重视大脑的学习指南

第二版

全面更新，涵盖HTML5

Head First HTML与CSS

只用一章开启你的Web职业生涯

当心常见的HTML和CSS陷阱和问题

100个谜题和练习帮你记住

创建基于标准的Web页面的初学者指南

发现为什么你的朋友对样式的理解可能是错的

避免验证错误的尴尬

Elisabeth Robson & Eric Freeman 著
徐阳 丁小峰 等译

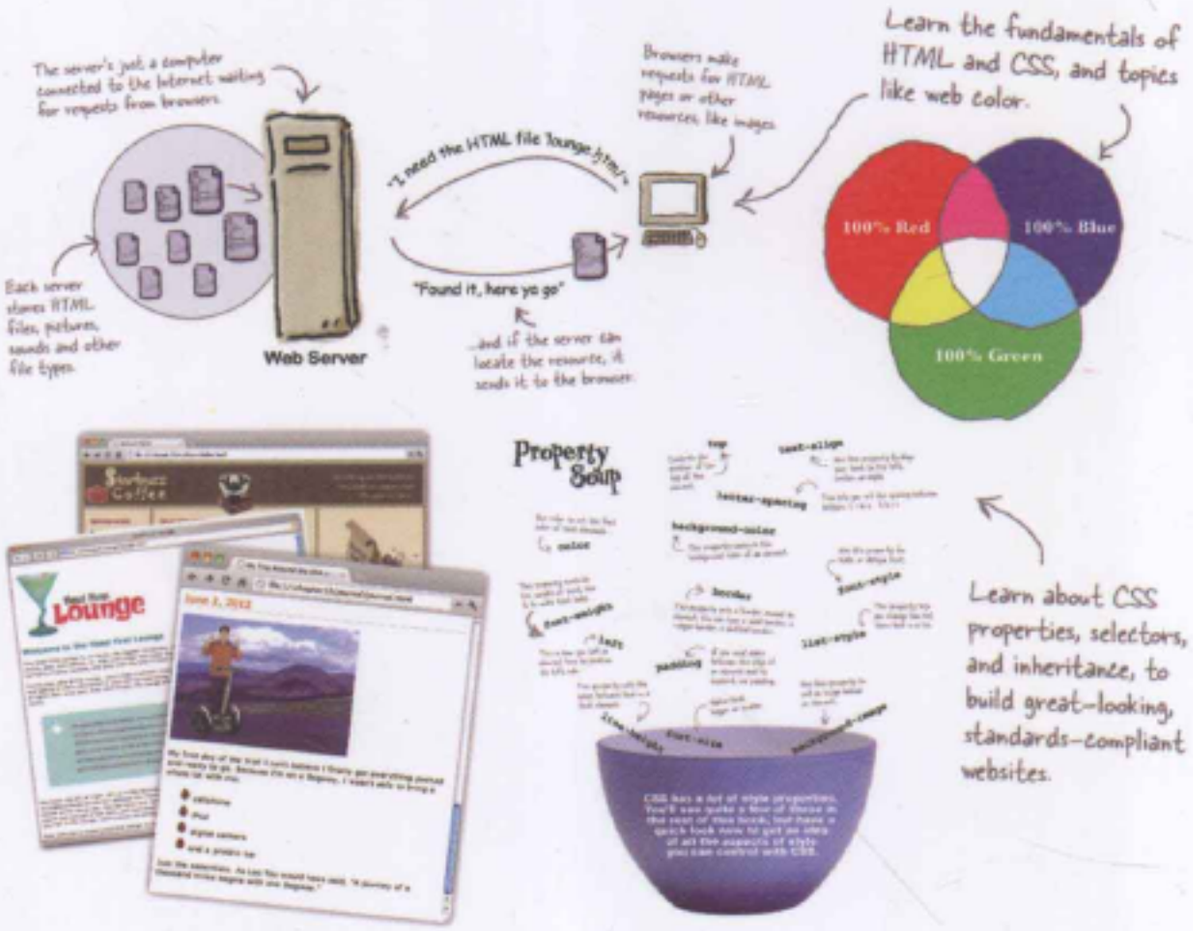
O'REILLY® 中国电力出版社

Head First HTML与CSS

Web Design & Development/HTML

从这本书能学到什么？

是不是已经厌倦了那些深奥的HTML书？你可能在抱怨，只有成为专家之后才能读懂那些书。那么，找一本新修订的《Head First HTML与CSS（第二版）》吧，来真正学习HTML。你可能希望学会HTML和CSS来创建你想要的Web页面，从而能与朋友、家人、粉丝和狂热的顾客更有效地交流。你还希望使用最新的HTML5标准，能够保证随时间维护和扩展你的Web页面，使它们在所有浏览器和移动设备中都能正常工作。



为什么这本书如此与众不同？

在这本书中，你会学到创建Web页面的秘密，最重要的是，你会用一种不至于让你昏昏欲睡的方式来学习。如果你读过Head First书，应该知道这会是一本怎样的书：它会采用一种专门为你的大脑而设计的丰富格式娓娓道来。通过使用认知科学和学习理论的最新研究成果，这本书会向你的大脑输入HTML和CSS，并让它牢牢记住。

O'REILLY®
oreilly.com.cn
headfirstlabs.com

O'Reilly Media, Inc.授权中国电力出版社出版

“《Head First HTML与CSS（第二版）》运用新颖的手法，介绍了Web页面标记和表现方面的前沿方法。它准确地把握了读者的困惑，并及时解决。丰富的图片和渐进的过程模拟了学习这个技术的最佳途径：你可以做一点改动，然后在浏览器上查看，了解每一个新特性的具体含义。”

——**Danny Goodman**，《Dynamic HTML:The Definitive Guide》的作者

“这本书介绍的都是专业人士才了解的信息，不过采用了一种循循善诱、幽默诙谐的方式来讲解，绝对不会让你觉得这些东西学不会或者让你无所适从。”

——**Christopher Schmitt**，《CSS Cookbook》与《Professional CSS》的作者

ISBN 978-7-5123-4477-8
9 787512 344778 >
定价：98.00元

Head First HTML与CSS

（第二版）

Elisabeth Robson
Eric Freeman 著
徐阳 丁小峰 等译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'Reilly Media, Inc. 授权中国电力出版社出版

中国电力出版社

图书在版编目（CIP）数据

Head First HTML与CSS：第2版 /（美）罗布森（Robson, E.），（美）弗里曼（Freeman, E.）著；徐阳等译. —北京：中国电力出版社，2013.9

书名原文：Head First HTML and CSS, Second Edition

ISBN 978-7-5123-4477-8

I. ①H… II.①罗… ②弗… ③徐… III. ①超文本标记语言－程序设计 ②网页制作工具 IV. ①TP312 ②TP393.092

中国版本图书馆CIP数据核字（2013）第108333号

北京市版权局著作权合同登记

图字：01-2013-3086号

书　　名/　Head First HTML与CSS（第二版）
书　　号/　ISBN 978-7-5123-4477-8
责任编辑/　刘炽
封面设计/　Karen Montgomery，张健
出版发行/　中国电力出版社
地　　址/　北京市东城区北京站西街19号（邮政编码100005）
印　　刷/　航远印刷有限公司
开　　本/　850毫米×980毫米　16开本　47.625印张　1021千字
版　　次/　2013年9月第1版　2013年9月北京第1次印刷
印　　数/　0001－3000册
定　　价/　98.00元（册）

浏览器战争是什么？你会在第6章找到答案。

致W3C，感谢你们结束了浏览器战争，另外感谢你们的睿智，
是你们提倡结构（HTML）与表现（CSS）分离……

还要感谢你们让HTML和CSS这么复杂，正因如此，人们才需要
这样一本书来学习。

《Head First HTML 与 CSS》的作者

Eric Freeman

Elisabeth Robson

按照“Head First”系列合作者Kathy Sierra的说法，Eric是“少有的奇才之一，不仅语言流畅，实践经验丰富，在很多领域都表现非凡，他是嬉皮士高手、副总裁、工程师，而且是名副其实的智多星。”

在专业领域，Eric最近刚刚离开任职近十年的一家媒体公司，他在迪斯尼公司担任Disney Online & Disney.com的CTO。Eric现在把时间全部投入到WickedlySmart，这是他与Elisabeth共同创建的一家公司。

经过培训，Eric已经成为一位计算机科学家，在耶鲁大学攻读博士学位期间曾与业界杰出人物David Gelernter同窗。他的论文被认为对寻找桌面隐喻*的替代品有着深远影响，这也是活动流的首个实现（活动流是他与Gelernter博士提出的一个概念）。

在闲暇时间，Eric对音乐深深着迷；在iPhone App Store上可以找到Eric最近与音乐“先锋”Steve Roach合作的一个项目，名为Immersion Station。

Eric与妻子和小女儿居住在华盛顿州双桥岛。他女儿是Eric工作室的常客，她特别喜欢打开他的合成器和音效开关。Eric对儿童教育和营养也很关注，总在想方设法做出改善。

可以给Eric写邮件（eric@wickedlysmart.com），或者访问他的网站（http://ericfreeman.com）。

* 桌面隐喻是在用户界面中用人们熟悉的桌面上的物品来清楚地表现计算机可处理的能力。

Elisabeth 是一位软件工程师、作家和培训师。从她作为耶鲁大学学生之日起就一直热衷于技术，她在耶鲁大学获得了计算机科学硕士学位，并设计了一个并发的可视化编程语言和软件体系结构。

Elisabeth从早期就一直从事Internet的工作，她合作创建了颇有声誉的网站：The Ada Project，这是最早帮助计算机科学领域的女性在线找工作和寻求指导信息的网站之一。

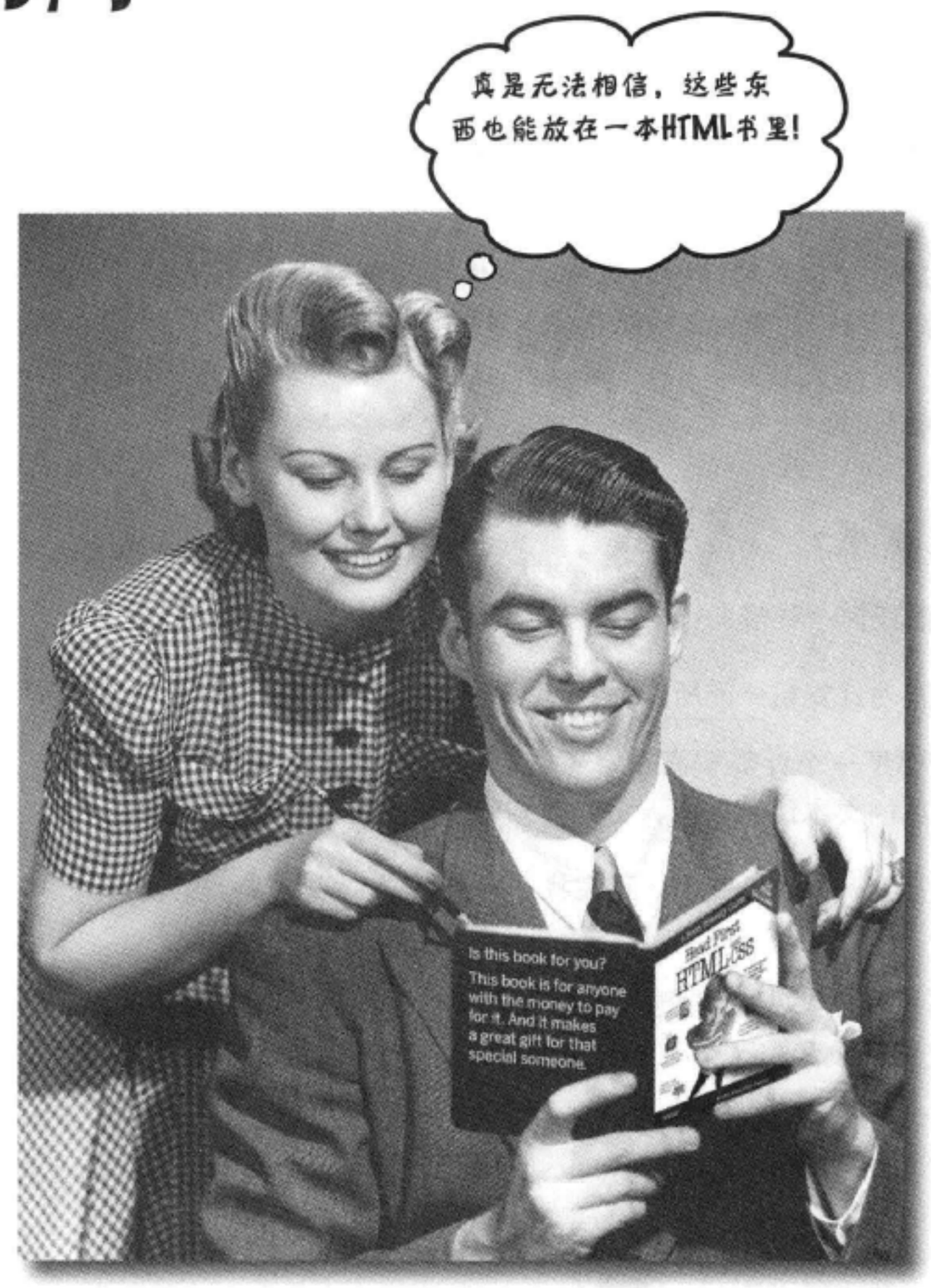
她目前是WickedlySmart的合作创始人，这是一个关注Web技术的在线教育项目，在这里她完成了有关的图书、文章、视频等。在此之前，作为O’Reilly Media的特殊项目主任（Director of Special Projects），Elisabeth曾经在各种技术专题发布过个人研讨文档和在线课程，这使她对创建学习体验来帮助人们理解技术越来越充满热情。在O’Reilly工作之前，Elisabeth主要在迪斯尼公司播洒她的仙女魔法粉，在那里，她带领数字媒体研发力量开展研究工作。

如果没有坐在计算机前，你就会发现Elisabeth总是带着她的相机在户外踏青、骑单车或者划皮艇，也可能在烹调素食大餐。

可以给她发Email（beth@wickedlysmart.com）或者访问她的博客（http://elisabethrobson.com）。

如何使用这本书

引子



有个问题真是听得我们耳朵都磨出茧了："你们到底为什么要把这样一些东西放在一本HTML书里呢？"现在就来回答这个问题。

谁适合看这本书？

如果对下面的所有问题都能肯定地回答“是”：

如果你的计算机是最近十年内的产品，答案就是“是”。

① 你有一台安装了**Web浏览器**和**文本编辑器**的计算机吗？

② 你是不是想**学习**、**理解**、**记住**并且用最好的技术和最新的标准来创建Web应用？

③ 你是不是更喜欢一种轻松的氛围，就像**在晚餐餐桌上交谈**一样，而不愿意被动地**听枯燥乏味的技术报告**？

这正是你想要的书。

谁可能不适合看这本书？

如果满足下面任何一种情况：

① 你是不是对计算机**一无所知**？

不要求你是一个计算机高手，不过起码应该了解文件夹和文件、简单的文本编辑应用，另外要知道如何使用Web浏览器。

② 你是不是在开发Web应用，正在找一本**参考书**？

③ 你是不是**对新鲜事物都畏首畏尾**？只喜欢简单的样式，而不敢尝试把条纹和格子混在一起看看？你是不是觉得，如果加入了拟人化的HTML标记，这样一本书肯定不是一本正经八百的技术书？

那么，这本书将不适合你。

[来自市场的声音：只要有信用卡，都可以拥有这本书。]

我们知道你在想什么。

“这算一本正经八百的书吗？”

“这些图做什么用？”

“我真能这样学吗？”

我们也知道你的大脑正在想什么。

你的大脑总是渴求一些新奇的东西。它一直在搜寻、审视、期待着不寻常的事情发生。大脑的构造就是如此，正是这一点才让我们不至于墨守成规，能够与时俱进。

如今，真正面对老虎、被老虎吃掉的可能性很小，不过你的大脑还是在时刻提防着。你只是不知道而已。

我们每天都会遇到许多按部就班的事情，这些事情很普通，对于这样一些例行的事情或者平常的东西，你的大脑又是怎么处理的呢？它的做法很简单，就是不让这些平常的东西妨碍大脑真正的工作。那么什么是大脑真正的工作呢？这就是记住那些确实重要的事情。它不会费心地去记乏味的东西，就好像大脑里有一个筛子，这个筛子会筛掉“显然不重要”的东西，如果遇到的事情枯燥乏味，这些东西就无法通过这个筛子。

那么你的大脑怎么知道到底哪些东西重要呢？打个比方，假如你某一天外出旅行，突然一只大老虎跳到你面前，此时此刻，你的大脑还有身体会做何反应？

神经元会“点火”，情绪爆发，释放出一些化学物质。

好了，这样你的大脑就会知道……

这肯定很重要！可不能忘记了！

不过，假如你正待在家里或者坐在图书馆里，这里很安全、很舒适，肯定没有老虎。你正在刻苦学习，准备应付考试。也可能想学一些比较难的技术，你的老板认为掌握这种技术需要一周时间，最多不超过十天。

这就存在一个问题。你的大脑很想给你帮忙。它会努力地把这些显然不太重要的内容赶走，保证这些东西不去侵占本不算充足的脑力资源。这些资源最好还是用来记住那些确实重要的事情，比如大老虎，再比如遭遇火灾等。就像你的大脑会让你记住绝对不能再穿着短裤去滑雪。

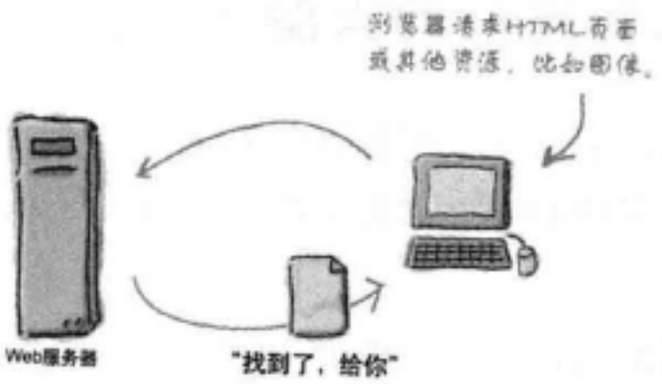
没有一种简单的办法来告诉大脑：“嘿，大脑，真是谢谢你了，不过不管这本书多没意思，也不管现在我对它多么无动于衷，但我确实希望你能把这些东西记下来。”

我们认为“Head First”的读者就是要学习的人。

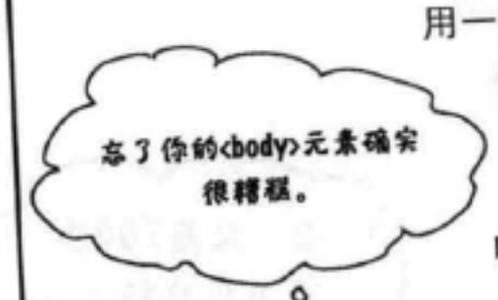
那么，怎么学习呢？首先必须获得知识，然后保证自己确实不会忘记。这可不是填鸭式的硬塞。根据认知科学、神经生物学和教育心理学的最新研究结果，学习的途径相当丰富，绝非只是通过书本上的文字。我们很清楚怎么让你的大脑兴奋起来。

下面是一些Head First学习原则：

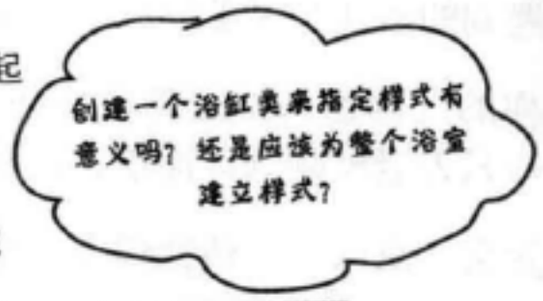
看得到。与单纯的文字相比，图片更能让人记得住，通过图片，学习效率会更高（对于记忆和传递型的学习，甚至能有多达89%的效率提升）。而且图片更能让人看懂。以往总是把图片放在一页的最下面，甚至放在另外的一页上，与此不同，如果把文字放在与之相关的图片内部，或者在图片的周围写上相关文字，学习者的能力就能得到多至两倍的提高，从而能更好地解决有关问题。



采用一种针对个人的交谈式风格。最新的研究表明，如果学习过程中采用一种第一人称的交谈方式直接向读者讲述有关内容，而不是用一种干巴巴的语调介绍，则学生在学习之后的考试中成绩会提高40%。正确的做法是讲故事，而不是做报告。要用通俗的语言。另外不要太严肃。如果你面对着这样两个人，一个是你在餐会上结识的很有意思的朋友，另一个人学究气十足，喋喋不休地对你说教，那么在这两个人中，你会更注意哪一个呢？



让学习的人想得更深。换句话说，除非你很积极地让神经元活动起来，否则你的头脑里什么也不会发生。必须引起读者的好奇，促进、要求，并鼓励读者去解决问题、得出结论、产生新的知识。为此，需要发出挑战，留下练习题和拓宽思路的问题，并要求读者完成一些实践活动，让左右脑都开动起来，而且要利用到多种思维。

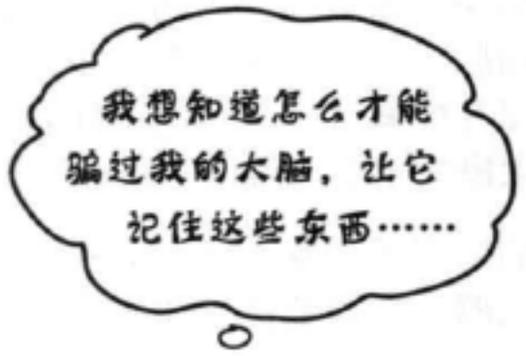


引起读者的注意，而且要让他一直保持注意。我们可能都有过这样的体验，“我真的想把这个学会，不过看过一页后实在是让我昏昏欲睡。”你的大脑注意的是那些不一般、有意思、有些奇怪、抢眼的、意料之外的东西。学习一项有难度的新技术并不一定枯燥。如果学习过程不乏味，你的大脑很快就能学会。

影响读者的情绪。现在我们知道了，记忆能力很大程度上取决于所记的内容对我们的情绪有怎样的影响。如果是你关心的东西，就肯定记得住。如果让你感受到了什么，这些东西就会留在你的脑海中。不过，我们所说的可不是什么关于男孩与狗的伤心故事。这里所说的情绪是惊讶、好奇、觉得有趣、想知道“什么……”还有就是一种自豪感，如果你解决了一个难题，学会了所有人都觉得很难的东西，或者发现你了解的一些知识竟是那些自以为无所不能的傲慢家伙所不知道的，此时就会有一种自豪感油然而生。

元认知：有关思考的思考

如果你真的想学，而且想学得更快、更深，就应该注意你怎样才会专注起来，考虑自己是怎样思考的，并了解你的学习方法。



我们中间大多数人长这么大可能都没有上过有关元认知或学习理论的课程。我们想学习，但是很少有人教我们怎么来学习。

不过,这里可以做一个假设，如果你手上有这本书，你想学习项目管理，而且可能不想花太多时间。如果你想把这本书中读到的知识真正用起来，就需要记住你读到的所有内容。为此，必须理解这些内容。要想最大程度地利用这本书或其他任何一本书，或者掌握学习经验，就要让你的大脑负起责来，要求它记住这些内容。

怎么做到呢？技巧就在于要让你的大脑认为你学习的新东西确实很重要，对你的生活有很大影响。就像老虎出现在面前一样。如若不然，你将陷入旷日持久的拉锯战中，虽然你很想记住所学的新内容，但是你的大脑却会竭尽全力地把它们拒之门外。

那么究竟怎样才能让大脑把HTML&CSS看作是一只饥饿的老虎呢？

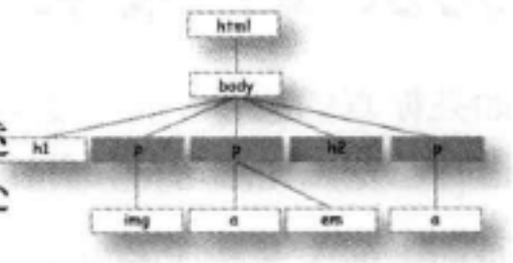
这有两条路，一条比较慢，很乏味。另一条路不仅更快，还更有效。慢方法就是大量地重复。你肯定知道，如果反反复复地看到同一个东西，即便再没有意思，你也能学会并记住。如果做了足够的重复，你的大脑就会说，“尽管看上去这对他来说好像不重要，不过，既然他这样一而再、再而三地看同一个东西，所以我假定这应该是重要的。”

更快的方法是**尽一切可能让大脑活动起来**，特别是开动大脑来完成不同类型的活动。如何做到这一点呢？上一页列出的学习原则正是一些主要的可取做法，而且经证实，它们确实有助于让你的大脑全力以赴。例如，研究表明，把文字放在所描述图片的中间（而不是放在这一页的别处，比如作为标题，或者放在正文中），这样会让你的大脑更多地考虑这些文字与图片之间有什么关系，而这就会让更多的神经元“点火”。让更多的神经元“点火”=你的大脑更有可能认为这些内容值得关注，而且很可能需要记下来。

交谈式风格也很有帮助，当人们意识到自己在与“别人”交谈时，往往会更专心，这是因为他们总想跟上谈话的思路，并能做出适当的发言。让人惊奇的是，大脑并不关心“交谈”的对象究竟是谁，即使你只是与一本书“交谈”，它也不会在乎！另一方面，如果写作风格很正统、干巴巴的，你的大脑就会觉得，这就像坐在一群人当中被动地听人做报告一样，很没意思，所以不必在意对方说的是什么，甚至可以打瞌睡。

不过，图片和交谈风格还只是开始而已，能做的还有很多。

我们是这么做的：



我们用了很多图，因为你的大脑更能接受看得见的东西，而不是纯文字。对你的大脑来说，一图胜千言。如果既有文字又有图片，我们就会把文字放在图片当中，因为文字处在所描述的图片中间时，大脑的工作效率更高，倘若把这些描述文字作为标题，或者“淹没”在别处的大段文字中，就达不到这种效果了。

我们采用了重复手法，会用不同方式，采用不同类型的媒体，运用多种思维手段来介绍同一个东西，目的是让有关内容更有可能储存在你的大脑中，而且在大脑中多个区域都有容身之地。

扮演浏览器

我们会用你想不到的方式运用概念和图片，因为你的大脑喜欢新鲜玩艺。在提供图和思想时，至少会含着一些情绪因素，因为如果能产生情绪反应，你的大脑就会投入更大的注意。而这会让你感觉到这些东西更有可能被记住，其实这种感觉可能只是有点幽默，让人奇怪或者比较感兴趣而已。

我们采用了一种针对个人的交谈式风格，因为当你的大脑认为你在参与一个会谈，而不是被动地听一场演示汇报时，它就会更加关注。即使你实际上在读一本书，也就是说在与书“交谈”，而不是真正与人交谈，但这对你的大脑来说并没有什么分别。

BULLET POINTS

在这本书里，我们加入了100多个实践活动，因为与单纯的阅读相比，如果能实际做点什么，你的大脑就会更乐于学习，更愿意去记。这些练习都是我们精心设计的，有一定的难度，但是确实能做出来，因为这是大多数人所希望的。

我们采用了多种学习模式，因为尽管你可能想循序渐进地学习，但是其他人可能希望先对整体有一个全面的认识，另外可能还有人只是想看一个例子。不过，不管你想怎么学，要是同样的内容能以多种方式来表述，这对每一个人都会有好处。

猜谜游戏

这里的内容不只是单单涉及左脑，也不只是让右脑有所动作，我们会让你的左右脑都开动起来，因为你的大脑参与得越多，你就越有可能学会并记住，而且能更长时间地保持注意力。如果只有一半大脑在工作，通常意味着另一半有机会休息，这样你就能更有效率地学习更长时间。

我们会讲故事，留练习，从多种不同的角度来看同一个问题，这是因为，如果要求大脑做一些评价和判断，它就能更深入地学习。

我们会给出一些练习，还会问一些问题，这些问题往往没有直截了当的答案，通过克服这些挑战，你就能学得更好，因为让大脑真正做点什么的话，它就更能学会并记住。想想吧，如果只是在体育馆里看着别人流汗，则对于保持你自己的体形肯定不会有什么帮助，正所谓临渊羡鱼，不如退而结网。不过另一方面，我们会竭尽所能不让你钻牛角尖，把劲用错了地方，而是能把功夫用在点子上。也就是说，你不会为弄明白一个难懂的例子而耽搁，也不会花太多时间去弄明白一段艰涩难懂而且通篇行话的文字，我们的描述也不会太过简洁而让人无从下手。

我们用了拟人手法。在故事中，在例子中，还有在图中，你都会看到人的出现，这是因为你本身是一个人，不错，这就是原因。如果和人打交道，相对于某件东西而言，你的大脑就会更为关注。

这里使用了一种80/20方法。我们相信你会成为一个一流的Web开发人员，这绝不会是你唯一的参考书。所以我们不打算面面俱到，涵盖所有内容，这里只提供你真正需要的东西。

把这一页撕下来，贴到你的冰箱上。

可以用下面的方法让你的大脑就范

好了，我们该做的已经做了，剩下的就要看你自己的了。下面的提示可以作为一个起点，听一听你的大脑是怎么说的，弄清楚对你来说哪些做法可行，哪些做法不能奏效。要做些新的尝试。

① 慢一点。你理解的越多，需要记的就越少。

不要光是看看就行了。停下来，好好想一想。书中提出问题的时候，你不要直接去翻答案。可以假想真的有人在问你这个问题。你让大脑想得越深入，就越有可能学会并记住它。

② 做练习，自己记笔记。

我们留了练习，但是如果这些练习的解答也由我们一手包办，那和有人替你参加考试有什么分别？不要只是坐在那里看着练习发呆。拿出笔来，写一写画一画。大量研究都证实，学习过程中如果能实际动动手，将改善你的学习。

③ 阅读“没有傻问题”部分。

顾名思义。这些问题不是可有可无的旁注，它们绝对是核心内容的一部分！千万不要跳过去不看。

④ 上床睡觉之前不要再看别的书，至少不要看其他有难度的东西。

学习中有一部分是在你合上书之后完成的（特别是，要把学到的知识长久地记住，这往往无法在看书的过程中做到）。你的大脑也需要有自己的时间，这样才能再做一些处理。如果在这段处理时间内你又往大脑里灌输了新的知识，那么你刚才学的一些东西就会丢掉。

⑤ 要喝水，而且要多喝点水。

能提供充足的液体，你的大脑才能有最佳表现。如果缺水（可能在你感觉到口渴之前就已经缺水了），学习能力就会下降。

⑥ 讲出来，而且要大声讲出来。

说话可以刺激大脑的另一部分。如果你想看懂什么，或者想更牢地记住它，就要大声地说出来。更好的办法是，大声地解释给别人听。这样你会学得更快，而且可能会有新的发现，这是以前光看不说的时候所没有的。

⑦ 听听你的大脑怎么说。

注意一下你的大脑是不是负担太重了。如果发现自己开始浮光掠影地翻看，或者刚看的东西就忘记了，这说明你该休息一会了。达到某个临界点时，如果还是一味地向大脑里塞，这对于加快学习速度根本没有帮助，甚至还可能影响正常的学习进程。

⑧ 要有点感觉。

你的大脑需要知道这是很重要的东西。要真正融入到书中的故事里。为书里的照片加上你自己的图题。你可能觉得一个笑话很蹩脚，不太让人满意，但这总比根本无动于衷要好。

⑨ 真正做些工作!

把这些知识应用到你的日常工作中去，利用你所学的方法完成项目决策。具体做些工作，你会得到书中练习和活动以外的经验。所需要的只是一支笔和一个要解决的问题——一个使用HTML和CSS可以妥善解决的问题。

重要说明

要把这看做是一个学习过程，而不要简单地把它看成是一本参考书。我们在安排内容的时候有意做了一些删减，只要是对有关内容的学习有妨碍，我们都毫不留情地把这些部分一律删掉。另外，第一次看这本书的时候，要从第一页从头看起，因为书中后面的部分会假定你已经看过而且学会了前面的内容。

我们先从基本HTML开始讲起，然后是基于标准的HTML5。

要编写基于标准的HTML，有很多技术细节需要了解，但是这些细节对于你学习基本HTML并没有帮助。我们的方法是先让你掌握HTML的基本概念（而不用担心那些细节），等你有了扎实的HTML基础后，再教你编写符合标准的HTML（HTML的最新版本是HTML5）。这有一个额外的好处：学习完基础之后，那些技术细节会更有意义。

开始使用CSS时，编写符合标准的HTML也很重要，所以我们强调在用CSS做具体工作之前，先要建立基于标准的HTML。

我们没有涵盖每一个HTML元素或属性，或者已创建的所有CSS属性。

有太多的HTML元素、太多的属性，还有太多的CSS属性。当然，它们都很有意思，不过我们的目标是写一本篇幅适量的书，起码重量不能超过读这本书的读者，所以这里没有面面俱到。我们的重点是核心HTML元素和对初学者最重要的CSS属性，确保你能真正地、深入地、扎实地了解如何使用以及何时使用这些内容。不管怎样，一旦读完这本《Head First HTML 与 CSS》，你就可以选择任何参考书，迅速补上我们没有谈到的元素和属性。

这本书提倡页面结构与页面表现的清晰分离。

如今，好的Web页面都使用HTML建立内容的结构，使用CSS提供样式和表现。20世纪90年代的Web页面通常使用另外一种模型，页面中HTML不仅用来指定结构，同时还用来建立样式。这本书会教你使用HTML建立结构，而用CSS指定样式，我们认为没有理由再教你那些过时的坏习惯。

建议你学习这本书时最好使用多种浏览器。

尽管我们会教你编写基于标准的HTML和CSS，但是你可能还会（而且有可能经常）发现不同的Web浏览器显示页面时会有一些细微的差别。所以，我们建议你至少选择两种现代浏览器，用它们分别测试你的页面。这会让你了解到不同浏览器之间的差异，另外还能获得一些经验，知道如何创建适用于多种不同浏览器的页面。

我们通常使用标记名表示元素名。

我们可能并不会说“a元素”或“‘a’元素”，而是使用一个标记名，如“<a>元素”。从理论上讲这可能并不对（因为<a>只是一个开始标记，而不是一个完整的元素），但是这样更有可读性，另外我们通常会在名字后面加上“元素”两个字来避免混淆。

书里的实践活动绝不是可有可无。

这里的练习和实践活动不是可有可无的装饰和摆设,它们也是这本书核心内容的一部分。其中有些练习和活动有助于记忆，有些能够帮助你理解，还有一些对于如何应用所学的知识很有帮助。千万不要把这些练习跳过不做。甚至填字游戏也很重要，它们可以帮助你将有关概念植入你的大脑。不过更重要的是，它们可以让你的大脑有机会在不同的上下文考虑这些词汇和概念。

我们有意安排了许多重复，这些重复非常重要。

Head First系列图书有一个与众不同的地方，这就是，我们希望你确确实实地掌握这些知识，另外希望在学完这本书之后你能记住学过了什么。大多数参考书都不太重视重复和回顾，但是由于这是一本有关学习的书，你会看到一些概念一而再、再而三地出现。

例子尽可能简洁。

有读者告诉我们，如果查了200行代码才能找到要理解的那两行代码，这是很让人郁闷的。这本书里大多数例子往往都开门见山，作为上下文的代码会尽可能的少，这样你就能一目了然地看到哪些东西是需要你学习的。别指望所有示例都非常健壮，要知道这里的代码甚至是不完整的。这些例子特意写得很简单，以方便你学习，而且它们的功能不一定完备。

所有示例代码和应用都已经在网上公布，以便你下载。所有代码都可以在`http://wickedlysmart.com/hfhtmlcss/`找到。

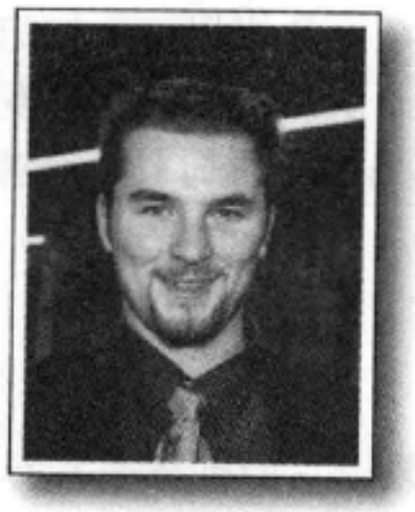
“Brain Power（头脑风暴）”练习没有答案。

有一些头脑风暴练习根本没有所谓正确的答案，另外一些练习中，头脑风暴实践活动的一部分学习过程就是让你确定：你的答案是否正确，以及在何种情况下正确。有些头脑风暴练习中，你会得到一些提示，为你指明正确的方向。

技术审校人员（第一版）


Louise Barr


Joe Konior


Valentin Crettaz


Corey McGlone


Barney Marispini


Pauline McNamara

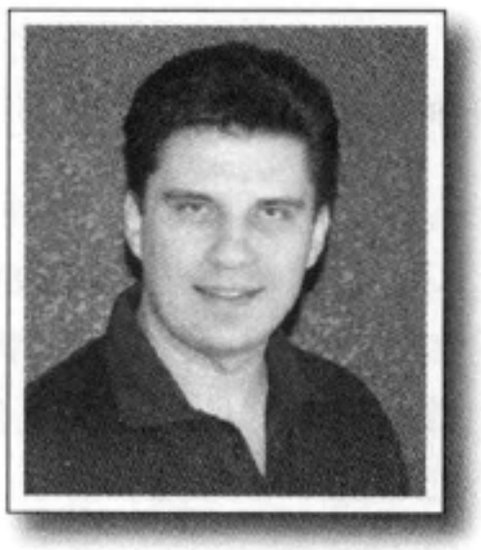
埃菲尔铁塔


Johannes de Jong

直言快语的极限审校团队队长


Ike Van Atta

Marcus Green


David O' Meara

Pauline获得“最严厉审校”奖

致我们的审校团队：

非常感谢我们的审校团队。Johannes de Jong组织并领导了整个项目，他作为“系列之父”，着力保证所有工作顺利进行。Pauline McNamara，整个项目的“协调人”，负责联系各个方面，他也是第一个指出我们的例子有点过于“花哨”。整个团队的工作充分表明我们是多么需要他们的技术经验以及对细节的关注。Valentin Crettaz、Barney Marispini、Marcus Green、Ike Van Atta、David O'Meara、Joe Konior和Corey McGlone在审校中一个纰漏都不放过，经过他们的努力，这本书更趋完美。你们真是太棒了！还要感谢Corey和Pauline，督促我们纠正了太多过于正式（确切地讲，有些不合适）的措辞。还要特别感谢JavaRanch提供了那么多优秀的资源。

非常感谢Louise Barr，最优秀的Web设计师，她让我们严谨地完成设计，同时正确地使用HTML和CSS（尽管你可能对我们现在的设计还是颇有微辞）。

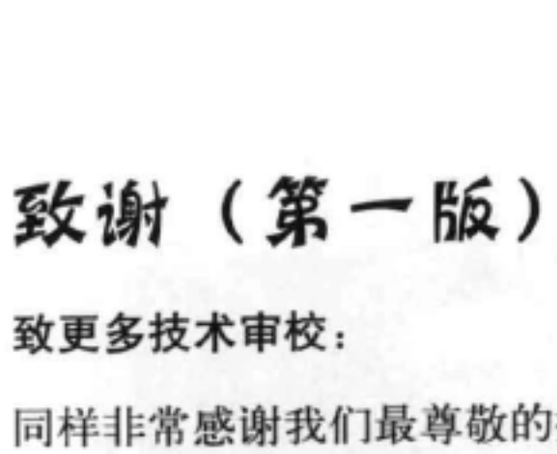
致谢（第一版）*

致更多技术审校：

同样非常感谢我们最尊敬的技术审校David Powers。对David我们真是又爱又恨，他让我们拼命工作，不过我们也得到了丰厚的回报。说真的，正是根据David的意见，我们对这本书做了大幅修改，现在这本书的技术含量绝对是之前的两倍。谢谢你，David。

致O'Reilly：

要向我们的编辑Brett McLaughlin致以最真诚的感谢，他牺牲了与家人共享天伦之乐的时间，为这本书扫清了所有障碍。Brett还为这本书的编辑相当费心费力（编辑Head First书可真不是件容易的事）。谢谢你，Brett，如果没有你，就不会有这本书。



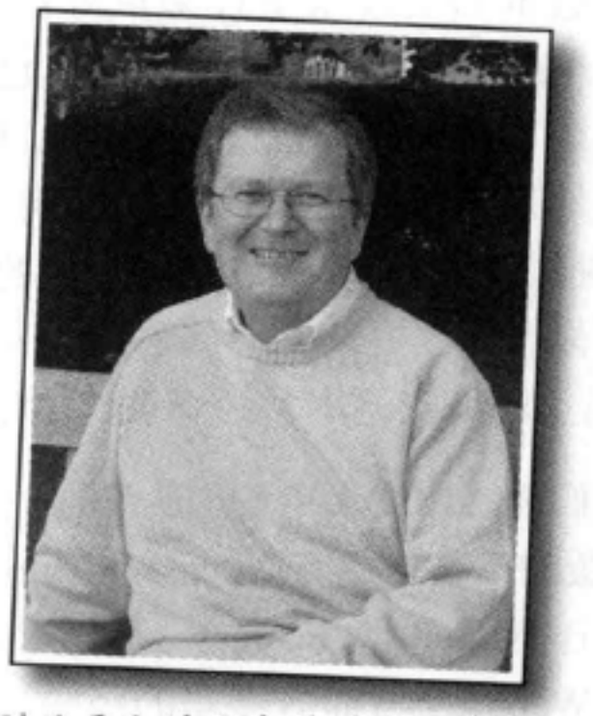
Brett McLaughlin

最诚挚地感谢整个O'Reilly团队：Greg Corrin、Glenn Bisignani、Tony Artuso和Kyle Hart全力做市场推广，我们非常欣赏他们别出心裁的方式。感谢Ellie Volkhausen充满灵感的封面设计，有着一如既往的独特风格，还要感谢Karen Montgomery，她让这本书的封面有了生命力。与以往一样，还要感谢你，Colleen Gorman，谢谢你为文字编辑付出的艰辛劳动（还以此为乐）。如果没有Sue Willing和Claire Cloutier的努力，我们不可能完成这样一本色彩丰富的书。

Head First书的致谢肯定少不了Mike Loukides，是他将Head First的概念变成了这样一个系列，另外要感谢Tim O'Reilly一如既往的支持。最后，还要感谢Mike Hendrickson，是他让我们加入Head First这个大家庭，而且坚信我们能够胜任这个工作。

致Kathy Sierra和Bert Bates：

最后，但绝不是最不重要，非常感谢Kathy Sierra和Bert Bates，不仅是我们的合作伙伴，你们绝对是这个系列的“核心人物”。感谢你们对我们的信任甚至多于你们的孩子。再次希望我们没有辜负你们的期望。那三天的讨论为这本书指明了方向，希望很快还有这样的机会。噢，对了，下一次能不能打个电话给LTJ，让他来西雅图旅行一趟？



最让人尊敬的技术审校 David Powers

别让柔和的毛衣欺骗了，这家伙相当强硬（当然是指技术方面）。

Bert Bates

Kathy Sierra

Kara

正在忙着完成《Head First Parelli》。

*之所以要感谢这么多人，这是因为我们发现了这样一条定律，书中致谢里提到的每个人都至少会买一本书，可能还会买好几本书，给亲戚和周围的所有人都送上一本。如果你希望我们在下一本书的致谢里提到你，而且你的家族人很多的话，则可以写信给我们。

技术审校人员（第二版）

要是不认识高水平的HTML与CSS大师David Powers，没有他当这本书的技术审校来审查错漏，我们简直夜不能寐。事实上，这本书第一版出版之后，已经过去了这么多年，我们不得不请了一位私家侦探才找到他（说来话长，不过最后总算在他的地下HTML与CSS研究室找到了）。说正经的，这本书中的所有技术错误都要归咎于作者（也就是我们），不过，我们可以给你打保票，David为确保我们不出错绝对是竭尽全力。重申一句，David对这本书的要求真是相当苛刻。

我们要感谢技术审校团队的每一个人。Joe Konior再次加入了这一版的工作，还有Dawn Griffiths（Head First C的合作者），另外还有Shelley Powers（他绝对是一个丰富的HTML与CSS宝库，在Web开发方面有多年的经验）。再说一次，你们太棒了！你们的反馈那么细致深入，对我们大有帮助。谢谢你们。

致谢（第二版）

最需要感谢的是我们的主编Mike Hendrickson，是他在各个方面让这本书顺利得以出版（而不只是写书这一个环节），在整个过程中有他一直陪伴，更重要的是（这也是所有编辑做的最重大的事情），他充分信任我们能完成这个任务！谢谢你，Mike，没有你，我们的书都不可能问世。十多年来，你一直在支持我们，实在让我们感激不尽！

当然，出版一本书绝对是个大工程，正是O'Reilly这些才能非凡而又友好温和的人在幕后辛勤地工作，才有了这样一本书。我们要向整个O'Reilly团队致以最诚挚的感谢：Kristen Borg（无与伦比的制作编辑）、非凡的Rachel Monaghan（校对）、Ron Strauss（感谢你缜密的索引）、Rebecca Demarest（感谢你提供的插图）、 Karen Montgomery（一流的封面设计者），最后但绝不是最不重要，还要感谢Louise Barr，她总能让我们的页面看起来更棒。

Safari®图书在线

Safari图书在线是一个按需提供资源的数字图书馆，从中可以很容易地搜索7500本技术书、参考书和视频，快速找到你想要的答案。

只需订阅，就可以从我们的在线图书馆阅读任何页面，观看任何视频。你可以在你的手机和移动设备上看书，能够在新书出版之前获得书目，还可以享有特权查看正在编写的书稿并向作者提出反馈意见。你可以剪切粘贴代码示例、整理最喜欢的图书、下载你需要的章节、为重要内容设置书签、创建笔记、打印页面，此外还有大量特性可以节省时间，让你从中受益。

O'Reilly Media已经将这本书（英文版）上传到Safari图书在线服务。要想通过数字方式全面访问这本书以及O'Reilly和其他出版商提供的其他类似图书，可以免费注册 http://my.safaribooksonline.com。

目录（概览）

	引子	xxv
1	Web语言：认识HTML	1
2	认识HTML中的“HT”：深入了解超文本	43
3	Web页面建设：构建模块	77
4	Web镇之旅：连接起来	123
5	认识媒体：为你的页面增加图像	163
6	严肃的HTML：标准及其他	219
7	加一点样式：CSS入门	255
8	扩大你的词汇量：增加字体和颜色样式	311
9	与元素亲密接触：盒模型	361
10	高级Web建设：div与span	413
11	摆放元素：布局与定位	471
12	现代HTML：HTML5标记	545
13	建立表格：表格与更多列表	601
14	实现交互：HTML表单	645
附录：其他十大主题（我们没有谈到的）		697

目录

引子

你的大脑与HTML和CSS。你想学些新东西，但是你的大脑总是帮倒忙，它会努力让你记不住所学的东西。你的大脑在想：“最好留出空间来记住那些确实重要的事情，比如要避开哪个野生动物，还有裸体滑雪是不是不太好。”那么，如何让你的大脑就范，让它认为如果不知道HTML和CSS你将无法生存？

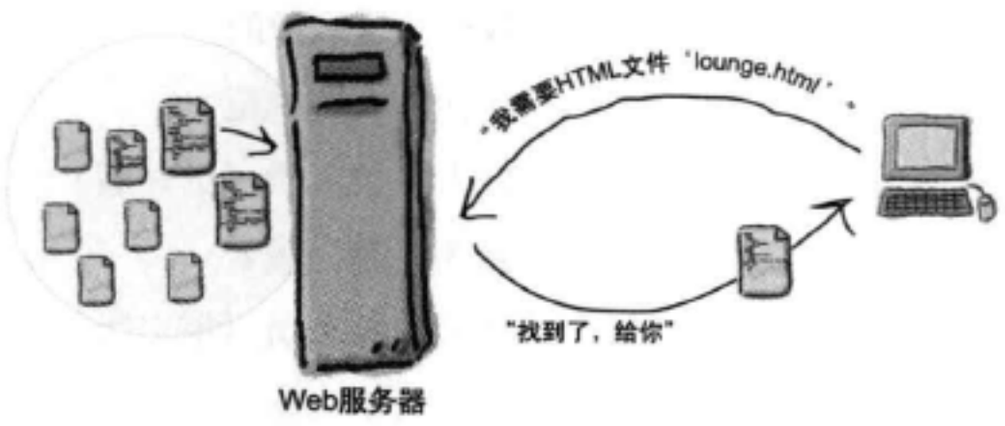
谁适合看这本书？	xxvi
元认知	xxix
我们是这么做的	xxx
让你的大脑就范	xxxi
技术审校人员（第一版）	xxxiv
致谢（第一版）	xxxv
技术审校人员（第二版）	xxxvi
致谢（第二版）	xxxvi

1 认识HTML

Web语言

要进入Web世界，在你面前只有一个障碍，就是要学会它的语言。超文本标记语言，或者简称为HTML。准备好了吗？我们要上几节语言课了。读完这一章之后，你不仅会了解一些基本的HTML元素，还能讲出带一点样式的HTML。说真的，等学完这本书，你就会像是从小在Web镇长大的一样，能自如地运用HTML。

Web让广播明星黯然失色 2
Web服务器能做什么？ 3
你写的代码（HTML） 4
浏览器创建的页面 5
你在Starbuzz咖啡馆交好运了 9
创建Starbuzz Web页面 11
创建一个HTML文件(Mac系统) 12
创建一个HTML文件（Windows系统） 14
现在，再回到Starbuzz…… 17
保存你的成果 18
在浏览器中打开你的Web页面 19
测试你的页面 20
完工了吗？ 23
另一个测试 24
标记剖析 25
认识style元素 29
给Starbuzz网站加点样式…… 30
测试样式…… 31
练习答案 38



2 深入了解超文本

认识HTML中的“HT”

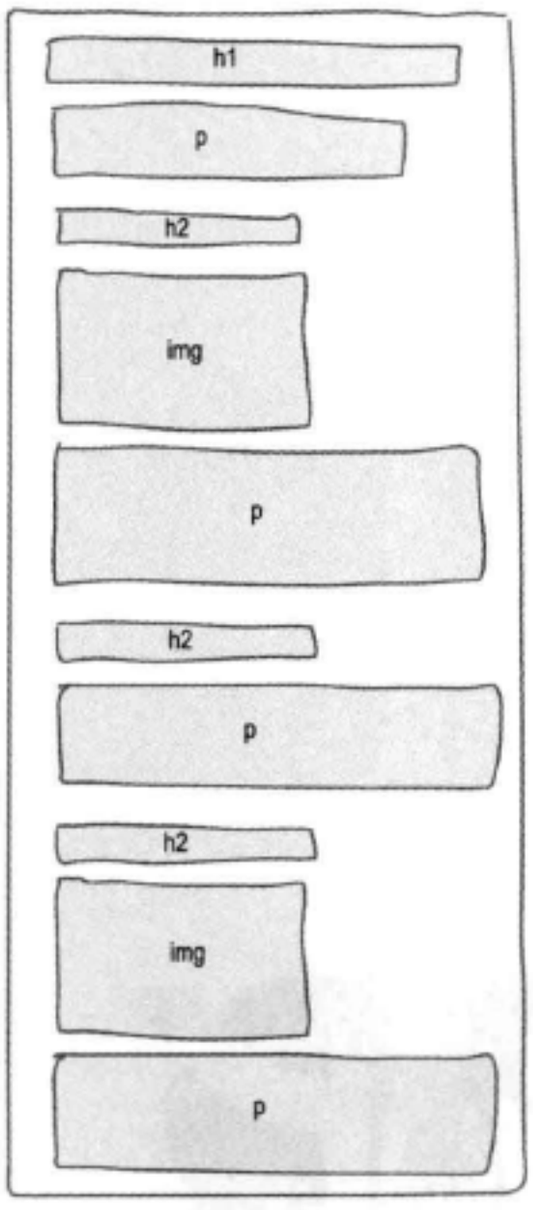
有人在说“超文本”？那是什么？哦，这就是Web的基础。第1章中我们简单认识了HTML，知道它是一种不错的标记语言（HTML中的“ML”），用来描述网页的结构。现在我们再来了解HTML中的“HT”，也就是超文本（hypertext），有了它，我们就能摆脱单个页面的束缚，链接到其他页面。顺便我们还要认识一个强大的新元素——<a>元素，了解“相对”是多么美妙的东西。好了，系好你的安全带，准备学习一些超文本吧。

Head First休闲室，全新改良	44
创建新的休闲室	46
我们做了什么？	48
了解属性	51
组织	56
组织休闲室……	57
技术难点	58
规划你的路径……	60
修复那些损坏的图像……	66
练习答案	73

3 构建模块

Web页面建设

我听说这本书会教我具体创建Web页面？你已经学到不少了：标记、元素、链接、路径……不过，如果不学以致用，创建一些一流的Web页面，这些知识就毫无意义。这一章我们要把重点放在建设上：你要从概念到蓝图来设计Web页面，浇注地基，完成建设，甚至最后还可以加几笔润色。戴上安全帽，系好安全带，我们会增加一些新工具，让你掌握一些更深入的内部知识，有了这些知识，你也能成为这个行业的佼佼者。



从日志到网站，以12迈的速度出发 79
粗略的设计草图 80
从草图到略图 81
从略图到网页 82
测试Tony的Web页面 84
增加一些新元素 85
认识<q>元素 86
很长的引用 90
增加<blockquote> 91
<q>和<blockquote> 谜案的真相 94
再回到Tony的网站…… 100
当然，也可以使用<p>元素创建列表…… 101
两步轻松构建HTML列表 102
城市列表测试 104
把一个元素放在另一个元素中称为“嵌套” 107
要理解嵌套关系，画一个图 108
使用嵌套确保标记匹配 109
练习答案 117

4 连接起来

Web镇之旅

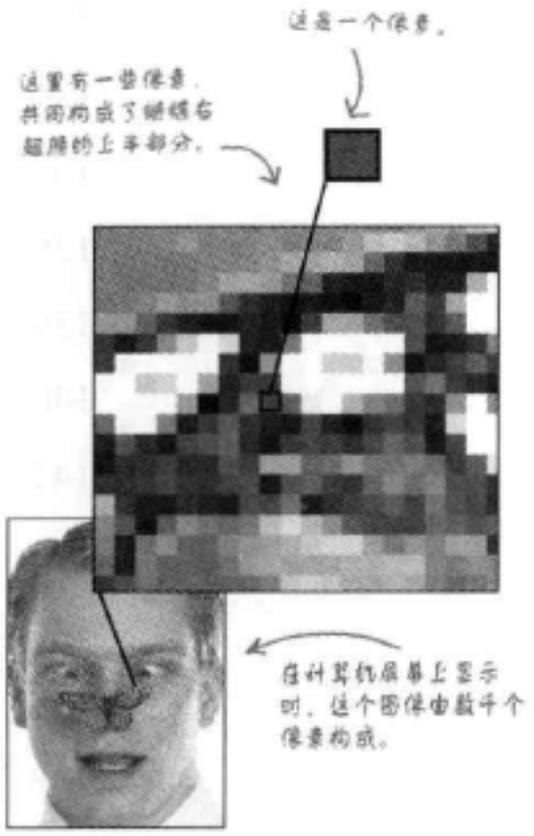
Web页面是互联网给我们的最好的一道菜。到目前为止，你只是在自己的计算机上创建HTML页面，所链接的页面也都在自己的计算机上。接下来我们会有所改变。这一章会鼓励你把这些Web页面发布到互联网上，让你的朋友、粉丝和客户都能真正看到。通过破解 h, t, t, p, :, /, /, w, w, w代码，我们还将揭开链接到其他页面的秘密。好了，收拾好行李，向下一站Web镇进发。

在Web上发布Starbuzz网站（或你自己的网站） 124
找一家托管公司 125
如何得到一个域名? 126
搬家 128
把你的文件复制到根文件夹 129
用两页尽可能讲清楚FTP 130
回到正题…… 133
我们的主干道, USA 134
什么是HTTP? 135
什么是绝对路径? 136
默认页面如何工作 139
Earl需要你帮他确定URL 140
如何链接到其他网站? 142
链接到Caffeine Buzz 143
现在来试一试…… 144
完善的Web页面 147
试一试标题…… 148
链接到一个页面 149
使用id属性为<a>创建目标 150
如何用id链接到元素 151
链接到一个新窗口 155
使用target打开新窗口 156
练习答案 160

5 为你的页面增加图像

认识媒体

笑一笑，说“茄子”。说真的，应该是笑一笑，说“gif”、“jpg”或“png”。你为Web“冲印照片”时，就会用到它们。这一章中，我们将学习如何为你的页面添加第一个媒体类型：图像。你是不是拍摄了一些数码照片想要发到网上？没问题。是不是需要在页面上加一个logo？小事一桩。不过在深入学习这些内容之前，是不是需要正式介绍一下<img>元素？真是抱歉，我们不是有意疏忽，只是一直没有找到“合适的机会”。作为补偿，下面用一整章专门讨论<img>。学完这一章，你就会全面地了解使用<img>元素及其属性的所有细节了。你还会亲眼看到这个小元素是怎样让浏览器做大量额外的工作来获取和显示图像的。



浏览器如何处理图像 164
图像是如何工作的 167
<img>: 不再只是相对链接 171
一定要提供候选格式 173
调整图像大小 174
创建超级粉丝网站：myPod 175
哇！图像太大了 178
打开图像 182
调整图像大小 183
修复myPod HTML 188
myPod的更多照片 190
把缩略图变成链接 196
为照片创建单独的页面 197
那么，怎么链接到图像呢？ 198
要使用什么格式？ 203
透明，还是不透明？这是个问题…… 204
等一下，Web页面背景色到底是什么？ 206
查看有蒙版的logo 207
把logo增加到myPod Web页面 208
练习答案 213

6 标准及其他

严肃的HTML

关于HTML还要知道些什么？很不错，现在你已经掌握了HTML。实际上，是不是该转向CSS了？来学习如何让这些乏味的标记看起来更漂亮。在此之前，我们需要确保你已经掌握了足够的HTML知识，有能力应对激烈的竞争。不要误解我们的意思，没错，你一直在写一流的HTML，不过还需要了解额外的一些知识，才能真正建立“工业标准”的HTML。还要利用这个机会确保你使用的是最新、最棒的HTML标准，也就是HTML5。保证了这一点，就可以确保HTML在所有浏览器上（至少是你关心的浏览器上） 能有更一致的显示，不仅如此，还可以在最新的iPhone、iPod之类的设备上（选择你最喜欢的设备）很好地显示。你的页面会加载得更快，可以很好地结合CSS，能够随着标准发展稳步走向未来。准备好了吗？这一章会让你从一个Web工匠变成真正的Web高手。

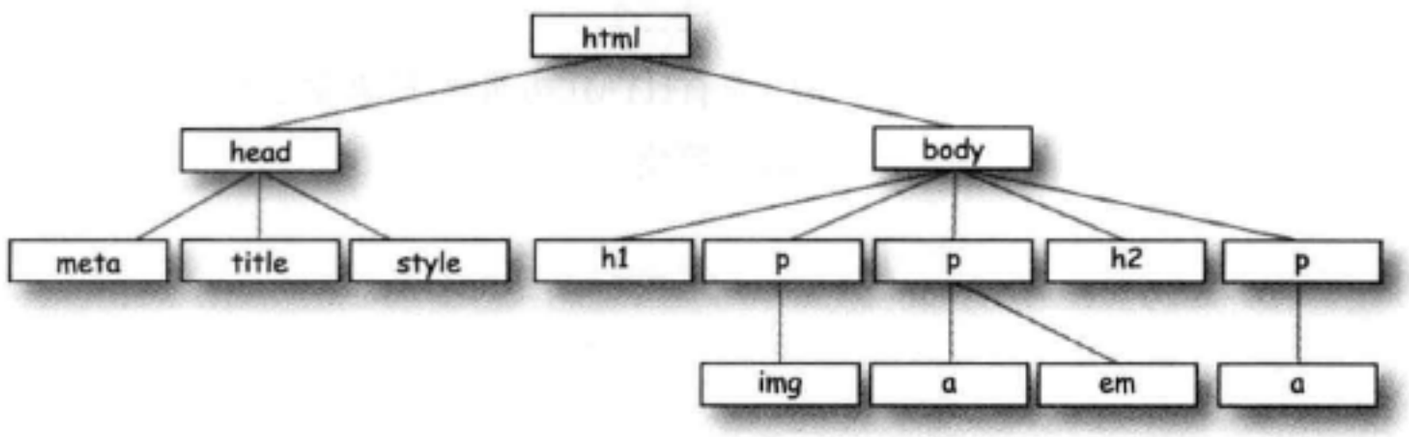
HTML简史 222
新的、改进的HTML5 doctype 227
HTML, 新的“活标准” 228
增加文档类型定义 229
测试doctype 230
认识W3C验证工具 233
验证Head First休闲室 234
唉呀，我们遇到一个问题…… 235
修正错误 236
就快完成了…… 237
增加一个<meta>指定字符编码 239
让验证工具（和很多浏览器）接受<meta>标记…… 240
事不过三? 241
叫所有HTML专业人员来拿手册…… 244
练习答案 251

7 CSS入门

加一点样式

我听说这本书会介绍CSS。到目前为止，你一直在专心学习HTML来创建Web页面结构。不过可以看到，浏览器对样式有很多要求。当然，我们可以请时尚人士来帮忙，不过没有必要。利用CSS，你就能完全掌控页面的表现，通常甚至不用对HTML做任何改变。真的这么容易吗？嗯，你得学习一种新的语言。毕竟，Web镇是一个讲多种语言的小镇。读完这一章对CSS语言学习的介绍后，你就能胸有成竹地站在主干道两边与Web镇的人们交流了。

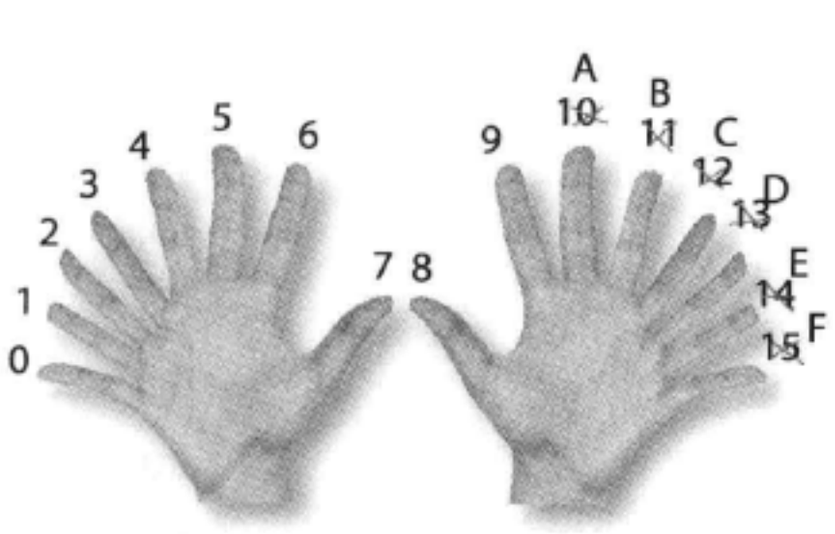
你已经离开堪萨斯 256
偶然听到的Web镇“交换空间”节目 258
结合HTML和CSS 259
把CSS放入HTML 261
为休闲室增加样式 262
再在欢迎消息下面加一条线 265
那么，选择器到底如何工作？ 267
通过图解来研究选择器 270
为清凉饮料和路线说明页面加入休闲室页面的样式 273
来谈谈继承…… 281
覆盖继承 284
把元素增加到greentea类 287
创建一个类选择器 288
更深入地研究类…… 290
关于应用样式的世界上最简短最快捷的指南 292
练习答案 303



8 增加字体和颜色样式

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 312
字体系列到底是什么？ 314
使用CSS指定字体系列 317
让Tony的旅行日志焕然一新 318
每个人都有不同的字体，我该如何处理？ 321
Web字体如何工作 323
如何为页面增加Web字体…… 325
调整字体大小 328
那么，我到底该如何指定字体大小呢？ 330
下面修改Tony的Web页面中的字体大小 332
改变字体粗细 335
为字体增加风格 337
让Tony的引用有一点斜体风格 338
Web颜色如何工作？ 340
如何指定Web颜色？来数数看有多少种方法…… 343
十六进制码速查指南 346
如何找到Web颜色 348
再回到Tony的页面…… 351
用不到一页的篇幅介绍 353
删除下划线…… 354
练习答案 357

9 盒模型

与元素亲密接触

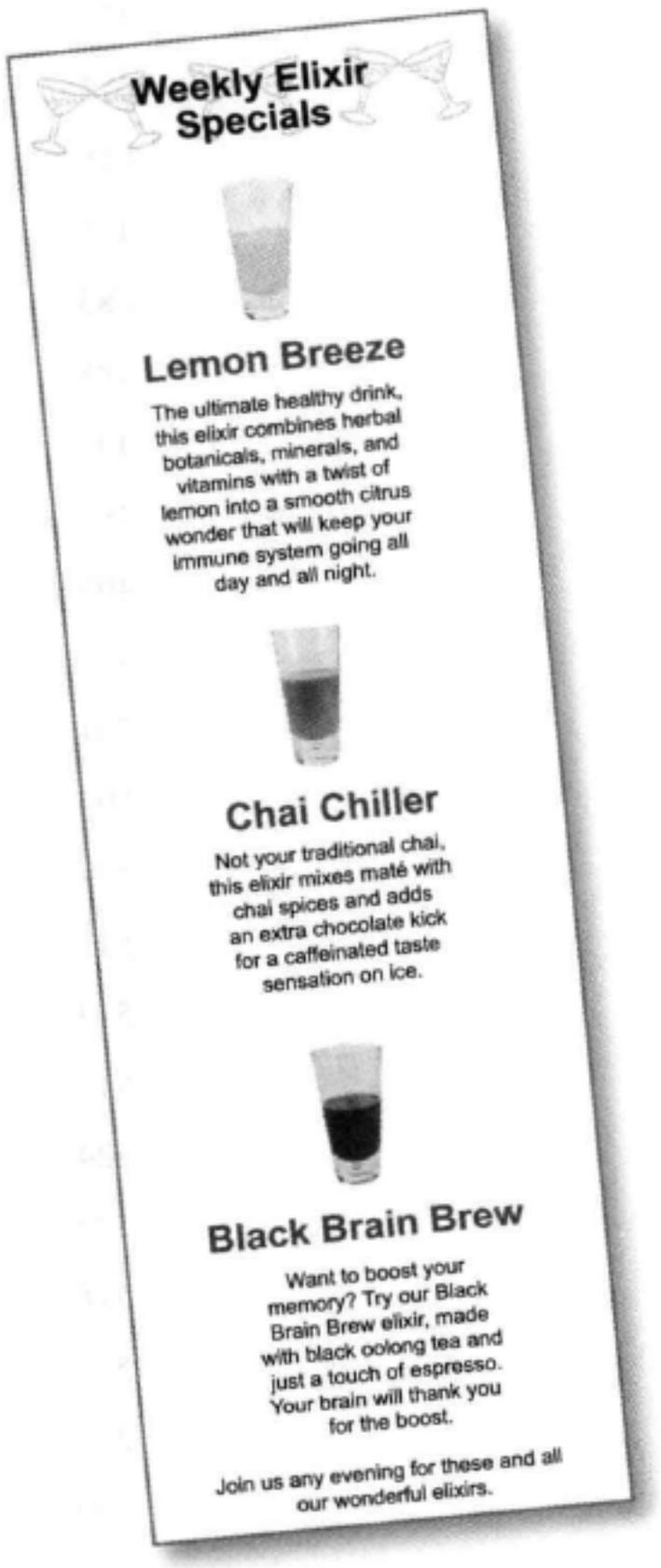
要推进Web建设，确实需要了解建筑材料。这一章中，我们会仔细研究我们的建筑材料：HTML元素。我们会把块元素和内联元素放在显微镜下，仔细观察它们的组成。你会看到，利用CSS，对于构造元素的各个方面几乎都可以控制。不过，我们并不会就此止步，你还会了解如何为元素提供唯一的身份。如果这些还不够，这一章还会告诉你什么时候要使用多个样式表，以及为什么要用多个样式表。好了，翻开下一页，与元素来个亲密接触吧。

休闲室要升级 362
先做一些简单的升级 364
查看新行高 366
准备全面翻新 367
仔细分析盒模型 368
对盒子能做哪些设置 370
创建guarantee样式 375
测试段落边框 376
保证段落的内边距、边框和外边距 377
增加一个背景图像 380
修正背景图像 383
如何只在左边增加内边距？ 384
如何只在右边增加外边距？ 385
边框简明指南 386
完善边框 389
休闲室页面中使用id 396
使用多个样式表 399
样式表，不再只面向浏览器…… 400
直接在CSS中增加媒体查询 401
练习答案 407

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）。

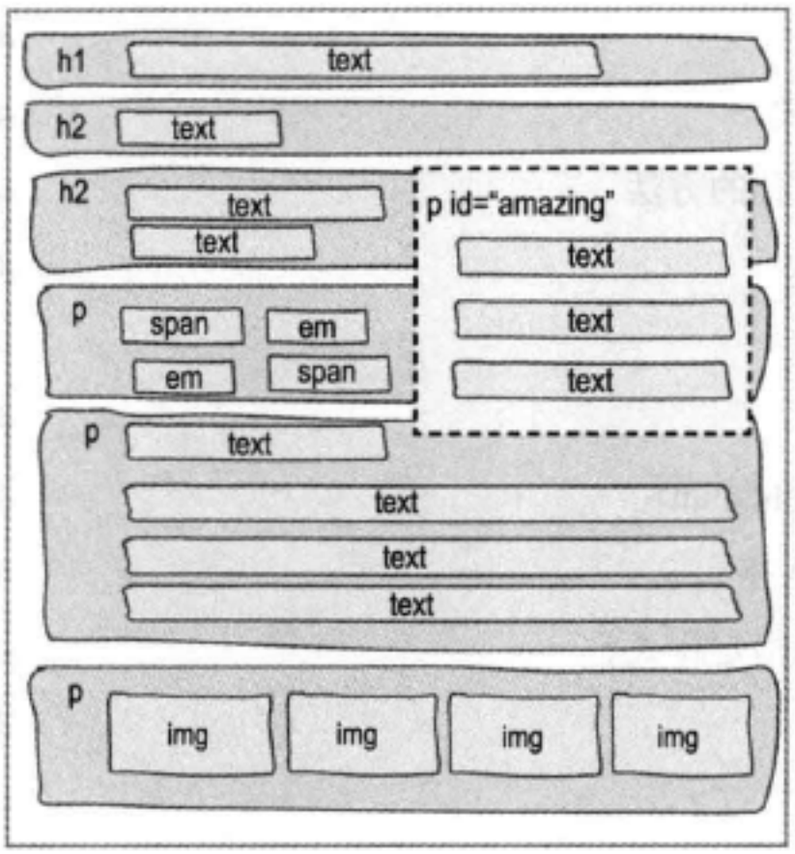


仔细观察饮料HTML 415
下面来研究如何将一个页面划分为逻辑区 417
增加边框 424
为饮料区增加一些真正的样式 425
处理elixir宽度 426
为elixirs增加基本样式 431
我们需要一种选择子孙的方法 437
改变饮料标题的颜色 439
修正行高 440
来点快捷方式 442
只需简单的3步来增加 448
<a>元素和它的多重人格 452
如何根据元素的状态指定样式？ 453
运用伪类 455
是不是该谈谈“层叠”了？ 457
层叠 459
欢迎参加“我有多特定？”游戏 460
综合在一起 461
练习答案 467

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。

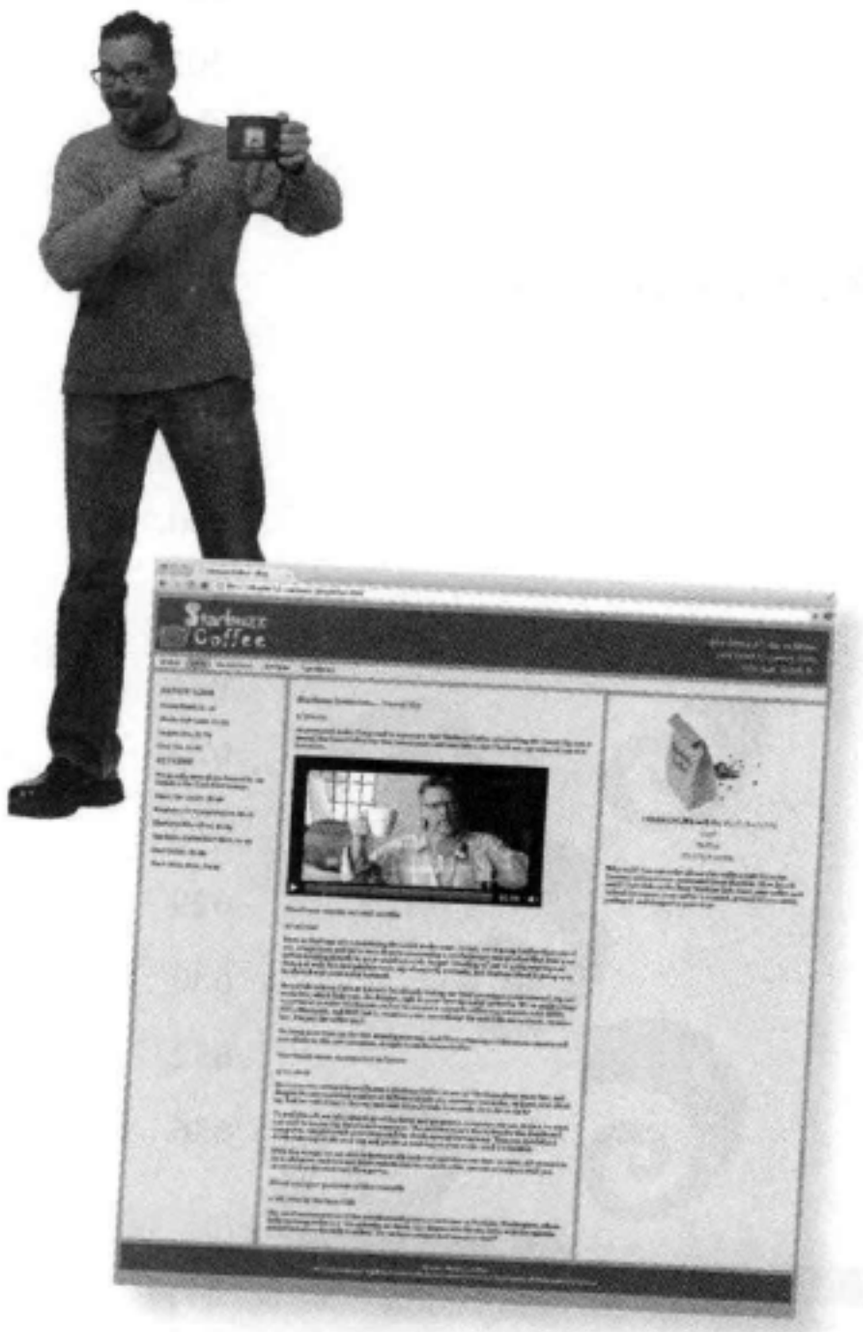


你做超级智力题了吗？ 472
使用流 473
内联元素呢？ 475
如何集成？ 476
如何浮动元素 479
新的Starbuzz页面 483
把边栏移动到页眉下面 488
修正两栏问题 491
设置主内容区的外边距 492
解决重叠问题 495
右紧左松 498
流体与冻结设计 501
绝对定位如何工作 504
修改Starbuzz CSS 507
CSS表格显示如何工作 511
为表格显示增加HTML结构 513
这些间距是怎么回事？ 517
页眉的问题 524
用float修正页眉图像 525
指定奖杯位置 528
固定定位如何工作？ 531
使用负的left属性值 533
练习答案 539

12 HTML5标记

现代HTML

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 546
更新Starbuzz HTML 551
如何为这些新元素更新CSS 554
建立博客页面的CSS 563
还需要为博客增加一个日期…… 565
为博客增加<time>元素 566
如何增加更多<header>元素 568
这些首部到底怎么了？ 570
页眉的最终测试 571
完成导航 574
谁需要GPS？测试导航 575
哇！看看这里的导航！ 577
创建新博客条目 580
灯光，摄像，开拍…… 581
<video>元素如何工作？ 583
仔细检查video属性…… 584
关于视频格式需要知道什么 586
视频格式竞争对手 587
如何处理所有这些格式…… 589
如何更具体地指定视频格式 590
练习答案 597

13 表格与更多列表

建立表格

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

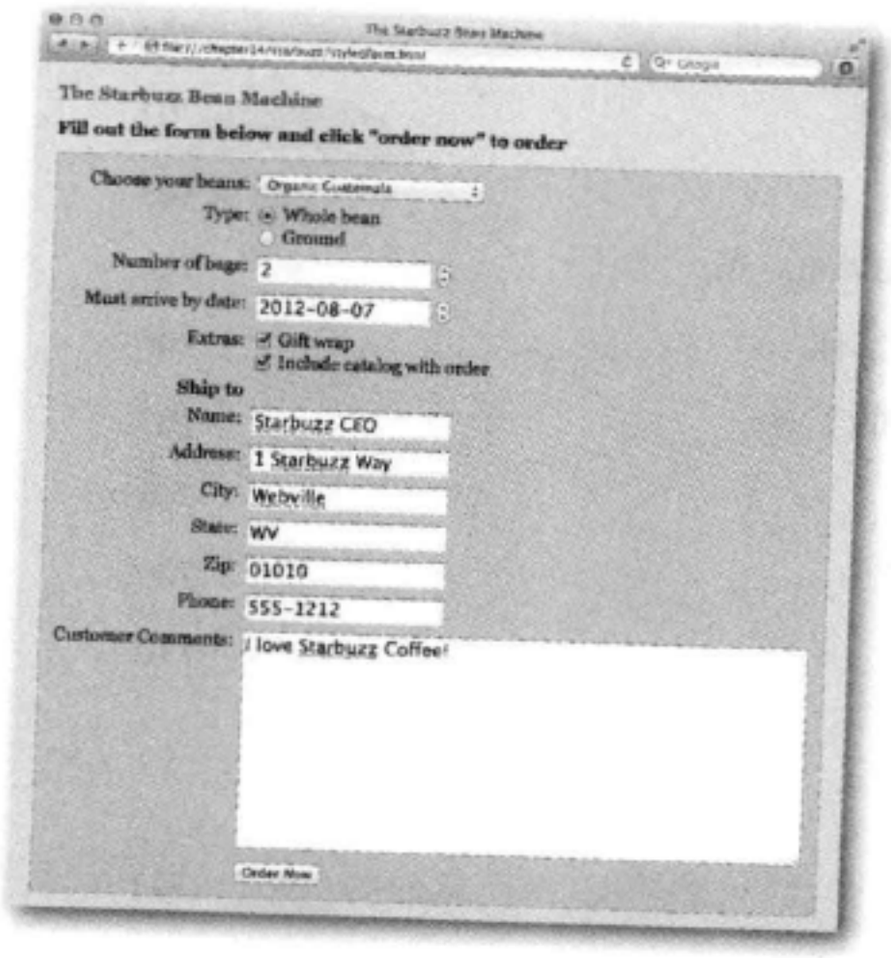
如何用HTML建立表格？ 603
用HTML创建一个表格 604
浏览器创建了什么？ 605
表格剖析 606
增加一个标题 609
开始指定样式之前，先把表格放在Tony的页面上 611
折叠边框 616
来点颜色怎么样？ 618
Tony有一个有趣的发现 620
再来看Tony的表格 621
如何让单元格跨多行 622
测试表格 624
天堂也有麻烦？ 625
覆盖嵌套表格表头的CSS 629
对Tony网站的最后润色 630
如果需要一个定制标记呢？ 632
练习答案 636

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

14 HTML表单

实现交互

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



表单如何工作 646
你写的HTML代码 648
浏览器创建的页面 649
<form>元素如何工作 650
准备建立Bean Machine表单 660
增加<form>元素 661
表单元素名如何工作 662
将这些<input>元素放在HTML中 664
为表单增加更多输入元素 665
增加<select>元素 666
允许客户选择全豆咖啡还是研磨咖啡 668
试试单选按钮 669
使用更多输入类型 670
增加数字和日期输入类型 671
完成表单 672
增加复选框和文本区 673
GET的实际使用 679
将表单元素放入HTML结构实现表格显示布局 684
用CSS建立表单样式 686
关于可访问性 688
表单中还可以有哪些元素? 689
练习答案 693

15 附录：其他

十大主题（我们没有谈到的）

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

#1 更多CSS选择器 698

#2 开发商特定的CSS属性 700

#3 CSS变换和过渡 701

#4 交互性 703

#5 HTML5 API和Web应用 704

#6 再来谈谈Web字体 706

#7 创建Web页面的工具 707

#8 XHTML5 708

#9 服务器端脚本 709

#10 音频 710

索引 711

1 认识HTML

Web语言

要进入Web世界，在你面前只有一个障碍，就是要学会它的语言。

超文本标记语言，或者简称为HTML。准备好了吗？我们要上几节语言课了。读完这一章之后，你不仅会了解一些基本的HTML元素，还能讲出带一点样式的HTML。说真的，等学完这本书，你就会像是从小在Web镇长大的一样，能自如地运用HTML。

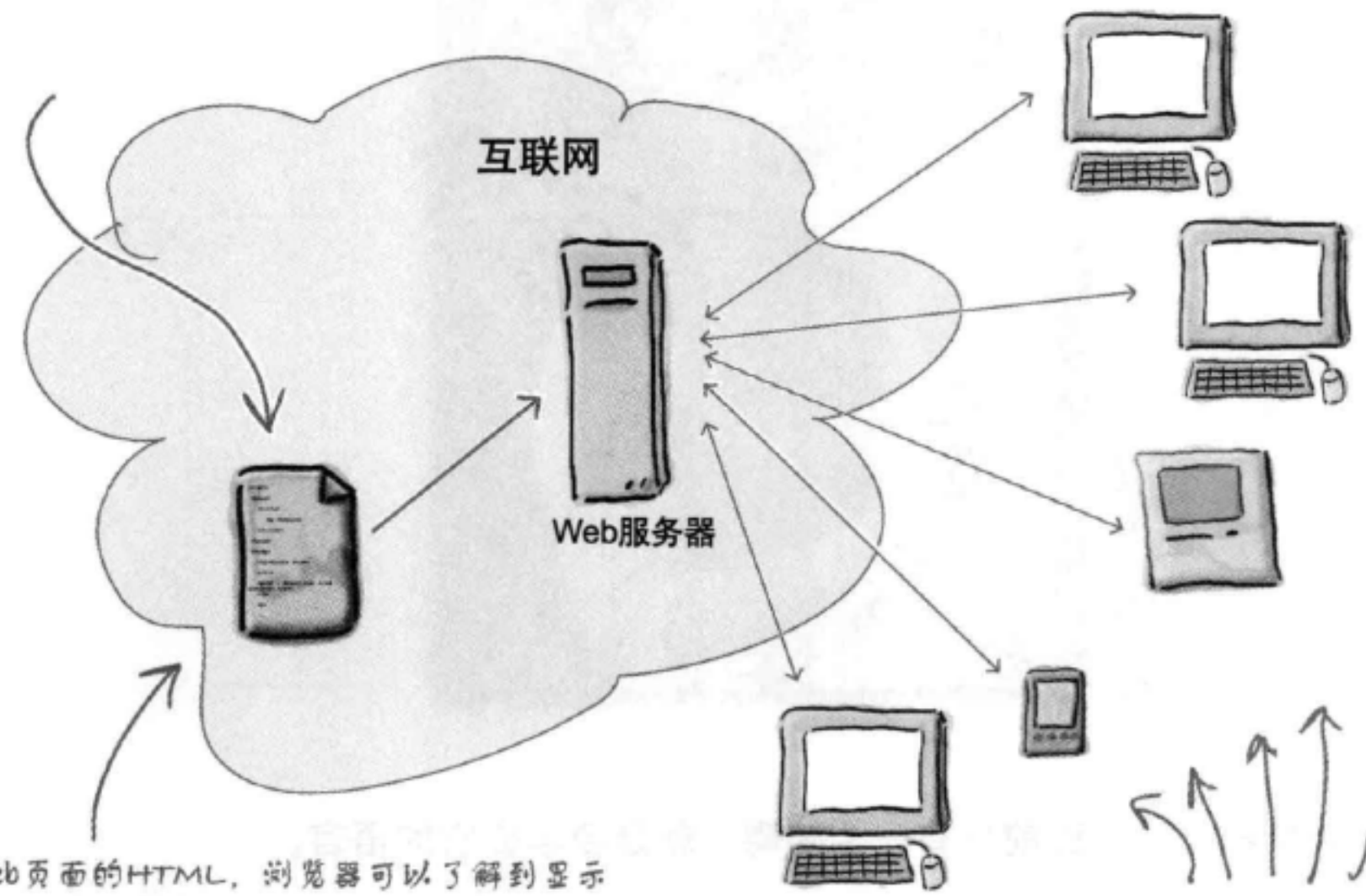
Web
电视让广播明星黯然失色

想表达某种想法？想出售某种商品？或者是不是需要一个有创意的销售渠道？进入Web世界吧，也许不用我们说你已经知道，Web早已经成为全球通用的交流方式。更棒的是，你能参与其中。

不过，如果你真的希望有效地使用Web，就必须对HTML有所了解，当然，还要对Web如何工作略知一二。下面先从宏观的角度来看看：

要建立Web页面，需要创建用超文本标记语言（HyperText Markup Language，简写为HTML）编写的文件，把它们放在一个Web服务器上（本书后面会介绍如何把你的文件放在一个服务器上。）

一旦把文件放在Web服务器上，任何浏览器就能通过互联网得到你的Web页面。

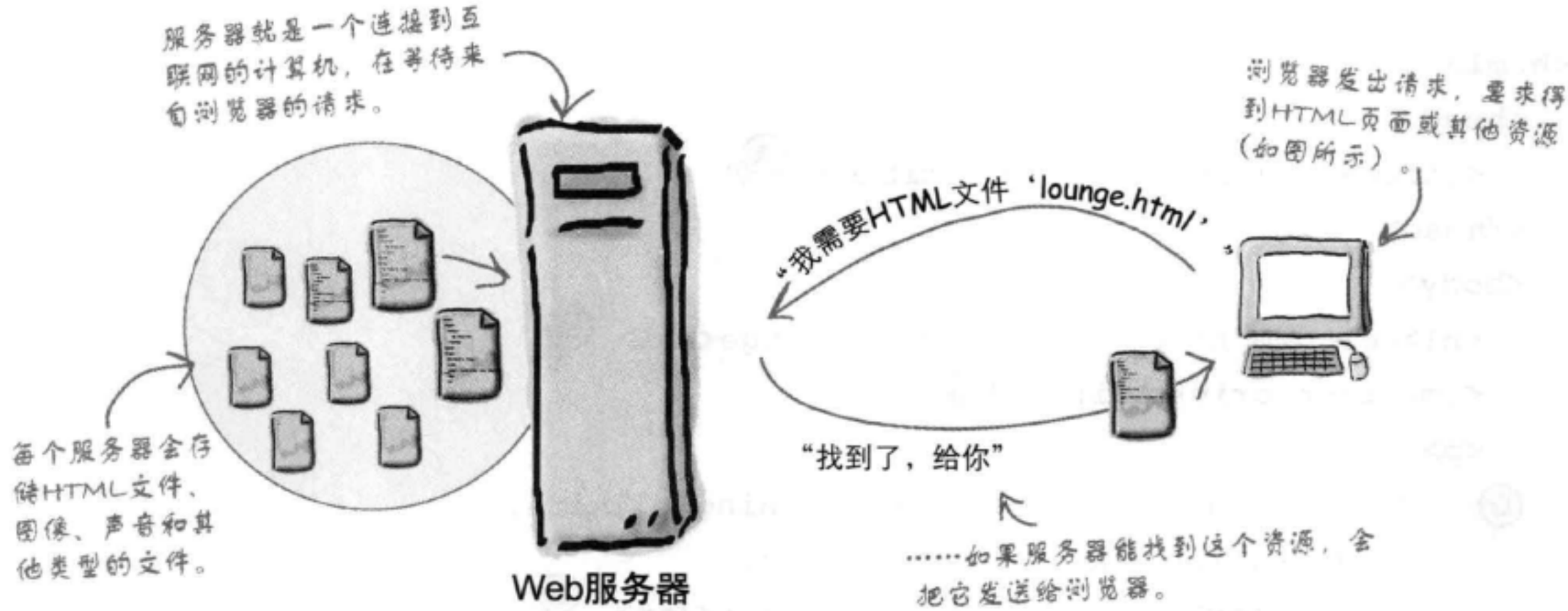


根据Web页面的HTML，浏览器可以了解到显示页面所需的全部信息。另外，如果编写得足够好，你的网页甚至在手机和移动设备上就能很好地显示，还允许有视觉缺陷的人在语音浏览器和屏幕放大器上查看。

大量PC和其他设备已经连接到互联网，它们都运行着Web浏览器。更重要的是，我们的朋友、家人、粉丝和潜在用户可能正在使用这些设备！

Web服务器能做什么？

Web服务器在互联网上有一份全天候的工作，夜以继日、不知疲倦地等待来自Web浏览器的请求。什么类型的请求？可能是请求Web页面、图像或声音，或者可能甚至是一个视频。服务器收到这些资源请求时，会查找所请求的资源，然后把找到的资源发回给浏览器。



Web浏览器能做什么？

你已经知道浏览器是如何工作的：你在网上冲浪，单击一个链接来访问某个页面。这个单击会导致浏览器向Web服务器请求一个HTML页面，获取这个页面，并在你的浏览器窗口中显示。

但是浏览器怎么知道如何显示一个页面呢？这里就要用到HTML了。HTML会告诉浏览器页面的所有内容和结构。下面来看这是如何做到的……

你写的代码（HTML）

如此说来，你已经知道HTML是浏览器显示页面的关键，但是HTML到底是什么样呢？它会做些什么？

下面来看一些HTML……假设你要创建一个Web页面，为Head First休闲室（Head First Lounge)做个宣传，这是当地大家常去的一个休闲场所，有美妙的音乐、清爽的饮料，还提供无线上网。下面是你编写的HTML代码：

```
<html>
  <head>
    <title>Head First Lounge</title>                    Ⓐ
  </head>
<body>
  <h1>Welcome to the Head First Lounge</h1>             Ⓑ
  <img src="drinks.gif">                                Ⓒ
  <p>
Ⓓ  Join us any evening for refreshing elixirs,
    conversation and maybe a game or
    two of <em>Dance Dance Revolution</em>.             Ⓔ
    Wireless access is always provided;
    BYOWS (Bring your own Web server).
  </p>
  <h2>Directions</h2>                                   Ⓕ
  <p>
Ⓖ  You'll find us right in the center of
    downtown Webville. Come join us!
  </p>
  </body>
</html>
```

我们并不指望你对HTML很了解。

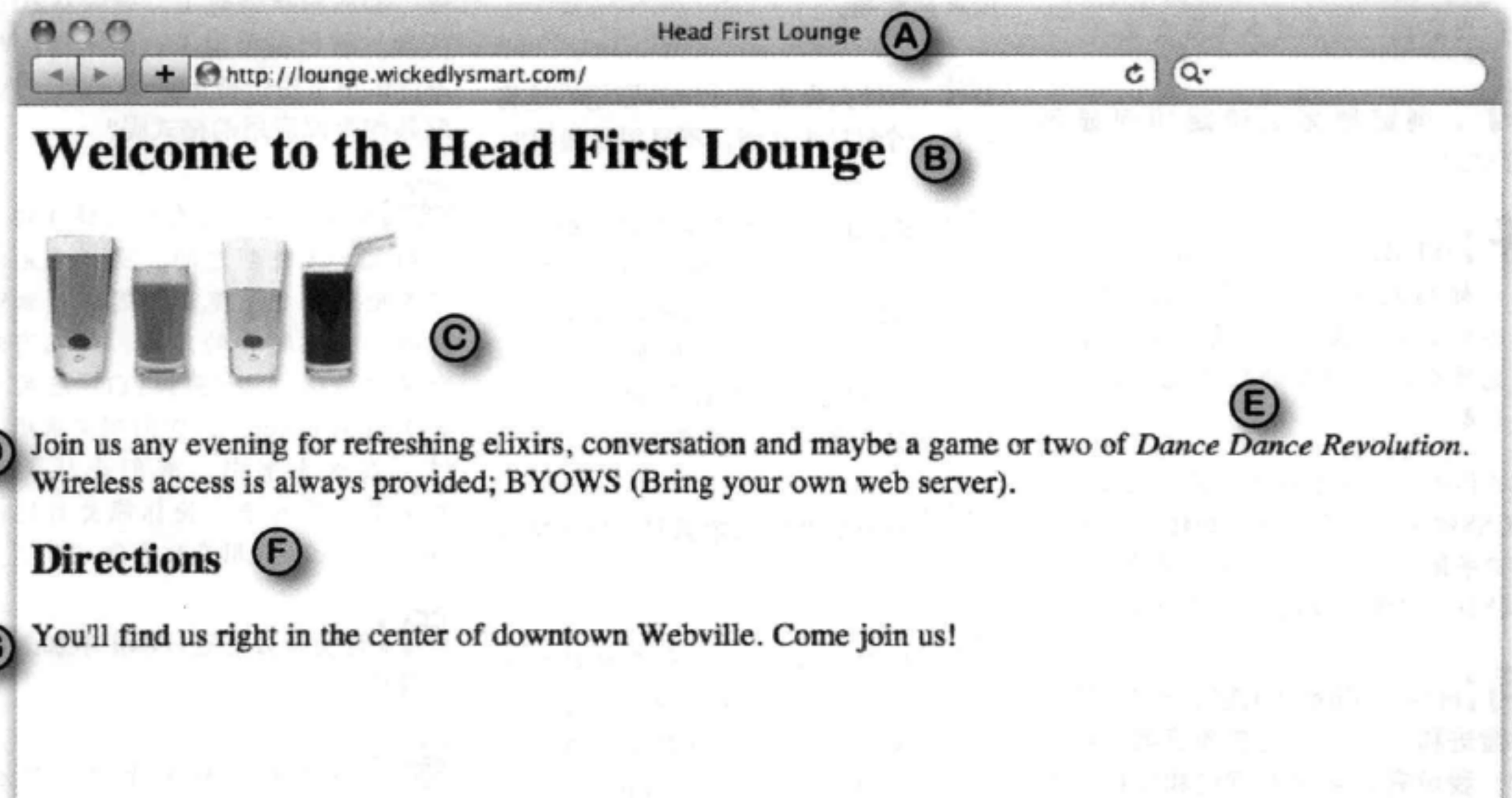
目前只需要你对HTML有个初步的认识，稍后我们会循序渐进地详细介绍所有内容。现在就来研究这个HTML，下一页会看到它在浏览器上如何表示。要特别注意每个字母标注，明确它们在浏览器中如何显示以及在哪里显示。

浏览器创建的页面

浏览器读到你的HTML时，它会翻译文本中的所有标记。标记就是尖括号括起来的词或字符，例如<head>、<p>、<h1>等。标记会告诉浏览器文本的结构和含义。所以并不是交给浏览器一大堆的文本，利用HTML，你可以用标记告诉浏览器哪些文本是标题，哪些文本是段落，哪些文本需要强调，或者图像需要放在什么位置。

下面来看浏览器如何翻译Head First休闲室中的标记：

注意HTML中的各个标记如何对应浏览器显示的内容。



there are no Dumb Questions

问：这么说，HTML就是一堆可以用来包围文本的标记，是吗？

答：对初学者来说是这样的。要记住，HTML（HyperText Markup Language）是超文本标记语言的缩写，所以HTML提供了一种用标记“标示”文本的方法，来告诉浏览器文本的结构是怎样的。不过，HTML还有另一个方面，也就是超文本（HyperText），这个内容我们会在这本书后面谈到。

问：浏览器怎么确定如何显示HTML？

答：HTML会告诉浏览器文档的结构：标题放在哪里，段落放在哪里，哪些文本需要强调等。有了这些信息，浏览器会按照内置的默认规则显示各个元素。

不过你不必受制于默认设置。完全可以用CSS增加你自己的样式和格式化规则，确定字体、颜色、大小和页面的很多其他特征。本章后面还会介绍CSS。

问：Head First休闲室的HTML有很多缩进和空格，不过在浏览器上显示时，我没有看到这些缩进和空格，怎么回事？

答：没错，这个问题问得好。浏览器会忽略HTML文档中的制表符、回车和大部分空格。实际上，它们会根据你的标记来确定在哪里换行或分段。

那么，既然浏览器会忽略这些空白符，我们为什么还要插入自己的格式呢？这是为了帮助我们在编辑HTML时能够更容易地读懂文档。随着HTML文档变得越来越复杂，你会发现，在某些地方加一些空格、回车和制表符，这会对提高HTML可读性很有帮助。

问：这么说，共有两级标题，<h1>和子标题<h2>，是吗？

答：实际上浏览器通常显示的标题一共有6级，从<h1>到<h6>，字体由大到小。除非你要创建一个非常复杂、庞大的文档，否则一般不会用到<h3>以后的标题。

问：为什么会需要<html>标记？这肯定是一个HTML文档，不是很明显吗？

答：<html>标记告诉浏览器你的文档确实是一个HTML文件。如果没有加这个标记，有些浏览器可以接受，但有一些是不允许的，本书后面介绍到“工业级强度HTML”时，你会看到包含这个标记相当重要。

问：一个文件怎么才算是HTML文件呢？

答：HTML文件是一个简单的文本文件。与字处理文件不同，其中没有嵌入特殊的格式。按照约定，我们会在文件名末尾加一个“.html”，让操作系统更清楚这个文件是什么。不过，可以看到，真正重要的是你在文件里放什么内容。

问：大家都在谈论HTML5。我们也会用HTML5吗？如果是这样，那么为什么不直接说“HTML-FIVE”，而不是“HTML”呢？

答：你现在学习的是HTML，实际上HTML5就是HTML的最新版本。HTML5最近很受关注，这是因为它让我们编写HTML的很多做法大大简化，而且还提供了一些新的功能，这些都会在这本书中介绍。另外，通过它的JavaScript应用编程接口（Application Programming Interfaces，API），它还提供了一些高级特性。这些在《Head First HTML5 Programming》一书有详细介绍。

问：标记看起来很傻，“所见即所得”的应用早就有了，好像从20世纪70年代就已经出现了，是不是？Web为什么不采用一种像Microsoft Word或其他类似应用的格式呢？

答：Web是基于没有任何特殊格式字符的文本文件创建的。这样才允许世界各地的各种浏览器都能够获取Web页面，并理解它的内容。现在有很多所见即所得（WYSIWYG）应用（比如Dreamweaver），它们确实很棒。不过，在这本书中，我们会从最原始的文本文件入手，使你能更好地理解Dreamweaver应用在后台做了什么。

问：有没有办法在HTML中加入自己的注释？

答：当然有，如果你把注释放在<!--和-->之间，浏览器就会把它们完全忽略。假设你想加入一个“Here's the beginning of the lounge content”注释，可以这样写：

```
<!-- Here's the beginning
of the lounge content -->
```

注意，注释可以写为多行。要记住，“<!--”和“-->”之间放置的所有内容（甚至HTML）都会被浏览器忽略。

Sharpen your pencil

也许你没想到，实际你对HTML并不陌生……

下面再给出Head First休闲室的HTML。研究下面这些标记，看你能不能猜出它们会告诉浏览器关于内容的哪些信息。把你的答案写在右边的空行上，我们已经帮你填上了前两个。

```
<html>                                                    告诉浏览器HTML从这里开始。
  <head>                                                  页面"head"开始（稍后介绍更多有关内容）。
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1>
    <img src="drinks.gif">
    <p>
      Join us any evening for refreshing elixirs,
      conversation and maybe a game or
      two of <em>Dance Dance Revolution</em>.
      Wireless access is always provided;
      BYOWS (Bring your own Web server).
    </p>
    <h2>Directions</h2>
    <p>
      You'll find us right in the center of
      downtown Webville. Come join us!
    </p>
  </body>
</html>
```

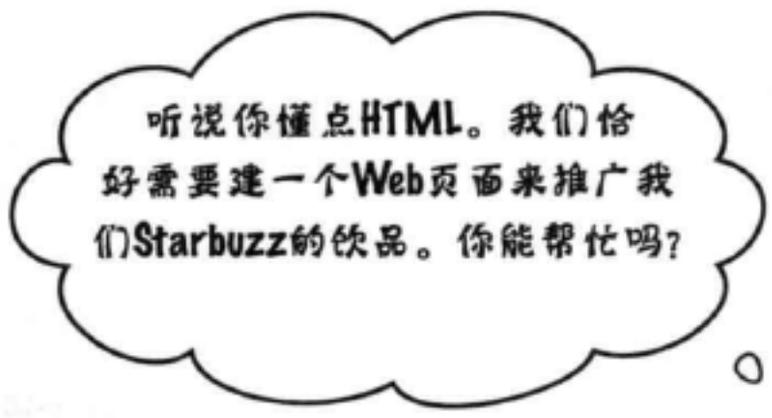
Sharpen your pencil 答案

```
<html>                                                  告诉浏览器HTML从这里开始。

  <head>                                                页面"head"开始（稍后介绍更多有关内容）。

    <title>Head First Lounge</title>                    为页面指定一个标题。

  </head>                                               head结束。

  <body>                                                页面主体（body）开始。

    <h1>Welcome to the Head First Lounge</h1>           告诉浏览器"Welcome to……"是一个标题。

    <img src="drinks.gif">                              在这里放置图像"drinks.gif"。

    <p>                                                 开始一个段落。

      Join us any evening for refreshing elixirs,

      conversation and maybe a game or

      two of <em>Dance Dance Revolution</em>.           强调Dance Dance Revolution。

      Wireless access is always provided;

      BYOWS (Bring your own Web server).

    </p>                                                段落结束。

    <h2>Directions</h2>                                 告诉浏览器"Directions"是一个子标题。

    <p>                                                 开始另一个段落。

      You'll find us right in the center of

      downtown Webville. Come join us!

    </p>                                                段落结束。

  </body>                                               页面主体结束。

</html>                                                 告诉浏览器HTML在这里结束。
```

你在Starbuzz咖啡馆交好运了

Starbuzz之所以名气很大，是因为它是发展最迅速的连锁咖啡店，随处都可以看到Starbuzz连锁店。也许你附近的街角有一家，街对面又能看到另一家。

实际上，Starbuzz发展得太快了，他们甚至没时间来建立自己的Web网站……你的好机会来了。很凑巧，你在Starbuzz买Chai Tea（印度香辣奶茶）时，刚好碰到了Starbuzz的CEO……



Starbuzz的CEO

做出你的决定，
选出你最优先的事情（只能单选）：

- ☐ A. 给狗狗洗个澡。
- ☐ B. 实现收支平衡。
- ☐ C. 抓住这个机会，从Starbuzz开启你崭新的Web职业生涯。
- ☐ D. 与牙医约个时间。

Starbuzz页面上该有些什么内容

太棒了！真高兴你能帮*我们。这是我们希望在第一个页面上发布的内容……

CEO在餐巾纸上潦草地写了些东西，然后把它递给你……

感谢你能助我们一臂之力！我们只需要这个Web页面显示一些简单的内容（请看下面），包括饮料名字、价格和介绍。

House Blend, $1.49
A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

Mocha Cafe Latte, $2.35
Espresso, steamed milk and chocolate syrup.

