有一种技术，可以只为<h1>指定第二个规则

不需要分解“h1, h2”规则，我们只需要只为h1再增加另一个规则，为它增加边框样式。

```
h1, h2 {
    font-family:    sans-serif;
    color:          gray;
}
```

第一个规则还是一样的。我们将使用一个合并规则指定<h1>和<h2>的font-family和color。

```
h1 {
    border-bottom: 1px solid black;
}
```

不过，现在要增加第二个规则，只为<h1>增加另一个属性：border-bottom属性。

```
p {
    color: maroon;
}
```

再试一试……

修改你的CSS，重新加载这个页面。你会看到，这个新规则在主标题的下面增加了一个黑色边框，这就为标题提供了一个漂亮的下划线，确实让它更为醒目了。

这里有黑色的下边框。

这里没有边框，这正是我们想要的。

there are no Dumb Questions

问：如果一个元素有多个规则，这是怎么处理的？

答：一个元素可以有多个规则，你可以根据需要来指定。每个规则会在现有的样式信息前面增加新的样式信息。一般来讲，要把元素的所有共同样式归组在一起，就像我们对<h1>和<h2>所做的处理，然后把一个元素特定的样式写在另一个规则中，如主标题的border-bottom样式。

问：这种方法有什么好处？单独地组织各个元素不是更好吗？这样你就能清楚地知道每个元素的样式是什么。

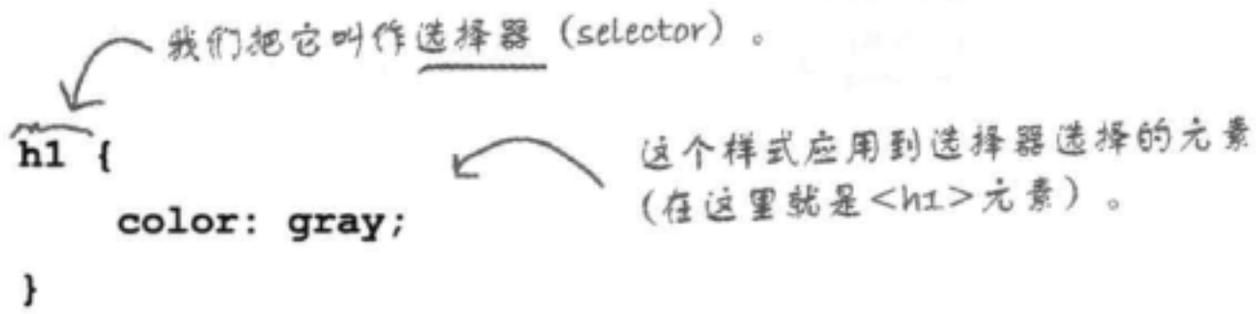
答：不见得。如果把共同的样式合并在一起，倘若它们有改变，你只需要在一个规则中修改。如果把它们分开，就必须修改多个规则，这很容易出错。

问：为什么要使用一个下边框为文本加下划线？文本不是有一个underline（下划线）样式吗？

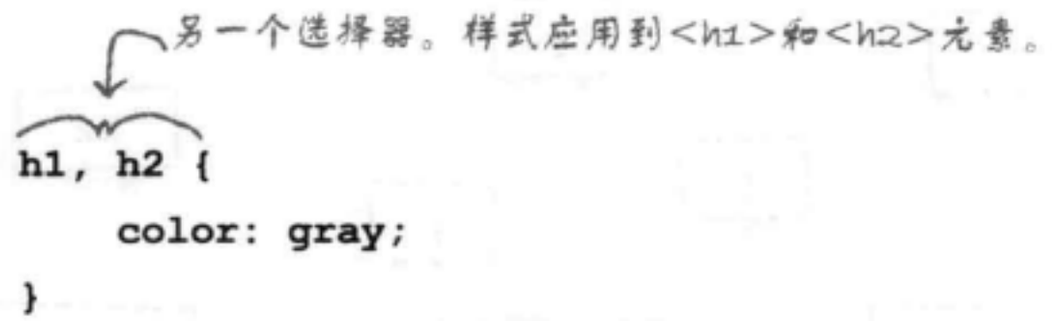
答：问得好。文本确实有一个underline（下划线）样式，我们当然也可以使用这个样式。不过，这两个样式在页面上的效果稍有不同。如果使用border-bottom，这条线会延伸到页面边缘。而使用underline时，下划线只出现在文本下面，不会继续延伸到页面边缘。设置文本下划线的属性名为text-decoration，值为“underline”时表示文本有下划线。你可以试一试，看看有什么不同。

那么，选择器到底如何工作？

你已经看到如何选择一个元素来增加样式，就像这样：

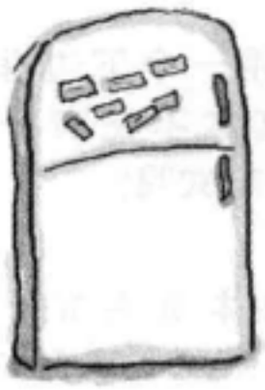


我们也见过选择多个元素的情况，如下所示：



你会看到，CSS允许你指定各种选择器，来确定将样式应用到哪些元素。了解如何使用这些选择器是掌握CSS的第一步，为此你需要知道要增加样式的HTML是如何组织的。毕竟，如果你对HTML没有一个清楚的认识，不知道其中有什么元素，以及它们之间有什么关系，又该如何选择元素来增加样式呢？

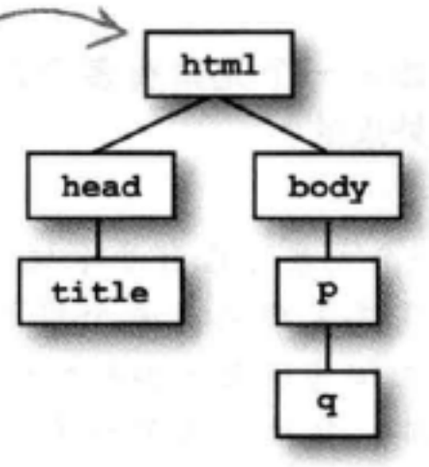
所以，先对休闲室HTML有一个明确的认识，然后再回来深入讨论选择器。

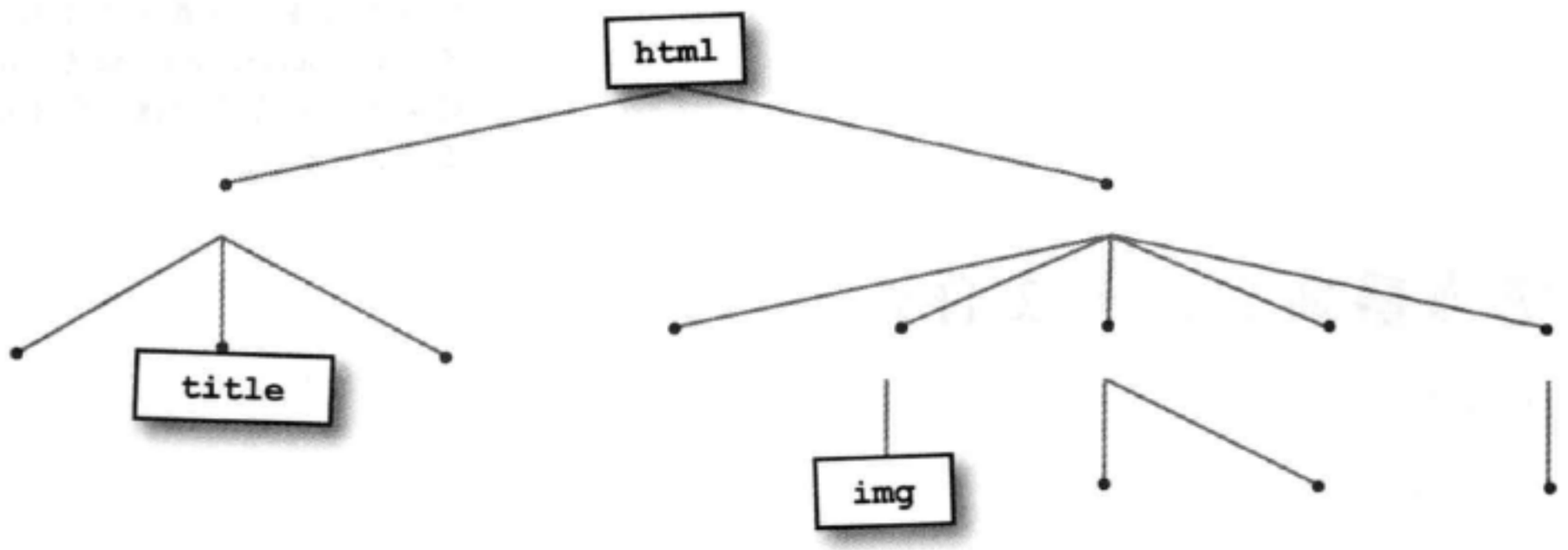


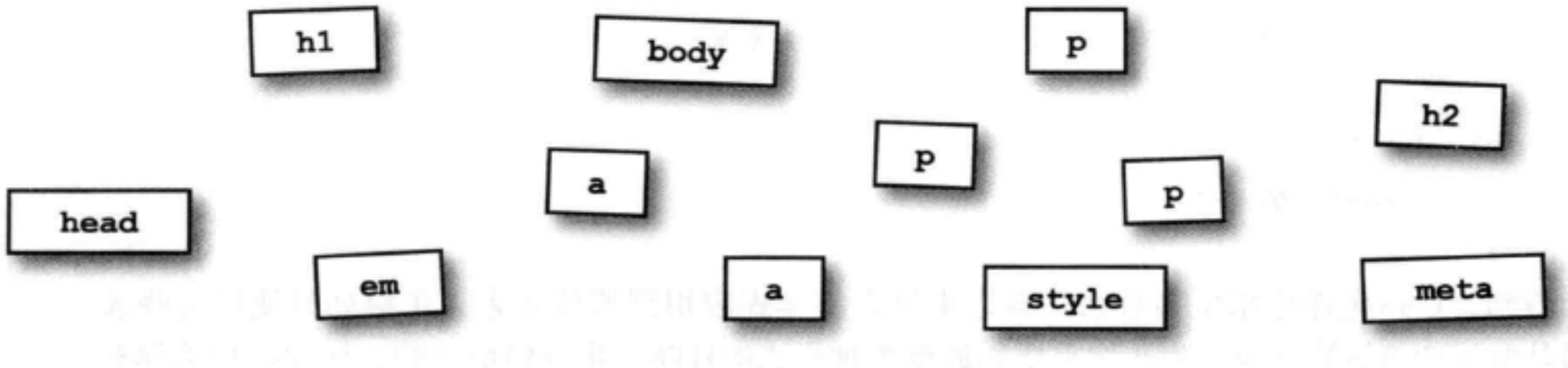
标记磁贴

还记得第3章中画的HTML元素结构图吗？现在再对休闲室主页面做同样的练习。你会在下面找到完成这个结构图需要的所有元素磁贴。使用右页中休闲室的HTML，完成下面的结构树。我们已经帮你放了几个磁贴。这一章最后会给出答案。

就像这样。







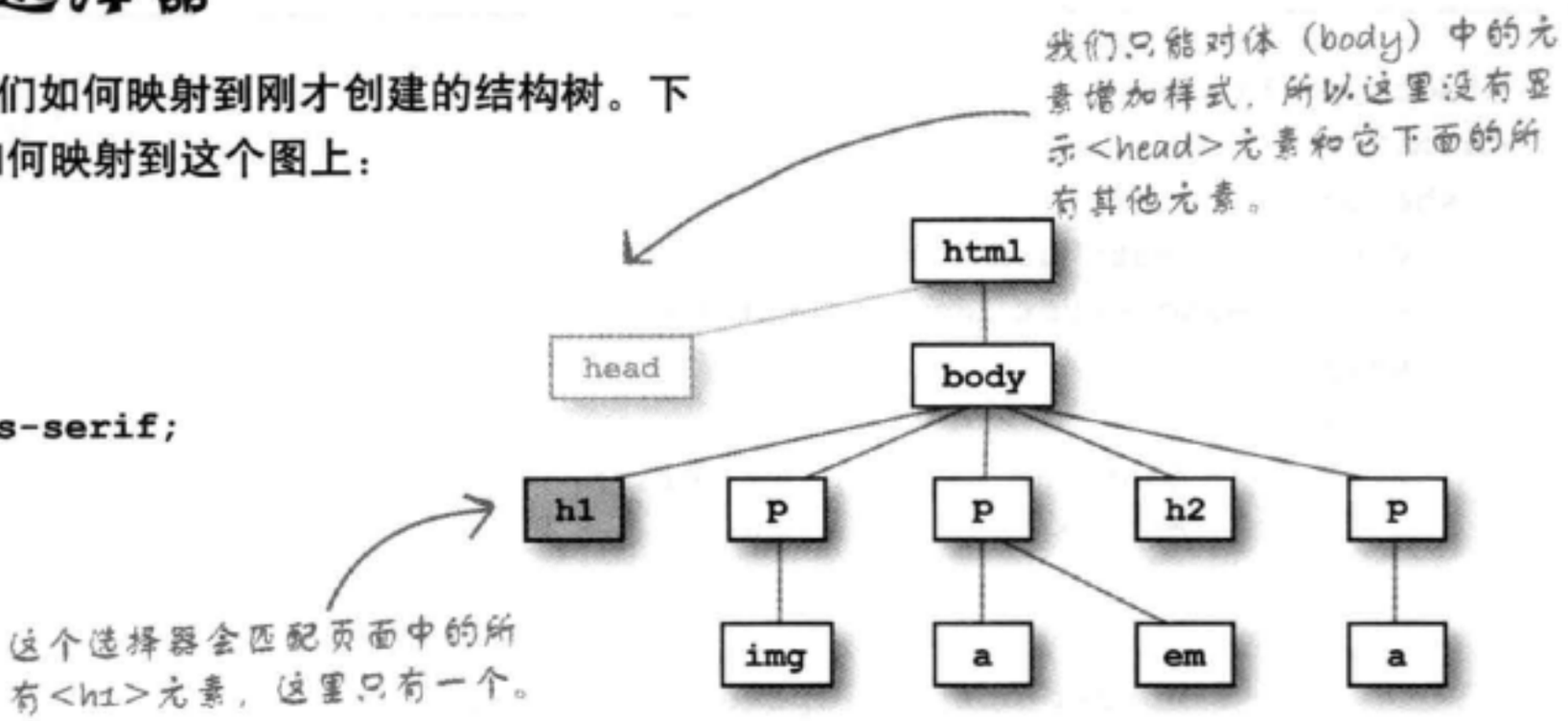
Head First休闲室的HTML。

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge</title>
    <style>
       h1, h2 {
             font-family:   sans-serif;
             color:         gray;
       }
       h1 {
             border-bottom: 1px solid black;
       }
       p {
             color: maroon;
       }
  </style>
</head>
<body>
  <h1>Welcome to the Head First Lounge</h1>
  <p>
     <img src="images/drinks.gif" alt="Drinks">
  </p>
  <p>
     Join us any evening for refreshing
     <a href="beverages/elixir.html">elixirs</a>,
     conversation and maybe a game or two
     of <em>Dance Dance Revolution</em>.
     Wireless access is always provided;
     BYOWS (Bring your own Web sever).
    </p>
    <h2>Directions</h2>
    <p>
       You'll find us right in the center of downtown
       Webville. If you need help finding us, check out our
       <a href="about/directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

通过图解来研究选择器

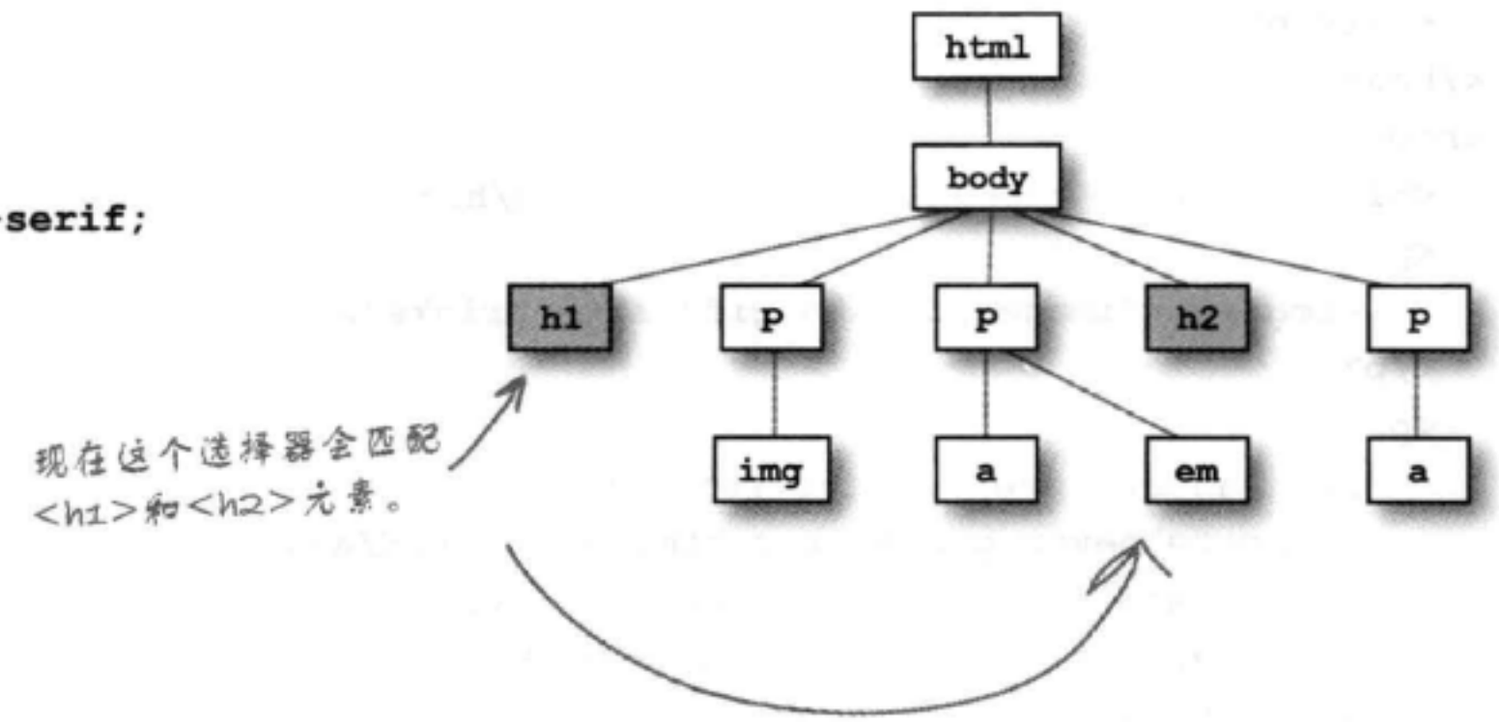
下面来看一些选择器，看看它们如何映射到刚才创建的结构树。下面显示了这个“h1”选择器如何映射到这个图上：

```
h1 {
    font-family: sans-serif;
}
```



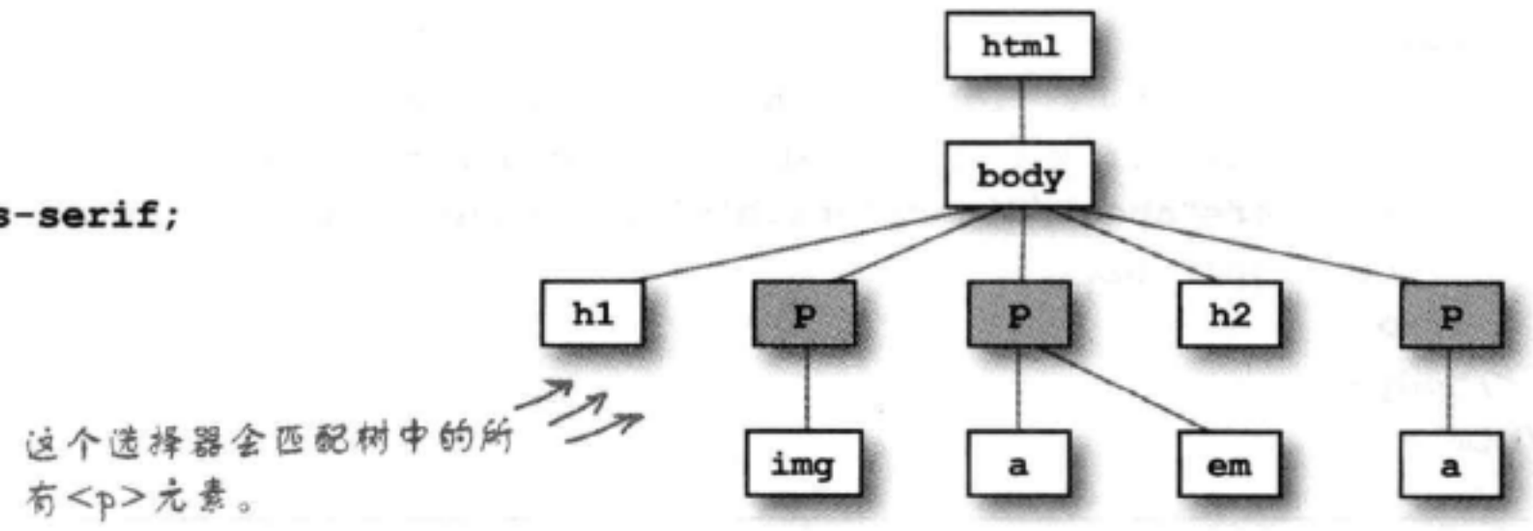
“h1, h2”选择器是这样的：

```
h1, h2 {
    font-family: sans-serif;
}
```



如果使用一个“p”选择器，就像这样：

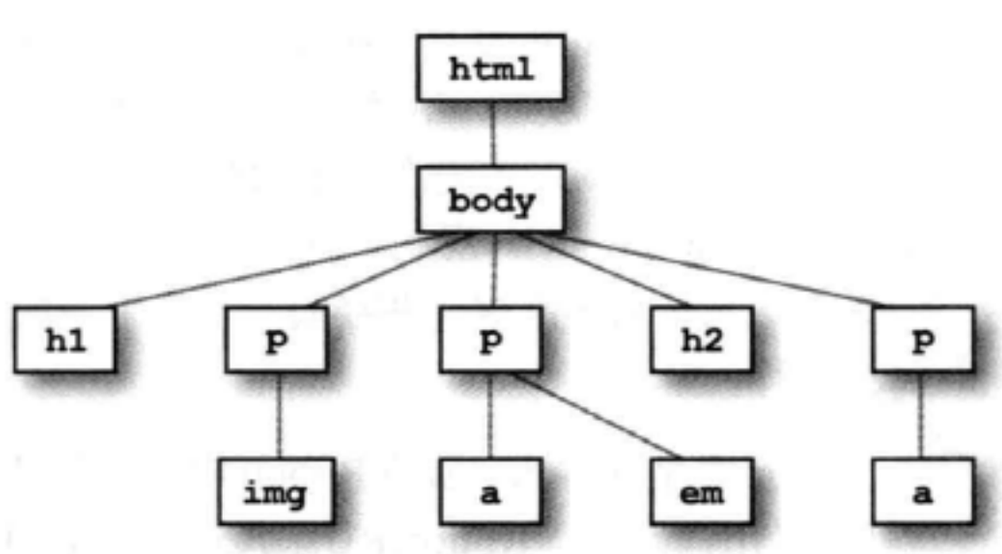
```
p {
    font-family: sans-serif;
}
```



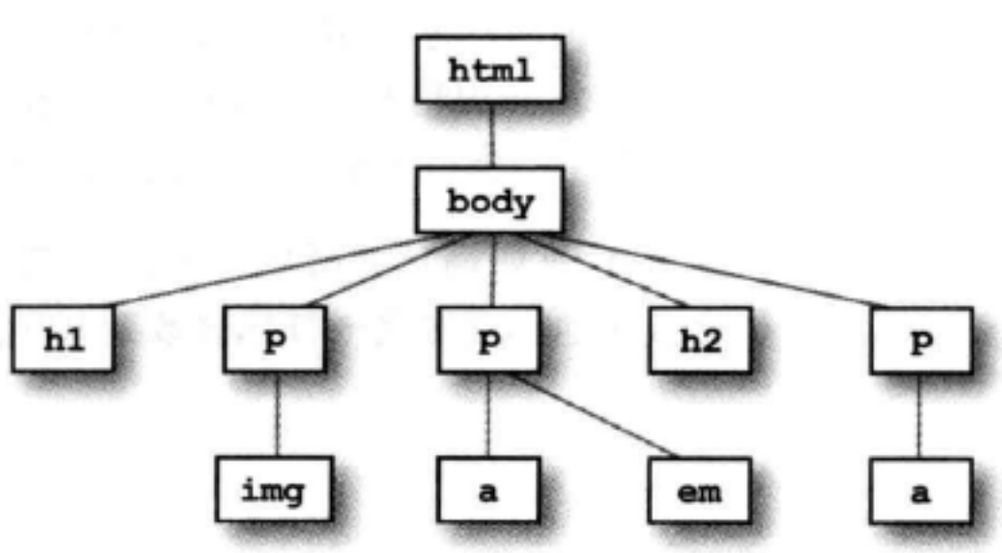
Sharpen your pencil

把这些选择器**选择**的元素涂上颜色：

```
p, h2 {
    font-family: sans-serif;
}
```




```
p, em {
    font-family: sans-serif;
}
```



蛮力与样式谜案

第4章谈到了RadWebDesign，他们在公司演示会上搞砸了，丢掉了RobotsRUs的生意。CorrectWebDesign接手负责整个RobotsRUs网站的建设，正在加班加点，要在这个月网站正式启动之前完善所有方面。不过，你还记得吧，RadWebDesign决定重整旗鼓，好好提高他们的HTML和CSS水平。他们决定自己修改RobotsRUs网站，这一次使用正确的HTML和样式，这只是为了掌握一些经验，希望下一次能接到其他咨询工作。

五分钟谜案

命运总是捉弄人，就在RobotsRUs大型网站正式启动的前一刻，又出问题了。RobotsRUs给CorrectWebDesign打来电话，通知了一个紧急消息，“我们要改变公司形象，现在网站上的所有颜色、背景和字体都要改”。而此时，网站包括将近100个页面，所以CorrectWebDesign做出回应，说他们需要几天的时间来修改网站。“我们没那么多时间，不可能再给你几天来修改！”CEO说。不得已，CEO决定再打电话给RadWebDesign寻求帮助，“虽然你们上个月的演示搞得一团糟糕，不过现在我们确实需要你们的帮助。你能帮CorrectWebDesign的人修改网站，让它有一个全新的面貌吗？”RadWebDesign回答说，他们还能做得更好，实际上，他们不到一小时就可以交付整个网站。

RadWebDesign是如何从耻辱中走出来，成为Web页面超级英雄的？是什么让他们能像飞一样迅速改变100个页面的外观？

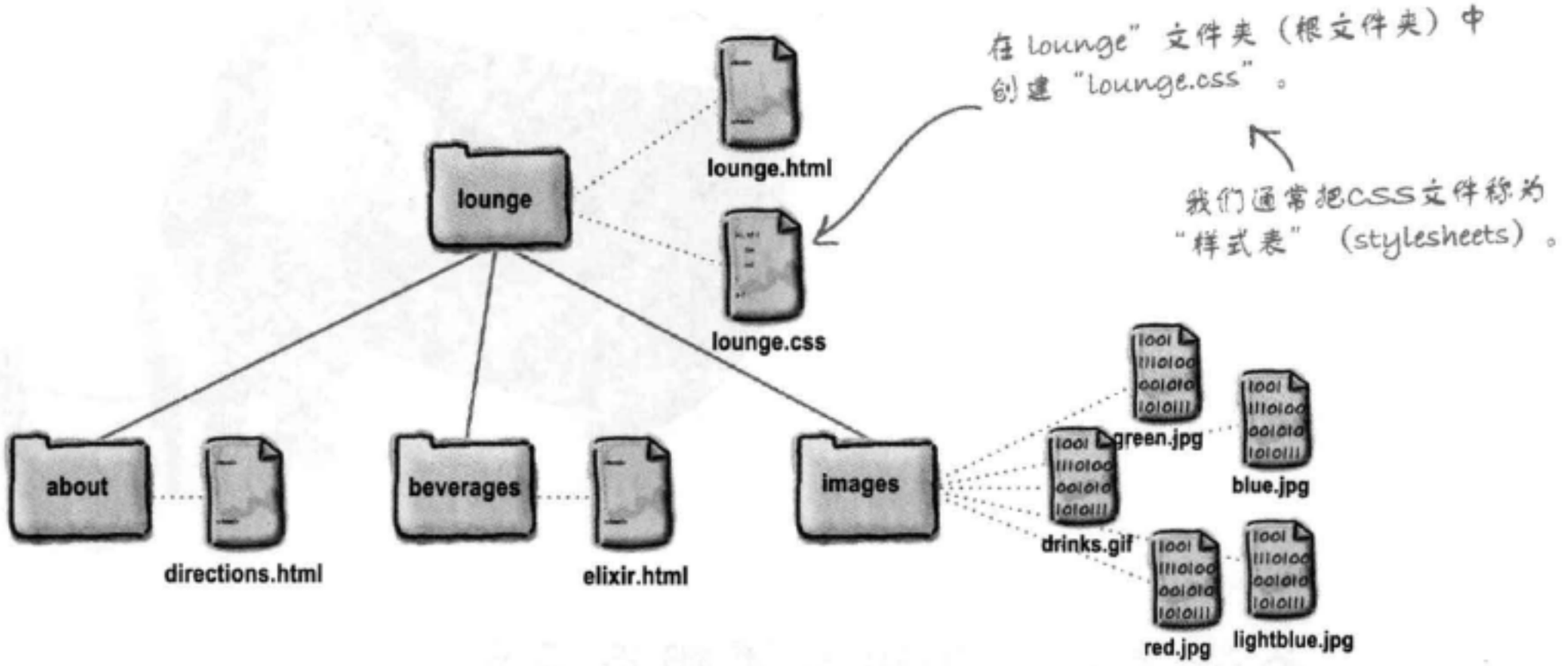
为清凉饮料和路线说明页面加入休闲室页面的样式

我们已经为“lounge.html”增加了所有样式，不过“elixir.html”和“directions.html”呢？它们要与主页外观一致。很容易……只需要把style元素和其中的所有规则复制到各个文件，是这样吗？别那么快下结论。如果这样做了，那么不论你什么时候需要改变网站的样式，都必须修改每一个文件，这可不是你想要的。不过，很幸运，有一种更好的办法。你可以这样做：

1. 取出“lounge.html”中的规则，把它们放在一个名为“lounge.css”的文件中。
2. 从“lounge.html”文件创建一个到这个文件的外部链接。
3. 在“elixir.html”和“directions.html”中创建同样的外部链接。
4. 对这3个文件好好做个测试。

创建“lounge.css”文件

你要创建一个名为“lounge.css”的文件，包含所有Head First休闲室页面的样式规则。为此，需要在你的文本编辑器中创建一个名为“lounge.css”的新的文本文件。



现在输入这些CSS规则，或者也可以从你的“lounge.html”文件复制粘贴这些CSS规则，把它们放在“lounge.css”文件中。完成后，从“lounge.html”文件中删除这些规则。

注意不要复制 <style>和</style>标记，因为“lounge.css”文件只包含CSS，不能包含HTML。

```
h1, h2 {
   font-family: sans-serif;
   color: gray;
}

h1 {
   border-bottom: 1px solid black;
}

p {
   color: maroon;
}
```

你的 “lounge.css” 文件会像这样。
要记住，没有<style>标记！

从“lounge.html”链接到外部样式表

现在我们需要告诉浏览器，它要利用外部样式表为这个页面增加样式。可以利用HTML <link>元素来做到。在你的HTML中使用<link>元素，如下所示：

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge</title>
    <link type="text/css" rel="stylesheet" href="lounge.css">
    <style>
    </style>

  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1>
    <p>
       <img src="drinks.gif" alt="Drinks">
    </p>
    .
    .
    .
    </p>
  </body>
</html>
```

这是链接到外部样式表的HTML。

不再需要<style>元素，将它删除。

其余的HTML仍然保持不变。

HTML放大镜

下面再仔细查看<link>元素，因为你之前还没有见过这个元素：

使用link元素“链入”外部信息。

这个信息的类型是“text/css”。换句话说，这是一个CSS样式表。在HTML5中，不再需要这个属性（这是可选的），不过你可能会在比较老的页面上看到它。

样式表放在这个href中（在这里，我们使用了一个相对链接，不过这也可以是一个完整的URL）。

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

rel属性指定了HTML文件与所链接的文件之间的关系。我们要链接到一个样式表，所以这里使用值“stylesheet”。

<link>是一个void元素。它没有结束标记。

从“elixir.html”和“directions.html”链接到外部样式表

现在你要从“elixir.html”和“directions.html”文件链接外部样式表，就像在“lounge.html”中一样。这里只需要记住一点，“elixir.html”在“beverages”文件夹中，“directions.html”在“about”文件夹中，所以它们都需要使用相对路径“../lounge.css”。

因此，现在要做的就是在这两个文件中增加以下<link>元素：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge Elixirs</title>
    <link type="text/css" rel="stylesheet" href="../lounge.css">
  </head>
  <body>
    .
    .
    .
  </body>
</html>
```

这是“elixir.html”。直接增加<link>行。

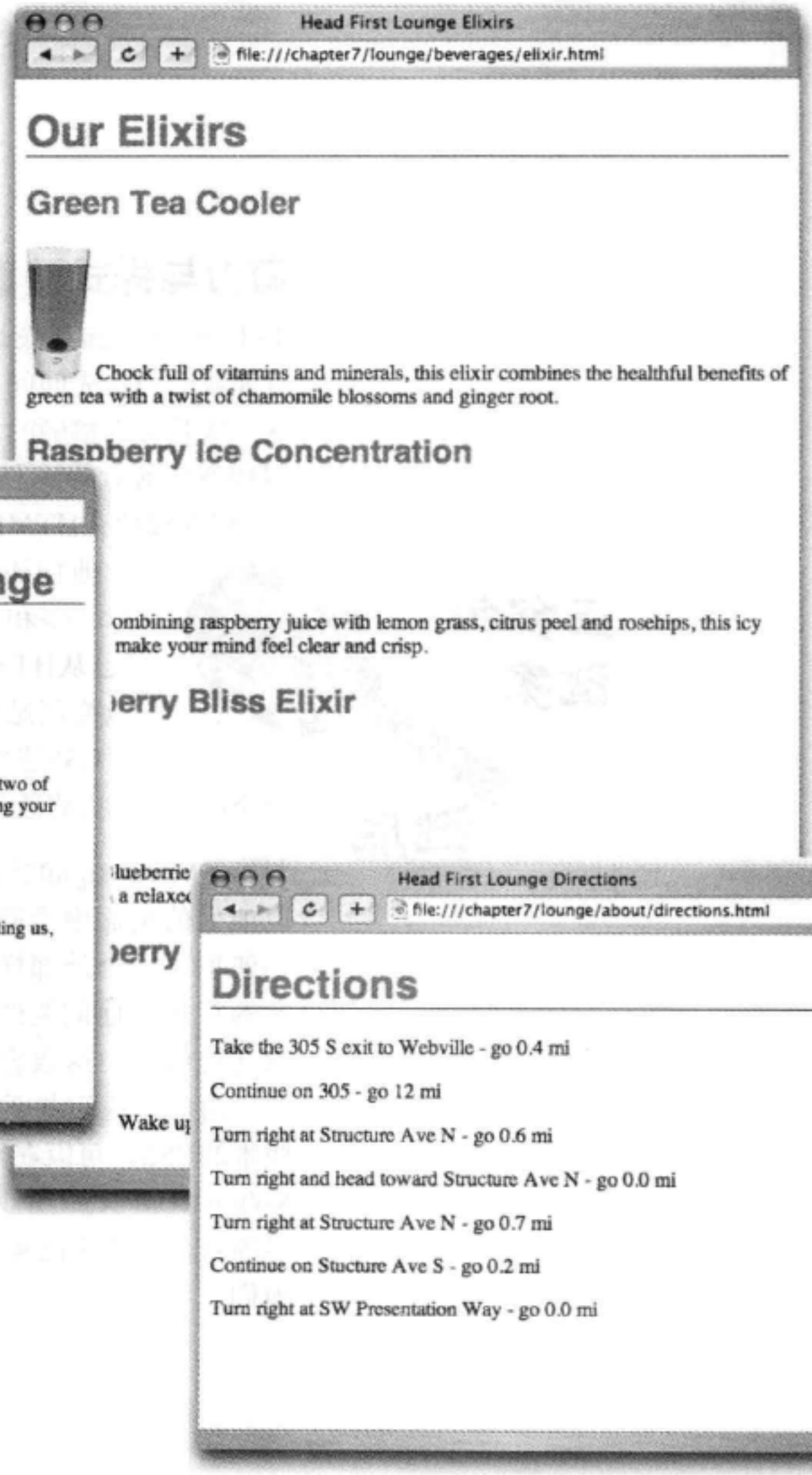
```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge Directions</title>
    <link type="text/css" rel="stylesheet" href="../lounge.css">
  </head>
  <body>
    .
    .
    .
  </body>
</html>
```

对“directions.html”做同样的处理。在这里增加<link>行。

测试整个休闲室……

保存这些文件，然后在浏览器中打开“lounge.html”。应该看不出样式有任何改变，不过现在样式来自一个外部文件。下面单击“elixirs”和“detailed directions”链接。

哇！只是在每个文件中修改了一行HTML，Elixirs和Directions页面就有了全新的样式！现在你看到CSS的巨大威力了吧。



蛮力与样式谜案

RadWebDesign到底是怎么变成web页面超级英雄的？或者可能我们应该先问问“从不犯错”的CorrectWebDesign公司这一次是怎么搞砸的。问题的根源在于CorrectWebDesign使用1998年前后的技术来创建RobotsRUs页面。他们把样式规则直接放在HTML中（每一次都复制粘贴），更糟糕的是，他们还使用了大量古老的HTML元素，比如<font>和 ，现在这些元素已经过时了（已从HTML删除）。所以，当接到电话要求他们改变网站外观时，这意味着要进入每一个Web页面，对CSS进行修改。不仅如此，这还意味着需要通查全部HTML来修改元素。

与RadWebDesign的做法比较一下：他们使用了HTML5，所以他们的页面中没有那些提供表现的古老HTML，另外他们使用了一个外部样式表。结果怎么样呢？要改变整个网站的样式，他们要做的只是进入这个外部样式表，对CSS做几处修改，这很容易，几分钟就能搞定，而不需要花费几天时间。他们甚至还有时间尝试多种设计，提供3个不同版本的CSS，可以在网站正式启动前预览它们的效果。这让RobotsRUs　CEO刮目相看，不仅向RadWebDesign承诺了更多项目，还答应把从生产线生产出来的第一个机器人赠送给他们。

Sharpen your pencil

现在你有了一个外部样式文件（或“样式表”），用它将所有段落字体改为“sans-serif”，与标题一致。要记住，改变字体样式的属性是“font-family”，对应sans-serif字体的值是“sans-serif”。答案见下一页。

标题使用sans-serif字体，这种字体没有衬线，看起来很清晰。

段落仍然使用默认的serif字体，这种字体有衬线，通常认为这种字体在计算机屏幕上阅读时会比较困难。

any

衬线（serif）。

Sharpen your pencil Solution

现在你有了一个外部样式文件（或“样式表”），用它将所有段落字体改为“sans-serif”，与标题一致。要记住，改变字体样式的属性是“font-family”，对应sans-serif字体的值是“sans-serif”。以下是我们的答案。

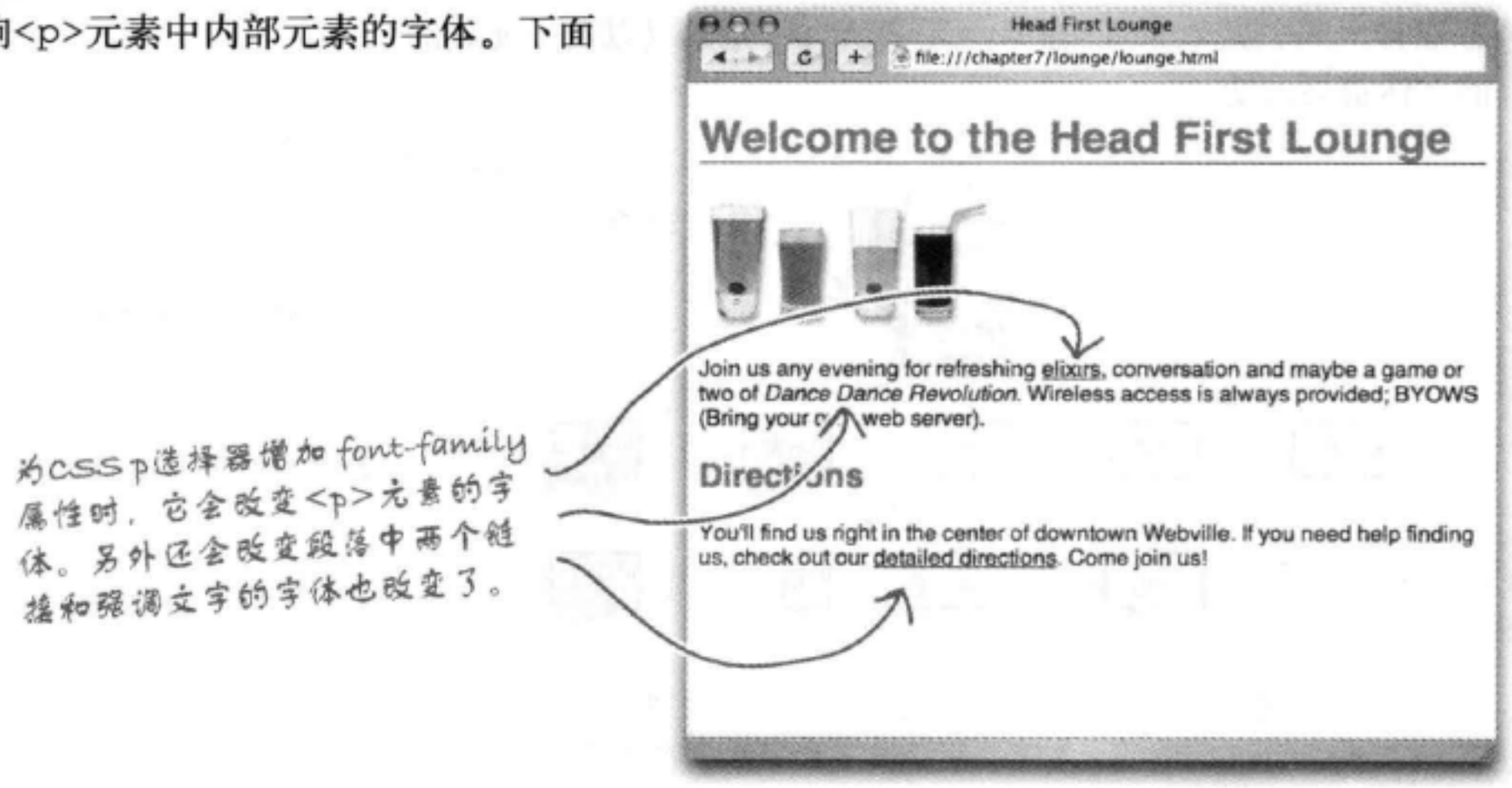
```
h1, h2 {
       font-family:  sans-serif;
       color:        gray;
}

h1 {
       border-bottom: 1px solid black;
}

p {
       font-family:  sans-serif;
       color:        maroon;
}
```

在“lounge.css”文件中为段落规则增加一个font-family属性。

我想知道这是否确实是最佳方案。为什么我们要为每一个元素指定font-family？如果有人在页面中增加了一个<blockquote>，是不是还得为它增加一个规则？不能告诉整个页面都采用sans-serif字体吗？

来谈谈继承……

注意到了吗？为p选择器增加font-family属性时，这也会影响<p>元素中内部元素的字体。下面来仔细看一下：



<p>元素中的元素从<p>继承了font-family样式

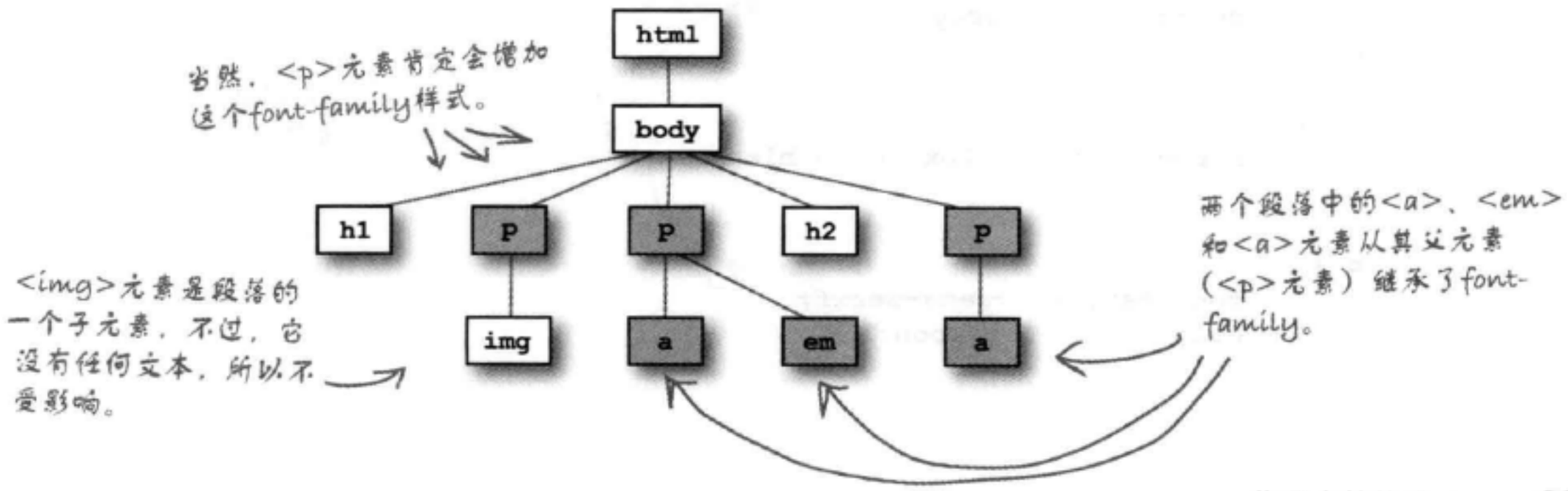
就像你可能会从你的父母那里继承蓝眼睛和金黄色的头发，同样的，元素也可以从它们的父元素继承样式。在这里，<a>和元素就从<p>元素继承了font-family样式，<p>元素就是它们的父元素。改变段落样式也会改变段落中元素的样式，这是有道理的，不是吗？毕竟，如果不是这样，你就只能事必躬亲，为整个网站中每一个段落中的每一个内联元素增加CSS规则……这太可怕了。

不是所有样式都能继承。只有一部分能继承，如 font-family。

更不用说还容易出错，很麻烦，另外还浪费时间。

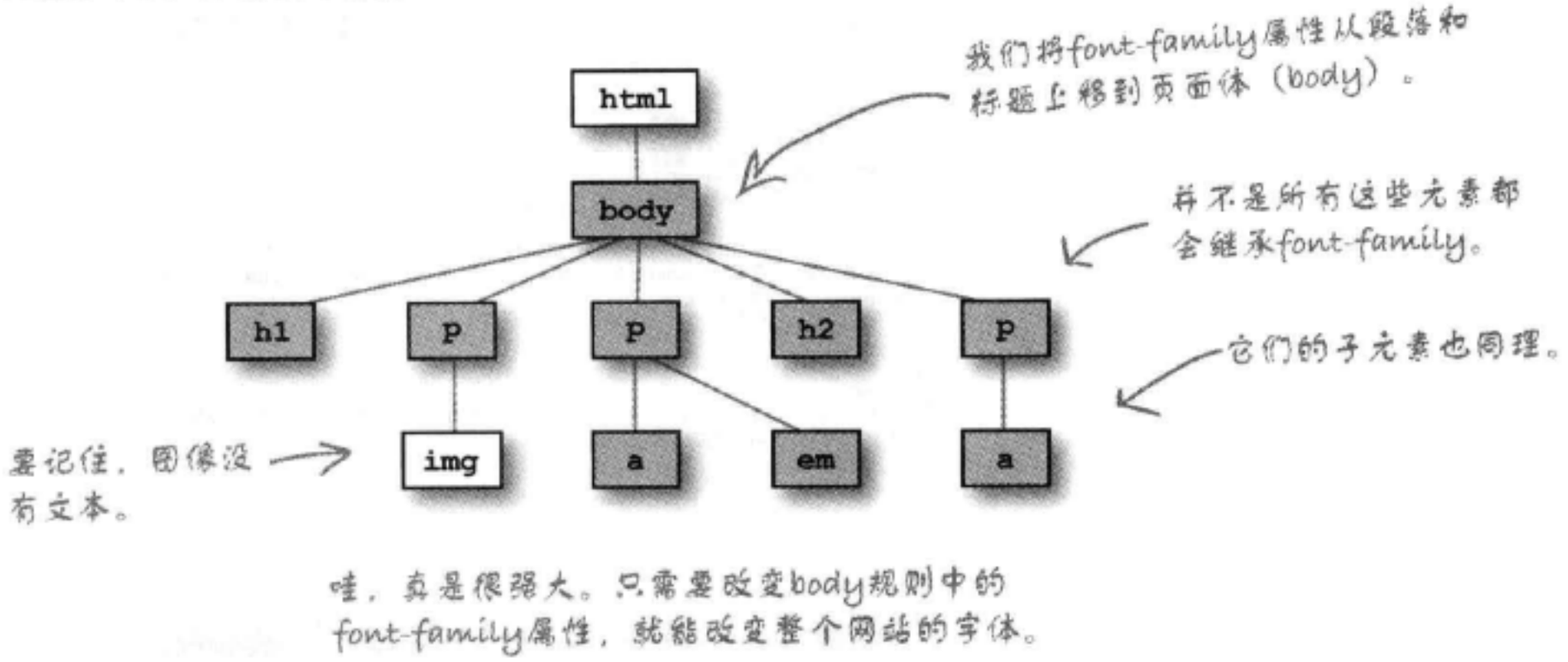
下面给出HTML树，看看继承到底是怎样工作的：

如果设置所有<p>元素的font-family，就会影响到下面阴影显示的所有元素。



如果在家族树中上移字体会有怎样的结果？

如果大多数元素都继承这个font-family属性，把它上移到<body>元素怎么样？这样一来，其效果是<body>元素所有子元素（以及子元素的子元素）的字体都会改变。



还等什么呢……试一试吧

打开你的“lounge.css”文件，增加一个新规则，选择<body>元素。然后从标题和段落规则删除font-family属性，因为现在不再需要它们了。

```
body {
      font-family:  sans-serif;
}

h1, h2 {
      font-family:  sans-serif;
      color:        gray;
}

h1 {
      border-bottom: 1px solid black;
}

p {
      font-family:  sans-serif;
      color:        maroon;
}
```

你要这么做：

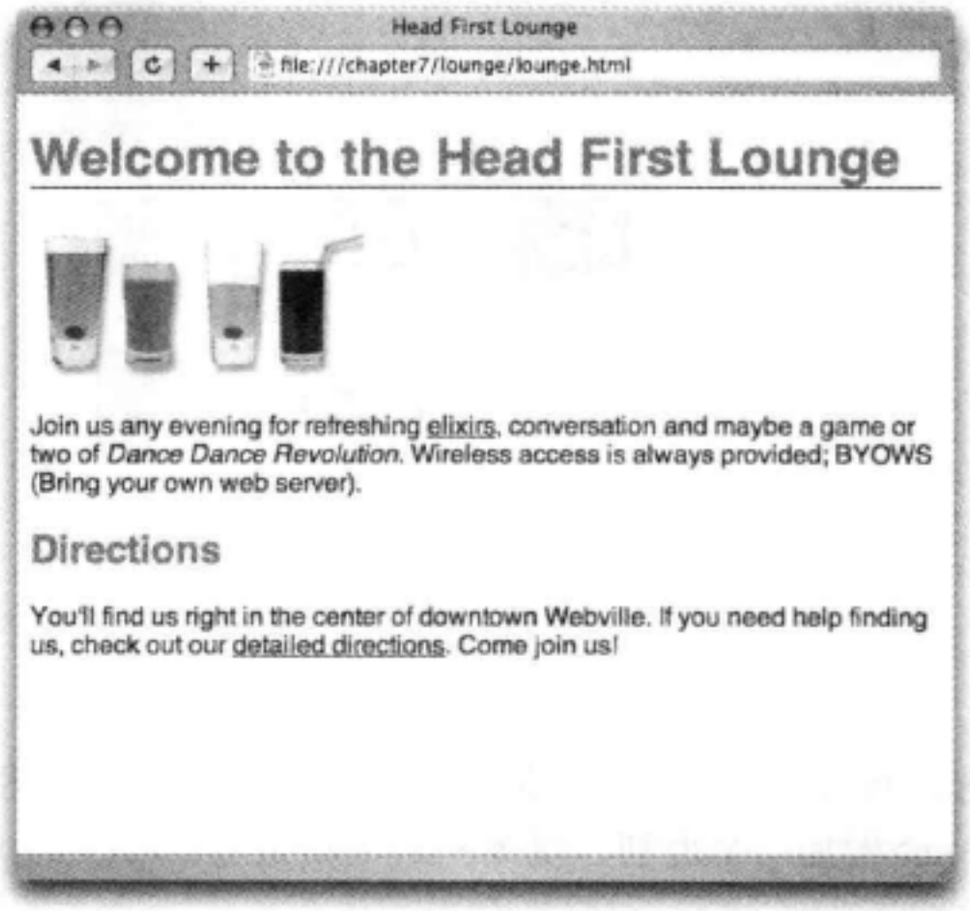
首先，增加一个新规则，选择<body>元素。然后增加font-family属性，值为sans-serif。

接下来，从“h1, h2”规则以及p规则删除font-family属性。

测试你的新CSS

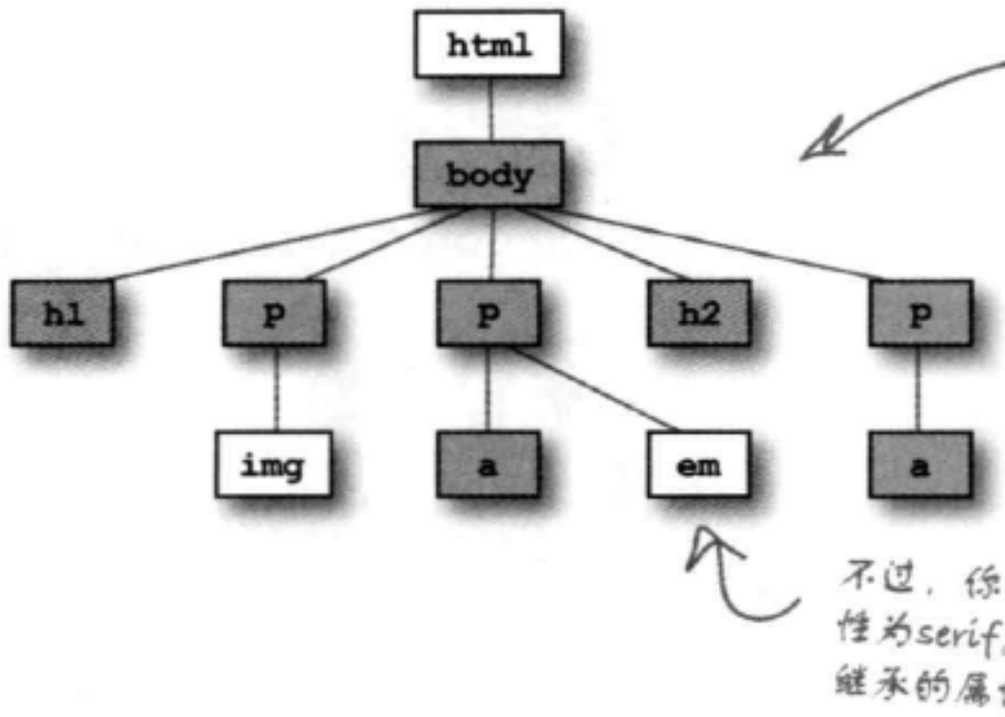
与以往一样，在“lounge.css”样式表中完成这些修改，保存，再重新加载“lounge.html”页面。你不会看到任何改变，因为样式还是一样的，只不过来自一个不同的规则。但是你会感觉你的CSS比以前好，因为现在如果向页面增加新元素，它们会自动继承sans-serif字体。

太惊奇了。这看上去没有任何区别，不过，这正是我们期望的，不是吗？你所做的只是把sans-serif字体上移到body规则中，让所有其他元素继承这个样式。



覆盖继承

通过将font-family属性上移到body规则中，这就为整个页面设置了这个字体样式。不过，如果你不希望每个元素都使用这个sans-serif字体，该怎么办呢？例如，你可能希望元素使用serif字体。



font-family属性在body规则中设置，所以体（body元素）中的所有元素都会从<body>继承这个font-family属性（sans-serif）。

不过，你希望元素的font-family属性为serif。这就需要用一个CSS规则覆盖继承的属性。

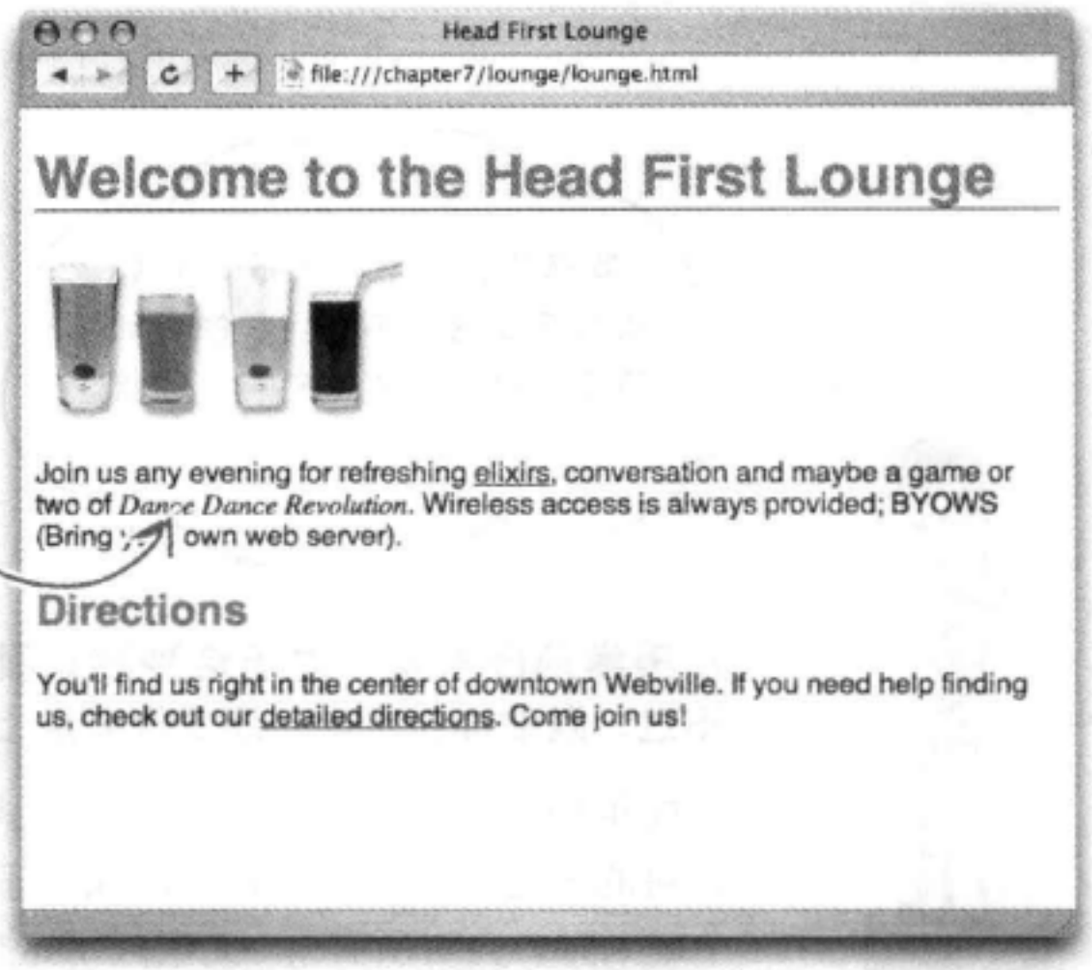
嗯，可以只为提供一个特定的规则来覆盖继承。下面为增加一个规则，覆盖body中指定的font-family：

```
body {
      font-family:  sans-serif;
}

h1, h2 {
      color:        gray;
}

h1 {
      border-bottom: 1px solid black;
}

p {
      color:        maroon;
}

em {
      font-family: serif;
}
```

要覆盖从body继承的font-family属性，可以增加一个新规则选择em，将font-family属性值设置为serif。

试一试

在CSS中为元素增加一个规则，font-family属性值设置为serif，然后重新加载你的“lounge.html”页面：



注意这里的“Dance Dance Revolution”文本，这是元素中的文本，现在它使用一种serif字体。

一般来讲，像这样在段落中间改变字体并不是一个好主意，所以完成测试之后再把CSS改回原样（去掉em规则）。

there are no Dumb Questions

问：我覆盖继承的值时，浏览器怎么知道对应用哪个规则？

答：对于CSS，总会使用最特定的那个规则。所以，如果对<body>有一个规则，对元素有一个更特定的规则，它就会使用这个更特定的规则。后面还会讨论如何知道哪个规则最特定。

问：我怎么知道哪些CSS属性能继承，哪些不能继承？

答：在这方面，如果有一本好的参考书会很方便，比如O’Reilly出版的《CSS Pocket Reference》。一般来讲，如果样式会影响你的文本外观，所有这些样式都能继承，如字体颜色（color属性）、刚才看到的font-family（字体系列），以及所有与字体相关的属性，如font-size（字体大小）、font-weight（字体粗细）和font-style（是否斜体）。其他属性不能继承，如边框，这是有道理的，不是吗？如果<body>元素有一个边框，这并不表示你希望体中的所有元素都有边框。很多情况下，可以根据你的常识来确定（或者可以试试看），随着你对各种属性以及它们的作用越来越熟悉，你就能很好地掌握哪些属性能继承而哪些不能。

问：如果我不希望继承，继承得到的属性是不是都能被覆盖？

答：是的，总能使用一个更特定的选择器覆盖从父选择器继承的属性。

问：这个内容有些复杂了。有没有办法增加注释，来提醒自己这些规则的作用？

答：有的。要在CSS中写注释，只需要把注释包围在/*和*/之间。例如：

```
/* 这个规则选择所有段落，指定它们的颜色为蓝色 */
```

注意，注释可以跨多行。还可以用注释包围CSS，浏览器会将其忽略，如下所示：

```
/* 这个规则没有任何效果，因为它在一个注释里

p { color: blue; }  */
```

一定要正确地结束注释。否则，CSS不会起作用！

从美学角度来讲，对于这种建议我们并不太赞同，不过，既然你是顾客，还是顾客至上。

你能单独地为这些段落分别设置样式，使文本的颜色与饮料的颜色一致吗？现在的问题在于，用p选择器使用一个规则时，会对所有<p>元素应用这个样式。所以，如何单独地选择这些段落呢？

这里要引入类（class）概念了。结合HTML和CSS，我们可以定义一类元素，并对属于该类的所有元素应用样式。那么，类到底是什么？可以把它想成是一个俱乐部，比如有人创办了一个“绿茶”俱乐部，要加入这个俱乐部，你就必须认可这个俱乐部的所有权利和责任，如遵循他们的样式标准。不管怎样，我们来创建这个类，你会看到它是如何工作的。

创建一个类有两步：首先，为HTML中的元素增加一个class属性，这样就会把这个元素增加到这个类中；其次，在CSS中选择这个类。下面一步一步来完成……

把元素增加到greentea类

打开“elixir.html”文件，找到“Green Tea Cooler”段落。我们希望把这个文本改为绿色。你要做的就是把<p>元素增加到一个名为greentea的类中。可以这样做：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge Elixirs</title>
    <link type="text/css" rel="stylesheet" href="../lounge.css">
  </head>
  <body>
    <h1>Our Elixirs</h1>
    <h2>Green Tea Cooler</h2>
    <p class="greentea">
          <img src="../images/green.jpg" alt="Green Tea">
          Chock full of vitamins and minerals, this elixir
          combines the healthful benefits of green tea with
          a twist of chamomile blossoms and ginger root.
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p>
          <img src="../images/lightblue.jpg" alt="Raspberry Ice">
          Combining raspberry juice with lemon grass,
          citrus peel and rosehips, this icy drink
          will make your mind feel clear and crisp.
    </p>
    <h2>Blueberry Bliss Elixir</h2>
    <p>
          <img src="../images/blue.jpg" alt="Blueberry Bliss">
          Blueberries and cherry essence mixed into a base
          of elderflower herb tea will put you in a relaxed
          state of bliss in no time.
    </p>
    <h2>Cranberry Antioxidant Blast</h2>
    <p>
          <img src="../images/red.jpg" alt="Cranberry Blast">
          Wake up to the flavors of cranberry and hibiscus
          in this vitamin C rich elixir.
    </p>
  </body>
</html>
```

要将一个元素加入一个类，只需要增加属性“class”，并提供类名，如“greentea”。

既然关于绿茶的段落属于greentea类，下面只需要提供一些规则，为这个类的元素指定样式。

创建一个类选择器

要在CSS中创建一个类，并选择这个类中的一个元素，可以编写一个类选择器，如下：

现在你可以选择属于某个类的<p>元素，为它们指定样式。所要做的就是为希望为绿色的<p>元素增加class属性，这样一来，就会应用这个规则。可以来试试：打开你的“lounge.css”文件，增加这个p.greentea类选择器。

```
body {
     font-family: sans-serif;
}

h1, h2 {
     color: gray;
}

h1 {
     border-bottom: 1px solid black;
}

p {
     color: maroon;
}

p.greentea {
     color: green;
}
```

试一试greentea

保存文件，然后重新加载页面，来试一试你的新类。

新的greentea类应用到这个段落。现在字体是绿色，与Green Tea Cooler饮料的颜色一致。也许这种样式设计不算太糟糕。

Sharpen your pencil

该轮到你了：为“elixir.html”中适当的段落增加两个类“raspberry”和“blueberry”，然后编写样式将这些文本分别设置为蓝色和紫色。raspberry的属性值是“blue”，blueberry的属性值是“purple”。把这些样式放在CSS文件的最后，即greentea规则的下面：先是raspberry规则，然后是blueberry规则。

我们知道，你可能在想，raspberry怎么是蓝色呢？嗯，如果Raspberry Kool-Aid是蓝色，对我们来说是完全可以的。说真的，如果把一大堆蓝莓混在一起，它们更应算是紫色而不是蓝色。就按我们说的去做吧。

更深入地研究类……

你已经写了一个规则，使用greentea类将这个类中的所有段落颜色改为“green”：

```
p.greentea {
        color: green;
}
```

不过，如果想对所有<blockquote>做同样的处理呢？可以这样做：

```
blockquote.greentea, p.greentea {
        color: green;
}
```

只需要再增加另一个选择器，来处理greentea类中的<blockquote>。现在这个规则会应用到greentea类中的<p>和<blockquote>元素。

在HTML中要这样写：

```
<blockquote class="greentea">
```

不用，有一种更好的办法。如果希望greentea类中的所有元素都有同一种样式，可以这样写规则：

```
.greentea {
        color: green;
}
```

如果省略所有元素名，只有一个点，后面是类名，那么这个规则会应用到这个类的所有成员。

太酷了！没错，这样是可以的。不过还有一个问题……你说过在一个类中就像在一个俱乐部里。嗯，我可以加入多个俱乐部。那么，一个元素能不能加入多个类呢？

可以，元素可以加入多个类。

一个元素要加入多个类，这很容易办到。假设你希望指定一个<p>元素属于greentea、raspberry和blueberry类。可以在开始标记中这样写：

```
<p class="greentea raspberry blueberry">
```

把各个类名放在class属性中，各个类名之间用一个空格分隔。类名的顺序并不重要。

这么说来，举个例子，我可以把一个<h1>放在“products”类中，这个类定义了字体大小和字体粗细，另外还可以把这个<h1>放在一个“specials”类中，如果某个饮料降价促销，就可以将它的颜色改为红色，是这样吗？

完全正确。如果你希望一个元素拥有不同类中定义的不同样式，就要使用多个类。对于你说的这个例子，与products关联的所有<h1>元素都有某种样式，不过并不是所有商品都同时降价促销。可以在一个单独的类中指定“specials”颜色，只把与促销商品关联的那些元素放在“specials”类中，为它们增加你希望的红色。

现在你可能想知道，如果一个元素属于多个类，所有这些类都定义了相同的属性，会有什么结果，比如上面的<p>元素。你怎么知道要应用哪个样式？你知道这些类都分别有一个color属性定义。所以，段落会是绿色、蓝色，还是紫色？

等我们对CSS有更多了解之后再来讨论这个问题，不过下一页你会看到一个简明指南，可以帮助你继续学习下面的内容。

关于应用样式的世界上最简短最快捷的指南

元素、文档树、样式规则，还有类……实在让人摸不着头脑。如何整理所有这些知识，从而能知道哪些样式要应用到哪些元素？之前我们说过，要全面地回答这个问题，你必须对CSS有更多了解，这些会在后面几章学习。不过在此之前，下面先简单了解一些应用样式的常识性规则。

首先，有没有某个选择器选择你的元素？

假设你想知道一个元素的font-family属性值。首先要检查：CSS文件中有没有一个选择器选择你的元素？如果有，而且它有一个font-family属性和值，那么这就是这个元素的font-family属性值。

继承情况呢？

如果没有与元素匹配的选择器，就要依赖于继承。所以，查看元素的父元素，以及父元素的父元素，依此类推，直到找到所定义的属性。如果找到了，这就是你要的值。

都没有？那就使用默认值

如果你的元素没有从它的任何祖先继承到这个值，就要使用浏览器定义的默认值。实际情况比我们描述的要复杂一些，不过这本书后面还会详细介绍。

如果多个选择器选择一个元素呢？

哈，之前就是这种情况，段落属于所有3个类：

```
<p class="greentea raspberry blueberry">
```

这里有多个选择器与这个元素匹配，它们都同样定义了color属性。这就是我们所说的冲突。哪个规则会胜出呢？嗯，如果一个规则比其他规则更特定，它就会胜出。不过，“更特定”是什么意思？后面有一章还会来讨论这个内容，告诉你如何确定一个选择器的特定程度，不过现在先来看一些规则，使你对规则的特定程度有所认识：

```
p { color: black;}

.greentea { color: green; }

p.greentea { color: green; }

p.raspberry { color: blue; }

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

这个规则会选择所有原来的段落元素。

这个规则选择greentea类的所有成员。这个规则更特定一些。

这个规则只选择greentea类中的段落，所以这比前一个规则更特定。

这些规则也只是选择一个特定类中的段落。所以它们与p.greentea规则特定程度相同。

如果仍然没有一个明确的赢家呢？

所以，如果一个元素只属于greentea类，会有一个明显的赢家：p.greentea选择器最为特定，所以文本将设置为绿色。不过，如果一个元素属于所有这3个类：greentea、raspberry和blueberry。这样一来，p.greentea、p.raspberry和p.blueberry都会选择这个元素，而且它们的特定程度还相同。现在你该怎么办？你会选择CSS文件中最后列出的那个规则。如果由于两个选择器有相同的特定性而无法解决冲突，就要利用样式表文件中规则的顺序来解决问题。也就是说，你要使用CSS文件中最后列出的规则（最靠后）。在这个例子中会使用p.blueberry规则。

在你的“elixir.html”文件中，把greentea段落改为包含所有类，就像这样：

```
<p class="greentea raspberry blueberry">
```

保存并重新加载，现在Green Tea Cooler段落是什么颜色？ _____

接下来，调整HTML中类的顺序：

```
<p class="raspberry blueberry greentea">
```

保存并重新加载，现在Green Tea Cooler段落是什么颜色？ _____

接下来，打开你的CSS文件，把p.greentea规则移到文件最下面。

保存并重新加载，现在Green Tea Cooler段落是什么颜色？ _____

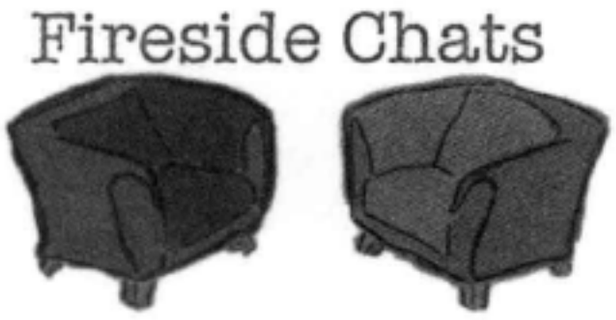
最后，把p.raspberry规则移到最下面。

保存并重新加载，现在Green Tea Cooler段落是什么颜色？ _____

完成之后，把greentea元素重写为原来的样子：

```
<p class="greentea">
```

保存并重新加载，现在Green Tea Cooler段落是什么颜色？ _____



今晚话题：CSS & HTML语言比较

CSS:

你看到了吗？我就像个魔术师！我从你的<style>元素中破茧而出，进入我自己的文件。你在第1章里居然还说我永远也无法逃脱！

必须把我链接进来？拜托，你要知道，没有我的样式，你的页面实在太难看了。

如果你专心地学习这一章，应该已经看到了，在我擅长的领域，我绝对能力非凡。

嗯，现在好多了。我也很高兴你能转变观念。

HTML:

别高兴得太早，还是需要我把你链接进来，你才能起作用。

再说一遍……我和我的所有元素只是要保证内容有结构，你谈论的却是头发亮不亮，指甲是什么颜色之类的问题。

好了，好了，这一点我承认，使用CSS确实能让我的工作容易些。那些过气的老样式元素确实让我很头疼。我很高兴可以为我的元素加样式，而不用在HTML中插入一大堆东西，当然有时候可能要有一个class属性。

不过，我还是忘不了你是怎么笑话我的语法的……<记得吗>?

CSS:

你得承认，HTML有点笨拙，不过从你作为一个20世纪九十年代早期的技术以来，你一直都是这样的。

你开玩笑吧？我表达能力非常强。我可以选择我想要的元素，然后准确地描述希望对它们如何加样式。你就等着看我实现的那些酷炫样式吧。

没错，等着瞧吧。我可以用各种各样有趣的方式对字体和文本设置样式。我还可以控制页面上每个元素如何管理它周围的空间。

哈哈，你以为我摆脱不了你的控制，只能在你的<style>标记中，是吗？你会看到，如果我愿意，完全可以让你的元素像狗一样坐下，大叫，甚至打滚儿。

HTML:

我倒不这么认为，我把这称为“经得住时间的考验”，而且你觉得CSS很高雅吗？我的意思是说，你只不过就是一堆规则。这也算一种语言吗？

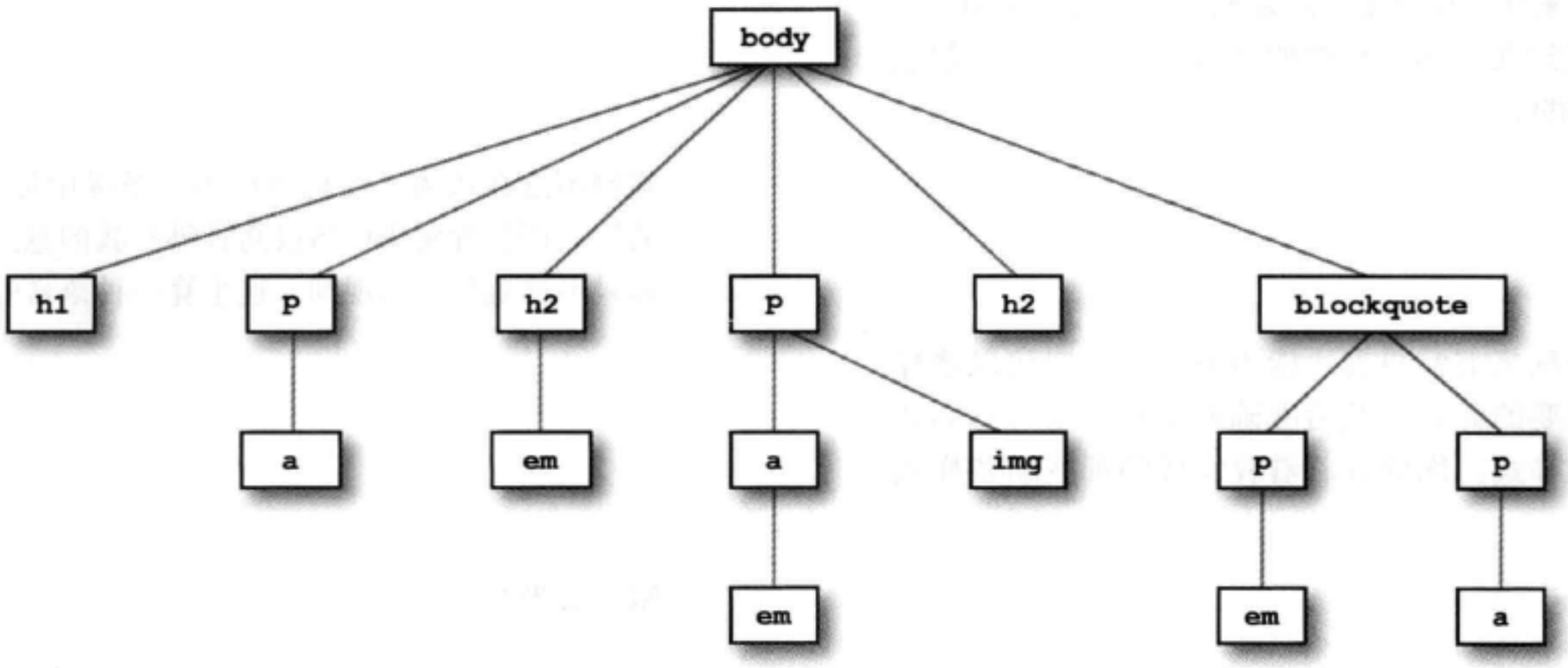
哦，是吗？

嗯……听起来你好像有点越俎代庖了。我可不喜欢你这样。毕竟，我的元素希望把握自己的人生。

嘿，停！保安……保安！

谁会继承？

嘿，<body>元素已经动身去浏览器了。不过他留下了很多子子孙孙，还让它们继承了颜色“green”。下面给出了他的家族树。把继承<body>元素green颜色的子孙元素标出来。别忘了先看下面的CSS。




```
body {
        color: green;
}

p {
        color: black;
}
```

这里是CSS。用它来确定上面的哪些元素会中彩，能得到继承的绿色（color属性）。

扮演浏览器

下面给出了CSS文件“style.css”，其中有一些错误。你的任务是扮演浏览器，找出所有错误。完成这个练习之后，对照本章最后的答案，看看有没有找出所有的错误。

如果你的CSS里有错误，通常这个错误以下的所有其他规则都会被忽略。所以通过这个练习，要养成检查错误的习惯。

文件“style.css”。

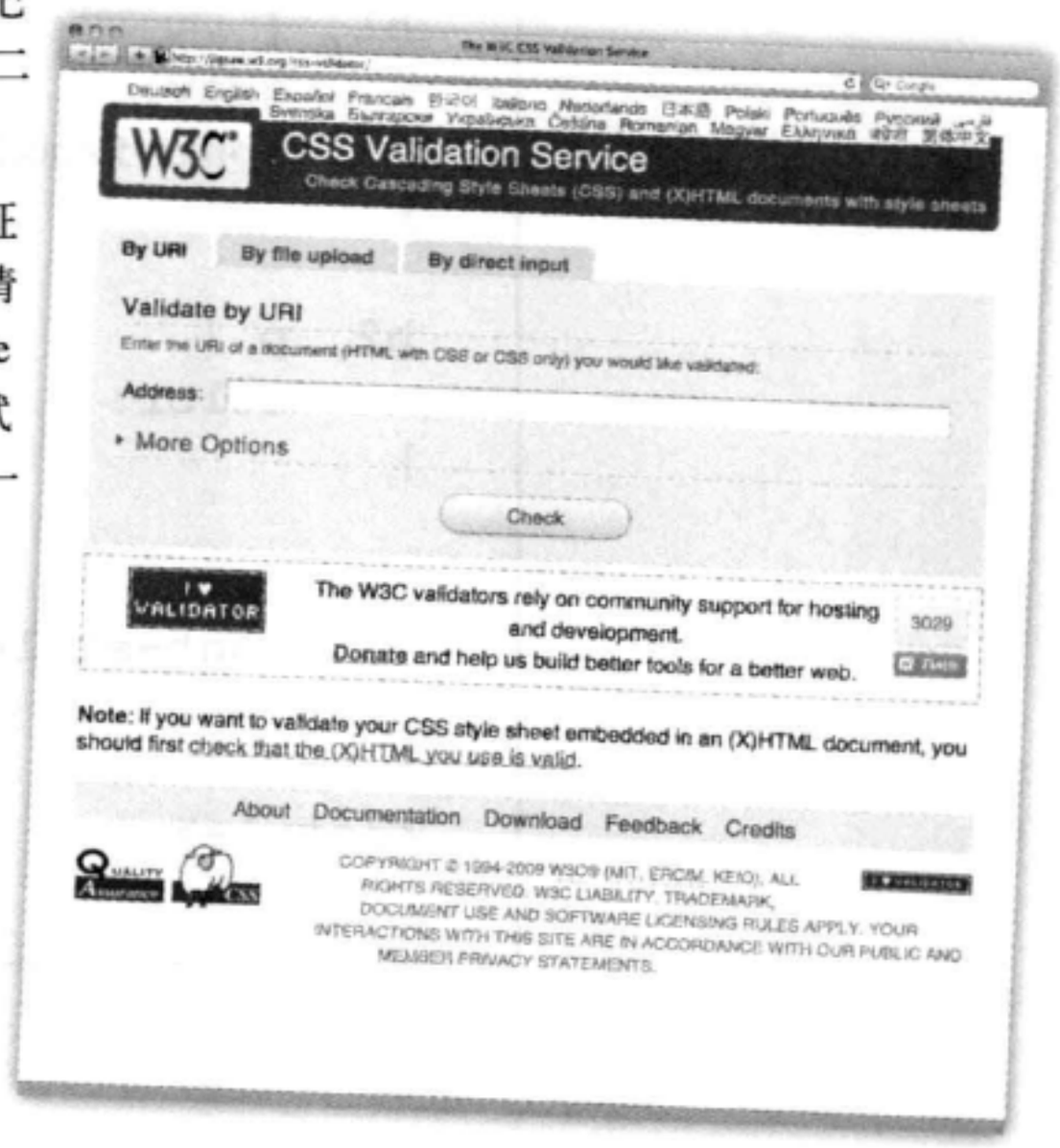
```
<style>

body {
    background-color: white

h1, {
    gray;
    font-family: sans-serif;
}

h2, p {
    color:
}

<em> {
    font-style: italic;
}

</style>
```

当然有！

W3C的那些人可不是混日子的，他们一直在努力工作。可以在这里找到他们的CSS验证工具：

```
http://jigsaw.w3.org/css-validator/
```

在你的浏览器中输入这个URL，相信进入这个页面时你肯定不会感到陌生。你会找到一个验证工具，与HTML验证工具的做法几乎完全相同。要使用这个CSS验证工具，只需要指向你的CSS URL，上传一个包含CSS的文件（第一个标签页），或者直接把CSS粘贴到表单中（第二个标签页），然后提交。



验证CSS时，不会像验证HTML那样遇到“意外情况”，比如需要doctype或字符编码。可以试试看（其实我们也会在下一页给出答案）。

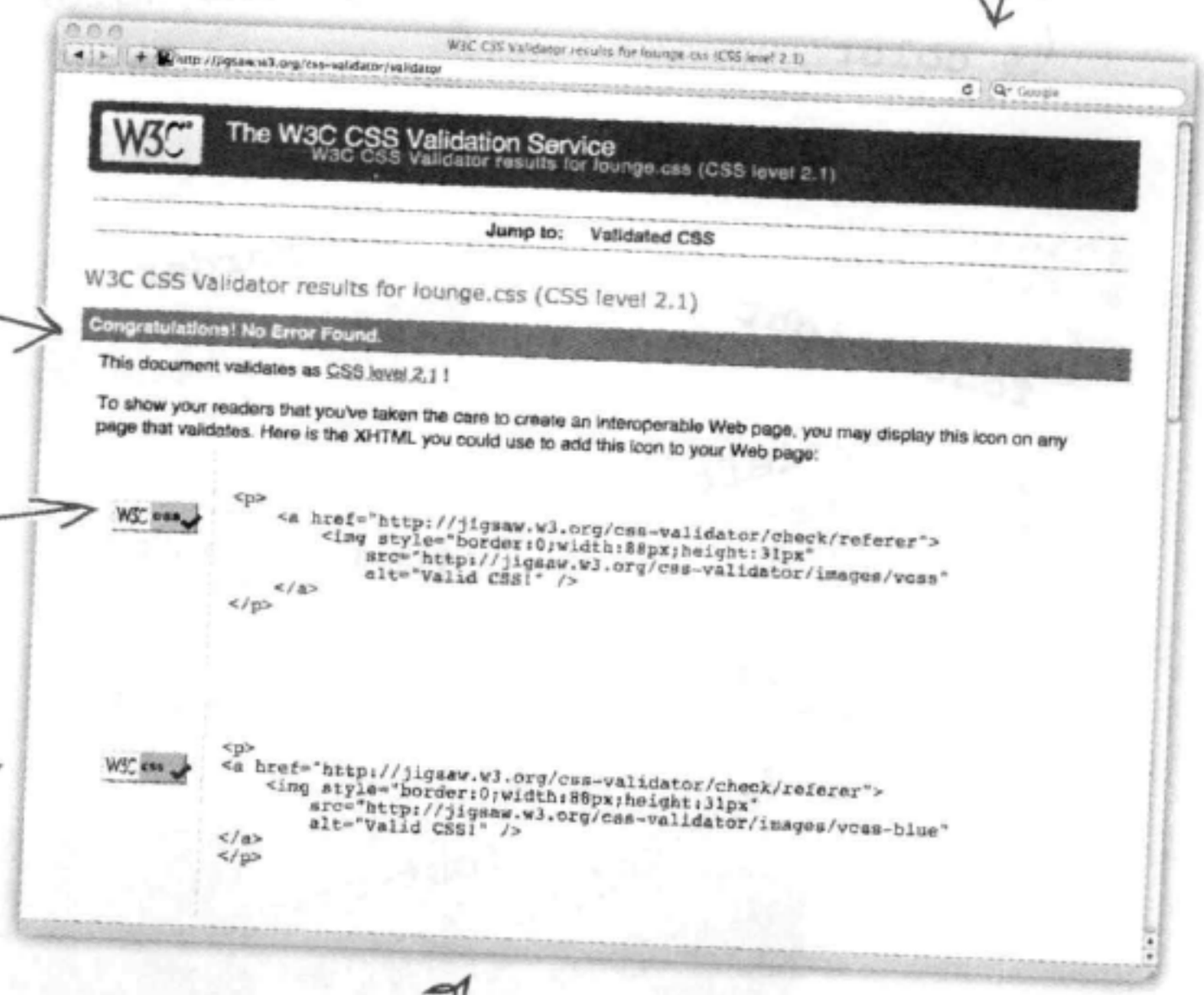
确保休闲室CSS合法

在结束这一章之前，如果Head First休闲室的CSS都能通过验证，你是不是会感觉好一些？当然，这是肯定的。可以用你喜欢的某种方法，把你的CSS提交给W3C。如果你的CSS在服务器上，可以在表单中键入URL。否则，可以上传CSS文件或者直接将CSS复制粘贴到表单中（如果上传文件，一定要确保指向你的CSS文件，而不是HTML文件）。完成后，单击Check（检查）。

如果你的CSS没能通过验证，利用前几页的CSS对照检查，查找你犯的小错误，然后重新提交。

哟！我们的CSS验证为CSS 2.1（验证工具还没有升级到CSS 3，不过等你读到这本书时如果验证工具已经升级，应该也能通过验证）。

这里有一些图标，如果你想炫耀你的CSS通过了验证，可以把它们放在你的Web页面中（验证HTML也能得到类似的图标）。



就像正确验证HTML时一样，CSS通过验证时也会得到这个“绿色的成功条”。只要看到绿色，就说明通过了！

there are no Dumb Questions

问：我需要在意那些警告吗？或者是不是得按警告所说的去做？

答：完全可以将它们忽略，不过你会发现其中一些警告不能算是“强制要求”，更应算是建议。只要有一点奇怪的地方，验证工具就会发出警告，所以只要记住就行了。

属性杂烩汤

top

控制元素顶部的位置。

text-align

使用这个属性将文本左对齐、居中或右对齐。

letter-spacing

这个属性能够在字母之间设置间距，就像这样：L i k e t h i s。

使用color来设置文本元素的字体颜色。

color

background-color

这个属性控制元素的背景颜色。

使用这个属性来设置斜体文本。

font-style

这个属性控制文本的粗细。可以用它设置粗体。

font-weight

border

这个属性在一个元素周围加边框。可以有一个实线边框、凸起边框、虚线边框……

这个属性允许你改变列表中列表项的外观。

list-style

left

利用这个属性指定一个元素的左边所在位置。

padding

如果在一个元素边缘和它的内容之间需要有空间，可以使用padding（内边距）。

这个属性设置一个文本元素中的行间距。

line-height

让文本更大或更小。

font-size

用这个属性在元素后面放置一个图像……

background-image

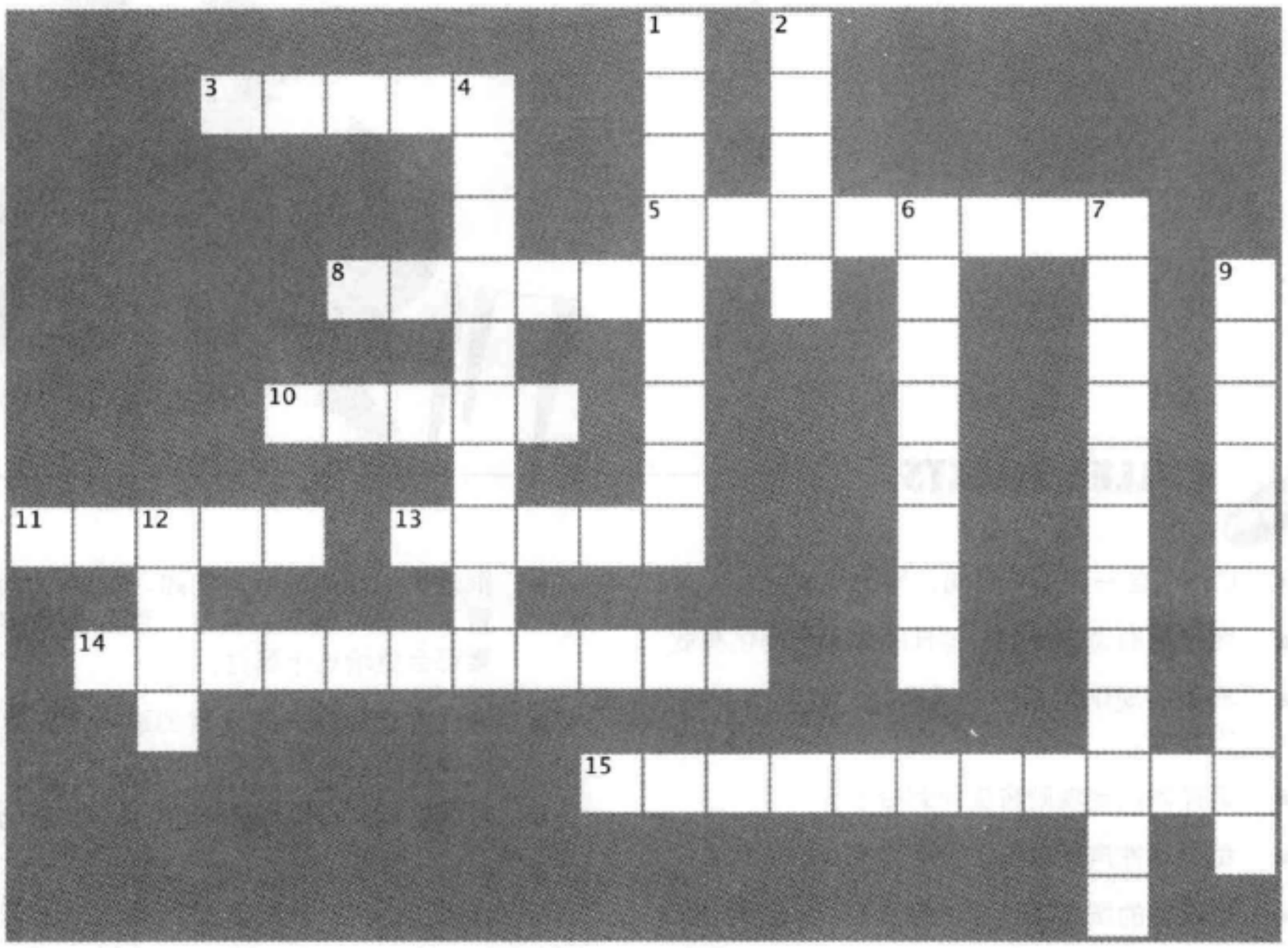
CSS有很多样式属性。这本书后面会看到其中大部分属性，不过现在可以先大体浏览一下，有一个初步的认识，大致了解利用CSS能够控制哪些方面的样式。

BULLET POINTS

- CSS包含一些简单语句，称为规则。
- 每个规则为选择的一些HTML元素提供样式。
- 典型的规则包括一个选择器，以及一个或多个属性和值。
- 选择器指定规则将应用到哪些元素。
- 每个属性声明以一个分号结束。
- 规则中的所有属性和值都放在{ }大括号之间。
- 可以使用元素名作为选择器，来选择任意元素。
- 通过用逗号分隔元素名，可以一次选择多个元素。
- 要在HTML中包含一个样式，最容易的办法就是使用<style>标记。
- 对于HTML以及相当复杂的网站，可能要链接到一个外部样式表。
- <link>元素用于包含一个外部样式表。
- 很多属性都能继承。例如，如果为<body>元素设置了一个可继承的属性，那么<body>的所有子元素都会继承这个属性。
- 通过为你想改变的元素创建一个更特定的规则，能覆盖该元素继承的属性。
- 可以使用class属性将元素增加到一个类。
- 通过在元素名和类名之间加一个“.”，可以选择该类中的一个特定元素。
- 使用“.classname”可以选择属性这个类的所有元素。
- 通过在class属性中放入多个类名，可以指定一个元素属于多个类，类名之间用空格分隔。
- 可以使用W3C验证工具验证CSS（http://jigsaw.w3.org/css-validator）。

HTML填字游戏

这里有一些线索，其中有一些脑筋急转弯，可以帮你从不同角度牢牢记住CSS!

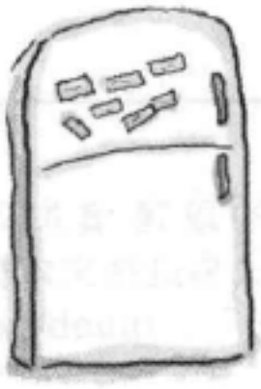


横向

3. 样式在这里定义。
5. 选择一个元素。
8. 每个规则定义了一组属性和 _____。
10. 定义一组元素。
11. 表示字体颜色的属性。
13. 一些字体的装饰部分。
14. 元素如何从其父元素得到属性。
15. 表示字体的属性。

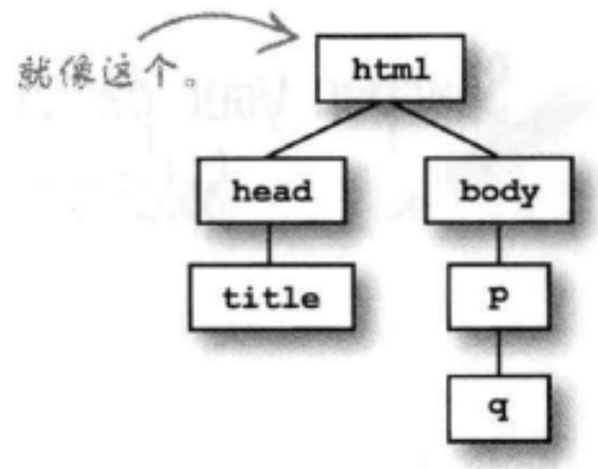
纵向

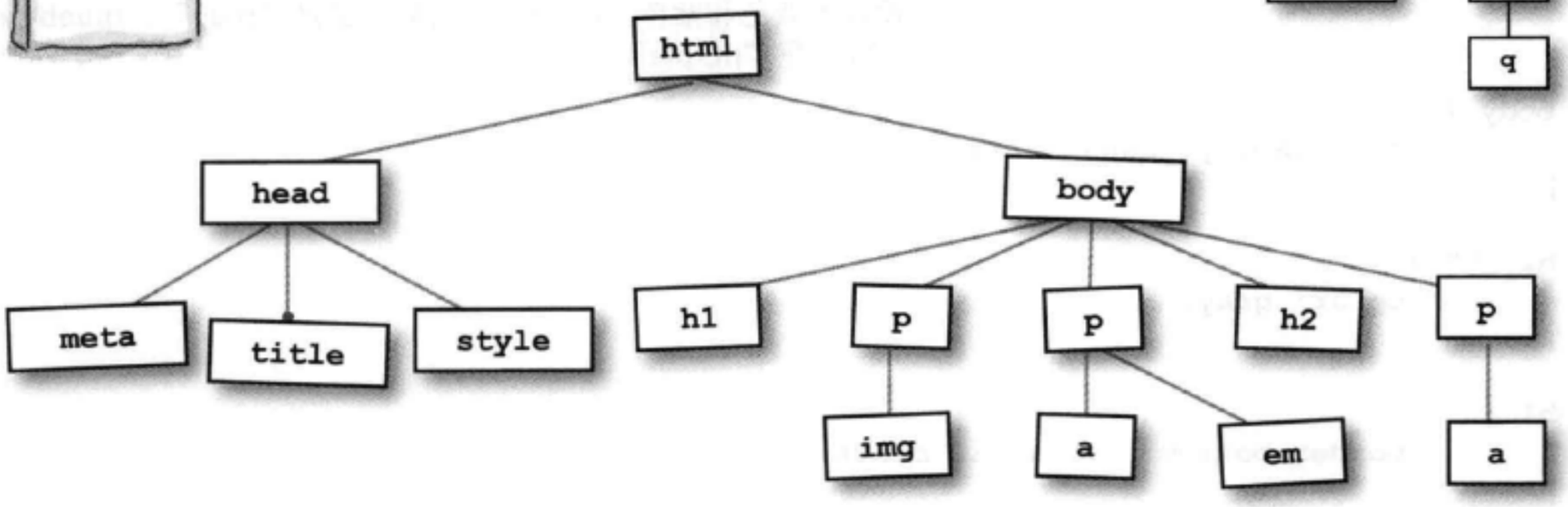
1. 没有上下衬线的字体。
2. 可以把CSS放在HTML文件的这些标记中。
4. 一个外部样式文件称为_____。
6. 利用继承，一个元素上设置的属性可以向下传递到它的_____。
7. 这一次胜出，因为他们使用了外部样式表。
9. 他们迫切希望有一些样式。
12. 使用这个元素来包含一个外部样式表。



标记磁贴答案

还记得第3章中画的HTML元素结构图吗？现在再对休闲室主页面做同样的练习。下面是我们的答案。

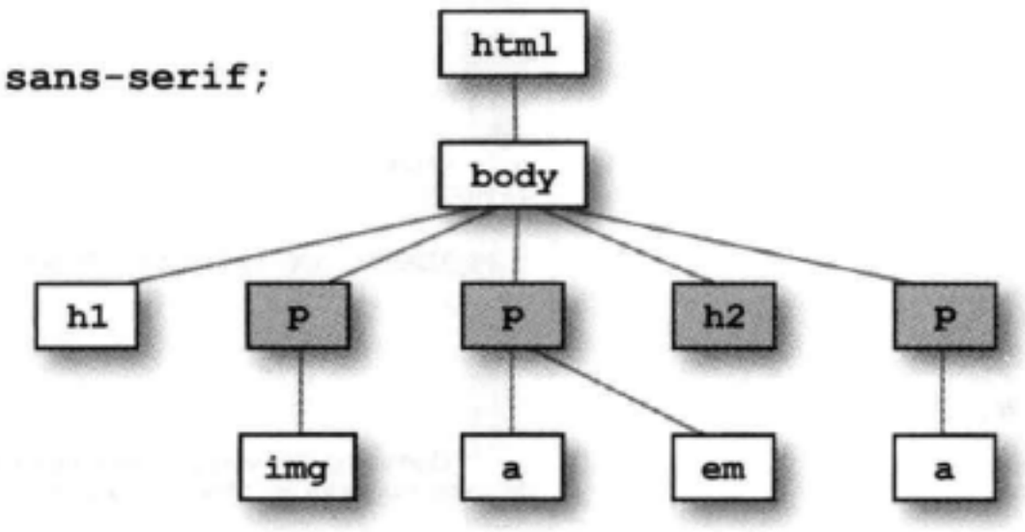




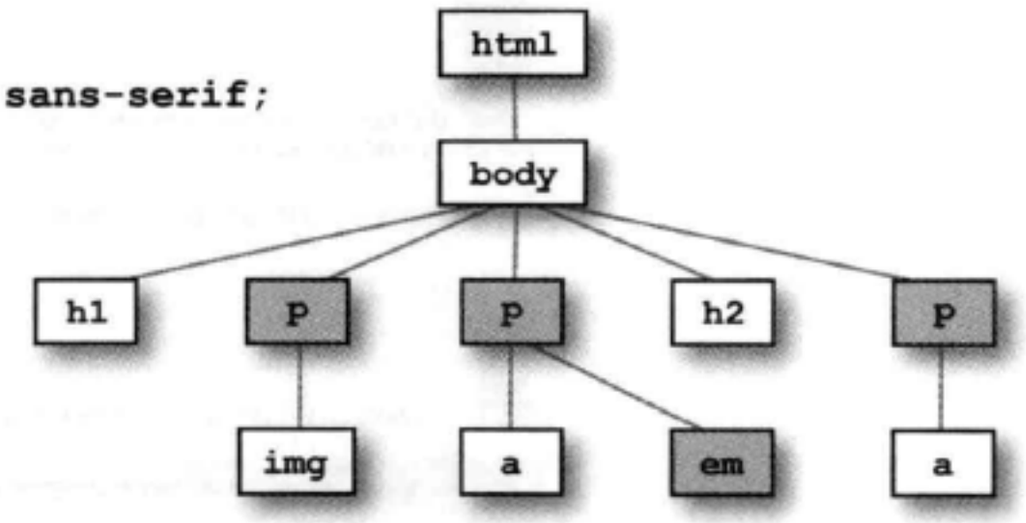
Sharpen your pencil Solution

所选的元素已经涂上颜色。

```
p, h2 {
    font-family: sans-serif;
}
```



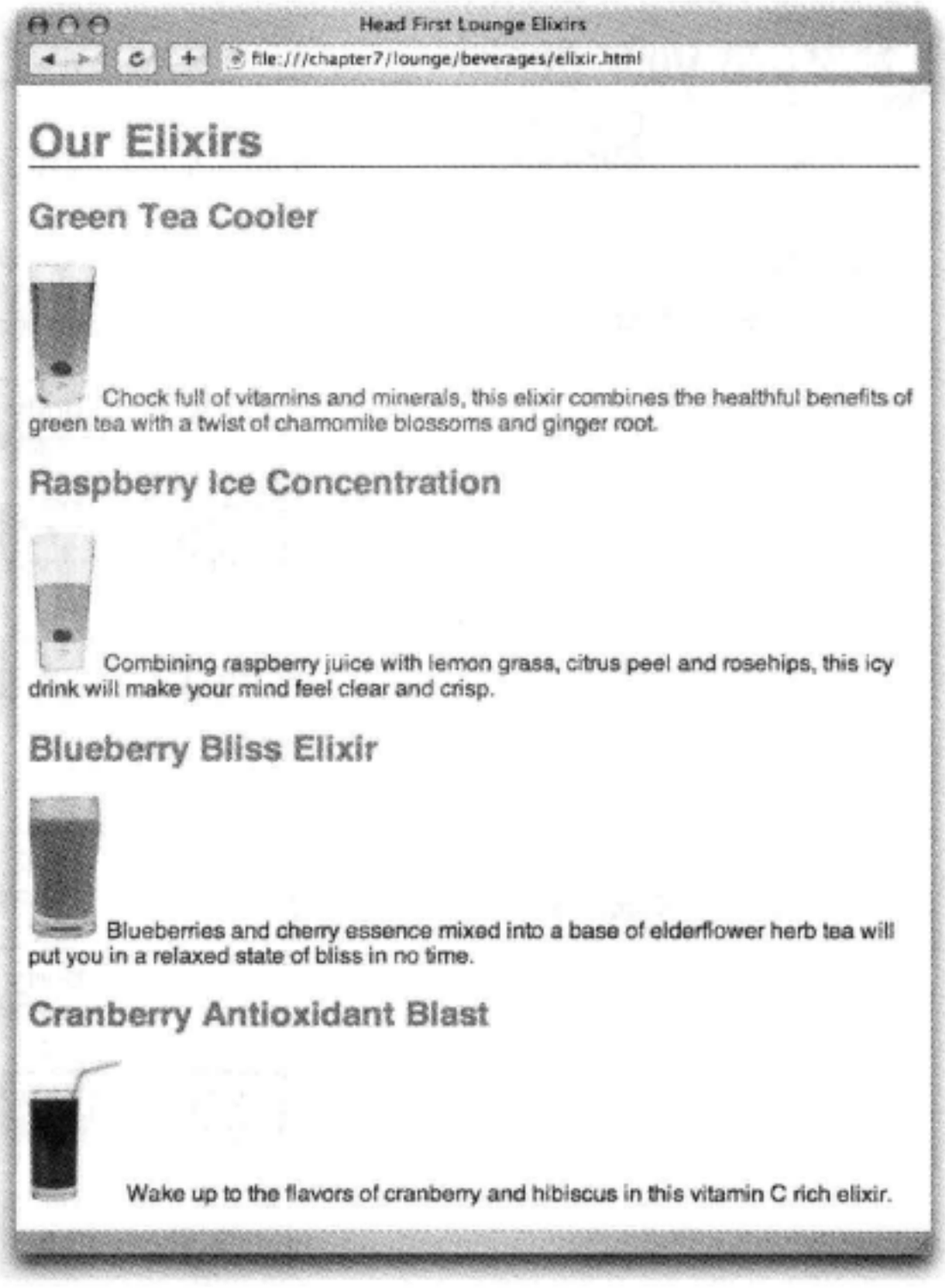

```
p, em {
    font-family: sans-serif;
}
```



Sharpen your pencil Solution

该轮到你了：为“elixir.html”中适当的段落增加两个类“raspberry”和“blueberry”，然后编写样式将这些文本分别设置为蓝色和紫色。raspberry的属性值是“blue”，blueberry的属性值是“purple”。

```
body {
        font-family: sans-serif;
}

h1, h2 {
        color: gray;
}

h1 {
        border-bottom: 1px solid black;
}

p {
        color: maroon;
}

p.greentea {
        color: green;
}

p.raspberry {
        color: blue;
}

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



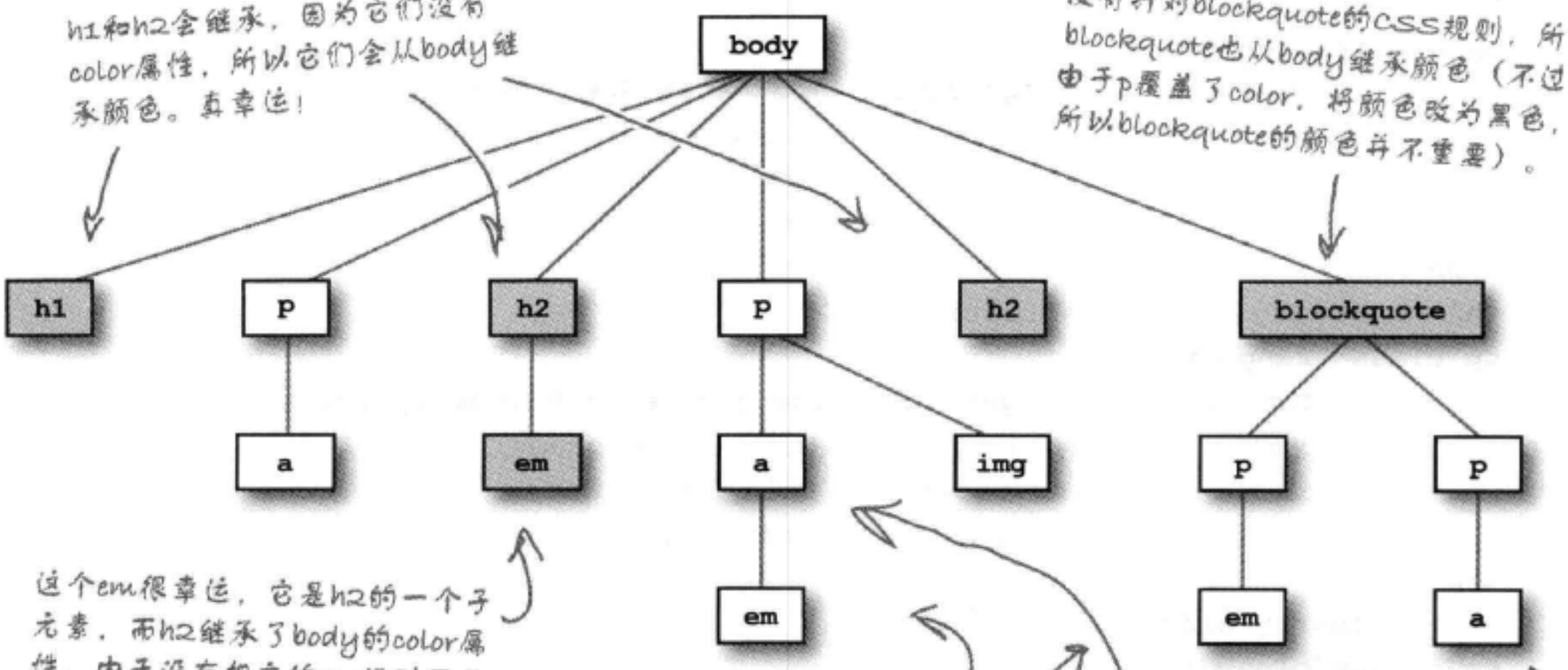
```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge Elixirs</title>
    <link type="text/css" rel="stylesheet" href="../lounge.css">
  </head>
  <body>
    <h1>Our Elixirs</h1>
    <h2>Green Tea Cooler</h2>
    <p class="greentea">
          <img src="../images/green.jpg" alt="Green Tea">
          Chock full of vitamins and minerals, this elixir
          combines the healthful benefits of green tea with
          a twist of chamomile blossoms and ginger root.
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p class="raspberry">
          <img src="../images/lightblue.jpg" alt="Raspberry Ice">
          Combining raspberry juice with lemon grass,
          citrus peel and rosehips, this icy drink
          will make your mind feel clear and crisp.
    </p>
    <h2>Blueberry Bliss Elixir</h2>
    <p class="blueberry">
          <img src="../images/blue.jpg" alt="Blueberry Bliss">
          Blueberries and cherry essence mixed into a base
          of elderflower herb tea will put you in a relaxed
          state of bliss in no time.
    </p>
    <h2>Cranberry Antioxidant Blast</h2>
    <p>
          <img src="../images/red.jpg" alt="Cranberry Blast">
          Wake up to the flavors of cranberry and hibiscus
          in this vitamin C rich elixir.
    </p>
  </body>
</html>
```

Exercise Solution

谁会继承？

嘿，<body>元素已经动身去浏览器了。不过他留下了很多子子孙孙，还让它们继承了颜色“green”。下面给出了他的家族树。把继承<body>元素green颜色的子孙元素标出来。别忘了先看下面的CSS。以下是我们的答案：

```
body {
        color: green;
}

p {
        color: black;
}
```



扮演浏览器答案

下面给出了CSS文件“style.css”，其中有一些错误。你的任务是扮演浏览器，找出所有错误。你找到所有错误了吗？

```
<style>

body {
    background-color: white
h1; {
    gray;
    font-family: sans-serif;
}

h2, p {
    color:
}

<em> {
    font-style: italic;
}

</style>
```

CSS中不能有HTML！<style>标记是HTML，在CSS样式表中无法工作。

缺少分号。

缺少}。

多余的逗号。

缺少属性名和冒号。

缺少属性值和分号。

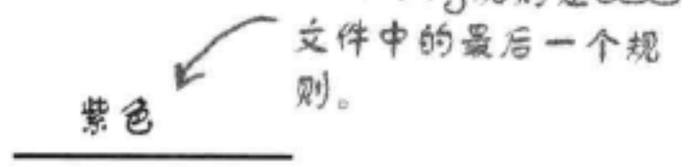
使用了HTML标记而不是元素名。这里应该是em。

CSS中不需要</style>标记。

在你的“elixir.html”文件中，把greentea段落改为包含所有类，就像这样：

```
<p class="greentea raspberry blueberry">
```

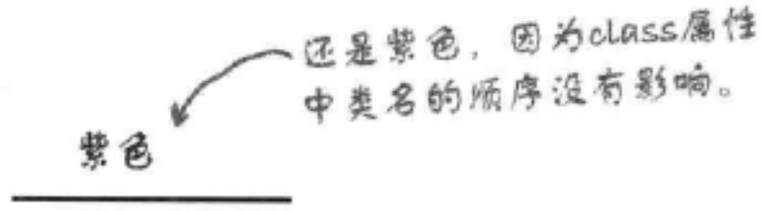
保存并重新加载，现在Green Tea Cooler段落是什么颜色？



接下来，调整HTML中类的顺序：

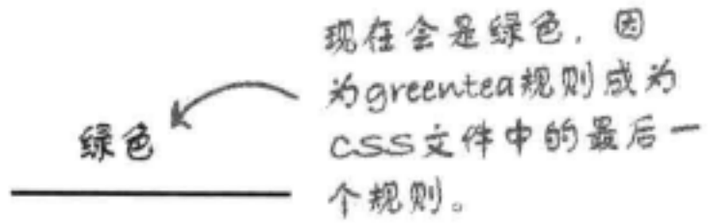
```
<p class="raspberry blueberry greentea">
```

保存并重新加载，现在Green Tea Cooler段落是什么颜色？



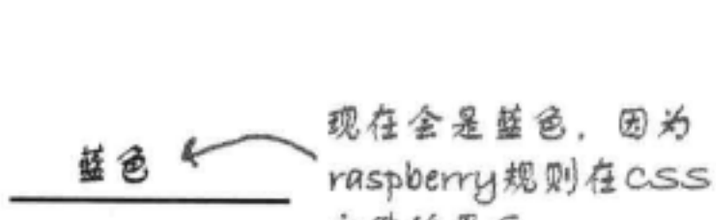
接下来，打开你的CSS文件，把p.greentea规则移到文件最下面。

保存并重新加载，现在Green Tea Cooler段落是什么颜色？



最后，把p.raspberry规则移到最下面。

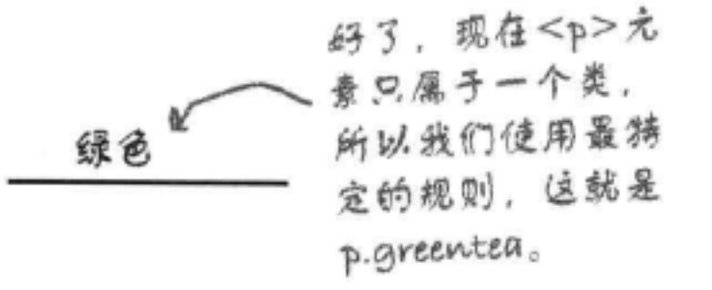
保存并重新加载，现在Green Tea Cooler段落是什么颜色？



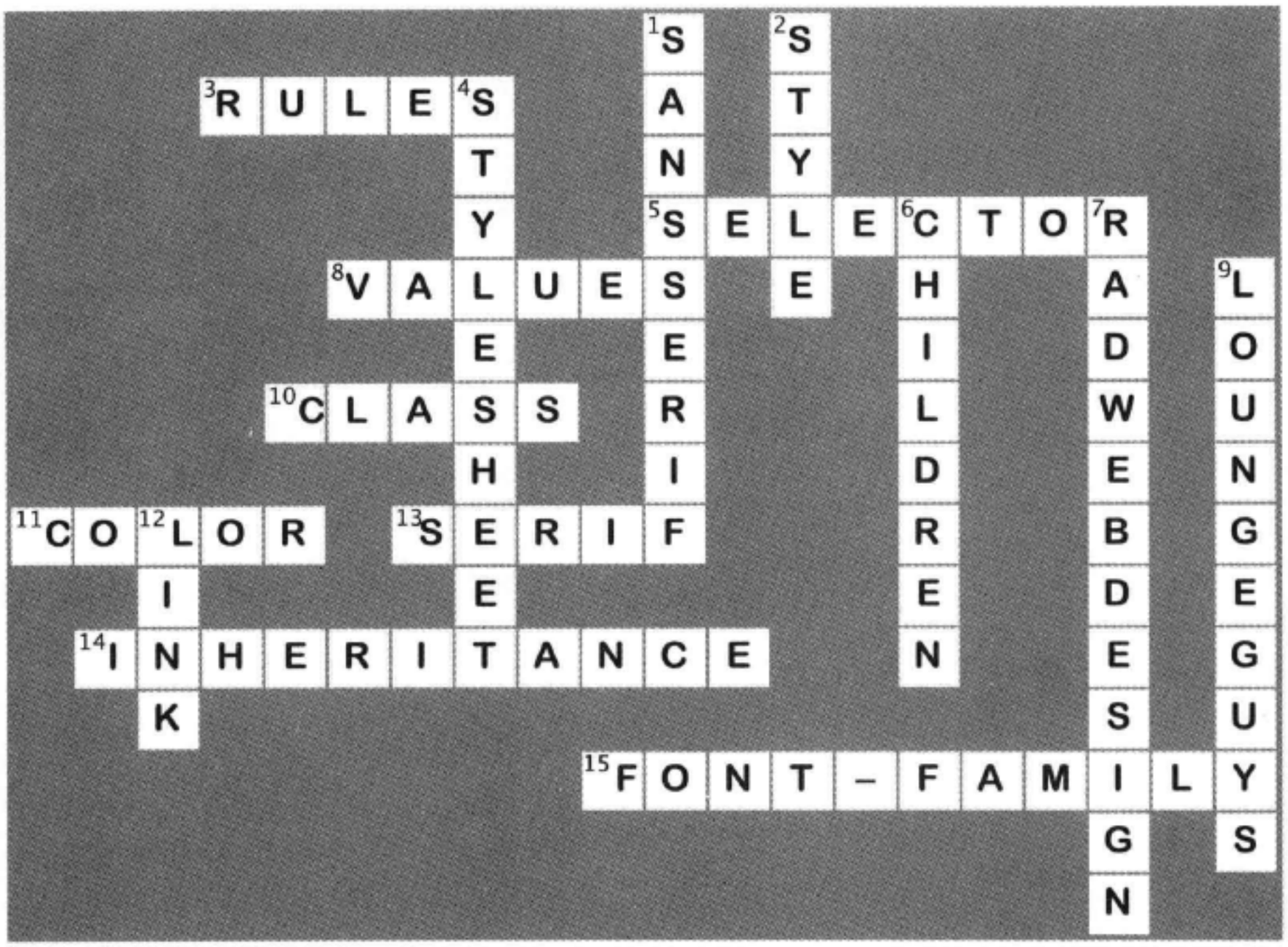
完成之后，把greentea元素重写为原来的样子：