Cappuccino, $1.89
A mixture of espresso, steamed milk and foam.

Chai Tea, $1.85
A spicy drink made with black tea, spices, milk and honey.

Sharpen your pencil

来看看餐巾纸上写了什么。你能确定它的大致结构吗？换句话说，有没有明显的标题？段落？有没有少点什么，比如页面标题？

在餐巾纸上加些标记（用铅笔），标出你看到的结构，再增加你认为缺少的内容。

答案可以在第1章最后找到。

* 如果你在上一页选择了A、B或者D，建议你把这本书捐给一个好一点的图书馆，或者冬天时拿来生火，也可以直接到Amazon上当二手书卖掉，换点钱回来。

创建Starbuzz web页面

当然，现在唯一的问题是你还没有具体创建过任何Web页面。
不过，这正是你决定深入学习HTML的原因，不是吗？

别担心，以下就是后面几页要做的工作：

1. 用你喜欢的文本编辑器创建一个HTML文件。

2. 录入Starbuzz CEO在餐巾纸上写的菜单。

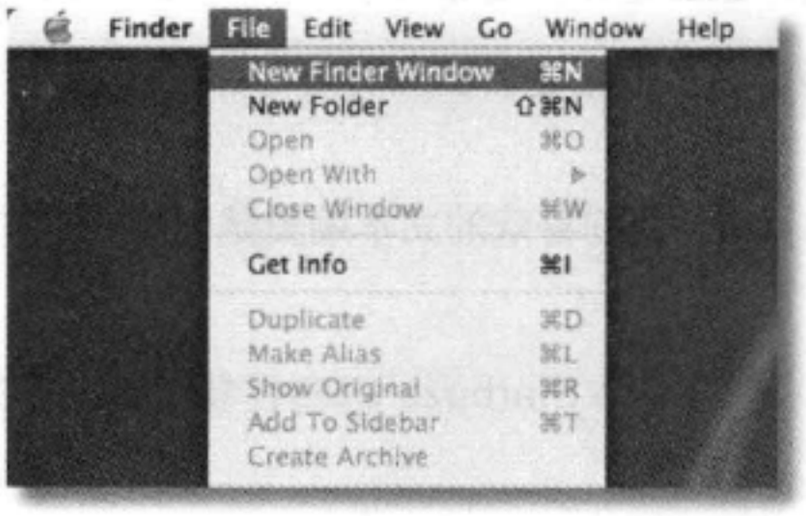
3. 将文件保存为“index.html”。

4. 在你喜欢的浏览器中打开“index.html”文件，然后等待见证奇迹的时刻吧。

创建一个HTML文件(Mac系统)

所有HTML文件都是文本文件。要创建一个文本文件，需要有一个能创建纯文本的应用，而不需要加入大量花哨的格式和特殊字符。你只要纯粹的纯文本。

这本书中，我们会使用Mac系统上的TextEdit。不过，如果你喜欢其他文本编辑器，则完全可以。另外，如果你运行的是Windows系统，则可以跳过这几页，直接看Windows系统的有关说明。



第1步：

导航到你的Applications文件夹。

TextEdit应用在Applications文件夹中。要找到这个文件夹，最容易的办法是从Finder（查找工具）的File菜单选择New Finder Window（新建查找窗口），然后在快捷方式中直接查找Applications（应用）。找到后，单击Applications。

第2步：

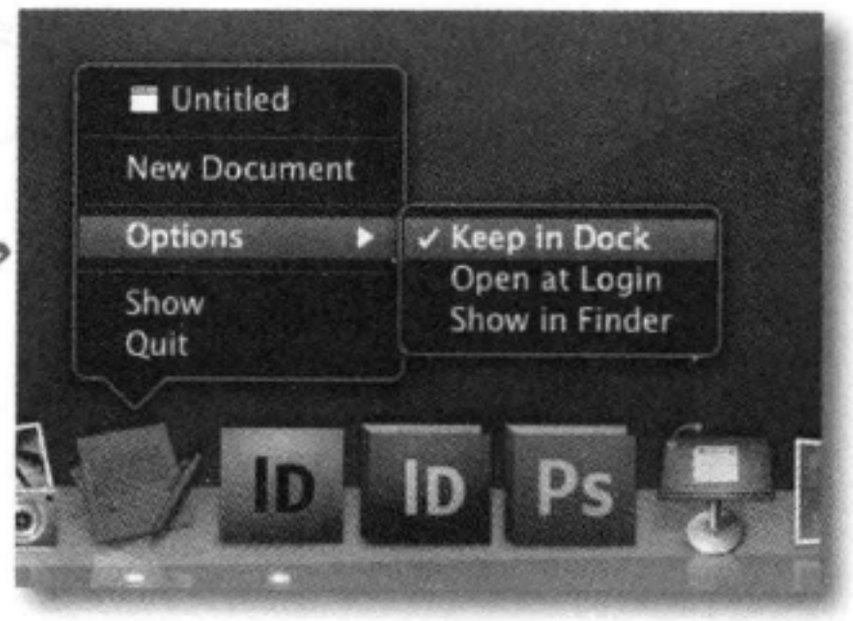
找到并运行TextEdit。

你的Applications文件夹中可能列出了很多应用，可以向下滚动，直到看到TextEdit。双击TextEdit图标，运行这个应用。

第3步（可选）：

让TextEdit一直在浮动条中。

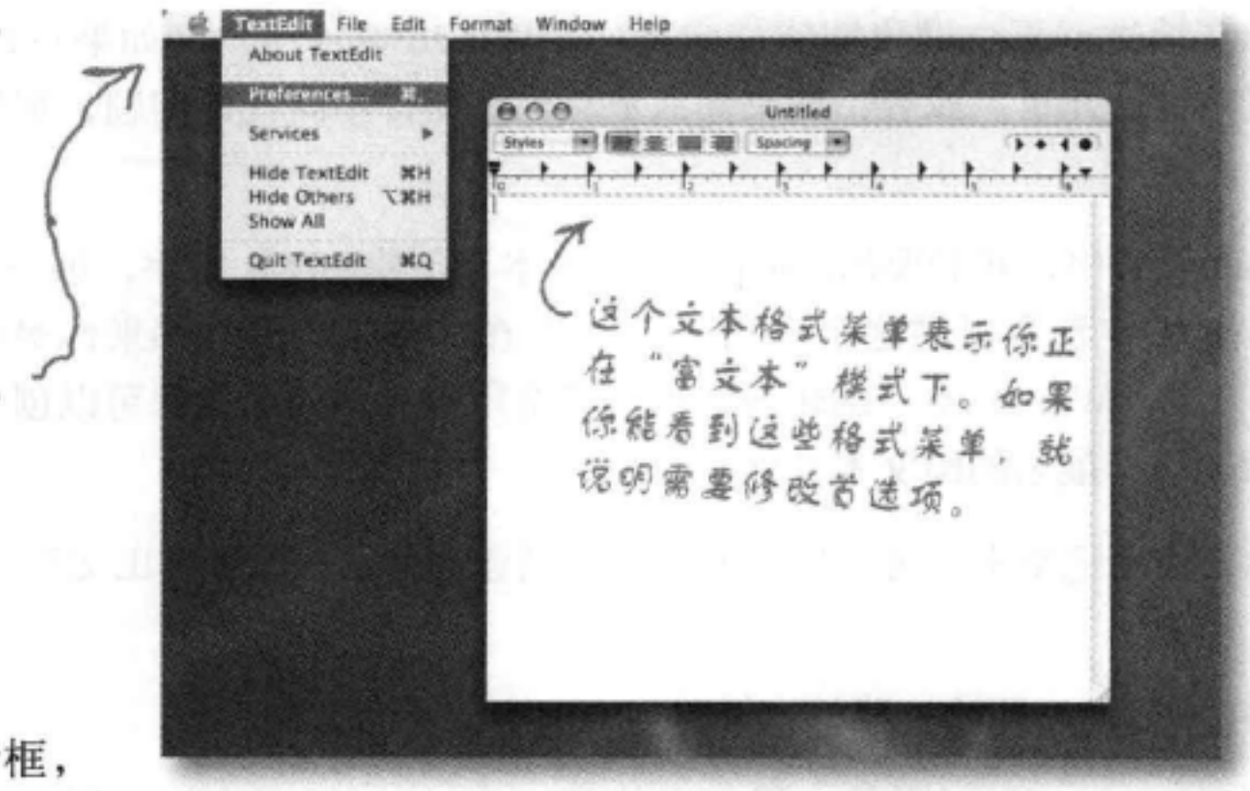
如果你想更为便捷，则可以单击浮动条中的TextEdit图标（一旦TextEdit应用运行，这个图标就会出现）。显示一个弹出菜单时，选择Options（选项），然后选择“Keep in Dock”（保持在浮动条中）。这样一来，这个TextEdit图标就会一直出现在你的浮动条上，你就不用每次需要用到它时，都从Applications文件夹中查找了。



第4步：

修改TextEdit首选项。

默认地，TextEdit会采用“富文本”模式，这表明当你保存文件时，它会向你的文件增加它自己的格式和特殊字符，这可不是你想要的。所以，你需要修改TextEdit首选项，让TextEdit把文件保存为纯文本文件。为此，首先从TextEdit菜单选择菜单项Preferences（首选项）。



第5步：

将首选项设置为纯文本。

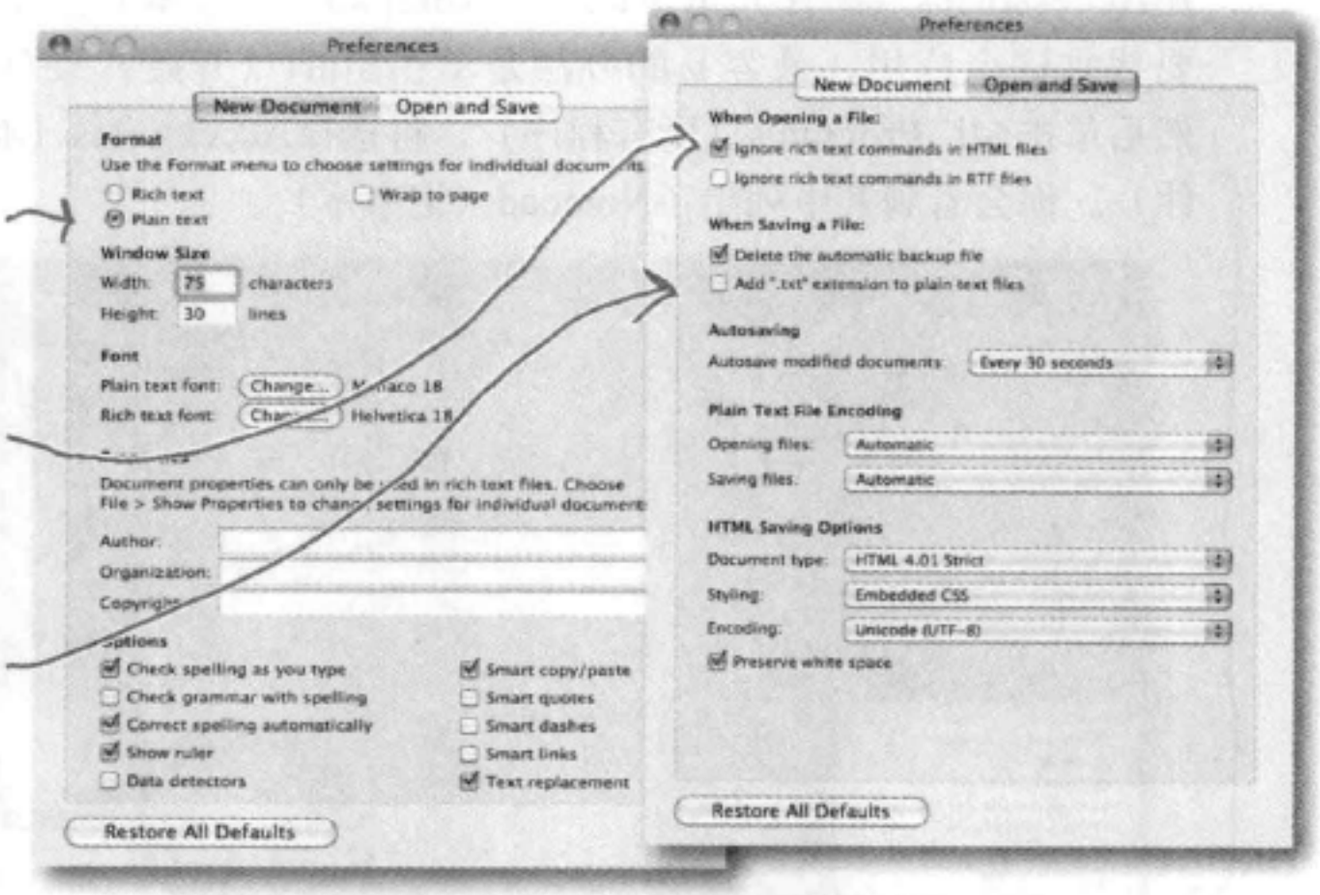
一旦看到首选项（Preferences）对话框，你就要做三件事。

首先，在New Document（新建文档）标签页中选择“Plain text”（纯文本）作为默认的编辑器模式。

然后，在“Open and Save”（打开并保存）标签页中，确保选中“Ignore rich text commands in HTML files”（忽略HTML文件中的富文本命令）。

最后，确保未选中“Add .txt extension to plain text files”（为纯文本文件增加.txt扩展名）。

就这么简单，现在单击左上角的红色按钮，关闭这个对话框。



第6步：

退出并重启。

现在退出TextEdit，从TextEdit菜单选择Quit（退出），然后重启应用。这一次，你会看到窗口最上面不再有复杂的文本格式菜单。你已经准备好了，现在可以创建HTML了。

创建一个HTML文件（Windows系统）

如果你在读这一页，则说明你肯定是一个Windows 7用户。如果不是，则可以跳过这几页。或者，如果你只是想坐在后排不问任何问题，那你可以随意。

或者Windows的其他版本。

在Windows 7中，我们要用Notepad（记事本）创建HTML文件，每一个Windows版本都会提供这个应用，价格公道，易于使用。如果你另有自己喜欢的 Windows 7编辑器，那也完全可以，只要能确保可以创建带“.html”扩展名的纯文本文件。

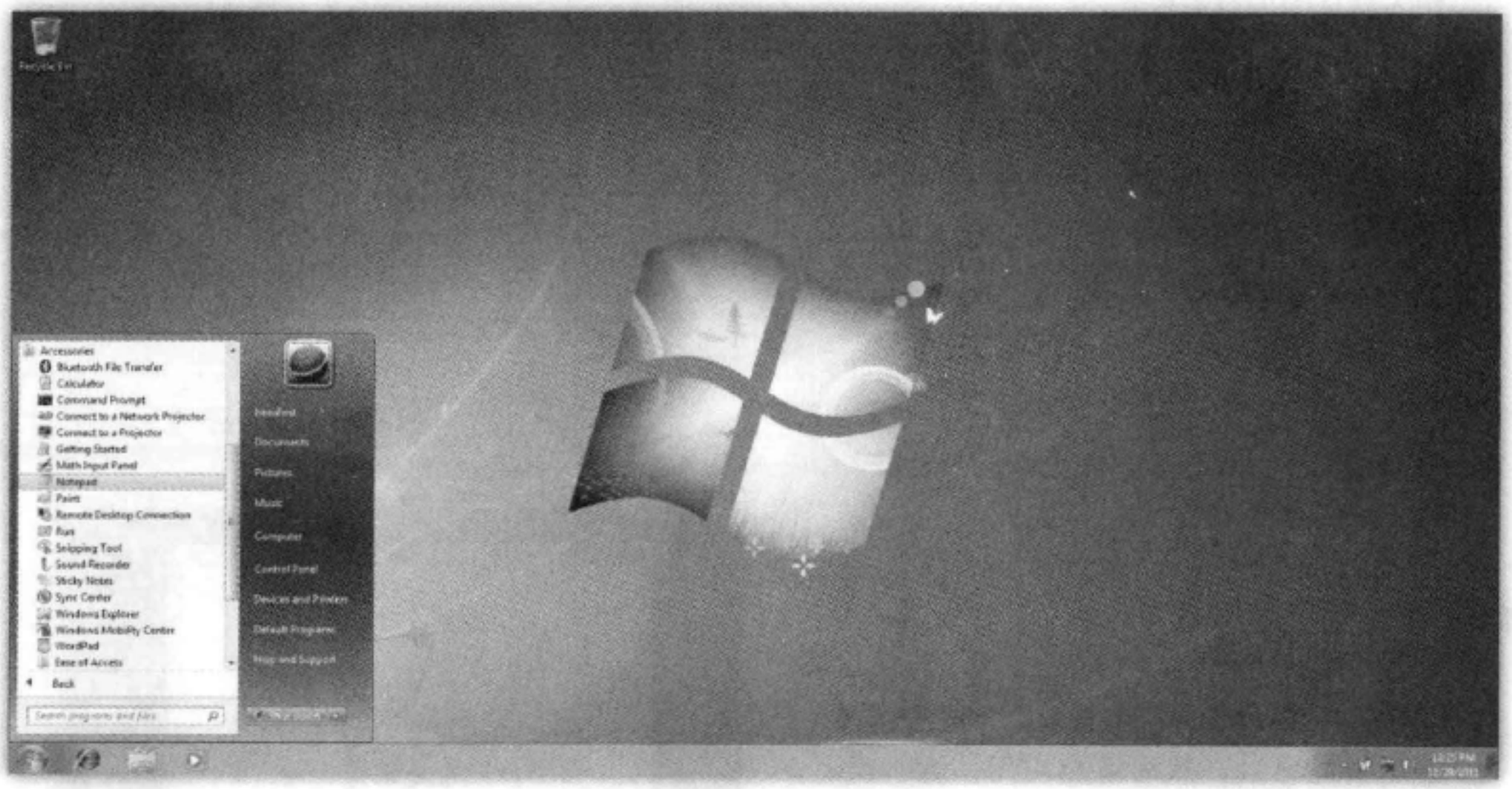
如果你在使用Windows的其他版本，也能找到这个记事本应用。

假设你要使用记事本，可以按下面的步骤创建你的第一个HTML文件。

第1步：

打开Start（开始）菜单，导航到Notepad（记事本）。

在Accessories（附件）里可以找到Notepad（记事本）应用。要找到这个应用，最容易的办法是单击Start（开始）菜单，然后单击All Programs（所有程序），再选择Accessories（附件）。你会看到其中列出了Notepad（记事本）。



第2步：

打开记事本。

一旦在Accessories（附件）文件夹找到记事本，单击这个图标，你就会看到一个空窗口，可以在这个窗口中输入HTML了。

不过建议最好完成这一步。

第3步：（可选）

不要隐藏已知文件类型的扩展名。

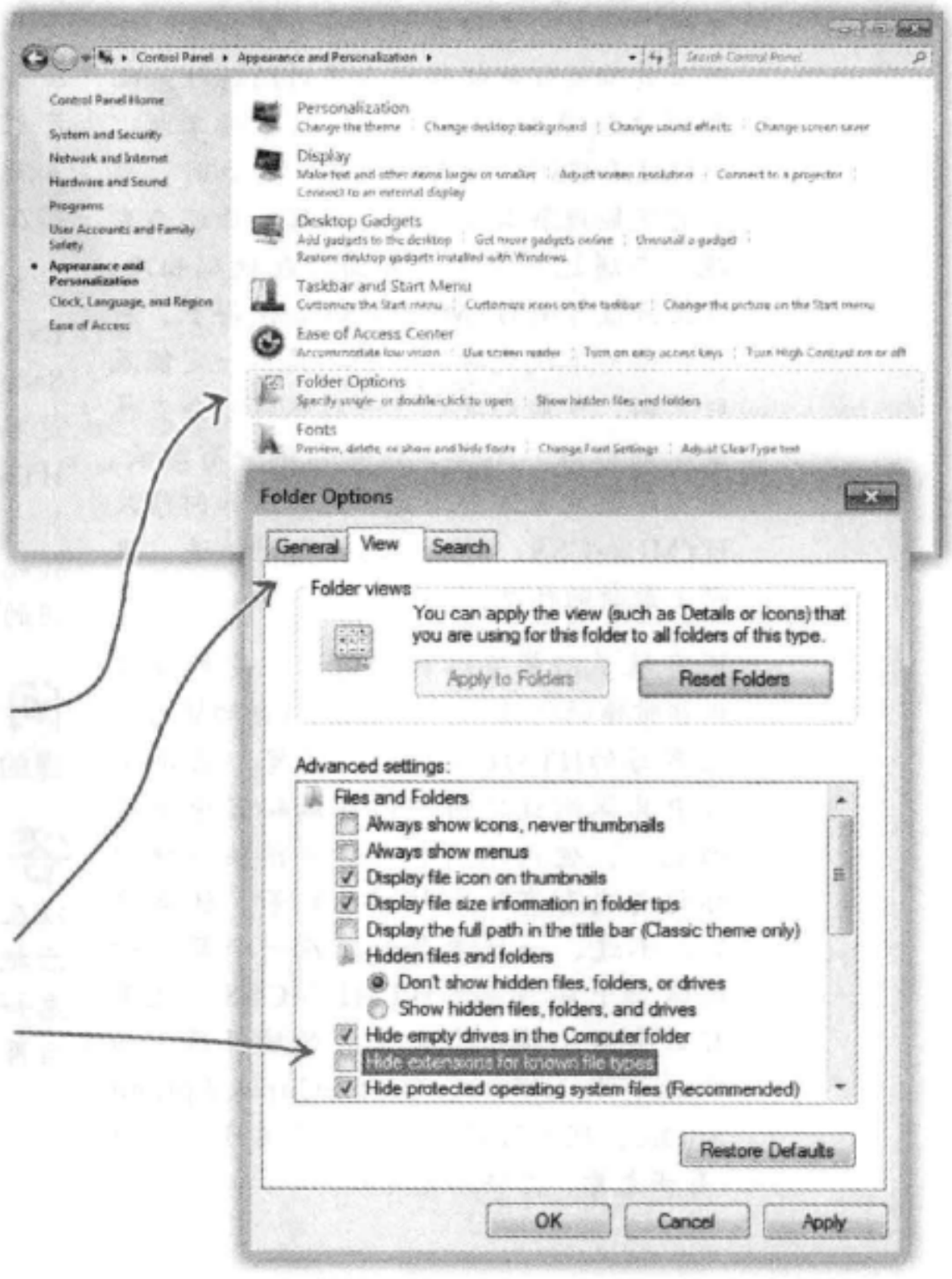
默认地，Windows File Explorer（文件资源管理器）会隐藏已知文件类型的文件扩展名。例如，一个名为“Irule.html”的文件在资源管理器中可能显示为“Irule”，而没有“.html”扩展名。

这可能会带来困惑，如果Windows显示这些扩展名，就不会那么让人糊涂了，所以下面改变你的文件夹选项，使文件扩展名可见。

首先，打开Folder Options（文件夹选项），单击Start（开始）按钮，再单击Control Panel（控制面板），接下来单击“Appearance and Personalization”（外观和个性化），然后单击Folder Options（文件夹选项）。

接下来，在View（查看）标签页中，在“Advanced settings”（高级设置）中向下滚动，直到看到“Hide extensions for known file types”（隐藏已知文件类型的扩展名），取消选中这个选项。

就这么简单。单击OK按钮保存首选项，现在就能在资源管理器中看到文件扩展名了。



there are no Dumb Questions

问：为什么我要使用简单的文本编辑器？难道不能用那些强大的工具创建Web页面吗？比如Dreamweaver和Expression Web？

答：你为什么要读这本书？是因为你想了解创建Web页面使用的具体技术，对不对？尽管那些工具确实都非常棒，但是它们会越俎代庖，很多事情都帮你做，尽管你省事了，但是学不到多少东西。在你真正掌握了HTML和CSS之前，应该先好好学学这些内容，而不要完全依赖一个“能干的”工具。

不过，一旦你已经成为这方面的行家，这些工具确实就能提供很多不错的特性，比如语法检查和预览。如果你已经掌握了HTML和CSS，查看“代码”窗口时，则你肯定能理解其中的所有内容，你还会发现，与通过一个用户界面修改代码相比，直接修改原始HTML和CSS会快得多。另外，当标准改变时，那些工具不一定能及时更新，可能到发布下一版本时才会支持最新的标准。不过，这对你不会有影响，因为即使没有这个工具你也知道如何修改HTML和CSS，所以你总能与时俱进，跟得上最新的标准。

还有很多功能完备的编辑器，它们能提供非常棒的特性，如剪贴板（自动插入你经常写的HTML片段）、预览（在浏览器中具体测试之前可以在编辑器中直接预览）、使用不同颜色区分语法（使得标记与内容有不同的颜色）等，还有很多。不过，一旦学会如何在一个简单的编辑器中编写基本HTML和CSS，就很有必要再试试一个更强大的编辑器，如Coda、TextMate、CoffeeCup或Aptana Studio。现在有很多这样的工具可供选择（有些免费，有些不免费）。

问：我有编辑器了，不过我要用哪一个浏览器呢？有好多浏览器——Internet Explorer、Chrome、Firefox、Opera、Safari，怎么选择啊？

答：简单的回答是，可以用你喜欢的任何浏览器。HTML和CSS是行业标准，这说明所有浏览器都会尽量以相同的方式支持HTML和CSS（只是为了得到最好的支持，要确保使用浏览器的最新版本）。

复杂的答案是，事实上，不同浏览器处理页面的方式存在细微的差别。如果你的用户可能用不同的浏览器访问你的页面，一定要在多个不同的浏览器中进行测试。有些页面在各种浏览器中看起来都一样，但有些可能并非如此。对HTML和CSS越深入,这些细微差别对你的影响就越大，我们将在这本书中讨论这些细小的差异。

大多数例子在所有主流浏览器（Internet Explorer、Chrome、Firefox、Opera和Safari）上都能正常工作（除非特别说明）。这些都是现代浏览器，提供了很好的HTML和CSS支持。作为一个Web开发人员，你可能需要在多个浏览器中测试你的代码，所以建议你下载并了解至少两个不同的浏览器！

问：这些文件是在我自己的计算机上创建的，怎么在网上浏览这些页面呢？

答：这是HTML的一个过人之处，你可以在自己的计算机上创建和测试文件，然后把它们发布到Web上。目前，我们只考虑如何创建文件以及文件中有些什么，稍后再介绍如何在Web上发布。

现在，再回到Starbuzz……

好了，你已经知道创建纯文本文件的基础知识，现在只需要在文本编辑器里放一些内容，保存文件，然后加载到你的浏览器里。

首先输入CEO在餐巾纸上写的那些饮料，这些饮料就是页面的内容。你要增加一些HTML标记，让这些内容有点结构，不过现在只需要输入基本内容。准备好了吧，首先在文件最上面增加“Starbuzz Coffee Beverages”。

像这样输入餐巾纸上的信息。

Untitled - Notepad

File Edit Format View Help

```
Starbuzz Coffee Beverages

House Blend, $1.49
A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

Mocha Cafe Latte, $2.35
Espresso, steamed milk and chocolate syrup.

Cappuccino, $1.89
A mixture of espresso, steamed milk and foam.

Chai Tea, $1.85
A spicy drink made with black tea, spices, milk and honey.
```

Windows

Untitled.txt

```
Starbuzz Coffee Beverages

House Blend, $1.49
A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

Mocha Cafe Latte, $2.35
Espresso, steamed milk and chocolate syrup.

Cappuccino, $1.89
A mixture of espresso, steamed milk and foam.

Chai Tea, $1.85
A spicy drink made with black tea, spices, milk and honey.
```

Mac

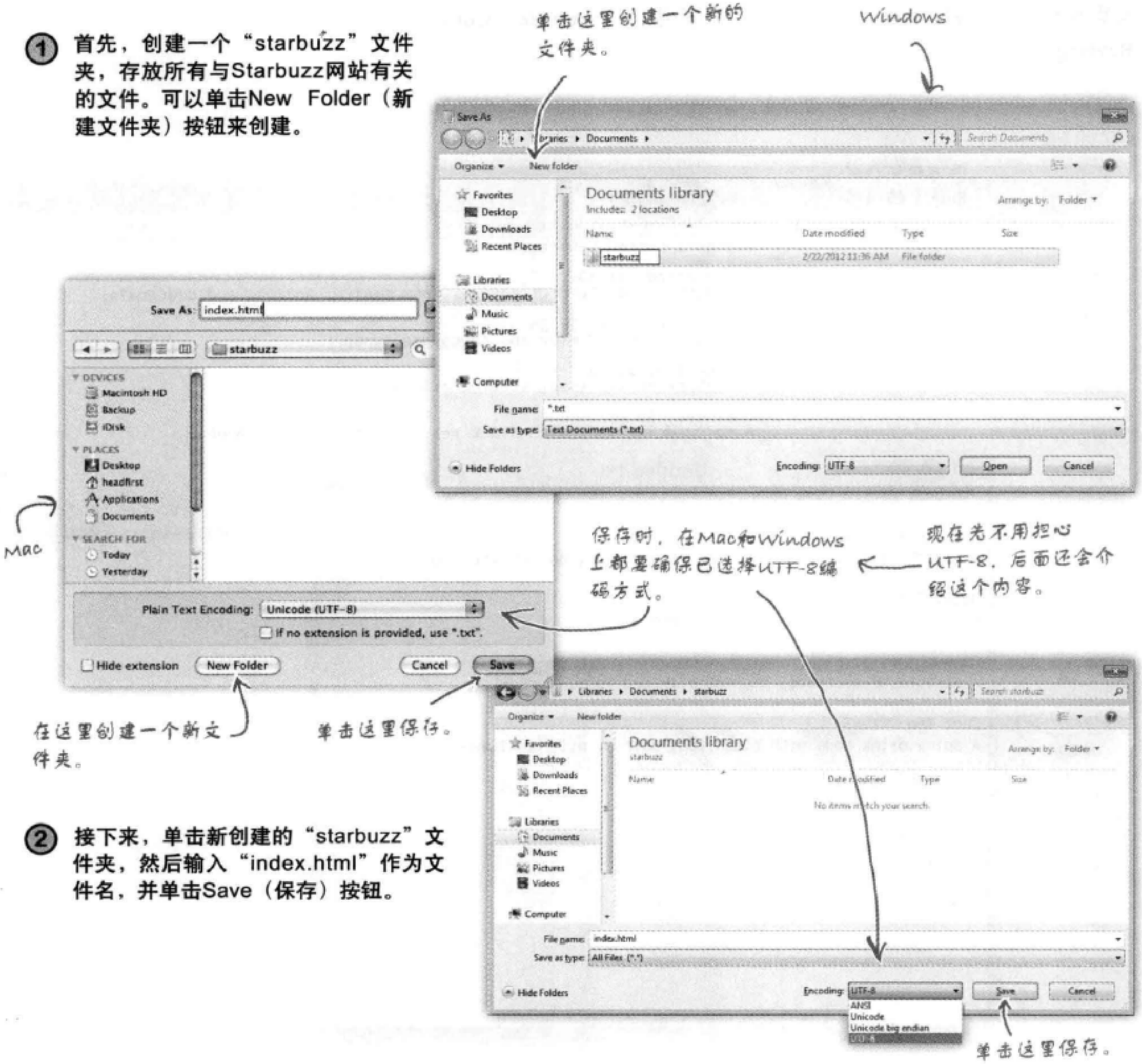
保存你的成果

一旦输入CEO在餐巾纸上写的饮料，接下来就要把你的成果保存到一个名为“index.html”的文件中。保存之前，可能需要创建一个文件夹，名为“starbuzz”，用来存放这个网站的文件。

首先，从File（文件）菜单选择Save（保存），你会看到一个Save As（另存为）对话框。接下来，按下面的步骤来做：

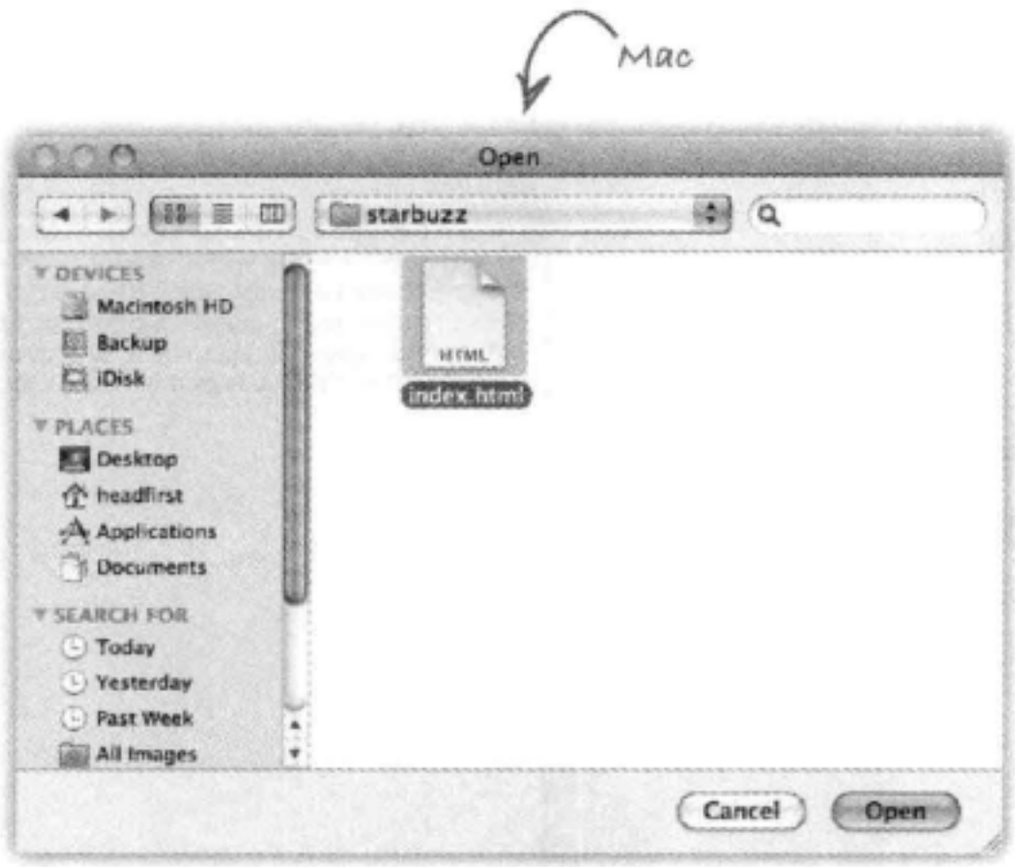
① **首先，创建一个“starbuzz”文件夹，存放所有与Starbuzz网站有关的文件。可以单击New Folder（新建文件夹）按钮来创建。**

② **接下来，单击新创建的“starbuzz”文件夹，然后输入“index.html”作为文件名，并单击Save（保存）按钮。**



在浏览器中打开你的Web页面

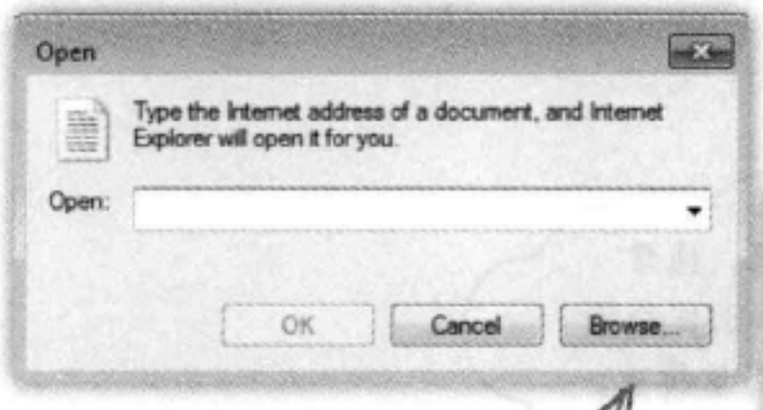
是不是准备打开你的第一个Web页面？可以用你喜欢的浏览器，从File（文件）菜单选择“Open File…”（打开文件……），如果使用 Windows 7和Internet Explorer，则要选择“Open…”（打开……），导航到你的“index.html”文件。选择这个文件并单击Open（打开）按钮。



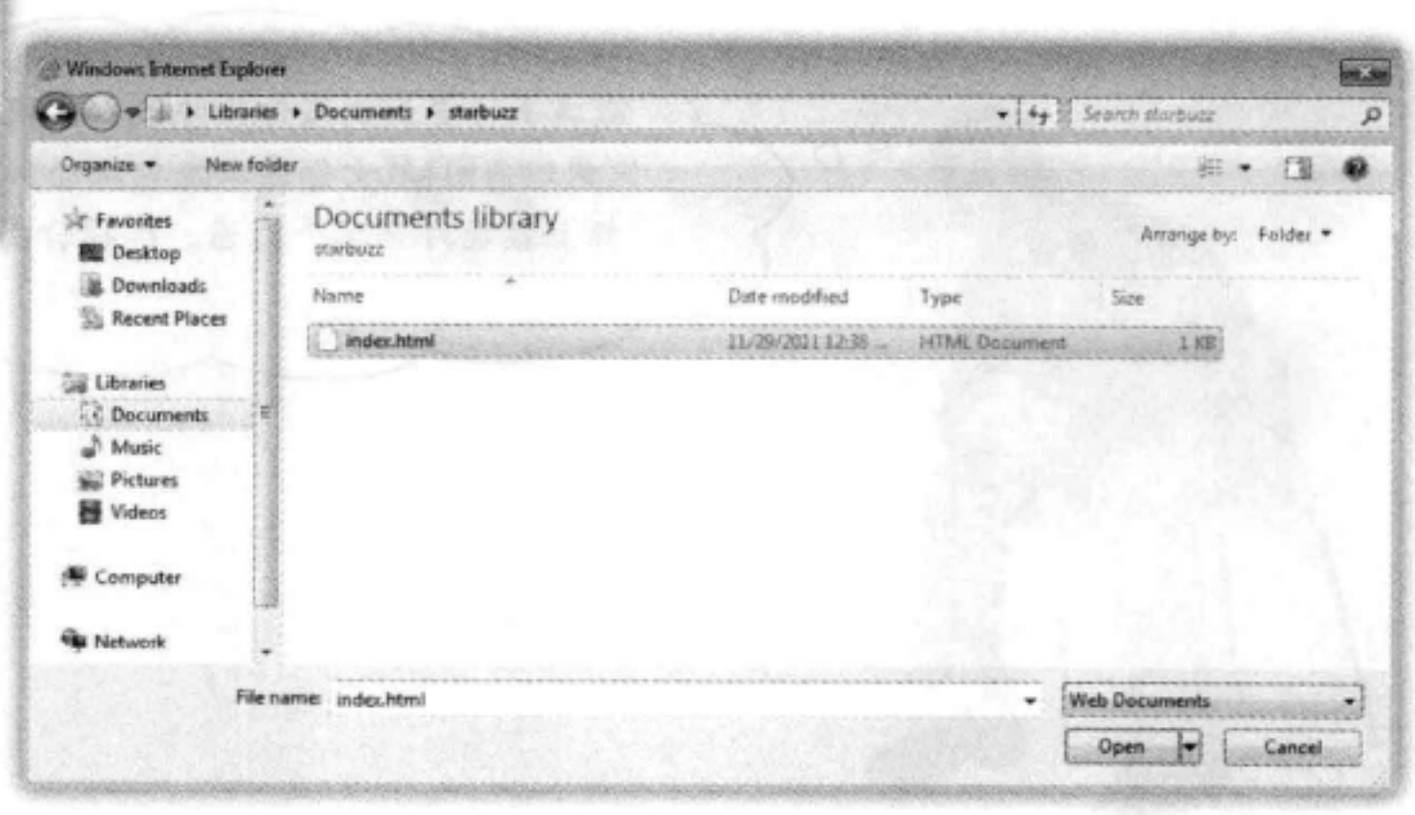
在Mac上，导航到你的文件，然后单击这个文件的图标选中它，再单击Open（打开）按钮。

在Windows Internet Explorer中，这个过程包括两步。首先，打开Open（打开）对话框。

Windows

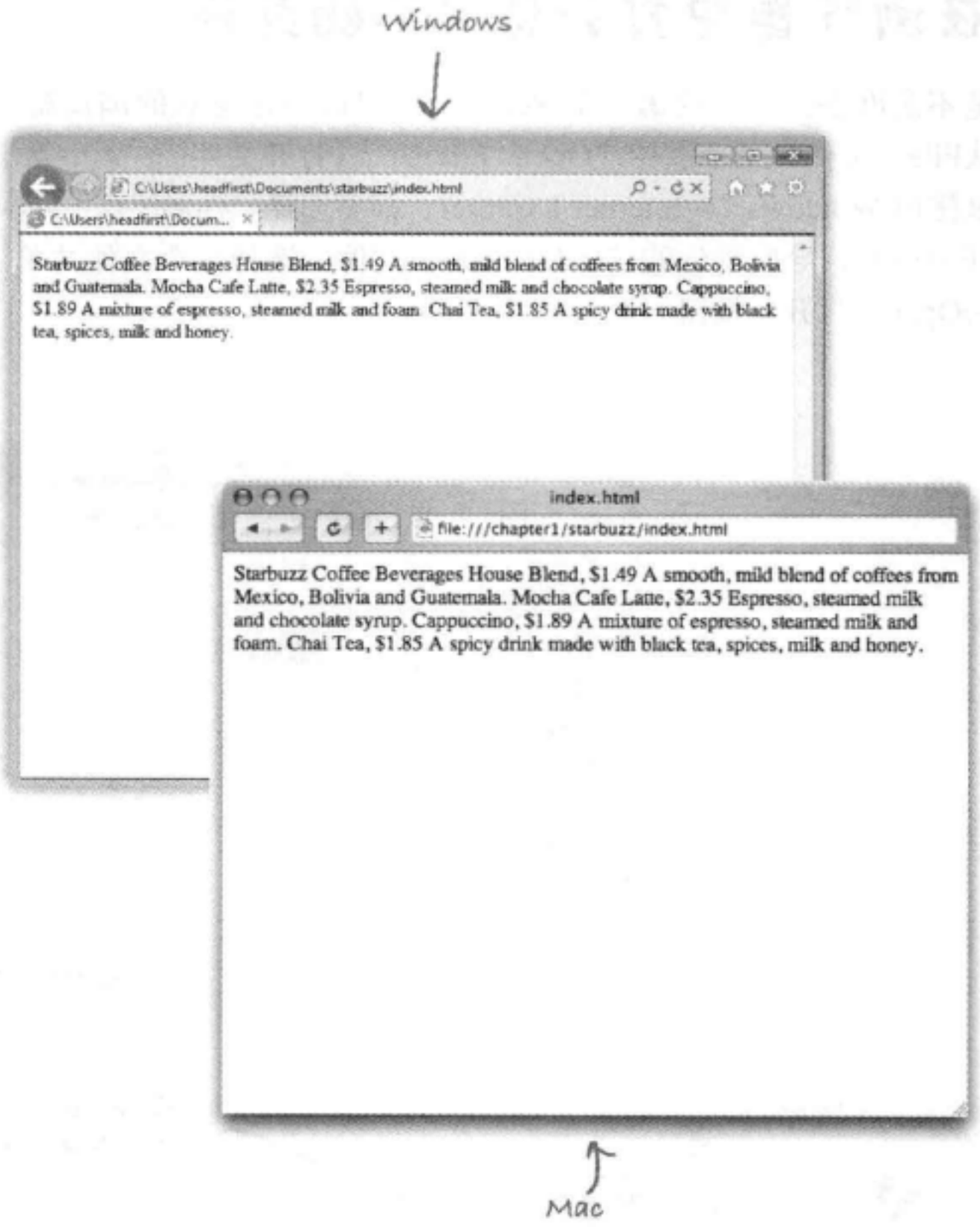


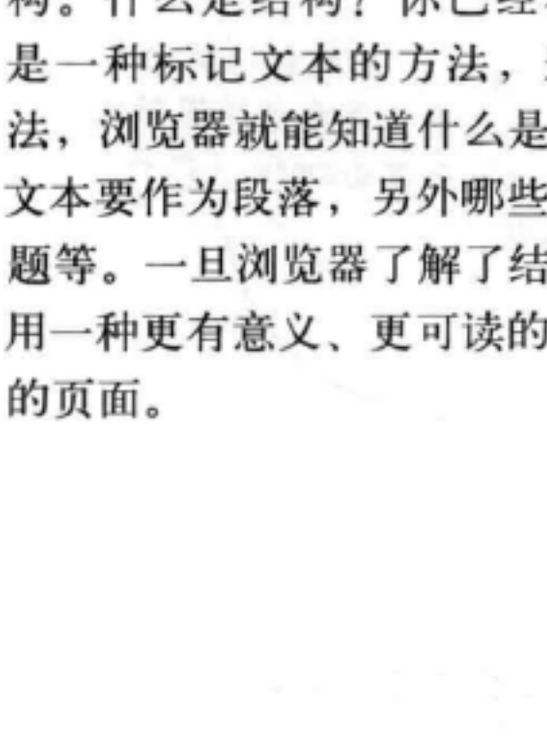
然后单击Browse（浏览）按钮，打开一个浏览对话框，再导航到保存文件的位置。



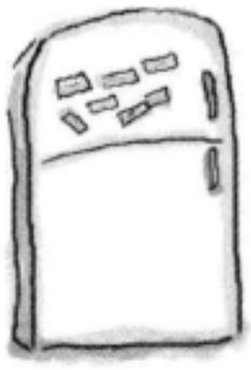
测试你的页面

成功了！你已经在浏览器中加载了页面，不过看起来有点……嗯……不太让人满意。这是因为，你目前所做的只是一些最基本的工作，仅仅创建了一个页面，再在浏览器中查看，仅此而已。到目前为止，你只是输入了Web页面的内容。现在该HTML发挥作用了。利用HTML，你可以告诉浏览器页面的结构。什么是结构？你已经看到了，这是一种标记文本的方法，通过这种方法，浏览器就能知道什么是标题，哪些文本要作为段落，另外哪些文本是子标题等。一旦浏览器了解了结构，它就能用一种更有意义、更可读的方式显示你的页面。





取决于你使用的操作系统和浏览器，通常只需双击HTML文件或者把它拖动到浏览器图标上就能打开这个页面。这样会简单得多。



标记磁贴

好吧，现在来增加结构……

你的任务是为Starbuzz餐巾纸上的文本增加结构。使用这一页最下面的冰箱磁贴标记文本，指示哪些部分是标题、子标题和段落文本。我们已经帮你放好了几个磁贴。完成这个任务并不需要用到下面的全部磁贴，有些磁贴可能用不上。

<h1> Starbuzz Coffee Beverages </h1>

House Blend, $1.49

A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

<h2> Mocha Cafe Latte, $2.35 </h2>

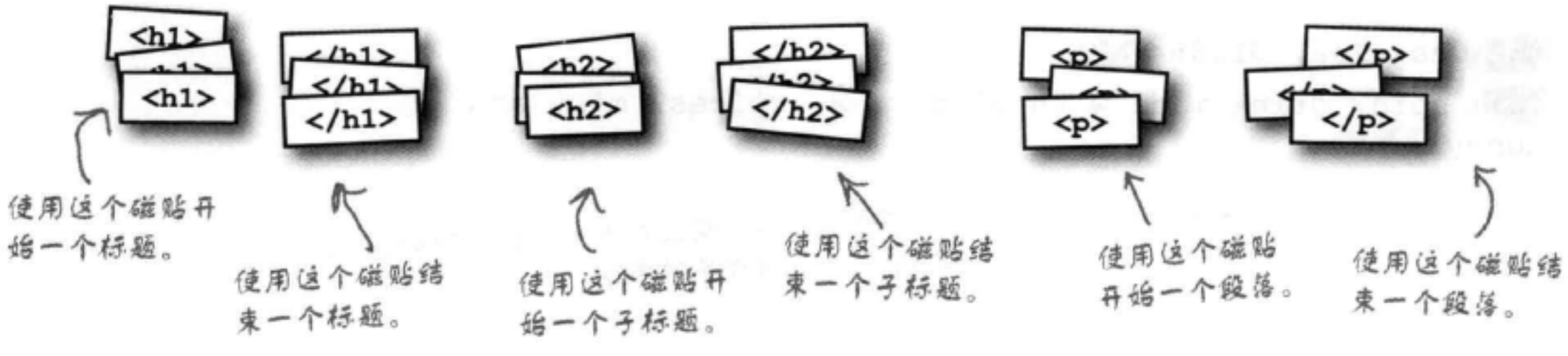
 Espresso, steamed milk and chocolate syrup.

Cappuccino, $1.89

A mixture of espresso, steamed milk and foam.

Chai Tea, $1.85

A spicy drink made with black tea, spices, milk and honey.



祝贺你，你已经写出你的第一个HTML！

看起来与冰箱磁贴好像差不多，不过你确实已经用HTML对文本做了标记。只不过可以看到，我们通常会把这些磁贴称为标记（tag）。检查下面的标记，与上一页中的磁贴做个比较。

使用<h1>和</h1>标记来标记标题。这两个标记之间的所有文本就是标题的具体内容。

```
<h1>Starbuzz Coffee Beverages</h1>

<h2>House Blend, $1.49</h2>
<p>A smooth, mild blend of coffees from Mexico, Bolivia
and Guatemala.</p>

<h2>Mocha Cafe Latte, $2.35</h2>
<p>Espresso, steamed milk and chocolate syrup.</p>

<h2>Cappuccino, $1.89</h2>
<p>A mixture of espresso, steamed milk and foam.</p>

<h2>Chai Tea, $1.85</h2>
<p>A spicy drink made with black tea, spices, milk and
honey.</p>
```

<h2>和</h2>标记包围一个子标题。可以把<h2>标记看作是<h1>标题的子标题。

和 标记包围一个文本块，这是一个段落。其中可以有一个或多个句子。

注意不用把匹配的标记放在同一行上。可以根据需要在它们之间放任意多的内容。

完工了吗？

你已经有了一个带标记的HTML文件，这就是一个Web页面吗？差不多吧。你已经见过<html>、<head>、<title>和<body>标记，现在只需要增加这些标记，让它变成一个一流的HTML页面……

首先，用<html>和</html>标记包围你的HTML。这会告诉浏览器文件的内容是HTML。

接下来增加<head>和</head>标记。首部（head）包含Web页面的有关信息，如页面标题。现在可以这样考虑：通过首部可以告诉浏览器关于Web页面的信息。

在head标记中放入title标记。title总出现在浏览器窗口的顶部。

```
<html>

  <head>
    <title>Starbuzz Coffee</title>
  </head>

  <body>
    <h1>Starbuzz Coffee Beverages</h1>
    <h2>House Blend, $1.49</h2>
    <p>A smooth, mild blend of coffees from Mexico,
       Bolivia and Guatemala.</p>

    <h2>Mocha Cafe Latte, $2.35</h2>
    <p>Espresso, steamed milk and chocolate syrup.</p>

    <h2>Cappuccino, $1.89</h2>
    <p>A mixture of espresso, steamed milk and foam.</p>

    <h2>Chai Tea, $1.85</h2>
    <p>A spicy drink made with black tea, spices,
       milk and honey.</p>
  </body>

</html>
```

首部包括<head>和</head>标记以及它们之间的所有内容。

页面主体包括<body>和</body>标记以及它们之间的所有内容。

页面主体包含Web页面的所有内容和结构，这就是你在浏览器中看到的部分。

另一个测试

修改你的“index.html”文件，在文件中增加<head>、</head>、<title>、</title>、<body>和</body>标记。一旦完成，就保存所做的修改，在浏览器中重新加载这个文件。

重新加载index.html文件，可以再次选择Open File（打开文件）菜单项，或者使用浏览器的重新加载按钮。

注意标题（你在<head>元素中指定的内容）会出现在这里。

现在看起来好一点了。浏览器会解释你的标记来提供页面显示，现在页面不仅有了结构，而且更可读。

标记剖析

好的，你已经见过不少标记，下面在显微镜下看一看这些标记究竟是如何工作的。

通常要用标记包围一些内容。这里我们使用标记告诉浏览器“Starbuzz Coffee Beverages”是一个顶级标题（即一级标题）。

这是结束标记，指示标题结束。在这里，</h1>标记会结束一个<h1>标题。之所以知道这是一个结束标记，是因为它在内容的后面出现，而且在“h1”前面还有一个“/”。所有结束标记都包含一个“/”。

这是开始标记，指示标题开始。

```
<h1> Starbuzz Coffee Beverages </h1>
```

标记包含标记名，标记名两边有尖括号（即<和>字符）。

整个称为一个元素。在这里，我们可以把它叫做<h1>元素。元素由包围内容的标记和标记之间的内容组成。

我们把开始标记和对应的结束标记称为匹配标记。

要告诉浏览器页面的结构，需要用成对的标记包围页面内容。

要记住：

元素 = 开始标记 + 内容 + 结束标记

there are no Dumb Questions

问：这么说匹配标记不需要在同一行上？

答：没必要。记住，浏览器并不关心制表符、回车和大多数空格。所以，标记可以在同一行上的任何位置开始和结束，或者也可以在不同行上开始和结束。只是要确保必须以一个开始标记开始，如<h2>，并以一个结束标记结束，如</h2>。

问：为什么结束标记要有一个额外的“/”？

答：结束标记中的“/”是为了让你和浏览器知道某个特定内容在哪里结束。否则，结束标记看上去就和开始标记一样了，是不是？

问：我注意到有些页面的HTML中开始标记和结束标记并不一定匹配。

答：嗯，标记原本应该匹配的。一般来讲，即使你写了不正确的HTML，浏览器也能很好地猜出你原本的意思。不过你会看到，如今写完全无误的HTML大有好处。如果你担心写不出完美的HTML，则大可不必担心。把代码发布到Web服务器展示给全世界之前，有很多工具可以帮你验证代码。现在你只需要养成好习惯，保证开始标记与结束标记总是匹配的。

问：嗯，那么这个休闲室例子里的<img src="drinks.gif">标记是怎么回事？你是不是忘记加结束标记了？

答：哦，你真厉害，一眼就发现了。有些元素会使用一种简写记法，只有一个标记。现在先不去管它，后面有一章会专门介绍这个内容。

问：元素就是一个开始标记 + 内容 + 结束标记，那么，是不是不能把标记嵌在其他标记里？比如<head>和<body>怎么能在<html>标记里呢？

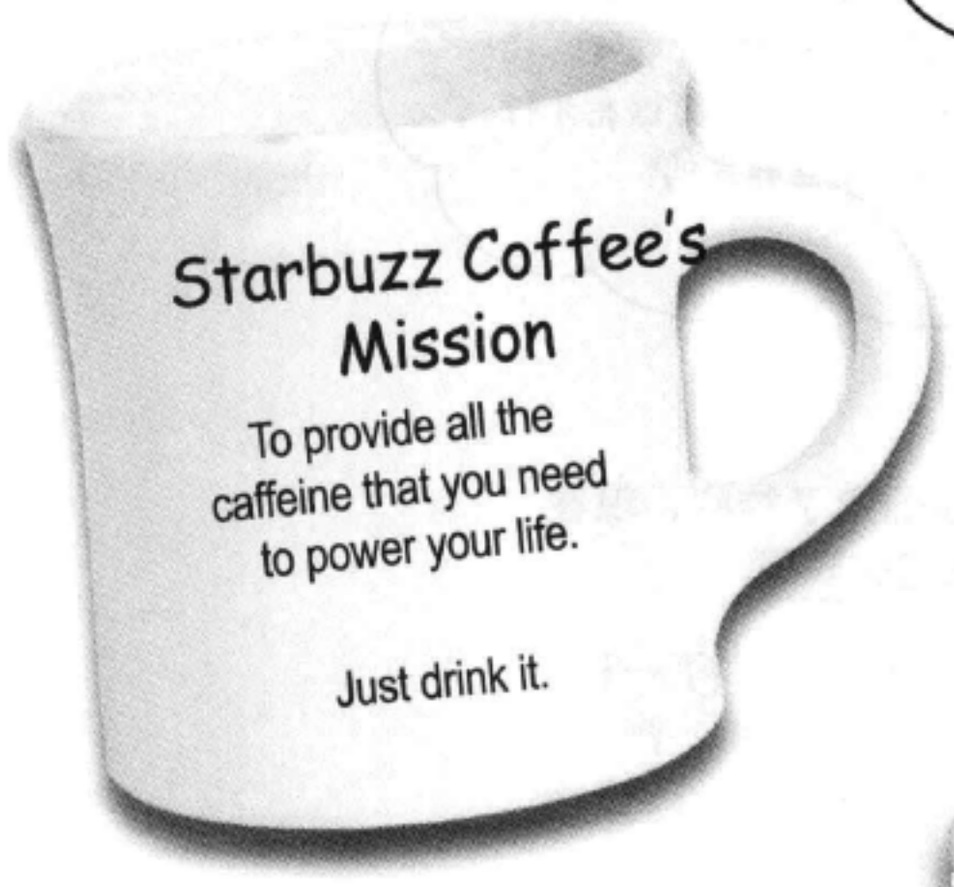
答：这是可以的，HTML标记通常都会像这样“嵌套”。好好想想，一个HTML页面有一个主体，其中包含一个段落，依此类推，这是很自然的啊。所以，很多HTML元素的标记之间都包含有其他HTML元素。后面有几章还会详细讨论这个问题，不过现在只需要注意页面中元素相互之间是如何关联的。

BRAIN POWER

标记可能比你目前为止看到的更有趣一些。下面是一个段落标记，还增加了一点额外的内容。你觉得这会用来做什么？

```
<p id="houseblend">A smooth, mild
blend of coffees from Mexico, Bolivia
and Guatemala.</p>
```

Exercise

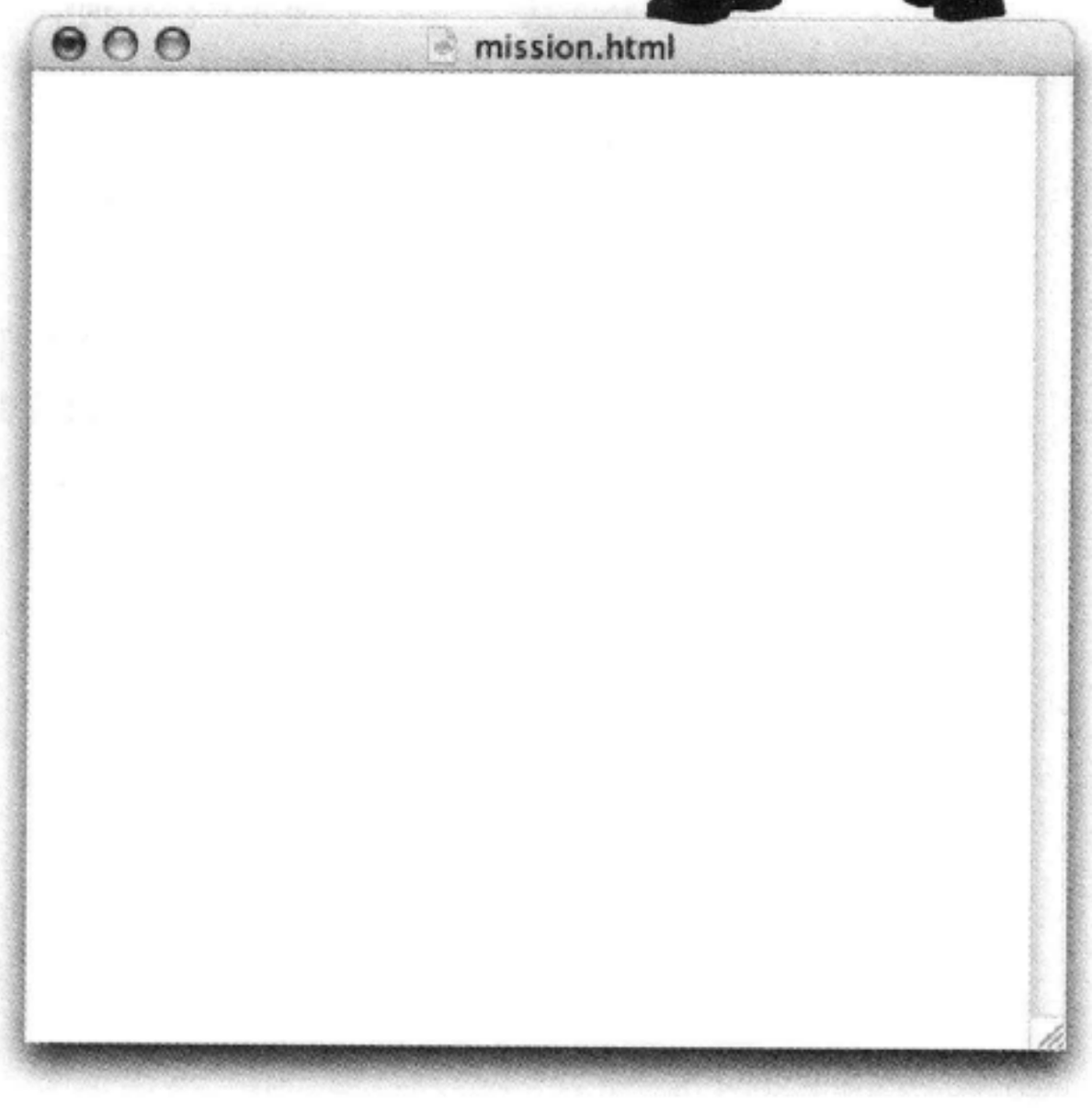


❶ 把新页面“mission.html”的HTML写在这里。

❷ 在一个文本编辑器里输入你的HTML，保存为“mission.html”，放在“index.html”文件所在的同一个文件夹下。

❸ 完成后，在浏览器里打开“mission.html”。

❹ 继续下一步之前，对照这一章最后给出的页面检查你的工作……



好的。我们已经确定了结构，现在来集中精力关注它的表现。

你已经知道了，HTML提供了一种方法来描述文件中内容的结构。浏览器显示HTML时，它会使用它自己内置的默认样式来表现这个结构。不过，如果完全依赖于浏览器，那么显然你没办法得到“本月最佳设计”的称号。

这里CSS就要发挥作用了。利用CSS，可以描述要如何表现你的内容。下面就来动手试试看，让这个Starbuzz页面看起来更为美观（同时开启你的Web职业生涯）。

CSS是层叠样式表（Cascading Style Sheets）的缩写。稍后会详细介绍这些是什么意思，不过现在只需要知道CSS提供了一种方法来告诉浏览器页面中的元素如何显示。

认识style元素

要增加样式，需要在页面中增加一个新的（和我们一起念）E-L-E-M-E-N-T，这就是<style>元素。下面再回到Starbuzz主页面，增加一些样式。试试看……

<style>元素放在HTML的首部里。

与其他元素类似，<style>元素有一个开始标记<style>，还有一个结束标记</style>。

<style>标记还有一个（可选的）属性，名为type，它能告诉浏览器你在使用什么类型的样式。由于要使用CSS，所以可以指定“text/css”类型。

在这里定义页面的样式。

```
<html>
  <head>
    <title>Starbuzz Coffee</title>
    <style type="text/css">


    </style>
  </head>
  <body>
    <h1>Starbuzz Coffee Beverages</h1>

    <h2>House Blend, $1.49</h2>
    <p>A smooth, mild blend of coffees from Mexico, Bolivia and
    Guatemala.</p>

    <h2>Mocha Caffe Latte, $2.35</h2>
    <p>Espresso, steamed milk and chocolate syrup.</p>

    <h2>Cappuccino, $1.89</h2>
    <p>A mixture of espresso, steamed milk and milk foam.</p>

    <h2>Chai Tea, $1.85</h2>
    <p>A spicy drink made with black tea, spices, milk and honey.</p>
  </body>
</html>
```

there are no Dumb Questions

问：元素可以有“属性”？这是什么意思？

答：通过属性，可以提供一个元素的附加信息。比如说，如果有一个<style>元素，属性允许你准确地指定是什么类型的样式。以后还会看到不同元素的更多属性：你只要记住一点，属性能提供元素的一些额外信息。

问：为什么必须指定样式类型（“text/css”）作为style元素的一个属性？难道还有其他类型的样式吗？

答：从前，HTML的设计者认为以后应该还会有其他样式，不过如今我们已经醒悟，事实表明，完全可以只使用<style>而不带类型属性，所有浏览器都知道你指的是CSS。很失望吧，让我们屏住呼吸，拭目以待<style type="50sKitsch">样式的到来吧。呵呵，开玩笑的。

给Starbuzz网站加点样式……

既然HTML首部里有了一个<style>元素，现在要做的就是提供一些CSS，让页面炫一些。你会发现，下面已经为你“烘制”好了一些CSS。看到成品标志时，就说明你看到的HTML和CSS可以原样输入。相信我们。后面还会学习这些标记是如何工作的，不过先看看它们能做什么。

现在先来看看这个CSS，再把它增加到“index.html”文件。输入这个CSS之后，保存文件。

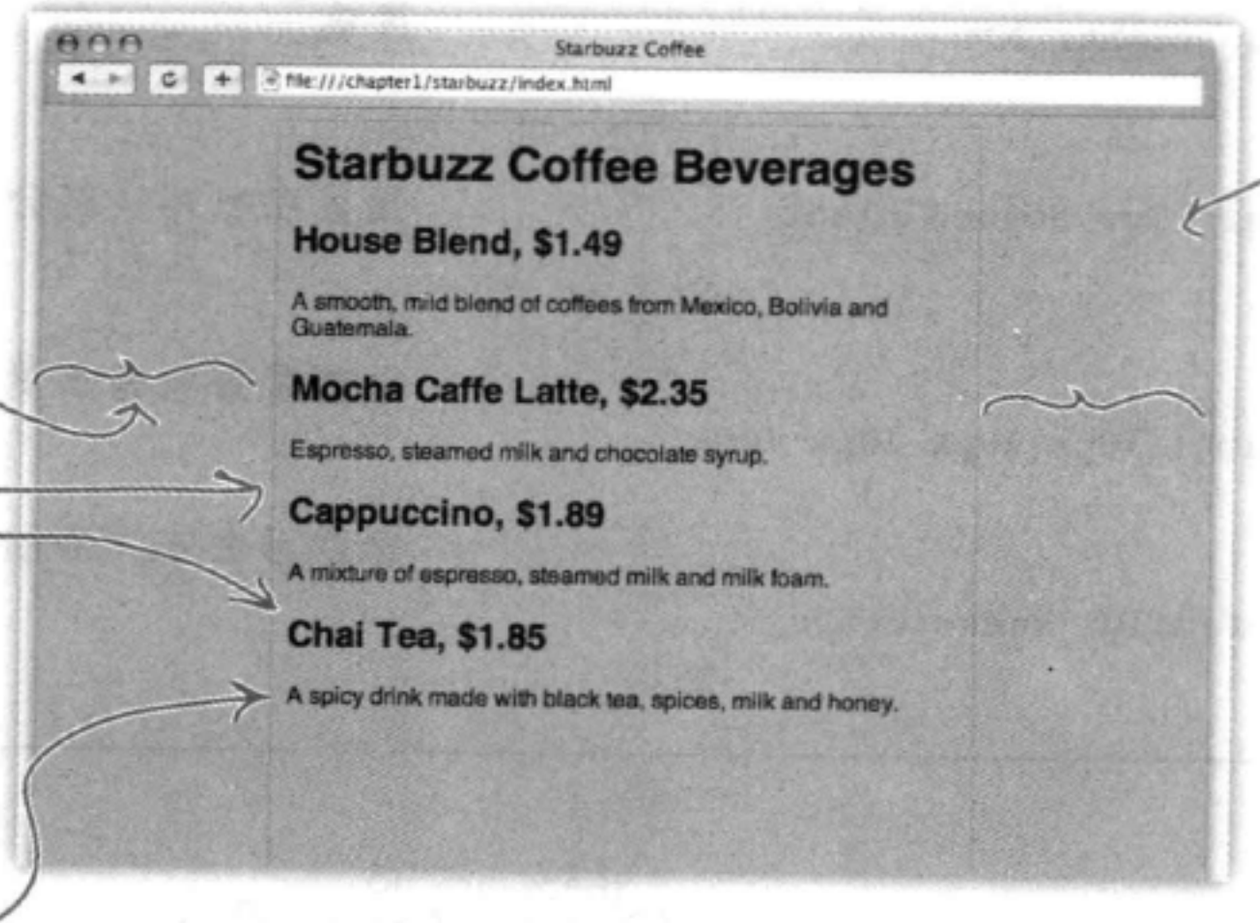
成品CSS

```
<html>
  <head>
    <title>Starbuzz Coffee</title>
    <style type="text/css">
      body {
        background-color: #d2b48c;
        margin-left: 20%;
        margin-right: 20%;
        border: 2px dotted black;
        padding: 10px 10px 10px 10px;
        font-family: sans-serif;
      }
    </style>
  </head>

  <body>
    <h1>Starbuzz Coffee Beverages</h1>

    <h2>House Blend, $1.49</h2>
    <p>A smooth, mild blend of coffees from Mexico, Bolivia and
    Guatemala.</p>

    <h2>Mocha Caffe Latte, $2.35</h2>
    <p>Espresso, steamed milk and chocolate syrup.</p>

    <h2>Cappuccino, $1.89</h2>
    <p>A mixture of espresso, steamed milk and milk foam.</p>

    <h2>Chai Tea, $1.85</h2>
    <p>A spicy drink made with black tea, spices, milk and honey.</p>
  </body>
</html>
```

CSS使用的语法与HTML完全不同。

测试样式……

又该做个测试了，下面重新加载你的“index.html”文件。这一次你会看到Starbuzz页面有了全新的面貌。



Watch it!

如果你用的浏览器是IE，则可能看不到这个边框。

Internet Explorer无法正确地显示页面主体周围的边框。要想看到这个边框，可以试着在Firefox、Chrome或Safari中加载这个页面。

WHO DOES WHAT?

尽管只是简单地看一眼CSS，你应该已经能猜出它能做什么了。把样式定义和相应的功能连连看。

```
background-color: #d2b48c;

margin-left: 20%;
margin-right: 20%;

border: 2px dotted black;

padding: 10px 10px 10px 10px;

font-family: sans-serif;
```

定义文本使用的字体。

定义页面主体周围的边框是虚线，颜色为黑色。

设置左右外边距分别占页面的20%。

设置背景色为土黄色。

在页面主体周围创建一些内边距。

there are no Dumb Questions

问：CSS看上去与HTML完全是两种不同的语言。为什么要用两种语言？这样一来，我就得学更多东西了，是吗？

答：你说的没错，HTML和CSS确实是完全不同的语言，不过这是因为它们的工作截然不同。就像你不能用英语算账，也不能用数学写诗一样，你不会用CSS来创建结构，或者使用HTML创建样式，因为这并不是当初设计它们的初衷。尽管这意味着你得学习两种语言，但你会发现，由于每个语言各有其擅长的方面，与试图使用一种语言兼顾这两方面的工作相比，实际上学习两种语言让它们各司其职反而更为容易。

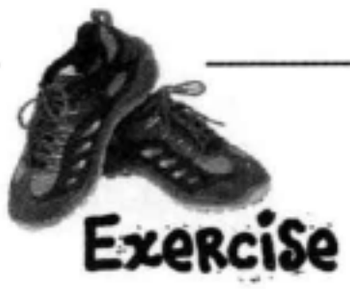
问：#d2b48c看起来可不像是个颜色。#d2b48c怎么会是“土黄色”呢？

答：用CSS指定颜色有很多不同的方式。最常用的一种方式称为“十六进制码”，#d2b48c就是一个十六进制码。这实际上就是土黄色。现在知道就行了，稍后还会告诉你#d2b48c为什么是一种颜色。

问：为什么CSS规则前面有一个“body”？这是什么意思？

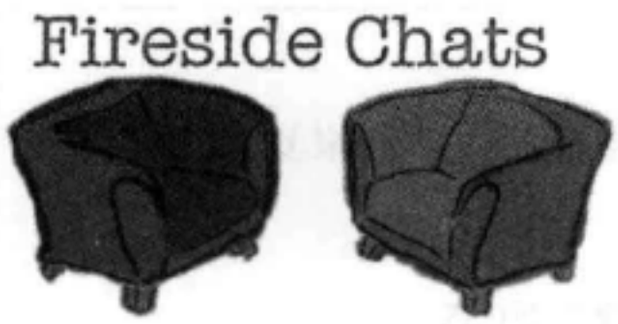
答：CSS中的“body”表示“{”和“}”之间的所有CSS要应用于HTML <body>元素中的内容。所以，将字体设置为sans-serif时，就是说页面主体中的默认字体是sans-serif。

稍后还会更详细地介绍CSS如何工作，请继续读下去。很快你就会发现自己能更熟悉地应用这些规则，这会让你做出漂亮的设计。



既然已经在Starbuzz“index.html”页面中加了一点样式，下面来更新你的“mission.html”页面，让它有同样的样式。

1. 把“mission.html”页面的HTML写在下面，然后增加新CSS。
2. 更新你的“mission.html”文件，包含新增加的CSS。
3. 完成后，在浏览器中重新加载“mission.html”。
4. 确保你的页面看起来与本章最后给出的宗旨页面是一样的。



今晚话题：HTML与CSS谈谈内容和样式

HTML

你好啊，CSS。很高兴你能来，因为我一直想澄清大家对我们的一些误解。

很多人认为我的标记就是告诉浏览器要如何显示内容。根本不是这样的！我说的是结构，不是表现。

嗯，看到人们是怎么以讹传讹的了吧；不管怎么说，即使没有CSS，只用HTML也完全可以得到一个精美的页面。

喂，我也很强大啊。为内容建立结构可比让它好看重要多了。样式太肤浅了，内容的结构才是重中之重。

嗬，真是个自负的家伙！嗯，我想从你这儿不能指望有什么收获了，实在无法交流，你一直都只是想用你鼓吹的样式来表现你的个性。

CSS

有吗？什么误解？

嘿，没错。我可不希望人们把我的功劳算到你头上！

“精美”可能有点言过其实了吧，你不觉得吗？我的意思是说，大多数浏览器显示纯HTML时看起来可都不怎么样。人们需要了解CSS是多么强大，还要知道我能多么轻松地给他们的网页增加漂亮的样式。

现实点吧！如果没有我，则网页会是多么乏味枯燥。不仅如此，没有我就没有办法为页面指定样式，也没有人会把你的页面当回事。一切看上去都那么笨拙呆板、不专业。

HTML

没错。实际上，我们是完全不同的语言，这很好，因为我可不希望你们样式设计人员弄乱我的结构元素。

对，对我来说，很显然每次看到CSS就像是外星人的语言。

你要是这么说，肯定会有上百万的Web开发人员反驳你。我的语法很简洁，与内容配合得很好。

嘿，听说过结束标记吗？

刚注意到，不管你到哪里，我都会用<style>标记把你包围起来。祝你逃跑愉快！

CSS

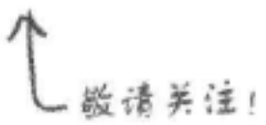
个性？好的设计和布局对页面的可读性和可用性会有非常大的影响。另外，你应该高兴，正是有了我灵活的样式规则，才允许设计人员对你的元素做各种各样有趣的处理，而不必把你的结构搅得一塌糊涂。

别担心，我们在两个完全不相关的世界里。

是啊，那么HTML也能算是一种语言吗？谁见过这么拙劣的东西，还有所有那些标记？

来看看CSS吧。多么高雅，多么简单，不会<在所有东西><周围>都加上那些笨拙的尖括号。<看><我><也><能><像><HTML先生><那样><说话><,><快><看><我><!>

哈！看着吧……因为，猜不到吧？我真的能逃掉……



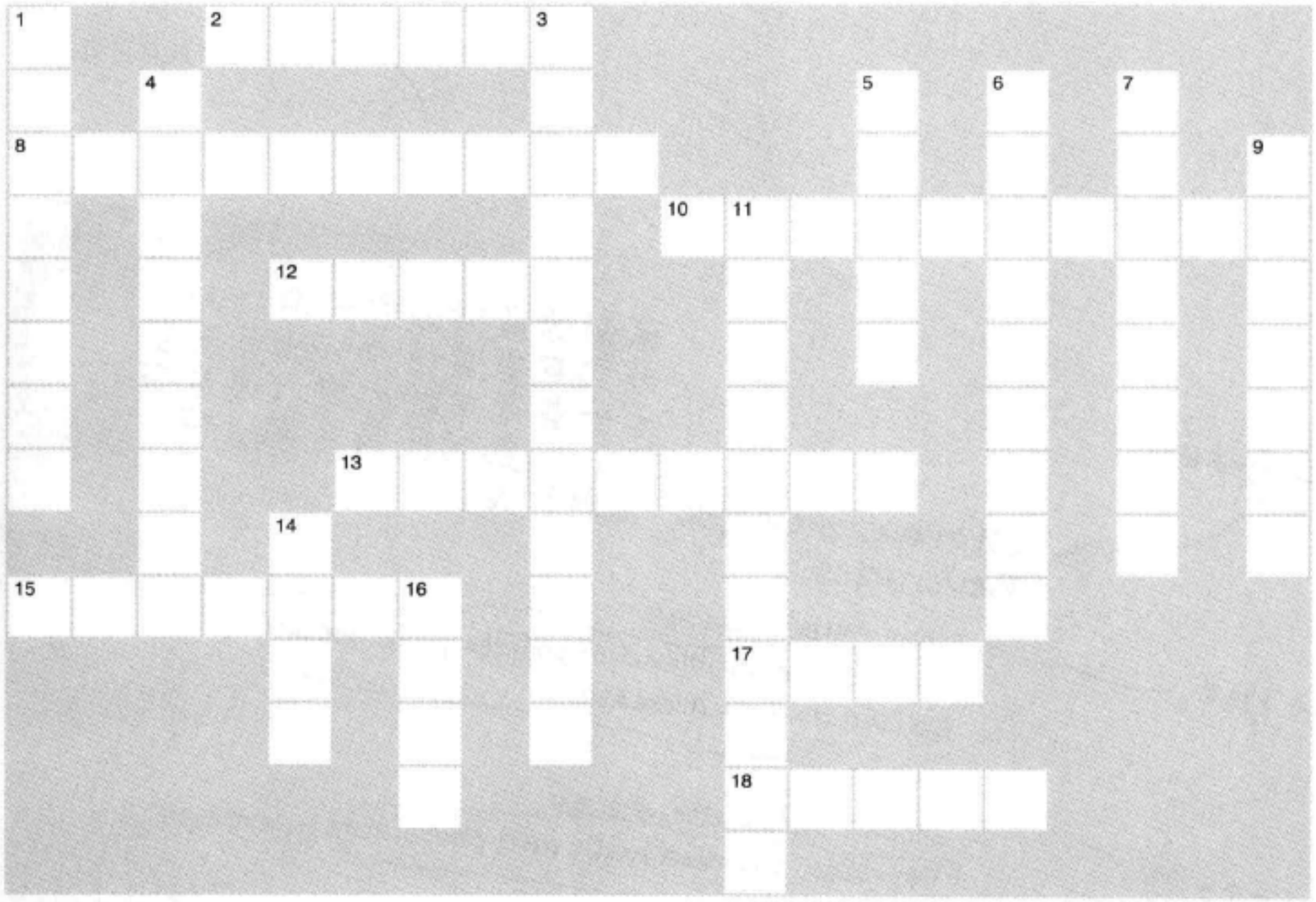
现在我们不仅有香浓的House Blend咖啡，还有一个网页向所有顾客介绍我们的咖啡。做得好。关于将来我还有一些更好的想法。另外，你能想一想怎么把这些页面发布到互联网上，让其他人都看到，能做到吗？

BULLET POINTS

- HTML和CSS是我们用来创建网页的语言。
- Web服务器存储并提供由HTML和CSS创建的网页。浏览器获取页面，并根据HTML和CSS显示网页的内容。
- HTML是超文本标记语言（HyperText Markup Language）的缩写，用来建立网页的结构。
- CSS是层叠样式表（Cascading Style Sheets）的缩写，用来控制HTML的表现。
- 通过HTML，我们利用标记来标示内容提供结构。我们把匹配标记以及它们包围的内容称为元素。
- 元素由3部分组成：一个开始标记、内容和一个结束标记。不过有些元素（比如<img>）有所例外。
- 开始标记可以有属性。我们已经见过一个属性：type。
- 结束标记在左尖括号后面、标记名前面有一个“/”，以明确这是结束标记。
- 所有页面都要有一个<html>元素，其中要有一个<head>元素和一个<body>元素。
- 网页的信息放在<head>元素里。
- <body>元素里的内容就是你将在浏览器里看到的东西。
- 大多数空白符（制表符、回车、空格）都会被浏览器忽略，不过可以利用空白符让你的HTML（对你）更有可读性。
- 可以在<style>元素中写CSS规则，为HTML网页增加CSS。<style>元素总要放在<head>元素里。
- 可以使用CSS在HTML中指定元素的特性。

HTML填字游戏

该坐下来让你的左脑活动活动了。这是一个标准的填字游戏，所有答案都在这一章里出现过。



横向

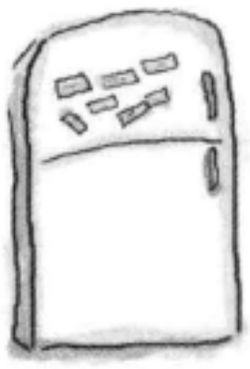
2. HTML中的"M"。
8. 标记可以有这些来提供附加信息。
10. 浏览器会忽略它们。
12. 通过这个元素来定义表现。
13. <p>元素的作用。
15. 两个标记及内容。
17. 页面上所看到的。
18. 我们强调了Dance _____ Revolution。

纵向

1. 它们共有6个。
3. 要控制_____时会使用CSS。
4. 标记内容来提供_____。
5. 对于每个页面总是出现在浏览器顶部。
6. 我们希望有的一种样式类型。
7. 开启你的Web职业生涯的公司。
9. 唯一可用的样式类型。
11. HTML中总是将它们分离。
14. 关于网页的信息。
16. 开始和结束。

Sharpen your pencil Solution

在餐巾纸上加些标记（用铅笔），标出你看到的结构，再增加你认为缺少的内容。



标记磁贴答案

你的任务是为Starbuzz餐巾纸上的文本增加结构。使用这一页最下面的冰箱磁贴标记文本，指示哪些部分是标题、子标题和段落文本。下面是我们的答案。

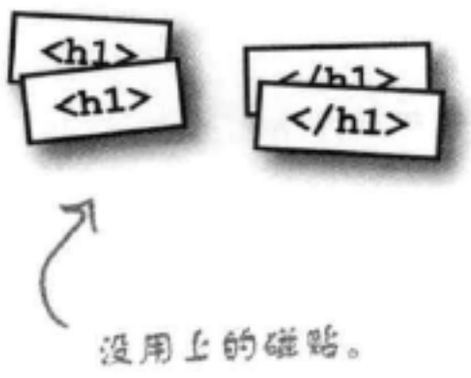
<h1> Starbuzz Coffee Beverages </h1>

<h2> House Blend, $1.49 </h2>
 A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala.

<h2> Mocha Cafe Latte, $2.35 </h2>
 Espresso, steamed milk and chocolate syrup.

<h2> Cappuccino, $1.89 </h2>
 A mixture of espresso, steamed milk and foam.

<h2> Chai Tea, $1.85 </h2>
 A spicy drink made with black tea, spices, milk and honey.



Exercise Solution

mission.html

```
<html>
  <head>
    <title>Starbuzz Coffee's Mission</title>
  </head>
  <body>
    <h1>Starbuzz Coffee's Mission</h1>
    <p>To provide all the caffeine that you need to
power your life.</p>
    <p>Just drink it.</p>
  </body>
</html>
```

这是完整页面的HTML源代码。

Starbuzz Coffee's Mission

file:///chapter1/starbuzz/mission.html

Starbuzz Coffee's Mission

To provide all the caffeine that you need to power your life.

Just drink it.

这是浏览器中显示的HTML。

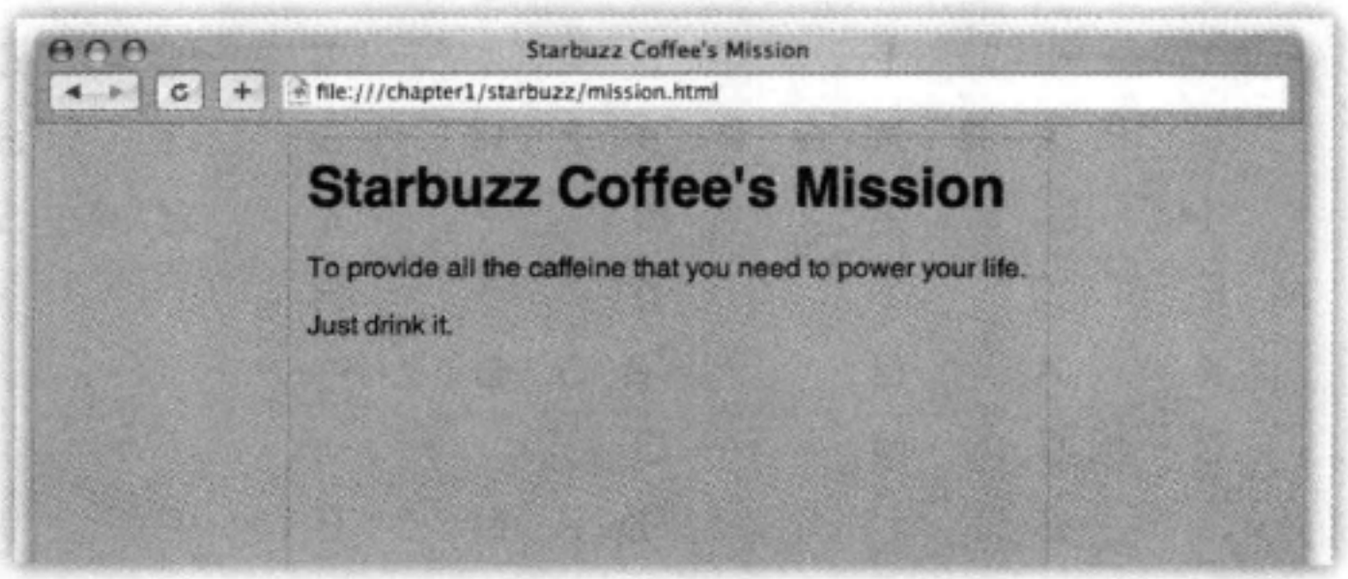
Exercise Solution

这里是完善后页面中的CSS。

mission.html

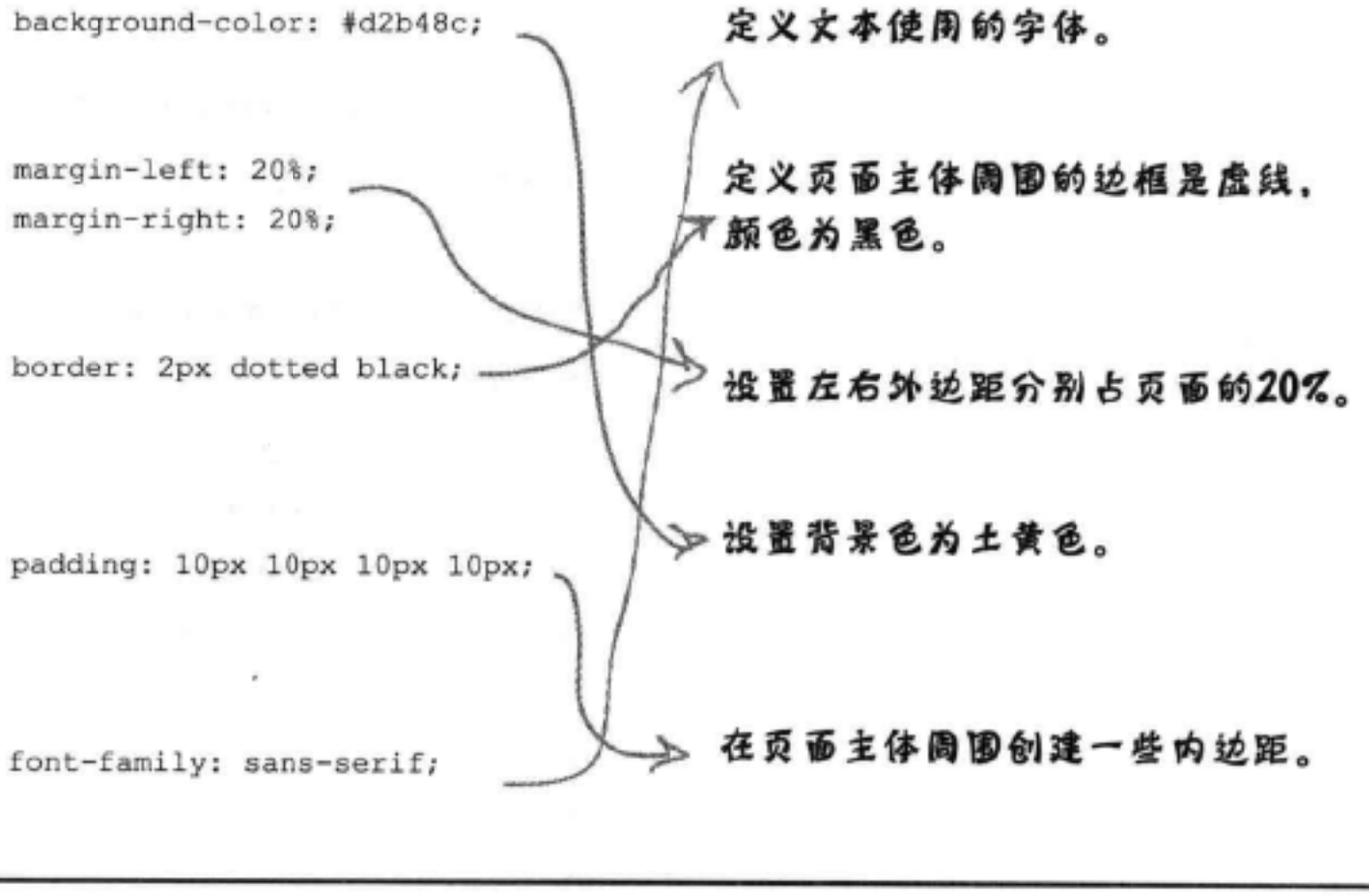
```
<html>
  <head>
    <title>Starbuzz Coffee's Mission</title>
    <style type= "text/css" >
     body {
       background-color: #d2b48c;
       margin-left: 20%;
       margin-right: 20%;
       border: 2px dotted black;
       padding: 10px 10px 10px 10px;
       font-family: sans-serif;
     }
   </style>
  </head>
  <body>
    <h1>Starbuzz Coffee's Mission</h1>
    <p>To provide all the caffeine that you need to power
your life.</p>
    <p>Just drink it.</p>
  </body>
```

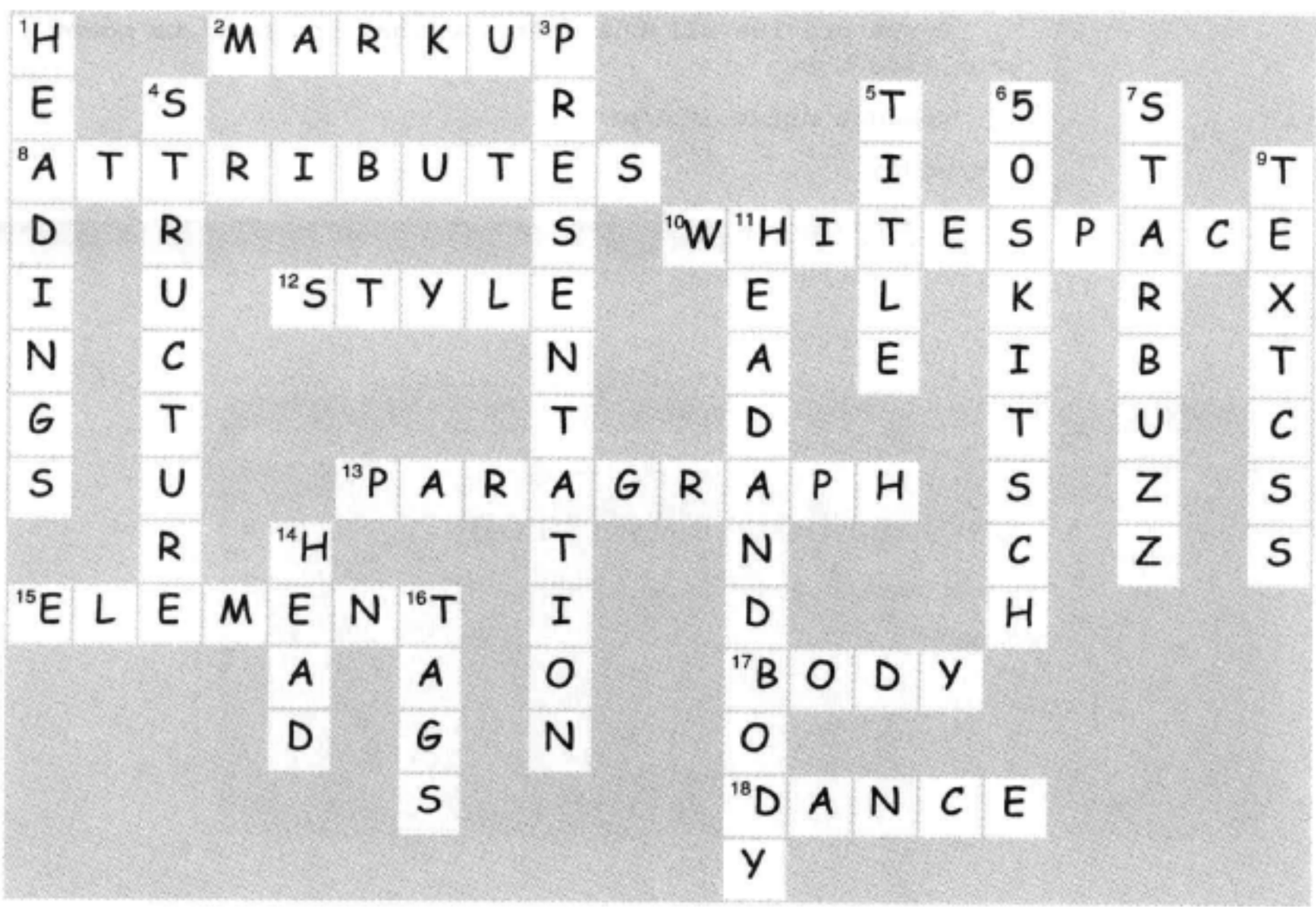
现在样式与Starbuzz主页面一致了。



WHO DOES WHAT?

尽管只是简单地看一眼CSS，你应该已经能猜出它能做什么了。把样式定义和相应的功能连连看。





认识HTML中的“HT”

有人在说“超文本”？那是什么？哦，这就是Web的基础。第1章中我们简单认识了HTML，知道它是一种不错的标记语言（HTML中的“ML”），用来描述网页的结构。现在我们再来了解HTML中的“HT”，也就是超文本（hypertext），有了它，我们就能摆脱单个页面的束缚，链接到其他页面。顺便我们还要认识一个强大的新元素——<a>元素，了解“相对”是多么美妙的东西。好了，系好你的安全带，准备学习一些超文本吧。

Head First休闲室，全新改良

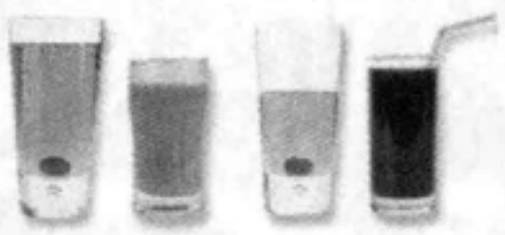
还记得Head First休闲室吗？那是一个不错的网站，不过，如果顾客能看到一个清凉饮料列表就更好了，不是吗？甚至还可以更棒，我们应该为顾客提供一些具体的路线说明，让他们能找到我们的休闲室。

这是改进后的新页面。

Head First Lounge

file:///chapter2/completelounge/lounge.html

Welcome to the New and Improved Head First Lounge



"elixirs" 链接指向另一个页面，其中包含一个完整的清凉饮料列表。

我们增加了两个新页面的链接，一个指向清凉饮料页面，另一个指向路线说明。

Join us any evening for refreshing elixirs, conversation and maybe a game or two of *Dance Dance Revolution*. Wireless access is always provided; BYOWS (Bring Your Own Web Server).

Directions

You'll find us right in the center of downtown Webville. If you need help finding us, check out our detailed directions. Come join us!

"detailed directions" 链接指向一个提供路线说明的HTML页面。

Head First Lounge Directions

file:///chapter2/completelounge/about/directions.html

Directions

Take the 305 S exit to Webville - go 0.4 mi

Continue on 305 - go 12 mi

Turn right at Structure Ave N - go 0.6 mi

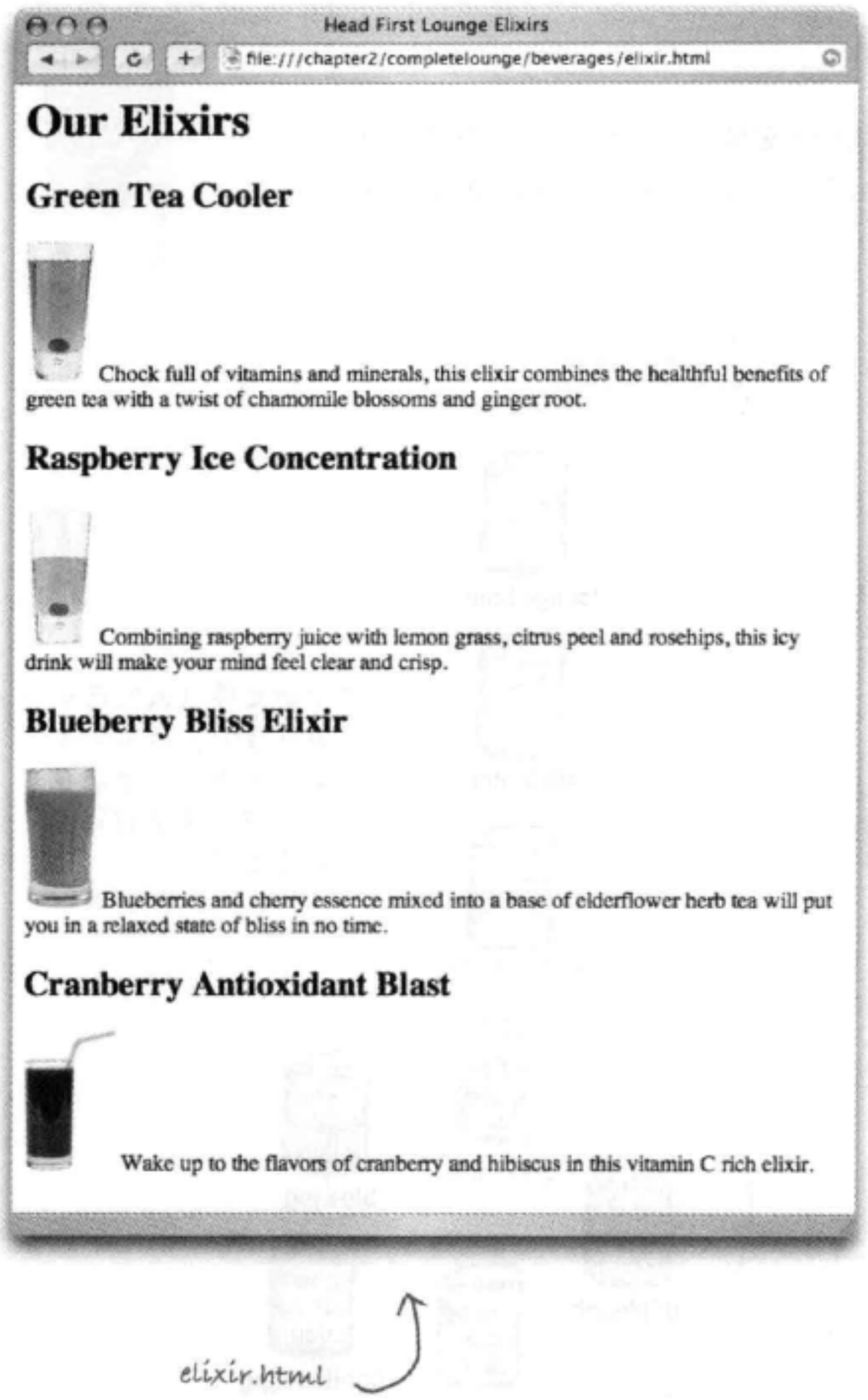
Turn right and head toward Structure Ave N - go 0.0 mi

Turn right at Structure Ave N - go 0.7 mi

Continue on Structure Ave S - go 0.2 mi

Turn right at SW Presentation Way - go 0.0 mi

directions.html



这个页面列出一些清凉健康饮料。可以先来一杯。

创建这个全新改良版的休闲室需要3步……

下面来改造原先的Head First休闲室页面，让它链接到这两个新页面。

1. 第1步很容易，因为我们已经为你创建了“directions.html”和“elixir.html”文件。可以在本书的源代码文件（http://wickedlysmart.com/hfhtmlcss）中找到这两个文件。

2. 接下来，需要编辑“lounge.html”文件，加入链接到“directions.html”和“elixir.html”所需的HTML。

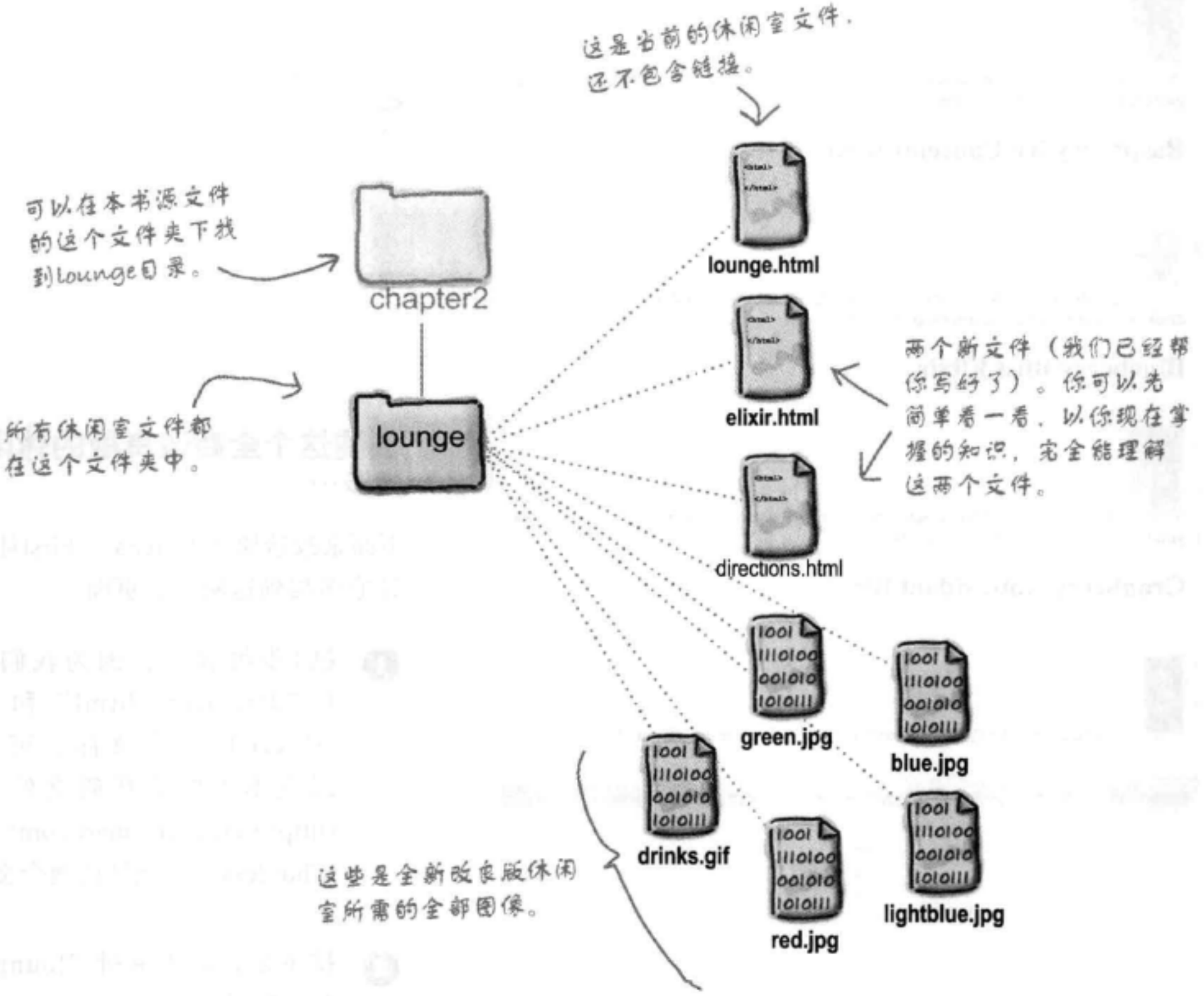
3. 最后，对这些页面做个测试，试试看新链接是否正常。完成后，我们再坐下来好好研究它是如何工作的。

翻开下一页， 我们开始吧……

创建新的休闲室

❶ 获取源文件。

从http://wickedlysmart.com/hfhtmlcss获取源文件。下载之后，查看文件夹“chapter2/lounge”，你会发现“lounge.html”、“elixir.html”和“directions.html”文件（还有一大堆图像文件）。



Head First休闲室还在发展壮大，你觉得把网站的所有文件都放在一个目录下能很好地组织这个网站吗？有没有别的办法？

❷ 编辑lounge.html。

在编辑器中打开“lounge.html”。增加下面突出显示的新文本和HTML。输入这些代码，下一页我们再来分析这是如何工作的。

```
<html>
  <head>
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
    <img src="drinks.gif">
    <p>
      Join us any evening for
      refreshing <a href="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 server).
    </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="directions.html">detailed directions</a>.
      Come join us!
    </p>
  </body>
</html>
```

在标题中增加文本“New and Improved”。

在这里增加清凉饮料链接的HTML。

要创建链接，需要使用<a>元素，稍后我们会讨论这个元素是如何工作的……

这里需要增加一些文本，指引顾客访问新增的路线说明。

在这里增加指向这些说明的链接，同样的，还是要使用一个<a>元素。

❸ 保存lounge.html，再做个测试。

完成上述修改后，保存文件“lounge.html”，再在浏览器中打开。可以做下面的尝试:

1. 单击清凉饮料链接，会显示新的清凉饮料页面。
2. 单击浏览器的“后退”按钮，会再次显示“lounge.html”。
3. 单击路线说明链接，会显示新的说明页面。

Ok，我已经加载了新的休闲室页面，单击了链接，一切都工作正常。不过我还想搞清楚这些HTML是如何工作的。

在幕后

我们做了什么？

① 下面一步一步来创建HTML链接。首先，需要把链接文本放在<a>元素中，就像下面这样：

```
<a>elixirs</a>
```

<a>元素用于创建指向另一个页面的链接。

```
<a>driving directions</a>
```

<a>元素的内容就是链接文本。在浏览器中，链接文本会显示有下划线，指示这是可单击的。

② 既然有了链接文本，现在需要增加一些HTML告诉浏览器这个链接指向哪里：

对于这个链接，浏览器会显示文本“elixirs”，单击这个文本时，用户将被带往“elixir.html”页面。

```
<a href="elixir.html">elixirs</a>
```

要通过href属性来指定链接的目标文件。

对于这个链接，浏览器会显示“driving directions”链接，单击时，用户会进入“directions.html”页面。

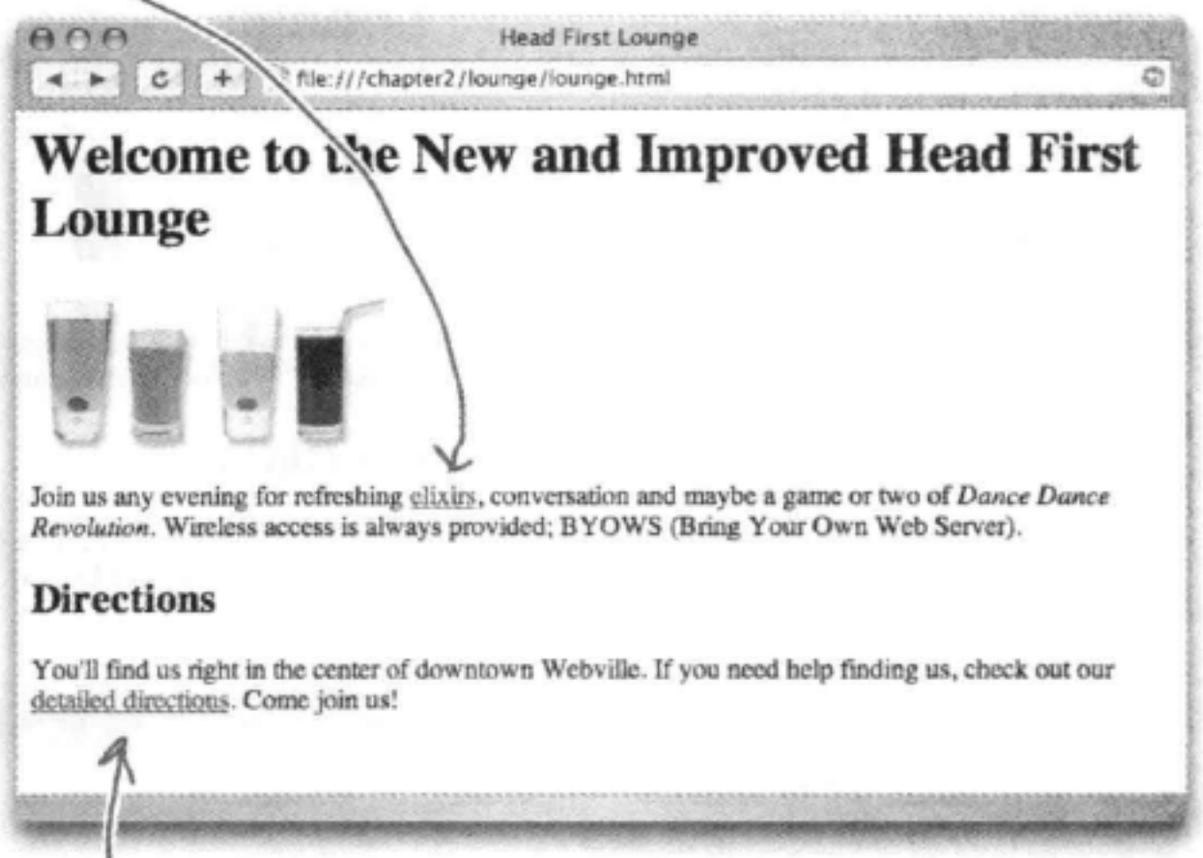
```
<a href="directions.html">driving directions</a>
```

浏览器做了什么？

在幕后

① 首先，浏览器会显示页面，如果遇到一个<a>元素，则会取这个元素的内容，把它显示为一个可单击的链接。

```
<a href="elixir.html">elixirs</a>
```



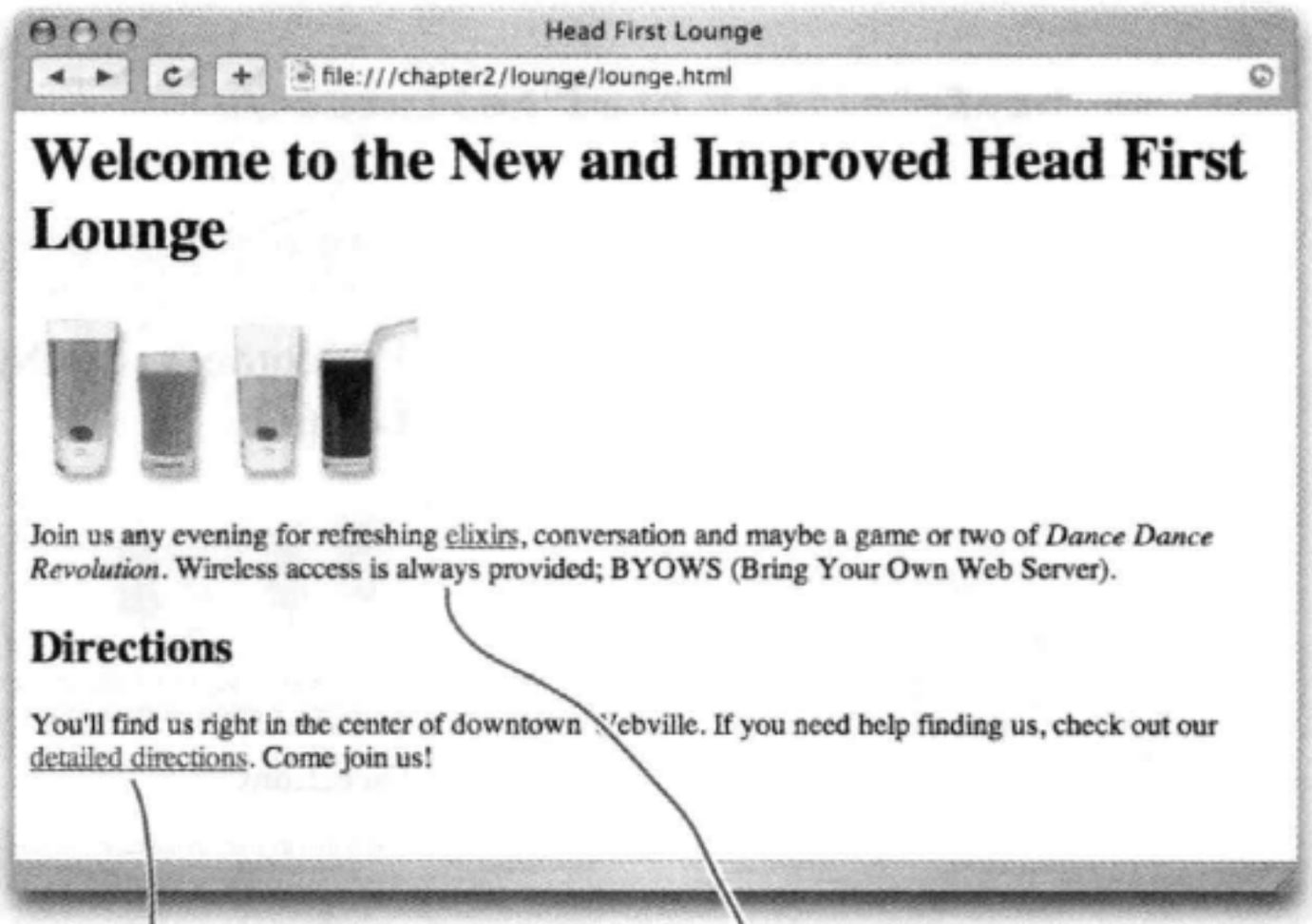
"elixirs" 和 "detailed directions" 都放在开始和结束<a>标记之间，所以在网页上它们会显示为可单击的文本。

```
<a href="directions.html">detailed directions</a>
```

使用<a>元素创建一个超文本链接，链接到另一个Web页面。
<a>元素的内容会成为Web页面中可单击的文本。
href属性告诉浏览器链接的目标文件。

在幕后

② 接下来，用户单击一个链接时，浏览器使用“href”属性来确定这个链接指向哪个页面。



用户单击“elixirs”链接或……

……单击“detailed directions”。

单击“detailed directions”时，浏览器获取href属性的值，在这里就是“directions.html”……

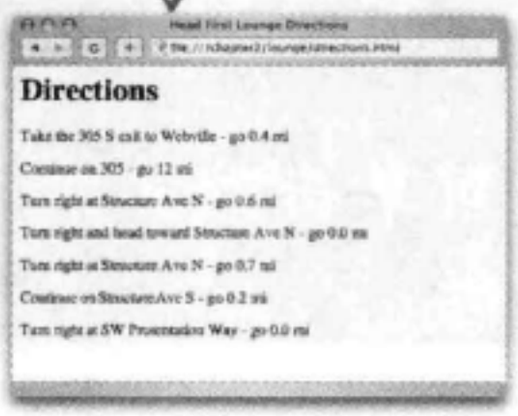
如果单击“elixirs”，则浏览器会得到href值“elixir.html”……

```
<a href="elixir.html">elixirs</a>
```

……并显示“elixir.html”页面。


```
<a href="directions.html">detailed directions</a>
```

……并加载“directions.html”。



了解属性

利用属性（Attributes），可以指定一个元素的附加信息。尽管我们还没有详细分析属性，不过之前已经见过属性的几个例子：

```
<style type="text/css">
```

type属性指定我们使用哪一种样式语言，这里就是CSS。

```
<a href="irule.html">
```

href属性告诉我们一个超链接的目标文件。

```
<img src="sweetphoto.gif">
```

src属性指定一个img标记显示的图像的文件名。

下面给出一个例子，以便你更好地了解属性如何工作：

如果<car>是元素呢？

如果<car>是一个元素，很自然地，你可能想写类似这样的标记：

```
<car>My Red Mini</car>
```

没有任何属性，我们能提供的仅仅是一个描述性的车名。

不过这个<car>元素只能提供一个描述性的车名，它没有告诉我们汽车的品牌、型号、是不是敞篷车，以及我们可能想知道的很多其他细节。所以，如果<car>确实是一个元素，则我们可能会像下面这样使用属性：

不过有了属性，就可以用各种信息定制这个元素。

```
<car make="Mini" model="Cooper" convertible="no">My Red Mini</car>
```

是不是好一些了？现在这个标记采用一种方便、易写的形式提供了更多信息。

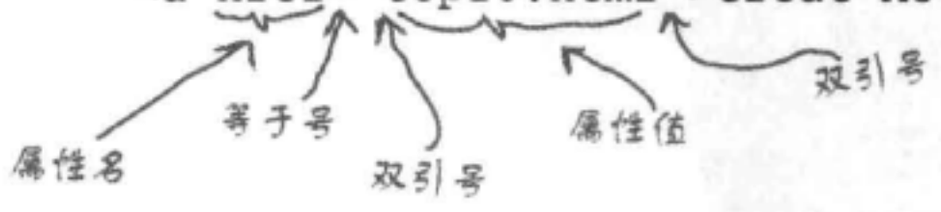
安全第一

属性的写法都是一样的：首先是属性名，后面是一个等于号，然后是用双引号括起来的属性值。

你可能看到Web上有一些不严谨的HTML没有加双引号，不过你可不能偷懒。如果不严谨，可能导致你以后会遇到很多问题（本书后面就会看到）。

这样做（最佳实践）

```
<a href="top10.html">Great Movies</a>
```



不要这样做

```
<a href=top10.html>Great Movies</a>
```

属性值周围没有双引号

there are no Dumb Questions

问：我能为一个HTML元素造新属性吗？

答：Web浏览器只认识每个元素的一组预定义的属性。如果你造了新属性，则浏览器不知道该怎么处理，另外在本书后面你会看到，这样做很可能给你带来麻烦。浏览器能识别一个元素或属性时，我们会说浏览器“支持“这个元素或属性。应该只使用浏览器支持的那些属性。

尽管如此，关于编写Web应用（这是《Head First HTML5 Programming》讨论的主题），现在HTML5已经支持定制数据属性，允许你为新属性构造定制的属性名。

问：谁来决定“支持”哪些属性？

答：有一些标准委员会专门考虑HTML的元素和属性。这些委员会由没其他事可干的这样一群人组成，他们不计得失、不计报酬，无私地投入时间和精力来确保建立一个通用的HTML路线图，使得所有组织都能使用这样一个标准实现他们的浏览器。

问：我怎么知道哪些属性和元素得到支持呢？另外，所有属性都能应用于任何元素吗？

答：一个给定元素只能使用某些特定的属性。可以这样来考虑：你不会对元素<toaster>使用属性“convertible”吧，是不是？也就是说，你只希望使用元素支持的那些有意义的属性。

通过这本书的介绍，你会逐步了解哪些元素支持哪些属性。学完这本书之后，还有很多不错的参考书可以帮你复习，如《HTML & XHTML: The Definitive Guide》(O’Reilly)。

属性闪亮登场

本周访谈:
href属性的自白

Head First：欢迎你，href。很荣幸能采访像你这么有名气的属性。

href：谢谢。我也很高兴来这里，可以摆脱所有链接。对一个属性来说，这实在是让我精疲力尽了。每次有人单击一个链接时，猜猜看是谁告诉浏览器下一步该去哪里？那就是我啊。

Head First：真是感谢你能在百忙中抽出时间。能不能从头说说……属性到底能做什么？

href：当然可以。嗯，属性可以用来定制一个元素。用<a>标记包围一些内容很容易，比如“Sign up now!”。我们会这么做：<a>Sign up now!。不过，如果没有我，就没有href属性，你就没办法告诉<a>元素链接的目标文件是什么。

Head First：不太明白……

href：……不过，有了属性，你就能提供元素的附加信息。就我而言，就是链接要指向哪里：Sign up now!。这说明，这个标签为“Sign up now!”的<a>元素要链接到“signup.html”页面。当今世界还有许许多多其他属性，不过在<a>元素中只能用我来告诉它要指向哪里。

Head First：噢。那么我还得问一句，希望你别介意，这个名字是怎么来的？href？它有什么特别的意思吗？

href：这是互联网家族的一个古老名字。它的含义是“超文本引用”（hypertextreference），不过我的朋友都叫我小名“href”。

Head First：那又是什么？

href：超文本引用就是互联网或你的计算机上的一个资源的别称。通常这个资源就是一个Web页面，不过我也可以指向PDF文档……实际上我可以指向各种各样的资源。

Head First：有意思。我们的读者目前为止看到的链接都只是指向他们自己的页面,怎么链接到Web上的其他页面和资源呢？

href：嘿，我得回去工作了，没有我，整个Web该成一团乱麻了。再说了，教他们如何链接不该是你的工作吗？

Head First：好吧，好吧，没错，我们马上去做……非常感谢你接受采访， href。

你已经创建了从“lounge.html”到“elixir.html”和“directions.html”的链接,现在再反过来。下面是“elixir.html”的HTML。请在清凉饮料页面的最下面增加一个标为“Back to the Lounge”的链接，指回到“lounge.html”。

```
<html>
  <head>
    <title>Head First Lounge Elixirs</title>
  </head>
  <body>
    <h1>Our Elixirs</h1>

    <h2>Green Tea Cooler</h2>
    <p>
      <img src="green.jpg">
      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="lightblue.jpg">
      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="blue.jpg">
      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="red.jpg">
      Wake up to the flavors of cranberry and hibiscus
      in this vitamin C rich elixir.
    </p>

    你的新HTML放在这里。

  </body>
</html>
```

完成后，再在“directions.html”中做同样的处理。

我们需要你的帮助来构造和解构<a>元素。希望你新学的<a>元素知识能有帮助。下面每一行分别给出了标签、目标文件和完整的<a>元素。请填入缺少的信息。我们已经帮你完成了第一行。

标签	目标文件	HTML中要写的代码
Hot or Not?	hot.html	`<a href="hot.html">Hot or Not?</a>`
Resume	cv.html	
	candy.html	`<a href=".....">Eye Candy</a>`
See my mini	mini-cooper.html	
let's play		`<a href="millionaire.html">.....</a>`

there are no Dumb Questions

问：我见过很多页面上还可以单击图像而不只是文本。能用<a>元素来实现吗？

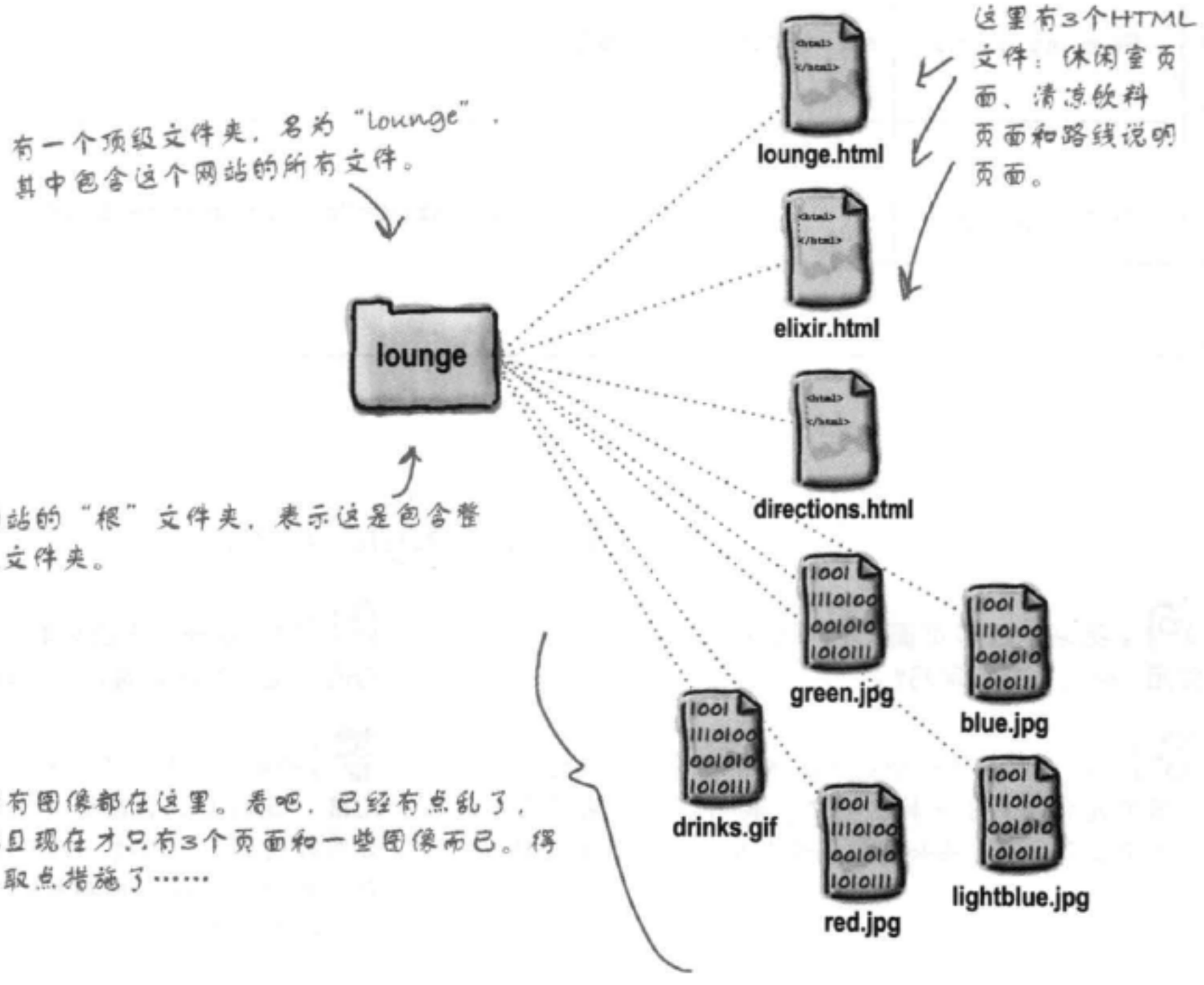
答：可以，如果把一个<img>元素放在<a>标记之间，这个图像就会像文本一样可单击。我们不打算花好几章来深入讨论图像，不过要知道，图像确实也可以作为链接。

问：这么说任何东西只要放在<a>标记之间就会成为可单击的，是吗？比如说，一个段落？

答：确实可以把一个<p>元素放在<a>元素中来链接整个段落。我们大多会在<a>元素中使用文本和图像（或者二者同时使用），不过，如果你需要链接一个<p>或<h1>元素，则完全可以。哪些标记可以嵌在其他标记里，这完全是另一个话题了，不过不用担心，很快就会谈到这个内容。

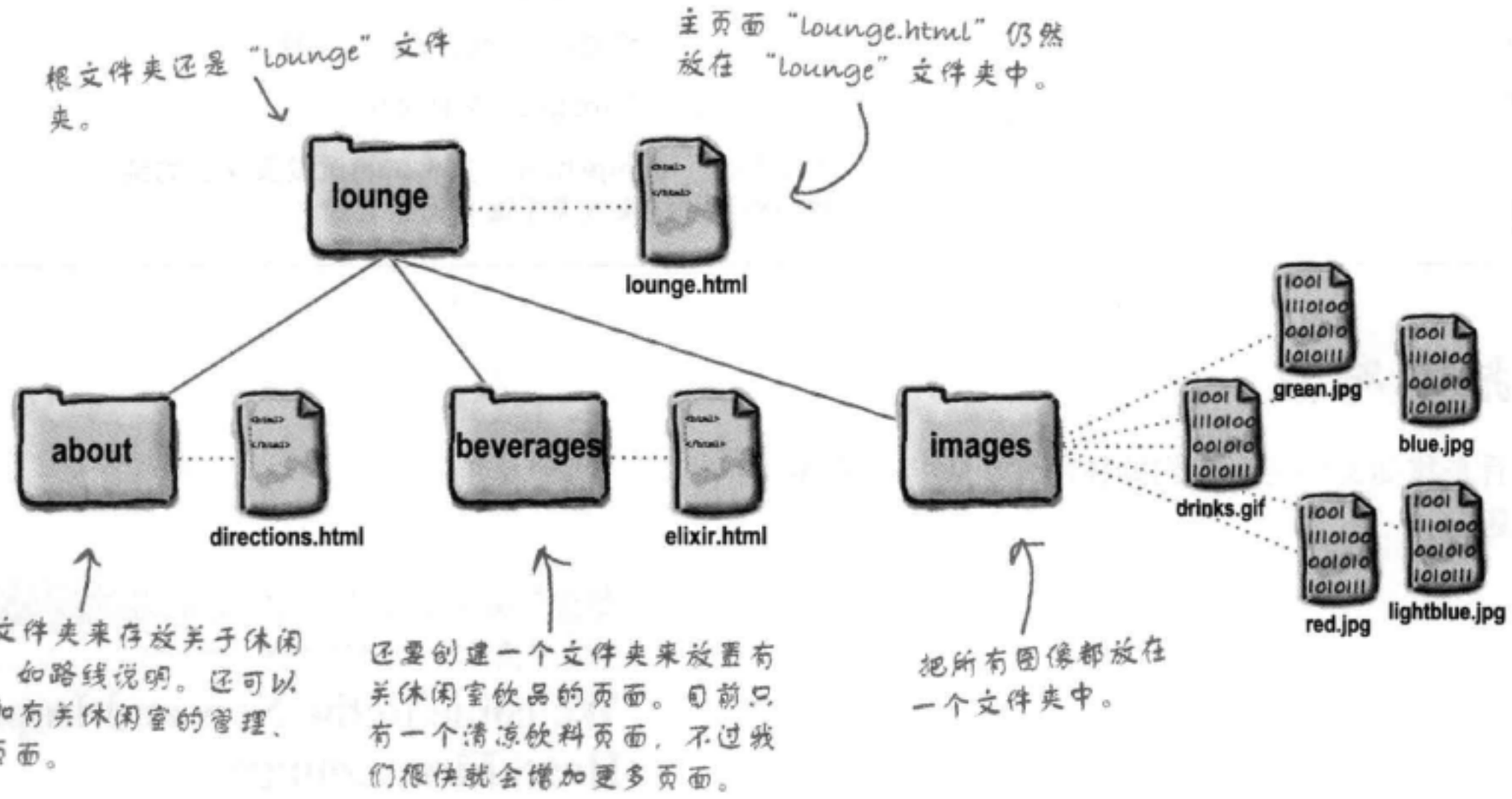
组织

开始创建更多HTML页面之前，先要组织好所有资源。到目前为止，我们一直都是将所有文件和图像放在一个文件夹中。你会发现，即使是规模很小的网站，如果把网页、图片和其他资源分别放在不同文件夹中维护，网站也会更容易管理。我们的现状是这样的：



组织休闲室……

下面让这个休闲室网站更有组织、更有条理。要记住组织网站的办法有很多，我们先从简单的做起，为页面创建一些文件夹，还要把所有图像归入一个位置。



there are no Dumb Questions

问：既然有一个文件夹来存放图像，为什么不创建另外一个“html”文件夹，把所有HTML文件都放在那个文件夹里呢？

答：也可以的。组织文件并没有“绝对正确”的方法。实际上，你会希望采用一种最适合你和用户的方式来组织网站。与大多数设计决策一样，你要选择足够灵活可以扩展的组织方案，同时还要力求简单。

问：那么，为什么不在各个其他文件夹中创建一个图像文件夹呢，比如在“about”和“beverages”下分别创建一个图像文件夹？

答：同样地，当然也可以这么做。我们希望某些图像能在多个页面上重用，所以把图像都放在根（顶级）目录下的一个文件夹里集中维护。如果你的网站需要在不同部分使用大量图像，则可以让每个分支都有自己的图像文件夹。

问：“每个分支”？

答：可以把这样来理解描述文件夹的方式看成是倒垂的树，它最上面是根，每个向下到一个文件或文件夹的路径就是一个分支。

现在你需要创建前面所示的文件和文件夹结构。要完成下面的工作：

1. 找到你的“lounge”文件夹，创建3个新的子文件夹，分别名为“about”、“beverages”和“images”。
2. 把文件“directions.html”移动到“about”文件夹中。
3. 把文件“elixir.html”移动到“beverages”文件夹中。
4. 将所有图像都移动到“images”文件夹中。
5. 最后，加载你的“lounge.html”文件，再试试页面上的链接。与我们得到的结果（见下面）做个比较。

技术难点

看来移动文件之后我们的休闲室页面出了很多问题。

图像没有正常显示。通常我们把这叫做“损坏的图像”。

另外，单击“elixirs”（或“detailed directions”）时情况更糟糕。我们会得到一个错误，指出无法找到这个页面。

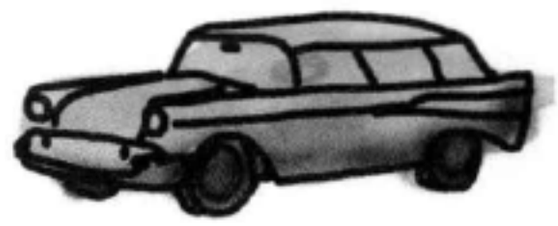
有些浏览器会把这个错误显示为一个web页面而不是一个对话框。

没错，我们要告诉浏览器这些页面的新位置。

到目前为止，我们使用的href值都指向相同文件夹中的页面。不过，网站通常会更为复杂，你还需要能够指向其他文件夹中的页面。

为此，需要追踪从页面到目标文件的路径。这可能意味着要下行一、两个文件夹，或者上行一、两个文件夹，不过不管怎样最后总能得到一个可以放在href中的相对路径。

规划你的路径……



如果你计划开着房车去度假，你会怎么做？你会找一个地图，从当前位置开始，画出一条到达目的地的路径。具体方向要由你的当前位置来定。如果你身处另一个城市，方向当然会不同，对不对？

OK，你当然可以访问Google Maps，不过在这里还是照我们说的做吧！

要确定链接的相对路径，也是同样的道理：从包含这个链接的页面开始，然后沿着一条路径，经过一些文件夹直到找到你需要指向的那个文件。

还有其他类型的路径。后面几章会介绍。

下面来追踪几个相对路径（顺便修复休闲室的问题）。

向下链接到一个子文件夹

① **从“lounge.html”链接到“elixir.html”。**

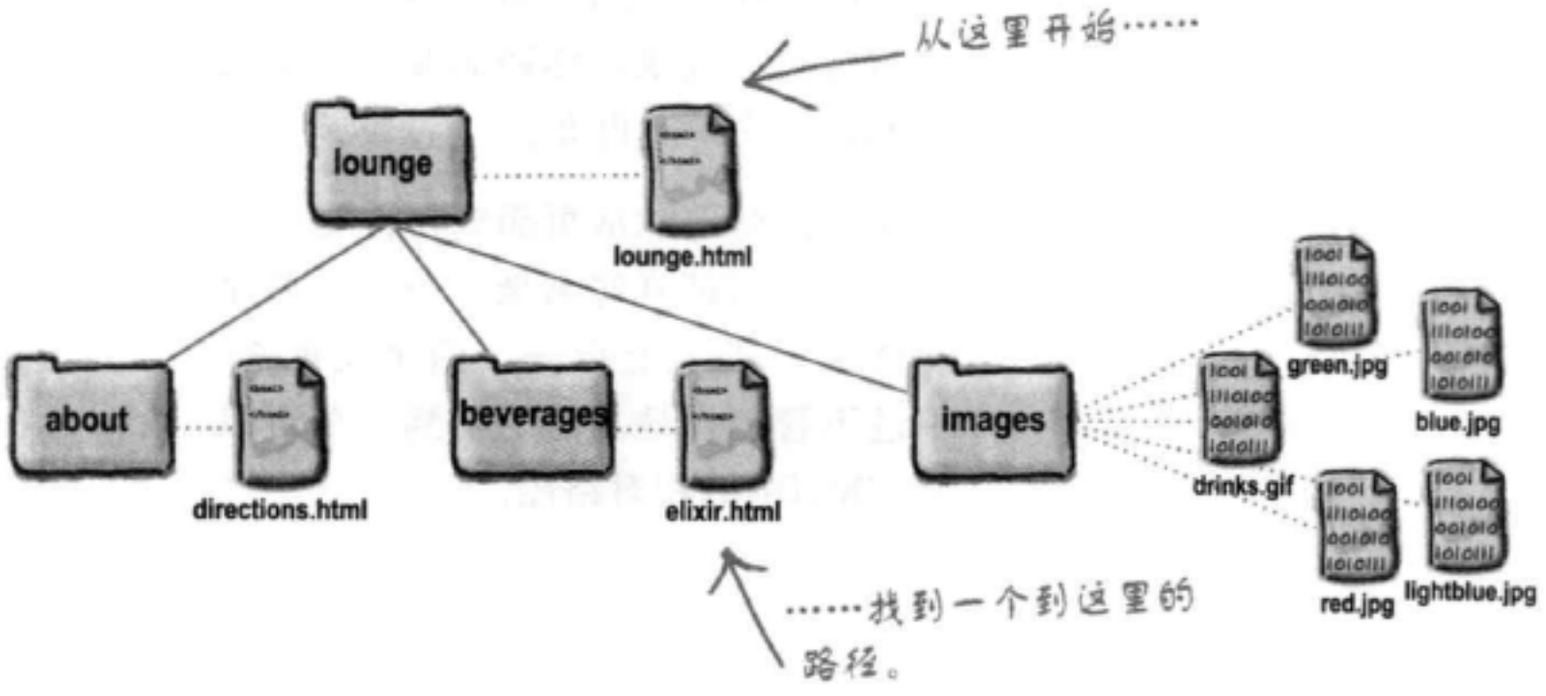
我们需要修复“lounge.html”页面中的 “elixirs”链接。目前这个 <a> 元素是这样的：

```
<a href="elixir.html">elixirs</a>
```

现在我们只使用了文件名“elixir.html”，这就告诉浏览器要在“lounge.html”所在的相同文件夹里查找文件。

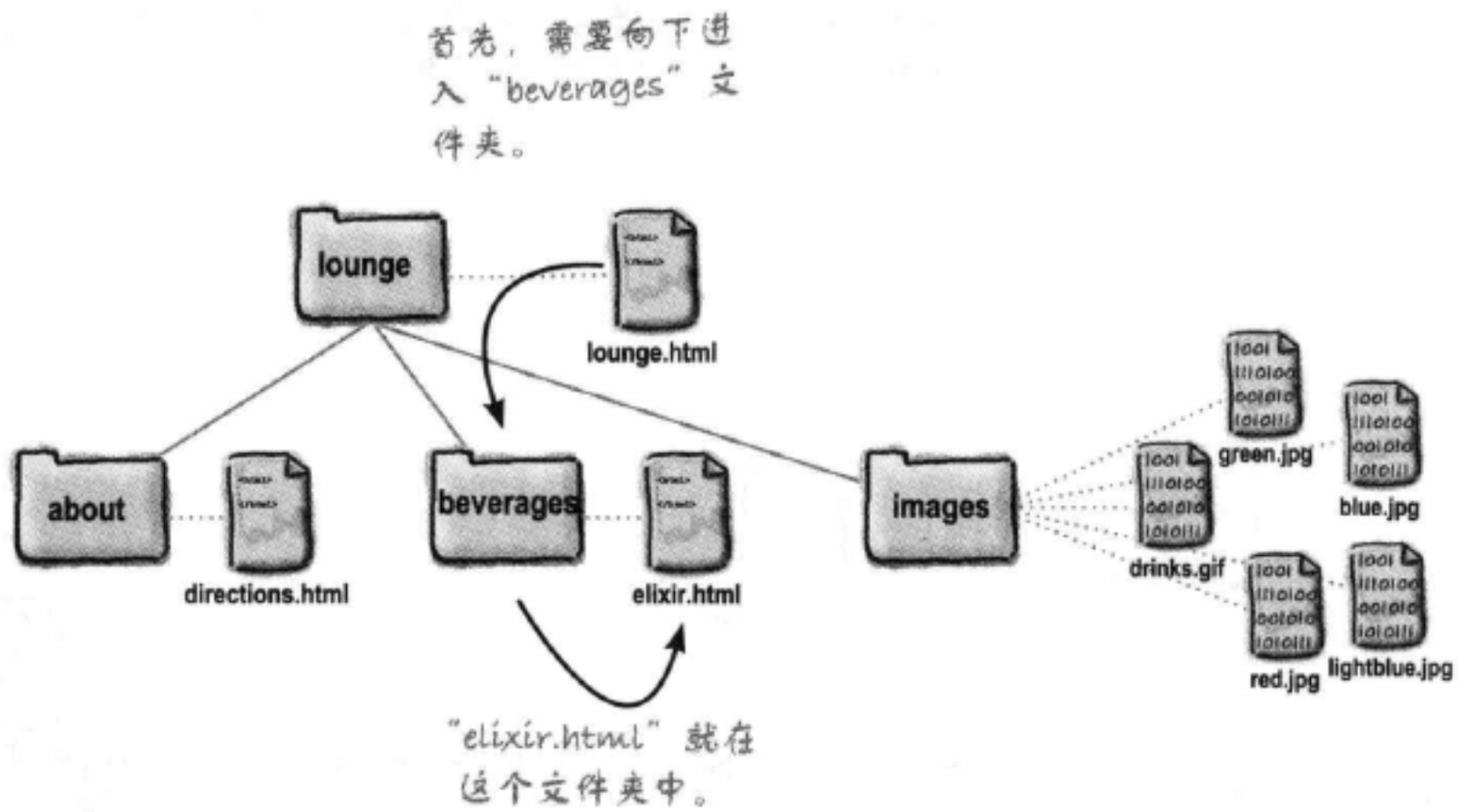
② **明确源文件和目标文件。**

重新组织休闲室时，我们把“lounge.html”仍然留在“lounge”文件夹中，但把“elixir.html”移动到“beverages”文件夹中，这是“lounge”的一个子文件夹。



③ 追踪从源文件到目标文件的一个路径。

下面追踪这个路径。要从“lounge.html”文件到达“elixir.html”，需要先进入“beverages”文件夹，然后在这个文件夹中可以找到“elixir.html”。

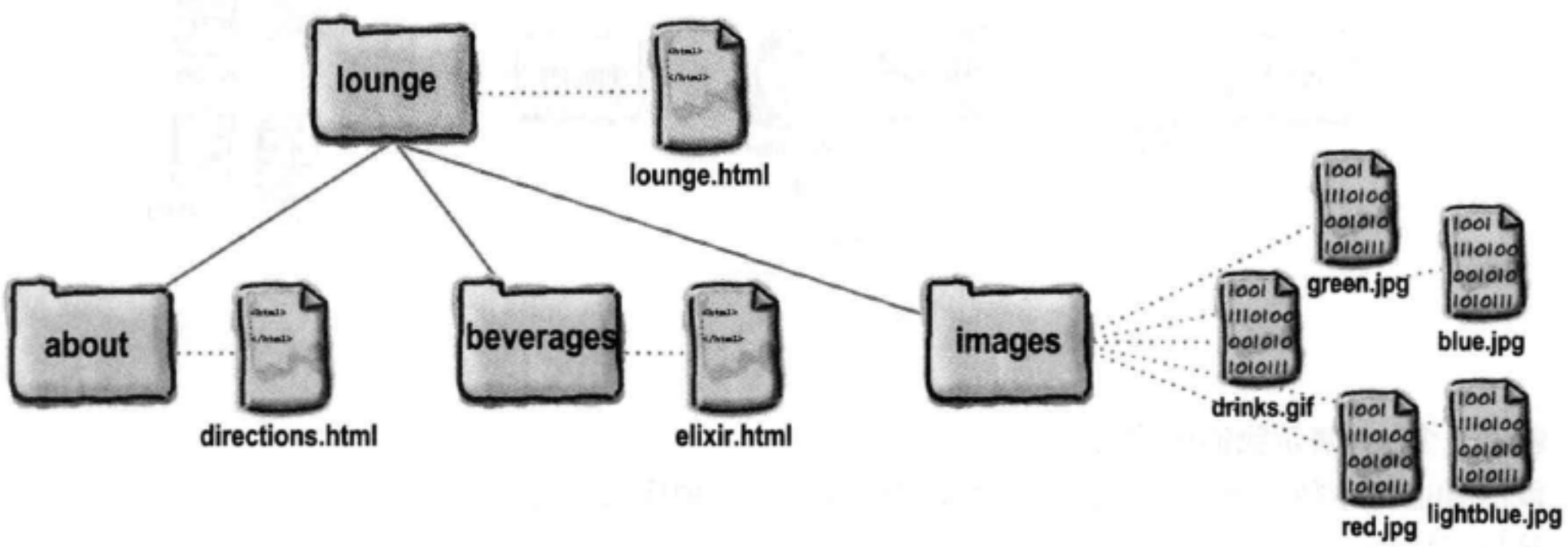


④ 创建一个href表示我们追踪到的路径。

既然知道了路径，就需要把它写成浏览器能理解的一种格式。可以这样写路径：

Sharpen your pencil

该轮到你了：追踪从“lounge.html”到“directions.html”的相对路径。找到后，完成下面的<a>元素。对照这一章最后给出的答案检查你做的对不对，然后修改“lounge.html”中的这两个<a>元素。




```
<a href="                                        ">detailed directions</a>
```

把你的答案写在这里。

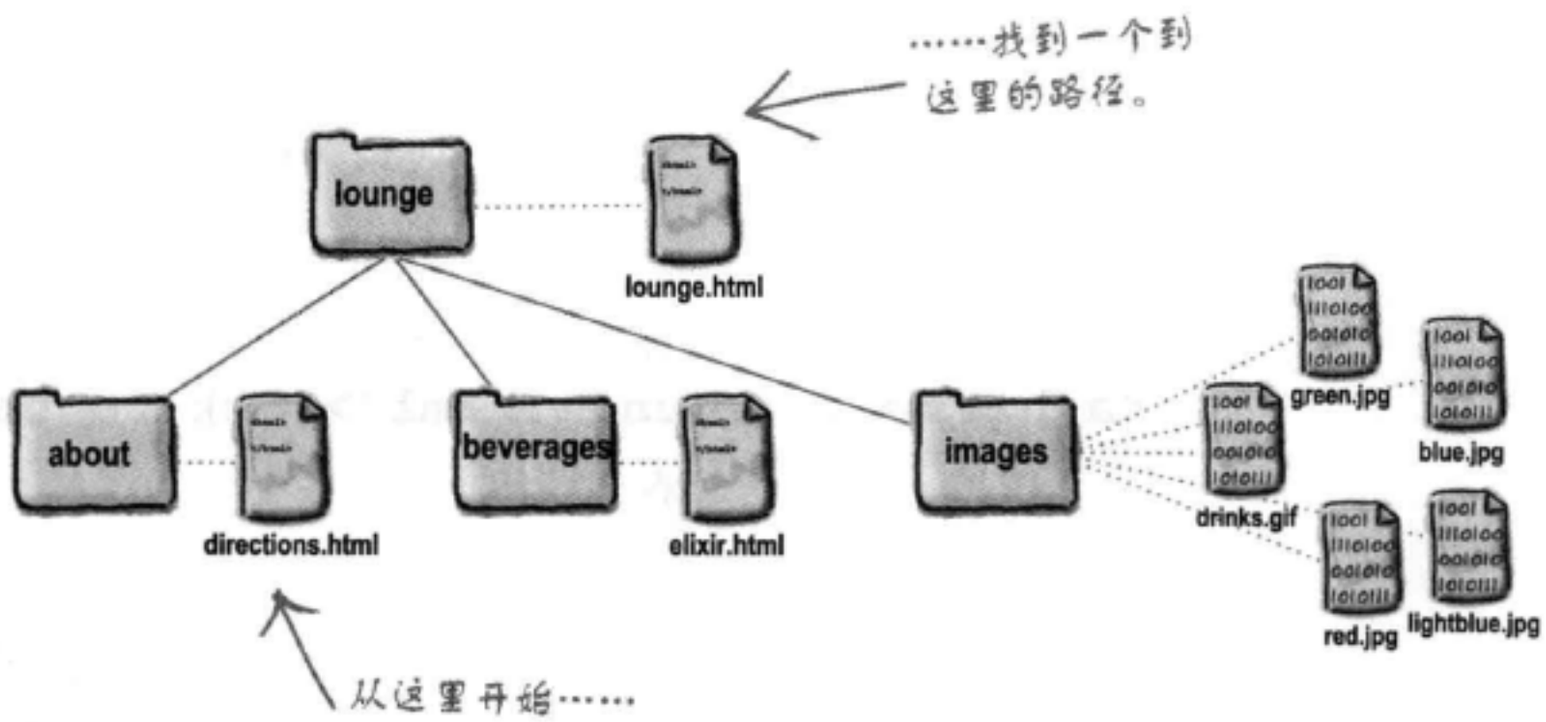
换个方向，向上链接到一个“父”文件夹

① **从“directions.html”链接到“lounge.html”。**
现在需要修复那些“Back to the Lounge”链接。“directions.html”文件中的<a>元素是这样的：

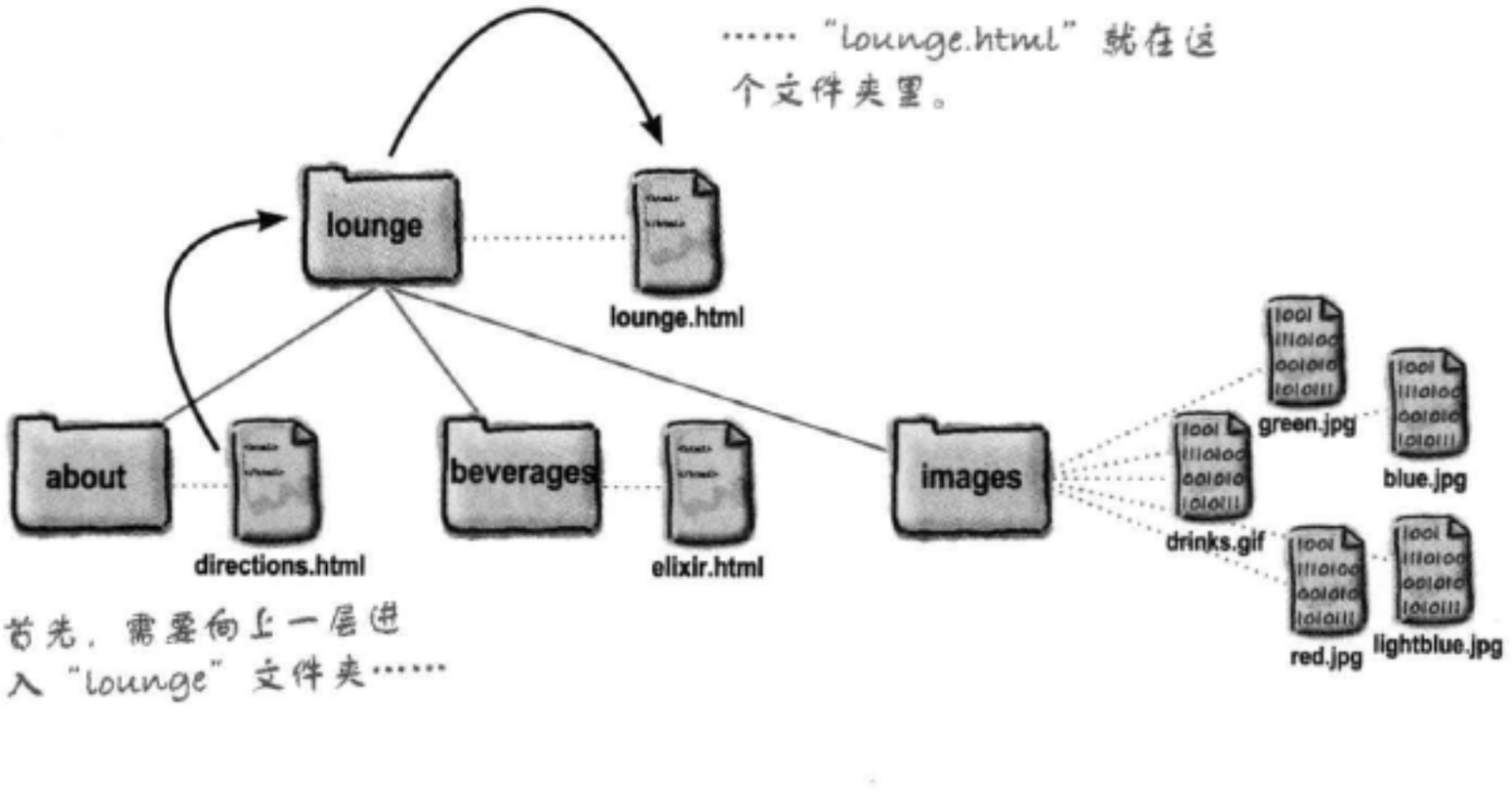
```
<a href="lounge.html">Back to the Lounge</a>
```

现在我们只使用了文件名“lounge.html”，这会告诉浏览器要在“directions.html”所在的相同文件夹里查找这个文件。肯定找不到。

② **明确源文件和目标文件。**
下面来看源文件和目标文件。源文件现在是“directions.html”文件，它在“about”文件夹下。目标文件是“lounge.html”文件，它在“directions.html”所在的“about”文件夹之上。

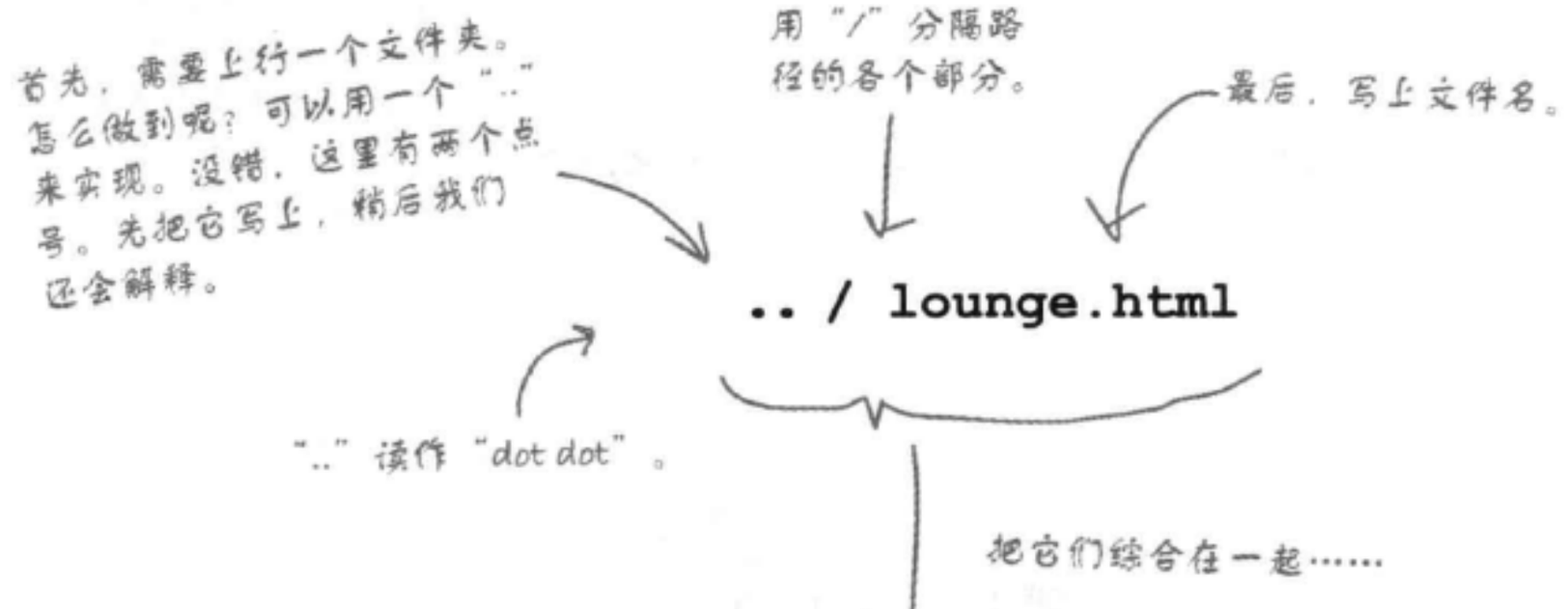


③ **追踪从源文件到目标文件的一个路径。**
下面来追踪这个路径。要从“directions.html”文件到“lounge.html”，需要向上进入“lounge”文件夹，可以在这个文件夹中找到“lounge.html”。



④ 创建一个href表示我们追踪到的路径。

差不多了。现在你已经知道了路径，需要把它写为浏览器能理解的一种格式。下面就来建立这个路径：




```
<a href="../lounge.html">Back to the Lounge</a>
```

现在单击这个链接时，浏览器会在上一层文件夹中查找"lounge.html"文件。

there are no Dumb Questions

问：什么是父文件夹？如果一个“fruit”文件夹里有一个“apples”文件夹，“fruit”就是 “apples”的父文件夹，是吗？

答：完全正确。文件夹（你可能还听过有人把它们叫做目录）通常按家族关系来描述。例如，就用你的例子来说，“fruit”是“apples”的父文件夹，“apples”是“fruit”的子文件夹。如果还有另外一个文件夹“pears”作为“fruit”的一个子文件夹，那么它就是“apples”的兄弟。可以把这想象成是一个家族树。

问：OK，我懂了，不过“..”是什么呢？

答：需要告诉浏览器你要链接的文件在父文件夹中时，可以用“..”表示“向上移至父文件夹”。换句话说，对浏览器来讲，这就是“父文件夹”。

在我们的例子中，我们希望从“directions.html”（在“about”文件夹中）链接到“lounge.html”（在“lounge”文件夹中，这也是 “about”的父文件夹）。所以必须告诉浏览器查看上一级文件夹，而告诉浏览器上行的方法就是使用“..”。

问：如果需要向上两级文件夹而不只是一级该怎么办呢？

答：每次要上行到父文件夹时都可以使用“..”。每一次使用“..”，就会向上到上一层父文件夹。所以，如果你想上行两级文件夹，可以输入“../..”。仍然要用“/”分隔每一部分，所以不要忘记这一点（浏览器可不知道“....”是什么意思）。

问：我上行了两个文件夹后，怎么告诉浏览器在哪里查找文件呢？

答：要把“../..”和文件名结合起来。所以，如果要链接到一个名为“fruit.html”的文件，它在向上两级的文件夹中，就可以写为 “../../fruit.html”。你可能觉得我们要把“../..”称为“祖父”文件夹，不过我们一般不这么讲，而是会说“父文件夹的父文件夹”，或者简称为“../..”。

问：我能向上走多远，有限制吗？

答：你能一直向上走到网站的根目录。在我们的例子中，根目录就是“lounge”文件夹。所以，最远只能向上走到“lounge”。

问：另一个方向呢，我能向下经过多少个文件夹，有限制吗？

答：嗯，你创建了多少层文件夹，就能下行多少层。如果你创建了10层，就可以写一个带你下行10个文件夹的路径。不过我们可不推荐这么做，如果真的有那么多层文件夹，可能说明你的网站组织太过复杂！

另外，有些浏览器对路径中的字符个数会做出限制。规范中建议当心超过255个字符，但现代浏览器能支持更大的长度。不过，如果你有一个大型网站，则还是要注意这一点。

问：我的操作系统使用“\”作分隔符；我是不是应该用“\”而不是“/”？

答：不是这样的。对于网页，只能使用“/”（斜线）。不要用“\”（反斜线）。不同的操作系统使用不同的文件分隔符（例如，Windows使用“\”而不是“/”），但是在Web上，我们选择了一个通用分隔符，所有操作系统都要使用这个分隔符。所以，不管你使用Mac、Windows、Linux还是其他操作系统，在HTML的路径中都要使用“/”。

该轮到你了：追踪“Back to the Lounge”链接从“elixir.html”到“lounge.html”的相对路径。这与“directions.html”文件中的那个返回主页面的链接有什么不同？

答案：没有不一样，它们完全相同。

修复那些损坏的图像……

休闲室基本上已经走入正轨了，接下来只需要修复那些不能正常显示的图像。

我们还没有深入分析<img>元素（后面几章会详细介绍），不过现在你只需要知道<img>元素的src属性有一个相对路径值，类似于href属性。

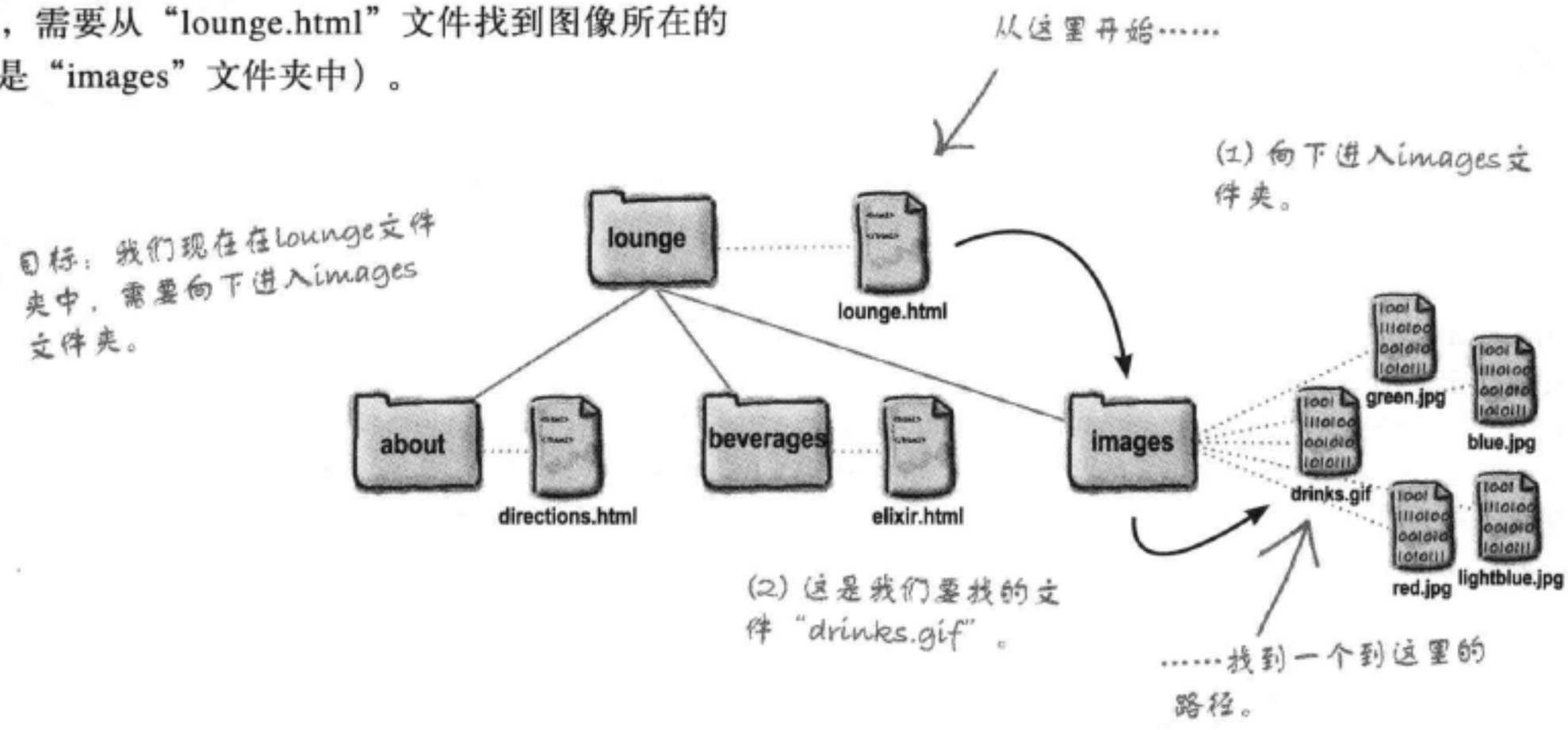
以下是“lounge.html”文件中的image元素：

```
<img src="drinks.gif">
```

这是相对路径，告诉浏览器图像的位置。与指定<a>元素中的href属性类似，我们也要对它做同样的处理。

查找从“lounge.html”到“drinks.gif”的路径

要查找路径，需要从“lounge.html”文件找到图像所在的位置（也就是“images”文件夹中）。

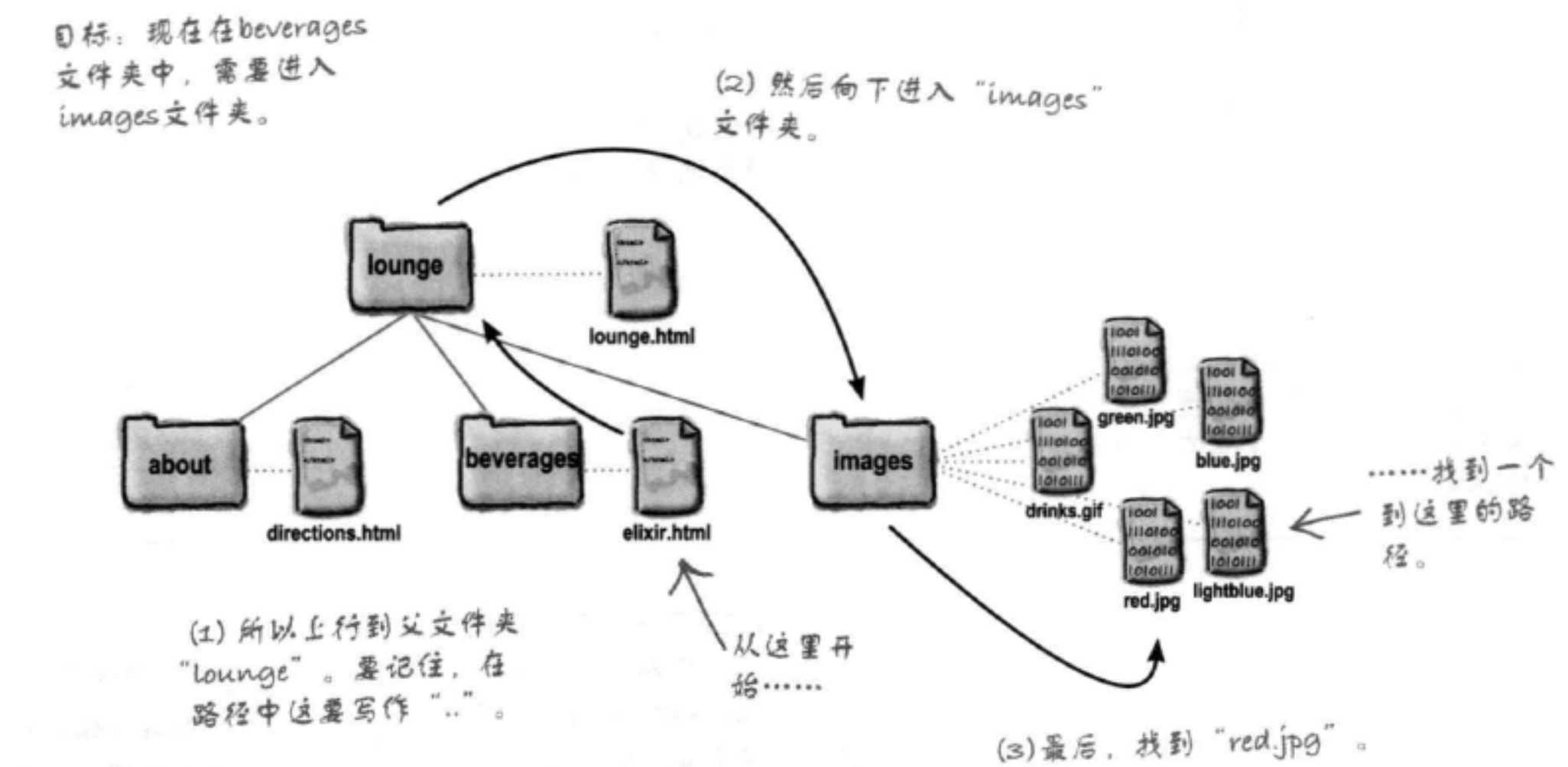


所以，把(1)和(2)综合在一起，可以得到路径“images/drinks.gif”，或

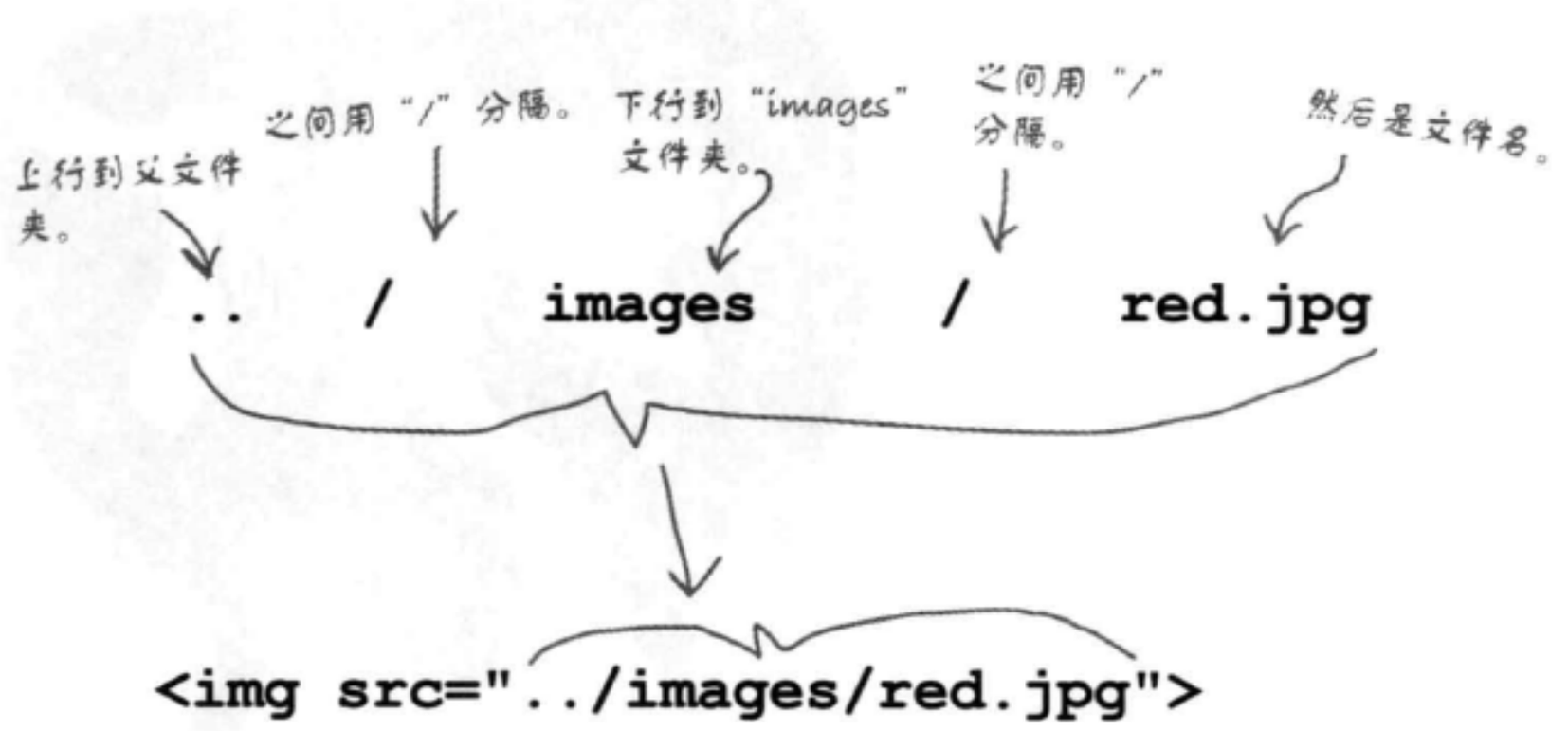
```
<img src="images/drinks.gif">
```

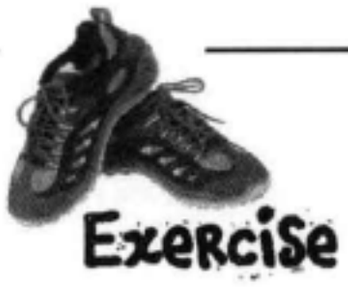
查找从“elixir.html”到“red.jpg”的路径

清凉饮料页面包含多种饮料的图像：“red.jpg”、“green.jpg”、“blue.jpg”等。下面来确定到达“red.jpg”的路径，其余图像的路径都是类似的，因为它们都在同一个文件夹中：



所以，把(1)、(2)和(3)综合在一起，可以得到：





现在已经修复了重组休闲室时破坏的所有链接，不过还需要修复“lounge.html”和“elixir.html”文件中的图像。要完成下面的工作：

1. 在“lounge.html”中，更新图像的src属性值为“images/drinks.gif”。
2. 在“elixir.html”中，更新图像的src属性，在每个图像名前面加上“../images/”。
3. 保存这两个文件，并在浏览器中加载“lounge.html”。现在应该能在页面间顺畅地导航，图像也能正常显示。

又及，如果还有问题，文件夹“chapter2/completelounge”中包含了一个可用的休闲室版本。对照这个版本仔细检查你的文件哪里有问题。

BULLET POINTS

- 想从一个页面链接到另一个页面时，要使用<a>元素。
- <a>元素的href属性指定了链接的目标文件。
- <a>元素的内容是链接的标签。这个标签就是你在网页上看到的链接文本。默认地，这个标签会有下划线，指示这是可以单击的。
- 文字或图像都可以用作链接的标签。
- 单击一个链接时，浏览器会加载href属性中指定的Web页面。
- 可以链接到相同文件夹中的文件，也可以链接到其他文件夹中的文件。
- 相对路径是相对于链接的源Web页面指向网站中其他文件的一个链接。就像在地图上一样，终点总是相对于起点。
- 使用“..”可以链接到源文件上一层文件夹中的一个文件。
- “..”表示“父文件夹”。
- 记住要用“/”（斜线）字符分隔路径中的各个部分。
- 指向一个图像的路径不正确时，会在Web页面上看到一个损坏的图像。
- 为网站选择的文件名和文件夹名中不要使用空格。
- 最好在构建网站初期组织网站文件，这样就不用在网站升级时修改一大堆的路径了。
- 组织网站有很多方法，具体如何组织由你决定。

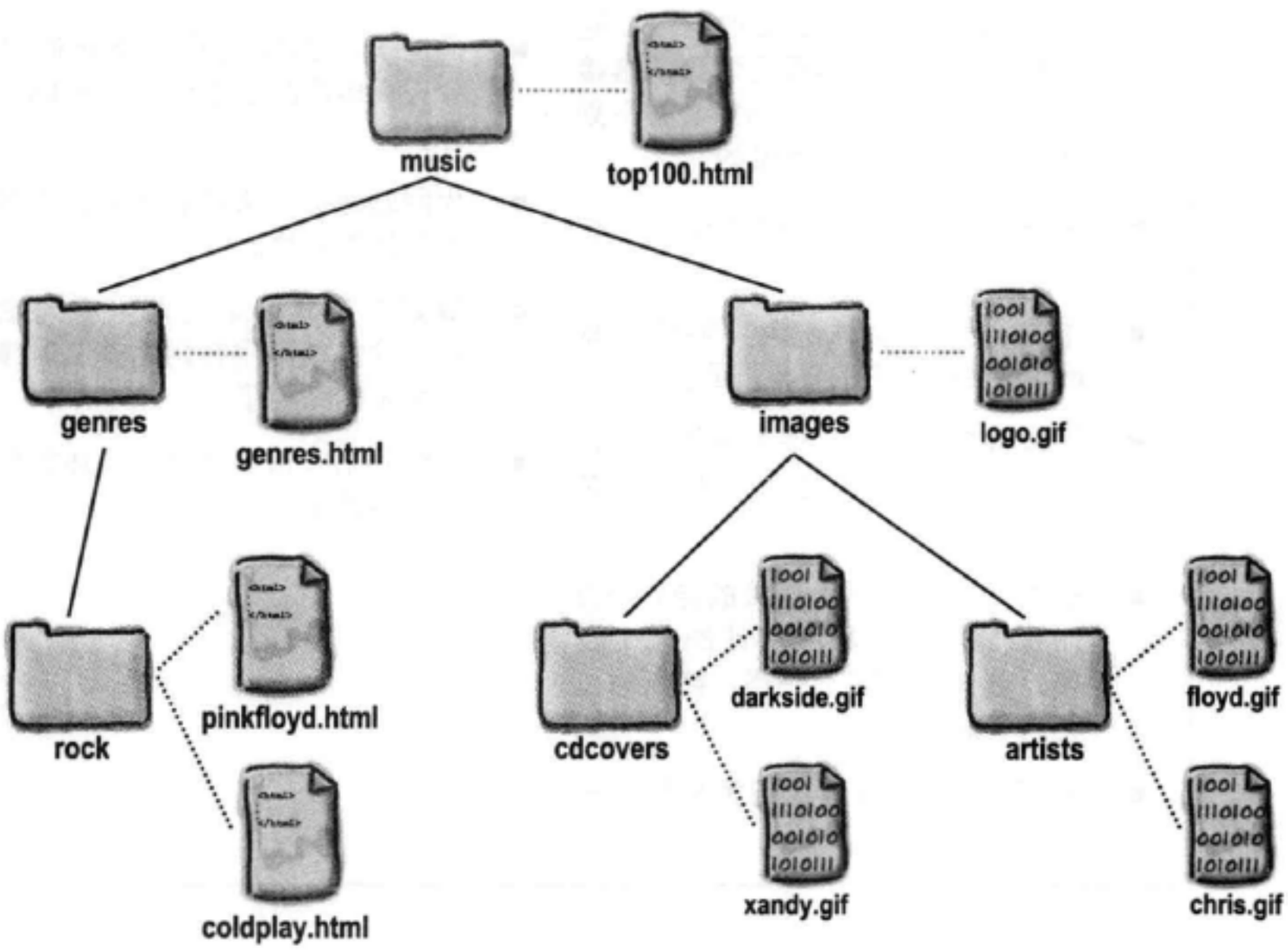
相对性的重大挑战

现在给你一个机会来测试你的相对性技能。我们为100强唱片建立了一个网站，所有文件都放在一个名为“music”的文件夹里。这个文件夹中有HTML文件、其他文件夹，还有图像。你的任务是找出我们需要的相对路径，以便从我们的Web页面链接到其他Web页面和文件。

这一页给出了这个网站的结构，下一页会测试你的能力，你会看到需要完成的任务。你要为每一对源文件和目标文件建立正确的相对路径。如果能成功地找出所有相对路径，则你将在相对路径赛场上夺冠。

祝你好运！

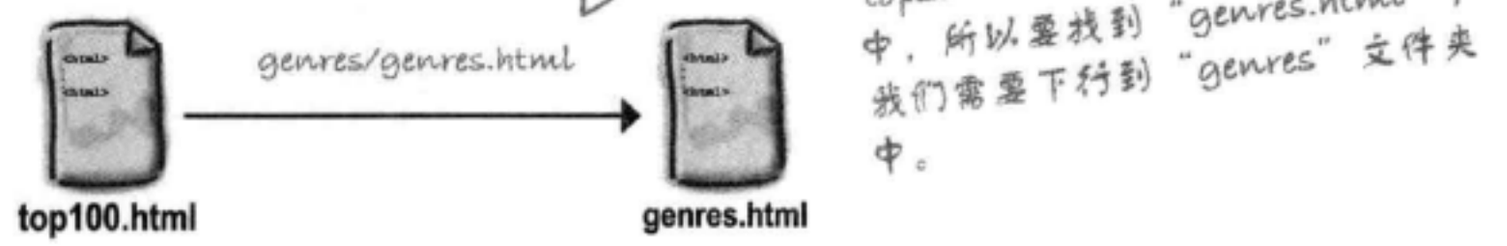
可以直接在这个网站结构图上画出路径。



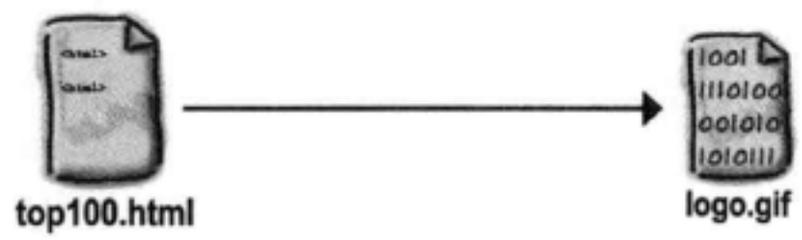
比赛就要开始了。

各就各位……预备……写！

例子

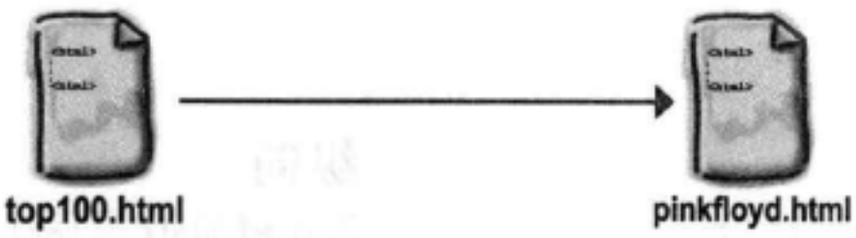


第一轮



第二轮

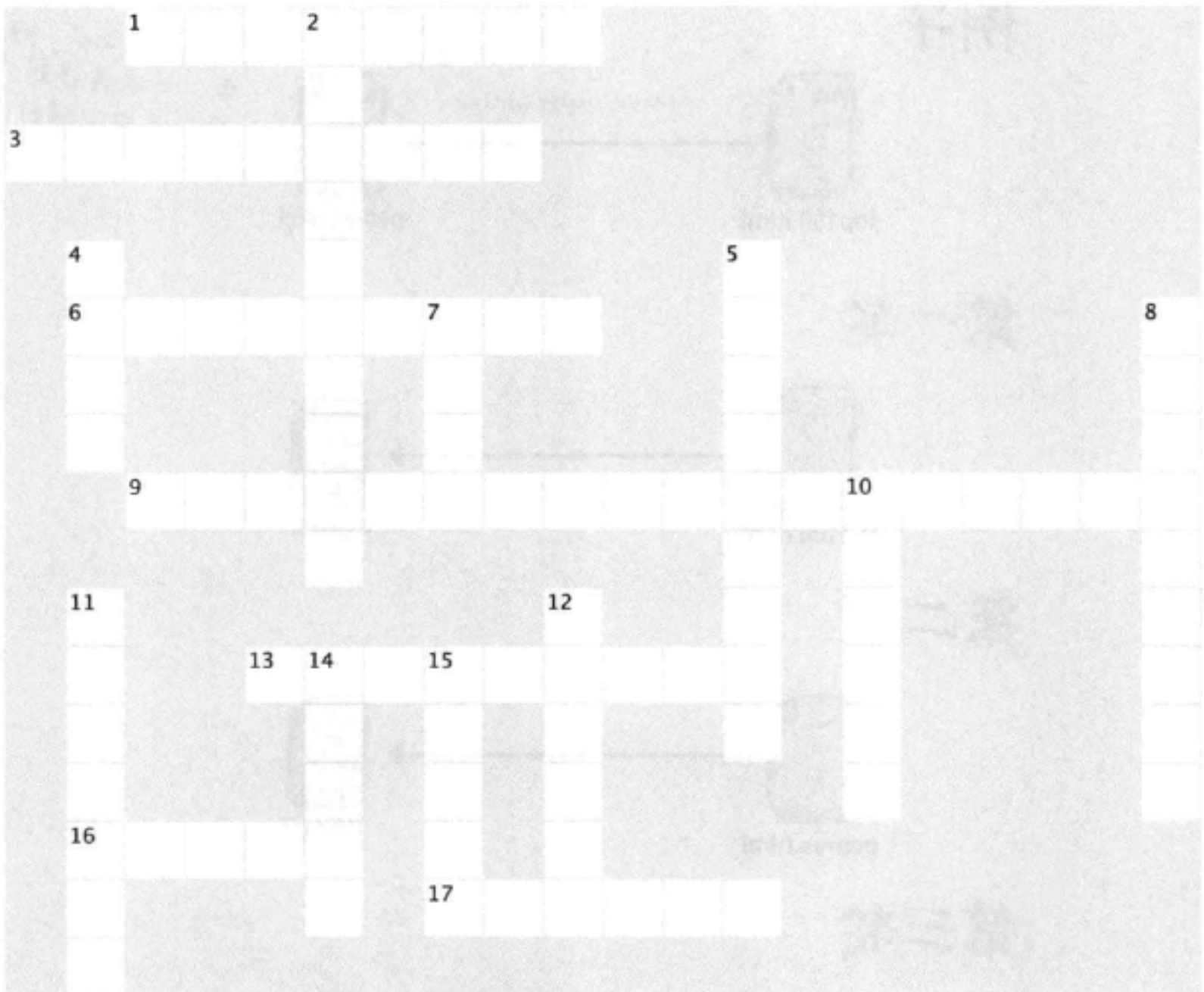
第三轮



加试赛

HTML填字游戏

填字游戏对你学习HTML有什么帮助？嗯，所有这些词都与HTML有关，而且都在这一章出现过。另外，线索里有一些脑筋急转弯，可以帮你从不同角度将HTML牢牢记住！



横向

1. “../myfiles/index.html”是这种链接。
3. 文件夹的别名。
6. blue drink的口味。
9. href表示的意思。
13. <a>和之间的所有内容。
16. 可以放在<a>元素中，就像文本一样。
17. 读作“..”。