```
<p class="greentea">
```

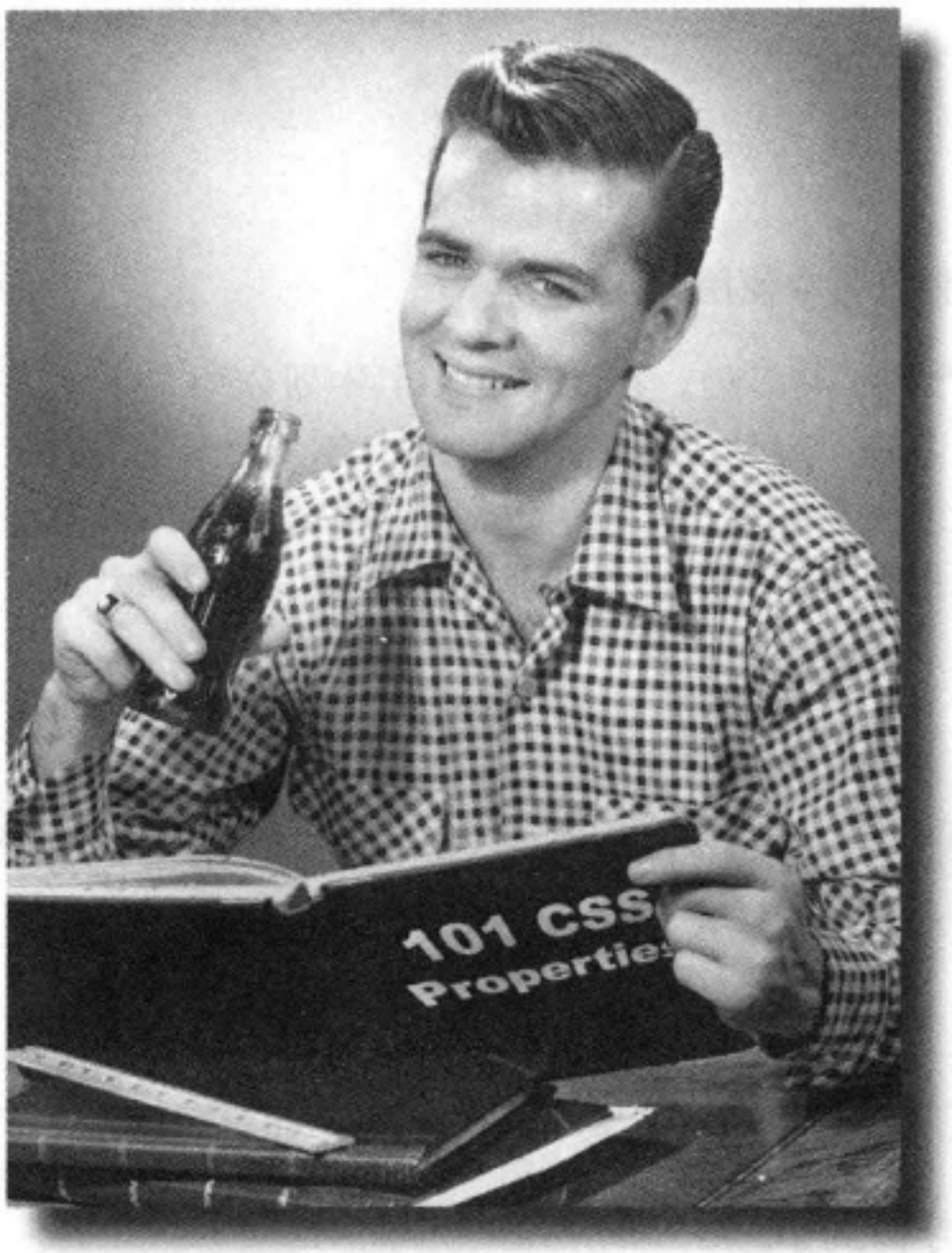
保存并重新加载，现在Green Tea Cooler段落是什么颜色？



HTML填字游戏答案


1S 2S
3RULE 4S
A T
N Y
Y 5SELE 6CTO 7R
8VALUES E HADL
E E I D O
10CLASS R L W U
H I D E N
11CO 12LOR 13SERIF R B G
I E E D E
14INHERITANCE N E G
K S U
15FONT-FAMILY
G S
N

扩大你的词汇量



你的CSS语言课程进行得很顺利。你已经掌握了CSS的基础知识，而且知道了如何创建CSS规则来选择和指定一个元素的样式。现在该扩大你的词汇量了，这意味着需要学习一些新的属性，了解它们能为你做什么。这一章中，我们会介绍影响文本显示的一些最常用的属性。为此，你需要对字体和颜色有所了解。你会发现，不必千篇一律地使用其他人都在用的字体，也不用按部就班地使用浏览器为段落和标题设置的默认字体大小和样式（真的很难看）。你还会看到除了让眼睛看着舒服外，还可以对颜色做很多其他设置。

从三万英尺的高空看文本和字体

有很多CSS属性专门用来帮助你设置文本样式。通过使用CSS，你可以控制文本的字体、风格和颜色，甚至可以控制文本上加的装饰，这些内容都会在这一章中谈到。我们先来研究显示页面使用的具体字体。你已经见过font-family属性，这一章中你会对如何指定字体有更多了解。

在深入分析之前，下面先三万英尺的高空宏观地看一下，哪些属性可以用来指定和改变字体的外观。然后我们会逐个地学习使用各个字体要注意的细节问题。

Andale Mono
Arial
Arial Black
Comic Sans
Courier New
Georgia
Impact
Times New Roman
Trebuchet MS
Verdana

用font-family属性定制页面中使用的字体。

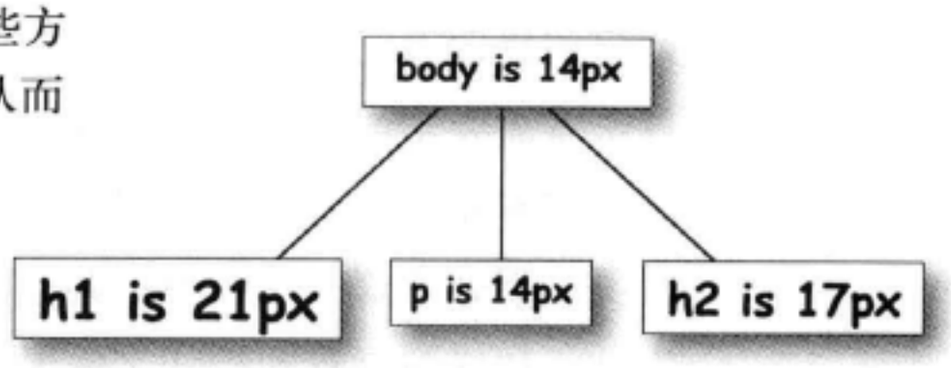
字体会对页面设计产生巨大影响。在CSS中，字体划分为“字体系列”（font family），你可以从中指定希望页面中各个元素使用的字体。大多数计算机上通常只安装了部分字体，所以在选择字体时要特别当心。为了有效地指定字体和最大程度地利用字体，这一章中我们会带着你学习时需要了解的全部知识。

不过，稍后还会看到，你还可以扩展浏览器可用的字体。

```
body {
      font-family: Verdana, Geneva, Arial, sans-serif;
}
```

用font-size属性控制字体大小。

字体大小（font size）对于Web页面的设计和可读性也有很大影响。用CSS指定字体大小有很多方法，这一章中我们会分别介绍这些方法，不仅如此，我们还会教你使用一种特殊方法指定字体，从而允许用户调整字体大小而不影响你的设计。

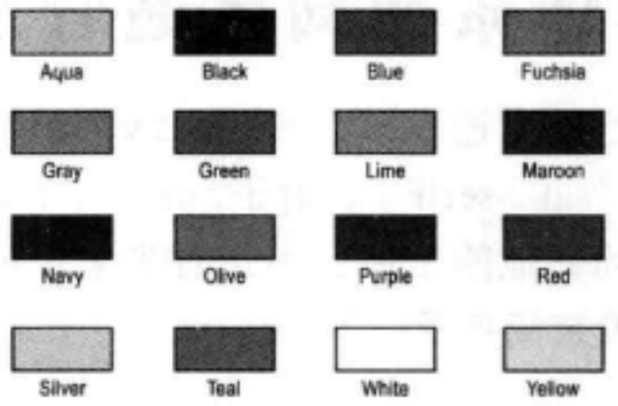



```
body {
      font-size: 14px;
}
```

用color属性为文本设置颜色。

可以用color属性改变文本颜色。为此，对Web颜色有所了解会很有帮助，我们会带你学习有关颜色的所有细节，包括神秘的颜色“十六进制码”。

```
body {
      color: silver;
}
```



用font-weight属性影响字体的粗细。

既然能在需要时为字体指定特定的粗细，怎么能满足于那些乏味平庸的字体呢？或者你的字体是不是看起来太粗了？可以让它苗条些，有正常的粗细。所有这些都可以利用font-weight属性轻松解决。

```
body {
      font-weight: bold;
}
```

lighter

normal

bold

bolder

使用text-decoration属性为文本增加更多风格。

通过使用text-decoration属性，可以对文本加一些装饰，包括上划线、下划线和删除线。

```
body {
      text-decoration: underline;
}
```

none

underline

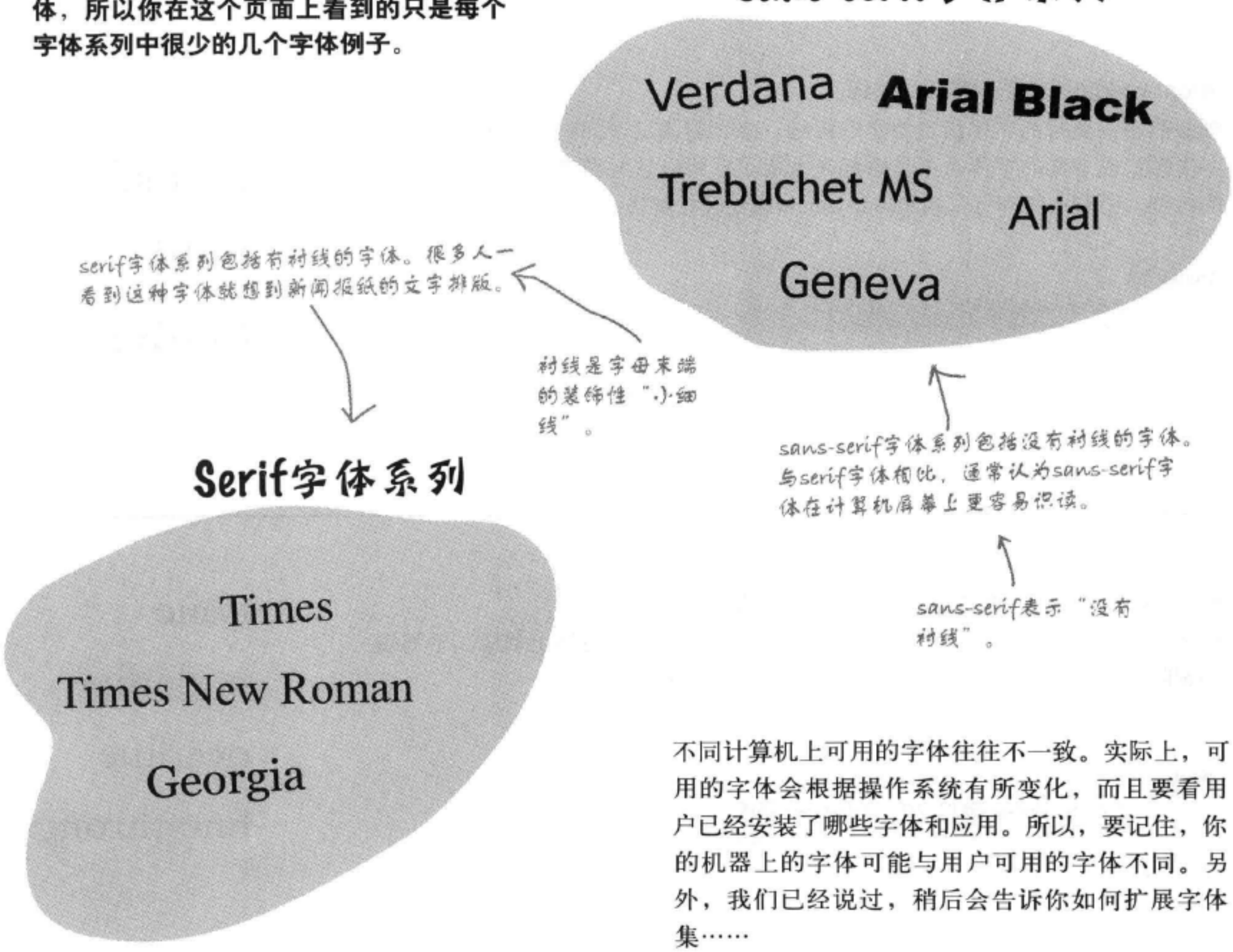
overline

~~line-through~~

字体系列到底是什么？

你已经见过font-family属性，到目前为止总是将这个属性的值指定为“sans-serif”。对于font-family属性还可以更有创意一些，可以指定为其他的字体，不过首先来了解字体系列是什么，这会很有帮助。下面就来简单看一下……

每个font-family包含一组有共同特征的字体。共有5个字体系列：sans-serif、serif、monospace、cursive和fantasy。每个字体系列都包括大量字体，所以你在这个页面上看到的只是每个字体系列中很少的几个字体例子。



不同计算机上可用的字体往往不一致。实际上，可用的字体会根据操作系统有所变化，而且要看用户已经安装了哪些字体和应用。所以，要记住，你的机器上的字体可能与用户可用的字体不同。另外，我们已经说过，稍后会告诉你如何扩展字体集……

Monospace字体系列

Courier

Courier New

Andale Mono

monospace字体系列中的字体包含固定宽度的字符。例如，一个“i”在水平方向所占的宽度与一个“m”所占的宽度是相同的。这些字体主要用于显示软件代码示例。

仔细查看这些字体系列：serif字体看起来很高雅、很传统，而sans-serif字体外观很清晰，可读性好。Monospace字体就好像是打字机打出来的。Cursive和fantasy字体给人一种有趣或者很有风格的感觉。

cursive字体系列包括看似手写的字体。有时会看到标题中使用这些字体。

Cursive字体系列

Comic Sans

Apple Chancery

Fantasy字体系列

LAST NINJA

Impact

fantasy字体系列包含有某种风格的装饰性字体。

字体磁贴

你的任务是帮助下面的虚构字体找到回家的路，回到它们自己的字体系列中。把左边的各个冰箱磁贴放在右边正确的字体系列中。继续学习下面的内容之前，请检查你的答案。如果有必要，可以再复习一下前两页的字体系列描述。

Bainbridge

Cartoon

Palomino

Angel

Iceland

Messenger

Savannah

Crush

Nautica

Quarter

Monospace 字体系列

Fantasy字体系列

Cursive 字体系列

Sans-serif字体系列

Serif字体系列

使用CSS指定字体系列

OK，这么说已经有多个字体系列，其中包括很多不错的字体。怎么在页面上使用这些字体呢？嗯，你在上一章已经见过font-family属性，在那里你为休闲室页面指定font-family为“sans-serif”。再来看一个更有意思的例子：

通常，你指定的font-family（即font-family规范）包含一个候选字体列表，它们都来自同一个字体系列。

```
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
}
```

使用font-family属性可以指定多个字体。只需要输入这些字体名，并用逗号分隔。

要完全按照这里的拼写输入字体名，大小写字母必须一致。

最后总是放一个通用的字体系列名，如“serif”、“sans-serif”、“cursive”或“monospace”。稍后你会了解这个通用字体系列名的作用。

font-family规范如何工作

浏览器会这样解释你在font-family规范中列出的字体：

查看用户计算机上是否有Verdana字体，如果有，这个元素（在这里就是<body>元素）就会使用这个字体。

如果Verdana不可用，再查找Geneva字体，如果这个字体可用，它将作为body元素的字体。

如果Geneva不可用，再查找字体Arial，如果这个字体可用，它将作为body元素的字体。

最后，如果前面指定的特定字体都没有找到，就使用浏览器的默认“sans-serif”字体。

```
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
}
```

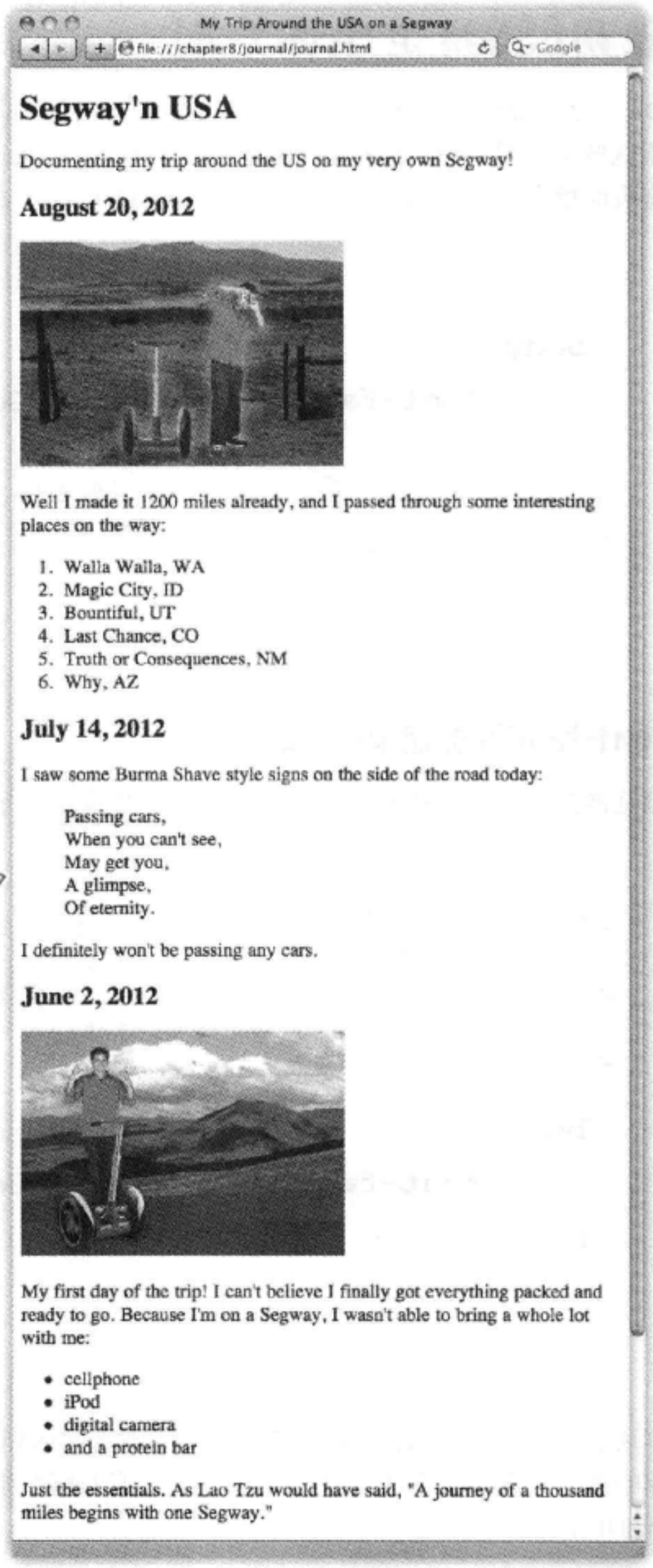
没有必要非得指定4种候选字体。你也可以指定2种、3种等。上一章中，我们只使用了一个字体（就是默认的sans-serif字体），不过我们不建议这么做，因为这样就没办法对所要使用的字体做太多控制。

利用font-family属性，你可以创建一个首选字体列表。我们希望大多数浏览器都能有你的第一个选择，不过，如果没有，至少可以确保浏览器能提供同一个字体系列中的一个通用字体。

下面在页面中加入一些字体……

让Tony的旅行日志焕然一新

既然知道了如何指定字体，下面再来看Tony的Segway'n USA页面，让它改头换面。我们会对Tony页面中的文本样式渐进地做一些小小的改动，尽管每一个改动都不会让页面发生显著变化，但是我们相信，到这一章结束时，你肯定会承认这个网站有了一个全新的面貌。下面来考虑可以在哪些方面进行改进，然后为Tony提供一个新的font-family。



要记住，我们没有对Tony的网站应用任何样式，所以在他的网站中，整个页面只有一个font-family (serif)。

标题字体的默认大小太大了，不利于建立一个漂亮的页面。

这里的引用只是有缩进。如果能增加一些font-style改进它的外观会更好。

除了照片外，这个页面相当乏味，所以我们还要增加一些字体颜色，让它更生动一点。

为Tony指定一个新的font-family

下面为Tony指定一个font-family。我们先从一些简洁的sans-serif字体开始。首先，在“chapter8/journal”文件夹中创建一个新文件“journal.css”，增加以下规则：

```
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
}
```

我们要设置<body>元素的font-family属性。要记住，<body>中的元素会继承这些字体。

大多数PC上都有Verdana字体……

……另外大多数Mac上都有Geneva。

Arial在PC和Mac上都很常见。

如果其他字体都找不到，就使用默认的sans-serif字体。

这里我们设置了一组sans-serif字体。

现在需要把Tony的旅行日志链接到这个新的样式表文件。为此，打开“chapter8/journal”文件夹中的文件“journal.html”。增加<link>元素，链入“journal.css”中的样式，如下所示：

我们已经更新了Tony的journal.html文件，让它成为真正的HTML5，这里加入了doctype和<meta>标记。

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <link type="text/css" rel="stylesheet" href="journal.css">
    <title>My Trip Around the USA on a Segway</title>
  </head>
  <body>
    .
    .
    .
  </body>
</html>
```

在这里链入新的“journal.css”文件。

完成这个修改后，保存文件，然后打开你的浏览器，加载这个页面。

测试Tony的新字体

在浏览器中打开增加了新CSS的页面，你会看到现在得到了一组漂亮的sans-serif字体。下面来介绍有哪些变化……

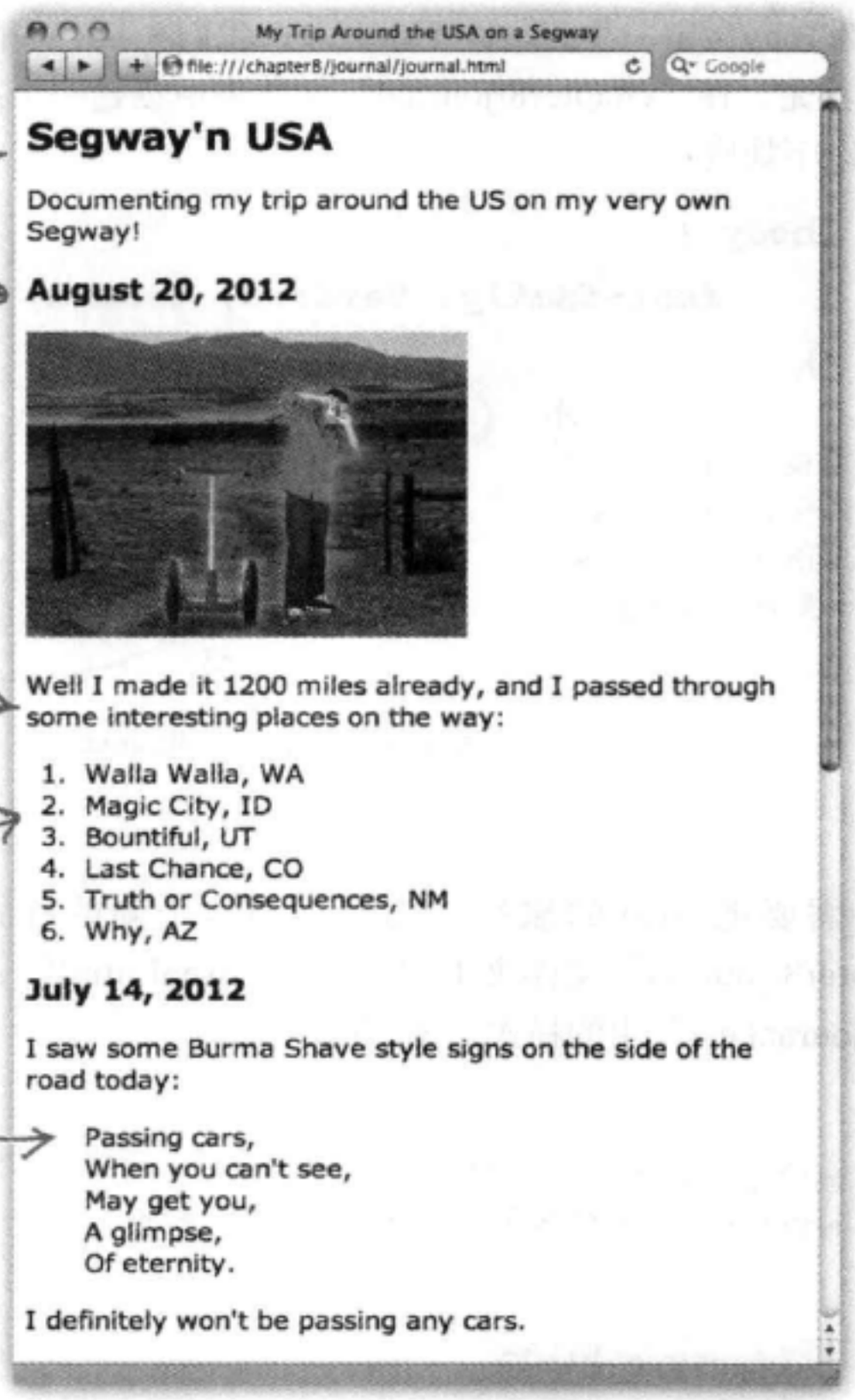
这些字体确实让Tony的Web页面焕然一新。没有字母上下的衬线（serif），标题现在更为清晰，不过在这个页面上看着还是有点太大。

段落文本也很清晰，很适合阅读。

由于font-family是一个可继承的属性，页面上所有元素现在都使用sans-serif字体，甚至包括列表元素……

……和<blockquote>。

如果你更喜欢serif字体，可以不受我们的影响。你可以调整font-family声明，仍然使用serif字体。



there are no Dumb Questions

问：这么说，font-family属性实际上就是一组候选字体，是吗？

答：没错。它实际上就是一个字体优先列表。第一个是你最希望使用的，后面是一个不错的替代字体，再后面是更多的替代字体。对于最后一个字体，应当指定最全面的通用“sans-serif”或“serif”，它与列表中指定的所有其他字体应当在同一个字体系列中。

问：如果一个字体名中包含多个单词，比如Courier New，我怎么指定这样一个字体呢？

答：只需要在font-family声明中的字体名两边加上引号，就像这样：font-family:"Courier New", Courier;

问：“serif”和“sans-serif”是真正的字体吗？

答：“serif”和“sans-serif”并不是具体的字体名。不过，如果font-family声明中指定的前几个字体都无法找到，浏览器就会选择一个实际字体来代替“serif”或“sans-serif”。取代它的字体是浏览器定义的该字体系列的默认字体。

问：我怎么知道使用哪个字体？Serif还是sans-serif？

答：没有固定的原则。不过，在计算机显示中，很多人认为sans-serif最适合体文本。你会发现很多设计中体文本都使用sans-serif字体，或者会混合使用serif字体和sans-serif字体。所以，这实际上取决于你的喜欢，以及你希望页面有怎样的外观。

每个人都有不同的字体，我该如何处理？

关于字体，很不幸的是，你没有办法控制用户计算机上安装什么字体。不仅如此，他们还往往使用不同的操作系统。例如，也许你的Mac上有某些字体，而在用户PC上很可能没有这些字体。

所以，如何处理这种情况呢？嗯，一种靠得住的策略是创建一个字体列表，其中包含最适合你的页面的字体，并希望用户已经安装了这些字体。不过，如果他们没有这些字体，至少我们可以依靠浏览器来提供相同字体系列中的一种通用字体。

下面来详细分析如何做到。你要确保你的`font-family`声明包含Windows和Mac（以及你的用户可能使用的任何其他平台，如Linux，可能还有移动设备）上都可能有的字体，而且最后要提供一个字体系列。

下面给出一个例子：

这些字体在Windows和Macintosh计算机上可能都有。

Andale Mono
Arial
Arial Black
Comic Sans
Courier New
Georgia
Impact
Times New Roman
Trebuchet MS
Verdana

这些字体最有可能出现在Macintosh计算机上。

Geneva
Courier
Helvetica
Times

下面再来看我们为Tony的页面声明的字体……

```
font-family: Verdana, Geneva, Arial, sans-serif;
```

(1) 我们希望使用Verdana，不过……

(2) 如果没有这个字体，Geneva也不错，不过这个字体可能在Mac上才有。如果没有这个字体……

(3) 也没关系，因为我们基本上可以相信Windows或Mac上都有Arial字体，不过如果这个字体也没有……

(4) 那也没关系，可以让浏览器为我们选择一个sans-serif字体。

现在我知道了，需要指定所有用户机器上都可用的字体，不过我真的希望能使用这个Emblema One字体，这个字体很酷，是我专门为主标题找的字体。我还是想用它，如果用户确实没有这个字体，可以使用一种其他字体作为退路，这样行吗？

可以，不过还有一种更好的办法……

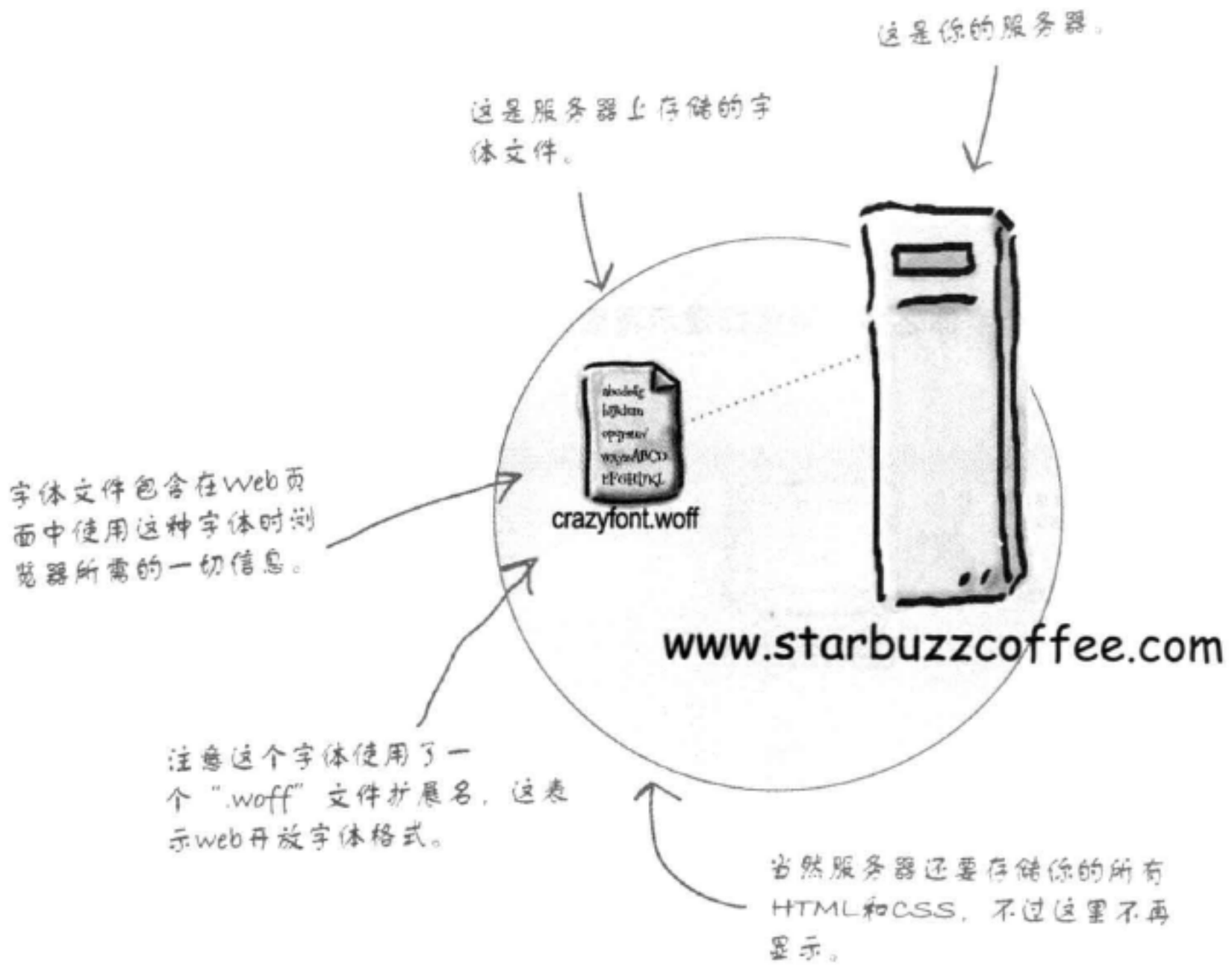
你的建议是可以的，不过很可能只适合一小部分用户。如果你确实必须使用这种“棒极了”的字体，或者排版对你的网站设计很重要，那么你可以使用Web字体（Web Font）向用户的浏览器提供一种字体。

为此，要用到CSS的一个比较新的特性：`@font-face`规则。这个规则允许你定义一种字体的名字和位置，然后可以在你的页面中使用。

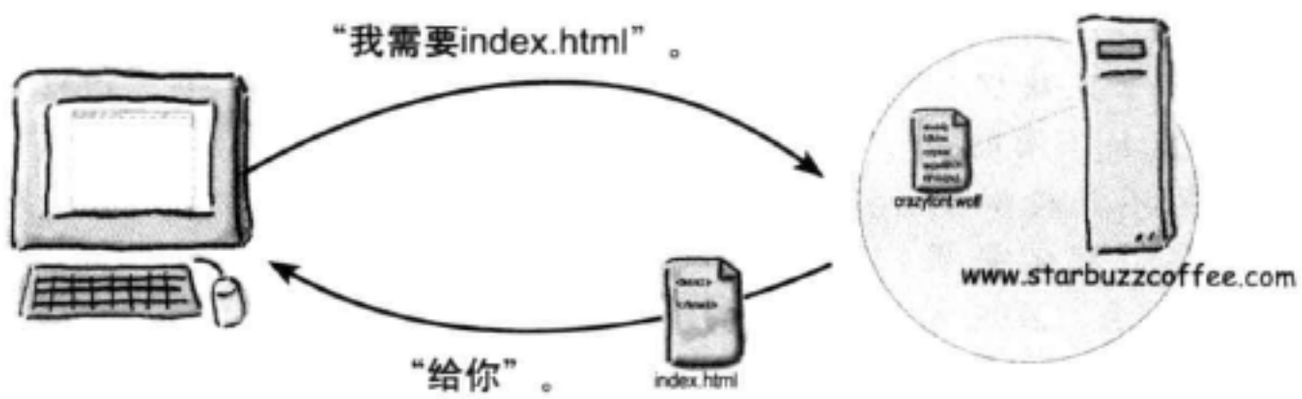
下面来看这是怎么做到的……

Web字体如何工作

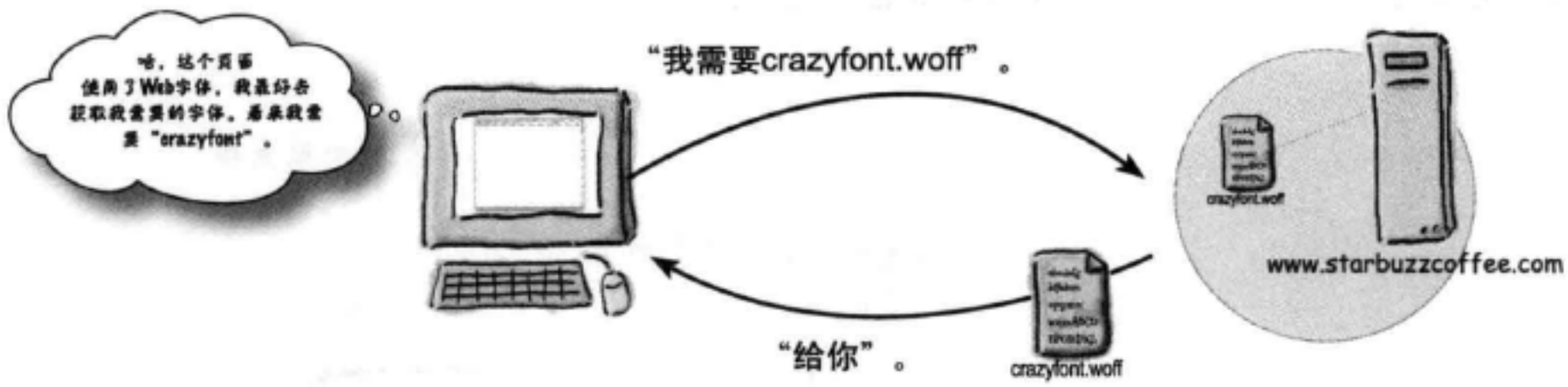
有了Web字体，你可以充分利用现代浏览器的一个新功能，能够向用户直接提供新的字体。一旦提供了新字体，浏览器就能使用Web字体，就像使用所有其他字体一样，甚至可以用CSS指定文本样式。下面详细分析Web字体如何工作：



❶ 要利用Web字体，浏览器首先获取一个引用这些字体的HTML页面。



2 浏览器再获取这个页面所需的Web字体文件。



3 现在，获取了这个字体之后，浏览器显示页面时就会使用这个字体。

there are no Dumb Questions

问：woff或者Web开放字体格式（Web open font format）到底是什么？

答：Woff是作为Web字体的标准字体格式出现的，你会看到，如今所有现代浏览器上都对Woff提供了支持。也就是说，这个领域以前一直缺少标准化，不同的浏览器会支持不同的字体格式。如果要为可能不支持woff的浏览器提供Web字体，就需要提供一个或多个格式作为候选。在这方面，Web字体托管服务会有很大帮助。

问：所以要使用一个Web字体，我必须在一个服务器上托管字体文件，是吗？

答：如果你只是要测试字体，实际上可以把这些字体作为本地文件，存储在你自己的文件系统中并引用（就像存储和引用本地图像一样）。不过，如果你想为Web上的用户提供字体，就必须把这些文件放在一个服务器上，或者利用一个托管服务，如Google的字体托管服务，这是免费的。

问：如果我要使用一个Web字体，能不能肯定我的用户一定能用这种字体？

答：只要他们有一个现代浏览器（另外不考虑网络连接或服务器问题），那么大多数情况下都可以确信这一点。不过，如果他们使用一些比较老的浏览器，或者不支持Web字体的移动设备，那就不行了，你仍然需要提供候选字体（稍后我们来介绍这个内容）。

如何为页面增加Web字体……

想为页面增加一种特殊的字体？下面一步一步说明如何使用Web字体和CSS中的`@font-face`规则来实现。

第1步：找到一个字体

如果还没有合适的字体，你可以像Tony一样，访问一些提供字体的网站，现在这些网站有很多，会提供免费以及需要授权的字体，允许你在页面中使用（有关的更多信息请参考本书附录）。我们将采纳Tony的建议，使用Emblema One，这是一个免费的字体。

第2步：确保有所需字体的所有格式

关于Web字体有一个好消息：`@font-face` CSS规则基本上已经成为所有现代浏览器的一个标准。不过也有一个坏消息：存储字体使用的具体格式还不是一个标准（不过已经离这个目标越来越接近了）。实际上，不同的浏览器对很多不同的格式提供了不同程度的支持（写这本书时）。下面是一些常用的格式（以及各自的文件扩展名）：

TrueType字体: .ttf

OpenType字体: .otf

TrueType和OpenType字体紧密相关，OpenType建立在TrueType基础之上（比TrueType更新）。

Embedded OpenType字体: .eot

Embedded OpenType (EOT)是OpenType的一种压缩形式。这种格式是专用的（Microsoft），仅IE提供支持。

SVG字体: .svg

Scalable Vector Graphics或SVG是一种通用图像格式，SVG字体使用这种格式表示字符。

Web开放字体格式: .woff

Web开放字体格式建立在TrueType基础之上，已经发展为Web字体的一个标准。大多数现代浏览器都对这种格式提供了很好的支持。

大多数现代浏览器上支持最为广泛的格式是Web开放字体格式，所以这也是我们推荐你使用的格式。你可以为较老的浏览器提供一个候选字体，我们将使用TrueType作为候选，因为这种字体在所有浏览器上也得到了很好的支持（IE除外）。

第3步：把你的字体文件放在Web上

你可能想把你的字体文件放在Web上，这样用户的浏览器就能访问这些字体。或者也可以利用网上的很多在线字体服务，这些服务会为你托管这些文件。不论哪一种情况，你都需要字体文件的URL。下面是Tony的文件，我们已经把这些文件放在`wickedlysmart.com`上：

```
http://wickedlysmart.com/hfhtmlcss/chapter8/journal/EmblemaOne-Regular.woff
http://wickedlysmart.com/hfhtmlcss/chapter8/journal/EmblemaOne-Regular.ttf
```

第4步：在CSS中增加@font-face属性

你已经得到了“Emblema One”字体的.woff和.ttf版本的URL，所以现在可以为“journal.css”文件增加一个@font-face规则。把这个规则增加到文件的最上面，放在body规则之上：

规则以@font-face开头。

与正常的规则不同，正常规则会选择一组元素并指定样式，而@font-face规则会建立一个字体，将指定一个font-family名，以便以后使用。

在@font-face规则中，我们使用font-family属性为这个字体创建一个名字。可以使用你喜欢的任何名字，不过通常最好与字体名一致，如“Emblema One”。

```
@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");
}
```

src属性告诉浏览器在哪里可以得到这个字体。对于浏览器可识别的每一个文件，我们要分别指定一个src值。在这里，我们将提供现代浏览器可以识别的.woff和.ttf字体。

@font-face规则告诉浏览器：要加载由src URL指定的字体文件。浏览器会尝试加载每一个src文件，直到找到它能支持的一个文件。一旦加载，会为这个字体分配font-family属性中指定的名字，在这里就是“Emblema One”。现在来看如何在页面样式中使用这个字体。

提示：你已经知道该如何使用！

第5步：在CSS中使用font-family名

一旦用@font-face规则在浏览器中加载一个字体，接下来就可以使用这个字体了，可以用font-family属性引用你指定的字体名。下面改变Tony页面中<h1>标题的字体，让它使用“Emblema One”字体。为此，我们要为<h1>增加一个规则，如下所示：

```
h1 {
  font-family: "Emblema One", sans-serif;
}
```

我们指定了字体名，这很正常，只不过这一次是使用@font-face加载的字体！为以防万一，我们还指定了sans-serif作为退路。

第6步：加载页面！

大功告成！现在来测试你的字体。重新加载Tony的日志页面，翻开下一页看看效果怎么样……

测试Tony日志页面中的Web字体

重新加载“journal.html”，你会看到页面最上面的<h1>标题会使用Emblema One字体。只用几行CSS就能达到这样的效果，真不错！

现在，Tony日志页面最上面的<h1>标题使用的是“Emblema One”字体。

Watch it!

TTF和WOFF字体格式在IE8及以前的版本中不可用。

如果你希望支持那些使用较老IE浏览器的用户，需要对Web字体多做一点工作，要使用一个EOT字体。

there are no Dumb Questions

问：@font-face规则不论看起来还是从表现上讲都不像一个CSS规则，不是吗？

答：你说对了，可以认为@font-face是一个内置的CSS规则，而不是一个选择器规则。@font-face并不选择一个元素。利用@font-face规则，可以获取一个Web字体，并为它分配一个font-family名。最前面的@就是一个很好的线索，说明这不是一个普通的CSS规则。

问：还有其他需要了解的内置CSS规则吗？

答：有。你还会看到两个常用的规则，分别是@import和@media。@import允许导入其他CSS文件（而不是HTML中通过一个<link>链入），另外@media允许创建特定于某些“媒体”类型的CSS规则，如印刷页、桌面屏幕或手机。后面还会介绍@media的更多内容。

问：Web字体看起来很棒，使用Web字体有什么缺点吗？

答：确实有一些缺点。首先，获取Web字体需要花费一些时间，所以第一次获取Web字体时，你的页面性能可能会受影响。另外，管理多个字体文件也是件痛苦的事情。最后，你可能会发现，移动设备和小型设备并不支持Web字体，所以你的设计里一定要考虑候选字体。

问：我能用@font-face使用多个定制字体吗？

答：可以。如果你使用@font-face来加载字体，对于你想用的每一种字体，要确保服务器上有相应的字体文件，而且要为每个字体创建一个单独的@font-face规则，所以要分别指定唯一的名字。不过，记住要确保只选择Web页面中真正需要的字体。每一个额外的字体都会额外增加页面的加载时间，所以如果页面中有多个Web字体，Web页面的加载会变慢。如果太慢，可能会让你的用户很不满！

问：你提到有些服务可以为我托管Web字体。你能再说详细点吗？

答：当然可以！FontSquirrel (http://www.fontsquirrel.com/)就是一个很好的网站，这里提供了很多开源、免费的字体，可以把这些字体上传到你的服务器。利用他们的字体库，可以很容易地提供一个给定字体的多个格式。Google Web字体服务（http://www.google.com/webfonts)也很不错，可以让Google为你完成管理字体和CSS的所有具体工作。在这种情况下，你只需要链接Google服务上你想要的字体，然后在你的CSS中使用相应的字体名就可以了，非常容易！

有关Web字体的更多内容请参考附录。

调整字体大小

现在Tony有了一组新的字体，接下来我们需要调整这些字体大小，因为大多数人都发现标题的默认字体有点太大了，至少从美学上讲不太美观。为此，你需要知道如何指定字体大小，具体来说，指定字体大小的方法有很多。下面先来看指定font-size的一些方法，然后讨论哪种方法最好，可以保证字体大小，还能做到用户友好。

如果你做得足够好，用户查看你的Web页面时，都能调整字体大小以方便阅读。后面几页会告诉你怎么做。

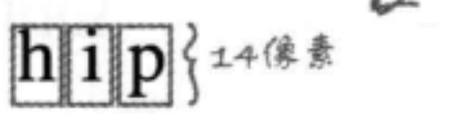
px

可以按像素指定字体大小，就像第5章中图像使用的像素尺寸一样。用像素指定字体大小时，就是在告诉浏览器字母高度是多少像素。

```
font-size: 14px;
```

在CSS中，指定像素数时，先要指定一个数，后面是“px”。这说明font-size应当是14像素高。

px必须紧跟在像素数后面。之间不能有空格。



设置一个字体高度为14像素，这说明字母的最低部分与最高部分之间有14像素。

body规则中要这样指定font-size。

```
body {
      font-size: 14px;
}
```

%

用像素指定字体大小时，会明确指出字体具体有多大，与之不同，用一个百分数指定字体大小时，会相对于另一个字体大小指出这个字体有多大。因此，

```
font-size: 150%;
```

说明这个字体大小应当是另一个字体大小的150%。不过，是另外哪一个字体大小呢？嗯，由于font-size是从父元素继承的一个属性，指定一个百分数字体大小时，就是相对于父元素的字体大小。下面来看这是如何工作的……

这里按像素指定了body的字体大小，并把一级标题的大小指定为150%。

```
body {
      font-size: 14px;
}
h1 {
      font-size: 150%;
}
```

em

还可以使用em指定字体大小，类似于百分数，这也是一个相对度量单位。使用em时，不是指定一个百分数。而是要指定一个比例因子。可以这样使用em：

不要把它与元素混淆了！

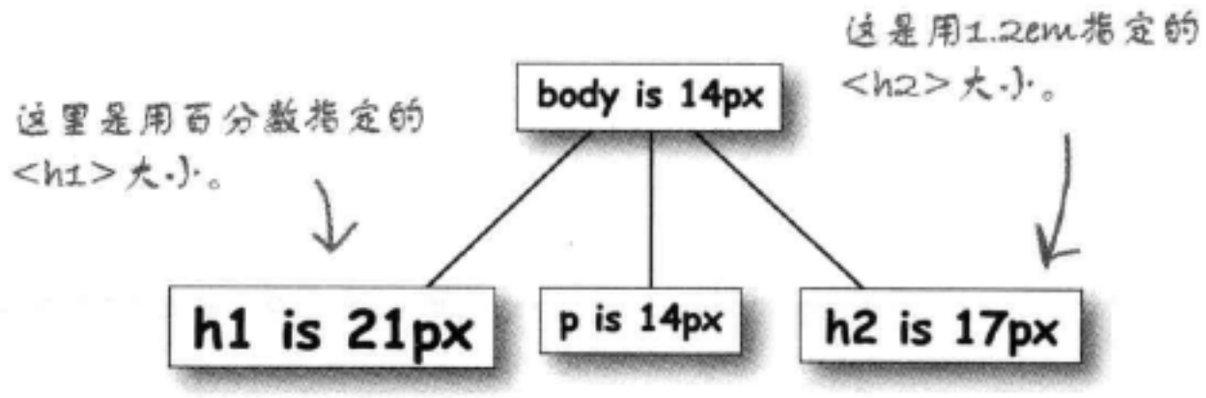
```
font-size: 1.2em;
```

这表示字体大小的比例是1.2。

假设你使用这种度量来指定<h2>标题的大小。<h2>标题的大小将是父元素字体大小的1.2倍，在这里就是1.2乘以14px，大约17px。

实际上是16.8，不过大多数浏览器会把它四舍五入为17。

```
body {
      font-size: 14px;
}
h1 {
      font-size: 150%;
}
h2 {
      font-size: 1.2em;
}
```



关键字

还有一种指定字体大小的方法：关键字。可以把一个字体大小指定为xx-small, x-small, small, medium, large, x-large或xx-large，浏览器会把这些关键字转换为像素值，它会使用浏览器中定义的默认值来完成这个转换。

各个关键字对应的大小通常有以下关系。每个大小大约比前一个大小大20%，small通常定义为大约12像素。不过，要记住，各个浏览器中这些关键字的定义并不一定相同，如果用户愿意，他们还可能重新给出他们自己的定义。

```
body {
      font-size: small;
}
```

大多数浏览器中，这会使体文本大小约为12像素。

xx-small

x-small

small

medium

large

x-large

xx-large

那么，我到底该如何指定字体大小呢？

指定字体大小确实有很多选择：px, em, 百分数和关键字。你要用哪一个呢？下面给出指定字体大小的一个小秘诀，利用这个秘诀，可以在大多数浏览器中得到一致的结果。

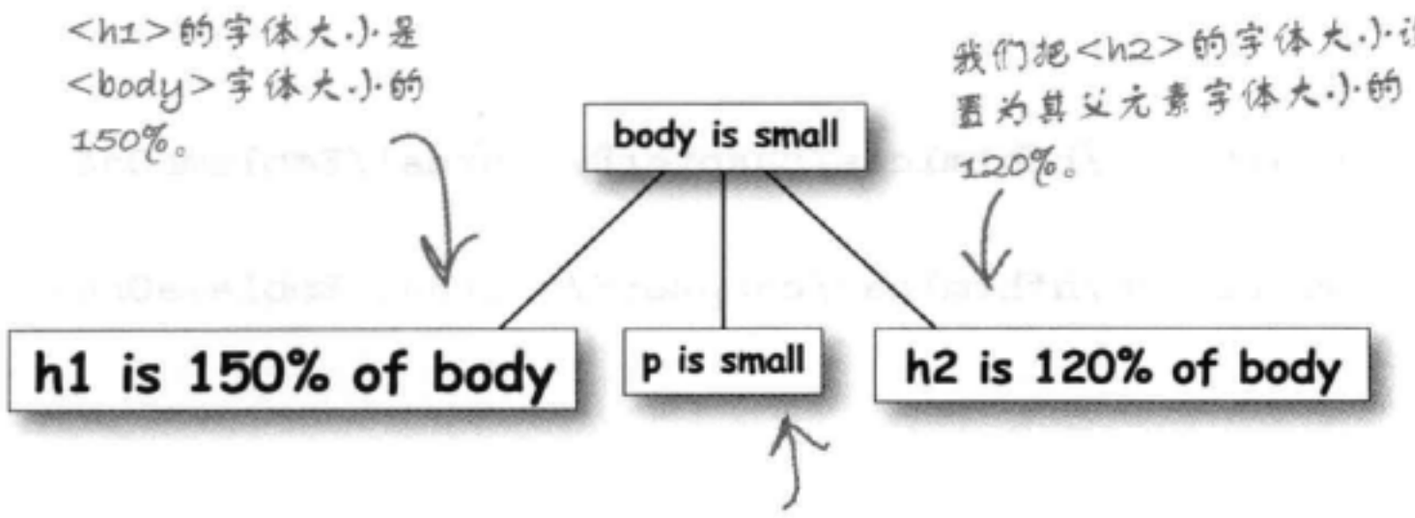
1. **选择一个关键字（我们推荐small或medium），指定它作为body规则中的字体大小。这相当于页面的默认字体大小。**
2. **使用em或百分数，相对于body字体大小指定其他元素的字体大小（选择em还是百分数由你决定，因为实际上这两种方法作用是一样的）。**

不错的秘诀，不过这有什么好处呢？是这样的：如果相对于body字体大小定义其他元素的字体大小，那么改变Web页面中的字体大小就会很容易，只需要改变body字体大小就可以了。如果你想重新设计页面，让字体更大一些呢？如果你的body字体大小值是small，只需要把它改为medium，看呐！每一个元素都会自动按比例增大，因为它们的字体大小是相对于body字体大小指定的。更棒的是，假设你的用户想调整页面上字体的大小。同样的，没问题，利用这个秘诀，页面上的所有字体都会自动调整大小。

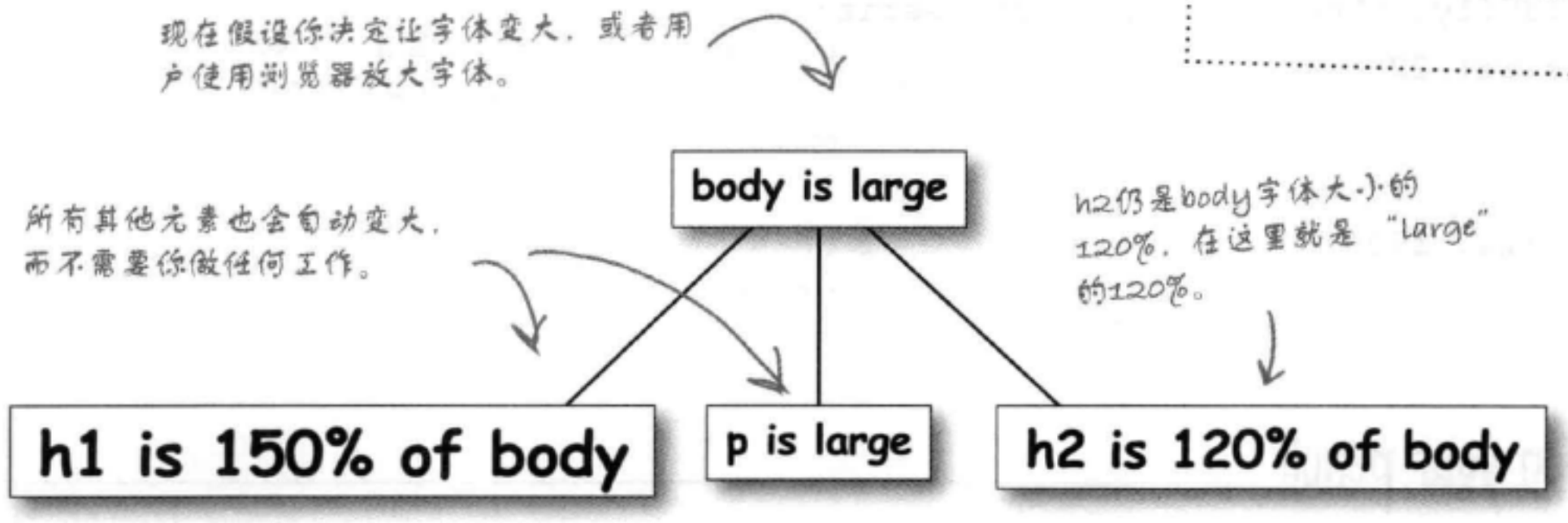
下面来看这是如何工作的。首先，为你的<body>元素设置一个大小。然后，相对于这个大小设置所有其他字体大小，就像这样：

```
body { font-size: small; }
h1 { font-size: 150%; }
h2 { font-size: 120%; }
```

这会给出类似这样的一个文档树：



现在假设你希望增大页面上字体的大小，或者可能用户想让字体变大。你会得到类似这样的一个文档树：



现在body字体大小改为large，所有其他元素的字体大小也会相对于bdoy字体大小而改变。这很棒，因为你不能逐个元素地改变所有其他元素的字体大小。你要做的只是改变body字体大小，如果你是一个用户，所有这一切都在后台发生。当你增大文本大小时，所有文本都会变大，因为所有元素都会相对地调整大小，所以页面在字体变大时看上去仍然很不错。

Watch it!

如果使用像素指定字体大小，老版本的Internet Explorer将不支持文本缩放。

很遗憾，如果用户使用老版本的Internet Explorer，倘若你的字体大小是用像素指定的，这些用户就无法调整字体的大小。这正是我们不建议按像素指定字体大小的一个原因。如果你使用像素，对于部分用户来说，页面的可用性会降低，尽管这种情况可能不会持续太久，因为用户已经陆续开始对他们的浏览器进行升级。

幸运的是，如果你遵循前面的秘诀，提供一个关键字来定义页面体的字体大小，再使用em或百分数为其他元素指定相对大小，这样当要求浏览器放大或缩小文本时，IE也能正确地缩放你的字体。

下面修改Tony的Web页面中的字体大小

现在该在Tony的Web页面中试试这些字体大小了。为“chapter8/journal”文件夹中的“journal.css”文件增加这些新属性。完成修改后，在浏览器中重新加载页面，查看字体大小的差别。如果看不出差别，仔细检查你的CSS，看看哪里有错误。

```
@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 {
    font-family: "Emblema One", sans-serif;
    font-size: 220%;
}
h2 {
    font-size: 130%;
}
```

按照前面给出的秘诀，<body>元素的font-size要使用small。这相当于基本字体大小。

另外相对于body字体大小设置其他字体。对于<h1>，我们尝试使用基本字体大小的220%作为一级标题的字体大小。

我们要让<h2>字体大小小于<h1>，是body字体大小的130%。

Sharpen your pencil

如果使用em而不是百分数来指定<h1>和<h2>的字体大小，相应的值是多少？

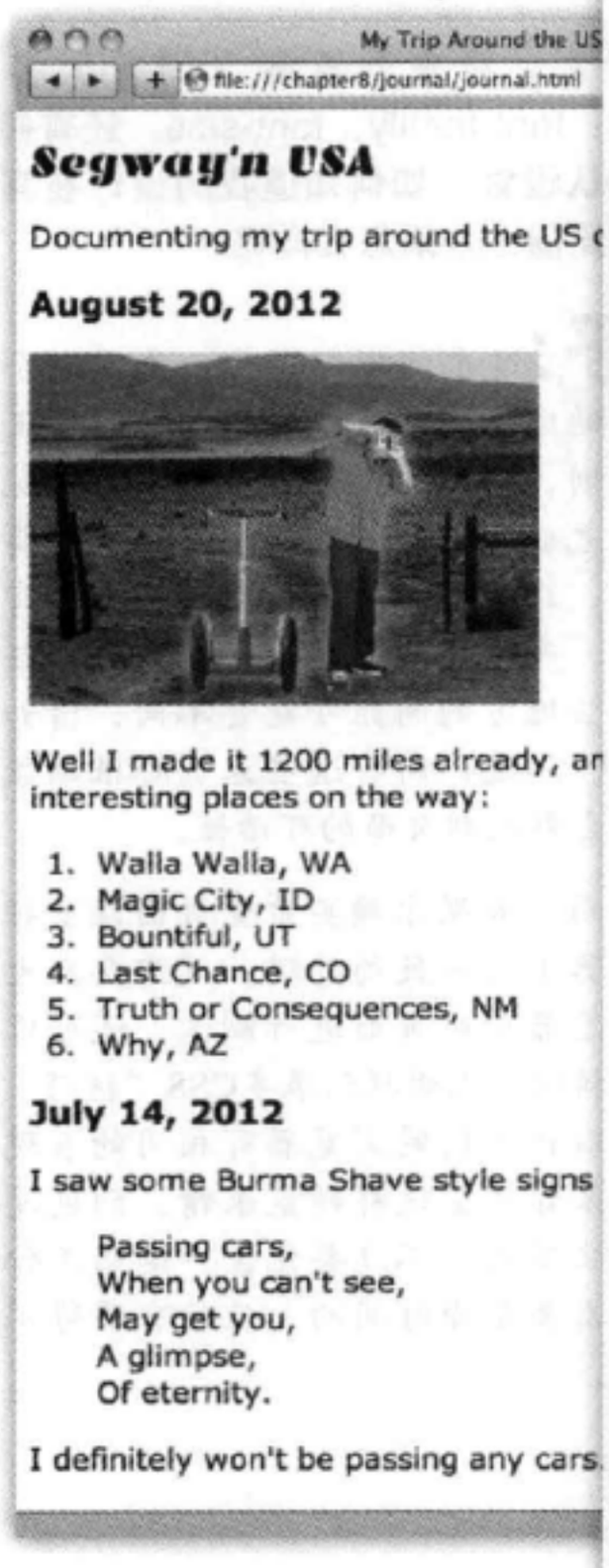
答案：<h1>将是2.2em，<h2>为1.3em。

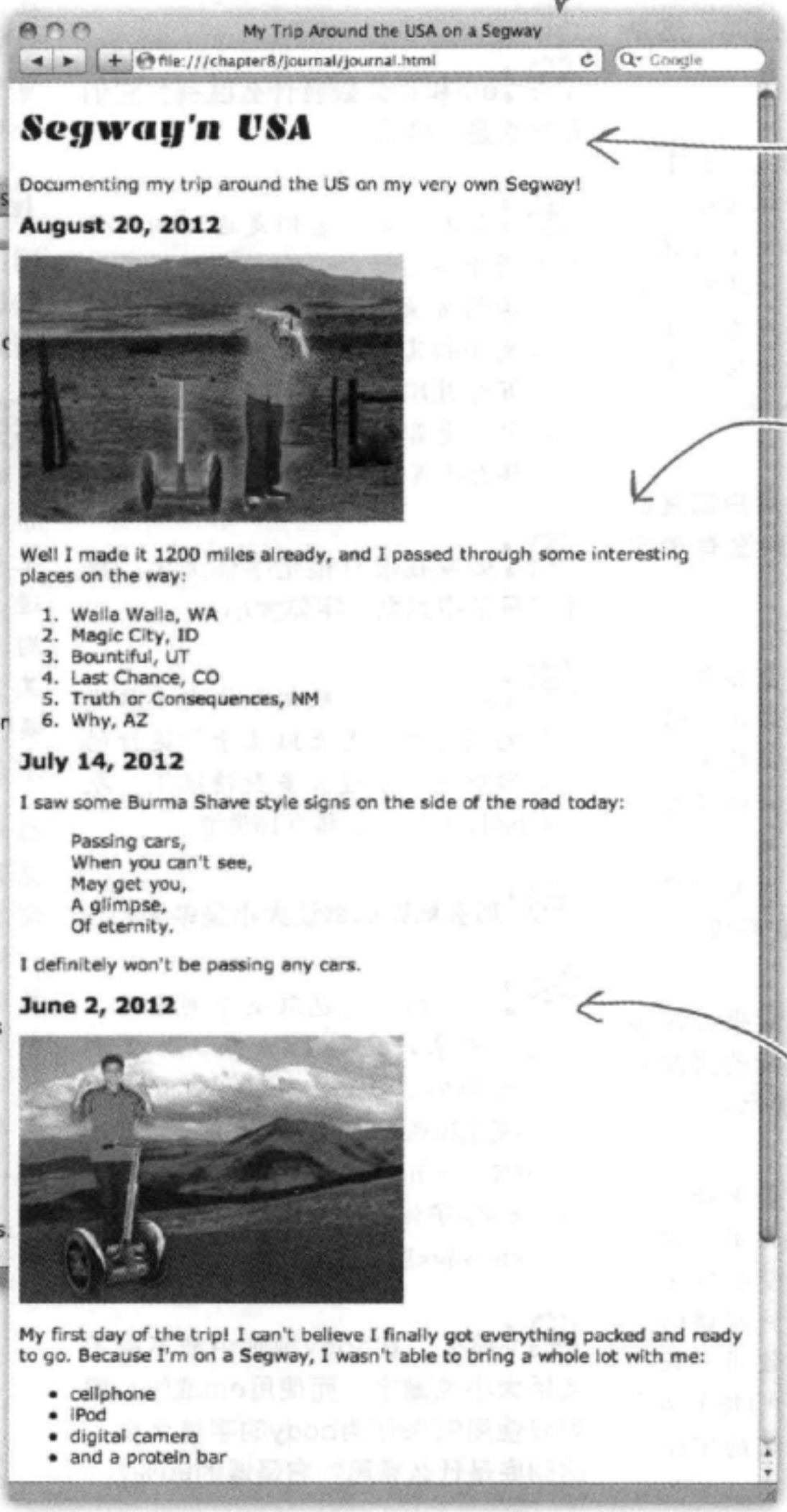
测试字体大小

这是改进后的日志页面，现在有了新的更小的字体。看看有什么不同……

这是更新字体后的新版本。这个设计看起来不那么丑陋了！

这是修改字体大小之前的上一个版本。





这个<h1>标题现在看起来好多了。它比<h2>标题大，不过在大小方面不会对体文本和页面带来压抑感。

体文本稍微小了一点。默认的体文本字体大小通常是16px，不过这取决于浏览器。使用"small"大小也很容易阅读，这可能大约为12px。

<h2>标题也变小了一点，与<h1>标题相比，大小很合适。

there are no Dumb Questions

问：这么说，通过在<body>元素中定义一个字体大小，实际上我是在为页面定义一个默认大小，是吗？这是怎么回事？

答：对，你说的没错。通过在<body>元素中设置一个字体大小，这样在设置其他元素的字体大小时就可以相对于其父元素来设置。这有什么好处呢？嗯，如果你需要改变字体大小，那么只需要改变body字体大小，所有其他元素都会按比例改变。

问：我们真的需要考虑用户调整浏览器字体大小吗？我从来没有考虑过。

答：是的。几乎所以浏览器都允许用户将页面的文本变大或变小，很多用户会利用这个特性。如果你采用一种相对方式定义字体，用户调整页面字体大小时就没有任何问题。不过要当心不要使用像素大小，因为有些浏览器在调整像素大小时会有麻烦。

问：我还是喜欢使用像素来设置字体大小，因为这样一来，我的页面会严格按我指定的字体大小显示。

答：这确实有点道理，如果每个元素的字体大小都使用像素指定，你会为每个元素准确选择你希望的字体大小。不过这样做要以为用户提供的灵活性为代价，有些用户（使用老版本Internet Explorer的那些用户)将无法选择一个适合显示、适合阅读的字体大小。

这样创建的页面维护也更困难一些，因为如果你突然想让一个页面中所有元素的字体大小变大，就必须做大量修改。

问：em和百分数有什么区别？它们看起来是一样的。

答：基本说来，它们是达到相同目的的两条不同途径。它们都提供了一种灵活的方法，可以相对于父元素的字体大小指定一个字体大小。很多人发现百分数比em更容易理解，另外在CSS中也更容易阅读。不过你完全可以选择自己喜欢的方法。

问：如果我没有指定字体大小，是不是只能得到默认字体大小？

答：是的，这些默认字体大小取决于你的浏览器，甚至取决于你运行的浏览器版本。不过大多数情况下，默认的body字体大小都为16像素。

问：那么标题的默认大小是多大？

答：同样的，这也取决于浏览器，不过一般来讲，<h1>是默认体文本字体大小的200%，<h2>是150%，<h3>是120%，<h4>是100%，<h5>是90%，<h6>是60%。注意默认地，<h4>字体大小与body字体大小相同，<h5>和<h6>要小一些。

问：那么，在body规则中能不能不使用大小关键字，而使用em或%？如果我使用90%作为body的字体大小，这到底是什么意思？它是谁的90%？

答：对，你可以这么做。如果在body规则中指定字体大小为90%，这将是默认字体大小的90%，我们刚说过，默认字体大小通常是16像素，所以它的90%就大约是14像素。如果你希望一个字体大小与关键字指定的大小稍有区别，就可以使用%或em。

问：看来浏览器之间有太多不同：font-family、font-size，还有各种默认设置。如何知道我的设计在其他浏览器上效果怎么样呢？

答：这个问题问得好。先做一个简单的回答：如果你遵循这一章给出的原则，你的大部分设计在其他浏览器上也会有很好的效果。不过，你要知道，在不同浏览器上看起来会稍有差别，字体可能稍稍大一些或小一些，某些地方的间距可能会不同，诸如此类。不过，所有这些差别都很细微，不会影响到页面的可读性。

然而，如果你确实希望页面在多种浏览器上有一致的外观，就需要在大量浏览器中对页面进行测试，还可以更为精细，你能找到很多CSS“技巧”，可以让不同的浏览器有相同的表现。如果你希望这样精益求精，倒也没有什么不对，不过要记住，这些工作都是需要花费时间的，往往有些得不偿失。

改变字体粗细

font-weight属性允许你控制文本的粗细。你应该知道，粗体文本看起来比正常文本更深，而且往往要“胖”一点。可以将元素的font-weight属性设置为bold，来使用粗体文本，如下所示：

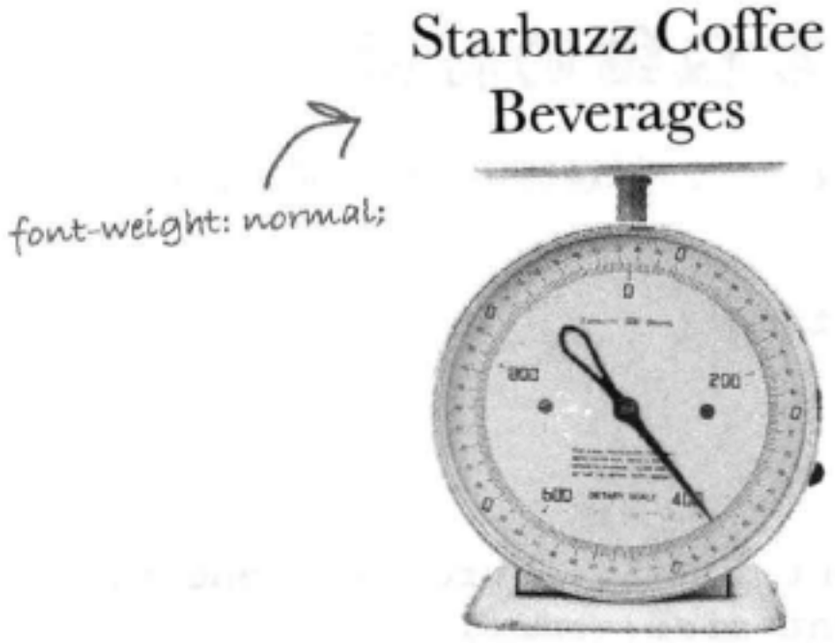
```
font-weight: bold;
```

也可以反过来。如果一个元素默认地设置为bold，或者从父元素继承了粗体，可以如下去掉粗体样式：

```
font-weight: normal;
```

还有两个相对font-weight属性：bolder和lighter。使用这两个属性值时，会相对于所继承的值使文本样式稍粗一些或者稍细一些。这些值很少使用，因为没有多少字体支持粗细程度的微小差别，实际上这两个值通常没有任何效果。

还可以把font-weight属性设置为100到900之间的一个数（100的倍数），不过同样的，这个特性也未得到字体和浏览器的广泛支持，所以通常并不使用。



Sharpen your pencil

编写CSS，将Tony页面中的二级标题从默认的bold值（粗体）改为normal（正常粗细）。接下来，将这个规则增加到你的CSS，做个测试。下一页会给出答案。

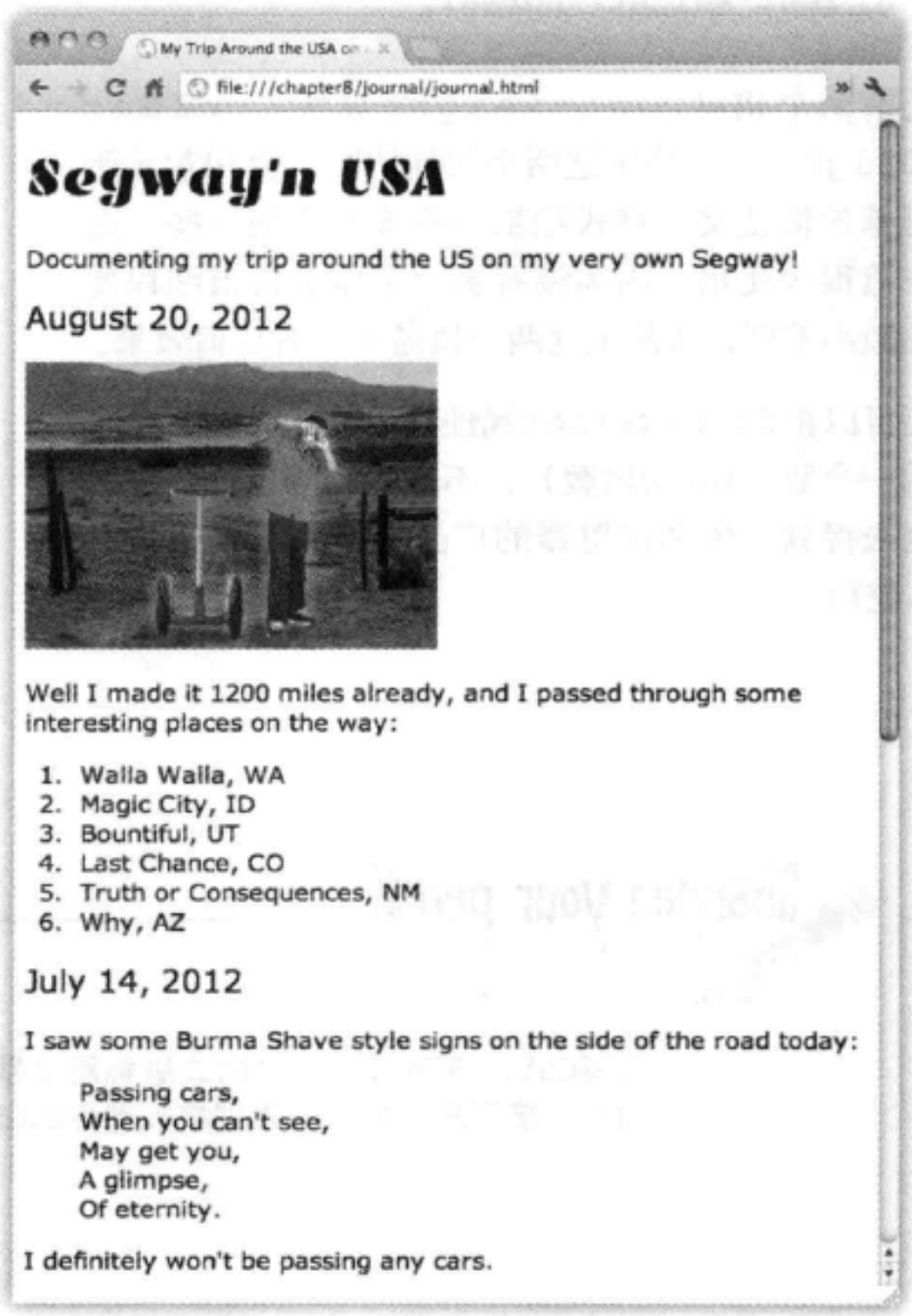
测试正常粗细的标题

完成以上修改，<h2>标题使用正常粗细之后（font-weight值为normal），CSS如下所示：

```
@font-face {
    ...
}
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
    font-size: small;
}
h1 {
    font-family: "Emblema One", sans-serif;
    font-size: 220%;
}
h2 {
    font-size: 130%;
    font-weight: normal;
}
```

为节省篇幅，这里省略了整个@font-face定义。

这里把<h2>标题的font-weight改为normal。



这是修改后的结果。<h2>标题现在看起来浅一些了。仍然能区分出它们是标题，因为它们的大小是体文本的130%。

为字体增加风格

你应该对斜体文本很熟悉了，对不对？斜体文本是倾斜的，有时还有额外的弯曲衬线。例如，比较下面这两种风格：

not italic
italic

斜体文本向右倾斜，另外衬线还有弯曲。

在CSS中可以使用`font-style`属性为文本增加斜体风格：

```
font-style: italic;
```

这里有一个常犯的错误，可能会把"italic"写成"italics"。如果是这样，你就看不到斜体文本。所以一定要记住检查拼写。

不过，并不是所有字体都支持斜体风格（italic），所以你得到的实际上称为倾斜（oblique）文本。倾斜文本也是倾斜的，不过这种字体并不是使用一组专门设计的倾斜字符，而是由浏览器将正常文字倾斜。请比较以下非倾斜和倾斜风格：

not oblique
oblique

倾斜风格中，普通文字向右倾斜。

也可以使用`font-style`属性得到倾斜文本，如下所示：

```
font-style: oblique;
```

实际上你会发现，取决于你选择的字体和浏览器，有时这两种风格看起来是一样的，有时则不同。所以，除非确实非得区分斜体和倾斜文本，这对你非常重要，否则完全可以任选一种使用。如果确实很重要，你就需要对不同的字体和浏览器组合进行测试，来得到最佳效果。

斜体（Italic）和倾斜（oblique）风格会使字体看起来是倾斜的。

除非可以控制用户使用的字体和浏览器，否则你会发现，不论你指定什么风格，结果并不确定，有时你会得到斜体，有时则是倾斜文本。

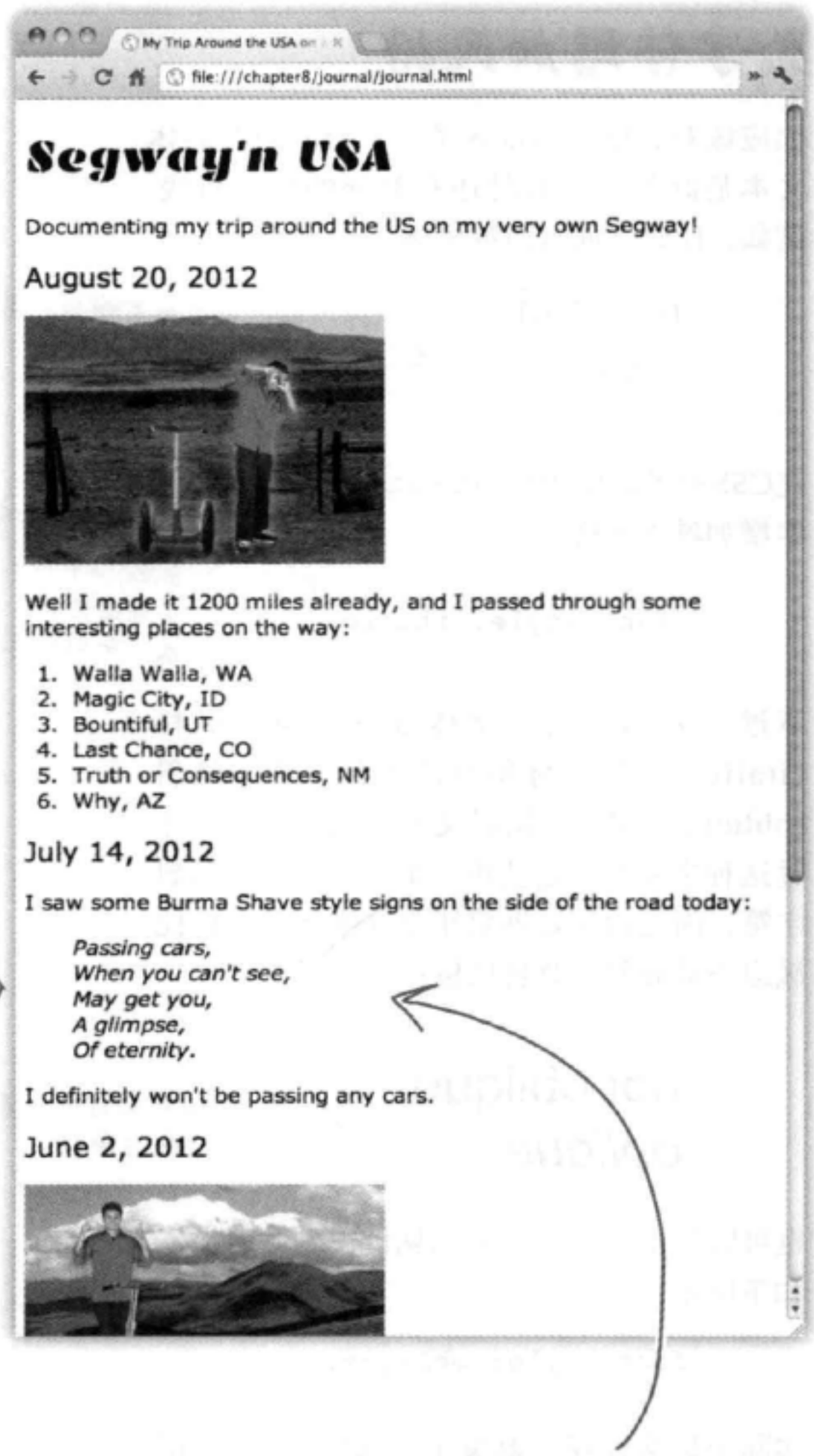
所以完全可以就用斜体，不用担心它们的差别（你可能根本无从控制）。

让Tony的引用有一点斜体风格

现在我们要使用font-style属性，让Tony的引用有些自己的风格。还记得<blockquote>元素中的柏玛刮胡膏广告词吗？我们要把这个广告词变成斜体风格，让它在文本中更为突出。为此，只需要设置<blockquote>的样式，指定font-style为italic，如下所示：

```
blockquote {
      font-style: italic;
}
```

把这个新的CSS规则增加到你的“journal.css”文件中，保存，并对页面进行测试。你会看到柏玛刮胡膏广告词变成了斜体；这是我们得到的测试结果。



这里展示了Tony页面中柏玛刮胡膏广告词的新风格。可以看到我们想要的倾斜文本。

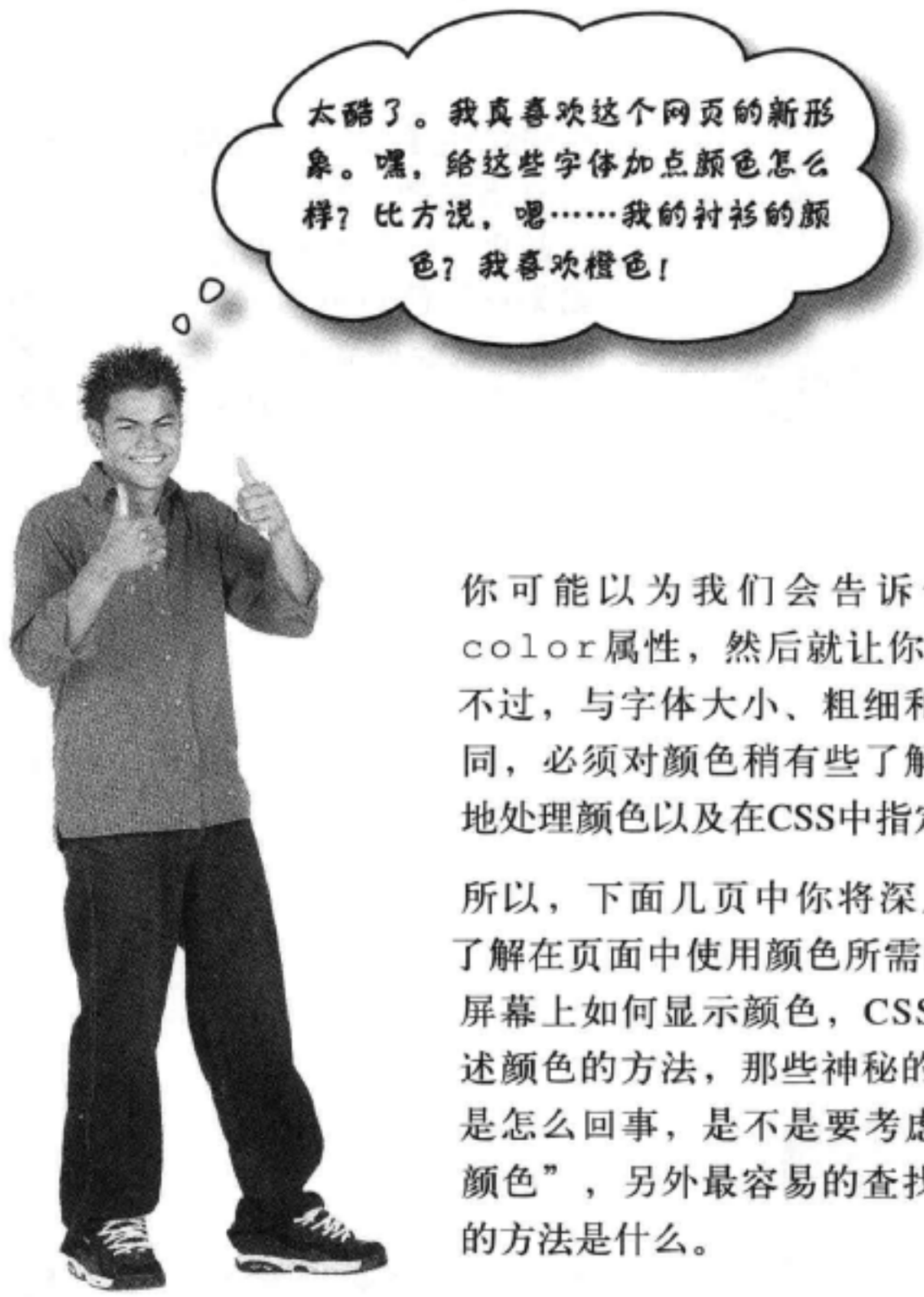
there are no Dumb Questions

问：<blockquote>的文本实际上在一个<p>中，这个<p>内嵌在这个<blockquote>里。它是怎么把段落变成斜体的呢？

答：要记住，默认地，大多数元素都会从其父元素继承得到字体样式，这个段落的父元素是<blockquote>元素。所以<blockquote>中的段落会继承这个斜体风格。

问：为什么不直接把文本放在<blockquote>中的一个元素中呢？这样是不是也一样，可以把<blockquote>变成斜体？

答：要记住，要用来指定结构。表示一些文字需要强调。现在我们所做的只是为<blockquote>指定样式，而不是指示应当强调<blockquote>中的文本。所以，也许你是对的，大多数浏览器上确实是斜体，但是要为<blockquote>中的文本指定样式，这可不是一种正确的方法。一定要记住，的样式可能会改变，所以不能指望总是斜体。

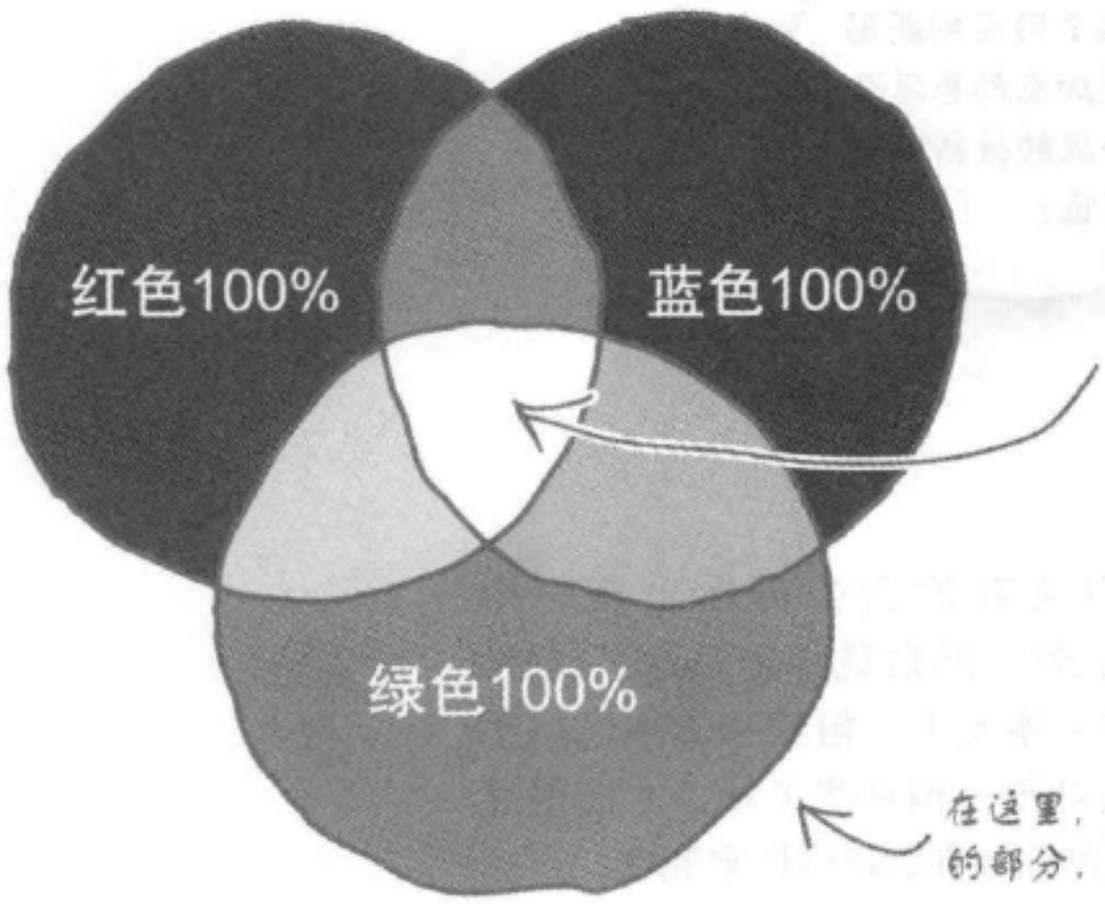


你可能以为我们会告诉你：有一个color属性，然后就让你用这个属性。不过，与字体大小、粗细和文本风格不同，必须对颜色稍有些了解，才能很好地处理颜色以及在CSS中指定颜色。

所以，下面几页中你将深入学习颜色，了解在页面中使用颜色所需的全部知识：屏幕上如何显示颜色，CSS中有哪些描述颜色的方法，那些神秘的十六进制码是怎么回事，是不是要考虑“Web安全颜色”，另外最容易的查找和指定颜色的方法是什么。

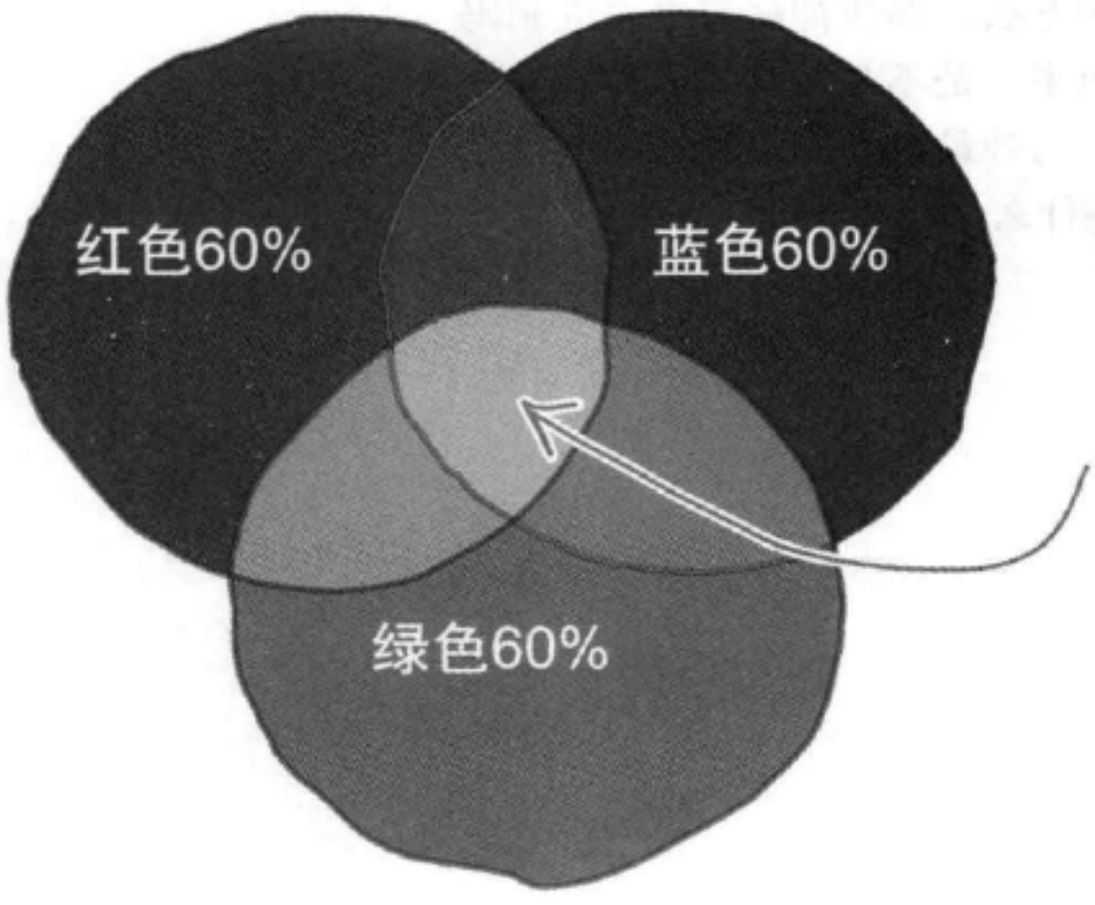
Web颜色如何工作？

你已经看到了，页面上很多地方都可以加颜色：背景颜色、边框颜色，还有很快要谈到的字体颜色。不过，这些颜色在计算机上究竟是如何工作的？下面来看一下。



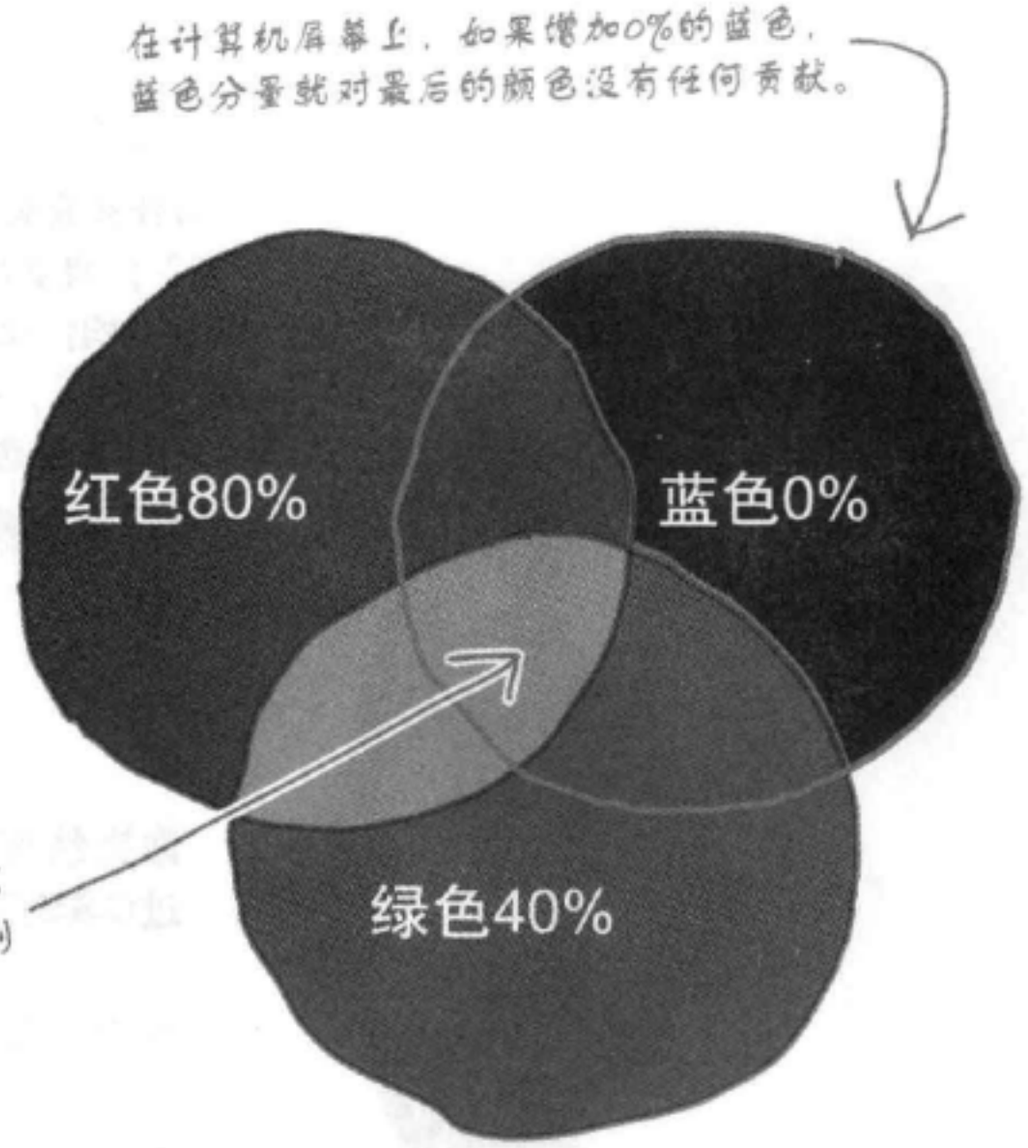
在这里，红色、绿色和蓝色混合在一起。请看中间的部分，可以看到它们是如何叠加的。

Web颜色是按构成颜色的红、绿、蓝三个分量所占数量来指定的。每种颜色会分别指定一个从0到100%的数值，然后把它们混合起来得到最终的颜色。例如，如果把红色100%、绿色100%和蓝色100%混合在一起，就会得到白色。注意，在计算机屏幕上，将颜色混合在一起会得到一种更亮的颜色。毕竟，光就是混合而成的！

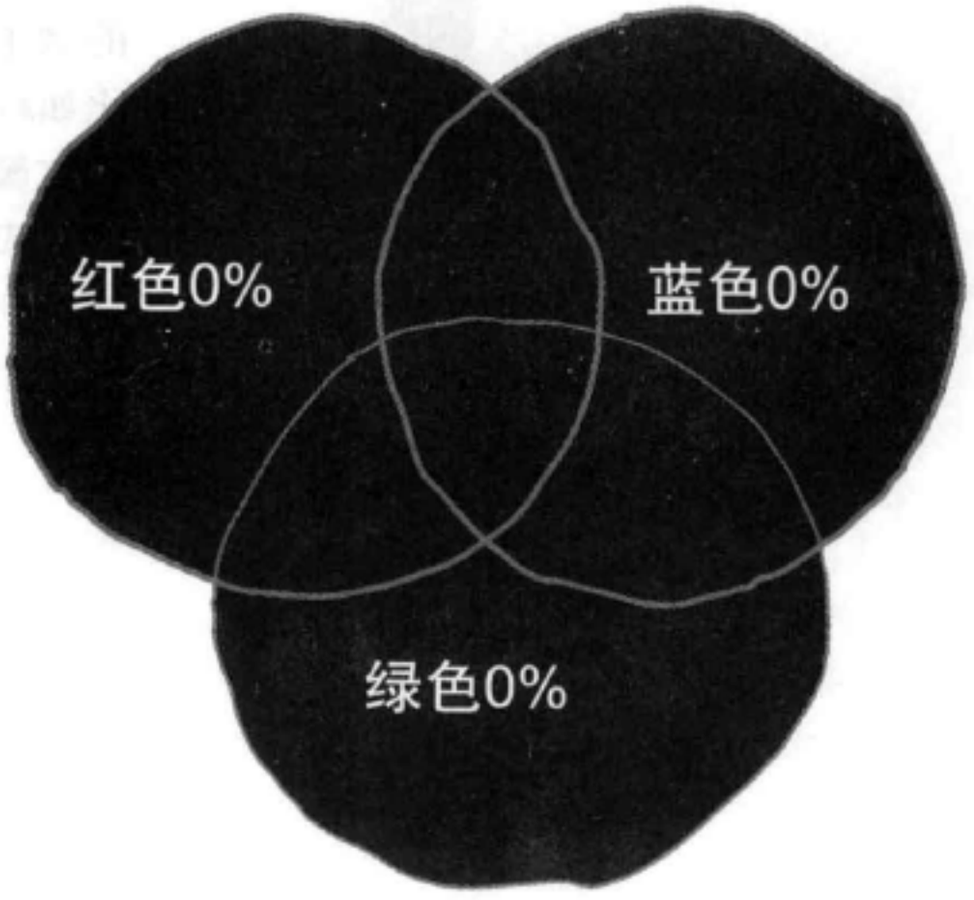


不过，如果每个颜色分量（红、绿和蓝）不是100%，比如只是60%，会得到什么颜色呢？不太白，是吗？换句话说，你会得到一个灰色，因为我们仍然是将这三种颜色等量相加，不过没有那么多光发送到屏幕上。

或者，假设把红色80%和绿色40%混合在一起。你觉得会得到一个橙色，是吗？嗯，你说的没错，确实是一个橙色。注意，如果一个颜色分量为0，它不会影响另外两个颜色分量。同样的，这是因为没有蓝光与红光和绿光混合。



那么如果红、绿和蓝色都是0%，混合在一起会得到什么？这说明，根本没有向屏幕发送任何光，所以会得到黑色。



你当然可以使用你喜欢的颜色名，不过CSS只定义了大约150个颜色名。

尽管看起来很多，但调色板很快就会落后，因为颜色不够用，以至于限制页面的表现力。我们会告诉你另外一种指定颜色的方法，允许你指定远远不止150种颜色。事实上，你可以使用一个包含1600万种颜色的调色板。

之前你已经在HTML中见过一些颜色的例子，没错，它们看起来有点奇怪，比如#fc1257。所以下面先来明确如何指定颜色，接下来你会了解到，利用颜色表、在线颜色选择器或你的照片编辑应用，可以很容易地选择颜色。

如何指定Web颜色？来数数看有多少种方法……

CSS提供了很多种指定颜色的方法。你可以指定颜色名，或者按红、绿、蓝相对百分比指定颜色，也可以使用一个十六进制码指定颜色，这是描述颜色红、绿、蓝分量的一种简写形式。

你可能以为现在Web已经确定了一种统一的格式，但实际上以上的这些格式都很常用，所以最好全面了解。不过，到目前为止，十六进制码是指定Web颜色最为常用的方法。但要记住，所有这些指定颜色的方法最终都是要告诉浏览器：一个颜色中红、绿、蓝分量分别是多少。

下面分别介绍在CSS中指定颜色的各种方法。

按名指定颜色

要在CSS中描述颜色，最直接的方法就是使用颜色名。有16种基本颜色和150种扩展颜色可以采用这种方法指定。假设你希望指定“silver”作为一个body元素的背景色；可以在CSS中这样写：

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

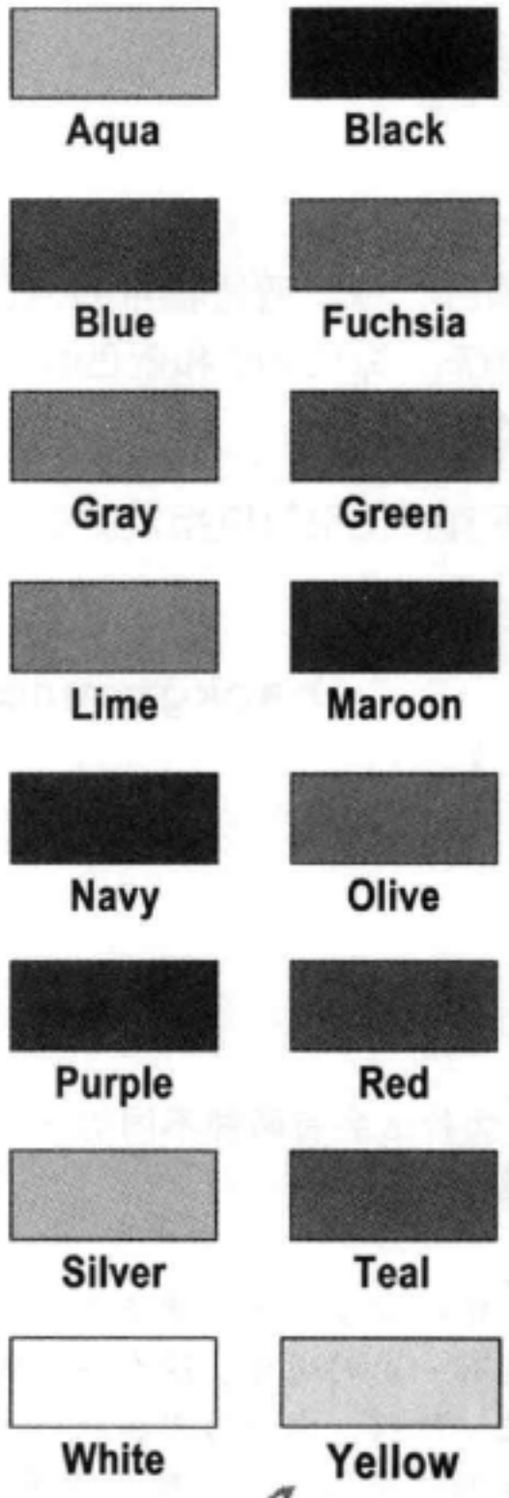
这里是body规则。

background-color属性。

按颜色名指定颜色。

所以，按名指定一个颜色时，只需要输入颜色名作为相应的属性值。CSS颜色名是不区分大小写的，所以输入silver，Silver或SILVER都是允许的。这里给出了可以在CSS中指定的16种基本颜色。要记住，这些颜色名只是预定义了红、绿、蓝三种颜色分量的多少。

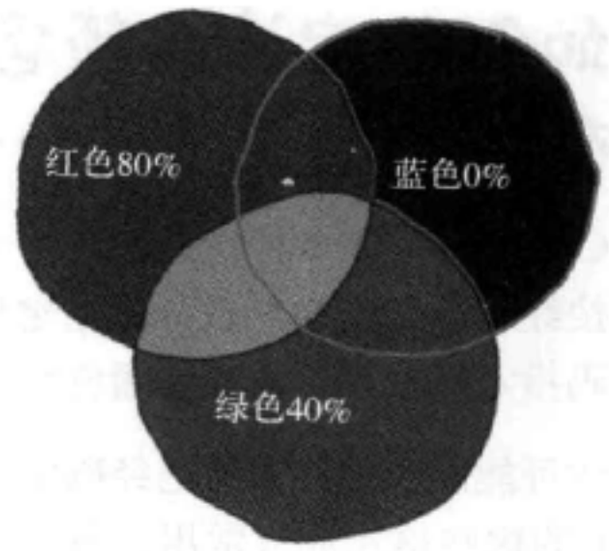
所有浏览器中都肯定有这16种颜色，不过可能只在较新的浏览器中能找到150种扩展颜色。



书上的颜色是印刷页反射的光。而在计算机上，光是从屏幕发出的，所以这些颜色看起来可能与Web页面上显示的颜色稍有不同。

用红、绿、蓝值指定颜色

还可以按红、绿、蓝分量的多少来指定一个颜色。所以，假设你想指定前几页上见过的橙色，它由红色80%，绿色40%和蓝色0%构成。可以这样做：




```
body {
    background-color: rgb(80%, 40%, 0%);
}
```

以"rgb"开头，这是red, green, blue的缩写。

然后在小括号里指定红、绿、蓝的百分比，每个分量后面都有一个百分号（%）。

还可以将红、绿、蓝值指定为0到255之间的一个数值。所以不用指定为红色80%，绿色40%和蓝色0%，可以使用红色204，绿色102，蓝色0来指定。

这些数值是从哪里来的？

255的80%就是204，

255的40%就是102，

255的0%当然是0。

可以如下直接使用数值指定颜色：

```
body {
    background-color: rgb(204, 102, 0);
}
```

还是以"rgb"开头。

要指定数值而不是百分数，直接输入数值，不要加%。

there are no Dumb Questions

问：为什么会有两种不同的方法来指定rgb值？百分数不是更直接吗？

答：有时百分数确实更直接，不过使用0到255之间的一个数也有一定的道理。这个数与1字节信息中包含的数值个数有关。所以，由于历史和技术方面的原因，通常使用255作为在颜色中指定红、绿、蓝值的度量单位。实际上，你可能已经注意到了，照片编辑应用通常就允许你指定从0到255的颜色值（如果还没有，稍后你就会知道该怎么做）。

问：我从来没有见过别人在CSS中使用rgb或者具体的颜色名。看起来所有的人都在使用类似#00fc9a的颜色码。

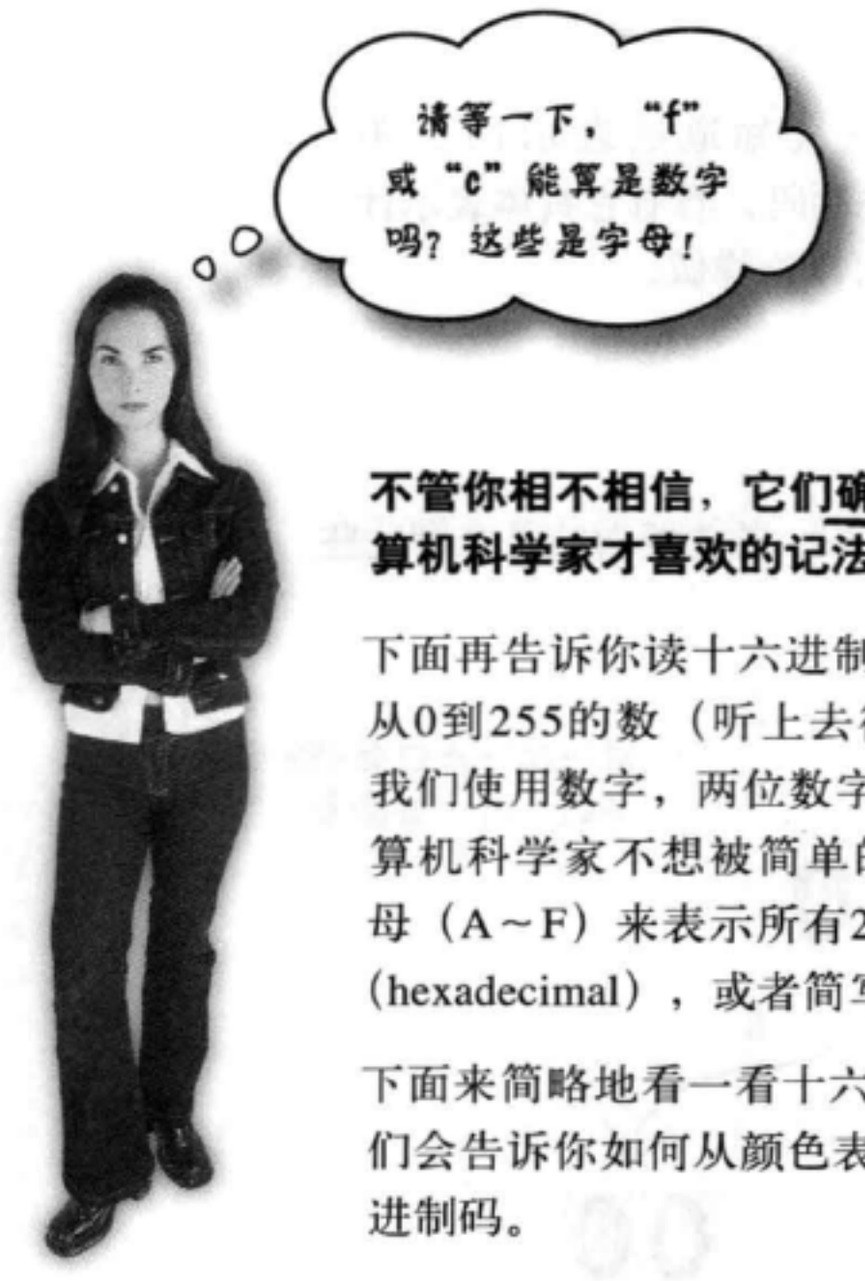
答：使用rgb百分数或数值越来越常见了，不过你说的没错，“十六进制码”仍然使用最广泛，因为人们认为这是一种很方便的指定颜色的方法。

问：是不是我一看到类似rgb(100, 50, 200)的颜色表示，就应该能知道这种颜色是什么，这很重要吗？

答：没必要。要知道rgb(100,50,200)具体表示什么颜色，最好的办法就是在你的浏览器中加载，或者使用一个在线颜色选择器或照片编辑应用查看。

使用十六进制码指定颜色

现在来考虑这些看起来很怪异的十六进制码。可以告诉你，它们的秘密是：一个十六进制码中，每组2位数字分别代表颜色的红、绿、蓝分量。所以前两位数字表示红色，接下来两位表示绿色，最后两位表示蓝色 。就像这样：



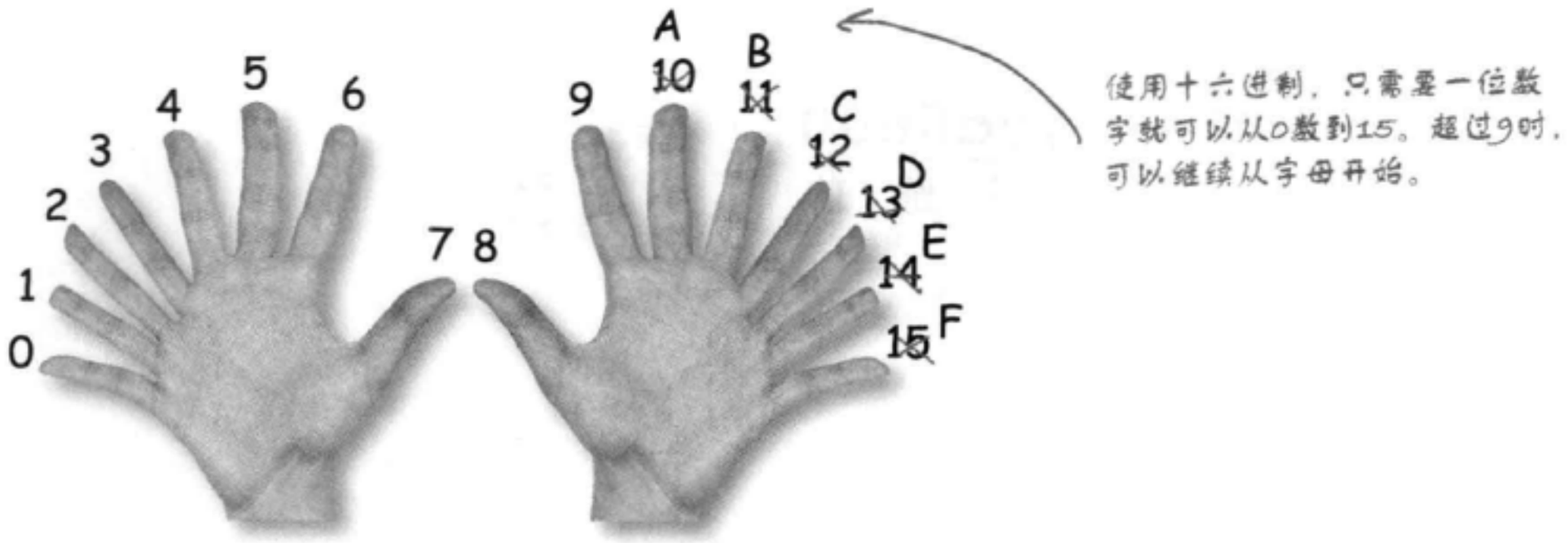
不管你相不相信，它们确实是数字，不过这里采用了只有计算机科学家才喜欢的记法。

下面再告诉你读十六进制码的秘密：每组两位数字表示一个从0到255的数（听上去很熟悉，是吗）。问题在于，如果我们使用数字，两位数字就只能表示到99，是不是？嗯，计算机科学家不想被简单的0～9束缚，他们决定借助一些字母（A～F）来表示所有256个值。这就是十六进制计数系统（hexadecimal），或者简写为“hex”。

下面来简略地看一看十六进制码究竟是如何工作的，然后我们会告诉你如何从颜色表或照片编辑应用中得到颜色的十六进制码。

十六进制码速查指南

关于十六进制码，首先要知道的是，它们并不是基于10个数字（0到9）。而是基于16个数字（0到F）。十六进制数是这样的：

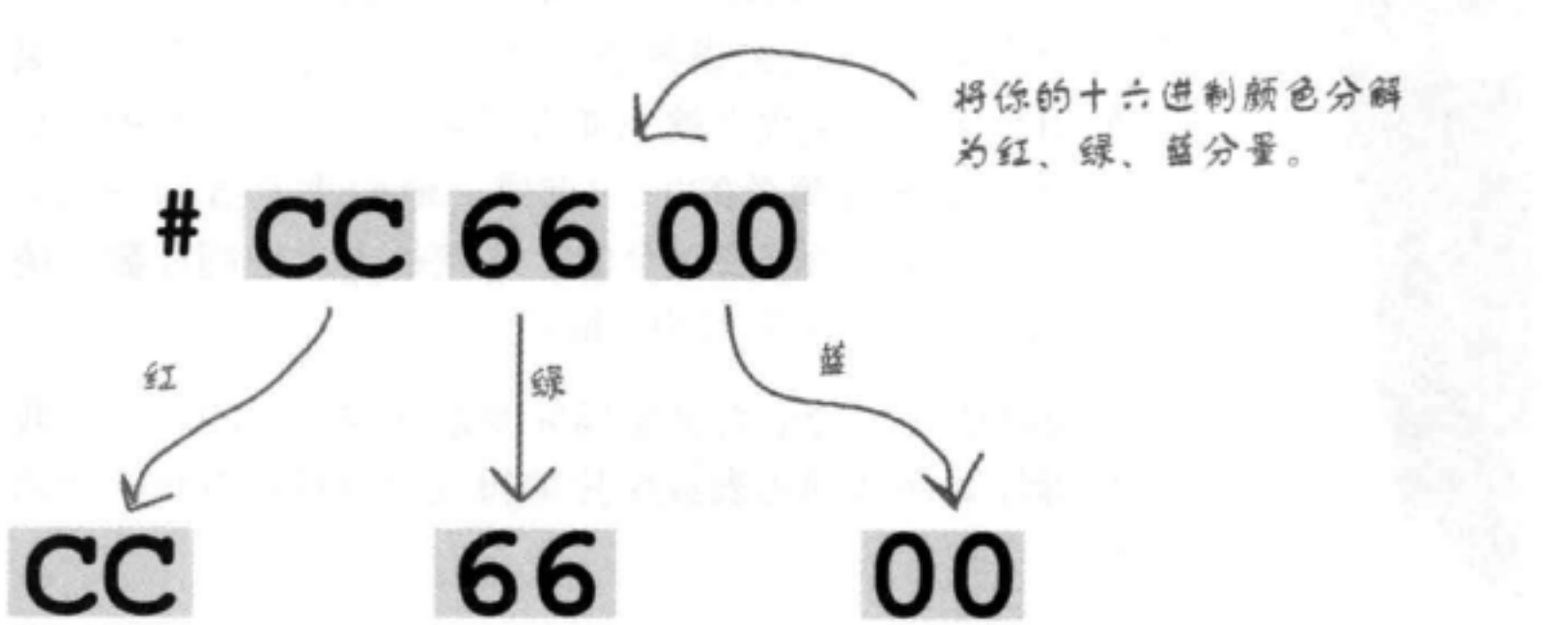


所以，如果你看到一个十六进制数，比如B，就会知道这表示11。不过，BB、E1或FF表示什么呢？下面来分解一个十六进制码，看看它具体表示什么。实际上，对于你可能遇到的十六进制颜色，你都可以这样做。