纵向

2. href和src是两个_____。
4. Web上工作最勤奋的属性。
5. 与href押韵。
7. 网站的顶层文件夹。
8. HTML中的“HT”。
10. 健康饮料。
11. 同级的文件夹。
12. 使用 ..可以到达_____目录。
14. <a>标记之间的文本作为一个_____。
15. 子文件夹也称为_____。

Exercise Solution

在清凉饮料页面的最下面增加一个标签为“Back to the Lounge”的链接，指回到“lounge.html”。下面是我们的答案。

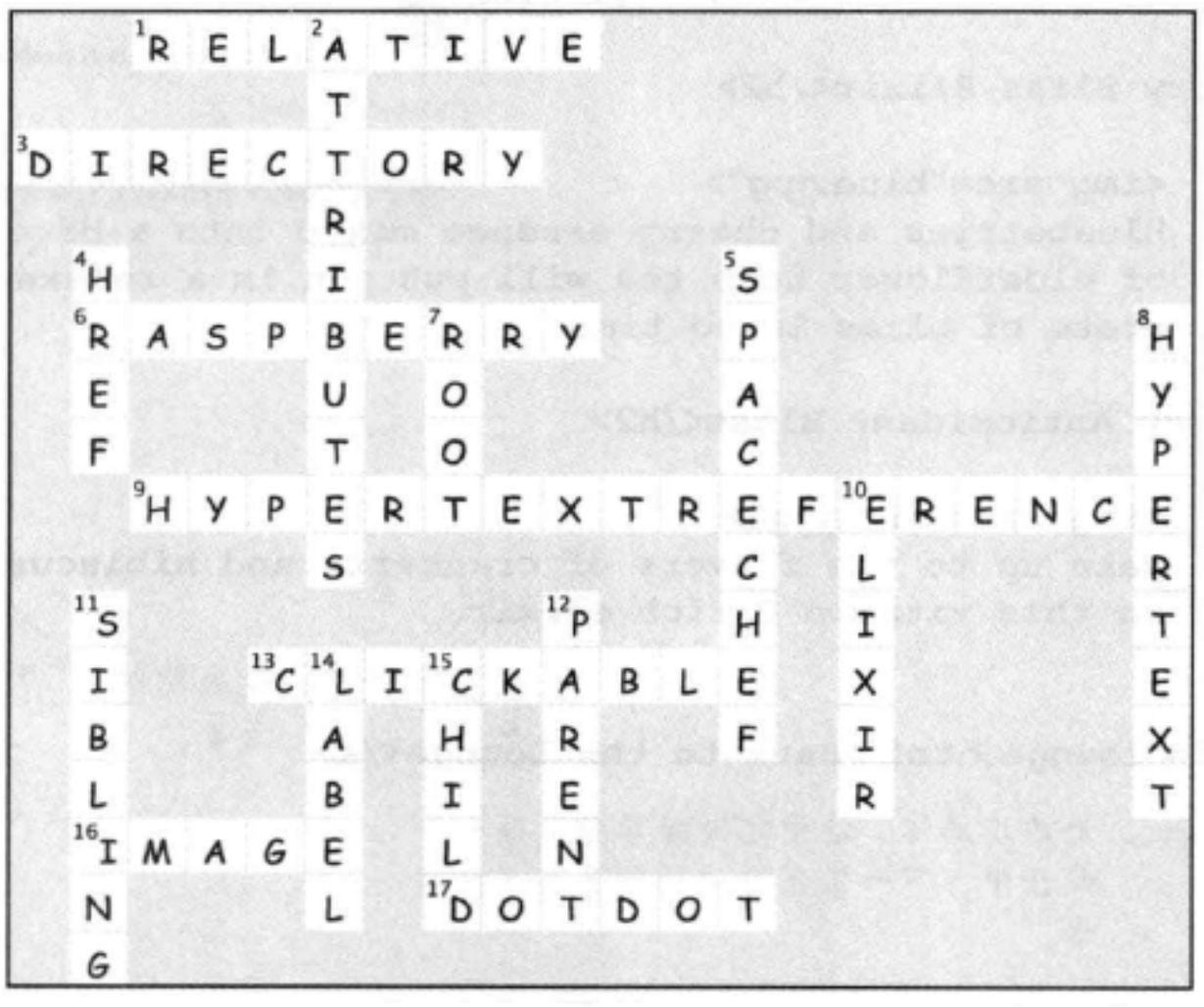
```
<html>
  <head>
    <title>Head First Lounge Elixirs</title>
  </head>
  <body>
    <h1>Our Elixirs</h1>

    <h2>Green Tea Cooler</h2>
    <p>
            <img src="green.jpg">
            Chock full of vitamins and mi
            combines the healthful benef
            a twist of chamomile blossom
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p>
            <img src="lightblue.jpg">
            Combining raspberry juice wi
            citrus peel and rosehips, thi
            will make your mind feel cle
    </p>
    <h2>Blueberry Bliss Elixir</h2>
    <p>
            <img src="blue.jpg">
            Blueberries and cherry essence mixed into a ba
            of elderflower herb tea will put you in a relaxe
            state of bliss in no time.
    </p>
    <h2>Cranberry Antioxidant Blast</h2>
    <p>
            <img src="red.jpg">
            Wake up to the flavors of cranberry and hibiscus
            in this vitamin C rich elixir.
    </p>
    <p>
        <a href="lounge.html">Back to the Lounge</a>
    </p>
  </body>
</html>
```

这是新加的<a>元素，指回到休闲室页面。

为力求简洁，这里把链接单独放在一个段落中。下一章还会讨论更多有关的内容。

练习答案

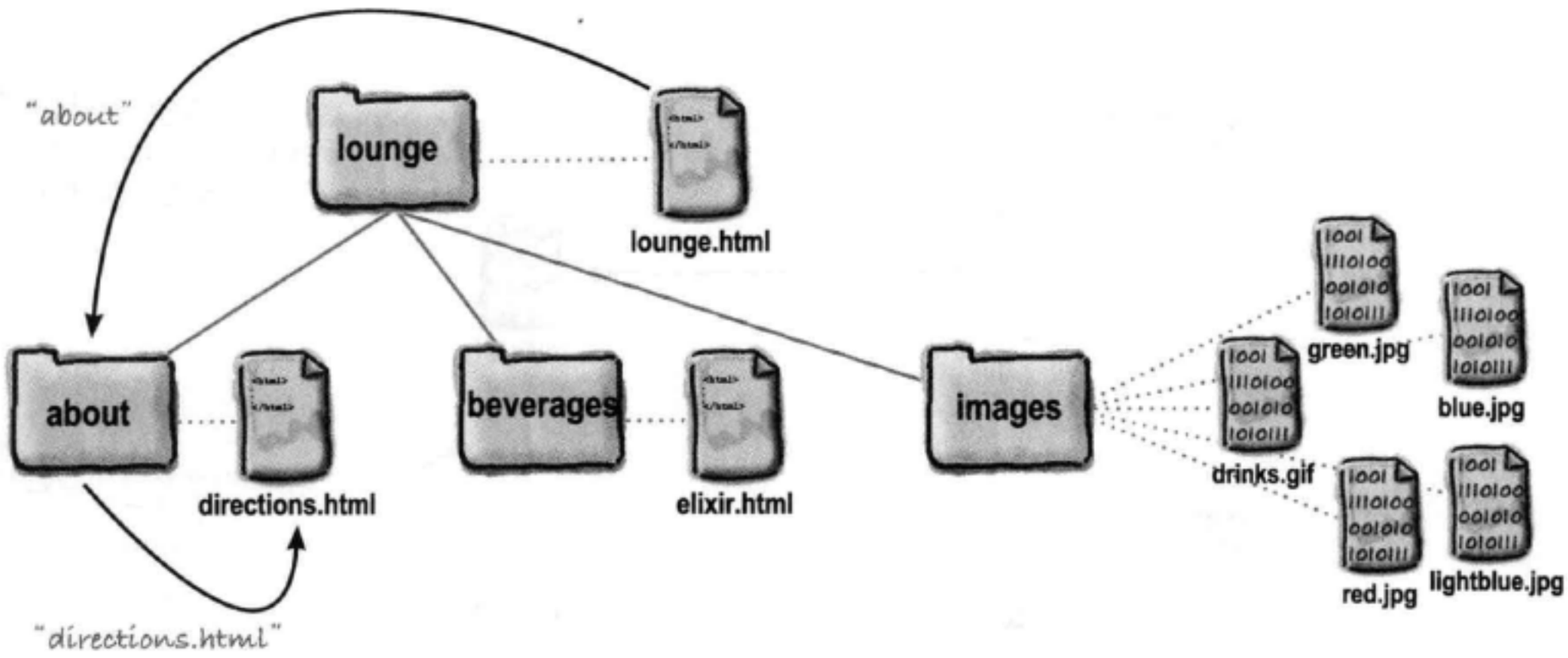
标签	目标文件	元素
Hot or Not?	hot.html	**Hot or Not?**
Resume	**cv.html**	Resume
Eye Candy	**candy.html**	**Eye Candy**
See my mini	**mini-cooper.html**	See my mini
let's play	millionaire.html	**** let's play ****



Sharpen your pencil Solution

追踪从“lounge.html”到“directions.html”的相对路径。找到后，完成下面的<a>元素。

下面是我们的答案。“lounge.html”中的两个<a>元素都改了吗？




```
<a href="about/directions.html">detailed directions</a>
```

答案写在这里

相对性的重大挑战答案

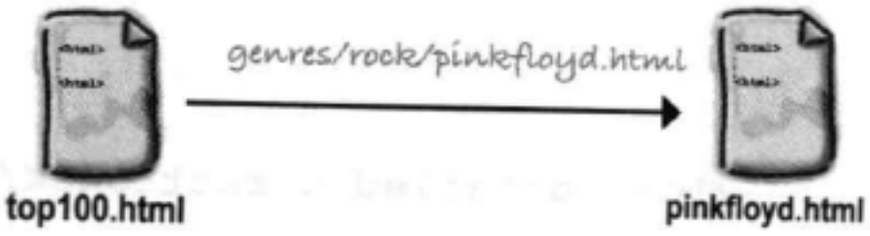
第一轮

top100.html在music文件夹中，所以要找到logo.gif，需要向下进入images文件夹。

第二轮

genres.html在genres目录中，所以要找到logo.gif，首先必须上行到music文件夹，然后向下进入images文件夹。

第三轮



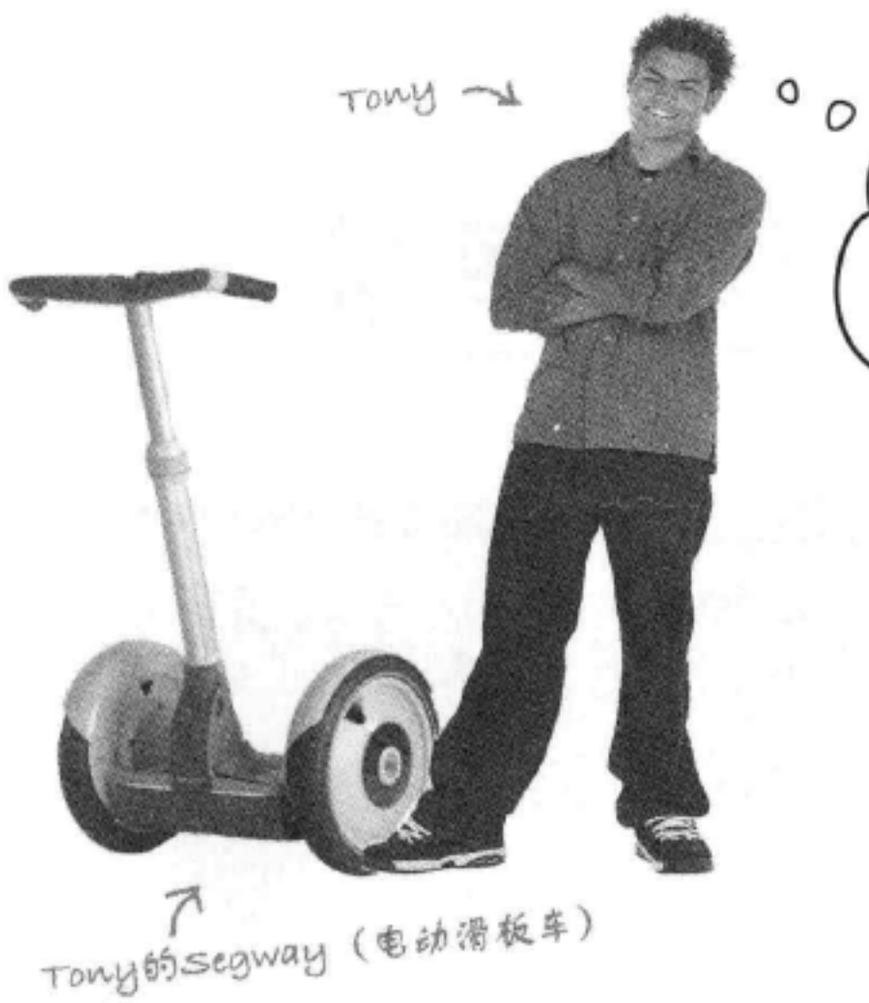
从top100.html需要下行到genres文件夹，然后进一步下行到rock目录，在这里可以找到pinkfloyd.html。

加试赛

这一题比较难。从coldplay.html（它在rock文件夹中），必须上行两层文件夹进找到music目录，然后下行进入images，最后再下行到artists目录，在这里才能找到图像chris.gif。哟，真不容易！

Web页面建设

我听说这本书会教我具体创建Web页面？你已经学到不少了：标记、元素、链接、路径……不过，如果不学以致用，创建一些一流的Web页面，这些知识就毫无意义。这一章我们要把重点放在建设上：你要从概念到蓝图来设计Web页面，浇注地基，完成建设，甚至最后还可以加几笔润色。带上安全帽，系好安全带，我们会增加一些新工具，让你掌握一些更深入的内部知识，有了这些知识，你也能成为这个行业的佼佼者。



Tony的Segway（电动滑板车）

要痛痛快快地玩我的新滑板车（Segway），再没有比骑着它上马路驰骋更过瘾的了！我要骑着它周游整个美国，还会把路上的所见所闻记在我的旅行日志里。我想把它们放到网页上，这样我的朋友和家人就能看到了。

Tony的旅行日志

Segway' n USA

Documenting my trip around the US on my very own Segway!

June 2, 2012

My first day of the trip! I can' t believe I finally got everything packed and ready to go. Because I' m on a Segway, I wasn' t able to bring a whole lot with me: cell phone, iPod, digital camera, and a protein bar. Just the essentials. As Lao Tzu would have said, "A journey of a thousand miles begins with one step Segway."

July 14, 2012

I saw some Burma Shave style signs on the side of the road today: "Passing cars, When you can' t see, May get you, A glimpse, Of eternity." I definitely won' t be passing any cars

August 20, 2012

Well I made it 1200 miles already, and I passed through some interesting places on the way: Walla Walla, WA, Magic City, ID, Bountiful, UT, Last Chance, CO, Why, AZ, and Truth or Consequences, NM.

一定要好好读读Tony的旅行日志，这一章会随处用到这些日志。

从日志到网站，以12迈的速度出发

← Segway建议的最高时速。

Tony在全力准备骑着他的Segway周游美国，已经无暇顾及其他了。你为什么不帮他一把，为他创建一个网页，好吗？

你要完成下面的工作：

1. 首先，画一个粗略的日志草图,这将作为页面设计的基础。
2. 接下来，创建HTML的基本构建模块（<h1>、<h2>、<h3>、<p>等），把你的草图翻译成HTML页面的略图（或蓝图）。
3. 一旦有了蓝图，接下来就可以把它翻译成真正的HTML了。
4. 最后，在完成的基本页面上还可以加一些改进，顺便认识一些新HTML元素。

Sharpen your pencil

先停一下！翻开下一页之前先完成这个练习。

仔细研究Tony的日志，想想看如何把这些信息展示在Web页面上。

在右边画出这个页面。不需要很漂亮、很精致，只需要一个大致的草图就可以了。假设他的所有日志都放在一个页面上。

从下面几个方面来考虑：

- 从大的结构元素来考虑页面的构成：标题、段落、图像，等等。
- 有没有办法把他的日志改为更适合在Web上发布。

在这里画出你的草图。

粗略的设计草图

Tony的日志看起来很像一个网页，要创建设计草图，我们要做的就是把他的所有日志都放在一个页面上，并安排好总体组织布局。看起来如果Tony某一天写日志，他会先写当天的日期作为标题，然后可能有一个图片，最后是对当天见闻的描述。下面来看这个草图……

Tony的旅行日志有一个大标题"Segway' n USA"，我们要把它作为标题放在页面顶部。

他还为旅行日志写了一个介绍。我们将把它作为一个小段落放在最上面。

每天Tony创建一个日志，包括日期，往往还有一个图片，最后是当天见闻的描述。所以要有一个标题、一个图像和另外一个文本段落。

有时他没有加图片。在这个日志中，只有一个标题（日期）和当天见闻的描述。

第3个日志与第1个看上去很类似，也有一个标题、一个图像和一个段落。

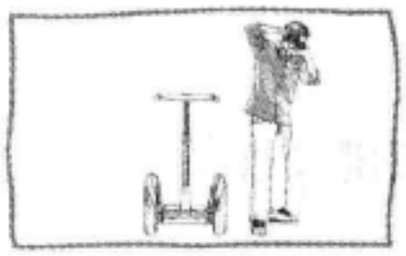
与Tony手写的日志不同，我们的页面长度是无限的，所以可以把很多个日志放在同一个Web页面上。他的朋友和家人只需要使用滚动条来浏览他的日志……

不过，注意我们把日志的顺序倒换了一下，这里按从最新到最早的顺序排列。这样一来，最新的日志会出现在页面最上面，用户不用滚动就能直接看到。

Segway' n USA

Documenting my trip around the US on my very own Segway!

August 20, 2012

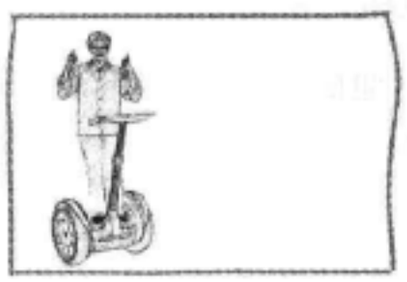


Well I made it 1200 miles already, and I passed through some interesting places on the way: Walla Walla, WA, Magic City, ID, Bountiful, UT, Last Chance, CO, Why, AZ and Truth or Consequences, NM.

July 14, 2012

I saw some Burma Shave style signs on the side of the road today: "Passing cars, When you can't see, May get you, A glimpse, Of eternity." I definitely won't be passing any cars.

June 2, 2012

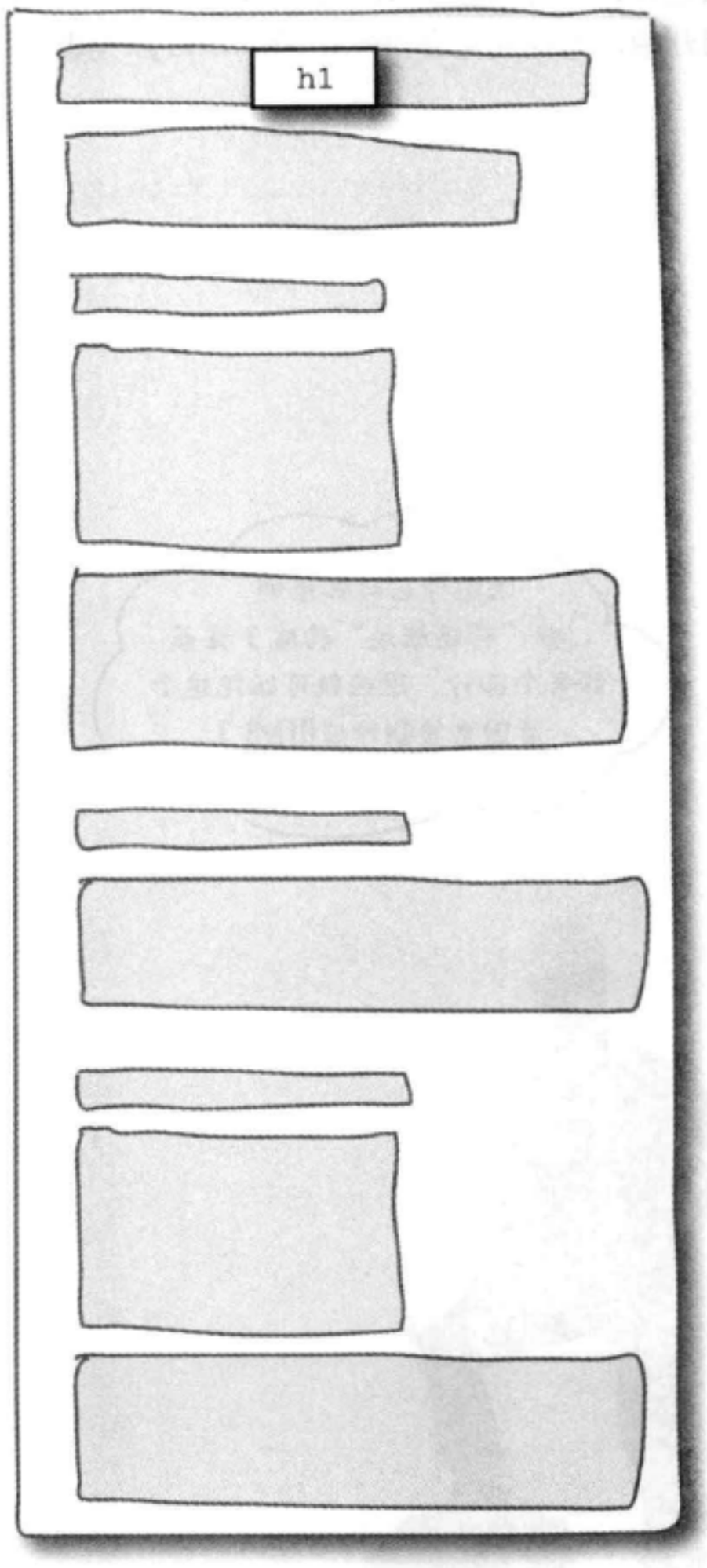


My first day of the trip! I can't believe finally got everything packed and ready to go. Because I'm on a Segway, I wasn't able to bring a whole lot with me: cell phone, iPod, digital camera, and a protein bar. Just the essentials. As Lao Tzu would have said, "A journey of a thousand miles begins with one Segway."

从草图到略图

既然有了页面的草图，下面就来分别研究各个部分，画出更精细的HTML页面略图（或蓝图）……

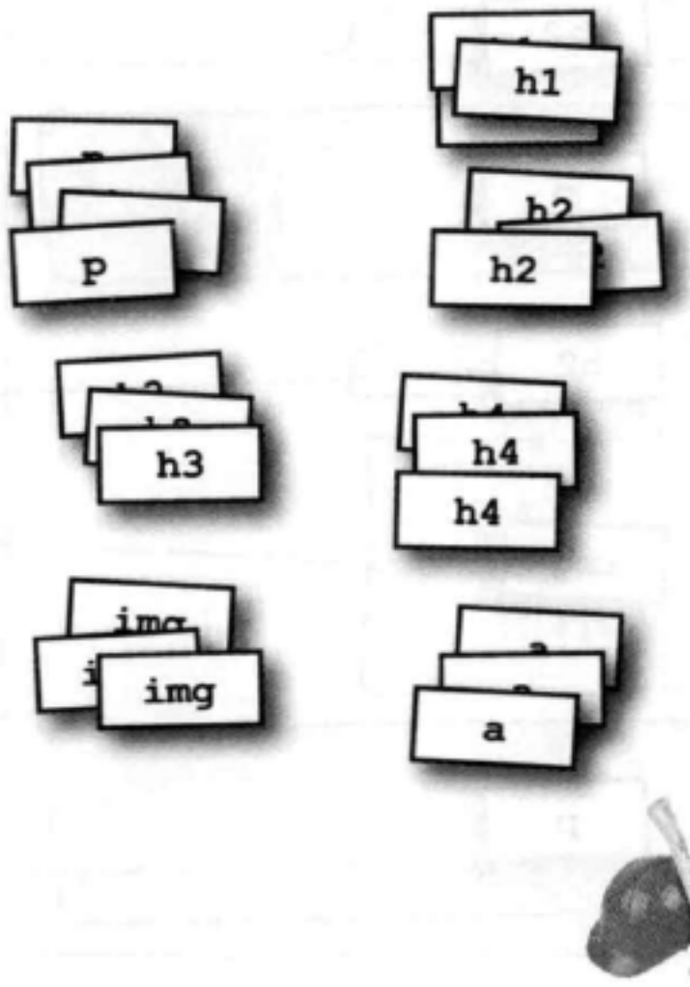
取草图的各个部分，在蓝图中创建相应的内容区。



现在要做的就是确定哪些HTML元素对应各个内容区，然后才能开始写HTML。

练习：Web构建

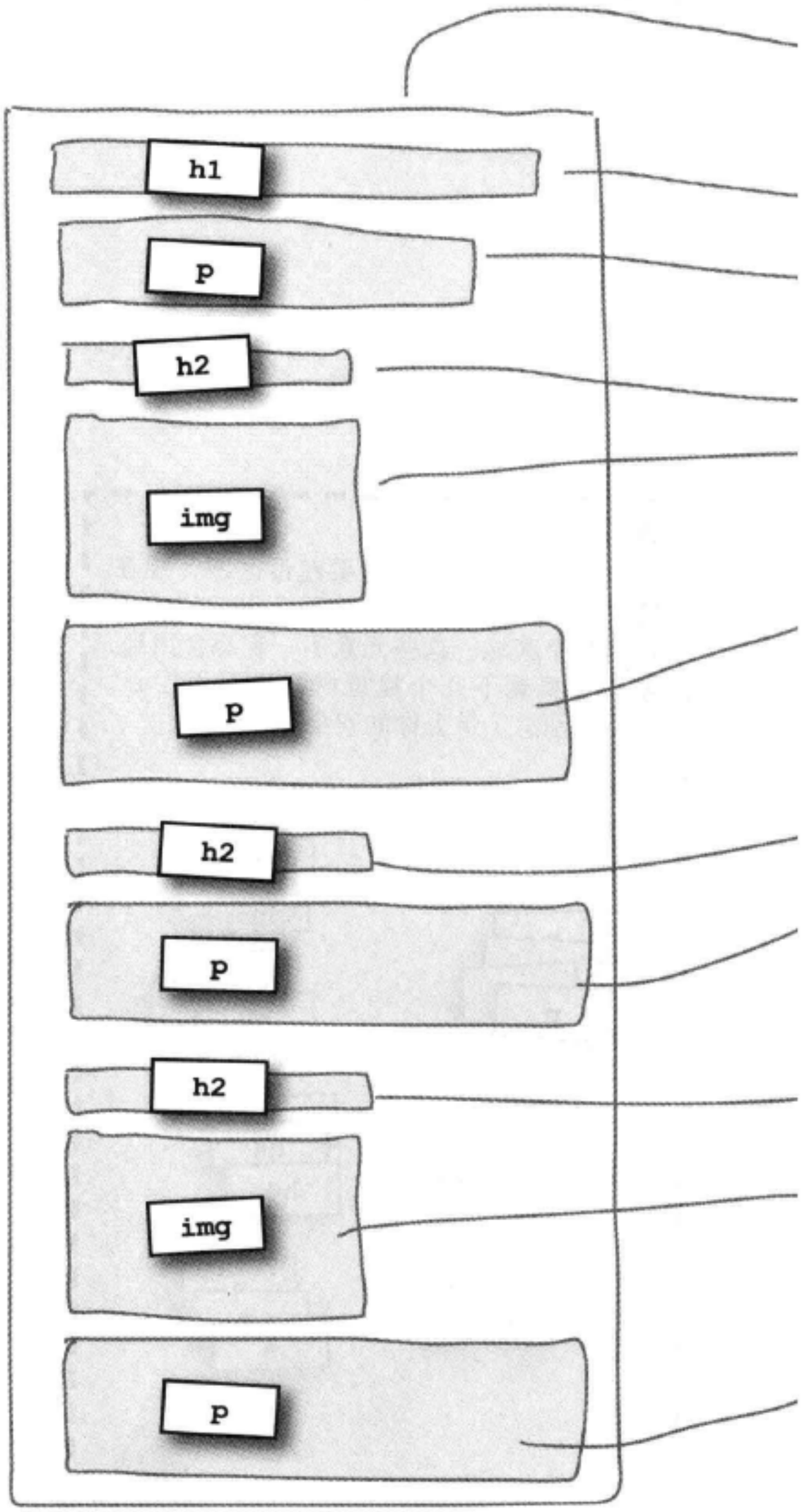
你已经确定了页面的主要建设区域，现在只需要落实建筑材料。使用下面的元素来标注各个区域。这些元素不一定都会用到，所以如果剩下几个建筑材料也不用担心。另外，别忘了带上你的安全帽。



从略图到网页

胜利在望了。你已经为Tony的网页创建了一个略图。现在只需要创建相应的HTML来表示这个页面，并填入Tony的文字。

开始之前，要记住每个Web页面都需要从<html>元素开始，而且要包含<head>和<body>元素。



不要忘了，<html>、<head>、<title>和<body>元素是必不可少的。

我们使用旅行日志的大标题作为这个web页面的标题。

```
<html>
  <head>
    <title>My Trip Around the USA on a Segway</title>
  </head>
  <body>

    <h1>Segway'n USA</h1>
    <p>
        Documenting my trip around the US on my very own Segway!
    </p>

    <h2>August 20, 2012</h2>
    <img src="images/segway2.jpg">
    <p>
      Well I made it 1200 miles already, and I passed
      through some interesting places on the way: Walla Walla,
      WA, Magic City, ID, Bountiful, UT, Last Chance, CO,
      Why, AZ and Truth or Consequences, NM.
    </p>

    <h2>July 14, 2012</h2>
    <p>
      I saw some Burma Shave style signs on the side of the
      road today: "Passing cars, When you can't see, May get
      you, A glimpse, Of eternity." I definitely won't be passing
      any cars.
    </p>

    <h2>June 2, 2012</h2>
    <img src="images/segway1.jpg">
    <p>
      My first day of the trip! I can't believe I finally got
      everything packed and ready to go. Because I'm on a Segway,
      I wasn't able to bring a whole lot with me: cell phone, iPod,
      digital camera, and a protein bar. Just the essentials. As
      Lao Tzu would have said, "A journey of a thousand miles begins
      with one Segway."
    </p>

  </body>
</html>
```

这里是Tony旅行日志的标题和描述。

标题

图像

描述

这里是Tony的最新日志。

这是第二个日志，这里没有图像。

最下面是Tony的第一个日志，包含图像 "segway1.jpg"。

最后，但绝不是最不重要，不要忘记结束<body>和<html>元素。

输入上面的HTML。把你的文件保存到 "chapter3/journal" 文件夹，文件名为 "journal.html"。你会发现 "segway1.jpg" 和 "segway2.jpg" 已经在 "images" 文件夹中。完成后，对这个页面做个测试。

测试Tony的Web页面

看看这个页面怎么样。Tony的所有日志都已经放在一个结构清晰、简单易读的网页中。

太棒了！看起来很不错。我真是迫不及待地想在我的页面里增加更多日志。

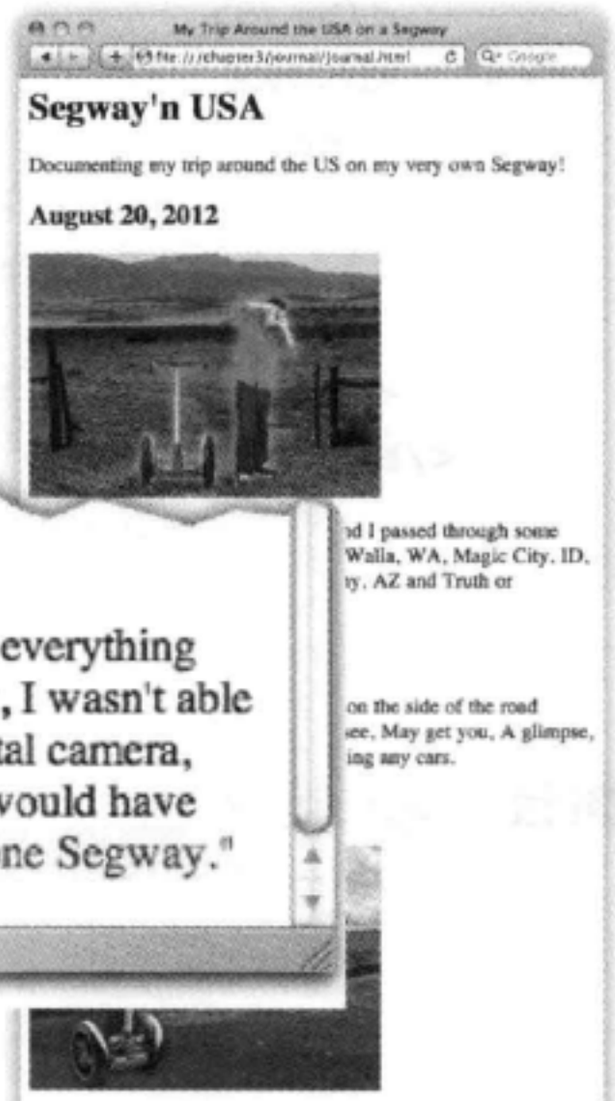
Tony在路上打来电话……

增加一些新元素

你已经掌握了HTML的基本元素，而且只经过简单的几步，你已经利用这些基本HTML元素（<p>、<h1>、<h2>和<img>）把手写的旅行日志变成了网上版本。

现在我们要让你的思维再拓宽一点，增加一些更常见的元素。下面再来看Tony的旅行日志，看看哪里还可以加以改进……

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



注意这里，Tony在第一篇日志的结尾引用了一小段文字。他在这里改写了老子的一句话："A journey of a thousand miles begins with one Segway"（千里之行，始于Segway）。

HTML有一个元素<q>可以支持这种引用。翻到一下页，我们来分析这个元素……

认识 <q> 元素

想在HTML里加段简短的引用？<q>元素正是你需要的。下面给出一小段测试HTML，来说明它是如何工作的：

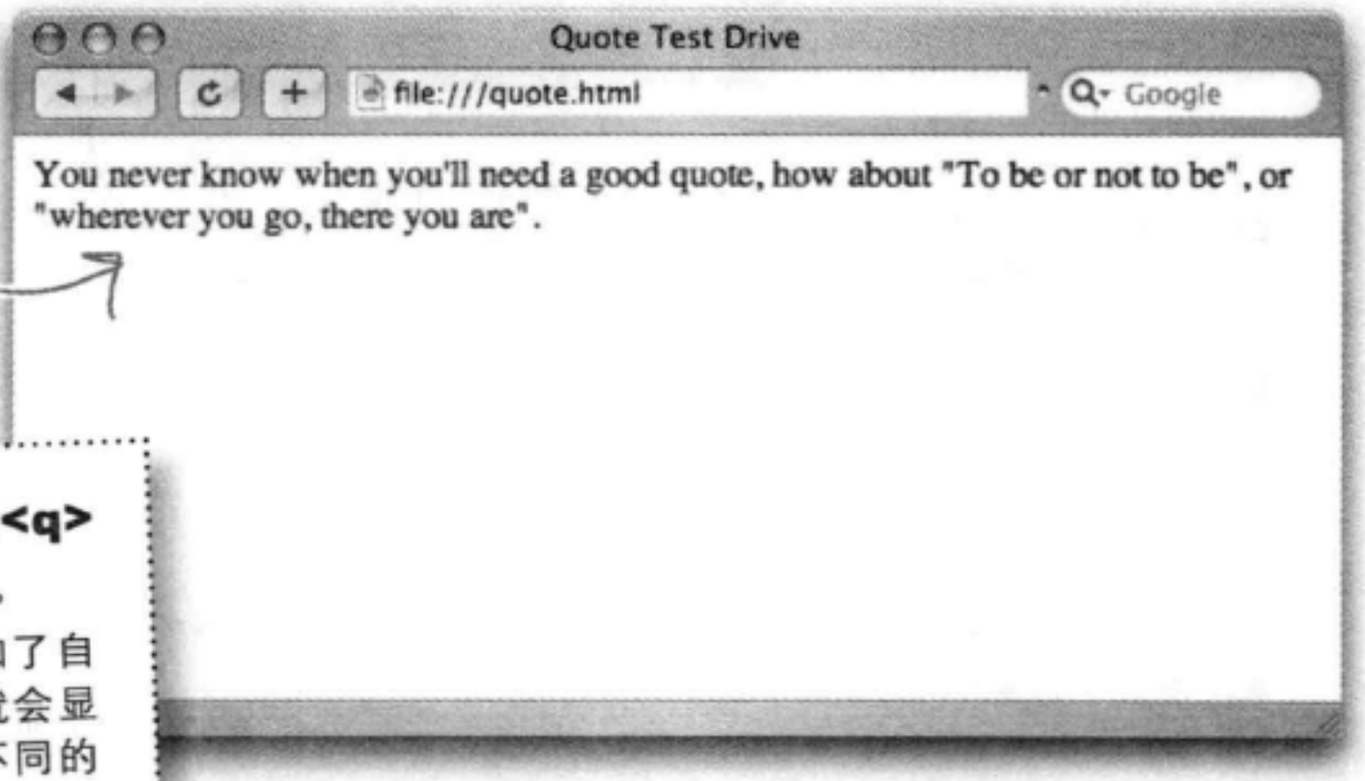
```
<html>
  <head>
    <title>Quote Test Drive</title>
  </head>
  <body>
    <p>
      You never know when you'll need a good quote, how
      about <q>To be or not to be</q>, or <q>Wherever you go, there you are</q>.
    </p>
  </body>
</html>
```

这个HTML中有两个引用……

每个引用都要用一个<q>开始标记和一个</q>结束标记包围。注意，这里没有在引用周围加双引号。

测试

这些引用在浏览器中会这样显示。注意浏览器会负责加上双引号。



Watch it!

并不是所有浏览器都会在<q>元素的内容两边加双引号。

这很糟糕，因为如果你增加了自己的双引号，有些浏览器就会显示两组双引号。建议你在不同的浏览器中对<q>做个测试，看看会得到什么结果。

等一下……你去掉了双引号，换成了一个<q>元素，只是为了显示双引号？我没看错吧？这样不是更复杂了吗？

不。这样可以让页面更结构化，更有意义。

人们之所以在文本中使用双引号，原因可能有很多，不过如果使用<q>，这就意味着这是某种特定的内容，它表示一段引用文字（在Tony的例子中，则是一段“改写的”引用）。

注意！使用双引号并不代表它确实是引用文字。

换句话说，通过标记引用，我们可以为它增加更多含义。在增加<q>元素之前，浏览器只知道这是一个文本段落，其中包含一些双引号字符。而现在，由于我们使用了<q>元素，浏览器知道了这些文字是真正的引用。

那又如何呢？嗯，既然浏览器知道这是一个引用，它就能用最合适的方式来显示。有些浏览器会在文本两边显示双引号，有些却不会，另外如果浏览器使用非英语语言，可能还会使用其他方法。不要忘记移动设备，比如手机，或者为有视力缺陷人士提供的语音HTML浏览器和大屏幕阅读器。在其他情况下这也很有用，如搜索引擎可能要搜寻包含引用的网页。页面中增加结构和含义绝对是有意义的。

最重要的一个原因是（本书后面谈到表现和CSS时就会看到），你可以对引用设置样式，使它的外观如你所愿。假设你希望引用文字用灰色斜体显示，怎么做呢？如果使用<q>元素在网页中指定了引用内容，就能轻松做到。

Exercise

下面是Tony的旅行日志。修改他引用的老子的话，现在改用<q>元素。在纸上完成后，再修改“journal.html”文件，然后测试这个页面。答案在本章最后给出。

```
<html>
  <head>
    <title>Segway'n USA</title>
  </head>
  <body>

    <h1>Segway'n USA</h1>
    <p>
      Documenting my trip around the US on my very own Segway!
</p>

<h2>August 20, 2012</h2>
<img src="images/segway2.jpg">
<p>
  Well I made it 1200 miles already, and I passed
  through some interesting places on the way: Walla Walla,
  WA, Magic City, ID, Bountiful, UT, Last Chance, CO,
  Why, AZ and Truth or Consequences, NM.
</p>

<h2>July 14, 2012</h2>
<p>
  I saw some Burma Shave style signs on the side of the
  road today: "Passing cars, When you can't see, May get
  you, A glimpse, Of eternity." I definitely won't be passing
  any cars.
</p>

<h2>June 2, 2012</h2>
<img src="images/segway1.jpg">
<p>
  My first day of the trip! I can't believe I finally got
  everything packed and ready to go. Because I'm on a Segway,
  I wasn't able to bring a whole lot with me: cell phone, iPod,
  digital camera, and a protein bar. Just the essentials. As
  Lao Tzu would have said, "A journey of a thousand miles begins
  with one Segway."
</p>
  </body>
</html>
```

出生后就被分开的元素

很多年前，一对孪生兄弟在Web镇降生，由于互联网路由器故障导致的一起意外事件，这对双胞胎出生后不久就分开了。他们渐渐长大，都不知道对方的存在，不过后来通过一系列匪夷所思的巧合，他们终于相遇，认出对方，不过他们决定保守这个秘密。

认出之后，他们很快发现彼此有太多共同之处。他们都娶了一个名叫Citation的妻子。他们都酷爱引用。哥哥`<q>`元素喜欢短引用，而弟弟`<blockquote>`喜欢比较长的引用，通常能记住书上或诗歌中的大段文字。

作为孪生兄弟，他们长相极其相似，所以他们决定制定一个邪恶的计划，相互交换身份。他们先拿自己的妻子做实验（细节不便详述），很成功,他们的妻子都没有发觉（或者至少假装没有发觉）。

接下来，他们想在工作场所测试这个交换计划，恰好他们都做同样的工作：在HTML文档中标记引用。所以，选好日子后，兄弟俩来到对方的工作场所，信心满满地实施他们的邪恶计划（毕竟，既然他们的妻子都没能区分，更何况老板呢），不过，这就出问题了。工作仅仅10分钟后，兄弟俩都被发现是冒名顶替的，并立即向标准权威机构发出了警报。

这对双胞胎怎么被发现的呢？
继续读下去，你会找到更多线索……

很长的引用

既然知道了怎么使用短引用，下面来看长引用。Tony给了我们一个柏玛刮胡膏广告的长引用。

在他的旅行日志里，Tony直接把柏玛刮胡膏引用放在段落里，不过如果把这个引用拿出来放在单独的“块”里不是更好吗？就像这样：

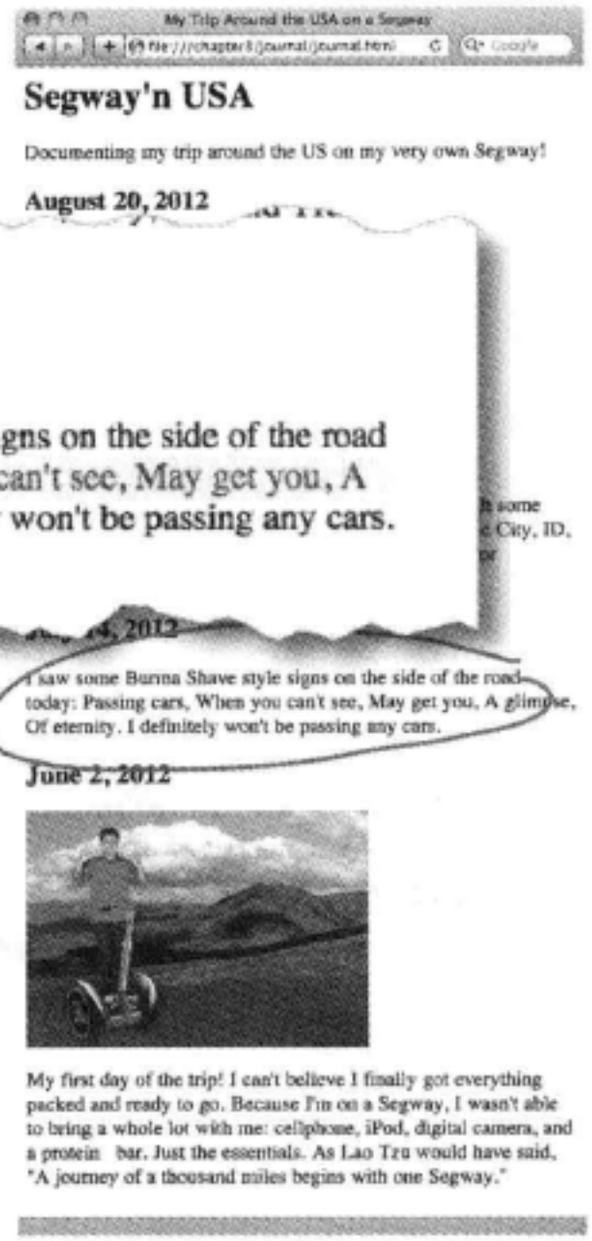
I saw some Burma Shave style signs on the side of the road today:

Passing cars,
When you can't see,
May get you,
A glimpse,
Of eternity.

I definitely won't be passing any cars.

如果你不知道“柏玛刮胡膏”广告，后面几页就会告诉你……

这里就用到了<blockquote>元素。与<q>元素不同（<q>用于短引用，作为现有段落的一部分），<blockquote>元素用于较长的引用，需要单独显示。



增加 <blockquote>

下面在 Tony的网上日志中加入一个<blockquote>。

❶ 打开你的“journal.html”文件，找到7月14日的日志。
修改这个段落，如下所示：

```
<h2>July 14, 2012</h2>
<p>
   I saw some Burma Shave style signs on the
   side of the road today:
</p>
<blockquote>
   Passing cars,
   When you can't see,
   May get you,
   A glimpse,
   Of eternity.
</blockquote>
<p>
   I definitely won't be passing any cars.
</p>
```

要插入<blockquote>元素，需要先结束这个段落。

接下来把柏瑞制胡麦广告词放在<blockquote>元素中。

每行文字都另起一行显示，这样更像柏瑞制胡麦广告。

最后，需要增加一个<p>标记，在<blockquote>之后开始下一个段落。

❷ 再做一次测试。在你的浏览器中打开“journal.html”，看看你的成果：

<blockquote> 创建了单独的一个文本块（与<p>类似），另外还把文字稍稍缩进，使它更像一个引用，这正是我们想要的……

不过我们的引用与原先预想的还有差距，因为所有行都在一起。我们希望它们分行显示。嗯，稍后再来解决这个问题……

there are no Dumb Questions

问：看我这样做对不对：我想在段落里加一些引用就使用<q>，如果希望引用在网页上自成一段就用<blockquote>，这样对吗？

答：说的没错。一般来讲，如果想引用一段或者多段文字，就要使用<blockquote>，不过如果只想把一个引用放在现有的文字里，作为其中的一部分，就可以使用<q>。

问：一个块引用中有多段？怎么做到呢？

答：很容易。只需要把段落元素放在<blockquote>中，一个段落元素对应一个段落。你可以自己试试看。

问：怎么知道我的引用或块引用在其他浏览器中会如何显示呢？听上去不同的浏览器好像会用不同的方式处理。

答：是的。互联网世界里就是这样。如果没有在不同的浏览器上试试看，你就无法知道你的引用会如何显示。有些浏览器使用双引号，有些使用斜体，还有一些不加任何效果。要想确定它们如何显示，唯一的办法就是指定样式，稍后会讨论怎么做到这一点。

问：我知道了，<blockquote>元素的文本会自成一段，还会缩进，那么为什么<blockquote>不像<q>元素那样嵌在段落中呢？

答：因为<blockquote>实际上就像一个新段落。可以这样来考虑，就好像你在一个字处理器中输入文字。输入一段之后，你按两次Enter键，开始一个新段落。要输入一个块引用，也要做同样的工作，并缩进引用。先记住这个内容，这是一个重点，稍后还会仔细讨论。

另外，要记住只是有些浏览器显示<blockquote>时会缩进。一些浏览器对于<blockquote>可能不使用缩进。所以，不要指望<blockquote>在所有浏览器中都有相同的显示。

问：我能结合使用引用元素吗？例如，能不能在<blockquote>元素中使用<q>元素？

答：当然可以。你可以把一个<q>元素放在<p>元素中，同样的，也可以将<q>嵌在<blockquote>中。如果你引用某个人的话，而他又引用了另外某个人的言论，可能就需要这样嵌套使用。不过把<blockquote>放在一个<q>中没有任何意义，对不对？

问：你说我们可以用CSS对这些元素指定样式，这么说来，如果我希望让<q>元素中的文本用灰色斜体显示，可以用CSS来做到。不过，难道不能用元素把引用置为斜体吗？

答：嗯，也可以，不过这么做不太好，因为你这样使用元素只是为了实现显示效果，而不是因为你写了要强调的文字。如果你引用某人的话，其中确实强调了某个词，或者如果你想强调引用中的某个重点，那么完全可以在引用中使用元素。不过，不要只是为了显示斜体而使用元素。利用CSS，有很多更好、更容易的方法来实现你希望的元素显示效果。

谜底： 出生后就被分开的元素

这对孪生引用兄弟为什么会这么快被人发现是冒牌的？

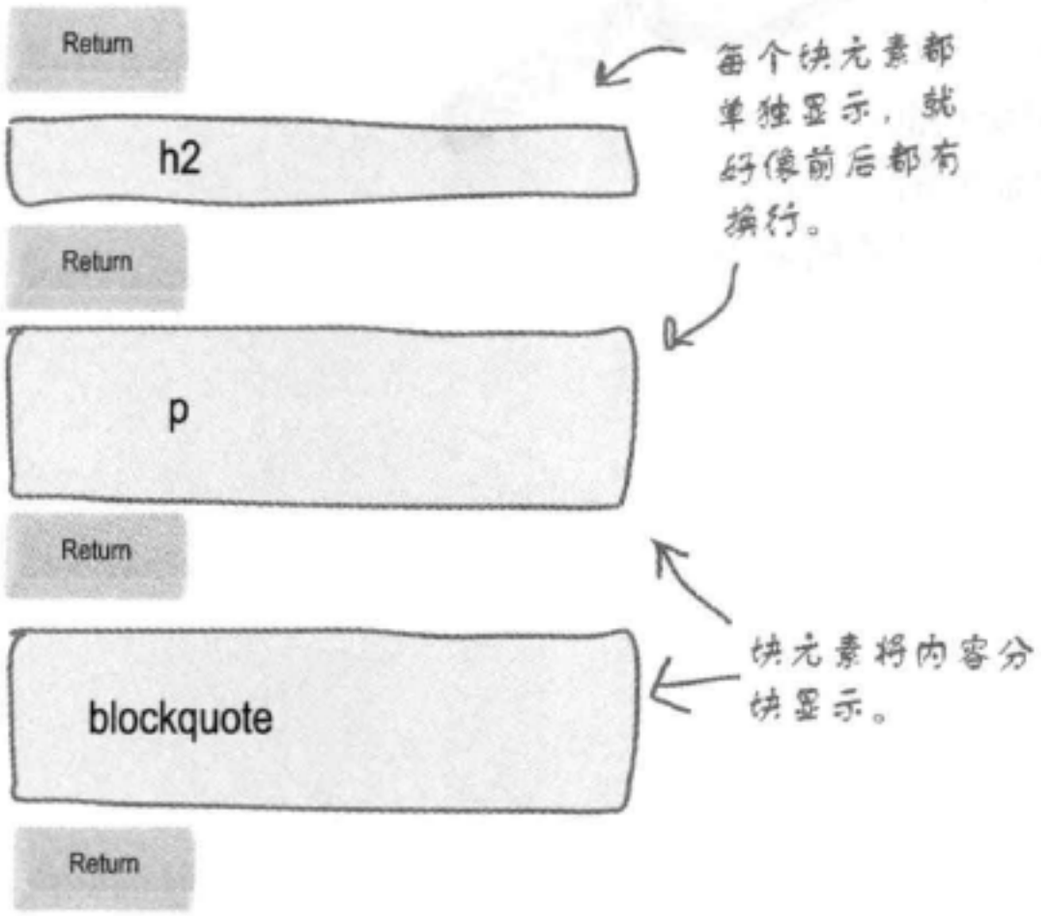
你可能已经猜到了，<q>和<blockquote>一开始工作想要标记文本时就被发现了。<q>通常只是些不起眼的小引用，现在却自成一块，格外突出，而<blockquote>的引用突然湮没在普普通通的文本段落中。接下来采访这个恶作剧的受害者时，一位编辑抱怨道，“就是因为这些疯子，我的整个引用页面变得乱七八糟。”两兄弟分别受到批评，回到各自原来的岗位。<blockquote> 和<q>向他们的妻子坦白后，她们随即驾车离开了小镇。故事就这样结束了（尽管结局不完满）。

<q>和<blockquote>谜案的真相

OK，揭开谜底的时候到了：<blockquote>和<q>实际上是两类不同的元素。<blockquote>元素是一个块（block）元素，而<q>元素是一个内联（inline）元素。有什么区别吗？块元素显示时就好像前后各有一个换行，而内联元素在页面文本流中总在“行内”出现。

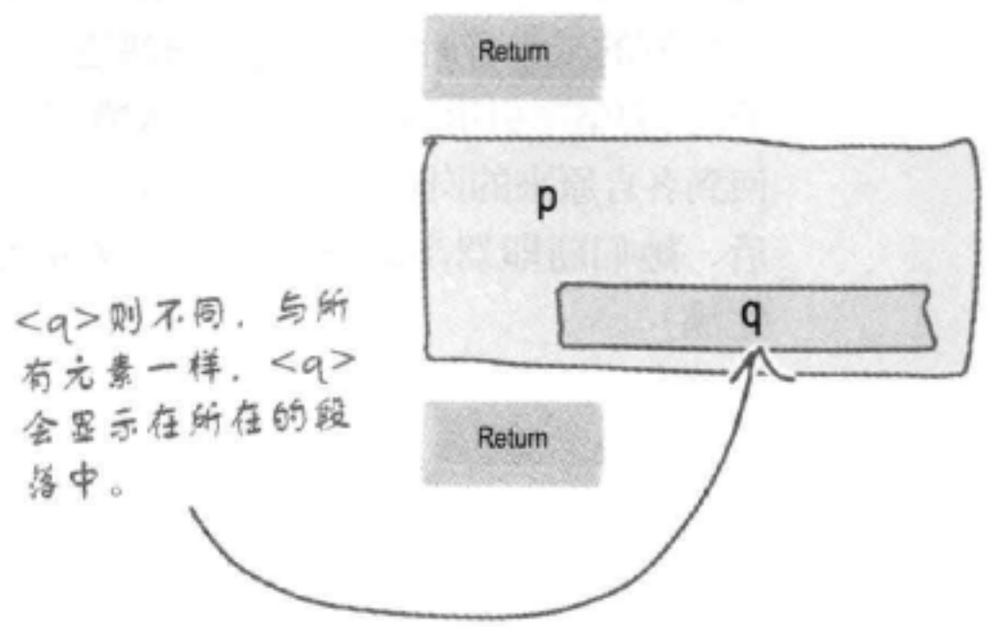
块元素：特立独行

<h1>、<h2>、…、<h6>、<p>和<blockquote>都是块元素。



内联元素：随波逐流

<q>、<a>和是内联元素。



要记住：块元素特立独行；内联元素随波逐流。

there are no Dumb Questions

问：我想我知道换行是什么，就像在打字机或计算机键盘上按Enter键，是吗？

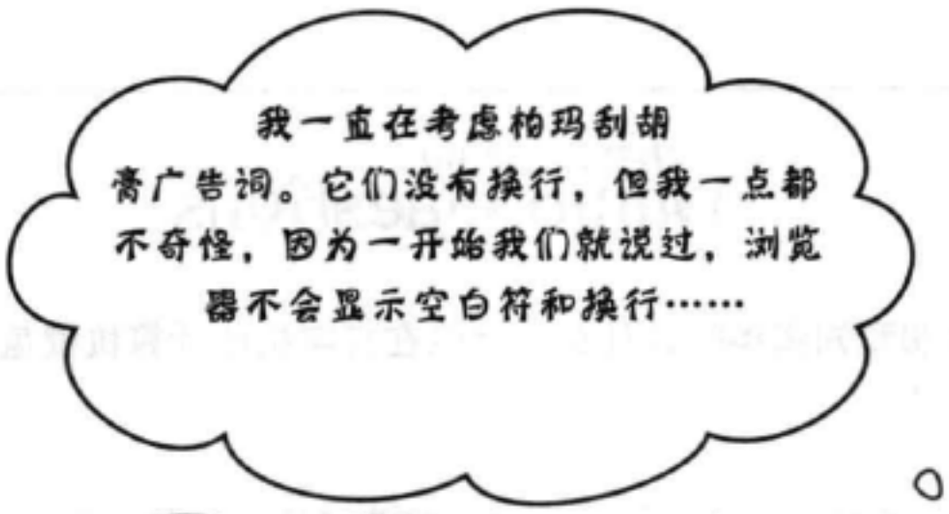
答：差不多吧。“换行”字面意思就是“换一行”，你说的没错，就像按Enter键一样。你已经知道，浏览器显示一个页面时，HTML文件中的换行符并不会出现，对不对？不过，现在你已经看到了，只要使用块元素，浏览器就会使用换行符来分隔每一“块”。

嗯，听起来不错，不过说这些换行、块和内联元素有什么用呢？该回来谈谈Web页面了吧？

你要了解HTML是如何工作的，不要低估掌握这些知识的重要性。很快你就快会看到，在页面上如何结合元素对于元素将显示为块元素还是内联元素有很大影响。接下来就会讨论这个内容。

另外，还可以这样考虑块元素和内联元素：块元素通常用作Web页面中的主要构建模块，而内联元素往往用来标记小段内容。设计一个页面时，一般先从较大的块开始（块元素），然后在完善页面时再加入内联元素。

在用CSS控制HTML的表现时，这些知识就能派上用场了。如果你清楚内联元素和块元素的区别，就可以轻松设计好布局，在别人为布局设计忙得焦头烂额时，你却能悠闲地喝着马提尼。



如果有这样一个元素，它的唯一任务就是在你需要时提供一个换行，你觉得怎么样？

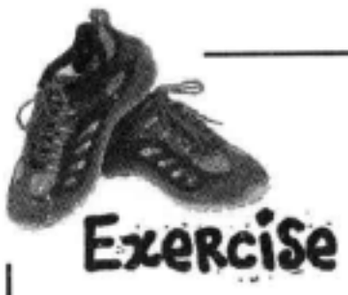
这样不是很好吗？你能让浏览器注意到并插入一些回车符。

事实上确实有这样一个元素，这就是
元素，它的作用正是如此。可以这样来使用：

```
<h2>July 14, 2012</h2>
<p>
   I saw some Burma Shave style signs on the
   side of the road today:
</p>
<blockquote>
   Passing cars, <br>
   When you can't see, <br>
   May get you, <br>
   A glimpse, <br>
   Of eternity. <br>
</blockquote>
<p>
   I definitely won't be passing any cars.
</p>
```

这是从Tony页面中节取的7月14日日志的片段。

在希望换行的地方增加一个
元素，这里就会插入一个“换行符”。



为Tony的日志增加
元素。修改之后，保存文件，再对页面做一个测试。

我们修改的结果如下。现在看起来像是真正的柏瑞利的广告了。

现在每一行后面都有一个换行。

My Trip Around the USA on a Segway

file:///chapter3/journal/journal.html

Google

Segway'n USA

Documenting my trip around the US on my very own Segway!

August 20, 2012

...lla, W... ...gic City, ... Boundar..., Last Chance, CO, Why, AZ and Truth or Consequences, NM.

July 14, 2012

I saw some Burma Shave style signs on the side of the road today:

Passing cars,
When you can't see,
May get you,
A glimpse,
Of eternity.

I definitely won't be passing any cars.

June 2, 2012

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

在第1章中我们说过，元素是开始标记+内容+结束标记。
算是一个元素吗？它没有任何内容，甚至还没有结束标记。

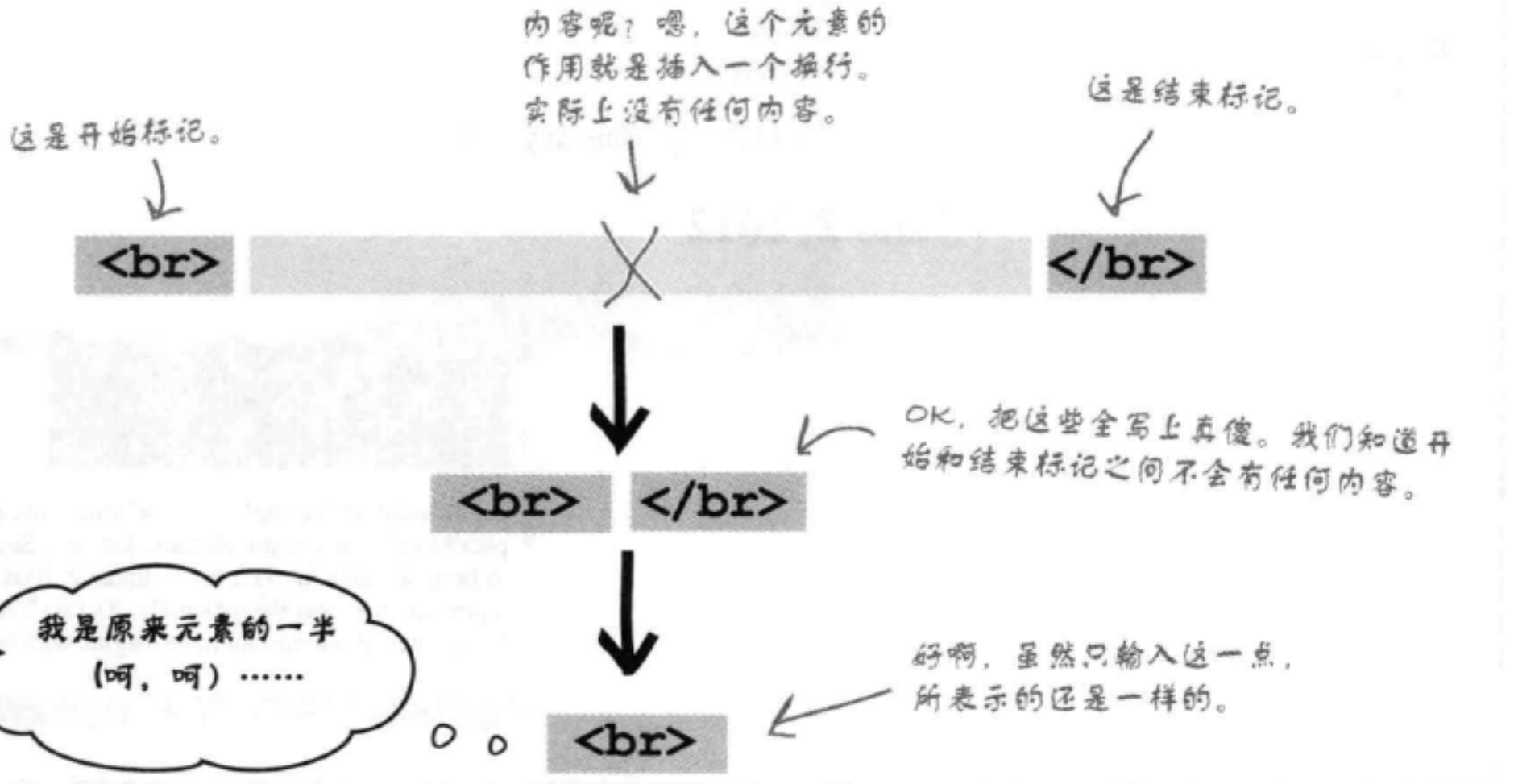
的确，它没有内容。

元素是一个没有任何内容的元素。为什么？因为它只是一个换行，没有其他内容。所以，如果一个元素设计为没有任何实际内容，我们会使用简写来表示这个元素，最后就会像
一样。毕竟，如果没有这个简写，每次需要一个换行时都需要写为
</br>，这有意义吗？

并不是唯一没有实际内容的元素，还有很多类似这样的元素，我们把这些元素叫做void元素。实际上，之前我们已经见过另一个void元素，那就是<img>元素。后面几章还会详细讨论<img>元素。

要记住，之所以用简写不是因为懒惰，更多的是为了提高效率。这样表示void元素会更高效（输入的字符更少，相应的页面中字符数也更少）。实际上，读过HTML之后，你会发现这样看起来也更轻松。

原先它们称为“空元素”，显然不太贴切，所以后来又改名为void元素。就个人而言，我们还是喜欢空元素这个名字。



**问：那么，
的唯一作用就是插入一个换行，是吗？**

答：没错，通常只有当你开始一个新的块元素（如 <p>、<h1>等）时，浏览器才会插入换行。如果你希望在文本中插入一个换行，就要使用
元素。

**问：为什么把
叫做“void”元素？**

答：因为它没有任何内容，按理说元素 = 开始标记+ 内容 + 结束标记。因为没有内容也没有结束标记，所以它是空元素（void）。就像一个“空位”，这里什么也没有，是空的。

**问：我还是不明白，请解释一下为什么
元素是“void”？**

答：考虑另外一个元素，比如<h1>（或者<p>或<a>）。这个元素的作用是标记内容，如：

```
<h1>Don't wait, order now</h1>
```

而
元素不同，它的作用就是在HTML中插入一个换行。没有需要标记的内容。我们不需要额外的尖括号和标记，所以可以把它简写为一种更方便的形式。如果你觉得 “void”这个名字很怪，没错：这个词来自计算机科学，原意是“无值”。

问：还有没有其他void元素？我觉得<img>肯定也是一个void元素，对吗？

答：没错，还有另外一些void元素。你已经见过我们使用了<img>元素，很快就会详细介绍这个元素。

问：我能把某个元素变成void元素吗？例如，如果有一个链接，那么我不想指定任何内容，可不可以直接写为 <a href="mypage.html"> ？

答：不行。世界上有两类元素：正常元素和void元素，正常元素比如<p>、<h1>和<a>，void元素如
和<img>。不能在这两个阵营之间来回切换。例如，如果你只输入<a href="mypage.html">，这只是一个开始标记，没有内容或结束标记（这是不对的）。如果写为，这就是一个空元素，这样是可以的，不过放在页面上没有什么用处！

**问：我见过有些页面上没有使用
而用了
。这是什么意思？**

答：意思完全是一样的。处理XHTML时，
的语法更为严格。只要看到
，都可以把它想象成是
，不过除非要规划或编写与XHTML兼容的页面（关于XHTML的更多信息参见附录），否则HTML中都应该使用
。

**设计为没有任何内容的元素称为void元素。需要使用void元素时，如
或<img>，只需使用一个开始标记。这是一种方便的简写形式，可以减少HTML中的标记数量。**

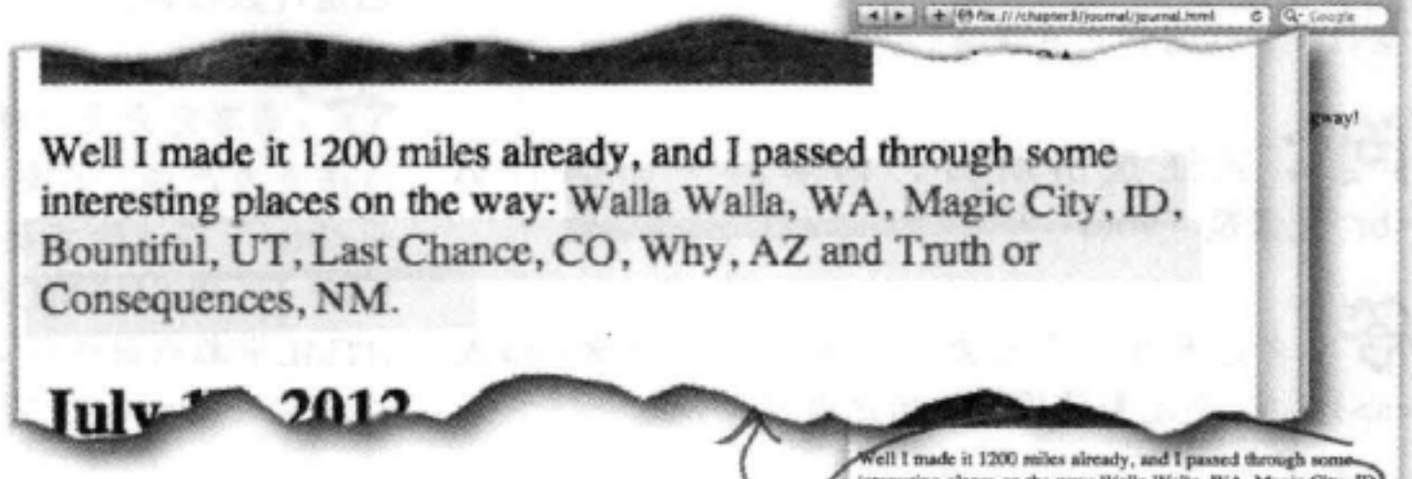
再回到Tony的网站……

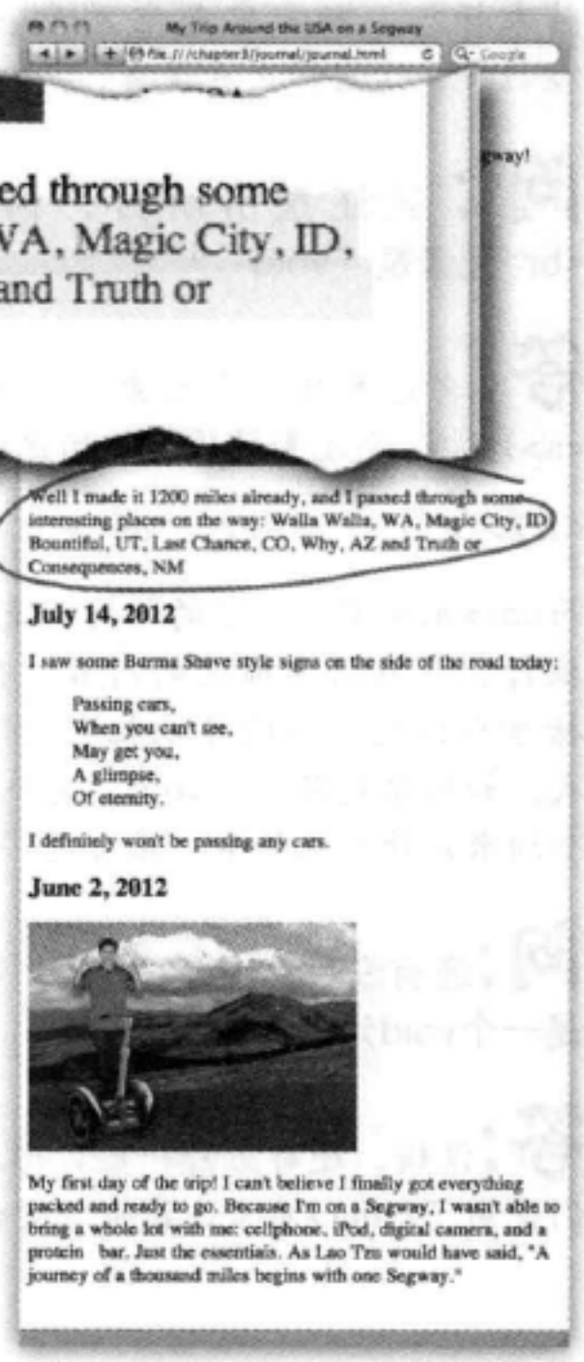
这一章已经做了不少工作了：已经设计并创建了Tony的网站，你还认识了一些新元素，另外了解了关于元素的一些内部信息，大多数创建Web页面的人对此可能都不清楚（比如块元素和内联元素，这些知识在后面的章节中很有用处）。

不过你的工作还没有结束。我们要让Tony的网站更上一层楼，看看还有哪些地方可以增加一些标记。

比如说？列表？尝试一下：

这里有一个列表。Tony在8月的旅行日志里写了他曾到过的城市的一个列表。





如果能标记这个文本，让浏览器知道这个文本是一个列表就好了！这样浏览器就能用一种更有用的方式显示列表项。比如：

Well I've made it 1200 miles already, and I passed through some interesting places on the way:

1. Walla Walla, WA
2. Magic City, ID
3. Bountiful, UT
4. Last Chance, CO
5. Why, AZ
6. Truth or Consequences, NM

注意这不仅是一个列表，还是一个有序列表。Tony是按特定的顺序游历这些城市的。

当然，也可以使用<p>元素创建列表……

使用<p>元素创建列表并不难。最后可能像这样：

```
<p>
1. Red Segway
</p>
<p>
2. Blue Segway
</p>
```

Segway最优选的两种颜色。

不过，出于多方面原因，我们不这么做

你现在应该有点常识了。我们总希望选择与内容结构含义最接近的HTML元素。如果这是一个列表，就使用一个列表元素。这样做可以让浏览器和你（稍后会看到）有更大的能力和灵活性，可以用一种有用的方式显示内容。

BRAIN POWER

为什么不用<p>建列表？（多选题）

- ☐ A. HTML专门提供了一个列表元素。如果使用这个元素，浏览器会知道这个文本是一个列表，就能用最佳方式来显示。
- ☐ B. 段落元素表示的是文本段落，而不是列表。
- ☐ C. 可能看上去不是一个列表，而是一堆有编号的段落。
- ☐ D. 如果想改变列表顺序，或者想插入一个新列表项，就必须对所有列表项重新编号。这太麻烦了。

答案：A，B，C，&D

两步轻松构建HTML列表

创建一个HTML列表需要两个元素，结合使用这两个元素就构成了列表。第一个元素用来标记每个列表项。第二个元素确定你创建的是哪种类型的列表：有序列表还是无序列表。

下面分两步在HTML中创建Tony的城市列表。

第一步：

将每个列表项放在一个 元素中。

要创建一个列表，需要把每个列表项放在单独的
元素中，这说明需要把内容用一个开始
标记和一个结束
标记包围起来。与所有其他HTML元素一样，标记之间的内容由你来定，可以很短，也可以很长，分为多行。

这里我们给出了Tony旅行日志的一个HTML片段。

在"journal.html"文件中找到这个HTML片段，跟着我们做相应的修改。

```
<h2>August 20, 2012</h2>
    <img src="images/segway2.jpg">
<p>
Well I've made it 1200 miles already, and I passed
through some interesting places on the way:
</p>
<li>Walla Walla, WA</li>
<li>Magic City, ID</li>
<li>Bountiful, UT</li>
<li>Last Chance, CO</li>
<li>Why, AZ</li>
<li>Truth or Consequences, NM</li>

<h2>July 14, 2012</h2>
<p>
I saw some Burma Shave style signs on the side of
the road today:
</p>
```

首先，将列表项移出段落。列表会独立显示。

……然后将每个列表项用一对
，
标记包围。

每个<li>元素会成为列表中的一个列表项。

第二步：

用<ol>或<ul>元素包围列表项。

如果使用一个<ol>元素包围列表项，则这些列表项将作为一个有序列表显示；如果使用<ul>，则列表将显示为一个无序列表。下面将列表项包围在一个<ol>元素中。

同样地，这里再给出Tony旅行日志的一个HTML片段。

```
<h2>August 20, 2012</h2>
    <img src="images/segway2.jpg">
<p>
Well I've made it 1200 miles already, and I passed
through some interesting places on the way:
</p>
<ol>
  <li>Walla Walla, WA</li>
  <li>Magic City, ID</li>
  <li>Bountiful, UT</li>
  <li>Last Chance, CO</li>
  <li>Why, AZ</li>
  <li>Truth or Consequences, NM</li>
</ol>
<h2>July 14, 2012</h2>
<p>
I saw some Burma Shave style signs on the side of
the road today:
</p>
```

我们希望这是一个有序列表，因为Tony是按一个特定顺序游历这些城市的。所以这里使用一个<ol>开始标记。

所有列表项放在<ol>元素中间，成为它的内容。

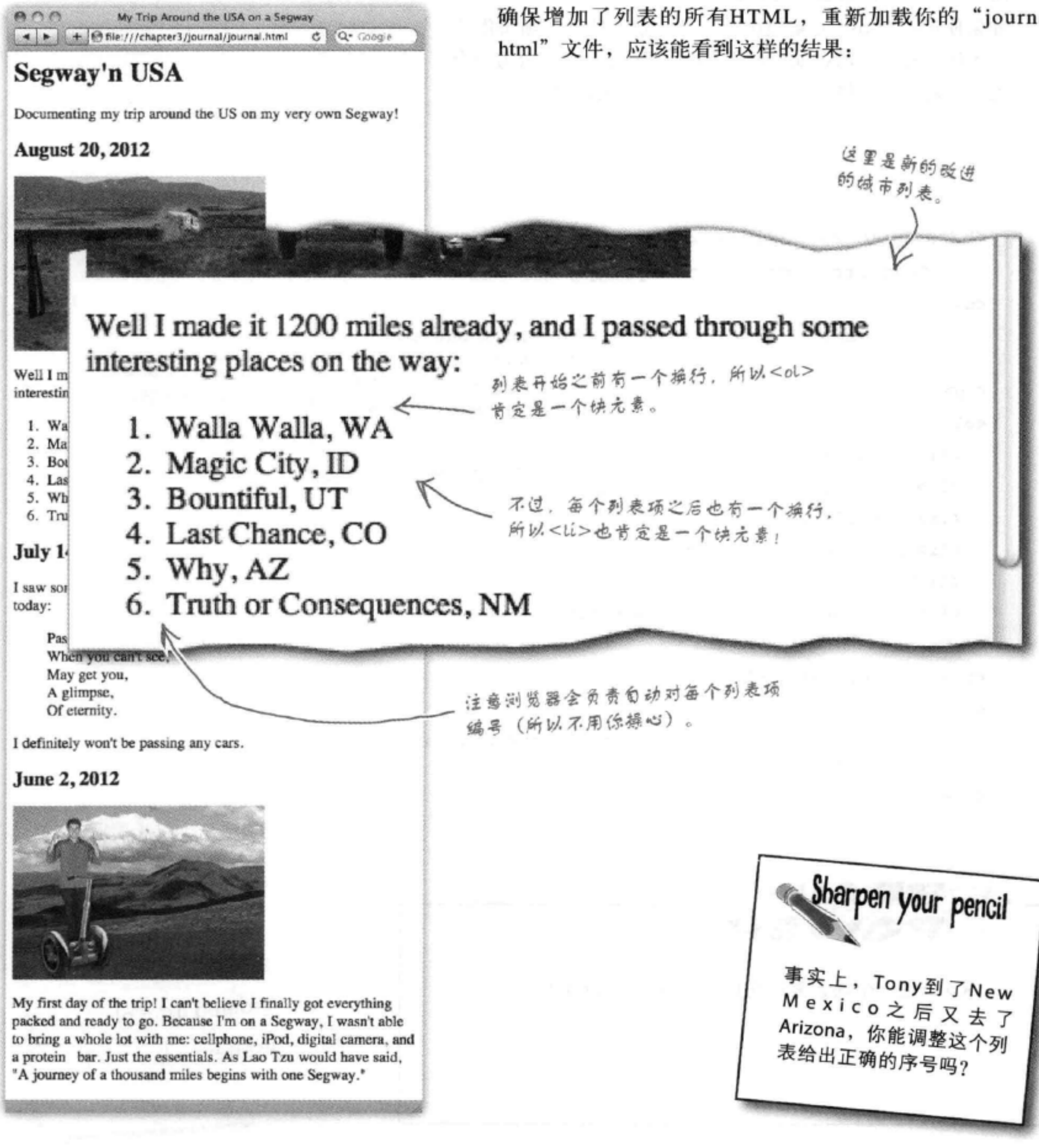
这里结束这个<ol>元素。

BRAIN POWER

是一个块元素还是内联元素？<li>呢？

城市列表测试

确保增加了列表的所有HTML，重新加载你的“journal.html”文件，应该能看到这样的结果：



Sharpen your pencil

事实上，Tony到了New Mexico之后又去了Arizona，你能调整这个列表给出正确的序号吗？

Exercise

这是Tony旅行日志中的另一个列表：cell phone, iPod, digital camera和一个protein bar。在他6月2日的日志中可以找到。这是一个无序列表。

下面给出了这个日志的HTML。请增加必要的HTML，把这些内容变成一个HTML无序列表（要记住，无序列表要用<ul>）。我们已经为你调整了一些文本的格式。

完成后，对照本章最后给出的答案检查你的HTML。再在“journal.html”文件中完成修改，并做个测试。

```
<h2>June 2, 2012</h2>
<img src="segway1.jpg">
<p>
      My first day of the trip! I can't believe I finally got
      everything packed and ready to go. Because I'm on a Segway,
      I wasn't able to bring a whole lot with me:

          cell phone
          iPod
          digital camera
          and a protein bar

      Just the essentials. As
      Lao Tzu would have said, <q>A journey of a
      thousand miles begins with one Segway.</q>
</p>
```

问：<ol>和<li>总要一起使用吗？

答：对，<ol>和<li>（或者<ul>和<li>）总是要一起使用。这些元素离开彼此就没有任何意义了。要记住，列表实际上就是一组列表项：<li>元素用来标识每个元素，<ol>元素把它们归为一组。

问：能把文本或其他元素放在<ol>或<ul>元素里吗？

答：不行，<ol>和<ul>元素设计为只能包含<li>元素。

问：无序列表呢？能不能用不同的项目符号？

答：可以。不过最好别这么干。后面介绍CSS和表现时还会再来讨论这个问题。

问：如果我想在列表中嵌一个列表呢？能行吗？

答：可以，当然可以。可以把一个<ol>或<ul>作为某个<li>的内容，这样就在列表中有了另一个列表（我们把这叫做嵌套列表）。

```
<ol>
   <li>Charge Segway</li>
   <li>Pack for trip
      <ul>
         <li>cell phone</li>
         <li>iPod</li>
         <li>digital camera</li>
         <li>a protein bar</li>
      </ul>
   </li>
   <li>Call mom</li>
</ol>
```

嵌套列表

这是一个<li>，它包围了嵌套列表。

问：我觉得我基本上已经了解浏览器如何显示块元素和内联元素，不过还是不清楚哪些元素放在其他元素里，就像你说的，哪些元素可以“嵌套”在另外哪些元素里。

答：这是掌握HTML最大的难点之一。你需要通过好几章来学习这个内容，我们会向你介绍很多方法，确保你能把这个关系真正搞懂。不过我们先回到正题，稍后再讨论嵌套。实际上，既然你感兴趣，接下来我们就会介绍这个内容。

问：这么说，HTML有有序列表和无序列表。还有其他类型的列表吗？

答：实际上，确实还有另外一类列表：定义列表。定义列表如下所示：

列表中每一项都有一个定义术语<dt>和一个定义描述<dd>。

```
<dl>
   <dt>Burma Shave Signs</dt>
      <dd>Road signs common in the U.S. in the 1920s and 1930s advertising shaving products.</dd>
   <dt>Route 66</dt>
   <dd>Most famous road in the U.S. highway system.
   </dd>
</dl>
```

可以输入这个列表，试一试。

问：柏玛刮胡膏是什么？

答：柏玛刮胡膏（Burma Shave）是20世纪初一家生产刮胡膏的公司。他们从1925年开始在路边树立广告牌来推广他们的产品，事实证明这些广告相当深入人心（当然也可能会让司机分心）。

这些广告4、5或6个一组，每行一句广告语。曾经一段时间，美国道路两旁有7000个这样的广告牌。现在大多数都已经没有了，不过偶尔还能看到剩余的少许几个。

把一个元素放在另一个元素中称为“嵌套”。

把一个元素放在另一个元素中时，我们称之为“嵌套”。我们可能这么说，“<p>元素嵌套在<body>元素中。”目前，你已经见过很多元素可以嵌套在其他元素中。之前我们将<body>元素嵌在<html>元素中，把<p>元素放在<body>元素中，把<q>元素嵌在一个<p>元素中，这样的嵌套还有很多。另外，还把<head>元素放在<html>元素中，<title>元素放在<head>中。HTML页面就是以这种方式构造的。

你对HTML了解得越多，很好地掌握嵌套就越显重要。不过不用担心，很快你就能很自然地以这种方式考虑元素了。

要理解嵌套关系，画一个图

画出网页中元素的嵌套关系就像画一棵家族树。最上面是曾祖父，下面是所有孩子和孙子。这里给出一个例子……

简单Web页面。

```
<html>
  <head>
    <title>Musings</title>
  </head>
  <body>
    <p>
       To quote Buckaroo,
       <q>The only reason
          for time is so
          that everything
          doesn't happen
          at once.</q>
    </p>
  </body>
</html>
```

现在把它画成图，每个元素画成一个方框，用线连接元素和嵌套在其中的另一个元素。

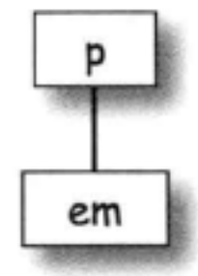
使用嵌套确保标记匹配

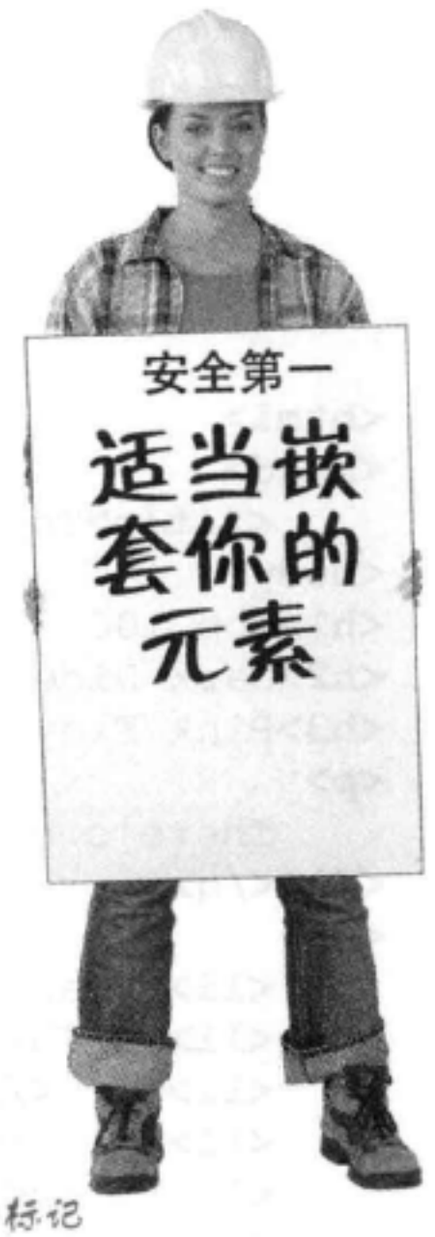
如果你了解元素如何嵌套，则第一个好处是你能避免标记不匹配。（后面还会有更多好处，请耐心等待）。

“标记不匹配”是什么意思？怎么会发生这种情况？下面来看一个例子：

```
<p>I'm so going to tweet <em>this</em></p>
```

这是HTML，嵌套在<p>中。



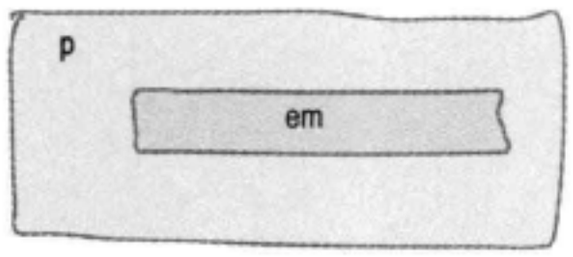


到目前为止，一切都还好，不过很容易出现手误，写出这样的HTML：

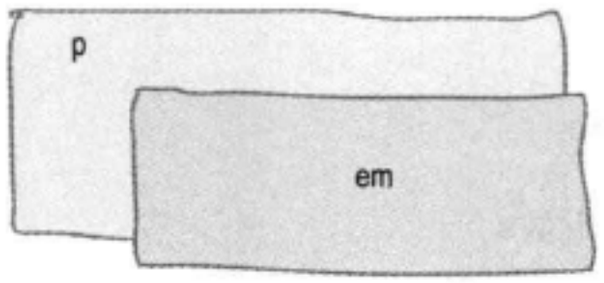
```
<p>I'm so going to tweet <em>this</p></em>
```

错误：<p>在标记之前结束！元素应当在<p>元素里面。

根据你对嵌套的了解，现在你应该知道元素需要完全嵌套或包含在<p>元素中。



正确：这里元素嵌在<p>里面。



错误：这里元素有一部分漏到<p>元素之外，这说明它没有正确地嵌套在<p>元素中。

那又怎样？

如果你喜欢玩俄罗斯转盘，那么把嵌套弄成这样一团糟也无所谓。不过如果你写HTML时没有正确地嵌套元素，也许你的页面在一些浏览器上能正常工作，但在另外一些浏览器上可能就不正常了。如果能很好地掌握嵌套，就能避免标记不匹配，确保你的HTML在所有浏览器上都能正常工作。在后面几章详细讨论“工业强度的HTML”时这一点会更为重要。

扮演浏览器

下面的HTML文件中有一些不匹配的标记。你的任务是扮演浏览器，找出所有错误。完成这个练习后，对照本章最后的答案看看你有没有找出所有错误。

```
<html>
<head>
    <title>Top 100</title>
<body>
<h1>Top 100
<h2>Dark Side of the Moon</h2>
<h3>Pink Floyd</h3>
<p>
    There's no dark side of the moon; matter   of fact <q>it's all dark.
</p></q>
<ul>
    <li>Speak to Me / Breathe</li>
    <li>On The Run</li>
    <li>Time</li>
    <li>The Great Gig in The Sky</li>
    <li>Money</li>
    <li>Us And Them</em>
    <li>Any Colour You Like</li>
    <li>Brain Damage</li>
    <li>Eclipse</li>
</ul>
</p>
<h2>XandY</h3>
<h3>Coldplay</h2>
<ol>
    <li>Square One
    <li>What If?
    <li>White Shadows
    <li>Fix You
    <li>Talk
    <li>XandY
    <li>Speed of Sound
    <li>A Message
    <li>Low
    <li>Hardest Part
    <li>Swallowed In The Sea
    <li>Twisted Logic
</ul>
</body>
</head>
```

一堆HTML元素身着盛装在玩一个派对游戏“我是谁？”它们给了一个线索，你要根据它们说的好好猜猜它们到底是谁。当然假设它们说的都是真话。确认它们的身份，填在右边的空格上。另外，对于每个参加者，还要写出这个元素是内联元素还是块元素。

今晚参加派对的有：

目前为止你见过的所有迷人的HTML元素都可能到场！

	名字	内联元素还是块元素？
我是1级标题。	.....	.....
我时刻准备链接到另一个页面。	.....	.....
用我强调文本。	.....	.....
我是个列表，不过我不关心顺序。	.....	.....
我是真正的换行符。	.....	.....
我是放在列表中的列表项。	.....	.....
我要保证我的列表项有序。	.....	.....
我的任务就是提供图像。	.....	.....
用我可以在段落中加引用。	.....	.....
用我来引用独立的文字。	.....	.....