第1步：

将十六进制颜色分解为它的3个分量。

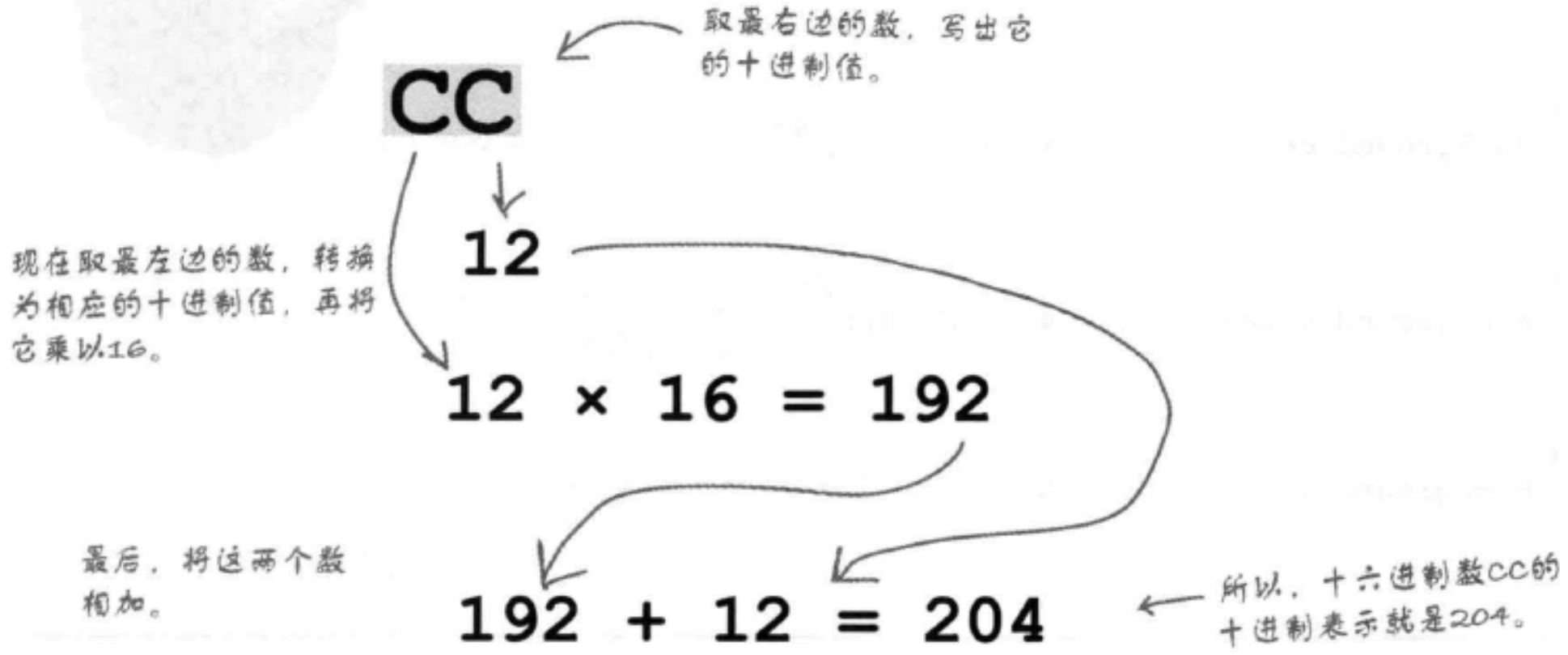
要记住，每个十六进制颜色由红、绿、蓝分量组成。首先要做的是分解这些分量。



第2步：

将每个十六进制数转换为相应的十进制数。

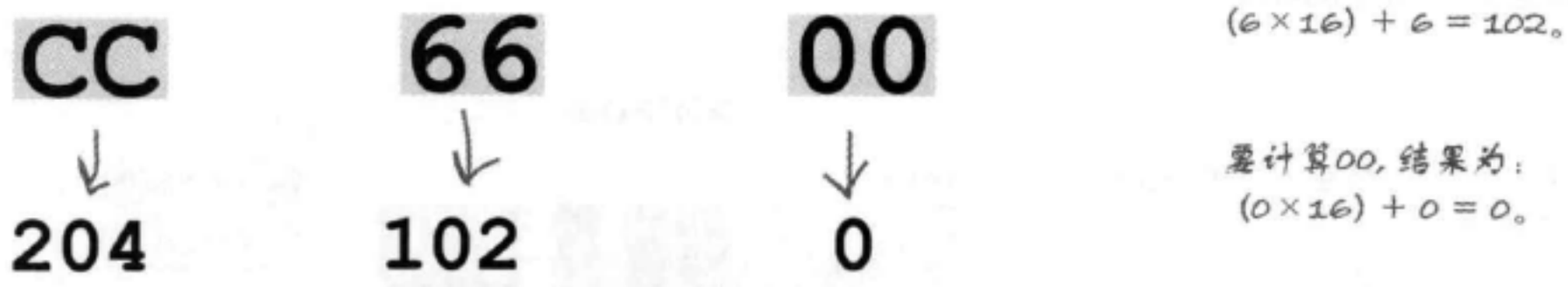
既然已经分解得到了分量，下面可以计算各个分量的值，得到0到255的一个数。先从红色分量的十六进制数开始：



第3步：

现在对另外两个值做同样的处理。

对另外两个值重复使用上面的方法。最后可以得到：



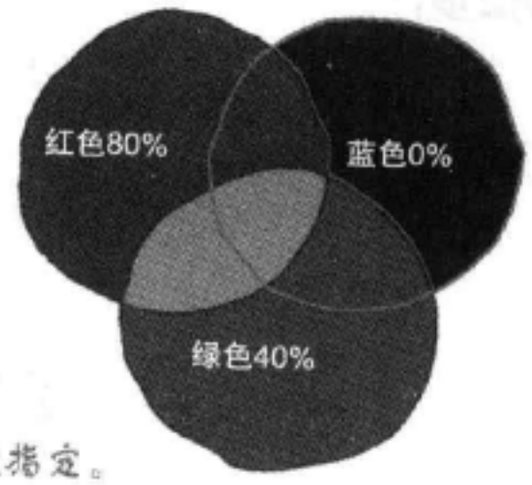
第4步：

没有第4步，已经完成了！

就这么简单。现在你得到了各个分量的数值，准确地知道这个颜色中红、绿、蓝各有多少。可以采用同样的方法对任何十六进制颜色进行分解。下面来看通常如何确定Web颜色。

综合在一起

现在你已经掌握了指定颜色的多种不同方法。以我们的橙色为例，它由红色80%，绿色40%和蓝色0%组成。在CSS中，可以采用任何一种方法指定这个颜色：




```
body {
      background-color: rgb(80%, 40%, 0%);
}
```

按红、绿、蓝百分数指定。

```
body {
      background-color: rgb(204, 102, 0);
}
```

按0～255的红、绿、蓝分量值指定。

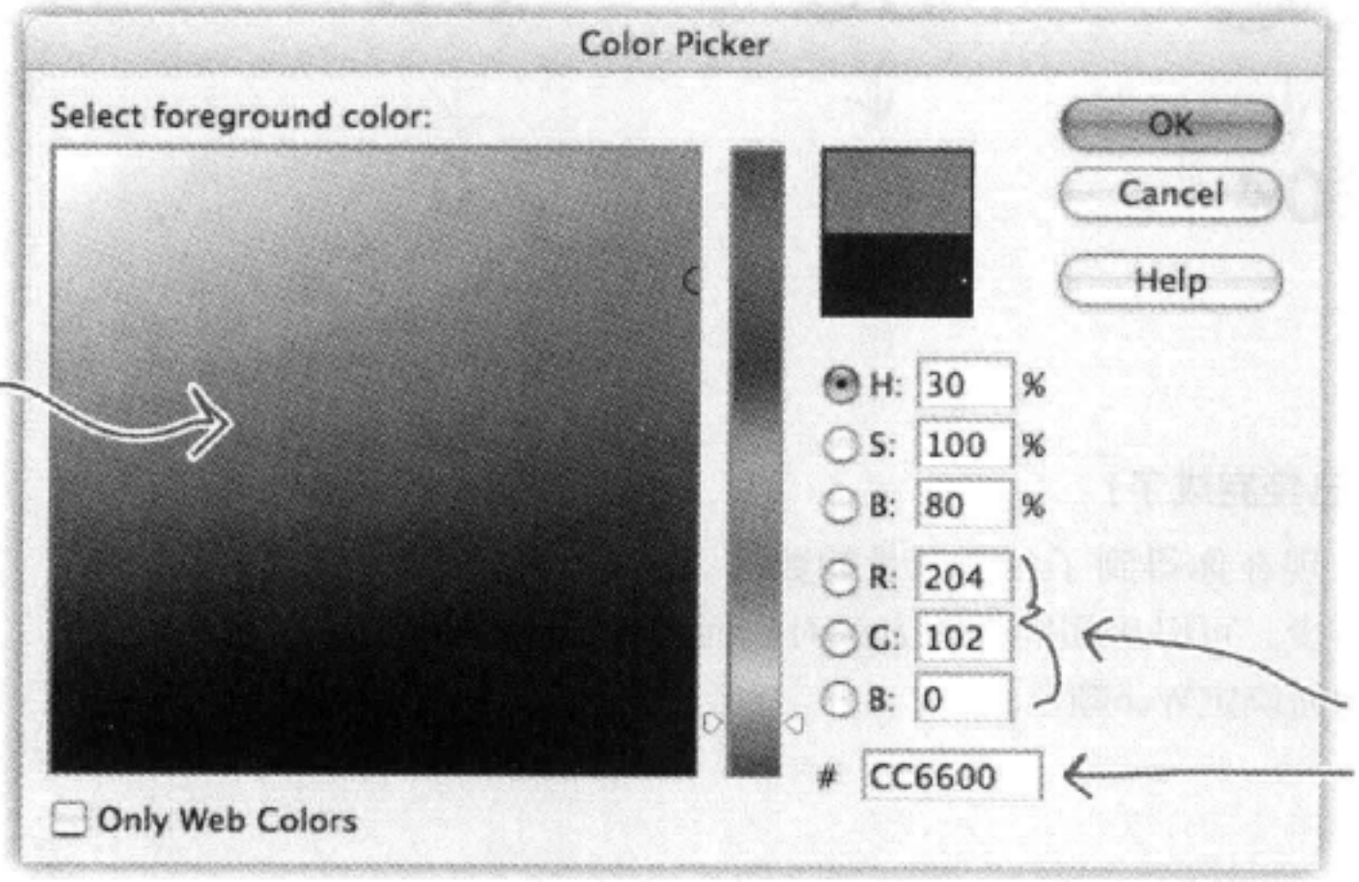
```
body {
      background-color: #cc6600;
}
```

使用一个十六进制码指定。

如何找到Web颜色

要找到Web颜色，有两种最常用的方法，可以使用一个颜色表，或者使用类似Photoshop Elements的应用。你可能还能找到很多Web页面允许你选择Web颜色，并完成rgb与十六进制码之间的转换。下面来看Photoshop Elements（大多数照片编辑应用都提供了同样的功能）。

大多数照片编辑应用都提供了一个颜色选择工具，允许你以图示方式使用一个或多个颜色光谱来选择颜色。



颜色选择工具还允许只选择“Web安全”颜色。稍后会介绍这个内容。

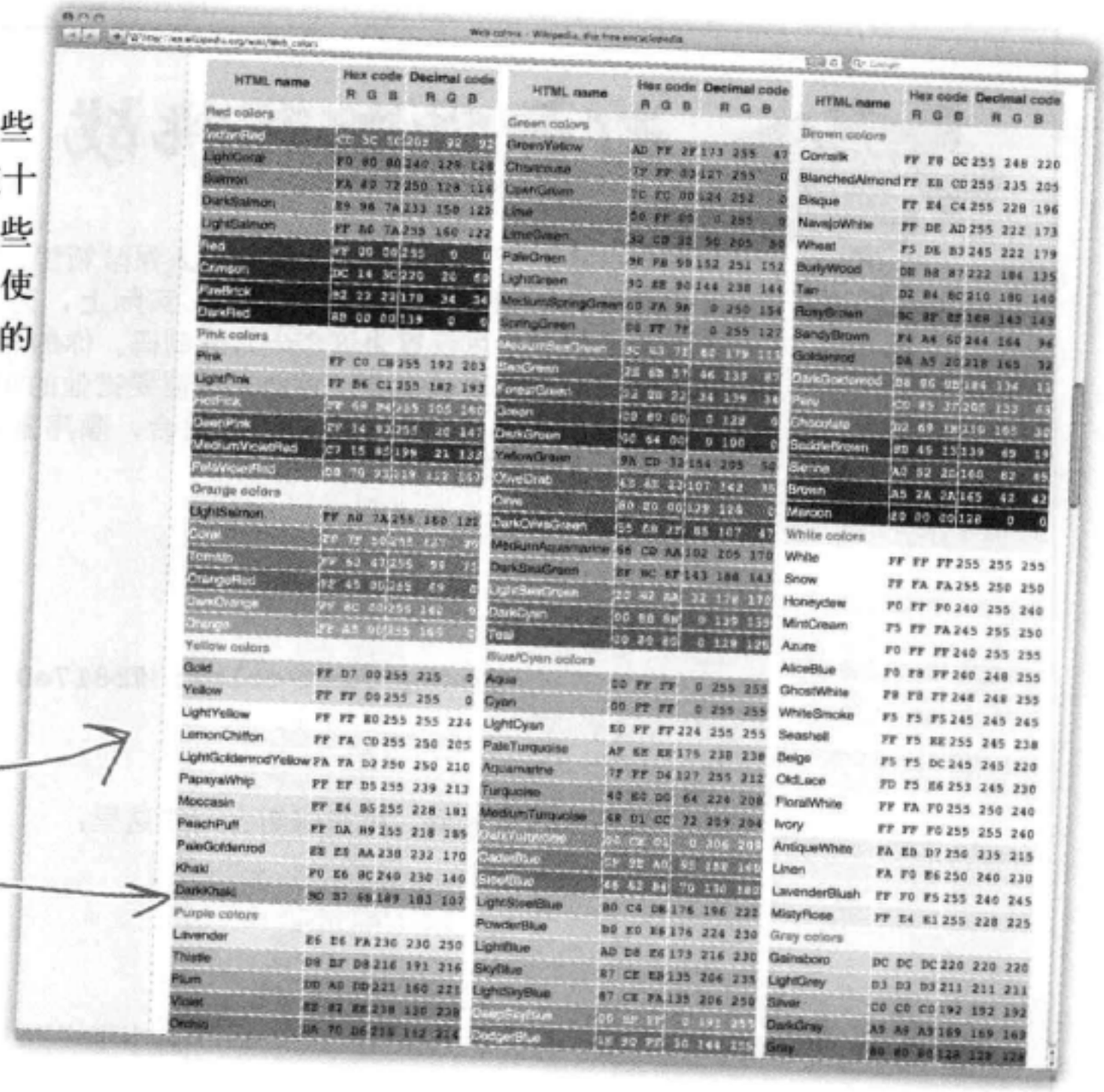
一旦选择了一个颜色，颜色选择器会为你显示这个颜色的rgb值和十六进制码。

使用在线颜色表

在Web上还可以找到一些有用的颜色表。这些颜色表通常会显示各种Web颜色，根据相应十六进制码的多种不同标准加以管理。使用这些颜色很容易，只需要选择你希望在页面中使用的颜色，然后把它的十六进制码复制到你的CSS中。

这个颜色表由Wikipedia维护（地址是http://en.wikipedia.org/wiki/Web_colors）。通过搜索“HTML color charts”（HTML颜色表），还可以找到很多其他的颜色表。

试试这个颜色名，看看在不同浏览器上能不能用。如果不行，就要用它的十六进制码。



there are no Dumb Questions

问：我听说，如果没有使用Web安全颜色，我的页面在其他浏览器上就不会有正确的显示。为什么还不讨论Web安全颜色呢？

答：让我们回到Web浏览器的早期阶段，那时很少有人能拥有支持大量颜色的计算机屏幕，所以当时创建了Web安全调色板，以确保页面在大多数显示屏上能有一致的显示。

如今，情况已经发生了巨大变化，大多数Web用户的计算机显示屏都支持数百万种颜色。所以，除非你有一些特殊的用户，你知道他们的显示屏只支持有限的颜色，否则完全可以不去过多地考虑Web安全颜色，可以把它们当作“古董”。

问：我现在知道该如何指定颜色了，不过怎么选择能合理搭配的字体颜色呢？

答：要准确地回答这个问题，需要一本书才能讲清楚，不过选择字体颜色有一些基本原则。最重要的是，对于文本和背景，要使用对比度最大的颜色，这样能帮助提高可读性。例如，白色背景上的黑色文本对比度就最高。没有必要一直使用黑白色，不过可以尝试让文本使用深色，而背景使用浅色。有些颜色配合在一起使用时，可能会产生很怪异的视觉效果（如蓝色和橙色，或者红色和绿色），所以在发布到网上展示给全世界之前，先让你的朋友看看颜色搭配的效果。

问：我见过类似#cb0的十六进制码，这是什么意思？

答：如果每一组分量中两位数字都相同，你可以使用简写。因此，例如，#ccbb00可以缩写为#cb0，或者#11eeaa可以缩写为#1ea。不过，如果十六进制码是#ccbb10，则不能使用缩写。

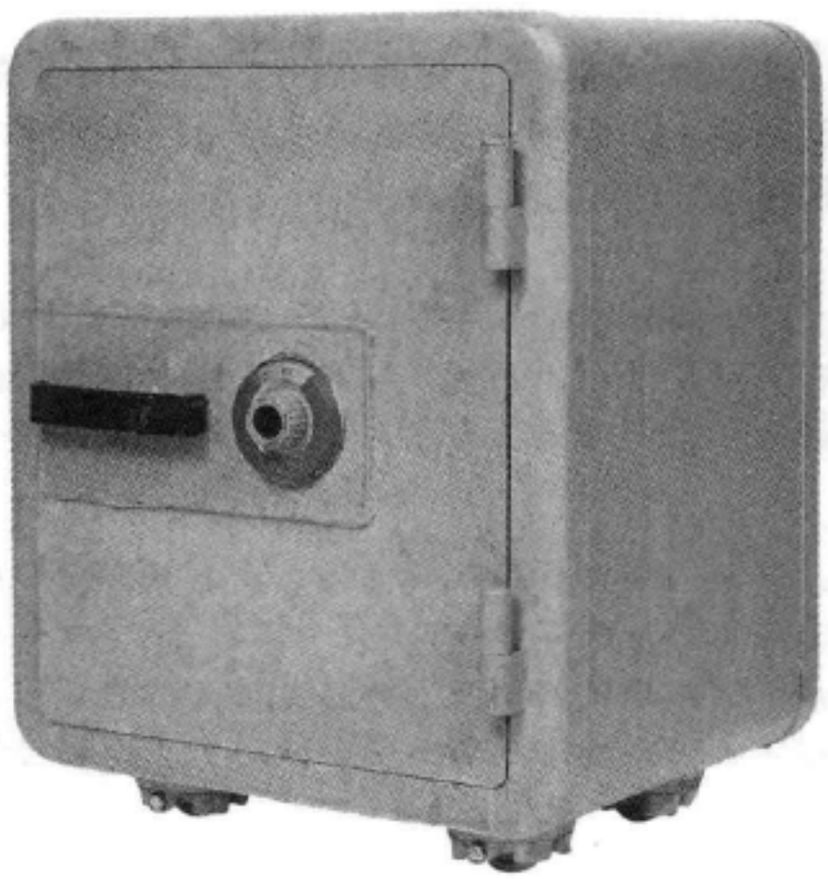
打开保险箱挑战

Evel博士的主计划就锁在他的个人保险箱里，你刚得到一个提示，他把保险箱密码组合编写成十六进制码。实际上，为了不忘记这个密码组合，他将主页的背景色设置为这个十六进制码。你的任务是破解他的十六进制码，找到保险箱的密码组合。为此，只需要把他的Web颜色转换成红、绿、蓝十进制数值，你就得到了他的密码组合，顺序是右—左—右。下面是他的主页的背景Web颜色：

```
body {
      background-color: #b817e0;
}
```

破解十六进制码，把密码组合写在这里：

右 ..... **左** ..... **右** .....



再回到Tony的页面……我们要把标题变成橙色，并增加一个下划线

既然你已经对颜色有了充分的了解，现在为Tony的Web页面加一点颜色吧。他喜欢橙色，希望在页面上加上橙色。不过我们不打算把所有文本都变成橙色—这可能很难看，而且在白色背景上也很难阅读。我们将在标题里加一点颜色。这种橙色比较深，文本和背景之间可以有很好的对比，另外与照片中的橙色（Tony的衬衫）也很搭配，这样就会在标题和照片之间创建一个颜色关系，将图像和文本联系在一起。此外为了确保突出标题，与日志内容区分开，我们还将在每个标题下面增加一个下划线。你还没有见过如何加下划线，不过先跟着我们照做，后面还会对文本装饰做更多说明。

下面是CSS中要完成的所有修改。在你的“journal.css”文件中完成这些修改。

```
@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;
    text-decoration: underline;
}

h1 {
    font-family: "Emblema One", sans-serif;
    font-size: 220%;
}

h2 {
    font-weight: normal;
    font-size: 130%;
}

blockquote {
    font-style: italic;
}
```

要让<h1>和<h2>都为橙色，所以把color属性放在一个共同的规则中。

这是Tony想要的橙色的十六进制码，，也就是rgb(80%, 40%, 0%)。

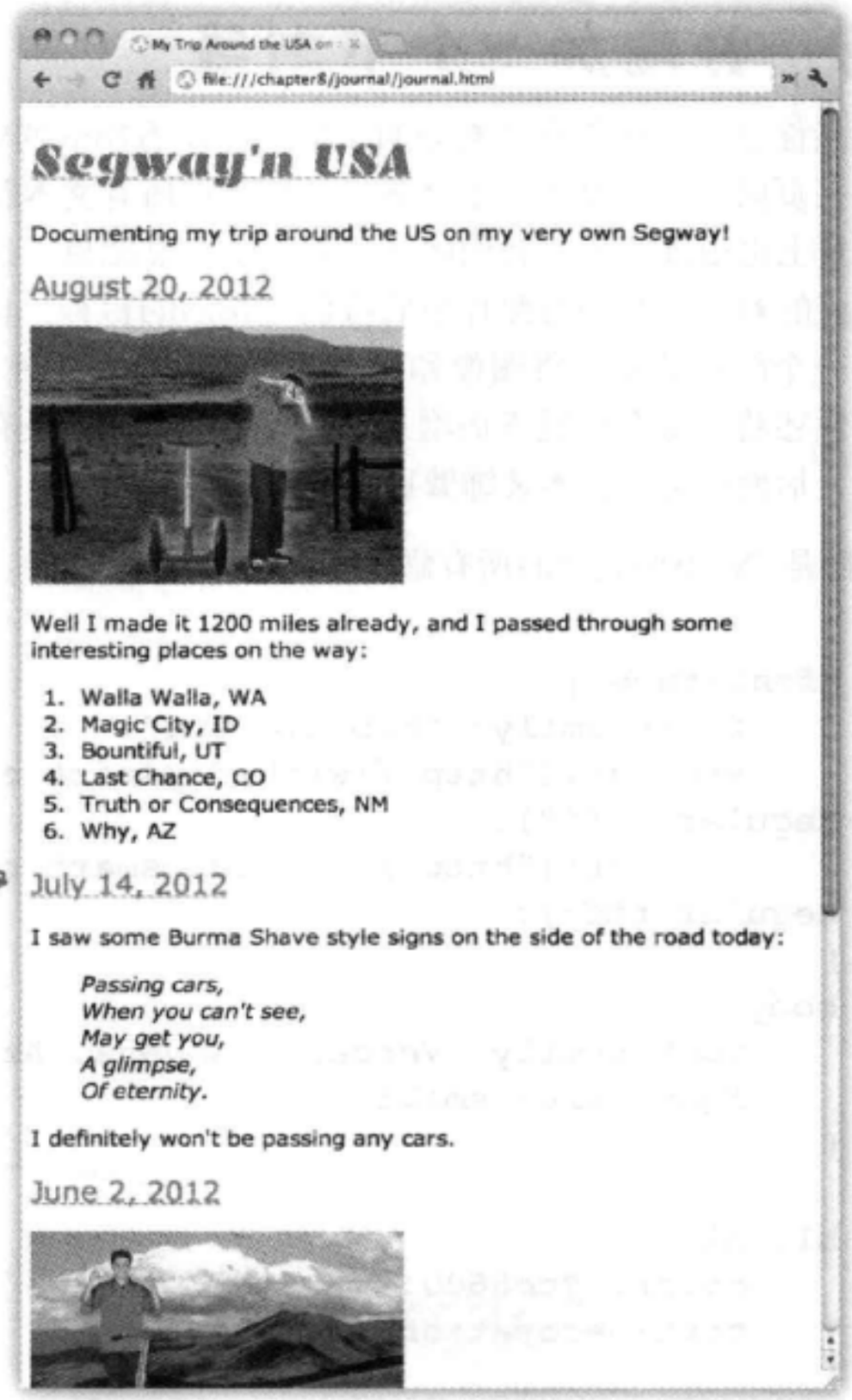
可以这样创建一个下划线。我们使用了text-decoration属性，将它设置为underline。

注意我们为<h1>和<h2>标题创建了一个新规则。这是一个很好的做法，因为这样可以减少重复。

测试Tony的橙色标题

对“journal.css”文件完成修改，在h1，h2规则中增加color属性，接下来重新加载这个Web页面，检查结果。

现在<h1>和<h2>标题都是橙色。这与Tony的橙色主题和衬衫很搭配。



这些标题还加了下划线。嗯……我们以为这是突出标题的一种好的做法，不过实际上它们看起来有点像可单击的链接，因为人们总是认为Web页面中有下划线的内容都是可单击的。

所以，加下划线可能不是一个很好的选择。下面快速了解一下其他的文本装饰，然后再重新考虑Web页面中的这些下划线。

Sharpen your pencil

这些颜色有什么共同点？可以在一个Web页面中分别尝试这些颜色（比如作为一个字体颜色），或者使用你的照片编辑应用的颜色选择工具来确定它们到底是什么颜色（直接把这些十六进制码输入到对话框中）。

#111111	#444444	#777777	#aaaaaa	#dddddd
#222222	#555555	#888888	#bbbbbb	#eeeeee
#333333	#666666	#999999	#cccccc	

用不到一页的篇幅介绍关于文本装饰所需了解的全部知识

文本装饰允许你为文本增加一些装饰性的效果，如下划线、上划线和删除线。要增加一个文本装饰，只需设置元素的text-decoration属性。就像这样：

```
em {
      text-decoration: line-through;
}
```

这个规则会使元素上有一个从文本中间穿过的横线。

一次可以设置多个装饰。假如你想同时对一个元素加下划线和上划线，可以如下指定文本装饰：

```
em {
      text-decoration: underline overline;
}
```

这个规则使得元素有一个下划线和一个上划线。

如果文本继承了你不想要的文本装饰，可以使用值“none”来去除装饰：

```
em {
      text-decoration: none;
}
```

利用这个规则，元素将没有任何装饰。

there are no Dumb Questions

问：如果有两个不同的规则，一个指定了上划线，另一个指定了下划线，它们会累加吗？这两个装饰都能得到吗？

答：不能。你要把这两个值合并到一个规则中，才能同时得到这两个文本装饰。对于text-decoration只会选择一个规则，不同规则中的装饰不会累加在一起。只有为text-decoration样式选择的规则才能确定使用什么装饰，所以要得到这两个装饰，唯一的办法就是在同一个text-decoration声明中同时指定。

问：我一直想问，为什么color属性不叫text-color？

答：color属性实际上控制着一个元素的前景色，所以它会控制文本和边框颜色，不过你也可以用border-color属性为边框指定自己的颜色。

问：我喜欢删除线装饰。能不能在我编辑的文本上使用这个装饰，来指示需要删除的内容？

答：当然可以，不过还有一种更好的方法。HTML提供了一个我们还没有谈到的元素：<del>，它能把HTML中的某些内容标记为要删除的内容。还有一个类似的元素，名为<ins>，这会标记要插入的内容。通常浏览器会分别用一个删除线和下划线指定这些元素的样式，你也可以用你喜欢的方式指定它们的样式。通过使用<del>和<ins>，在指定样式的同时还可以指出内容的含义。

删除下划线……

下面去掉让人误解的下划线，像在休闲室网页上一样增加一个漂亮的下边框。为此，打开你的“journal.css”文件，对合并的h1，h2规则完成以下修改：

```
h1, h2 {
      color: #cc6600;
      border-bottom: thin dotted #888888;
      text-decoration: underline;
}
```

在<h1>和<h2>元素下面增加一个下边框。可以像读英语一样读这个规则：“在下边框上增加一条细的虚线，颜色为#888888”……

下一章中，我们还会详细讨论边框。坚持一下，就快完工了！

删除文本装饰。

这是我们新的“下划线”，与文本装饰下划线相比，这样更美观，也不容易让人误解。

现在<h1>和<h2>元素下面有一个下边框，而不是下划线。

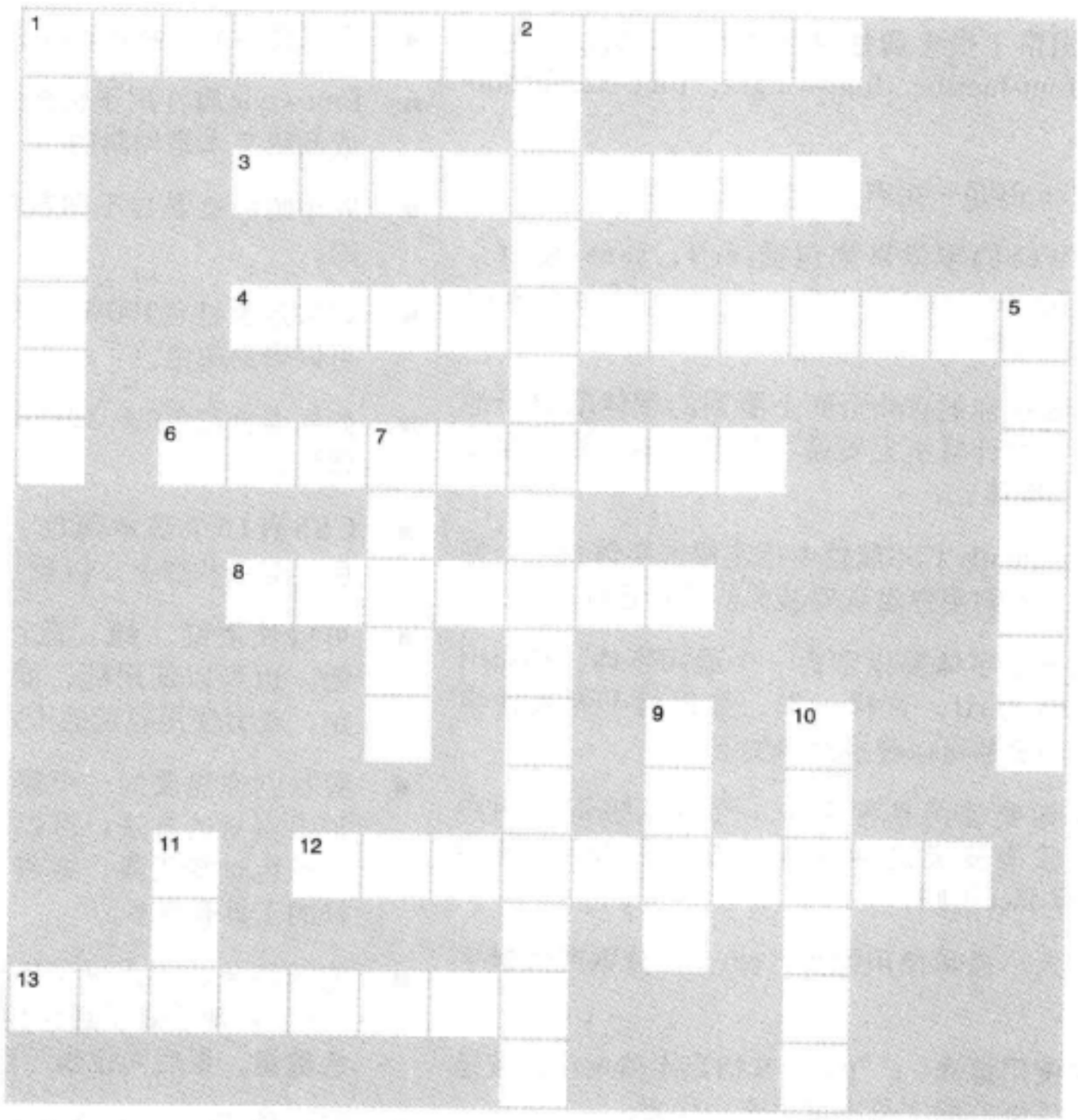
注意这些下边框一直延伸到页面边缘，而不只是在文本下面出现。为什么？下一章会告诉你答案。

BULLET POINTS

- CSS提供了很多属性对字体的外观加以控制，包括font-family，font-weight，font-size和font-style。
- font-family是一组有共同特征的字体。
- 用于Web的字体系列包括serif，sans-serif，monospace，cursive和fantasy。serif和sans-serif字体最为常用。
- 访问者在你的Web页面上看到的字体取决于他们自己的计算机上安装了哪些字体，除非你使用Web字体。
- 在font-family CSS属性中指定候选字体是一个好主意，以防用户没有安装你的首选字体。
- 最后一个字体要指定为一个通用字体，如serif或sans-serif，这样一来，如果找不到其他字体，浏览器可以替换适当的字体。
- 如果你要使用某种字体，而默认情况下用户可能没有安装这种字体，可以在CSS中使用@font-face规则。
- 字体大小通常使用像素、em、百分数或关键字指定。
- 如果使用像素 （“px”)来指定字体大小，就是在告诉浏览器字母高度是多少像素。
- em和%是相对字体大小，所以使用em和%指定字体大小时，就意味着字体大小要相对于其父元素的字体大小指定。
- 使用相对字体大小可以让你的页面更可维护。
- 在body规则中使用字体大小关键字来设置基本字体大小，这样如果用户希望文本更大或更小，所有浏览器就能按比例缩放字体大小。
- 可以使用font-weight CSS属性设置文本为粗体。
- font-style属性用于创建斜体或倾斜文本。斜体或倾斜文本是倾斜的。
- Web颜色是混合不同数量的红、绿、蓝色得到的。
- 如果混合红色100%，绿色100%和蓝色100%，可以得到白色。
- 如果混合红色0%，绿色0%和蓝色0%，可以得到黑色。
- CSS有16个基本颜色，包括黑色、白色、红色、蓝色和绿色，以及150种扩展颜色。
- 可以使用红、绿、蓝百分数指定你想要的颜色，也可以使用红、绿、蓝数值（0~255）指定，或者使用颜色的十六进制码来指定颜色。
- 要找到你想要的一个颜色的十六进制码，有一种很容易的方法，可以使用一个照片编辑应用的颜色选择工具，或者某个在线Web工具，这样的工具有很多。
- 表示颜色的十六进制码有6位，每一位取值为0~F。前两位表示红色数量，接下来两位表示绿色数量，最后两位表示蓝色数量。
- 可以使用text-decoration属性为文本创建一个下划线。有下划线的文档通常会被用户误以为是链接文本，所以要谨慎使用这个属性。

HTML填字游戏

这一章你已经学到不少知识：字体、颜色、粗细和风格。现在再来做一个填字游戏，把这些知识牢牢记住。



横向

1. 类似的字体归组为_____。
3. CSS中使用_____ 规则从Web加载字体。
4. 在font-family属性中指定字体时，就是在指定_____。
6. 一般认为这种字体在计算机显示屏上更清晰，更容易阅读。
8. 可以按像素，em或_____指定字体。
12. 下划线和删除线都是文本_____的例子。
13. em和%都是 _____ 大小。

纵向

1. Web页面中几乎从来不用的字体系列。
2. 不能很好地处理像素字体大小的浏览器。
5. 十六进制码使用 _____个数。
7. 带有小衬线的字体。
9. 从#111111到#EEEEEE颜色都是不同程度的_____。
10. 控制字体的粗细。
11. 这个元素可以用来标记要删除的文本。

字体磁贴答案

你的任务是帮助下面的虚构字体找到回家的路，回到他们自己的字体系列中。把各个冰箱磁贴放在正确的字体系列中。继续学习下面的内容之前请检查你的答案。下面给出答案：

Monospace 字体系列

Messenger

Bainbridge

Fantasy字体系列

Crush

Cursive 字体系列

Cartoon

Sans-serif字体系列

Iceland

Angel

Nautica

Serif字体系列

Savannah

Quarter

Palomino

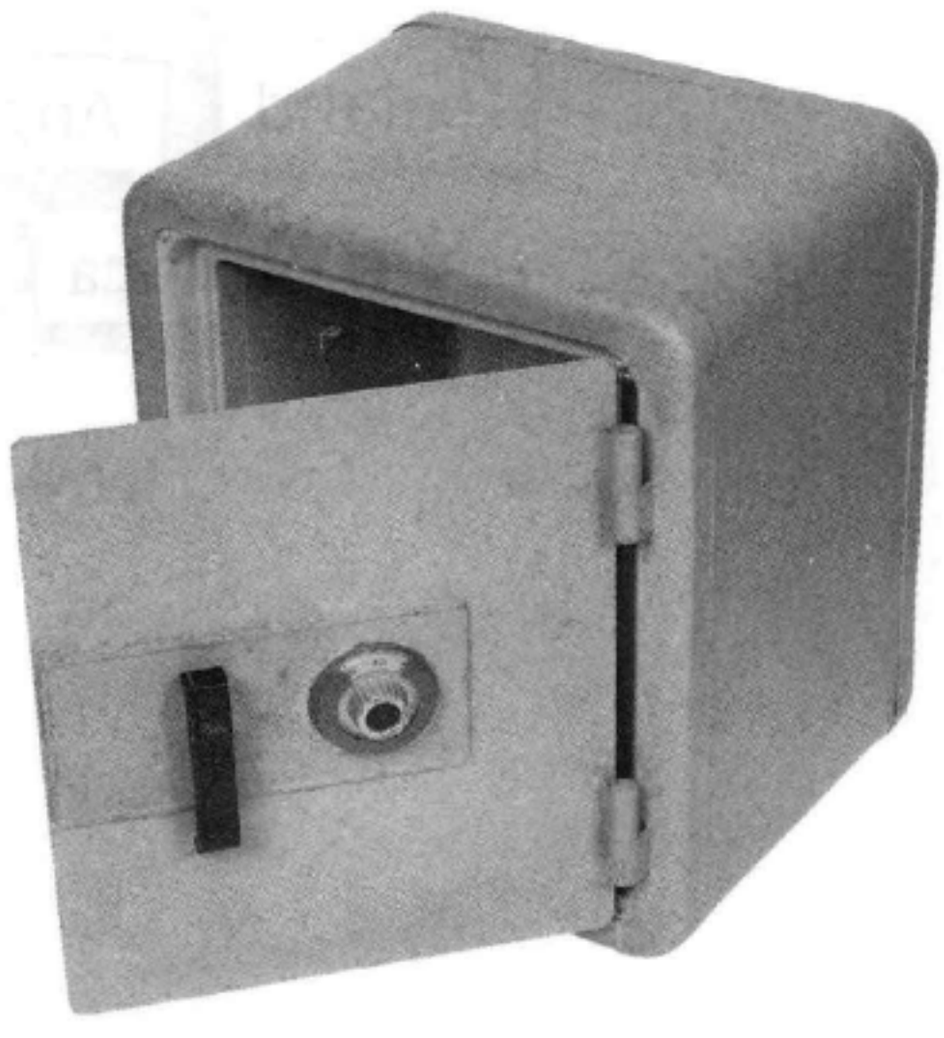
打开保险箱挑战答案

Evel博士的主计划就锁在他的个人保险箱里，你刚得到一个提示，他把保险箱密码组合编写成十六进制码。实际上，为了不忘记这个密码组合，他将主页的背景色设置为这个十六进制码。你的任务是破解他的十六进制码，找到保险箱的密码组合。为此，只需要把他的Web颜色转换成红、绿、蓝十进制数值，你就得到了他的密码组合，顺序是右－左－右。下面是他的主页的背景Web颜色：

```
body {
        background-color: #b817e0;
}
```

破解十六进制码，在这里写出密码组合：

右 (11 × 16) + 8 = 184　左 (1 × 16) + 7 = 23　右 (14 × 16) + 0 = 224



Sharpen your pencil Solution

这些颜色有什么共同点？可以在一个Web页面中分别尝试这些颜色，或者使用颜色选择工具来确定它们到底是什么颜色（直接把这些十六进制码输入到对话框中）。

如果颜色的十六进制码中每一位都相同（都是同一个数），这些颜色都是灰色，从深灰（接近黑色）到浅灰色（接近白色）。

HTML填字游戏答案

1F	O	N	T	F	A	M	2I	L	I	E	S							
A							N											
N			3@	F	O	N	T	F	A	C	E							
T							E											
A			4A	L	T	E	R	N	A	T	I	V	E	5S				
S							N							I				
Y		6S	A	N	7S	S	E	R	I	F				X				
					E		T							T				
			8P	E	R	C	E	N	T					E				
					I		X							E				
					F		P		9G		10W			N				
							L		R		E							
		11D		12D	E	C	O	R	A	T	I	O	N					
		E					R		Y		G							
13R	E	L	A	T	I	V	E				H							
							R				T							

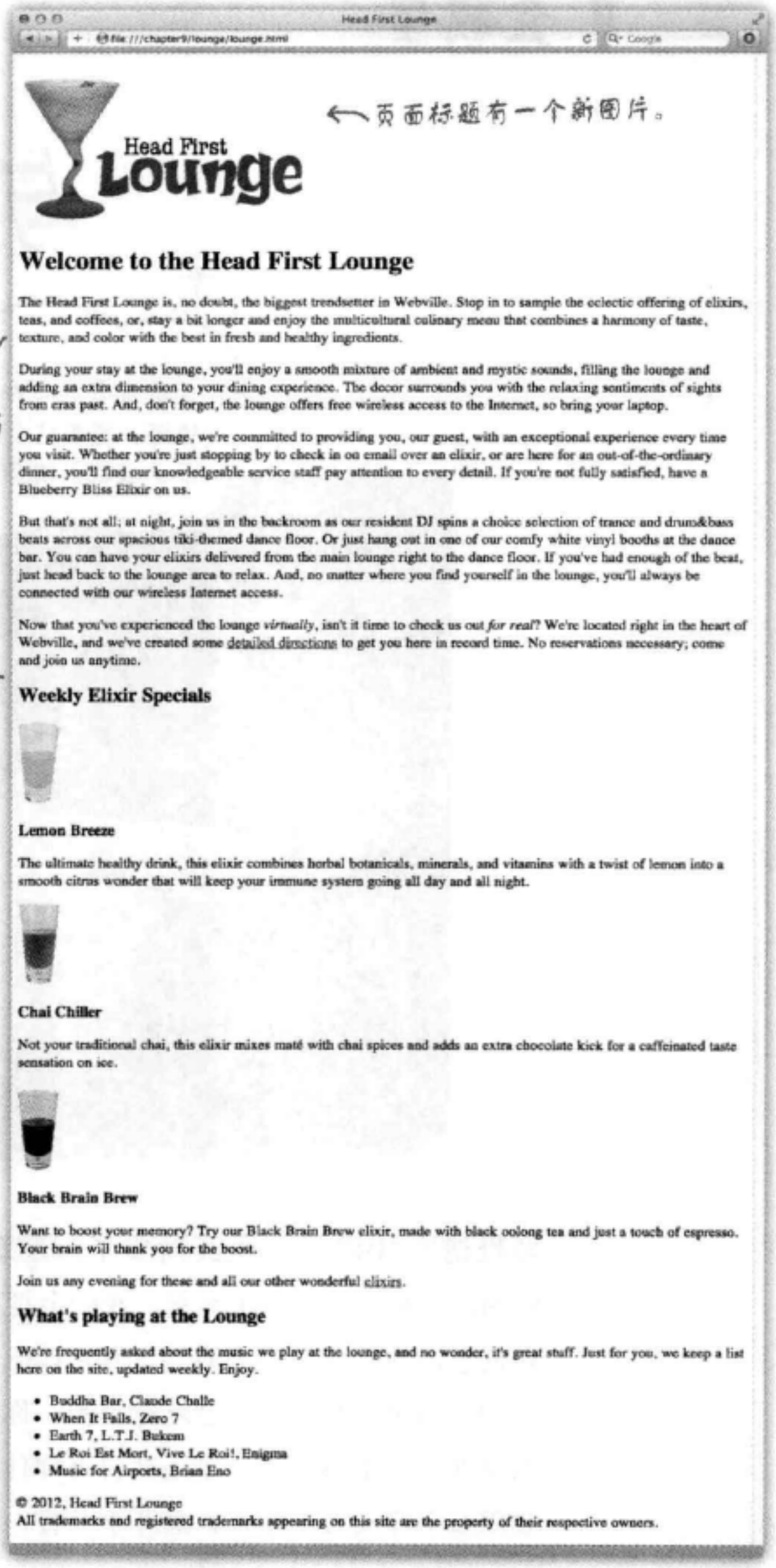
9 盒模型

与元素亲密接触

要推进Web建设，确实需要了解建筑材料。这一章中，我们会仔细研究我们的建筑材料：HTML元素。我们会把块元素和内联元素放在显微镜下，仔细观察它们的组成。你会看到，利用CSS，对于构造元素的各个方面几乎都可以控制。不过，我们并不会就此止步，你还会了解如何为元素提供唯一的身份。如果这些还不够，这一章还会告诉你什么时候要使用多个样式表，以及为什么要用多个样式表。好了，翻开下一页，与元素来个亲密接触吧。

休闲室要升级

经过前8章的学习，你已经取得了长足进步，Head First休闲室也有了很大改进。实际上，在接下来的两章中，我们还将对它做一个全面升级，不仅会在主页面中补充新的内容，还会从头开始建立全新的样式。为了满足你的好奇心，在正式开始之前，先让你看一眼。请对照检查这两个页面，这一页给出了包含所有新内容的休闲室页面，不过没有加任何样式。在下一页上，你会看到增加了样式的版本，这就是下一章结束时将要创建的最终页面。



标题是青绿色（aquamarine），与网站颜色主题一致。字体也是可读性很好的sans-serif。

对这个段落设置了很多样式，可以帮助它从其余文本中“脱颖而出”，另外也让页面更生动。看起来它的字体是一种serif字体，这与主文本不同。

饮料的样式有很大调整，现在显示的饮料让人禁不住都想喝一杯。

饮料被移到一边。这是怎么做到的？

音乐CD和艺术家现在也指定了样式。

页脚居中，用很小的字体显示。

改进后的超炫新休闲室

不难看了吧。现在这个休闲室设计对你来说可能有点……嗯，怎么说呢，是不是太“炫”了？不过，嘿，这可是一个休闲室，当然要让人看着舒服才行。你可能认为这个设计看起来相当复杂，不过想想看，这些技术也可以用来改造你的页面。这两章学完后，类似这样的设计对你来说就是小菜一碟了。

创建新休闲室

正式开始建设之前，先来熟悉这个新休闲室。你需要做到：

1. 查看“chapter9/lounge”文件夹，你会看到包含所有新内容的文件“lounge.html”。在你的编辑器中打开这个文件，浏览一下。所有内容应该都不陌生，这里有标题、段落和一些图像，还有一个列表。
2. 这一章主要是为这个HTML增加样式，所以需要有地方来存放你的CSS。为这个休闲室创建的所有新样式都放在样式表文件“lounge.css”中，所以你会看到，“lounge.html”<head>中的<link>元素还在，不过以前版本的“lounge.css”样式表已经不见了。

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

要记住，这个<link>元素告诉浏览器查找一个名为“lounge.css”的外部样式表。

3. 接下来，需要在“chapter9/lounge”文件夹中创建新的“lounge.css”文件。这个文件将包含为新休闲室创建的所有新CSS。

先做一些简单的升级

现在可以开始对这个休闲室增加样式了。下面增加一些CSS规则，为后面的工作奠定基础，比如字体系列、大小和一些颜色，这会使休闲室迅速改观（也可以很好地复习上一章的内容）。所以，打开你的“lounge.css”文件，增加以下规则。

```
body {
      font-size:   small;
      font-family: Verdana, Helvetica, Arial, sans-serif;
}

h1, h2 {
      color: #007e7e;
}

h1 {
      font-size: 150%;
}

h2 {
      font-size: 130%;
}
```

这是页面的默认字体大小。

休闲室将采用sans-serif字体系列。我们先选择了几个候选字体，声明的最后指定了通用sans-serif字体。

我们将<h1>和<h2>元素的颜色设置为一种青绿色，使它与logo中的酒杯颜色一致。

现在为<h1>和<h2>指定便于阅读的标题字体大小。因为我们为这两级标题设置了两个不同的大小，所以需要不同的规则，不能把这些设置增加到<h1>和<h2>的合并规则中。

快速测试

下面做一个快速测试，看看这些样式对页面有什么影响。确保完成以上所有修改，然后保存并测试。

标题现在是sans-serif字体，而且采用了与logo一致的颜色，这就为页面创建了一个主题。

段落文本也是sans-serif字体，因为每个元素都会继承<body>的font-family属性。

<h2>标题也指定了新颜色，并使用sans-serif字体，不过小一点。

对<h3>没有应用任何样式，所以它会从<body>继承font-family属性。

这个链接使用默认的蓝色，看起来很突兀，很怪异。我们会修正这个问题（稍后）。

再做些调整

在做更大变动之前，我们再对这个休闲室做一个调整。这个调整涉及一个新属性，之前你还从来没见过，不过既然已经学到这里了，说明你已经有了足够的经验，所以我们不打算再把你当小孩子看，每次遇到一个新属性都手把手地教你。好了，你自己来试试吧。

我们要做的是：需要调整整个页面上文本的行高，使得各行之间有更大的垂直间距。为此，要在body规则中增加一个line-height属性：

增加文本的行高可以改善可读性。这样做还可以使页面不同部分之间形成对比，产生反差（稍后会看到这是如何做到的）。

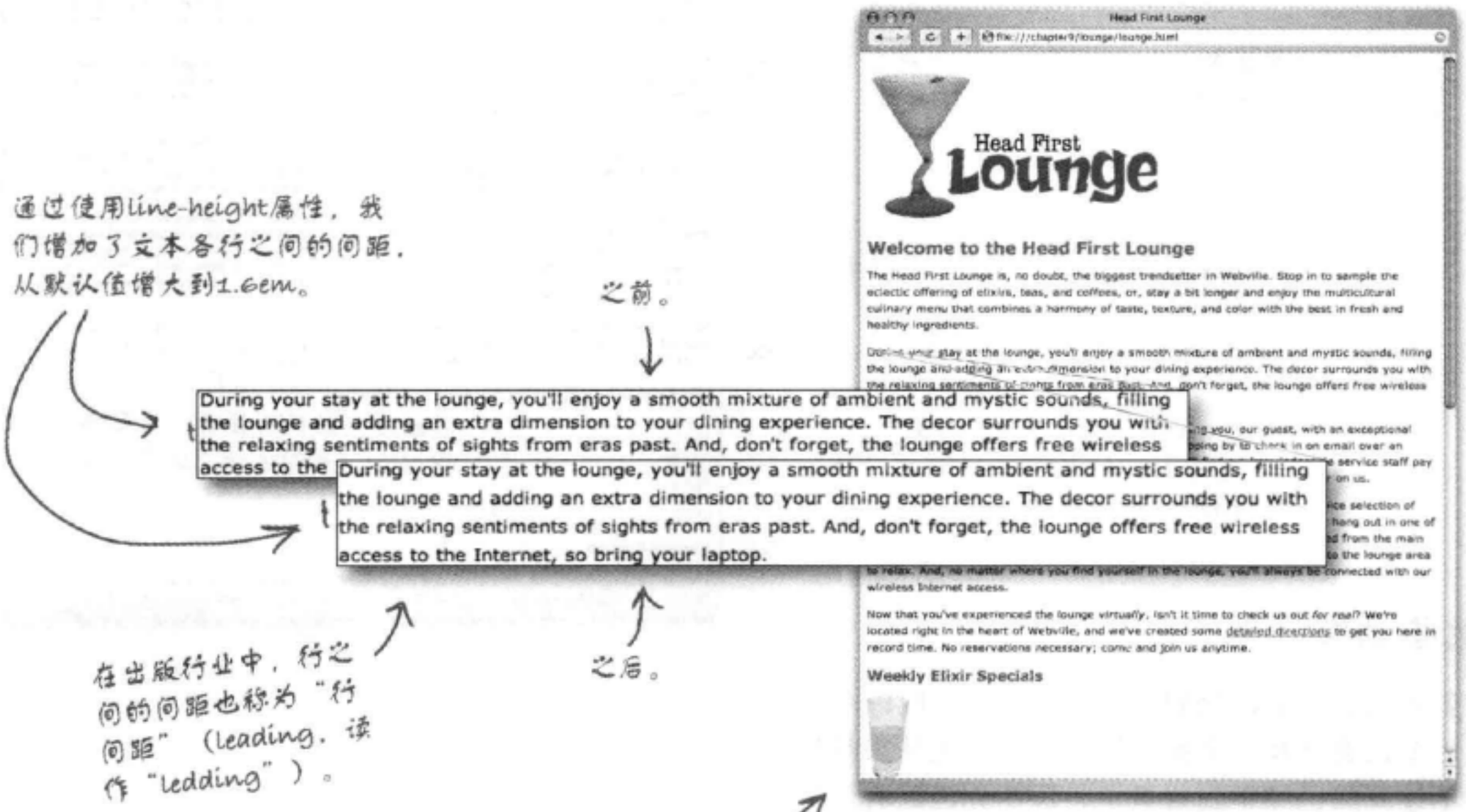
```
body {
        font-size:   small;
        font-family: Verdana, Helvetica, Arial, sans-serif;
        line-height: 1.6em;
}
```

这里将各行之间的间距改为1.6em。换句话说，就是字体大小的1.6倍。

查看新行高

你可能已经猜到了，line-height属性允许你指定文本中各行之间的垂直间距量。类似于其他与字体相关的属性，可以按像素指定行高，也可以使用em或百分数值相对于字体大小来指定。

下面来看休闲室增加line-height属性后的效果。确保在CSS文件中增加这个line-height属性，然后保存。刷新页面时应该能看行高增加了。



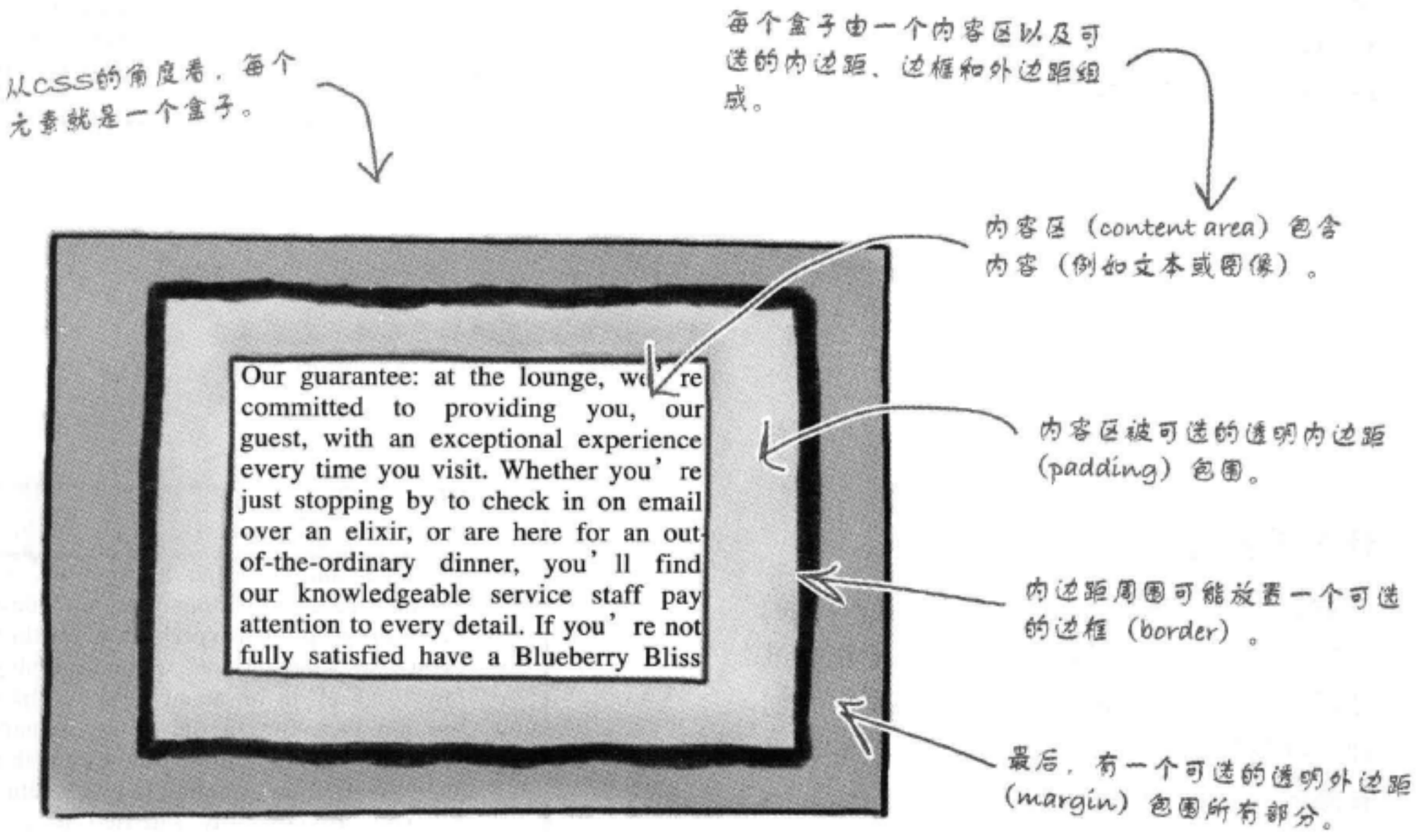
尝试使用几个不同的line-height值，如200%、.5em和20px，看看效果。哪一个最好？哪一个最差？哪一个最可读？完成后，一定要把line-height值再改回1.6em。

准备全面翻新

虽然这一章只是刚刚开始，你才只学了几页，但这个新休闲室已经有了很多文本样式，祝贺你！

接下来会更有意思。我们不再只是改变简单的元素属性，如大小、颜色和装饰，而是要对元素显示的一些基本方面真正做些调整。现在你要面对挑战了。

不过，要接受挑战，必须先了解盒模型（box model）。什么是盒模型？这就是CSS看待元素的一种方式。CSS将每个元素看作由一个盒子表示。下面来看这是什么意思。



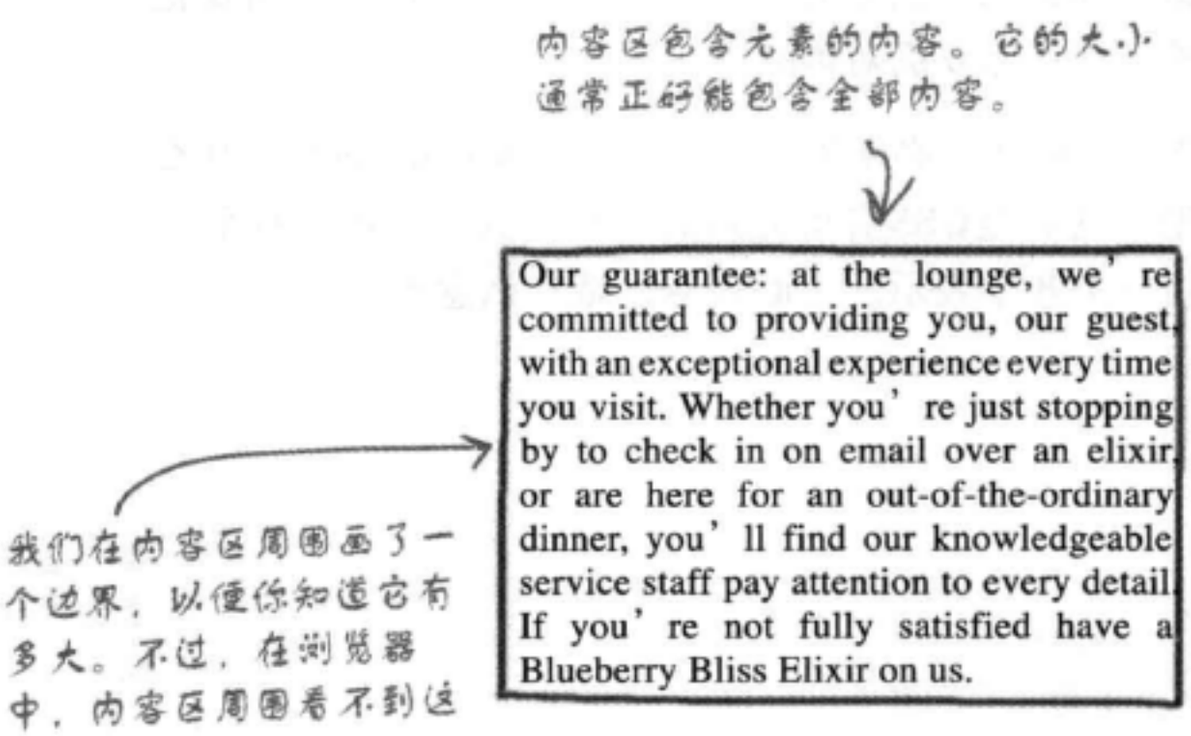
所有元素都被当作盒子：段落、标题、块引用、列表、列表项等。甚至内联元素（如和链接）在CSS看来也是盒子。

仔细分析盒模型

利用CSS，你可以对盒子的所有方面加以控制：内容周围内边距的大小、元素是否有边框（什么类型的边框，以及边框的大小），另外元素之间有多大的外边距。下面来看盒子的每一部分，并了解它们的作用：

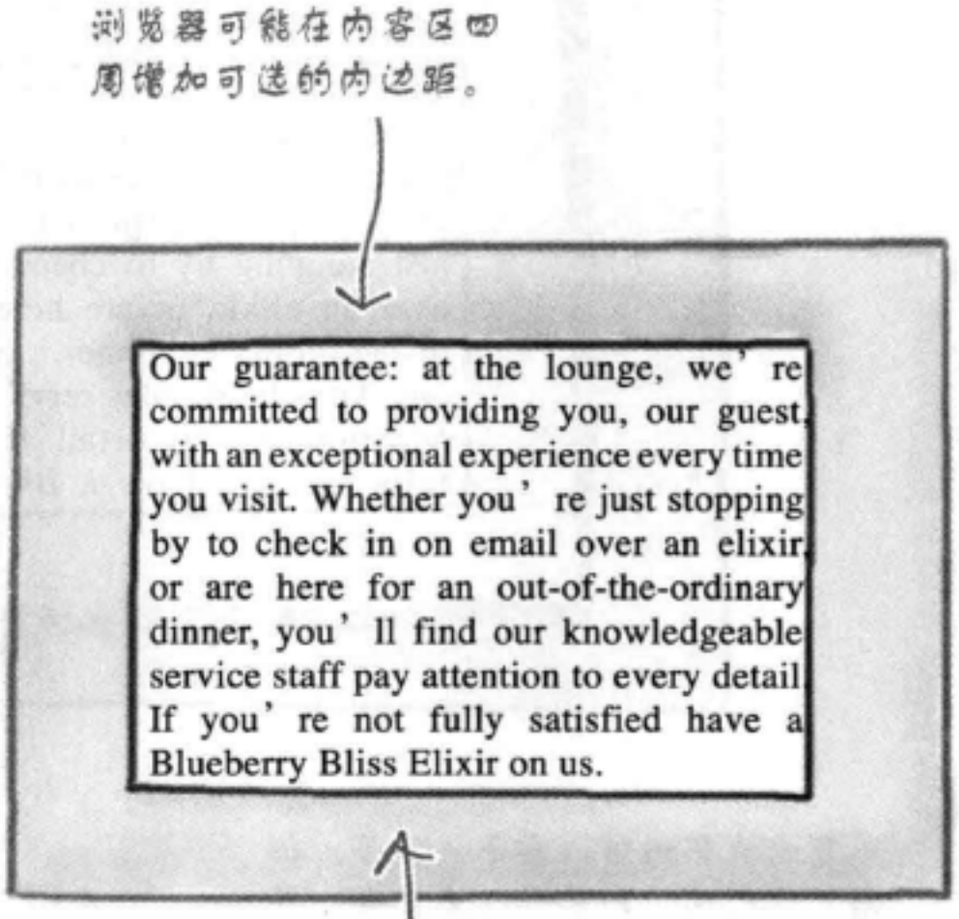
什么是内容区？

每个元素都会有一些内容，如文本或图像，这个内容会放在一个盒子里，这个盒子的大小正好能包含所有内容。注意，在内容区中，内容与盒子边缘之间没有空间。



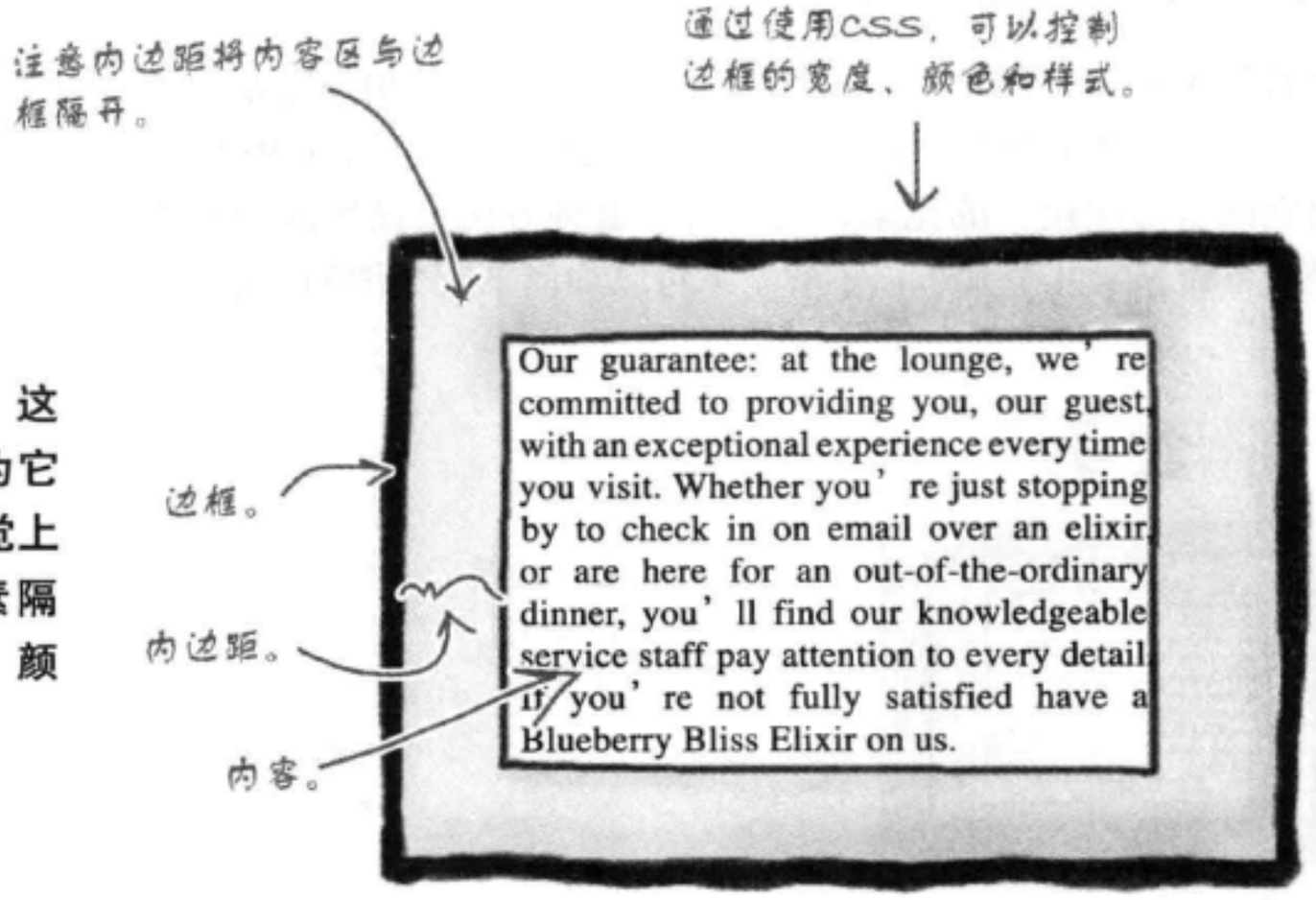
什么是内边距？

所有盒子在内容区周围可能有一层内边距。内边距是可选的，所以不一定有，不过通过使用内边距，可以在内容与盒子边框之间创建一些看得到的空间。内边距是透明的，没有颜色，也没有自己的装饰。



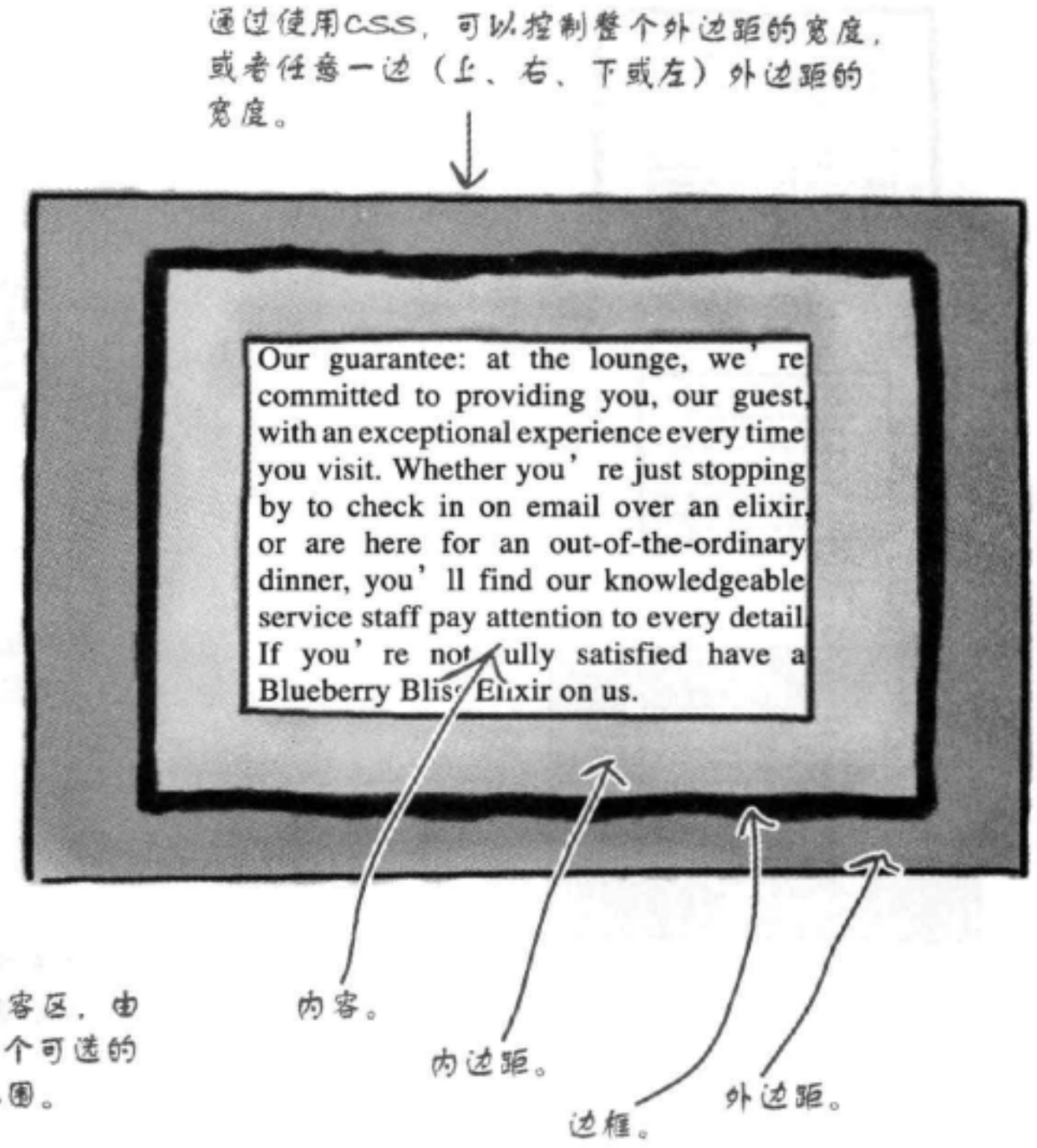
什么是边框?

元素周围可以有一个可选的边框。这个边框会包围内边距，另外，因为它是围绕内容的一条线，这就从视觉上使内容与同一页面上的其他元素隔开。边框可以有各种不同的宽度、颜色和样式。



什么是外边距?

外边距也是可选的，包围着边框。利用外边距，可以在同一个页面上的不同元素之间增加空间。如果两个盒子紧挨着，外边距就相当于它们之间的空间。类似于内边距，外边距也是透明的，本身没有颜色或装饰。

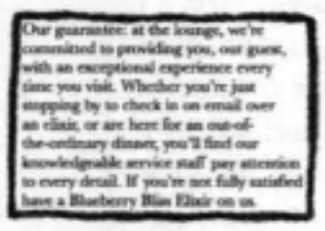


对盒子能做哪些设置

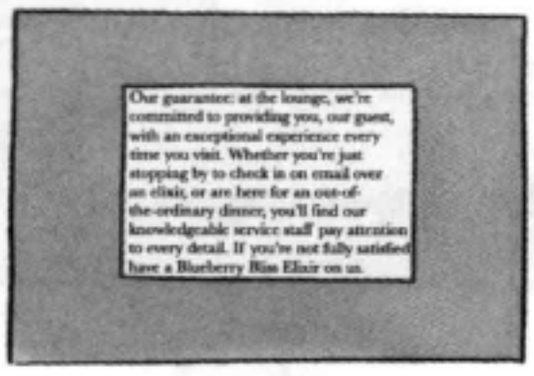
盒模型看起来可能很简单，只有内容、内边距、边框和外边距。不过，如果把它们都结合在一起，对于一个可能有内部空间（内边距）和外围空间（外边距）的元素，就会有无数种方法来设置这个元素的布局。下面来看几个盒子，了解一下可以如何来改变你的元素。

可以指定盒子的样式，包括内边距、边框和外边距。

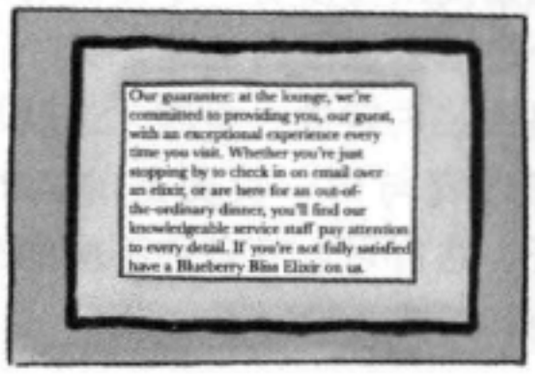
或者只有内边距和一个边框。



或者只有边框。

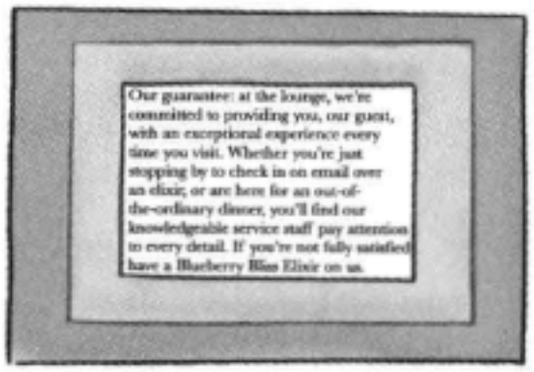


或者有外边距，但没有边框和内边距。

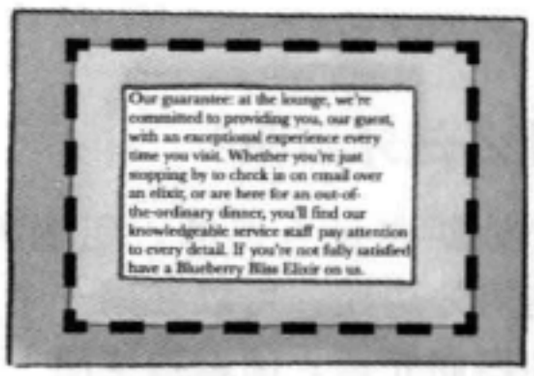


可以有实线边框，边框可以是粗线或细线。

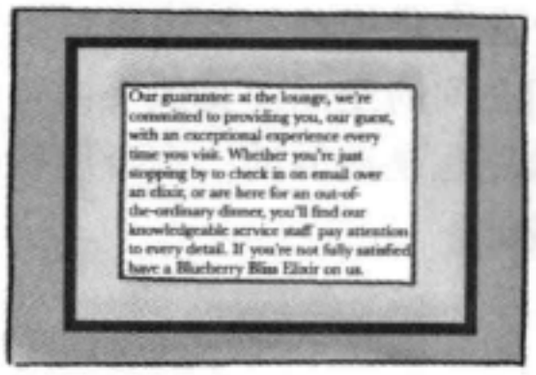
或者根本没有边框。



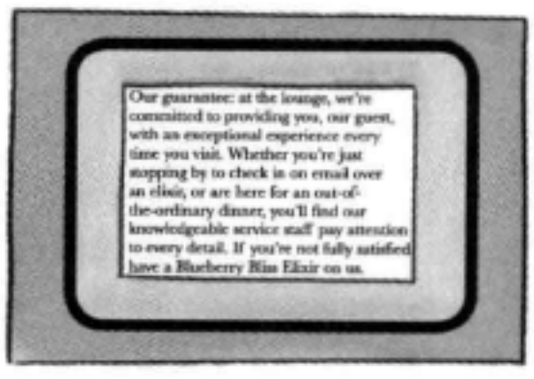
或者可以从8种不同样式的边框中选择，如虚线边框。



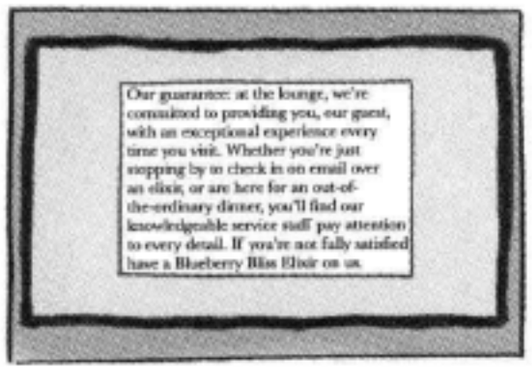
或者对边框指定颜色。



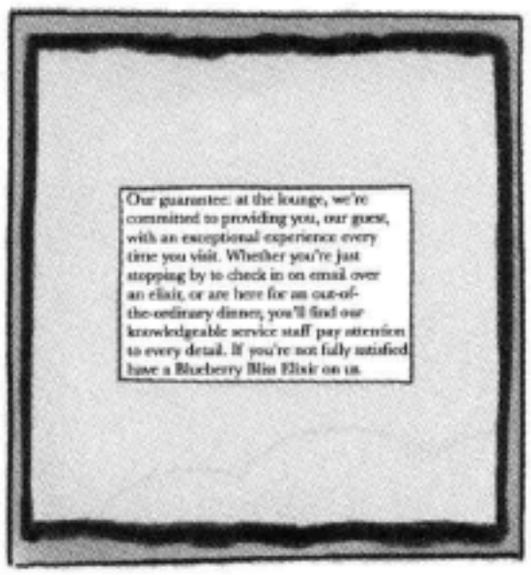
或者甚至对边框创建圆角。



内边距



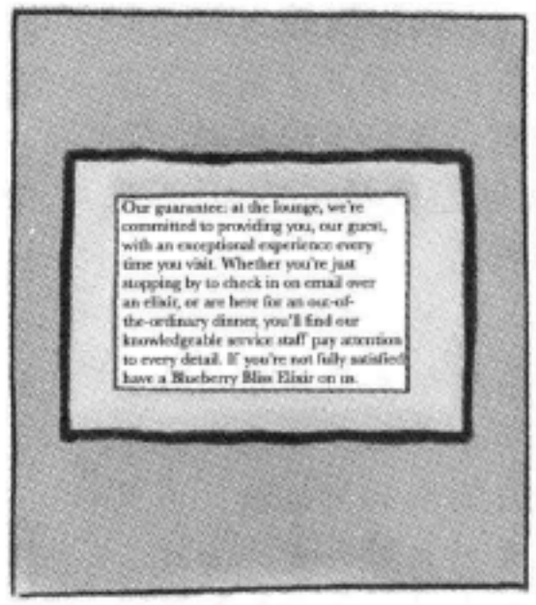
利用CSS，可以控制内容区任意一边的内边距。这里有很大的左右内边距。



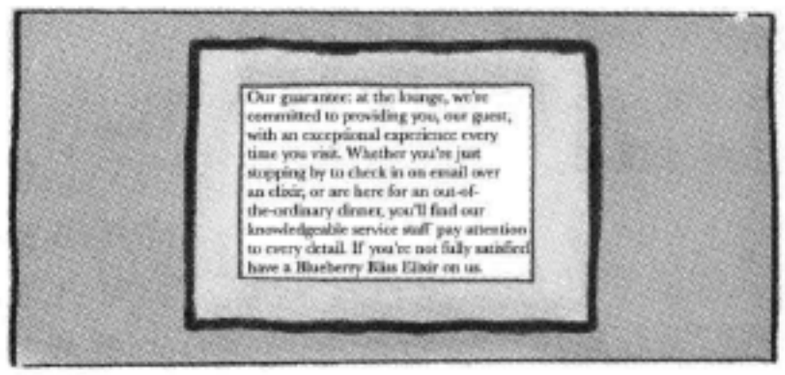
这里有很大的上下内边距。

外边距

对外边距也可以有同样的控制。这里有很大的上下外边距。

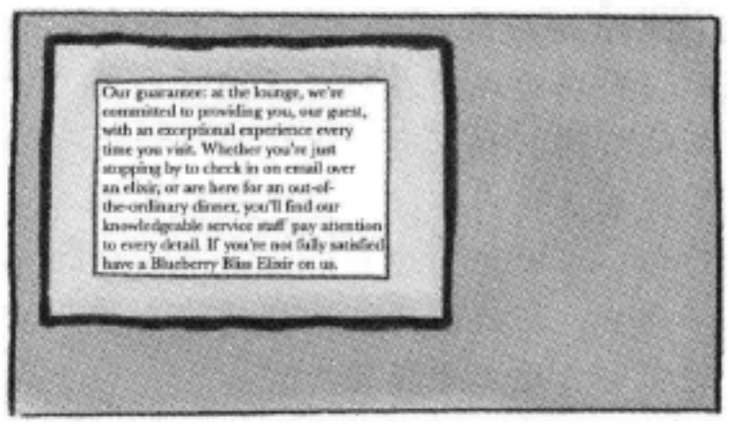


这里有很大的左右外边距。



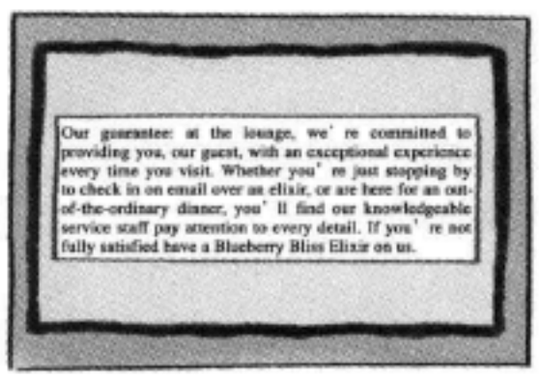
这里利用上边和左边的内边距将内容移到右下角。

与内边距一样，可以独立地指定四边的外边距，就像这样。



内容区

甚至可以采用多种方法控制宽度和高度。在这里，内容区很宽。



这里内容区高而窄。

there are no Dumb Questions

问：看起来，如果我想为Web浏览器开发软件，就必须了解关于盒子的这些内容，这好像很重要，不过这对于我创建更棒的Web页面有什么帮助呢？

答：如果想超越采用浏览器默认布局的简单Web页面，你需要能够控制元素在页面上如何摆放，以及它与其他元素的相对位置。为此，需要对每个元素的内边距和外边距的各个方面加以调整。所以，要创建有意思的Web页面设计，你肯定需要了解盒模型。

问：内边距和外边距有什么区别？它们看起来是一样的。

答：外边距提供元素之间的间距，而内边距是在内容周围增加额外的空间。如果有一个边框，内边距就在边框内部，而外边距在边框外部。可以把内边距看作是元素的一部分，而外边距包围你的元素，将它与其他元素隔开。

问：我知道它们都是可选的，不过如果要有边框或外边距，是不是先得有内边距？

答：没有必要，它们都是完全可选的，相互之间没有依赖关系。所以你可以有一个边框而没有内边距，或者可以有一个外边距而没有边框，这些都是可以。

问：我还不太清楚如何摆放元素，另外如何利用外边距来做到。

答：先留着这个问题。尽管通过这一章的学习，你会对外边距与其他元素如何交互有一点认识，不过这个内容将在第11章介绍元素定位时再详细讨论。

问：这么说，除了大小，看来不能对内边距和外边距指定样式，是吗？

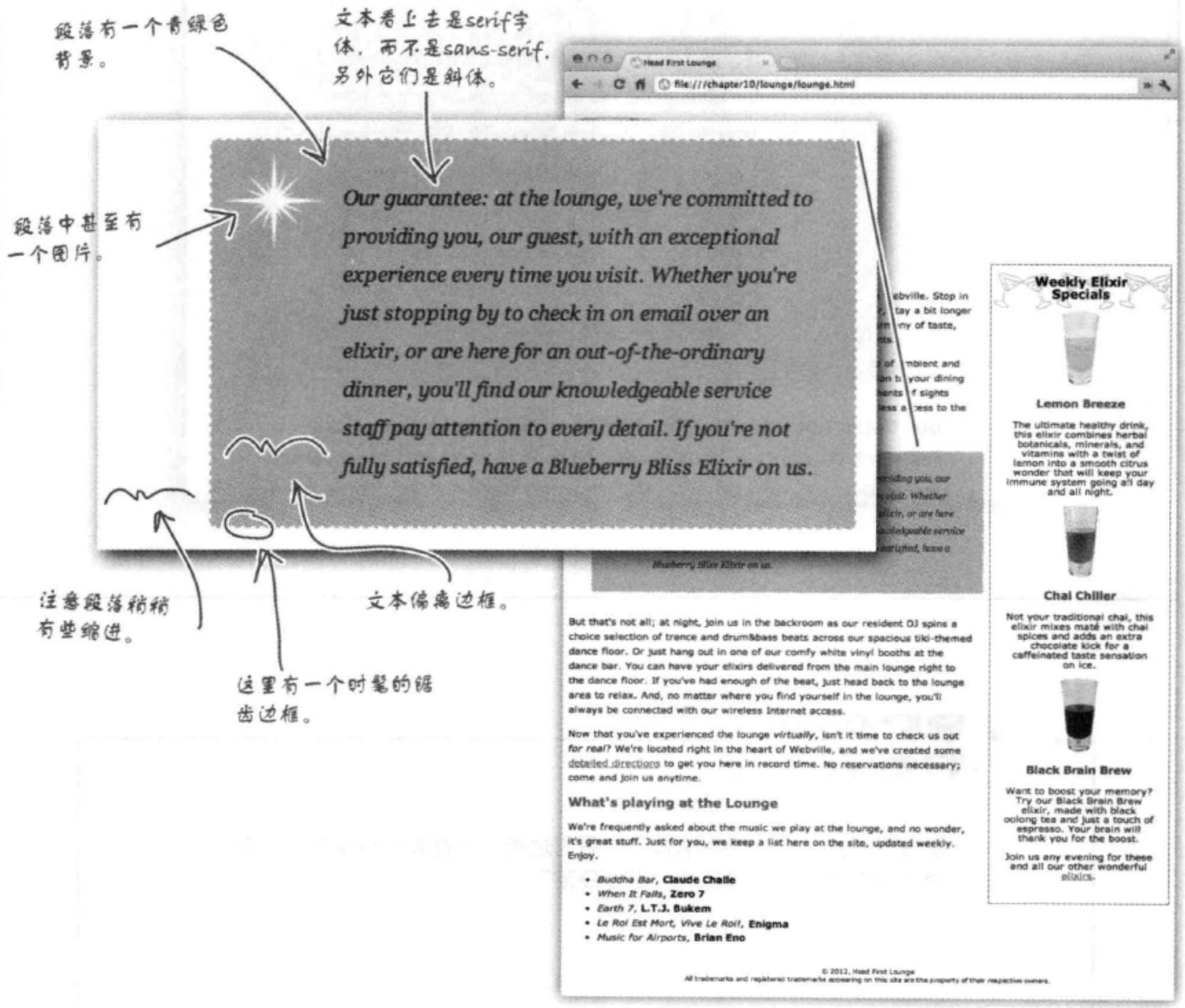
答：基本正确。内边距和外边距都是用来提供更多可见的空间，不能对内边距和外边距指定颜色，也不能加任何装饰。不过，由于它们是透明的，所以他们会呈现背景颜色或背景图像。内边距和外边距之间有一个区别：元素的背景颜色（或背景图像）会延伸到内边距下面，但不会延伸到外边距。稍后你会看到具体是怎样的。

问：内容区的大小是不是仅由其中内容的大小来确定？

答：浏览器会使用多个规则来确定内容区的宽度和高度，稍后我们会更深入地讨论这个问题。不过可以先简单回答这个问题，尽管内容是确定一个元素大小的主要途径，不过如果你需要对元素的大小有所控制，完全可以自行设置宽度和高度。

再回到休闲室……

我们已经做好了准备，来应对休闲室页面的挑战，下面再回到休闲室。在这一章最前面查看休闲室页面时，你有没有注意到那个有特殊样式的蓝色段落？这个段落包含的文本是休闲室对顾客做出的保证，显然他们希望充分强调他们的承诺。下面再仔细查看这个段落，然后着手来构建。



Sharpen your pencil

看看你能不能找出这个段落的内边距、边框和外边距。标出所有内边距和外边距（包括左、右、上和下）。

don't forget, the lounge offers free wireless access to the Internet, so bring your laptop.

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

But that's not all; at night, join us in the backroom as our resident DJ spins a choice selection of trance and drum&bass beats across our spacious tiki-themed dance floor. Or just hang

BRAIN POWER

学习下一页之前，请考虑如何使用内边距、边框和外边距把一个普通的段落变成这个醒目的“保证段落”。

创建guarantee样式

下面先对这个保证段落的样式做一些小改动，来熟悉如何设置段落的盒子。为此，你要把这个段落增加到一个名为guarantee的类中，这样我们就可以只为这个段落创建一些定制样式。然后再增加边框和一个背景颜色，这样能让你清楚地看到这个段落是一个怎样的盒子。接下来我们再处理其余的样式。你需要完成以下工作：

❶ **打开你的“lounge.html”文件，找到以“Our guarantee”开头的段落。为这个元素增加一个class属性“guarantee”，如下所示：**

增加一个class属性，值为“guarantee”。要记住，利用类，你可以独立于其他段落对这个段落增加样式。

```
<p class="guarantee">
    Our guarantee: at the lounge, we're committed to providing
    you, our guest, with an exceptional experience every time you
    visit. Whether you're just stopping by to check in on email
    over an elixir, or are here for an out-of-the-ordinary dinner,
    you'll find our knowledgeable service staff pay attention to every
    detail. If you're not fully satisfied, have a Blueberry Bliss
    Elixir on us.
</p>
```

❷ **保存你的“lounge.html”文件，再打开“lounge.css”文件。现在要为保证段落增加一个边框和背景颜色。在样式表最下面增加以下CSS，然后保存。**

前3个属性会为guarantee类中的所有元素增加一个边框。到目前为止，这个类只有这一个段落。

```
.guarantee {
    border-color:     black;
    border-width:     1px;
    border-style:     solid;
    background-color: #a7cece;
}
```

我们要建立一个黑色边框……

……1个像素宽……

……而且是实线。

我们还要为这个元素指定一个背景色，这样可以帮助你查看内边距和外边距之间的差别，另外能让这个保证段落看起来更漂亮。

测试段落边框

在浏览器中重新加载这个页面，现在你会看到保证段落有了一个青绿色背景，另外它周围有一个很细的黑色边框。下面来仔细观察……

看起来这个段落在内容周围没有内边距，
文本和边框之间没有间距。

the lounge and adding an extra dimension to your dining experience. The decor surrounds you with the relaxing sentiments of sight[illegible] from eras past. And, don't forget, the lounge offers free wireless access to the Internet, so bring your laptop.

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

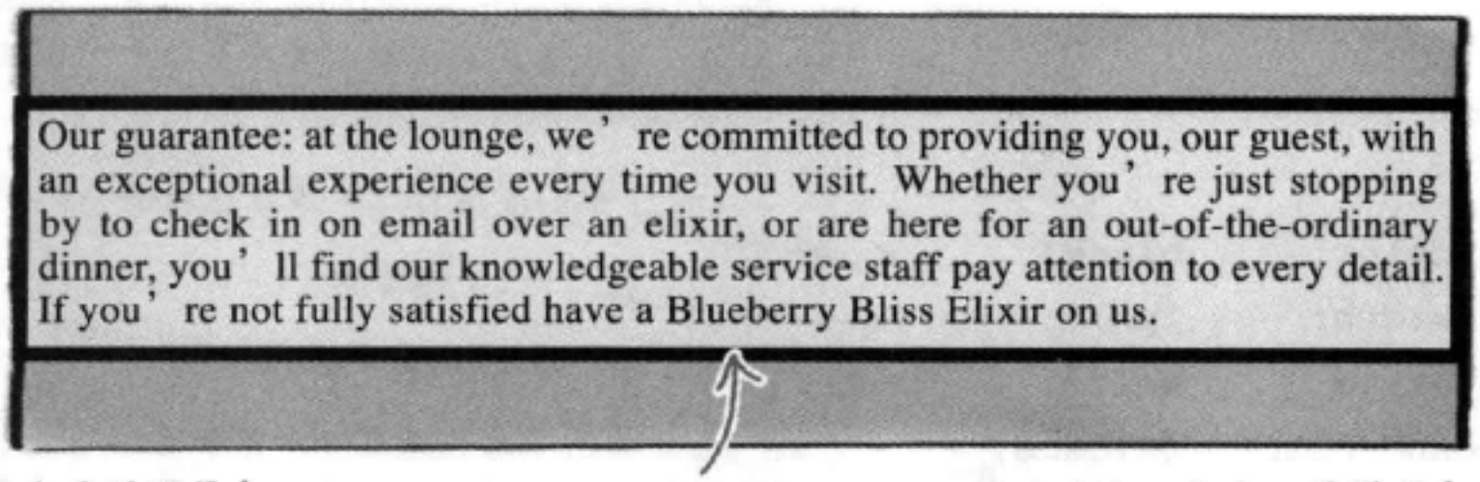
But that's not all; at night, join us in the backroom as our resident DJ spins a choice selection of trance and drum&bass beats across our spacious tiki-themed dance floor. Or just hang out in one of our comfy white vinyl booths at the dance bar. You can have your elixirs delivered from the main

不过看来这个段落元素上面
和下面确实有外边距。

这个段落的左右两边与浏览器窗
口边缘之间没有明显的外边距。

如果把它画成一个盒模型示意图，这个段落如下所示：

上下有外边距。

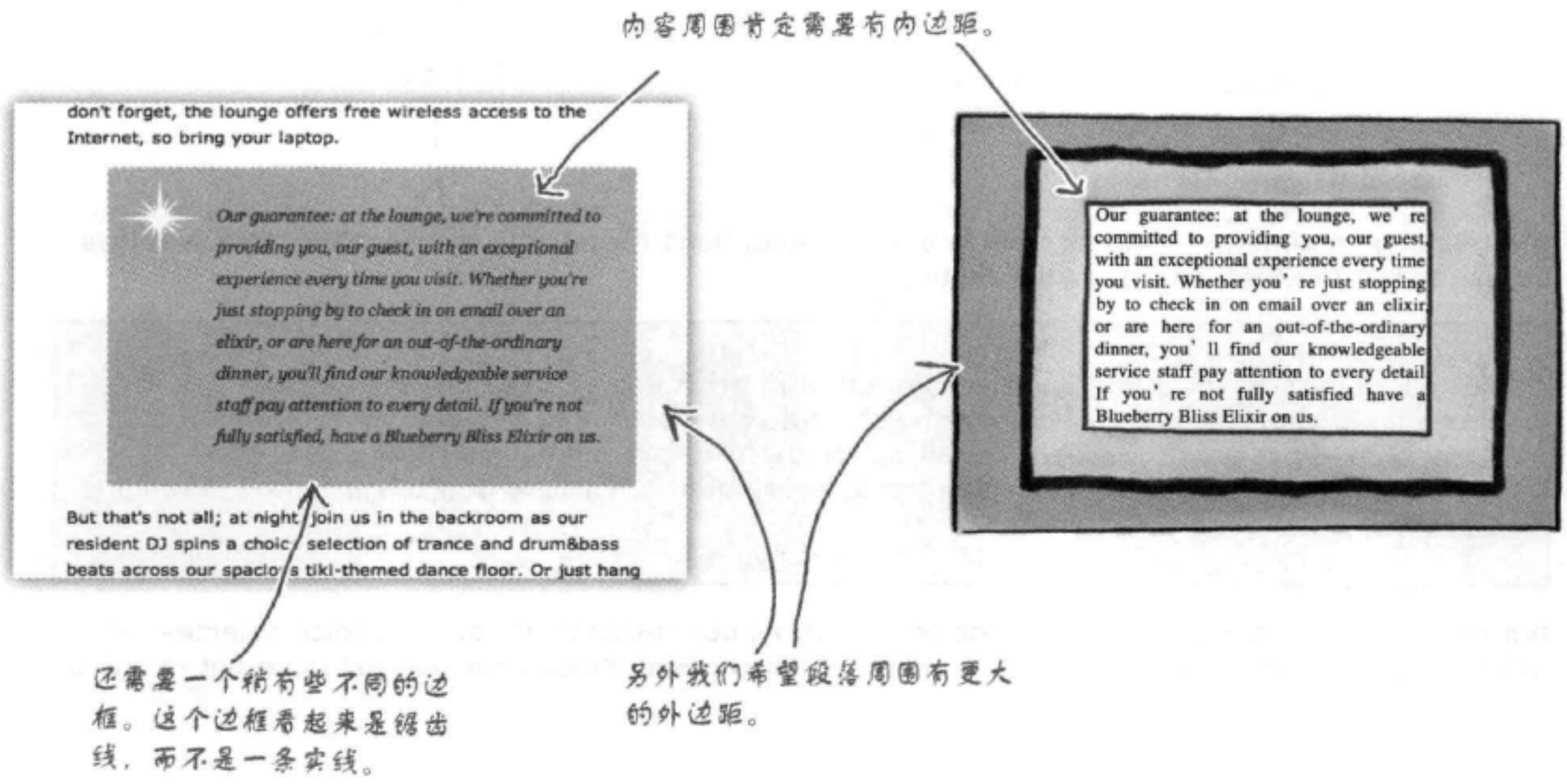


但是左右外边距很小。

这里有边框，不过它直接包围内容，这说明内
边距设置得非常小，或者根本没有内边距。

保证段落的内边距、边框和外边距

既然你已经看到了目前保证段落上的内边距、边框和外边距是如何设置的，下面再来考虑我们真正想要的保证段落是什么样子。



增加一些内边距

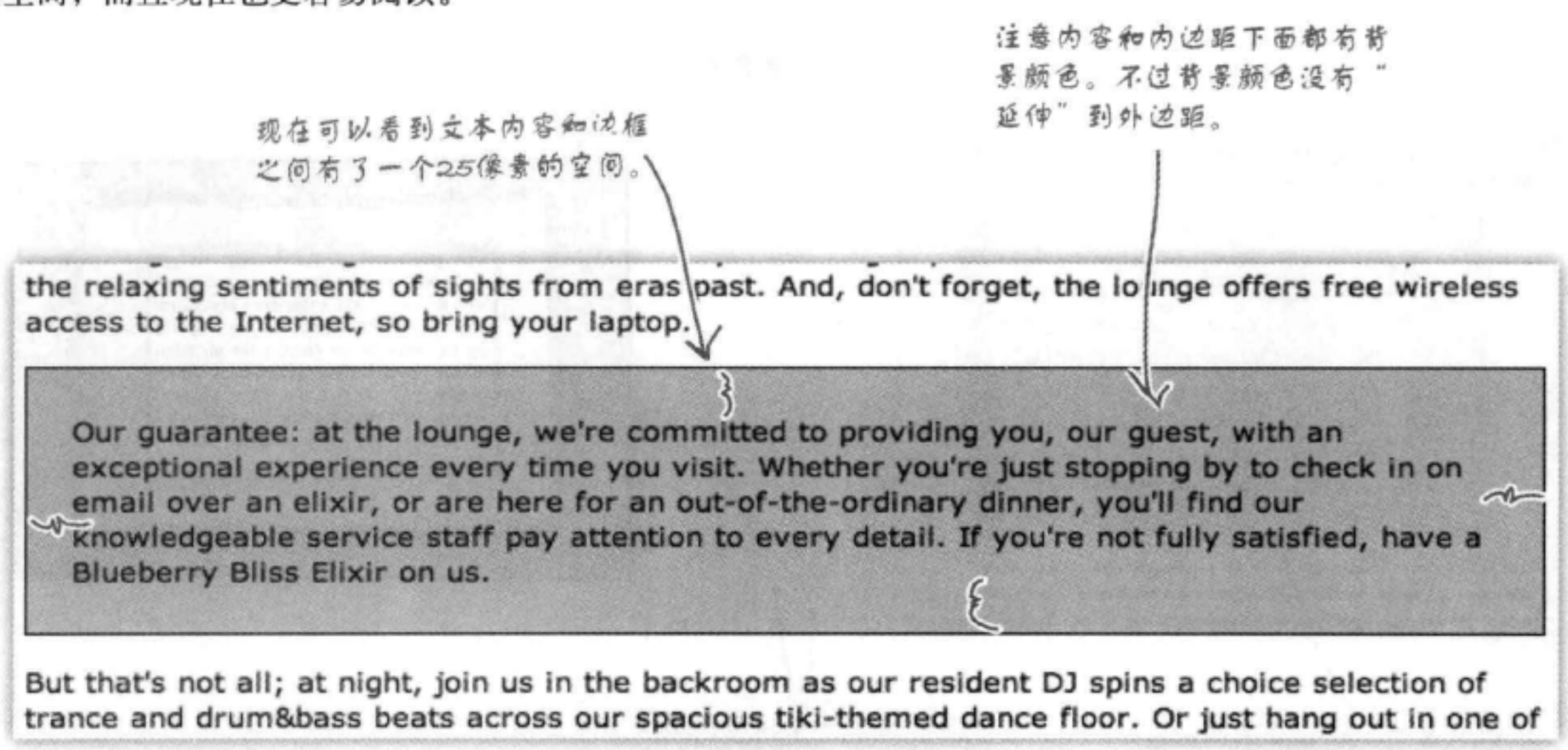
下面先来增加内边距。CSS有一个padding属性，可以用来对内容四周设置相同的内边距。可以将这个属性设置为一个像素数，或者也可以设置为一个百分数。我们将使用像素，把内边距设置为25像素。

```
.guarantee {
      border-color:     black;
      border-width:     1px;
      border-style:     solid;
      background-color: #a7cece;
      padding:          25px;
}
```

在内容的四周（上、右、下和左）增加25像素的内边距。

测试内边距

在浏览器中重新加载这个页面时，你会注意到，现在保证段落中的文本四周多了一点呼吸空间。文本和边框之间有了一些空间，而且现在也更容易阅读。



现在来增加一些外边距

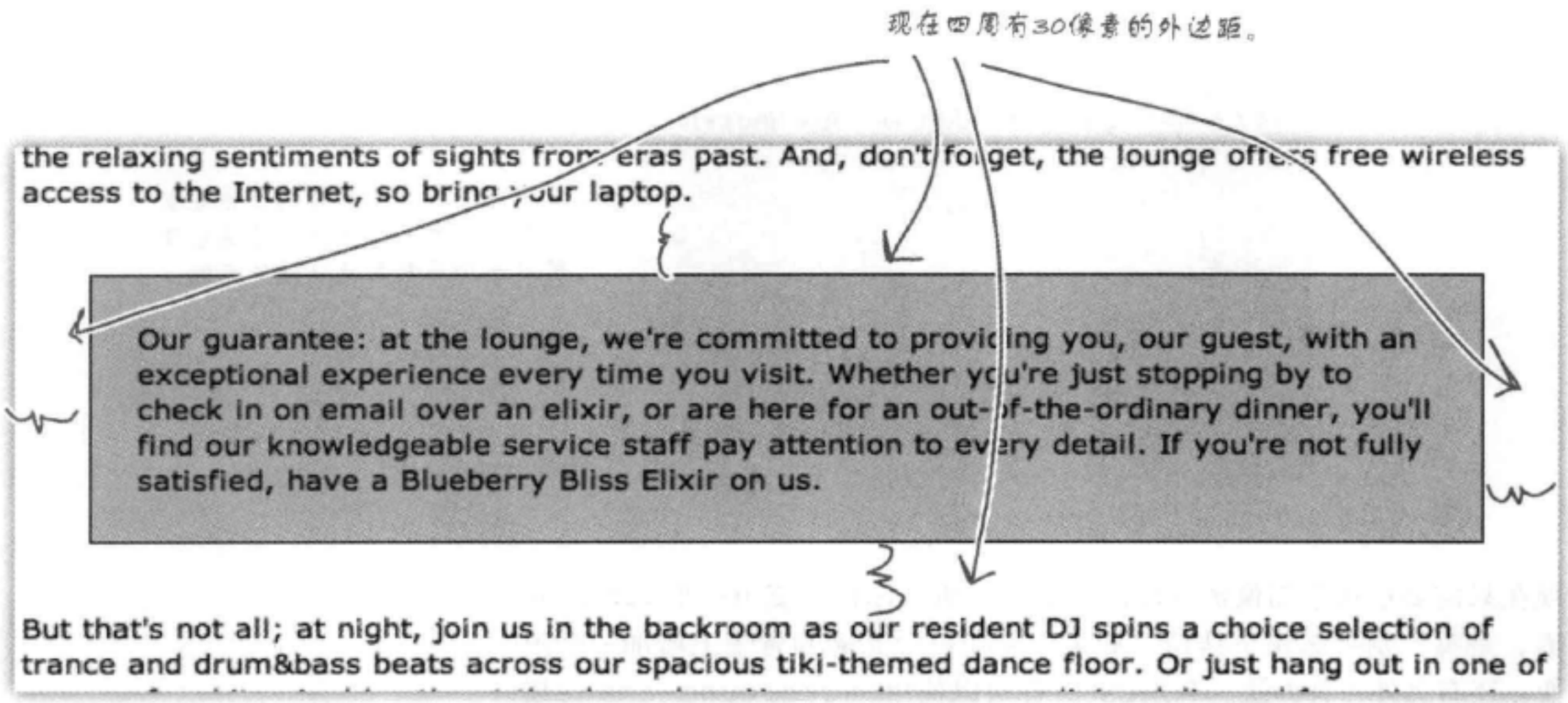
使用CSS很容易增加外边距。类似于内边距，可以将外边距指定为一个百分数或者按像素指定。这里将在整个保证段落周围增加一个30像素的外边距。可以这样做：

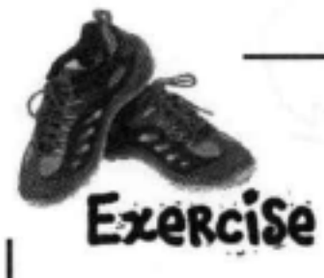
```
.guarantee {
      border-color:     black;
      border-width:     1px;
      border-style:     solid;
      background-color: #a7cece;
      padding:          25px;
      margin:           30px;
}
```

我们要在内容四周（上、右、下和左）增加30像素的外边距。

测试外边距

重新加载休闲室页面时，你会看到，这个段落在页面上确实显得比较突出了。由于有外边距，这个段落就像是插在其余文本中一样，再加上背景颜色，更使它不像一个普通段落，而像是一个“插图”。可以看到，仅仅加了几行CSS，你就取得了如此卓越的成就。





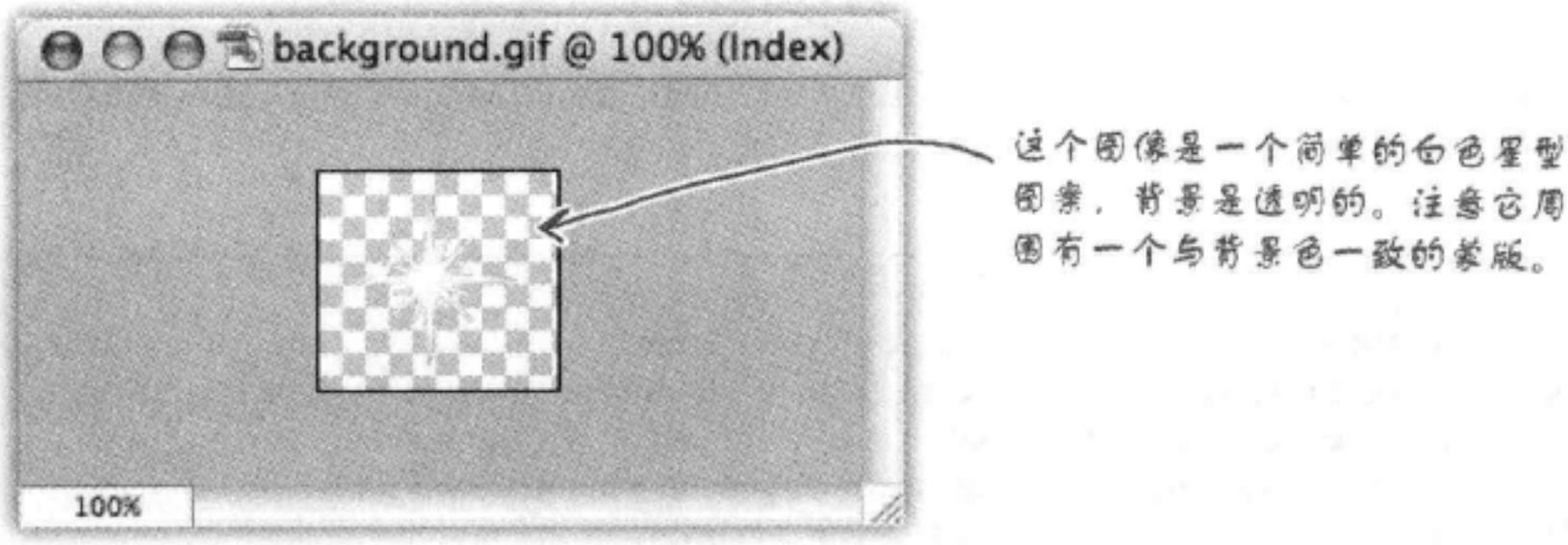
Exercise

如果查看这个保证段落的最终版本，可以看到它有一种斜体serif字体，另外行高大于页面的其余元素（如果再仔细一点），还可以看到文本是灰色的。在下面编写CSS，将行高设置为1.9em，字体风格设置为斜体，字体颜色设置为#444444，字体系列设置为Georgia，“Times New Roman”，Times, serif。对照这一章最后给出的答案检查你写的CSS，然后输入并测试。

增加一个背景图像

就快成功了。还需要做什么？我们还需要在段落中放入一个白色的“保证星”图片。另外还要处理边框，现在还是一个黑色的实线。下面先来处理图像。

如果查看“chapter9/lounge/images”文件夹，可以找到一个名为“background.gif”的GIF图像，如下所示：

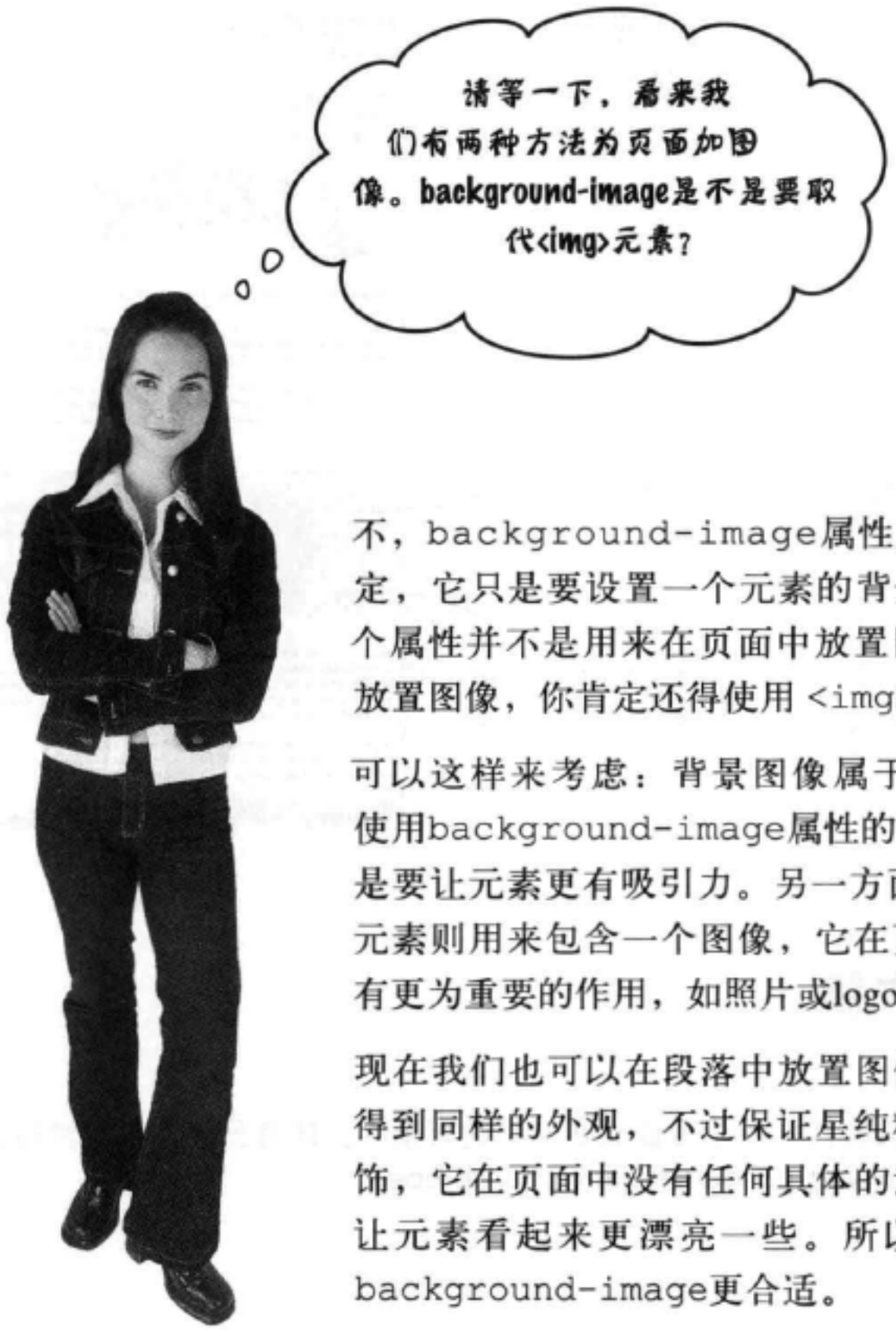


现在只需要把这个图像放入段落元素中，所以你打算使用一个<img>元素，是吗？别那么快下结论。如果你要在一个元素的背景上增加一个图像，还有另外一种办法。利用CSS，你可以使用background-image属性为任何元素增加一个背景图像。下面来试一试，看看能不能奏效：

这些是上一页练习中增加的属性。

```
.guarantee {
      line-height:      1.9em;
      font-style:       italic;
      font-family:      Georgia, "Times New Roman", Times, serif;
      color:            #444444;
      border-color:     black;
      border-width:     1px;
      border-style:     solid;
      background-color: #a7cece;
      padding:          25px;
      margin:           30px;
      background-image: url(images/background.gif);
}
```

把它增加到CSS中，保存并重新加载页面。



不，background-image属性用途非常特定，它只是要设置一个元素的背景图像。这个属性并不是用来在页面中放置图像，要想放置图像，你肯定还得使用 <img>元素。

可以这样来考虑：背景图像属于表现方面，使用background-image属性的唯一理由就是要让元素更有吸引力。另一方面，<img>元素则用来包含一个图像，它在页面中可能有更为重要的作用，如照片或logo。

现在我们也可以在段落中放置图像，可能会得到同样的外观，不过保证星纯粹是一个装饰，它在页面中没有任何具体的意义，只会让元素看起来更漂亮一些。所以这里使用background-image更合适。

测试背景图像

嗯，这的确是一个有意思的测试，我们看到了背景图像，不过它会重复很多次。下面会更深入地研究如何使用CSS背景图像，然后你就能修正这个问题了。

这是背景上的保证星图像。注意它在段落的背景颜色之上，这是因为这个图像有一个透明背景，所以下面的颜色可以透出来。

还要注意，类似于背景颜色，这些背景图像也只出现在内容区和内边距下面，不会在边框以外以及外边距中出现。

CSS放大镜

background-image属性把一个图像放在元素的背景中。还有另外两个属性也会影响背景图像：background-position和background-repeat。

```
background-image: url(images/background.gif);
```

background-image属性设置为一个URL，这可以是一个相对路径，也可以是一个完整的URL(http://…)。

注意URL两边不需要加引号。

修正背景图像

默认地，背景图像会重复。好在background-repeat属性有一个no-repeat值。另外，浏览器还会默认地把背景图像放在元素的左上角，这正是我们希望的，不过下面增加一个background-position属性来试试看。

```
.guarantee {
      line-height:        1.9em;
      font-style:         italic;
      font-family:        Georgia, "Times New Roman", Times, serif;
      color:              #444444;
      border-color:       black;
      border-width:       1px;
      border-style:       solid;
      background-color:   #a7cece;
      padding:            25px;
      margin:             30px;
      background-image:   url(images/background.gif);
      background-repeat:  no-repeat;
      background-position: top left;
}
```

这里要增加两个新属性。

我们希望背景图像不重复。

另外希望它在左上角。

background-position属性会设置图像的位置，可以按像素指定，也可以指定为一个百分数，或者还可以使用关键字指定，如top、left、right、bottom和center。

```
background-position: top left;
```

将这个图像放在元素的左上角。

CSS中有很多不同的方法来指定位置，后面两章我们还会进一步讨论。

默认地，背景图像会“平铺”，也就是反复重复来填满整个背景空间。background-repeat属性可以控制这种平铺行为。

```
background-repeat: repeat;
```

设置图像在水平和垂直方向上重复。这是默认行为。

这是另外几个可用的background-repeat值。

```
no-repeat
repeat-x
repeat-y
inherit
```

图像显示一次，根本不重复。

图像只在水平方向上重复。

图像只在垂直方向上重复。

按父元素的设置来处理。

再来测试背景图像

再来一次吧。这一次看起来我们离目标更近了。不过，由于这是一个背景图像，文本可能会出现在图像上面。怎么修正这个问题呢？这里就要用到内边距了！利用内边距，你可以在内容区周围增加可见的空间。下面增加左边的内边距，看看能不能一次全部搞定。

如何只在左边增加内边距？

对于内边距、外边距甚至边框，CSS在每一个方向（上、右、下和左）都提供了一个属性。要在左边增加内边距，可以使用padding-left属性，如下所示：

```
.guarantee {
    line-height:        1.9em;
    font-style:         italic;
    font-family:        Georgia, "Times New Roman", Times, serif;
    color:              #444444;
    border-color:       black;
    border-width:       1px;
    border-style:       solid;
    background-color:   #a7cece;
    padding:            25px;
    padding-left:       80px;
    margin:             30px;
    background-image:   url(images/background.gif);
    background-repeat:  no-repeat;
    background-position: top left;
}
```

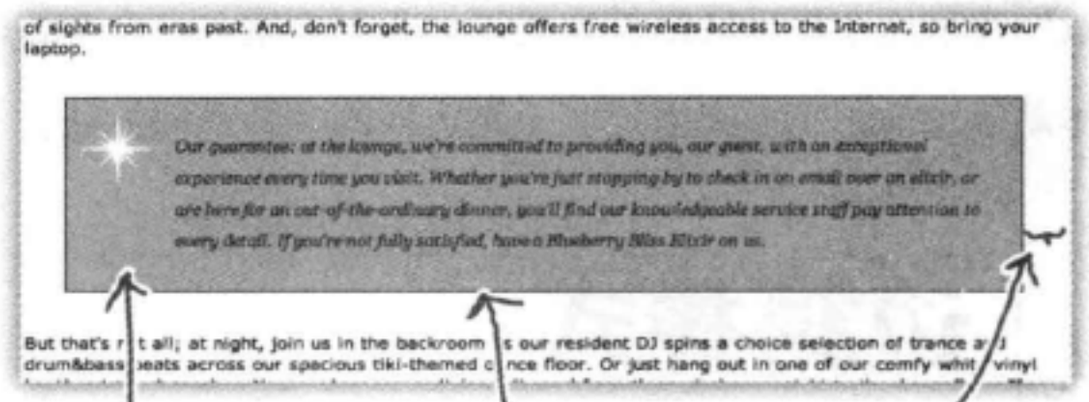
这里使用padding-left属性在左边增加内边距……

注意我们首先将四周的内边距设置为25像素，然后为左边指定一个padding-left属性。

这里顺序很重要。如果交换了顺序，就会先设置左边的内边距，然后才将四周的通用padding属性设置回25像素，这也包括左边（这样一来，左边的内边距就被覆盖了）！

成功了吗？

保存所做的修改，重新加载页面。你会看到段落左边有更大的内边距，现在文本与保证星分开放置，不再出现在保证星上面。有些情况下应该使用内边距而不是外边距，这就是一个很好的例子。如果需要在内容区本身周围有更大的可见空间，就要使用内边距，另一方面，如果希望元素与页面边缘之间有更大空间，这种情况下就要使用外边距。实际上，我们可以在右边使用更大的外边距，让这个段落更突出。下面就来做这个处理，完成这个工作之后就只需要修正边框了。



如何只在右边增加外边距？

就像处理内边距一样，可以增加另一个属性margin-right，来增加右外边距。

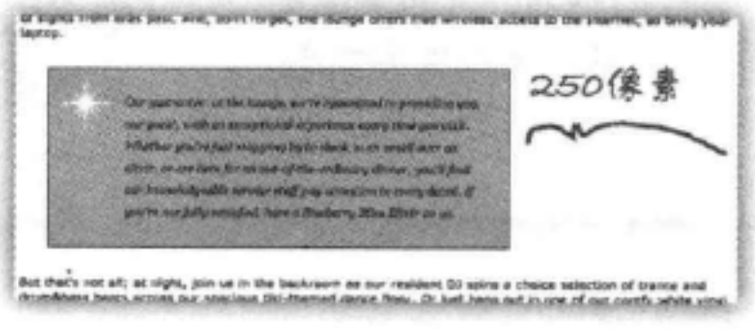
看出规律了吧？首先有一个属性来控制所有4个边，另外如果你想单独对每个边进行设置，实际上对于每个边还分别有一个属性。

```
.guarantee {
    line-height:        1.9em;
    font-style:         italic;
    font-family:        Georgia, "Times New Roman", Times, serif;
    color:              #444444;
    border-color:       black;
    border-width:       1px;
    border-style:       solid;
    background-color:   #a7cece;
    padding:            25px;
    padding-left:       80px;
    margin:             30px;
    margin-right:       250px;
    background-image:   url(images/background.gif);
    background-repeat:  no-repeat;
    background-position: top left;
}
```

要记住，我们已经将外边距设置为30像素。

现在我们要覆盖右外边距的设置，将它设置为250像素。

增加新的margin-right属性，并重新加载页面。现在段落右边会有250像素的外边距。



边框简明指南

再做一件事这个保证段落就完美了，我们要增加一个更好的边框。

具体动手之前，下面来看控制元素边框的各种不同方法。

边框样式

border-style属性可以控制边框的视觉样式。共有8种可用的边框样式，包括实线、虚线、脊线和槽线。

```
border-style: groove;
```

要指定一个边框样式，只需要使用border-style属性，并提供以下某个样式值。

solid样式（实线）就是一个实线边框。

用solid吧，这是最经典的。

dotted样式（虚线，也称为点线）看起来像一系列小点。

一旦拥有dotted，别无所求。

dashed样式（破折线）就是围绕边框的一组破折线。

忘了dotted吧，请使用dashed。

double样式（双线）有两条线。

用double吧，我会带来双倍快乐。

groove样式（槽线）看起来就像页面中的一个槽（不过在书上很难看出来）。

我采用的就是groove边框。

inset样式（内凹）看起来向页面凹进去。

我是唯一的“内凹”样式：inset。

outset样式（外凸）看起来像是从页面凸出来一样。

用我（outset）吧，我从一开始就更胜一筹。

ridge样式（脊线）看起来像页面上一个凸起的山脊。

我更有意思，我有ridge。

边框宽度

border-width属性控制边框的宽度。可以使用关键字或像素来指定边框宽度。

```
border-width: thin;
border-width: 5px;
```

可以使用关键字thin, medium或thick指定边框宽度，或者用像素数指定。

thin
medium
thick

1px
2px
3px
4px
5px
6px

边框颜色

border-color属性设置边框的颜色。这与设置字体颜色类似，可以使用颜色名、rgb值或十六进制码来指定颜色。

```
border-color: red;
border-color: rgb(100%, 0%, 0%);
border-color: #ff0000;
```

使用border-color指定边框的颜色。可以使用任意一种常用方法来指定颜色。

指定某一边的边框

border-top-color
border-top-style
border-top-width

border-right-color
border-right-style
border-right-width

border-bottom-color
border-bottom-style
border-bottom-width

border-left-color
border-left-style
border-left-width

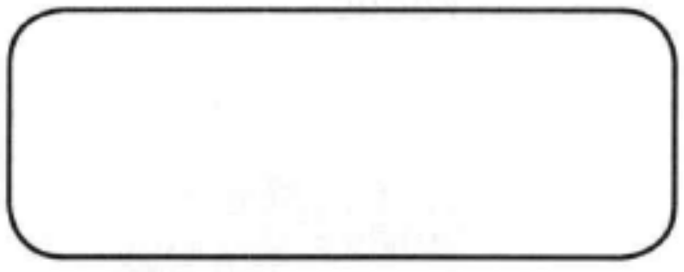
就像外边距和内边距一样，还可以指定任意一边（上、右、下和左）的边框样式、宽度和颜色。

```
border-top-color: black;
border-top-style: dashed;
border-top-width: thick;
```

这些属性只针对上边框。可以独立地指定任意一边的边框。

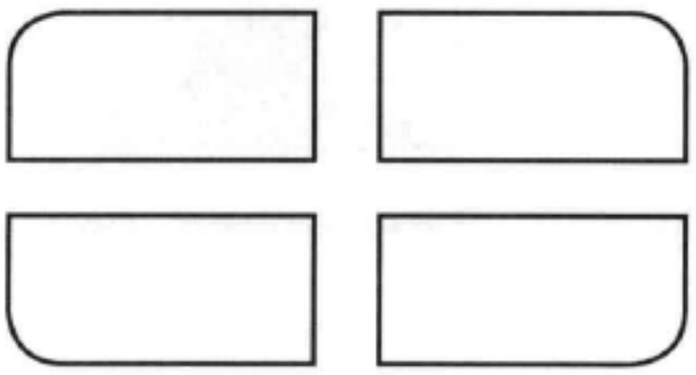
指定边框圆角

可以在四个角上都创建圆角，或者只对一个角或这4个角的任意组合创建圆角。



可以用一个数指定4个角。

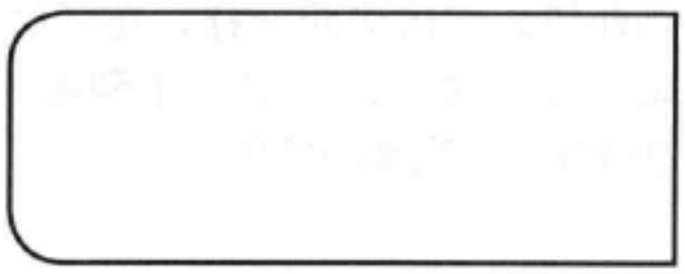
```
border-radius: 15px;
```



或者，可以分别指定每一个角。注意你可以使用px或em来指定半径大小。

```
border-top-left-radius: 3em;
border-top-right-radius: 3em;
border-bottom-right-radius: 3em;
border-bottom-left-radius: 3em;
```

如果使用em，边框半径的度量相对于元素的字体大小，与使用em指定font-size时是一样的。



```
border-top-left-radius: 15px;
border-top-right-radius: 0px;
border-bottom-right-radius: 0px;
border-bottom-left-radius: 15px;
```

通过使用border-radius，可以得到各种有趣的形状。

完善边框

我们的保证段落就要完成了。现在我们要做的就是为它提供一个锯齿线边框。不过，这是哪一种样式呢？可用的样式包括solid、double、dotted、dashed、groove、ridge、inset和outset。那么怎么让它看起来像锯齿呢？实际上这里有一个技巧：我们会使用dashed（破折线）边框，把它的颜色设置为白色（与页面背景颜色一致）。可以像下面这样做。首先设置边框为破折线型。在“lounge.css”中找到`border-style`属性，修改如下：

```
border-style: dashed;
```

这里将边框从实线改为虚线。

保存这个文件，应该能看到像这样的一个边框：

现在，要得到有锯齿的边框，只需要将这个边框的颜色设置为白色。这会使边框看起来像是切入背景色一样。可以来试一试：找到`border-color`属性，将它设置为`white`。

```
border-color: white;
```

这里将边框颜色从黑色改为白色。

保存文件，再次重新加载。现在应该能看到这个锯齿边框了。

Watch it!

不同浏览器对thin、medium和thick的具体大小可能有不同定义。

不同的浏览器对于关键字thin、medium和thick可能有不同的默认大小，所以如果边框大小对你来说确实很重要，就应该考虑使用像素大小来指定。

测试漂亮的边框

祝贺你！

棒极了！你让一个平平常常的HTML段落变得漂亮时髦得多，而且仅仅用了15行CSS代码！

真是一条漫长的学习之路，所以现在我们建议你休息一下。来杯冰茶，花一点点时间消化学到的东西，等你回来后，我们将讨论更多CSS重点。

Exercise

在你品味冰茶的同时，别让手闲着，为保证段落增加一个border-radius。下面给出了保证段落的几个例子，它们分别设置了不同的border-radius值。请写出CSS来创建这个例子中看到的边框。我们已经提供了每个例子中创建圆角所用的border-radius值。

30px

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

在这里写出你的CSS。

40px

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

40px

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

2em

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

欢迎回来，你来的正好。我们正要去听对类的采访……

类闪亮登场

本周访谈：
类总是对的吗？

Head First：嘿，类，你知道的，我们已经用你很久了，不过还是对你不太了解。

类：嗯，没有太多需要了解的。如果你想创建一个“组”（先这么叫吧），以便增加样式，就可以使用类，把你的元素放在这个类中，这样你就能为这个类中的所有元素一起指定样式了。

Head First：这么说利用类可以对一组元素应用一个或多个样式属性，是吗？

类：完全正确，假设你的页面中有一些假日主题区，比如一个万圣节区，一个圣诞节区。你可以把所有万圣节元素增加到halloween类，而将所有圣诞节元素增加到christmas类，然后单独地对这些元素指定样式。比如说，万圣节元素用橙色，圣诞节元素用红色，当然要编写分别应用到各个类的规则。

Head First：很有道理。我们在这一章中刚刚看过一个很好的例子，不是吗？

类：我不清楚；那时我刚好不在，你得提醒我一下。

Head First：嗯，Head First休闲室页面上有一个段落，其中包含休闲室做出的保证，他们希望这个段落能从其他段落中“脱颖而出”。

类：听起来还不错……不过我得问你一个问题：是有好几个段落还是只有一个？

Head First：嗯，我觉得作为保证来讲，没有理由长篇大论写好几段吧，另外我没有看到页面中其他内容也应用这种样式，所以应该只有一个段落。

类：嗯，我可不希望这样。要知道，如果你想对多个元素重用某些样式，才能真正发挥类的作用。如果只有一个元素需要指定样式，这种情况下使用类就有些大材小用了。

Head First：等一下，看起来类做得很好……怎么会不对呢？

类：呵，别害怕。你要做的只是把class属性换成一个id属性。不用一分钟就能搞定。

Head First：id属性？我以为它们只用于那些链接目标呢，就像第4章的那些链接，不是吗？

类：id有很多用途。它们是元素的唯一标识符。

Head First：你能再给我多讲讲id属性吗？这对我来说都是新内容。我的意思是说，整个一章里我都只是在错误地使用class！

类：嘿，别担心，这是一个常犯的错误。实际上，你只需要知道，想要对多个元素使用某个样式时，就要用到class。但是如果只有一个元素需要加样式，或者页面上只有一个元素，那就应该使用id。id属性恰恰就是用来唯一地命名元素。

Head First：OK，我觉得我明白了，不过这很重要吗？我是说，用class好像也没问题。

类：因为有些东西你可能希望页面上只能有一个。你提到的保证段落就是一个这样的例子，不过还有更好的例子，比如页面上的页眉或页脚，或者导航条。这些东西在页面上不会有两个。当然，即使只有一个元素，你也可以对它使用类，不过别人可能会向这个类增加另一个元素，这样一来，你的元素就没有唯一的样式了。另外，如果你要指定HTML元素的位置（这个内容我们还没有讨论），这也很重要。

Head First：嗯，好吧，类。这次谈话确实让我受益非浅。看来我们确实需要把那个段落从class改为id。再次感谢你能接受采访。

类：随时愿意效劳，Head First！

对下面的元素选择使用class还是id：

id	class		id	class	
☐	☐	包含页面页脚的一个段落。	☐	☐	包含电影评论的一组<p>元素。
☐	☐	包含公司简介的一组标题和段落。	☐	☐	包含任务列表的一个 元素。
☐	☐	包含“每日图片”的<img>元素。	☐	☐	包含Buckaroo Banzai引用的一些<q>元素。

答案：页脚、每日图片和任务列表最好使用id。

id属性

因为你已经在<a>元素上用过id，另外已经知道如何使用class属性，所以使用id属性并不需要再学习多少新知识。假设你的页面上有一个页脚。所有页面都只有一个页脚，所以听上去这很适合使用id。可以为包含页脚文本的段落增加标识符footer，如下所示：

为元素指定id与将元素增加到一个类很类似。唯一的区别是，这个属性名为id，而不是class。一个元素不能有多个id，另外页面上不允许多个元素都有相同的id。

there are no Dumb Questions

问：这很重要吗？为什么仅仅为了证明在页面上的唯一性就需要一个id？使用类也完全可以做到这一点，不是吗？

答：嗯，你当然可以用类来“模拟”一个唯一的id，不过最好不要这么做，原因有很多。假设你在与一个团队共同开发一个Web项目。某个团队成员看到一个类时，他会认为其他元素可以重用这个类。另一方面，如果他看到的是一个id，就会知道这对应一个唯一的元素。id之所以很重要还有另外一些原因，不过后面几章才会具体解释这方面的内容。例如，要在一个页面上指定元素位置，你就需要每个要定位的元素都有唯一的id。

问：可不可以这样：一个元素有一个id，同时属于一个类？

答：当然可以。可以这样来考虑：id只是一个元素的唯一标识符，不过这并不妨碍这个元素属于一个或多个类（就像你有一个唯一的名字，但你完全可以加入一个或多个俱乐部）。

不过如何在CSS中使用id呢？

用id选择一个元素与用类选择一个元素基本上是一样的。可以简单复习一下：如果有一个名为specials的类，选择使用这个类的元素有多种方法。可以只选择这个类中的某些元素，如下所示：

```
p.specials {
        color: red;
}
```

这只会选择specials类中的段落。

或者，也可以选择属于specials类的所有元素，如下所示：

```
.specials {
        color: red;
}
```

这会选择specials类中的所有元素。

使用id选择器是非常相似的。要按id来选择一个元素，需要在id前面使用一个#字符（可以与类做个比较，按类选择时，需要在类名前使用一个点号[.]）。假设你想选择id为footer的任意元素：

```
#footer {
        color: red;
}
```

这会选择id为"footer"的任意元素。

或者可以只选择id为footer的一个<p>元素，如下所示：

```
p#footer {
        color: red;
}
```

这会选择一个id为"footer"的<p>元素。

class和id还有一点差别：id选择器只与页面中的一个元素匹配。

休闲室页面中使用id

我们的保证段落确实应该有一个id，因为页面中这个保证段落只用到一次。尽管我们本该从一开始就这样设计，不过现在改起来也很简单。

第1步：将“lounge.html”文件中的class属性改为id。

只是将class属性改为id。

```
<p id="guarantee">
    Our guarantee: at the lounge, we're committed to providing
    you, our guest, with an exceptional experience every time you
    visit. Whether you're just stopping by to check in on email
    over an elixir, or are here for an out-of-the-ordinary dinner,
    you'll find our knowledgeable service staff pay attention to every
    detail. If you're not fully satisfied, have a Blueberry Bliss Elixir
    on us.
</p>
```

第2步：将“lounge.css”中的“.guarantee”类选择器改为一个id选择器。

只是将选择器的“.”改为“#”。

```
#guarantee {
      line-height:        1.9em;
      font-style:         italic;
      font-family:        Georgia, "Times New Roman", Times, serif;
      color:              #444444;
      border-color:       white;
      border-width:       1px;
      border-style:       dashed;
      background-color:   #a7cece;
      padding:            25px;
      padding-left:       80px;
      margin:             30px;
      margin-right:       250px;
      background-image:   url(images/background.gif);
      background-repeat:  no-repeat;
      background-position: top left;
}
```

第3步：保存所做的修改，重新加载页面。

嗯，看起来并没有不同。不过现在你是不是感觉好多了？原本就应该这样。

there are no Dumb Questions

问：为什么选择器要写为#guarantee而不是p#guarantee？

答：这两个都可以，它们会选择同一个元素。在这个页面上，我们知道肯定会有一个段落指定为这个id，所以用哪一个选择器都是可以的（#guarantee要简单一些）。不过，在一组更复杂的页面中，可能会出现这种情况：有些页面将这个唯一的id指定给一个段落，而在另外一些页面上可能会把这个id分配给一个列表或块引用。所以你可能需要为这个id定义多个规则，根据页面上不同类型的元素应用不同的规则，如p#someid和blockquote#someid。

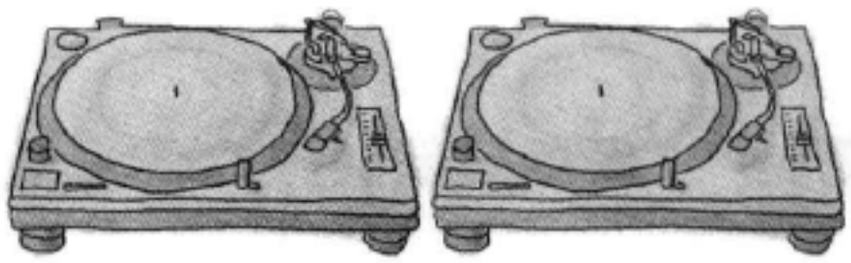
问：是不是总得先从类开始，然后如果知道这个元素是唯一的，再改成id？

答：不用，设计页面时你通常都会知道一个元素是不是唯一的。这一章之所以这么做，只是因为之前你还不懂id。不过你不觉得吗？我们不仅很好地引入了id，而且这个故事编得还很完满！

问：类和id名有什么规则？

答：类名要以一个字母开头，不过id名可以以一个数字或字母开头。id和类名都可以包含字母、数字以及_字符，但不能有空格。所以“number1”是可以的，“main_content”也可以，但“header content”不行。一定要记住，id必须是唯一的！

混合样式表

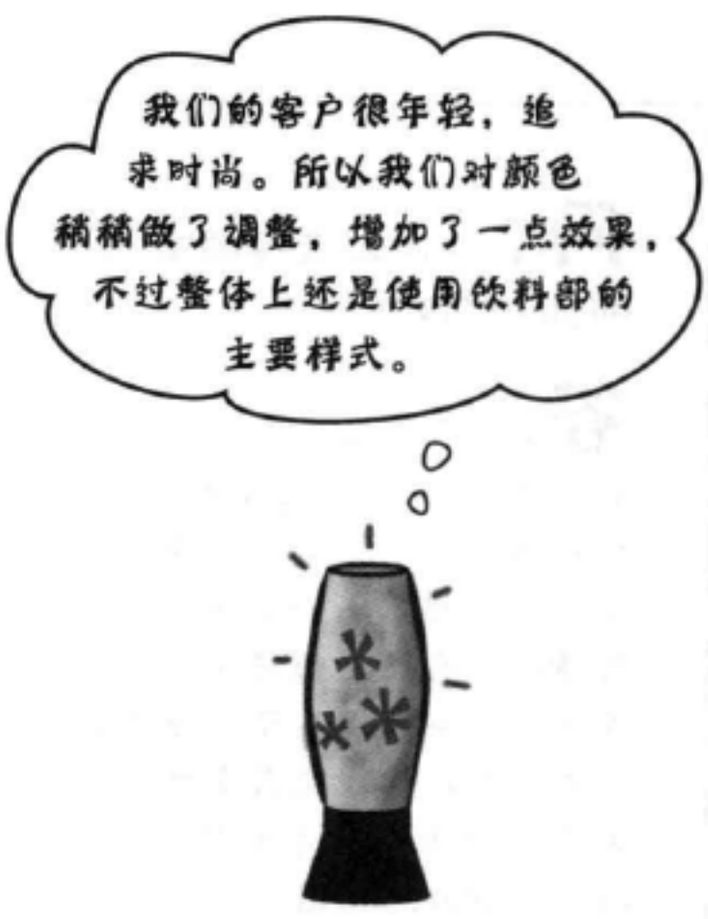


结束这一章之前，下面做个有意思的事情，我们要混合一些样式表。到目前为止，你一直都只是使用一个样式表。嗯，有人说过不能用多个样式表吗？你完全可以指定一组样式表用于任何HTML。不过，你可能想知道为什么会有人希望这么做。这有很多充分的理由。下面先说第一个原因……

假设Head First休闲室开张了，有了特许经营权，完成了首次公开募股等（这些都要归功于你，当然还要归功于你非凡的Web工作）。这将是一个庞大的公司网站，有数百个页面，显然你希望利用外部CSS样式表对这些页面增加样式。由于公司有很多分部，这些分部希望以自己的方式对样式有所调整。另外休闲室经营者们也希望对样式有一些控制。可能会是这样：

总公司

饮料部



西雅图休闲室
（饮料部的一部分）

使用多个样式表

你想先从总公司样式开始，然后允许分部和休闲室经营者们覆盖和修改这些样式，如何做到呢？可以使用多个样式表，如下所示：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge</title>
    <link type="text/css" href="corporate.css" rel="stylesheet">
    <link type="text/css" href="beverage-division.css" rel="stylesheet">
    <link type="text/css" href="lounge-seattle.css" rel="stylesheet">
  </head>
  <body>
  .
  .
  .
  </body>
</html>
```

在你的HTML中，可以指定多个样式表。这里我们有3个样式表。

整个公司有一个样式表。

饮料部可以对总公司样式做一些补充，甚至可以覆盖公司的一些样式。

西雅图休闲室又对样式表做了自己的调整。

顺序很重要！一个样式表会覆盖在它上面链接的样式表中的样式。

there are no Dumb Questions

问：这么说样式表的顺序很重要，是吗？

答：没错，这些样式表从上到下排列，最下面的样式表最优先。所以，假设总公司和饮料部的样式表中<body>元素都有一个font-family属性，那么饮料部的样式优先，因为它在HTML中最后链接。

问：简单的网站也需要这样吗？

答：你可能会惊奇地发现确实如此。有时会有一个样式表作为页面的基础样式。要修改样式，并不是修改这个样式表，而是链接这个样式表，然后在它下面提供你自己的样式表，指定你想修改的样式。

问：你能再多说说吗？如何确定一个特定元素的样式？

答：我们在第7章简单讨论过，现在可以再补充一点，文件中链接样式表的顺序很重要。在下一章中，等你学完CSS的另外一些细节后，我们就会介绍浏览器如何知道哪个元素要应用哪个样式。

样式表，不再只面向浏览器……

之所以希望有多个样式文件，实际上还有一个原因：你可能想针对将要显示页面的设备类型（桌面PC、笔记本电脑、平板电脑、手机或者甚至页面的印刷版本）来调整页面的样式。嗯，要做到这一点，可以利用一个media属性，在<link>元素中增加这个属性，只使用适用于指定设备的样式文件。下面来看一个例子：

media属性允许你指定应用这个样式表的设备类型。

通过创建一个“媒体查询”来指定设备类型，媒体查询要与设备匹配。

```
<link href="lounge-mobile.css" rel="stylesheet" media="screen and (max-device-width: 480px)">
```

这个查询指定了有屏幕的设备（而不是其他设备，比如说打印机或3D眼镜，或者盲文阅读机）……

……而且屏幕宽度不超过480像素。

类似的，我们可以创建一个查询来匹配打印机设备，如下所示：

```
<link href="lounge-print.css" rel="stylesheet" media="print">
```

lounge-print.css文件只有当……

……媒体类型为“print”时才会使用，这说明我们要通过打印机查看页面。

查询中还有很多属性可以使用，如min-device-width、max-device-width（刚刚用过），以及显示方向[orientation，这可以是横向（landscape）或纵向（portrait）]，此外还有很多其他的属性。要记住，可以根据需要为HTML增加多个<link>标记，涵盖你要支持的所有设备。

直接在CSS中增加媒体查询

要为CSS指定有特定属性的设备，还有一种方法：不是在link标记中使用媒体查询，还可以直接写在CSS中。下面给出一个例子：

使用@media规则……

……后面是你的媒体查询。

对于与这个查询匹配的设备，将所有适用的规则放在大括号里。

所以，如果设备屏幕宽度大于480px就会使用这些规则……

……如果设备屏幕宽度小于等于480px，就会使用这些规则……

……如果要打印这个页面，就会使用这些规则。

所有其他规则会应用于所有页面，因为它们并未包含在一个@media规则中。

```
@media screen and (min-device-width: 481px) {
    #guarantee {
        margin-right: 250px;
    }
}
@media screen and (max-device-width: 480px) {
    #guarantee {
        margin-right: 30px;
    }
}
@media print {
    body {
        font-family: Times, "Times New Roman", serif;
    }
}

p.specials {
        color: red;
}
```

采用这种方式，@media规则中只包含特定于一种媒体类型的CSS规则。在CSS文件中，要把对所有媒体类型都通用的规则放在@media规则下面，这样一来，就不会不必要地重复规则了。另外，浏览器加载页面时，它会通过媒体类型来确定页面适用的规则，而将不匹配的规则忽略。

媒体查询是目前标准组织在积极发展的一个领域，所以要密切关注指定设备的最佳实践，这方面还在不断演进发展。

Watch it!

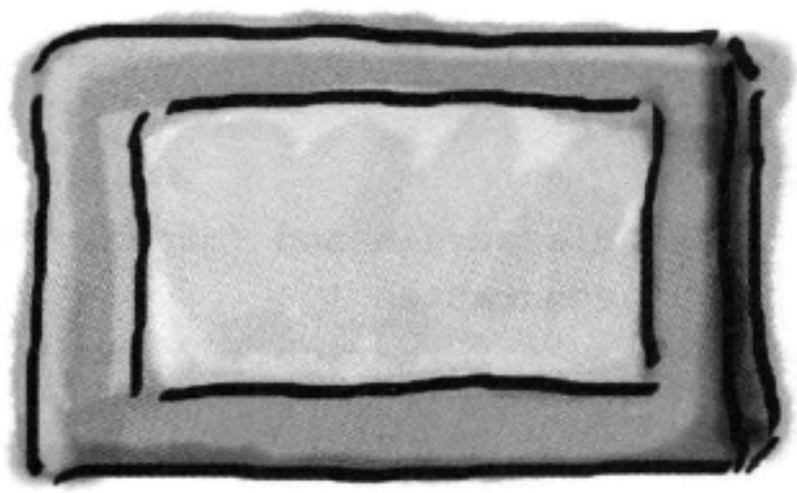
IE8及以前版本不支持媒体查询。

请查看下面的设备以及相应的规格。你能设计一组媒体查询指定以下各个设备吗？

智能手机：480 × 640像素

平板电脑，横向或纵向：1024 × 768像素

桌面PC: 1280 × 960像素



互联网电视: 2650 × 1600像素，横向

```
<link rel="stylesheet" href="lounge-smartphone.css"
      media="                                      ">
<link rel="stylesheet" href="lounge-tablet-portrait.css"
      media="                                      ">
<link rel="stylesheet" href="lounge-tablet-landscape.css"
      media="                                      ">
<link rel="stylesheet" href="lounge-pc.css"
      media="                                      ">
<link rel="stylesheet" href="lounge-tv.css"
      media="                                      ">
```

你的答案写在这里！

there are no Dumb Questions

问：真是太酷了。这样我就能为不同的设备建立不同的样式表，是吗？

答：是的，你可以建立多个样式表，然后在你的HTML中链接这些样式表。要确定使用哪个样式表，这个工作可以交给浏览器来完成，它会根据媒体类型和你在媒体查询中指定的特征选择使用合适的样式表。

问：除了max-device-width和min-device-width，还有其他媒体属性吗？

答：是的，还有很多，包括最大和最小宽度（与device-width不同，稍后就会看到），最大和最小高度、方向、颜色、宽高比等。可以查看CSS3媒体查询规范来了解所有细节（http://www.w3.org/TR/css3-mediaqueries/），另外也可以参考《Head First Mobile Web》中给出的例子。

问：要为不同的媒体类型和特征指定不同的CSS规则，使用<link>还是@media，哪一个更好？

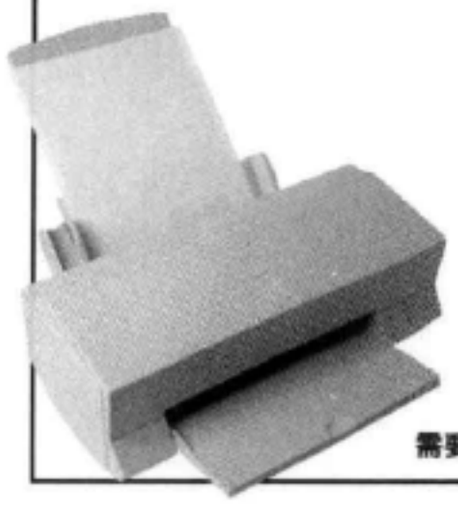
答：这两个都行。不过要注意，如果把所有规则都放在一个文件中，再使用@media规则将它们分开，你的CSS会变得非常庞大。通过为不同的媒体类型使用不同的<link>元素，就能按照媒体类型在不同文件中组织CSS。所以，如果你的CSS文件相当庞大，我们就建议使用<link>元素来指定不同的样式表。

在你的“chapter9/lounge”文件夹中，可以找到一个“lounge-print.css”文件。打开“chapter9/lounge”文件夹中的“lounge.html”，为媒体类型“print”增加一个新链接，指向这个样式表。还要为链接到“lounge.css”的<link>元素增加属性media=“screen”，这样就有了两个样式表，一个针对屏幕，另一个面向打印机。接下来保存文件，重新加载页面，并选择浏览器的Print选项。打开打印机看看结果！

```
<link   type="text/css"   href="lounge-print.css"
      rel="stylesheet" media="print">
```

这是“lounge.html”文件中要增加的新链接。

这是打印版本。利用CSS，打印时页面的外观完全改变了。区分结构和表现确实真是很值得。



需要激光打印机，不随书赠送。

对不同屏幕宽度测试不同CSS文件

max-device-width和min-device-width媒体属性依赖于设备的实际屏幕（而不是你的浏览器窗口宽度）。如果你更关心浏览器大小呢？嗯，可以使用max-width和min-width属性，它们表示浏览器窗口本身的最大和最小宽度（而不是屏幕大小）。下面来看这是如何工作的：在你的“chapter9/lounge”文件夹中，可以找到一个文件“lounge-mobile.css”。再次打开你的lounge.html文件，修改文档<head>中的<link>元素，如下所示：

```
<link type="text/css" rel="stylesheet" href="lounge.css"
      media="screen and (min-width: 481px)">
<link type="text/css" href="lounge-mobile.css" rel="stylesheet"
      media="screen and (max-width: 480px)">
<link type="text/css" href="lounge-print.css" rel="stylesheet" media="print">
```

现在请在你的浏览器中重新加载“lounge.html”页面。确保浏览器窗口相当大。你会看到正常的休闲室页面。

接下来，让浏览器窗口变窄（宽度小于480像素）。休闲室页面会怎么样呢？注意到差别了吗？请在下面描述将浏览器窗口变窄并加载页面时的结果。为什么这个版本更适合移动浏览器？

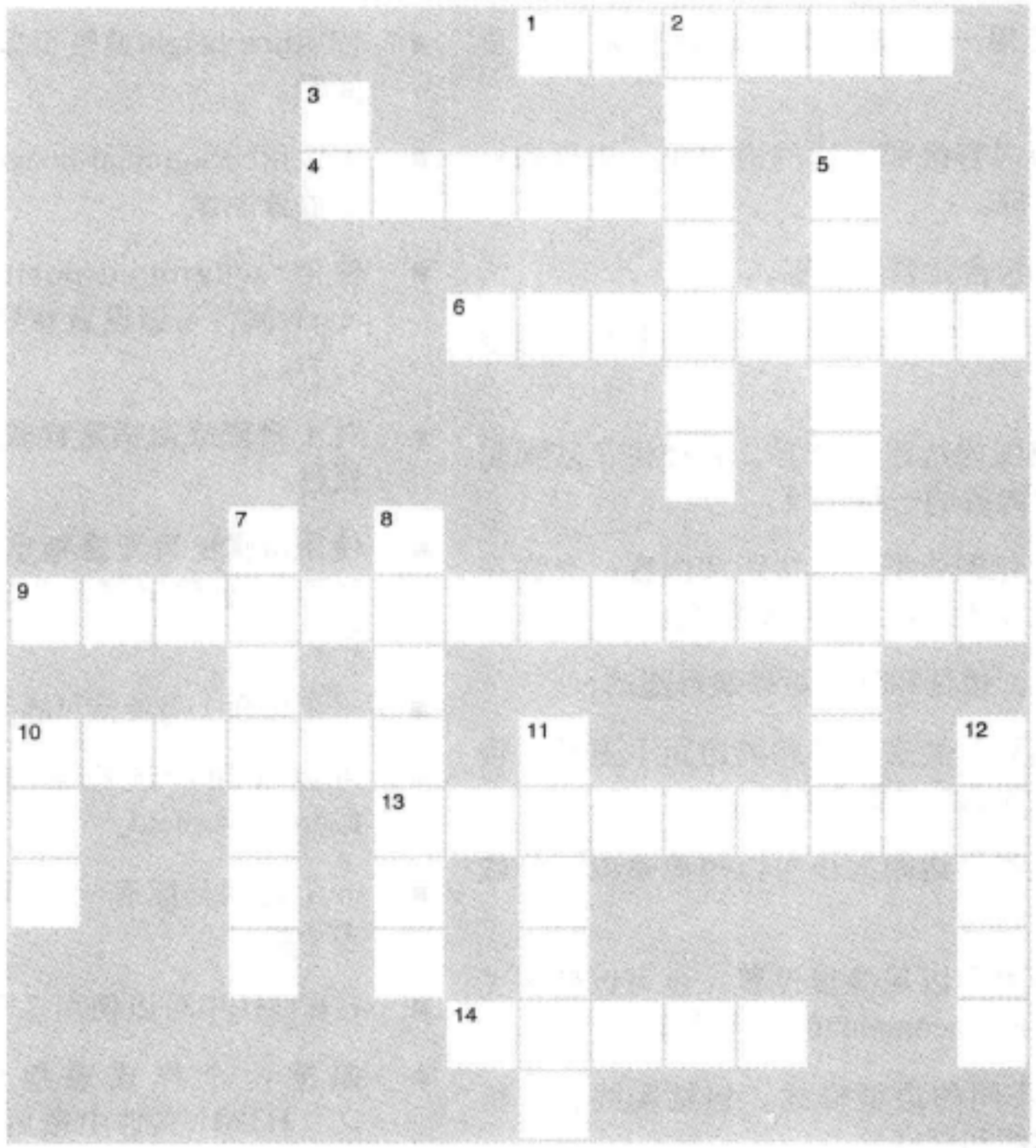
一定要使用一个现代浏览器！如果你在使用IE，这就意味着必须使用IE9以上的版本。

BULLET POINTS

- CSS使用一个盒模型来控制元素如何显示。
- 盒子由内容区和可选的内边距、边框和外边距组成。
- 内容区包含元素的内容。
- 内边距用来在内容区周围创建可见的空间。
- 边框包围内边距和内容，它提供了从视觉上分离内容的一种手段。
- 外边距包围边框、内边距和内容，允许在元素和其他元素之间增加空间。
- 内边距、边框和外边距都是可选的。
- 元素的背景会在内容和内边距下显示，但不会延伸到外边距下面。
- 内边距和外边距大小可以用像素或百分数设置。
- 边框宽度可以用像素设置，也可以使用关键字thin、medium和thick来指定。
- 有8种不同的边框样式，包括实线、破折线、虚线和脊线。
- 对于外边距、内边距或边框，CSS提供了相应的属性，可以一次对所有四个边（上、右、下、左）完成设置，也可以单独设置任意一边。
- 使用border-radius属性可以对有边框的元素创建圆角。
- 使用line-height属性可以增加文本行之间的间距。
- 可以用background-image属性在元素的背景上放置图像。
- 使用background-position和background-repeat属性可以设置背景图像的位置和平铺行为。
- 对于希望成组指定样式的元素要使用class属性。
- 使用id属性为元素指定一个唯一的名字。还可以使用id属性为元素提供唯一的样式。
- 一个页面上有给定id的元素只能有一个。
- 可以使用id选择器按id选择元素。例如#myfavoriteid。
- 一个元素只能有一个id，不过它可以属于多个类。
- 在HTML中可以使用多个样式表。
- 如果两个样式表包含冲突的属性定义，HTML文件中最后链接的样式表最为优先。
- 可以在<link>元素中使用媒体查询或者使用CSS中的@media规则来指定设备。

HTML填字游戏

你的HTML和CSS技能已经大大增强。完成一个填字游戏来强化记忆。所有答案都能在这一章中找到。



横向

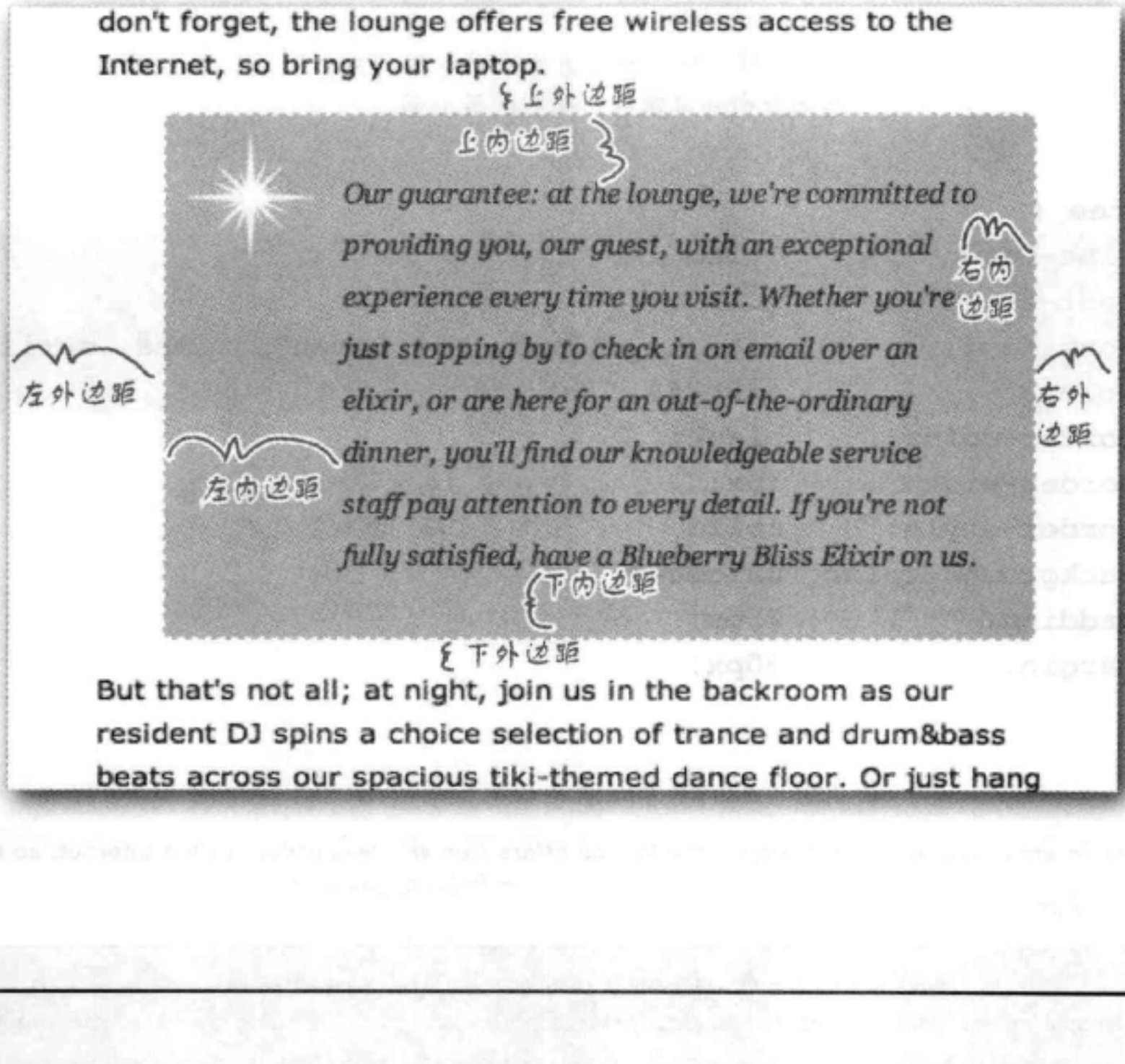
1. 默认地，背景图像会做这件事。
4. 要创建一个“锯齿”边框，要使用_____边框样式。
6. 内边距、边框和外边距都是 _____。
9. 如果你还不太满意，你会拿哪种饮料？
10. 在内边距和外边距之间。
13. 我们把 _____ 类改为一个id。
14. 要为不同的设备使用不同的样式，需要使用 _____ 查询。

纵向

2. 内容和边框之间的空间。
3. 如果希望元素有唯一的样式，要使用这种选择器。
5. 用来增加文本行之间间距的属性。
7. 行之间间距的印刷术语。
8. 保证段落中的首选字体。
10. CSS把每个元素看作是一个 _____。
11. 保证段落上使用的边框样式。
12. 对应其他类型_____的可选 <link>属性。

Sharpen your pencil Solution

看看你能不能找出这个段落的内边距、边框和外边距。标出所有内边距和外边距（左、右、上、下）：



如果查看这个保证段落的最终版本，可以看到它有一种斜体serif字体，另外行高大于页面的其余元素（如果再仔细一点），还可以看到文本是灰色的。在下面编写CSS，将行高设置为1.9em，字体风格设置为斜体，字体颜色设置为#444444，字体系列设置为Georgia, “Times New Roman”, Times, serif。下面给出答案……你测试过吗？

可以在规则的任意位置增加新属性。我们把这些新属性增加到规则的最上面。

```
.guarantee {
        line-height:      1.9em;
        font-style:       italic;
        font-family:      Georgia, "Times New Roman", Times, serif;
        color:            #444444;
        border-color:     black;
        border-width:     1px;
        border-style:     solid;
        background-color: #a7cece;
        padding:          25px;
        margin:           30px;
}
```

注意，如果一个字体名中包含有空格，要加上引号。

of sights from eras past. And, don't forget, the lounge offers free wireless access to the Internet, so bring your laptop.

增加了行高。

一种斜体serif字体。

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

灰色使文本看起来更柔和。

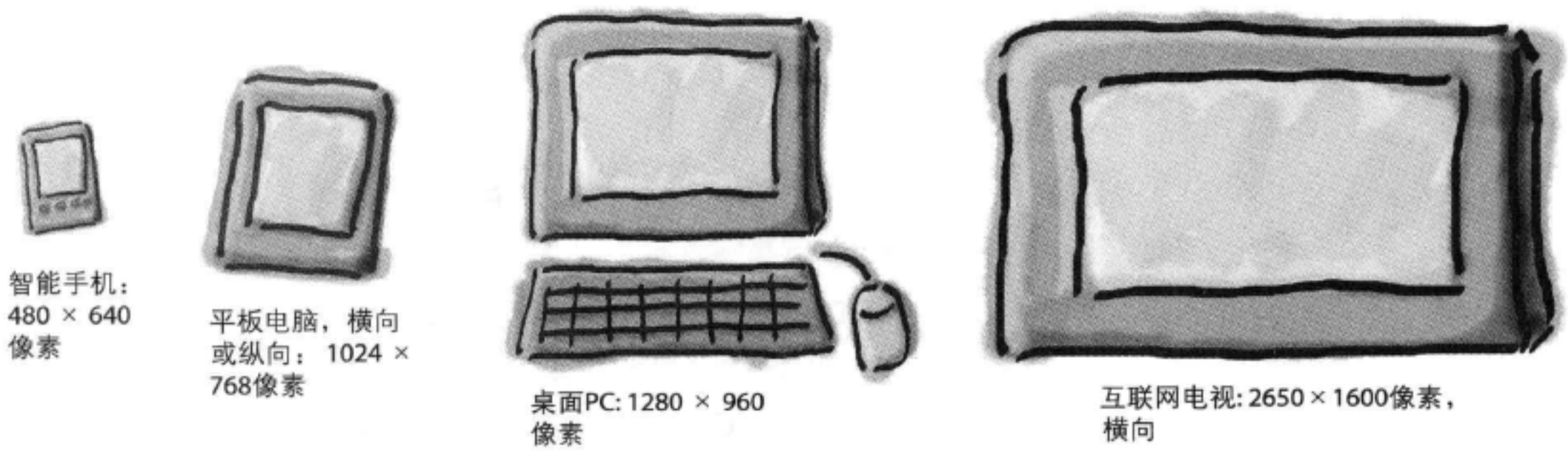
But that's not all; at night, join us in the backroom as our resident DJ spins a choice selection of trance and drum&bass beats across our spacious tiki-themed dance floor. Or just hang out in one of our comfy white vinyl

HTML填字游戏答案

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

Exercise Solution

请查看下面的设备以及相应的规格。你能设计一组媒体查询指定以下各个设备吗？



智能手机：480 × 640像素

平板电脑，横向或纵向： 1024 × 768像素

桌面PC: 1280 × 960像素

互联网电视: 2650 × 1600像素，横向

```
<link rel="stylesheet" href="lounge-smartphone.css"
      media="screen and (max-device-width: 480px)">
<link rel="stylesheet" href="lounge-tablet-portrait.css"
      media="screen and (max-device-width: 1024px) and (orientation:portrait)">
<link rel="stylesheet" href="lounge-tablet-landscape.css"
      media="screen and (max-device-width: 1024px) and (orientation:landscape)">
<link rel="stylesheet" href="lounge-pc.css"
      media="screen and (max-device-width: 1280px)">
<link rel="stylesheet" href="lounge-tv.css"
      media="screen and (max-device-width: 2650px)">
```

各种设备对媒体查询的支持还在不断发展，所以请跟踪网上最新最棒的技术。

这是我们的答案。你的答案一样吗？实际上有很多方法可以达到目的，只是特定程度有所不同。如果你的做法与我们的不同，你觉得相比之下你的做法怎么样？

Exercise

在你品味冰茶的同时，别让手闲着，为保证段落增加一个border-radius。下面给出了保证段落的几个例子，它们分别设置了不同的border-radius值。请写出CSS来创建这个例子中看到的边框。我们已经提供了每个例子中创建圆角所用的border-radius值。

在这里写出你的CSS。

30px

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

```
border-top-left-radius: 30px;
border-top-right-radius: 0px;
border-bottom-right-radius: 0px;
border-bottom-left-radius: 30px;
```

40px

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

```
border-top-left-radius: 40px;
border-top-right-radius: 40px;
border-bottom-right-radius: 40px;
border-bottom-left-radius: 40px;
```

40px

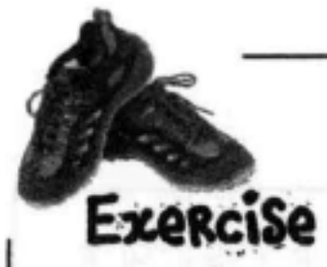
Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

```
border-top-left-radius: 0px;
border-top-right-radius: 40px;
border-bottom-right-radius: 40px;
border-bottom-left-radius: 40px;
```

2em

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

```
border-top-left-radius: 0em;
border-top-right-radius: 2em;
border-bottom-right-radius: 0em;
border-bottom-left-radius: 2em;
```



max-device-width和min-device-width媒体属性依赖于设备的实际屏幕（而不是你的浏览器窗口宽度）。如果你更关心浏览器大小呢？嗯，可以使用max-width和min-width属性，它们表示浏览器窗口本身的最大和最小宽度（而不是屏幕大小）。下面来看这是如何工作的：在你的“chapter9/lounge”文件夹中，可以找到一个文件“lounge-mobile.css”。再次打开你的lounge.html文件，修改文档<head>中的<link>元素，如下所示：

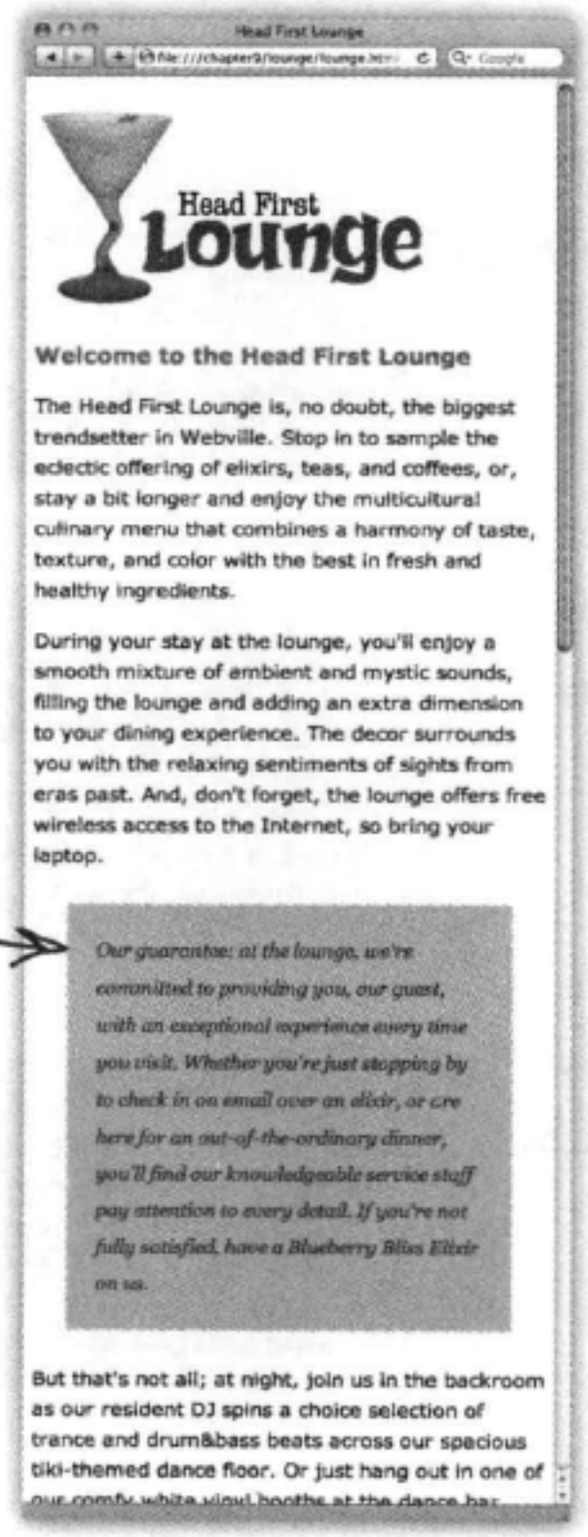
```
<link type="text/css" rel="stylesheet" href="lounge.css"
      media="screen and (min-width: 481px)">
<link type="text/css" href="lounge-mobile.css" rel="stylesheet"
      media="screen and (max-width: 480px)">
<link type="text/css" href="lounge-print.css" rel="stylesheet" media="print">
```

现在请在你的浏览器中重新加载“lounge.html”页面。确保浏览器窗口相当大。你会看到正常的休闲室页面。

接下来，让浏览器窗口变窄（宽度小于480像素）。休闲室页面会怎么样呢？注意到差别了吗？请在下面描述将浏览器窗口变窄并加载页面时的结果。为什么这个版本更适合移动浏览器？

让休闲室页面变窄，宽度小于480像素时，保证段落会改变样式。右内边距会从250px减少到30px（与其余外边距一致）。背景的星星图像消失了，左边额外的内边距也没有了。

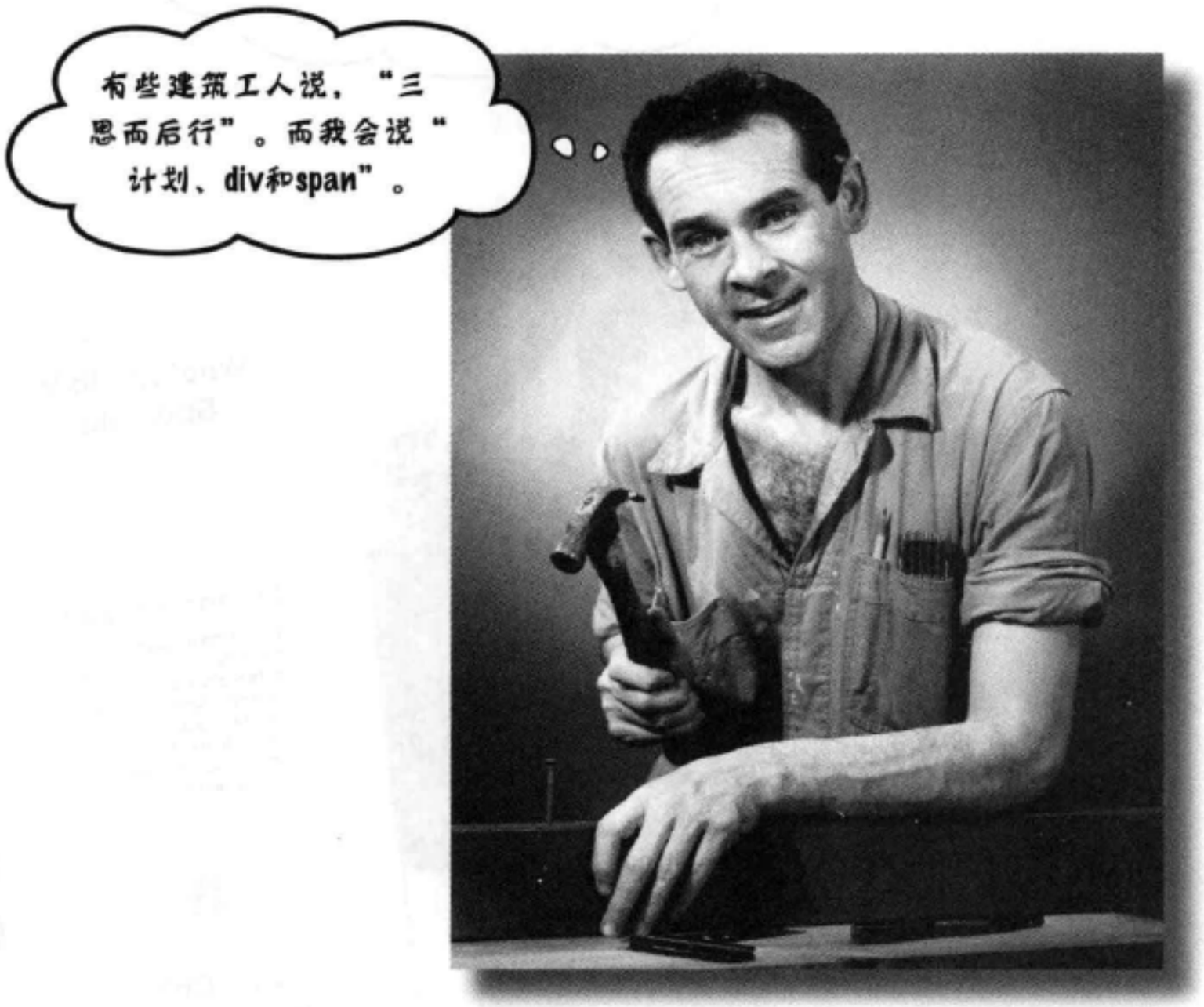
这个版本更适合移动浏览器，因为如果使用为更宽屏幕设计的CSS，保证段落会变得太窄。通过删除背景图像和额外的外边距和内边距，段落将更容易阅读。再说了，最重要的还是内容，不是吗？



一定要使用一个现代浏览器！如果你在使用IE，这就意味着必须使用IE9以上的版本。

10 div与span

高级Web建设



该建个大工程了。这一章中，我们要介绍两个新的HTML元素： 和。它们绝对不是简简单单的“小玩艺”，而是堪称举足轻重的钢铁支架。利用 和，你能构建重要的支撑架构，一旦有了这些架构，就能用各种新颖、强大的方式为它们增加样式。我们已经注意到，你的CSS工具箱开始有点满了，所以很有必要为你展示一些捷径，让你能更容易地指定属性。这一章还会请到一些特殊的客人——伪类（pseudo-classes），利用它们你可以创建一些非常有趣的选择器（如果你觉得“伪类”这个名字很不错，想把它当做你的下一个乐队的名字，嘿嘿，太晚了，我们已经捷足先登）。

这是特色饮料的宣传单。哇，这个设计与页面的其余部分可大不一样。它很窄，文本居中，而且有红色标题，所有内容用青绿色边框包围，最上面甚至还有一些鸡尾酒图片。

仔细观察饮料HTML

Alice的要求太离谱了，是不是？她居然希望我们把现在的休闲室HTML变成像宣传单一样。嗯……听上去很有挑战性，不过我们有CSS帮忙，所以倒是可以试一试。但在具体处理样式之前，先对现有的HTML有个大概了解。下面就是特色饮料的HTML片段；可以在“chapter10/lounge”文件夹的“lounge.html”中找到：

特色饮料部分以一个<h2>标题开始。

```
<h2>Weekly Elixir Specials</h2>

<p>
        <img src="images/yellow.gif" alt="Lemon Breeze Elixir">
</p>
<h3>Lemon Breeze</h3>
<p>
        The ultimate healthy drink, this elixir combines
        herbal botanicals, minerals, and vitamins with
        a twist of lemon into a smooth citrus wonder
        that will keep your immune system going all
        day and all night.
</p>

<p>
        <img src="images/chai.gif" alt="Chai Chiller Elixir">
</p>
<h3>Chai Chiller</h3>
<p>
        Not your traditional chai, this elixir mixes mat&eacute;
        with chai spices and adds an extra chocolate kick for
        a caffeinated taste sensation on ice.
</p>

<p>
        <img src="images/black.gif" alt="Black Brain Brew Elixir">
</p>
<h3>Black Brain Brew</h3>
<p>
        Want to boost your memory? Try our Black Brain Brew
        elixir, made with black oolong tea and just a touch
        of espresso. Your brain will thank you for the boost.
</p>

<p>
        Join us any evening for these and all our
        other wonderful
        <a href="beverages/elixir.html"
           title="Head First Lounge Elixirs">elixirs</a>.
</p>
```

我们有3种饮料，它们有相同的结构。

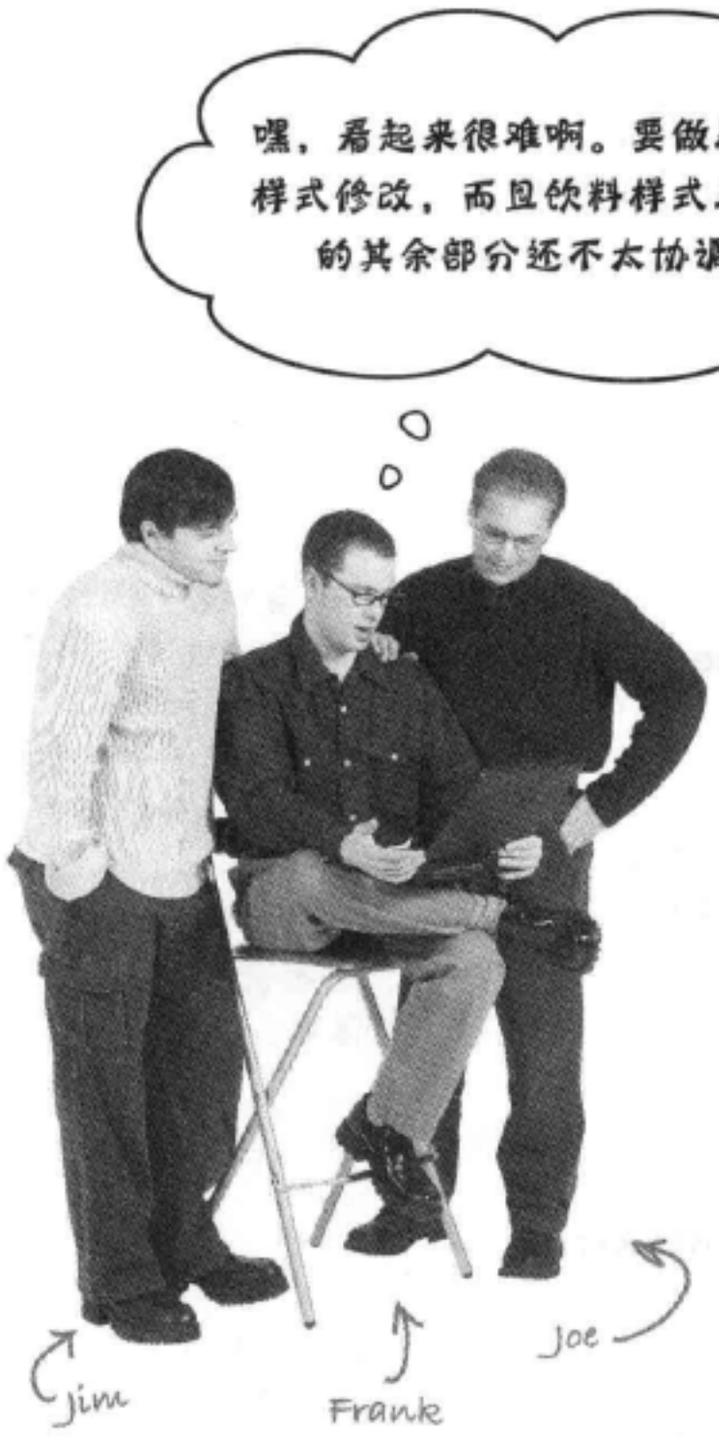
每种饮料有一个<p>元素，其中有一个图像。

……另外还有一个包含饮料名字的<h3>标题……

……还有一个介绍，这也放在一个段落中。

每个饮料都重复这个结构。

最后，在最下面还有一个段落，其中包含一些文本，还有一个链接指向具体的饮料页面。



Jim：别急，Frank，你知道我们可以创建一、两个类，然后可以区别于页面的其余部分，单独为所有饮料元素指定样式。

Frank：这倒是真的。也许情况还不算太糟糕。我相信肯定有一个简单的属性可以让文本居中，另外我们已经知道如何处理彩色文本。

Jim：等一下，那些边框呢？

Frank：小菜一碟。我们刚学过如何设计边框。记住，每个元素都可以有一个边框。

Joe：嗯，我可不这么认为。如果你仔细看那些HTML，有一大堆的<h2>、<h3>和<p>元素。如果每个元素上都加上单独的边框，它们看上去就是一大堆单独的盒子。

Frank：你说的没错，Joe。我们需要一个合适的元素，把所有这些元素都嵌在其中，这样我们就可以只对这个元素设置一个边框。如此一来，页面饮料部分中的所有内容就会有一个边框。

Jim：嘿，你真棒，Frank，我知道为什么你的薪水高了。我们能不能把这些饮料内容嵌在一个<p>元素或一个<blockquote>中？

Frank：嗯，这会破坏页面的结构和含义。饮料单并不是一个段落，也不是一个块引用。好像有点棘手……

Frank：……实际上，我觉得我们的方向还是对的。我一直在读一本关于HTML和CSS的书，我刚看到这一节谈到一个名叫 的新元素。这可能正是我们需要的工具。

Joe： ，那是什么？听起来好像是数学计算方面的。

Frank：也可以这么说，因为 确实允许你将页面划分为逻辑区或逻辑分组。

Jim：嘿，听上去正是我们要的！

Frank：好极了，下面我来教你们如何把一个页面划分为几个逻辑区，然后再告诉你们关于 我知道些什么……

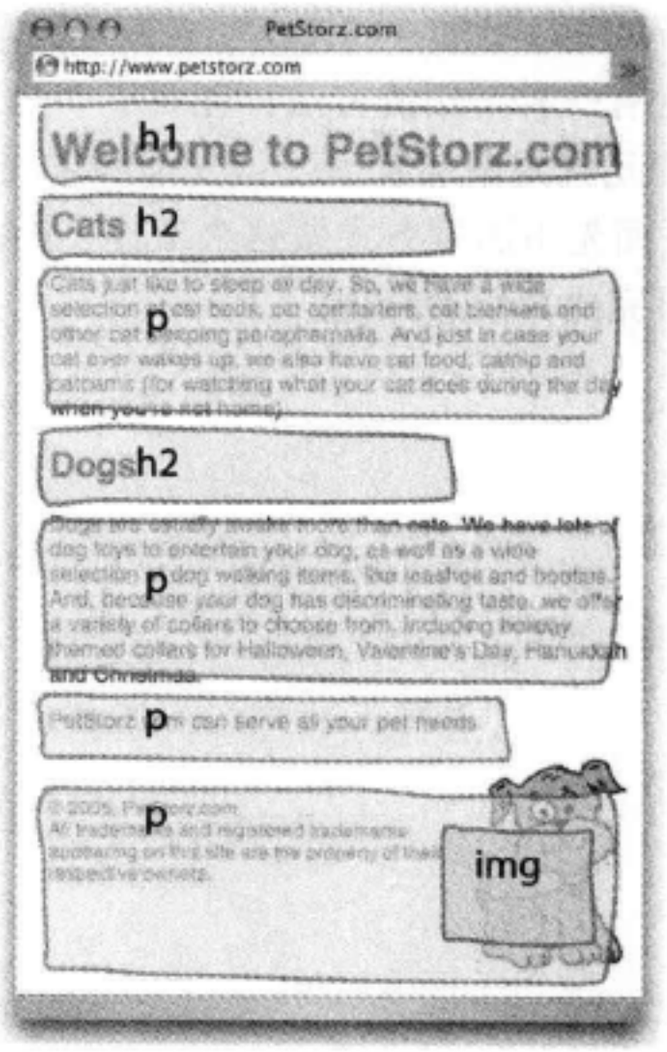
下面来研究如何将一个页面划分为逻辑区

来看右边的Web页面，这是为PetStorz.com建立的一个Web页面，接下来几页我们会研究如何为这个页面增加一些结构，首先找出一些逻辑区，然后把这些逻辑区分别包围在一个 元素中。

这是一个看着很普通的页面：有很多标题和段落，还有一个图像。

不过，如果只看这个页面的结构，确实无法了解页面的太多信息。哪些元素构成了页眉？页面上有没有页脚？哪些是内容区？

我们画出了这个PetStorz页面的略图。

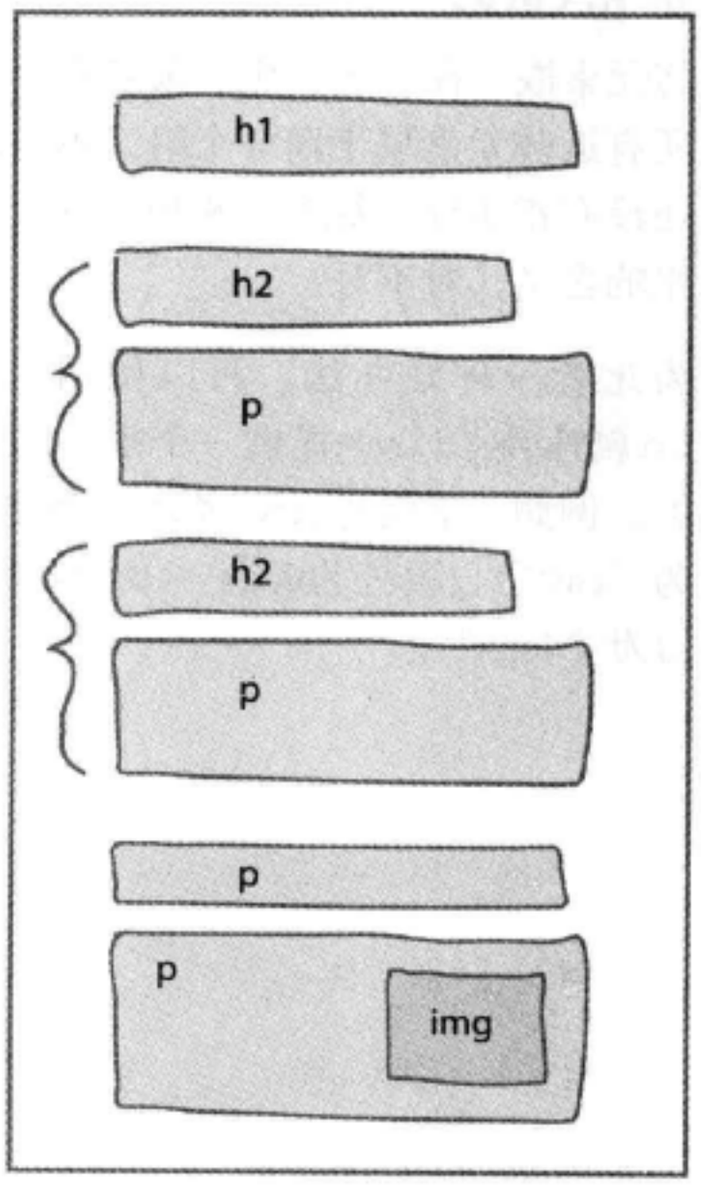


找出逻辑区

OK，所以你的任务就是找出这个页面中的“逻辑区”（logical section）。什么是逻辑区？逻辑区就是页面上彼此相关的一组元素。例如，在PetStorz.com的Web页面上，有一些元素用于“猫猫”区（cats），另外一些用于“狗狗”区（dogs）。下面来看一下。

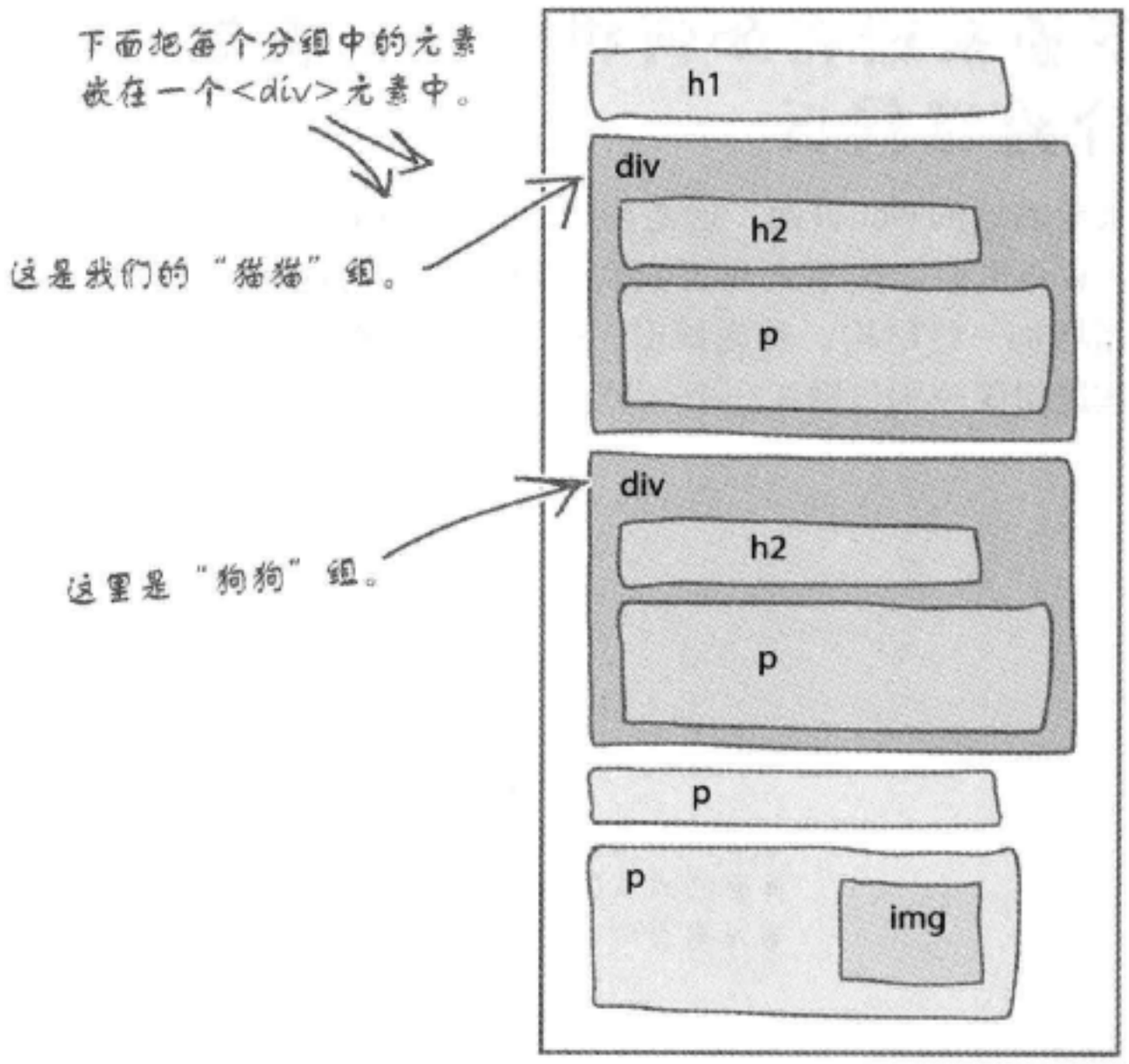
PetStorz页面有两个主要的内容区，一个对应“猫”，另一个对应“狗”。还有另外一些区域，不过稍后再来讨论那些区域。

在这里，“猫猫”区和“狗狗”区都包括两个元素：一个标题和一个段落。不过通常这些逻辑分组还可以包含更多的元素。



使用 标记逻辑区

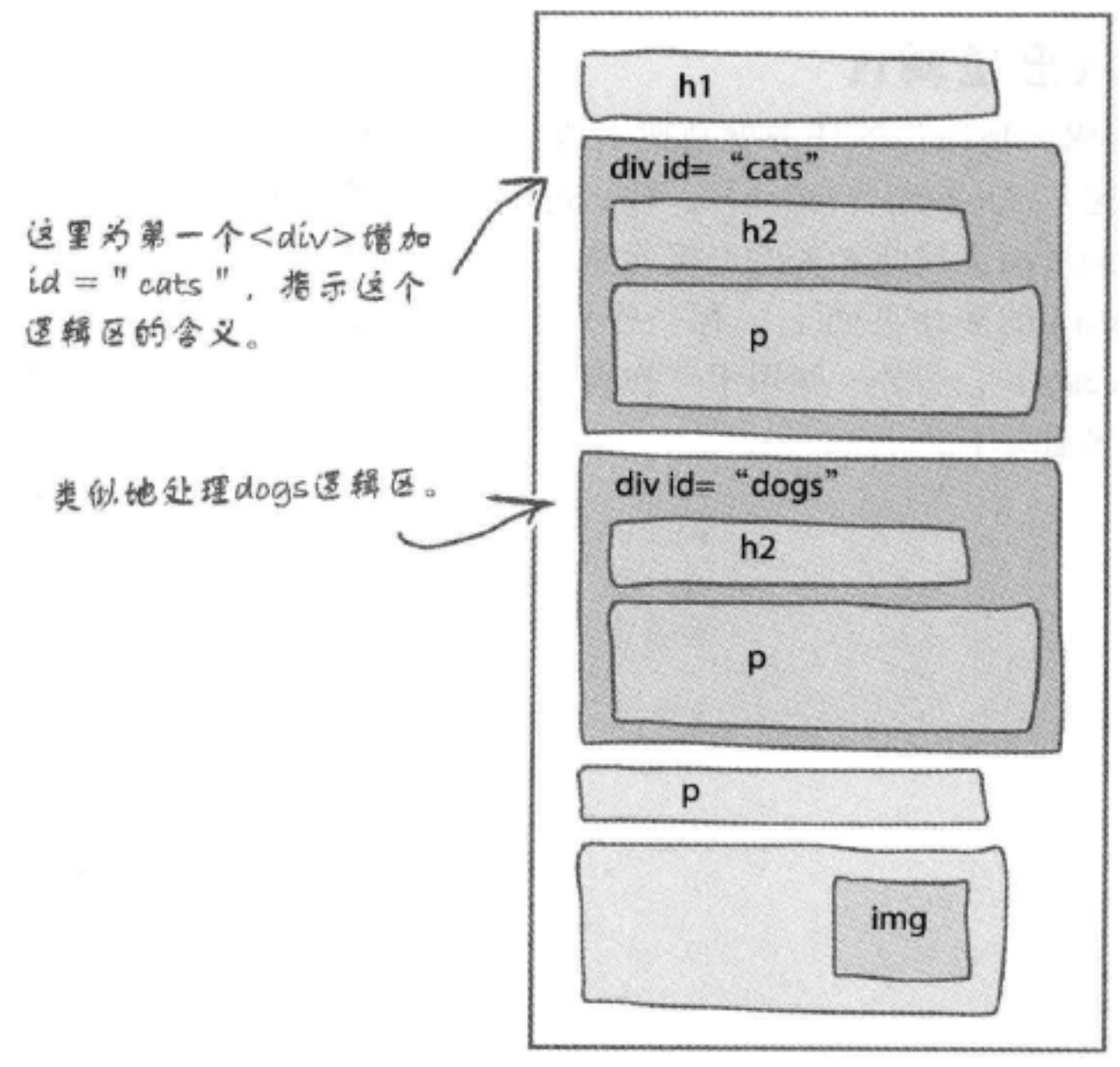
既然已经知道各个逻辑区包括哪些元素，下面可以增加一些HTML来标记这个结构。这有一种常用的方法：在属于一个逻辑区的元素周围放置 开始和结束标记。下面先用图解方式做这个工作，后面几页再完成真正的标记。



标出

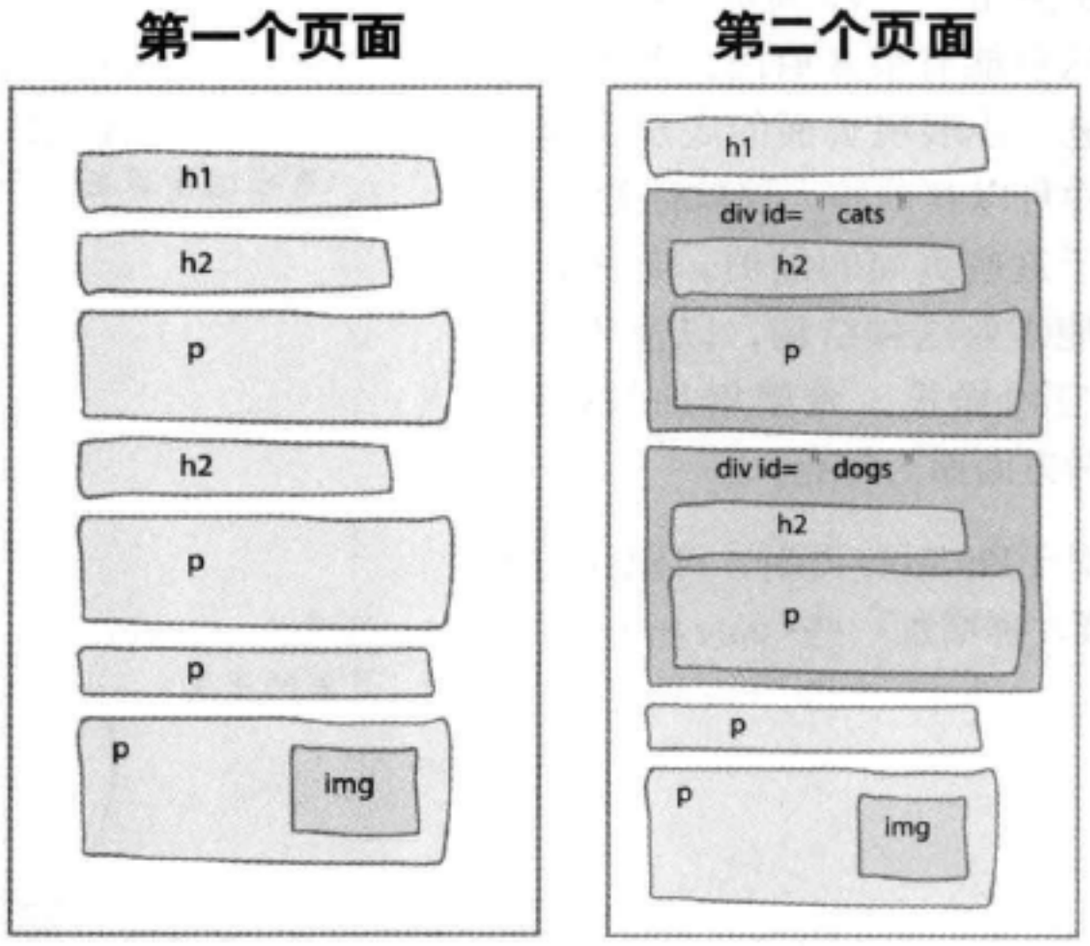
将元素嵌入在 中，就是在指示所有这些元素属于同一个组。不过你还没有指定任何标签，来说明这个分组的含义，对不对？

为此有一种好办法，可以使用一个id属性为 提供一个唯一的标签。例如，下面为cats 指定id为“cats”，另外为dogs 指定id为“dogs”。



经Starbuzz CEO的介绍，现在请你来做顾问，考虑如何修改PetStorz主页面的样式。如果只让你看第一个页面，你能不能很快了解PetStorz web页面？

如果让你看第二个页面呢？



增加一些样式

OK，现在已经为PetStorz页面增加了一些逻辑结构，你还提供了标签，为每个 指定了一个唯一的id。这就是所需的全部准备工作，接下来就可以对 中包含的一组元素指定样式了。

这里有两个规则，分别对应一个 。每个 由一个id选择器来选择。

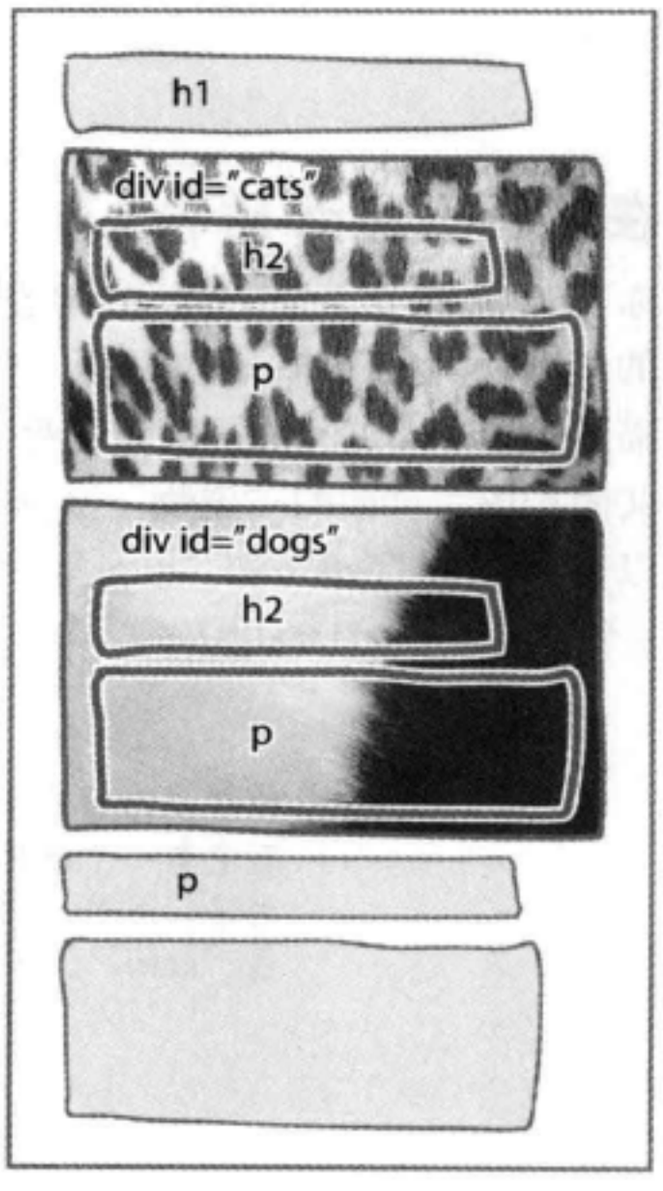
```
#cats {
   background-image: url(leopard.jpg);
}

#dogs {
   background-image: url(mutt.jpg);
}
```

这两个规则都分别设置了background-image属性。对于cats，我们设置了一个猎豹图像，另外为dogs 设置了一个小狗图像。

现在 有一点样式。

通过对 设置背景，它会透过 中包含的元素显示出来。

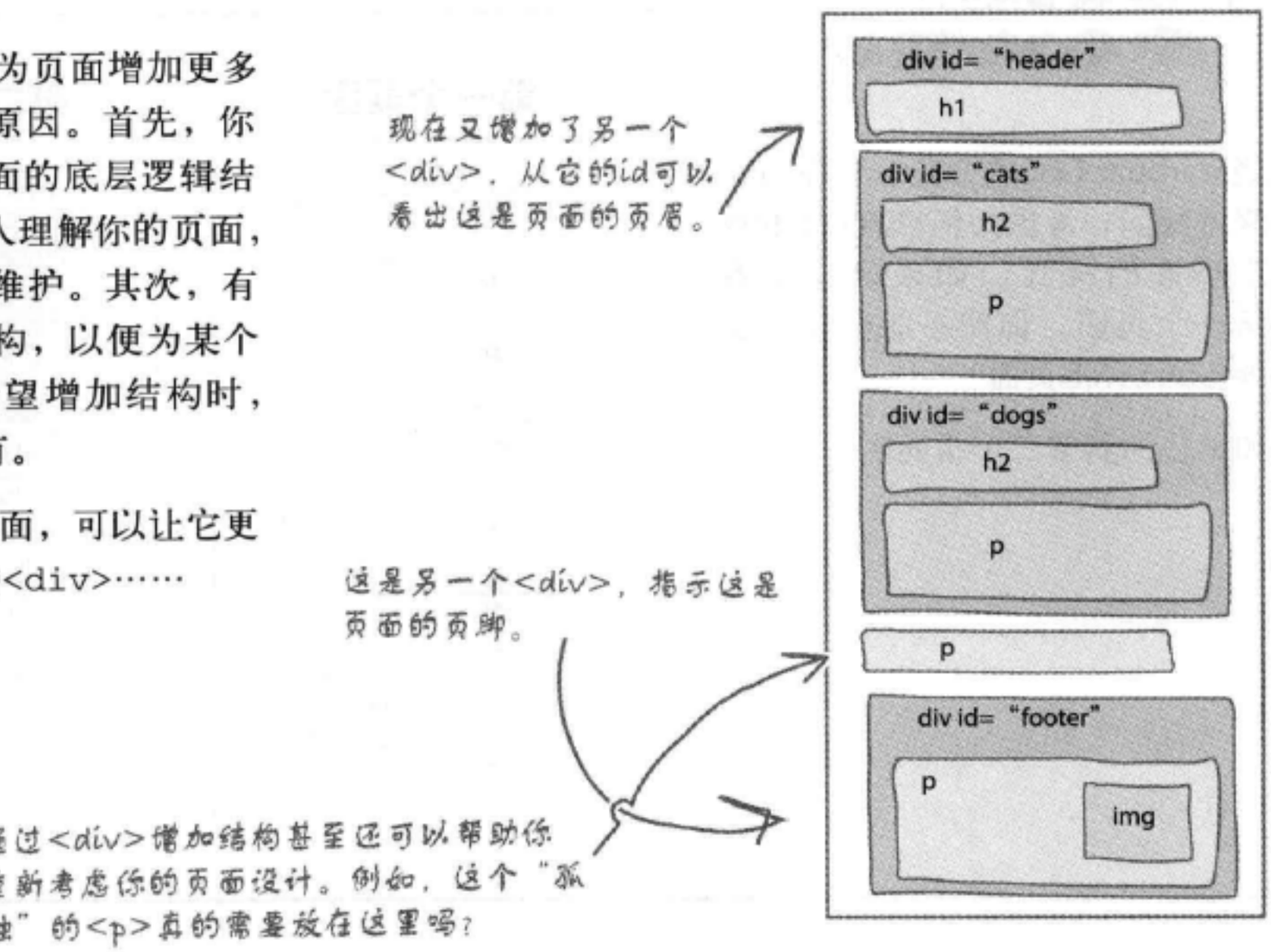
就像所有子元素一样， 中的元素也会从 继承一些属性（如font-size、color等）。



展现更多结构

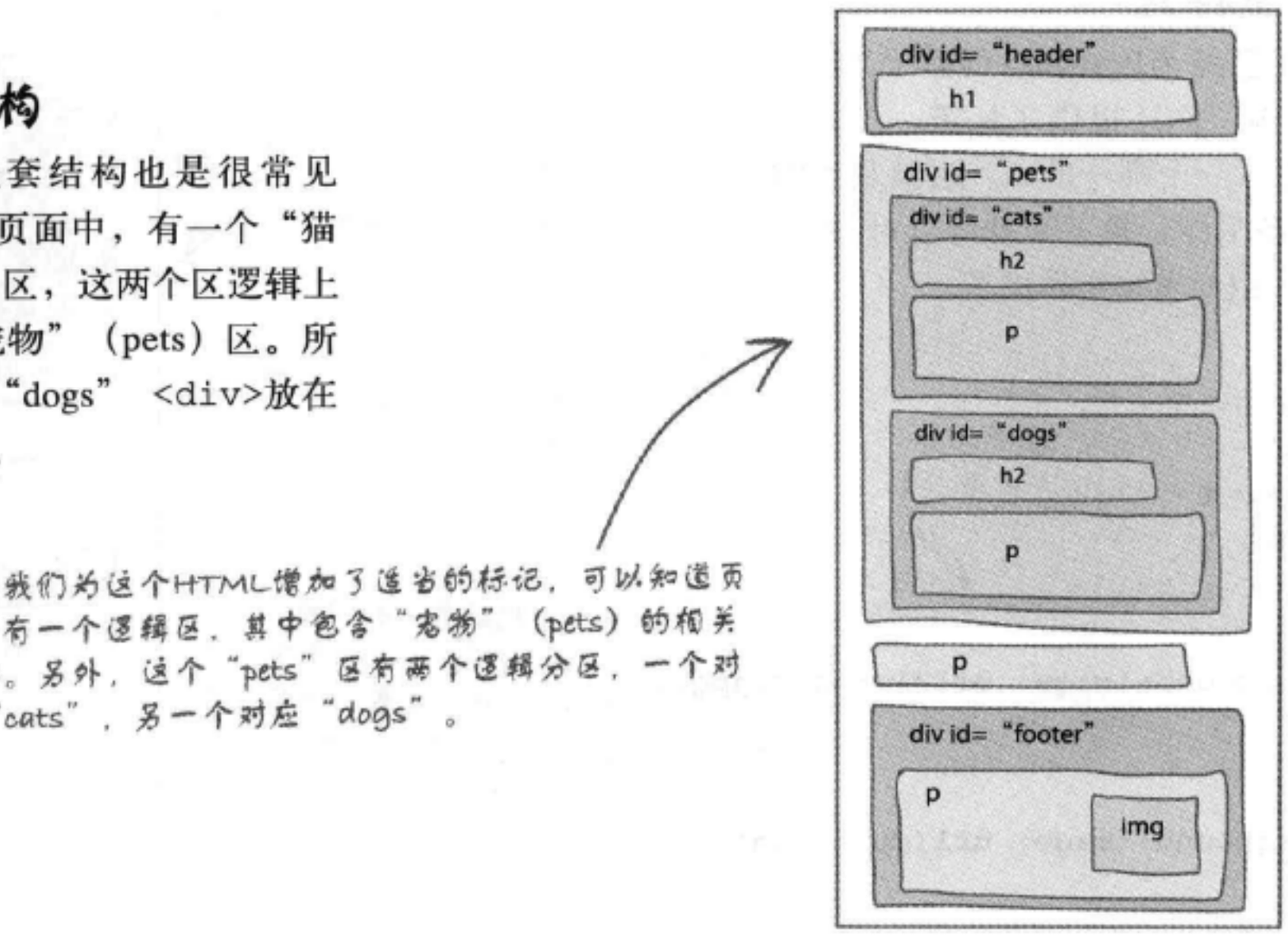
你可能希望用 为页面增加更多结构，这可能有很多原因。首先，你可能想进一步展现页面的底层逻辑结构，这样可以帮助别人理解你的页面，还有助于这些页面的维护。其次，有时你可能需要这种结构，以便为某个逻辑区应用样式。希望增加结构时，通常这两方面原因都有。

所以，对于PetStorz页面，可以让它更上一层楼，再增加一些 ……



在结构上增加结构

你不必就此止步。嵌套结构也是很常见的。例如，在PetStorz页面中，有一个“猫猫”区和一个“狗狗”区，这两个区逻辑上共同构成了页面的“宠物”（pets）区。所以，可以把“cats”和“dogs” 放在一个“pets” 中。



there are no Dumb Questions

问：这么说， 就相当于一个容器，可以把元素一同放在这个容器里，是吗？

答：没错。实际上，我们通常就把 描述为“容器”。它们不仅相当于逻辑容器，可以用来将一堆相关的元素（如“cat”元素）放在一起，另外对 指定样式和定位时（见下一章），你会看到它们还相当于图形容器。

问：页面中除了用标题和段落等建立的结构外，我还应该用 增加更高一层的结构，是吗？

答：这个不好说。如果确实有实际作用，就可以增加结构，不过不要为了加结构而不必要地增加结构。一定要在保证完成任务的前提下让结构尽可能简单。例如，如果为PetStorz页面增加一个“pets”区包含“cats”和“dogs”区会有帮助，就要尽可能增加这层结构。不过，如果它不能提供任何实际的好处，就只会让页面更复杂。

使用 一段时间后，你就会对什么时候使用以及使用多少 才合适有点感觉了。

问：你把 放在一个类里，而不是为它指定一个id，这样行吗？

答：嗯，要记住，一个元素有一个id，同时可以属于一个或多个类，所以这个选择并不互相排斥。没错，很多情况下会创建 并把它们放在类中。假设你的音乐播放列表页面中有很多唱片区，你可以把组成唱片的所有元素放在一个 中，然后把这些 放在一个“albums”类中。这样可以标识唱片在哪里，而且可以利用类对它们同时指定样式。不仅如此，你还可以为每个唱片指定一个唯一的id，从而能为它单独的应用额外的样式。

问：我对 嵌在 中的做法还不太明白，比如“pets”、“cats”和“dogs”之间的关系，这到底是怎么回事。你能再做些解释吗？

答：当然可以。你习惯于元素嵌套在其他元素中，对吗？比如一个<p>嵌套在一个<body>中，而<body>又嵌套在一个<html>元素中。你甚至见过列表嵌套在列表中。 实际上也没有不同；还是将一个元素嵌套在另一个元素中。对于PetStorz页面，我们使用 来显示更大的结构块（“cats”和“dogs”嵌套在一个“pets”区中）。或者，也可以使用 将一个啤酒区嵌套在饮料区中，而饮料区进一步嵌套在酒单区中。

不过，要理解为什么需要像这样嵌套 ，最好的办法是在遇到合适的情况时具体使用 。记着这一点，稍后就会看到一个需要嵌套 的例子。

在页面中要使用 ，但不要滥用。如果这样做有助于你将页面分解为逻辑区，从而保证结构清晰并便于指定样式，那么可以增加更多的结构。如果只是为了在页面中创建大量结构而增加 ，就只会让页面复杂，而没有任何实际好处。

再回到休闲室……

关于 的理论已经讲得够多了，下面在休闲室页面里增加一个 。记住，我们打算把所有饮料元素放在一个组中，然后为它指定样式，使它看起来就像饮料宣传单一样。所以打开“chapter10/lounge”文件夹中的“lounge.html”文件，找到饮料元素，在它们周围插入开始和结构 标记。

这里是开始标记，我们为它指定id = "elixirs"，来标识这个 。

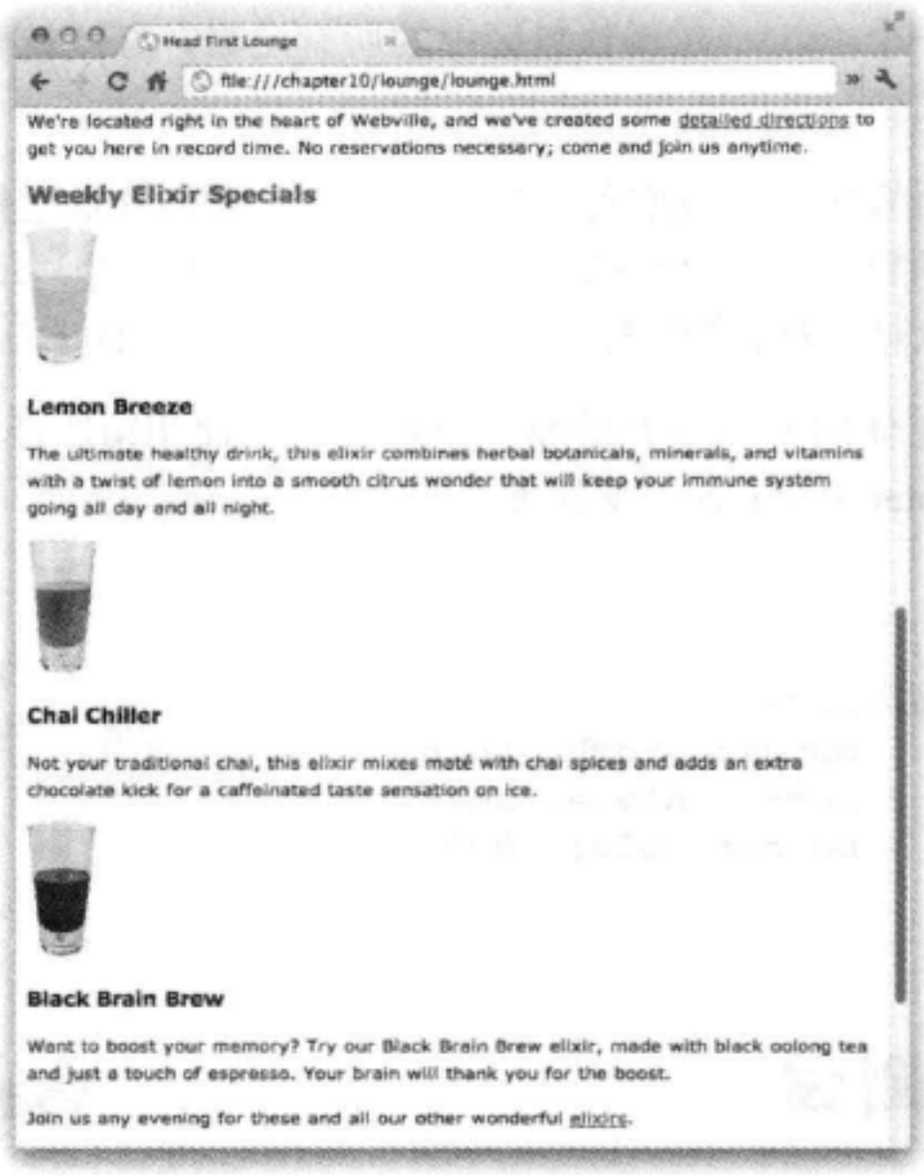
```
<div id="elixirs">
    <h2>Weekly Elixir Specials</h2>

    <p>
        <img src="images/yellow.gif" alt="Lemon Breeze Elixir">
    </p>
    <h3>Lemon Breeze</h3>
    <p>
        The ultimate healthy drink, this elixir combines
        herbal botanicals, minerals, and vitamins with
        a twist of lemon into a smooth citrus wonder
        that will keep your immune system going all
        day and all night.
    </p>

    <p>
       <img src="images/chai.gif" alt="Chai Chiller Elixir">
    </p>

    <h3>Chai Chiller</h3>
    <p>
        Not your traditional chai, this elixir mixes mat&eacute;
        with chai spices and adds an extra chocolate kick for
        a caffeinated taste sensation on ice.
    </p>

    <p>
       <img src="images/black.gif" alt="Black Brain Brew Elixir">
    </p>

    <h3>Black Brain Brew</h3>
    <p>
        Want to boost your memory? Try our Black Brain Brew
        elixir, made with black oolong tea and just a touch
        of espresso. Your brain will thank you for the boost.
    </p>

    <p>
        Join us any evening for these and all our
        other wonderful
       <a href="beverages/elixir.html"
          title="Head First Lounge Elixirs">elixirs</a>.
    </p>
</div>
```

这里是结束标记。

要记住，我们只显示了整个文件中的一个HTML片段。打开“lounge.html”时，你会看到页面的所有标记。

测试

很容易，不是吗？现在我们有了一个更为结构化的页面，下面打开浏览器，看看页面怎么样……



嗯……根本没有任何变化！不过没关系， 只是纯粹的结构，它在页面中没有“外观”或默认样式。

也就是说， 只是一个块元素，可以对它应用你希望的任何样式。所以，只要知道如何对一个块元素指定样式（你恰好知道），就会知道如何对 指定样式。

BRAIN POWER

要记住，这里的目标是对页面上的饮料内容调整样式，使它们看起来像宣传单。

在绕道去学习 之前，我们要确定如何在整个饮料内容周围加一个边框。既然“lounge.html”中已经有了一个 ，该如何增加边框呢？

增加边框

既然有一个 包围饮料区中的所有元素，好戏就要开场了，你可以对它指定样式。

对于饮料宣传单，我们首先想要“复制”的部分是边框，它要包围饮料区中的所有饮料，对不对？嗯，现在实际上已经有一个 元素包围这个饮料区，所以可以对它指定样式，增加一个边框。下面就来试一试。

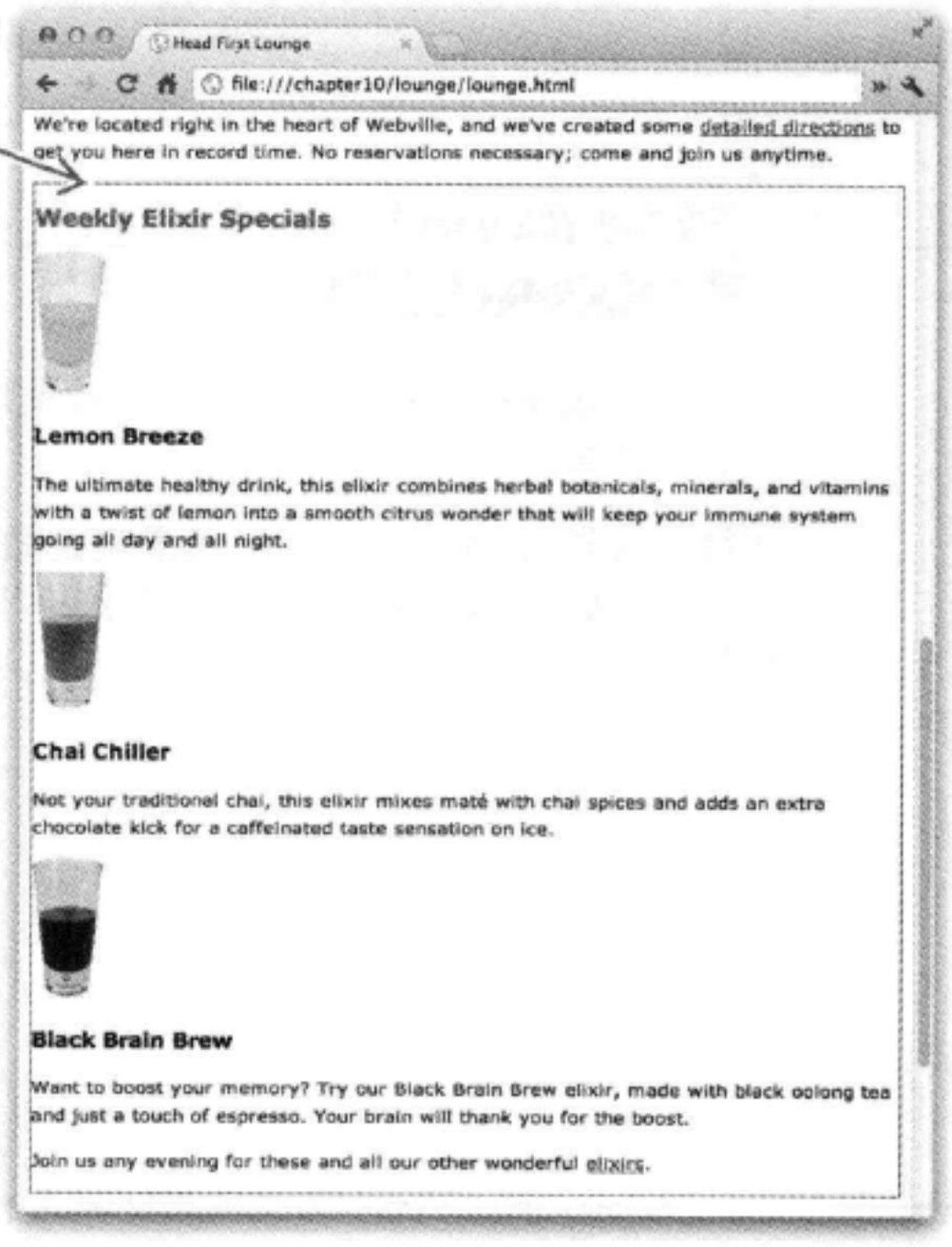
你需要在休闲室CSS中增加一个新规则，使用id选择这个 元素。打开“chapter10/lounge”文件夹中的“lounge.css”文件，在最后增加以下规则：

```
#elixirs {
    border-width: thin;
    border-style: solid;
    border-color: #007e7e;
}
```

把这个规则增加到CSS文件的最下面。它会使用id选择elixirs 元素，并用我们喜欢的青绿色增加一个细实线边框。

边框测试

增加以上CSS之后，保存文件，然后重新加载“lounge.html”文件。



这里是你刚为elixirs 元素增加的边框。

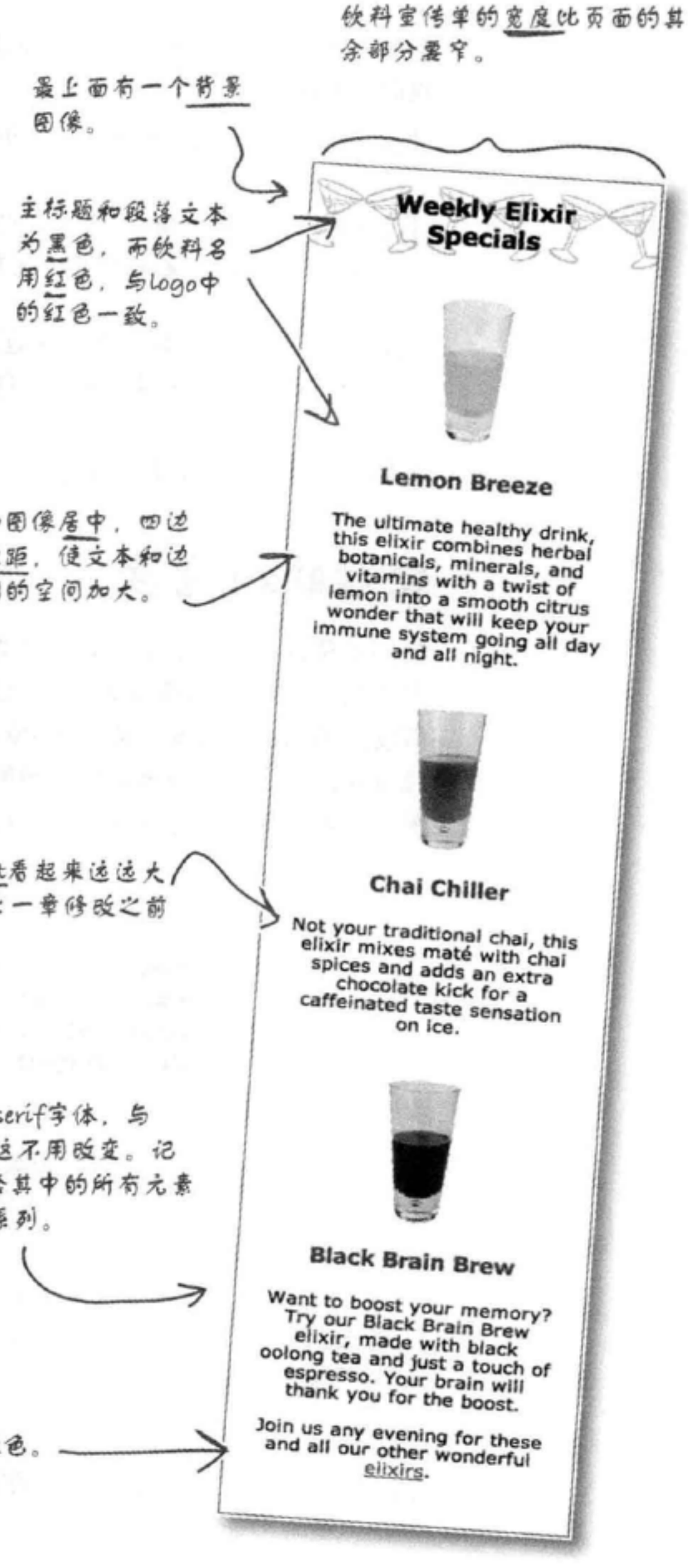
为这个 增加了一个可见的边框，不过还没有内边距和外边距，这些也需要增加。

注意这个边框包围了 元素中的所有元素。像所有其他元素一样，这个 也是一个盒子，所以增加边框时，边框会包围内容，这里的内容就是 中的所有元素。

为饮料区增加一些真正的样式

到目前为止都还不错。我们已经想办法在整个饮料区周围加上了边框。现在来看如何使用 为整个饮料区指定样式，而且要独立于页面的其余部分。

显然我们需要处理一些内边距问题，因为边框直接包围着内容。此外还有很多其他样式需要处理。下面来看需要我们考虑的所有问题……



进攻策略

新样式真不少，下面在逐个攻克之前先共同制订一个进攻策略。我们需要做到：

- ☐ 首先，要改变elixirs 的宽度，让它更窄一些。
- ☐ 接下来，处理你熟悉的一些样式，如内边距和背景图像。我们还会处理文本对齐，这方面的内容你之前还没有见过。
- ☐ 然后只剩下文本行高和标题颜色了。你会看到需要稍稍提升你的CSS选择器技能，才能完成这些改变。

要做的事情很多，下面开始吧。

处理elixir宽度

我们希望elixirs区很窄，这样才像休闲室那个细长的宣传单。大约是一个典型浏览器窗口宽度的1/4应该就可以了。所以，假设你将浏览器设置为宽度800像素，这就约为200像素。你已经设置过内边距、边框和外边距的宽度，不过之前你还没有设置过元素的宽度。为此要使用width属性，如下所示：