我刚创建了一个Web页面来解释从这本书中学到的知识，我希望在页面中提到<html>元素。这会不会把嵌套弄乱？需要用双引号或者其他什么字符把它括起来吗？

你说的没错，这确实会有问题。

因为浏览器要用<和>来开始和结束标记，如果在HTML内容中使用这两个字符，就会有问题。不过HTML提供了一种简便的方法，可以使用一种称为字符实体（character entity）的简单缩写来指定这样一些特殊字符。它是这样工作的：对于被认为“特殊”的字符，或者你可能希望在Web页面中使用某个字符但在你的编辑器里无法输入（比如版权符号），就可以查找相应的缩写在HTML中直接输入。例如，>字符的缩写是 >，<字符的缩写是<。

所以，假如你希望在页面中输入“The <html> element rocks.”。通过使用字符实体，可以这样输入：

```
The &lt;html&gt; element rocks.
```

还有一个重要的特殊字符需要知道，这就是&字符（与字符）。如果希望HTML内容中出现一个&，则可以使用字符实体&，而不要直接使用字符&。

那么版权符号呢（版权符号是&copyright;）？还有所有其他符号和外文字符呢？可以在下面这个URL找到一些常用的字符实体：

http://www.w3schools.com/tags/ref _ entities.asp

或者，如果需要更详尽的字符实体清单，则可以访问这个URL：

http://www.unicode.org/charts/

there are no Dumb Questions

问：哇，我以前都不知道浏览器居然还可以显示这么多不同的字符。www.unicode.org网站上有太多的不同字符和语言。

答：要当心。只有当你的计算机或设备安装了正确的字体时，你的浏览器才有可能显示所有这些字符。所以，你可能以为www.w3schools.com页面上的所有基本实体在任何浏览器上都可用，但实际上并不能保证所有这些实体都能显示。不过，假设你了解你的用户，应该很清楚他们的机器上通常使用哪些外文字符。

问：你说&是特殊字符，我得用字符实体&来取代它，不过要输入实体我又必须使用&。这么说，如果我要输入>实体，就必须输入>，是吗？

答：不，不是这样的！&之所以特殊正是因为它是所有实体的第一个字符。所以在实体名中使用&是完全可以的，但不能单独使用&。

只要记住，输入实体时都会用到&，而如果内容中确实需要一个&，就要使用&。

问：我在www.w3cschools.com上查看了字符实体，注意到每个实体还有一个编号。这个编号做什么用？

答：你可以在HTML中使用这个编号，比如&#100，也可以使用实体名，它们的作用都是一样的。不过，并不是所有实体都有名字，所以如果没有实体名，那么使用这个编号就是唯一选择了。

破解位置挑战

为了实现统治世界的欲望，Evel博士建立了一个私人网页，供他的党羽使用。你接收到截获的一个HTML片段，其中可能包含他的行踪线索。根据你掌握的HTML专业知识，现在要求你破解这个代码，找出他的位置。下面是他的主页的部分文字：

```
There's going to be an evil henchman meetup next
month at my underground lair in
&#208;&epsilon;&tau;&#114;&ouml;&igrave;&tau;. Come
join us.
```

提示：访问http://www.w3schools.com/tags/ref_entities.asp 和/或输入这段HTML，看看浏览器会显示什么。

元素大杂烩

使用这个元素来标记你想用不同方式展示的文本，比如你想强调一个要点。

使用这个元素来标记你希望特别强调的文本。

<pre>
希望浏览器按你输入的方式原样显示文本时，使用这个元素来指定文本的格式。

只要需要建立链接，就需要<a>元素。

<a>

<time>
这个元素告诉浏览器这个内容是一个日期或时间，或者同时包含日期和时间。

这是一个void元素，用来提供换行。

使用这个元素提供短引用……你懂的，比如 "to be or not to be," 或 "No matter where you go, there you are." 。

<q>

需要显示一个列表？比如说，菜谱中的配料列表，或者一个任务列表？可以使用<ul>元素。

<p>
给我一个段落，谢谢。

如果需要一个有序列表，就要使用<ol>元素。

<img>
这个元素用来在页面中包含一个图像，比如照片。

code元素用来显示计算机程序代码。

<code>

对应列表中的列表项，如巧克力、热巧克力、巧克力酱……

这个元素用于长引用，就是你希望作为一大段文字来强调的内容，比如从书中节选的一部分。

<blockquote>

这里有一大堆你已经知道的元素，还有一些你没有见过的元素。

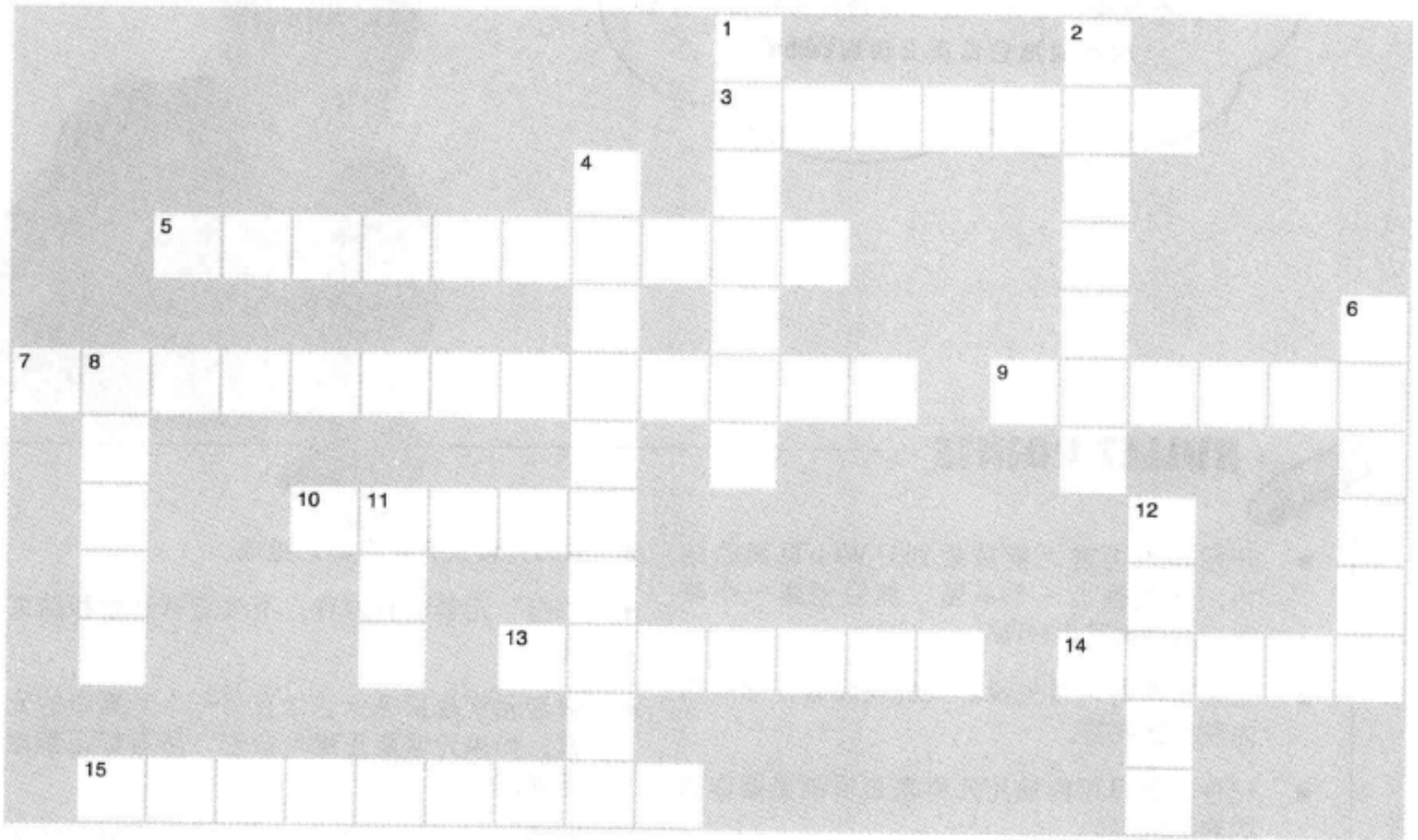
要记住，HTML的一半乐趣在于动手实践！所以你要自己创建一些HTML文件来试试。

BULLET POINTS

- 开始输入内容之前要规划好Web页面的结构。首先画出一个草图，然后创建一个略图，最后再写出HTML。
- 规划页面时，首先设计大的块元素，然后用内联元素完善。
- 记住，要尽可能使用元素来告诉浏览器你的内容的含义。
- 一定要使用与内容含义最接近的元素。例如，如果需要一个列表，就不要使用段落元素。
- <p>、<blockquote>、<ol>、<ul>和<li>都是块元素。它们单独显示，在内容前后分别有一个换行（默认地）。
- <q>和是内联元素。这些元素中的内容与其包含元素的其余内容放在一起。
- 需要插入你自己的换行时，可以使用
元素。
-
是一个“void”元素。
- void元素没有内容。
- void元素只有一个标记组成。
- “空”元素没有内容。不过它有开始和结束标记。
- 嵌套元素是指完全包含在另一个元素中的元素。如果元素能正确地嵌套，所有标记都能正确匹配。
- 要结合两个元素建立一个HTML列表：使用<ol>和<li>建立有序列表，使用<ul>和<li>可以建立一个无序列表。
- 浏览器显示一个有序列表时，它会为列表创建序号，所以无需你费心。
- 可以在列表中建立嵌套列表，将<ol>或<ul>元素放在<li>元素中。
- 要对HTML内容中的特殊字符使用字符实体。

HTML填字游戏

该让你的右脑歇一歇了，让左脑开动起来：所有单词都与HTML有关，而且都在本章出现过。



横向

1. Tony的交通工具。
8. 著名的路边广告牌。
10. <q>是这种类型的元素。
11. 另一个void元素。
13. 没有内容的元素。
14. 页面的主要构建模块。
15. 使用<ol>建立这种列表。

纵向

2. 驾车一同离开小镇的人。
3. Segway的最大时速。
4. Tony不会做这些。
5. 把一个元素放在另一个元素中叫做什么？
6. 需要两个元素。
7. 提供引用的块元素。
9. 使用<ul>建立这种列表。
12. void元素没有这些。

Exercise Solution

下面是Tony改写的老子的话，这里使用<q>元素实现。你对你的答案做了测试吗？

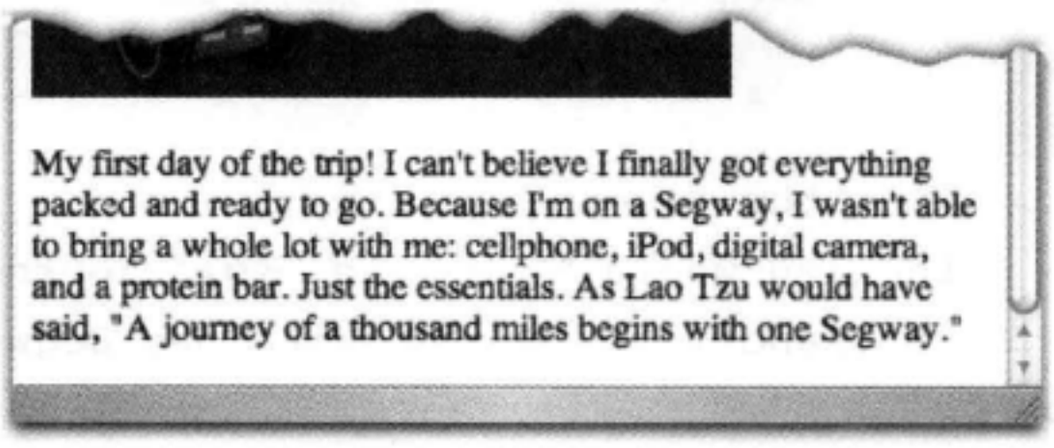
这是改动过的部分……

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

我们在引用开始之前增加了<q>开始标记，并在最后增加了</q>结束标记。

注意我们还去掉了双引号。

这里是我们的测试……



OK，尽管看起来没有不同，不过你现在感觉是不是好多了？

Exercise Solution

这是Tony旅行日志中的另一个列表：cell phone、 iPod、 digital camera和一个protein bar。在他6月2日的日志中可以找到。这是一个无序列表。

再在“journal.html”文件中完成这些修改，和你的答案一样吗？

```
<h2>June 2, 2012</h2>
<img src="segway1.jpg">
<p>
      My first day of the trip! I can't believe I finally got
      everything packed and ready to go. Because I'm on a Segway,
      I wasn't able to bring a whole lot with me:
</p>
<ul>
        <li>cell phone</li>
        <li>iPod</li>
        <li>digital camera</li>
        <li>and a protein bar</li>
</ul>
<p>
      Just the essentials. As
      Lao Tzu would have said, <q>A journey of a
      thousand miles begins with one Segway.</q>
</p>
```

首先，结束上一个段落。

开始这个无序列表。

把各个列表项分别放在一个<li>元素中。

结束这个无序列表。

需要开始一个新段落。

扮演浏览器

答案

```
<html>
<head>
    <title>Top 100</title>
<body>
<h1>Top 100
<h2>Dark Side of the Moon</h2>
<h3>Pink Floyd</h3>
<p>
    There's no dark side of the moon; matter of fact <q>it's all dark.
</p></q>
<ul>
    <li>Speak to Me / Breathe</li>
    <li>On The Run</li>
    <li>Time</li>
    <li>The Great Gig in The Sky</li>
    <li>Money</li>
    <li>Us And Them</em>
    <li>Any Colour You Like</li>
    <li>Brain Damage</li>
    <li>Eclipse</li>
</ul>
</p>
<h2>XandY</h3>
<h3>Coldplay</h2>
<ol>
    <li>Square One
    <li>What If?
    <li>White Shadows
    <li>Fix You
    <li>Talk
    <li>XandY
    <li>Speed of Sound
    <li>A Message
    <li>Low
    <li>Hardest Part
    <li>Swallowed In The Sea
    <li>Twisted Logic
</ul>
</body>
</head>
```

缺少</head>结束标记。

缺少</h1>结束标记。

和<q>没有正确嵌套： 标记应该在</q>标记后面。

有一个结束，这里原本应该是一个结束
标记。

这里有一个结束 ，但没有与之匹配的开始 标记。

这些标题的</h2>和</h3>结束标记弄混了。

开始了一个
列表，但它与一个结束
标记匹配。

所有
结束标记都丢了。

这与以上列表的<ol>开始标记不匹配。

这是前面缺少的</head>标记，不过还缺少一个结束</html>标记。

Exercise Solution

一堆HTML元素身着盛装在玩一个派对游戏“我是谁？”它们给了一个线索，你要根据它们说的好好猜猜它们到底是谁。

今晚参加派对的有：

目前为止你见过的很多迷人的HTML元素都可能到场！

我是1级标题。

我时刻准备链接到另一个页面。

用我强调文本。

我是个列表，不过我不关心顺序。

我是真正的换行符。

我是放在列表中的列表项。

我要保证我的列表项有序。

我的任务就是提供图像。

用我可以在段落中加引用。

用我来引用独立的文字。

名字	内联元素还是块元素？
h1	block
a	嗯……
em	inline
ul	block
br	嗯……
li	block
ol	block
img	inline
q	inline
blockquote	block

嗯，看起来像是一个内联元素，但是<a>还可以包围块元素，而不只是文本。所以，根据具体的上下文，<a>既可以是内联元素，也可以是块元素。

糊涂了？
是块元素和内联元素之间的一个模糊地段。它确实会创建一个换行，不过不会像有两个<p>元素那样把文本分成单独的两块。

我们还没有详细讨论这个元素，不过，没错，<img>确实是内联元素。先记住这一点，我们在第5章还会专门介绍。

Exercise Solution

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

破解位置挑战

你可能已经查出每个实体，或者已经输入这个HTML。不管怎样，答案看来都是Detroit！

```
There's going to be an evil henchman meetup
next month at my underground lair in
&#208;&epsilon;&tau;&#114;&ouml;&igrave;
&tau;. Come join us.
```

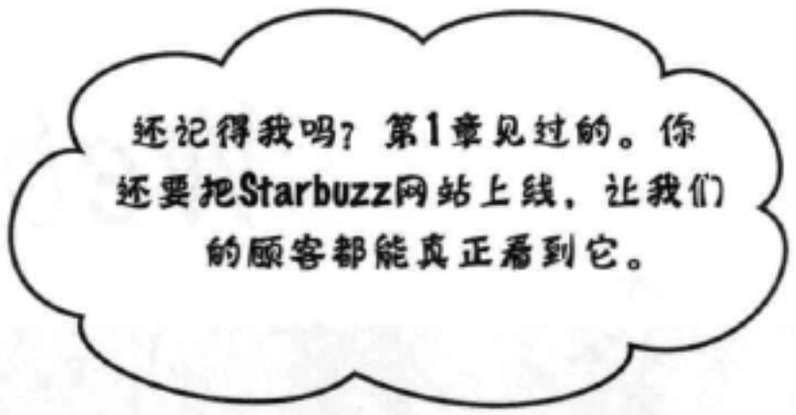
evel.html

There's going to be an evil henchman meetup next month at my underground lair in Ðετröιτ. Come join us.

Web镇之旅

Web页面是互联网给我们的最好的一道菜。到目前为止，你只是在自己的计算机上创建HTML页面，所链接的页面也都在自己的计算机上。接下来我们会有所改变。这一章会鼓励你把这些Web页面发布到互联网上，让你的朋友、粉丝和客户都能真正看到。通过破解 h, t, t, p, :, /, /, w, w, w代码，我们还将揭开链接到其他页面的秘密。好了，收拾好行李，向下一站Web镇进发。

提醒一句：一旦到了Web镇，你可能就永远不会回来了。别忘了给我们寄明信片。



在Web上发布Starbuzz网站（或你自己的网站）

你自己都没有想到，居然这么快就要把Starbuzz网站（或者也许更棒，要把你自己的网站）发布到Web上。你要做的就是找到一个“Web托管公司”（以后我们就把它叫做“托管公司”），在他们的服务器上托管你的页面，然后把页面从你的计算机复制到他们的某个服务器上。

当然，如果了解你的本地文件夹如何“映射”到服务器上的文件夹，另外知道一旦把页面放在服务器上，该如何指示浏览器访问你的页面，那就好办了。不过关于这些，我们都会告诉你。现在，先来讨论如何发布到Web上。你要做这样几件事：

1. **找一家托管公司。**
2. **为你的网站选一个名字（比如www.starbuzzcoffee.com）。**
3. **想办法把文件从你的计算机上传到托管公司的服务器上（这样的方法有很多）。**
4. **把你的新网站告诉你的朋友、家人和粉丝，开始享受你的成果。**

我们会带你一步一步完成网站的发布，即使你不打算现在就让网站上线，也可以跟着我们一起学一学，因为你会学到以后需要知道的一些重要知识。来吧，做好准备，从HTML开始我们的Web之旅吧……

找一家托管公司

要在Web上发布网页，需要一个在Web上全天候工作的服务器。你最好的选择是找一家托管公司，让他们负责考虑保证服务器一直运行的诸多细节问题。不过，不用担心，找一家托管公司很容易，也不贵。

哪家公司？嗯，我们希望你在Head First Hip Web Hosting公司完成托管，开玩笑的，这家公司实际上并不存在。所以，你得自己做点功课。尽管找一家公司托管你的页面并不难，不过这有点类似于选择有线电视公司：有太多的选择和方案。你需要货比三家，找到最合适的价位和服务。

有个好消息，即使你几乎身无分文也可以开始，可以等以后需要额外特性时再进行升级。尽管我们不会对具体哪一家提供商最好给出建议，不过可以告诉你寻找提供商时要注意的一些事项，另外我们还列出了一些比较受欢迎的提供商，见http://wickedlysmart.com/hosting-providers/

市场部门提示：当然了，如果哪家托管公司给我们一大笔钱，我们也可以在这里给它做广告！

轻松一刻

学习这本书并不要求把页面发布到Web上。

如果你的页面能真正发布到Web上，肯定很有意思，不过，只用你自己的计算机也完全能顺利学完这本书。

不管是否真正发布网站，都应当好好学习后面几页，了解网站发布的全过程。

一分钟托管指南

我们肯定没办法把选择托管公司需要了解的方方面面全部告诉你（毕竟，这是一本关于HTML和CSS的书），不过我们可以给你指出一个正确的方向。选择托管公司时要考虑以下几点。

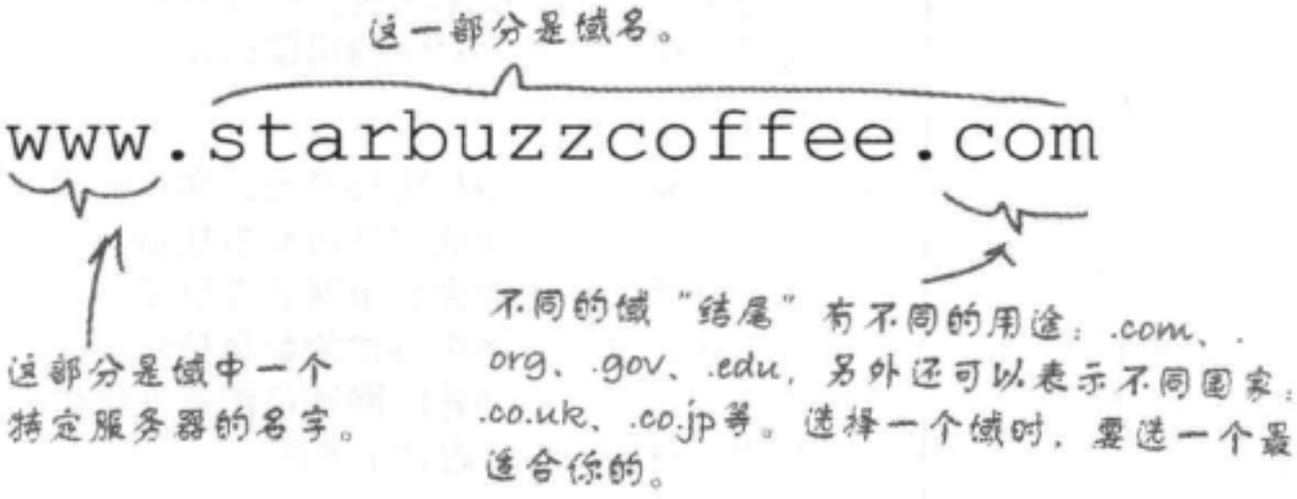
- 技术支持：这家托管公司有没有一个好的系统来处理你的技术问题？比较好的公司会通过电话或邮件对你的问题迅速做出回应。
- 数据传输：这是指托管公司允许你在一定时间内向访问者发送的页面和数据量。大多数托管公司可能会为小网站提供最基本的方案，这些方案中提供的数据传输量并不大。如果你要创建的网站可能有大量访问者，就要仔细考虑这个方面。
- 备份：这家托管公司是否会定期对你的页面和数据做备份，从而在服务器出现硬件故障时能够恢复？
- 域名：托管公司定价中是否包括一个域名？有关的更多内容会在下一页介绍。
- 可靠性：大多数托管公司都声称保证网站99%以上的时间内都能正常运行。
- 赠品：是否附送了其他赠品，比如Email地址、论坛或脚本语言支持（这对你将来会很重要）？

域名

你好，我的名是……

也许你从未听说过“域名”这种说法，但实际上你肯定早已经见过并用过无数次域名了，要知道……google.com、facebook.com、amazon.com、disney.com还有一些你不希望我们提到的网站都是域名。

那么域名到底是什么？就是一个唯一的名字，可以用来定位你的网站。下面给出一个例子：



我们要重视域名，这有两方面原因。如果你希望你的网站有独一无二的名字，就需要有自己的域名。域名还会用来将你的页面链接到其他网站（后面几页会介绍这个内容）。

还有一件事你要知道。域名由一个集中的权威机构（ICANN）控制，以确保一次只能有一个人使用某个域名。另外（你可能已经预料到了），要保留你的域名，每年还需要交纳少许注册费。

如何得到一个域名？

简单的答案是，这个问题可以交给你的托管公司来考虑。他们通常会提供域名注册作为其服务中的一项。不过，还有上百家公司很乐于提供帮助，可以在这里找到这些公司的一个清单：

http://www.internic.net/regist.html

与寻找托管公司一样，恐怕也得你亲自去寻找和注册你自己的域名。你会发现，交给托管公司来办是最容易的。

there are no Dumb Questions

问：为什么把它叫做“域名”而不是“网站名”？

答：因为它们完全是两码事。如果你看到www.starbuzzcoffee.com，这就是一个网站名，不过如果只有“starbuzzcoffee.com”部分，就是域名。还可以创建使用相同域名的其他网站，比如corporate.starbuzzcoffee.com或employees.starbuzzcoffee.com。所以域名可以用于多个网站。

问：如果我想获得Starbuzz的域名，是不是会得到www.starbuzzcoffee.com？看起来所有人都在使用www开头的网站。

答：重申一次，不要把域名和网站名弄混了：starbuzzcoffee.com是一个域名，而www.starbuzzcoffee.com是一个网站名。购买域名就像买下一块地，比如说100mainstreet.com。在这块地上，你可以随你所愿建很多网站，例如：home.100mainstreet.com、toolshed.100mainstreet.com和outhouse.100mainstreet.com。所以www.starbuzzcoffee.com只是starbuzzcoffee.com域上的一个网站。

问：那么，域名到底有什么好处呢？我真的需要一个域名吗？我的托管公司说我可以直接使用他们的域名，比如www.dirtcheaphosting.com，这样可以吗？

答：如果能满足你的需要的话，则使用他们的域名倒也无妨。不过（特别强调一下）这样做有一个缺点：如果你想选择另一家托管公司，或者如果这家托管公司破产了，以前知道你的网站的人就没办法找到你的网站了。另一方面，如果你有自己的域名，就可以带着它选择新的托管公司（而且你的用户甚至根本不知道你已经换了托管公司）。

问：如果域名是独一无二的，这说明有人可能已经在用我想用的域名，怎么查出来呢？

答：这个问题问得好！大多数提供域名注册服务的公司都允许你搜索一个域名是否已经被占用（有点像搜索未用的车牌号）。可以在http://www.internic.net/regist.html找到这些公司的一个清单。

这里有一个练习，你要自己来完成。我们个人很乐于帮忙，不过作为这本书的作者，有些问题我们实在无法给出答案（另外也无法在你外出度假时帮你喂猫）。

自己试一试

现在要找到一家托管公司，为网站获得一个域名。要记住，你可以访问Wickedly Smart来得到一些建议和资源。另外还要记住，即使没有完成这个练习，你也能顺利读完这本书（不过确实应该做一做）。

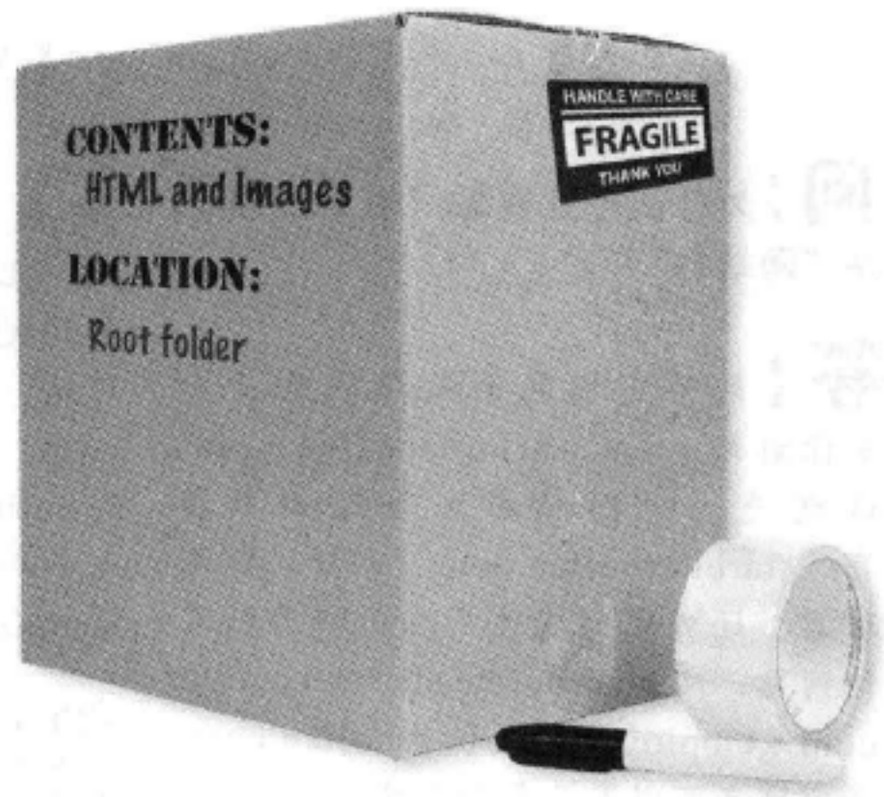
我的Web托管公司： .....

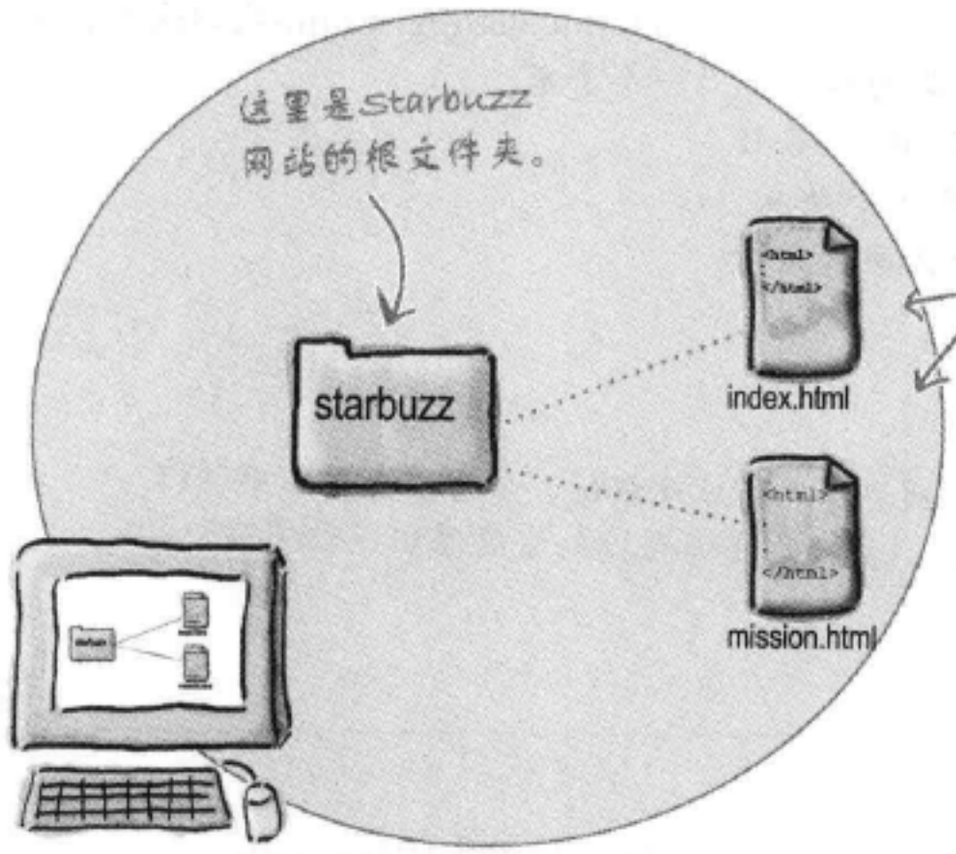
我的域名： .....

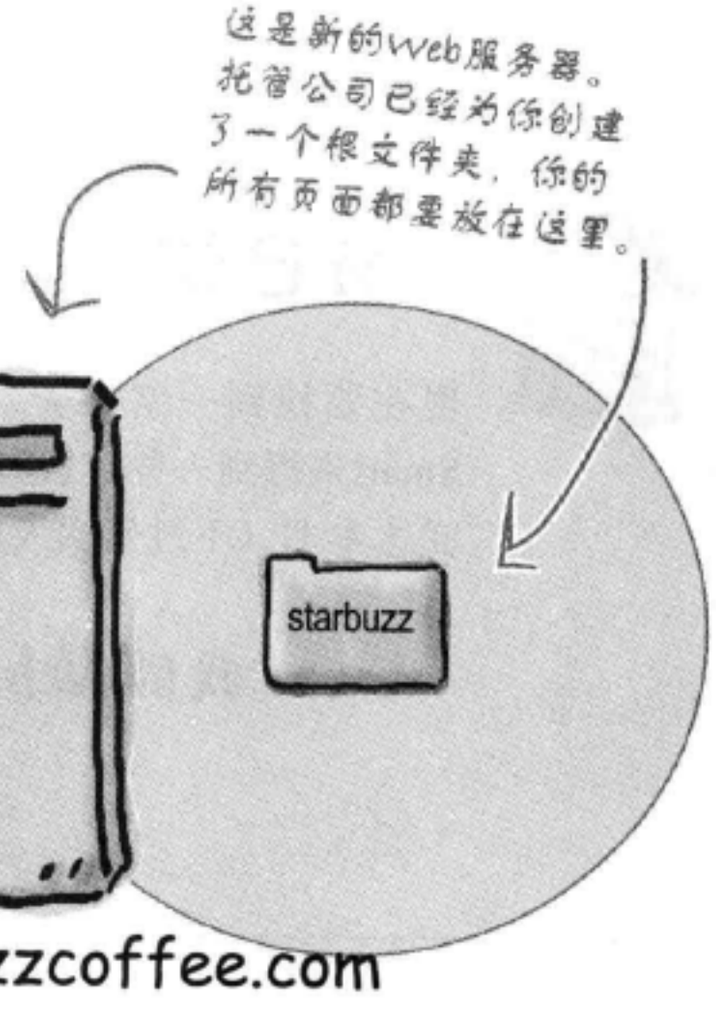
搬家

祝贺你！你已经找到了托管公司，得到了一个域名，另外已经找好了一个服务器来维护你的Web页面（即使还没有，也请先跟着我们继续学习，因为现在要讲很重要的内容）。

现在该做什么呢？嗯，当然是搬家。取下“出售”的牌子，整理好所有文件，我们要把它们搬到新服务器上去。与平常的搬家一样，我们的目标是把东西从一个地方（比如说老房子的厨房）搬到另一个地方（如新家的厨房）。在Web上，我们只考虑把文件从你自己的根文件夹转移到Web服务器上的根文件夹。下面回到Starbuzz，一步一步来搬。现在的情况是这样的：







there are no Dumb Questions

问：等一下，再问一次“根文件夹”是什么？

答：到现在为止，根文件夹一直都是页面的顶级文件夹。在Web服务器上，根文件夹更为重要，因为根文件夹中的内容可以从网上访问到。

问：我的托管公司好像把我的根文件夹命名为“mydomain_com”。这样有问题吗？

答：没问题。托管公司可能会对你的根文件夹取很多不同的名字。重点在于你要知道你的根文件夹在服务器的什么位置，而且可以把你的文件复制到这里（稍后就介绍这个内容）。

问：我再确认一下。我们把这个网站的所有页面都放在一个文件夹里，这个文件夹称做根文件夹。现在我们要把它们统统复制到服务器的根文件夹，是这样吗？

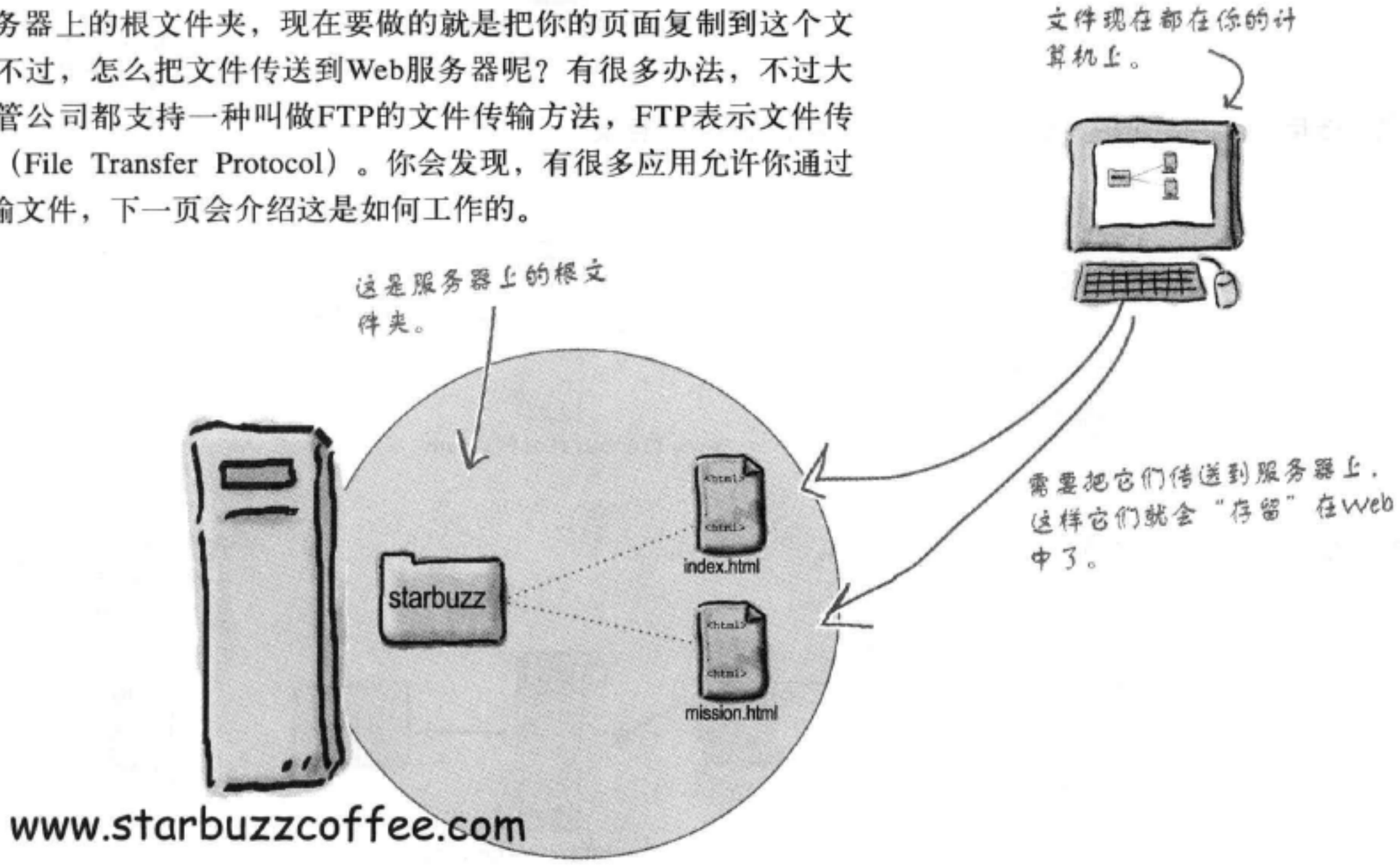
答：完全正确。你要把你自己的计算机上的所有页面统统都放到托管公司服务器上，具体来说要把它们放在对应你的网站的根文件夹下。

问：子文件夹呢？比如“images”文件夹？这些也要复制吗？

答：没错，实际上你要把你自己的根文件夹下的所有页面、文件和文件夹都复制到服务器上。所以，如果你的计算机上有一个“images”文件夹，那么服务器上也应该有这样一个文件夹。

把你的文件复制到根文件夹

现在离把Starbuzz网站发布到Web上只有一步之遥了：你已经找到托管公司服务器上的根文件夹，现在要做的就是把你的页面复制到这个文件夹。不过，怎么把文件传送到Web服务器呢？有很多办法，不过大多数托管公司都支持一种叫做FTP的文件传输方法，FTP表示文件传输协议（File Transfer Protocol）。你会发现，有很多应用允许你通过FTP传输文件，下一页会介绍这是如何工作的。

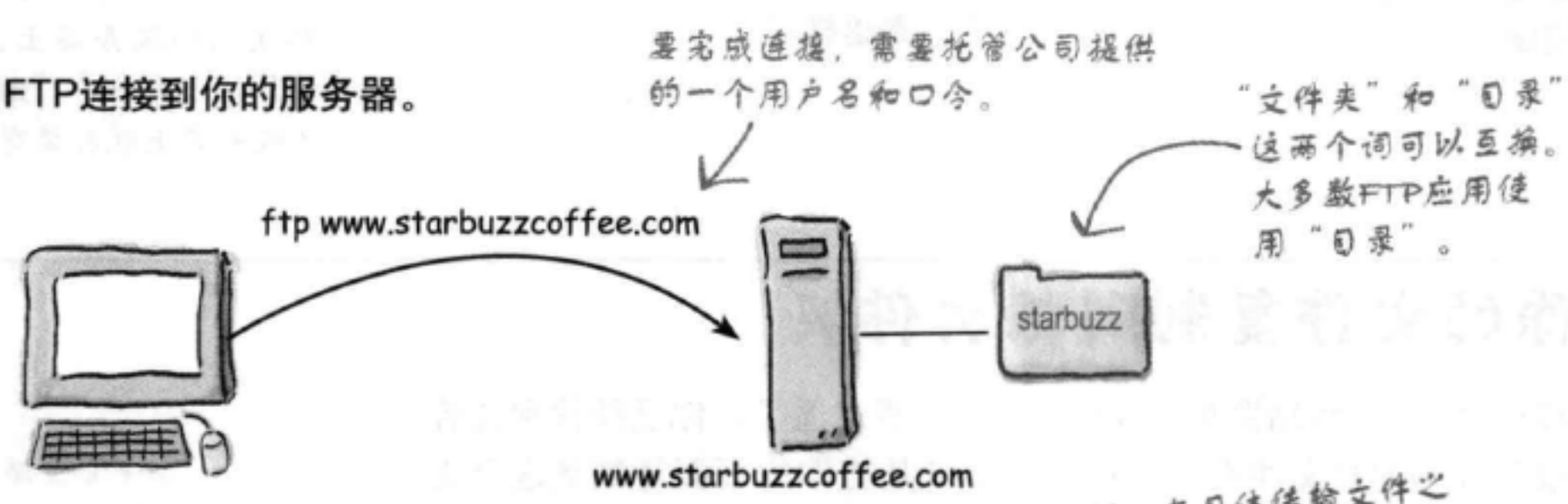


用两页尽可能讲清楚FTP

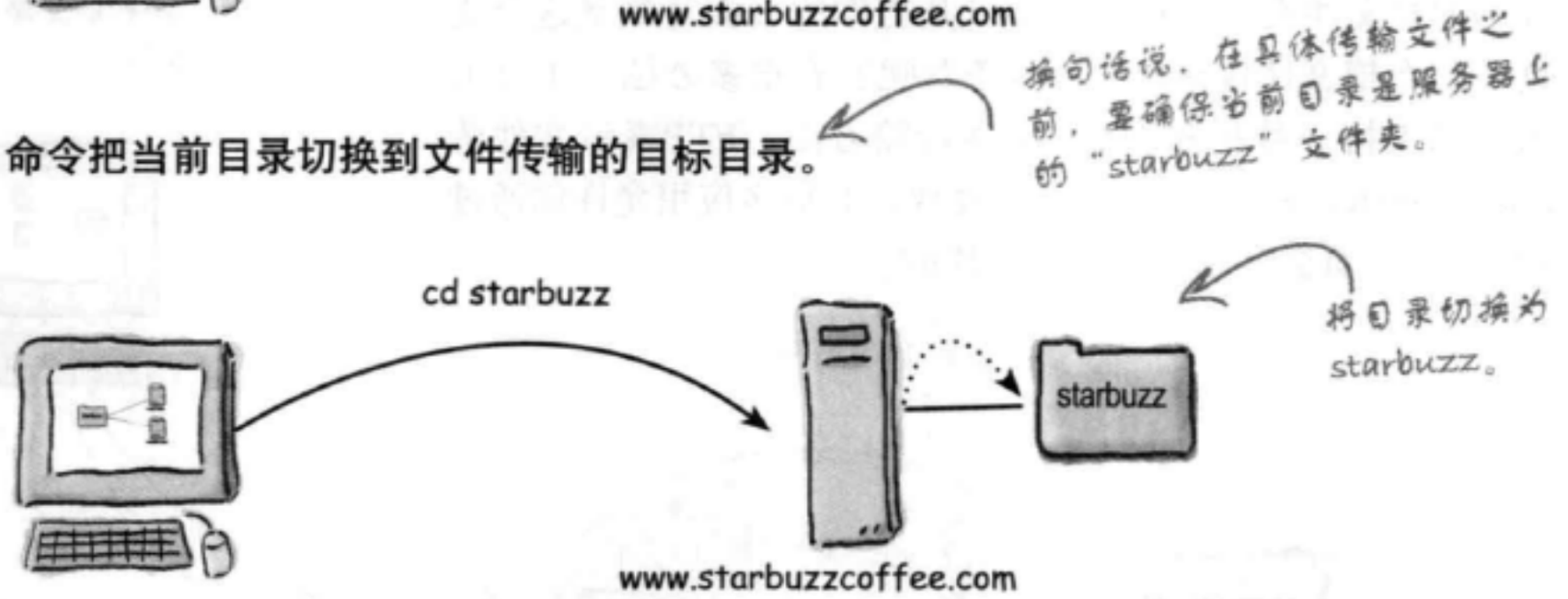
严格地说，这是一本关于HTML和CSS的书，不过我们不想让你独自面对挑战。所以，下面给出一个简短的指南，告诉你如何使用FTP将文件上传到Web。 要记住，你的托管公司可能会对如何将你的文件上传到他们的服务器给出一些建议（你已经付过钱了，可以让他们来帮你）。这几页之后，我们还会回到正题上，继续讨论HTML和CSS，直到全书结束（我们保证）。

假设你已经找到一个FTP应用。有些FTP应用是命令行应用，有些有完备的图形界面，还有一些甚至内置于Dreamweaver和Expression Web之类的应用中。它们都使用相同的命令，不过有些应用要求你自己输入命令，而另外一些允许你使用图形界面。下面概要地介绍一下FTP如何工作：

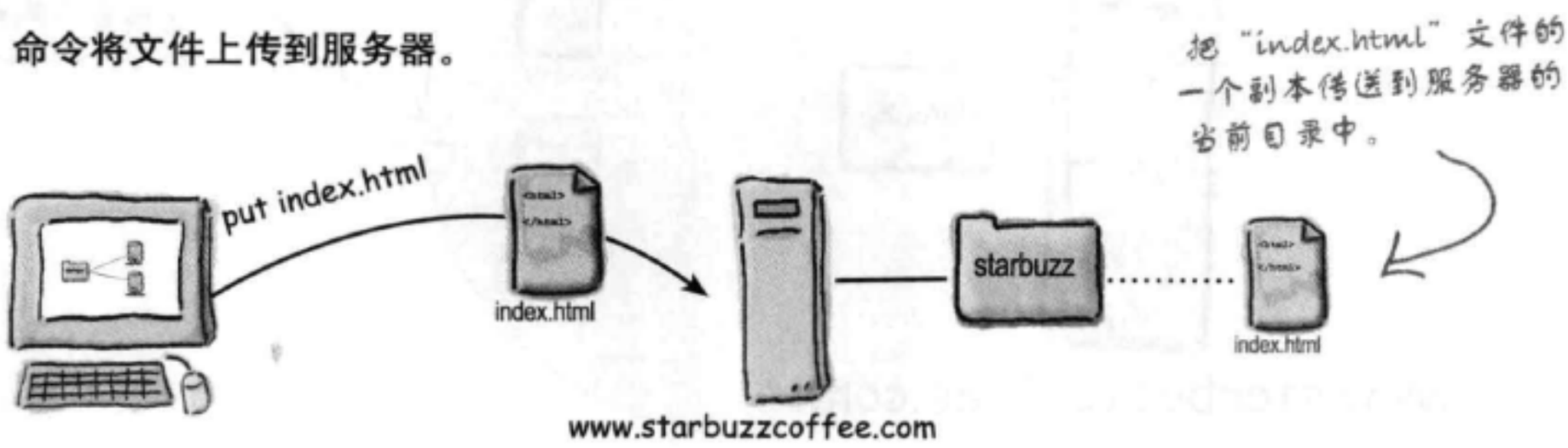
① 首先，使用FTP连接到你的服务器。



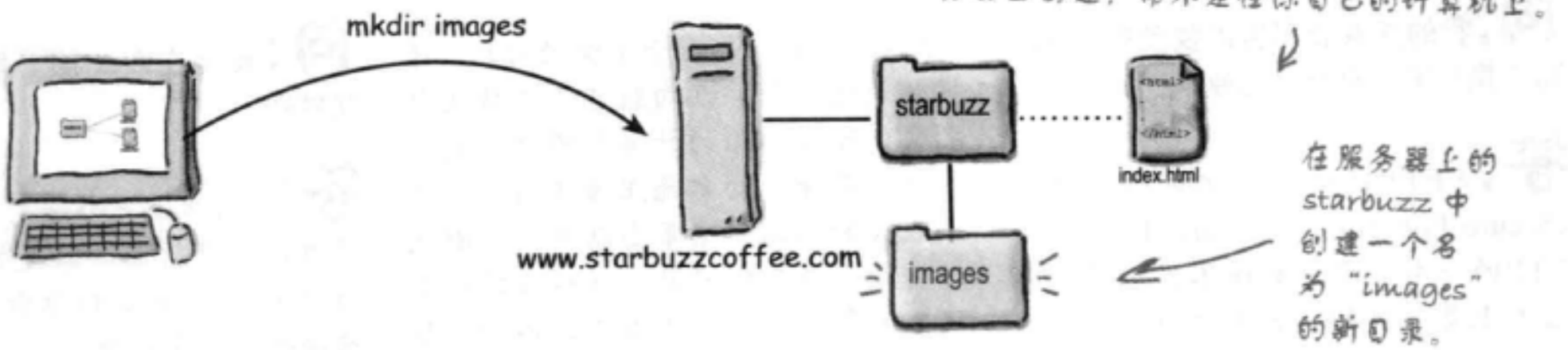
② 使用“cd”命令把当前目录切换到文件传输的目标目录。



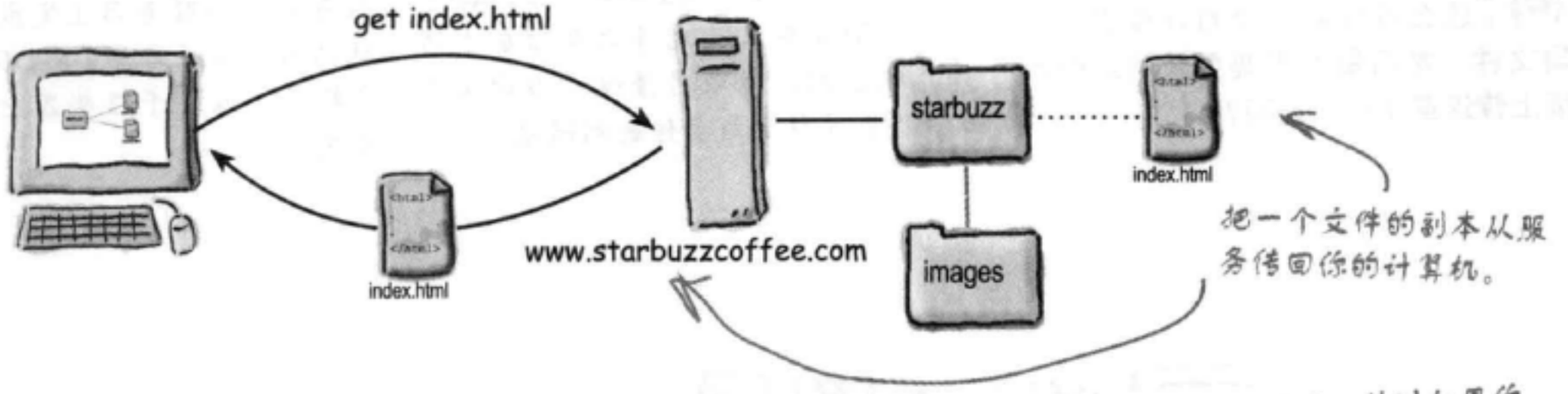
③ 使用“put”命令将文件上传到服务器。



④ 还可以用 “mkdir” 命令在服务器上创建一个新目录。



⑤ 还可以用 “get” 命令获取文件。



下面把这些步骤综合起来。这里给出一个从命令行应用使用FTP的例子：

大多数FTP应用都提供了更为友好的图形界面，所以如果你在使用这样一个FTP应用，可以略过这个例子不看。

```
File Edit Window Help Jam
%ftp www.starbuzzcoffee.com
Connected to www.starbuzzcoffee.com
Name: headfirst
Password:******
230 User headfirst logged in.
ftp> dir
drwx------  4096 Sep  5 15:07 starbuzz
ftp> cd starbuzz
CWD command successful
ftp> put index.html
Transfer complete.
ftp> dir
-rw-------  1022 Sep  5 15:07 index.html
ftp> mkdir images
Directory successfully created
ftp> cd images
CWD command successful
ftp> bye
```

连接并登录。
查看当前目录里有什么。
一个名为starbuzz的目录。
切换到starbuzz目录。
传入index.html。
查看这个目录，现在包含有index.html。
为图像创建一个目录，然后用bye命令退出。

FTP命令

不论你是在命令行上输入FTP命令，还是使用一个有图形界面的FTP应用，执行的命令或操作基本上都是一样的。

- dir: 得到当前目录的文件列表。
- cd: 切换到另一个目录。“..”也表示上一层目录。
- pwd: 显示当前目录。
- put <filename>: 将指定的文件传送到服务器。
- get <filename>: 从服务器获取指定的文件，传回你的计算机。

there are no Dumb Questions

问：我的托管公司告诉我使用SFTP，而不是FTP。有什么区别吗？

答：SFTP就是安全文件传输协议（Secure File Transfer Protocol），这是FTP的一个更安全的版本，不过工作基本上是一样的。购买之前要确保你的FTP应用支持SFTP。

问：这么说我要在我的计算机上编辑文件，然后每次想要更新网站时都要上传这些文件，是吗？

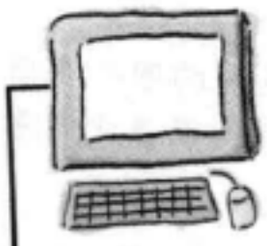
答：是这样的，对于小型的网站，通常就是这么做。在向服务器上传文件之前，可以用你的计算机测试你做的修改，确保一切都能正常工作。对于比较大的网站，通常会创建一个测试网站和一个真实网站，这样在转向真实网站之前，可以先在测试网站上预览修改的结果。

如果你使用类似Dreamweaver或Coda等工具，则这些工具允许你在你自己的计算机上测试所做的修改，当你保存文件时，文件会自动传送到网站。

问：我能在Web服务器上直接编辑文件吗？

答：这通常不是个好主意，因为在你有时间预览页面和修复错误之前，访问网站的人会看到你做的所有修改，错误也会一览无余。

不过，一些托管公司确实允许你登录服务器，在服务器上完成修改。想要这么做，你还需要了解DOS或Linux命令提示（取决于服务器上运行的操作系统）。



流行的FTP应用

下面是Mac和Windows上最流行的一些FTP应用：

对于Mac OS X:

- Fetch (http://fetchsoftworks.com/) 是Mac上最流行的FTP应用之一。收费。
- Transmit (http://www.panic.com/transmit/)。收费。
- Cyberduck (http://cyberduck.ch/)。免费。
- 对于Windows:
- Smart FTP (http://www.smartftp.com/download/)。收费。
- WS_FTP (http://www.wsftple.com/)。基本版免费，专业版收费。
- Cyberduck (http://cyberduck.ch/)。免费。

大多数FTP应用都有一个测试版，可以先下载试用，再考虑是否购买。

自己试一试

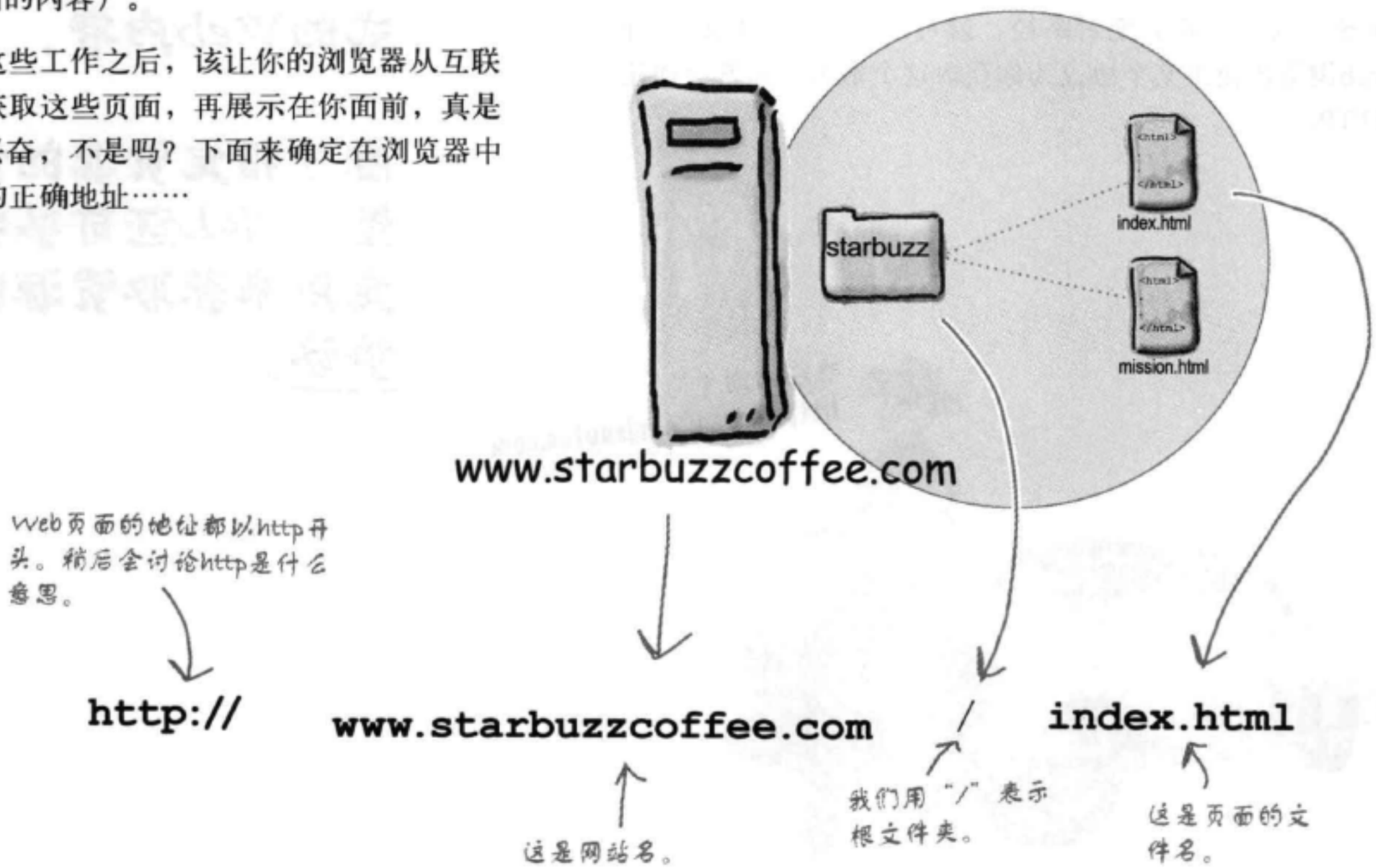
这是给你留的另一个家庭作业（做完一项就打一个勾）：

- ☐ 你要知道你的根文件夹在托管公司服务器上的位置。
- ☐ 找到从你的计算机向服务器传送文件的最佳方法（和最佳工具）。
- ☐ 现在将Starbuzz的“index.html”和“mission.html”文件传送到服务器的根文件夹中。

回到正题……

Web之旅结束了，现在我们再回到正道上来。目前你已经有了两个Starbuzz页面：“index.html”和“mission.html”，它们都在你的根文件夹中（如果不是，则请先跟着我们继续学习下面的内容）。

完成这些工作之后，该让你的浏览器从互联网上获取这些页面，再展示在你面前，真是令人兴奋，不是吗？下面来确定在浏览器中输入的正确地址……



我们的主干道，USA

听到“h”“t”“t”“p”“冒号”“斜杠”“斜杠”是不是很熟悉，已经听到过无数次了吧？那么这到底是什么意思？首先要知道，你在浏览器中输入的Web地址称为URL，或统一资源定位符（Uniform Resource Locators）。

如果让我们取名，则我们可能会叫它“Web地址”， 不过没人问过我们，所以我们也只好沿用这个名字，把它叫做统一资源定位符。下面来剖析一个URL：

```
http://www.starbuzzcoffee.com/index.html
```

URL的第一部分指出用来获取资源的协议。

第二部分是网站名。现在你对这一部分应该很了解了。

第3部分是从根文件夹到资源的绝对路径。

要定位Web上的某个资源，只要你知道维护这个资源的服务器，以及资源的绝对路径，就可以创建一个URL，让Web浏览器使用某个协议为你获取这个资源，通常会使用HTTP。

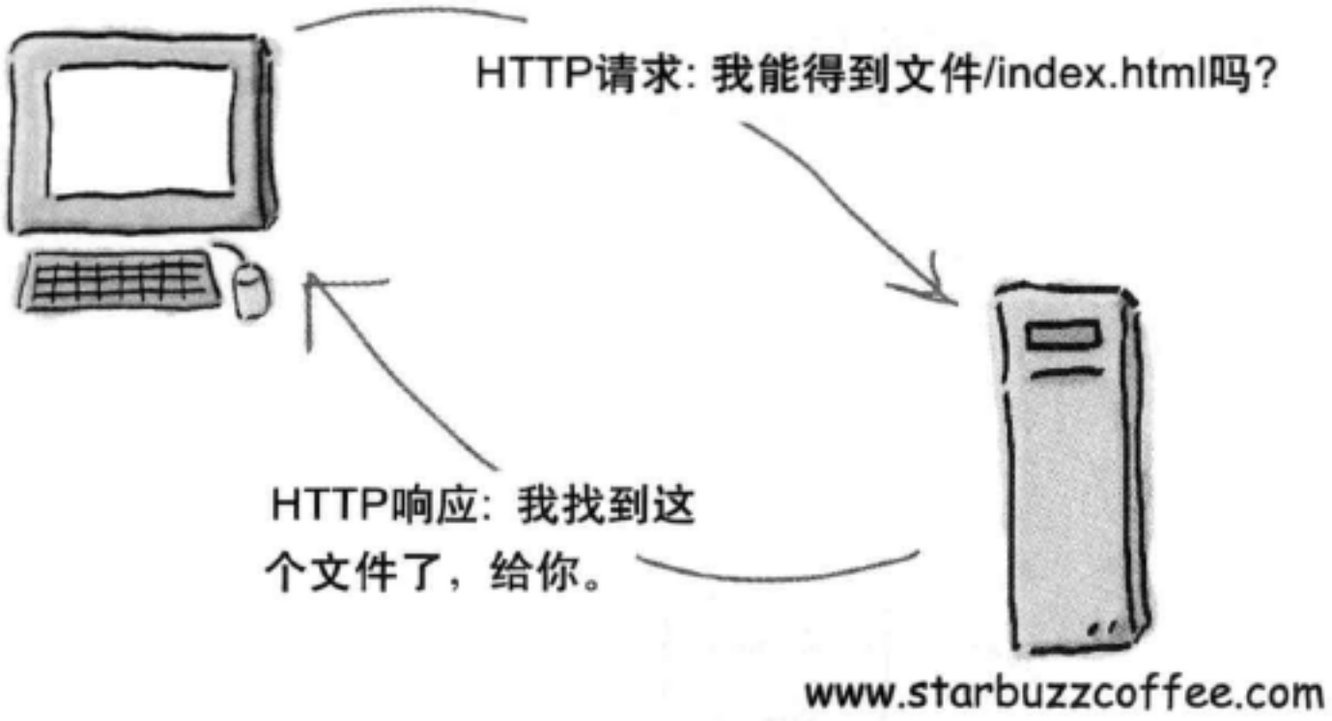
统一资源定位符（Uniform Resource Locator，URL）是一个全局地址，可以用来定位Web上的任意资源，包括HTML页面、音频、视频和很多其他形式的Web内容。

除了指定资源的位置，URL还可以指定用来获取资源的协议。

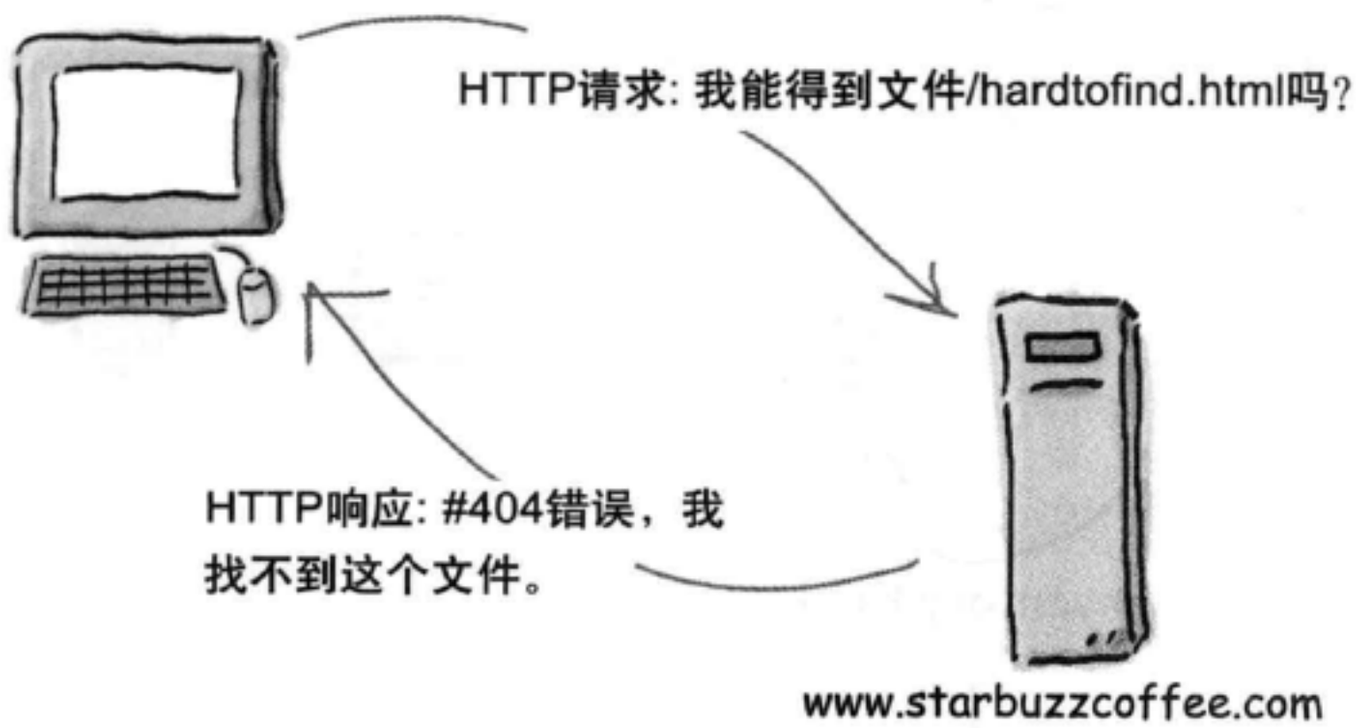
什么是HTTP?

HTTP也称为超文本传输协议（HyperText Transfer Protocol）。换句话说，这是在Web上传输超文本文档的公认的一种方法（协议）。尽管“超文本文档”通常只是指HTML页面，但这个协议实际上还可以用来传输图像或Web页面可能需要的任何其他文件。

HTTP是一个简单的请求和响应协议。它是这样工作的：



所以说，每次在浏览器的地址栏中输入一个URL时，浏览器就会使用HTTP向服务器请求相应的资源。如果服务器找到这个资源，就会把它返回给浏览器，由浏览器显示。如果服务器找不到呢?



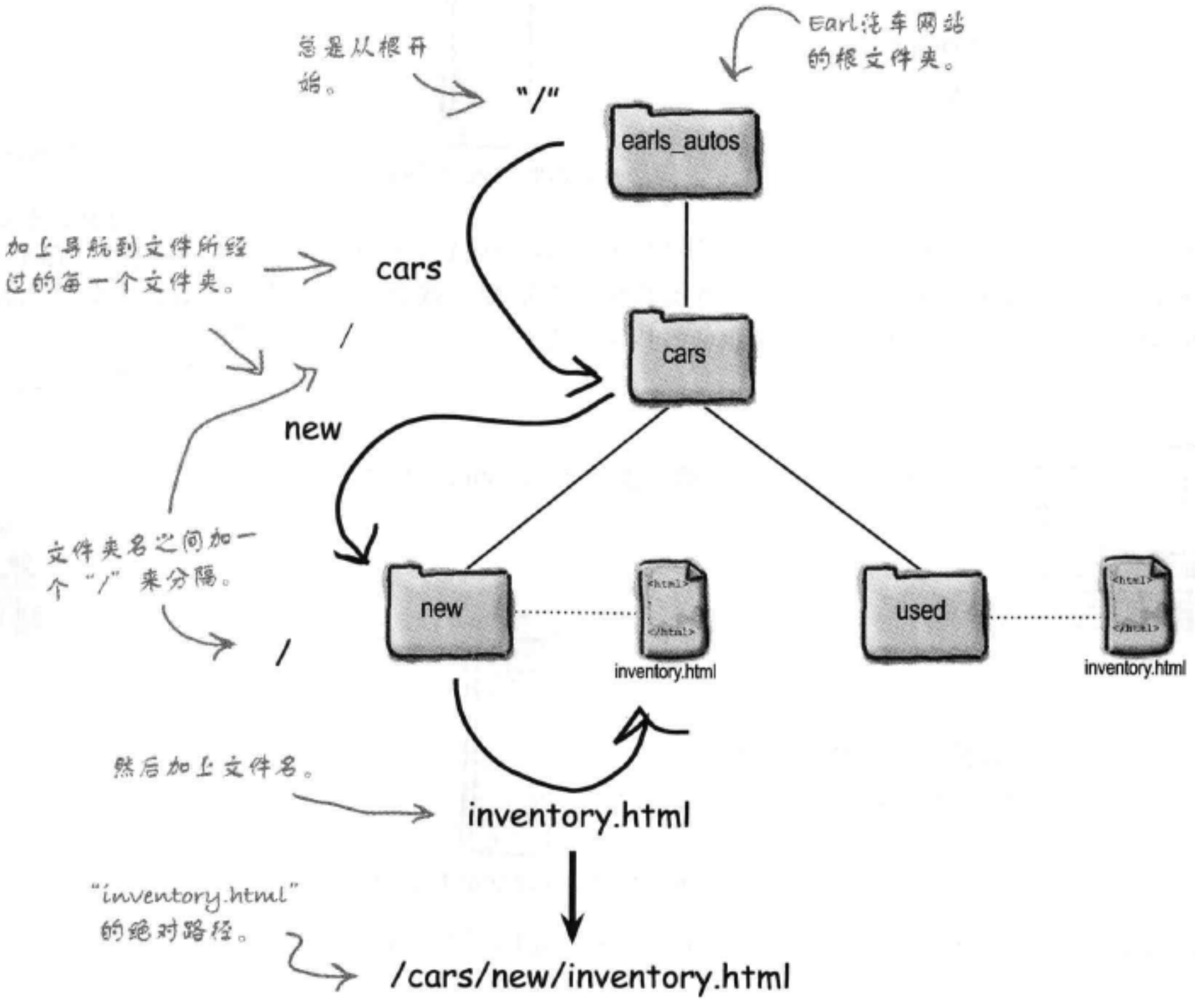
如果无法找到资源，则服务器会向浏览器报告我们熟悉的“404错误”。

什么是绝对路径？

上一次我们讨论路径时，是要编写HTML用<a>元素建立链接。现在我们要讨论的路径是URL中的绝对路径部分，就是在协议（HTTP）和网站名（www.starbuzzcoffee.com）之后的最后一部分。

绝对路径告诉服务器如何从你的根文件夹到达某个特定的页面或文件。例如，以Earl的汽车网站为例。假设你想查看Earl的库存，看看你订的Mini Cooper是否到货。为此，你需要确定到文件“inventory.html”的绝对路径，这个文件在“new”文件夹下。你要做的就是从根文件夹开始追踪文件夹，直到找到“inventory.html”文件所在的“new”文件夹。这个路径由沿路经过的所有文件夹组成。

所以，看起来这包括根（我们用“/”表示根）、“cars”和“new”，最后是文件本身“inventory.html”。下面把它们综合在一起：



there are no Dumb Questions

问：绝对路径有什么意义？

答：服务器需要绝对路径来找到你请求的文件。如果服务器没有得到绝对路径，它就不知道去哪里查找。

问：我觉得每个部分我都明白了（协议、服务器、网站和绝对路径），不过把它们联系在一起还是有些困难。

答：如果把所有这些部分结合在一起，你就有了一个URL，利用URL，你可以请求浏览器从Web获取一个页面 （或者其他类型的资源）。怎么做到的？协议部分告诉浏览器应当使用什么方法来获取资源（大多数情况下，协议都是HTTP）。网站部分（由服务器名和域名组成）告诉浏览器要从互联网上的哪个计算机获取资源。最后绝对路径会告诉服务器我们要找哪个页面。

问：我们学过可以在<a>元素的href属性中放入相对路径。既然它们不是绝对路径，服务器是怎么找到的呢？

答：嗯，这个问题问得好。单击一个相对链接时，在后台浏览器会根据这个相对路径和所单击页面的路径创建一个绝对路径。所以，所有Web服务器看到的都是绝对路径，这要归功于浏览器。

问：如果在HTML中加入绝对路径，那么会不会对浏览器有帮助？

答：哈，这个问题也很妙，不过先把这个问题放一放，稍后再做解释。

Sharpen your pencil

让你久等了。现在该拿你的新URL试一试了。开始之前，先完成下面的填空，然后输入这个URL（如果还没有输入的话）。如果遇到问题，则现在可以请托管公司帮忙一起来解决问题。如果还没有找托管公司，则你可以先填上www.starbuzzcoffee.com，再在浏览器中输入这个URL。

.....　　://　　.....　　.....

协议　　网站名　　绝对路径

我希望访问者只需输入“http://www.starbuzzcoffee.com”，而不用再输入“index.html”。有没有办法做到？

可以的。有一种情况我们还没有谈到，就是如果浏览器向Web服务器请求一个目录而不是文件时会发生什么。例如，浏览器可能会请求：

http://www.starbuzzcoffee.com/images/

或

http://www.starbuzzcoffee.com/

Web服务器接收到一个类似这样的请求时，它会尝试查找这个目录中的一个默认文件。通常默认文件名为“index.html”或“default.htm”，如果服务器找到这样一个文件，就会把它返回给浏览器显示。

所以，要从根目录（或任何其他目录）默认地返回一个文件，只需要把这个文件命名为“index.html”或“default.htm”。

要记住，我们讨论Web服务器或FTP时，通常会使用“目录”而不是“文件夹”。不过它们都是一样的意思。

根目录中的images目录。

根目录本身。

不过你要了解你的托管公司希望你如何命名默认文件，因为这取决于他们使用哪种类型的服务器。

不过我请求的是“http://www.starbuzzcoffee.com”，看起来稍有不同。末尾没有“/”。

确实如此。如果服务器接收到这样一个末尾没有“/”的请求，而且这个目录确实存在，服务器就会帮你加上末尾的斜线。所以如果服务器接收到以下请求：

http://www.starbuzzcoffee.com

它会把这个请求改为

http://www.starbuzzcoffee.com/

这会让服务器查找一个默认文件，最后会返回这个文件，就好像你之前输入了以下地址：

http://www.starbuzzcoffee.com/index.html

默认页面如何工作

在幕后

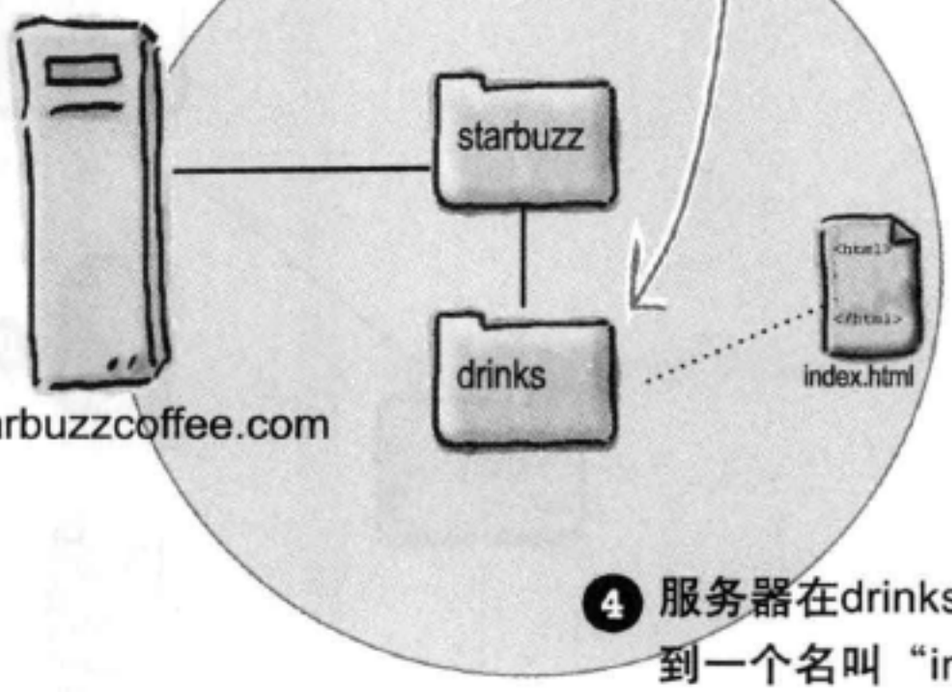
❶ 用户在浏览器中输入
http://www.starbuzzcoffee.com/drinks/

❷ HTTP请求: 我能得到文件 “/drinks/”吗?

❸ 服务器说，“这看上去像是个目录，这个目录里有默认文件吗？”

❹ 服务器在drinks目录中找到一个名叫“index.html”的默认文件。

❺ HTTP响应: 你在请求一个目录，不过我发现这个目录中有“index.html”，所以我会把这个文件发给你。



there are no Dumb Questions

问：这么说，输入URL http://www.mysite.com访问我的网站的人都会看到我的“index.html”页面，是吗?

答：对。或者也可能是“default.htm”，这取决于你的托管公司使用哪种Web服务器（注意“default.htm”通常结尾没有“l”。这是Microsoft Web服务器的一个怪癖）。

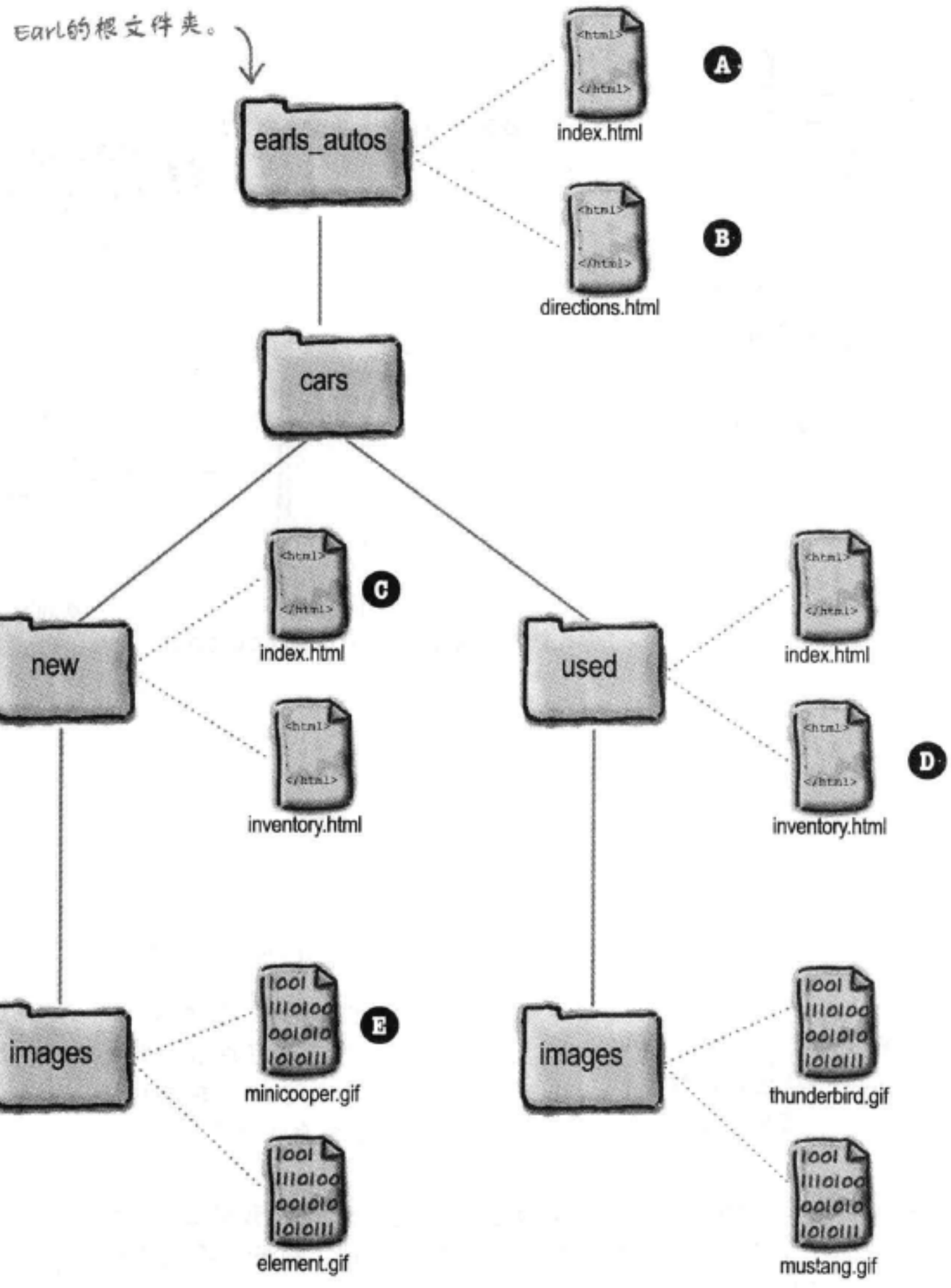
问：所以我告诉别人我的URL时，加上“index.html”部分是不是更好些?

还有其他一些可能的默认文件名，如“index.php”,你开始写脚本来生成页面时就会用到。这超出了本书的范畴，不过你将来很可能还会遇到。

答：不。最好不要加。如果将来你改变了Web服务器，而它使用另一个默认文件名，比如“default.htm”，则会怎么样呢？或者如果你开始写脚本，则使用“index.php”作为默认文件名，又该如何？这样一来，你原来给的URL就不再有效了。

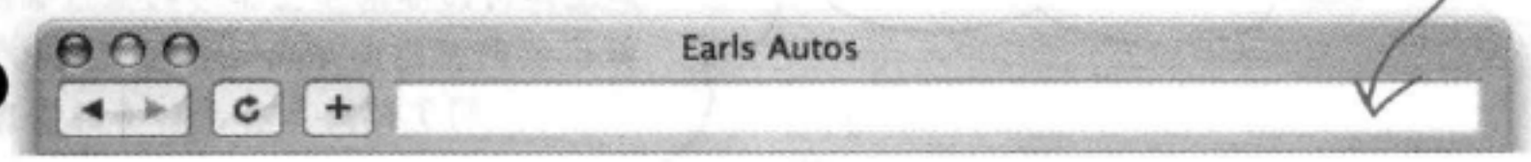
Earl需要你帮他确定URL

Earl只知道自己的名字（Earl），但不知道U-R-L。他需要你的帮助来确定下面各个文件的URL，分别标为A、B、C、D和E。在右边写出从www.earlsautos.com获取相应文件所需的URL。

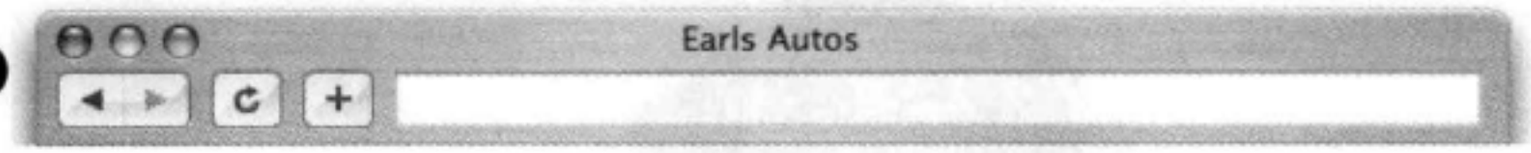


在这里写出URL。

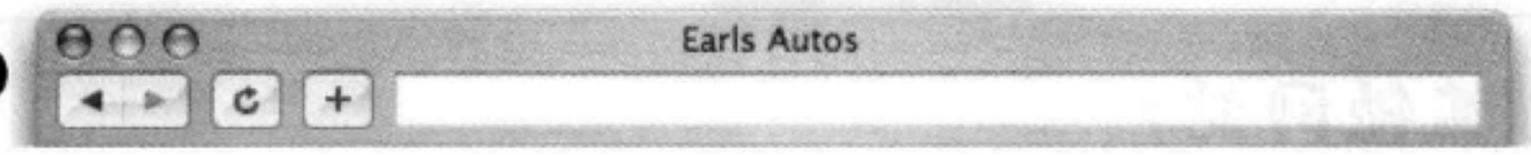
A



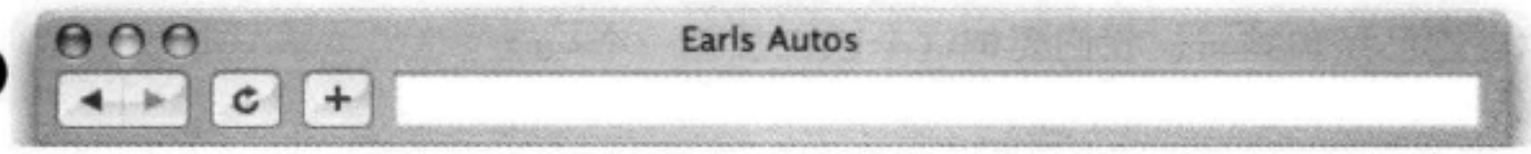
B



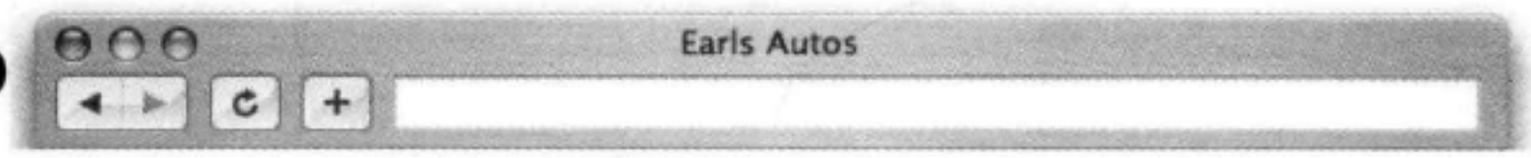
C

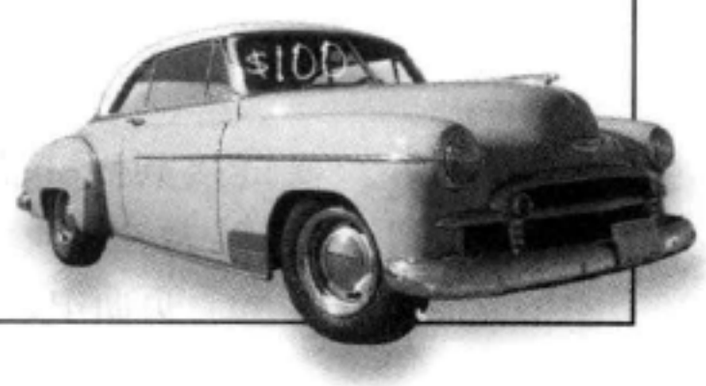


D



E





如何链接到其他网站？

URL并不只是在浏览器中输入，还可以在HTML中使用。当然，我们已经得到提示，Starbuzz　CEO又给你布置了一个新任务：要建立一个从Starbuzz主页面到咖啡信息网站（`http://wickedlysmart.com/buzz`）的链接。你可能已经猜到了，我们要把这个URL放在一个<a>元素中。可以这样做：

```
<a href="http://wickedlysmart.com/buzz">Caffeine Buzz</a>
```

最普通、最平常的<a>元素。

我们在href中放入一个URL。单击“Caffeine Buzz”，将从wickedlysmart.com/buzz获取一个页面。

就这么简单。要链接到Web上的资源，只需要它的统一资源定位符（URL），把这个URL放在<a>元素中作为`href`属性值。下面把它增加到Starbuzz“index.html”页面中。

链接到Caffeine Buzz

打开你的Starbuzz“index.html”文件（在“chapter4/starbuzz”文件夹中），在最后增加两个新链接：一个指向“mission.html”宗旨页面的相对链接，以及一个指向Caffeine Buzz的链接。完成下面的修改，然后在你的浏览器中加载“index.html”文件。单击链接，感受一下Caffeine Buzz。

```
<html>
    <head>
        <title>Starbuzz Coffee</title>
        <style type="text/css">
            body {
                background-color: #d2b48c;
                margin-left: 20%;
                margin-right: 20%;
                border: 2px dotted black;
                padding: 10px 10px 10px 10px;
                font-family: sans-serif;
            }
        </style>
</head>

<body>
    <h1>Starbuzz Coffee Beverages</h1>
    <h2>House Blend, $1.49</h2>
    <p>A smooth, mild blend of coffees from Mexico,
        Bolivia and Guatemala.</p>

    <h2>Mocha Cafe Latte, $2.35</h2>
    <p>Espresso, steamed milk and chocolate syrup.</p>

    <h2>Cappuccino, $1.89</h2>
    <p>A mixture of espresso, steamed milk and foam.</p>

    <h2>Chai Tea, $1.85</h2>
    <p>A spicy drink made with black tea, spices,
        milk and honey.
    </p>
    <p>
        <a href="mission.html">Read about our Mission</a>.
        <br>
        Read the <a href="http://wickedlysmart.com/buzz">Caffeine Buzz</a>.
    </p>
    </body>
</html>
```

这是指向“mission.html”文件的链接。这里使用了一个相对路径来链接“mission.html”。

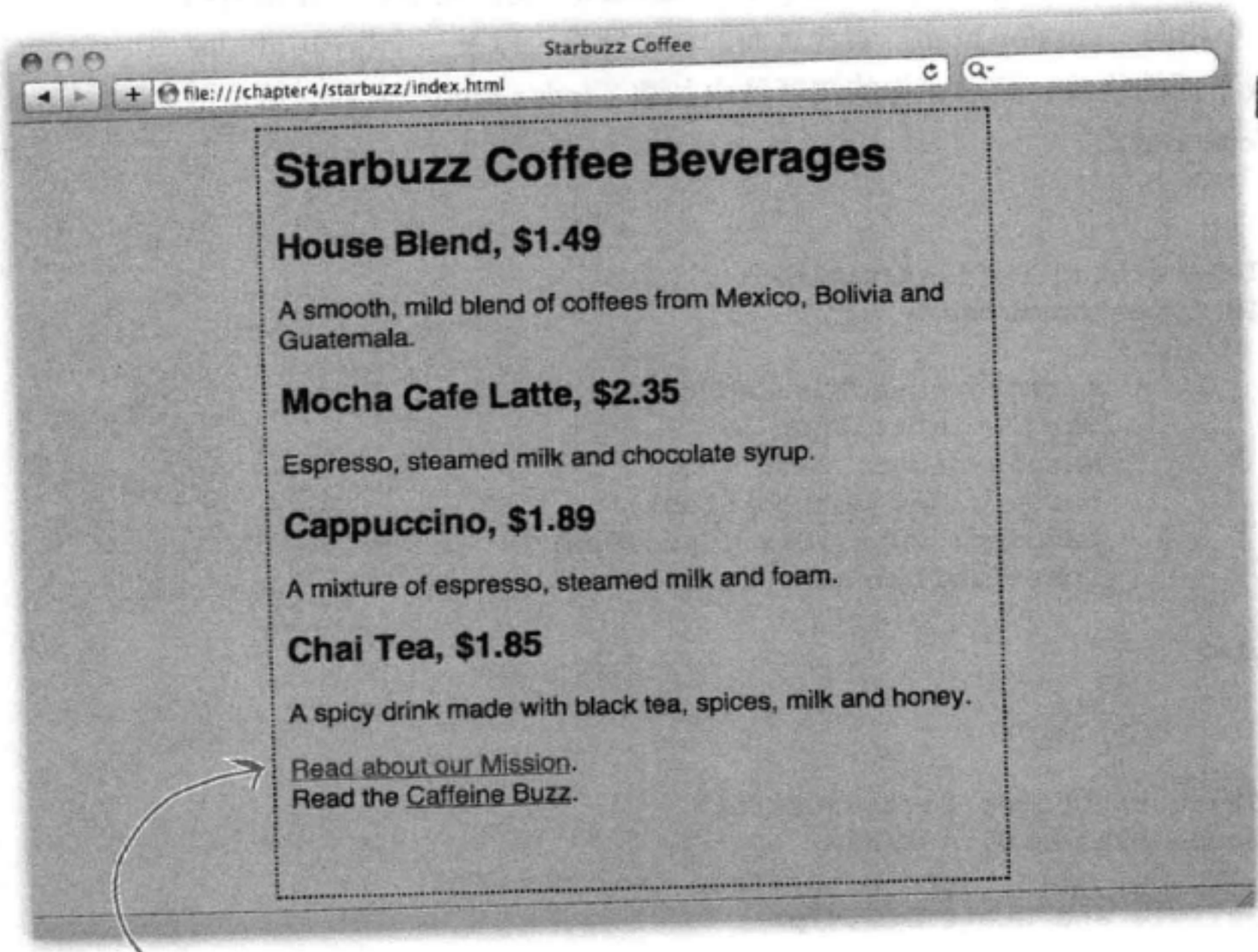
我们增加了一个
，让这两个链接分两行显示。

这里增加了一些结构，把链接和文本组织到一个段落中。

在这里增加指向wickedlysmart.com/buzz页面的链接。

现在来试一试……

这是增加了新链接的页面，与我们计划的完全一致。



这里是新链接。注意，我们只链接了单词“caffeine Buzz”，所以它看上去与另一个链接有所不同。

单击这个链接时，浏览器会向wickedlysmart.com/buzz发出一个HTTP请求，然后显示结果。

there are no Dumb Questions

问：看来现在链接页面有两种方法：相对路径和URL。

答：相对路径只用来链接同一网站内的页面，而URL通常用来链接其他网站。

问：如果我对网站内部页面和外部页面都统一使用URL，那么不是更容易吗？这样也行，是不是？

答：当然，这样确实是可以的，不过出于几方面的原因你可能不想这么做。一个问题是，如果一个Web页面中有很多URL，它们会很难管理：URL很长，不容易编辑，而且也会影响HTML的可读性（对页面编写者而言）。

另外，如果一个网站都使用URL链接到本地页面，移动这个网站或者改变它的名字时，就必须修改所有这些URL来反映新的网站位置。如果使用相对路径，只要页面仍在原来的文件夹中，就不必对<a>元素的href属性做任何修改，因为链接都是相对的。

所以，应当使用相对链接来链接同一网站中的页面，而使用URL来链接其他网站上的页面。

问：还见过其他协议吧？开始使用Web服务器之前我总看到“file:///”。

答：没错。问得好。浏览器从你的计算机本地读取文件时会使用file协议。例如，文件URL“file:///chapter4/starbuzz/index.html”会告诉浏览器文件“index.html”位于“/chapter4/starbuzz/”路径。取决于你的操作系统和浏览器，这个路径可能有所不同。

如果你想输入一个文件URL，则要注意重要的一点是与HTTP不同，文件URL有3个斜线而不是2个。可以这样来记：如果删去一个HTTP URL中的网站名，那么也会有3个斜线。

问：还有其他协议吗？

答：对，很多浏览器可能支持使用FTP获取页面，还有一个mail协议，可以通过Email发送数据。大多数情况下都会使用HTTP。

问：我见过这样的URL，http://www.mydomain.com:8000/index.html。为什么这里还有一个“:8000”？

答：“:8000”是一个可以放在HTTP URL中的可选的“端口”。可以这样来考虑端口：网站名就像一个地址，端口则像是这个地址的邮箱号（例如，在一个复合公寓里）。通常Web上的所有东西都会传送到一个默认端口（80），不过有时Web服务器会配置为在另外一个不同的端口接收请求（如8000）。这种情况经常在测试服务器上出现。正常的Web服务器几乎都在端口80接收请求。如果你没有指定端口，则默认为80。

相对与绝对

PlanetRobots公司现在面对这样一项任务，要为它的两个分公司分别开发一个网站（PlanetRobot Home和PlanetRobot Garden），最后决定与外包公司签约来完成网站建设。RadWebDesign是一家看上去很有经验的公司，承接了Home公司网站，他们只用URL写网站的内部链接（毕竟，URL更复杂，所以应该更好一些）。另一家公司CorrectWebDesign经验较少，不过经过了很好的培训，由它接手PlanetRobot Garden网站的建设，他们使用相对路径来编写网站内部所有页面间的链接。