```
#elixirs {
        border-width: thin;
        border-style: solid;
        border-color: #007e7e;
        width: 200px;
}
```

width属性允许你指定元素内容区的宽度。在这里，我们将内容宽度指定为200像素。

在elixirs 上设置这个样式。所以elixirs 中的内容将是200像素宽，浏览器的布局规则会起作用，使嵌在 中的所有元素都在这个宽度范围内。

试一试。打开“lounge.css”，在最下面增加这个规则。

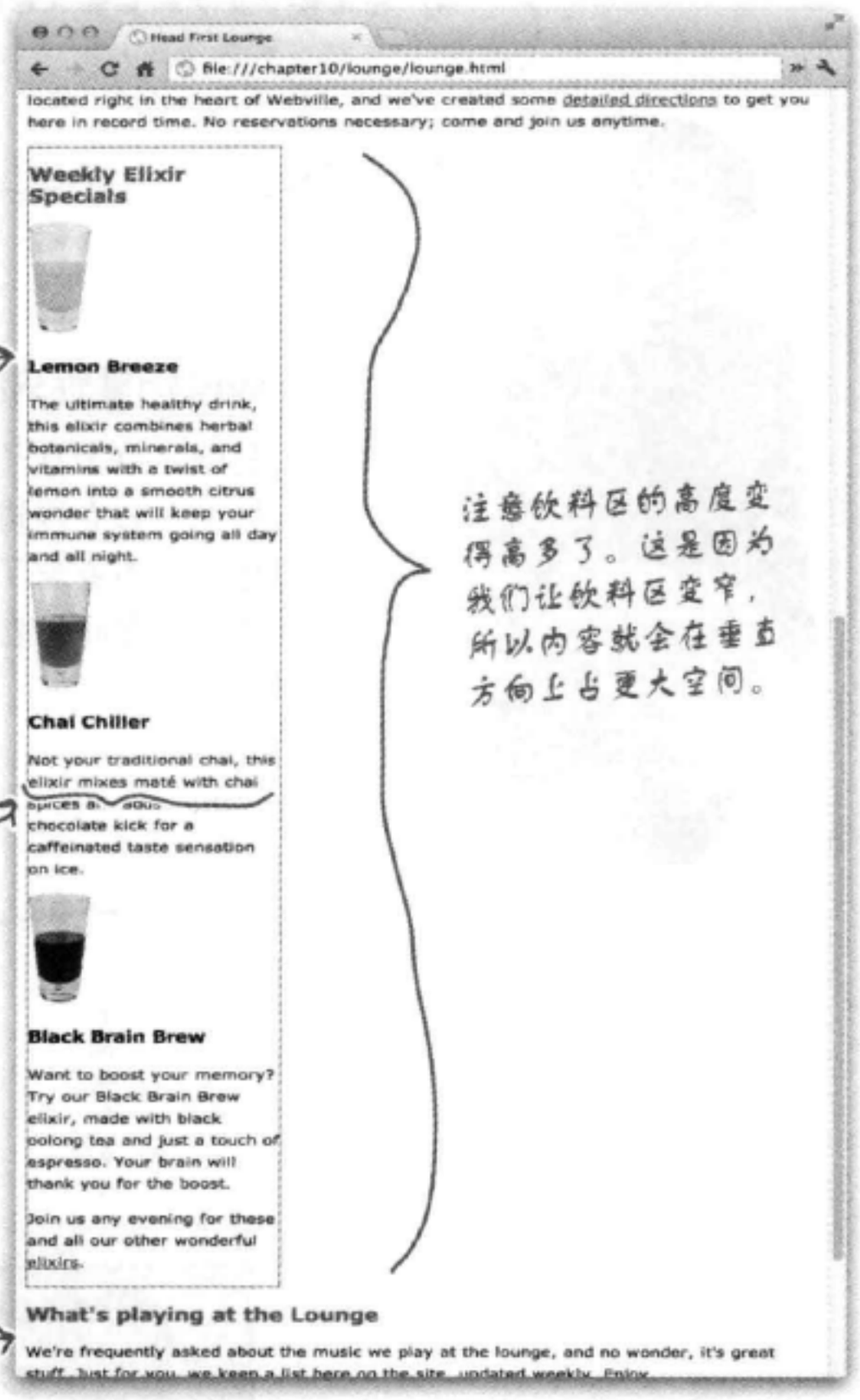
测试宽度

接下来，保存CSS，然后重新加载“lounge.html”文件。你会看到饮料区变得更窄，这是由于你为它指定的宽度。现在 中内容的宽度为200像素。还可以看到一些有意思的行为……

现在elixirs 中的所有内容都放在200像素宽的一个空间里。即使你调整浏览器窗口，让它很宽或者很窄，这也不会改变。试试看！

200像素。

如果让浏览器窗口更宽，请对这个 与其他元素的行为做个比较。这些段落会自动扩展，占满浏览器的宽度。稍后还会讨论这个内容……



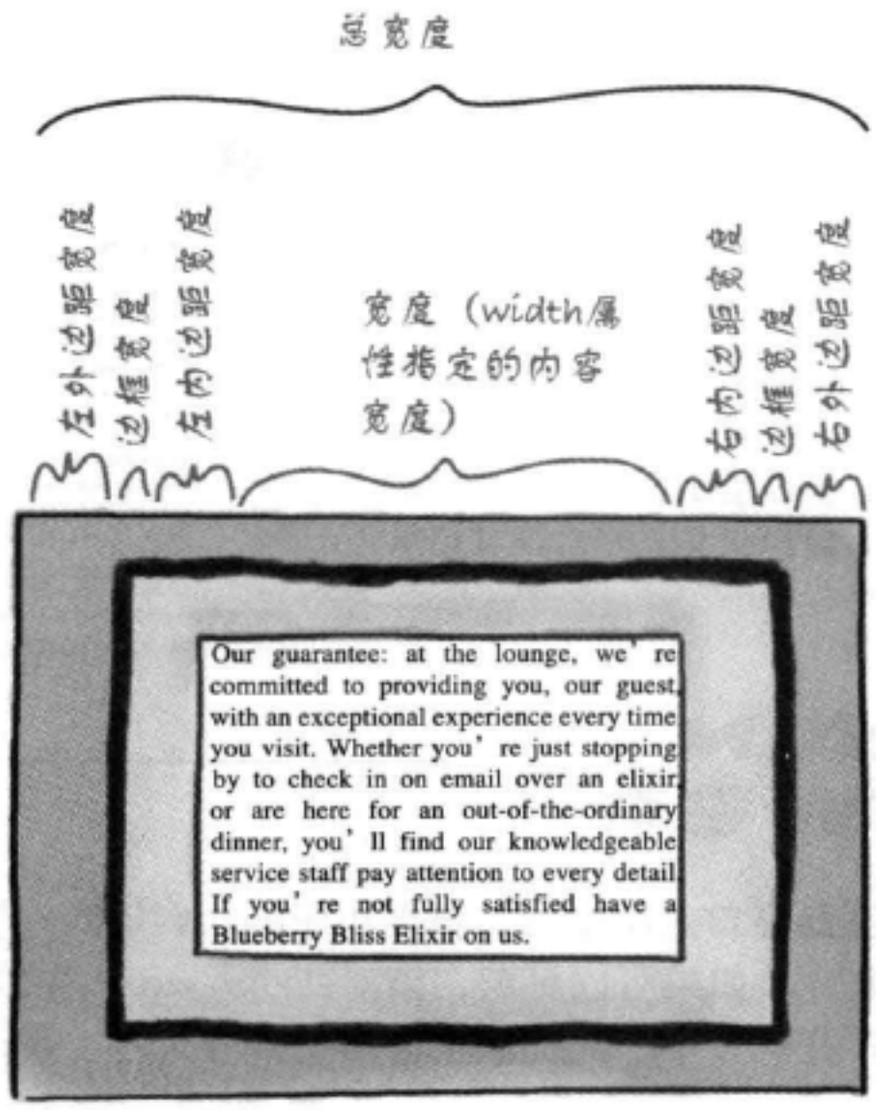
注意饮料区的高度变得高多了。这是因为我们让饮料区变窄，所以内容就会在垂直方向上占更大空间。

BRAIN POWER

能不能调整浏览器窗口大小使窗口宽度小于elixirs 的宽度？有些浏览器不允许把窗口调到那么窄，有些则允许。如果可以更窄，请对elixirs 中的文本和页面的其余文本做个比较。不论将窗口调整到多宽或多窄，其他段落会自行调整大小，而elixirs 的宽度永远是200像素，不多也不少。

width属性只指定内容区的宽度。

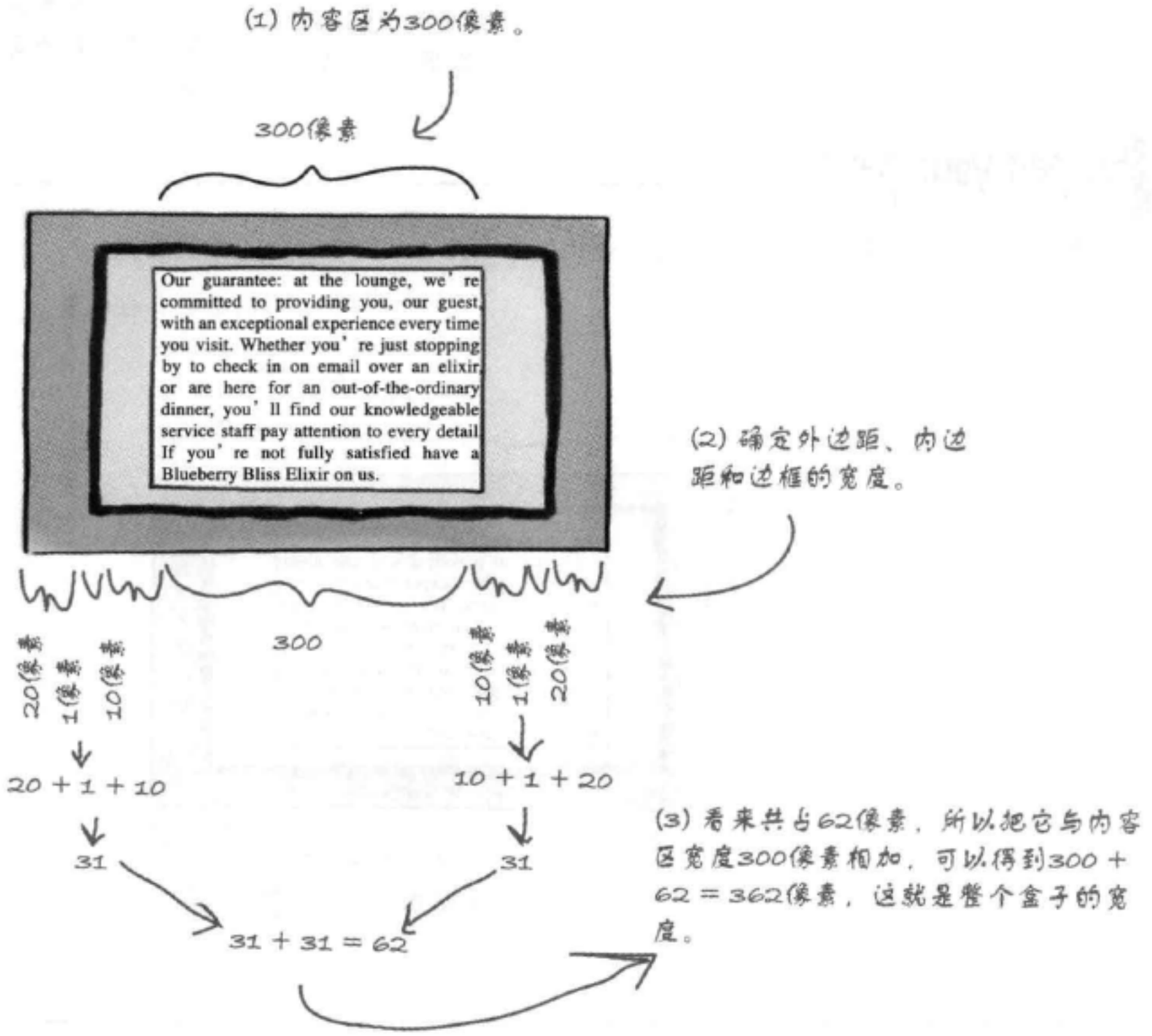
要确定整个盒子的宽度，需要将内容区的宽度加上左和右内边距、左和右外边距以及边框的宽度。不要忘了，边框宽度要加两次，因为左边和右边都有边框。



不用指定。你可以指定内容区、内边距、边框和外边距的宽度。所有这些加在一起就是整个元素的宽度。

假设在一个CSS规则中使用`width`属性将内容区宽度设置为300像素。

假设将外边距设置为20像素，内边距设置为10像素，另外有1像素宽的边框。这个元素盒子的宽度是多少？嗯，就是内容区的宽度，加上左和右内边距、左和右外边距以及左和右边框的宽度。下面来看如何计算……



there are no Dumb Questions

问：如果没有设置一个元素的宽度，从哪里得到它的宽度呢？

答：一个块元素的默认宽度是“auto”，这说明它会延伸占满可用的空间。考虑我们之前构建的Web页面，每个块元素都可以延伸占满浏览器的整个宽度，这正是“auto”在起作用。现在先记住这一点，因为我们会在下一章详细讨论这个内容。只要记住，“auto”允许内容填满可用的所有空间（考虑到内边距、边框和外边距之后）。

问：如果没有外边距、内边距或边框呢？

答：那么内容的宽度就是盒子的整个宽度。如果内容区的宽度是300像素，而且没有内边距、边框或外边距，那么整个盒子的宽度就是300像素。

问：指定宽度有多种不同方法，这些方法有什么区别？

答：你可以指定一个具体大小，通过按像素指定，或者也可以指定一个百分数。如果使用百分数，那么宽度会计算为元素所在容器宽度的一个百分比（容器可以是<body>、 等）。

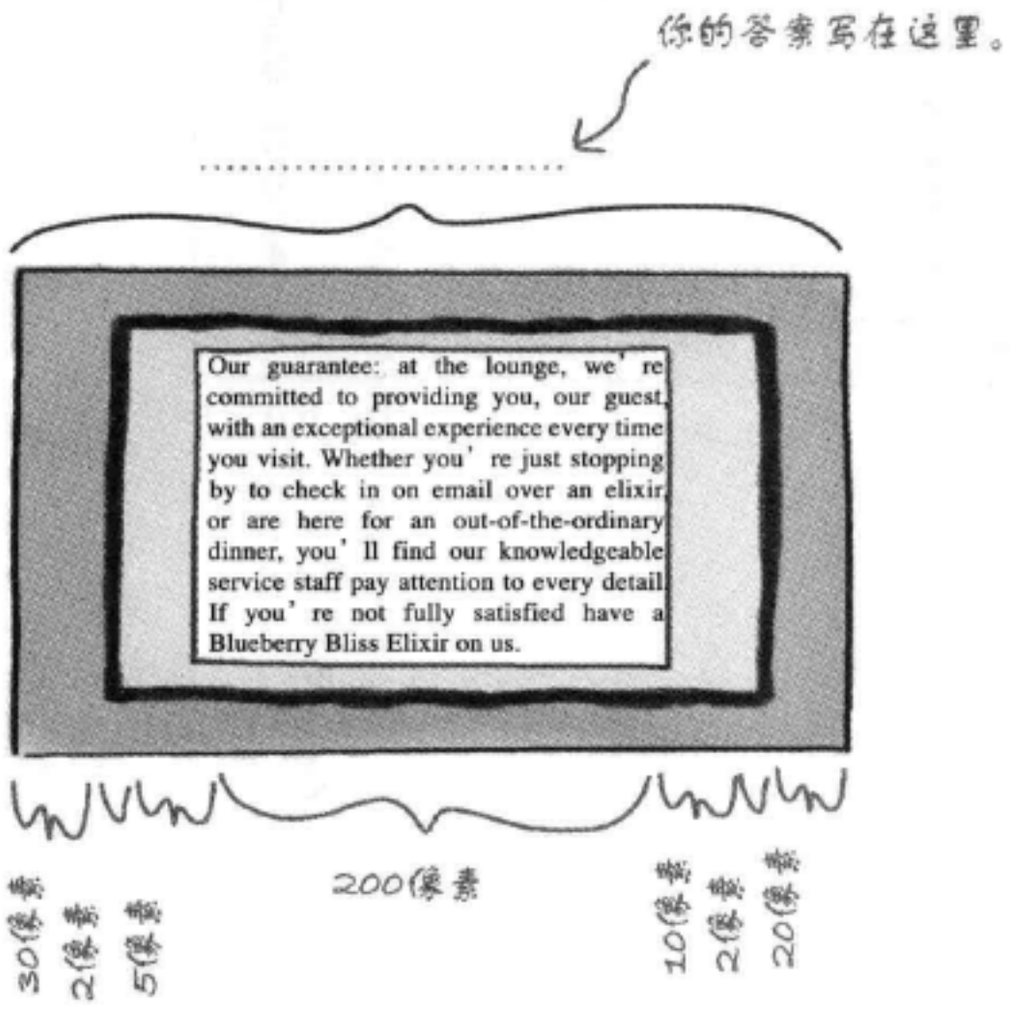
问：高度呢？

答：一般来讲，一个元素的高度是默认的，也就是auto（自动），浏览器在垂直方向上会延伸内容区，使所有内容都可见。将width设置为200像素之后，查看饮料区，你会看到这个 变得高多了。

可以显式地设置一个高度，不过这会有风险，如果你指定的高度不够大，不足以放下内容，内容底部有可能“溢出”到其他元素中。一般地，不用指定元素的高度，就让它们默认为auto。

Sharpen your pencil

这里有一个盒子，已经标出的所有宽度。整个盒子的宽度是多少？



为elixirs增加基本样式

我们已经解决了宽度问题。还要做什么？

- [x] 首先，要改变elixirs 的宽度，让它更窄一些。
- [ ] 接下来，处理你熟悉的一些样式，如内边距和背景图像。我们还会处理文本对齐，这方面的内容你之前还没有见过。
- [ ] 然后只剩下文本行高和标题颜色了。你会看到需要稍稍提升你的CSS选择器技能，才能完成这些改变。

接下来要完成这一步。

现在来重点强调一些基本样式，如内边距和文本对齐，还要在elixirs 中加上鸡尾酒杯背景图像。其中大部分内容你已经很熟悉了，下面来简单看一下CSS：

要记住，你要将所有这些样式应用到elixirs ，使它们只影响 及其中包含的元素，而不是整个页面。

```
#elixirs {
   border-width:       thin;
   border-style:       solid;
   border-color:       #007e7e;
   width:              200px;

   padding-right:      20px;
   padding-bottom:     20px;
   padding-left:       20px;

   margin-left:        20px;

   text-align:         center;

   background-image:   url(images/cocktail.gif);
   background-repeat:  repeat-x;
}
```

的默认内边距为0像素，所以我们要增加一些内边距，为内容提供一点空间。注意，是边没有增加内边距，因为那里空间已经足够大了，这要归功于<h2>标题的默认外边距（返回去查看上一个测试，你会看到<h2>上面有很大空间）。不过在右边、下边和左边确实需要加内边距。

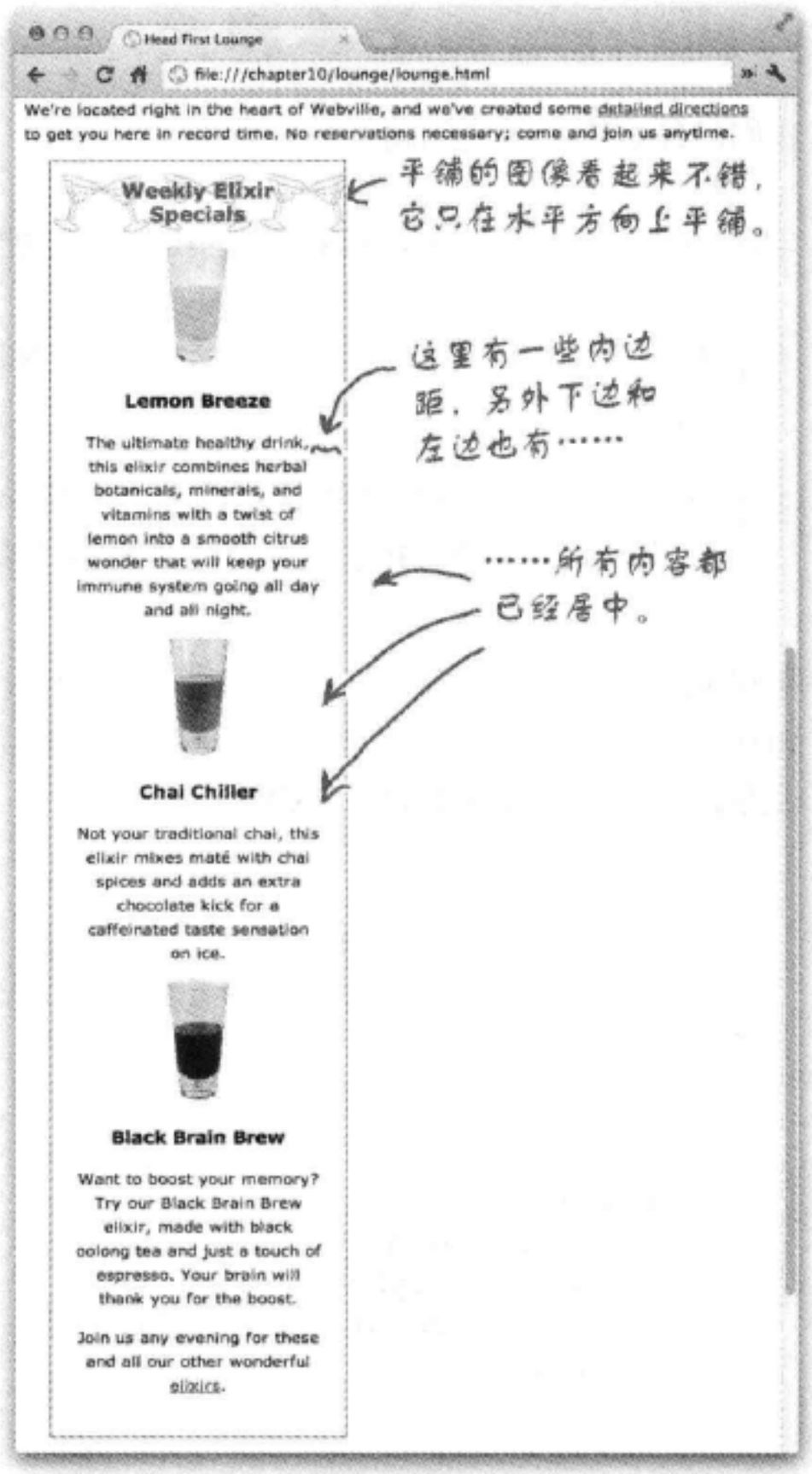
我们在左边增加了一些外边距，使elixirs相对于页面其他内容稍有些缩进。稍后会看到这很有用……

使用块元素的text-align来对齐其中包含的文本。这里我们打算让文本居中。

最后指定在背景中使用的一个图像，在这里就是鸡尾酒图像。我们将background-repeat属性设置为repeat-x，这使得图像只在水平方向上平铺。

测试新样式

现在为你的“lounge.css”文件增加这些新属性，并重新加载页面。下面来看有哪些改变： 中的标题、图像和文本都居中了，而且因为现在有一些内边距，所以有了更多的呼吸空间。我们还在最上面利用平铺的鸡尾酒图像增加了一点装饰。



问得好……这看起来有点不对劲，是不是？不过事实上，text-align会对块元素中的所有内联内容对齐。所以，我们对 块元素设置了这个属性后，它的所有内联内容都会居中。只是要记住，尽管这个属性的名字是text-align，但实际上它适用于对任何类型的内联元素对齐。还要记住一点：text-align属性只能在块元素上设置。如果直接在内联元素（如<img>）上使用，则不起作用。

目光真敏锐。 元素中的所有文本都在嵌套块元素中，不过它们现在都对齐了。这是因为，这些块元素继承了 的text-align属性。所以，这是有区别的，并不是 本身将标题和段落中的文本对齐（它不会这么做，因为标题和段落是块元素），实际上是标题和段落继承了text-align值“center”，然后是它们自己将内容居中。

那又怎样？嗯，可以想想看，这就为你提供了很多使用 的方法，因为你可以用 包围一些内容，然后对 应用样式，而不是对单个元素分别应用样式。当然要记住，默认地，并不是所有属性都能继承，所以这一点并不适用于所有属性。

Sharpen your pencil

既然你已经了解了宽度，那么elixirs盒子的总宽度是多少？首先，我们知道内容区宽度是200像素。另外我们还设置了一些左和右内边距，这会影响宽度，还将边框宽度设置为“thin”。假设thin边框为1像素宽（大多数浏览器中都是这样）。外边距呢？我们设置了一个左外边距值，但是没有右外边距，所以默认地，右外边距为0像素。

以下是所有与宽度有关的属性。你的任务是确定elixirs 的总宽度。

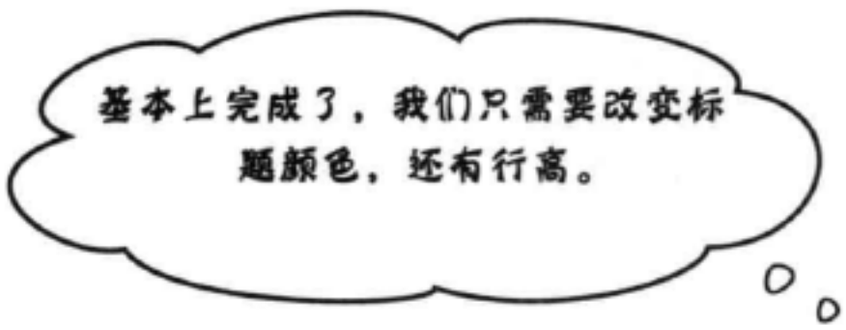
```
border-width:   thin;

width:          200px;

padding-right:  20px;
padding-left:   20px;

margin-left:    20px;
```

快完工了……

我们的饮料区就快要改造好了。还有什么没有做？

- [x] 首先，要改变elixirs 的宽度，让它更窄一些。
- [x] 接下来，处理你熟悉的一些样式，如内边距和背景图像。我们还会处理文本对齐，这方面的内容你之前还没有见过。
- [ ] 然后只剩下文本行高和标题颜色了。你会看到需要稍稍提升你的CSS选择器技能，才能完成这些改变。 ← 现在来完成最后一步。

听上去相当容易，是不是？毕竟，这些你以前都做过。实际上，只要你知道可以对 设置样式，而且这些样式会被其中的元素继承，这一步很快就能完成。



Frank：对，真有意思。饮料主标题<h2>颜色是青绿色，因为CSS中已经有一个<h2>规则。不过我们希望它是黑色。然后还要让饮料区中的<h3>变成红色。

Jim：嗯，没问题，我们只需要再加几个规则。

Frank：不过请等等……如果我们改变<h2>规则，或者增加一个<h3>规则，就会改变整个页面上的标题颜色。但现在我们只是想改变饮料区的这些颜色。

Jim：噢，对啊。嗯……我们可以使用两个类。

Frank：看来可以，不过有点麻烦。只要elixirs 中需要一个新标题，就得记得把它增加到类中。

Jim：对，生活就是这样的。

Frank：实际上，Jim，在你使用类之前，应该先看一下子孙选择器，我想它们用在这里更适合。

Jim：子孙选择器？

Frank：对，这也是一种指定选择器的方法，就像“选择elixirs 中的一个<h2>元素”。

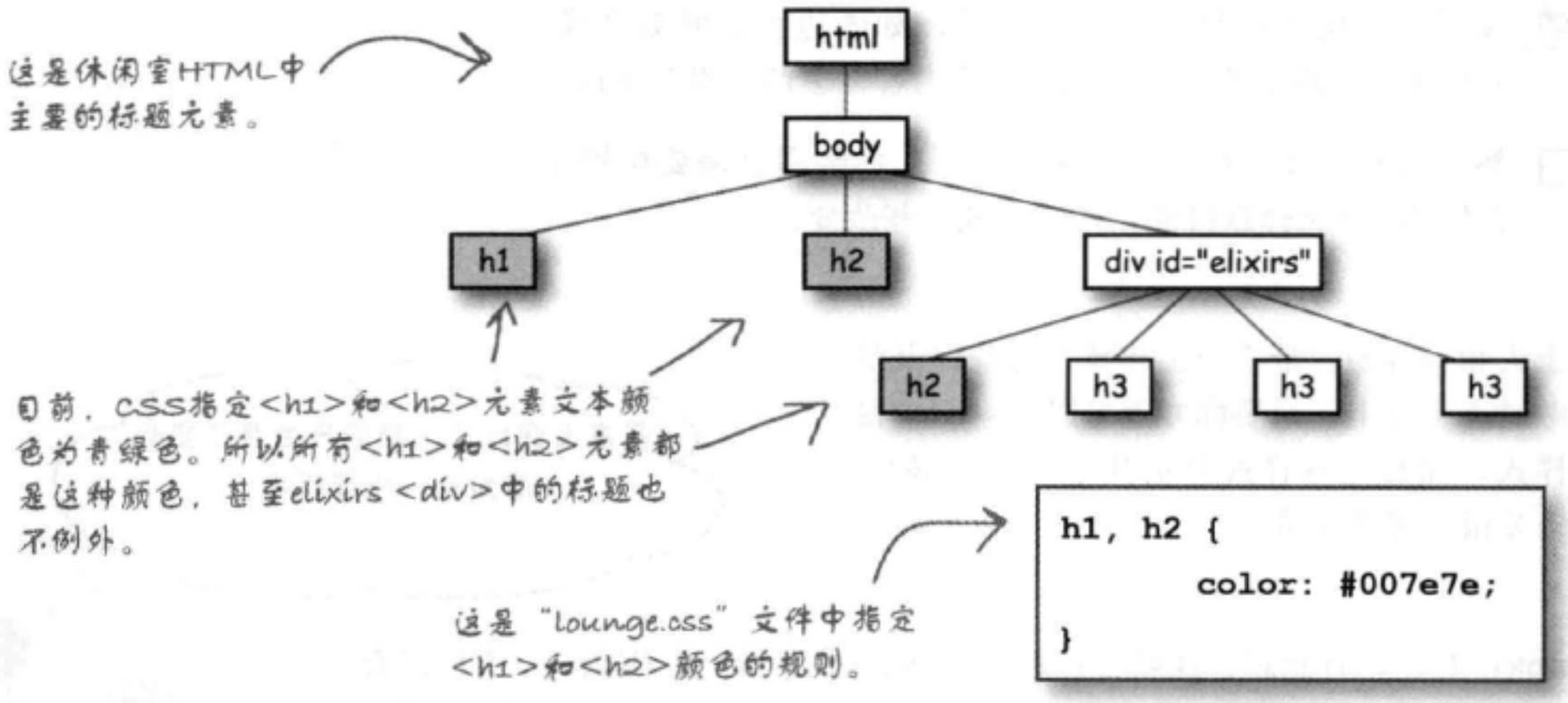
Joe：我不明白。

Frank：没关系，我们一步一步解释……

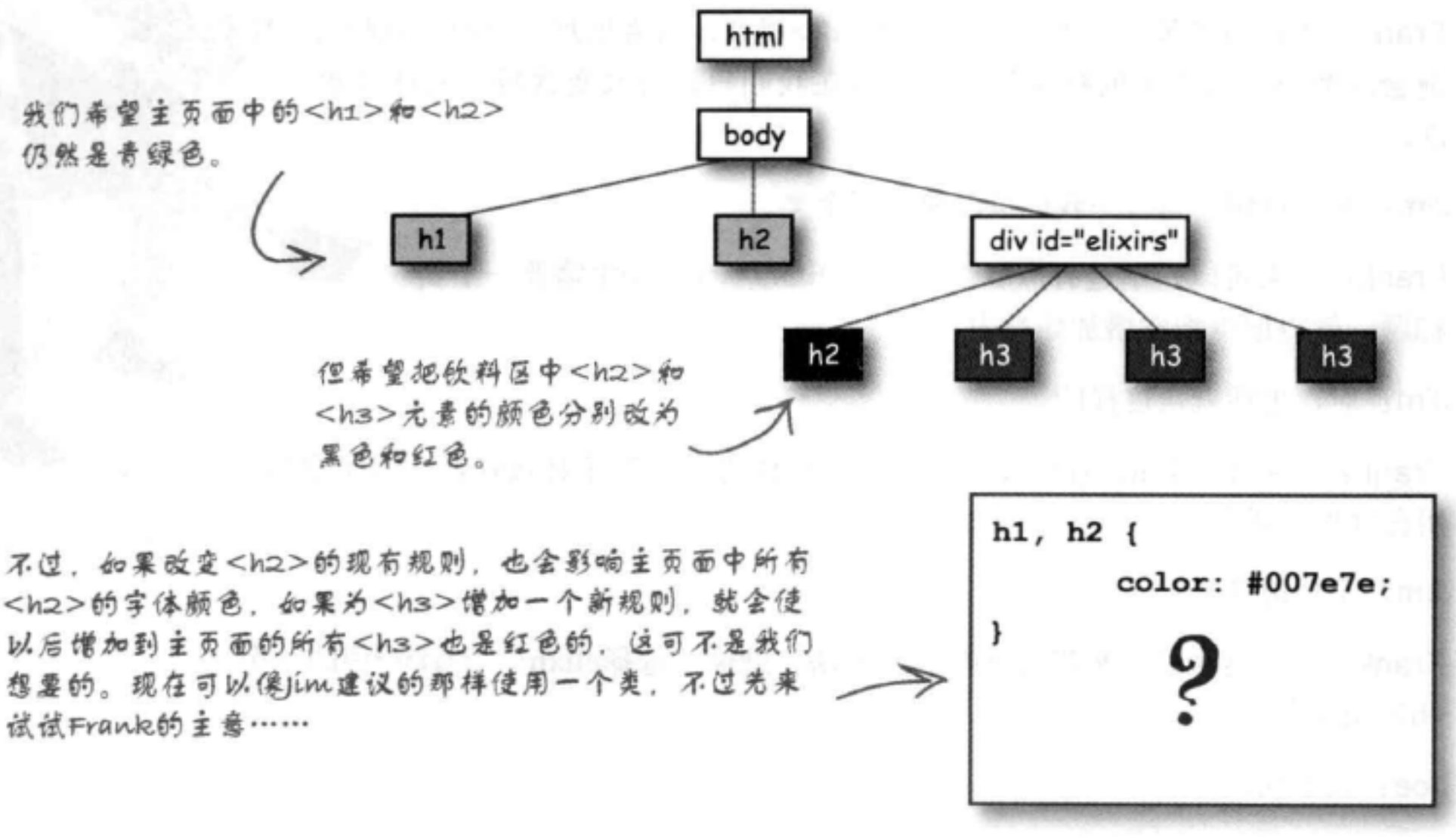
我们的目标是什么？

下面来简单看一下需要如何处理标题颜色。

我们现在有什么

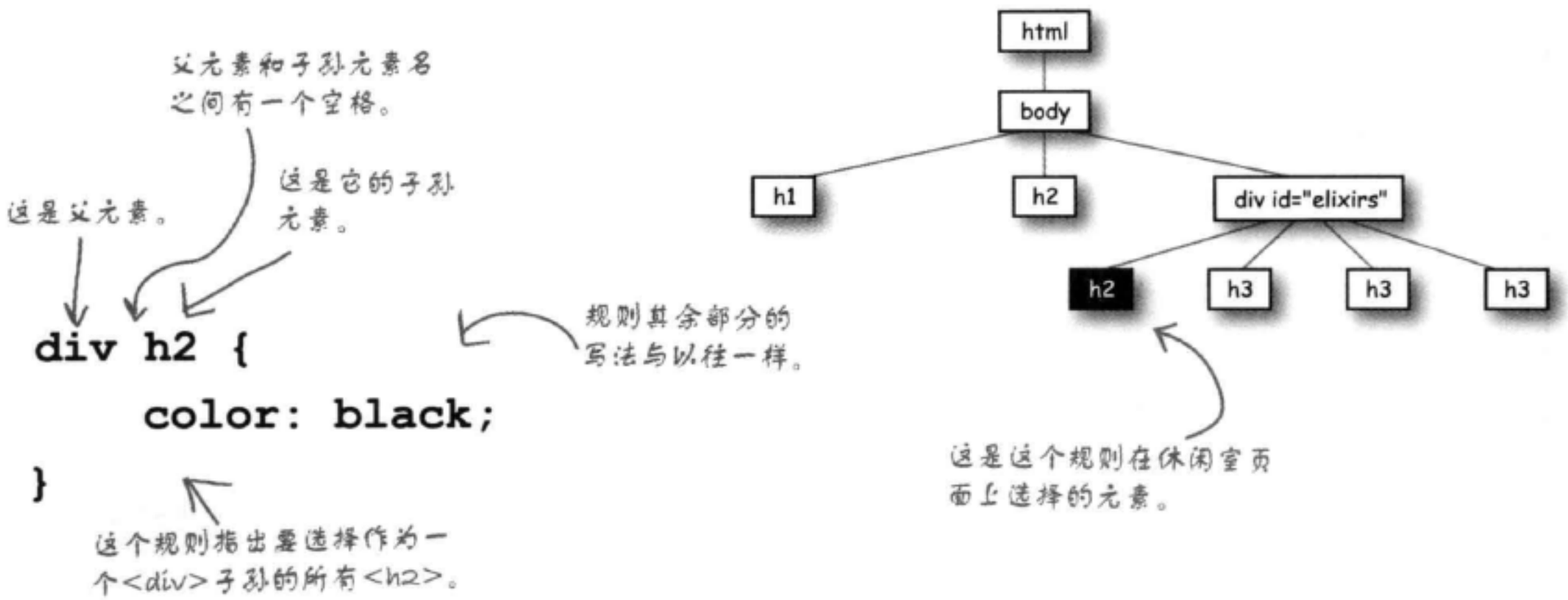


我们想要有什么

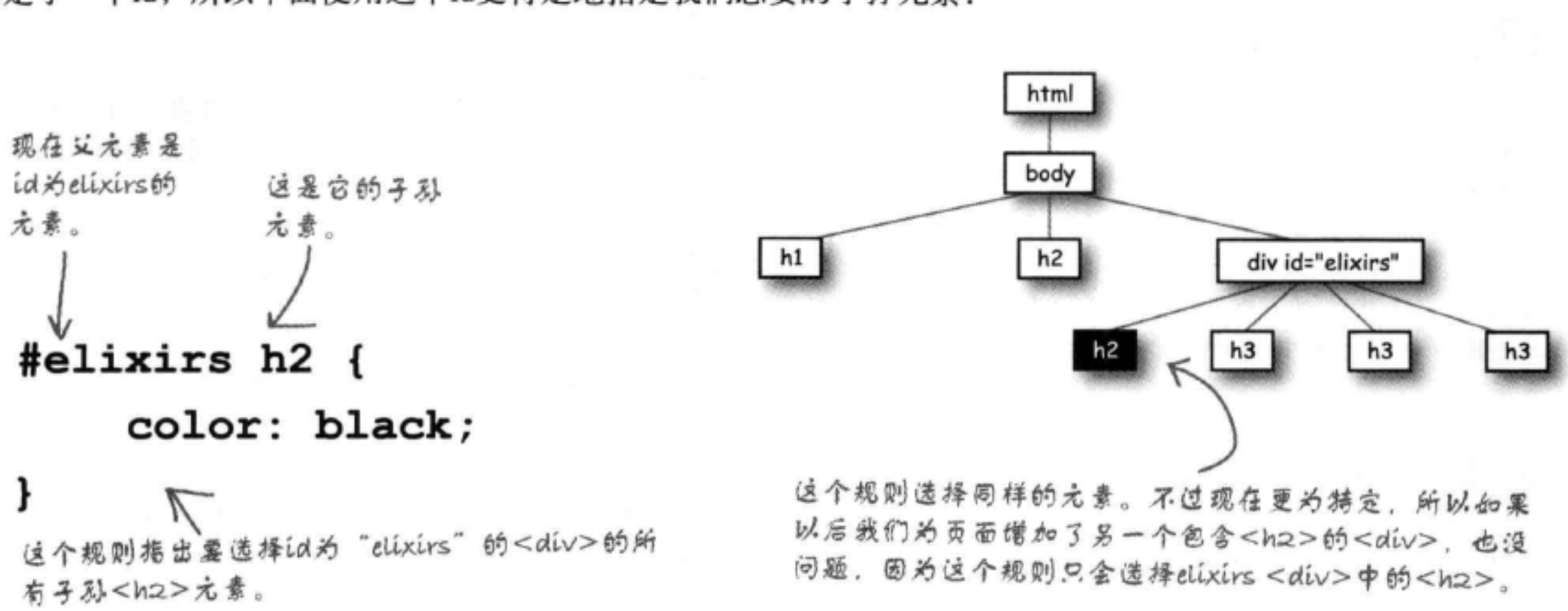


我们需要一种选择子孙的方法

我们真正想要的是能有一种方法告诉CSS：我们只想选择某些元素的子孙元素，这有点像指定你的遗产只能由你的一个女儿或儿子的孩子们继承。子孙选择器的写法如下所示。



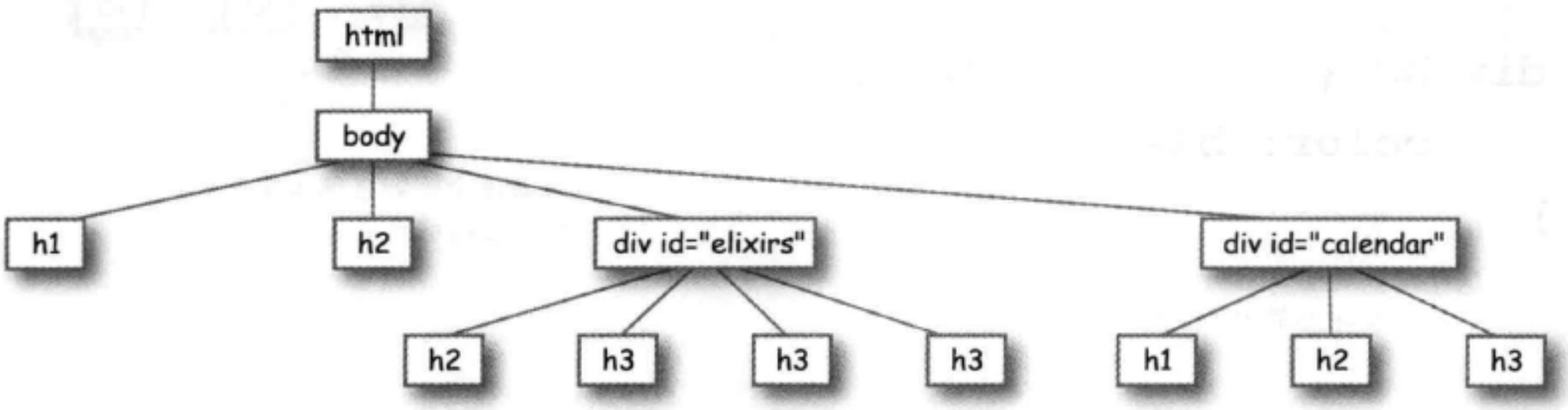
现在这个规则还有一个问题，如果有人在“lounge.html”文件中创建了另一个 ，她就会得到黑色的<h2>文本，尽管她可能并不想这样。不过，我们已经为elixirs 指定了一个id，所以下面使用这个id更特定地指定我们想要的子孙元素：



Sharpen your pencil

该轮到你了。请写出选择器，只选择elixirs 中的<h3>元素。在你的规则中，将color属性设置为#d12c47。另外在下面的图中标出选择的元素。

你的CSS规则写在这里。



there are no Dumb Questions

问：子孙一般表示孩子、孙子、曾孙子。不过这里只选择了孩子，是吗？

答：问得好。选择器“#elixirs h2”表示选择的<h2>元素可以是elixirs的所有子孙，所以这个<h2>可以是 的直接孩子，也可以嵌套在一个<blockquote>或另一个嵌套的 中（这就成为一个孙子），依此类推。所以子孙选择器会选择一个元素中嵌套的所有<h2>，而不论它嵌套得有多深。

问：嗯，有没有一种办法选择直接的孩子？

答：有。例如，你可以使用“#elixirs>h2”，这样一来，只有当<h2>是一个id为“elixirs”的元素的直接孩子时，才会选择这个<h2>。

问：如果我需要更复杂的选择呢？比如要选择一个<h2>，要求它是一个<blockquote>的孩子，而且<blockquote>必须在elixirs中，可以做到吗？

答：方法还是一样的。只需要使用更多子孙选择器，如下所示：

```
#elixirs blockquote h2 {
    color: blue;
}
```

这会选择<blockquote>下一级的<h2>元素，而且这个<blockquote>是一个id为“elixirs”的元素的子元素。

改变饮料标题的颜色

既然了解了子孙选择器，下面将饮料区中的<h2>标题设置为黑色，将<h3>标题设置为红色。需要这样做：

```
#elixirs h2 {
    color: black;
}

#elixirs h3 {
    color: #d12c47;
}
```

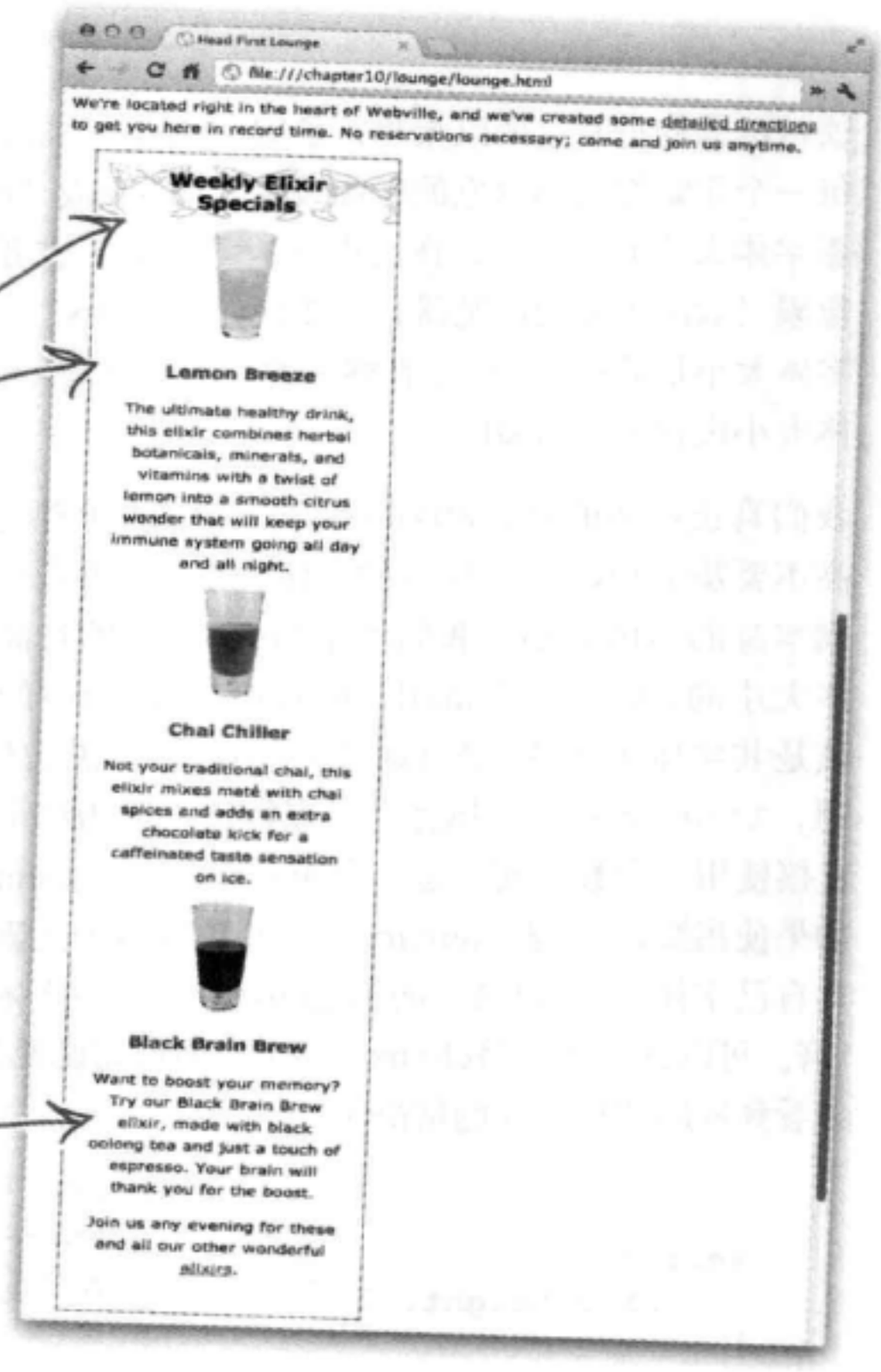
在这里使用子孙选择器来指定只选择elixirs 中的<h2>和<h3>元素。我们使用十六进制码将<h2>设置为黑色，将<h3>设置为红色。

快速测试……

将这些新属性增加到“lounge.css”文件的最下面，保存，并重新加载“lounge.html”。

现在饮料区中有了黑色和红色的标题，而且对主页面中的<h2>标题没有影响，主页面标题还是青绿色。

现在要做的就是修正行高。



修正行高

应该记得，上一章中我们将休闲室文本的行高设置得比正常稍高一点。这样看起来很不错，不过在饮料区中我们希望文本仍采用正常的行高，与宣传单一致。听上去很容易，是不是？只需要设置 的line-height属性就一切OK了，因为行高会继承。现在只有一个问题，标题也会继承这个行高，最后就会得到这样的结果：

```
#elixirs {
    line-height: 1em;
}
```

如果设置整个 的line-height属性，那么 中的所有元素都会继承这个属性，包括标题。可以注意到，标题中的行高太小了，这两行挤到一起了。

饮料标题的行高之所以太小，是因为elixirs 中的每一个元素都会继承它的行高设置，即1em或“elixirs元素字体大小的一倍”，在这里就是“small”或者大约12像素（取决于你的浏览器）。要记住，elixirs 的字体大小是从<body>元素继承的，而<body>元素的字体大小设置为“small”。

这些是元素的字体大小。我们将body的字体大小设置为“small”，所以elixirs也会继承这个字体大小。

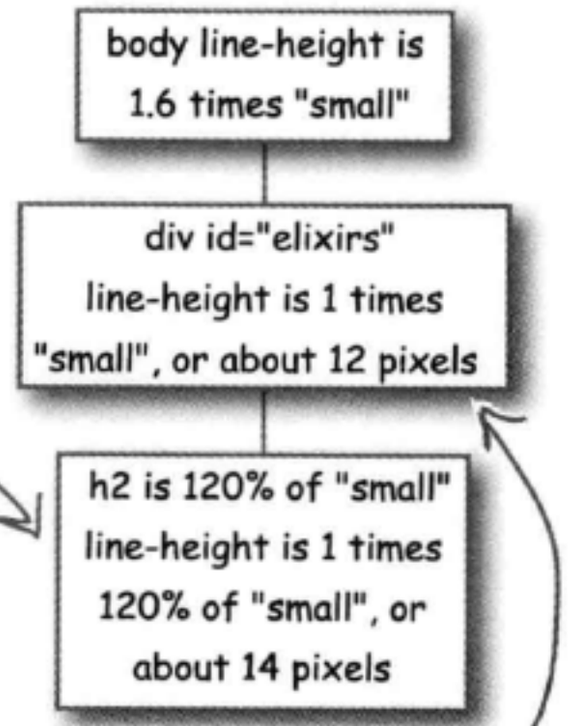
<h2>的line-height设置为elixirs字体大小的1倍，也就是“small”或大约12像素。

我们真正希望的是，elixirs 中的所有元素的行高不要基于elixirs 的字体大小，而要基于各个元素本身的字体大小。我们希望<h2>标题的行高为其字体大小的1倍（即“small”的120%），<p>的行高应该是其字体大小的1倍（即“small”）。怎么做到呢？嗯，line-height属性有一点特殊，因为你可以对它直接使用一个数，而不是一个相对度量（比如em或%）。如果使用数1，就表示elixirs 中的各个元素行高是其自己字体大小的1倍，而不是elixirs 字体大小的1倍。可以试一下，将elixirs 的行高设置为1，你会看到标题的行高问题解决了。

```
#elixirs {
    line-height: 1;
}
```

为elixirs 增加line-height为1，改变其中各个元素的line-height。

我们希望<h2>的line-height是它自己字体大小的1倍，也就是14像素（small的120%）。



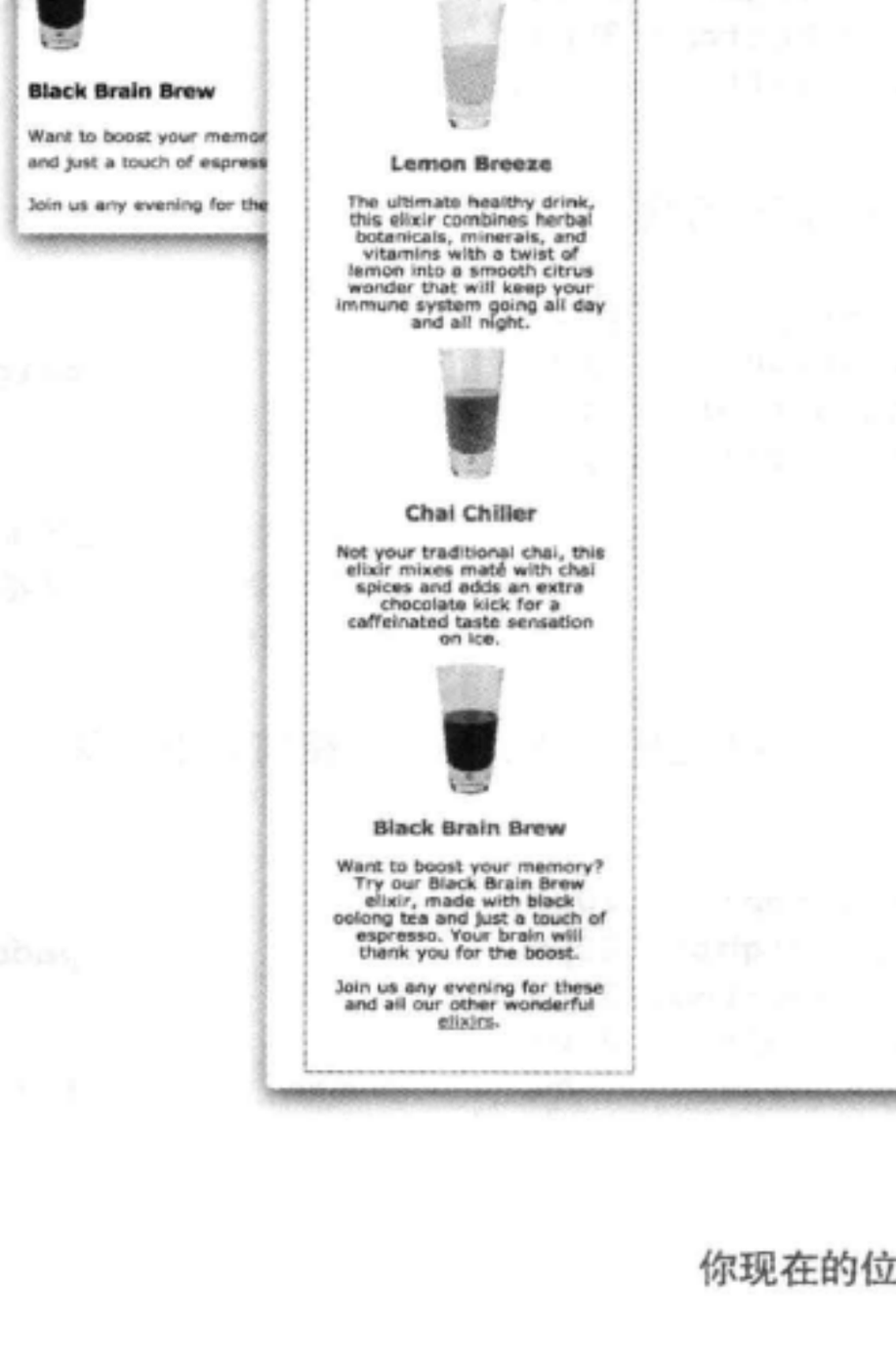
p元素的font-size是“small”（p继承了elixirs 的font-size），所以它的line-height为12像素，这正是我们想要的。

看看我们的成果……

来看我们的饮料区。它已经完成了“变身”，现在看起来与宣传单相差无几了。另外，除了为HTML增加了一个 和一个id属性外，我们只是增加了一些CSS规则和属性就做到了。

现在你应该认识到CSS有多么强大，另外将结构（HTML）与表现（CSS）分离之后，你的Web页面是多么灵活。只需要改变CSS，就可以让HTML有一个全新的面貌。

记得吧，这正是我们开始时展示的饮料区……



……这是我们现在得到的。

来点快捷方式

你可能已经注意到，看起来CSS属性太多了。例如，padding-left、padding-right、padding-bottom和padding-top。外边距属性也一样。另外还有background-image、background-color和background-repeat呢？它们可以为元素背景设置不同的属性值。另外你注意到了吗？输入这些属性确实有些枯燥。最好把时间花在更值得的事情上，而不是无聊地敲键盘，是不是？

```
padding-top:    0px;
padding-right:  20px;
padding-bottom: 30px;
padding-left:   10px;
```

只为了指定4个数，就要输入这么多字符。

嗯，这一章会给你一个特殊奖励。你会学到如何不费吹灰之力就能指定所有这些值。方法如下：

原来要这样指定内边距。

```
padding-top:    0px;
padding-right:  20px;
padding-bottom: 30px;
padding-left:   10px;
```

这是新的改进方式，把它们写为一个简写形式。

```
padding: 0px 20px 30px 10px;
```

可以对外边距使用同样的简写：

```
margin-top:    0px;
margin-right:  20px;
margin-bottom: 30px;
margin-left:   10px;
```

```
margin: 0px 20px 30px 10px;
```

与内边距类似，通过使用简写，只用一个属性就可以指定所有外边距值。

如果所有四个边上的内边距或外边距值都相同，还可以更简短：

```
padding-top:    20px;
padding-right:  20px;
padding-bottom: 20px;
padding-left:   20px;
```

```
padding: 20px;
```

这说明，盒子四边的内边距都是20像素。

如果所有内边距值都相同，可以这样写。

不过还不只这些……

要简写外边距（或内边距）还有另外一种常用方法：

上下是一样的。

```
margin-top:    0px;
margin-right:  20px;
margin-bottom: 0px;
margin-left:   20px;
```

左右也相同。

上下外边距以及左右外边距分别都是一样的，所以可以使用这样的简写。

```
margin: 0px 20px;
```

上和下　左和右

我们提到的边框属性呢？也可以对它们使用简写。

```
border-width:  thin;
border-style:  solid;
border-color:  #007e7e;
```

将边框属性重写为一个属性。可以采用你喜欢的任何顺序。

```
border: thin solid #007e7e;
```

边框简写比外边距或内边距更灵活，因为你可以按你喜欢的任何顺序来指定。

这些都是合法的边框简写形式。

```
border: solid thin #007e7e;

border: #007e7e solid thin;

border: solid thin;

border: #007e7e solid;

border: solid;
```

……不要忘记背景的简写

对背景也可以使用简写：

```
background-color:  white;
background-image:  url(images/cocktail.gif);
background-repeat: repeat-x;
```

类似于边框，简写中这些值可以采用任何顺序，还可以指定另外一些值，如background-position。

```
background: white url(images/cocktail.gif) repeat-x;
```

更多简写形式

如果没有提到字体的简写，关于简写形式的介绍肯定不全面。可以看看字体需要的所有属性：font-family，font-style，font-weight，font-size，font-variant，另外不要忘记line-height。嗯，有一个简写可以将所有这些属性包装成一个属性。方法如下：

```
font: font-style font-variant font-weight font-size/line-height font-family
```

这是字体简写中的属性。这里顺序并不重要，只不过……

必须指定字体大小。

最后，需要增加字体系列。只需要指定一个字体（必要），不过强烈建议指定一些候选字体。

这些值都是可选的。可以指定这些属性的任意组合，不过它们必须出现在font-size属性前面。

line-height属性是可选的。如果你想指定一个行高，只需要在font-size属性后面加一个/，然后指定你想要的行高。

font-family名之间要使用逗号分隔。

下面就来试一试。以下是休闲室页面body元素的字体属性：

```
font-size: small;
font-family: Verdana, Helvetica, Arial, sans-serif;
line-height: 1.6em;
```

现在把它们对应到简写形式中：

我们没有使用这些属性，不过这没关系，它们都是可选的。

```
font: font-style font-variant font-weight font-size/line-height font-family
```

现在写出最后的简写形式：

```
font: small/1.6em Verdana, Helvetica, Arial, sans-serif;
```

这就是简写版本。哇，确实是简写，是不是？现在你就能腾出时间来爬山了（或者去海边度假）。

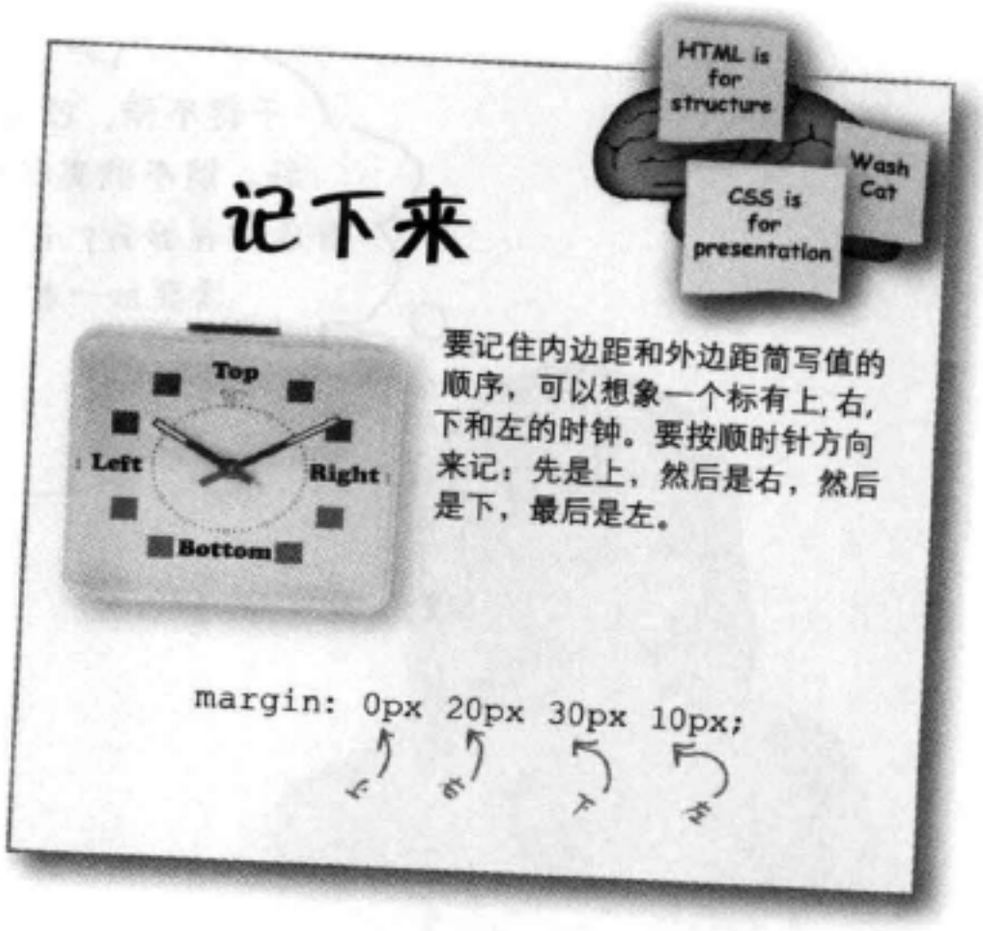
there are no Dumb Questions

问：一定要用简写形式吗？

答：没必要。有些人觉得长形式更可读。简写确实有一些好处，可以缩小CSS文件的大小，当然输入会更快，因为现在需要输入的内容少多了。不过，如果存在问题，倘若有不正确的值，或者顺序有误，简写形式会比较难“调试”。所以，你要使用更适合的形式，因为它们都是完全合法的。

问：简写形式更复杂，因为我必须记住顺序，还要知道哪些可选哪些不可选。怎么才能记住呢？

答：嗯，你会惊奇地发现它很快会成为你的第二本能，不过，我们这个“行当”的人都有一个小秘密，我们喜欢找一个“参考手册”。如果你需要快速查找属性名或一个属性的语法，可以找一本简明的参考手册，查出你想要的信息。我们特别喜欢Eric Meyer的《CSS Pocket Reference》。这是一个非常好的参考手册，而且短小精悍。



现在要学以致用，让你掌握的新知识发挥作用。我们注意到休闲室页面最下面有一个很小的版权信息部分，它要作为这个页面的页脚。增加一个 让它单独成为一个逻辑区。完成之后，再用以下属性为它指定样式：

```
font-size: 50%;
text-align: center;
line-height: normal;
margin-top: 30px;
```

将文本设置得很小。你知道的，就是那种极小的字体。

将文本居中。

还要将line-height设置为“normal”，这是之前没有见过的一个关键字。“normal”允许浏览器选择一个适当的行高大小，通常要根据字体来确定。

增加一些上外边距，为页脚提供一些呼吸空间。

完成后，再来看看整个“lounge.css”文件。有没有哪些地方你希望用简写来简化？如果有，请完成修改。

另一个休闲室任务

休闲室常驻DJ。

What's playing at the Lounge

We're frequently asked about the music we play at the lounge, and no wonder, it's great stuff. Just for you, we keep a list here on the site, updated weekly. Enjoy.

- *Buddha Bar*, **Claude Challe**
- *When It Falls*, **Zero 7**
- *Earth 7*, **L.T.J. Bukem**
- *Le Roi Est Mort, Vive Le Roi!*, **Enigma**
- *Music for Airports*, **Brian Eno**

所有CD名都用斜体字体风格显示。

所有艺术家都用粗体显示。

BRAIN POWER

要对“What's playing at the Lounge”部分中的CD和艺术家指定样式，你认为最佳的方法是什么？

Frank：是的，不过这有点像只是为了缩进文本而使用<blockquote>。我的意思是说，我们并不想强调和重点强调CD和艺术家，只是想使用斜体和粗体。另外，要是有人改变了和的样式呢？如果是这样，CD和艺术家也会采用新样式显示了。

Jim：嗯，我考虑过这个问题，不过我实在想不出还有什么别的办法。我是说，它们是同一个列表项中的文本。好像没有办法分别指定样式吧。

Frank：什么意思？

Jim：我们只能对元素指定样式，这里只有一些文本，比如“Music for Airports, Brian Eno”。我们需要一个元素来包围每一部分文本，才能对它们分别指定不同的样式。

Frank：噢，对，对，我明白你的意思了。

Jim： 我想可以这样做：

```
<div class="cd">"Music for Airports</div>
<div class="artist">Brian Eno</div>
```

不过这是一个块元素，所以会带来换行。

Frank：哈，我觉得找到切入点了，Jim。还有一个类似于 的元素，不过是针对内联元素的。这个元素叫做。它能完美地解决这个问题。

Jim：我不明白。它是怎么做的？

Frank：嗯，可以利用创建内联字符和元素的逻辑分组。我们可以试一试……

只需简单的3步来增加

允许你为块级内容创建逻辑划分，元素则采用类似的方式建立内联内容的逻辑分组。要搞清楚它的用法，下面来为音乐推荐部分增加样式。首先我们要增加元素包围CD和艺术家，然后写两个CSS规则指定这些的样式。你需要做到：

1. 将CD和艺术家嵌在单独的元素中。
2. 将一个增加到“cd”类，另一个增加到“artist”类。
3. 要创建一个规则为“cd”类指定斜体样式，另外创建一个规则指定“artist”类为粗体。

第1步和第2步：增加

打开你的“lounge.html”文件，找到“Who's playing at the Lounge”标题。在它下面，可以看到一个推荐音乐无序列表。现在列表是这样的：

每个列表项包含一个CD名，一个逗号，然后是艺术家。

```
<ul>
<li>Buddha Bar, Claude Challe</li>
<li>When It Falls, Zero 7</li>
<li>Earth 7, L.T.J. Bukem</li>
<li>Le Roi Est Mort, Vive Le Roi!, Enigma</li>
<li>Music for Airports, Brian Eno</li>
</ul>
```

下面为第一个CD和艺术家增加：

增加一个开始标记，并指定class属性和“cd”值。

接下来，在CD名后面增加一个结束标记。

对艺术家做同样的处理。把艺术家嵌在一个元素中，只不过这一次要把这个放在“artist”类中。

```
<ul>
<li><span class="cd">Buddha Bar</span>, <span class="artist">Claude Challe</span></li>
<li>When It Falls, Zero 7</li>
<li>Earth 7, L.T.J. Bukem</li>
<li>Le Roi Est Mort, Vive Le Roi!, Enigma</li>
<li>Music for Airports, Brian Eno</li>
</ul>
```

第3步：指定的样式

继续学习后面的内容之前，保存这个文件，在浏览器中重新加载页面。类似于 ，也没有默认样式，所以应该看不到任何变化。

现在来增加一些样式。将这两个规则增加到“lounge.css”文件最下面：

我们要为这两个新类（cd和artist）分别增加一个规则。

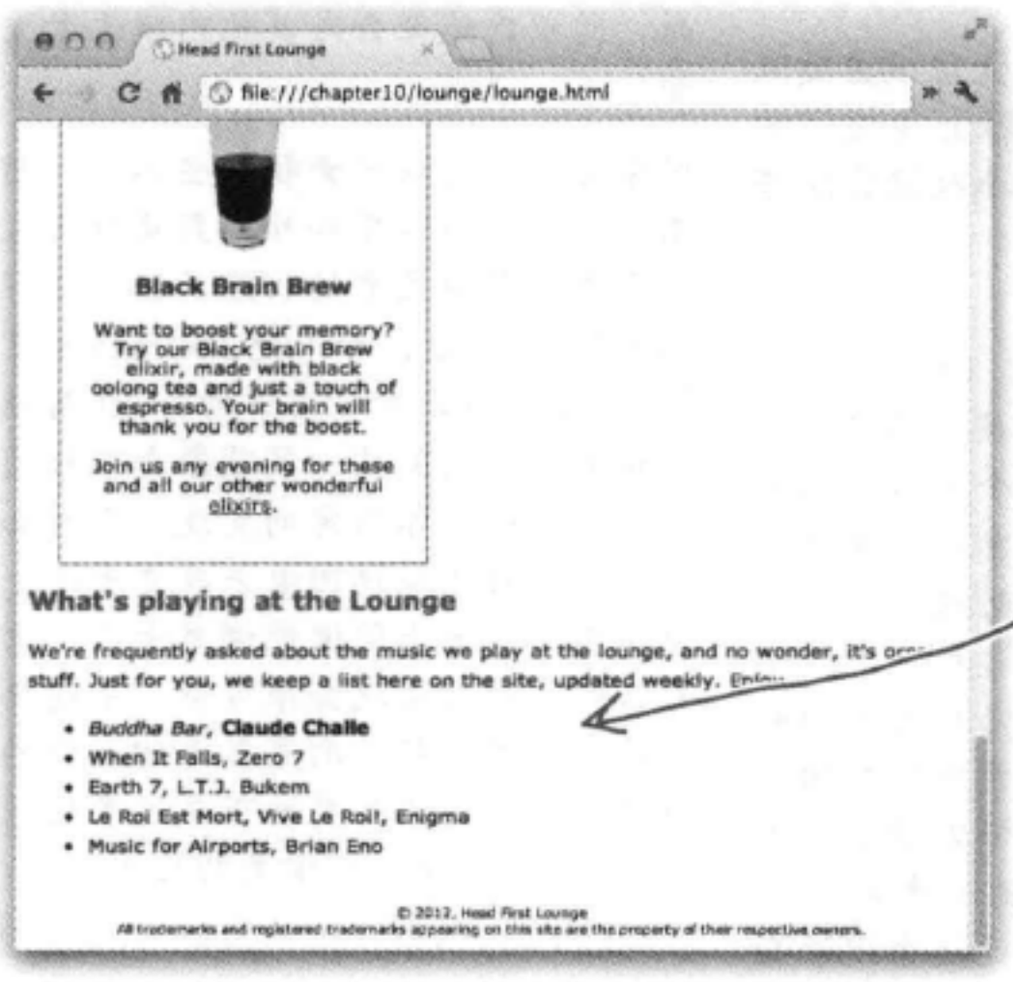
对于CD，要设置字体风格为斜体。

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

.artist {
        font-weight: bold;
}
```

对于艺术家，要设置font-weight为bold。

测试span

就这么简单。保存并重新加载。你会看到这样的结果。



现在第一个音乐推荐已经有了正确的样式。

Sharpen your pencil

你的任务是为其余音乐推荐增加元素，再测试你的页面。这一章最后会给出答案。

```
<ul>
<li><span class="cd">Buddha Bar</span>, <span class="artist">Claude Challe</span></li>
<li>When It Falls, Zero 7</li>
<li>Earth 7, L.T.J. Bukem</li>
<li>Le Roi Est Mort, Vive Le Roi!, Enigma</li>
<li>Music for Airports, Brian Eno</li>
</ul>
```

there are no Dumb Questions

问：为什么我要用而不是另外一个内联元素，比如或？

答：通常，你总希望用与内容含义最接近的元素来标记内容。所以，如果要强调某些文字，就可以使用。如果想强调一个重点，可以使用。不过，如果你想要的只是改变某些文字的样式，比如一个歌迷网站上的唱片名或艺术家名字，就应该使用，并把你的元素放在适当的类中，对它们分组并指定样式。

问：能不能对元素设置类似width之类的属性？实际上我想问的是，一般的内联元素能不能设置这些属性？

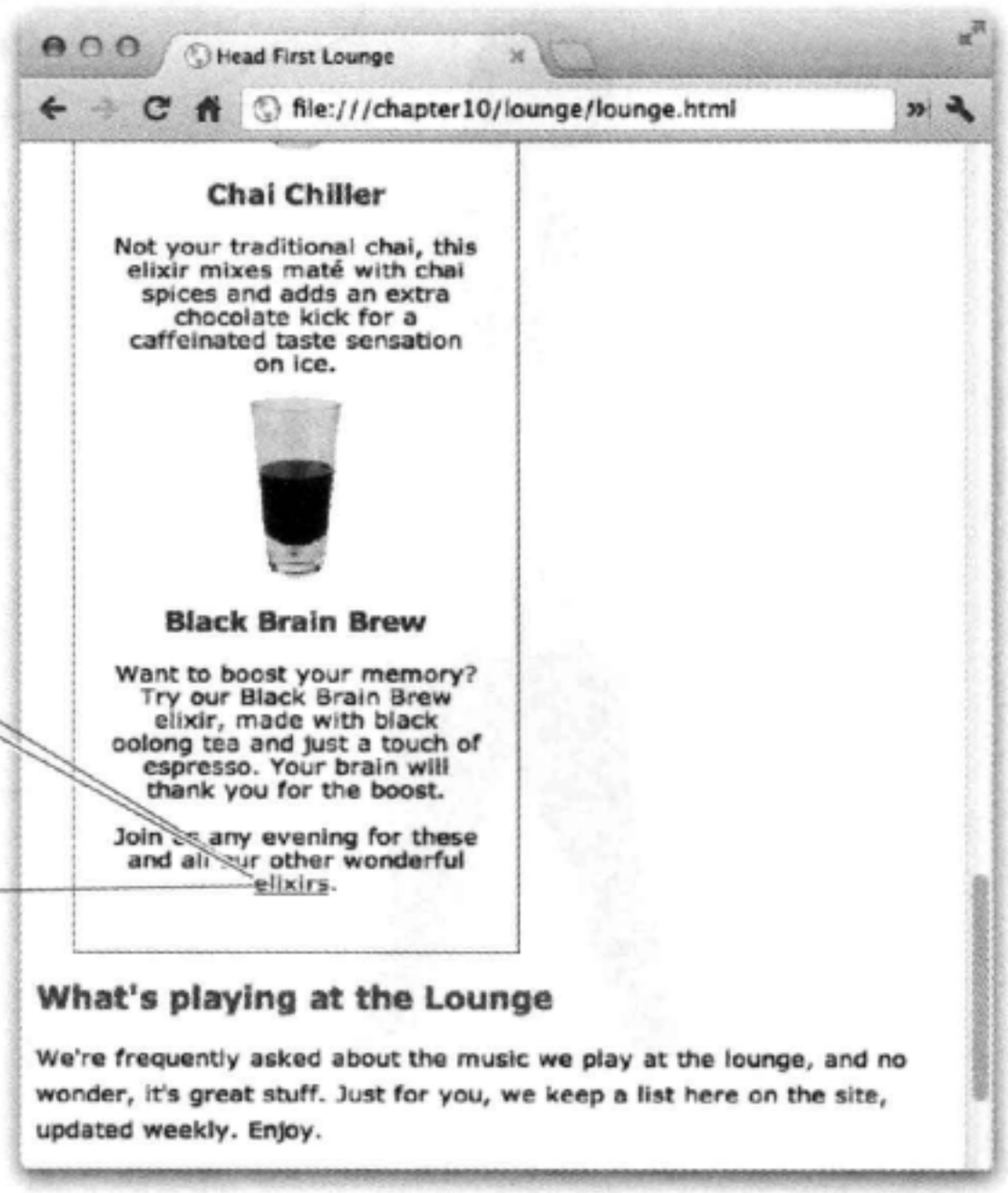
答：可以设置内联元素（如、和）的宽度，不过在对这些元素定位之前（这个内容将在下一章学习），你可能注意不到宽度改变的效果。另外还可以对这些元素增加外边距、内边距以及边框。内联元素上的外边距和内边距与块元素稍有不同，如果一个内联元素四周都增加外边距，只能看到左边和右边会增加空间。你也可以对内联元素的上边和下边增加内边距，不过这个内边距不会影响包围它的其他内联元素的间距，所以内边距会与其他内联元素重叠。

图像与其他内联元素稍有些不同。图像的宽度、内边距和外边距属性都表现得更像是块元素的相应属性。记得在第5章介绍过：如果使用<img>元素的width属性或者CSS中的width属性来设置图像的宽度，浏览器会缩放图像，使它适合你指定的宽度。有时如果你不能自己编辑图像来改变大小，但又希望页面上图像能调整大小，浏览器的这种做法就会很方便。不过要记住，如果依赖于浏览器来缩放你的图像，可能会不必要地下载过多的数据（如果图像大于你需要的尺寸）。

考虑一下<a>元素。它的样式与其他元素有没有不同的地方？

<a>元素和它的多重人格

你注意到了吗？在样式方面，链接的表现稍有些不同。链接是元素世界里的变色龙，取决于具体环境，它们会瞬间改变样式。下面来仔细看一下：



这里有一个你没有单击过的链接。这称为“未访问的链接”，或者就称为“链接”，默认地，这个链接是蓝色的。

Join us any evening for these and all our other wonderful elixirs.

Join us any evening for these and all our other wonderful elixirs.

这里是一个之前单击过的链接。我们把这种链接称为“已访问的链接”。通常，相对于未访问的链接，已访问的链接会用不同的颜色显示，以便区分二者的差别。大多数浏览器中，已访问的链接默认为紫色。

Join us any evening for these and all our other wonderful elixirs.

Head First Lounge Elixirs

如果把鼠标停在一个链接上面，但不单击，这称为“悬停”。在一些浏览器上，鼠标悬停在链接上时你会看到一个工具提示，它会显示“title”属性的文本，如果看得更仔细一些，会发现悬停在链接上时它有一种不同的样式。

与其他元素不同，<a>元素的样式会根据它的状态改变。如果这个链接从来没有单击过，它会有一种样式，如果已经单击过，则有另外一种样式。另外如果悬停在一个链接上，可能还有一种不同的样式。也许对<a>元素指定样式比你看到的更复杂，是吗？说对了……下面就来看一下。

如何根据元素的状态指定样式？

一个链接可以有多种状态：可能未访问、已访问或者处于悬停状态（还有另外几种状态）。那么，如何利用所有这些状态呢？例如，如果能为已访问和未访问的链接指定不同的颜色，或者用户悬停在一个链接上时能突出显示这个链接，肯定很棒。如果有办法……

嗯，当然有，如果我们告诉你这需要用到伪类，你可能会合上书，不想再读下去，觉得今天晚上已经学得够多了，是吗？不过坚持一下！就当我们从来没有说过伪类这个词，下面来看如何对链接指定样式：

注意这里有元素<a>，后面是一个:（冒号），然后是我们想选择的状态。要确保这些选择器中没有空格（例如，a:link是不行的）。

```
a:link {
    color: green;
}

a:visited {
    color: red;
}

a:hover {
    color: yellow;
}
```

这个选择器应用于处于未访问状态的链接。

这个选择器应用于已访问的链接。

悬停在一个链接上时会应用这个选择器。

Exercise

将这些规则增加到“lounge.css”文件最下面，然后保存并重新加载“lounge.html”。试着单击链接，并把鼠标悬停在链接上，看看它们的不同状态。需要说明，必须清空你的浏览器历史记录，才能看到未访问的链接的颜色（绿色）。

there are no Dumb Questions

问：如果我把<a>元素当成一个普通元素来指定样式，会怎么样呢？比如说：

```
a {  color: red; }
```

答：当然可以这么做，不过这样一来，你的链接在所有状态下看起来都一样，这会使你的链接无法做到用户友好，因为没办法区分哪些访问过，哪些没有访问过。

问：你提到的另外几种链接状态是什么？

答：还有另外两种状态：focus和active。浏览器将焦点放在你的链接上时就是焦点（focus）状态。这是什么意思？有些浏览器允许按下Tab键来轮流访问页面上的所有链接。浏览器访问到某个链接时，这个链接就拥有“焦点”。设置一个焦点伪类值对于提高可访问性很有帮助，因为需要使用键盘（而不是使用鼠标）来访问链接的人会知道他们何时选择到正确的链接。用户第一次单击一个链接时，就处于活动（active）状态。

问：难道链接不能同时处于多种状态吗？例如，我的链接可能已经访问过，而且鼠标悬停在它上面，另外用户可能正在单击它，这些情况可能同时发生。

答：当然可以。你要按规则的顺序来确定应用哪个样式。所以一般认为适当的顺序是：link, visited, hover, focus，然后是active。如果使用这个顺序，就能得到你期望的结果。

问：OK，我明白了。那么什么是伪类呢？

答：这只是CSS语言中最让人困惑的术语之一。不过，你已经看到，对链接加样式相当简单。所以下面来讨论伪类……

伪类闪亮登场

本周访谈:
来认识伪类

Head First：欢迎你，伪类（Pseudo-class）。很高兴能请到你。我得承认，他们刚开始让我做这个采访时，我大脑一片空白。伪类？我唯一能想到的是80年代Phil Collins的一首歌。

伪类：嗯，别记错了，那叫Sussudio，我的名字是Pseudo。

Head First：唉呀。真不是故意的。我们就从这里开始吧。你能多给我们讲讲这个Pseudo是从哪里来的吗？

伪类：Pseudo（伪）通常表示一个东西看上去像是真的，不过其实它不是。

Head First：那你的姓是怎么来的？类？

伪类：所有人都知道CSS类是什么。这就是你创建的一个分组，可以放入一些元素，这样就能对它们一起指定样式。把“pseudo”（伪）和“class”（类）加在一起就有了一个伪类：它就像一个类，但并不是真正的类。

Head First：既然它表现得像一个类，说它不是真正的类又是什么意思？

伪类：OK，打开一个HTML文件，找找类:visited或:link，或者:hover。找到了就告诉我一声。

Head First：我找不到啊。

伪类：不管怎么样，a:link、a:visited甚至a:hover都允许你指定样式，就好像它们是类一样。所以这些就是伪类。换句话说，你可以对伪类指定样式，但是没有人在HTML中真正输入这些伪类。

Head First：很好，那它们是怎么工作的呢？

伪类：这要归功于你的浏览器。浏览器会仔细检查所有<a>元素，把它们增加到正确的伪类中。如果一个链接已经访问过，没问题，它会放在:visited 伪类中。用户把鼠标悬停在一个链接上？没问题，浏览器会把这个链接扔进:hover 伪类中。噢，现在用户不再悬停了？浏览器又会把它从“hover”伪类中拽出来。

Head First：哇，真没想到。这么说这些类确实存在，浏览器会在后台向这些类增加和删除元素，是吗？

伪类：没错。一定要知道这一点，这非常重要；否则，浏览器又怎么根据状态为链接指定适当的样式呢？

Head First：那么，Pseudo，你是不是只能处理链接？

伪类：不，我还能处理其他元素。现代浏览器已经对其他类型的元素提供了类似:hover等伪类支持，另外还有一些其他的伪类。例如，伪类:first-child对应元素的第一个子元素，如<blockquote>中的第一个段落，甚至还可以用:last-child 伪类选择<blockquote>的最后一个段落。我很全面的，真的。

Head First：噢，这次采访真是让我学到了很多。谁知道那首真名叫“Sussudio”的歌?谢谢你接受我们的采访，伪类。

运用伪类

好了，我们还是实话实说。你刚刚学习了这本书里最重要的内容：伪类。为什么，不，绝对不是因为它们允许你根据浏览器的决定（确定元素属于哪些“类”）来指定这些元素的样式，比如:link或:first-child。另外，不，也不是因为它们为你提供了有效的方法，可以根据访问者使用页面时发生的情况来对元素指定样式，如:hover。真正的原因是：下一次你参加设计会议时，由于你对伪类有了透彻的理解并因此在会议上侃侃而谈时，所有人的目光肯定都会转向你。我的意思是，你会得到升职加薪……至少，你的同伴们将对你充满敬重与钦佩。

所以，下面就来好好运用这些伪类。你已经向你的“lounge.css”增加了一些伪类规则，它们对链接的外观确实有巨大影响，不过对于休闲室网站来说可能还不太合适。下面来对样式稍稍做些调整：

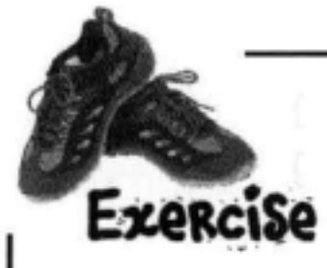
哈，这里变化很大。这里结合了一个伪类和一个子孙选择器。第一个选择器要选择嵌套在一个id为“elixirs”的元素中的所有未访问过的<a>元素。所以我们只是对elixirs中的链接指定样式。

```
#elixirs a:link {
    color: #007e7e;
}

#elixirs a:visited {
    color: #333333;
}

#elixirs a:hover {
    background: #f88396;
    color: #0d5353;
}
```

我们利用这两个选择器来设置颜色。对于未访问的链接，是漂亮的青绿色……

……对于已访问的链接，我们要使用深灰色。

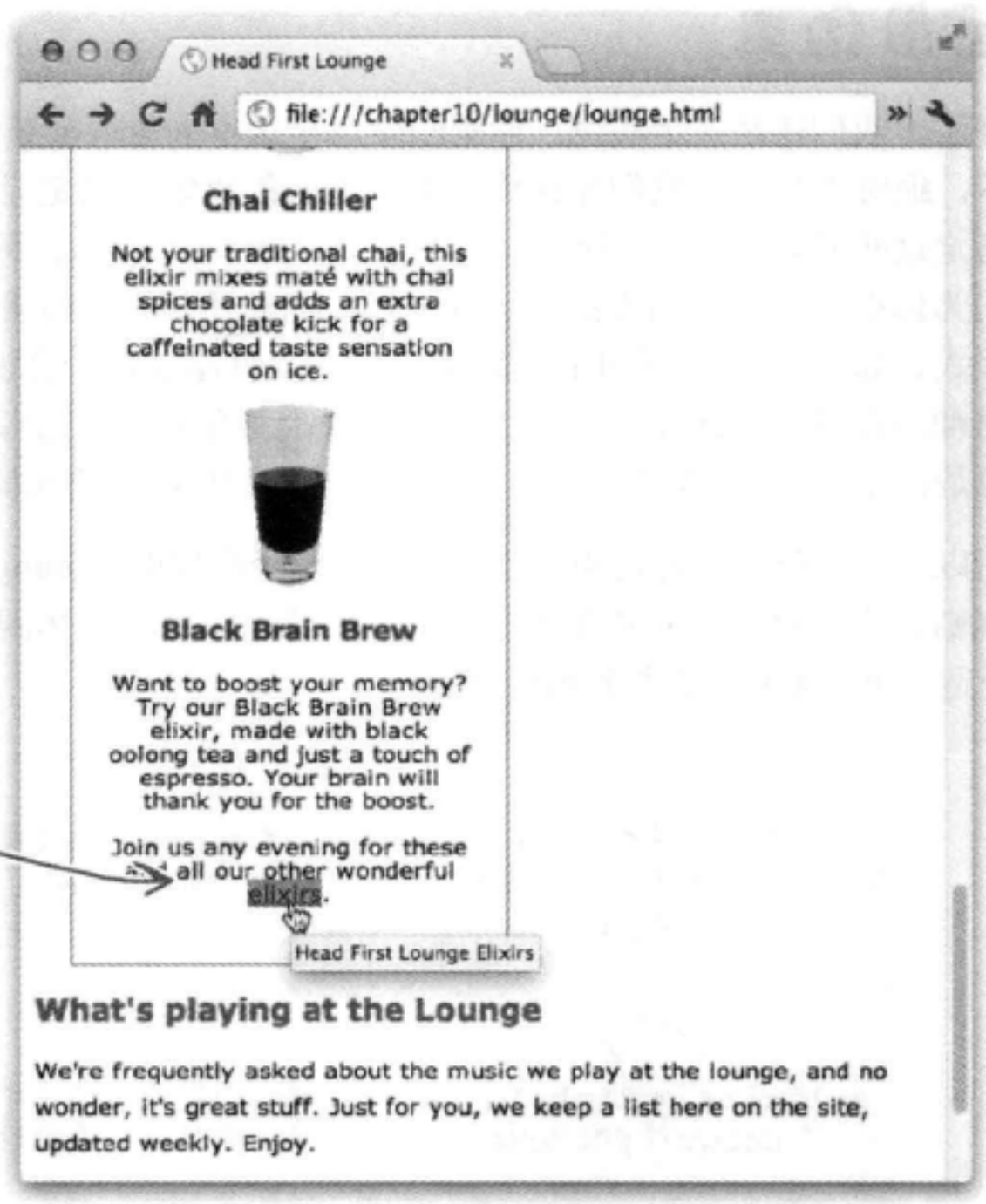
现在来看一个有意思的规则。用户悬停在链接上时，我们要把背景改为红色。这样一来，鼠标移过链接时链接会非常醒目。可以试一下！



打开你的“lounge.css”，使用新的子孙选择器和新样式定义修改a:link、a:visited和a:hover规则。保存文件，重新加载页面，然后翻开下一页。

测试链接

重新加载时，你会看到饮料区有一些新样式。要记住，要想看到未访问的链接，必须先清空浏览器的历史记录。否则，浏览器知道你以前访问过这些链接。



现在未访问的链接是绿色，已访问的链接是灰色。另外鼠标停在链接上时还会有一个非常酷的红色背景，使它非常突出。

Sharpen your pencil

你的任务是为休闲室页面上的“detailed directions”链接指定样式。就像饮料链接一样，我们希望所有未访问的链接是青绿色，所有已访问的链接为灰色。不过，我们不希望休闲室的其他链接有悬停样式……这是饮料链接特有的。那么如何做到呢？请完成下面的填空，为“detailed directions”链接和你以后可能为休闲室页面增加的所有其他链接指定样式。对照检查这一章最后的答案，然后在你的休闲室文件中完成修改。

```
_____ { _____ : #007e7e; }
_____ { _____ : #333333; }
```

是不是该谈谈“层叠”了？

噢，好的，好的。这本书读到这里已经很多页了（准确地讲是457页），我们还没有告诉你层叠样式表中的“层叠”到底是什么。说实话，你必须对CSS有更多了解才能充分理解层叠的含义。不过，你储备的知识已经差不多了，不用再等了。

要理解层叠，还有最后一个信息必须知道。你已经知道如何使用多个样式表来更好地组织你的样式，或者支持不同类型的设备。不过实际上用户访问你的页面时还有另外一些样式表。下面来看一下：

首先是你为页面写的所有样式表。

作者（就是你）

注意，如果读者在一个属性声明的最后加上“!important”，他就能覆盖你的样式。

不过有些浏览器还允许用户为HTML元素创建自己的样式。如果你的样式表中没有定义这些样式，就会使用用户创建的样式表。

读者（你的用户）

最后，你已经知道浏览器本身会维护一组默认样式，如果你没有为一个元素定义样式，就会使用这些默认样式。如果根本没有任何作者或读者样式表，也会使用这些样式。

浏览器确定要对一个元素应用哪个样式时，它会使用所有这些样式表。最优先的是作者的样式（也就是你的样式），然后是读者的样式，最后才是浏览器的默认样式。

浏览器

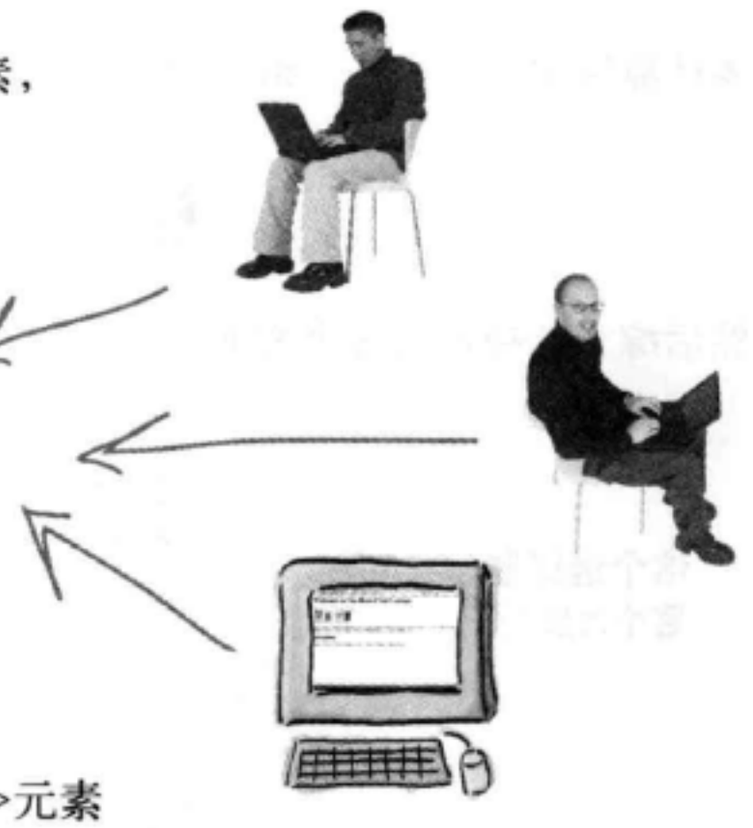
我们来复习一下，作为页面作者，我们可以对我们的HTML使用多个样式表，用户也可能会提供自己的样式，另外浏览器也有默认样式。而且不仅如此，还可以有多个选择器应用到同一个元素。那么如何确定一个元素究竟应用哪些样式呢？

实际上，这就是在问层叠的作用，只是问法不同。给定一组样式表中的一组样式，浏览器就是以层叠方式来确定具体使用哪一个样式。要回答这个问题，我们需要把方方面面都综合起来，这包括各种可用的样式表，规则，还有这些规则中的各个属性声明。

在后面两页中，我们会一步一步说明这是如何工作的，使你了解所有的细节。这些细节涉及到大量排序，还需要确定对于一个元素来说哪些规则最为特定。不过这是有回报的：等学完后面两页，你就能弄清楚为什么没有像你预期的那样应用某些样式，不仅如此，你会比99%的Web页面开发人员都更懂层叠（绝不是开玩笑）。

层叠

在这个练习中，你要“扮演浏览器”。假设你的页面上有一个<h1>元素，你想知道这个元素的font-size属性。你会这么做：



第1步：

收集所有样式表。

这一步你需要所有样式表，包括：Web页面作者写的样式表，读者增加的样式表，还有浏览器的默认样式（要记住，你现在是浏览器，所以你能得到所有这些样式表）。

第2步：

找到所有匹配的声明。

特别地，我们要找font-size属性，所以要查看所有可能选择<h1>元素的选择器的font-size声明。检查所有样式表，找出所有匹配<h1>而且有font-size属性的规则。

第3步：

现在对所有匹配的规则排序。

既然得到了所有匹配的规则，现在按作者、读者和浏览器对这些规则排序。换句话说，如果是你（页面的作者）写的规则，它们就比读者写的规则更重要。相应地，读者的样式则比浏览器的默认样式更重要。

记住，我们提到过，读者可能在他的CSS属性上加上!important，如果是这样，排序时这些属性最为优先。

第4步：

现在按特定性对所有声明排序。

记住，之前在第7章中简单讨论过这个内容。凭直觉你可能认为，如果一个规则能更准确地选择一个元素，那么这个规则就更为特定，例如，子孙选择器“blockquote h1”就比“h1”选择器更特定，因为前者只选择<blockquote>中的<h1>。不过这里有一个小秘诀，你可以按照这个秘诀计算一个选择器的特定性，下一页就会介绍如何来计算。

第5步：

最后，对于冲突的规则，按照它们在各自样式表中出现的顺序进行排序。

现在只需要对冲突的规则排序，各个样式表中后出现的规则（更靠近最下面）更重要。所以，如果在样式表中增加一个新规则，它会覆盖在它之前的所有规则。

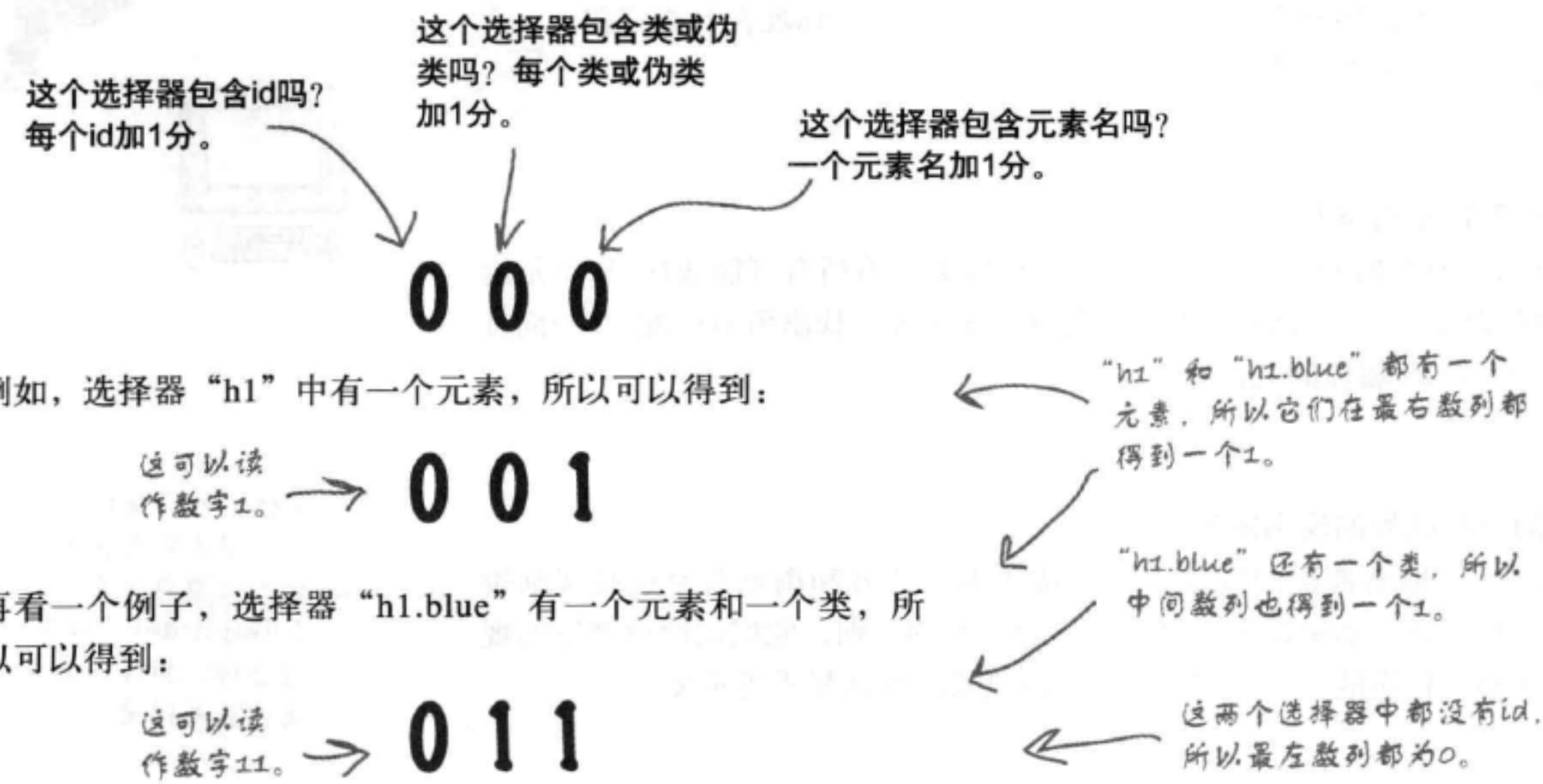
就是这样！得到的有序列表中的第一个规则就是胜出者，它的font-size属性就是你要使用的属性。现在来看如何确定一个选择器究竟有多特定。

欢迎参加“我有多特定?”游戏

要计算特定性，先从一组3个数开始，如下所示：

0 0 0

然后综合选择器的各个方面，如下所示：



例如，选择器“h1”中有一个元素，所以可以得到：

0 0 1

再看一个例子，选择器“h1.blue”有一个元素和一个类，所以可以得到：

0 1 1

综合考虑所有id、类和元素之后，得到的特定性数越大，这个规则就越特定。所以，由于“h1.blue”的特定性值为11，所以它比“h1”（特定性值为1）更特定。

Sharpen your pencil

使用上面的规则，计算以下选择器的特定性：

h1.greentea	_____	ol li p	_____	em	_____
p img	_____	.green	_____	span.cd	_____
a:link	_____	#elixirs h1	_____	#sidebar	_____

there are no Dumb Questions

问：怎样才算一个特定性数大于另一个特定性数？

答：就把它们读作真正的数：100（一百）大于010（十），它又大于001（一），依此类推。

问：那么类似“h1, h2”的规则呢？它的特定性是多少？

答：可以把它看作是两个单独的规则：一个“h1”规则，特定性为“001”，还有一个“h2”规则，特定性也为“001”。

问：你能再多讲讲关于!important的内容吗？

答：读者可能会覆盖一个样式，在他的属性声明最后放置一个“!important”，就像这样：

```
h1 {
    font-size: 200%
!important;
}
```

这会覆盖作者的样式。

问：我没办法得到读者的样式表，该怎么来理解层叠的做法呢？

答：你确实得不到，不过可以这样来考虑，如果读者覆盖了你的样式，这确实超出了你的控制范围。所以要适当地设计你的页面，让读者知道你希望他们使用你的样式。如果读者仍然选择覆盖你的样式，就随他们去吧，也许他们会得到更好的结果，但也可能更糟。

综合在一起

哦！ 哦 ！该给个例子了。假设你想知道这个<h1>元素的color属性：

```
<h1 class="blueberry">Blueberry Bliss Elixir</h1>
```

下面一步一步完成层叠：

记住，你是浏览器，因为你要确定如何显示这个<h1>元素。

第1步：

收集所有样式表。

```
h1 {
    color: #efefef;
}

h1.blueberry {
    color: blue;
}
```

通常你是作者（写CSS的人）。不过现在你暂时是浏览器。

作者

```
h1 {
    color: black;
}
```

浏览器

这是你（暂时如此）。

```
body h1 {
    color: #cccccc;
}
```

读者

使用浏览器的人。

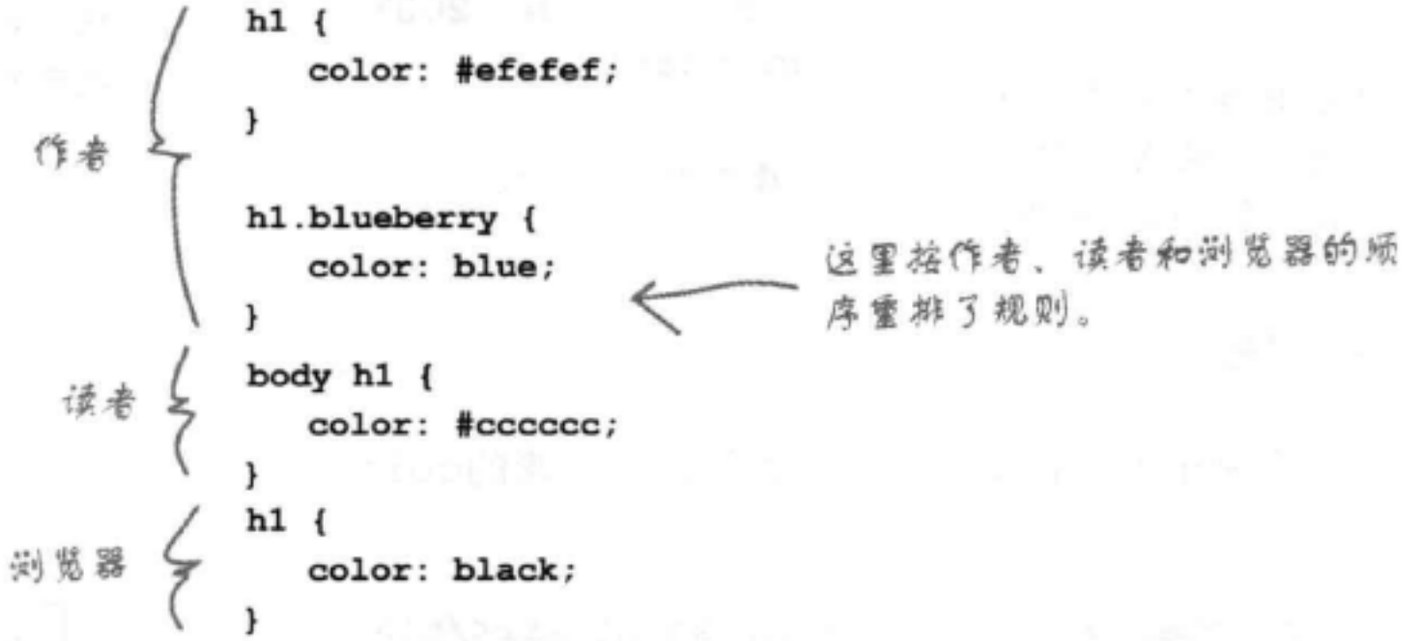
第2步：

找到所有匹配的声明。

这里是所有可能匹配<h1>元素而且包含color属性的规则。

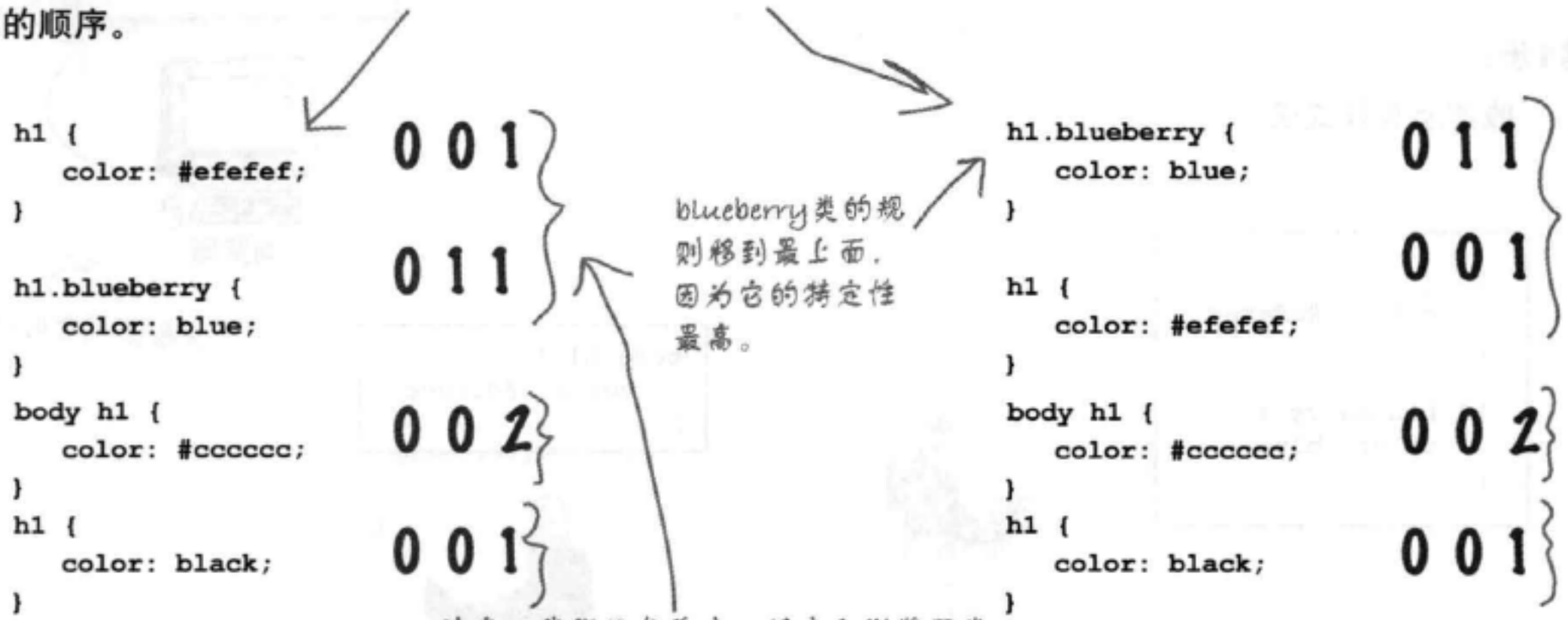
第3步：

现在对所有匹配的规则按作者、读者和浏览器的顺序排序。



第4步：

现在按特定性对所有声明排序。为此，需要首先计算各个特定性得分，然后重排规则的顺序。



第5步

最后，对于冲突的规则，按照它们在各个样式表中出现的顺序排序。

这里就可以了，因为目前没有冲突的规则。blueberry规则得分11，显然胜出。如果有两个规则得分都是011，最后出现的规则将成为赢家。

```
h1.blueberry {
   color: blue;
}

h1 {
   color: #efefef;
}

body h1 {
   color: #cccccc;
}

h1 {
   color: black;
}
```

作者

读者

浏览器

我们找到一个赢家……

通过对元素的初选、多次排序以及确定特定性，“h1.blueberry”规则排名第一。所以<h1>元素的color属性将是blue。

there are no Dumb Questions

问：再问一次：我知道在CSS文件中位置越低，优先级越高，不过我的HTML中有多个指向样式表的链接，这该怎么处理？

答：不论是否在同一个CSS文件中，规则总是从上到下排列。可以假装你把所有CSS都按链接文件的顺序插入到你的文件中。这就是规则出现的顺序。

问：这么说，完成特定性排序时，并不会对所有规则重新排序？

答：没错，不会完全重排。可以把每次排序看作是对之前所做工作的进一步完善。所以，首先按作者、读者和浏览器的顺序对规则排序，然后在各个类别中，再按特定性排序，对于有相同特定性的元素，则根据样式表中的顺序再次进行排序。

问：读者真的会建立他们自己的样式表吗？

答：总的来说并不会。不过有些情况下，有视力障碍的人可能会这样做，当然总会有一些人喜欢追求完美。不过，因为每个读者只会控制他自己看到的效果，所以这不应该影响你的设计。

问：这么多内容，我真的都需要记住吗？

答：对于如何综合所有这些样式表，你会慢慢有些直觉，这种直觉会与日俱增，让你走得更远。不过，偶尔你也会看到页面上有一个你不明白的样式，那时就需要进一步提高自己的水平了。你肯定能很好地处理层叠，在此之前，你要清楚地知道页面中有些什么。

如果层叠没有提供值

那么，如果经过所有这些步骤，我还是没有找到包含特定属性声明的规则，来得到我想要确定的属性值，那该怎么办？

哈，这个问题问得好。实际上我们在第7章已经简单提到过。如果在层叠的所有规则中都没有找到匹配的属性，就要使用继承了。还记得吗？并不是所有属性都能继承，比如边框属性。不过，对于能继承的属性（如color，font-family，line-height等），浏览器会查看这个元素的祖先，从它的父元素开始，尝试找到这个属性的值。如果找到了，这就是你要用的属性值。

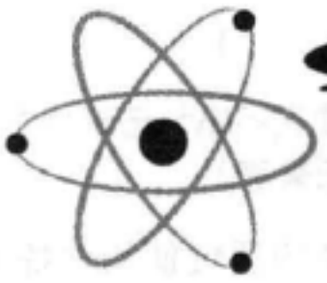
我明白了。嘿，不过如果这个属性不能继承呢？或者如果我在祖先元素的规则中也找不到这样一个值呢？那该怎么办？

如果是这样，就只能依靠浏览器样式表中设置的默认值了，所有浏览器都应该对每个元素设置了默认样式。

噢，那么到底为什么把这叫做“层叠”呢？

之所以选择“层叠”这个名字，是因为来自多个样式表的样式都“层叠”在页面上，对各个元素会应用最特定的样式（如果你还是不清楚这为什么叫做层叠，也不要沮丧。其实我们自己也不是很清楚。把它叫做“CSS”就行了，我们继续前进）。

停下来！进入下一章之前先完成这个练习！



这是一个特殊的智力题，正是由于它很特殊，我们希望你在学习下一章之前考虑。你需要做到：

1. 打开文件“lounge.html”，找到elixirs 。
2. 将整个elixirs 区移到文件最上面，让它在包含休闲室logo的段落下面（紧挨着那个段落）。
3. 保存并重新加载页面。有什么变化吗？
4. 打开文件lounge.css”。
5. 找到“#elixirs”规则。
6. 在规则的最下面增加以下声明：

```
float: right;
```

7. 保存文件，并在浏览器中重新加载页面。

有什么变化？你认为这个属性有什么作用？

BULLET POINTS

- 元素用于将相关的元素归组在一起，放在逻辑区中。
- 创建逻辑区有助于标识主内容区以及页面的页眉和页脚。
- 可以使用 元素将需要共同样式的元素归组在一起。
- 使用嵌套 元素为文件增加更多结构，这有利于保证结构清晰或者方便增加样式。不过除非确实需要，否则不要过多地增加结构。
- 一旦用 元素将内容区归组在一起，类似于其他块元素，可以对这些 增加样式。例如，对于包含在 中的一组元素，可以使用嵌入这些元素的 的边框属性，对这组元素增加一个边框。
- width属性设置一个元素内容区的宽度。
- 一个元素的总宽度是内容区宽度，加上所增加的内边距、边框和外边距的宽度。
- 一旦设置一个元素的宽度，它不会延伸来占满浏览器窗口的整个宽度。
- text-align是块元素的一个属性，用来将这个块元素中的所有内容对齐，可以居中，左对齐或右对齐。这个属性可以由所有嵌套的块元素继承。
- 可以使用子孙选择器来选择嵌套在其他元素中的元素。例如，子孙选择器

```
div h2 {...}
```

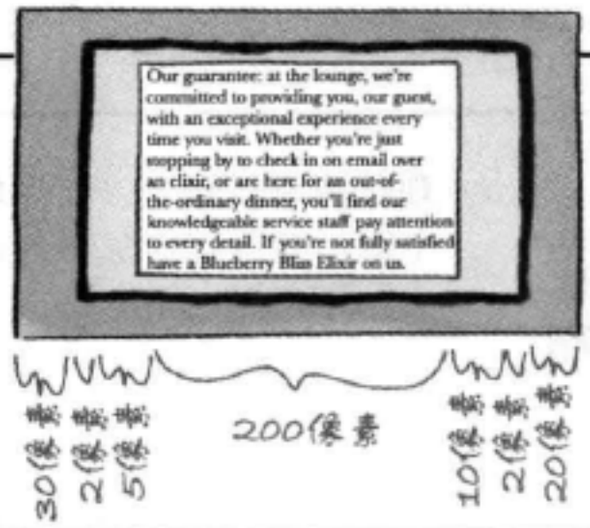
- 会选择嵌款在 元素中的所有<h2>（包括子元素、孙子元素等）。
- 可以对相关的属性使用快捷方式。例如，padding-top、padding-right、padding-bottom和padding-left都与内边距有关，可以用一个快捷规则来指定：padding。
- 内边距、外边距、边框、背景和字体属性都可以用快捷方式指定。
- 内联元素与 元素类似；它用于将相关的内联元素和文本归组在一起。
- 类似于 ，可以将元素增加到类（或者为元素指定唯一的id），对它们增加样式。
- 有些元素有不同的状态，<a>元素就是这样一个例子。<a>元素的主要状态包括未访问、已访问和悬停。
- 可以用伪类单独地为各个状态指定样式。伪类最常用于<a>元素，:link对应未访问的链接，:visited对应已访问的链接，:hover对应悬停状态。
- 伪类还可以用于其他元素，而不仅限于<a>。
- 另外一些伪类包括:hover, :active, :focus, :first-child和last-child伪类等。

HTML填字游戏去度假了

因为你要做一个“超级智力题”，所以这一章我们让HTML填字游戏放假。不用担心，下一章它还会回来的。

Sharpen your pencil Solution

这里有一个盒子，已经标出的所有宽度。你的任务是确定整个盒子的宽度。下面给出答案。



30 + 2 + 5 + 200 + 10 + 2 + 20 = 269

像素

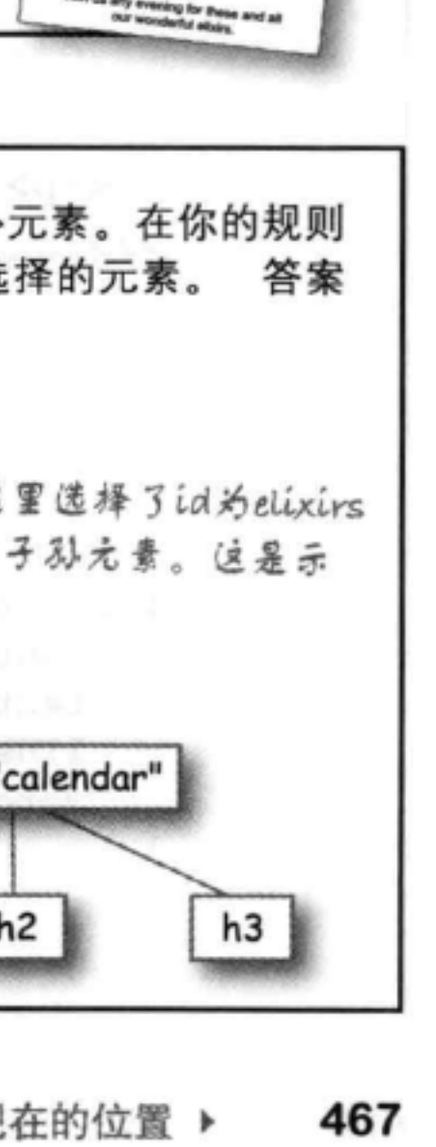
Sharpen your pencil Solution

既然你已经了解了宽度，那么elixirs盒子的总宽度是多少？首先，我们知道内容区宽度是200像素。另外我们还设置了一些左和右内边距，这会影响宽度，还将边框宽度设置为“thin”。假设thin边框为1像素宽（大多数浏览器中都是这样）。外边距呢？我们设置了一个左外边距值，但是没有右外边距，所以默认地，右外边距为0像素。

你的任务是确定elixirs 的总宽度。下面是我们的答案：

20 + 20 + 200 + 1 + 1 + 0 + 20 = 262

左内边距　右内边距　内容区　左边框　右边框　右外边距　左外边距

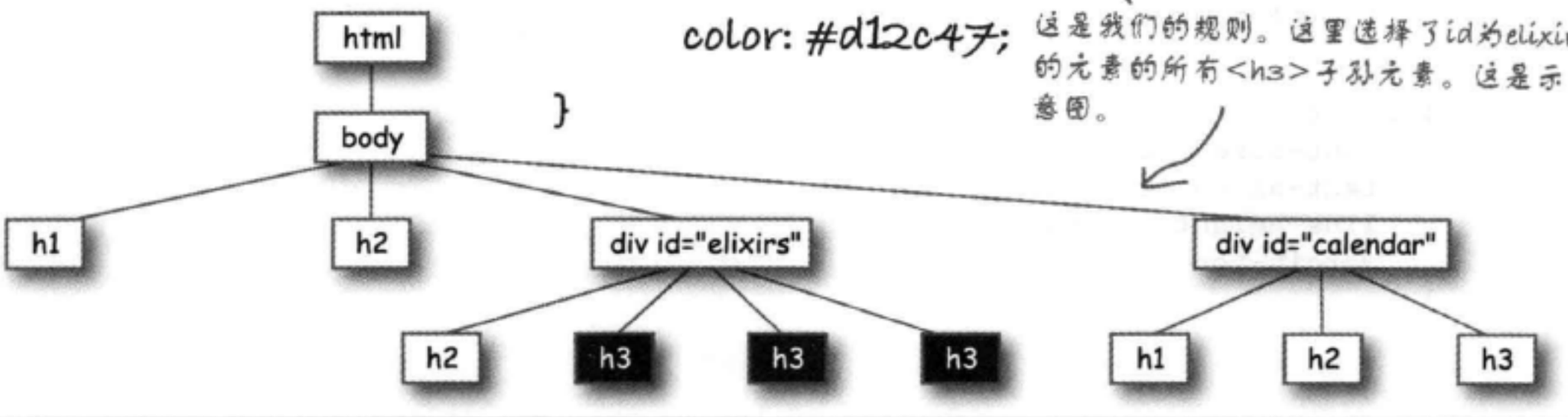


Sharpen your pencil Solution

该轮到你了。请写出选择器，只选择elixirs 中的<h3>元素。在你的规则中，将color属性设置为#d12c47。另外在下面的图中标出选择的元素。 答案如下：

```
#elixirs h3 {
    color: #d12c47;
}
```

这是我们的规则。这里选择了id为elixirs的元素的所有<h3>子孙元素。这是示意图。



现在要学以致用，让你掌握的新知识发挥作用。我们注意到休闲室页面最下面有一个很小的版权信息部分，它要作为这个页面的页脚。增加一个 让它单独成为一个逻辑区。完成之后，再用以下属性为它指定样式：

```
font-size: 50%;
text-align: center;
line-height: normal;
margin-top: 30px;
```

文本设置得很小。你知道的，就是那种极小的字体。

文本居中。

还要将line-height设置为“normal”。

增加一些上外边距，为页脚提供一些呼吸空间。

用 标记包围版权信息。

指定id为“footer”。

```
<div id="footer">
  <p>
    &copy; 2012, Head First Lounge<br>
    All trademarks and registered trademarks appearing on
    this site are the property of their respective owners.
  </p>
</div>
```

更好的方案是将<p>改为<small>，这是专门为“极小字体”设计的一个元素。可以试试看！

这是页脚的CSS。

```
#footer {
   font-size: 50%;
   text-align: center;
   line-height: normal;
   margin-top: 30px;
}
```

Sharpen your pencil Solution

你的任务是为其余音乐推荐增加元素，再测试你的页面。以下给出答案。

```
<ul>
<li><span class="cd">Buddha Bar</span>,
    <span class="artist">Claude Challe</span></li>
<li><span class="cd">When It Falls</span>,
    <span class="artist">Zero 7</span></li>
<li><span class="cd">Earth 7</span>,
    <span class="artist">L.T.J. Bukem</span></li>
<li><span class="cd">Le Roi Est Mort, Vive Le Roi!</span>,
    <span class="artist">Enigma</span></li>
<li><span class="cd">Music for Airports</span>
    <span class="artist">Brian Eno</span></li>
</ul>
```

What's playing at the Lounge

We're frequently asked about the music we play at the lounge, and no wonder, it's great stuff. Just for you, we keep a list here on the site, updated weekly. Enjoy.

- *Buddha Bar*, **Claude Challe**
- *When It Falls*, **Zero 7**
- *Earth 7*, **L.T.J. Bukem**
- *Le Roi Est Mort, Vive Le Roi!*, **Enigma**
- *Music for Airports*, **Brian Eno**

Sharpen your pencil Solution

你的任务是为休闲室页面上的“detailed directions”链接指定样式。就像饮料链接一样，我们希望所有未访问的链接是青绿色，所有已访问的链接为灰色。不过，我们不希望休闲室的其他链接有悬停样式……这是饮料链接特有的。那么如何做到呢？请完成下面的填空，为“detailed directions”链接和你以后可能为休闲室页面增加的所有其他链接指定样式。答案如下：

```
a:link    { color   : #007e7e; }
a:visited { color   : #333333; }
```

Sharpen your pencil Solution

使用上面的规则，计算以下选择器的特定性。这里给出了答案。

h1.greentea	011	ol li p	003	em	001
p img	002	.green	010	span.cd	011
a:link	011	#elixirs h1	101	#sidebar	100

11 布局与定位

摆放元素

让你的HTML元素学点新技巧。我们不再只是让那些HTML元素原地不动，它们该起来了，来帮我们创建一些有真正布局的页面。怎么做呢？嗯，你已经对 和结构元素相当熟悉，而且也知道了盒模型是如何工作的，是吧？现在就该充分利用这些知识来完成一些真正的设计。不，我们并不会大谈更多的背景和字体颜色，我们说的是使用多栏布局的那种成熟、专业的设计。在这一章中你可以把之前学到的知识汇集起来加以应用。

你做超级智力题了吗？

如果你没有做上一章最后的**超级智力题**，那就还得返回去完成这个练习，这是必不可少的。

没错，既然已经打好了基础，扫清了障碍，上一章的最后我们给你留下了一个悬念。我们让你把elixirs 上移，放在logo下面，然后再在CSS中为elixirs规则增加一个小属性，如下：

```
float: right;
```

看呐，这么一个小小的属性居然带来这么大变化！突然之间，它从一个非常普通的Web页面变成了一个两栏的漂亮页面。一下子变得更可读，看起来也更舒服。

这是什么魔法？这样一个看起来一点儿都不起眼的属性怎么会产生如此巨大的效果？我们能不能用这个属性对页面做更有意思的处理？嗯，当然可以，不过首先，你需要了解浏览器在页面上如何摆放元素。了解这些内容之后，我们就能讨论可以采用哪些方法改变页面的布局，另外在页面上如何定位元素。

有一个好消息：你已经了解块元素和内联元素，甚至还知道盒模型。这些正是浏览器建立页面布局的基础。现在你只要知道浏览器如何安排页面上的所有元素，确定它们应该放在哪里。

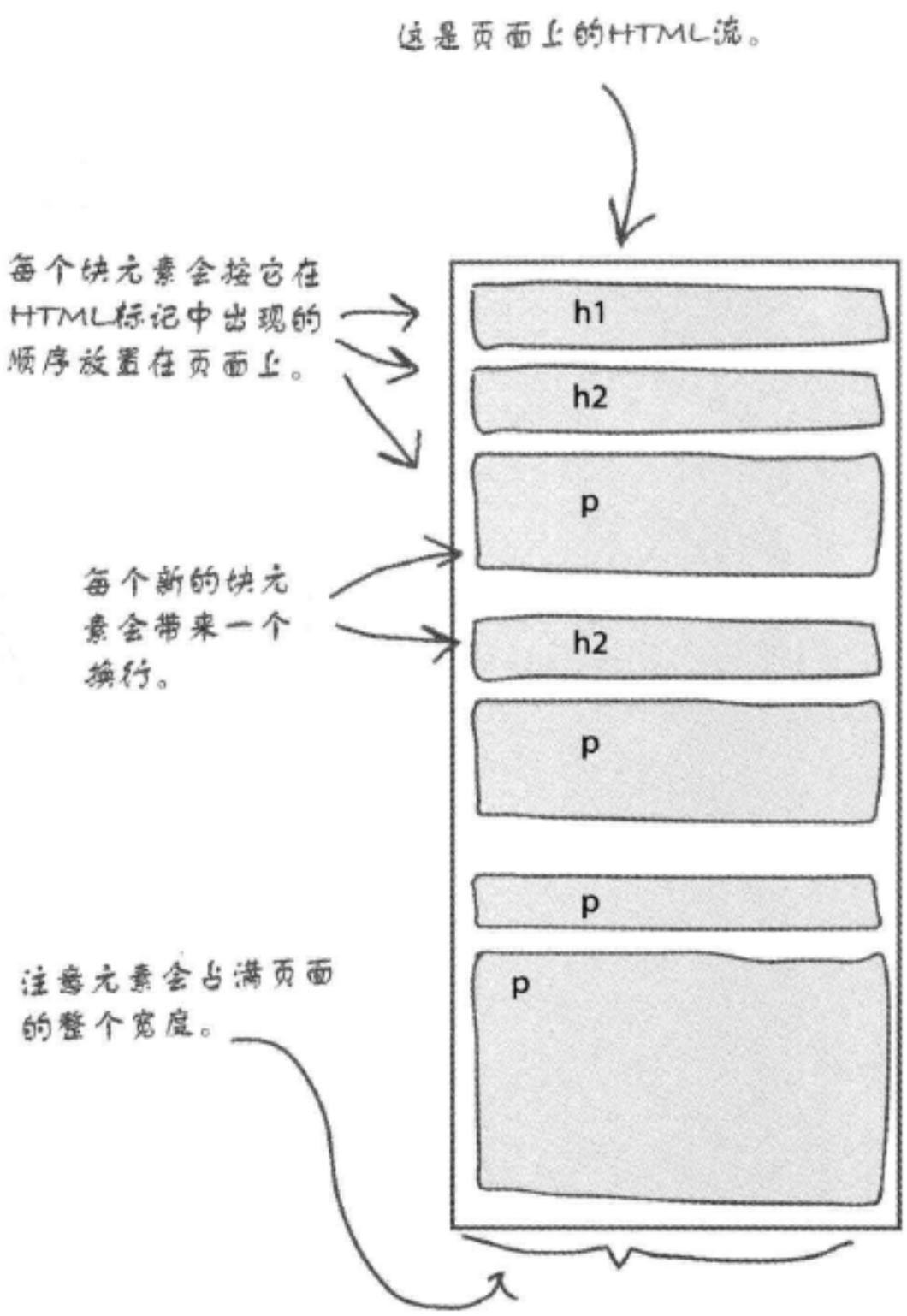
使用流

正是流（Flow）让CSS有了现在的强大威力。这是所有生物创建的一个能量域。它包围着我们，穿透我们。它把整个银河系连在一起……噢，抱歉，有点扯远了。

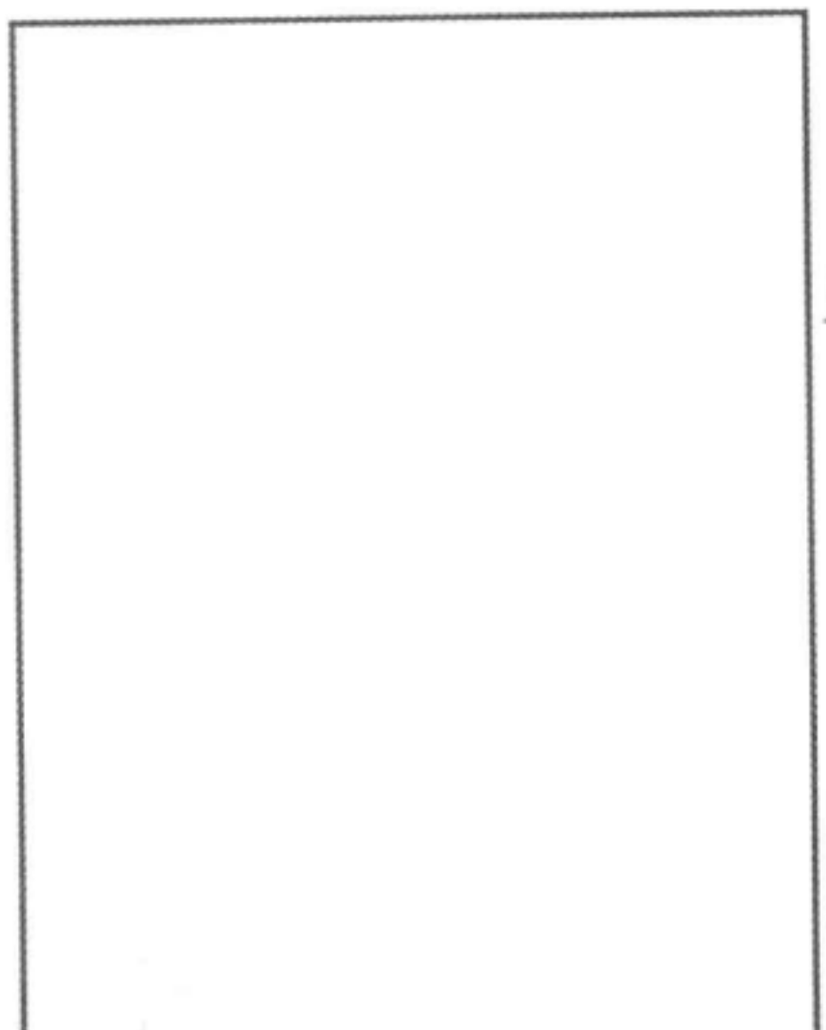
流实际上就是浏览器在页面上摆放HTML元素所用的方法。浏览器从HTML文件最上面开始，从上到下沿着元素流逐个显示所遇到的各个元素。现在先来考虑块元素，它会在每个块元素之间加一个换行。所以首先会显示文档中的第一个元素，然后是一个换行，然后是第二个元素，接下来又是一个换行，如此继续，从文件最上面一直到文件末尾逐个显示。这就是流。

这里有一个简短的“缩略版”HTML。

```
<html>
    <head>...</head>
    <body>
        <h1>...</h1>
        <h2>...</h2>
        <p>...</p>
        <h2>...</h2>
        <p>...</p>
        <p>...</p>
        <p>...</p>
        <p>...</p>
    </body>
</html>
```



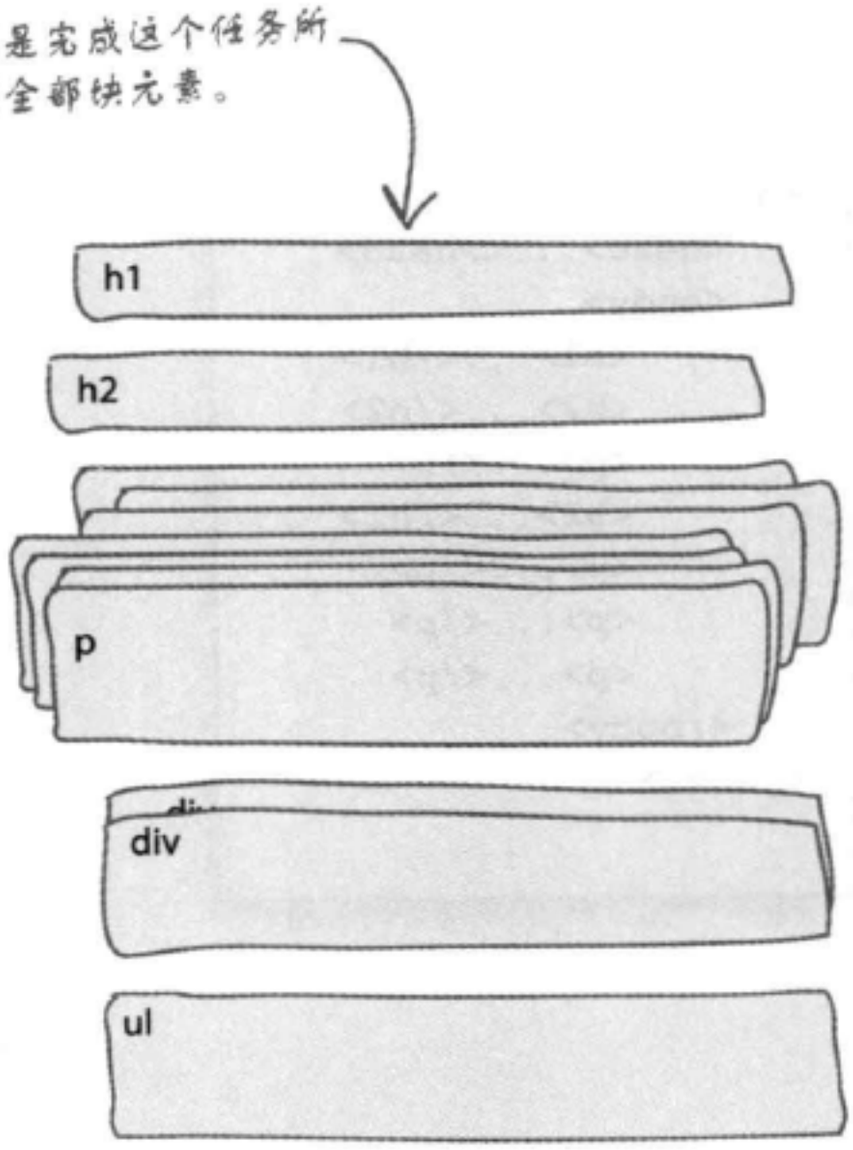
这是你的页面，在这里画出“lounge.html”中的块元素流。



扮演浏览器

打开“lounge.html”文件，找到所有块元素。让各个块元素逐个流入下面的页面。只考虑直接嵌套在body元素中的块元素。可以先忽略CSS中的“float”属性，因为你还不知道它要做什么。继续学习后面的内容之前，先检查你的答案。

这里是完成这个任务所需的全部块元素。



内联元素呢？

你已经知道了块元素从上向下流，各元素之间有一个换行。很简单吧。那么内联元素呢？

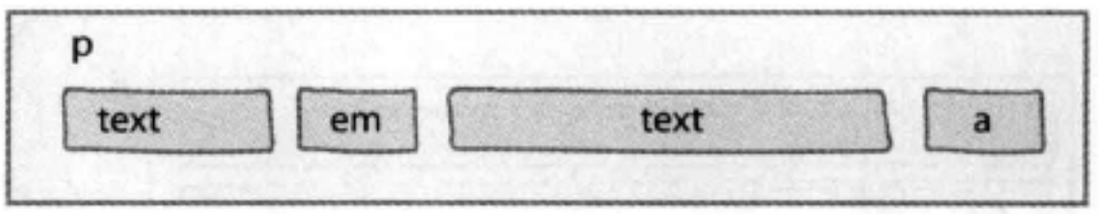
内联元素在水平方向上会相互挨着，总体上会从左上方流向右下方。具体是这样的：

这是另一个HTML片段。

```
<p>
Join us <em>any evening</em> for
these and all our other wonderful <a
href="beverages/elixir.html" title="Head
First Lounge Elixirs">elixirs</a>.
</p>
```

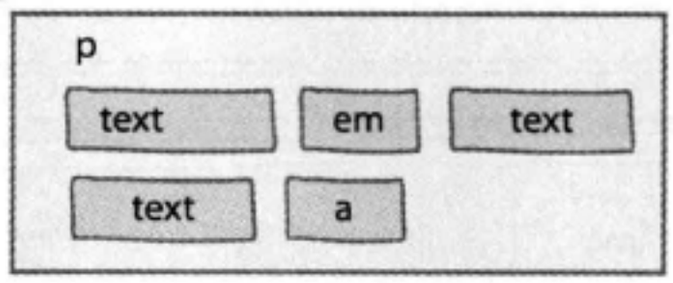
如果将这个<p>元素的内联内容流入页面，会从左上方开始。

内联元素在水平方向上相互挨着摆放（只要右边还有空间能够放下）。



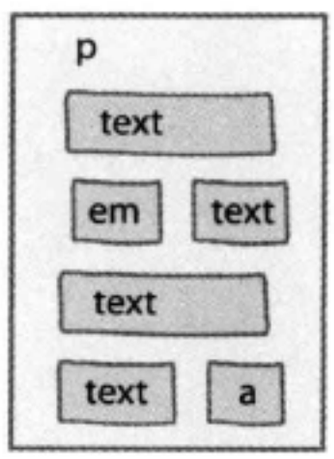
这里有足够的空间，在水平方向上可以放下所有内联元素。注意，文本是内联元素的一种特殊情况。浏览器会把它分解为适当大小的内联元素，以适应给定的空间。

那么，如果让浏览器窗口稍稍窄一点呢，或者用width属性缩小内容区的大小呢？这样一来，放置内联元素的空间会缩小。下面来看具体会怎样。



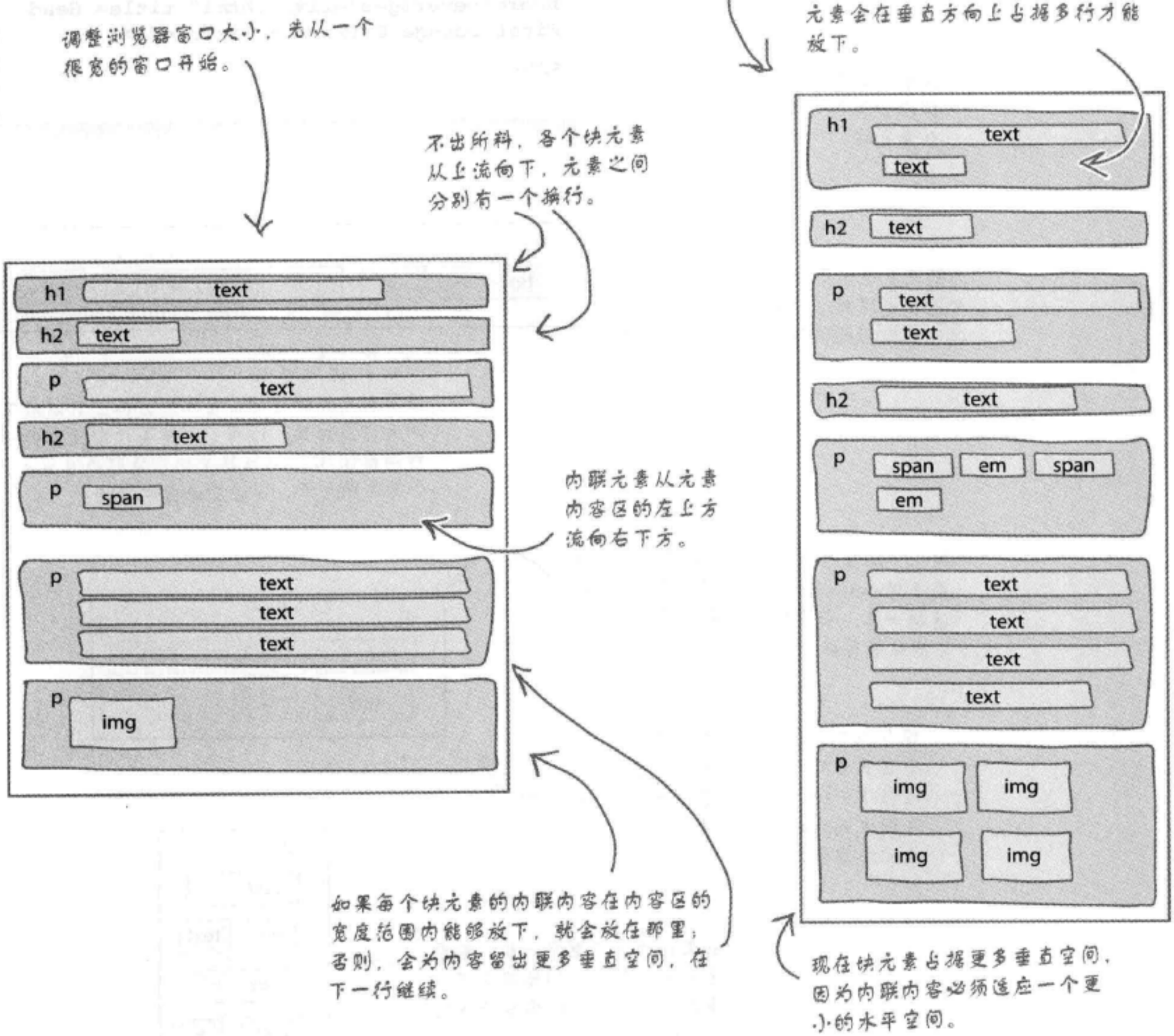
现在内容会从左向右流，直到没有更多空间为止，剩下的内容会放在下一行。注意浏览器必须用不同方式分解文本，使它能刚好适合内容区大小。

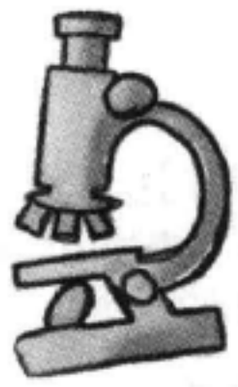
如果让内容区更窄一些，看看会有什么结果。浏览器会根据需要使用多行，将内容流入这个空间。



如何集成？

现在你已经知道了块元素和内联元素如何流入页面，下面把它们集成在一起。我们将使用一个典型的页面，其中包含标题、段落和一些内联元素（如span、一些强调元素，甚至还有一些图像）。另外当然不能忘了内联文本。



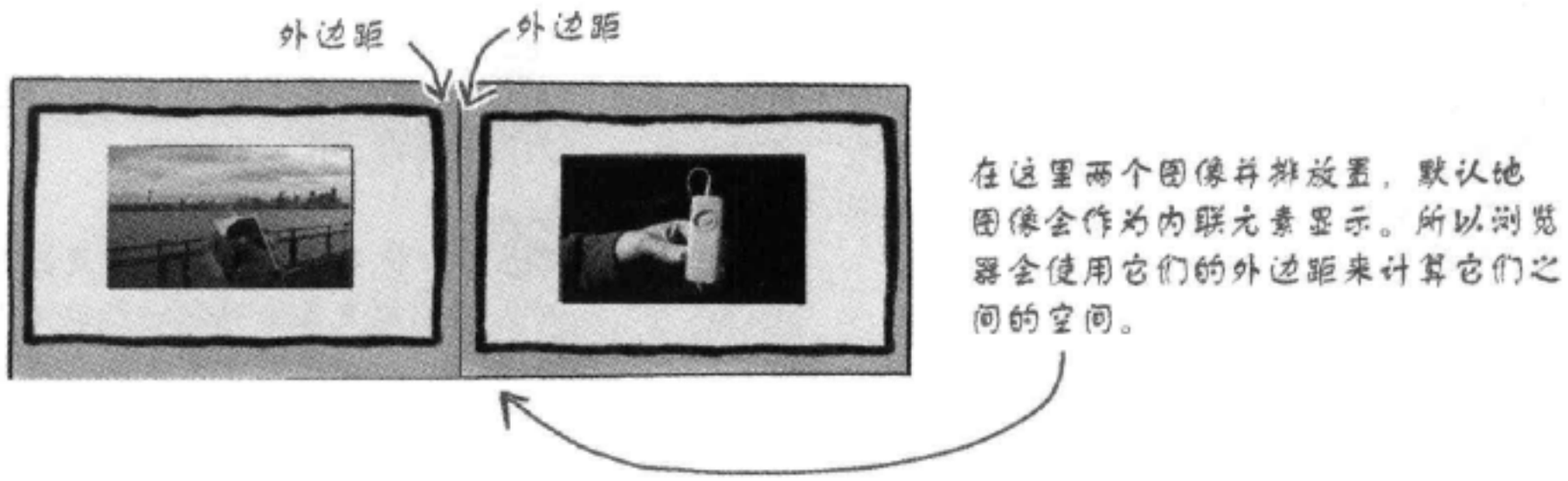


关于流和盒模型还需要知道……

下面稍稍放大，看看浏览器摆放块元素和内联元素的另外一个方面。看起来浏览器会根据页面上放置的元素类型对外边距做不同的处理。

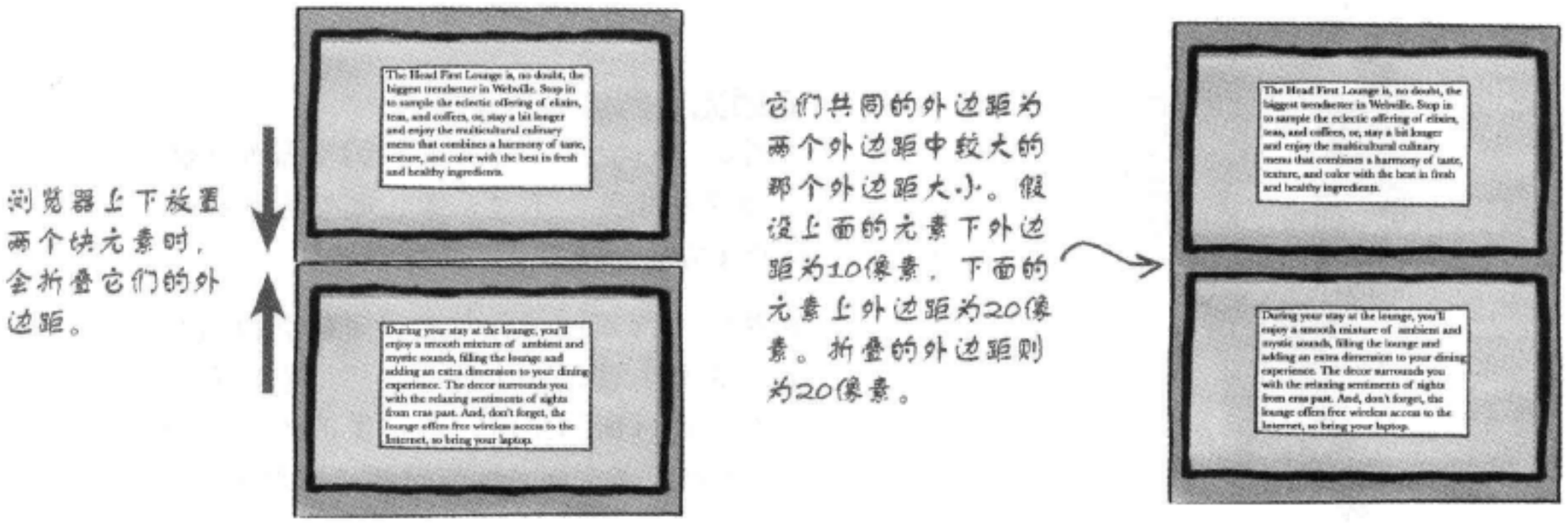
浏览器并排放置两个内联元素时……

如果浏览器执行任务，并排放置两个内联元素，而且这些元素都有外边距，浏览器会像我们期望的那样做。它会在这些元素之间创建足够的空间，考虑到它们的外边距。所以，如果左边的元素外边距为10像素，右边的元素外边距为20像素，那么这两个元素之间就会有30像素的空间。



浏览器上下放置两个块元素时……

这里变得更有意思了。浏览器上下放置两个块元素时，它会把它们共同的外边距折叠在一起。折叠的外边距高度就是最大的外边距高度。



there are no Dumb Questions

问：这么说，如果有一个块元素的外边距为0，它下面的块元素的上外边距为20，那么它们之间的外边距就是20，是吗？

答：没错。如果一个外边距较大，那么两个块元素之间的外边距就是二者中较大的那一个（即使其中一个元素的外边距为0）。不过，如果外边距相同，比如说，都是10像素，它们就会折叠在一起，总共也是10像素。

问：内联元素可以有外边距吗？

答：当然可以，不过你会发现通常并不会设置内联元素的外边距。只有一个例外，这就是图像。对于图像，通常不仅会设置外边距，还会设置内边距和边框，这很常见。尽管这一章我们不会对内联元素设置外边距，但是稍后会为一个图像设置边框。

问：如果一个元素嵌套在另一个元素中，它们都有外边距怎么办？会折叠吗？

答：是的，确实会折叠。可以用下面这种方法来确定元素的外边距何时会折叠：只要两个垂直外边距碰到一起，它们就会折叠，即使是一个元素嵌套在另一个元素中也不例外。注意，如果外面的元素有一个边框，那么两个元素的外边距就不会碰到一起，这样也就不会折叠。但是如果你把这个边框去掉，这两个外边距就会折叠。刚开始看到这种情况时往往会迷惑不解，所以要把它记住，等遇到这种情况时就不会莫名其妙了。

问：既然文本（text）的内容不是元素，为什么文本会作为内联元素工作？

答：即使是文本内容，浏览器也要让它流入页面，对不对？所以浏览器会确定一行能流入多少文本，然后把这行文本当作一个内联元素。浏览器甚至会在它周围创建一个小盒子。你已经看到，如果调整页面的大小，文本会重新适应内容区，所有这些块也会随之改变。

要理解float，必须先了解流。

这可能是一个小属性，不过它的工作方式与浏览器如何将元素和内容流入页面紧密相关。不过，嘿，既然你现在已经了解了流，下面就来解释 float。

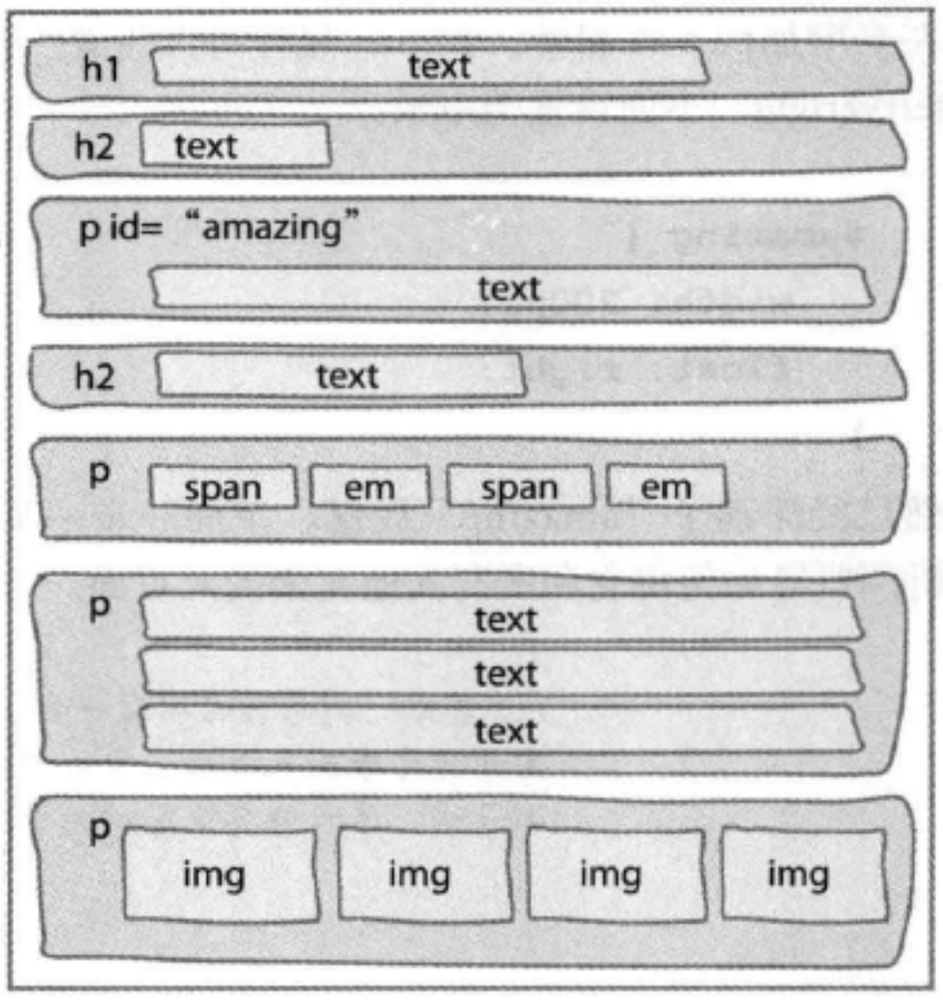
先给出一个简单的答案：float属性首先尽可能远地向左或向右（根据float的值）浮动一个元素。然后它下面的所有内容会绕流这个元素（所谓绕流，就是像流体一样绕着这个元素流动）。当然，还有更多细节问题，下面就来仔细分析……

如何浮动元素

先来逐步分析一个例子，了解如何浮动一个元素，然后查看这样做会对页面流有什么影响。

首先，指定一个标识

下面取页面中的一个段落，指定一个id。我们想把它叫做“amazing floating paragraph”（惊人的浮动段落），不过还是简短点，就叫它“amazing”吧。

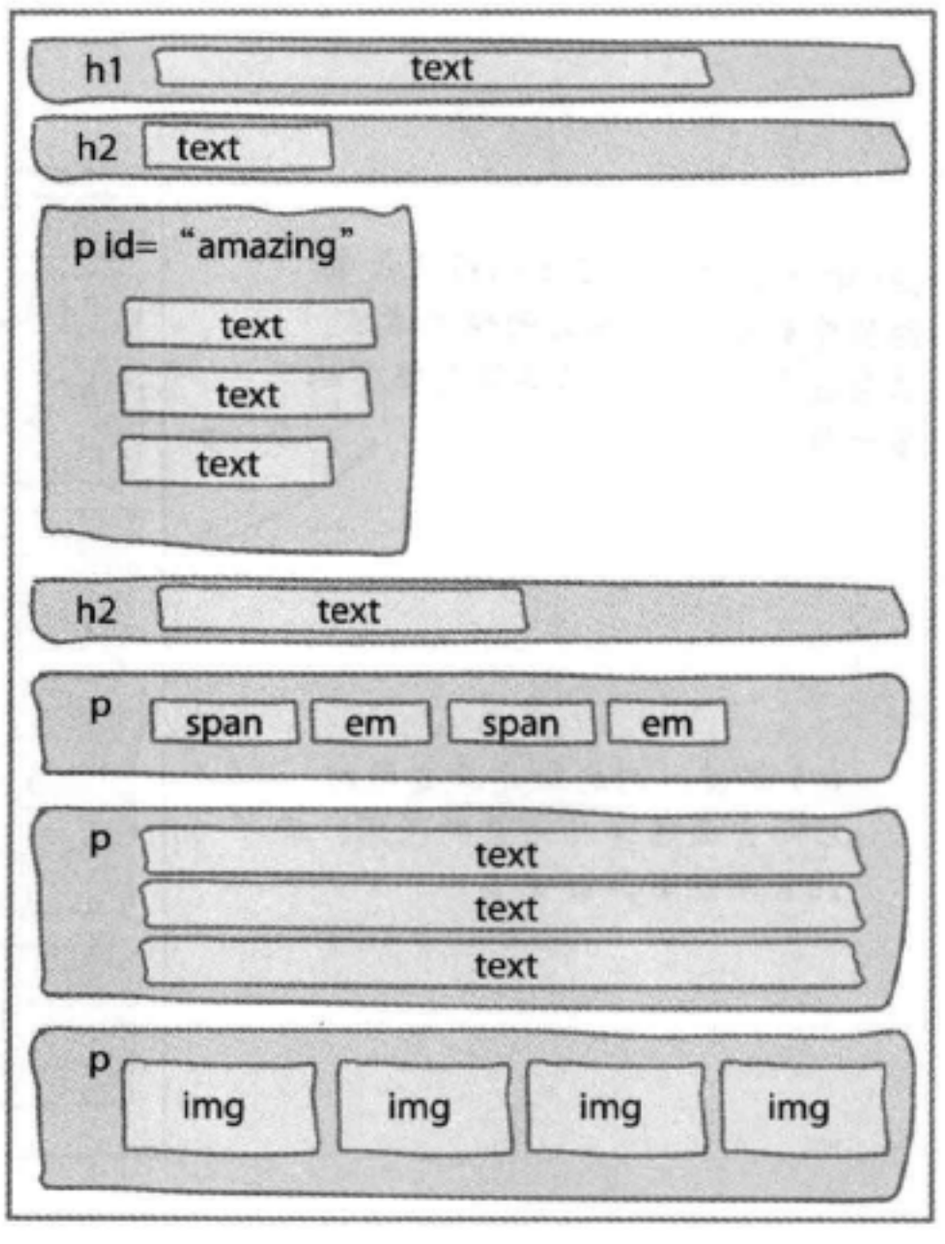


现在指定一个宽度

对于所有浮动元素都有一个要求：它必须有一个宽度。我们设置这个段落宽度为200像素。规则如下：

```
#amazing {
    width: 200px;
}
```

现在这个段落的宽度为200像素，其中包含的内联内容必须调整，以适应这个宽度。要记住，段落是一个块元素，所以不会有元素上移到它旁边，因为所有块元素前后都会有换行。

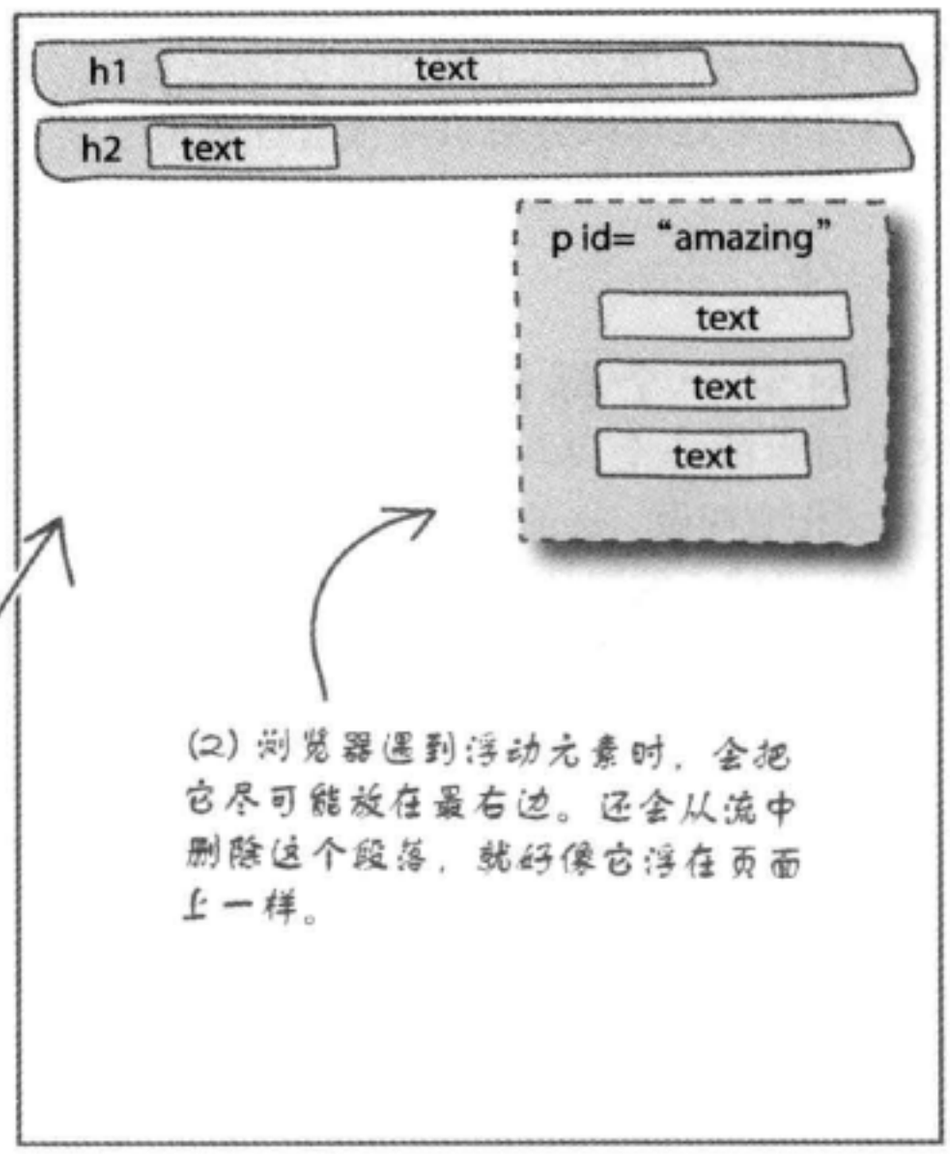


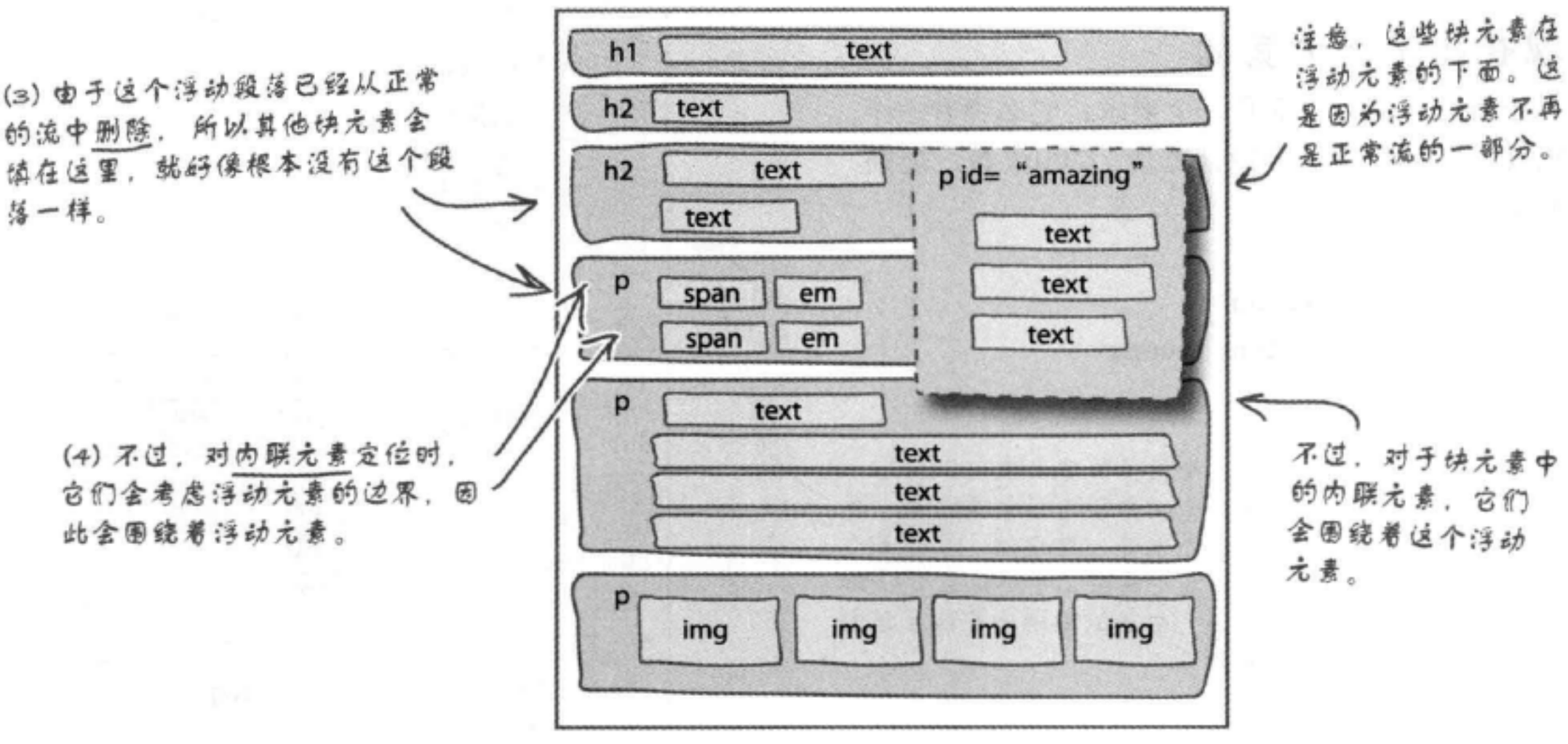
现在让它浮动

现在来增加float属性。float属性可以设置为left或right。下面设置为right：

```
#amazing {
   width: 200px;
   float: right;
}
```

既然已经浮动了“amazing”段落，下面一步一步分析浏览器如何将它和所有其他元素流入页面。





在休闲室的幕后

现在你已经了解了流，也知道了浮动元素在页面上如何摆放。下面再回到休闲室，看看这些内容如何结合在一起。

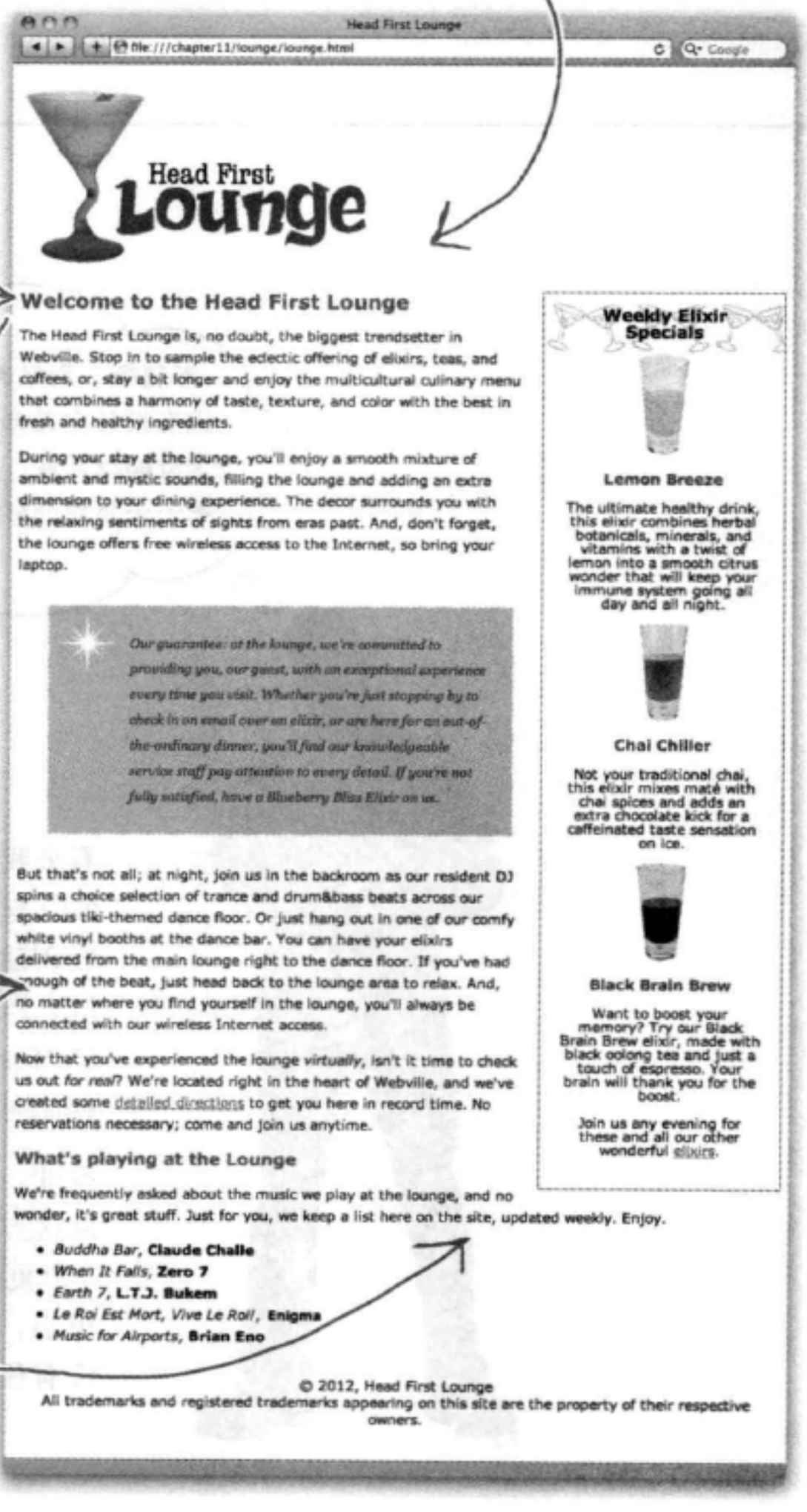
记住，除了将elixirs 设置为向右浮动，我们还把elixirs 上移，让它紧接着放在页面顶部的logo下面。

通过移动这个 ，我们可以把它浮动到右边，然后让整个页面围绕着这个 。如果elixirs 仍然留在音乐推荐下面，那么页面的大部分元素都放置完毕之后才会浮动这个饮料区。

HTML中所有这些元素都在饮料区后面，所以它们会围绕着饮料区。

记住，elixirs 在页面最前面开始浮动，所有其他元素都在它下面，不过内联内容流入页面时会考虑饮料区的边界。

另外要注意，这些文本会围绕饮料区底部，因为这些文本包含在一个块元素中，而这个块元素的宽度与页面相同。如果你的文本没有围绕饮料区，试着将浏览器窗口变窄，直到文本围绕在饮料区下方。



将elixirs　移回到原来的位置，仍然放在主要内容下面，然后保存并重新加载页面。现在元素会浮动到哪里？对照这一章最后的答案检查你的结果，然后把elixirs 再放回到页眉下面。

看来我们又有了一个新任务。Starbuzz网站确实可以有所改进。当然，你之前的工作很棒，已经创建了一个典型的从上向下流动的页面。不过，既然你已经了解了流，应该能让Starbuzz Coffee页面有一个全新的面貌，与上一个设计相比对用户更友好。

不过，这里有一个小秘密……我们已经做了一点这方面的工作。我们已经创建了这个网站的一个更新版本。你的任务是提供所有布局。别担心，我们会帮助你理解目前所做的全部工作，没有任何新内容，都是你之前见过的。

新的Starbuzz页面

下面来简单看一下我们目前有些什么。先来看现在的页面是怎样的，然后会粗略地查看标记以及为标记增加样式的CSS。

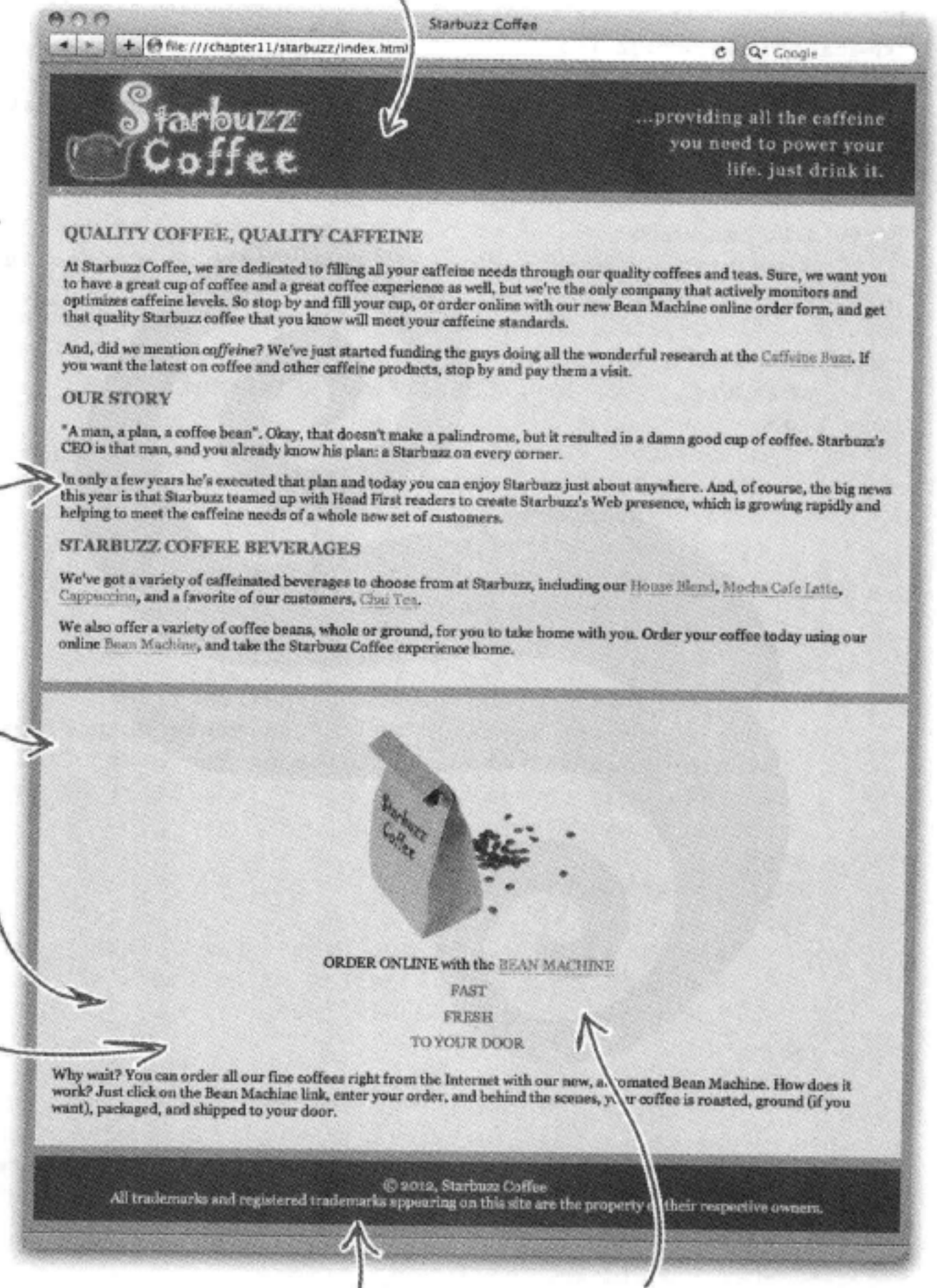
现在有一个页眉，它有一个新的漂亮的Starbuzz logo，还有公司的宗旨声明。这实际上是一个GIF图像。

现在有4个区：页眉、主内容区，一个广告区（推销一种名叫"Bean Machine"的新产品），还有一个页脚。

每个区有一个 ，可以单独指定样式。

看起来页面整体有一个背景色，另外每个 使用了一个图像作为背景。

这里是"Bean Machine"区。它链接到Starbuzz Coffee的一个新页面，可以在那里在线订购咖啡豆。这个链接现在还不起作用，因为下一章才会构建这个Bean Machine页面。



这里是页脚。它没有使用背景图像，只有背景颜色。

注意我们用一种很有趣的方式来指定链接的样式，它有一个虚线下划线……

查看标记

现在来看这个新的Starbuzz HTML标记。我们取出了各个逻辑区，分别放在一个单独的 中，每个 有自己的id。除了 和，这里所有其他内容实际上与第5章中并没有不同。下面来简单看一下，先熟悉结构，然后翻开下一页查看CSS样式。

这些都是标准的HTML“日常处理”……

……后面是对应页眉的 ，然后是对应主内容区的 。

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Starbuzz Coffee</title>
    <link type="text/css" rel="stylesheet" href="starbuzz.css">
</head>
<body>

  <div id="header">
    <img src="images/header.gif" alt="Starbuzz Coffee header image">
  </div>

  <div id="main">
    <h1>QUALITY COFFEE, QUALITY CAFFEINE</h1>
    <p>
      At Starbuzz Coffee, we are dedicated to filling all your caffeine needs through our
      quality coffees and teas. Sure, we want you to have a great cup of coffee and a great
      coffee experience as well, but we're the only company that actively monitors and
      optimizes caffeine levels. So stop by and fill your cup, or order online with our new Bean
      Machine online order form, and get that quality Starbuzz coffee that you know will meet
      your caffeine standards.
    </p>
    <p>
      And, did we mention <em>caffeine</em>? We've just started funding the guys doing all
      the wonderful research at the <a href="http://buzz.wickedlysmart.com"
      title="Read all about caffeine on the Buzz">Caffeine Buzz</a>.
      If you want the latest on coffee and other caffeine products,
      stop by and pay them a visit.
    </p>
    <h1>OUR STORY</h1>
    <p>
      "A man, a plan, a coffee bean". Okay, that doesn't make a palindrome, but it resulted
      in a damn good cup of coffee. Starbuzz's CEO is that man, and you already know his
      plan: a Starbuzz on every corner.
    </p>
    <p>
      In only a few years he's executed that plan and today
      you can enjoy Starbuzz just about anywhere. And, of course, the big news this year
      is that Starbuzz teamed up with Head First readers to create Starbuzz's Web presence,
      which is growing rapidly and helping to meet the caffeine needs of a whole new set of
      customers.
    </p>
    <h1>STARBUZZ COFFEE BEVERAGES</h1>
    <p>
      We've got a variety of caffeinated beverages to choose
      from at Starbuzz, including our
```

这里仍然是主内容区。

```
        <a href="beverages.html#house" title="House Blend">House Blend</a>,
        <a href="beverages.html#mocha" title="Mocha Cafe Latte">Mocha Cafe Latte</a>,
        <a href="beverages.html#cappuccino" title="Cappuccino">Cappuccino</a>,
        and a favorite of our customers,
        <a href="beverages.html#chai" title="Chai Tea">Chai Tea</a>.
    </p>
    <p>
        We also offer a variety of coffee beans, whole or ground, for you to
        take home with you. Order your coffee today using our online
        <a href="form.html" title="The Bean Machine">Bean Machine</a>,
         and take the Starbuzz Coffee experience home.
    </p>
</div>

<div id="sidebar">
    <p class="beanheading">
        <img src="images/bag.gif" alt="Bean Machine bag">
        <br>
        ORDER ONLINE
        with the
        <a href="form.html">BEAN MACHINE</a>
        <br>
        <span class="slogan">
            FAST <br>
            FRESH <br>
            TO YOUR DOOR <br>
        </span>
    </p>
    <p>
        Why wait? You can order all our fine coffees right from the Internet with our new,
        automated Bean Machine. How does it work? Just click on the Bean Machine link,
        enter your order, and behind the scenes, your coffee is roasted, ground
        (if you want), packaged, and shipped to your door.
    </p>
</div>

<div id="footer">
    &copy; 2012, Starbuzz Coffee
    <br>
    All trademarks and registered trademarks appearing on
    this site are the property of their respective owners.
</div>

</body>
</html>
```

这里是对应Bean Machine区的 。我们指定了一个id="sidebar"。嗯，想知道这是什么意思吗？

最后是构成页面页脚的 。

查看样式

下面好好看一下为这个新Starbuzz页面指定样式的CSS。仔细分析各个CSS规则。尽管这个新Starbuzz页面看起来可能挺高级，不过你会发现实际上这些都只是你已经了解的一些简单CSS。

```
body {
        background-color: #b5a789;
        font-family:      Georgia, "Times New Roman", Times, serif;
        font-size:        small;
        margin:           0px;
}

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

#main {
        background:       #efe5d0 url(images/background.gif) top left;
        font-size:        105%;
        padding:          15px;
        margin:           0px 10px 10px 10px;
}

#sidebar {
        background:       #efe5d0 url(images/background.gif) bottom right;
        font-size:        105%;
        padding:          15px;
        margin:           0px 10px 10px 10px;
}

#footer {
        background-color: #675c47;
        color:            #efe5d0;
        text-align:       center;
        padding:          15px;
        margin:           10px;
        font-size:        90%;
}

h1 {
        font-size:        120%;
        color:            #954b4b;
}

.slogan { color:          #954b4b;}

.beanheading {
        text-align:       center;
        line-height:      1.8em;
}
```

首先，在body中建立一些基本样式：一个背景颜色和一些字体，另外设置body的外边距为0。这样可以确保页面边界周围没有额外空间。

接下来，对应各个逻辑区分别有一个规则。在每个规则中，我们稍稍调整了字体大小，增加了内边距和外边距，另外在main和sidebar 中还指定了一个背景图像。

然后，设置标题的字体和颜色。

然后是对类slogan设置的颜色，这会在sidebar 中使用，另外还定义了beanheading类，这也在sidebar 中使用。

Starbuzz CSS的最后两个规则中，我们使用了a:link和a:visited伪类来指定链接样式。

```
a:link {
        color:              #b76666;
        text-decoration:    none;
        border-bottom:      thin dotted #b76666;
}
a:visited {
        color:              #675c47;
        text-decoration:    none;
        border-bottom:      thin dotted #675c47;
}
```

将text-decoration设置为none，删除链接的默认下划线，另外……

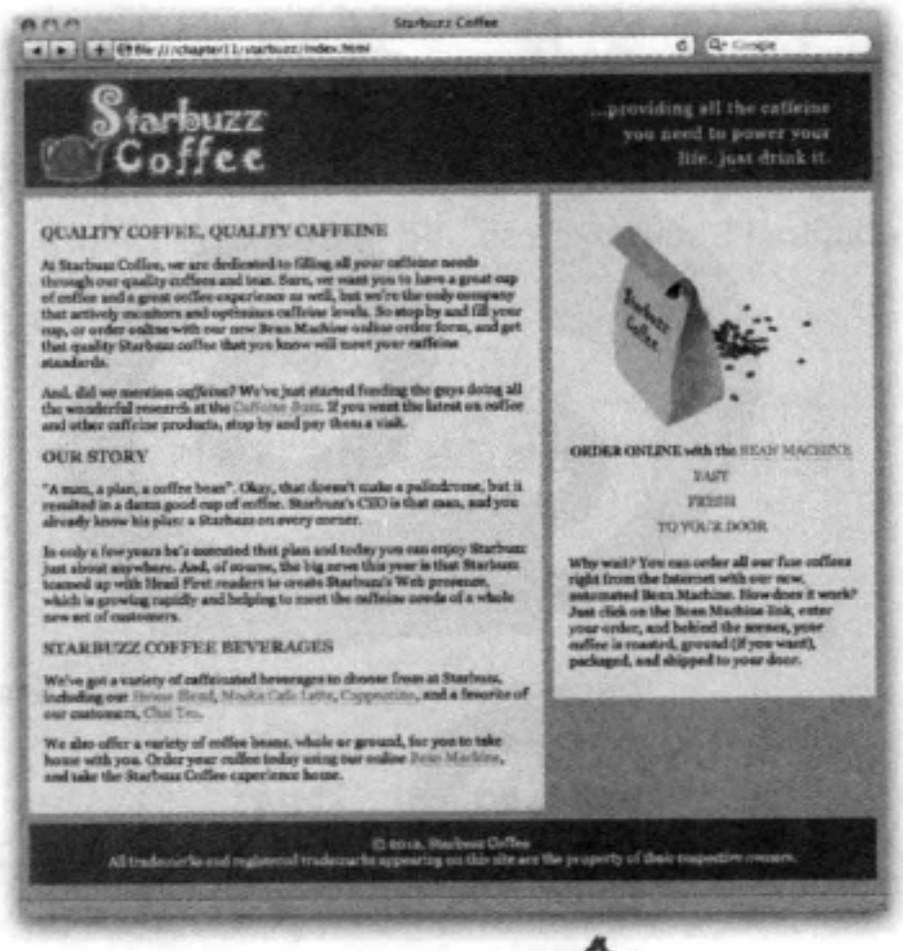
……通过使用一个虚线下边框而不是下划线，来得到漂亮的虚线下划线效果。这是对内联元素使用边框属性的一个很好的例子。

只对这个<a>元素设置 border-bottom。

让Starbuzz更上一层楼

我们的目标是：把Starbuzz Coffee变成类似右边的这个网站。为此，我们需要把Bean Machine边栏移到右边，来得到一个漂亮的两栏页面。你已经成功地对休闲室做过这种处理，对不对？所以，以它为基础，现在你需要做到：

1. 使用id为要浮动的元素指定一个唯一的名字。这个工作已经做过了。
2. 要让这个元素在某个元素后面浮动，确保将浮动元素的HTML放在那个元素（在这里就是Starbuzz页眉）的下面。
3. 设置元素的宽度。
4. 将元素浮动到左边或右边。看起来你希望它浮动到右边。

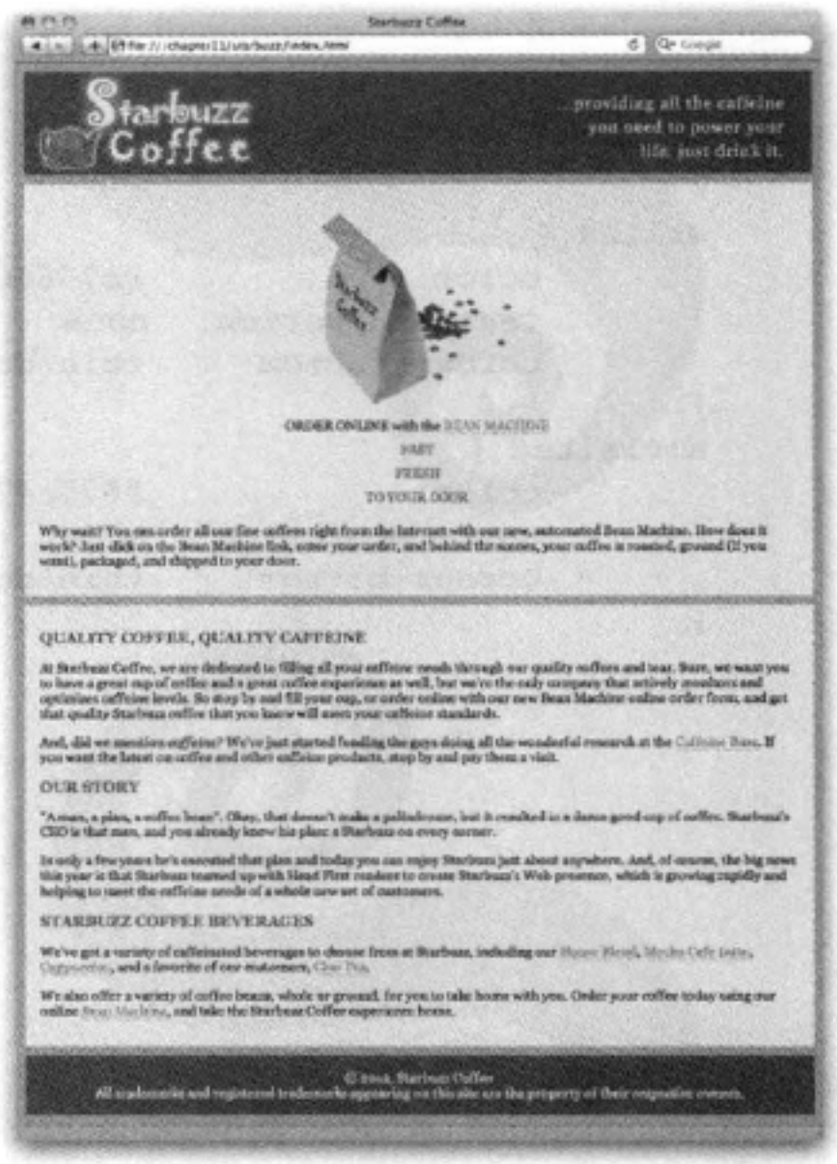


这里我们得到了一个漂亮的两栏页面，两栏明显是分开的。

下面就开始吧。只需要简单的几步，我们就能让Starbuzz CEO非常满意，还会免费送来一些印度香辣奶茶。

把边栏移动到页眉下面

浮动一个元素时，如果希望它在某个元素后面，就要移动浮动元素的HTML，让它紧挨着放在那个元素下面。在这里，边栏需要放在页眉下面。所以，先在编辑器里找到sidebar ，把这个 整个移到header 下面。在“chapter11/starbuzz”文件夹下的文件“index.html”中可以找到这个HTML。完成之后，请保存并重新加载页面。



现在边栏应该在主内容区的上面。

设置边栏宽度，并让它浮动

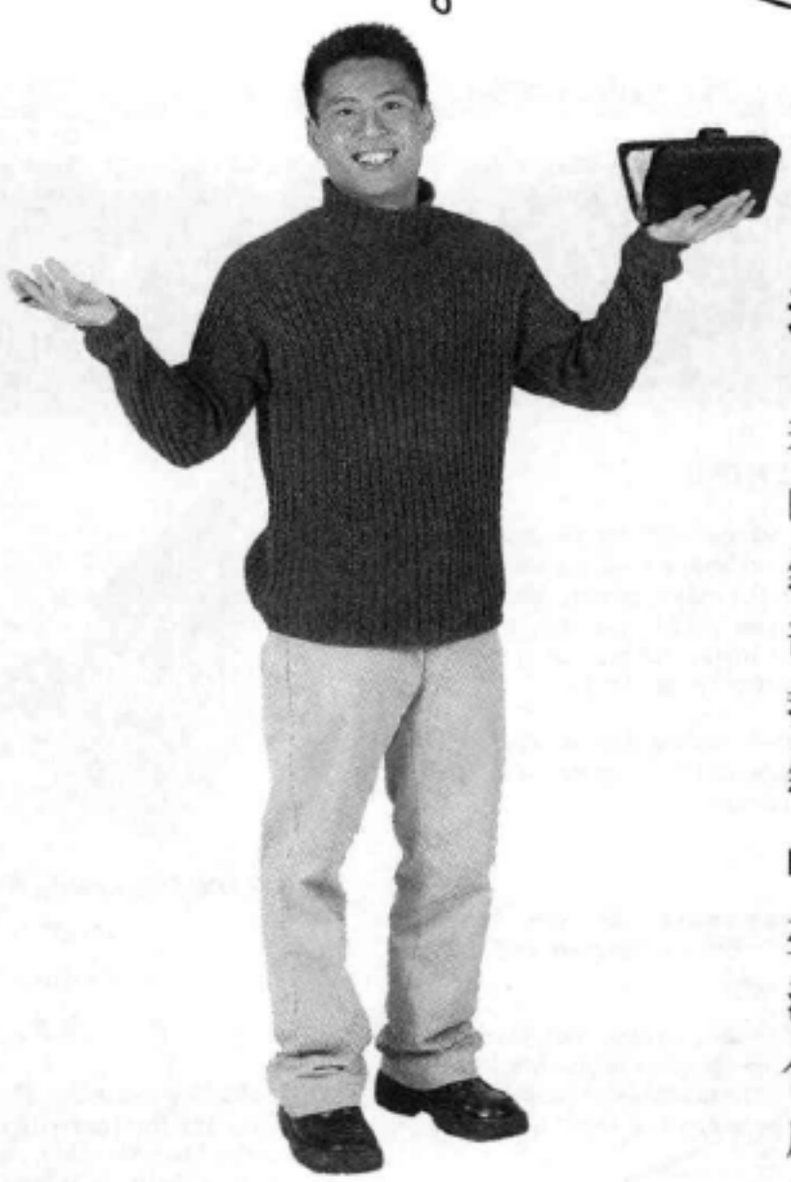
下面把这个边栏的宽度设置为280像素，为了浮动边栏，需要为“chapter11/starbuzz.css”增加float属性，如下：

我们要使用一个id选择器选择id为“sidebar”的元素，我们知道这就是对应边栏的 。

```
#sidebar {
        background: #efe5d0 url(images/background.gif) bottom right;
        font-size:  105%;
        padding:    15px;
        margin:     0px 10px 10px 10px;
        width:      280px;
        float:      right;
}
```

设置内容区宽度为280像素。

然后将这个边栏浮动到右边。记住，这会使边栏在页眉下面尽可能向右浮动，另外还会从正常流中删除边栏。HTML中边栏下面的所有其他内容都会上移，围绕着边栏。



这确实是个好主意，不过存在几个问题。

表面看来，这个想法很不错。我们要做的只是对主内容 设置一个宽度，然后把它浮动到左边，再让页面的其余部分绕着主内容 流动。这样一来，我们就能保留页面现在的顺序，而且仍然能得到两栏。

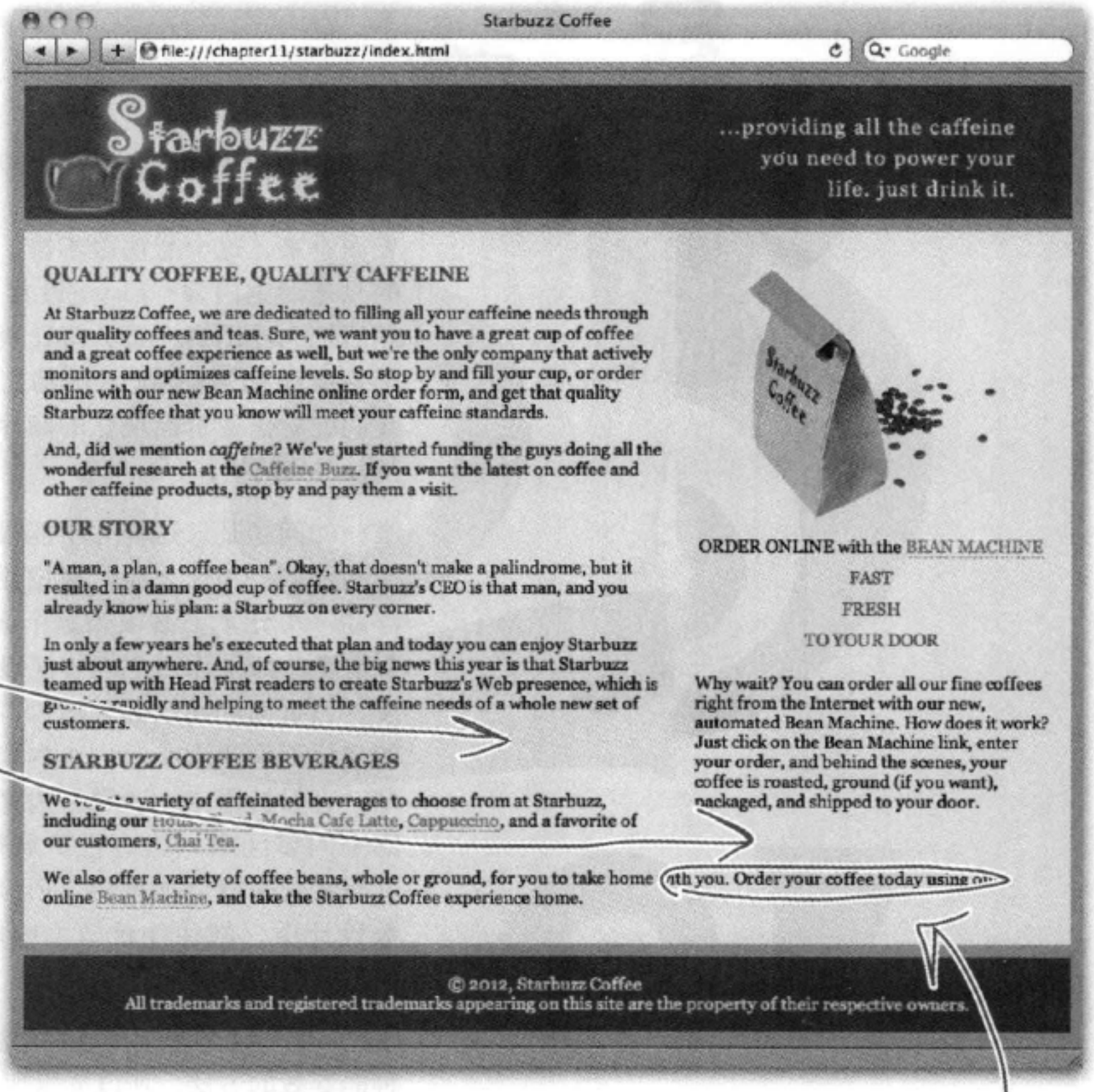
唯一的问题是，这样并不能得到一个漂亮的页面。为什么呢？要记住，如果要浮动一个元素，必须为这个元素设置一个宽度，如果你对主内容区设置一个宽度，当页面的其余部分随着浏览器宽度的变化而调整大小时，主内容区的宽度却一直保持固定。通常，边栏会设计得比主内容区窄，当边栏扩展得太宽时，页面看起来就会很糟糕。所以，在大多数设计中，都往往希望主内容区扩展，而不是边栏。

不过，我们还是想利用这种想法，找到一种能奏效的方法。所以先保留这个想法。我们还要再谈谈为什么需要考虑页面中各个部分的顺序。

测试Starbuzz

要为“chapter11/starbuzz”文件夹中的“starbuzz.css”文件增加新的边栏属性，然后重新加载Starbuzz页面。下面来看我们得到的页面……

嗯，看起来还不错，不过如果再往前翻3页，你会发现这与我们原先想要的页面并不完全相同。



主内容区和边栏分别在左边和右边，不过看起来并不像两栏。

注意这两个部分的背景图像拼到一起了。两栏之间没有分开。

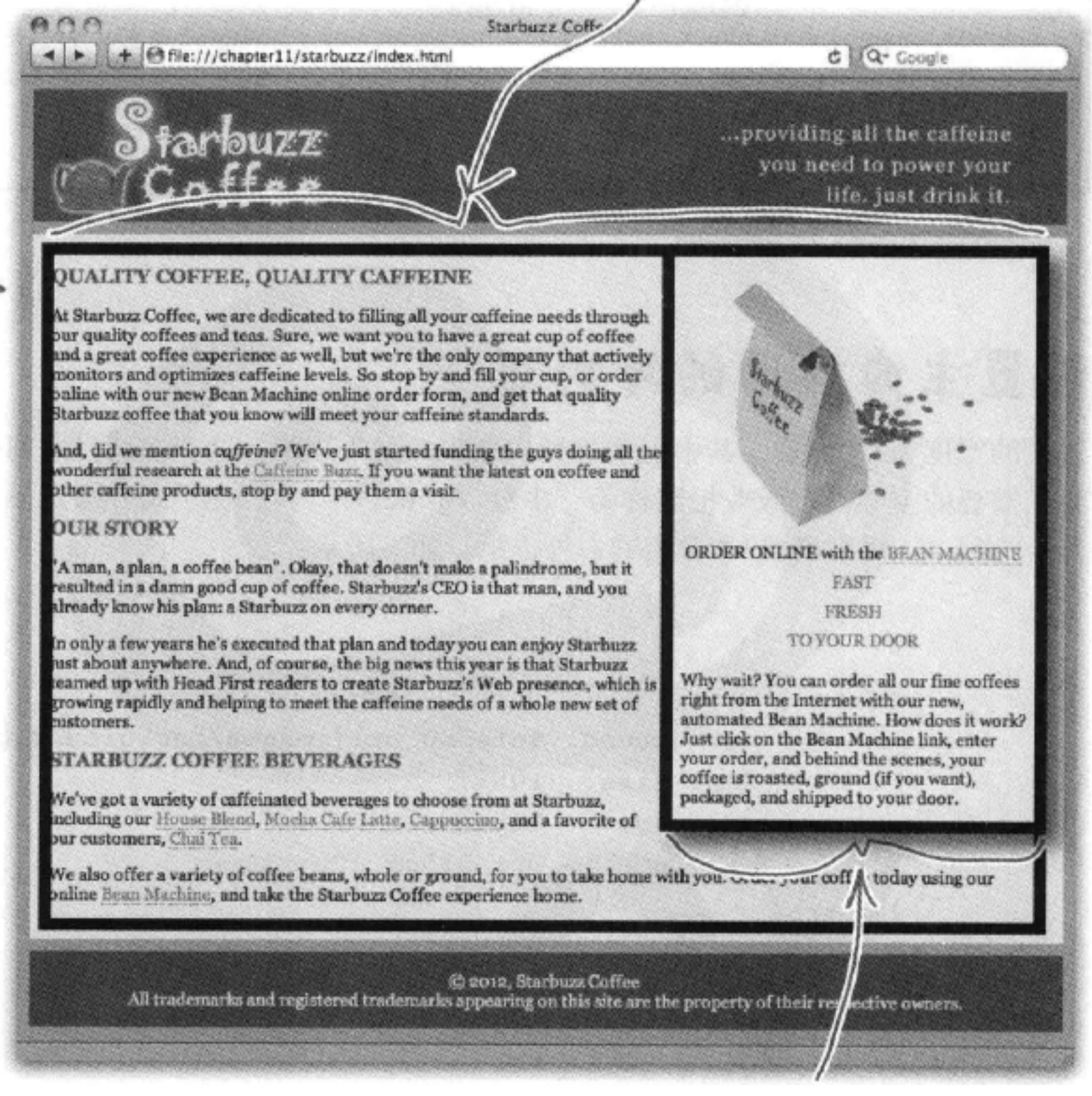
文本围绕在边栏下面，这也使得页面看起来不像有两栏。
嗯，休闲室就是这么做的，也许只能这样吧？

修正两栏问题

现在我们可能已经认识到，页面布局就像一门艺术。尽管有很多技术可以用来摆放块元素，不过没有哪一种技术是完美的。所以，我们要使用一种得到广泛应用的通用技术来解决我们的问题。你会看到，这种方法并不完美，不过大多数情况下都能得到不错的结果。在此之后，你会看到如何使用另外一些方法来解决这个两栏问题，这些方法都各有自己的优点。最重要的是，你要了解这些技术，知道它们为什么有效，这样才能很好地应用这些技术来解决你自己的问题，甚至可以在必要时适当调整。

首先要记住的是，边栏浮在页面上，主内容区在它下面扩展。

那么，如果为主内容区指定适当的外边距，让外边距至少与边栏宽度相同，怎么样？这样一来，它的内容就会扩展到边栏附近的位置，而不会一直扩展下去。



这样主内容区和边栏就会分开，因为外边距是透明的，不会显示背景图像，所以页面本身的背景颜色会透过来，这正是我们想要的（翻回到前几页，你会看到这样做的效果）。

下面让外边距与边栏宽度相等。

Sharpen your pencil

我们希望在主内容区上设置适当的外边距，使它与边栏宽度相同。不过边栏有多大？嗯，我们希望你还没忘上一章的内容。下面是计算边栏宽度所需的全部信息。对照这一章最后给出的答案检查你的结果。

```
#sidebar {
        background: #efe5d0 url(images/background.gif) bottom right;
        font-size:  105%;
        padding:    15px;
        margin:     0px 10px 10px 10px;
        width:      280px;
        float:      right;
}
```

在这个规则中可以找到计算边栏宽度所需的全部信息。

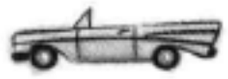
设置主内容区的外边距

边栏的宽度为330像素，其中包括边栏10像素的左外边距，这会在两栏之间提供我们需要的空隙［出版业把这称为“中缝（gutter）”］。在“starbuzz.css”文件中为#main规则增加330像素的右外边距，如下所示：

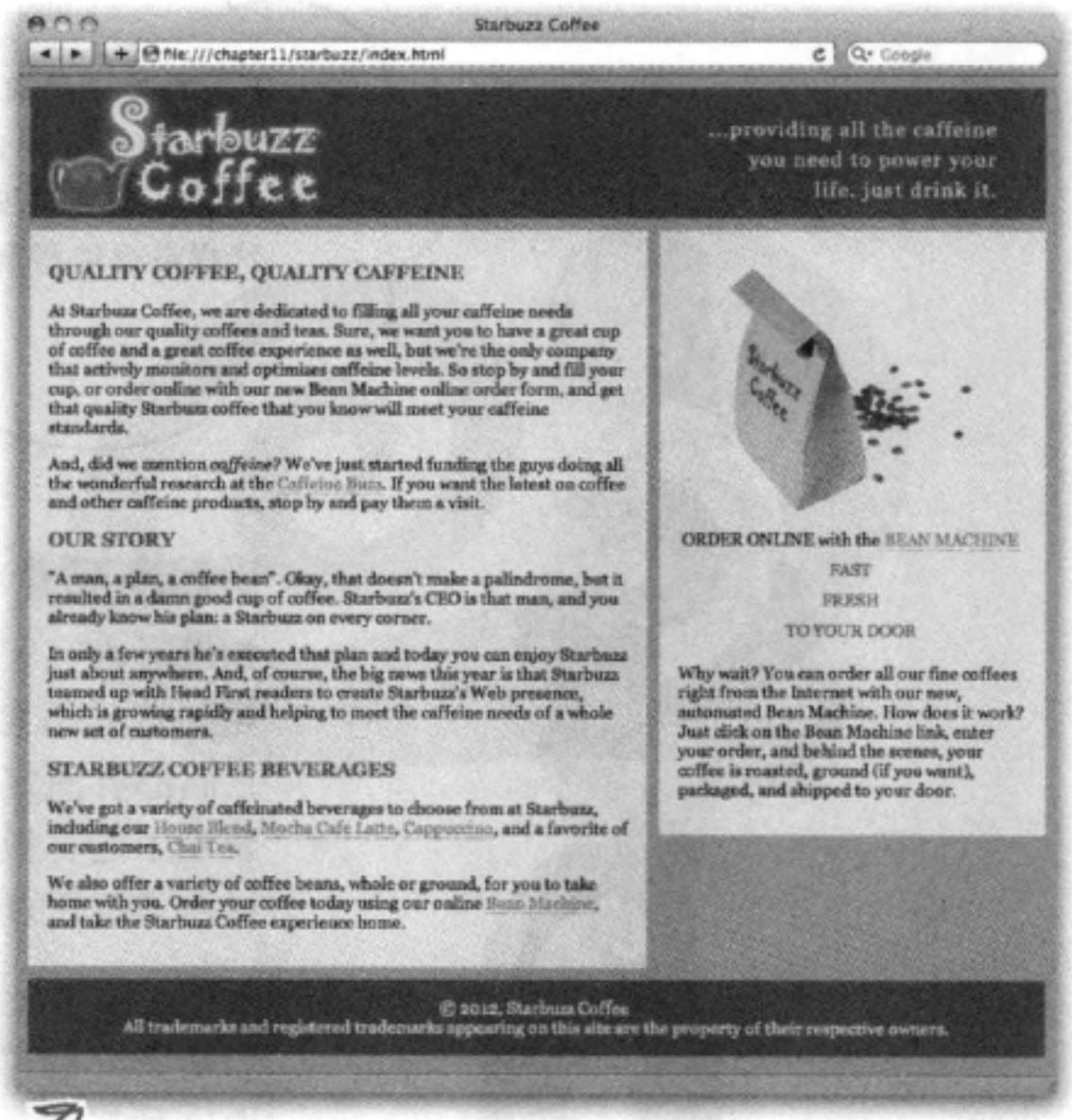
```
#main {
        background: #efe5d0 url(images/background.gif) top left;
        font-size:  105%;
        padding:    15px;
        margin:     0px 330px 10px 10px;
}
```

我们把右外边距改为330像素，与边栏的大小一致。

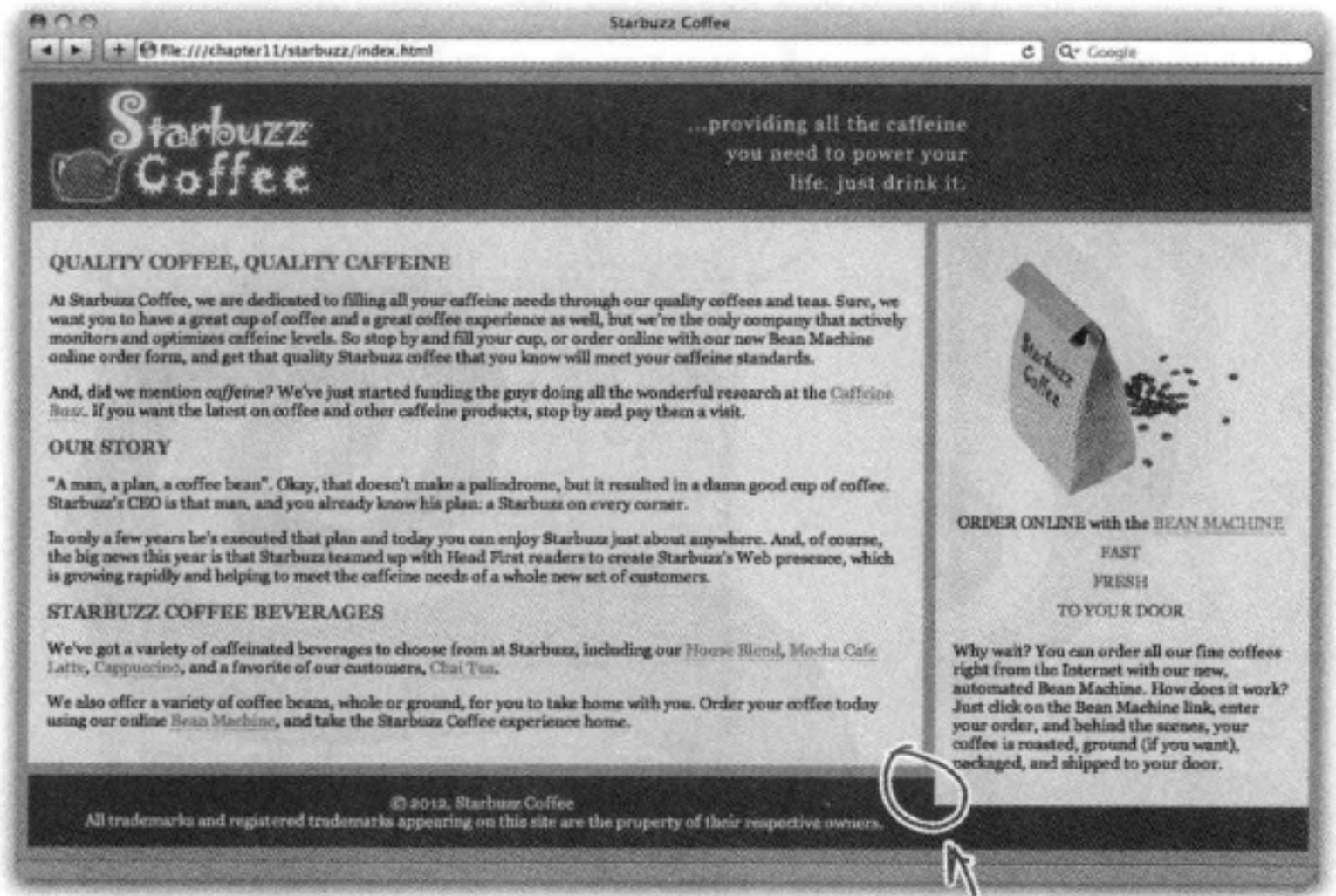
试一试



与以往一样，保存你的“starbuzz.css”文件，然后重新加载“index.html”。现在应该能看到两栏之间有一个漂亮的中缝。下面再来仔细考虑这是如何做到的。边栏向右浮动，所以它会向右浮动到尽可能远，整个 都从正常流中删除，浮在页面最上面。现在主内容 仍占浏览器窗口的整个宽度（因为块元素都是这样），不过我们为它指定了一个外边距，与边栏一样宽，这样会减少内容区的宽度。结果就是我们会得到一个漂亮的两栏页面。尽管你很清楚main 仍然在边栏下面，不过这个秘密只有我们知道，只要你不告诉别人，我们也会帮你保守这个秘密。



通过扩大main 的外边距，我们创造了一种两栏布局的假象，在两栏之间加了一个中缝。



我们遇到一个问题。将浏览器调整到很宽时，页脚和边栏开始重叠。

唉呀，还有一个问题

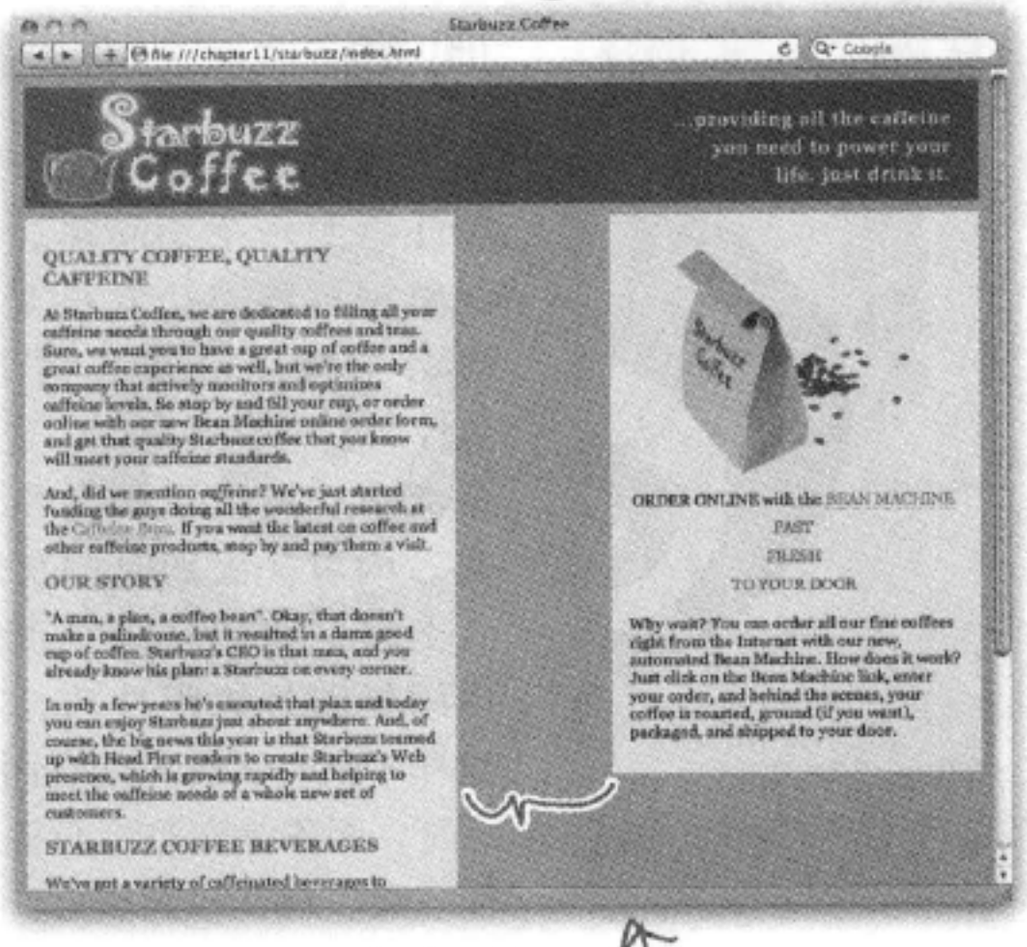
测试这个页面时，你可能已经注意到一个小问题。如果调整浏览器，让它很宽，页脚会上移出现在边栏下面。为什么？嗯，要记住，边栏不在流中，所以页脚会将它忽略，内容区太短时，页脚就会上移。我们可以对页脚使用同样的外边距技巧，不过这样一来，页脚只能在内容区下面，而不是在整个页面的下面。唉，现在该怎么办？

等一下。想办法解决这个问题之前，我想问一句，为什么那么麻烦非得使用外边距？为什么不直接设置主内容区的宽度？这样不也能得到同样的效果吗？

听起来不错……不过你试试看就知道了。

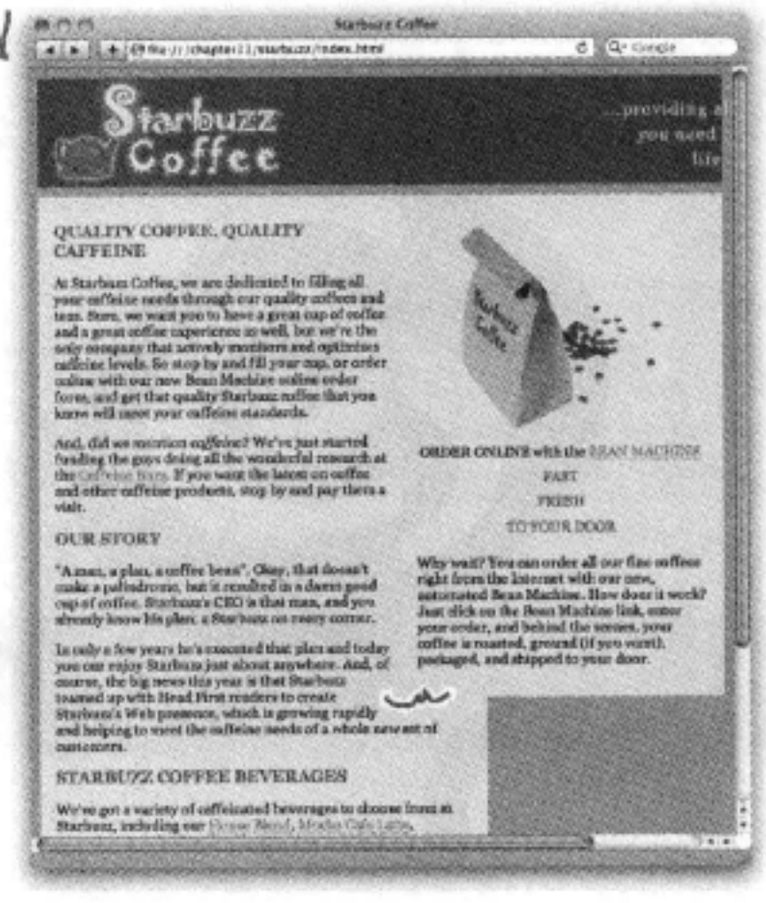
如果对内容区和边栏都设置宽度，会有一个问题，这样不允许页面正确地扩展和收缩，因为它们都有固定的宽度。查看下面的截屏图，它显示了这样做的效果（或者更确切地讲，这里展示了不好的效果）。

不过这种想法是好的。你的思路没有错，在这一章后面介绍“流体和冻结”布局时我们还会考虑这种想法。如果先锁定另外一些部分，就可以利用很多办法来实现你的想法。



浏览器很宽时，这两部分分得太开。

浏览器窗口很小时，这两部分开始重叠。

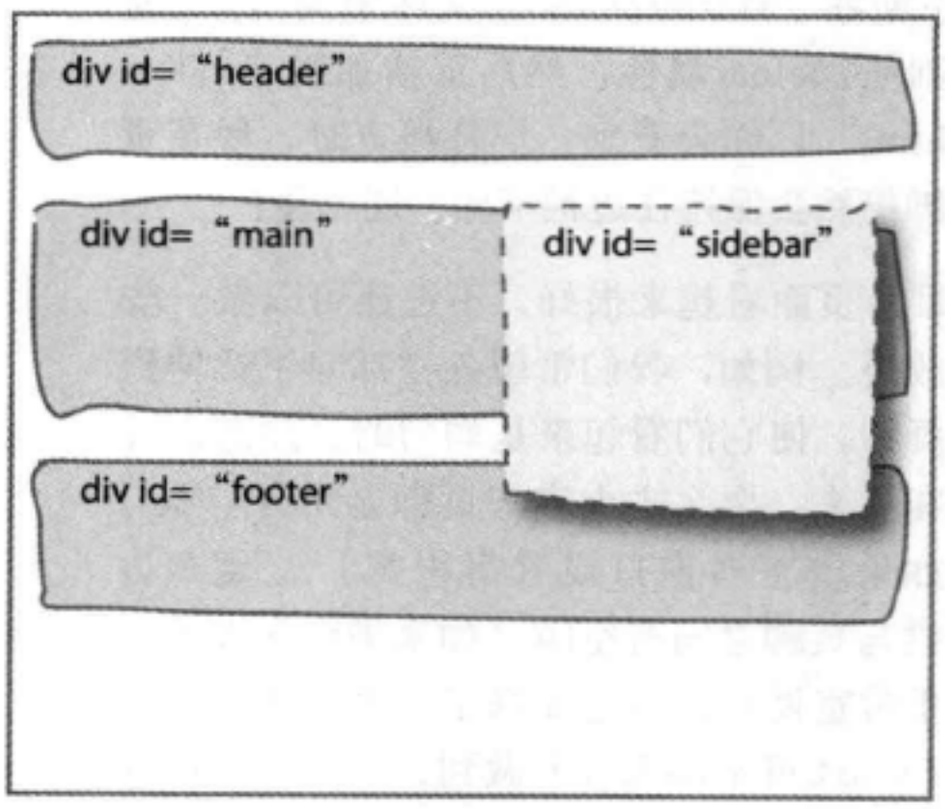


解决重叠问题

要修正这个重叠问题，我们要用到另一个CSS属性：clear属性，之前还没有见过这个属性，它的工作是这样的……

这是我们现在得到的页面。"main" 很短，以至于footer 会上移，与sidebar 重叠。

出现这种情况是因为，边栏已经从流中取出。所以浏览器会正常摆放main和footer ，而忽略边栏（不过要记住，浏览器布置内联元素时，会考虑到边栏的边框，让内联元素围绕着边栏）。

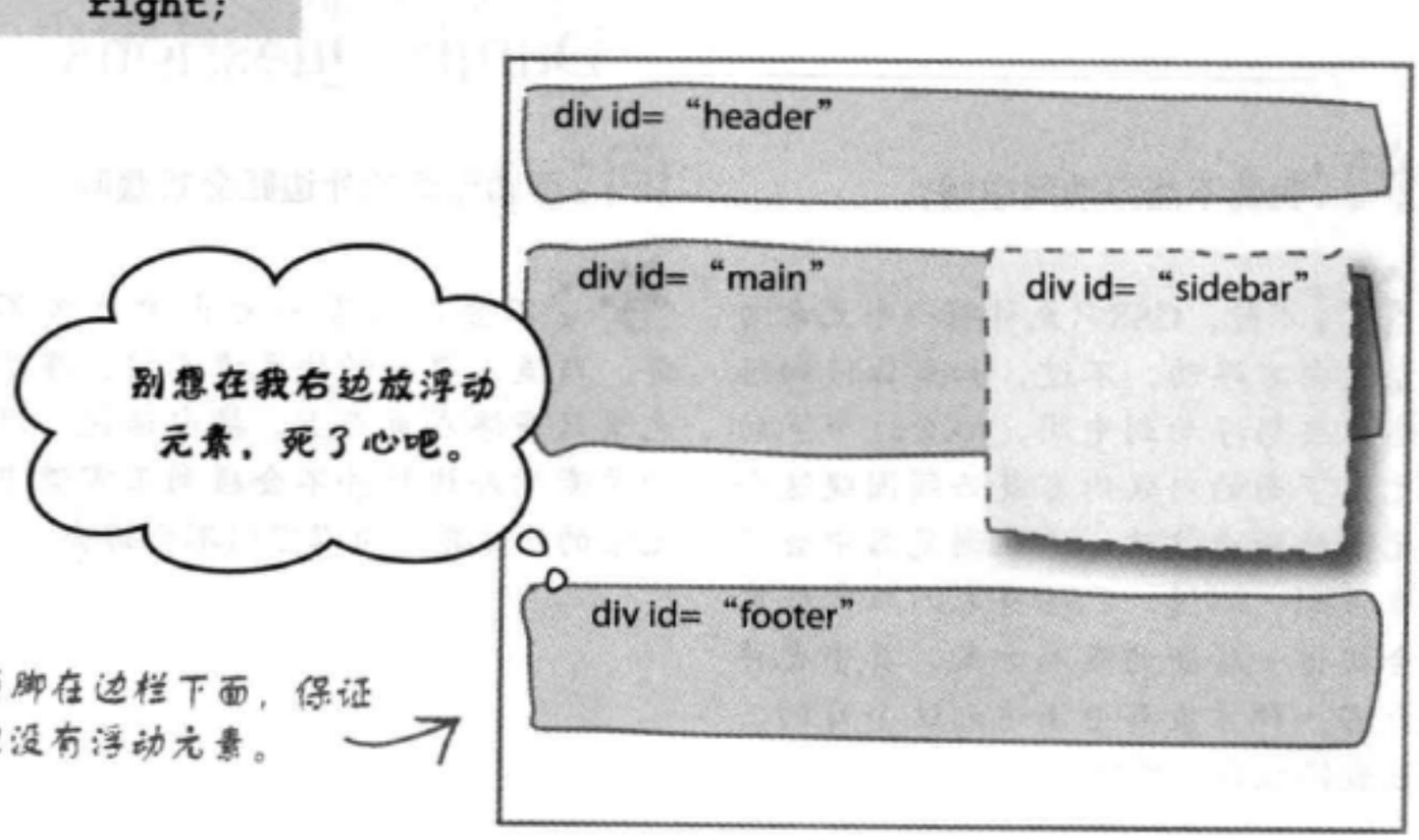


可以使用元素的CSS clear属性来提出请求：当元素流入页面时，在这个元素左边、右边或两边不允许有浮动内容。下面来试一下……

```
#footer {
        background-color: #675c47;
        color:            #efe5d0;
        text-align:       center;
        padding:          15px;
        margin:           10px;
        font-size:        90%;
        clear:            right;
}
```

这里为footer规则增加了一个属性，指出页脚右边不允许有浮动内容。

现在浏览器在页面上放置元素时，它会查看页脚右边有没有一个浮动元素，如果有，就会把页脚下移，直到它右边没有浮动内容为止。现在，不论浏览器多宽，页脚都总在边栏下面。



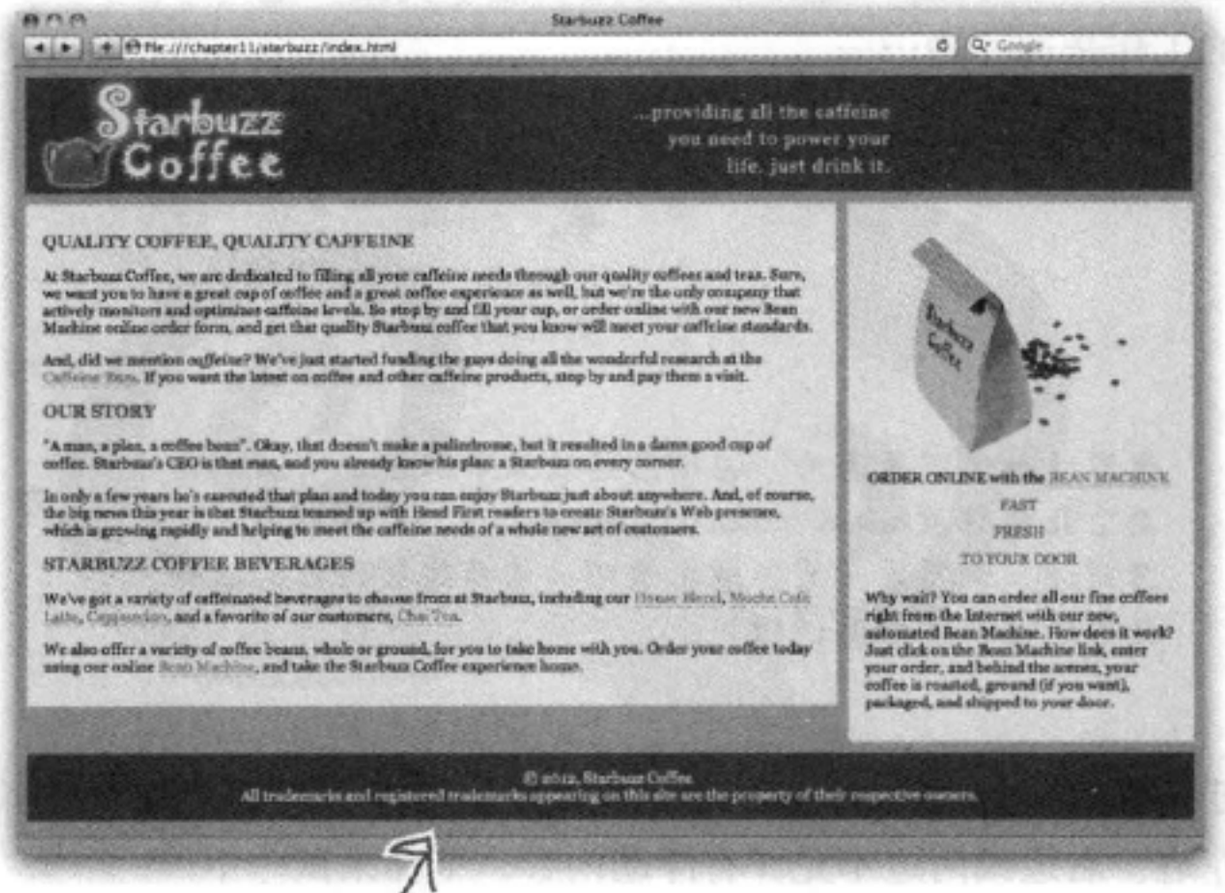
现在页脚在边栏下面，保证它右边没有浮动元素。

试一试

下面在“starbuzz.css”文件中为footer规则增加clear属性，然后重新加载“index.html”。你会看到，屏幕很宽时，现在页脚仍然会保持在边栏下面。还不错！

目前页面看起来很好，不过还可以做一些改进。例如，我们希望各栏都向下延伸到页脚，使它们看起来是均匀的。注意，现在看来，要么主内容与页脚之间有空隙（如果浏览器窗口设置得很宽），要么边栏与页脚之间有空隙（如果浏览器设置为正常宽度）。不过要修正这个问题，使用`float`可不那么容易做到，所以我们换种思路，来看使用其他CSS技术设计页面布局的方法。你会看到，CSS中有很多种方法，每种方法都有其长处和短处。

对你来说，重要的是要了解这些技术，从而能够在需要的时候和场合适当地加以应用。



现在我们的页脚问题已经解决了。页脚总在边栏下面，而不论浏览器设置得多窄或多宽。

there are no Dumb Questions

问：我能不能浮动到中间？

答：不行，CSS只允许将一个元素向左或向右浮动。不过，如果你仔细想想，要想浮动到中间，那么这个浮动元素下面的内联内容就必须围绕这个元素的两边流动，这在浏览器中会很难做到。不过，CSS将来的版本可能会提出一些新的布局方案，其中也许会有一种方案有办法达到这个目的，让我们拭目以待吧。

问：浮动元素的外边距会折叠吗？

答：不会，很容易看出为什么不会。与流入页面的块元素不同，浮动元素只是浮在页面上。换句话说，浮动元素的外边距并不会碰到正常流中元素的外边距，所以它们不会折叠。

不过这引出一个很好的想法，可以找出布局中的一个常见错误。如果有一个主内容区和一个边栏，通常会对它们分别设置一个上外边距。然后，如果浮动边栏，它仍然有一个外边距，但这个外边距不会再与它上面的空间折叠。这样一来，如果你不记得浮动元素不会折叠外边距，最后边栏和主内容很可能会有不同的外边距。

问：可以浮动内联元素吗？

答：可以，当然可以。最好的例子（也是最常见的例子）就是浮动图像。可以试一下，将图像在段落中向左或向右浮动，你会看到文本会围绕在它周围。不要忘记增加内边距，给图像一点空间，可能还可以增加一个边框。还可以浮动任何其他内联元素，不过这种情况不常见。

问：能不能把浮动元素想成是被块元素忽略的元素，但是内联元素知道它们在哪里，这样考虑对吗？

答：可以，这样考虑很好。嵌套在块元素中的内联内容总会围绕着浮动元素，内联元素会留意浮动元素的边界，而块元素会正常流向页面。有一个例外，对一个块元素设置clear属性时，这会导致块元素下移，直到它右边、左边或两边没有浮动元素挨着它（具体要取决于clear的值）。

没错。出现这种情况是由于我们是按这个顺序摆放 的。

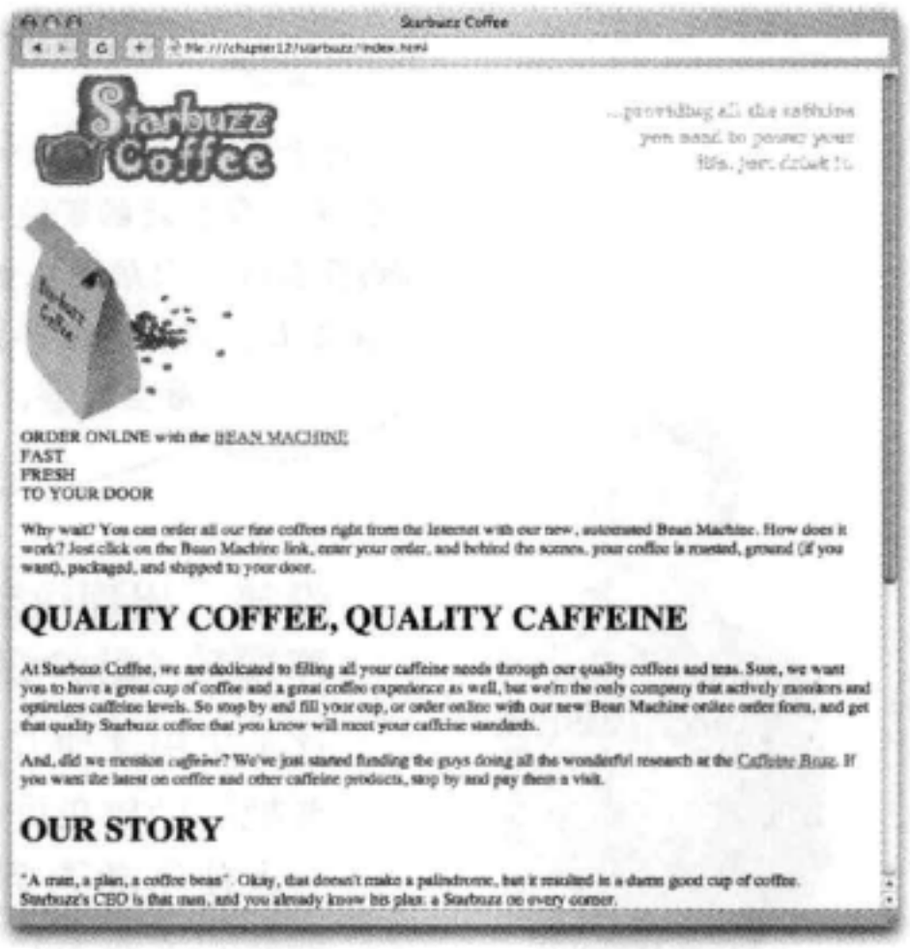
这是采用这种方式设计页面的缺点之一，由于我们需要把边栏放在页眉下面，而且要在主内容之前，如果有人使用功能受限的浏览器（PDA、小的移动设备、屏幕阅读器等），他们看到的页面就会采用我们写页面时使用的元素顺序，边栏在最前面。不过，大多数人更希望先看到主内容，然后才是某种形式的边栏或导航。

所以，下面来看另一种设计方法，这就要用到之前你提到的想法：对主内容使用float “left”。

看呐，没有CSS！

你想知道如果用户在不利的条件下（比如使用的浏览器不支持CSS）会看到怎样的页面吗？可以打开你的“index.html”文件，从<head>删除<link>，保存文件，然后在浏览器中重新加载页面。现在你可以看到这些元素的实际顺序（或者从屏幕阅读器听到的顺序）。来试一试。完成之后一定要把<link>元素放回原位（毕竟，这一章是关于CSS的）。

这是没有CSS的Starbuzz页面。大多数方面都还好。这个页面还是很有可读性，不过Bean Machine出现在主内容的前面，这可能不是我们想要的。



右紧左松

下面让Starbuzz页面换一下，让主内容向左浮动。你会看到右紧左松记忆法在CSS世界里也是适用的……嗯，起码适用于我们的边栏。我们会如下转换页面……只需要简单的几步。

第1步：从边栏开始。

实际上我们要交换边栏和主内容区的角色。内容区要有一个固定的宽度，而且要设置为浮动，边栏则要围绕着内容。我们还要使用同样的外边距技术让这两个部分视觉上是分离的。不过，开始改变CSS之前，先打开你的“index.html”文件，将“sidebar” 向下移到“main” 下面。完成之后，需要对sidebar CSS规则做以下修改：

```
#sidebar {
        background:   #efe5d0 url(images/background.gif) bottom right;
        font-size:    105%;
        padding:      15px;
        margin:       0px 10px 10px 470px;
        width:        200px;
        float:        right;
}
```

由于边栏现在要在主内容下面流动，所以需要把大外边距移到边栏上（也就是要为边栏设置很大的外边距）。主内容区的总宽度为470像素（可以利用你的空闲时间自己完成这个计算。计算方法与计算边栏的总宽度时是一样的。你应该知道我们打算将主内容区的宽度设置为420像素）。

要为主内容 设置一个固定的宽度，所以删除边栏的width属性，另外删除float属性。

第2步：处理主内容区。

现在需要浮动主内容 。方法如下：

```
#main {
        background: #efe5d0 url(images/background.gif) top left;
        font-size:    105%;
        padding:      15px;
        margin:       0px 10px 10px 10px;
        width:        420px;
        float:        left;
}
```

我们要把右外边距从330像素改回为10像素。

需要设置一个明确的宽度，因为我们打算浮动这个元素。这里使用420像素。

我们要把主内容 浮动到左边。

第3步：处理页脚。

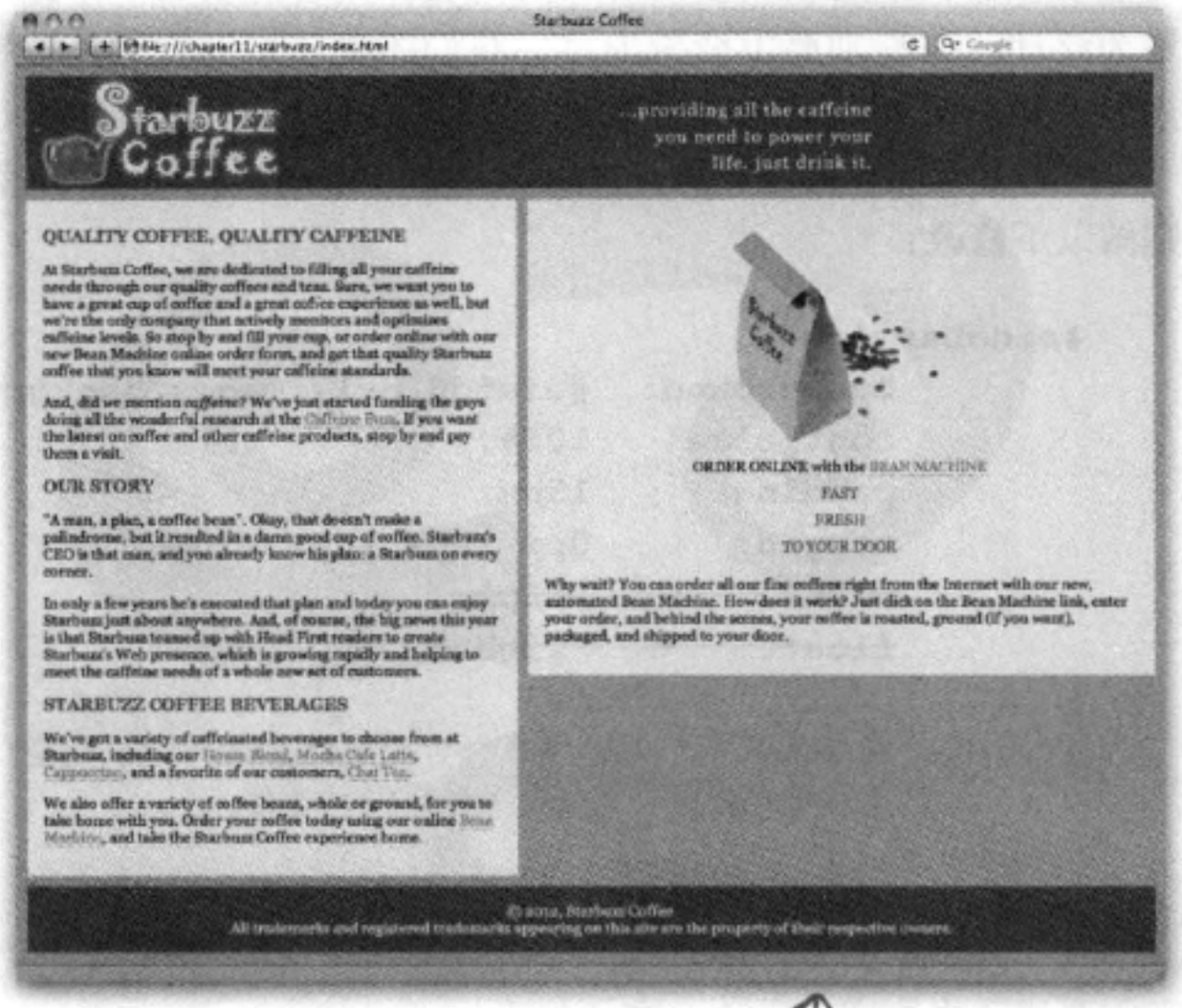
现在只需要调整页脚，让它清空左边的所有浮动元素，而不是右边。

```
#footer {
        background-color: #675c47;
        color:            #efe5d0;
        text-align:       center;
        padding:          15px;
        margin:           10px;
        font-size:        90%;
        clear:            left;
}
```

改变clear属性，现在值为left而不是right。这样一来，页脚就会给主内容区让路。

快速测试

我们已经说过，这种将内容浮动到左边的方法存在一些问题。在继续学习后面的内容之前，先来做一个快速的测试，看看这种方法具体如何工作。完成对“starbuzz.css”文件的修改，然后在浏览器中重新加载“index.html”。好好感觉一下窗口大小调整得很窄、正常和很宽时页面会有怎样的表现。



实际上，这看起来还不错，现在 的顺序是正确的。不过边栏过于扩展，这不太好，看起来还是固定更好一些。边栏通常用于导航，所以边栏扩展时看起来不是很好。

将sidebar 浮动到右边时，设计很漂亮也很紧凑，允许内容扩展。不过如果将主内容浮动到左边，这个设计感觉过于松散，它将允许边栏扩展。

BRAIN POWER

从设计来讲，第一种设计会比较好，而从信息角度讲，第二种会更好一些（因为 的摆放顺序是正确的）。有没有一种方法可以同时得到这两方面的好处：能不能让边栏有一个固定的宽度，而且main 仍然是HTML中的第一个元素，可以做到吗？要达到这个目的，需要在设计上如何取舍？

流体与冻结设计

目前为止，我们采用的所有设计都称为流体布局（liquid layouts），因为不论我们将浏览器调整到多大的宽度，布局都会扩展，填满整个浏览器。这些布局很有用，因为通过扩展，它们会填充可用空间，使用户能够充分利用他们的屏幕空间。不过，有时让布局锁定可能更重要，这样一来，当用户调整屏幕大小时，你的设计仍能保持原样。这称为冻结布局（frozen layouts）。冻结布局会锁定元素，让它们冻结在页面上，这样这些元素根本不能移动，我们就能避免由于窗口扩展带来的很多问题。下面来试一试冻结布局。

要把当前页面变成一个冻结页面，只需要对HTML做一点补充，另外在CSS中增加一个新规则。

HTML修改

在你的HTML中，要增加一个新的 元素，id为“allcontent”。顾名思义，这个 要包围页面中的所有内容。所以将这个开始 标记放在header 前面，结束标记放在footer 下面。

```
<body>
  <div id="allcontent">
    <div id="header">
     ...rest of the HTML goes here...
    </div>
  </div>
</body>
```

增加一个新的 ，id为“allcontent”，它将包围<body>中的所有其他元素。

这个 结束footer 。

CSS修改

现在我们要限制“allcontent” 中所有元素和内容的大小，使它们有一个固定的宽度：800像素。下面的CSS规则可以达到这个目的：