项目接近尾声时，PlanetRobots发出一个紧急消息："我们被控商标侵权，所以要把域名为RobotsRUs。我们的新Web服务器将是www.robotsrus.com。"CorrectWebDesign只花了五分钟做了一些小改动，就一切OK，开始准备RobotsRUs公司总部的新网站发布会。RadWebDesign则不同，他们一直工作到凌晨4点才修复他们的页面，不过还好，总算在发布会之前完成了工作。但在发布演示中，最可怕的事情出现了：RadWebDesign项目小组组长在演示网站，他单击了一个链接，结果得到一个"404—Page Not Found"（文件未找到）错误。RobotsRUs CEO很不高兴，讽刺RadWebDesign可能应该改名叫BadWebDesign（糟糕的Web设计），然后问CorrectWebDesign能不能协助修复Home网站。

怎么回事？只是改了Web服务器名，为什么RadWebDesign会搞得一团糟？

完善的Web页面

你是不是说过“Web职业生涯”？你已经很好地完成了Starbuzz CEO要求的任务，现在已经建立了一个知名度很高的网站（你的名气也不小）。

不过你可能不想就此止步。你希望你的网站有那种专业的“完善性”，这正是普通网站与最佳网站之间的差距。在这本书中你还会看到可以用很多方法对网站“抛光”，让它更加出色，不过这里先介绍一种改进链接的方法。

为链接增加标题以便访问

对于我们要单击的链接，如果能得到更多信息就好了。这对那些有视力障碍使用屏幕阅读器的人来说尤其重要，因为他们通常不希望把整个URL读出来：（“h” “t” “t” “p” “:” “slash” “slash” “w” “w” “w” “dot”），而链接的标签通常只能提供有限的描述，如“Caffeine Buzz”。

<a>元素有一个title属性，就是用来提供链接信息的。有些人可能被这个属性名搞糊涂了，因为<head>中还有一个名为<title>的元素。之所以名字相同，这是因为它们本来就是相关的，通常建议title属性值与所链接的Web页面的<title>元素值相同。不过这不是一个严格的要求，在title属性中提供你自己的更具体的描述可能更有意义。

可以像这样为<a>增加title属性：

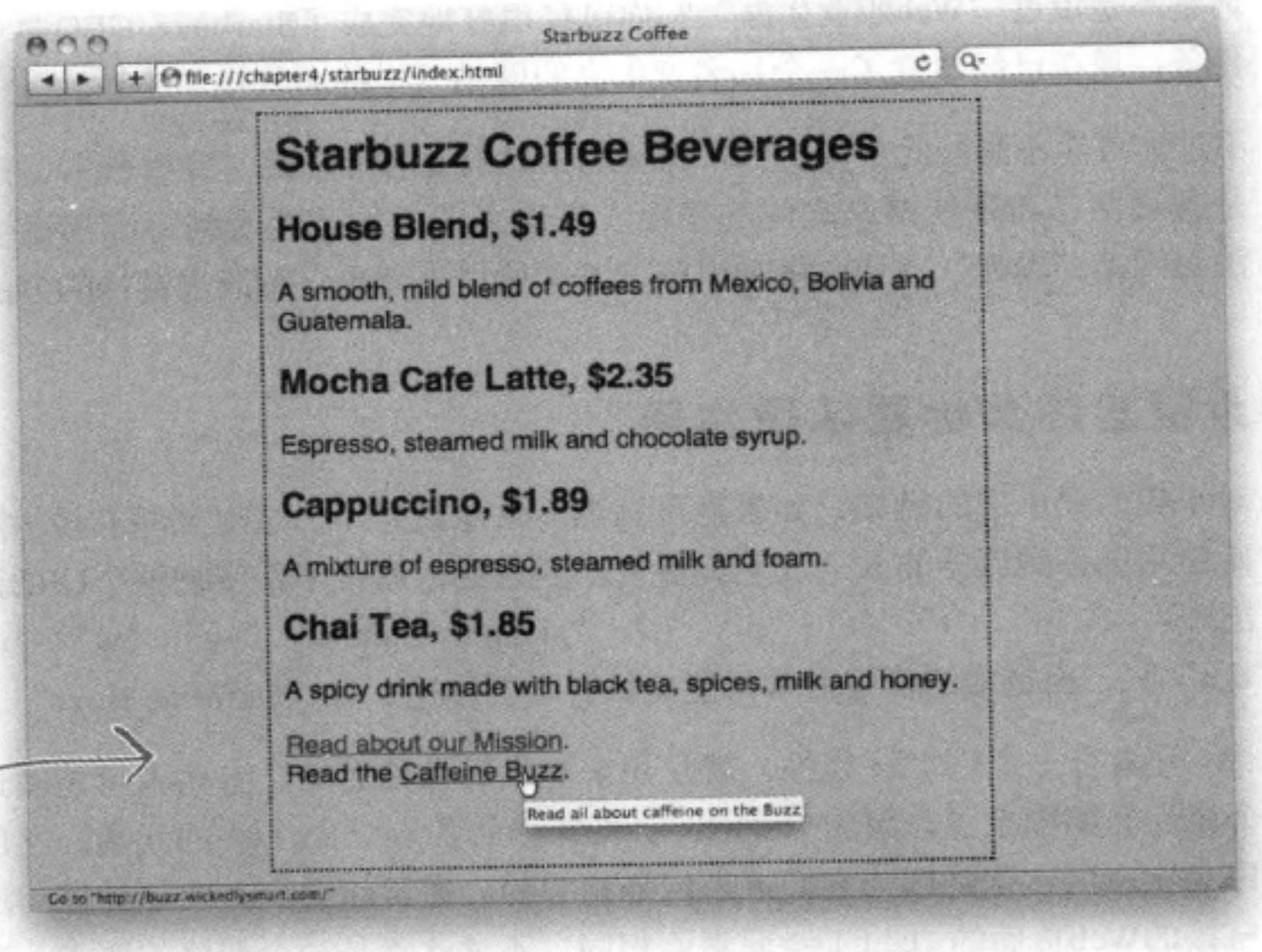
```
Read the <a href="http://wickedlysmart.com/buzz"
         title="Read all about caffeine on the Buzz">Caffeine Buzz</a>
```

title元素有一个值，这是所要链接的页面的文本描述。

既然有了一个title属性，下面就来看访问者如何使用这个属性。不同的浏览器对title有不同的用法，不过很多浏览器都会显示一个工具提示。在你的“index.html”文件中完成上面的修改，然后重新加载页面，看看你的浏览器会如何处理。

试一试标题……

对于大多数浏览器来说，鼠标移动到一个链接上时，标题会显示为一个工具提示。要记住，对于有视力障碍的人来说，浏览器可能会大声读出链接标题。



大多数浏览器中，标题会显示为一个工具提示。把你的鼠标移到链接上，停留一秒，就可以看到这个工具提示。

完善链接的Head First指南

进一步完善链接时要记住以下技巧：

- 保证链接标签很简洁。不要把整个句子或大段文字放在链接里。一般来讲，要保证只有几个单词。可以在title属性中提供额外的信息。
- 保证链接标签是有意义的。不要使用类似“单击这里”或“这一页”之类的链接标签。用户往往会首先粗略扫一眼页面，看看有没有链接，然后才会通读页面。所以，通过提供有意义的链接，可以改善页面的可用性。可以这样测试一下你的页面，只读页面上的链接，能理解吗？或者是不是还需要读它们周围的文字？
- 不要把链接放在一起，用户将很难区分放在一起的链接。

Exercise

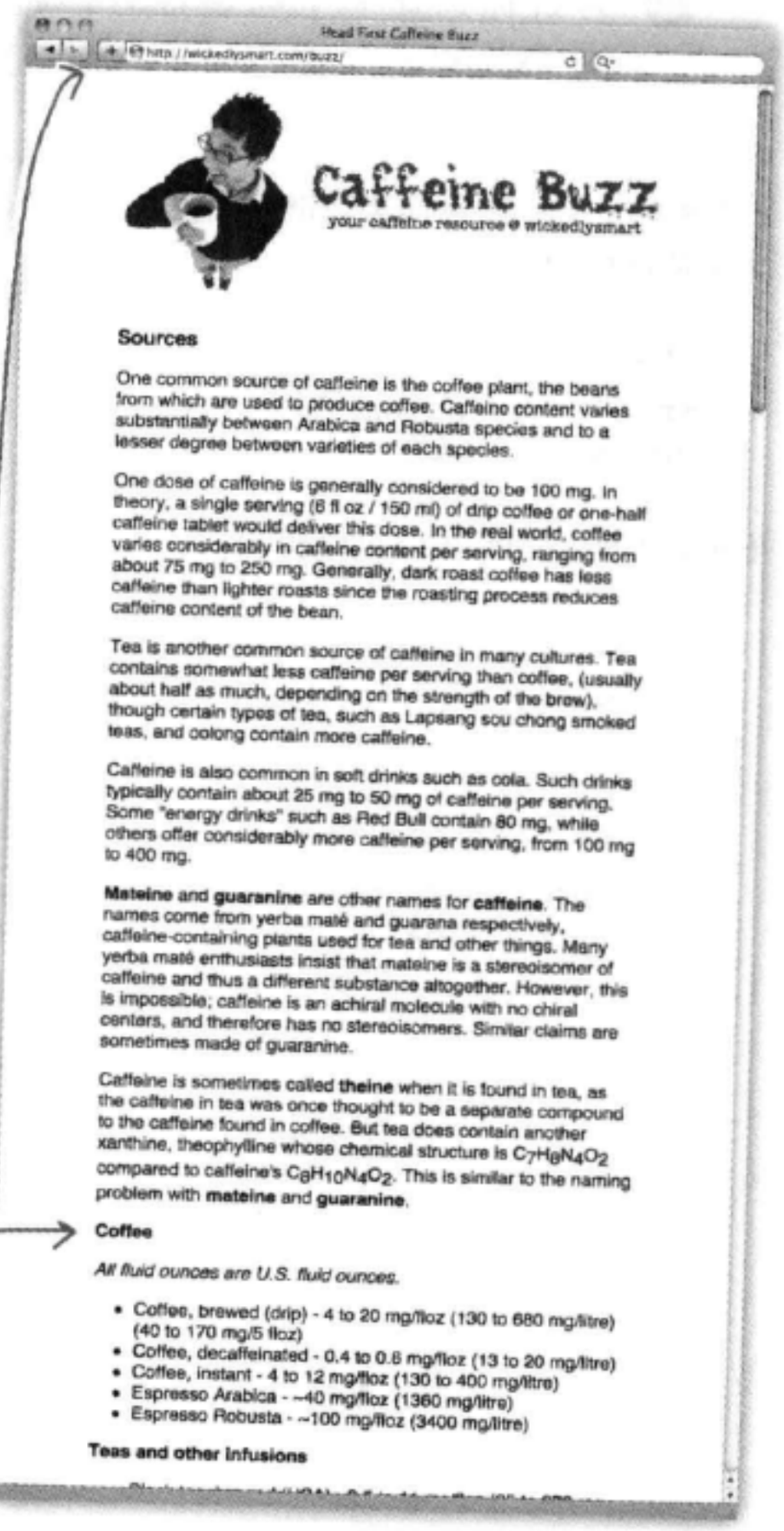
打开你的Starbuzz“index.html”文件，为“mission.html”链接增加一个标题，设置为“Read more about Starbuzz Coffee’s important mission”。注意这个宗旨页面链接的标签还不够简洁。可以把这个链接的标签简写为“our Mission”。对照本章最后给出的答案，测试你做的修改。

链接到一个页面

到目前为止，只要链接到另一个页面，浏览器就会加载整个页面，并从头开始显示。

不过CEO希望你能直接链接到页面中的某个特定位置：Coffee部分。

听上去不太可能？别担心，这可是Head First，我们有办法。怎么做呢？嗯，我们还没有告诉你关于<a>元素的全部呢。实际上<a>元素可以带一个id属性，允许你直接访问页面中的一个特定点。



使用id属性为<a>创建目标

我们还没有介绍过id属性，这是一个重要的属性，它有一些特殊的特性，本书后面还会更详细地讨论id的其他特殊特性。不过现在可以认为它是一种唯一标识元素的方法。带id的元素有一个特殊特性：你可以直接链接这些元素。下面来看如何使用id属性在页面中为<a>创建目标。

1. 找到页面中你希望创建锚点的位置。这可以是页面上的任何文本，不过通常是标题。
2. 为目标选择一个标识符名，如“coffee”或“summary”或“bio”，并在元素的开始标记中插入一个id属性。

下面来试一试。假设你希望链接Starbuzz页面上的Chai Tea（香辣奶茶），现在是这样的：

这是“index.html”中关于Chai Tea标题和描述的片段。

```
<h2>Chai Tea, $1.85</h2>
<p>A spicy drink made with black tea, spices, milk
and honey.</p>
```

按照前面所说的两步，可以得到：

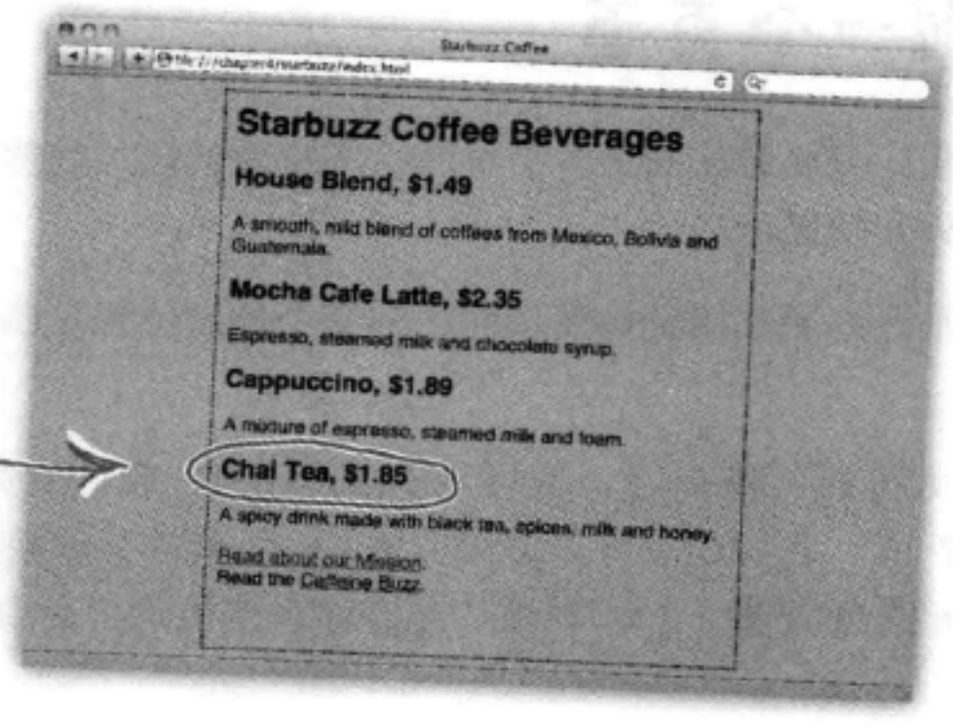
为标题的开始标记增加id。

为这个目标指定标识符“chai”。

id必须是唯一的，这很重要。也就是说，“chai” id必须是这个页面中唯一的“chai” id！

```
<h2 id="chai">Chai Tea, $1.85</h2>
<p>A spicy drink made with black tea, spices, milk and
honey.</p>
```

通过指定id，你建立了一个目标，指向“index.html”页面中的Chai Tea标题。



如何用id链接到元素

你已经知道如何使用相对链接或URL链接到页面。不论是相对链接还是URL，要链接到页面中的一个特定目标，只需在链接最后加一个#，再加上目标标识符。所以，如果你想从任何Starbuzz Coffee Web页面链接到“chai”目标，可以这样写<a>元素链接：

```
<a href="index.html#chai">See Chai Tea</a>
```

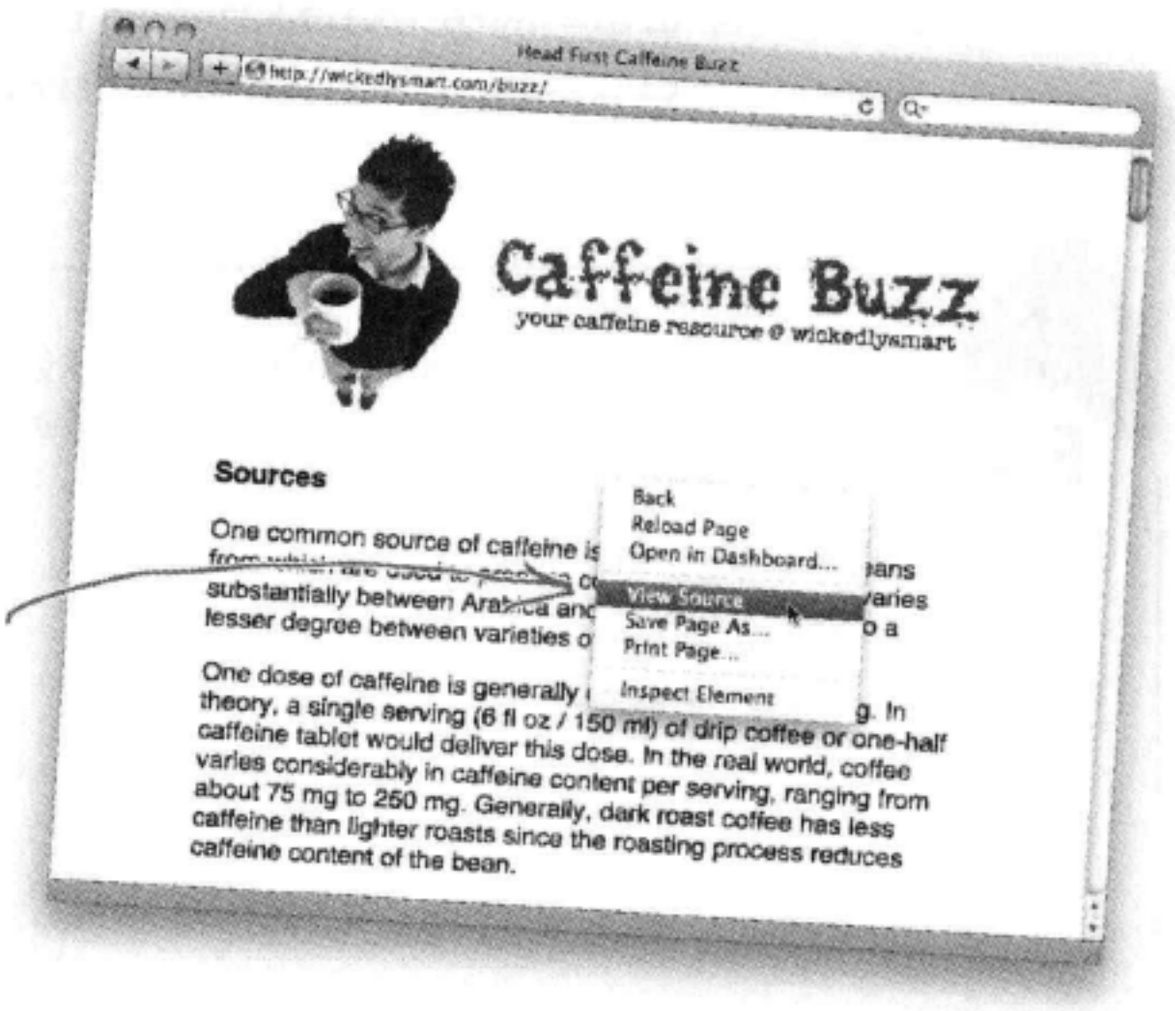
遗憾的是，链接到Chai Tea并不会给人留下太深印象，因为整个页面很小，所以可以在浏览器中完整地显示。下面我们要链接到http://wickedlysmart.com/buzz的Coffee部分。你需要做到：

特定目标的主要好处是链接到长文件中的某个位置，使得访问者不必向下滚动文件来查找所需的部分。

1. 明确Coffee标题的id。
2. 修改 Starbuzz Coffee “index.html”文件中现有的<a>元素，指向目标标题。
3. 重新加载“index.html”页面，然后测试链接。

查找目标标题

要找到目标标题，需要查看wickedlysmart.com/buzz页面和这个页面的HTML。怎么看呢？几乎所有浏览器都有一个“查看源代码”（View Source）选项。所以，可以访问这个页面，完全加载后，选择“查看源代码”选项，你就会看到这个页面的标记。



大多数浏览器中，可以用鼠标右键单击选择“查看源代码”（View Source）。

既然得到了HTML……

向下滚动，直到看到Coffee部分；如下所示：

这是Caffeine Buzz页面的一个小片段。

这里是Coffee部分。可以看到它的标题和下一段的开头。

哈，这里是目标标题，名为“Coffee”。

```
...
This is similar to the naming problem
with <b>mateine</b> and <b>guaranine</b>.
</p>

<h3 id="Coffee">Coffee</h3>
<p>
<i>All fluid ounces are U.S. fluid ounces.</i>
</p>
```

调整“index.html”中的链接

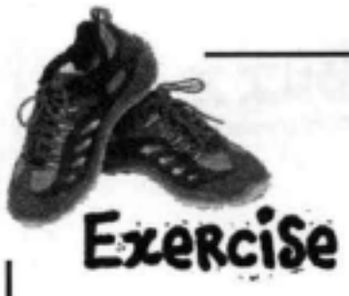
现在要做的就是调整指向Caffeine Buzz的链接，在后面增加目标锚名，如下所示：

这是Starbuzz “index.html”文件的一个片段。

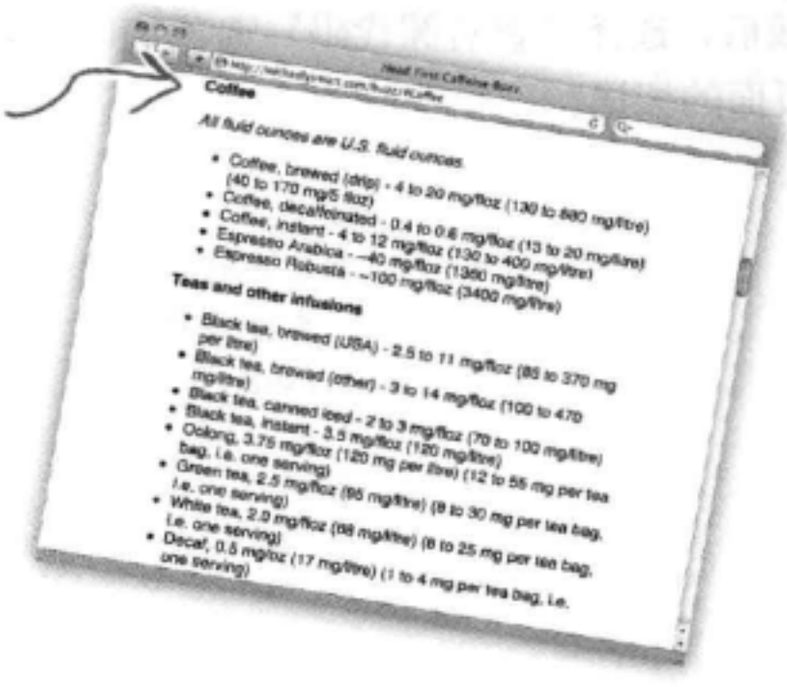
wickedlysmart.com/buzz的默认文件是index.html。所以我们把它增加到URL，这样才能结合使用目标id。

在href中增加#以及目标id。

```
Read the <a href="http://wickedlysmart.com/buzz/index.html#Coffee"
         title="Read all about caffeine on the Buzz">Caffeine Buzz</a>
```



对Starbuzz “index.html”文件完成以上修改。重新加载页面，单击“Caffeine Buzz”链接。会把你直接带到Caffeine Buzz首页的Coffee部分。



there are no Dumb Questions

问：如果一个元素里有两个属性，属性的先后顺序重要吗？例如，title属性是不是总在href后面？

答：所有元素中，属性的顺序都不重要（如果重要，那我们可就太头疼了）。所以，你可以用你喜欢的任何顺序。

问：如果元素不是<a>，那么怎么为这样一个元素创建一个工具提示呢？

答：可以为任何元素增加title属性，所以如果希望某个元素上有一个工具提示，比如一个标题，则可以在<h1>开始标记中增加一个title属性，就像对<a>一样。有些元素不只是使用title属性作为工具提示，可能还有其他用途，不过工具提示是最常见的用法。

问：可以为任意元素增加id属性吗？

答：可以。例如，如果为元素增加一个id，那么还可以链接到段落中间。通常不太需要这么做，不过如果你愿意也无不可。

问：能不能链接到一个链接，也就是在目标中为一个<a>元素增加id属性？

答：可以！

问：我注意到，在id名中，你使用的是“chai”，所有字母都是小写，而Caffeine Buzz用的是“Coffee”，有一个大写的“C”。这有影响吗？

答：在id属性中可以使用大写和小写字符的任意组合。只是要确保保持一致，在href和目标id中要用相同的大小写字母（正是因为这个原因，每次都全用小写字母会更容易一些）。如果没有做到一致，那就别指望你的链接在所有浏览器上都能正常工作。关于你选择的id名，最重要的一点是它在你的页面中必须唯一。

问：我可以在文档中放置一个指向相同文档中某个目标的链接吗？

答：当然可以。实际上，通常会在页面最上面（比如，在页面的顶部）定义一个目标“top”，并在页面最下面定义一个目标“Back to top”。在长文档中，往往会有整个页面的一个目录，这也很常见。例如，要链接到同一个页面上的“top”目标，可以写为Back to top。

问：为什么需要为Buzz URL增加“/index.html”来创建指向目标标题的链接呢？为什么不能直接写为：http://wickedlysmart.com/buzz#Coffee？

答：不行，这不一定能正常工作，因为浏览器会帮你在URL末尾加斜线，这有可能取代id引用。不过，你可以这样写：http://wickedlysmart.com/buzz/#Coffee，类似于使用“index.html”创建的链接，这会得到同样的结果。如果你不知道默认文件是否是“index.html”，这就会很方便。

问：如果一个Web页面没有提供一个目标，但我还需要链接到页面的某个特定部分，该怎么做呢？

答：这是办不到的。如果没有目标（换句话说，没有带id的元素），就无法指示浏览器前往Web页面中的一个特定位置。你可能需要联系页面作者，让他们增加这样一个目标（甚至可以更棒，你还可以告诉他们怎么加目标）。

问：能不能有一个类似“Jedi Mindtrick”的目标id，或者是不是id只能有一个单词？

答：要在大多数浏览器上得到一致的结果，id一定要用一个字母开头（A~Z或a~z），后面可以是任意字母、数字、横线、下划线、冒号或点号。所以尽管不能加空格，不能有类似“Jedi Mindtrick”的名字，但这个限制不算太苛刻，因为你还可以使用“Jedi-Mindtrick”、“Jedi_Mindtrick”、“JediMindtrick”之类的名字。

问：怎么告诉别人可以链接到哪些目标呢？

答：这没有既定的方法，实际上，“查看源代码”仍然是发现链接目标最古老也是最好的方法。

问：总得使用文字作为<a>元素的内容吗？

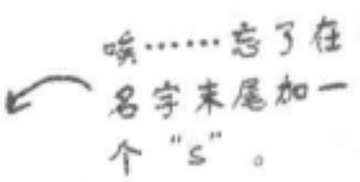
答：不是的。<a>元素能够由文字和图像（内联内容）创建链接，最近（在HTML5中）则进一步更新，现在还可以从块元素（如<p>和<blockquote>）创建链接！所以可以用<a>由各种不同内容创建链接。

五分钟谜案之谜底

相对与绝对

RadWebDesign的演示为什么会搞砸了？嗯，因为他们使用URL来指定href而不是相对链接，必须把每一个链接从http://www.planetrobots.com改为http://www.robotsrus.com。

能不容易出错吗？凌晨3点，有人呵欠连天，不小心输入了http://www.robotsru.com（命运真是捉弄人，CEO在演示中单击的恰恰就是这个链接）。



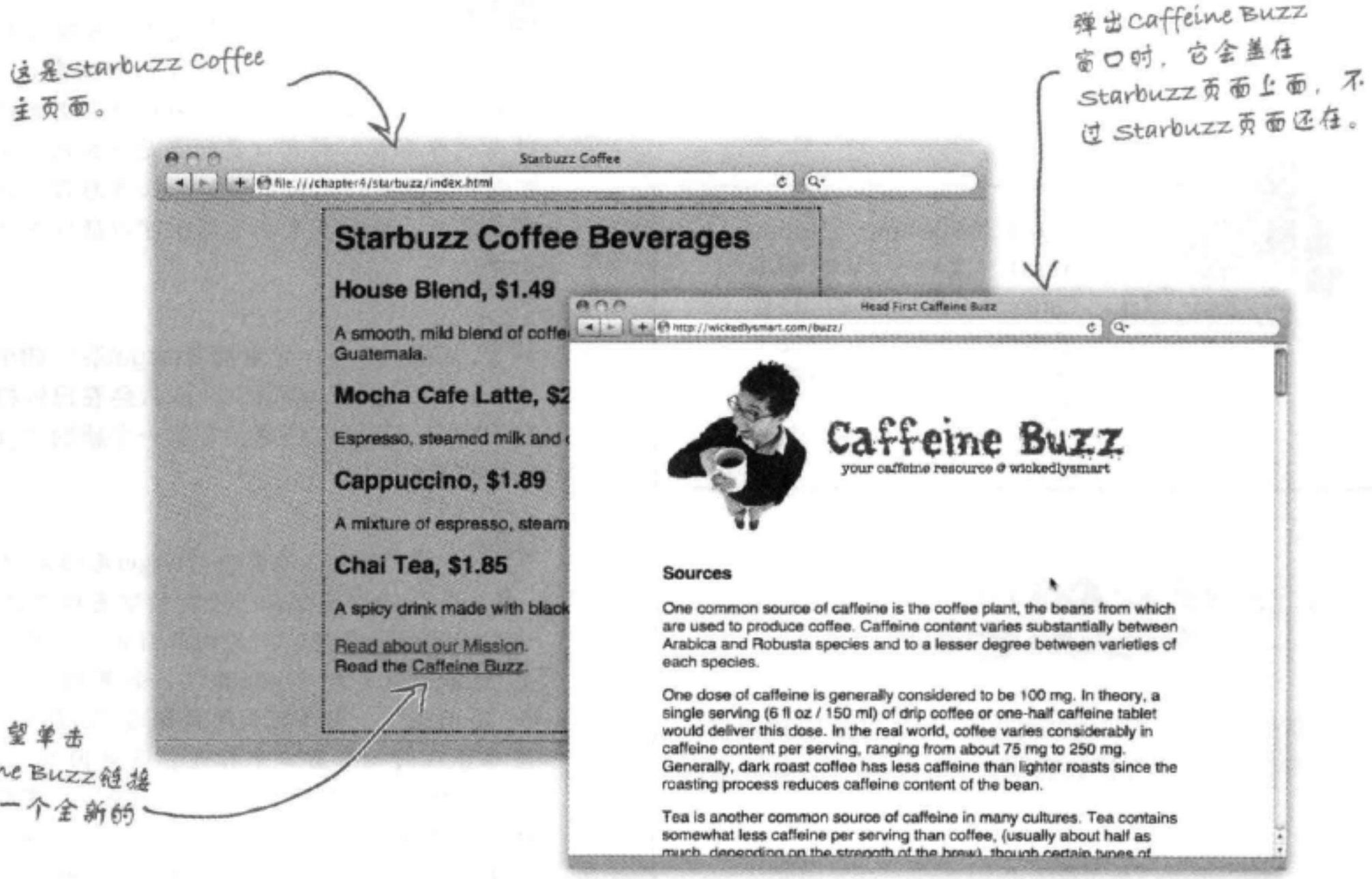
CorrectWebDesign则不同，他们使用相对链接来建立所有内部链接。例如，从公司宗旨声明到产品页面的链接为<a href="../products.html">，不论网站叫做PlanetRobots还是RobotsRUs，这个链接都能正常工作。所以，CorrectWebDesign所要做的只是在一些页面上更新公司名字。

所以RadWebDesign灰头土脸地离开了演示会场，与之相反，CorrectWebDesign离开会场时则是眉开眼笑，又接了好几单业务。不过，故事还没有结束。在发布会之后RadWebDesign拜访了一些小咖啡馆和书店，决定不再闭门造车，还选了一本关于HTML和CSS的书攻读。后来怎么样了？请关注后面几章的“蛮力与样式迷案”。

你做到了，链接到了Buzz网站，真不赖……
我知道我总在要求修改，不过说真的，这是最后一次了。能不能在我单击时让Buzz网站在另一个窗口中显示？我不希望我的Starbuzz页面消失。

链接到一个新窗口

Starbuzz CEO又有一个新的要求（网站总是不断有新需求）。他希望这样：你单击Starbuzz Coffee页面中的“Caffeine Buzz”链接时，Starbuzz Coffee页面并不消失。而是出现一个全新的窗口，在这个窗口中打开Caffeine Buzz页面，如下所示：



使用target打开新窗口

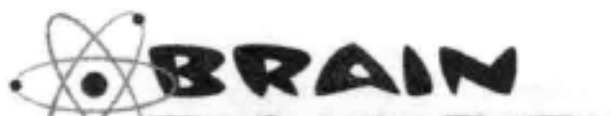
要在一个新窗口中打开一个页面，需要告诉浏览器你要打开的窗口名。如果没有告诉浏览器要使用哪个特定的窗口，浏览器就会在同一个窗口中打开这个页面。可以为<a>元素增加一个target属性，告诉浏览器使用一个不同的窗口。target属性值会告诉浏览器页面的“目标窗口”。如果使用“_blank”作为目标，浏览器就总是打开一个新窗口显示页面。下面来仔细看一下：

```
<a target="_blank" href="http://wickedlysmart.com/buzz"
   title="Read all about caffeine on the Buzz">Caffeine Buzz</a>
```

target属性告诉浏览器在哪里打开href属性中链接指示的Web页面。如果没有target，浏览器就会在同一个窗口中打开这个链接。如果目标为“_blank”，浏览器就会在一个新窗口中打开链接。

Exercise

打开你的Starbuzz “index.html”文件。为链接到Caffeine Buzz页面的<a>标记增加target属性。现在来试一试，打开一个新窗口了吗？



BRAIN POWER

你能想到使用target属性在一个新窗口打开页面有什么优点和缺点吗？

there are no Dumb Questions

问：我得到一个新的标签页，而不是新窗口。是不是哪里做错了？

答：不，你没做错。大多数浏览器现在有都有一个默认设置，会在一个标签页中打开新窗口，而不是一个全新的浏览器窗口，因为用户好像更喜欢这样。不过新的标签页和新窗口实际上是一样的，只是标签页可以共享原窗口的窗口边框。如果想强制打开一个全新的窗口，则大多数浏览器上可以通过首选项设置来实现。

问：如果多个<a>元素都有target呢？如果已经打开了一个“_blank”新窗口，那么会在已经打开的这个窗口中打开页面？还是会在另一个新的“_blank”窗口中打开？

答：如果所有<a>元素中的target都指定为“_blank”，那么每个链接都会在一个新的空窗口中打开。不过，这个问题问得很好，因为由此引出一个重点：不一定要把target指定为“_blank”。如果指定另一个名字，如“coffee”，那么有相同目标名“coffee”的所有链接都会在同一个窗口中打开。这是因为，为target指定一个特定的名字时，如“coffee”，实际上就是在对显示链接页面的新窗口命名。“_blank”则是一种特殊情况，告诉浏览器总是使用一个新窗口。

target属性闪亮登场

本周访谈：
使用target很不好吗？

Head First：你好，Target！很高兴你能接受我们的采访。

Target：我也很高兴。你们还对我这么感兴趣，真让我感动。

Head First：怎么这么说呢？

Target：嗯，说实话，我没有以前那么流行了。

Head First：为什么你会这么想？

Target：我觉得，这是因为用户现在都想自己来控制什么时候打开窗口。它们不希望总是意外地弹出新窗口。

Head First：噢，这确实很烦人，我们听到过很多人曾抱怨，因为屏幕上有太多窗口，他们简直没有办法找到原来的页面。

Target：不过，去掉这些窗口并不难……只需要单击那个小关闭按钮。这有什么难的！

Head First：没错，不过用户不知道已经打开一个新窗口，他们可能就会糊涂。有时新窗口会完全覆盖原来的窗口，这就很难说清楚到底出了什么问题。

Target：嗯，浏览器在这方面越来越棒了。

Head First：哦，是吗？

Target：浏览器通常会在一个新的标签页中打开新页面，不过是在同一个浏览器窗口中，而不在一个全新的窗口中打开。

Head First：啊，这确实会有帮助，因为看到打开一个新标签页就不那么容易让人糊涂了，用户希望什么时候访问都可以。与打开一个新窗口不同，这样不会让人那么困惑。

Head First：不过，这对使用屏幕阅读器的人有帮助吗？

Target：你是指有视力障碍的人使用的浏览器吧？

Head First：没错。打开一个新窗口时，有些屏幕阅读器会发出一个声音，不过另外一些会完全忽略新窗口，也可能直接跳到新窗口。不管怎样，对于无法亲眼看到的人来说，都会很迷惑。我不知道他们如何处理标签页。

Target：[唉]是啊，我们还没办法提供好的工具来满足每一个人的需要，特别是有视力障碍的人。也就是说，我们需要能够把用户带到我们网站以外的其他页面，很多网站的做法都是打开另一个窗口（或者如果浏览器支持，可以打开另一个标签页）。

Head First：对，我们需要你，不过我们还需要做得更好，不让用户困惑。

Target：我希望Web标准和浏览器小组能让情况好起来。

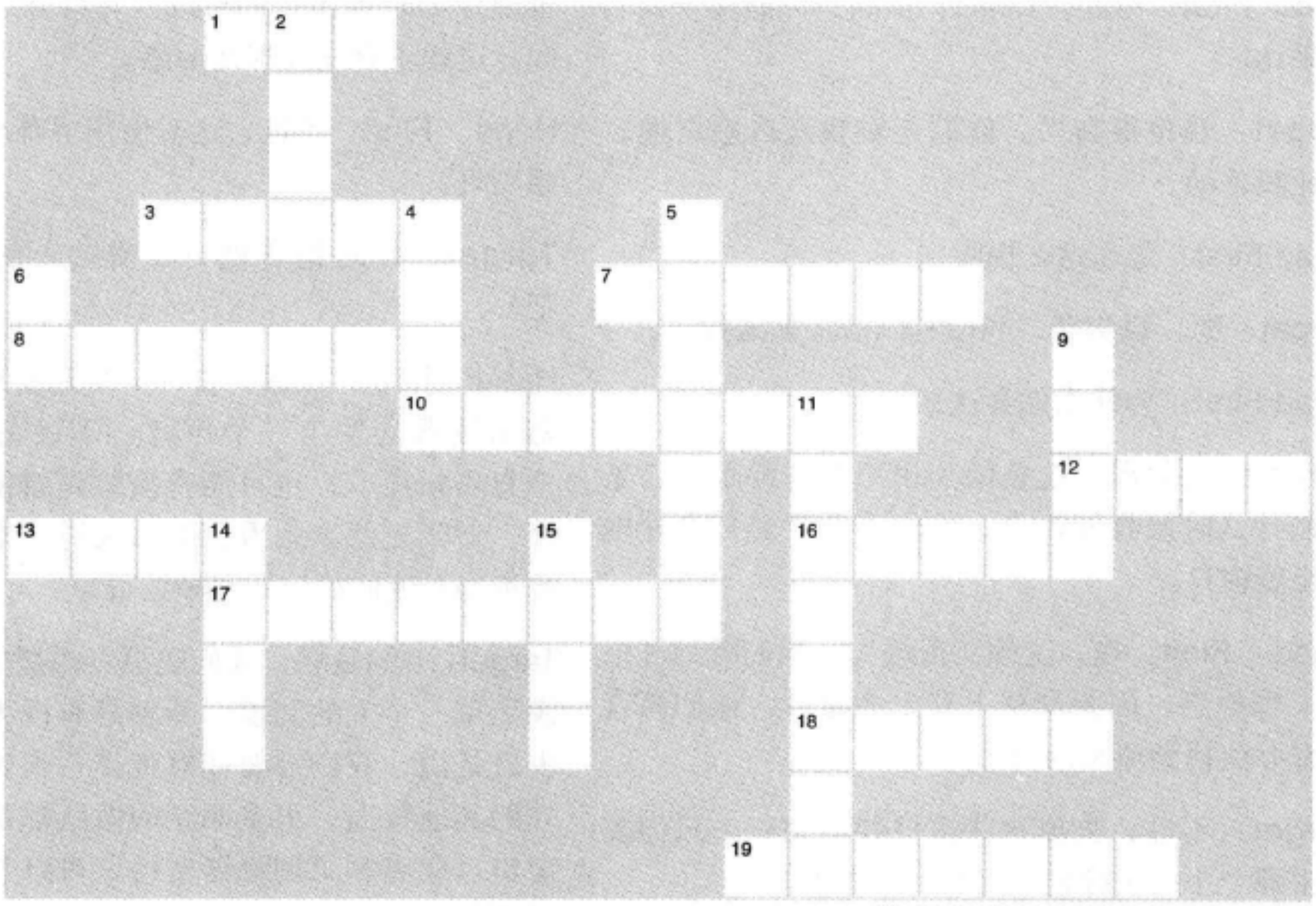
Head First：我想现在我们只需要记住在适当的情况下用你，不过还要考虑到那些可能有视力障碍的人，所以不能过度地用你。

Target：说得对。你让我松了一口气，非常感谢你帮我把这句话说出来！

Head First：随时愿意效劳，Target！

HTML填字游戏

来锻炼锻炼你的左脑。



横向

1. 一个资源的Web地址。
3. 一个Mac FTP应用。
7. Web上的唯一名。
8. 请求一个目录时得到的文件。
10. 你从Web镇寄回的东西。
12. 网站的顶级目录。
13. 这一章之前一直使用的协议。
16. 人们可以扫描这些而不是通读文本。
17. 从根开始的路径。
18. 控制域名。
19. 请求一个目录时得到的文件。

纵向

2. 网站的顶级目录。
4. 请求/响应协议。
5. 保证链接标签 _____。
6. 使用这个属性使一个元素成为目标。
9. Earl出售这些。
11. 链接到相同服务器上的页面时要使用这种链接。
14. 拼读URL的错误方式。
15. 提供大量信息的咖啡因网站。

BULLET POINTS

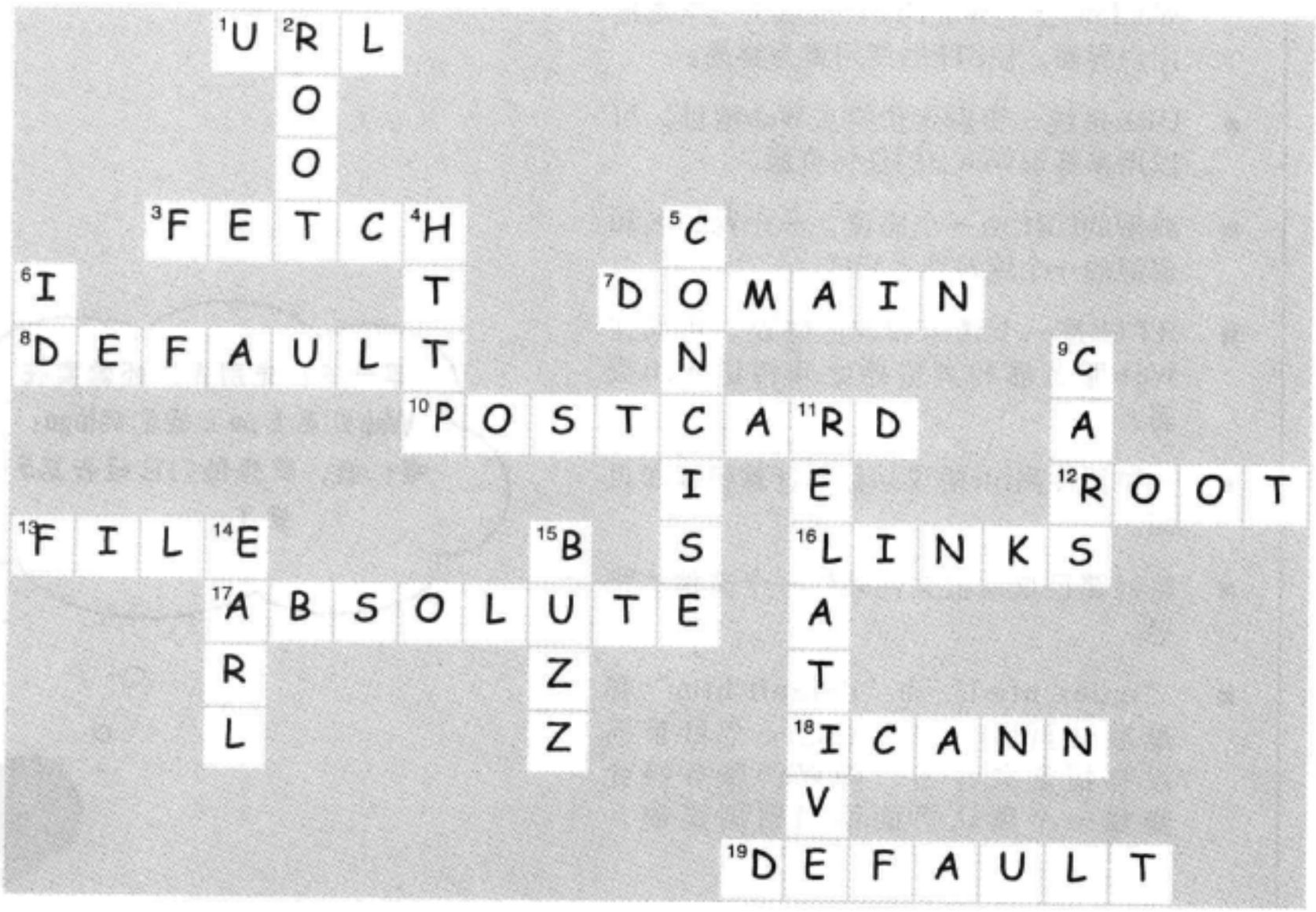
- 要把网站发布到Web上，通常最好的办法就是找一家托管公司来托管你的Web页面。
- 域名是一个唯一的名字，如amazon.com或starbuzzcoffee.com，用来唯一标识网站。
- 托管公司可能会为你的域创建一个或多个Web服务器。服务器通常命名为“www”。
- 文件传输协议（File Transfer Protocol，FTP）是向服务器传输web页面和内容的常用方法。
- FTP应用（如Mac上的Fetch for Mac或Windows上的WS_FTP）提供了一个图形用户界面，使FTP的使用更为容易。
- URL是统一资源定位符或Web地址，可以用来标识Web上的任何资源。
- 典型的URL由一个协议、一个网站名和资源的一个绝对地址组成。
- HTTP是一个请求和响应协议，用来在Web服务器和浏览器之间传送Web页面。
- 浏览器使用file协议从你的计算机读取页面。
- 绝对路径是从根文件夹到一个文件的路径。
- “index.html”和“default.htm”都是默认页面。如果指定一个目录而没有指定文件名，则Web服务器会查找一个默认页面返回到浏览器。
- <a>元素的href属性中可以使用相对路径或URL来链接其他Web页面。对于你的网站中的其他页面，最好使用相对路径，对外部链接才使用URL。
- 可以用id属性在页面中创建一个目标。使用#后面加一个目标id，可以链接到页面中的那个位置。
- 为了便于访问，可以在<a>元素中使用title属性提供链接的一个描述。
- 使用target属性在另一个浏览器窗口中打开链接。不要忘了，对于使用各种不同设备和浏览器的用户，target属性可能会有问题。

Sharpen your pencil Solution

让你久等了。现在该拿你的新URL试一试了。开始之前，先完成下面的填空，然后输入这个URL（如果还没有输入的话）。如果遇到问题，则现在可以请托管公司帮忙一起来解决问题。如果还没有找托管公司，你可以先填上www.starbuzzcoffee.com，再在浏览器中输入这个URL。

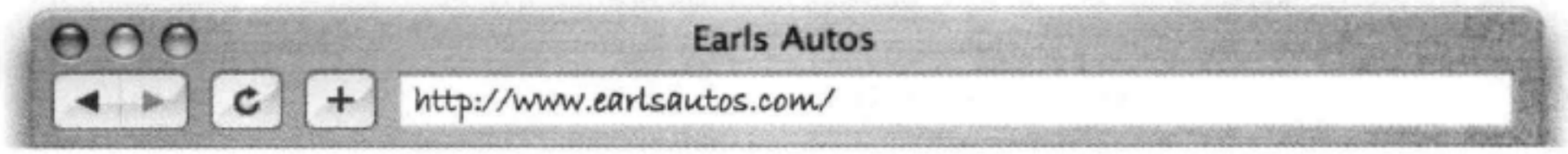
http **协议** :// www.starbuzzcoffee.com **网站名** /index.html **绝对路径**

这里写上你的网站名。



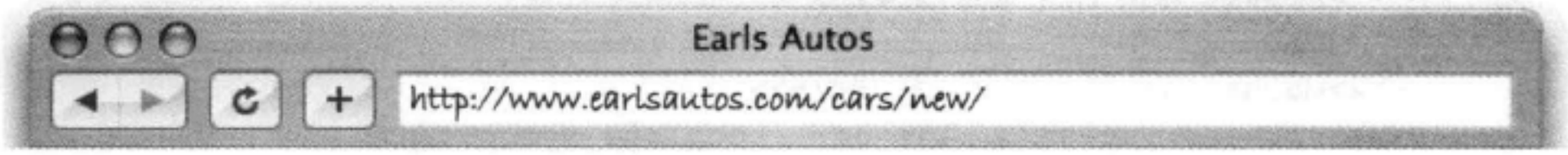
Earl需要你帮他确定URL

A

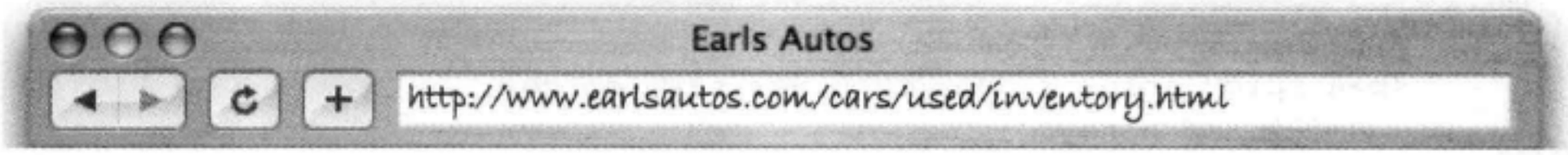


B

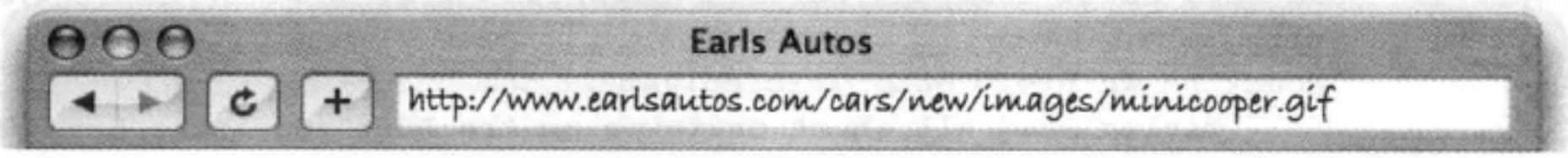
C



D



E



Exercise Solution

为“mission.html”链接增加一个标题，设置为“Read more about Starbuzz Coffee's important mission”。注意这个宗旨页面链接的标签还不够简洁。可以把这个链接的标签简写为“our Mission”。下面是我们的答案，你对这些修改测试过吗？

```
<html>
    <head>
        <title>Starbuzz Coffee</title>
        <style type="text/css">
            body {
                background-color: #d2b48c;
                margin-left: 20%;
                margin-right: 20%;
                border: 1px dotted gray;
                padding: 10px 10px 10px 10px;
                font-family: sans-serif;
            }
        </style>
    </head>
    <body>
        <h1>Starbuzz Coffee Beverages</h1>
        <h2>House Blend, $1.49</h2>
        <p>A smooth, mild blend of coffees from Mexico,
           Bolivia and Guatemala.</p>

        <h2>Mocha Cafe Latte, $2.35</h2>
        <p>Espresso, steamed milk and chocolate syrup.</p>

        <h2>Cappuccino, $1.89</h2>
        <p>A mixture of espresso, steamed milk and foam.</p>

        <h2>Chai Tea, $1.85</h2>
        <p>A spicy drink made with black tea, spices,
           milk and honey.
        </p>
        <p>
            Read about <a href="mission.html"
        title="Read more about Starbuzz Coffee's important mission">our Mission</a>.
            <br>
            Read the <a href="http://wickedlysmart.com/buzz"
                title="Read all about caffeine on the Buzz">Caffeine Buzz</a>.
        </p>
    </body>
</html>
```

为宗旨页面链接增加一个title属性。

把“Read about”移出<a>元素。

5 为你的页面增加图像

认识媒体

笑一笑，说“茄子”。说真的，应该是笑一笑，说“gif”、“jpg”或“png”。你为Web“冲印照片”时，就会用到它们。这一章中，我们将学习如何为你的页面添加第一个媒体类型：图像。你是不是拍摄了一些数码照片想要发到网上？没问题。是不是需要在页面上加一个logo？小事一桩。不过在深入学习这些内容之前，是不是需要正式介绍一下<img>元素？真是抱歉，我们不是有意疏忽，只是一直没有找到“合适的机会”。作为补偿，下面用一整章专门讨论<img>。学完这一章，你就会全面地了解使用<img>元素及其属性的所有细节了。你还会亲眼看到这个小元素是怎样让浏览器做大量额外的工作来获取和显示图像的。

浏览器如何处理图像

浏览器对<img>元素的处理与其他元素稍有不同。以<h1>或<p>元素为例，浏览器在页面上看到这些标记时，只需要把它们显示出来就可以了，很简单。不过，浏览器看到一个<img>元素时，会做不同的处理：浏览器在页面中显示图像之前，必须先获取这个图像。

要了解这是如何做到的，最好的办法就是通过一个例子来说明。下面再来简单地回顾一下Head First休闲室的清凉饮料页面，其中有4个<img>元素：

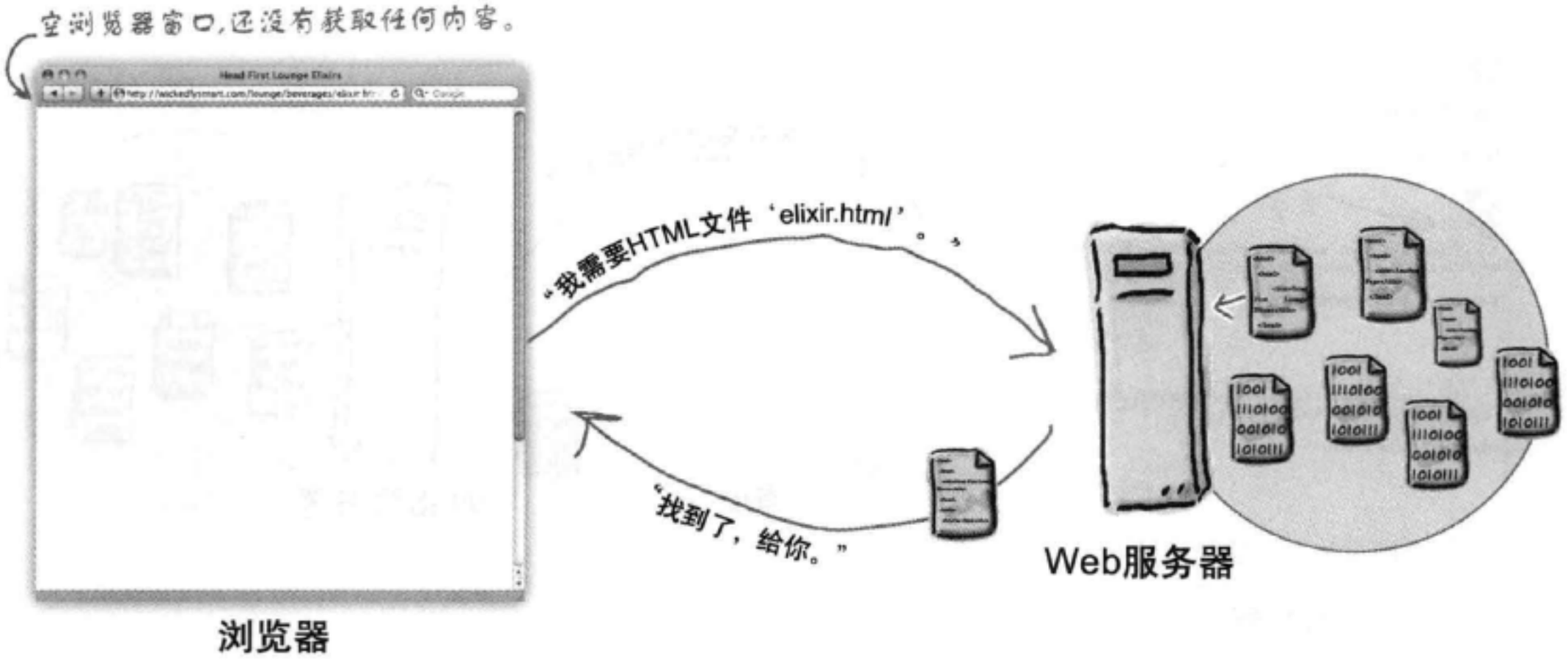

```
<html>
  <head>
    <title>Head First Lounge Elixirs</title>
  </head>
  <body>
    <h1>Our Elixirs</h1>
    <h2>Green Tea Cooler</h2>
    <p>
      <img src="../images/green.jpg">
      Chock full of vitamins and minerals, this elixir combines
      the healthful benefits of green tea with a twist of chamomile
      blossoms and ginger root.
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p>
      <img src="../images/lightblue.jpg">
      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">
      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">
      Wake up to the flavors of cranberry and hibiscus in
      this vitamin C rich elixir.
    </p>
    <p>
      <a href="../lounge.html">Back to the Lounge</a>
    </p>
  </body>
</html>
```

这个HTML中有4个图像。

现在我们来幕后看一看，一步一步了解浏览器收到http://wickedlysmart.com/lounge/请求时如何获取和显示这个页面：

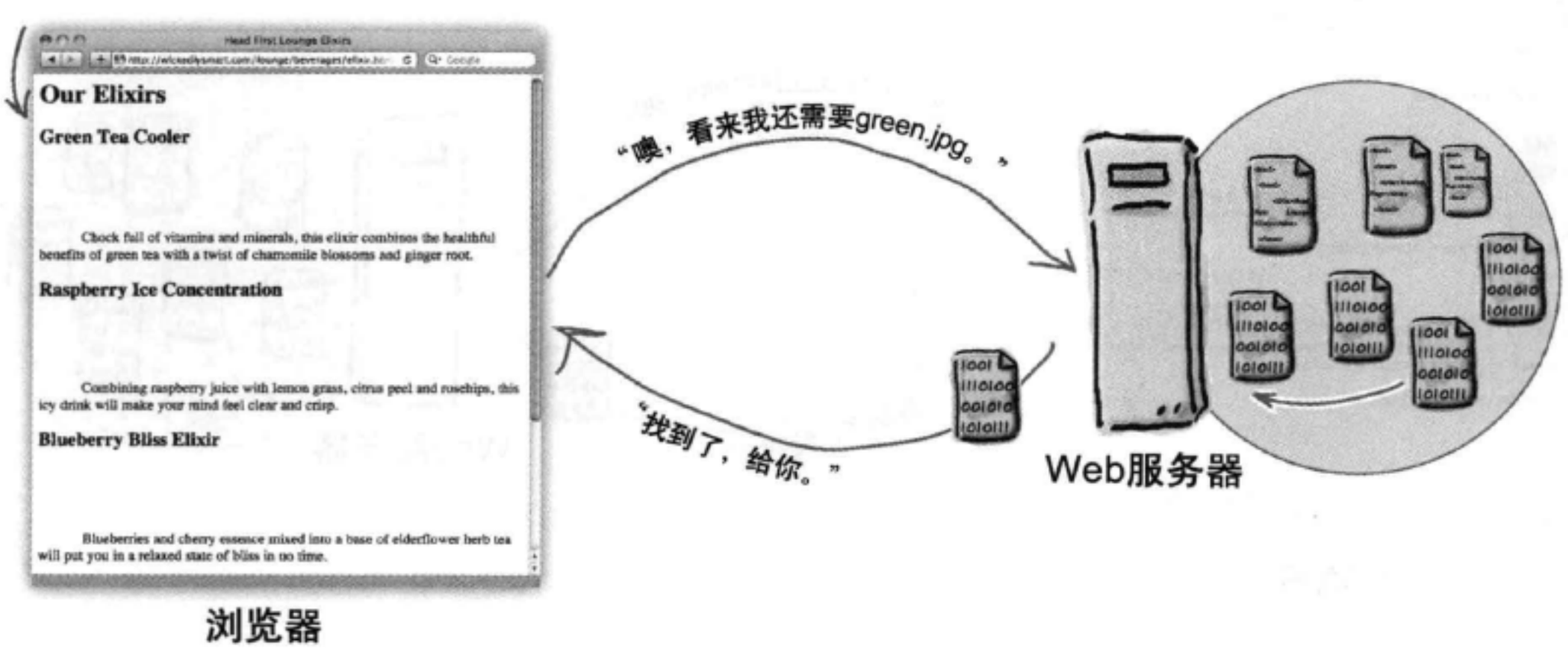
在幕后

① 首先，浏览器从服务器获取文件“elixir.html”。

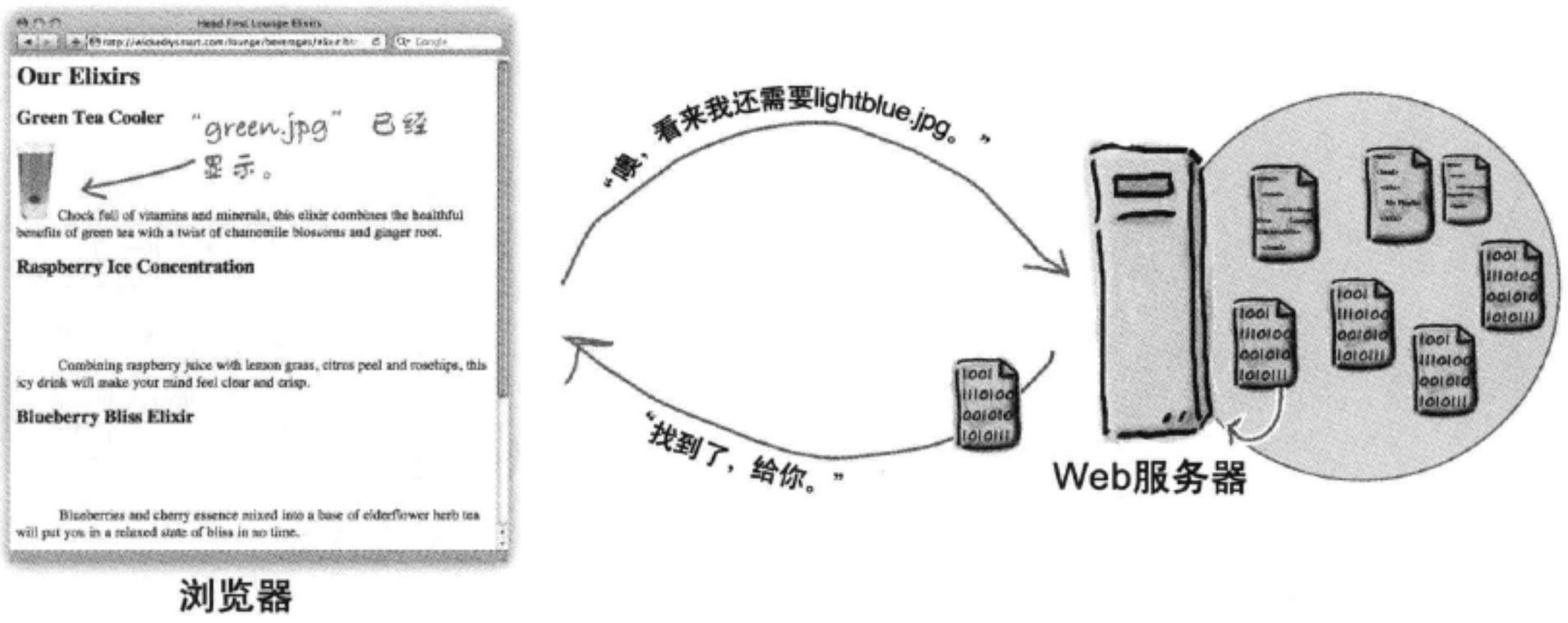


---

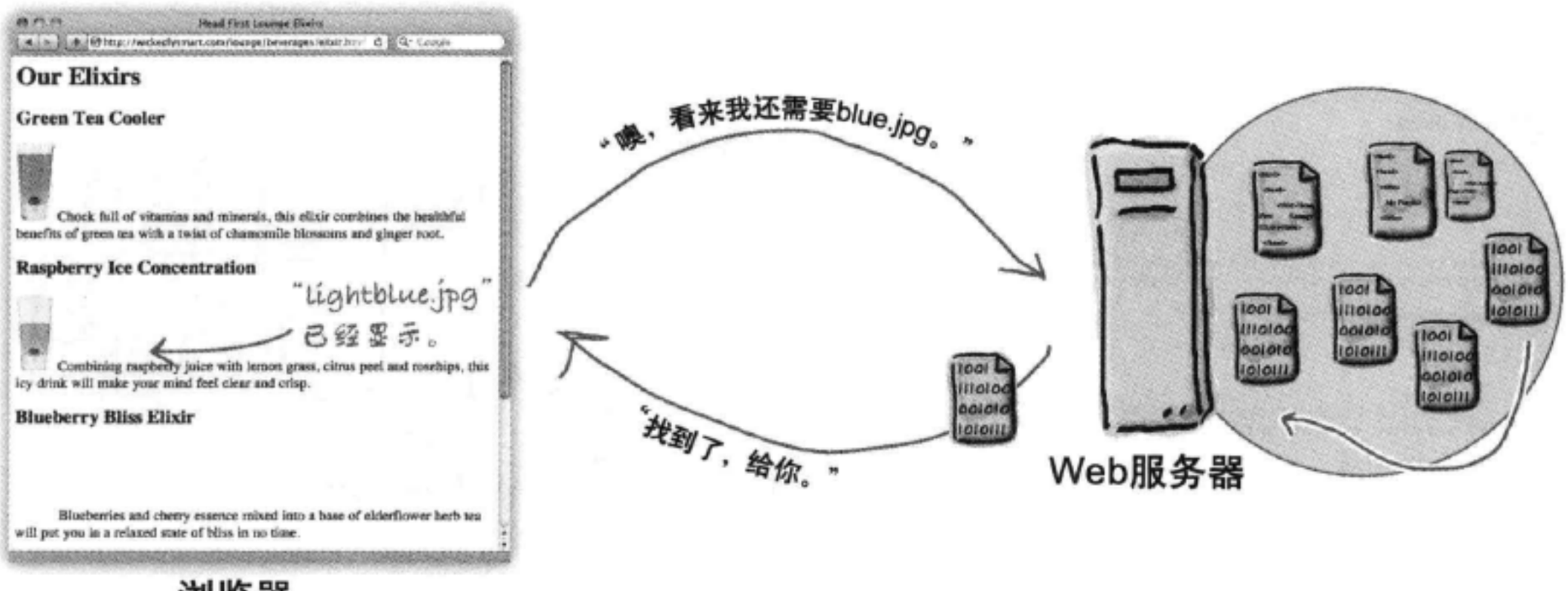
② 接下来，浏览器读取“elixir.html”文件，显示这个文件，发现其中有4个图像需要获取。所以它需要从Web服务器逐个得到这些图像，先从“green.jpg”开始。



③ 获取到“green.jpg”之后，浏览器显示这个图像，然后转向下一个图像：“lightblue.jpg”。



④ 现在浏览器已经获取了“lightblue.jpg”，所以它会显示这个图像，然后转向下一个图像：“blue.jpg”。对页面上的每一个图像都会重复这个过程。



图像是如何工作的

图像就是image，对吗？嗯，实际上，世界上图像格式实在太多了，这些图像格式都各有优缺点。幸运的是，Web上最常用的只是其中3种格式：JPEG、PNG和GIF。唯一比较困难的是要确定在什么情况下使用哪一种格式。

那么，JPEG、PNG和GIF有什么不同呢？

照片和复杂图像使用

JPEG

最适合连续色调图像，如照片。

可以表示包含多达1600万种不同颜色的图像。

这是一种“有损”格式，因为缩小文件大小时会丢掉图像的一些信息。

不支持透明度。

文件比较小，以便Web页面更高效地显示。

不支持动画。

单色图像、logo和几何图形使用

PNG或GIF

PNG最适合单色图像和线条构成的图像（如logo、剪贴画和图像中的小文本）。

PNG可以表示包含上百万种不同颜色的图像。PNG有3种：PNG-8、PNG-24和PNG-32，取决于你需要表示多少种颜色。

PNG会压缩文件来缩小文件大小，不过不会丢掉信息。所以这是一种“无损”格式。

允许将颜色设置为“透明”，使图像下面的东西可以显示出来。

与相应的JPEG文件相比，PNG文件更大一些，不过取决于使用的颜色数，可能比相应的GIF文件小，也可能更大。

类似于PNG，GIF最适合单色图像和线条构成的图像（如logo、剪贴画和图像中的小文本）。

GIF可以表示最多256种不同颜色的图像。

GIF也是一种“无损”格式。

GIF也支持透明度，不过只允许一种颜色设置为“透明”。

GIF文件往往比相应的JPEG文件大。

支持动画。

真正的图像格式请起立！

本周访谈: 混乱的图像格式

Head First：嘿，大家好。我想这可能是我们头一次同时介绍三位佳宾！

JPEG：你们好，GIF和PNG。

GIF：我不知道为什么要和另外这两个家伙一起接受采访。所有人都知道GIF是Web最早的图像格式。

JPEG：哈！除非你能很好地表示复杂图像，比如说照片，那时人们可能才会重新对你另眼相看吧，不过我实在不相信你用区区256种颜色能够做到。

Head First：PNG，快来帮忙啊？你太安静了，到现在还没说一句话呢……

PNG：嗯，第一名一般就不用说什么。我可以像JPEG那样表示复杂图像，还可以像GIF那样是无损的。所有优点都集于我一身。

Head First：无损的？

PNG：对，用一种无损格式存储图像时，你不会丢失图像中的任何信息或细节。

GIF：我也是！我也是无损的，你知道的。

Head First：嗯，为什么会有人需要无损格式呢？

JPEG：凡事总有取舍。有时你可能只想要一个很小的文件，可以快速下载，而且质量很不错。我们并不总需要最完美的质量。JPEG图像已经让人很满意了。

PNG：当然，当然，不过你看过那些线条、logo、小文本和单色图像没有？如果用JPEG格式，则它们看起来可不怎么样。

Head First：等一下，JPEG提出一个有意思的问题。我要问一下，GIF和PNG，你们的文件大吗？

PNG：我承认我的文件有时可能比较大，不过我提供了3种格式：PNG-8、PNG-24和PNG-32，你可以选择适当的文件大小。

GIF：对我来说，听上去很复杂，这得让你的用户记更多东西。

PNG：嗯，GIF，如果所有图像都只使用256种颜色，那该多好！不过这样不行。

GIF：嘿，对于线图、图形之类的图像，很容易用8位颜色表示，而且我在这方面确实很棒。

JPEG：哈，你上一次见到用GIF格式存储的照片是什么时候？人们已经发现的你短处了，GIF。

GIF：我提到过我可以透明了吗？你可以取走我的一部分，我下面的东西就能透出来。

PNG：在这方面，你可比不上我，GIF。我可以把任意种颜色设置为透明，而你仅限于一种颜色。

GIF：一种还是多种颜色，谁在意呢？一种就够了。

PNG：如果你希望图像里有无锯齿透明区，则只设置一种颜色透明就不行！

GIF：嗯？

PNG：是的，你要知道，因为我允许将多种颜色设置为透明，所以透明区的边缘是平滑的。

Head First：听上去是一个很不错的特性。你能做到吗，JPEG？

JPEG：不行，不过我不担心这一点，没有多少照片需要这么做。这是对logo来讲的。

PNG：嗬，我可发现Web上到处都在用我的透明性。

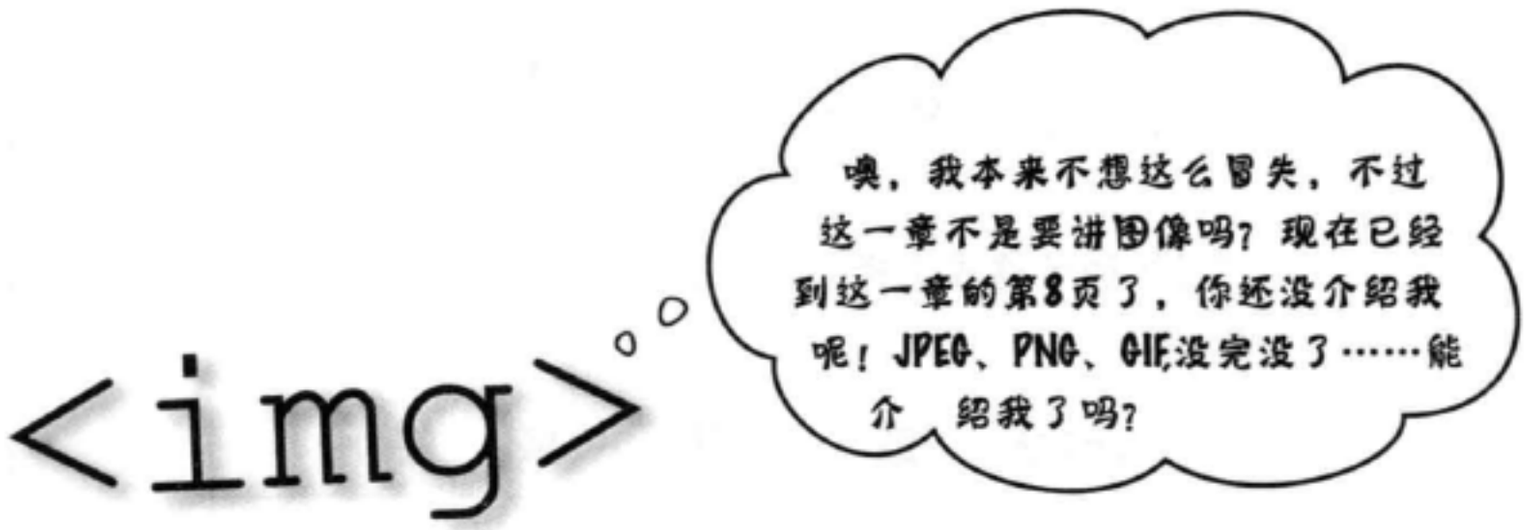
Head First：嗯，下次再做三人采访之前我可得三思了，不过根据你们所说，GIF和PNG，你们很擅长表示logo和文本图像；JPEG，你擅长照片；另外，PNG，如果我们需要透明和很多种颜色时，则你会很方便。好了，再见！

PNG、JPEG、GIF：等一下，别急着走！

哪种图像格式？

祝贺你：你当选为今天的“超级图像格式选择能手”。对下面的每个图像，请选择在Web上显示的最佳格式。

	JPEG	或 PNG	或 GIF
	☐	☐	☐
	☐	☐	☐
	☐	☐	☐
	☐	☐	☐
	☐	☐	☐



现在就来正式介绍：<img>元素。

这个介绍真是拖了很久。可以看到，与处理其他HTML标记相比，处理图像要做更多工作。不管怎样，现在先不说这些了……来认识认识<img>元素吧。

首先来仔细研究这个元素（不过你现在对<img>如何工作可能已经了解得不少了）：

仅此而已吗？还不完全。你可能还希望了解几个属性，当然你还希望知道如何使用<img>元素引用Web上的图像（这些图像可能不在你自己的网站上）。不过，说实在的，你已经掌握了使用<img>元素的基本知识。

下面逐步分析使用<img>元素的几个重点，然后学以致用。

<img>: 不再只是相对链接

src属性不只是用于相对链接，还可以在src属性中放入URL。图像与HTML页面一同存储在Web服务器上，所以Web上的每个图像都有自己的URL，就像Web页面一样。

如果要指向另一个不同网站上的图像，通常要使用这个图像的URL（要记住，对于相同网站上的链接和图像，最好使用相对路径）。

要使用URL链接一个图像，如下所示：

```
<img src="http://www.starbuzzcoffee.com/images/corporate/ceo.jpg">
```

要用URL包含一个图像，只需要把图像的整个URL都放在src属性中。

URL是图像的路径，所以最后的文件名总是一个图像文件名。与默认Web页面不同，这里没有默认图像之说。

Sharpen your pencil

这里有一个“练练笔”练习，它确实与铅笔有关（哦，还有图像）。这个练习提到这样一个问题：给你一根普通的新铅笔，如果用它画一条连续的线，把整个铅笔用完，这条线会有多长？

这和图像有什么关系？要找出答案，先写一些HTML。这个问题的答案就包含在URL http://wickedlysmart.com/hfhtmlcss/trivia/pencil.png图像中。你的任务是把图像增加到这个HTML，并找出答案：

```
<html>
  <head>
    <title>Sharpen your pencil trivia</title>
  </head>
  <body>
    <p>How long a line can you draw with the typical pencil?</p>
    <p>
      <image>
    </p>
  </body>
</html>
```

把image元素放在这里。

there are no Dumb Questions

问：这么说，<img>元素很简单，就是提供了一个途径，可以通过它指定需要在页面上显示的图像的位置，是吗？

答：是的，总结起来就是这样。我们还会讨论这个元素中可以增加的一些属性。以后你还会看到，可以采用很多方法利用CSS改变一个图像的视觉样式。

不过，关于图像本身还有很多需要了解的知识。这些不同的图像格式有什么作用？为什么要使用某一种格式而不是其他格式？这些图像文件会有多大？如何准备在Web页面中使用的图像？

问：我们已经了解到，void元素就是没有内容和结束标记的元素。我们还知道<img>元素是void元素。不过它不是有内容（图像）吗？

答：嗯，更确切地讲，void元素是指HTML页面中在开始标记和结束标记之间没有任何内容的元素。没错，图像也是内容，不过<img>元素只是指向图像。图像并不是HTML页面本身的一部分。实际上，浏览器显示页面时，图像会取代<img>元素。另外要记住，HTML页面是纯文本，所以图像绝对无法直接作为页面的一部分。它是单独存在的。

问：再回到前面的例子，加载一个有图像的页面……我加载一个Web页面时，并没有看到图像一个接一个地加载，为什么？

答：浏览器通常会同时获取图像。也就是说，浏览器会同时请求多个图像。按照现有的计算机和网络速度，这一切发生得太快，所以你通常会看到页面与图像一同显示出来。

问：如果我看到Web页面上的一个图像，那么如何确定它的URL，从而能链接到它呢？

答：大多数浏览器都允许用鼠标右键单击图像，这会弹出一个上下文菜单，提供一些选项。在这些选项中，你会看到“Copy Image Address”（复制图像地址）或“Copy Image Link”（复制图像链接），这两个选项会把图像的URL放在你的剪贴板里。要找到URL，另一种方法是用鼠标右键单击并选择“Open Image in New Window”（在新窗口打开图像），这会在一个浏览器窗口中打开图像。你可以从浏览器的地址栏得到这个图像的URL。最后一种做法是使用浏览器的“查看源代码”（View Source）菜单选项，检查HTML。不过要记住，你找到的可能是图像的一个相对链接，所以需要使用网站域名和图像路径“重新构造”它的URL。

问：为什么JPEG照片要比GIF或PNG照片好，或者为什么GIF或PNG logo要优于JPEG logo？

答：“优于”通常是结合图像质量和文件大小而言的。JPEG照片一般比相同质量的PNG或GIF照片要小，而PNG或GIF logo通常看起来比JPEG格式效果更好，而且文件也可能更小。

问：在GIF和PNG之间如何选择呢？看起来它们很相似。

答：PNG是最新的图像格式，也很有意思，因为它既能支持照片也能很好地支持logo。而且它还提供了比GIF更高级的透明性。现在所有主流浏览器都支持PNG，而几年前情况可大不一样。

要在GIF和PNG中做出选择，有几个方面需要考虑。首先，PNG的压缩要稍稍优于GIF，所以对于一个颜色数相同的图像（也就是说，最多256种颜色），PNG文件可能更小一些。不过，如果需要更多颜色，而GIF无法提供，而且也不能选择JPEG（例如，你需要透明性），那么除了PNG就别无选择了。不过，如果你需要动画，那么还得使用GIF，因为GIF是唯一支持动画并得到广泛支持的格式。

一定要提供候选格式

Web上有一点可以肯定，你永远也无法准确地知道人们会用什么浏览器和设备来查看你的页面。访问者很有可能使用移动设备，有视力障碍的人可能还要用屏幕阅读器，另外有些浏览器可能在网速极慢的环境中运行（可能只能获取一个网站的文本，而无法得到图像），还有可能使用手机、能上网的T恤……谁知道还会有什么！

不过尽管存在这些不确定性，你完全可以做好准备，沉着应对。尽管浏览器可能无法显示你的页面上的图像，但还有一种候选做法。你可以使用`<img>`元素的`alt`属性为访问者提供一些指示，告诉他们图像里有什么信息。可以这样做：

```
<img src="http://wickedlysmart.com/hfhtmlcss/trivia/pencil.png"
     alt="The typical new pencil can draw a line 35 miles long.">
```

alt属性需要指定描述这个图像的一些文本。

如果图像未能显示，就会使用这个文本来取代它。这有些像你通过电话为某个人读一个Web页面，读到图像时，你会读出alt文本来代替图像。

在这个练习中，你会看到，如果无法显示图像，则你的浏览器将如何处理alt属性。基本原理是这样的：如果无法找到图像，则会取而代之显示alt属性。不过，并不是所有浏览器都会这样做，所以你的结果可能有所不同。你需要完成下面的工作。

1. 得到上一个练习的HTML。
2. 更新image元素，使它包含alt属性“The typical new pencil can draw a line 35 miles long.”
3. 将图像名从“pencil.png”改为“broken.png”。这个图像并不真正存在，所以你会发现图像无法显示。
4. 在浏览器中重新加载页面。
5. 最后，下载几个其他的浏览器，再来试一试。是不是得到了不同的结果？

例如，你可以试一试Firefox (http://www.mozilla.org/)或Opera (http://www.opera.com/)。

查看本章最后给出的结果……

调整图像大小

你还需要知道<img>元素的另一个属性。实际上，应该是一对属性：width和height。可以使用这些属性提前告诉浏览器你的页面中一个图像的大小。

可以这样使用width和height：

```
<img src="images/drinks.gif" width="48" height="100">
```

width属性告诉浏览器在页面中显示图像的宽度。

height属性告诉浏览器在页面中显示图像的高度。

宽度和高度都使用像素数指定。如果对像素还不熟悉，则这一章后面我们还会更详细地介绍。你可以为一个图像增加width属性和height属性。如果没有指定宽度和高度，则浏览器在页面中显示这个图像之前会自动确定图像的大小。

there are no Dumb Questions

问：既然浏览器可以确定图像大小，我又何必使用这些属性呢？

答：很多浏览器上，如果在HTML中提供了width和height，浏览器在显示图像之前就可以开始建立页面布局。如果没有指定，则浏览器在知道了图像大小之后，通常需要重新调整页面布局。要记住，浏览器是在下载了HTML文件并开始显示页面之后才下载图像。浏览器下载图像之前，除非你告诉它，否则它无法知道图像的大小。

还可以提供比图像实际尺寸更大或更小的宽度和高度值，这样浏览器就会缩放图像来满足你指定的大小。需要按与原尺寸不同的大小显示一个现有图像时，很多人都会这么做。不过，以后你会看到，出于很多原因，最好不要使用width和height来达到这个目的。

问：我必须成对使用这些属性吗？能不能只指定width或height？

答：可以的，不过，既然你想告诉浏览器某一个尺寸，再提供第二个尺寸又何妨呢？工作量是相同的（而且只提供一个宽度或一个高度并没有多少好处，除非你想把图像缩放到某个特定的宽度或高度）。

问：我们说过很多次，希望用HTML来提供结构，而不要用于提供表现。这些属性看起来是表现属性，我理解的不对吗？

答：这取决于你要如何使用这些属性。如果你把图像宽度和高度设置为正确的尺寸，那么它们只是提供信息。不过，如果你使用width和height属性来调整浏览器中图像的大小，就是在用这些属性提供表现。在这种情况下，可能最好考虑使用CSS来得到同样的结果。

创建超级粉丝网站:myPod

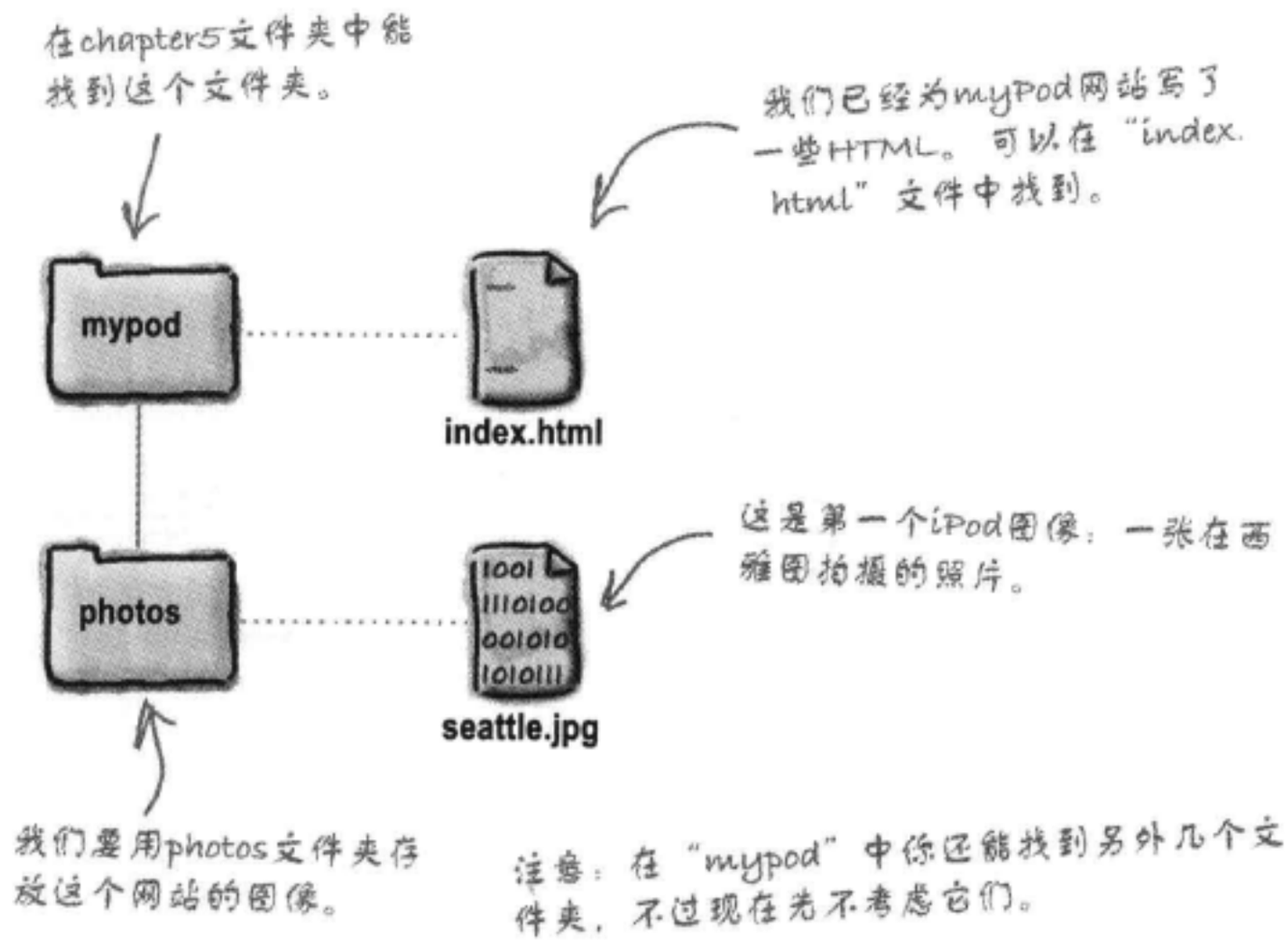
有iPod的人都非常喜爱他们的iPod，不论到哪里都带着它。想象一下，现在要创建一个名为“myPod”的新网站，显示你的朋友带着他们的iPod在世界各地最喜欢的景点拍摄的照片。

首先需要些什么？只需要一些HTML知识、一些图像，还有对iPod的喜爱。

我们已经为这个网站写了一些HTML，不过还没有加入图像，所以要请你来。不过在你增加这些图像之前，先明确几点，在本书示例源码中查找“chapter5”文件夹。你会找到一个名为“mypod”的文件夹。打开“mypod”文件夹，你会看到下面的内容：

iPhone也行！

我的iPod在西雅图！你能看到西雅图的标志性建筑“太空针”。不过你看不到628家咖啡店。



检查myPod的“index.html”文件

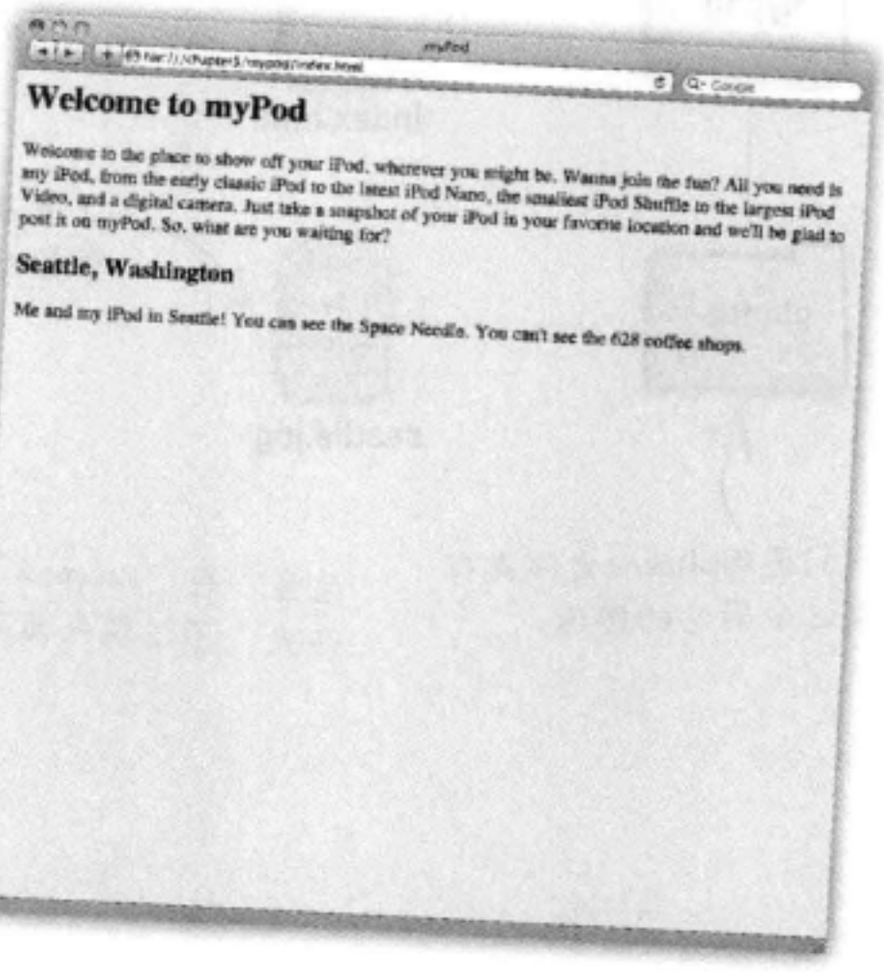
打开文件“index.html”，你会看到myPod已经有内容了。
下面是现在的HTML：

```
<html>
  <head>
    <title>myPod</title>
    <style type="text/css">
      body { background-color: #eaf3da;}
    </style>
  </head>
  <body>

    <h1>Welcome to myPod</h1>
    <p>
      Welcome to the place to show off your iPod, wherever you might be.
      Wanna join the fun? All you need is any iPod from the early classic
      iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
      iPod Video, and a digital camera. Just take a snapshot of your iPod in
      your favorite location and we'll be glad to post it on myPod. So, what
      are you waiting for?
    </p>

    <h2>Seattle, Washington</h2>
    <p>
      Me and my iPod in Seattle! You can see the
      Space Needle. You can't see the 628 coffee shops.
    </p>

  </body>
</html>
```

这里加入了一些成品CSS。现在直接输入就可以了。这些CSS的作用就是让页面有一个淡绿色背景。后面几章我们还会仔细讨论CSS，我保证！

这个HTML看起来很熟悉，因为这里使用的都是基本构建模块：<h1>、<h2>和<p>。

这是浏览器中显示的结果。还不错，不过我们还需要图像。



Sharpen your pencil

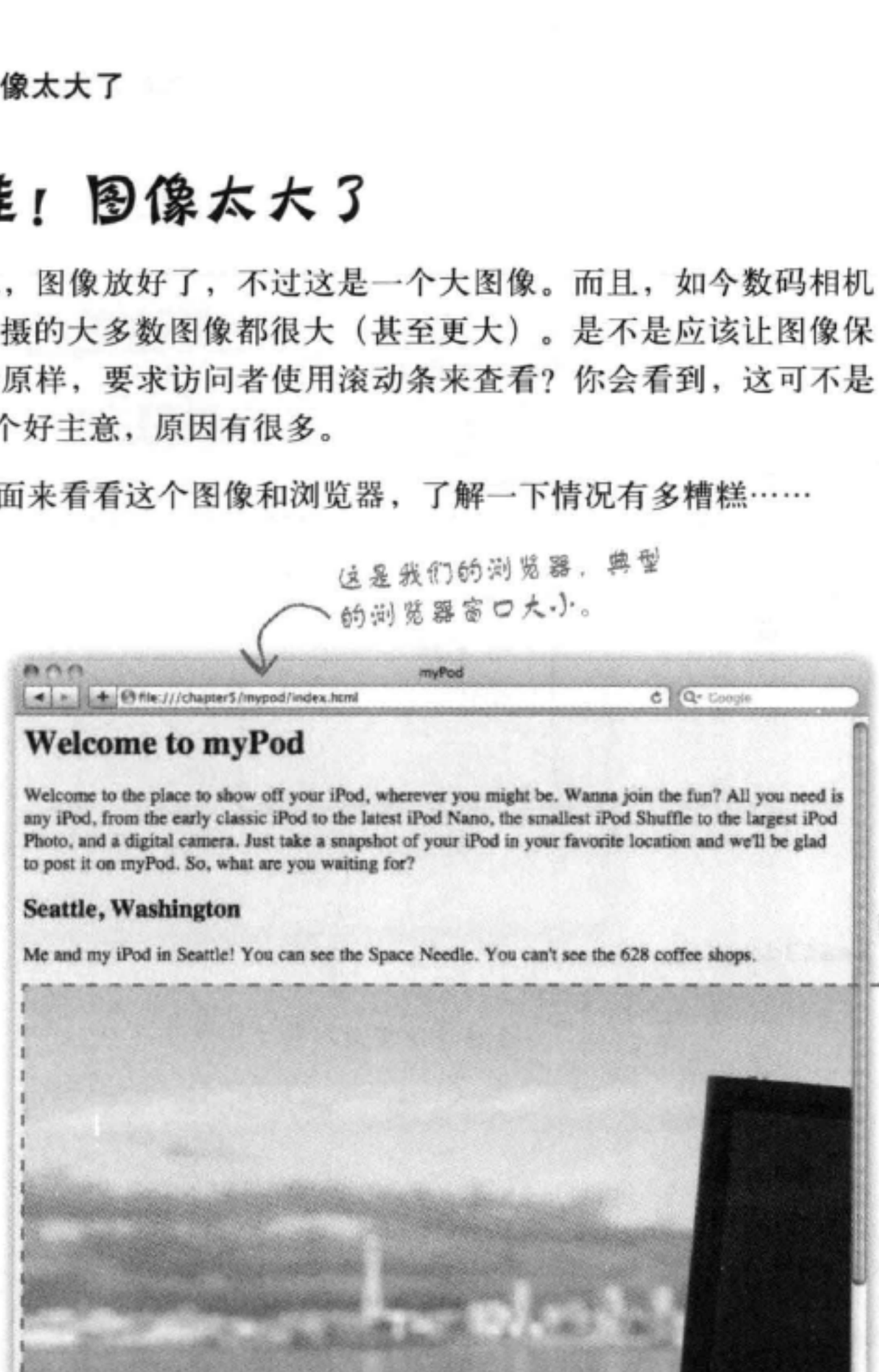
可以看到，myPod网站的大部分HTML都已经写好，准备运行了。现在要做的就是为你希望加入的每张照片增加一个<img>元素。目前为止只有一张照片“seattle_video.jpg”，所以来增加一个元素，把这个图像放在下面的页面中。完成后，在你的浏览器中加载这个页面，你就能看到西雅图的风景了。


```
<html>
   <head>
      <title>myPod</title>
      <style type="text/css">
         body { background-color: #eaf3da;}
      </style>
   </head>
   <body>

      <h1>Welcome to myPod</h1>
      <p>
         Welcome to the place to show off your iPod, wherever you might be.
         Wanna join the fun? All you need is any iPod from, the early classic
         iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
         iPod Video, and a digital camera. Just take a snapshot of your iPod in
         your favorite location and we'll be glad to post it on myPod. So, what
         are you waiting for?
      </p>

      <h2>Seattle, Washington</h2>
      <p>
         Me and my iPod in Seattle! You can see the
         Space Needle. You can't see the 628 coffee shops.
      </p>

      <p>
                                  你的<img>元素要放在
                                  这里。
      </p>

   </body>
</html>
```

哇！图像太大了

嗯，图像放好了，不过这是一个大图像。而且，如今数码相机拍摄的大多数图像都很大（甚至更大）。是不是应该让图像保持原样，要求访问者使用滚动条来查看？你会看到，这可不是一个好主意，原因有很多。

下面来看看这个图像和浏览器，了解一下情况有多糟糕……



Watch it!

如果图像在浏览器窗口中刚好能放下，说明你的浏览器可能打开了“auto image resize”（自动调整图像大小）选项。有关的更多内容稍后介绍……

there are no Dumb Questions

问：让用户用滚动条查看图像有什么不对？

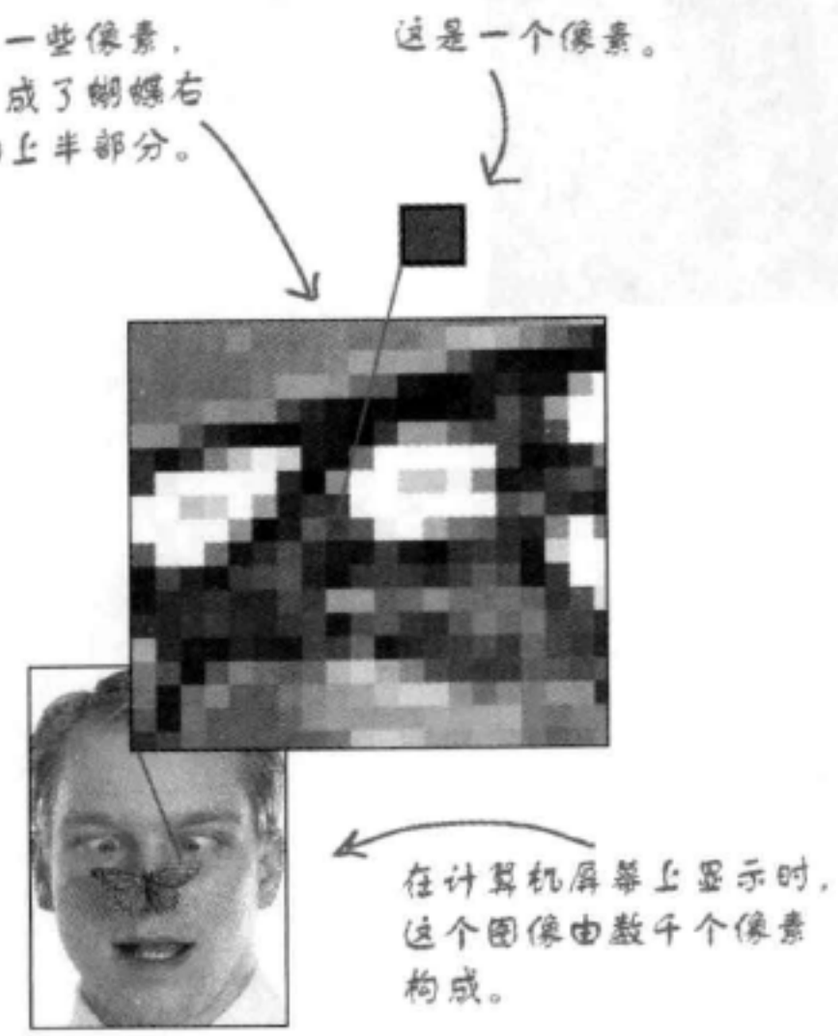
答：一般来讲，如果页面中有很大的图像，则这个页面会很难使用。访问者没有办法一次看到完整的图像，不仅如此，使用滚动条也很麻烦。另外大图像还需要服务器和浏览器之间传输更多数据，这会耗费很多时间，可能导致你的页面显示速度很慢，特别是对于那些通过拨号上网或使用其他慢速连接的用户。

问：为什么不能直接使用width属性和height属性来调整页面上图像的大小？

答：因为浏览器在缩放图像使之适应页面大小之前，仍然需要获取整个大图像。

问：你刚才说浏览器窗口宽度是800像素，这是什么意思？

答：你的计算机显示屏是由数百万个称为像素的点组成的。如果仔细看你的显示屏，就应该能看到它们：



尽管屏幕大小和分辨率可能有变化（有些人的显示器小，有些则很大），大多数人通常都会把浏览器宽度设置为800到1280像素之间。所以一般经验是将图像最大宽度设置为800像素（Web页面也一样，不过这个内容在后面一章再做介绍）。

问：像素数与屏幕上图像的大小有关系吗？

答：一般经验是每英寸96像素，不过如今的显示器分辨率都很高，所以还可以更高。我们习惯使用标准的72像素/英寸（ppi），不过为了应对现代显示器，又创造了一个CSS像素的概念。CSS像素是1英寸的1/96（96 ppi）。所以对于一个3"宽 × 3"高的图像，要使用 96 (像素) × 3 (英寸) = 288 × 288像素。

问：嗯，那么我的图像应该设置为多大呢？

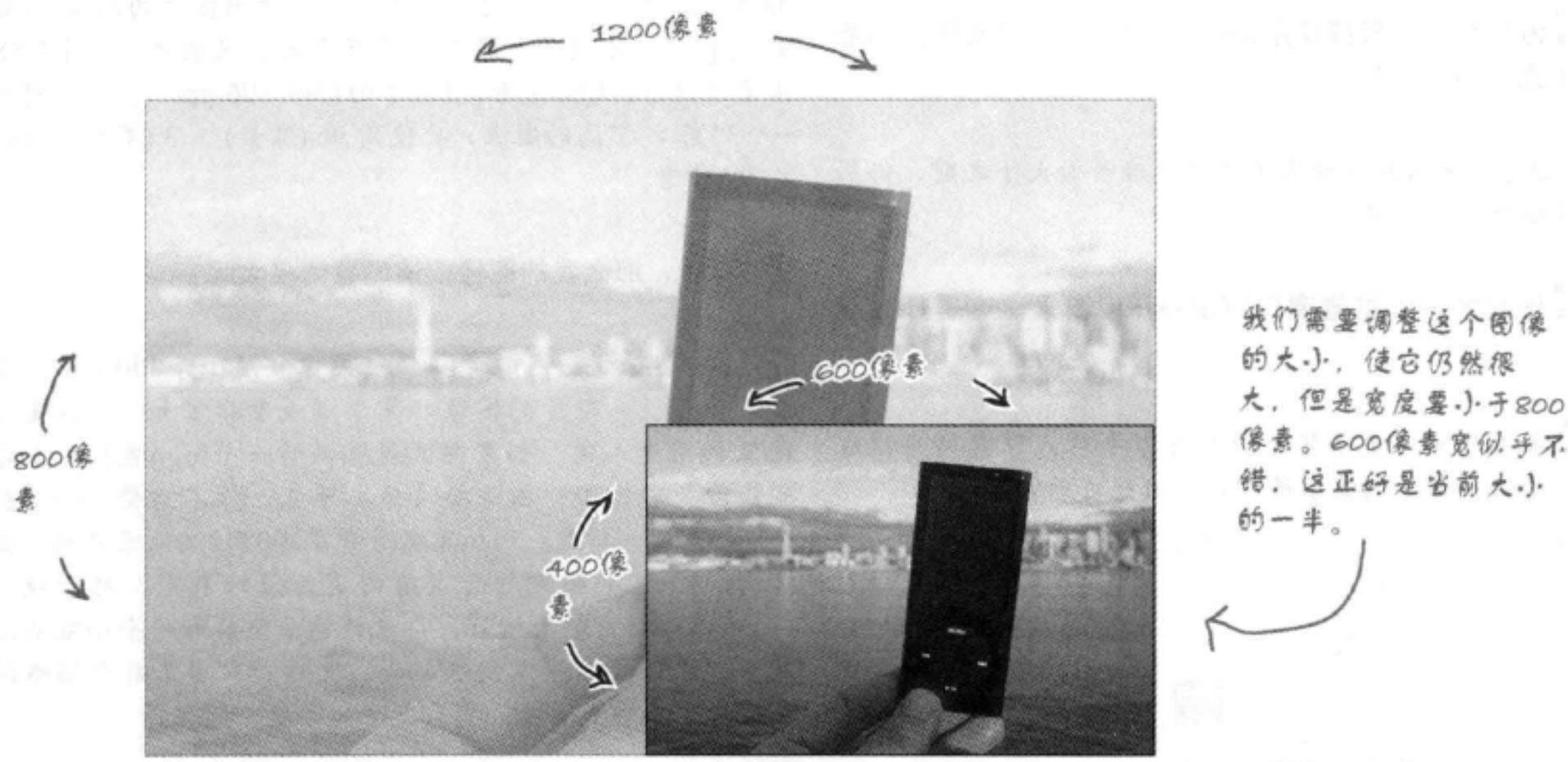
答：一般来讲，往往希望保证图像宽度小于800像素。当然，你可能希望你的图像非常小（或者非常大），这要由图像的用途而定。如果图像是页面的一个logo呢？那么你可能希望它很小，但是要清楚。毕竟，你不需要logo与整个Web页面一样宽。logo宽度通常在100到200像素之间。归根结底，这个问题的答案要由页面设计来定。对于照片（通常你希望它越大越好），也许你希望能有一个小缩略图页面，这个页面可以快速加载，允许用户单击各个缩略图来查看原来的大图像。稍后就要讨论这个问题。

问：我觉得我的浏览器会自动调整西雅图图像的大小，因为它在窗口里看起来刚好合适。为什么我的浏览器会这么做？

答：有些浏览器有一个特性，可以对不适合浏览器宽度的图像调整大小。不过很多浏览器不这么做，所以不要依赖这个特性。即使每个浏览器确实有这个特性，这样做也不好，这会在服务器和浏览器之间不必要地传输更多数据，导致你的页面加载很慢，可用性降低。另外要记住，越来越多的人在移动设备上查看Web页面，太大的图像会影响这些设备上的数据使用。

调整图像大小以适合浏览器

下面调整这个图像的大小，让它能更适合浏览器窗口。现在这个图像是1200像素宽、800像素高（稍后你就会了解这是如何确定的）。因为我们希望图像的宽度小于800像素，所以要确定一个适当的宽度，能适合我们的myPod Web页面。myPod网站的目的就是浏览在不同场合拍摄的iPod的照片，所以我们可能希望看到足够大的图像。如果把图像大小缩小一半为600像素宽、400像素高，则宽度上会基本占满浏览器，不过两边还会有一些空白。听起来还不错？下面就来调整图像的大小……



你要做的工作：

1. 用一个照片编辑应用打开这个图像。
2. 把图像大小缩小为一半（600像素×400像素）。
3. 将图像保存为“seattle_video_med.jpg”。

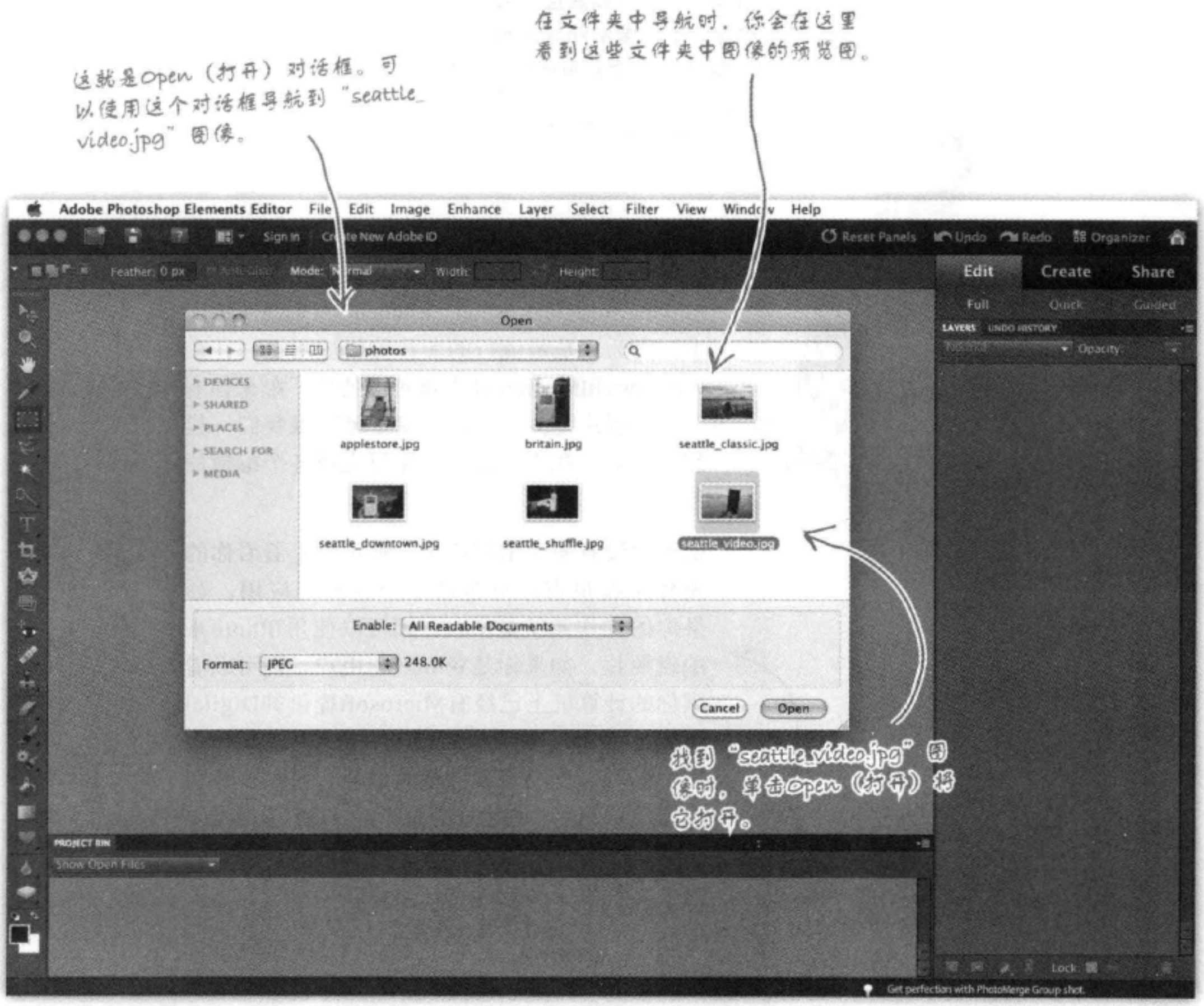
问得好，市场上有很多照片编辑应用（有一些还是免费的），它们都差不多。我们要用Adobe公司的Photoshop Elements来调整图像大小，因为这是最流行的照片编辑应用之一，在Windows和Macintosh上都可以使用。如果你有自己的照片编辑应用，则完全可以跟我们一起继续学习，在你自己的应用里完成各个编辑任务。

如果你没有照片编辑应用，则可以先看看你的操作系统里有没有包含一个这样的应用。如果你的操作系统是Mac，则可以使用iPhoto来编辑照片。如果你是Windows用户，则可以看到你的计算机上已经有Microsoft提供的Digital Image Suite。如果确实没有一个可用的编辑应用，则可以先跟着我们继续学习，每一步都可以使用示例文件夹中已经包含的HTML和图像。

如果没有Adobe Photoshop Elements，但还想跟着我们学习这一章，则可以先下载这个软件的试用版，能免费使用30天。下载URL为：http://www.adobe.com/go/tryphotoshop_elements。

打开图像

首先，启动你的照片编辑应用，打开“seattle_video.jpg”图像。在Photoshop Elements中，可以选择File（文件）菜单下的“Open…”（打开……）菜单项，这会打开一个Open（打开）对话框。可以使用这个对话框导航到“chapter5/mypod/photos”文件夹中的图像“seattle_video.jpg”。

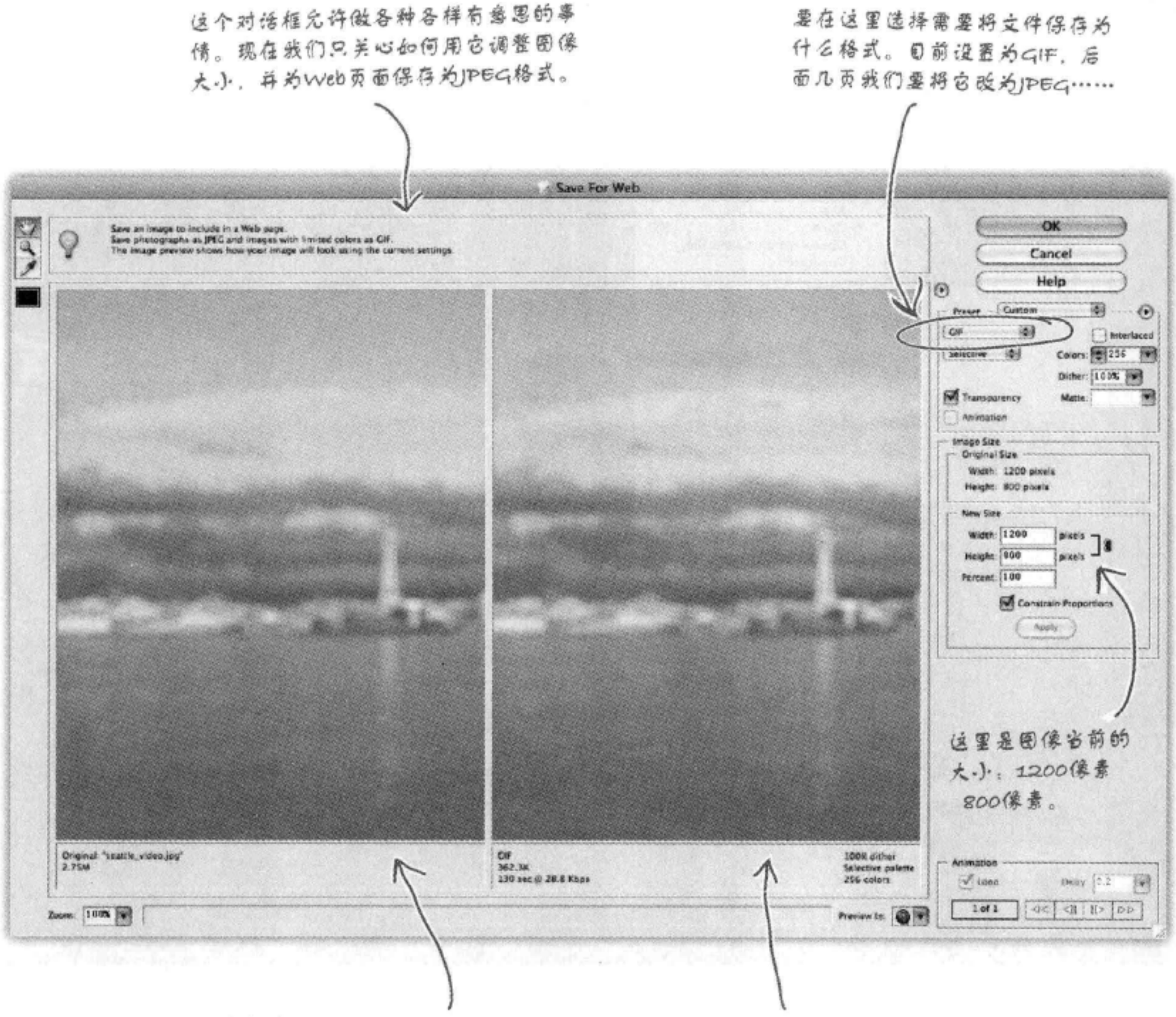


调整图像大小

现在“seattle_video.jpg”已经打开，我们要用“Save for Web”（保存为Web格式）对话框调整图像大小并保存。要打开这个对话框，只需要从File（文件）菜单选择“Save for Web”（保存为Web格式）单项。

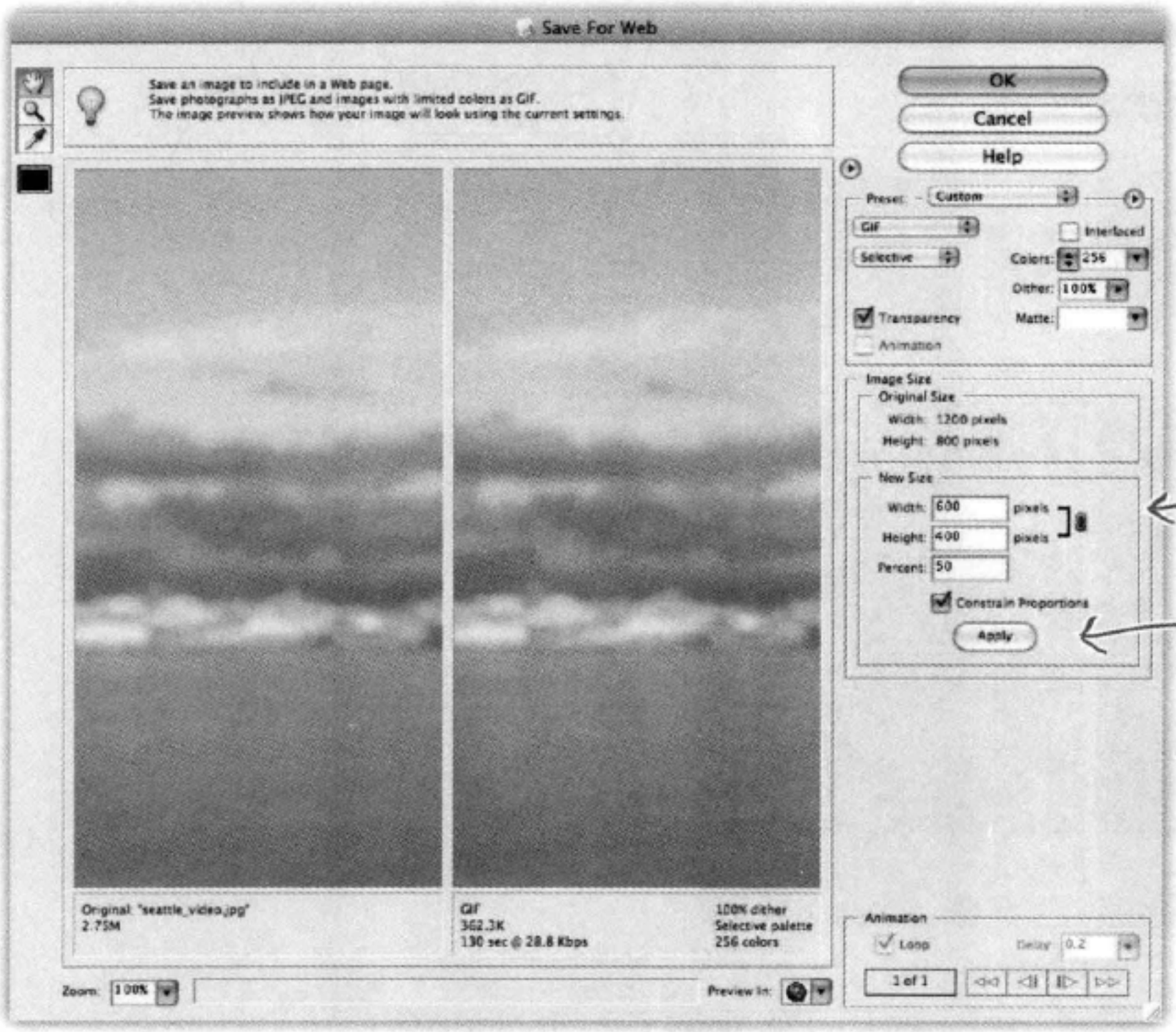
调整图像大小（续……）

选择“Save for Web”（保存为Web格式）菜单项之后，应该会看到下面的对话框，在使用之前先来熟悉一下。



可以看到，这个对话框里内置有很多功能。下面来充分利用这个对话框。要调整图像的大小，需要把宽度改为600像素，高度改为400像素。然后需要用JPEG格式保存这个图像。下面先来调整图像的大小……

(1) 在这里调整图像大小，改为宽度600，高度400。如果选中了Constrain Proportions（保持纵横比），那么你只需要输入新的宽度600，Elements就会帮你把高度改为400。



(2) 一旦正确地设置了宽度和高度，就单击Apply（应用），让Elements知道这是你想要的大小。

这不会影响原来的图像，只会影响你要保存的那个文件。

必须单击Apply（应用）来缩小图像大小，否则图像将仍按原宽度和高度保存。

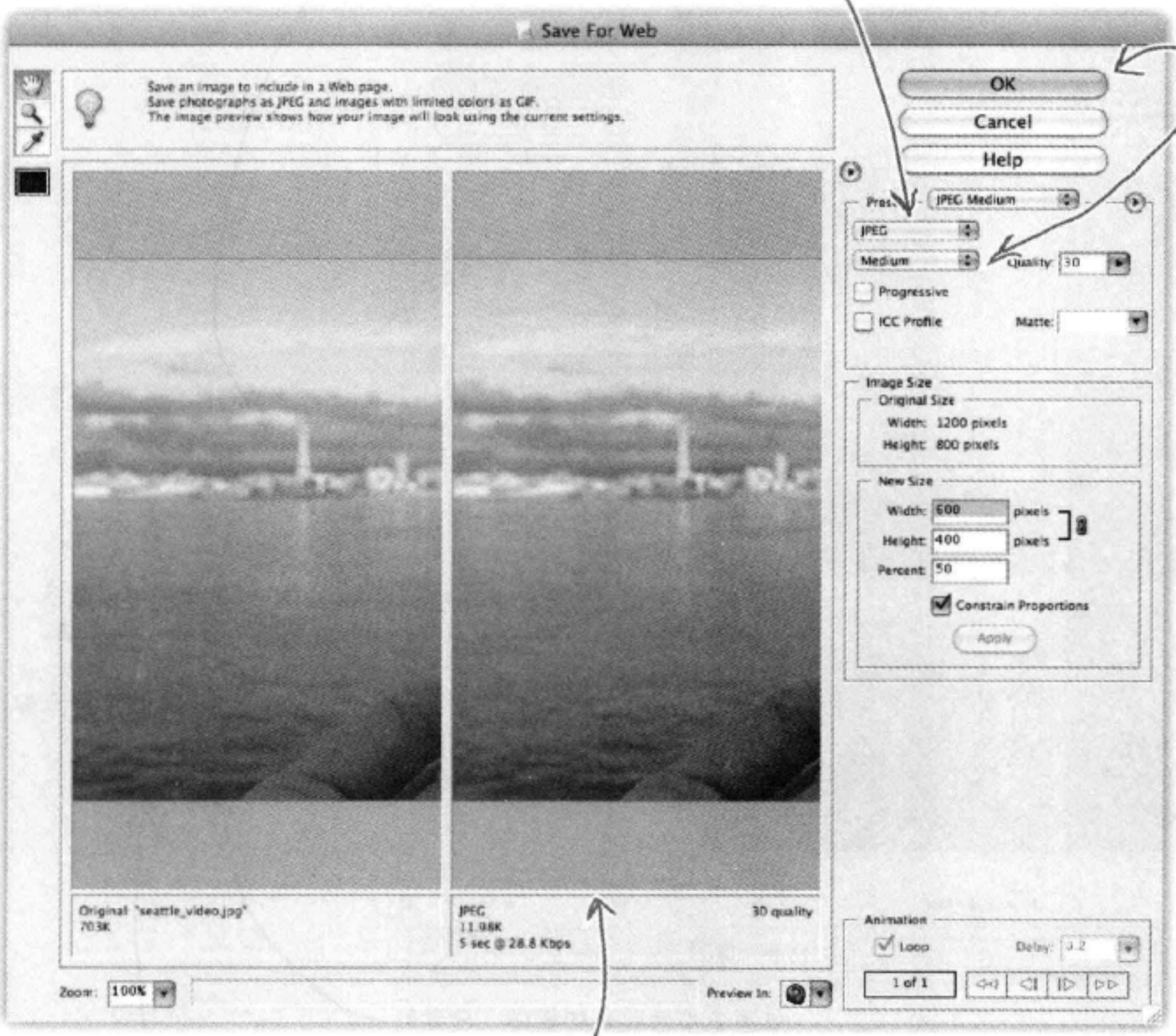
大小调整完毕，现在来保存

现在你只需要按正确的格式（JPEG）保存图像。为此，需要选择JPEG格式，将质量设置为Medium（中）。稍后我们再来讨论质量问题。

(1) 现在已经设置了图像大小，只需要选择图像格式。目前的设置是保存为GIF，现在就像我们一样，将它改为JPEG。

(2) 将质量设置为 Medium（中）。

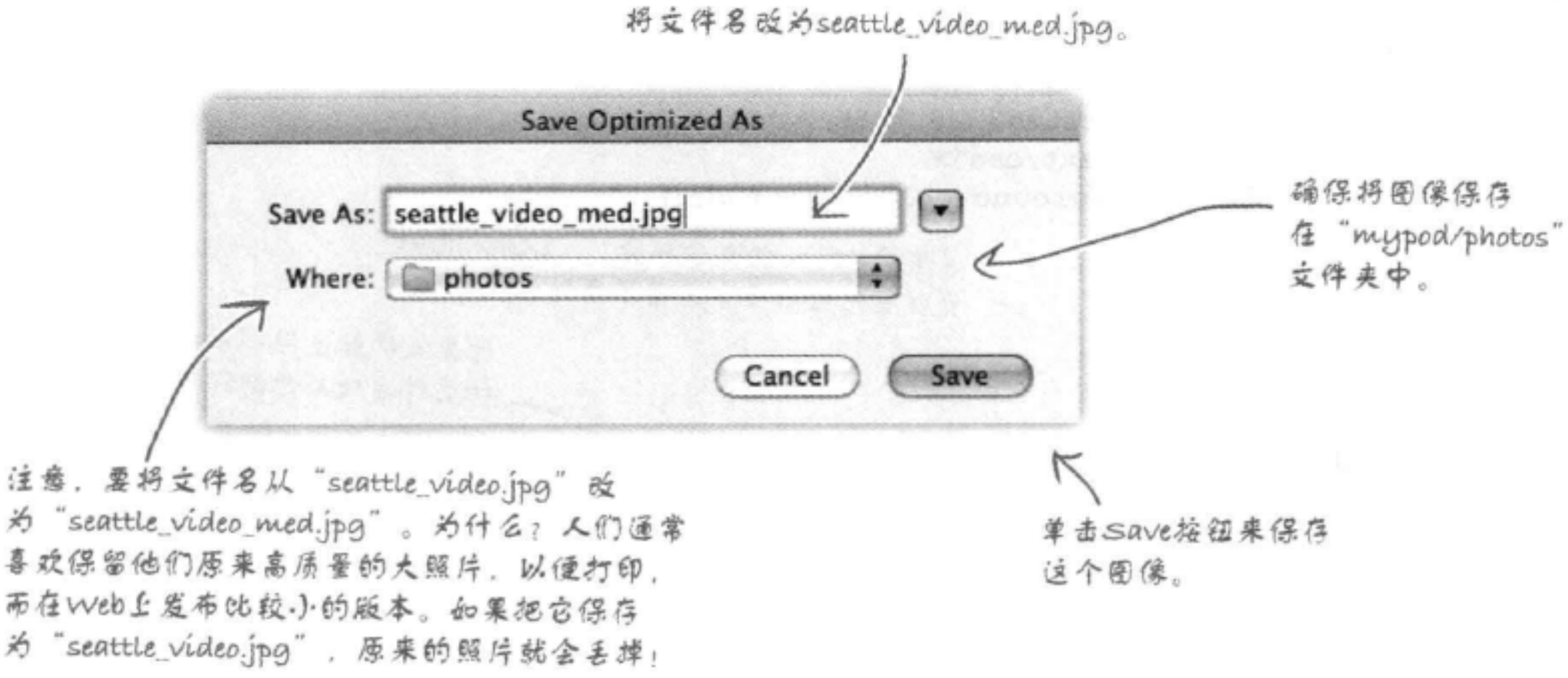
(3) 这就可以了，单击OK按钮，来看下一页。



注意，上一步单击Apply（应用）按钮时，会调整图像大小并重新显示。

保存图像

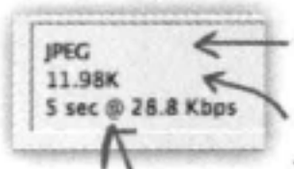
单击OK按钮之后，会打开一个Save（保存）对话框。将图像保存为“seattle_video_med.jpg”，这样就不会覆盖原来的照片了。



there are no Dumb Questions

问：能不能再讲讲“Save for Web”（保存为Web格式）中的质量设置？

答：JPEG格式允许指定你需要的图像质量等级。质量越低，文件大小就越小。如果查看“Save for Web”（保存为Web格式）对话框中的预览窗口，则可以看到改变质量设置时，质量和文件大小会随之改变。



要对质量设置和不同图像格式有所认识，最好的办法就是实践，用你自己的图像做些试验。你会很快发现，对于你的图像和所要开发的Web页面类型，需要怎样的质量等级。同时你还能逐渐了解何时使用JPEG格式而不是其他格式。

问：在JPEG选项对话框中，Quality（质量）标签旁边的数字30是什么意思？

答：30就是Photoshop Elements所认为的Medium（中）质量。JPEG实际上使用1%～100%范围内的数值，Low（低）、Medium（中）和High（高）等只是表示很多照片编辑应用常用的一些值。

问：不能直接用<img>元素的width属性和height属性来调整图像的大小吗？

答：可以使用width属性和height属性来调整一个图像的大小，不过这不是一个好主意。为什么？因为如果这样做，则仍然会下载完整的大图像。让浏览器来完成调整图像大小的工作，就像你打开了浏览器上的自动调整大小选项一样（如果浏览器支持这个特性）。width属性和height属性实际上是帮助浏览器确定要为这个图像预留多大的空间。如果使用这两个属性，则它们应当与图像的实际宽度和高度一致。

修复myPod HTML

一旦保存了图像，就可以关闭Photoshop Elements。现在只需要修改myPod “index.html”页面，在其中包含照片“seattle_video_med.jpg”的新版本。下面是“index.html”文件的一个片段，只给出了需要你修改的部分。

```
<html>
  <head>
    <title>myPod</title>
    <style type="text/css">
      body { background-color: #eaf3da;}
    </style>
  </head>
  <body>
    .
    .
    .
    <h2>Seattle, Washington</h2>
    <p>
      Me and my iPod in Seattle! You can see the
      Space Needle. You can't see the 628 coffee shops.
    </p>

    <p>
      <img src="photos/seattle _ video _ med.jpg" alt="My iPod in Seattle, WA">
    </p>

  </body>
</html>
```

这里是HTML的其余部分，“index.html”文件里已经有了，这里不再列出。

你要做的就是将<img>元素中的文件名改为你刚保存的图像名：“seattle_video_med.jpg”。

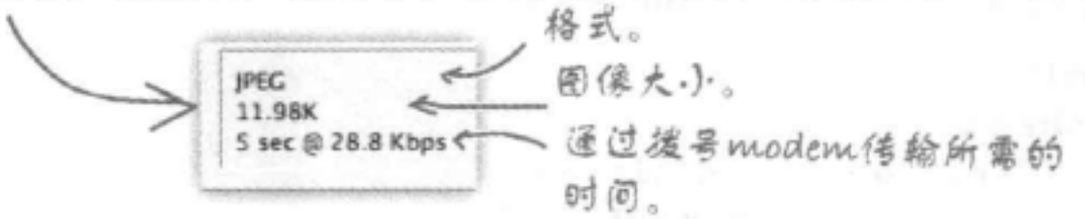
现在来试一试……

完成以上修改，保存“index.html”文件，并在浏览器中重新加载。看起来好多了。现在图像大小正合适，很方便访问者查看，再不会是一个庞大的照片让访问者不知所措了。

现在图像可以在浏览器窗口中很好地显示。而且文件大小也小多了，这样可以帮助页面更快地加载。

哪一种图像格式？

这一次的任务：在Photoshop Elements中打开文件“chapter5/testimage/eye.jpg”。打开“Save for Web”（保存为Web格式）对话框，选择JPEG的各种质量设置（Low、Medium、High等），再试试PNG-24和GIF，填写下面的空格。你会在图像下面的预览窗口中找到这些信息。完成后，确定对于这个图像哪种设置最合适。



格式	质量	大小	时间	优胜者
PNG-24	N/A			☐
JPEG	Maximum			☐
JPEG	High			☐
JPEG	Medium			☐
JPEG	Low			☐
GIF	N/A			☐

也试试PNG-8！

myPod的更多照片

又为myPod提供了一批新的照片：3张来自西雅图，还有一张是英国的一个朋友拍摄的。这些照片大小已经调整，宽度小于800像素。为它们增加<img>元素（你会发现这些图像已经在photos文件夹中）：

这里还可以增加你自己的一些照片。不过要记住首先要调整照片的大小。

```
<html>
  <head>
    <title>myPod</title>
    <style type="text/css">
      body { background-color: #eaf3da;}
    </style>
  </head>
  <body>
    <h1>Welcome to myPod</h1>
    <p>
      Welcome to the place to show off your iPod, wherever you might be.
      Wanna join the fun? All you need is any iPod, from the early classic
      iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
      iPod Video, and a digital camera. Just take a snapshot of your iPod in
      your favorite location and we'll be glad to post it on myPod. So, what
      are you waiting for?
    </p>

    <h2>Seattle, Washington</h2>
    <p>
      Me and my iPod in Seattle! You can see the
      Space Needle. You can't see the 628 coffee shops.
    </p>

    <p>
      <img src="photos/seattle_video_med.jpg" alt="My video iPod in Seattle, WA">
      <img src="photos/seattle_classic.jpg" alt="A classic iPod in Seattle, WA">
      <img src="photos/seattle_shuffle.jpg" alt="An iPod Shuffle in Seattle, WA">
      <img src="photos/seattle_downtown.jpg" alt="An iPod in downtown Seattle, WA">
    </p>

    <h2>Birmingham, England</h2>
    <p>
      Here are some iPod photos around Birmingham. We've obviously got some
      passionate folks over here who love their iPods. Check out the classic
      red British telephone box!
    </p>

    <p>
      <img src="photos/britain.jpg" alt="An iPod in Birmingham at a telephone box">
      <img src="photos/applestore.jpg" alt="An iPod at the Birmingham Apple store">
    </p>
  </body>
</html>
```

这里把所有西雅图的照片放在一起。

伯明翰的照片也做同样的处理……

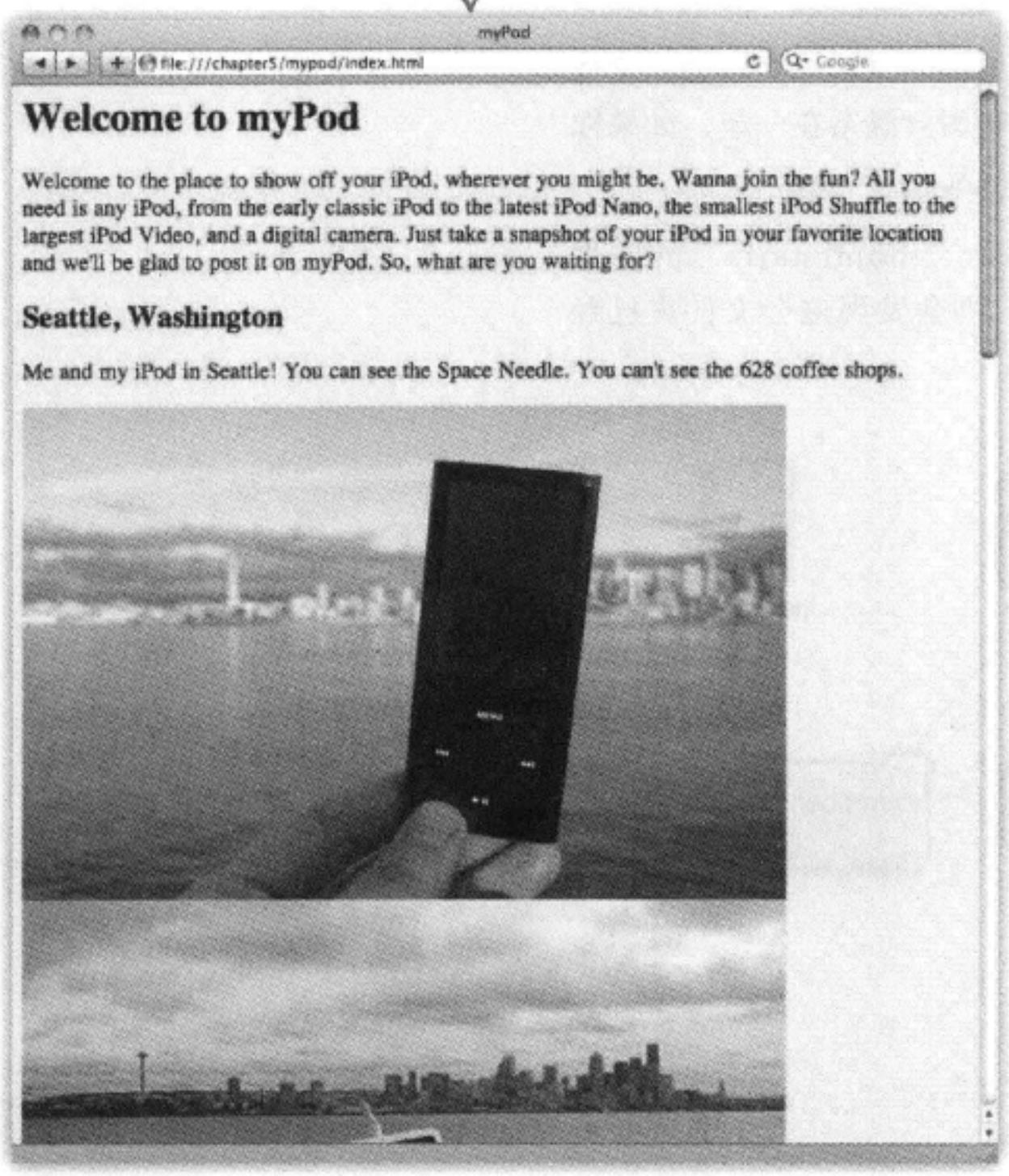
再试试myPod

现在不需要我们再多说，相信你肯定早就知道了：下面在浏览器中重新加载这个页面。哇，多几个图像就大不一样了，是不是？这个页面开始有点意思了。

不过，这并不表示你已经大功告成。现在已经在页面上放置了一组图像，不过即使已经调整了这些图像的大小，它们还是太大了。增加更多的图像时，不仅页面加载越来越慢，用户还必须大量滚动才能看到全部图像。如果用户能看到每个照片的一个小“缩略”图像，然后单击这个缩略图来查看更大的图像，这样不是更好吗？

现在加上所有图像，整个页面是这样的。

近看页面是这样的。



修改网站，使用缩略图

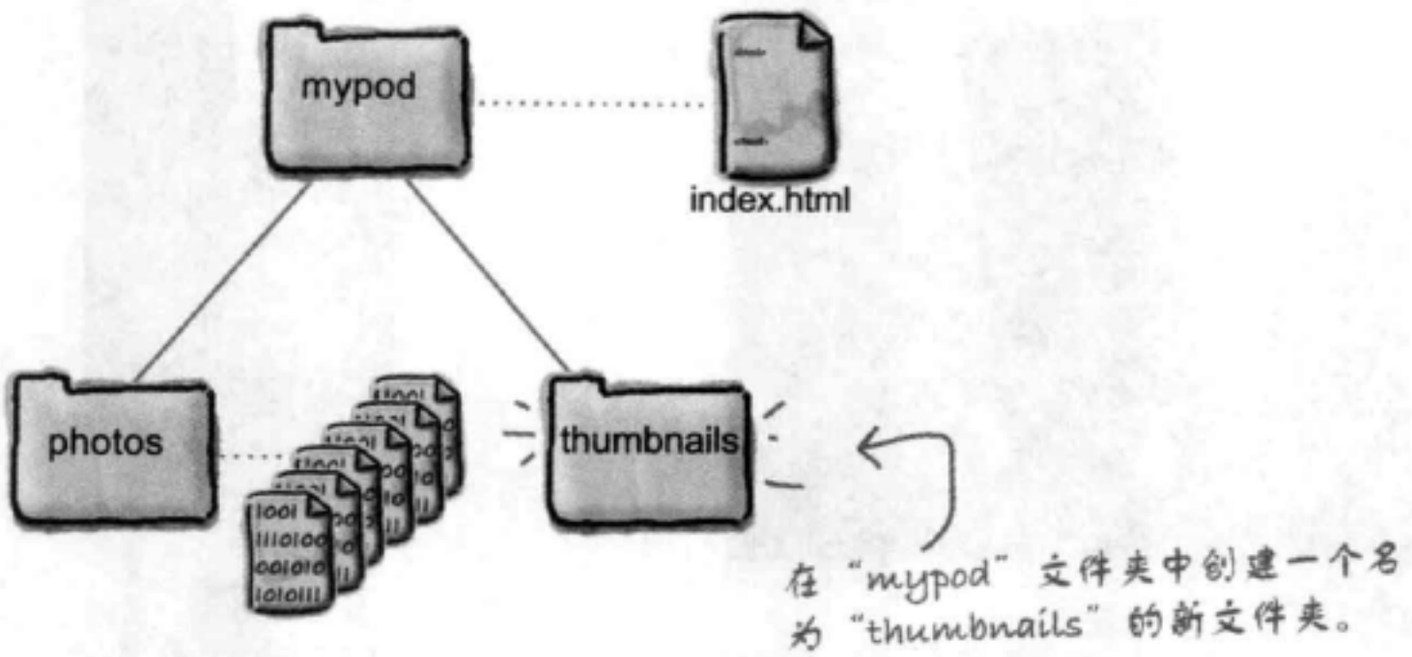
现在要让这个页面更可用，把每个照片替换为一个更小的图像（我们把它叫做缩略图），然后创建从这个缩略图到各个更大照片的链接。你要按以下步骤完成这个工作：

1. 为缩略图创建一个新目录。
2. 把各个照片的大小调整为150 像素 × 100像素，把它保存到“thumbnail”文件夹中。
3. 将“index.html”中各个<img>元素的src设置为照片的缩略版本。
4. 增加从各个缩略图到一个新页面的链接，这个新页面中包含相应的大照片。

为缩略图创建一个新目录

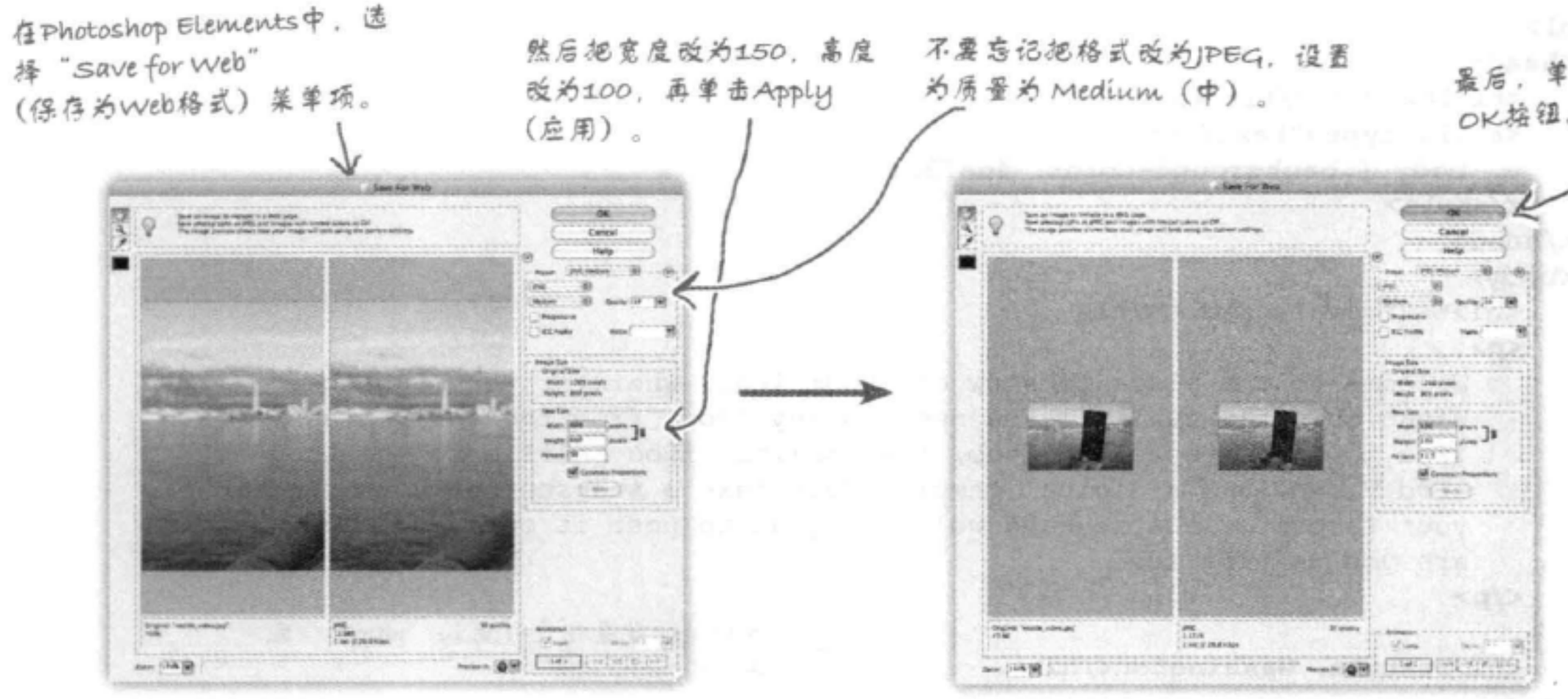
为了更有条理、更有组织，就为这些缩略图创建一个单独的文件夹。否则，图像文件夹中大图像和缩略图会混杂在一起，如果你增加了大量照片，则情况会变得相当混乱。

在“mypod”文件夹下创建一个名为“thumbnails”的文件夹。如果你使用这本书的示例文件，则会发现这个文件夹已经在“mypod”文件夹下了。



创建缩略图

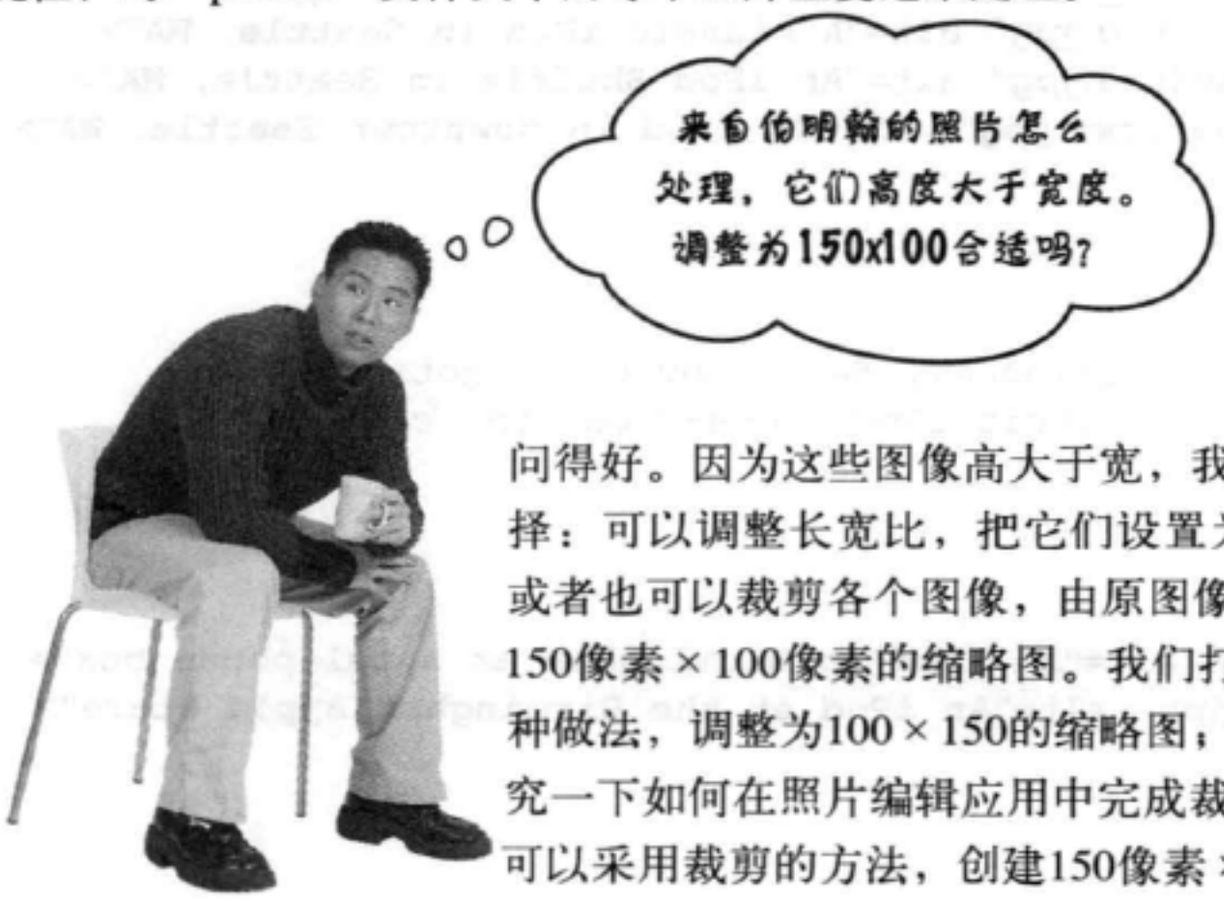
已经有了放置缩略图的地方，下面来创建这些缩略图。首先在你的照片编辑应用中打开“seattle_video_med.jpg”。使用之前创建600像素×400像素版本时同样的方法把它的大小调整为150像素×100像素：



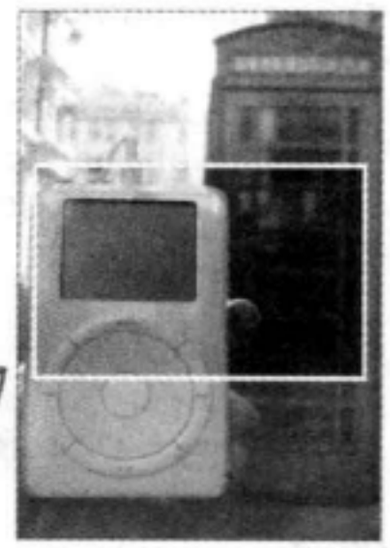
调整图像大小后，选择OK按钮，并按同名保存，不过要保存到thumbnail文件夹中。要当心：如果把它保存到“photos”文件夹，就会替换原来的大图像。

如果使用示例文件，则你会发现这些缩略图已经在“thumbnails”文件夹中，所以你不用这样处理每一张照片（毕竟，你是要学习HTML，而不是批量处理照片）。

现在，对“photos”文件夹中的每个照片重复这个处理。



问得好。因为这些图像高大于宽，我们有两种选择：可以调整长宽比，把它们设置为100×150，或者也可以裁剪各个图像，由原图像裁剪出一个150像素×100像素的缩略图。我们打算采出前一种做法，调整为100×150的缩略图；如果你想研究一下如何在照片编辑应用中完成裁剪，则完全可以采用裁剪的方法，创建150像素×100像素的缩略图像。



修改HTML，使用缩略图

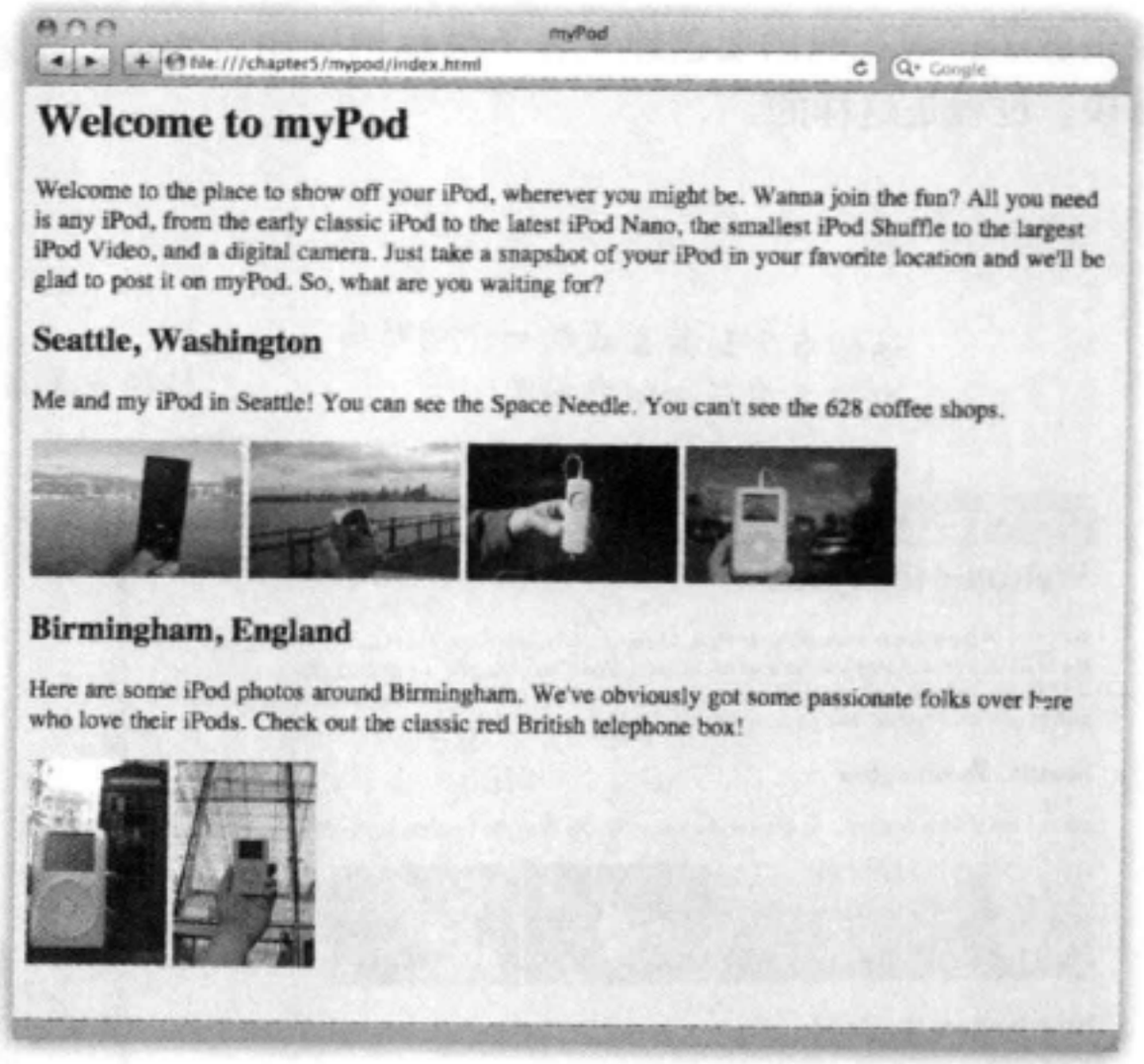
现在只需要修改HTML，使<img>元素从“thumbnails”文件夹而不是“photos”文件夹中取图像，因为你现在使用的是相对路径，如“photos/seattle_video_med.jpg”，所以方法很简单：对于每一个<img>元素，你只需要把文件夹从“photos”改为“thumbnails”。

```
<html>
  <head>
    <title>myPod</title>
    <style type="text/css">
      body { background-color: #eaf3da;}
    </style>
  </head>
  <body>
    <h1>Welcome to myPod</h1>
    <p>
      Welcome to the place to show off your iPod, wherever you might be.
      Wanna join the fun? All you need is any iPod, from the early classic
      iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
      iPod Video, and a digital camera. Just take a snapshot of your iPod in
      your favorite location and we'll be glad to post it on myPod. So, what
      are you waiting for?
    </p>

    <h2>Seattle, Washington</h2>
    <p>
      Me and my iPod in Seattle! You can see the
      Space Needle. You can't see the 628 coffee shops.
    </p>

    <p>
      <img src="thumbnails/seattle_video_med.jpg" alt="My video iPod in Seattle, WA">
      <img src="thumbnails/seattle_classic.jpg" alt="A classic iPod in Seattle, WA">
      <img src="thumbnails/seattle_shuffle.jpg" alt="An iPod Shuffle in Seattle, WA">
      <img src="thumbnails/seattle_downtown.jpg" alt="An iPod in downtown Seattle, WA">
    </p>

    <h2>Birmingham, England</h2>
    <p>
      Here are some iPod photos around Birmingham. We've obviously got some
      passionate folks over here who love their iPods. Check out the classic
      red British telephone box!
    </p>

    <p>
      <img src="thumbnails/britain.jpg" alt="An iPod in Birmingham at a telephone box">
      <img src="thumbnails/applestore.jpg" alt="An iPod at the Birmingham Apple store">
    </p>
  </body>
</html>
```

你要做的就是把文件夹从“photos”改为“thumbnails”。

再来测试myPod

哈……好多了。访问者可以一眼看到所有可用的图片。他们还能更容易地区分不同城市的照片。现在我们需要想办法从每个缩略图链接到相应的大图像。



等一下，你不觉得你在捣鬼吗？图像原来是上下摆放的，现在变成并排的了。

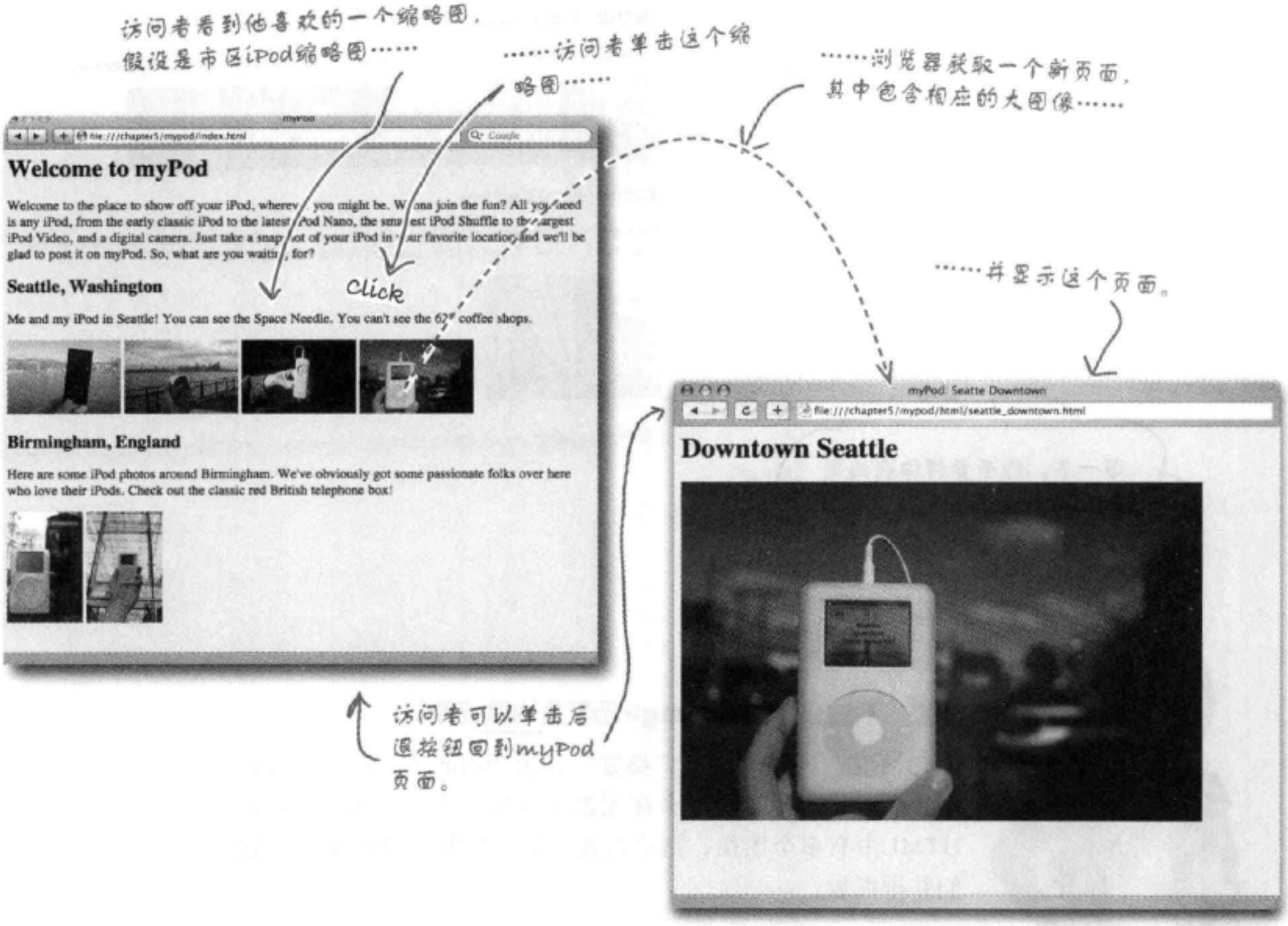
没错，不过应该记得<img>元素是内联元素。

换句话说，我们并没有“捣鬼”。因为<img>显示为内联元素，所以显示元素时不会在元素前后插入换行。因此，如果HTML中有多个图像，浏览器窗口足够宽时，浏览器会把它们并排摆放。

那些更大的照片之所以没有并排摆放，是因为浏览器没有足够的空间来并排显示它们。实际上，它只能从上到下显示各个大图像。浏览器总是在块元素前后在垂直方向上显示间距，如果再看看前面的截屏图，则你会看到那些图像尽管上下摆放，但是相互之间没有间距。这也说明了<img>是一个内联元素。

把缩略图变成链接

差不多了。现在只需要创建从各个缩略图到相应的更大版本的一个链接。过程是这样的：



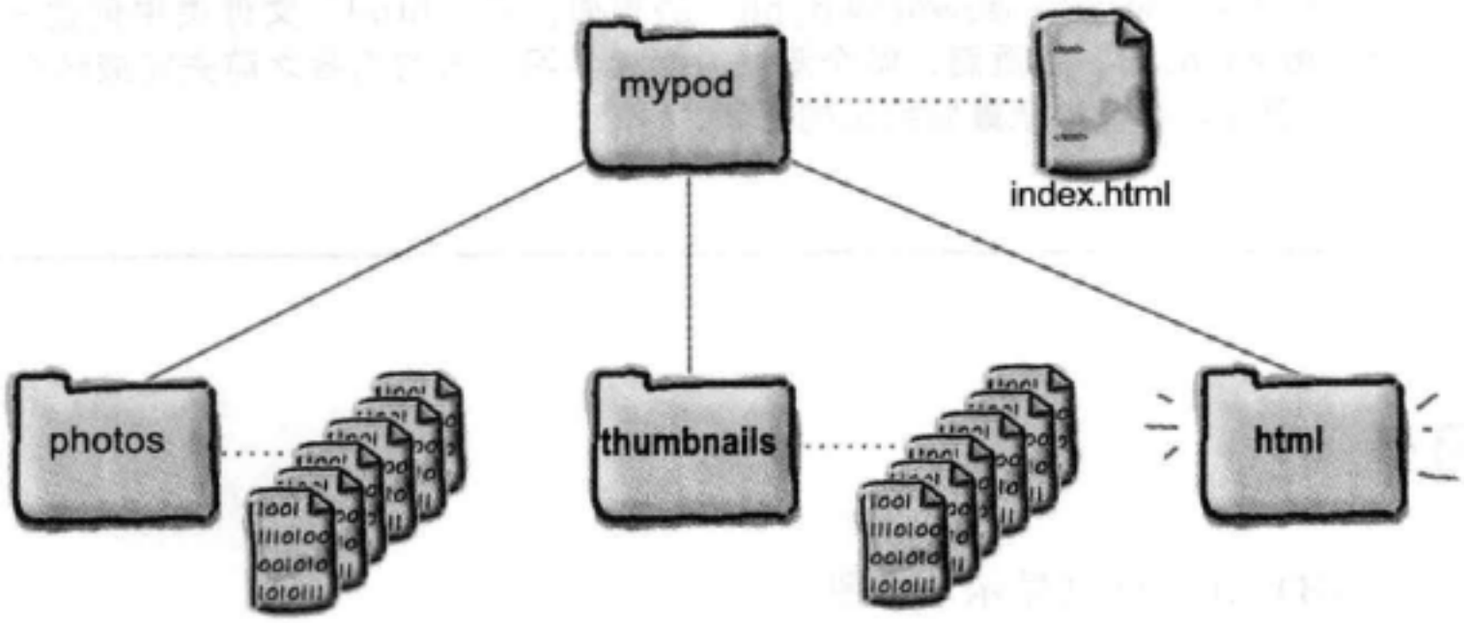
为此你需要两样东西：

1. **对应每个照片要有一个页面，其中显示各个照片，还要有一个标题来描述照片内容。**
2. **“index.html”中从每一个缩略图到相应照片页面的一个链接。**

下面先来创建页面，然后再回来完成所有链接。

为照片创建单独的页面

首先，在“mypod”文件夹下创建一个名为“html”的新文件夹，存放所有这些单独的页面:



现在我们要为每个照片创建一个HTML文件。如果照片名为“seattle_video_med.jpg”，那么相应的HTML文件就命名为“seattle_video_med.html”，以保持一致。在每个HTML文件中，要有一个标题来描述这个照片，后面是照片本身。下面是第一张西雅图照片的HTML。所有其他页面也采用同样的格式:

```
<html>
  <head>
    <title>myPod: Seattle Ferry</title>
    <style type="text/css"> body { background-color: #eaf3da; } </style>
  </head>
  <body>
    <h1>Seattle Ferry</h1>
    <p>
      <img src="../photos/seattle_video_med.jpg" alt="A video iPod on the ferry">
    </p>
  </body>
</html>
```

页面标题，应当能够描述这个照片。

下面依然是我们之前提供的成品CSS，只是为了让页面保持一致的颜色。

这里为页面提供一个描述性的标题。

这里是<img>元素，指向大照片“seattle_video_med.jpg”。下面为这个图像提供一个描述性的alt属性。

注意我们需要在相对路径中使用“..”，因为“html”文件夹是“photos”文件夹的“兄弟”，所以使用相对链接时必须进入上一层文件夹，然后再向下进入“photos”。

如果查看本章示例文件中的“html”文件夹，则你会发现其中已经提供了所有的照片页面，但有一个页面除外——对应“seattle_downtown.jpg”的页面。在“html”文件夹中创建一个名为“seattle_downtown.html”的页面，做个测试。继续学习下面的内容之前先完成这个工作。如果遇到问题，则可以参考本章最后给出的答案。

那么，怎么链接到图像呢？

你已经有了大照片，小缩略图，还有一组HTML页面来显示单个照片。现在需要把它们综合在一起，让“index.html”中的这些缩略图链接到“html”文件夹中的页面。不过怎么做到呢？

要链接到一个图像，需要把<img>元素放在<a>元素中，就像这样：

这是一个<a>开始标记，放在<img>元素前面。

href链接到对应照片“seattle_downtown.html”的新页面（在“html”目录中）。

这是对应“seattle_downtown.jpg”缩略图的<img>元素，就像在“index.html”文件中一样。

```
<a href="html/seattle_downtown.html">
  <img src="thumbnails/seattle_downtown.jpg"
       alt="An iPod in downtown Seattle, WA">
</a>
```

这是结束<a>标记。

<img>元素直接嵌套在<a>元素中。

一旦把<img>元素放在一个<a>元素中，浏览器就会把这个图像当作一个可单击的链接。单击这个图像时，浏览器会获取href中的页面。

为“index.html”增加图像链接

这是最后一步了。你还需要在“index.html”文件中用<a>元素包围各个缩略图的<img>元素。要记住，各个<a>元素的href应当链接到“html”文件夹中包含更大图像的页面。要确保你的链接、缩略图和页面能正确对应。

下面是完整的“index.html”文件。你只需要增加用灰色标注的HTML。

```
<html>
   <head>
      <title>myPod</title>
      <style type="text/css">
         body { background-color: #eaf3da;}
      </style>
</head>
<body>

   <h1>Welcome to myPod</h1>
   <p>
      Welcome to the place to show off your iPod, wherever you might be.
      Wanna join the fun? All you need is any iPod, from the early classic
      iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
      iPod Video, and a digital camera. Just take a snapshot of your iPod in
      your favorite location and we'll be glad to post it on myPod. So, what
      are you waiting for?
   </p>

   <h2>Seattle, Washington</h2>
   <p>
      Me and my iPod in Seattle! You can see the
      Space Needle. You can't see the 628 coffee shops.
</p>

<p>
<a href="html/seattle_video_med.html">
   <img src="thumbnails/seattle_video_med.jpg" alt="My video iPod in Seattle, WA">
</a>
<a href="html/seattle_classic.html">
   <img src="thumbnails/seattle_classic.jpg" alt="A classic iPod in Seattle, WA">
</a>
<a href="html/seattle_shuffle.html">
   <img src="thumbnails/seattle_shuffle.jpg" alt="A iPod Shuffle in Seattle, WA">
</a>
```

增加缩略图链接

```
<a href="html/seattle_downtown.html">
    <img src="thumbnails/seattle_downtown.jpg" alt="An iPod in downtown Seattle, WA">
</a>
</p>

<h2>Birmingham, England</h2>
<p>
    Here are some iPod photos around Birmingham. We've obviously got some
    passionate folks over here who love their iPods. Check out the classic
    red British telephone box!
</p>

    <p>
    <a href="html/britain.html">
      <img src="thumbnails/britain.jpg" alt="An iPod in Birmingham at a telephone box">
    </a>
    <a href="html/applestore.html">
      <img src="thumbnails/applestore.jpg" alt="An iPod at the Birmingham Apple store">
    </a>
    </p>
  </body>
</html>
```

用一个<a>元素包围每一个缩略图图像。要当心，每个链接里的href要正确！

把这些<a>元素增加到你的“index.html”文件。保存文件，然后在浏览器中加载页面，感受一下你的myPod网站吧!

there are no Dumb Questions

问：用<a>元素包围文本时，会有一个下划线。为什么图像没有相应的标志呢？

答：确实，Internet Explorer会在图像周围加一个边框，显示这是一个链接（我们的浏览器是Safari，它不会这么做）。如果你的浏览器会在链接图像周围加一个边框或者在下面加一条线，但你并不喜欢这样，可以再等几章，后面会教你如何用CSS来改变这种效果。还要注意，鼠标经过图像时，光标会改变，指示你可以单击这个链接图像。大多数情况下，用户都可以通过上下文和鼠标光标了解到一个图像是链接（即使图像周围没有边框）。

问：能不能直接链接到JPEG图像，而不是那些HTML页面？我想浏览器肯定很聪明，能直接显示图像。

答：你说的没错，确实可以直接链接到图像，就像这样：...。如果这样做，单击链接时，浏览器就会在一个空页面显示这个图像。不过，一般认为直接链接到一个图像并不是一个好的做法，因为通常你需要为所显示的图像提供一些上下文。

好主意。实际上，我们已经有一个myPod logo。

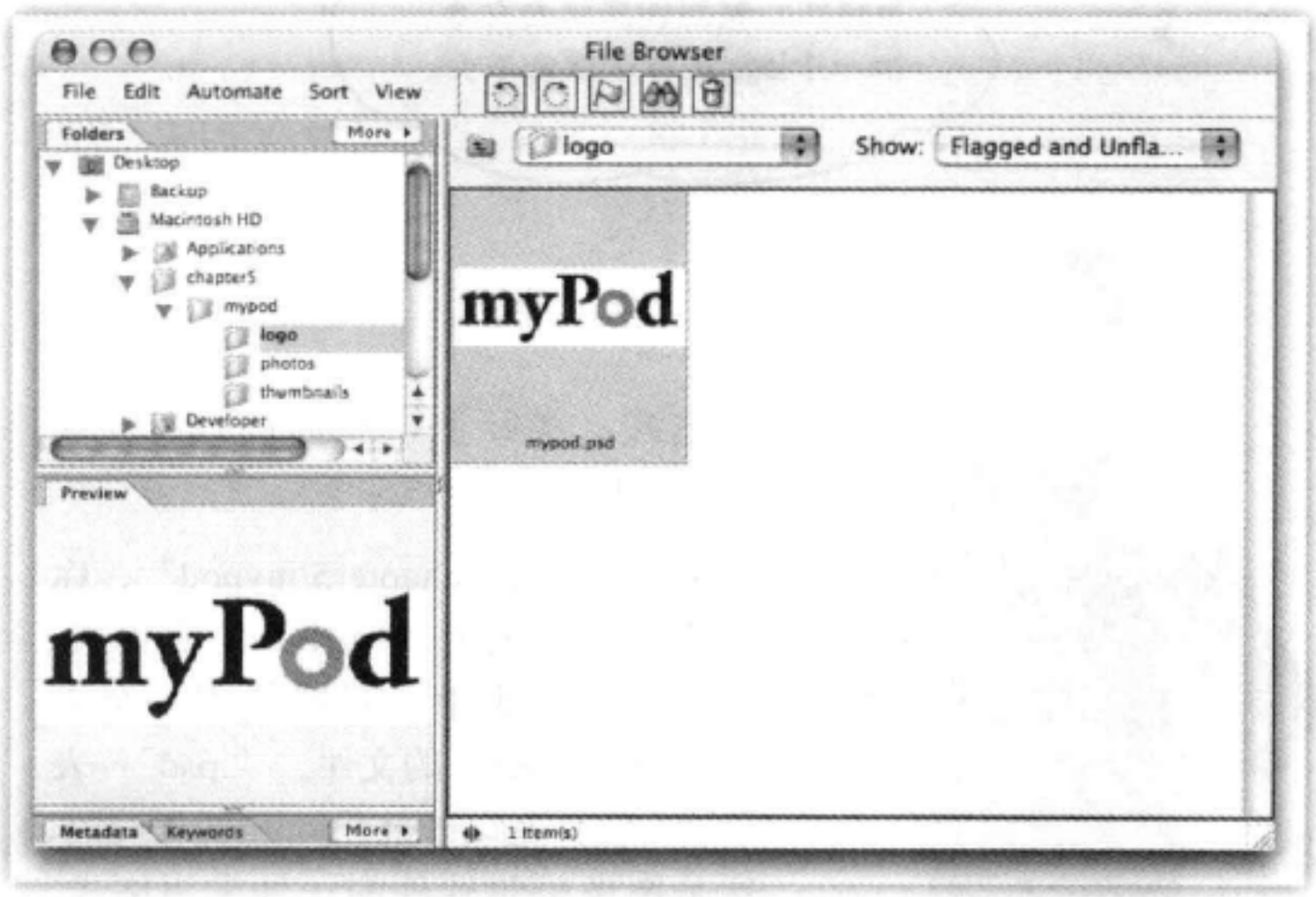
再来看文件夹“chapter5/mypod”，你会找到一个名为“logo”的文件夹。在这个文件夹中，可以找到一个名为“mypod.psd”的文件。“.psd” 表示这个文件是以Photoshop格式保存的，这是图像编辑软件的一种常用格式。不过Photoshop格式文件主要用于处理数字图像，而不是用于Web页面，所以我们必须做些工作，由它得到一个为“为Web准备的”图像。

很多照片编辑应用都支持.psd文件，所以即使你没有Photoshop Elements，也可以继续读下面几页。如果你的应用无法打开“.psd”文件，则可以在“logo”文件夹中找到每一步的图像。

打开myPod logo

下面来看myPod logo：在Photoshop Elements中打开“chapter5/mypod/logo”文件夹中的“mypod.psd”：

在“chapter5/mypod”文件夹中可以找到“logo”文件夹。



如果你的照片编辑应用无法打开这个文件，也请跟着我们继续学习，这些原则也同样适用于其他格式。

进一步分析……

不错的logo。这里有一些文字，还有两个圆，一个灰色，另一个白色（这显然是受了iPod上经典的单击式触摸转盘控件的启发）。

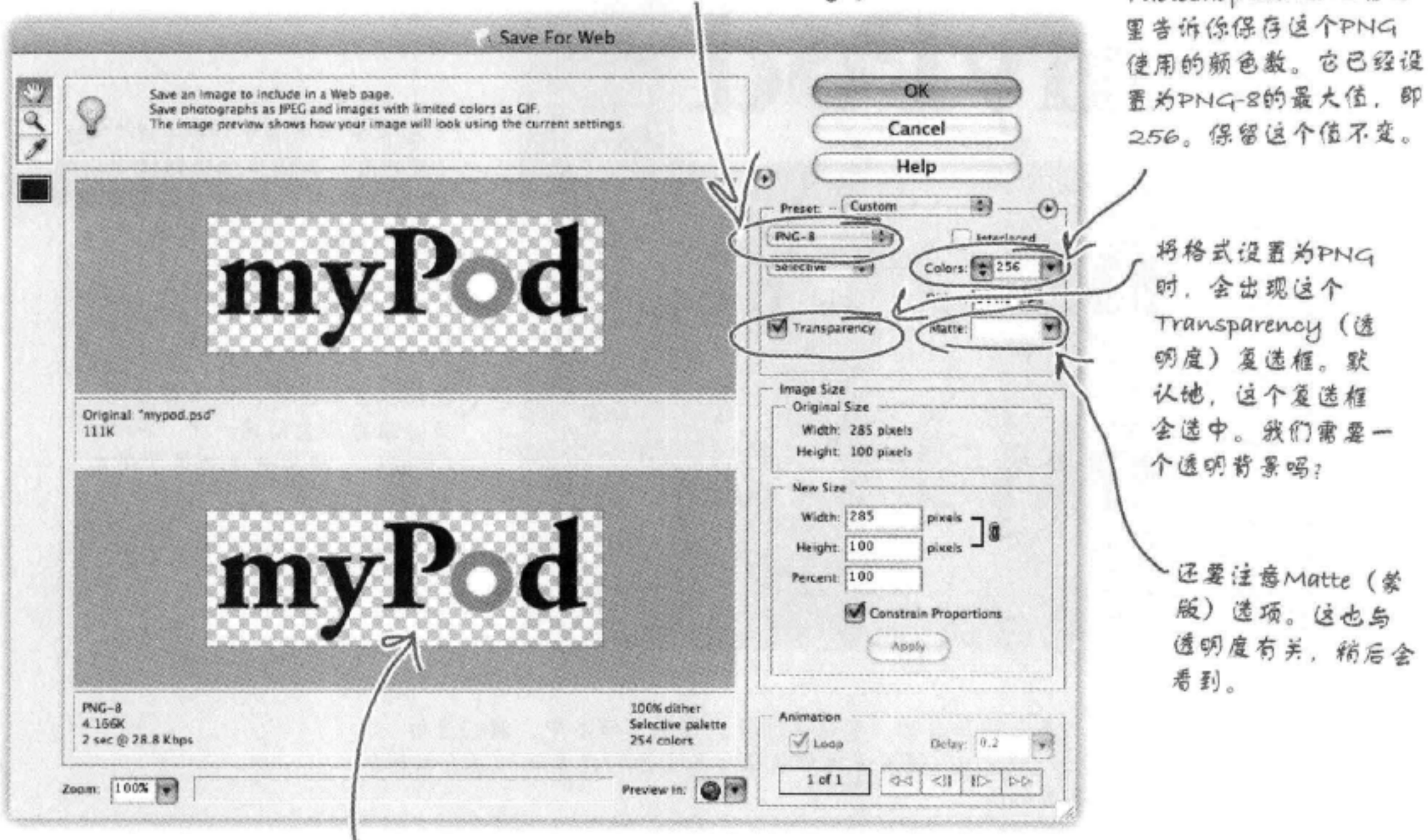
不过，背景里的棋盘图案是什么？大多数照片编辑应用都会以这种方式告诉你这个区域是透明的。为logo选择图形格式时要记住这些……

只要看到这种棋盘图案，就说明图像中有一个透明区域。

要使用什么格式？

你已经知道，在决定如何保存这个图像时有几种选择：可以使用JPEG、PNG或GIF。这个logo只使用了3种颜色，另外有文本和一些几何形状。根据你对PNG和GIF这两种格式的了解，你可能已经倾向于PNG或GIF了。这两种格式都可以。图像质量相同时，PNG文件可能稍小一点，所以我们选择PNG。另外，因为我们只有3种颜色，所以完全可以使用只支持256种颜色的PNG-8，使用这种格式可以让文件大小进一步缩小。

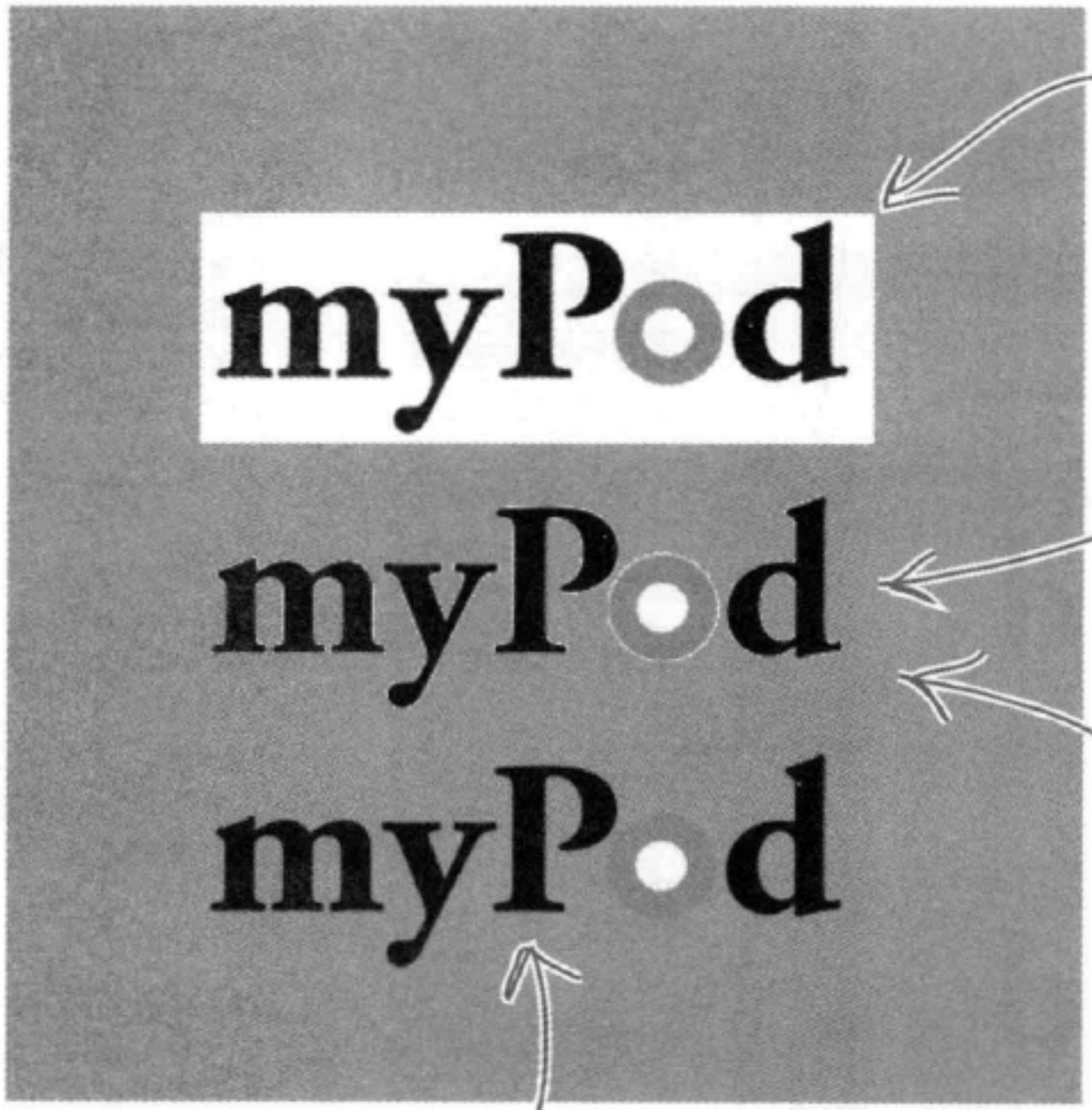
因此，选择File（文件）菜单下的“Save for Web”（保存为Web格式）选项，再在格式下拉框中选择PNG-8。你会看到更多选项。下面来看一下……



透明，还是不透明？这是个问题……

myPod logo要放在一个淡绿色背景中，所以你可能认为透明比较好，是吗？嗯，下面来比较一下，看看使用“Save for Web”（保存为Web格式）对话框中的几种选项时logo看起来怎么样：

下面用3种不同方式保存这个logo，并在一个有绿色背景的Web页面中显示。



没有透明度，情况很糟糕。很显然，白色背景在绿色页面上很不合适（不过，在白色页面上可能还可以）。

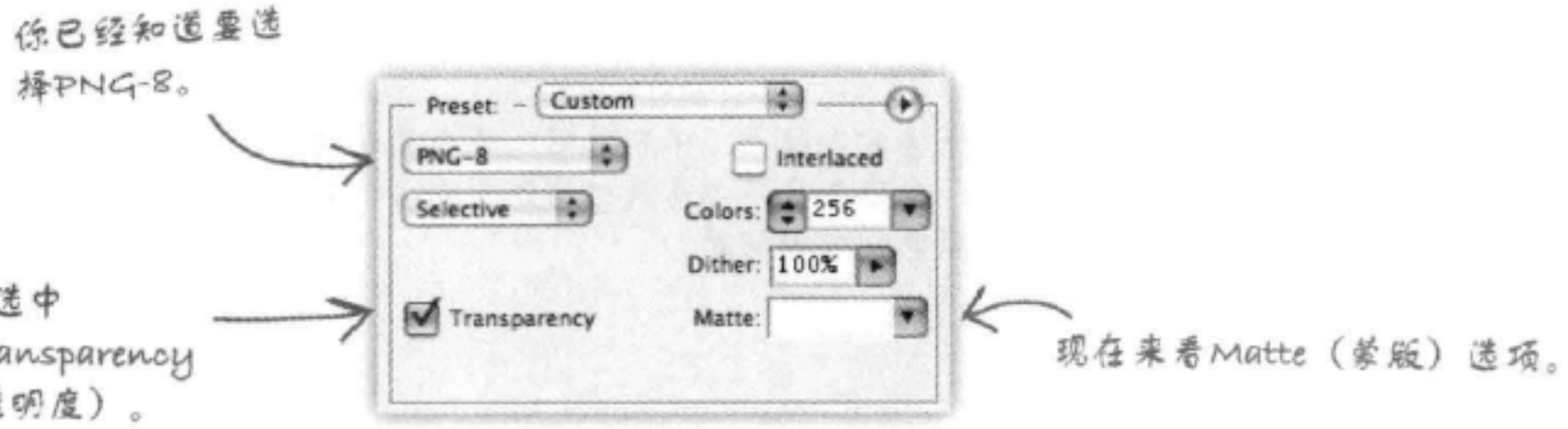
下面来看如果选中Transparency（透明度）并保存会得到什么。好一些……不过，logo中字母周围的那些白色“光晕”是怎么回事？

之所以会有光晕，这是因为照片编辑应用会创建一个“matte”（蒙版），根据背景颜色柔化文本边缘。不过，应用在这个logo上时，它假设是根据一个白色背景柔化文本边缘。

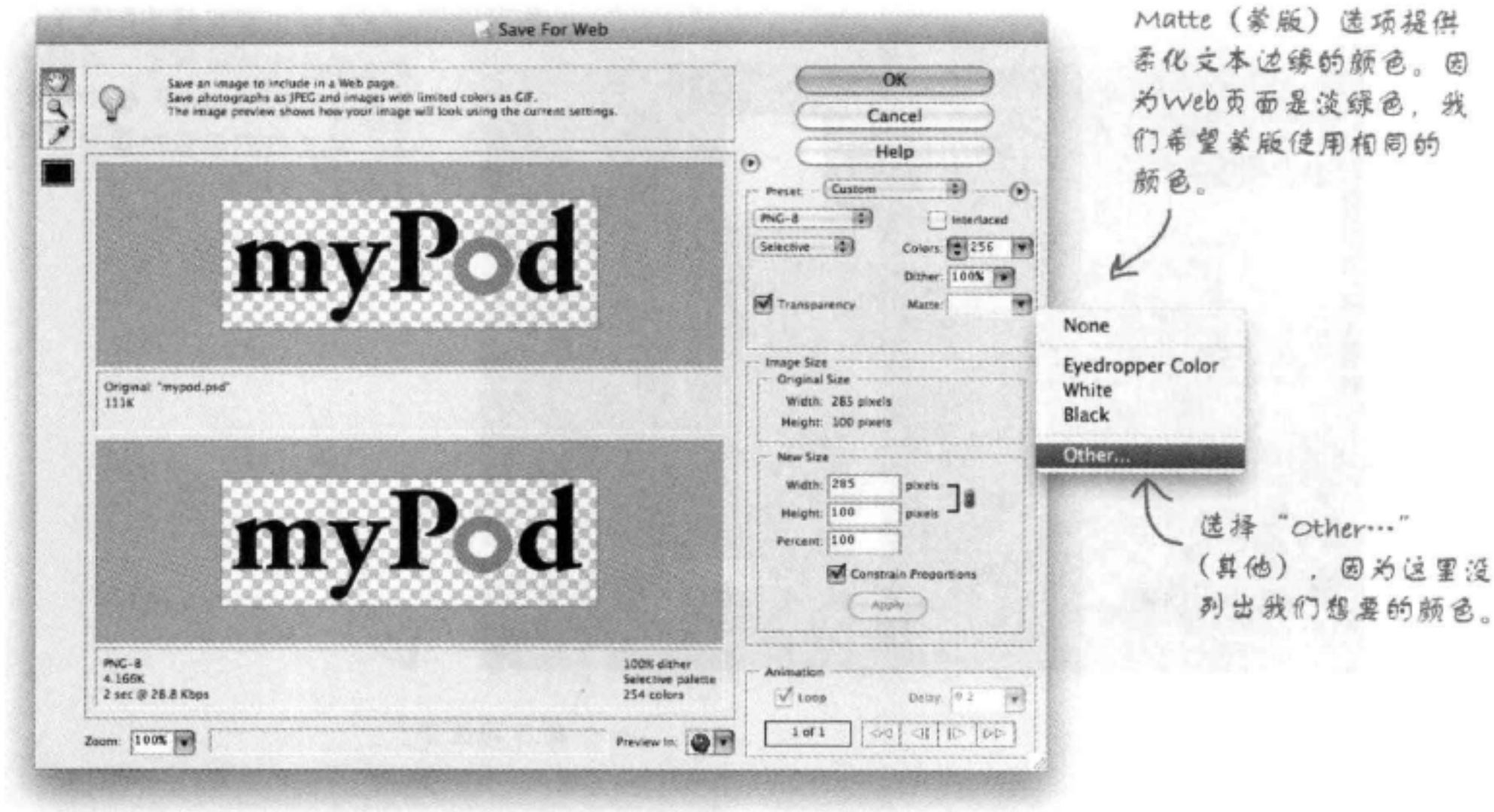
哈，这才是我们想要的，看起来很棒。在这个版本中，我们告诉Photoshop Elements使用绿色背景在文本周围创建蒙版。怎么做到的？下面就来告诉你。