```
#allcontent {
        width:              800px;
        padding-top:        5px;
        padding-bottom:     5px;
        background-color:   #675c47;
}
```

我们要把“allcontent”的宽度设置为800像素。这样做的效果是将其中包含的所有内容限制在800像素范围内。

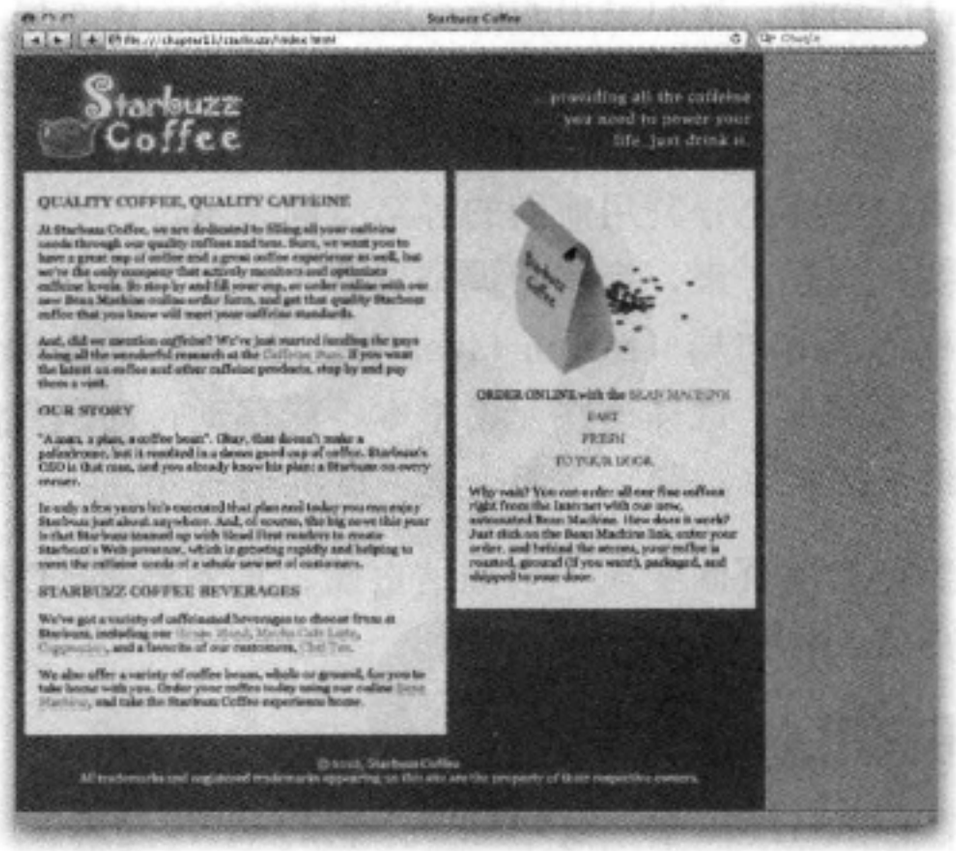
由于这是第一次指定这个 的样式，所以下面增加一些内边距，并让它有自己的背景颜色。你会看到这将有助于把整个页面联系在一起。

外围的“allcontent” 宽度总是800像素，即使浏览器大小调整了，这个宽度也不变，这样一来，我们就有效地将这个 以及其中包含的所有内容冻结在页面上。

冻结测试

将这个规则增加到“starbuzz.css”的最下面，然后重新加载“index.html”。现在你可以看到为什么我们会把它叫做冻结布局。不论浏览器怎么调整大小，它都不会移动。

现在不论浏览器如何调整大小，“allcontent” 的宽度都是800像素。另外，由于所有其他 都包含在“allcontent”中，它们也会限制在这800像素的空间中。所以这个页面实际上“冻结”为800像素。



这当然能解决边栏扩展的问题，看起来也还不错。不过，浏览器很宽时，这会有点奇怪，因为右边有很多空白空间。

不过我们还没有完工，还有一点提升空间。

流体和冻结之间的状态是什么？当然是凝胶！

冻结布局有一些优点，不过浏览器加宽时看起来也不太好。我们已经做了一个修正，这是Web上很常见的一种设计。这种设计介于冻结和流体之间，名字也很贴切，叫做凝胶（Jello）。凝胶布局会锁定页面中内容区的宽度，不过会将它在浏览器中居中。实际上与其解释这种布局会有怎样的表现，不如先把布局改为凝胶布局来试一试，这样理解起来会更容易，下面就动手做这个修改吧：

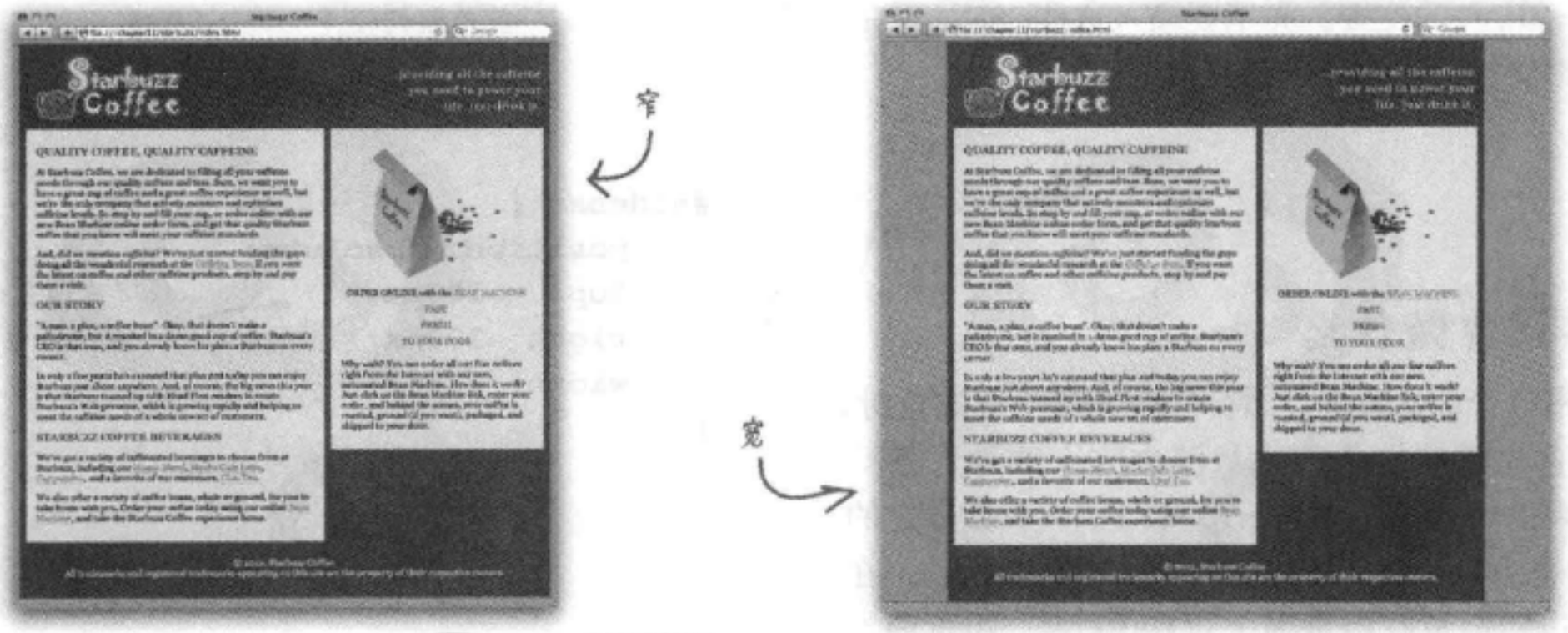
```
#allcontent {
        width: 800px;
        padding-top:    5px;
        padding-bottom: 5px;
        background-color: #675c47;
        margin-left:  auto;
        margin-right: auto;
}
```

“allcontent” 上没有固定的左右外边距，我们将外边距设置为“auto”。

如果你还记得，我们讨论指定内容区宽度为“auto”时，浏览器会根据需要扩展内容区。外边距为“auto”时，浏览器会确定正确的外边距是多少，另外还会确保左和右外边距相同，所以内容会居中。

测试凝胶布局

为“starbuzz.css”文件增加两个外边距属性，然后重新加载页面。现在试着调整浏览器的大小。很不错，是不是？



所以，如果我们希望内容有正确的顺序，就必须使用扩展的边栏，或者必须使用凝胶布局，是吗？有没有其他办法？

利用CSS，实现一种布局通常有很多种方法，分别有自己的长处和短处。实际上，我们正打算介绍另一种创建两栏布局的常用技术，它能让内容保持正确的顺序，同时避免流体布局存在的一些问题。不过，你会看到，这种方法同样要做一些折衷。

利用这种新技术，我们不会浮动元素。实际上，我们要使用CSS的一个特性，可以在页面上精确地定位元素。这称为绝对定位（absolute positioning）。我们还可以使用绝对定位来实现一些漂亮的效果，而不只是建立多栏布局，后面还会看到这样一些例子。

为此，我们要回到这一章开始时给出的原始HTML和CSS。在“chapter11/absolute”文件夹中可以找到这些文件的最新副本。一定要再看一看这些文件，记住它们原来是什么样子。应该记得我们有一大堆 ：一个对应页眉，一个对应主内容，一个对应页脚，还有一个对应边栏。另外要记住在原来的HTML中，sidebar 放在主内容区下面，这是我们认为的最佳位置。

绝对定位如何工作

下面先搞清楚绝对定位会做些什么，另外是怎么做的。这里有一个简短的CSS，使用绝对定位来指定sidebar 的位置。先不要输入这些CSS，现在我们只希望你对它如何工作有点认识：

首先我们使用position属性指定这个元素要绝对定位。

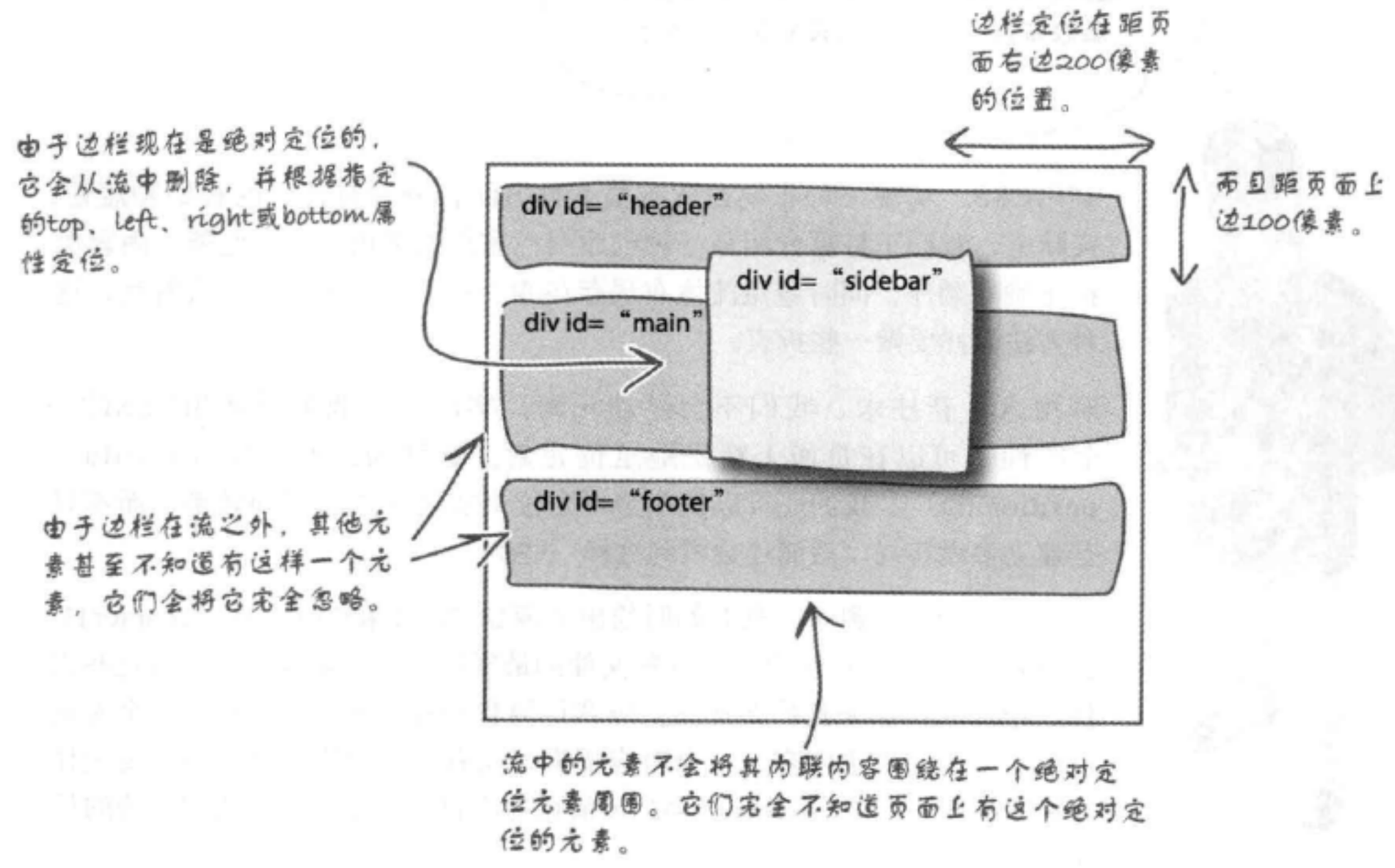
```
#sidebar {
    position: absolute;
    top:    100px;
    right: 200px;
    width: 280px;
}
```

接下来设置top和right属性。

另外为 指定一个宽度。

这个CSS会做什么

现在来看这个CSS做些什么。一个元素绝对定位时，浏览器首先要做的是将它从流中完全删除，然后浏览器将这个元素放置在top和right属性指定的位置上(也可以使用bottom和left指定位置)。在这里，边栏会放在距页面上边100像素、距页面右边200像素的位置上。我们还设置了 的宽度，这与设置浮动元素样式时一样。



绝对定位的另一个例子

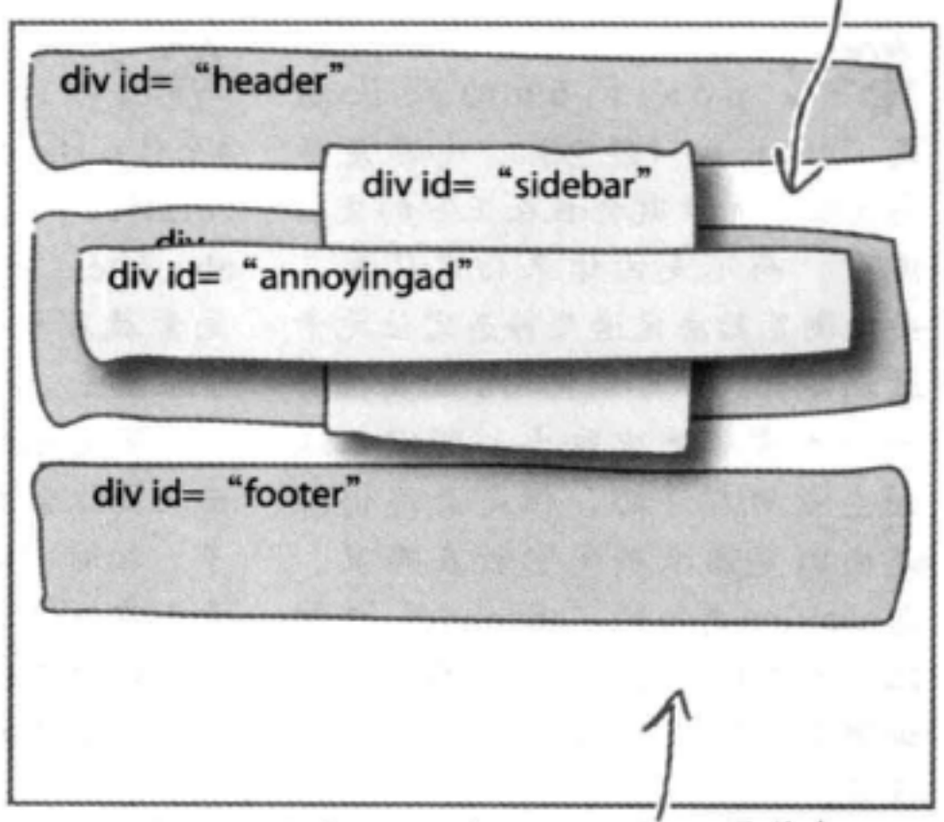
下面来看另一个例子。假设我们有另一个 ，id为“annoyingad”。可以这样指定它的位置：

```
#annoyingad {
    position: absolute;
    top:    150px;
    left:   100px;
    width: 400px;
}
```

这个讨厌的广告位于距左边100像素、距上边150像素的位置上。它比边栏稍宽一点，宽度为400像素。

与边栏一样，我们把这个“annoying ad” 精确地放在页面上的某个位置。在它下面，页面正常流中的所有元素根本不知道页面上有这个绝对定位元素。这与浮动元素有所不同，因为流中的元素会调整它们的内容来适应浮动元素的边界。不过绝对定位元素对其他元素没有任何影响。

现在有另一个 ，也是绝对定位，距左边100像素，距上边150像素。



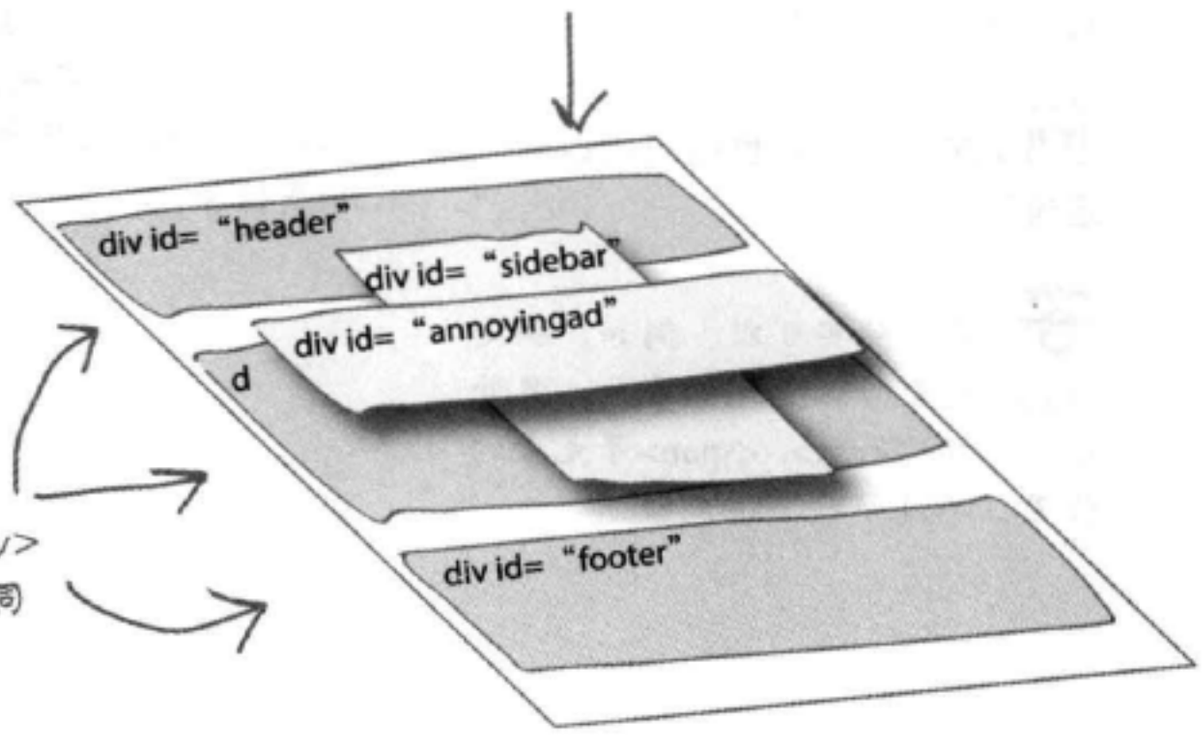
注意annoyingad 覆盖在sidebar 之上。

谁在上面？

关于绝对定位元素还有一个有意思的问题，绝对定位元素可以分层放置，一个元素可以放在另一个绝对定位元素上面。不过，如果页面上同一个位置有多个绝对定位元素，你怎么知道它们是如何分层的呢？换句话说，谁在上面？

每个定位元素都有一个名为z-index的属性，这会指定它在一个虚拟z轴上的位置（上面的元素“更靠近”你，z-index更大）。

sidebar和annoyingad 在页面上分层显示，annoyingad的z-index大于sidebar，所以annoyingad在上面。



header, main和footer 都在流中，所以在页面的同一个平面上。

there are no Dumb Questions

问：position属性的默认设置是什么？

答：position的默认值是“static”（静态）。如果是静态定位，元素就会放在正常的文档流中，而不是由你来指定位置，要由浏览器决定这些静态定位元素放在哪里。你可以使用float属性将一个元素从流中取出，可以让这向左或向右浮动，但是最终仍然是由浏览器来确定它放在哪里。与position属性的“absolute”值相比，使用绝对定位时，将由你来告诉浏览器究竟应该把元素放在什么位置。

问：我是不是只能指定 的位置？

答：可以对任何元素指定绝对位置，包括块元素和内联元素。只是要记住，一个元素绝对定位时，会把它从页面的正常流中删除。

问：这么说，我也能对内联元素定位了？

答：对，确实可以。例如，指定<img>元素的位置就很常见。另外还可以指定, 等元素的位置，不过一般不这么做。

问：除了static和absolute，还有没有其他的position属性值？

答：实际上，position属性有4个值：static, absolute, fixed和relative。你已经了解了static和absolute。固定（Fixed）定位是将元素放在相对于浏览器窗口的一个位置上（而不是相对于页面），所以固定元素永远也不会移动。后面几页你会看到固定定位的一个例子。相对（Relative）定位会让元素正常地流入页面，不过在页面上显示之前要进行偏移。相对定位常用于更高级的定位和特殊效果。

问：必须像对浮动元素一样为绝对定位元素指定宽度吗？

答：不，绝对定位元素不用指定宽度。不过，如果没有指定宽度，默认的，块元素会占浏览器的整个宽度（当然要减去你指定的距左边或右边的偏移量）。这可能正是你想要的，也可能不是。所以如果你想改变这种默认行为，就要设置width属性值。

问：指定位置时必须使用像素来指定吗？

答：不，指定元素位置还有一种常用方法——可以使用百分数。如果使用百分数，改变浏览器宽度时，元素的位置可能会改变。例如，如果浏览器设置为800像素宽，元素的left位置设置为10%，那么元素就会放在距浏览器窗口左边80像素的位置。不过，如果你的浏览器大小调整为400像素宽，宽度就会缩减为400像素的10%，也就是距浏览器窗口左边40像素。

百分数还常用于指定宽度。如果不需要为元素或外边距指定特定的宽度，就可以使用百分数，让主内容区和边栏的大小更为灵活。在两栏和三栏布局中会经常看到这种用法。

问：是不是必须知道如何使用z-index才能使用绝对定位？

答：不用，z-index通常用在一些CSS高级用法中，特别是涉及Web页面脚本时，所以这有点超出了这本书的范围。不过这也是绝对定位中的一部分，所以最好对z-index有点了解（稍后我们还会再对z-index作简单的介绍）。

使用绝对定位

前面创建了Starbuzz页面的浮动版本，现在我们要用类似的技术创建一个两栏的Starbuzz页面。不过，这一次我们要用绝对定位。以下是我们要做的事情：

1. 首先，将sidebar 绝对定位。实际上，我们要把它放在之前使用浮动技术时的同一个位置上。
2. 接下来，为主内容指定另一个更大的外边距，让边栏位于外边距空间之上。
3. 最后，做一个测试，与浮动版本比较，看看有什么不同。

修改Starbuzz CSS

我们的HTML已经准备好了，sidebar 已经放在我们希望的正确位置上（在重要的主内容下面）。现在要做的就是对CSS做一些修改，我们要让边栏绝对定位。打开“starbuzz.css”文件，对sidebar规则做以下修改：

记住，我们要使用文件原来的版本，可以在“chapter11/absolute”文件夹中找到。

可以在“absolute”文件夹中修改，或者将文件“index.html”和“starbuzz.css”复制到“starbuzz”文件夹，在那里完成修改，我们就是这样做的。

```
#sidebar {
    background: #efe5d0 url(images/background.gif) bottom right;
    font-size:  105%;
    padding:    15px;
    margin:     0px 10px 10px 10px;
    position:   absolute;
    top:        128px;
    right:      0px;
    width:      280px;
}
```

OK，现在要指定边栏绝对定位，放在距页面上边128像素、右边0像素的位置上。我们还希望边栏有一个宽度，所以让它与浮动版本有相同的宽度：280像素。

稍后你会看到这个128是从哪里来的……

距右边0像素可以确保边栏紧挨着浏览器的右边。

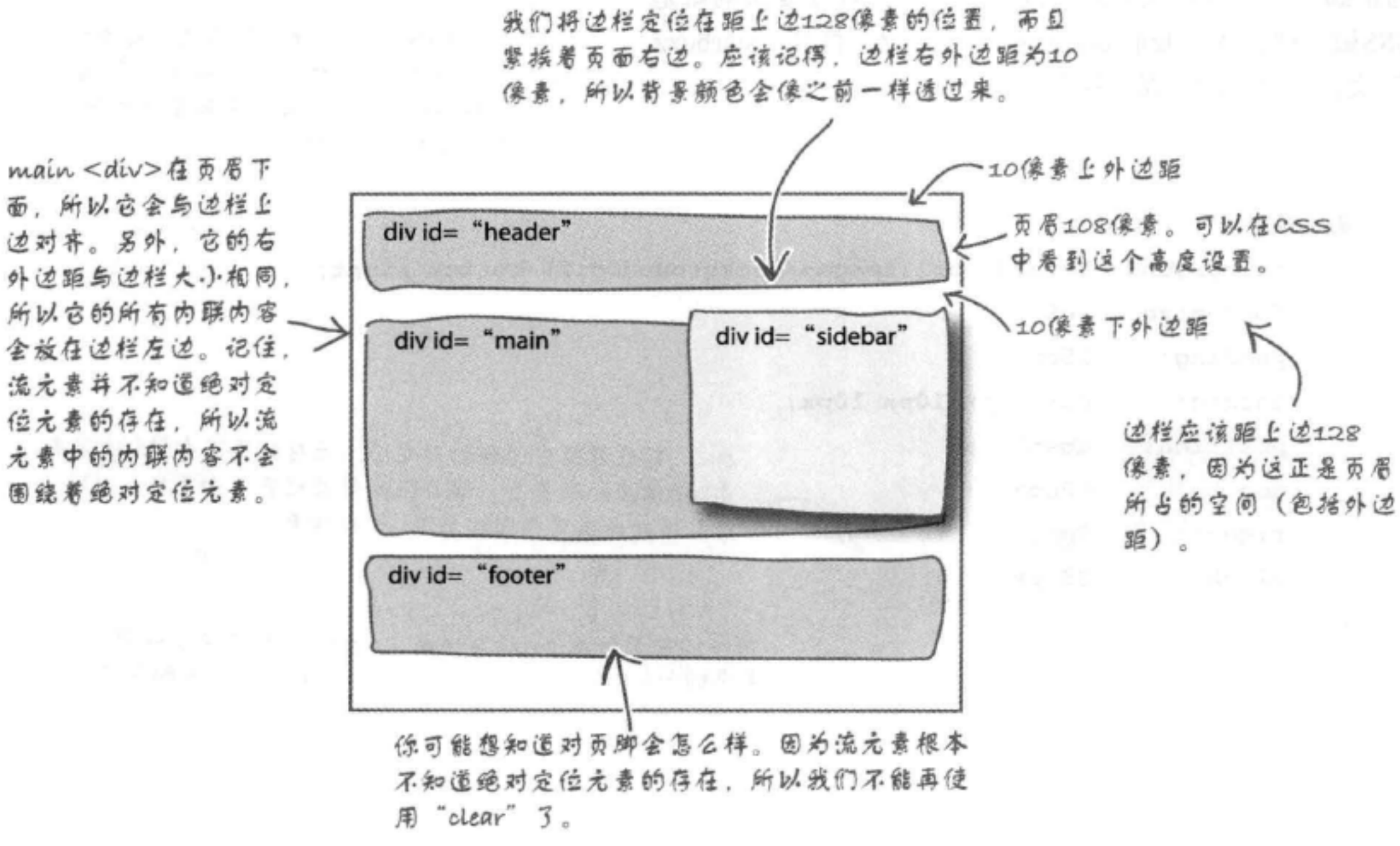
现在只需要调整main

实际上，没有多少调整工作要做。我们只是要像浮动版本中一样增加一个外边距。所以，将main 的右外边距改为330像素，就像上一次一样。

```
#main {
    background: #efe5d0 url(images/background.gif) top left;
    font-size:  105%;
    padding:    15px;
    margin:     0px 330px 10px 10px;
}
```

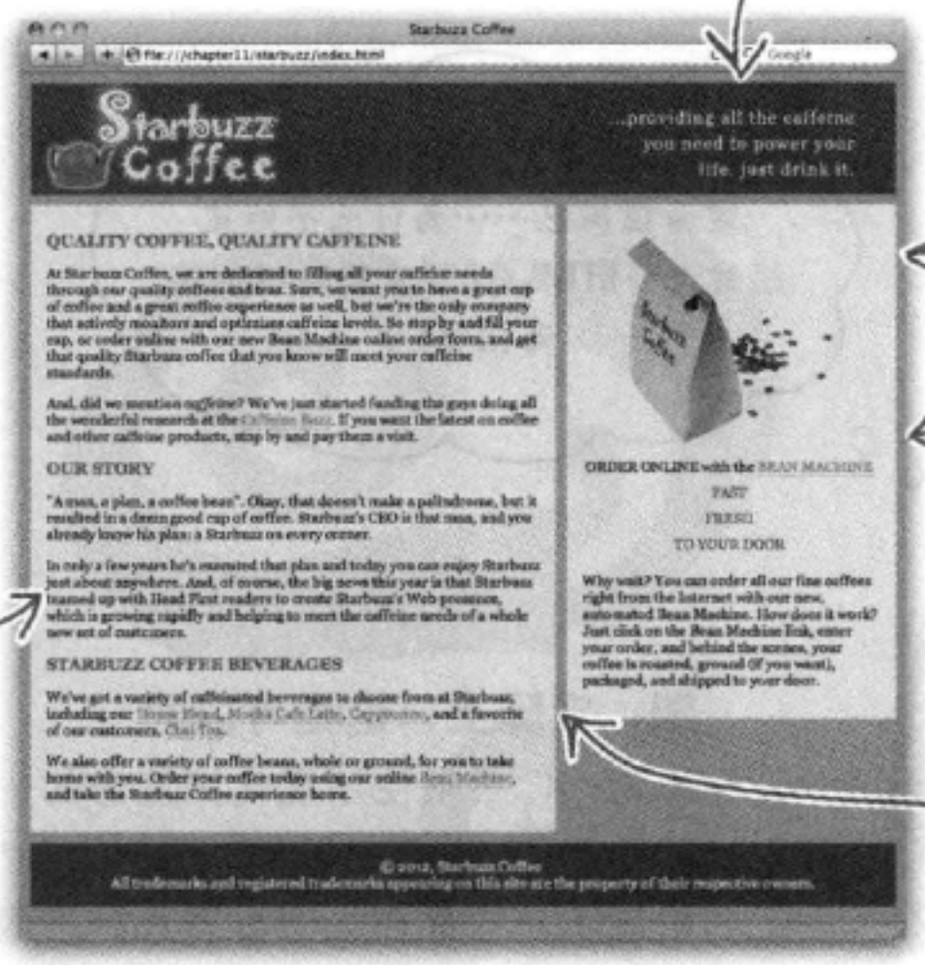
通过为main 指定一个大的外边距，会留出一些空间来放置边栏。这与浮动版本中使用的技术是一样的。唯一的区别是sidebar 放在外边距空间上。

现在只需要修改外边距，然后保存。不过在完成测试之前，下面来考虑这对绝对定位的边栏有什么影响。



来测试绝对定位

保存这个新的CSS，然后在浏览器中重新加载“index.html”。下面来看结果：



哇，这看起来与浮动版本惊人的相似。不过，你知道现在边栏是绝对定位。

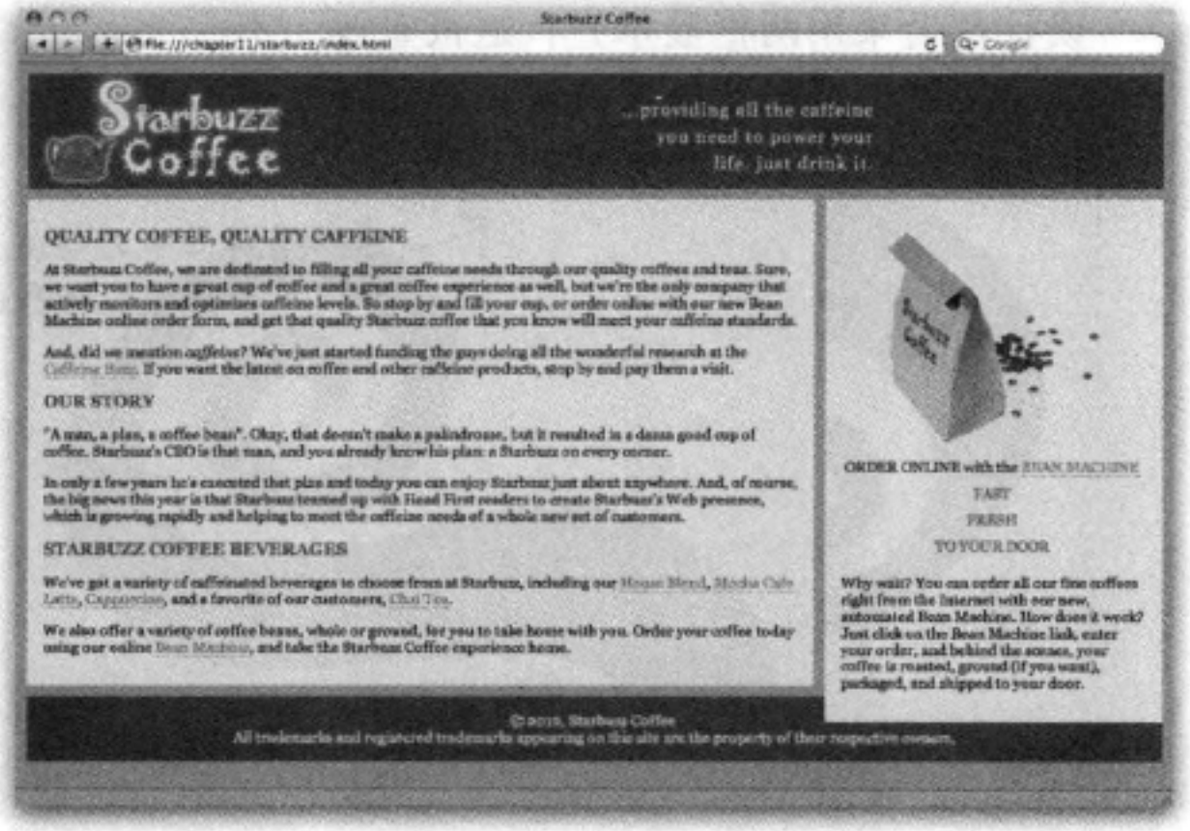
调整浏览器大小时，边栏总是在距上边128像素的位置，而且紧挨着页面右边。

边栏10像素的右外边距，所以它与页面边界之间有空隙。

主内容区的右外边距与边栏宽度相同，所以边栏可以放在那个外边距空间之上。

两栏之间也有不错的中缝。

不过，现在页脚有问题了。浏览器足够宽时，绝对定位的边栏会向下覆盖页脚的上面部分。遗憾的是，这一次我们不能依靠`clear`属性，因为流元素完全不知道绝对定位元素的存在。



浏览器很宽时，主内容区的垂直空间缩小，边栏会向下覆盖页脚。

嗯，实际上，这是可以的……

为此，你必须使用现代浏览器的一个很新的功能：CSS表格显示。这是什么？CSS表格显示允许你在一个有行和列的表格中显示块元素（稍后会看到），另外，通过将内容放在一个CSS表格中，可以很容易地用HTML和CSS创建多栏设计。

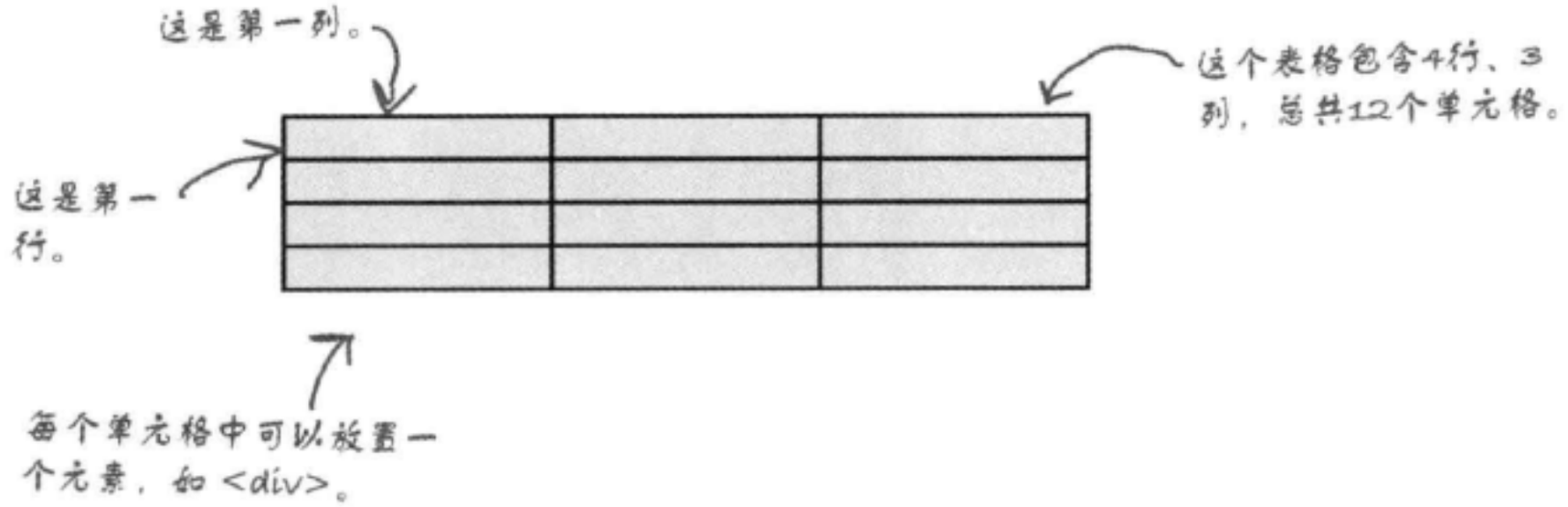
目前，所有现代浏览器都支持这个特性。

现在你可能在想“为什么你不早点告诉我们？”嗯，你应该先了解浏览器如何建立流以及如何显示内容（因为并不是每一个设计都会采用两栏显示），这很重要。不过，既然你已经了解了布局，现在可以使用CSS表格显示来调整我们的页面了。

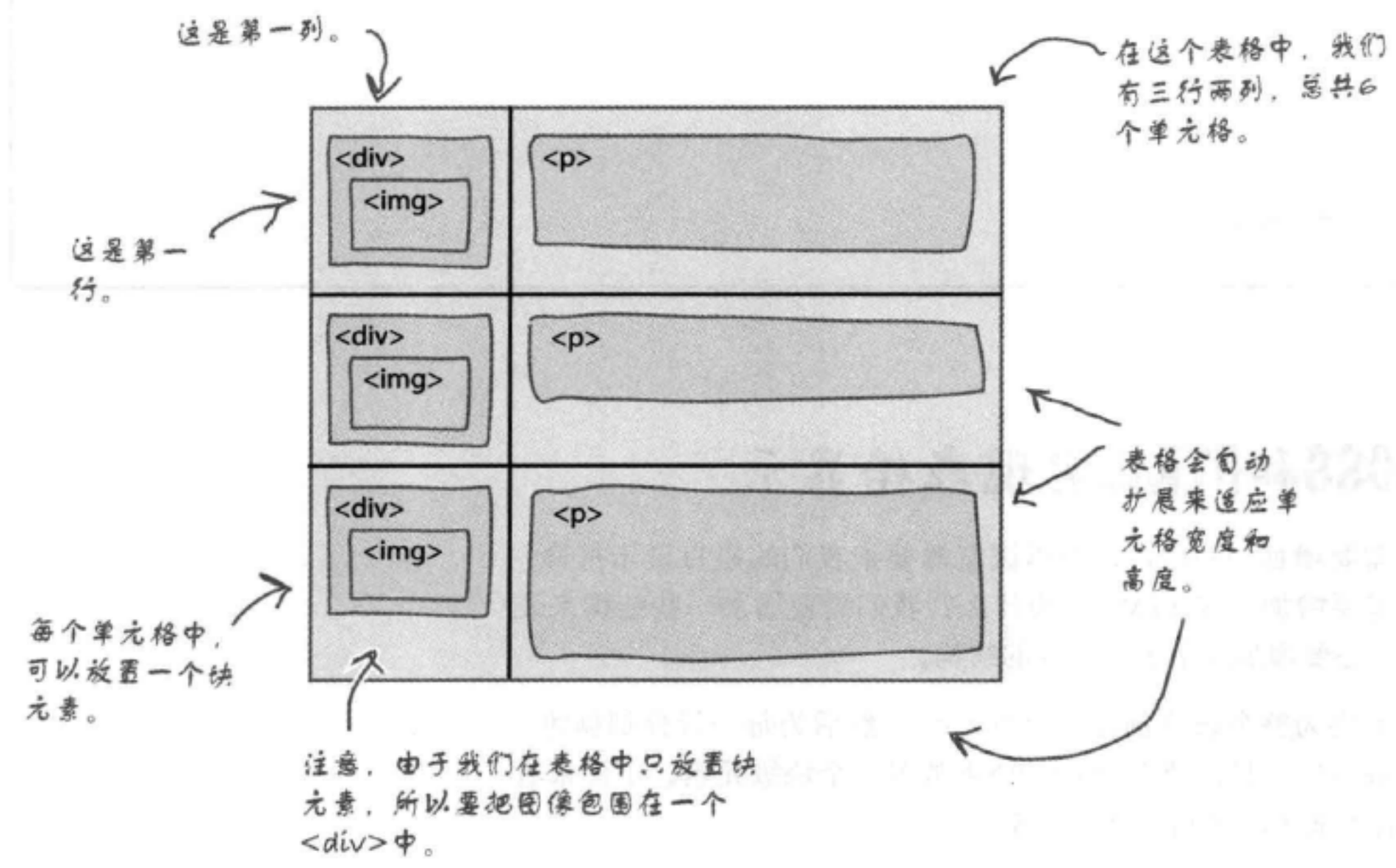
与所有其他布局解决方案一样，甚至表格显示也有自己的优缺点。

CSS表格显示如何工作

可以把表格想成是一个电子表格，包含行和列，各行和列交叉的位置有一个单元格。在一个电子表格中，可以在各个单元格中放入值，如一个数或一些文本。对于CSS表格显示，每个单元格会包含一个HTML块元素。



假设你的页面有3个图像，还有3个段落，你希望把它们放在三行两列中。理论上讲，可以使用表格这样做：



既然你已经了解了CSS表格显示，下面画出草图，指出如何把Starbuzz页面的两栏（“main”和“sidebar”）放在一个表格中。学习后面的内容之前，先对照这一章最后给出的答案检查你的结果……

在这里画出你的表格。

如何创建CSS和HTML实现表格显示

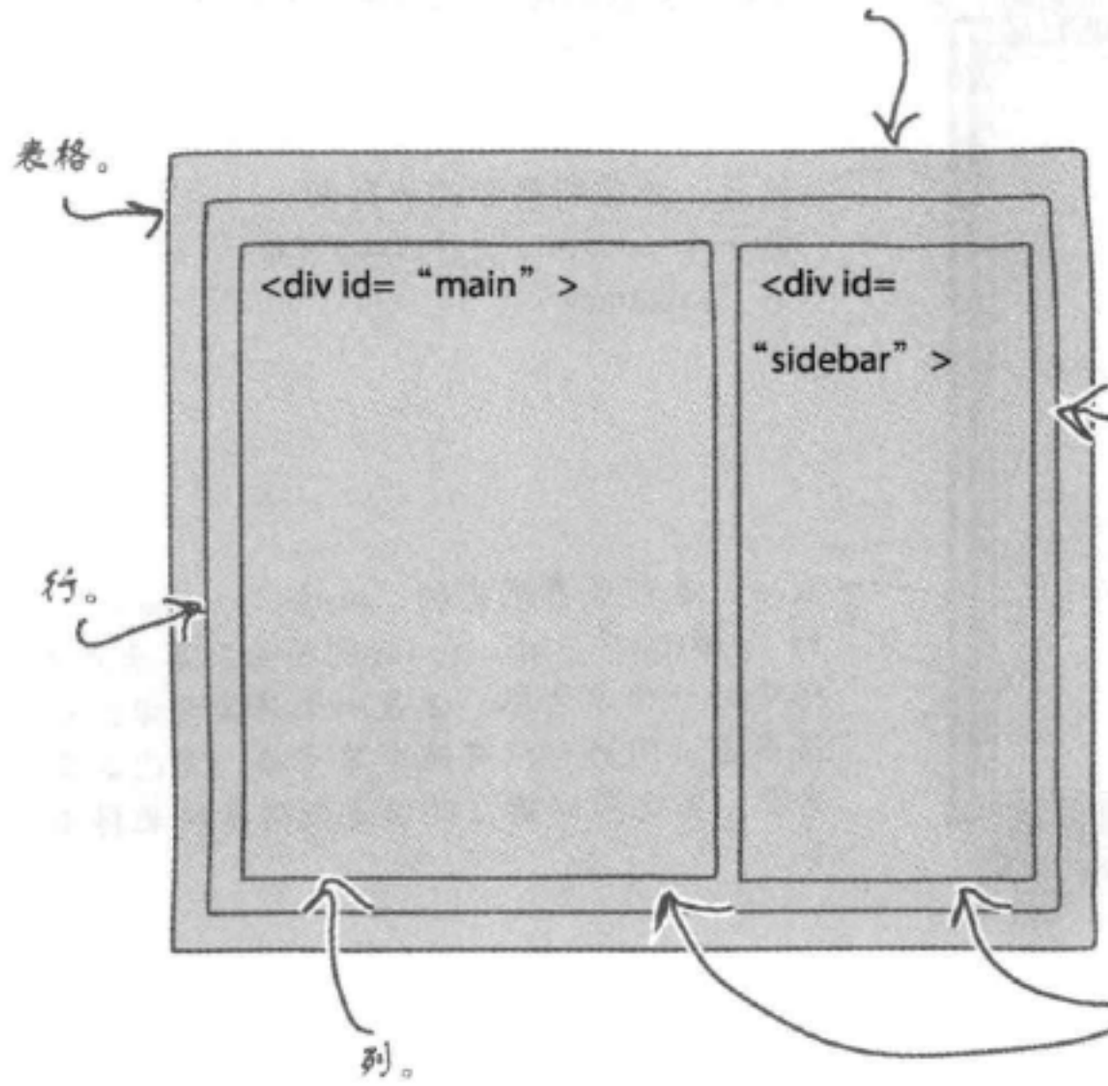
不难想到，我们需要增加一些CSS，告诉浏览器要把我们的栏目显示得像一个表格，不过还需要增加一些HTML。为什么？我们需要增加一些结构来表示表格的行和列，还要增加这个表格本身的结构。

做法很简单，只需要为整个表格创建一个 ，然后为每一行分别创建一个 。对于每一列，只需要在行 中放置一个块级元素。下面来看这个HTML，然后再来考虑我们需要的CSS。

为表格显示增加HTML结构

下面一步一步介绍如何使用HTML增加结构，来支持CSS表格显示：

1 首先，创建一个 表示整个表格，行和列要嵌套在这个 中。



2 接下来，对于表格中的每一行，要创建一个 ，其中包含行内容。Starbuzz页面中只有一行。

3 然后，对于每一列，只需要一个块元素作为该列内容。我们已经有两个块元素可以使用：“main” 和“sidebar” 。

Sharpen your pencil

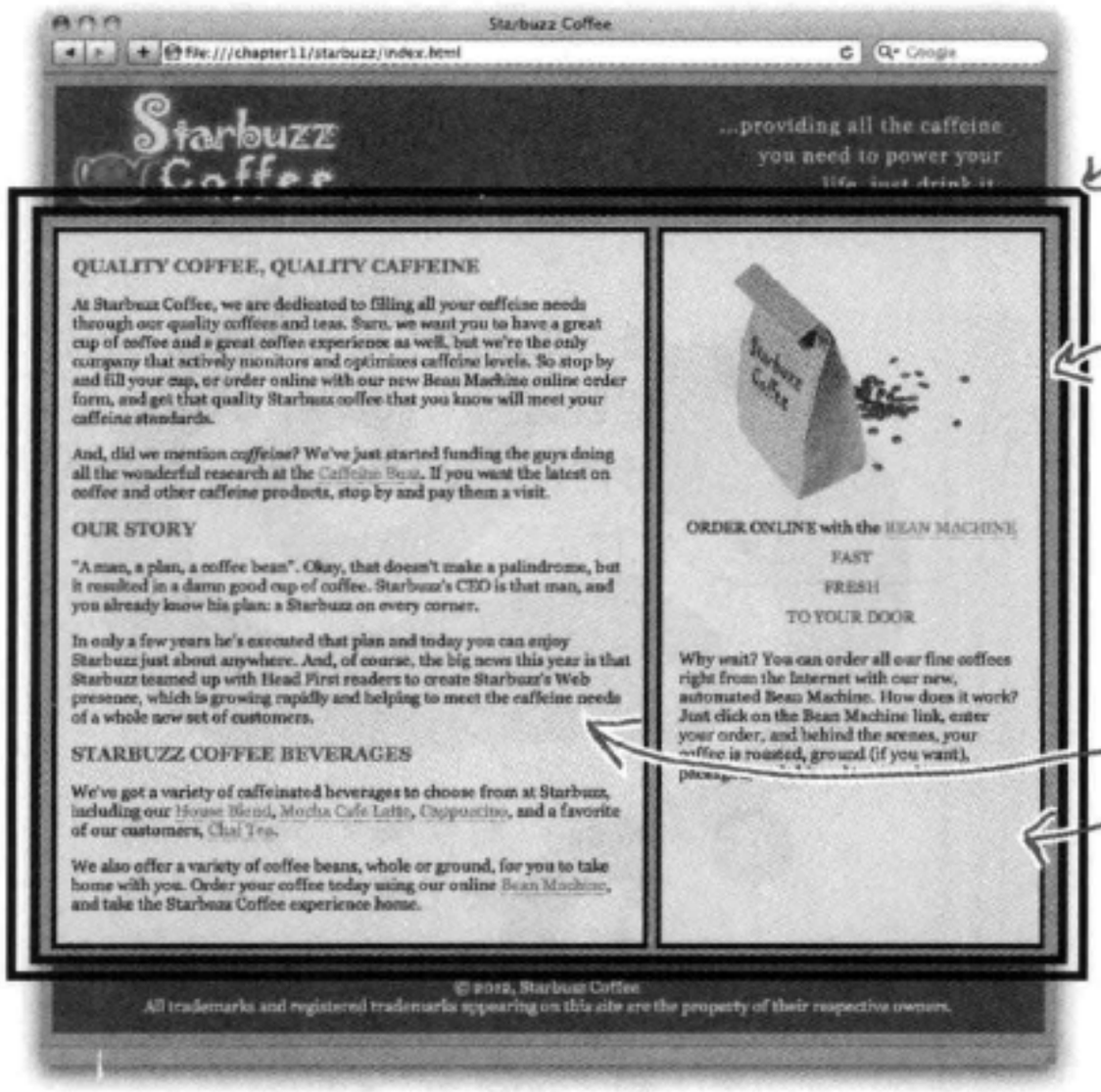
现在该轮到你了：在下面写出建立Starbuzz表格结构所需的HTML。

在这里写出实现Starbuzz表格显示布局所需的HTML。

Sharpen your pencil Solution

现在该轮到你了：在下面写出建立Starbuzz表格结构所需的HTML。

这是我们的答案！



首先，把要作为一个表格显示的所有内容包围在一个 中，名为"tableContainer"。

然后，为我们需要的一行创建一个 ，这个 名为"tableRow"。

最后，各列放置现有的"main" 和"sidebar" ，它们将分别显示为表格中的一个单元格。这是一个非常简单的表格布局，因为它只有两个单元格，不过如果需要，完全可以建立比这复杂得多的表格布局。

下面写出HTML……

这些HTML没有给出，不过可以知道这里是页眉……

```
...
<div id="tableContainer">
  <div id="tableRow">
    <div id="main">
      ...
    </div>
    <div id="sidebar">
      ...
    </div>
  </div>
</div>
...
```

增加新的 （id为"tableContainer"），包围"main"和"sidebar" 。

然后增加新 （id为"tableRow"），也要包围"main"和"sidebar" ，不过要嵌套在"tableContainer" 中。

确保正确地嵌套结束 标记！

……这里是页脚。要确保页眉和页脚不包含在新 中。

如果使用CSS创建表格显示

既然知道了如何增加HTML结构来支持CSS表格显示，下面来看如何为各个元素指定CSS，创建这个表格显示。

❶ **首先，为表格增加一个 ，id为“tableContainer”。这个 包含行和列。如下指定“tableContainer” 的样式：**

```
div#tableContainer {
  display: table;
}
```

“tableContainer”是最外层 ，表示整个表格结构。

❷ **接下来，为行增加一个 ，id为“tableRow”。我们只有一行，其中有两个单元格，所以只需要一个行 。如果有多行，则需要多个行 。这个行 的样式如下指定：**

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

“tableRow” 表示表格中的一行。我们的表格中只有一行，所以只需要这一个规则。如果你有多行，可以考虑使用一个类（例如div.tableRow），这样就可以用一个规则指定所有行的样式。

❸ **最后，使用现有的“main”和“sidebar” 作为单元格，对应于行中的各列。这些 的样式指定如下：**

```
#main {
  display: table-cell;
  background: #efe5d0 url(images/background.gif) top left;
  font-size: 105%;
  padding: 15px;
  margin: 0px 10px 10px 10px;
}
#sidebar {
  display: table-cell;
  background: #efe5d0 url(images/background.gif) bottom right;
  font-size: 105%;
  padding: 15px;
  margin: 0px 10px 10px 10px;
}
```

“main”和“sidebar” 是表格中的列，所以它们分别作为表格单元格显示。

再回到Starbuzz……

现在为Starbuzz页面加入表格显示，看看各栏看起来怎么样。为此，我们还要回到这一章开始时创建的Starbuzz HTML和CSS，所以打开“chapter11/tabledisplay”得到HTML和CSS的最新副本。编辑“index.html”，增加两个 来包围“main” 和“sidebar” ，外面的 名为“tableContainer”，里面的名为“tableRow”。接下来，打开“starbuzz.css”文件，把以下规则增加到CSS中：

如果需要，可以参考前几页的HTML。

```
#tableContainer {
  display: table;
  border-spacing: 10px;
}
#tableRow {
  display: table-row;
}
#main {
  display: table-cell;
  background: #efe5d0 url(images/background.gif) top left;
  font-size: 105%;
  padding: 15px;
  margin: 0px 10px 10px 10px;
  vertical-align: top;
}
#sidebar {
  display: table-cell;
  background: #efe5d0 url(images/background.gif) bottom right;
  font-size: 105%;
  padding: 15px;
  margin: 0px 10px 10px 10px;
  vertical-align: top;
}
```

display: table属性告诉“tableContainer” 要像表格一样摆放。

border-spacing属性为表格中的单元格增加10像素的边框间距。可以把border-spacing看作是常规元素的外边距。因为我们要使用单元格上的border-spacing，所以不再需要 上的外边距（见下面）。

“tableRow” 是表格中的第一行（也是唯一的一行）。

“main” 和“sidebar” 都是表格中的单元格。“main”是“tableRow”的第一列（因为它在HTML中最先出现），“sidebar”在第2列。

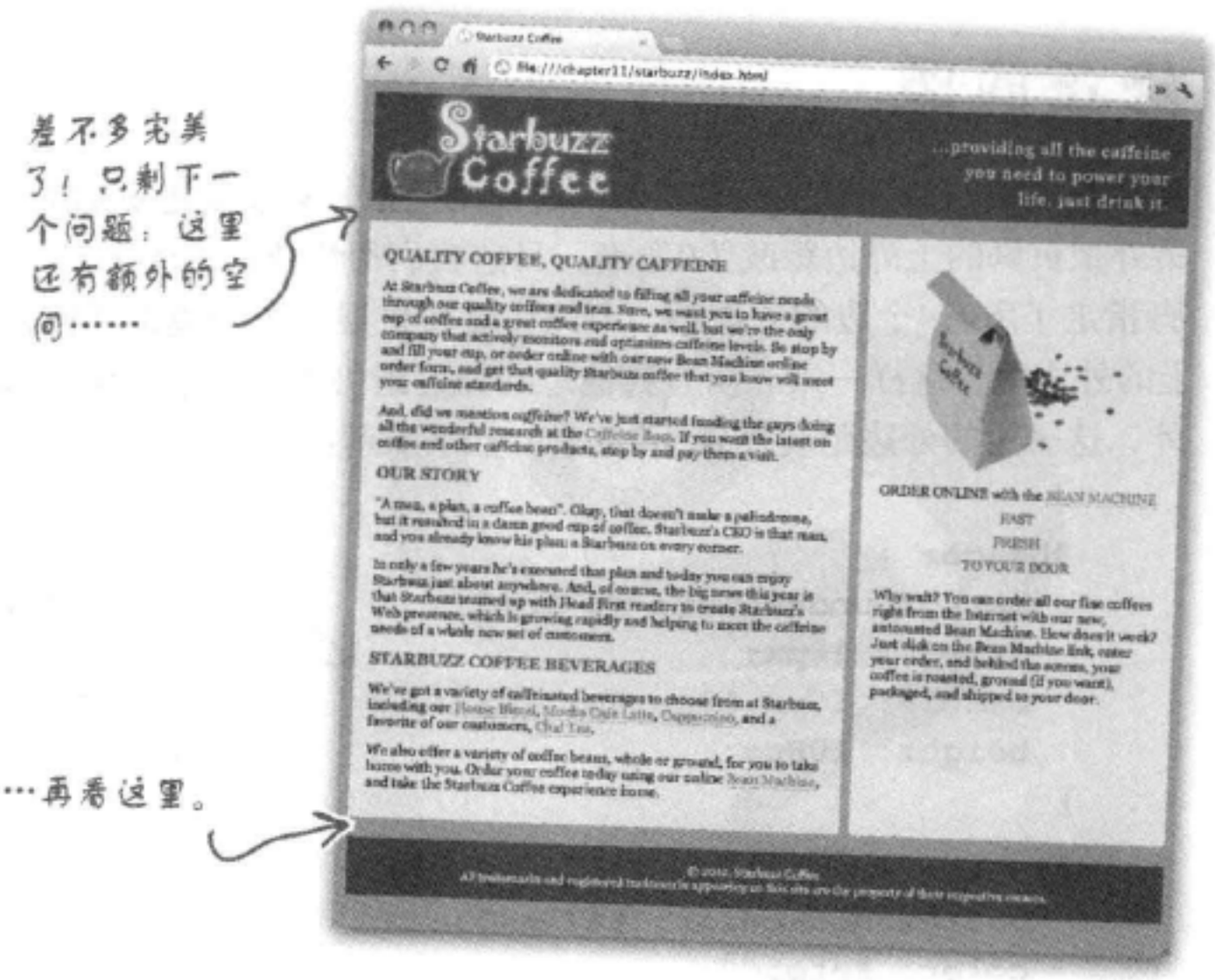
可以删除“main”和“sidebar”的外边距。

我们需要增加一个属性vertical-align，确保表格两个单元格中的所有内容相对于单元格上边对齐（而不是与中间或下边对齐）。

快速测试……

太棒了！现在这两列看起来（几乎）完美了。试着让浏览器变宽，然后再变窄。注意两列的高度总是一样的，而且我们不会再遇到某一列与页脚重叠的问题。另外，对于移动用户，内容也会以正确的顺序显示！

这里只有一个小小的问题，不过很容易修正，注意到页眉和这两列之间的空隙了吧？另外页脚与这两列之间的间距好像也有点太大了……

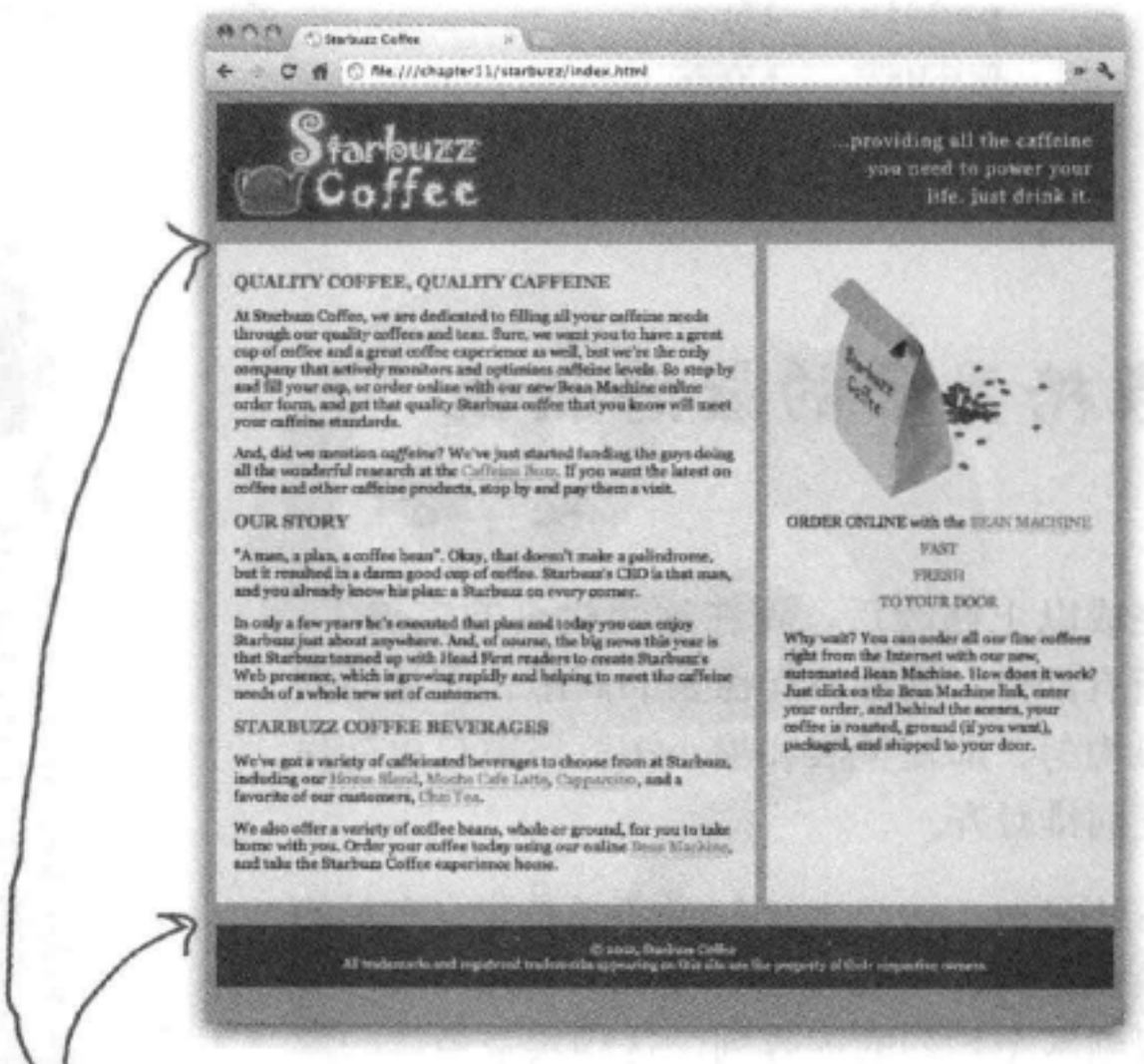


这些间距是怎么回事？

现在我们的“header” 有10像素的下外边距，另外“footer” 有10像素的上外边距。增加表格布局之前，我们指定了“main”和“sidebar” 的外边距，使它们的上外边距都为0像素，所以它们与“header”之间总的外边距为10像素，另外还有10像素的下外边距。现在应该记得，相互挨着的块元素的垂直外边距会折叠。这说明，尽管这两列有10像素的下外边距，而且页脚有10像素的外边距，但这些外边距会折叠为10像素，所以两栏与页脚之间总的空间也是10像素。

从“main”和“sidebar” 删除外边距时，我们在“tableContainer”　　中使用border-spacing属性创建了一个10像素的间距。这会在单元格之间增加10像素的空间，另外在边界周围也会增加10像素的空间。

不过border-spacing和外边距创建的空间不会折叠！所以最后页眉和两列之间就有20像素的空间，另外两列与页脚之间也有20像素的空间。幸运的是，修正这个问题相当容易。



修正间距

要修正页眉与两列以及页脚与两列之间的间距，只需要把页眉的下外边距改为0像素，另外把页脚的上外边距改为0像素。目前我们用快捷规则margin: 10px为页眉和页脚指定了所有4个边的外边距。所以，需要扩展这个外边距属性，单独地指定各个边上的外边距，这样一来，所有其他边的外边距仍为10像素，但与两列相邻的那一边除外，这一边的外边距要设置为0像素。如下所示：

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

#footer {
  background-color: #675c47;
  color: #efe5d0;
  text-align: center;
  padding: 15px;
  margin: 10px;
  margin:  0px 10px 10px 10px;
  font-size: 90%;
}
```

并不是让页眉所有4个边的外边距都是10像素，现在只让另外3个边的外边距为10像素，而下边除外，要把下外边距设置为0像素。

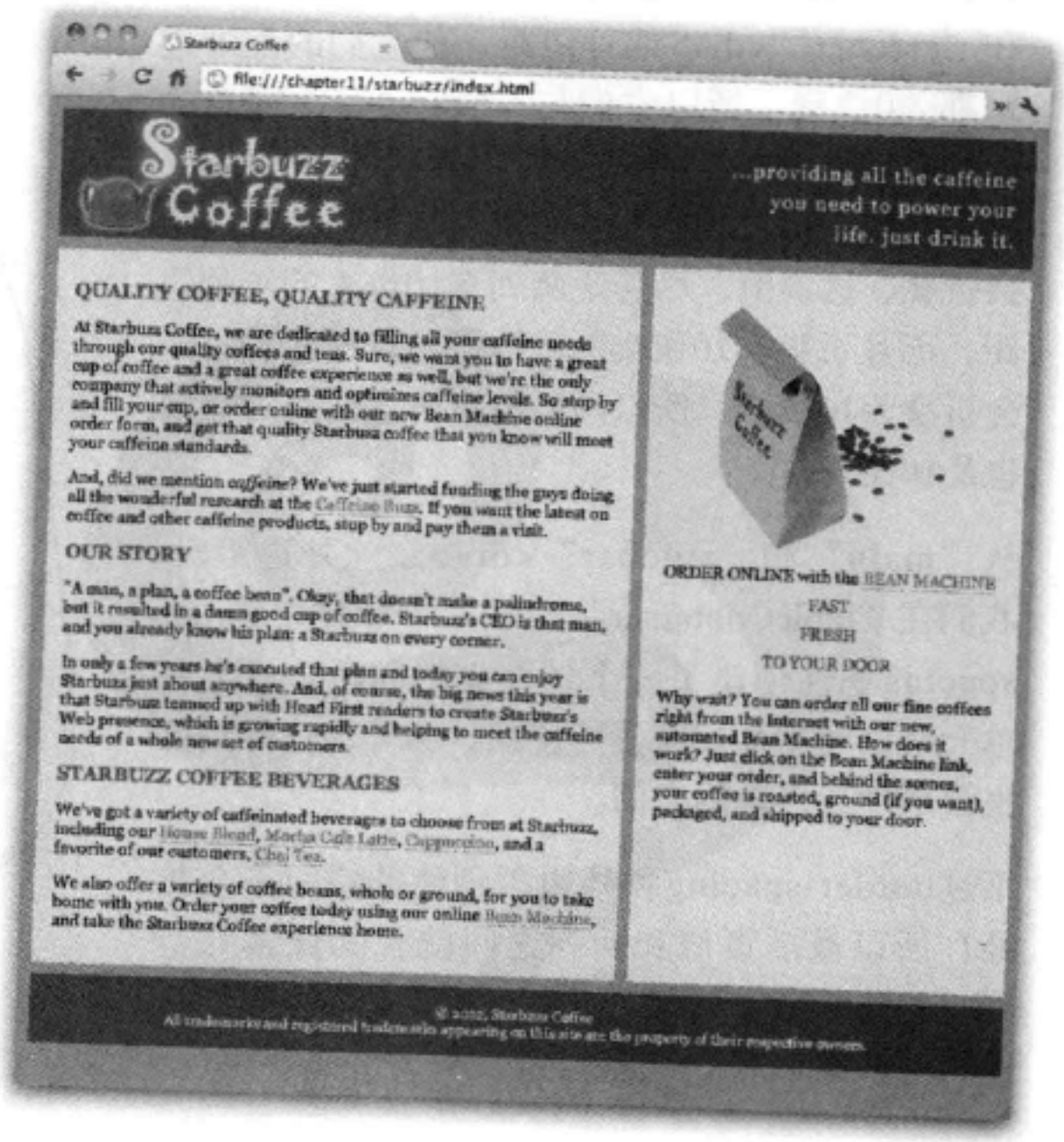
类似的，对于页脚，现在另外3个边的外边距仍为10像素，但上外边距为0像素。

表格显示的最终测试

完成以上修改后，现在这两栏真的完美了！所有部分之间都有10像素的间距，而且两列很均匀，即使扩展和缩小浏览器窗口也能保证同排对齐。

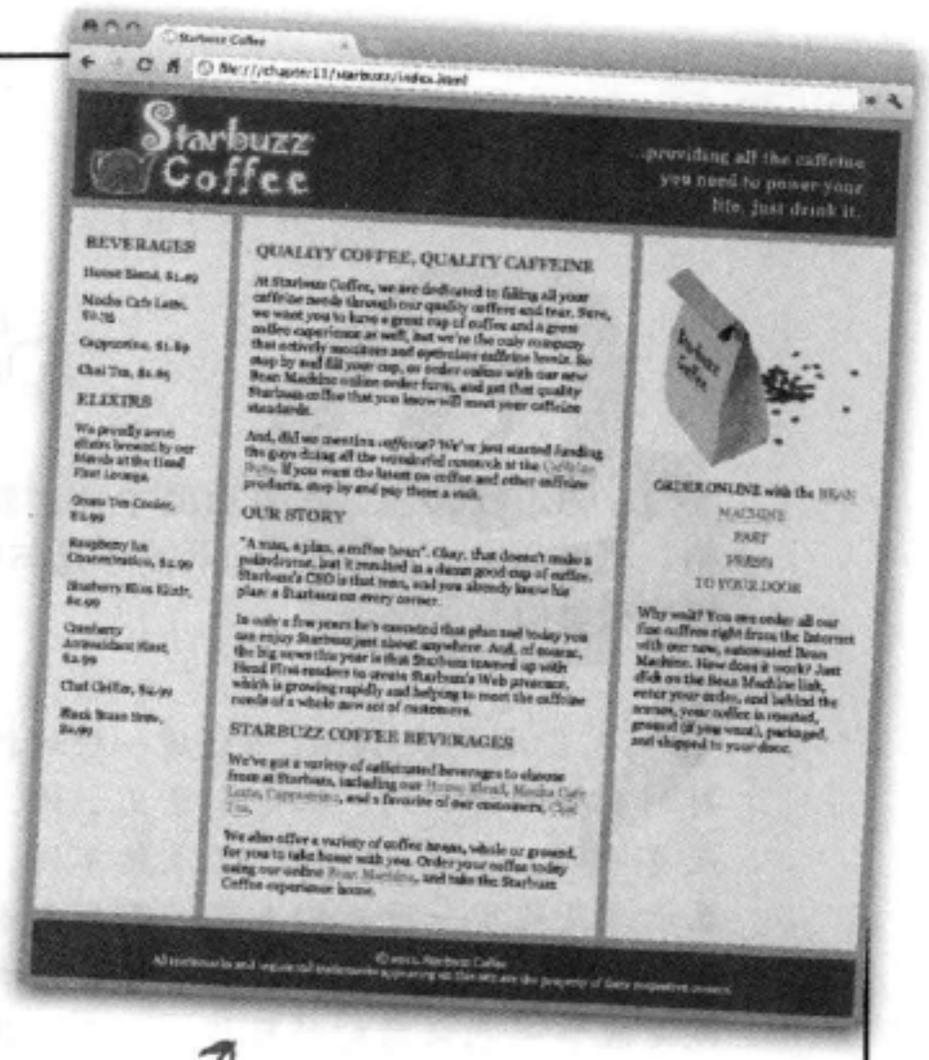
不过display: table并不总是建立布局的最佳工具，但在这里，要在Starbuzz页面中得到两个均匀的内容列，这确实是最佳的解决方案。

太完美了！



Exercise

Starbuzz CEO决定再为Starbuzz Coffee页面增加一列，提供饮料单。他希望新加的这一列放在左边，宽度是浏览器窗口的20%。你的任务是在现有页面正确的位置上增加新的饮料单HTML，然后完成下面的CSS，确保它显示为一个表格单元格，就像另外两列一样。对照这一章最后给出的答案检查你的结果。



CEO希望Starbuzz页面像这样，左边有一个新列，其中包含饮料单。

这个饮料单的HTML。

```
<div id="drinks">
  <h1>BEVERAGES</h1>
    <p>House Blend, $1.49</p>
    <p>Mocha Cafe Latte, $2.35</p>
    <p>Cappuccino, $1.89</p>
    <p>Chai Tea, $1.85</p>
  <h1>ELIXIRS</h1>
    <p>
      We proudly serve elixirs brewed by our friends
      at the Head First Lounge.
    </p>
    <p>Green Tea Cooler, $2.99</p>
    <p>Raspberry Ice Concentration, $2.99</p>
    <p>Blueberry Bliss Elixir, $2.99</p>
    <p>Cranberry Antioxidant Blast, $2.99</p>
    <p>Chai Chiller, $2.99</p>
    <p>Black Brain Brew, $2.99</p>
</div>
```

新的CSS……你需要完成这个CSS！

```
#drinks {
    _____________________________;
    background-color: #efe5d0;
    width: 20%;
    padding: 15px;
    vertical-align: top;
}
```

填空，让drinks 作为页面上的第一列。

there are no Dumb Questions

问：嗯，我知道这本书后面才会讲到HTML表格，不过CSS display: table是不是与使用HTML表格很类似？

答：它们都是在HTML中创建一种结构，能够映射到表格的行和列，从这个意义上讲，它们确实是类似的。不过，与HTML表格不同，CSS表格显示只是使用一种类似表格的布局来表现这个结构中的内容。HMTL表格面向的是表格数据，也就是应当有表格结构的数据。所以，使用CSS表格显示只是创建某种表现布局的一种方法，而HTML表格则是建立数据的结构。HTML表格的内容将在第13章介绍。

问：如果我的表格显示里需要不只一行，该怎么办呢？

答：如果需要在多行中显示内容，只需要增加更多HTML结构来支持多行。如果再看一下Starbuzz HTML，你会看到这里一行有两列（或者，增加了Beverages列之后有3列）。要想再增加一行，可以类似“tableRow” ，再增加一个 ，嵌套在“tableContainer” 中，其中包含的列数要与第一行的列数相同。如果要增加多行，可以像这样增加更多 。

问：为什么要在CSS中使用vertical-align: top为每个单元格增加垂直对齐？

答：我们为每个单元格增加了vertical-align: top，这是为了确保所有内容都与单元格上边对齐。如果每个单元格都按这种方式对齐，那么Starbuzz页面上每列中的内容就会在顶端对齐，看上去会更专业。如果没有增加垂直对齐方式，你会看到，浏览器中的默认对齐方式设置为中间对齐。当然，有些情况下，这可能正是你想要的！垂直对齐可以设置为top（顶端对齐），middle（中间对齐）或bottom（底端对齐）。

问：一个单元格中放多少内容会有影响吗？

答：没什么影响。你可能想确保各个列的内容数量不要过于参差不齐，比如说与其他列相比，某一列的内容过多，使页面看起来很不均衡。不过，总的说来，各列放多少内容要由你以及你希望的页面外观来决定。

问：我能不能控制列的宽度？

答：可以，用width属性就能控制列的宽度。在增加Beverages列的练习中，其实你已经这样做过，你可能注意到我们将这一列的宽度设置为20%。可以像这样设置各个列的宽度（最好确保所有列的宽度加在一起是100%）。通过使用百分数，你的表格仍能随着浏览器窗口大小的调整扩展和收缩。

CSS布局工具箱的策略

你已经看到了，使用HTML和CSS建立页面布局有很多种可以使用的方法。要改变页面的布局，我们并不需要对HTML做太多修改。除了移动某个内容（处理浮动边栏）和增加两个 （实现表格显示布局）之外，完全可以用CSS处理内容的表现。关键在于，你的HTML应当负责为内容建立结构，而CSS负责处理布局。选择哪种方法来建立布局要由你决定，最终要取决于你选择哪种布局，以及你希望这种布局有多大的灵活性。

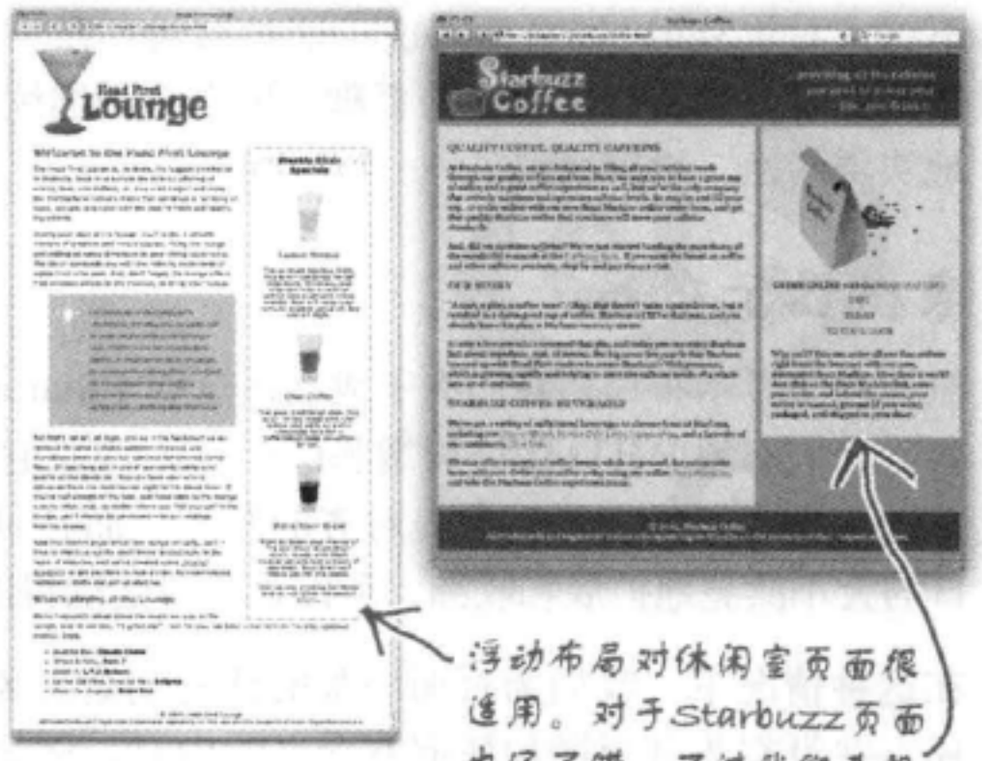
下面复习一下前面的内容。

浮动布局

我们对休闲室页面使用了浮动布局，将elixirs 浮动到页面中主内容的右边。在这种情况下，float很合适，因为我们希望主内容围绕着elixirs ，它确实出色地完成了这种效果。float还有一种用法（不过我们还没有这样用过），它非常适合在一个文本段落中浮动图像，让文本围绕着这个图像。

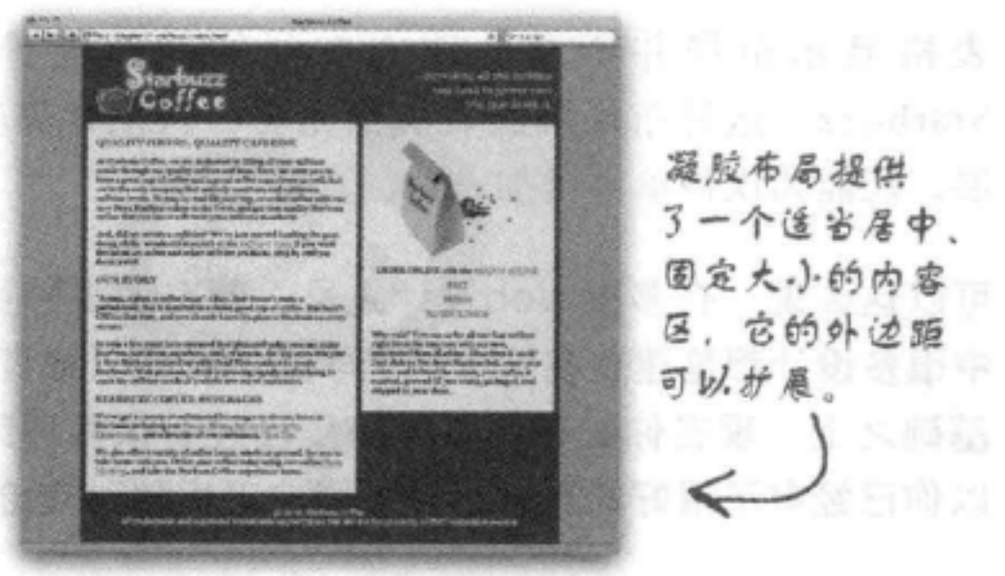
接下来我们使用float使sidebar 浮在Starbuzz页面中，并使用clear确保这个浮动边栏不会与页脚重叠。

这种方法有一个很大的缺点，我们必须把需要浮动的整个 移到页面主内容之上，如果这种顺序并不反映页面中内容的相对重要性，这种做法往往不是最优的。另外还有一个潜在的缺点，使用float时，将无法创建两个高度相同的列，所以如果你想达到这个目标，就需要选择其他方案。



凝胶布局

接下来我们创建了一种冻结布局，由一个固定大小的 包围页面中的所有内容，然后利用auto属性值允许外边距扩展，把它调整为一种凝胶布局。这样可以得到一个很漂亮的布局，Web上很多页面就采用了这种设计。例如，你会看到很多博客都是采用这种方式建立的。这样也解决了内容顺序的问题。这种方法的缺点在于，内容不会扩展来填满整个浏览器窗口（不过很多人可能根本不认为这是一个缺点）。



CSS布局工具箱的策略（续）

绝对布局

然后我们使用绝对定位得到了一个流体布局，这也能保证内容的顺序正是我们希望的。通过将边栏设置为一个特定的宽度，并将它定位在主内容右边，就有了一个可以随页面大小扩展和收缩的主内容区，而边栏会一直保持固定的大小，而且固定在浏览器窗口右侧。如果你希望页面的某一部分大小固定，而另外一部分可以扩展和收缩，这就是一个很好的布局选择，或者如果你需要精确地指定某个元素的位置（稍后会看到如何做到），也很适合选择绝对定位。

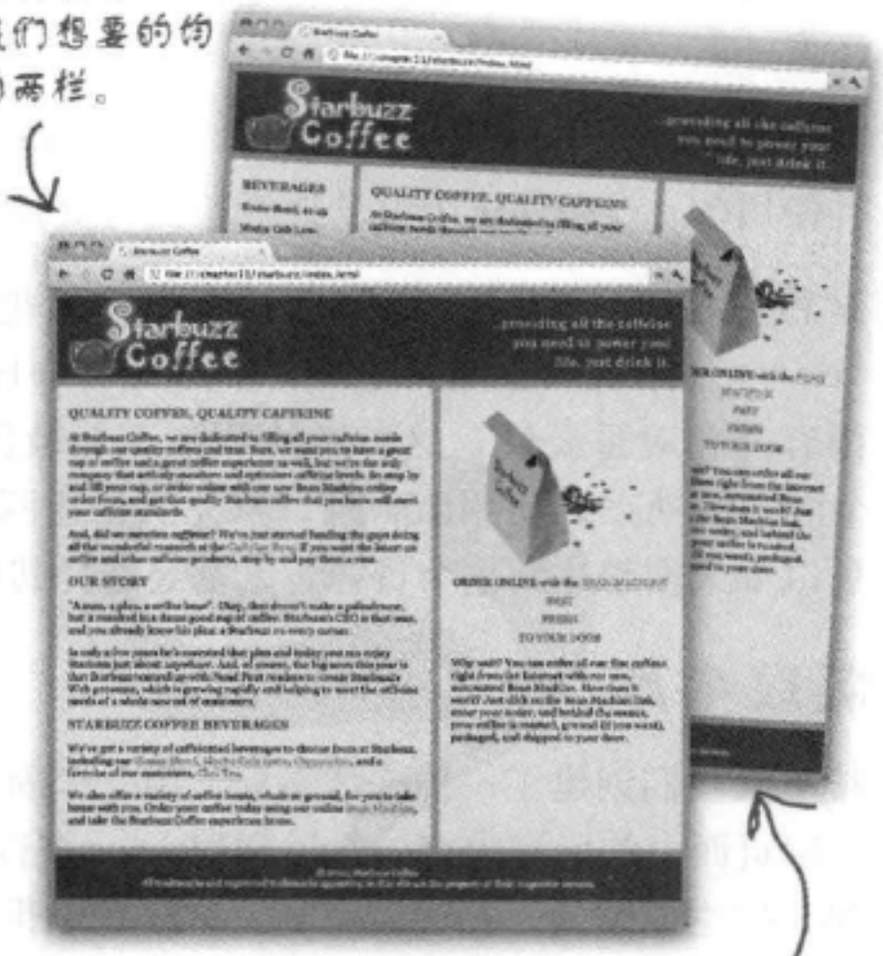
利用绝对定位，可以提供一个漂亮的流体主内容区，还有一个固定的边栏。

不过，对于Starbuzz页面来说，这种布局存在一个缺点：浏览器变宽时，边栏会再次覆盖页脚。所以我们继续探索如何完美地实现两栏，然后找到了……

表格显示布局

利用表格显示布局，我们终于成功地建立了Starbuzz的布局。确实，我们必须向HTML结构增加两个 ，它才能正常工作，不过这样我们就能得到完美对齐的两列，而且可以随浏览器窗口的大小漂亮地扩展和收缩，所以这绝对是值得的。

采用表格显示布局，现在可以得到我们想要的均匀的两栏。



表格显示很容易扩展，可以增加更多列（或更多行）。

在这种情况下，我们为页面增加的结构完全是为了支持这种布局，并没有为页面增加任何有含义的内容。你会发现，通常都会这样使用 （实际上，读到下一章时，你会发现如今更是这样，而几年前都没有这么普遍）。不过，不要滥用 ，你希望根据需要选择最好的布局，要得到你想要的布局应当尽可能少增加 。

表格显示布局并不总是建立布局的最佳选择，不过对于Starbuzz，这种布局确实表现很完美，甚至允许我们轻松地扩展，还能为饮料单再增加一列。真是很棒！

可以这么说，有多少Web设计人员，就有多少种页面设计，不过其中很多设计都是建立在你在这里学到的布局（或者它们的一些变种）基础之上。现在你的布局工具箱里已经有很多种策略可供选择，所以你已经有了很好的基础，可以解决老板随时交给你的布局任务！

对，你说的没错。

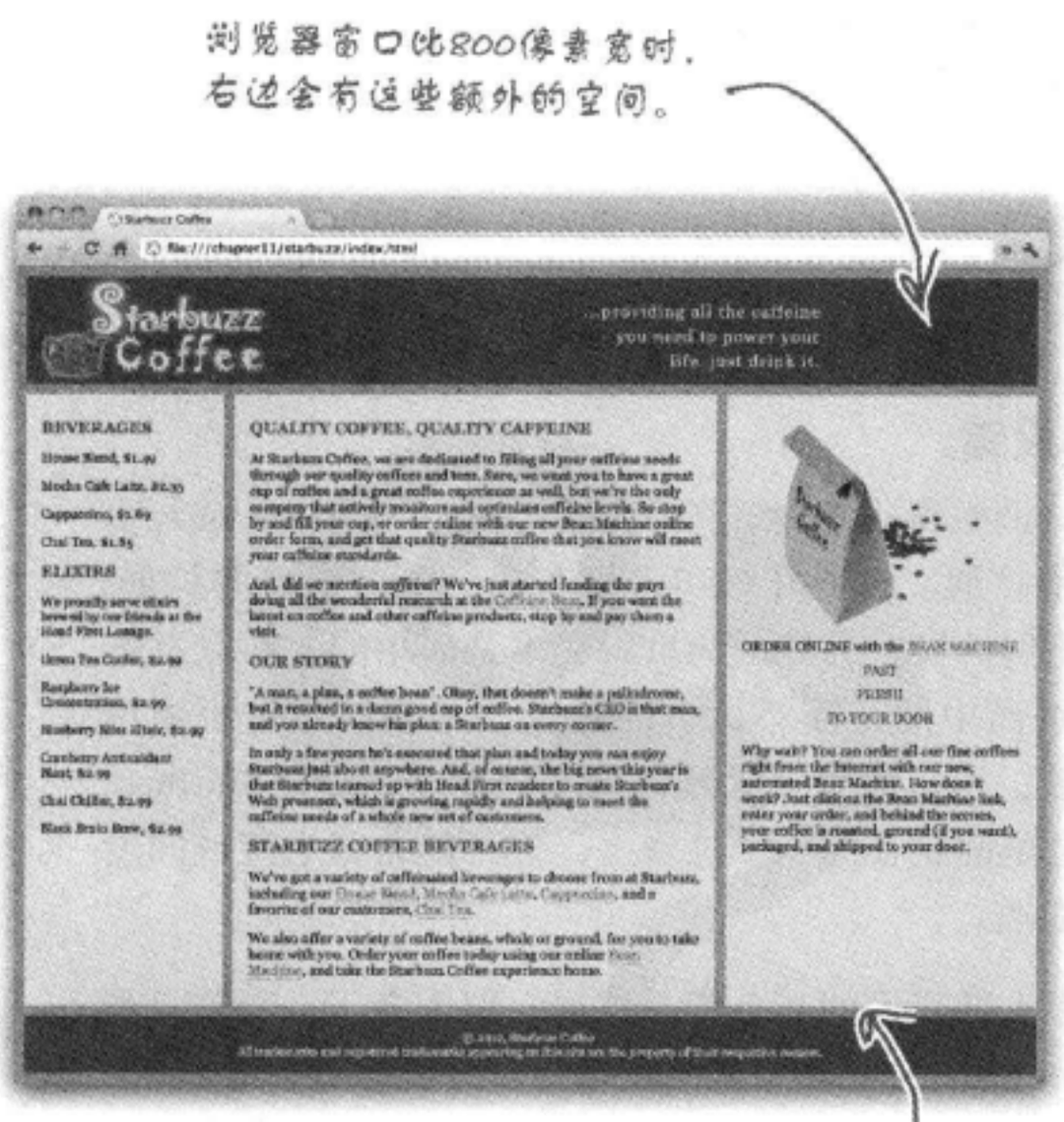
将浏览器窗口调整得很宽时，除了页眉以外，这个Starbuzz页面都能很好地扩展。归功于CSS表格布局，随着窗口的扩展，各列也会成比例扩展。另外因为页脚文本是居中的，所以页脚看起来总在页面中间，而不论页面是宽还是窄。不过页眉没能很好地扩展。可以看到背景颜色确实扩展了，但是Starbuzz口号看起来总是固定在同一个位置上，你可能希望它应该向后移到窗口的右边。

页眉之所以没有随页面的其余部分扩展，原因在于，页眉只是一个包含了Starbuzz logo和口号的图像，这个图像正好是800像素宽。如果浏览器窗口宽度大于800像素，你会看到右边出现额外的空间。类似地，如果浏览器窗口比800像素窄，就会看到图像紧贴着浏览器窗口的边缘。

我们能修正这个问题吗？

页眉的问题

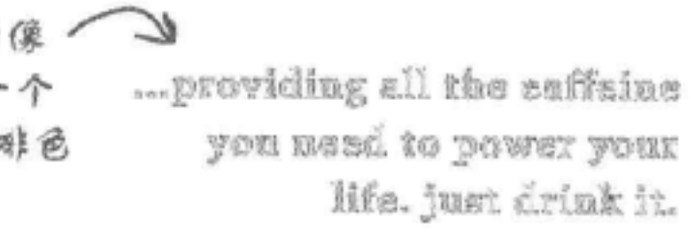
下面来看看页面的变化。打开你的浏览器窗口，先让窗口比页眉图像宽，然后缩小窗口，让它比页眉图像窄。你会看到页眉的表现还不尽如人意。



BRAIN POWER

如果把页眉图像分成两个不同的图像，一个包含logo，另一个包含口号，请你考虑如何在 元素中摆放这两个图像，让它们都有正确的位置（也就是说，不论浏览器窗口有多宽或多窄，logo一直在页眉左边，而口号一直在页眉的右边），有什么办法吗？

可以很容易地将页眉分为两个gif图像（它们都有透明的背景，另外还有一个套版，可以很好地结合页眉中的咖啡色背景颜色）。

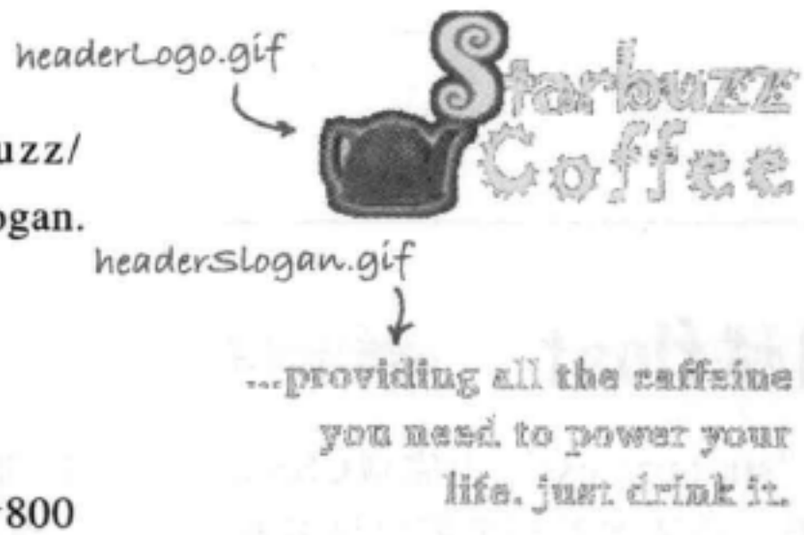


用float修正页眉图像

用CSS解决一个布局问题通常会有多种策略，这里也不例外。我们要用float解决这个问题。在最后采用CSS表格布局之前，我们已经用过float来摆放Starbuzz页面的某些部分。不过你完全可以混合使用多种不同的策略，如在一个页面上同时使用表格显示和float。实际上，这种方式非常常见。下面来看如何做到。

❶ 将页眉图像分为两个图像

这一步我们已经帮你做好了。你会在“chapter11/starbuzz/images”文件夹中找到图像“headerLogo.gif”和“headerSlogan.gif”。



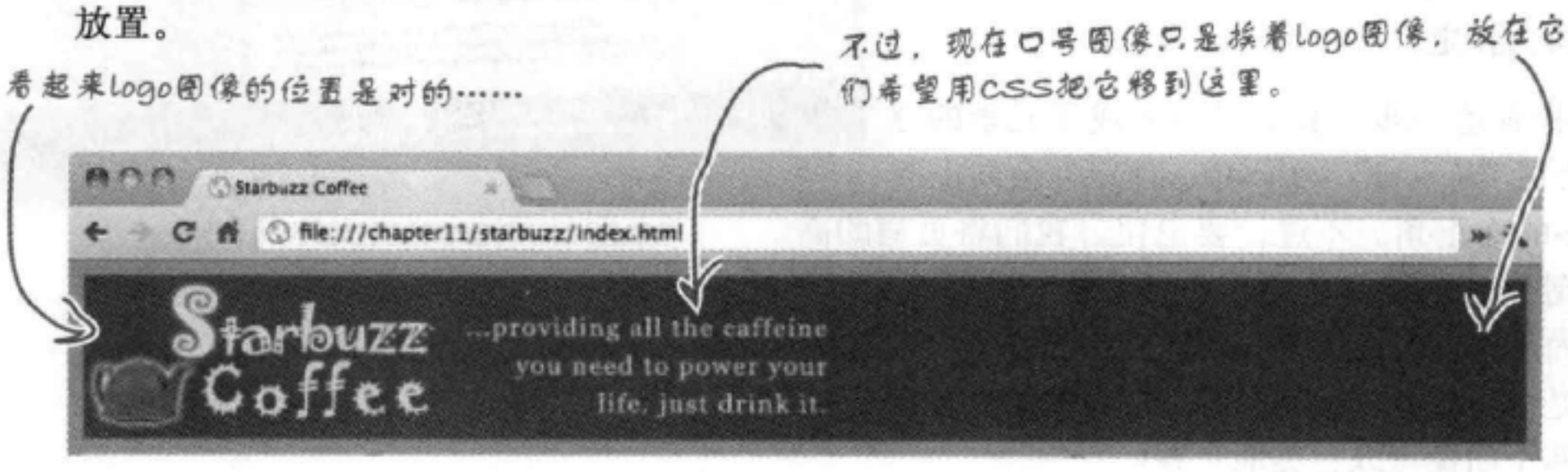
❷ 更新HTML来使用这些图像

接下来，需要更新HTML，替换现在的页眉图像，这是一个800像素宽的图像，把它换成第1步中创建的两个图像。我们要为每个图像分别指定一个id，以便在CSS中选择这些图像。

```
<div id="header">
  <img src="images/header.gif" alt="Starbuzz Coffee header image">
  <img id="headerLogo"
       src="images/headerLogo.gif" alt="Starbuzz Coffee logo image">
  <img id="headerSlogan"
       src="images/headerSlogan.gif"
       alt="Providing all the caffeine you need to power your life.">
</div>
```

❸ 用CSS修正图像

最后，需要把这些图像正确地摆放在页眉中。如果现在加载页面，你会看到页眉中会出现这两个图像，它们相互挨着，靠页面左边放置。



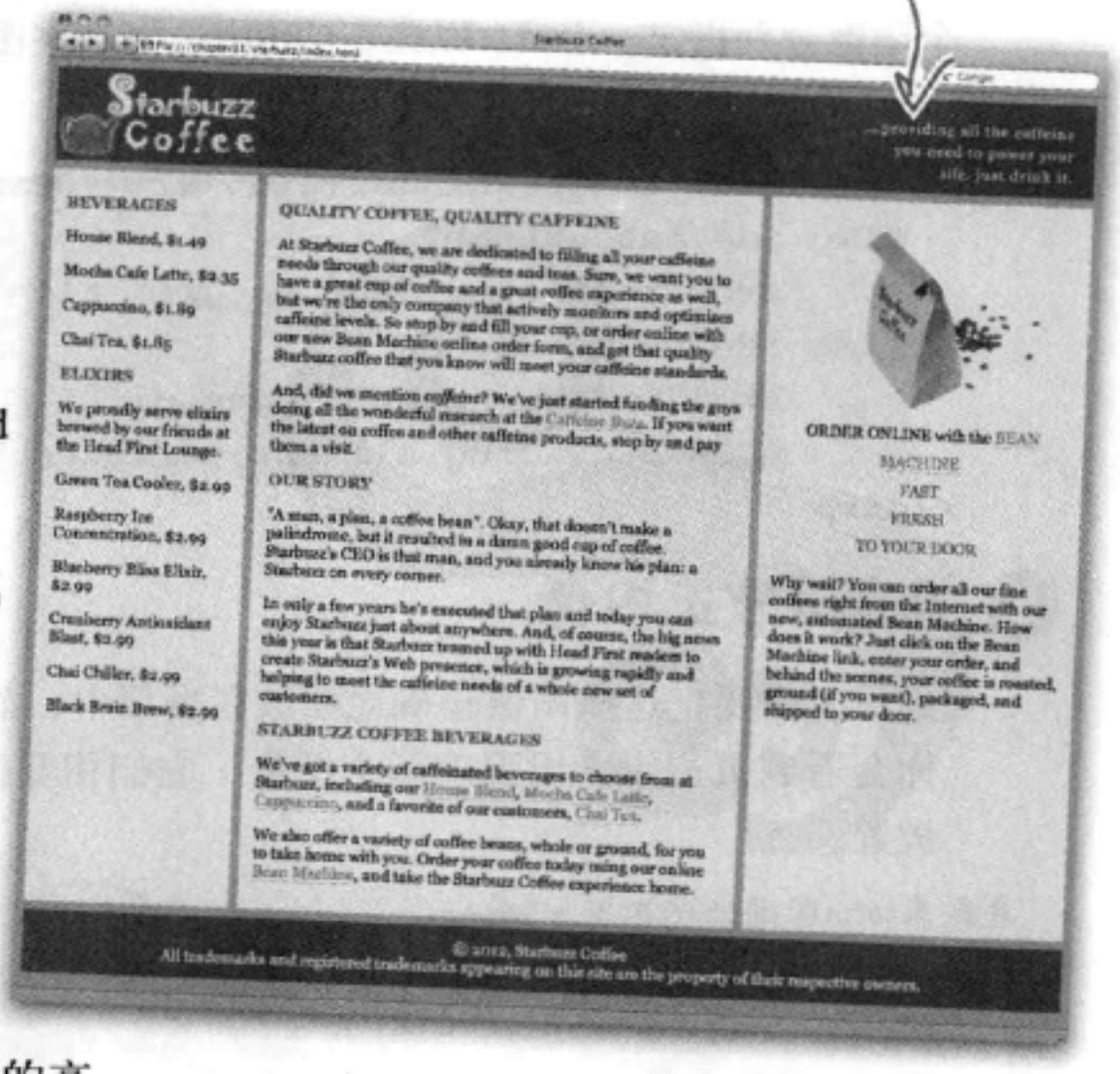
这个CSS很容易，你可能闭上眼睛都能写出来，毕竟这一章你已经有了很多布局经验。编写CSS来修正页眉中的图像。你已经知道要用float，完成下面的填空，补全规则的其余部分，将图像放在正确的位置上。学习后面的内容之前，先对照这一章最后给出的答案检查你的结果。

```
_____________________ {
    float: _________;
}
```

测试float

在“starbuzz.css”中完成CSS更新，重新加载Starbuzz页面。应该能看到页眉口号图像会一直移到页面的右侧，这才是它本来的位置，而且更棒的是，即使浏览器窗口变得相当大，它也会一直留在右边。成功了！

现在口号图像会移到右边，而且一直留在那里，即使你改变了浏览器窗口大小，它也坚守在右边。



页眉中float是如何工作的

还记得这一章前面介绍过浮动一个元素的步骤吧：

为元素指定一个标识。我们为希望浮动的图像指定了 id “headerSlogan”。请检查这一步。

为元素指定一个宽度。这一次实际上我们没有必要显式地指定宽度（不过当然也可以显式指定）。为什么呢？因为image元素默认的有一个指定宽度，这就是图像本身的宽度。CSS可以识别出图像有一个宽度，所以我们没有必要自己再去指定。

浮动元素。请检查这一步。我们已经完成了元素的浮动。<img>嵌套在“header” 中，所以它会向上浮动到这个 的右上角。不过，要记住，我们将页眉的高度设置为恰好等于这两个图像的高度。另外，前面已经解释过，页面中的其他内联内容会围绕着浮动元素。在这里，页眉中的其他内联内容就是logo图像，它正好与与口号图像以及页眉的高度相同。所以这两个图像可以完美地对齐！

there are no Dumb Questions

问：为什么我们不用为页眉下面的“tableContainer” 增加“clear: right”？

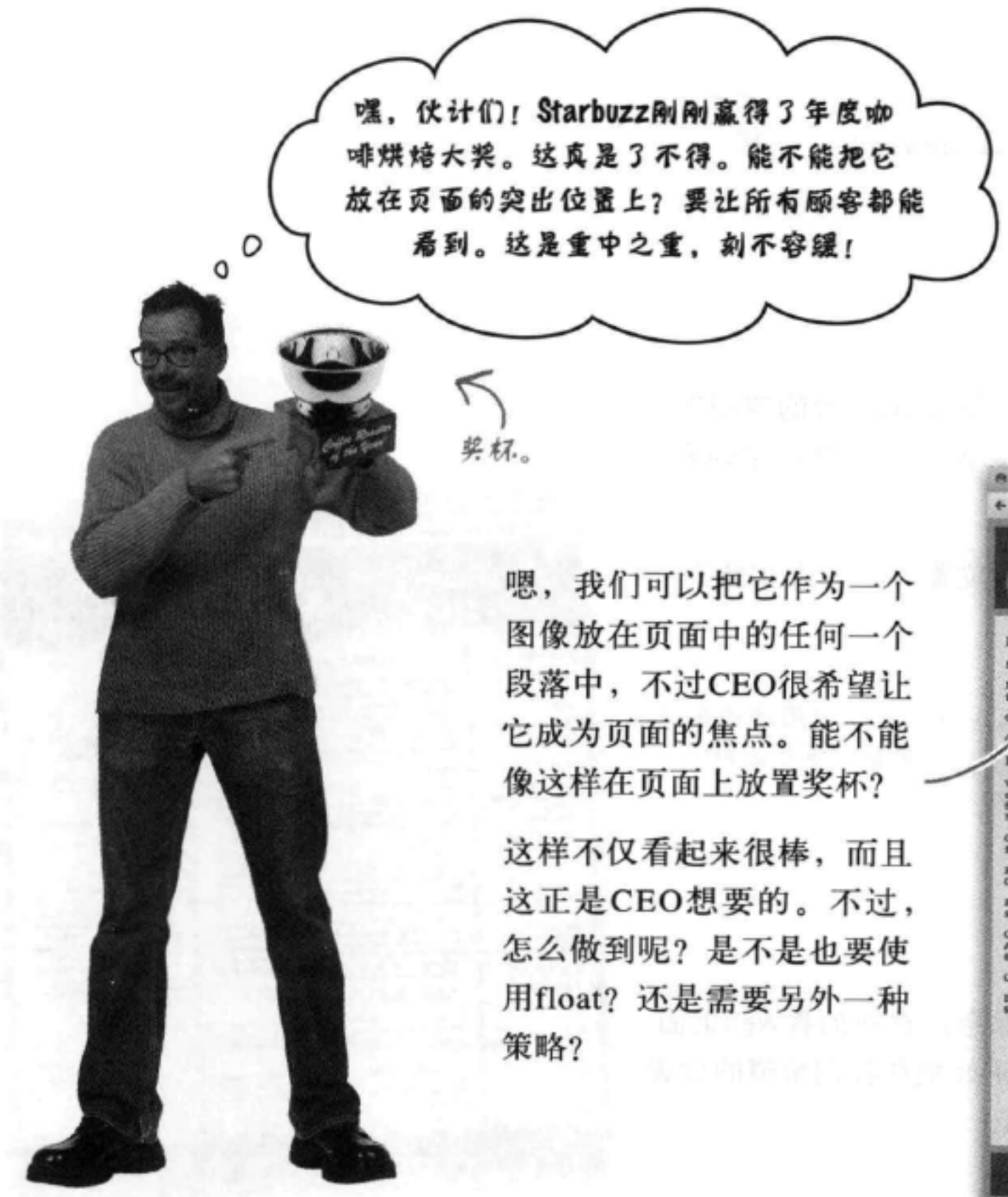
答：因为我们要浮动的图像恰好与页眉中的另一个图像高度相同，都是108像素，所以对于页面中的其他内容来说，再没有空间可以上移和围绕这个浮动元素。两个图像所占的垂直空间相同，所以页面中的其他元素都能稳定在它们自己的位置上。

问：如果我想浮动一个位于文本段落中间的图像，可以吗？

答：如果是这样，文本会围绕着这个图像。这与我们在休闲室页面中浮动elixirs 时是一样的。还记得页面其余部分中的文本是怎么围绕 的吗？浮动图像时，情况也是一样的。

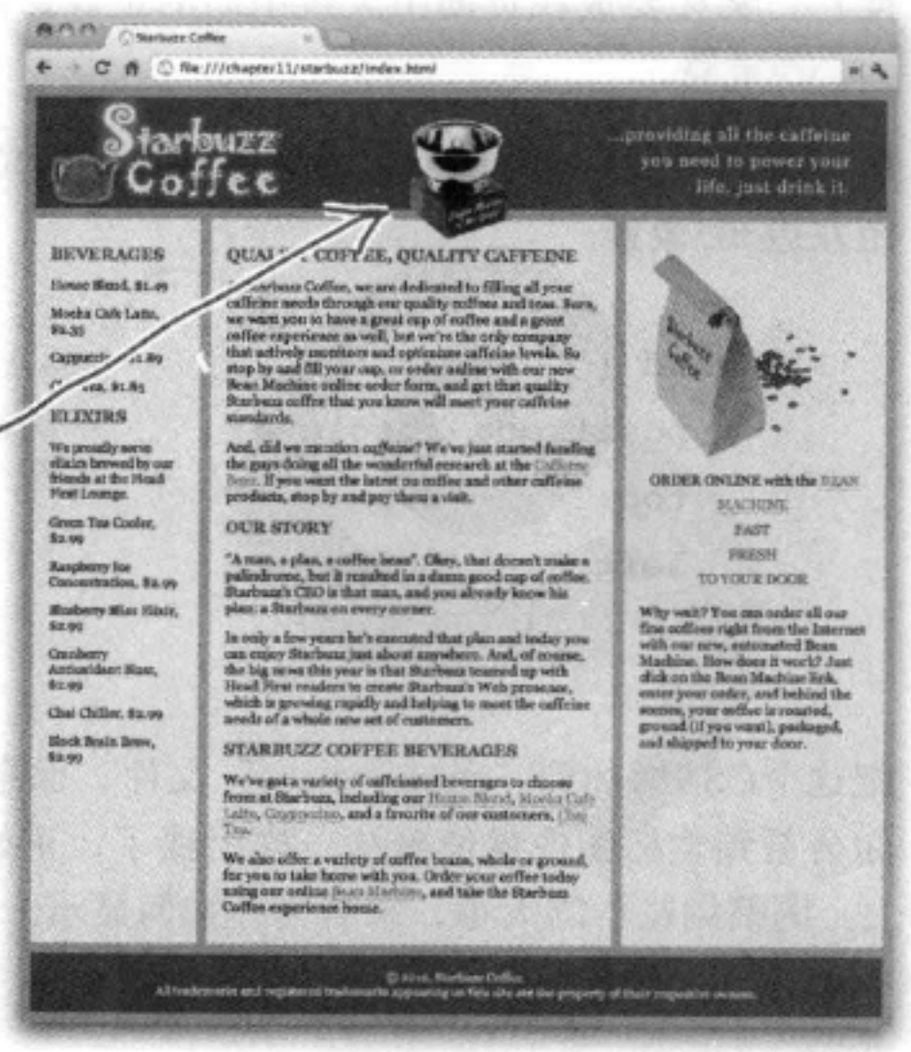
问：能不能使用前面讨论的另外一种布局策略来定位页眉图像？

答：当然可以。CSS中解决一个问题的办法通常有很多。另外一种可用的策略是使用绝对定位。下面我们就来看如何对一个图像绝对定位。



嗯，我们可以把它作为一个图像放在页面中的任何一个段落中，不过CEO很希望让它成为页面的焦点。能不能像这样在页面上放置奖杯？

这样不仅看起来很棒，而且这正是CEO想要的。不过，怎么做到呢？是不是也要使用float？还是需要另外一种策略？



增加奖杯

注意，这个奖杯的位置很特殊，它与页眉和页面主要部分都有重叠。要把一个浮动图像放在这个位置上会相当困难。不仅如此，我们还知道，这个奖杯不应该影响页面中所有其他元素的流。

听起来这正是绝对定位最擅长的事情。通过使用绝对定位，你可以把元素放在页面上你希望的任何位置，而且由于它不在流中，所以不会影响到页面上的任何其他元素。看来对页面稍稍增加些内容就可以了，而不会破坏原有的一切。

下面来试试看，首先增加一个新的 ，把它放在页眉下面（CEO认为这非常重要，所以它在内容顺序中应该最优先）。下面给出这个 ：

```
<div id="award">
    <img src="images/award.gif"
         alt="Roaster of the Year award">
</div>
```

这个 包含奖杯图像。

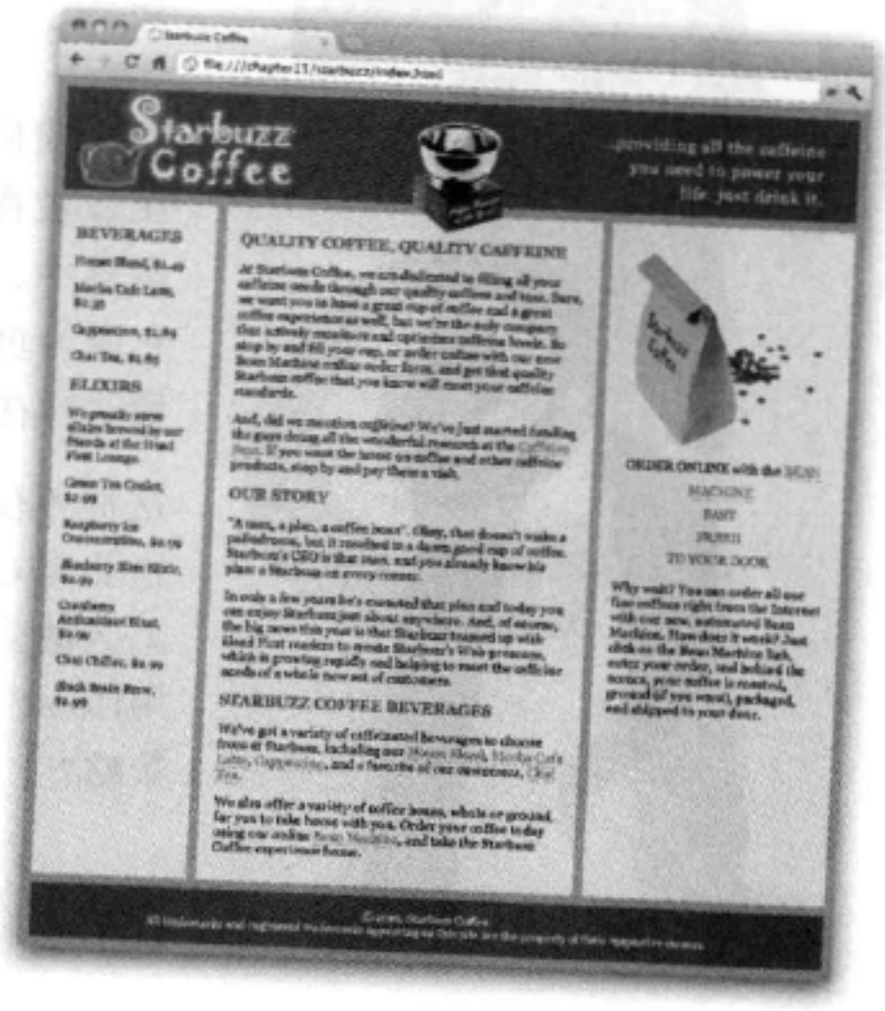
指定奖杯位置

我们希望，浏览器窗口宽度为800像素时（这是浏览器的典型宽度），奖杯要放在页面中大致中间的位置，而且要刚好与主内容 重叠。

所以我们使用top和left属性指定奖杯的位置，距上边30像素，距左边365像素。

```
#award {
    position: absolute;
    top:      30px;
    left:     365px;
}
```

对award 使用绝对定位，距上边30像素，距左边365像素。



把这个CSS增加到“starbuzz.css”文件，保存，重新加载Web页面。你会看到奖杯图像就像魔法一样出现了，正好放在我们希望的位置上。调整浏览器的大小，来看奖杯如何显示。

there are no Dumb Questions

问：听上去绝对定位比浮动要好，因为我可以对元素放在哪里有更多控制。是不是应该更倾向于绝对定位而不是浮动呢？

答：不一定。这要看你需要什么。如果确实需要一个元素出现在页面中某个精确的位置上，那就该选择绝对定位。不过，如果你有其他的需要，例如，希望文本围绕一个图像，利用绝对定位就不容易做到了。在这种情况下，你肯定希望使用浮动。你会发现这两种方法都经常用到。

问：我尝试着对两个 元素绝对定位，其中一个总是显示在另一个上面。有没有办法改变哪一个在上面？

答：有，每个定位元素都有一个“z-index”属性，这是元素在一个虚拟z轴上的顺序（可以认为这个z轴从屏幕指向你）。要这样使用这个属性：

```
#div1 {
    position: absolute;
    top:      30px;
    left:     30px;
    z-index:  0;
}
#div2 {
    position: absolute;
    top:      30px;
    left:     30px;
    z-index:  1;
}
```

这些规则会把id为“div2”的元素放在id为“div1”的元素上面。

问：我怎么知道默认的页面上各个元素的z-index是多少？

答：通常你不会知道，除非用开发工具检查相应的CSS，来确定浏览器如何计算页面上各个元素的z-index属性。不过大多数情况下，你并不关心元素的z-index，除非你要对元素建立某种特定的分层，或者遇到像这里放置奖杯之类的情况。通常只需要把z-index设置为1，这就足以确保元素出现在页面中其他元素的上面，不过，如果有多个元素，而且你需要自行定位和分层，就要更慎重地考虑这些元素的z-index值。

问：有没有最大的z-index值？

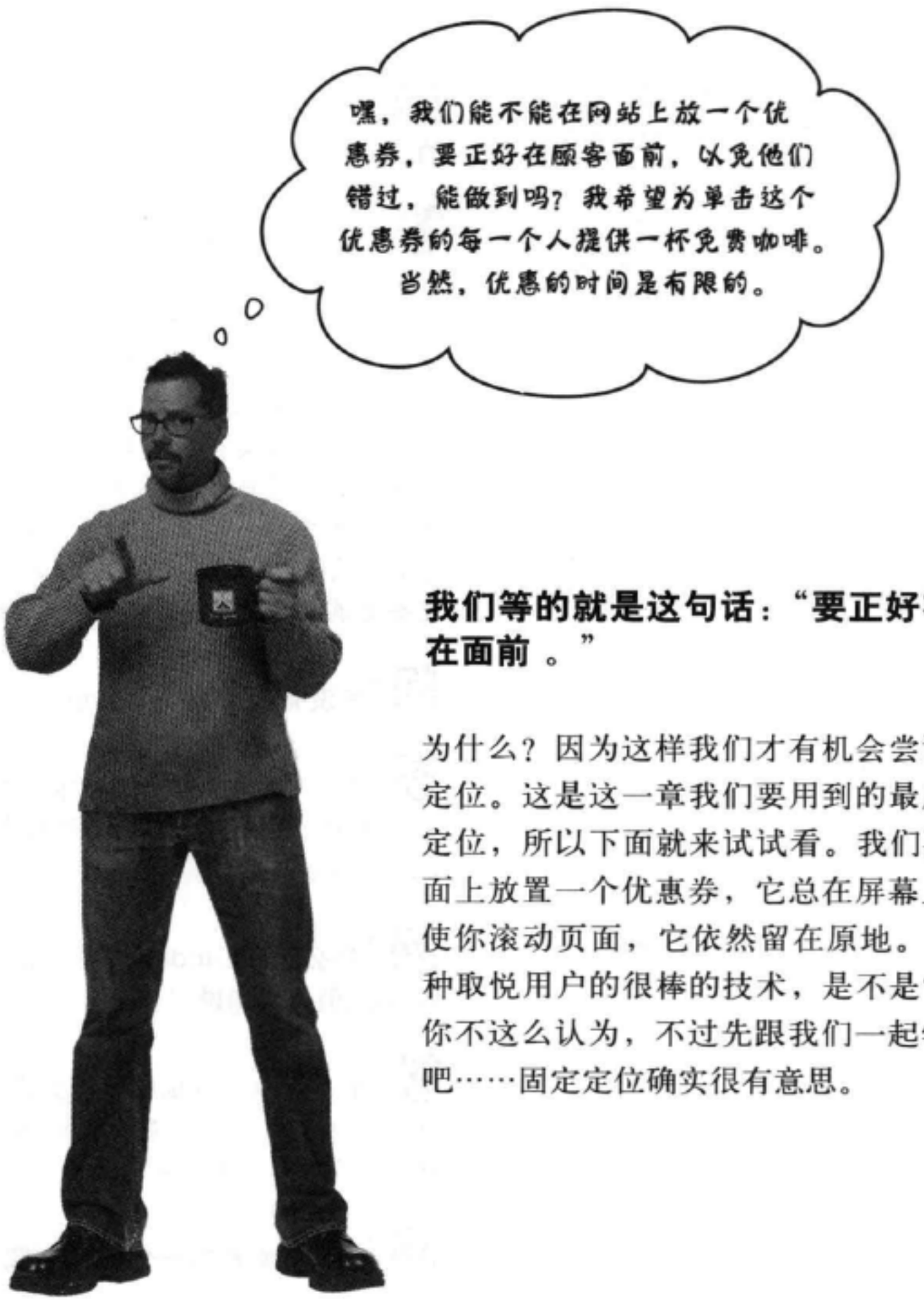
答：有，不过这是一个非常大的数，而实际来讲，你永远也不会用到那么大的z-index值。

问：那么负的z-index值呢？能不能有负z-index值，比如说–1？

答：能，可以有负值！对于负值，之前的原则同样适用（也就是说，值越大，层次就高，在屏幕上离你就越近）。

问：任何元素都有一个z-index吗？

答：不，只有使用CSS绝对定位、相对定位或固定定位的元素有z-index。接下来你会看到一个固定定位的例子！



我们等的就是这句话：“要正好在顾客在面前 。”

为什么？因为这样我们才有机会尝试固定定位。这是这一章我们要用到的最后一种定位，所以下面就来试试看。我们要在页面上放置一个优惠券，它总在屏幕上，即使你滚动页面，它依然留在原地。这是一种取悦用户的很棒的技术，是不是？也许你不这么认为，不过先跟我们一起学下去吧……固定定位确实很有意思。

固定定位如何工作？

与绝对定位相比，固定定位很简单。使用固定定位时，也像绝对定位一样要为元素指定你希望的位置，不过这个位置是距浏览器窗口边界的一个偏移量，而不是距页面边界的距离。这就有一个有趣的效果，一旦采用固定定位方式放置内容，它就会一直留在你原先指定的位置，不再移动，即使你滚动页面它也原地不动。

把浏览器窗口称为视窗，这会让你的朋友和同事对你刮目相看。试试吧！这是可以的，而且W3C也会点头同意。

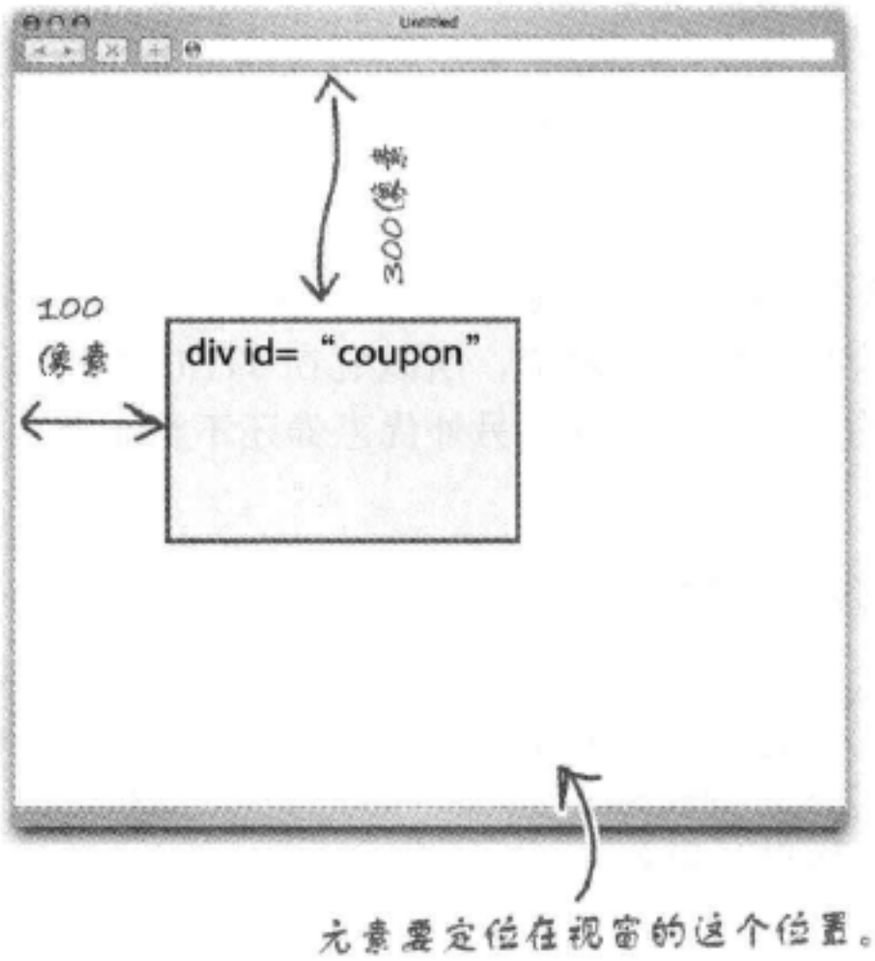
所以，假设有一个 ，id为“coupon”。可以把这个 固定在距视窗上方300像素，距左边100像素的一个位置上，就像这样：

这是选择coupon 的id选择器。

我们要使用固定定位。

```
#coupon {
    position: fixed;
    top:      300px;
    left:     100px;
}
```

将优惠券定位在距上边300像素、距左边100像素的位置上。就像完成绝对定位时一样，也可以使用right和bottom指定位置。



元素要定位在视窗的这个位置。

一旦指定了元素的位置，下面就有意思了：你可以随便滚动页面……这个元素不会移动。你可以调整窗口大小……这个元素仍然不会移动。拿起你的显示器，使劲摇晃……它依然不会移动。OK，最后一个说法是开玩笑的，别当真。不过，我们要强调的重点是，固定定位元素不会移动。只要显示页面，它们就永远在原地不动。

现在，你肯定已经在考虑要用固定定位来实现一些有趣的效果，不过首先要完成一个任务。下面把这个优惠券放在Starbuzz页面上。

把优惠券放在页面上

现在我们要在页面上增加这个免费咖啡优惠券（Free Coffee Coupon）。首先创建一个 来包含这个优惠券：

这是id为"coupon"的 。

里面有一个优惠券图像，可以在"chapter11/starbuzz/images"文件夹中找到这个图像文件。

```
<div id="coupon">
    <a href="freecoffee.html" title="Click here to get your free coffee">
      <img src="images/ticket.gif" alt="Starbuzz coupon ticket">
    </a>
</div>
```

把图像包围在一个<a>元素中，这样用户单击这个图像时就会进入一个页面，可以在那个页面中打印优惠券。

下面把这个 增加到"index.html"文件的最后，但要放在页脚上面。因为我们要对它定位，所以它在HTML中的位置只会对那些不支持定位的浏览器有影响，另外优惠券还不至于重要到非得放在页面最上面。

现在编写CSS来指定优惠券的位置：

```
#coupon {
    position: fixed;
    top: 350px;
    left: 0px;
}

#coupon a, img {
    border: none;
}
```

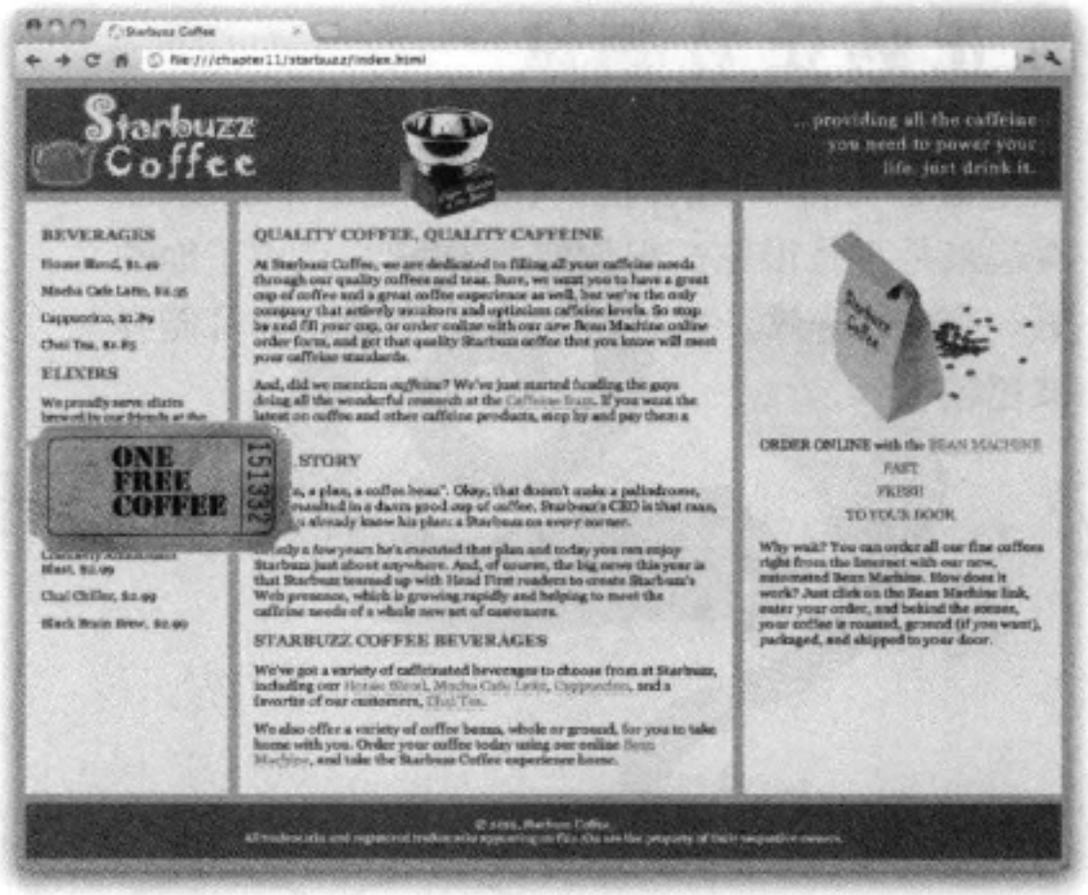
可以设置优惠券的固定位置，距视窗上边350像素，它的左边紧挨着视窗边界放置。所以需要指定距左边0像素。

还需要指定图像和链接的样式。否则，图像上就会有边框，因为它是可单击的。所以，下面设置图像的边框为none，对链接也做同样的设置。对于这两种元素我们要同一个属性，所以把这两个规则合并为一个。

记住，CSS中有一条规则，要求关闭text-decoration，而使用边框来建立链接的下划线。这里会覆盖对应coupon 中链接的规则，指出我们不希望这个链接上有任何边框。如果你忘了链接的其他规则，可以回顾一下原来的CSS。

把优惠券放在页面上

将这些新的coupon规则增加到“starbuzz.css”文件中，保存，然后重新加载页面。你可能要让浏览器小一点，这样才能看到。即使滚动页面，优惠券仍保持在原地不动。单击这个优惠券会带你进入“freecoffee.html”页面。



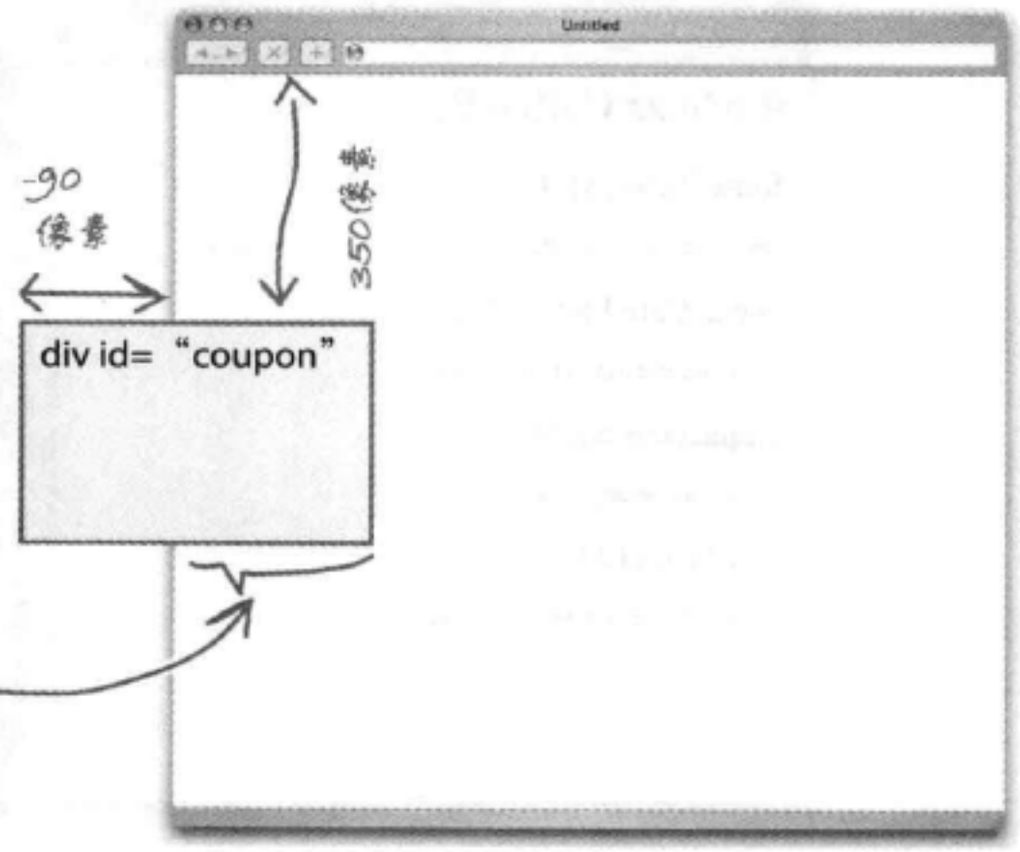
这看起来很不错，不过如果优惠券能偏到左边，它会显得更时髦，这样一来，它看起来就像是从视窗一边飞出的。为了达到这个目的，可以进入我们的照片编辑软件，去掉图像的左边来创建这种效果。或者也可以使用一个负偏移量，使图像的左边定位在比视窗边界更左的位置。好吧，就这么做。

使用负的left属性值

指定一个负的属性值与指定正值是一样的，只需要在前面加一个负号。如下：

```
#coupon {
    position: fixed;
    top: 350px;
    left: -90px;
}
```

通过指定-90像素，我们在告诉浏览器把这个图像定位在距视窗边界左边90像素的位置。

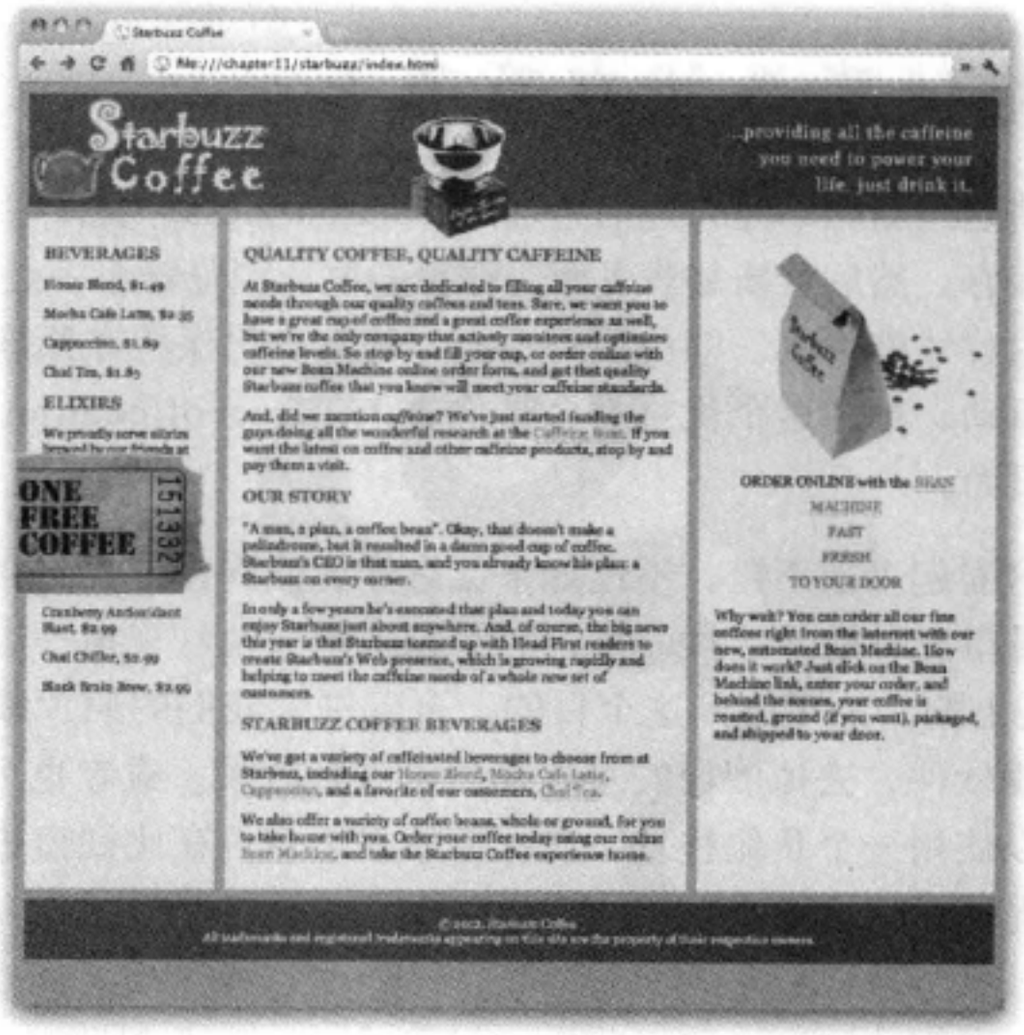


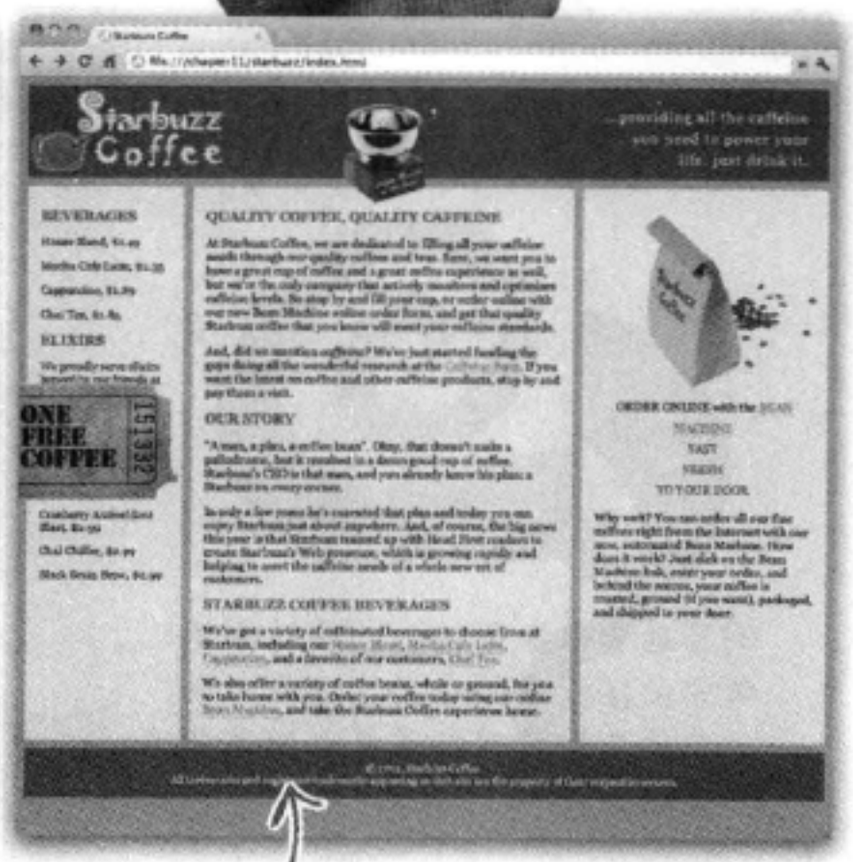
浏览器会很高兴地为你把这个图像定位在视窗的左边，这样一来，你只能看到仍留在屏幕上的那部分图像。

真正的正负测试

确保指定负的left属性值，保存并重新加载页面。看起来是不是很炫？祝贺你，你刚刚完成了你的第一个CSS特效。当心，乔治·卢卡斯，你可能会被超越哦！

不过要记住，使用固定定位“盖住”内容的做法对用户并不是最友好的，不过确实很有意思。

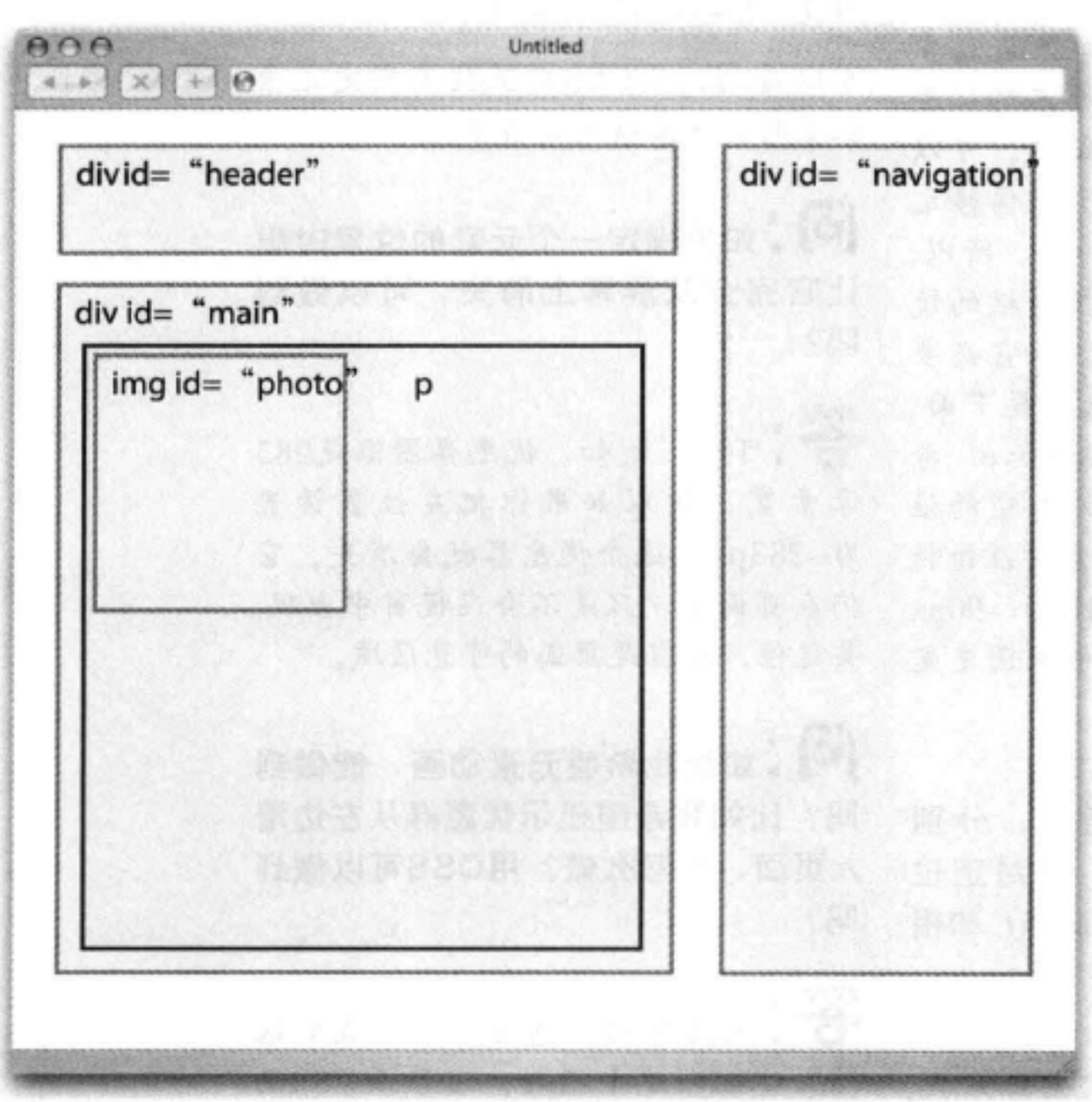




哇！真是天壤之别！

Sharpen your pencil

现在把关于浮动和绝对定位的所有知识汇总在一起，来完成这个测试！来看下面的Web页面。这里有4个元素，分别有自己的id。你的任务是将这些元素分别与右边的CSS规则正确匹配，为各个规则填入正确的id选择器。对照这一章最后的答案检查你的结果是否正确。



填入选择器完成这个CSS。

```
.....{
    margin-top:   140px;
    margin-left:  20px;
    width:        500px;
}

.....{
    position: absolute;
    top:      20px;
    left:     550px;
    width:    200px;
}


.....{
    float: left;
}

.....{
    position: absolute;
    top:      20px;
    left:     20px;
    width:    500px;
    height:   100px;
}
```

there are no Dumb Questions

问：固定优惠券确实很酷，不过有点烦人。有没有别的办法对它定位，而且不要与内容重叠，比如说放在饮料栏的下面？

答：当然有办法。可以使用相对定位指定优惠券的位置，把它放在饮料栏的下方。我们没有介绍这种定位，不过这与绝对定位很类似，只是元素仍然在页面流中（还在它原本的位置上），然后按你指定的量偏移。可以使用top、left、bottom或right偏移元素，就像对元素绝对定位一样。所以，假设你希望优惠券出现在饮料栏的饮料下面，就要移动优惠券，让它嵌套在“drinks” 中，放在最下面，然后把position属性设置为relative。再由你将优惠券准确地放在你希望的位置上，可以用top: 20px指定它在饮料下面20像素的位置，另外用left: –90px让它在页面的左边（就像使用固定定位一样）。

问：这么说，有4种定位，分别是静态定位（static）、绝对定位（absolute）、固定定位（fixed）和相对定位（relative），是吗？

答：没错。如果没有指定定位方式，会默认为静态定位。这会将所有内容正常地流入页面。绝对定位将元素完全从页面流中取出，允许你为它指定一个绝对位置，这是相对于离它最近的父元素指定的（这一般是<html>，除非你自行指定了另外一个父元素）。固定定位则是相对于浏览器窗口，把元素放在一个特定的固定位置上，而相对定位会相对于其外围包含元素来定位，元素仍在正常的页面流中，然后再按你指定的量偏移元素。还可以结合使用这些定位技术。例如，我们说过，绝对定位的元素要相对于位置最近的父元素来定位，还记得吧？完全可以将一个 放在另一个 中，对外围 使用相对定位（它仍在页面流中），然后用绝对定位指定内部 的位置，这样你就能相对于父 对它定位了。

可以看到，用CSS定位技术指定元素位置时，方法确实相当多。

问：如果指定一个元素的位置时想让它完全从屏幕上消失，可以做到吗？

答：可以！例如，优惠券图像是283像素宽，所以如果你把左位置设置为–283px，这个优惠券就会消失。它仍在页面上，只是不会在视窗中出现。要记住，视窗是页面的可见区域。

问：如果我希望元素动画，能做到吗？比如我希望显示优惠券从左边滑入页面，该怎么做？用CSS可以做到吗？

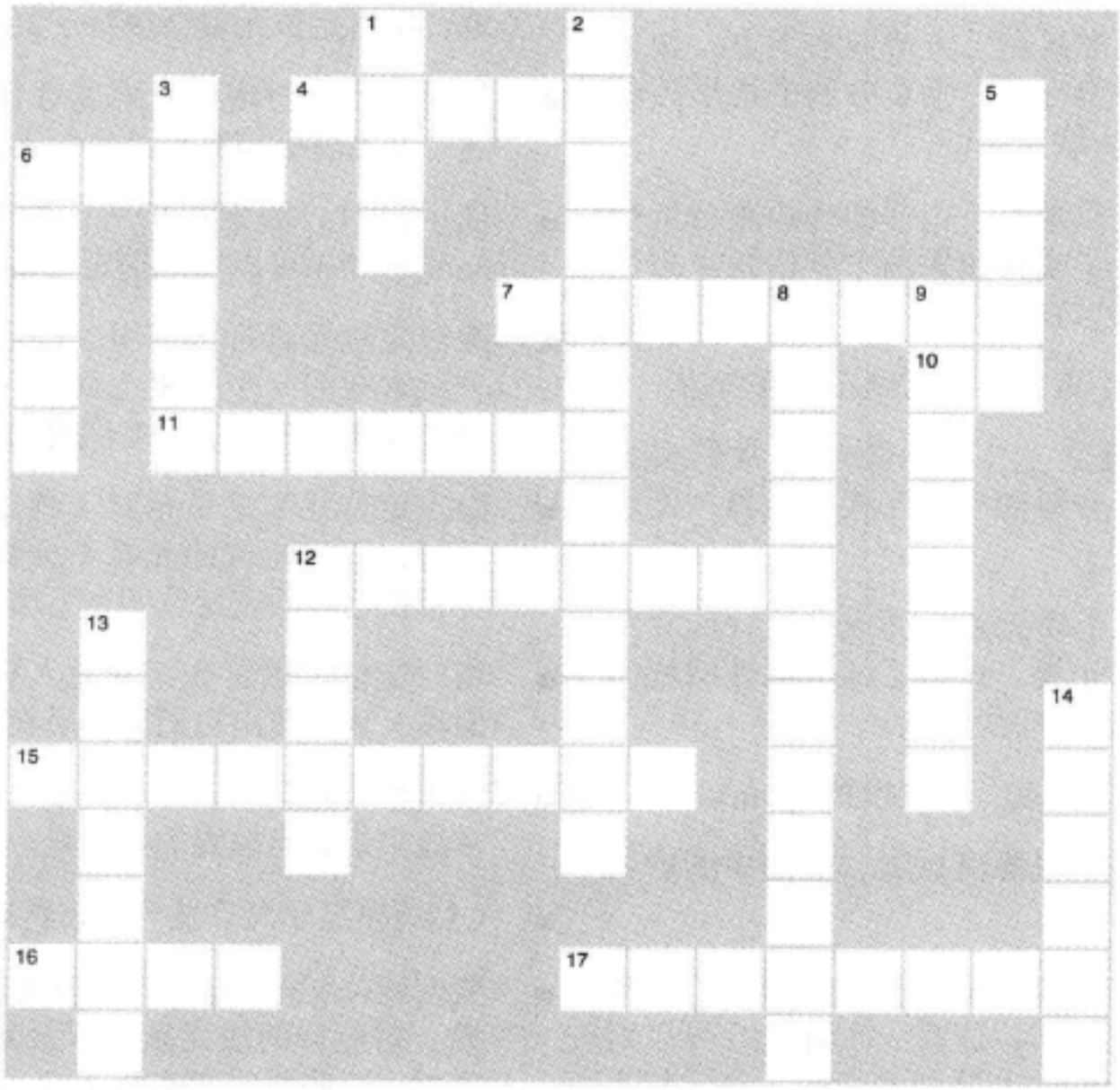
答：说实在的，确实可以，我们很高兴你能问这个问题。这涉及CSS动画，超出了这本书的范围，不过要知道，CSS3为元素引入了基本动画，提供了变换和过渡特性，这让我们Web开发人员欣喜不已。这些特性功能还很有限，不过，利用CSS动画确实可以完成一些很酷的效果。如果你想要的效果用CSS做不到，就要用到JavaScript了，这又是另一个内容。我们会在附录中对CSS变换和过渡做一个简要的介绍，吊吊你的胃口。

BULLET POINTS

- 浏览器使用流在页面中放置元素。
- 块元素从上向下流，各元素之间有一个换行。默认的，每个块元素会占浏览器窗口的整个宽度。
- 内联元素在块元素内部从左上方流向右下方。如果需要多行，浏览器会换行，在垂直方向上扩展外围块元素，来包含这些内联元素。
- 正常页面流中两个块元素上下相邻的外边距会折叠为最大外边距的大小，或者如果两个外边距大小相同，则会折叠为一个外边距。
- 浮动元素会从正常流中取出，浮动到左边或右边。
- 浮动元素放在块元素之上，不会影响正常的页面流。不过，内联内容会考虑浮动元素的边界，围绕着这个浮动元素。
- clear属性用来指定一个块元素左边或右边（或者左右两边）不能有浮动元素。设置了clear属性的块元素会下移，直到它旁边没有块元素。
- 浮动元素必须有特定的宽度，不能设置为auto。
- 流体布局是指，扩展浏览器窗口时，页面中的内容会扩展以适应页面。
- 冻结布局是指，其中内容的宽度是固定的，不会随着浏览器窗口扩展或收缩。这有一个好处，可以对设计提供更多控制，不过也要付出代价，这样就不能有效地使用浏览器宽度了。
- 凝胶布局是指，其中内容宽度是固定的，但是外边距会随着浏览器窗口扩展或收缩。凝胶布局通常会把内容放在中央。这与冻结布局有同样的好处，不过通常更美观。
- position属性可以设置为4个值：static（静态）、absolute（绝对）、fixed（固定）和relative（相对）。
- 静态定位是默认方式，将元素放在页面的正常流中。
- 绝对定位允许将元素放在页面上的任何位置。默认地，绝对定位元素会相对于页面边界来放置。
- 如果一个绝对定位元素嵌套在另一个定位元素中，这个元素就会相对于外包含元素定位。
- 使用绝对、固定和相对定位时，属性top、right、bottom和left可以用来指定元素的位置。
- 绝对定位元素可以使用z-index属性分层放置，使一个元素在另一个元素上面。z-index值越大，说明它层次越高（在屏幕上离你越近）。
- 固定定位元素总是相对于浏览器窗口定位，页面滚动时，固定定位的元素不会移动。页面中的其他内容会在这些固定定位元素下面正常滚动。
- 相对定位元素首先正常流入页面，然后按指定的量偏移，从而留出它们原先所在的空间。
- 使用相对定位时，left、right、top和bottom是指距正常流中该元素位置的偏移量。
- CSS表格显示允许按一种表格布局来摆放元素。
- 要创建CSS表格显示，需要使用对应表格的一个块元素，对应行的块元素，以及对应单元格的块元素。通常，这些块元素都是 元素。
- 如果需要建立多栏布局，而且内容栏是均匀的，表格显示就是一个很好的布局策略。

HTML填字游戏

这真是超负荷的一章，有太多的东西要学习。做做这个填字游戏，来帮你把学到的知识牢牢记住。所有答案都可以在这一章中找到。



横向

4. 流体和冻结之间的一个状态。
6. 浏览器使用这种方法指定页面上静态元素的位置。
7. 在优惠券上使用这种偏移量来得到一种特效。
10. 通常用来标识要定位的元素。
11. 盒子上下放置时，这些会折叠。
12. 两个内联元素相邻放置时，它们的外边距不会_____。
15. 绝对定位是相对于 _____ 块元素定位。
16. 内联元素从左_____开始流动。
17. 保证元素在流中的定位类型。

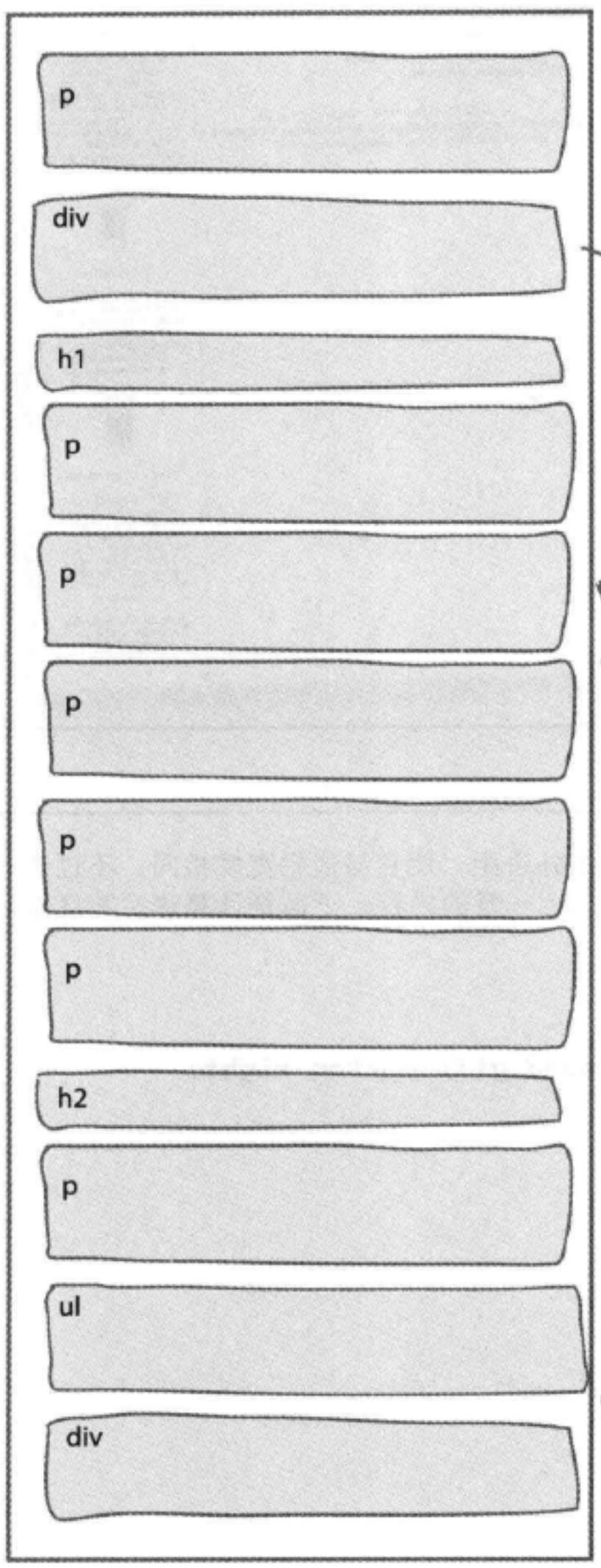
纵向

1. 建立页面布局时，这种特殊的内联元素会分组到盒子中。
2. 使用 _____ 在表格显示中创建单元格之间的间距。
3. 块元素从上流到_____。
5. 相对于视窗的定位类型。
6. 将元素从流中删除，将它设置到某一边。
8. 一般来讲，建立分栏布局较好的技术是_____。
9. 浏览器窗口的另一个名字。
12.用来修正页脚重叠问题的属性。
13.内联内容会围绕 _____元素。
14.描述定位元素分层行为的属性。

扮演浏览器答案

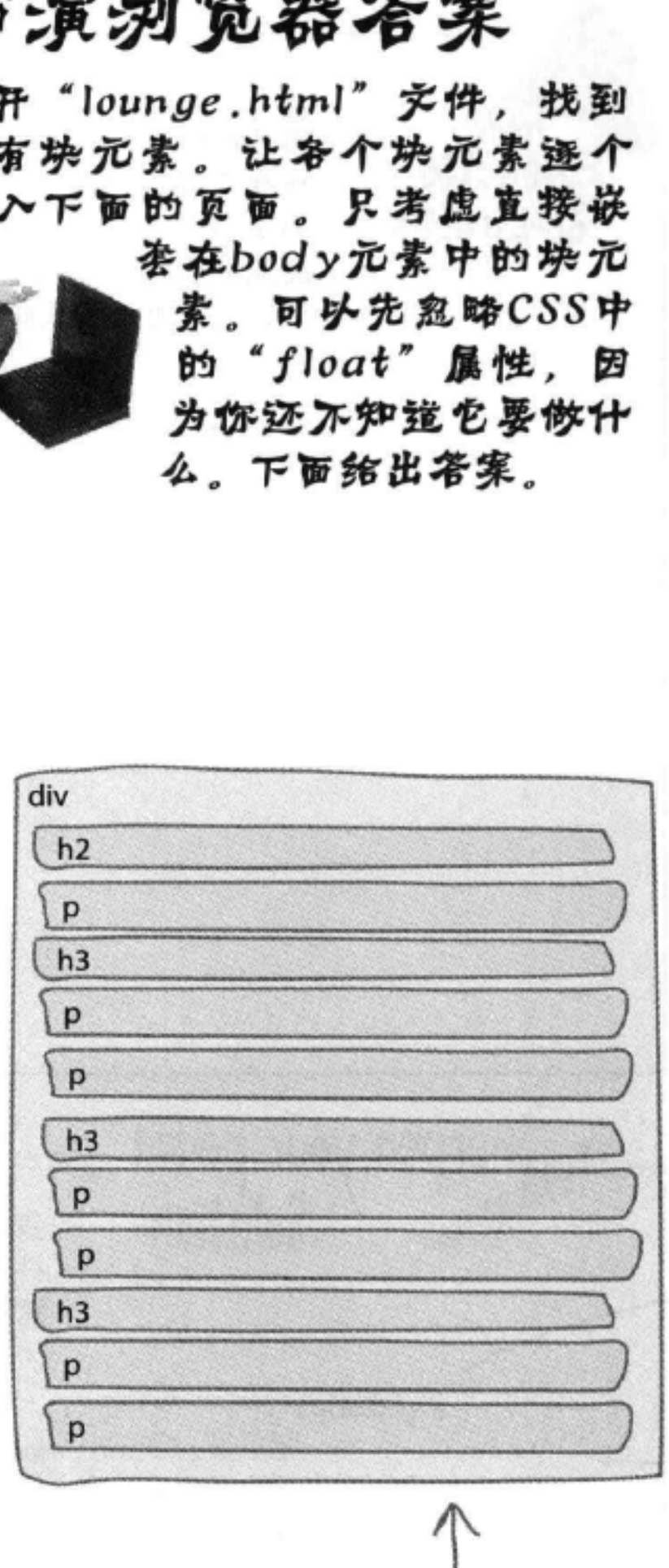
打开“lounge.html”文件，找到所有块元素。让各个块元素逐个流入下面的页面。只考虑直接嵌套在body元素中的块元素。可以先忽略CSS中的“float”属性，因为你还不知道它要做什么。下面给出答案。

这是你的页面，在这里画出“lounge.html”中的块元素流。



“lounge.html”文件中的各个块元素从上向下流，各元素之间有一个换行。

其中一些元素还包含嵌套的块元素，如<ul>、elixirs 和footer 。

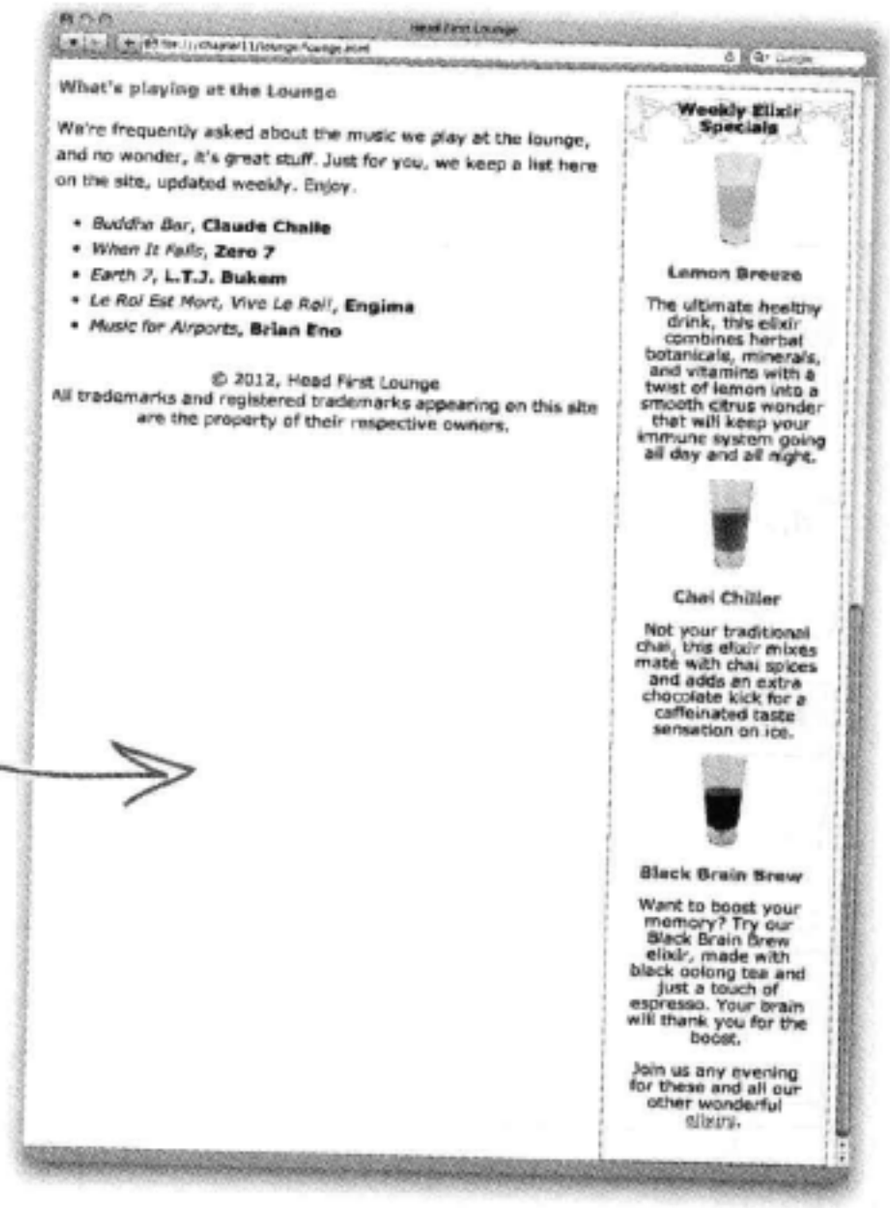


我们没有要求你画出嵌套的块元素流，不过如果你想更进一步，这里给出了elixirs 中的元素流。

Exercise Solution

将elixirs 移回到原来的位置，仍然放在主要推荐音乐下面，然后保存并重新加载页面。现在元素会浮动到哪里？应该能看到饮料区会在主内容下面，旁边是音乐推荐和页脚。

这个 浮动到右边，放在主要内容下面，HTML中的其余内容（音乐推荐和页脚）浮在它周围。



Sharpen your pencil Solution

我们希望在主内容区上设置适当的外边距，使它与边栏宽度相同。不过边栏有多大？嗯，我们希望你还没忘上一章的内容。下面是计算边栏宽度所需的全部信息。下面给出答案。

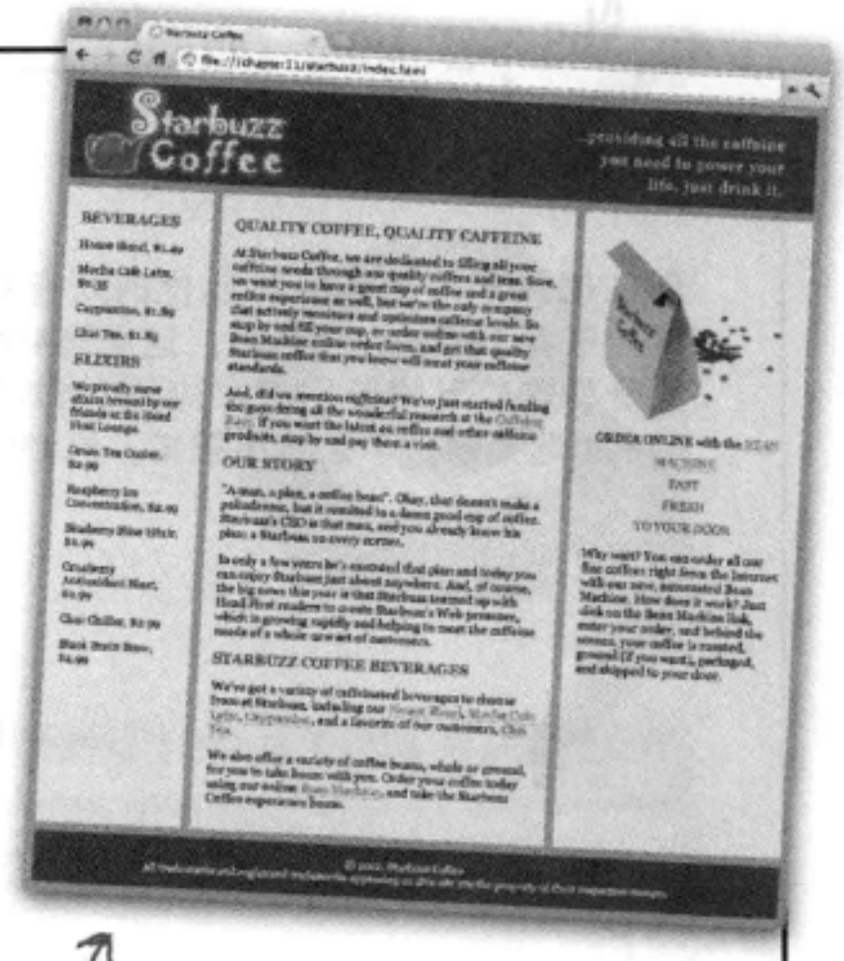
```
#sidebar {
        background: #efe5d0 url(images/background.gif) bottom right;
        font-size:  105%;
        padding:    15px;
        margin:     0px 10px 10px 10px;
        width:      280px;
        float:       right;
}
```

15 + 15 + 280 + 0 + 0 + 10 + 10 = 330

左内边距　右内边距　内容区　左边框　右边框　右外边距　左外边距

Exercise Solution

Starbuzz CEO决定再为Starbuzz Coffee页面增加一列，提供饮料单。他希望新加的这一列放在左边，宽度是浏览器窗口的20%。你的任务是在现有页面正确的位置上增加新的饮料单HTML，然后完成下面的CSS，确保它显示为一个表格单元格，就像另外两列一样。下面给出我们的答案。



这是CEO希望得到的Starbuzz页面，包含饮料单的新列在左边。

把这个HTML增加到"tableRow" 中，放在"main" 上面，这样它的内容就会最先出现，作为页面的第一列（也是表格布局中的第一个单元格）。

```
<div id="tableContainer">
  <div id="tableRow">
    <div id="drinks">
     <h1>BEVERAGES</h1>
      <p>House Blend, $1.49</p>
      <p>Mocha Cafe Latte, $2.35</p>
      <p>Cappuccino, $1.89</p>
      <p>Chai Tea, $1.85</p>
     <h1>ELIXIRS</h1>
      <p>
        We proudly serve elixirs brewed by our friends
        at the Head First Lounge.
      </p>
      <p>Green Tea Cooler, $2.99</p>
      <p>Raspberry Ice Concentration, $2.99</p>
      <p>Blueberry Bliss Elixir, $2.99</p>
      <p>Cranberry Antioxidant Blast, $2.99</p>
      <p>Chai Chiller, $2.99</p>
      <p>Black Brain Brew, $2.99</p>
    </div>
    <div id="main">
      ...
```

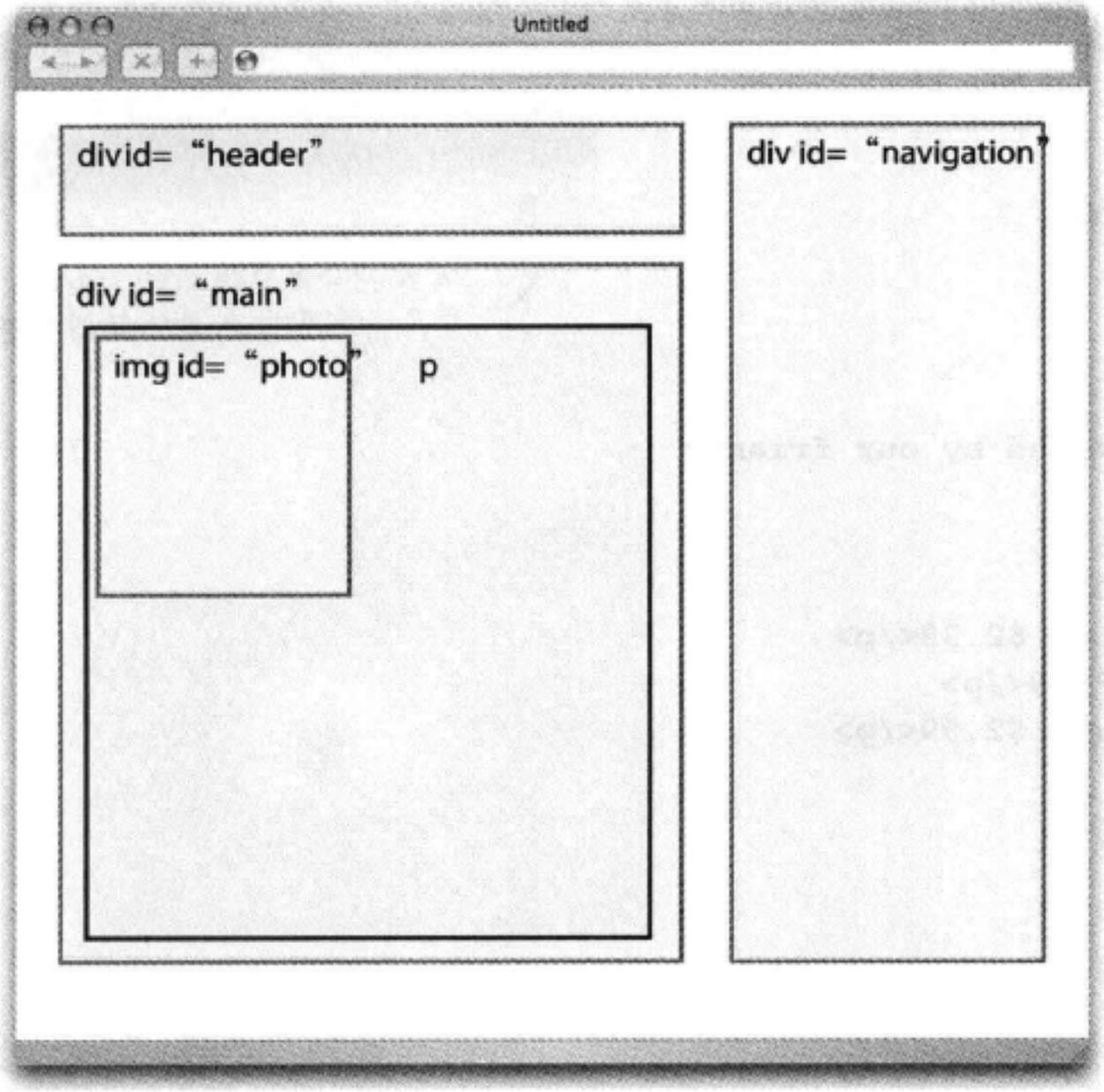
新的CSS……你需要完成这个CSS！

为了让drinks 显示为页面中的第一列，我们要将display设置为table-cell。

```
    #drinks {
        display: table-cell;
        background-color: #efe5d0;
        width: 20%;
        padding: 15px;
        vertical-align: top;
    }
```

Sharpen your pencil Solution

现在把关于浮动和绝对定位的所有知识汇总在一起，来完成这个测试！来看下面的Web页面。这里有4个元素，分别有自己的id。你的任务是将这些元素分别与右边的CSS规则正确匹配，为各个规则填入正确的id选择器。下面给出答案。你都做对了吗？



填入选择器完成这个CSS。

```
#main {
    margin-top:  140px;
    margin-left: 20px;
    width:       500px;
}

#navigation {
    position: absolute;
    top:      20px;
    left:     550px;
    width:    200px;
}

#photo {
    float: left;
}

#header {
    position: absolute;
    top:      20px;
    left:     20px;
    width:    500px;
    height:   100px;
}
```

这个CSS很容易，你可能闭上眼睛都能写出来，毕竟这一章你已经有了很多布局经验。编写CSS来修正页眉中的图像。你已经知道要用float，完成下面的填空，补全规则的其余部分，将图像放在正确的位置上。下面给出我们的答案。

```
#header img#headerSlogan ____ {
    float: ___right___;
}
```

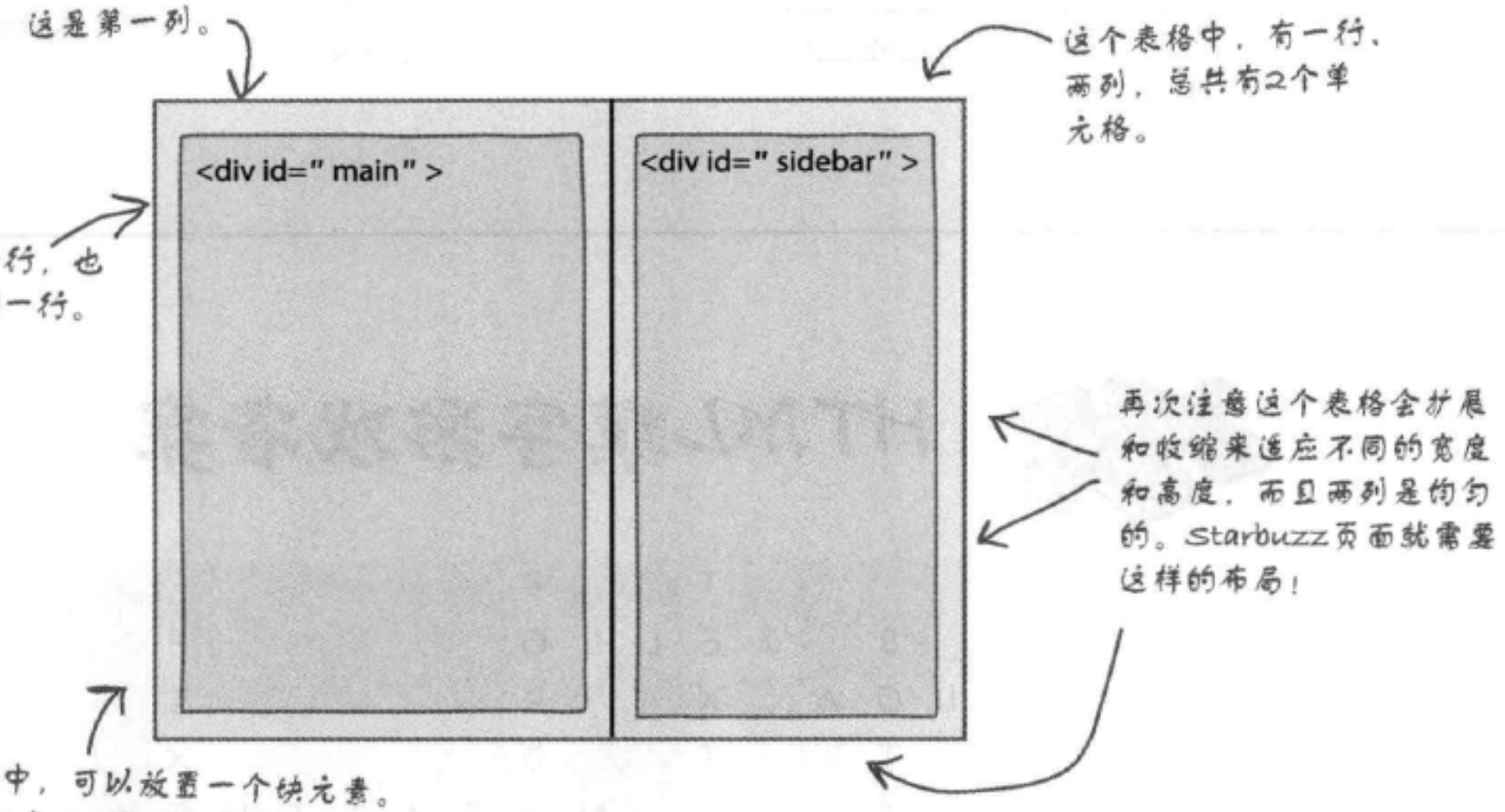
如果你愿意，这里还可以使用#headerSlogan作为选择器。

HTML填字游戏答案

					[1]T			[2]B								
		[3]B		[4]J	E	L	L	O							[5]F	
[6]F	L	O	W		X			R							I	
L		T			T			D							X	
O		T					[7]N	E	G	A	[8]T	I	[9]V		E	
A		O						R			A		[10]I		D	
T		[11]M	A	R	G	I	N	S			B		E			
								P			L		W			
				[12]C	O	L	L	A	P	S	E		P			
	[13]F			L				C			D		O			
	L			E				I			I		R		[14]Z	
[15]C	O	N	T	A	I	N	I	N	G		S		T		I	
	A			R				G			P				N	
	T										L				D	
[16]L	E	F	T					[17]R	E	L	A	T	I	V	E	
	D										Y				X	

Sharpen your pencil Solution

既然你已经了解了CSS表格显示，下面画出草图，指出如何把Starbuzz页面的两栏（“main”和“sidebar”）放在一个表格中。学习后面的内容之前，先对照这一章最后给出的答案检查你的结果……



12 HTML5标记

你肯定听到过关于HTML5铺天盖地的宣传。另外，这本书读到这里已经篇幅不短，你可能在怀疑是不是书买错了。现在要明确重要的一点，目前为止你在这本书里学到的所有东西都是HTML，更确切地讲，这些都满足HTML5标准。不过，HTML5标准对HTML标记有一些补充，这些方面我们还没有谈到，这一章就来介绍这些内容。HTML5新增的这些特性中，大多都是演进发展而来，你会发现，如果努力完成了这本书中之前的所有工作，掌握这些新内容对你来说也会很轻松。当然也有一些创造性的新特性（比如视频），这些也会在这一章中讲到。下面我们就开始吧，好好看一看这些新的内容！

重新考虑HTML结构

在学习更多的标记之前，下面先退一步……关于结构，我们已经说了不少，不过 确实是好结构吗？毕竟，浏览器并不知道你的 是一个页脚，它只知道这是一个 ，对不对？好像还不太让人满意，是不是？

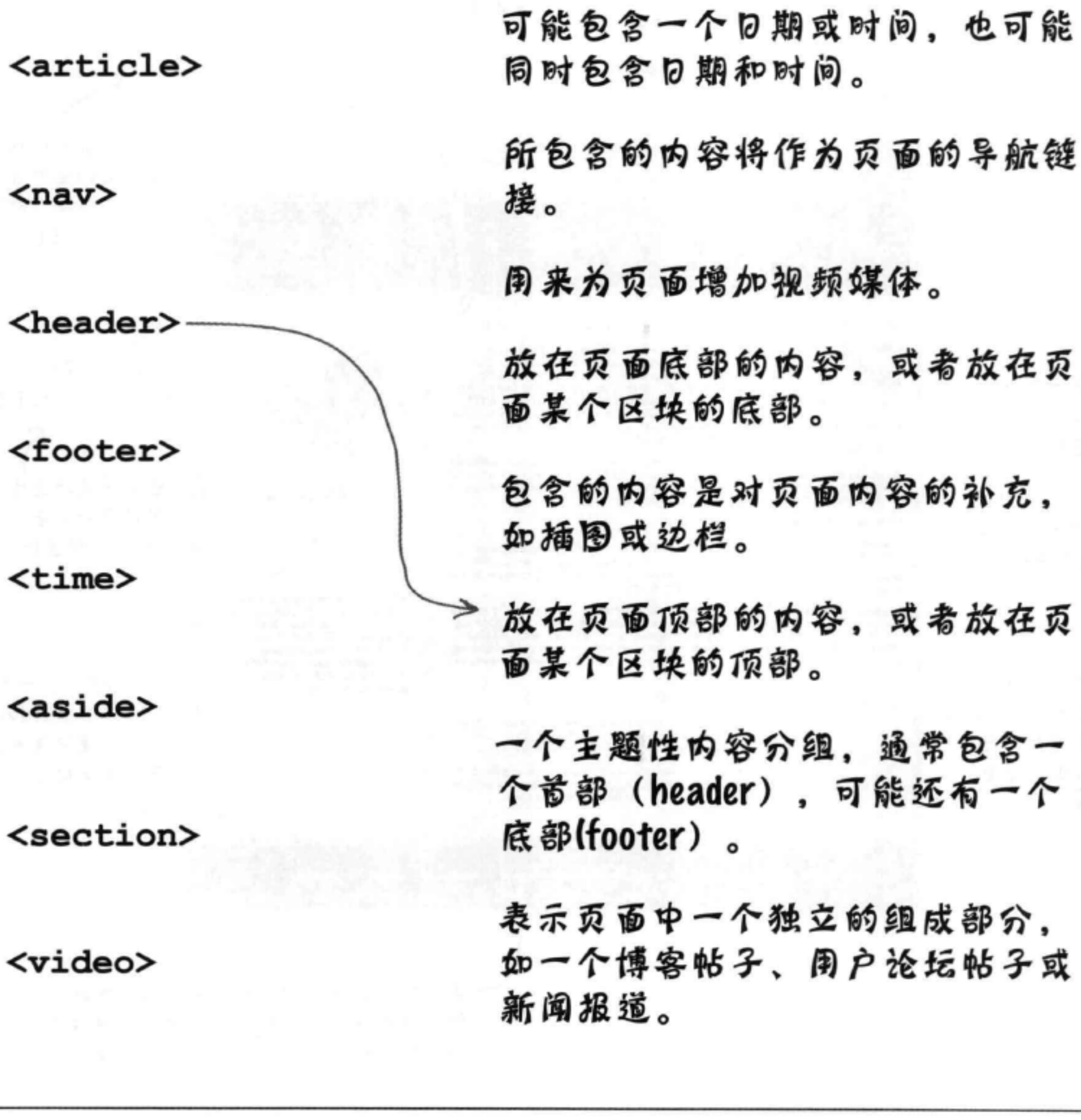
新的HTML5标记正是考虑到这一点，首先明确了人们如何利用 建立页面结构，相应地提供了更特定、更适合某些结构的标记。可以看到，浏览器（或搜索引擎，或屏幕阅读器）看到页面中的id="navigation"时，它们根本不知道这个 将用于导航。还不如把它叫做id="goobledygoop"。

所以，标准委员会仔细研究了人们究竟如何使用 元素（它们可能用于页眉、导航、页脚、文章等），然后增加了一些新的元素来表示这些结构。这说明，利用HTML5，我们可以对页面稍稍做些调整，把原来的 换成一些更特定的元素，能够更明确地指示其中包含什么类型的内容。

好好考虑一下之前 的用法。另外，查看一些Web页面，看看这些页面是如何使用 的。假设你想采用最常见的模式，把 改为真正的HTML元素。例如，可以把所有 元素都改为<footer>元素。列出你希望HTML增加的所有新元素。当然，你并不想增加太多新元素，只涵盖最常见的用途就足够了。另外写出增加这些新元素的优点（或缺点）：

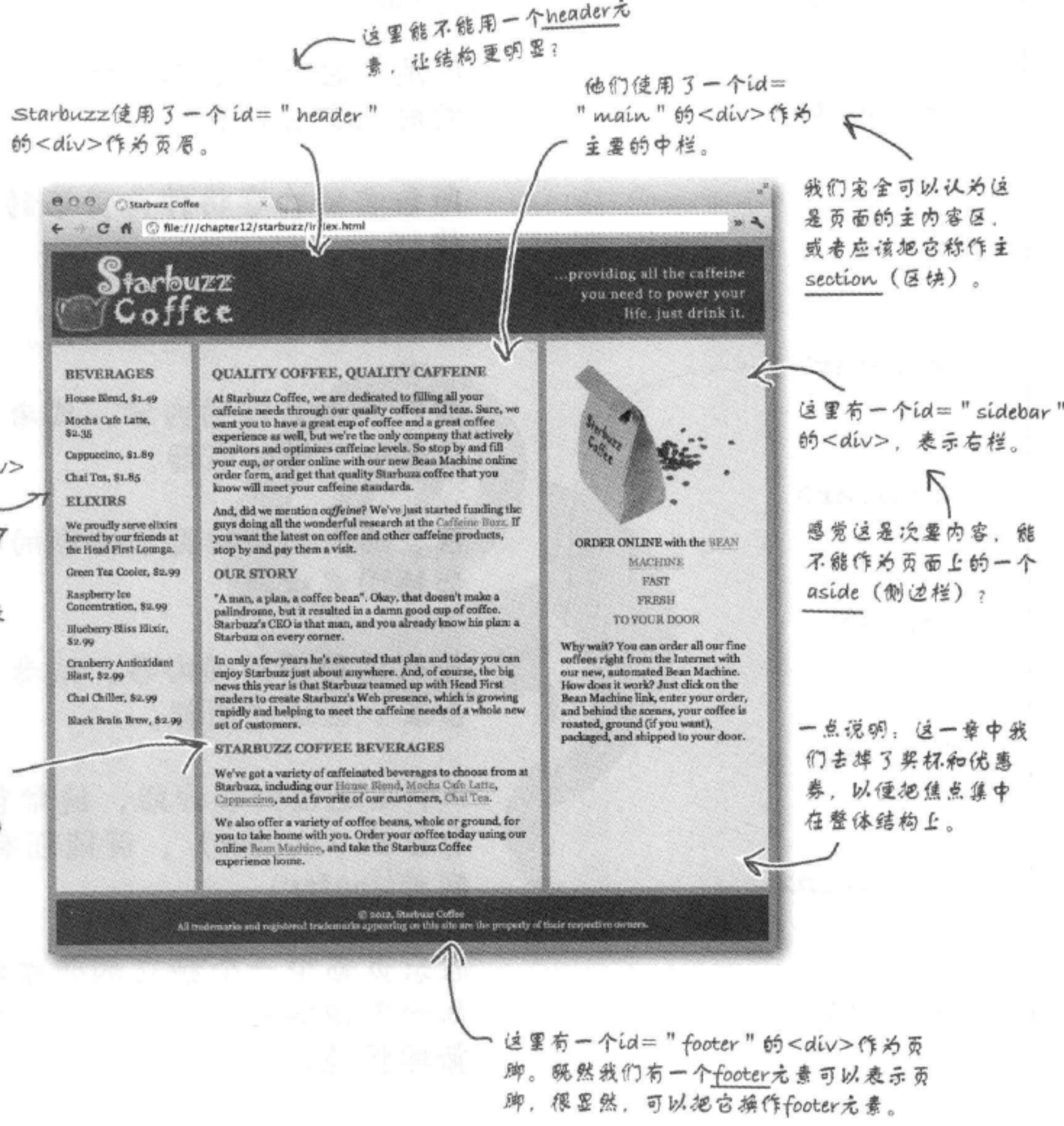
WHO DOES WHAT?

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：

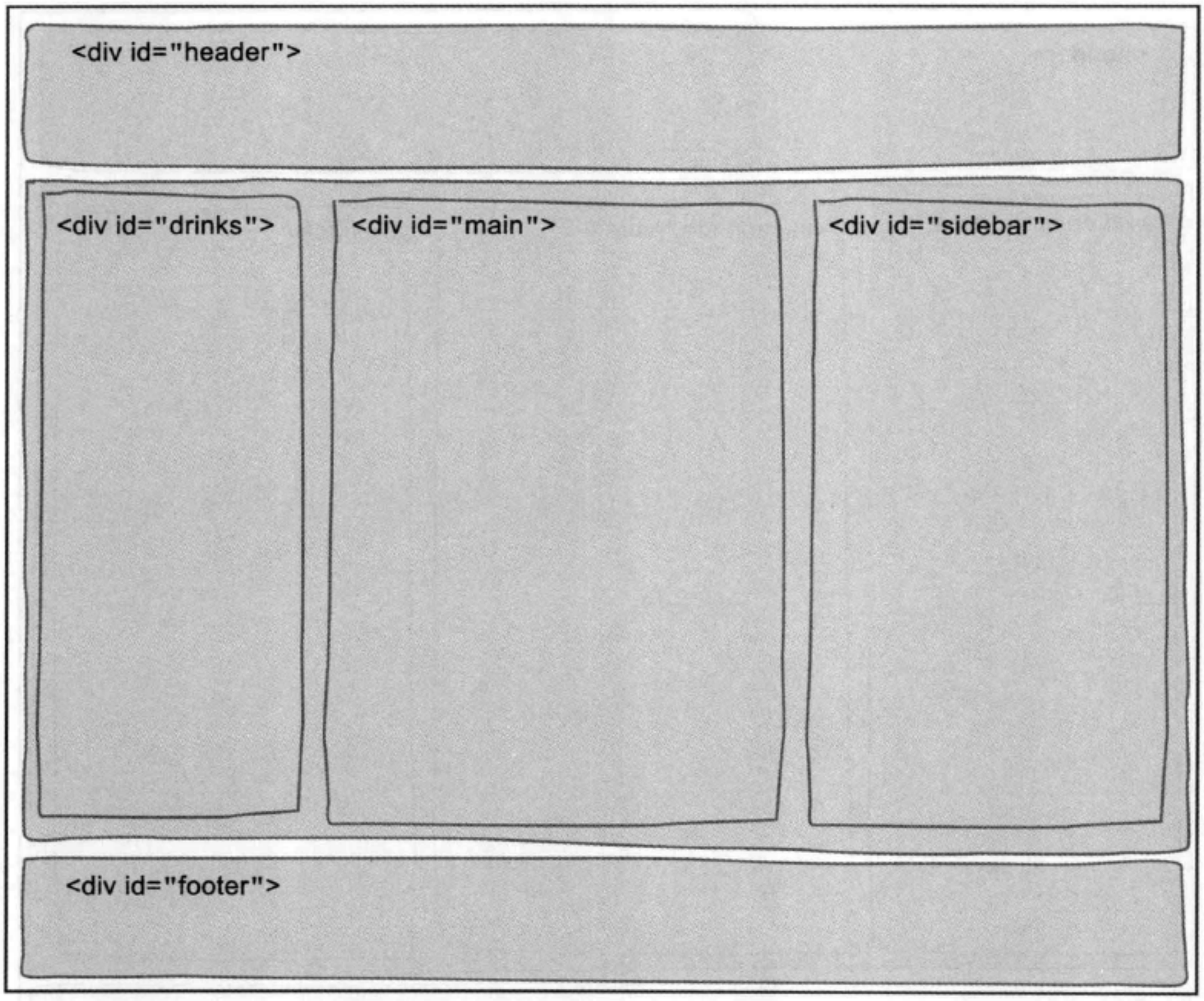


现代Starbuzz

Starbuzz Coffee是一个现代的新兴公司，既然如此，是不是也应该在页面中使用最新最潮的标记？下面来看他们在哪些地方可能错过了使用HTML5的机会：



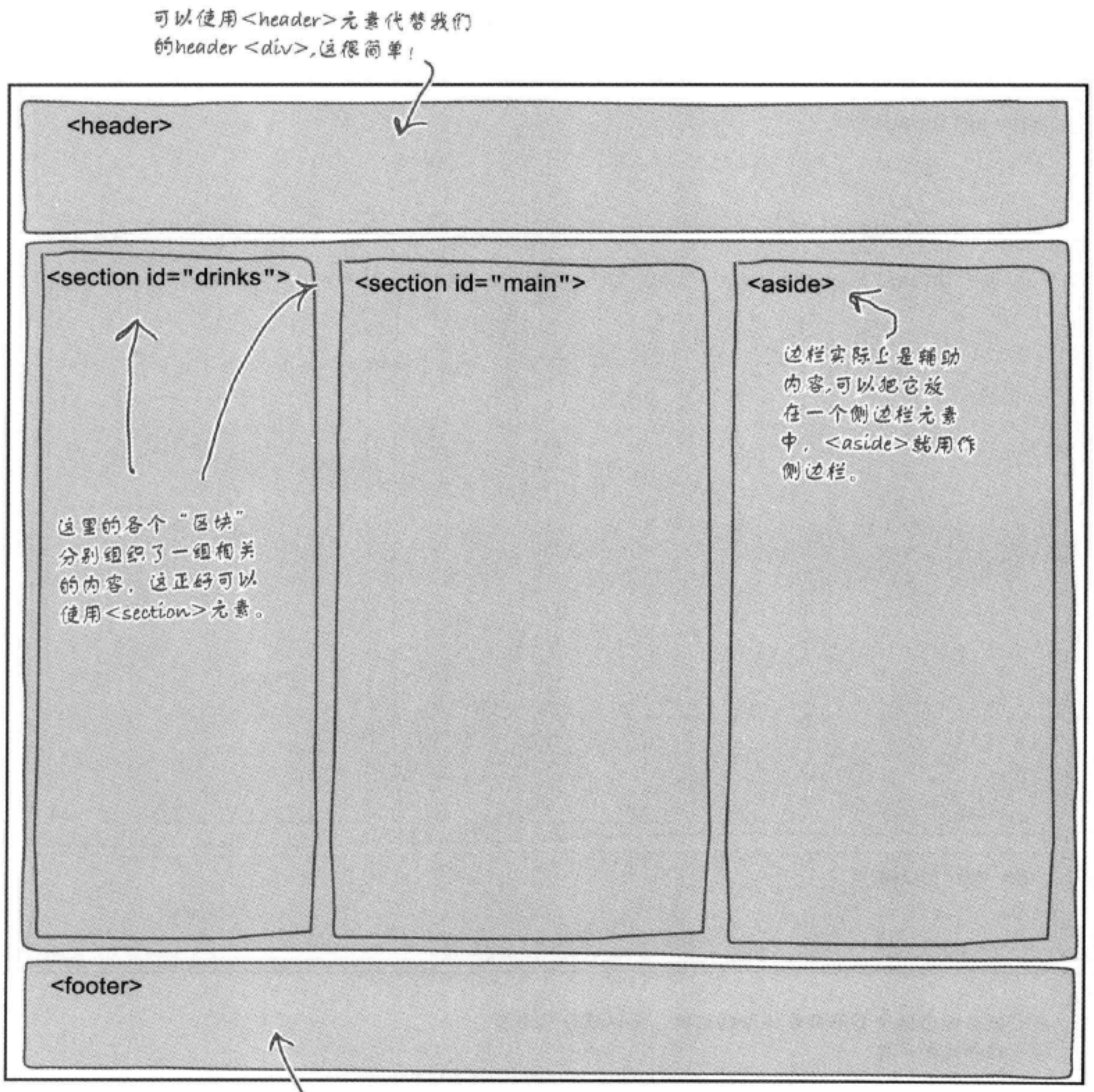
到目前为止，你对新的HTML5元素已经有所了解，利用你掌握的全部知识，看看能不能修改Starbuzz页面来利用这些新元素。只需要在这一页上简单标出来。



我们没有给出这个页面非常详细的结构，所以现在只考虑这个粗粒度的结构。

Exercise Solution

到目前为止，你对新的HTML5元素已经有所了解，利用你掌握的全部知识，看看能不能修改Starbuzz页面来利用这些新元素。只需要在这一页上简单标出来。



更新Starbuzz HTML

下面把这些新元素增加到Starbuzz HTML中，先从<header>、<footer>和<aside>元素开始。稍后还会讨论<section>元素，不过现在先保留饮料和主内容 。打开Starbuzz "index.html"文件，完成以下修改：

❶ 增加<header>元素。

先把 替换为一个<header>元素。如下：

```
<div id="header">
<header>
<img id="headerLogo"
     src="images/headerLogo.gif" alt="Starbuzz Coffee logo image">
<img id="headerSlogan"
     src="images/headerSlogan.gif" alt="Providing all the...">
</header>
</div>
```

删除 标记，把它们替换为<header>标记。

❷ 增加<footer>元素。

对 做同样的处理，只是要把它替换为一个<footer>元素：

```
<div id="footer">
<footer>
    &copy; 2012, Starbuzz Coffee
    <br>
    All trademarks and registered trademarks appearing on
    this site are the property of their respective owners.
</footer>
</div>
```

❸ 把边栏改为一个<aside>。

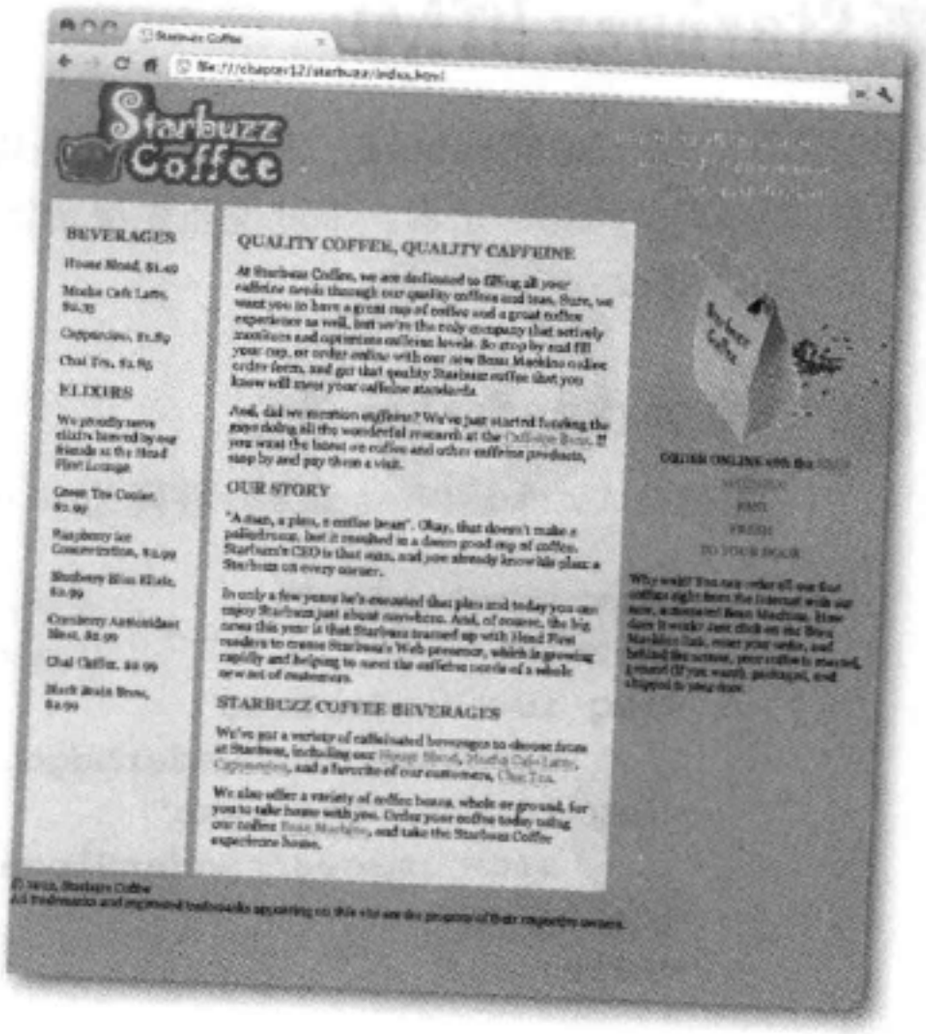
现在把"sidebar" 改为一个<aside>元素：

```
<div id="sidebar">
<aside>
    <p class="beanheading">
        <img src="images/bag.gif" alt="Bean Machine bag">
        ...
    </p>
    <p>
        ...
    </p>
</aside>
</div>
```

我们希望少浪费些纸，相应地少砍些树，所以对内容做了一点缩减，你要保留页面中原有的所有内容，只把 标记改为这里的<aside>标记。

测试新标记

我们只是稍稍做了些调整，不过有没有感觉你的HTML更时新、更清晰、更现代了？下面在浏览器中加载你的页面，来做个测试。



唉呀……看来还有些问题。

怎么回事？就是因为你说HTML5好，让我们都转到HTML5来。但是这个页面看起来实在不怎么样。

别担心，我们只是有些操之过急了。这个页面看起来不太妙，那是因为我们只改变了HTML，但是还没有调整CSS。可以这样来考虑，原来我们有一大堆 ，分别有自己的id，CSS就依赖于这些id建立样式，现在其中一些 已经没有了。所以我们需要改写CSS，要针对那些新元素而不是原来的 指定样式。现在就来做这个工作。

在继续学习之前……

Watch it!

较老的浏览器不支持这一章中将用到的HTML5新元素。

这一章中用到的元素是HTML5新增的，在较老的浏览器上并没有得到很好的支持（如IE8及更早版本，Safari 3及更早版本等）。如果你认为使用你的Web页面的人有可能还在用这些很老的浏览器，就不要使用这些新元素。

智能手机（如Android和iPhone）上的移动浏览器都支持这些新元素，所以，如果你面对的主要用户群体是移动用户，那就太好了！

可以查看http://caniuse.com/#search=new%20elements，了解浏览器是否支持这一章中使用的新元素，以及支持情况有没有更新。

如何为这些新元素更新CSS

下面更新CSS来反映我们的新元素。不用担心；我们保证CSS文件中的基本内容都是正确的。现在只需要稍稍修改选择器：

```
body {
  background-color: #b5a789;
  font-family:      Georgia, "Times New Roman", Times, serif;
  font-size:        small;
  margin:           0px;
}
#header {
header {
  background-color: #675c47;
  margin: 10px 10px 0px 10px;
  height:           108px;
}
#header img#headerSlogan {
header img#headerSlogan {
  float: right;
}

...

#sidebar {
aside {
  display:          table-cell;
  background:       #efe5d0 url(images/background.gif) bottom right;
  font-size:        105%;
  padding:          15px;
  vertical-align:   top;
}
#footer {
footer {
  background-color: #675c47;
  color:            #efe5d0;
  text-align:       center;
  padding:          15px;
  margin: 0px 10px 10px 10px;
  font-size:        90%;
}
...
```

首先，从header规则中删除#号。原来针对的是一个id为"header"的 ，现在要选择一个名为header的元素。

为了少砍树，这里节省些篇幅……就省这里是其余的CSS。

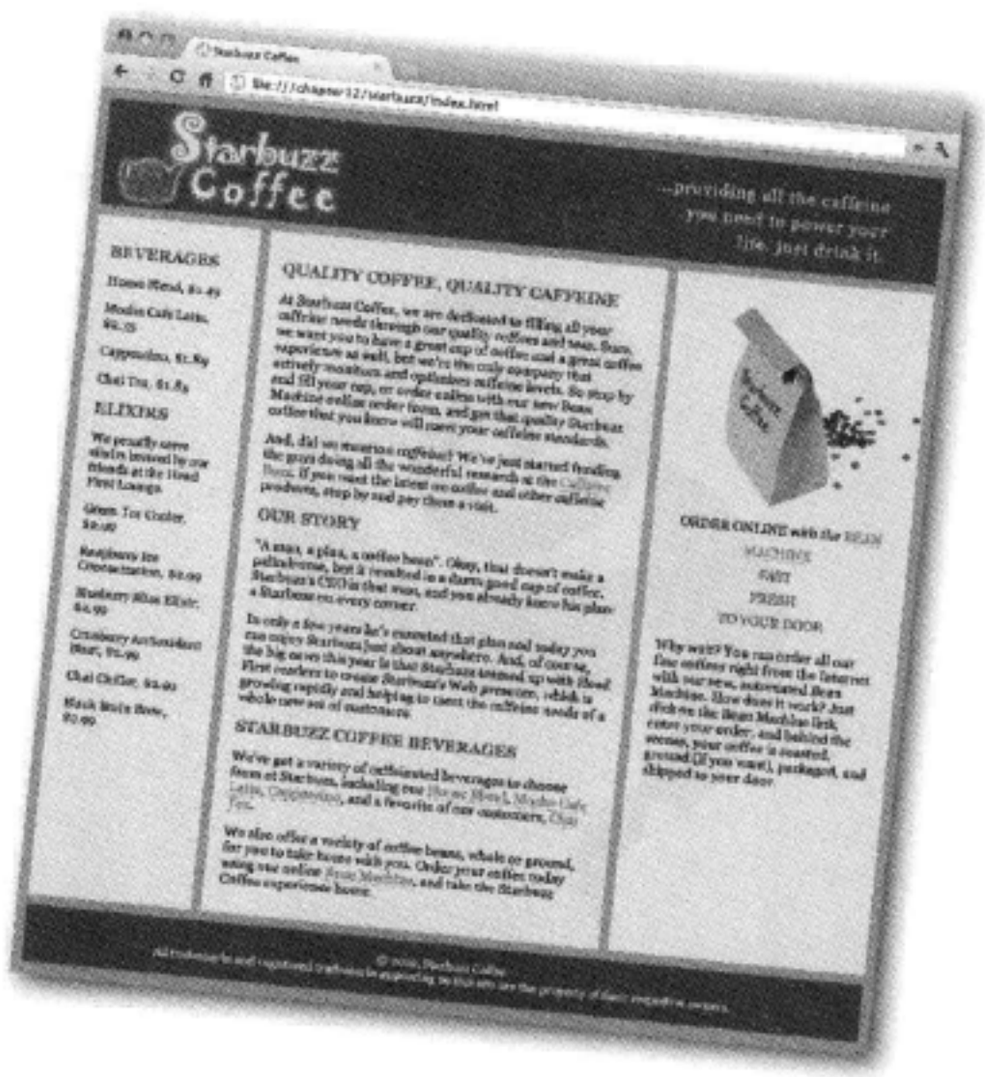
这里也需要修改，原来针对一个id为"sidebar"的元素，现在要改为选择一个aside元素。

最后，需要选择footer元素。

测试#2

哈……好多了。

好的，我们需要做的就是这些。下面再来试一试，这一次你会看到页面又恢复正常了。实际上，这就像我们增加HTML5标记之前的页面。



BRAIN POWER

既然在页面上没有任何视觉效果，增加这些新的HTML5标记有什么意义呢？

Fireside Chats

今晚话题：HTML5和HTML4.0的争论

HTML5

哈，老朋友，HTML4.01。之前你干得不错，不过现在我来了。

我已经开始起步了。

确实，人们才刚开始用那些新元素。记住，它们并不是用来改变这个世界，只是要明确Web开发人员在做什么。

我在考虑这里的 ……

我不是说要完全剔除 。没错，它在某些方面确实很棒，可以把内容分组在一起，来一同指定样式，不过如果你想把页面上的某个内容标识为一个文章呢？或者如果你想把页面划分为区块，该怎么办？

HTML4.01

何止是干得不错？看看Web吧，现在仍然是HTML4.01的天下。

是吗？那些新元素呢？很多我都还根本没有见过呢。

嘿，<p>不明确吗？这是一个段落。还有什么能比这更明确？

也没有任何问题。别去烦它。

你我都知道，对于如何使用这些元素，大家都很迷茫，另外，实际上这些工作 都能做到。

HTML5

对，用 确实也能做到，不过，那些新元素有一个好处，比如说，如果使用一个<article>元素，浏览器、搜索引擎、屏幕阅读器，还有Web开发人员，他们就能很肯定地知道这是一个文章。

记住，我们要物尽其用，要用最适合的元素来完成任务，对不对？使用这些新元素，就能最为明确地表示结构，而且我们的工具也能正确发挥作用。

嘿，这正是你理解不到位的地方。以<aside>元素为例，这个元素用来标记页面上的补充内容。现在假设有一个屏幕大小有限的移动手机，如果浏览器知道这个内容是一个<aside>，你可能会看到，这个内容会“塞”到页面最下面，这样你就能首先看到更重要的内容。相反，如果这个内容放在 中，那就什么情况都有可能发生，要看这个内容在HTML文件中的具体位置。

现在浏览器就能知道页面上的主内容与<aside>之间的不同。所以它会对<aside>中的内容区别对待。例如，搜索引擎可能会优先考虑页面上的主要内容，而不是<aside>中的内容。

不，不，不只是侧边栏，这适用于所有新的HTML标记：header、footer、section、 article、time都可以用不同方式处理。

屏　蔽

HTML4.01

那又怎么样？看起来还是一样的。

正确发挥作用？比如什么？提供完全相同的显示？

我还是看不出这有什么意义。

不错，这么说，利用HTML5，我们就能知道如何处理侧边栏，是吧？

嗯，我想该把页脚放在

屏　蔽

致编辑：它们有些失控了，能不能让它们回到正题上来，完成这个讨论？

Sharpen your pencil

你已经把“header”、“footer”和“sidebar” 分别替换为<header>、<footer>和<aside>元素。现在需要把“drinks”和“main” 换成<section>元素，还要更新你的CSS。暂时先保留用于表格显示的所有 ，我们还需要这些 保持页面的正确布局。

没有<section>元素的HTML。

划掉下面不需要的HTML和CSS，换成增加<section>元素所需的HTML和CSS。

```
<div id="tableContainer">
  <div id="tableRow">
    <div id="drinks">
      ...
    </div>
    <div id="main">
      ...
    </div>
    <aside>
      ...
    </aside>
  </div> <!-- tableRow -->
</div> <!-- tableContainer -->
```

目前对应#drinks和#main的CSS。

```
#drinks {
  display:          table-cell;
  background-color: #efe5d0;
  width:            20%;
  padding:          15px;
  vertical-align:   top;
}

#main {
  display:          table-cell;
  background:       #efe5d0
     url(images/background.gif) top left;
  font-size:        105%;
  padding:          15px;
  vertical-align:   top;
}
```

这些区块还需要id吗？如果需要，说说为什么？

Sharpen your pencil Solution

你已经把“header”、“footer”和“sidebar” 分别替换为<header>、<footer>和<aside>元素。现在需要把“drinks”和“main” 换成<section>元素，还要更新你的CSS。暂时先保留用于表格显示的所有 ；我们还需要这些 保持页面的正确布局。

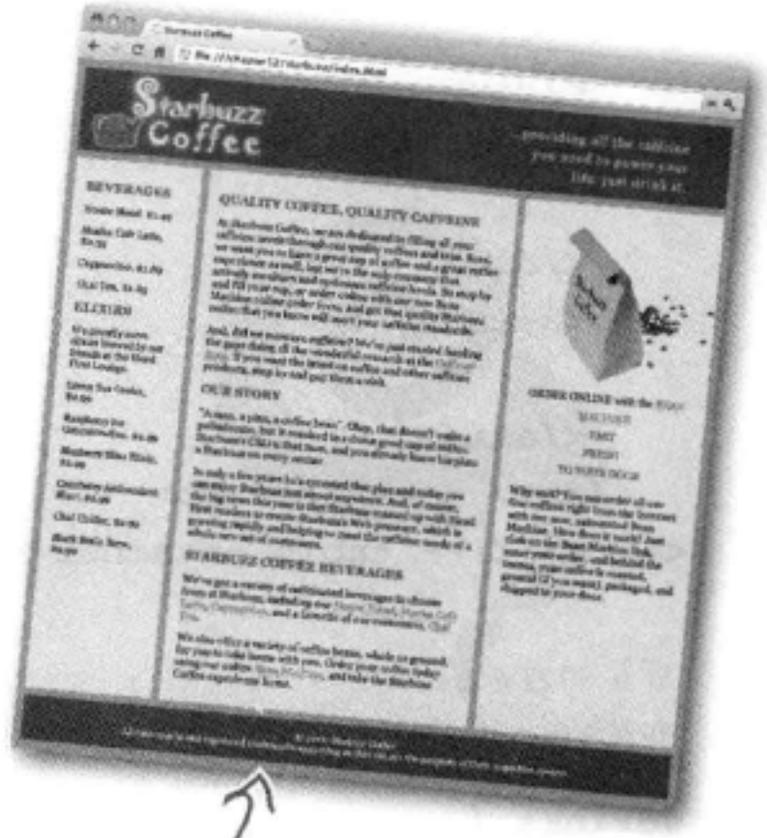
下面是我们的答案：

换成<section>元素的HTML。

```
<div id="tableContainer">
  <div id="tableRow">
    <section id="drinks">
      ...
    </section>
    <section id="main">
      ...
    </section>
    <aside>
      ...
    </aside>
  </div> <!-- tableRow -->
</div> <!-- tableContainer -->
```

我们所做的就是把“drinks”和“main” 换成<section>。

这里保留了id，因为还需要唯一标识各个<section>，以便指定样式。



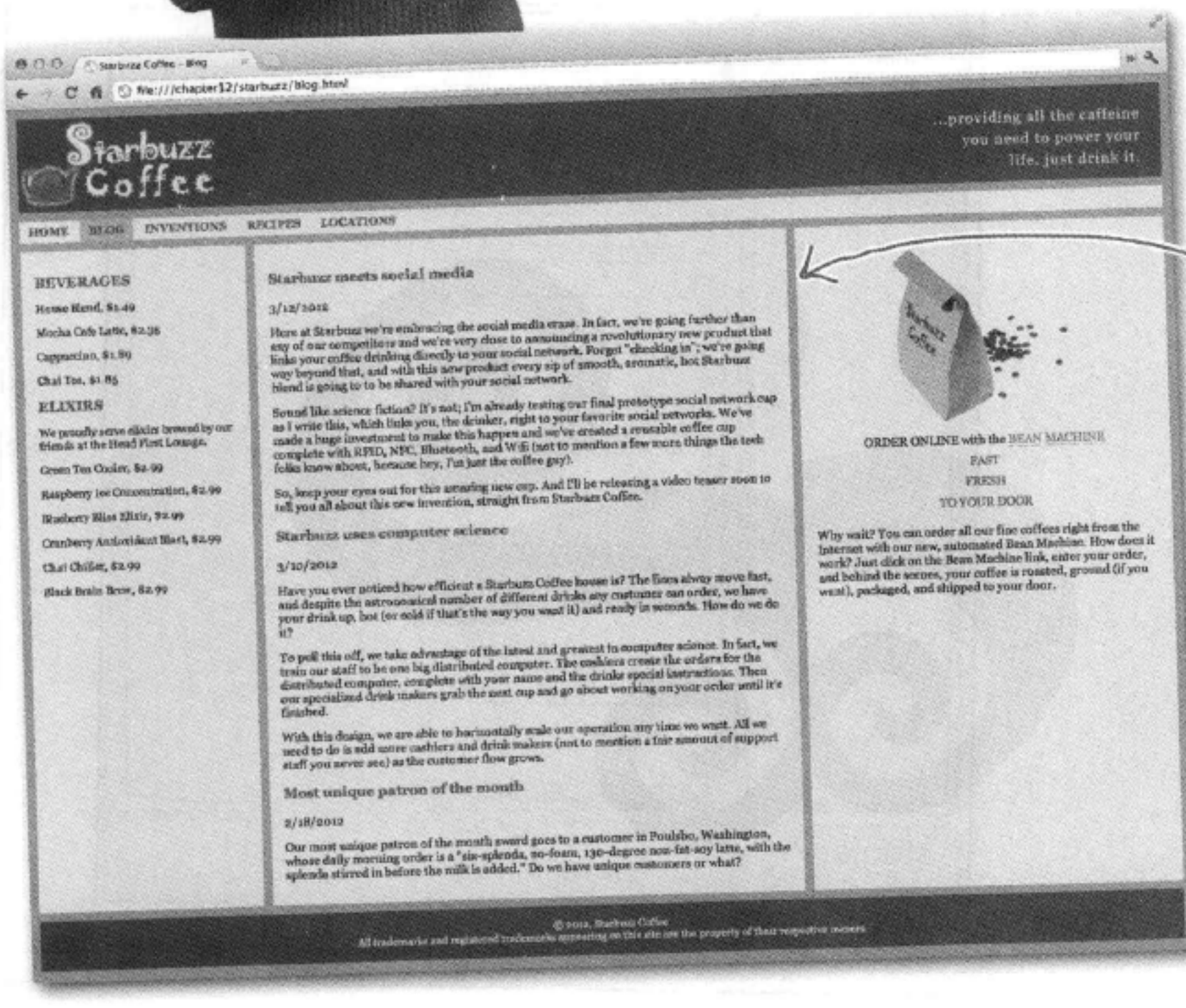
这是现在的页面！看起来完全一样，不过你已经知道现在换成了新的HTML5元素，是不是感觉好多了？

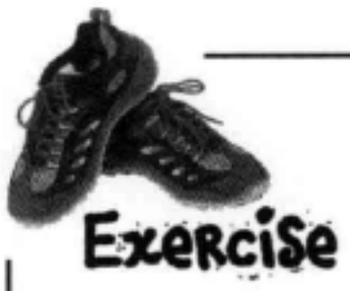
为这两个区块更新的CSS。

```
section#drinks {
  display:          table-cell;
  background-color: #efe5d0;
  width:            20%;
  padding:          15px;
  vertical-align:   top;
}

section#main {
  display:          table-cell;
  background:       #efe5d0 url(images/background.gif) top left;
  font-size:        105%;
  padding:          15px;
  vertical-align:   top;
}
```

CSS可以原封不动！因为我们使用了id，所以现有的规则也能选择这两个元素。不过，我们还在id选择器前增加了标记名，这只是为了更清楚地表示这里在使用<section>。

有意思，刚好你问到这个问题，因为很多新的HTML5元素正好非常适合创建博客。不过，介绍具体的标记之前，下面先来考虑这个博客可能是什么样子，要确保它与当前的Starbuzz设计一致。为此，我们要创建一个新页面，左边还是同样的“drinks”<section>，右边也是一样的<aside>，我们要改变的只是中间的内容，让它作为博客。下面来看看这个设计：



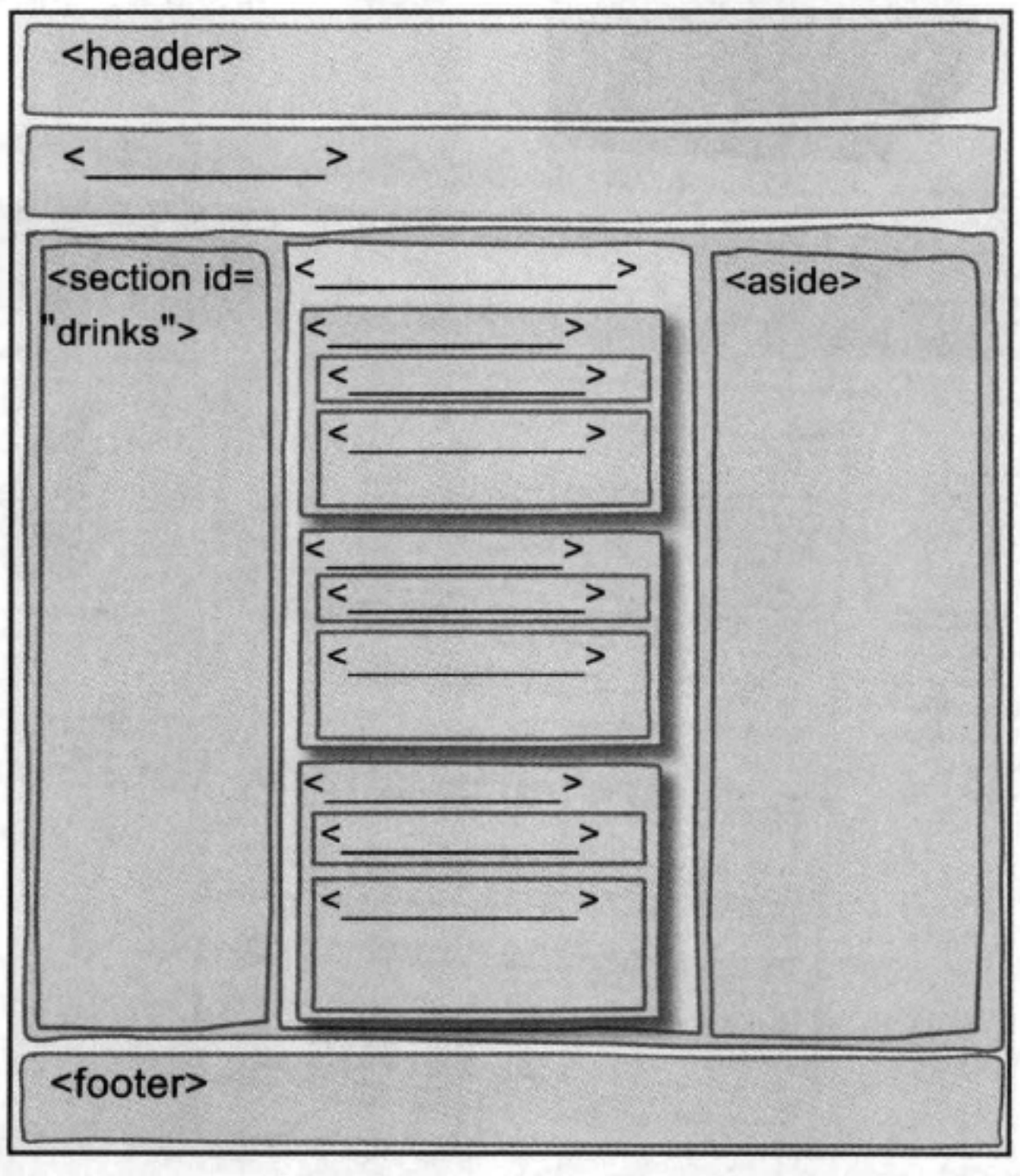


你的任务是选择你认为最适合建立这个新博客的元素。在下面的图表中填空，标出你会选择哪些元素。需要说明，每个博客帖子都有一个标题，另外至少有一个文本段落。

请从下面的列表中选择元素：

```
<header>        <aside>
<footer>        <section>
<article>       <div>
<nav>           <h1>
<time>          <p>
```

新的博客页面。这与主页面很相似，只不过中间部分现在是一些博客帖子，另外在页眉下面有一个导航菜单。



你的任务是选择你认为最适合建立这个新博客的元素。在下面的图表中填空，标出你会选择哪些元素。需要说明，每个博客帖子都有一个标题，另外至少有一个文本段落。

请从下面的列表中选择元素：

<header>
<footer>
<article>
<nav>
<time>
<aside>
<section>

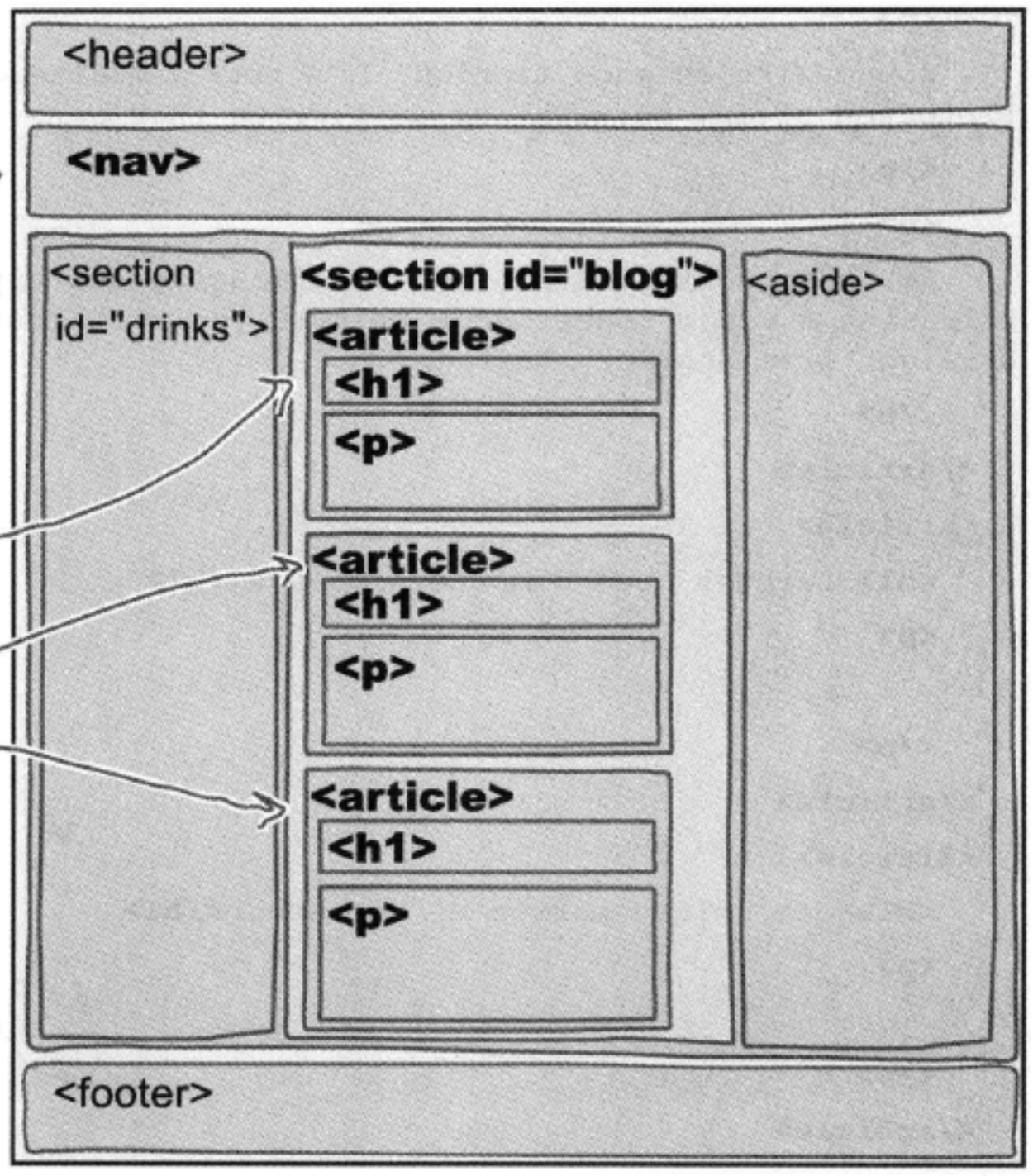
<h1>
<p>

新的博客页面。这与主页面很相似，只不过中间部分现在是一些博客帖子，另外在页眉下面有一个导航菜单。

我们使用<nav>元素作为导航菜单。

将页面的博客“区块”放在一个<section>元素中，因为<section>用于把相关的内容组织在一起。

把各个博客帖子放在各自的<article>元素中，因为每个帖子都是一个独立的元素（也就是说，你可以取出这些文章，而不会影响其余文章的可读性）。



构建Starbuzz博客页面

从上一个练习可以知道，我们要用一个<section>元素实现博客区块（中栏），另外要对各个博客帖子分别使用一个<article>元素。下面先来完成这个工作，稍后再来讨论导航。我们已经为你创建了“blog.html”文件，实际上就是先创建文件“index.html”的一个副本，将其中的“main”<section>替换为一个“blog”<section>。可以从本书下载源码中找到完整的“blog.html”，这里给出其中的一部分：

```
<section id="blog">
  <article>
    <h1>Starbuzz meets social media</h1>
    <p>
      Here at Starbuzz we're embracing the social media craze. In fact,
we're going further than any of our competitors and we're very close……
    </p>
    <p>
      Sound like science fiction? It's not; I'm already testing our final
prototype social network cup as I write this……
    </p>
    <p>
      So, keep your eyes out for this amazing new cup. And I'll be
releasing a video teaser soon to tell you all about this new invention,
straight from Starbuzz Coffee.
    </p>
  </article>
  <article>
    <h1>Starbuzz uses computer science</h1>
    <p>
      ...
    </p>
  </article>
  <article>
    <h1>Most unique patron of the month</h1>
    <p>
      ...
    </p>
  </article>
</section>
```

我们使用一个<section>元素来实现中栏，这类似于index.html文件中的“main”。

这里只显示了部分博客帖子。

每个博客帖子都有自己的<article>元素。

在各个<article>中，使用<h1>作为标题，另外使用<p>作为文本段落。很简单吧！不过，这比一大堆 含义更明确，对不对？

可以从wickedlysmart.com下载“blog.html”文件，其中有完整的博客帖子文本。

建立博客页面的CSS

你可能已经注意到，“index.html”文件和“blog.html”文件都链接到同一个CSS文件“starbuzz.css”。下面简单看一下“blog.html”：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Starbuzz Coffee - Blog</title>
    <link rel="stylesheet" type="text/css" href="starbuzz.css">
  </head>
  ...
```

这是指向CSS的链接……

……既然在建博客，这里换一个页面标题。

目前，我们还没有为id为“blog”的新区块增加任何CSS，所以现在就来做这个工作。很清楚，我们希望“blog”<section>的样式与主页上的“main”<section>类似，所以可以直接重用同样的规则，把blog区块的规则增加到现有的main区块规则中，如下：

```
section#main, section#blog {
  display:        table-cell;
  background:     #efe5d0 url(images/background.gif) top left;
  font-size:      105%;
  padding:        15px;
  vertical-align: top;
}
```

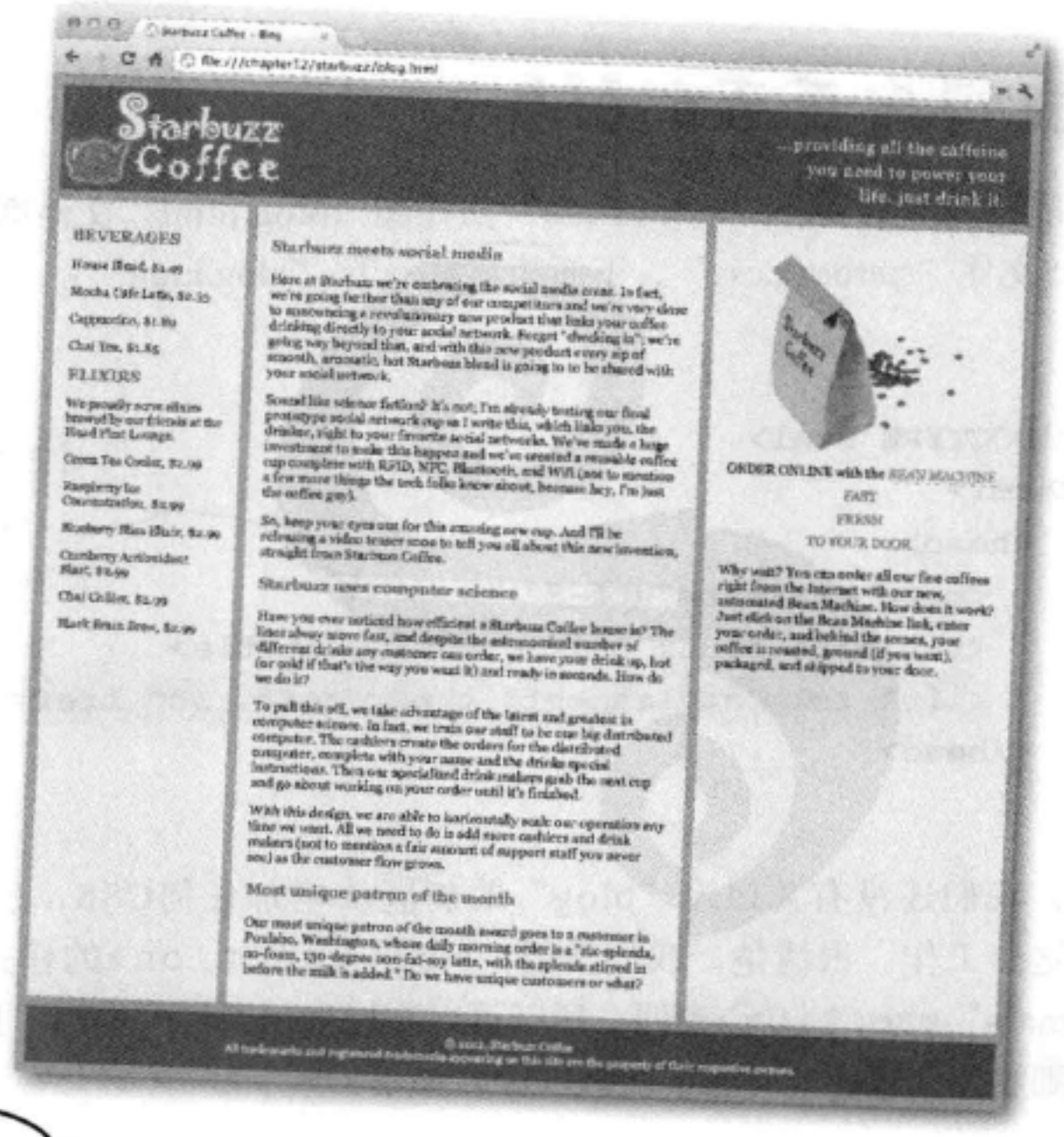
通过使用两个选择器（用逗号分隔），可以对这两个<section>元素使用相同的规则。这说明，所选择的两个元素都要应用这些属性。

尽管这两个元素（“main”<section>和“blog”<section>）在不同的页面上，也可以这样选择，因为这两个页面都链接到相同的CSS文件。

就这么简单！页面上“blog”<section>需要的所有其他样式都已经在CSS中了，另外我们不需要为<article>增加任何特殊的样式。所以现在该……

测试博客页面

我们已经创建了一个新的博客页面，而且对页面做了一些简单的调整（增加了<section>和<article>元素），下面保存页面，再在浏览器中加载这个页面。



可以看到，<section>、<article>和<aside>等元素都像 一样，有类似的默认样式。也就是说，它们本身没有多少样式！不过，这些元素确实可以为页面中的内容增加含义信息。

区块（section）和文章（article）有什么区别？

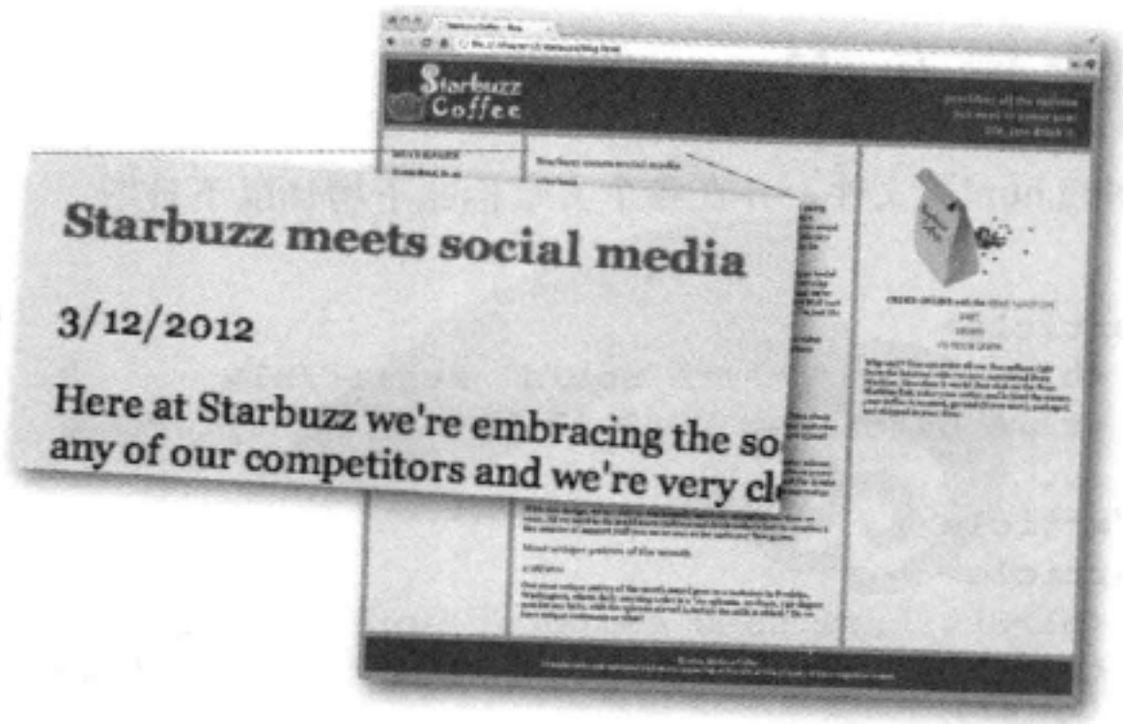
是的，这确实让人很困惑。可以明确告诉你，关于这个问题并没有透彻明了的答案。实际上，使用<article>和<section>的方法有很多。不过，通常可以这样来考虑：使用<section>可以把相关的内容分组在一起，另一方面，要用<article>包含独立的内容，如一个新闻报道、一个博客帖子或者一个简短的报告。

在Starbuzz页面中，每一栏都包含相关的内容，所以我们把各栏作为页面的一个区块。另外各个博客帖子要作为文章，因为它们是独立的（甚至可以想象取出其中一个文章，把它转发到另一个网站或博客上）。

你的标准可能有所不同，不过一般来讲，要组织相关的内容，就要使用<section>，而对于独立的内容，则使用<article>。如果你需要把感觉并不相关的内容组织在一起，往往可以用 作为后备。

还需要为博客增加一个日期……

注意到了吗？在我们的博客设计中，还为每个博客帖子增加了一个日期。在HTML5之前，日期往往可以采用自由的方式创建，增加日期时可以根本不做标记，或者使用一个甚至<p>来标记。不过现在我们有了一个专门完成这个任务的元素：<time>元素。



<time>元素的简要指南

下面再来仔细分析<time>元素。它有一个重要的属性：datetime，这个元素对于datetime属性中使用的值有些挑剔，所以很有必要了解一些细节问题。

如果元素内容没有采用官方Internet日期/时间格式来写，就必须有datetime属性。

如果使用datetime属性来指定一个日期和/或一个时间，可以写你希望的任何元素内容。通常，这可能是某个与日期或时间相关的文本，如“February 18, 2012”或者甚至可以是“yesterday”或“now”。

```
<time datetime="2012-02-18">2/18/2012</time>
```

这就是指定日期的官方Internet格式，包含日、月和年。

```
2012-02
```
可以只指定年和月，或者只指定年。

```
2012
```

这里是使用官方格式表示日期和时间的另外一些方法。

```
2012-02-18 09:00
```
可以按24小时制增加一个时间。

```
2012-02-18 18:00
05:00
```
可以只指定一个时间。

```
2012-02-18 05:00Z
```
如果在日期和时间后面有一个“Z”，这表示UTC时间。(UTC = GMT)