保存透明PNG

你知道我们想要一个PNG版本的透明logo，另外还知道我们需要使用一个蒙版避免文本周围出现光晕。下面来分析“Save for Web”（保存为Web格式）对话框中的PNG面板。



Matte（蒙版）选项允许你为文本周围的蒙版选择颜色，我们希望蒙版颜色就是Web页面的背景色。



等一下，Web页面背景色到底是什么？

还记得myPod “index.html”文件中的成品CSS吗？这个CSS就是用来将页面的背景色设置为淡绿色，从这里可以得到颜色：

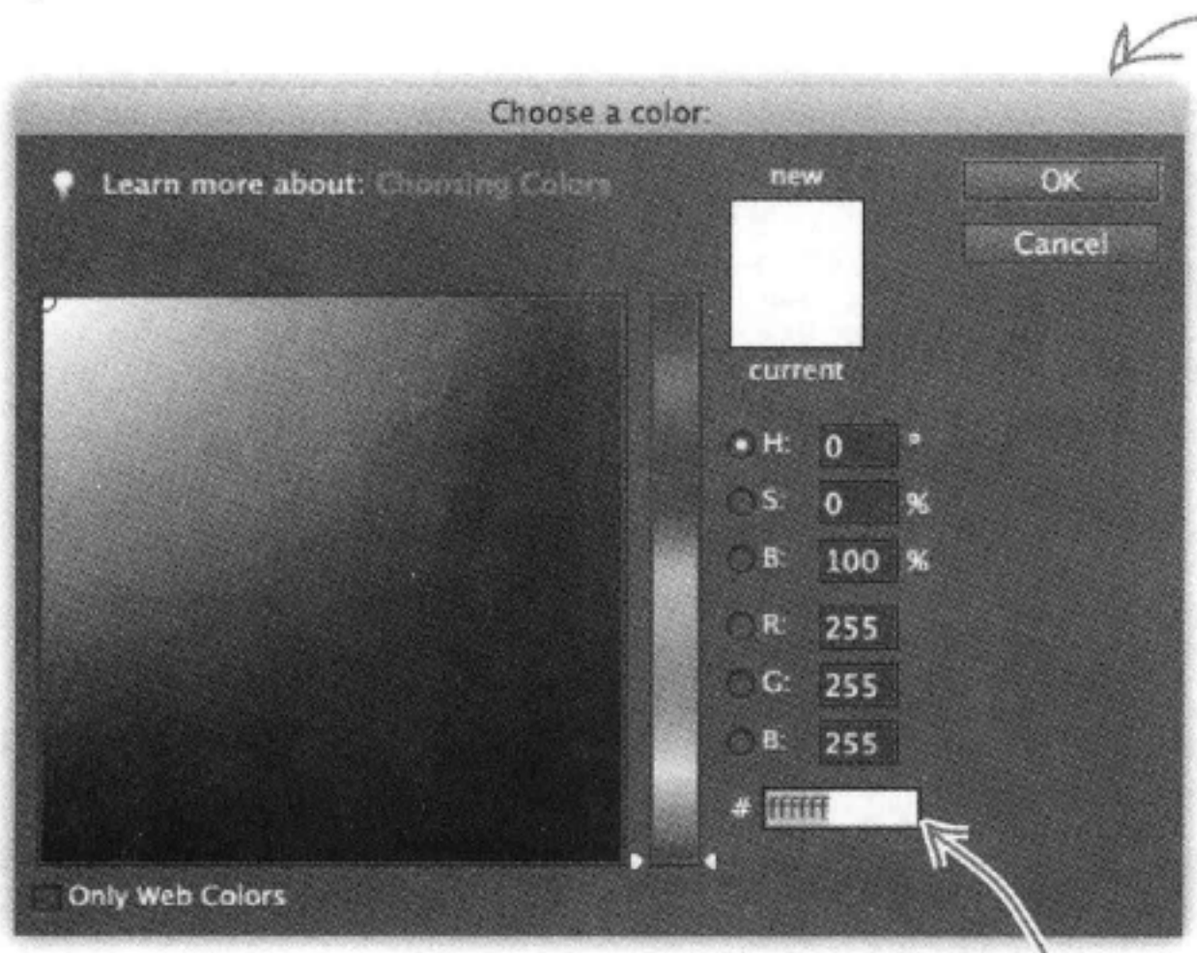
```
<style type="text/css">
   body { background-color: #eaf3da; }
</style>
```

这里就是背景色。

什么？你说这是淡绿色，不可能吧？现在先记住我们说的，后面几章还会再来讨论这个内容，并解释有关颜色的种种问题。

设置蒙版颜色

单击Matte（蒙版）下拉菜单并选择“Other…”（其他）菜单项时，Photoshop Elements会打开Color Picker（颜色选取工具）对话框。

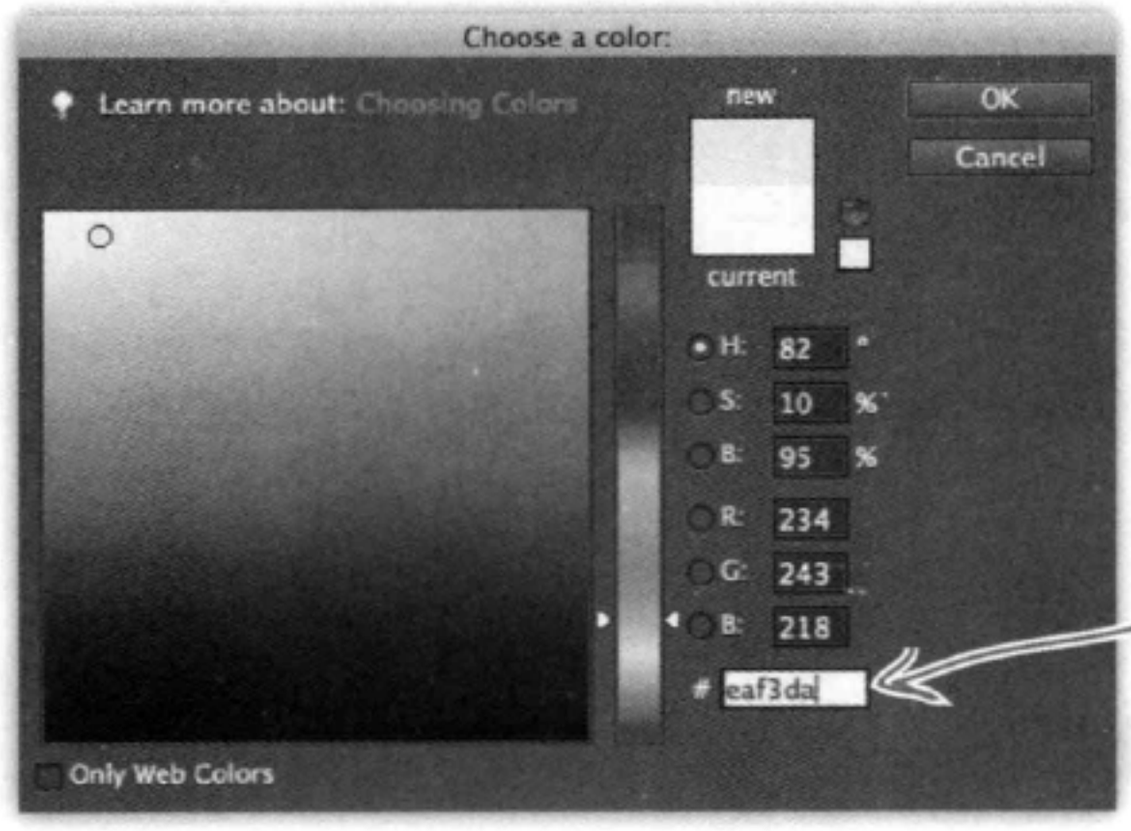


颜色选取工具提供了很多不同方法来选择蒙版颜色。我们希望把它设置为Web页面的背景色，而且我们已经知道页面背景色是eaf3da……

……输入到这里。

设置蒙版颜色（续）

把颜色“eaf3da”输入到“颜色选取工具”对话框。你会看到颜色变成了myPod页面的背景色。

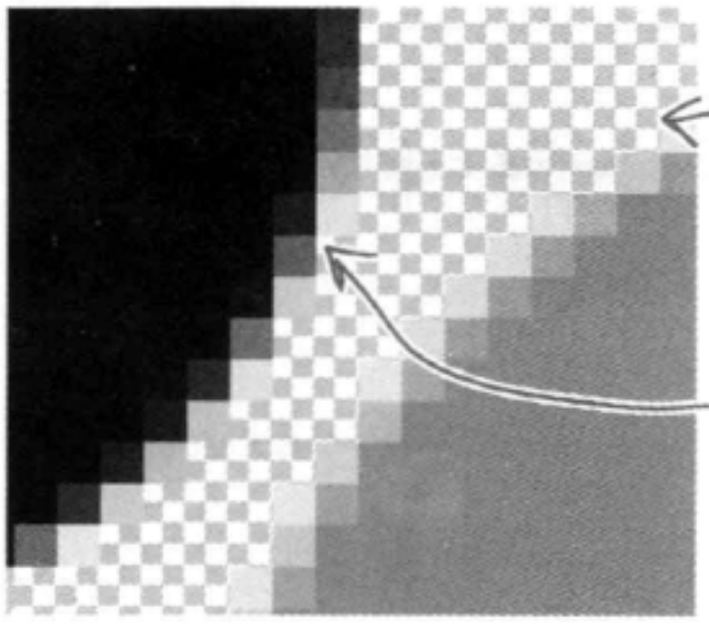


在这里输入这些字母。这个输入框专门设计用来输入Web格式的颜色。这些字母可以是大写，也可以是小写。大小写没有影响。

一旦将颜色输入到颜色选取工具中，就单击OK按钮，这个修改将应用到logo。

查看有蒙版的logo

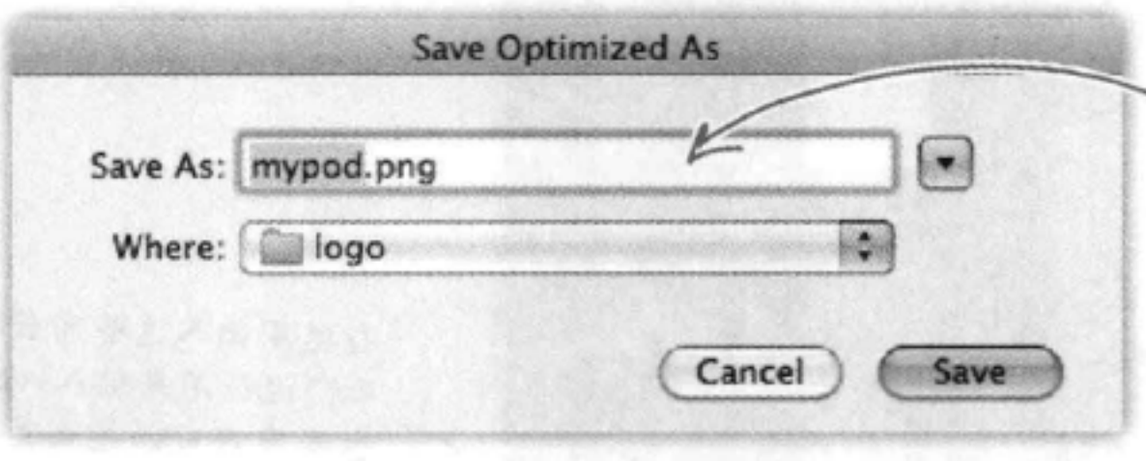
现在再在预览窗口中仔细查看这个logo。你会看到，Photoshop Elements已经在生硬的边线周围加上了一个淡绿色的蒙版，当myPod logo在Web页面中显示时，这个蒙版会让logo中的文本外观更柔和、更漂亮。



现在，仔细查看这个logo，你会看到蒙版与myPod Web页面的淡绿色背景一致。

保存logo

OK，你已经在“Save for Web”（保存为Web格式）对话框中完成了所需的全部调整，下面单击OK按钮，把这个图像保存为“mypod.png”。



Elements会自动将你的文件扩展名改为“.png”。把这个图像保存为“mypod.png”，放在“logo”文件夹中。

把logo增加到myPod Web页面

现在你要做的就是把这个logo增加到myPod Web页面。我们把它增加到最上面，让它出现在网站描述和iPod图像之上。这样一来，用户访问你的myPod页面上，第一眼就能看到它。

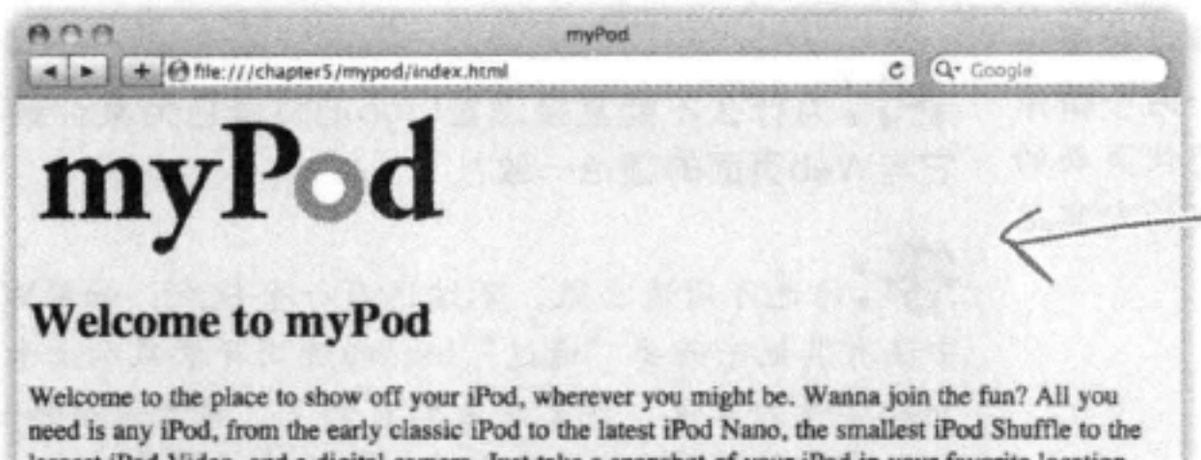
```
<html>
  <head>
    <title>myPod</title>
    <style type="text/css">
      body { background-color: #eaf3da;}
    </style>
  </head>
  <body>
    <p>
      <img src="logo/mypod.png" alt="myPod Logo">
    </p>

    <h1>Welcome to myPod</h1>
    .
    .
    .
  </body>
</html>
```

把这个logo图像增加到myPod Web页面的最上面。要记住，logo要使用正确的相对路径（在“logo”文件夹中），另外要增加一个alt属性来描述图像。

这里是“index.html” HTML的其余部分……

最后再来试一试

来做个测试吧！在浏览器中重新加载这个页面，看看你的myPod透明PNG logo效果怎么样。



不错啊，之前的努力没有白费。现在你的myPod Web页面上有了一个漂亮的logo。

there are no Dumb Questions

问：我真的需要了解所有这些关于图像格式的知识才能写出好的Web页面吗？

答：不。即使没有图像，你也完全可以构建出一流的Web页面。不过，图像是Web中很重要的一部分，所以对图像如何工作有所了解肯定会有帮助。有时仅仅是一两个图像就会使页面大不相同，也许一个好页面和一个最佳页面的差距就在于这一、两个图像。关于图像需要了解的内容很多，不过学起来并不难。

问：为什么需要柔化文本的边缘？

答：请看下面两个myPod logo的版本：

myPod

myPod

可以看到，上面的版本边缘很生硬，有锯齿，可读性较差。这是计算机屏幕上显示文本的默认方式。第二个版本使用了一种称为反锯齿（anti-aliasing）的技术来柔化它的边缘。经过反锯齿处理的文字在计算机屏幕上更可读，眼睛看起来也更舒服些。

问：什么情况下会用到蒙版呢？

答：反锯齿处理可以相对于背景色柔化边缘。如果把上一个问题中logo的第二个版本放在一个彩色背景中，则你会看到它会有白色边缘。Photoshop Elements中的Matte（蒙版）允许你指定要文本背景的颜色，所以柔化文本时，会根据这个颜色来完成柔化。

问：这种技术只适用于文本吗？

答：不，它还适用于图形中可能导致“锯齿”的所有线条。查看myPod logo中的圆，它也使用了蒙版。

问：为什么不能直接设置logo的背景色为某种颜色，让它与Web页面的颜色一致？

答：你也可以这么做，不过这有一个缺点：如果Web页面中还有其他东西要“透过”logo的透明背景显示出来，倘若logo背景为单色，就无法看到它们了。这里没有给出例子，不过我们讨论CSS时，你会看到这方面的例子。

问：在我创建使用蒙版的logo之后，如果背景色有改变则会怎么样呢？

答：如果背景色只是稍有调整，那么可能注意不到任何差别；不过，如果颜色有很大变化，就必须用新的蒙版颜色重新创建PNG。

如果把一个透明的图像放在Web页面中，则要确保这个图像的蒙版颜色与Web页面的背景色一致。

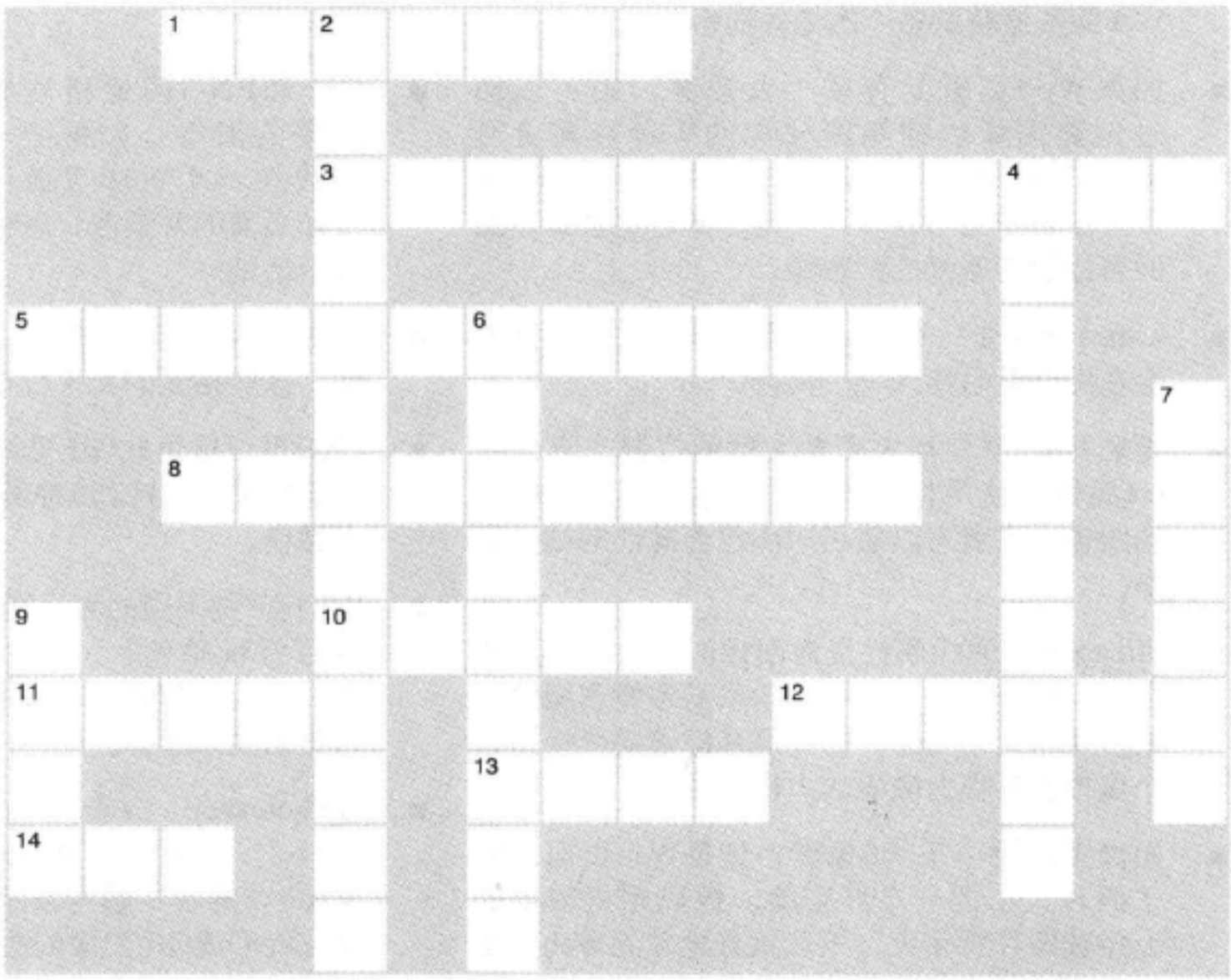
透明图像可以使用PNG或GIF格式。

BULLET POINTS

- 使用<img>元素在Web页面中放置图像。
- 浏览器对<img>元素的处理与其他HTML元素稍有不同。读取HTML页面之后，浏览器会从Web服务器获取各个图像并显示。
- 如果Web页面上有多个大图像，则可以通过创建图像的缩略图使你的Web页面更可用，下载也更快，缩略图是一些小图像（大图像的缩小版本），用户单击这些缩略图时可以看到原来的大图像。
- <img>元素是一个内联元素，这说明浏览器不会在图像前后插入一个换行。
- 要利用src属性指定图像文件的位置。可以在src属性中使用相对路径包含你自己的网站中的图像，或者可以使用URL包含其他网站的图像。
- <img>元素的alt属性是对图像的一个有意义的描述。在一些浏览器中，如果无法找到图像，就会显示这个描述，另外屏幕阅读器会使用这个属性为有视力障碍的人描述图像。
- 图像宽度要小于800像素，这是Web页面中关于照片大小的一个好经验。数码相机拍摄的大多数照片都太大，不能很好地放在Web页面中，所以需要调整它们的大小。
- 有很多照片编辑应用，Photoshop Elementsis就是其中之一，可以用来调整图像的大小。还可以使用很多免费的联机工具调整图像大小。可以在网上搜索“free online image editor”（免费联机图像编辑器）。
- 对于浏览器来说太大的图像会使Web页面很难使用，而且下载和显示都很慢。
- JPEG、PNG和GIF是Web浏览器广泛支持的3种图像格式。
- JPEG格式最适合保存照片和其他复杂图像。
- GIF或PNG格式最适合保存logo和其他包含单色、线条或文本的简单图形。
- JPEG图像可以按不同质量压缩，所以可以很好地权衡图像质量和文件大小，来满足你的需要。
- GIF和PNG图像格式允许建立一个有透明背景的图像。如果把一个有透明背景的图像放在一个Web页面中，图像后面的东西（如页面的背景色）就会透过图像的透明部分显示出来。
- GIF和PNG是无损格式，这说明相比于JPEG文件，这些格式的文件往往更大。
- PNG可以提供比GIF更好的透明度控制，而且不像GIF只支持256种颜色，PNG可以支持更多颜色。
- PNG有3种不同的大小选择：PNG-24 （支持数百万种颜色）、PNG-16（支持数千种颜色），以及PNG-8（支持256种颜色），可以根据需要来选择。
- 在Photoshop Elements中，使用“Save for Web”（保存为Web格式）对话框中的Matte（蒙版）颜色菜单来选择合适的颜色，柔化PNG或GIF图像的边缘。
- 图像可以用作指向其他Web页面的链接。要由图像创建一个链接，可以使用 <img>元素作为<a>元素的内容，将链接放在<a>元素的href属性中。

HTML填字游戏

该让你的右脑歇一歇了，让左脑开动起来：所有单词都与HTML有关，而且都在本章出现过。



横向

1. 利用JPEG，你可以控制这方面。
3. 大多数浏览器会以这种方式获取图像。
5. PNG和GIF有这个特性，而JPEG没有。
8. 用一支铅笔可以画多少英里。
10. Web服务器会为每一个 _____发出一个请求。
11.屏幕上最小的元素。
12. 使用Photoshop Elements对图像做这个处理。
13. 最喜爱的MP3播放器。
14. 更适合包含单色、线条和小文本的图像。

纵向

2. alt属性可以改善 _____。
4. 页面上的小图像。
6. 柔化文本边缘的技术。
7. 图像越大，传输时间就_____。
9. 更适合有连续色调的照片。

哪一种图像格式？

答案

祝贺：你当选为今天的“超级图像格式选择能手”。对下面的每个图像，请选择在Web上显示的最佳格式。

	JPEG	或 PNG	或 GIF
 有大量连续灰色阴影的照片。	☑	☐	☐
 只有两种颜色和一些文本，肯定是一个PNG或GIF。不透明？选择PNG的话，可以得到更小的文件。	☐	☑	☑
 有很多颜色的照片。肯定是一个JPEG或PNG。如果你想有透明背景，那就得使用PNG。	☑	☑	☐
 只是一个简单的黑白图标，这是一个PNG或GIF。如果需要它透明，可能希望对边缘做反锯齿处理，PNG在这方面可能更擅长。	☐	☑	☑
 这个图像介于它们之间。它包含大量连续颜色（适合采用JPEG），不过也算是一个几何图形（GIF），另外可能会以某种需要透明度的方式使用这个图像（PNG）。	☑	☑	☑

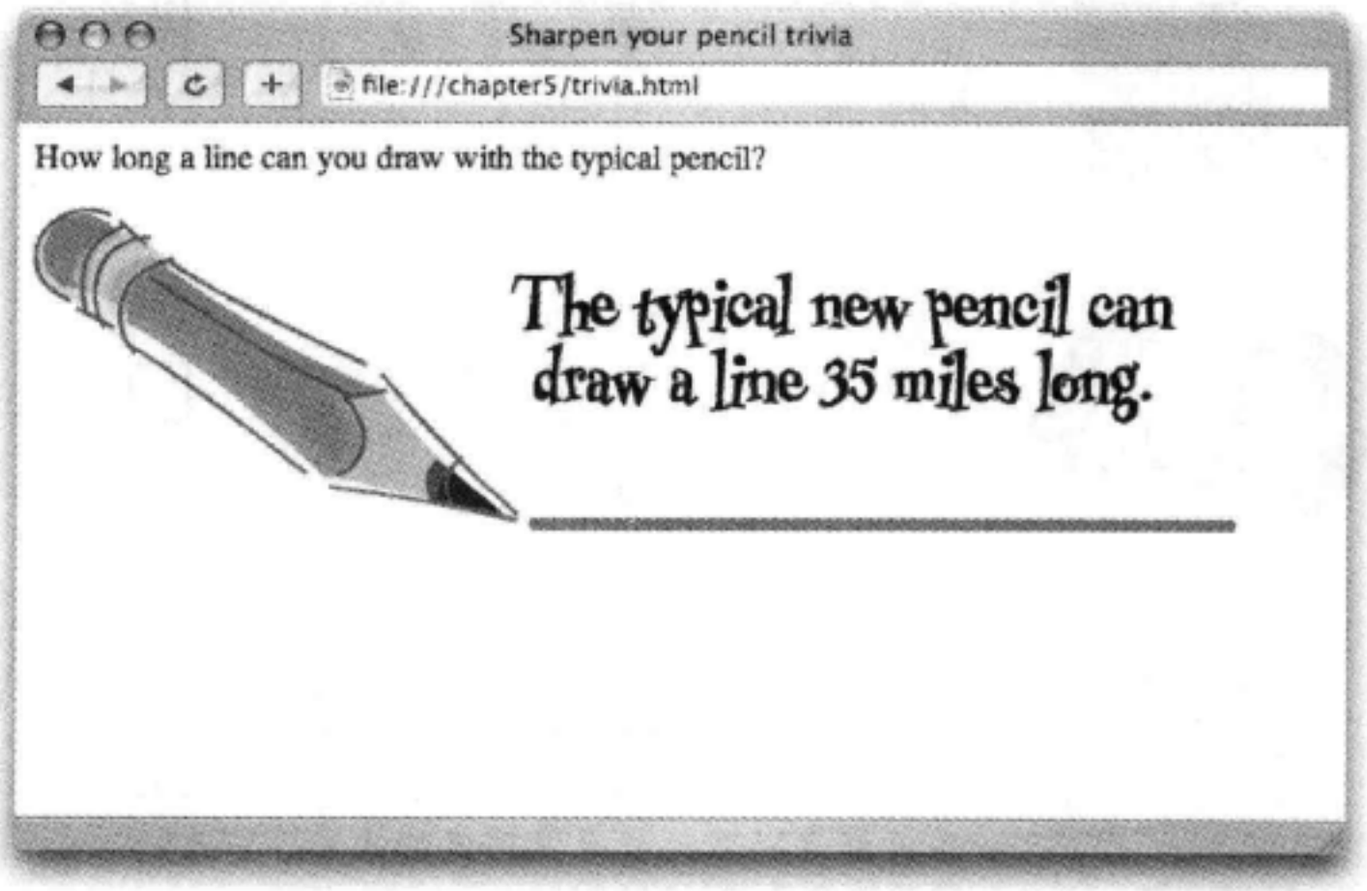
Sharpen your pencil Solution

这里有一个“练练笔”练习，它确实与铅笔有关（哦，还有图像）。这个练习提到这样一个问题：给你一根普通的新铅笔，如果用它画一条连续的线，则把整个铅笔用完，这条线会有多长？

这和图像有什么关系？要找出答案，先写一些HTML。这个问题的答案就包含在URL为http://wickedlysmart.com/hfhtmlcss/trivia/pencil.png图像中。你的任务是把图像增加到这个HTML，并找出答案。下面是我们的答案。

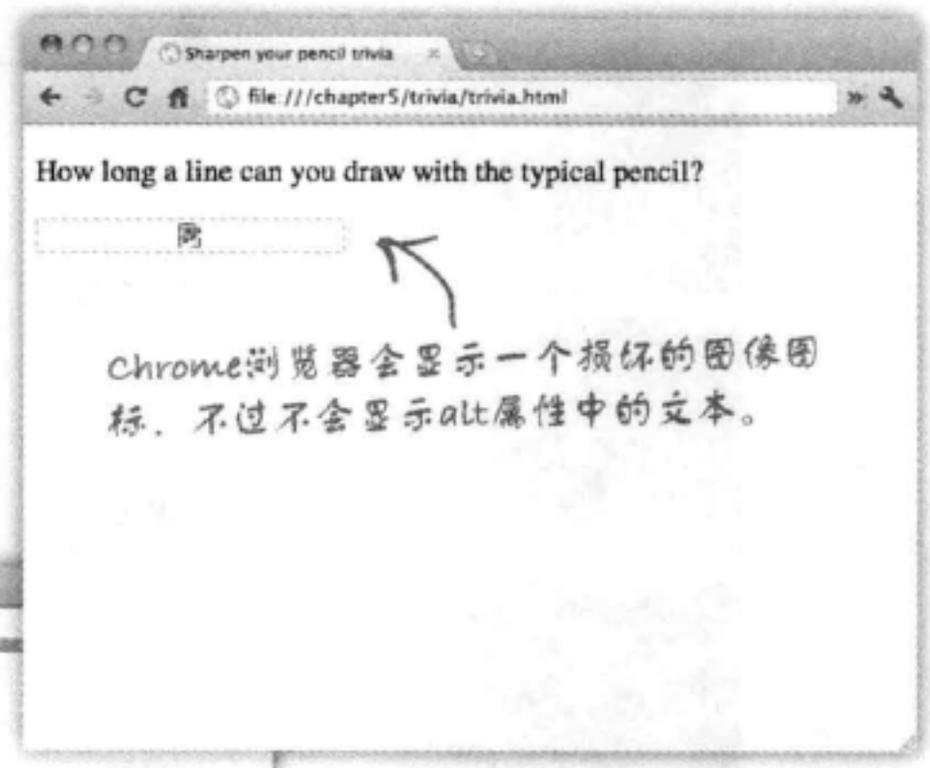
```
<html>
  <head>
    <title>Sharpen your pencil trivia</title>
  </head>
  <body>
    <p>How long a line can you draw with the typical pencil?</p>
    <p>
      <img src="http://www.headfirstlabs.com/trivia/pencil.gif">
    </p>
  </body>
</html>
```

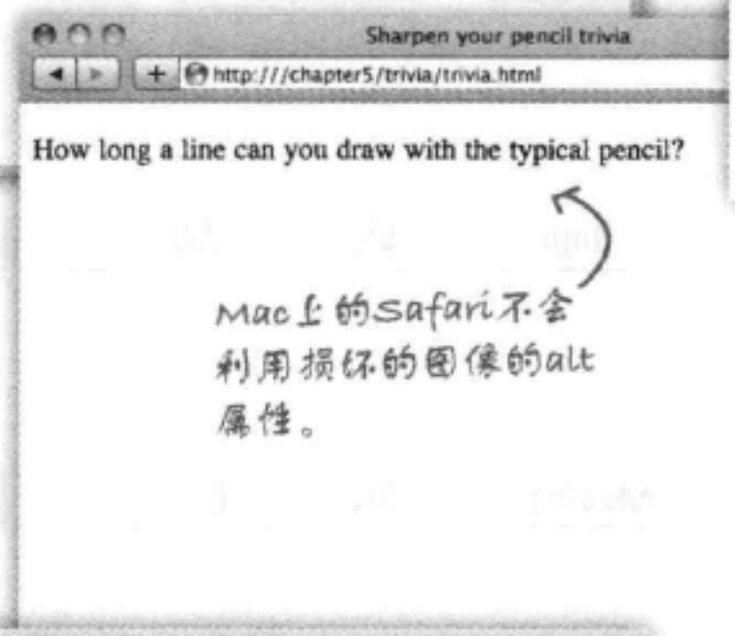
如果把图像放在这里，加载页面时你就会看到答案。

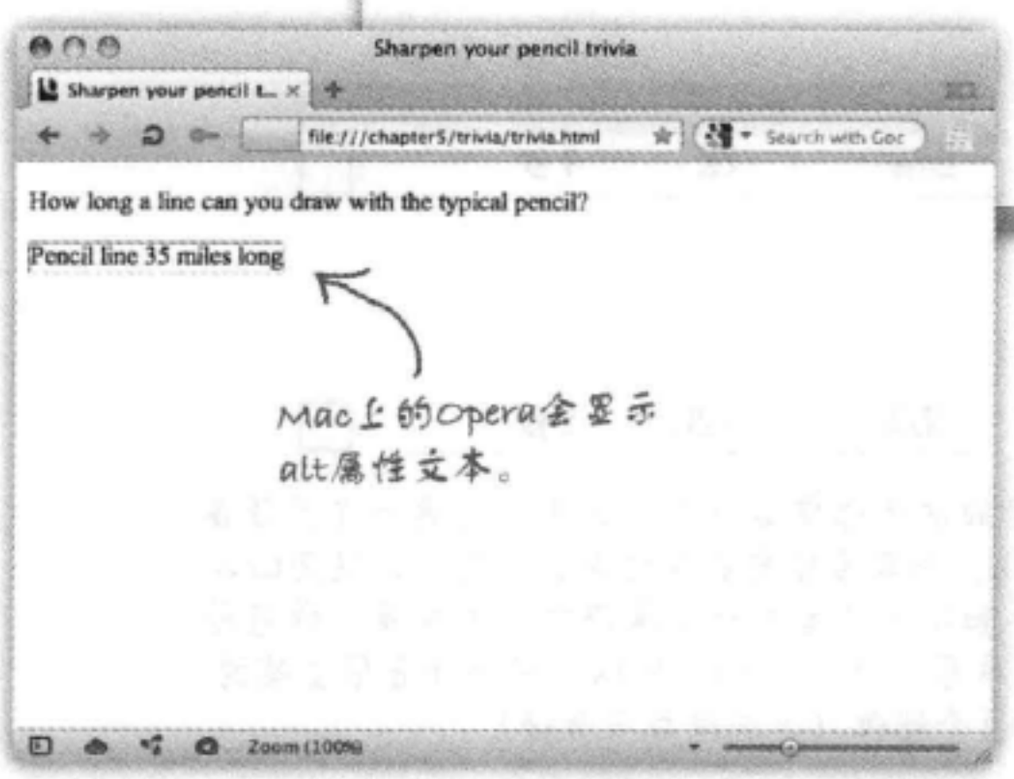


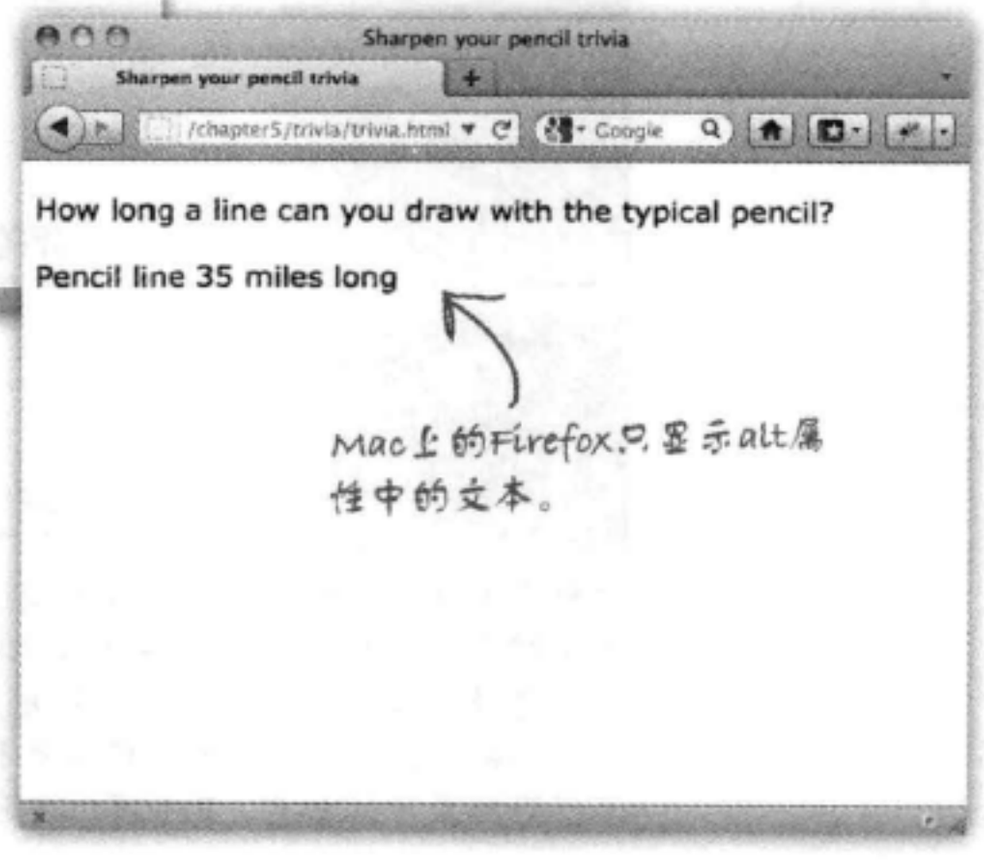
来源： http://www.papermate.com

以下是一些不同浏览器中无法正常显示图像时得到的结果。大多数情况下，浏览器都能使用额外的alt属性信息来改善显示。我们为什么关心这个问题呢？毕竟，这是Web页面中存在的一个错误，我们要修复它，对不对？嗯，在真实世界里，情况往往并不完美。有些东西会损坏，互联网连接可能在页面加载过程中断开，或者有视力障碍的用户需要听到图像里有什么，因为他们看不见。







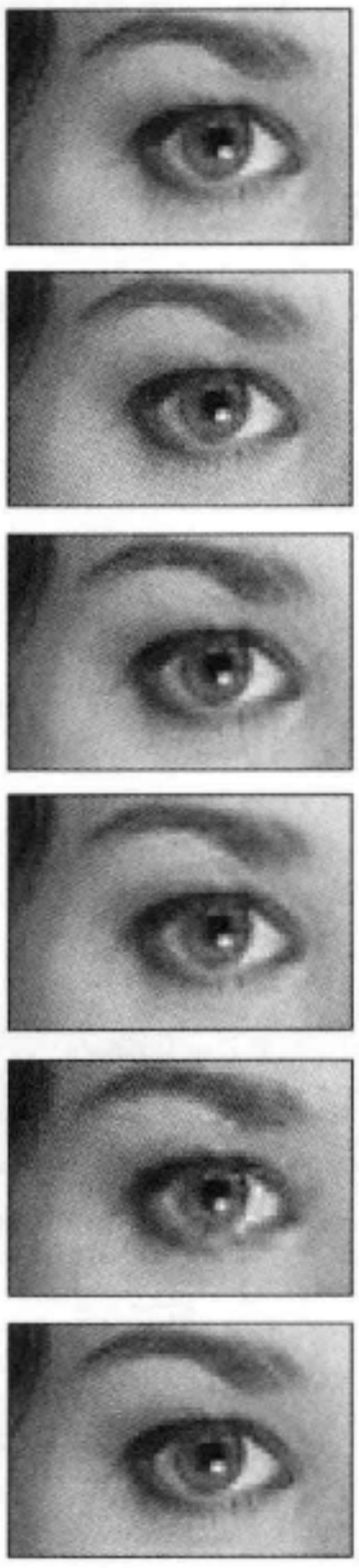


哪一种图像格式？

答案

这一次的任务：在Photoshop Elements中打开文件“chapter5/testimage/eye.jpg”。打开“Save for Web”对话框，选择JPEG的各种质量设置（Low、Medium、High等），再试试PNG-24和GIF，填写下面的空格。你会在图像下面的预览窗口中找到这些信息。完成后，确定对于这个图像哪种设置最合适。

注意，取决于你使用软件的哪个版本，得到时间可能有所不同。



注意到了吗？从JPEG Maximum变到Low时，图像质量在下降。

格式	质量	大小	时间	优胜者
PNG-24	N/A	32K	13秒	☐
JPEG	Maximum	21K	8秒	☐
JPEG	High	6K	3秒	☐
JPEG	Medium	3K	2秒	☑
JPEG	Low	2K	1秒	☐
GIF	N/A	22K	9秒	☐

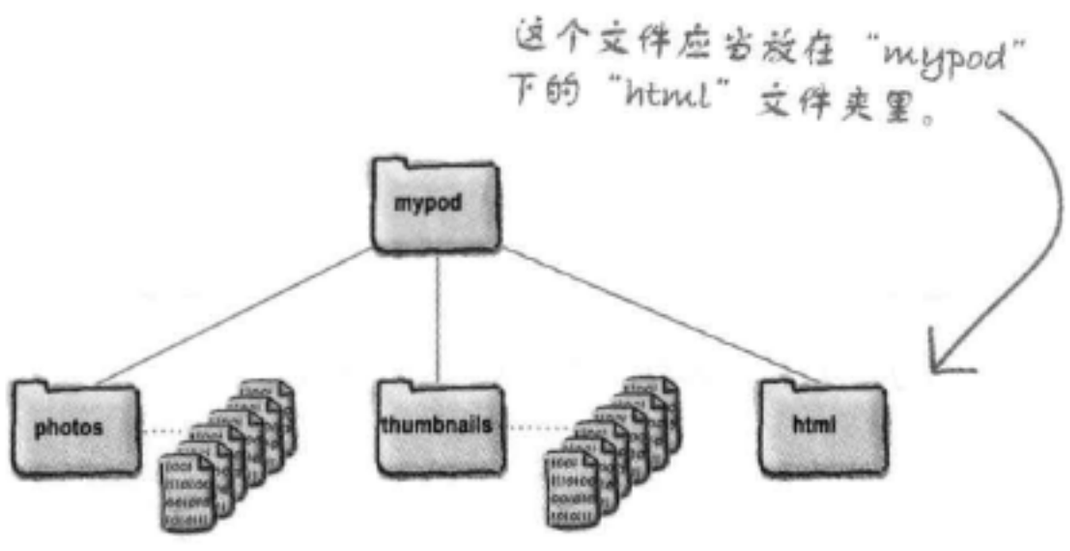
优胜者真的是Medium吗？不一定。这完全取决于你需要什么。如果你想要一个真正高质量的图像，那么你可能要选择Very High。如果希望有最快的网站，则可以使用Low。我们选择Medium的原因是，它是文件大小和图像质量的一个很好的折衷方案。你可能认为Low就足够了，或者可能认为有必要把质量上调到High。所以如何选择是很主观的。不过有一点可以肯定：PNG和GIF不太适合这个图像（这应该并不奇怪）。

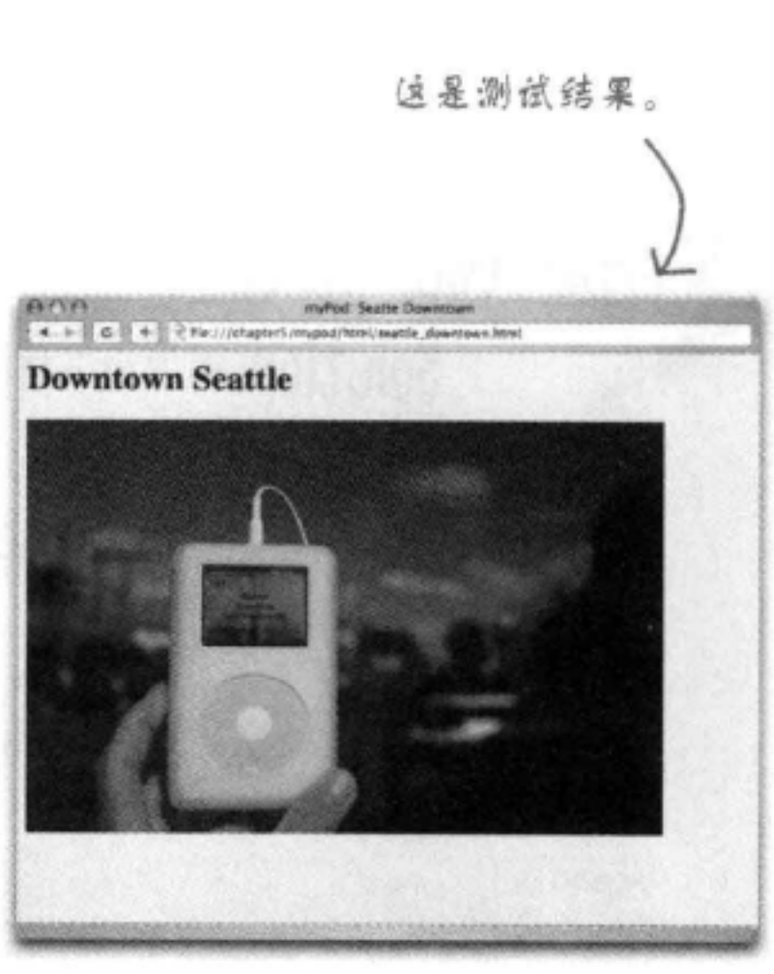
Exercise Solution

如果查看本章示例文件中的“html”文件夹，那么你会发现其中已经提供了所有的照片页面，但有一个页面除外——对应“seattle_downtown.jpg”的页面。在“html”文件夹中创建一个名为“seattle_downtown.html”的页面，做个测试。继续学习下面的内容之前先完成这个工作。下面是我们的答案。

这里给出了HTML，这个文件应当命名为“seattle_downtown.html”。

```
<html>
   <head>
      <title>myPod: Seattle Downtown</title>
      <style type="text/css"> body { background-color: #eaf3da; } </style>
   </head>
   <body>
      <h1>Downtown Seattle</h1>
      <p>
      <img src="../photos/seattle _ downtown.jpg" alt="An iPod in downtown Seattle, WA">
      </p>
   </body>
</html>
```





HTML填字游戏答案

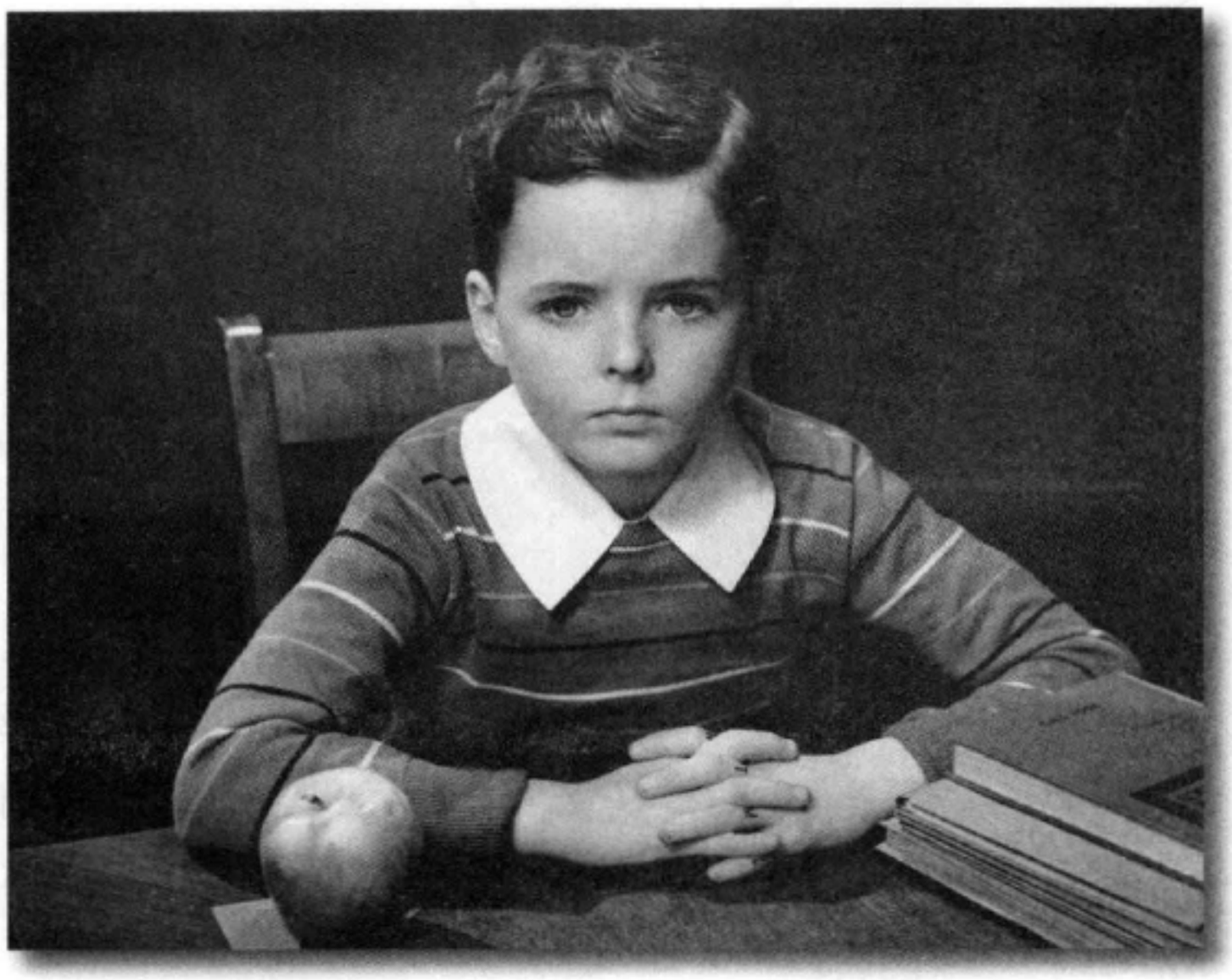
			[1]Q	U	[2]A	L	I	T	Y									
					C													
					[3]C	O	N	C	U	R	R	E	N	[4]T	L	Y		
					E									H				
[5]T	R	A	N	S	P	[6]A	R	E	N	C	Y			U				
				S		N								M		[7]L		
			[8]T	H	I	R	T	Y	F	I	V	E		B		O		
					B		I							N		N		
[9]J					[10]I	M	A	G	E					A		G		
[11]P	I	X	E	L		L				[12]R	E	S	I	Z	E			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

Sharpen your pencil

Solution 可以如下将图像“seattle.jpg”增加到文件“index.html”中。

```
<h2>Seattle, Washington</h2>
<p>
    Me and my iPod in Seattle! You can see rain clouds and the
    Space Needle.  You can't see the 628 coffee shops.
</p>
<p>
    <img src="photos/seattle.jpg" alt="My iPod in Seattle, WA">
</p>
```

严肃的HTML



关于HTML还要知道些什么？很不错，现在你已经掌握了HTML。实际上，是不是该转向CSS了？来学习如何让这些乏味的标记看起来更漂亮。在此之前，我们需要确保你已经掌握了足够的HTML知识，有能力应对激烈的竞争。不要误解我们的意思，没错，你一直在写一流的HTML，不过还需要了解额外的一些知识，才能真正建立“工业标准”的HTML。还要利用这个机会确保你使用的是最新、最棒的HTML标准，也就是HTML5。保证了这一点，就可以确保HTML在所有浏览器上（至少是你关心的浏览器上）能有更一致的显示，不仅如此，还可以在最新的iPhone、iPod之类的设备上（选择你最喜欢的设备）很好地显示。你的页面会加载得更快，可以很好地结合CSS，能够随着标准发展稳步走向未来。准备好了吗？这一章会让你从一个Web工匠变成真正的Web高手。

Jim：做好准备，应对重大时刻？

Frank：是啊，你知道的，要保证它完全合法，而且能适应HTML5。

Jim：我们的HTML很不错了……看这里，看它在浏览器里的表现。很漂亮，不是吗？

Joe：对，我也这么想……他只是想给我们找点事干。

Frank：说真的，虽然我不想承认，不过我觉得这一次老板说的没错。

Jim, Joe：嗯？

Frank：到目前为止，我们一直都忽略了这样一个事实，在这个领域有很多标准。甚至HTML也有很多版本，比如HTML 4.01，还有现在的HTML5。对我们来说，是不是该做的都做了？确实能保证符合HTML5吗？

Joe：喂，这意味着还要做更多工作。我们已经受够了。实际上，页面看起来够好的了,我甚至在一些更新的设备上都已经做过测试。

Frank：也许是吧，不过我的意思是说，我觉得这样反而能帮助我们减少将来的工作。

Joe：哦？真的吗？怎么会这样？

Frank：嗯，如果能确保HTML与当前最新标准保持一致，在标准发展过程中，我们就不用做那么多的修改。另外，我们还要确保所有其他方面没有问题；你知道的，这包括我们的语法等方面。有太多不同的浏览器，而且这些浏览器还有不同的版本，所以如果HTML中有错误，我们的页面在不同浏览器上显示时肯定会不一样。用CSS为HTML增加表现效果时，如果HTML有问题，则这些差别甚至会更严重。

Joe：这么说，通过确保遵循“标准”，为客户显示页面时就能少遇到麻烦，不能正确显示页面的问题也会少得多，是这样吗？

Frank：没错。

Jim: 如果这样能让客户少在半夜打来求助电话，听上去这个主意也还不坏。

Joe：好的，那么从哪里开始呢？难道我们现在没有遵循标准吗？我们的HTML有什么问题？

Frank：可能没什么问题，不过老板希望用最新的HTML5，所以我们需要明确正在使用HTML的哪个版本，如果不是HTML5，我们就得改为HTML5。另外，如果我们做到了这一点，使用CSS时日子也会好过得多。

如果你编写了正确的HTML，各种浏览器都能一致地显示你的页面，不过如果HTML中有错误，或者你在HTML中做了非标准的事情，则在不同浏览器中页面通常会有不同的显示。你觉得为什么会出现这种情况？

HTML简史

HTML 1.0～2.0

那可真是很早以前了。关于HTML需要知道的东西并不多，甚至在汽车后备箱里都能全部放下。页面不好看，不过至少已经支持超文本。没有人关心表现，几乎Web上的每一个人都有他们自己的“主页”。那时甚至课桌上的铅笔、曲别针和即时贴数量都被认为是“Web内容”（你肯定认为我们是在开玩笑）。

HTML 3

那时正经历漫长、残酷的“浏览器战争”。Netscape 和Microsoft都在试图争霸世界。毕竟，谁统治了浏览器，谁就能统治宇宙，是不是?

陷入这场争端的受害者是Web开发人员。在这场战争中，每个浏览器公司都在不断增加自己的专用扩展包，试图保持领先，军备竞赛就此展开。谁能坚持到最后？不仅如此，那个时候，通常你必须写两个单独的Web页面：一个用于Netscape浏览器，另一个用于Internet Explorer。真是很不好。

HTML 4

哈……浏览器大战终于结束了，万维网协会（World Wide Web Consortium）来解救我们了，我们亲切地称它为W3C。他们的计划是：创建一个唯一的HTML“标准”，让这个世界恢复平静。

这个计划的关键是什么？将HTML的结构和表现分解到两种语言，一种语言用于实现结构（HTML），另一种语言用于表现（CSS），另外要求获得最大利益的浏览器制造商采用这些标准。

不过他们的计划奏效了吗?

嗯，差不多吧……只是稍有些调整（见HTML 4.01）。

1989　1991　1995　1998

从这一章开始，我们的目标是写正确的HTML5。与往常一样，这个世界在不断变化，所以我们还会介绍今后的走向。

HTML 4.01

XHTML 1.0

HTML5

这段日子过得很惬意，HTML 4.01在1999年闪亮登场，成为接下来十年中HTML的“必备”版本。

4.01与4.0并没有太大变化，只是在一些方面做了些修补。不过，与HTML的早期阶段相比（那时可真是艰辛，就好像必须光脚在6英尺深的雪地里艰难跋涉，而且都是上坡路），HTML 4.01可以让我们晚上睡个好觉，因为我们知道，几乎所有浏览器（至少你关心的那些浏览器）都能很好地显示你的内容。

正当我们感觉安逸的时候，一个新兴事物开始引起所有人的注意。这个新兴事物就是XML。实际上，它让HTML开始心烦意乱，它们两个终于在“拉郎配式”的婚姻中不情愿地结合在一起，XHTML 1.0就此诞生。

XHTML承诺，由于它的严格，再加上它提供的一些新方法，只要遵循这个标准，Web的所有争端将就此平息。

唯一的问题是，大多数人们很讨厌XHTML。他们并不想要一种编写Web页面的新方法，只是希望改进HTML 4.01已有的特性。Web开发人员对HTML的灵活性更感兴趣，而不是XHTML的严格性。另外，这些开发人员越来越希望把时间用来创建更像是应用的Web页面，而不只是文档（稍后还会讨论Web应用）……

当然，由于没有得到大家的支持，这场婚姻的结局并不好，很快被HTML的新版本所取代，这就是HTML5。由于它支持HTML 4.01标准的大部分特性，而且还提供了一些新特性，可以体现Web新的发展，所以HTML5得到了大家的欢迎，这正是开发人员一直想要的。另外，基于一些新特性，如支持类似博客的元素、新的视频和图形功能，以及一大堆用来构建Web应用的功能，HTML5注定成为大家公认的标准。

说实话，HTML和XML的分手还是让很多人感到诧异，曾在一段时间让人们很困惑HTML5到底是什么。不过现在已经都清楚了，所以继续读下去，看看HTML5对你来说意味着什么，你将如何加入这个欢乐的世界。

1999 2001 2009 2012 ????

未来会怎样？我们会不会在会飞的汽车里工作，能不能只吃营养药片代替进餐？继续读下去，自己来找答案吧。

浏览器闪亮登场

本周访谈：
为什么你这么关心我用哪个HTML版本？

Head First：很高兴能请到你，浏览器。你知道的，“HTML版本”已经成为一个热门的话题。这是怎么回事？毕竟，你是一个Web浏览器。我把HTML交给你，你尽你所能显示这些HTML就可以了。

浏览器：当个浏览器可真不容易……有那多么的Web页面，而且很多还是用HTML的老版本写的，或者标记里可能还有错误。就像你说的，不管怎样，我的任务就是努力显示每一个页面。

Head First：那么，有什么问题呢？看来你做得不错啊。

浏览器：有些情况下是这样，不过，你有没有在不同的浏览器上查看你的页面？如果你使用比较老的浏览器，或者页面中有不正确的标记，就可能遭遇这样一种尴尬的情况：你的页面在一个浏览器上看起来很棒，但在另一个浏览器上可就不怎么样了。

Head First：真的吗？为什么会这样？难道你做的不一样吗？

浏览器：我们确实在做同样的事情，前提是我们显示的是正确、最新的页面。前面我说过，如果你把有问题的页面交给我，情况就很难讲了。告诉你原因吧：我们浏览器都听HTML规范的，它会告诉我们如何显示正确的HTML，但是如果是不正确的HTML，我们就不清楚该怎么做了。所以你会在不同的浏览器上看到截然不同的表现。

Head First：噢。那么有什么解决办法吗？我们确实希望页面能很好地显示。

浏览器：很容易。提前告诉我你在使用哪个版本的HTML。你会惊奇地发现，居然有那么多的页面没有这样做。另外要确保你的页面没有任何错误，你知道的，就是标记不匹配之类的错误。

Head First：我们怎么告诉你在使用哪个版本呢？特别是现在我们都要转向HTML5了。

浏览器：嗯，HTML5确实让问题简单一些了。

Head First：是吗？新版本的HTML有什么帮助？我原来以为另一个版本只会让问题更困难呢。

浏览器：嗯，语言的新版本确实会带来“成长的烦恼”，因为所有人都得跟上最新的标准。不过HTML5则有所简化，你能更容易地告诉我你在使用哪个版本的HTML。HTML5标准还明确指出了Web页面中可能出现的很多错误，所以所有浏览器都能更一致地处理这些错误。

Head First：噢，那么这是不是意味着，我们写HTML时不用再担心犯错误了，是吗？

浏览器：不是这样的！尽管我们能更好地处理错误，但是并不表示你就能偷懒。 你还是需要保证你的页面与标准一致，而且没有错误。如果不能做到，你可能会在不同浏览器上得到不一致的结果，另外别忘了还有那些移动设备上的浏览器。

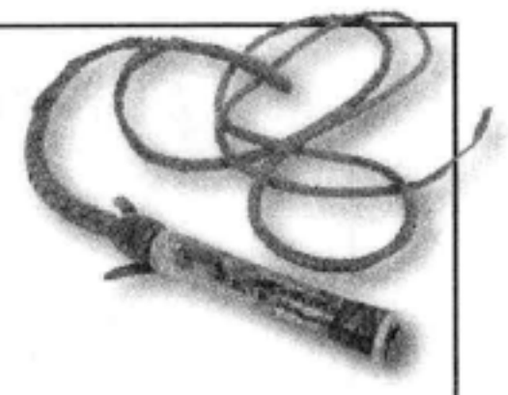
Head First：回答我前面的问题吧，怎么告诉你我们在用什么版本？

浏览器：嗯，原来这可麻烦呢……

Head First：嘿，拜托，观众们都可是大忙人，我们没时间了，快一点儿好吗！

浏览器：好的，好的，你可以用一个doctype告诉我你正在使用的HTML版本。只是这么一点点“标记”，要放在HTML文件的最上面。好了，既然时间到了，你自己试试看吧！

HTML考古



我们挖到了一些古老的HTML 4.01和XHTML 1.1页面。这些页面使用了一个doctype，放在HTML页面的最上面，告诉浏览器所使用的HTML版本。我们已经把doctype挑出来了，你可以好好分析一下。请看下面的doctype……

这里为浏览器指定这个页面的文档类型。

这表示这个<html>是页面中的根（第一个）元素。

这表示HTML 4.01标准是公共可用的。

这部分表示我们在使用HTML 4.01版本，另外这个HTML标记用英语编写。

可以把这些写在一行上，或者如果你愿意，也可以像我们一样加一个回车。不过要确保只能在加引号的部分之间按Enter键。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
                      "http://www.w3.org/TR/html4/strict.dtd">
```

注意这不是一个HTML元素。在开始的“<”后面有一个“!”，这说明它与众不同。

这指向一个文件，标识这个特定的标准。

类似于HTML DOCTYPE，这是一个公共文档类型。

这仍然是一个HTML版本（结合XML的版本）。

这个对应XHTML 1.1版本。

```
<!DOCTYPE html
      PUBLIC "-//W3C//DTD XHTML 1.1//EN"
             "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

另外它有一个URL指向XHTML 1.1的定义。

关于XHTML的更多内容，请参考附录。

Sharpen your pencil

我们认为与其告诉你HTML5的doctype定义，还不如让自己动手把它找出来。再来看下面的HTML 4.01 doctype定义：

记住，这是"html"的doctype。

这表示这个标准是公共可用的。

这部分说明我们在使用HTML 4.01版本，而且这个标记用英语编写。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
    "http://www.w3.org/TR/html4/strict.dtd">
```

这指向标识这个标准的一个文件。

要记住，doctype定义要放在HTML文件的最上面，告诉浏览器你的文档类型是什么，在这里就是HTML 4.01。通过使用doctype，浏览器可以更准确地解释和显示页面。

所以，下面要发挥你的推理能力了，你认为HTML5的doctype定义是什么样？把它写在下面（下一页具体介绍时你可以翻回来对照检查一下，不过现在不要偷看答案）：

你的答案写在这里。

新的、改进的HTML5 doctype

OK，准备好了吧。下面给出HTML5 doctype：

```
<!doctype html>
```

只有一行，别漏了。

真是很简单！

你的“练练笔”练习的答案是这样吗？你肯定会说，这个要简单得多。哇，现在每次需要一个doctype时甚至不用查找都能记住。

有些人记不住老的doctype，所以把它当作纹身纹在手掌上，对于他们，我们只能表示深深的同情。

问得好。不，这可不是印刷错误，下面来解释为什么：你很清楚原来的doctype相当复杂，版本号和丑陋的语法混杂在一起。不过，随着HTML5的到来，doctype已经得到简化，所以现在我们要做的就是告诉浏览器我们在使用“HTML”，不用再担心特定的版本号或语言，也不用指向某个标准。

真的能这样吗？我们可以只指定“HTML”，而不再需要其他的部分，真的吗？浏览器不需要其他信息吗？嗯，实际上，当浏览器看到：

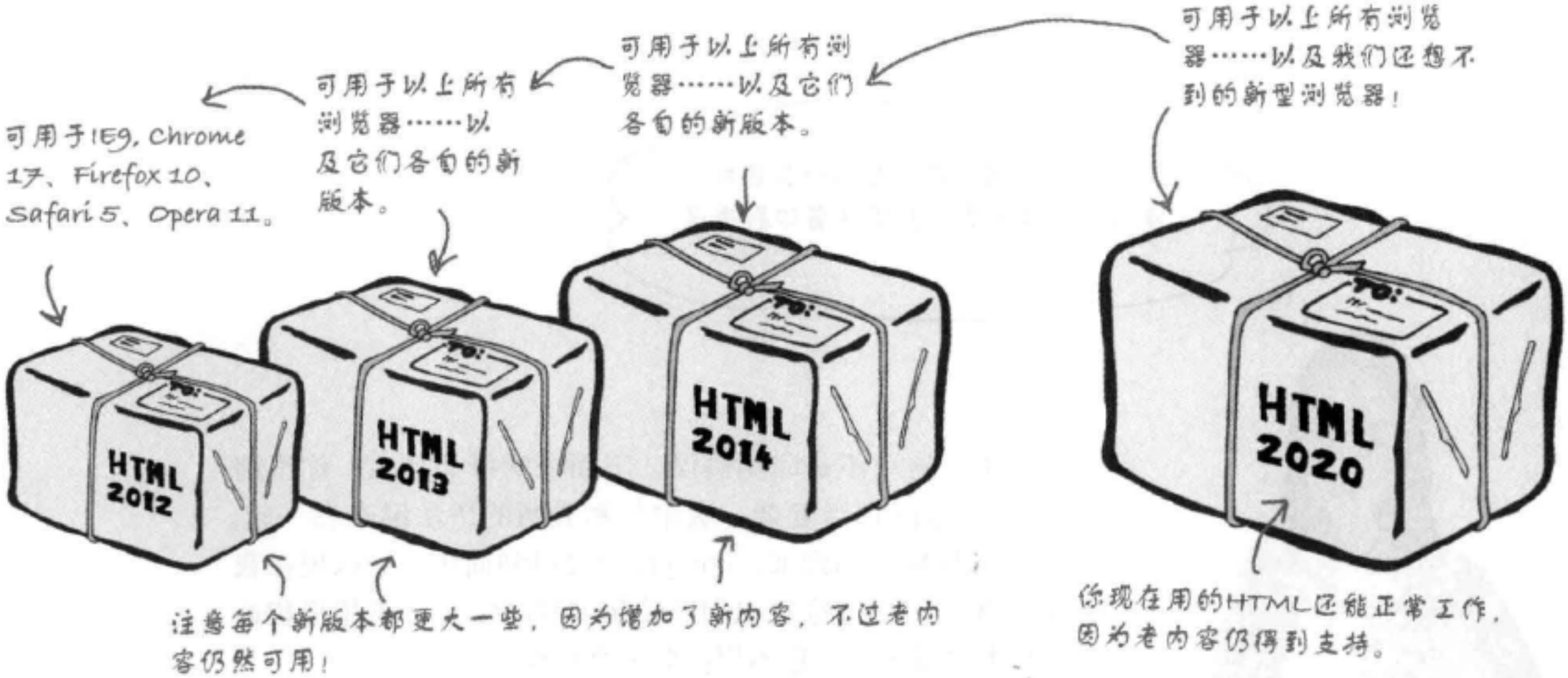
```
<!doctype html>
```

它就认为你在使用标准HTML。不再考虑版本号，也不考虑标准在什么位置；实际上，HTML标准将变成一个“活的标准”，这意味着它会根据需要继续发展和变化，不过不再有固定的版本号。现在你可能会想，“活的标准到底是什么意思？它是怎么工作的呢？”下一页就会告诉你……

HTML，新的“活标准”

你没听错……HTML不会再有版本6、7、8……制订标准的人已经把这个规范变成了一个活的标准，它会随着技术的发展形成相应文档。所以，不再有版本号。你甚至可以不再把它叫做HTML5，因为从现在开始它只是“HTML”。

现在，你可能在考虑这在实际中如何工作。毕竟，如果规范在不断变化，这对于可怜的浏览器来说意味着什么？另外，对于你们Web开发人员又意味着什么？这里的关键是向后兼容性。向后兼容性（backwards compatibility）表示我们可以继续向HTML增加新的内容，浏览器（最终）会支持这个新内容，不过它们还会继续支持原来的内容。所以，你今天写的HTML页面将会继续正常工作，甚至以后增加了新的特性之后也仍然能很好地工作。



there are no Dumb Questions

问：如果规范明天改变了怎么办？我该怎么做？

答：如果你今天编写了稳定可靠的HTML，即使明天规范改变了，比如增加了一个新元素，你还能像以往一样使用你的HTML。是否用这个新元素要由你来决定。

如果规范对你之前的工作有改变，比如改变了一个元素或属性的工作方式，那么浏览器会继续支持你使用的老方式，同时支持新的方式。这正是“向后兼容性”的含义。现在，如果对现有特性的改变尽可能少，而且你（作为一个Web开发人员）始终跟进最新规范，保持与规范一致，随着规范的改变修改你的页面，这当然很好，不过关键是在规范不断变化的同时，你的HTML还会继续正常工作。

问：规范到底是什么？

答：规范就是一个文档，指定了HTML标准是什么。也就是说，HTML中有哪些元素和属性等。这个文档由万维网协会（World Wide Web Consortium，简称为W3C）维护，不过任何人都可以为它做出贡献，都可以结这个标准如何发展进言。

增加文档类型定义

说的够多了，下面在HTML里加上这个doctype。

这就是doctype行。把它作为第一行增加到“lounge.html”文件中。

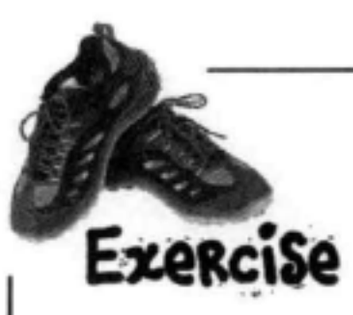
可以写为DOCTYPE或doctype，这两个都可以。

```
<!doctype html>
<html>
  <head>
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
       <img src="drinks.gif">
    <p>
       Join us any evening for refreshing
       <a href="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 server).
    </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="directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

测试doctype

对“chapter6/lounge”文件夹中的“lounge.html”文件完成上述修改，然后在你的浏览器中加载这个页面。

哇，没有任何差别。嗯，我们并不希望有差别，因为doctype所做的就是让浏览器明确知道你在使用HTML5。



Exercise

还要为“directions.html”和“elixir.html”文件增加这个doctype。来好好做个测试。就像“lounge.html”一样，你看不到任何差别（不过今天晚上你能睡得更好一点）。

HTML5闪亮登场

本周访谈：
HTML5有什么过人之处？

Head First：HTML5，你是HTML“最新最棒”的版本，所有的人都在为你欢呼，不过我们的读者想知道你到底有什么过人之处。

HTML5：首先，我增加了一堆新元素，还有一些新属性。

Head First：看来我们还没有使用任何新元素，是吧？

HTML5：你现在使用的所有元素都是HTML5标准的一部分，所以你使用的就是HTML5元素。不过，你说的没错，确实还没有用到任何新元素……

Head First：为什么不用？难道我们不该尽快使用所有这些最新的元素吗？

HTML5：没必要。应该记得（第3章中提到过）：一定要物尽其用，使用最合适的元素完成它最擅长的工作！这些最新的元素有一些特定的工作。 其中一些要用来为页面增加更多结构和含义。比如我的新<article>元素，它就专门用于博客帖子和新闻报道之类的内容。

Head First：我们也可以在第3章Tony的博客中使用这些新元素，对吗？

HTML5：那是当然了……我相信你以后肯定会增加这些新元素。

Head First：我想读者们可能还很困惑，因为他们在这本书里学的是HTML，是不是应该学HTML5？

HTML5：不！HTML5只是演进路上的下一步，你学到的所有东西在HTML5中都是一样的。HTML5只是增加了一些新东西。实际上，你不应该再说“HTML5”了。我就是HTML的最后一个版本，所以请叫我HTML。现在如果还讲HTML5只会让人糊涂。

Head First：请等一下，关于HTML5做了那么多宣传，你真的建议我们把你叫做HTML吗？

HTML5：没错。你已经知道，我是一个活的标准，而版本号是死的。嗯，确切地讲，我是一个活的HTML标准，而不是HTML5。

Head First：我懂了。我们的读者确实应该继续学习HTML5，噢，抱歉，应该是继续学习HTML，他们目前为止学到的所有知识都很重要，而且接下来要学习的所有新知识都是最新、最棒的HTML技术。

HTML5：完全正确。

Head First：不过，我还得问一句。我听说你的一些新内容是用来构建Web应用的。这是什么意思？

HTML5：最重要的是，我不再只是用来建立Web页面，现在我被设计为可以建立成熟的Web应用。

Head First：这有什么区别？

HTML5：Web页面主要是静态页面。页面上有一些图像和一堆链接，可能还有一些漂亮的效果，比如菜单。不过，总的来讲，页面还只是用来阅读。另一方面，Web应用则用来交互，用来完成具体工作。它们就像你的桌面计算机上的应用一样，只使用Web应用就能在Web上完成工作。

Head First：能不能给我一个例子？

HTML5：社交媒体应用、地图应用、游戏……这样的例子不计其数。

Head First：在HTML5之前这些做不到吗？

HTML5：嗯，有些可以做到，不过构建这些应用所需的很多特性在HTML5中才第一次成为标准。在此之前，即使某些特性存在，也实属偶然。

Head First：不过，我不认为我们会在这本书里构建任何应用。

HTML5：确实如此，但你可以看看《Head First HTML5 Programming》。那本书就是在介绍如何用我来构建Web应用！

Head First：我们肯定会的！感谢你接受我们的采访，HTML5。

Jim：太好了，真是很容易。不过，也有一点虎头蛇尾吧……我们把这个doctype放在文件最上面，告诉浏览器我们的页面是HTML，不过然后呢？什么都没有改变。

Frank：对，你看不到任何改变，但它确实告诉了浏览器我们在使用标准HTML。浏览器可以最大程度地使用这个信息。另外，老板希望我们写完全合法的HTML，为此我们也需要这个doctype。

Jim：OK，仅此而已吗？然后呢？我们现在是在写工业标准HTML吗？

Frank：据我所知是这样，不过现在开始有点意思了。现在只有一样东西可能导致失败，这就是我们可能在页面中引入的错误。假设我们忘记加一个结束标记会怎么样？标记名敲错了呢？

Jim：噢，没错，嗯，如果确实存在这些错误，则难道我们不知道吗？

Frank：不一定，浏览器看到错误时，它很善于绕开这些错误。

Jim：如果我把人员都召集起来，则我们一起对整个页面完成审查，这样好不好？

Frank：可能不需要……现在有很多工具可以帮助你验证页面。

Jim：验证？

Frank：对，就是通查页面，确保所有标记都是合法的。确保与标准保持一致。这有些像HTML的一个拼写检查工具。

Jim：听上去是个不错的主意。从哪里可以得到这些工具呢？

Frank：W3C制订标准的那些人提供了一个验证工具，这是免费的。

Jim：太好了，下面就来完成验证吧。

认识W3C验证工具

下面来看W3C验证工具，用它来验证我们的休闲室文件。要得到这个验证工具，只需将浏览器指向http://validator.w3.org。

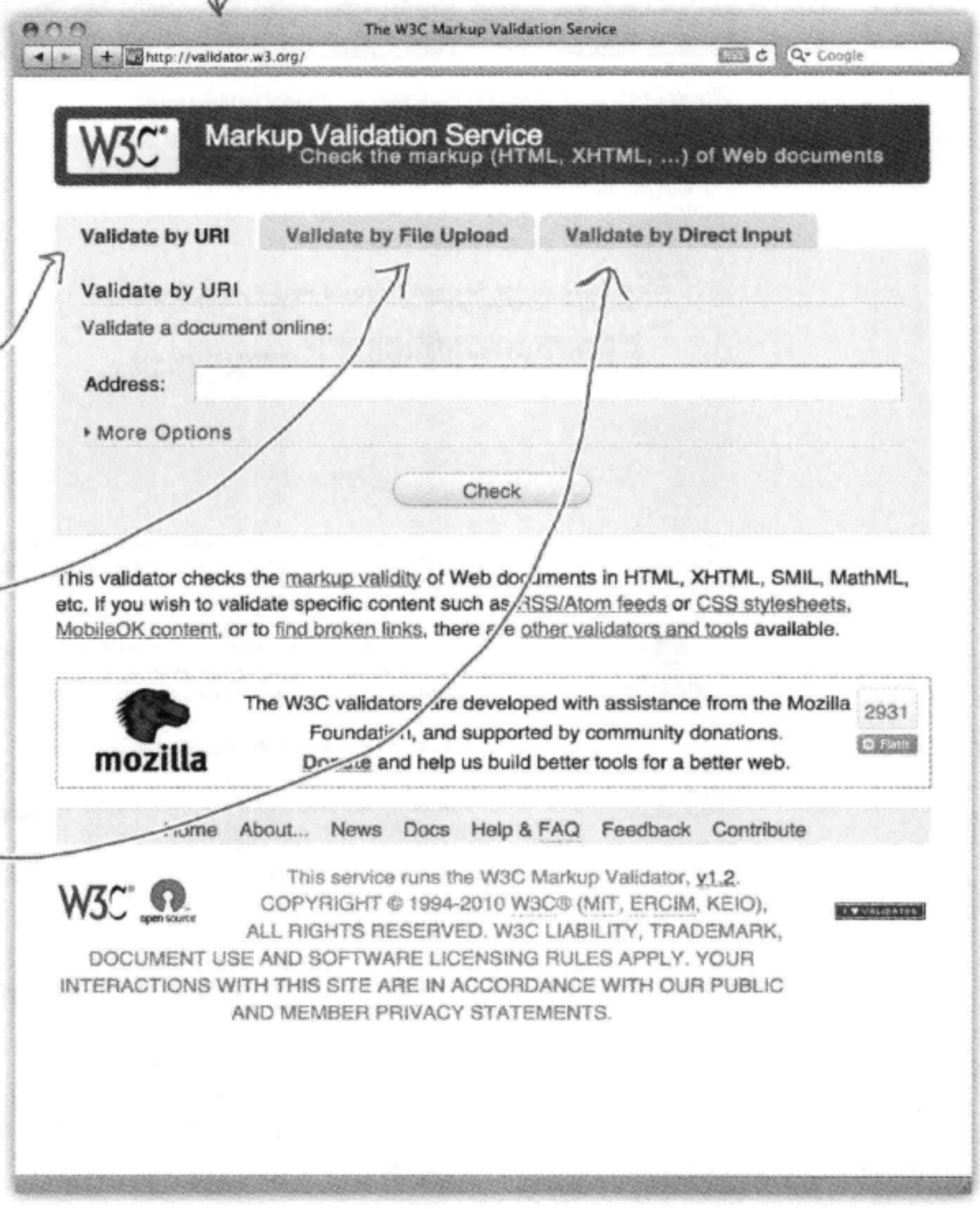
W3C验证工具位于http://validator.w3.org。

检查HTML有3种方法：

(1) 如果你的页面在Web上，则可以在这里输入URL，单击Check（检查）按钮，这个验证服务会获取你的HTML，并完成检查。

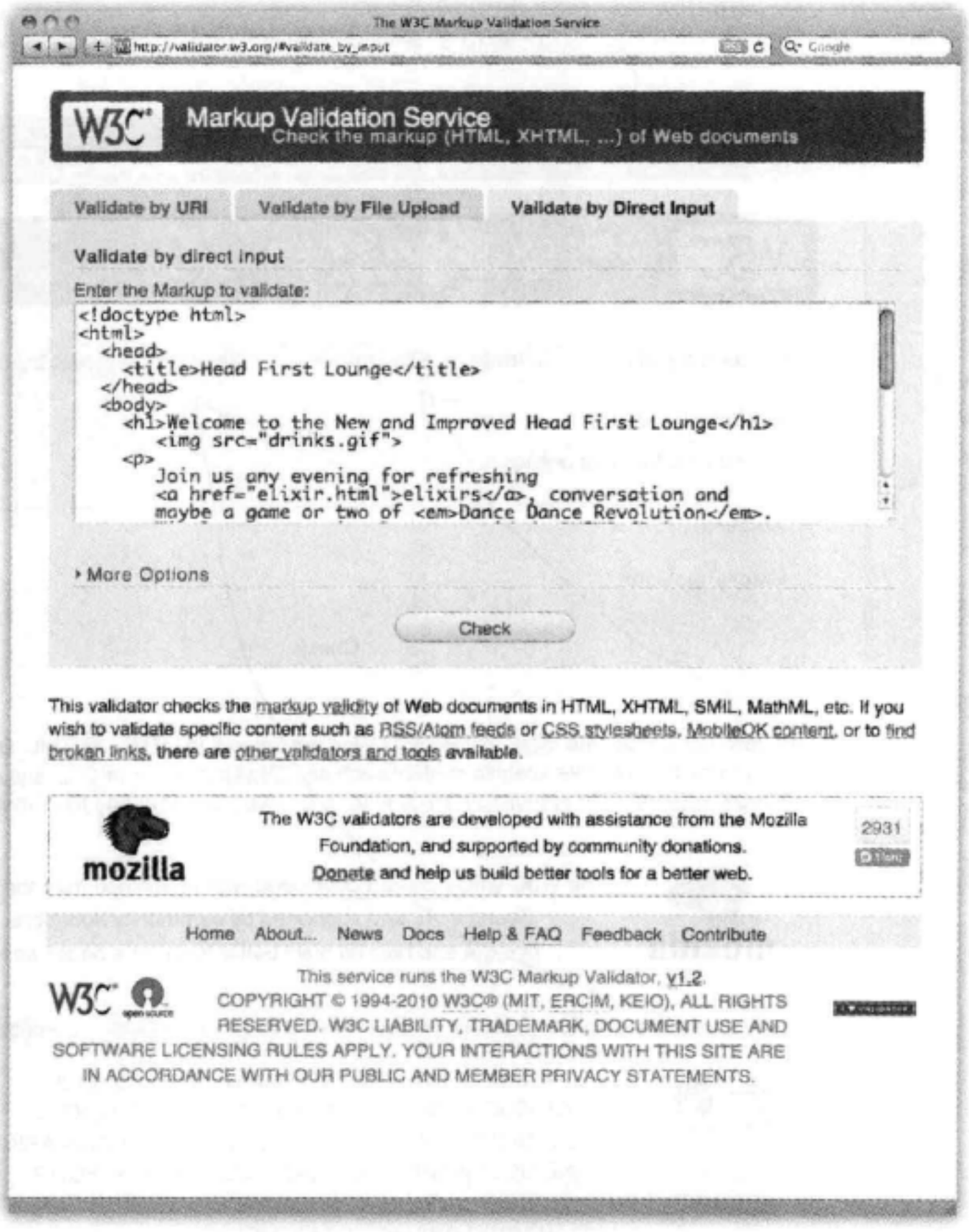
(2) 可以选择第2个标签页，从你的计算机上传一个文件。选择这个文件后，单击Check（检查），浏览器会上传这个页面，由验证服务检查。

(3) 或者，也可以选择第3个标签页，将你的HTML复制粘贴到这个标签页上的表单中。然后单击Check（检查），验证服务就会检查你的HTML。



验证Head First休闲室

我们要使用第3个标签页“Validate by Direct Input”（直接输入验证）来验证“lounge.html”文件。这说明，需要从“lounge.html”将HTML复制粘贴到这个标签页上的表单中，跟着我们一起来试一试……

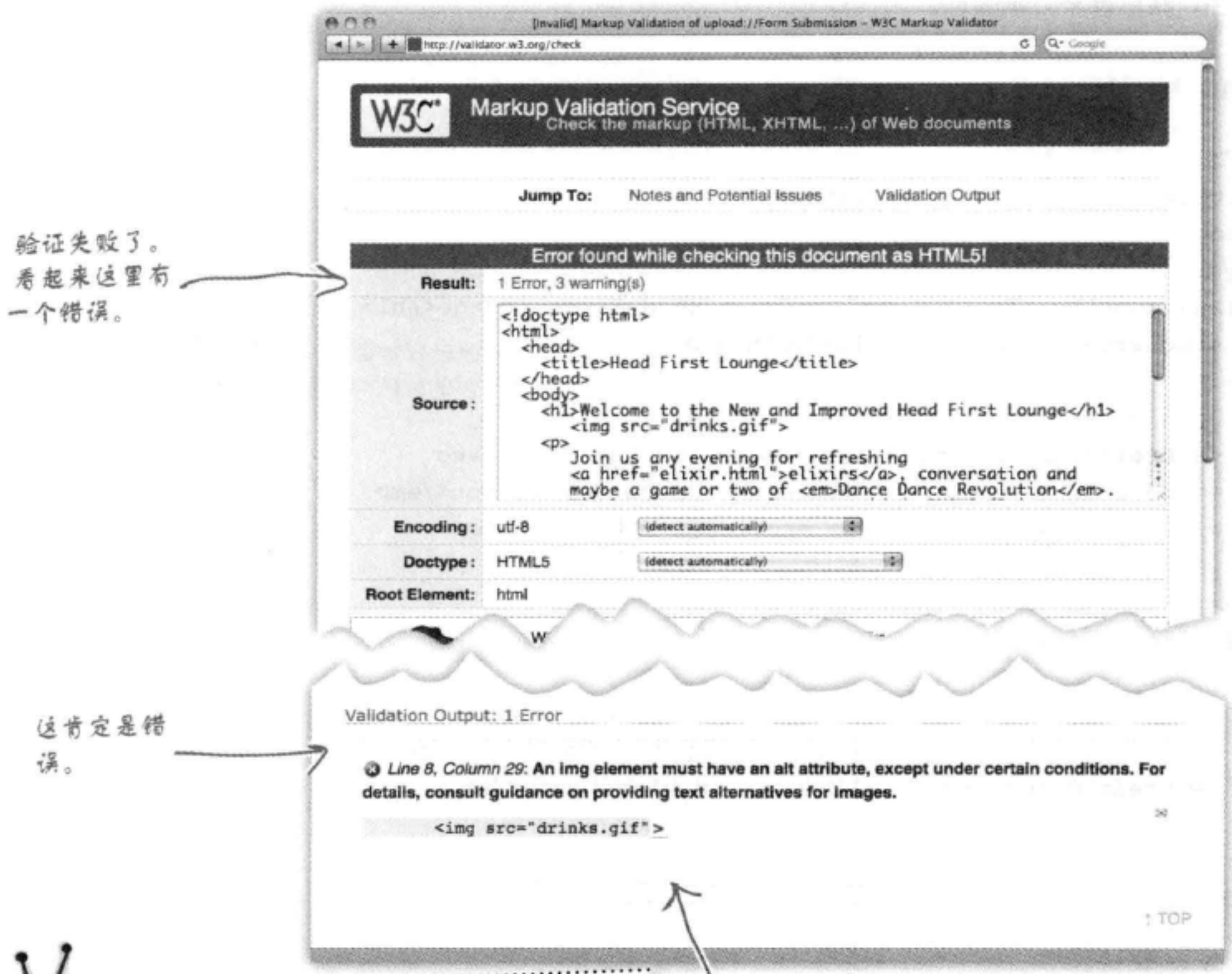


这里我们使用了第3种方法。单击“Validate by Direct Input”（直接输入验证）标签页，粘贴“lounge.html”的代码，现在这个HTML代码最上面有HTML5的doctype。我们已经等待这个时刻很久了……这个Web页面合法吗？绝对没有问题吗？单击Check（并翻开下一页）按钮，看看结果……

如果方法1或2更方便，也完全可以使用这两种方法。

唉呀，我们遇到一个问题……

页面上出现红色肯定不是好事。看来这个页面没有通过验证。我们最好来看一下……



验证失败了。看起来这里有一个错误。

这肯定是错误。

还不算太糟糕。看来我们必须在<img>元素中使用alt属性。

Watch it!

W3C一直在修改这个验证工具。

因为W3C在频繁修改验证工具，你看到的错误消息可能不完全一样。不用担心，即使你没有看到上面的错误，也要跟着我们继续学习，因为后面几页的所有内容都很重要。

修正错误

OK，看来修正这个错误很简单。只需要在HTML5中为<img>元素增加一个alt属性。下面打开“lounge.html”，完成修改，保存，然后再来尝试验证。

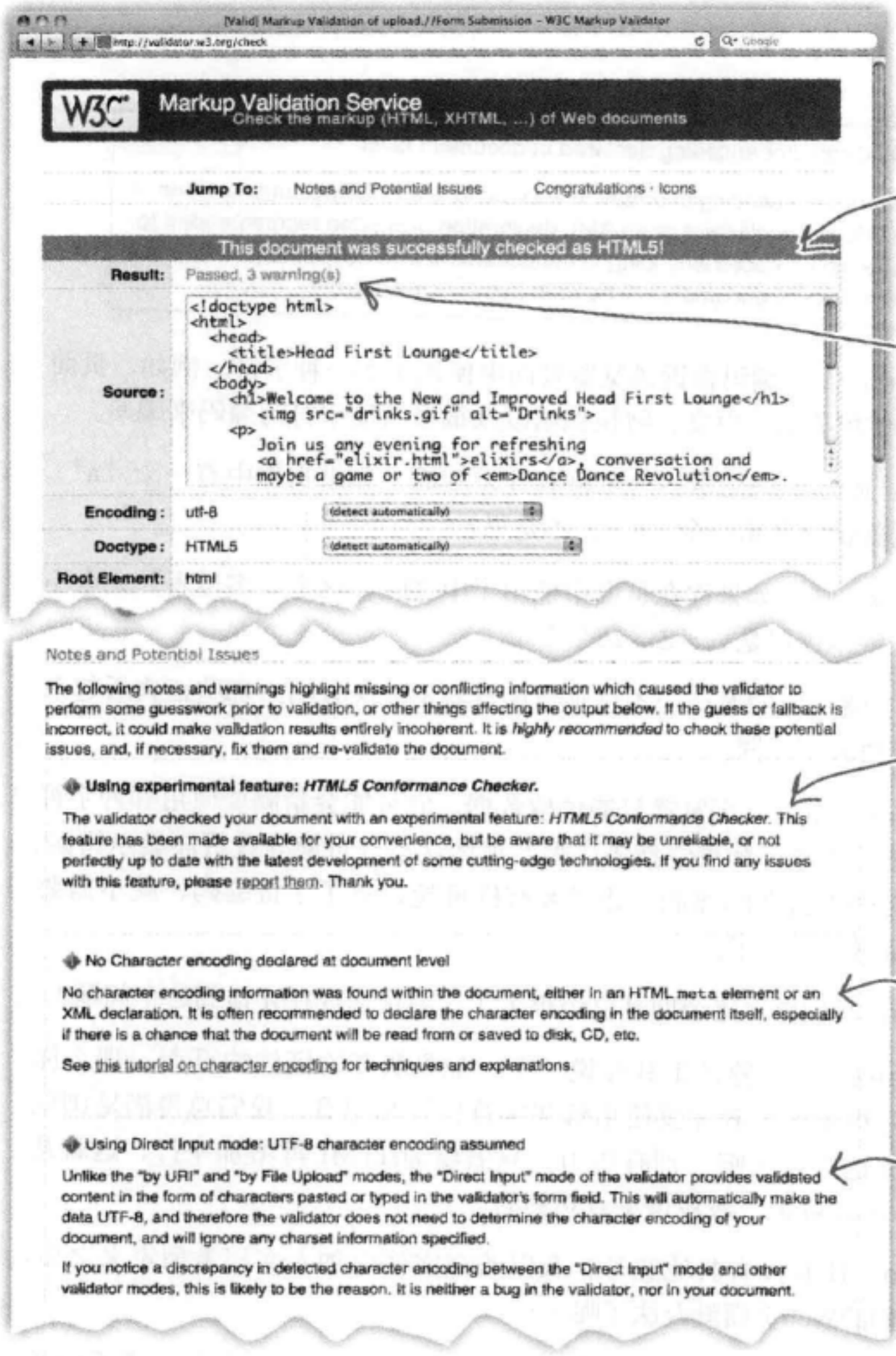
```
<!doctype html>
<html>
  <head>
    <title>Head First Lounge</title>
  </head>
  <body>
  <h1>Welcome to the New and Improved Head First Lounge</h1>
    <img src="drinks.gif" alt="Drinks">
  <p>
    Join us any evening for refreshing
    <a href="elixir.html">elixirs</a>, conversation and
    maybe a game or two of <em>Dance Dance Revolution</em>.
    Wireless access is always provided; BYOWS (Bring
    your own Web server).
  </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="directions.html">detailed directions</a>.
    Come join us!
  </p>
</body>
</html>
```

你已经知道alt属性是什么，把它增加到<img>元素中。

在你看来，为什么HTML5中alt属性是必要的？

就快完成了……

成功了！现在页面上出现一个绿条，这肯定是好事。不过，还有3个警告。看来我们还需要注意几个问题。下面来看一下：



通过了！不过……

……还有一些警告，我们要向下滚动，看一下这些警告……

这里不用担心，这是一个标准警告，只要验证工具在W3C看来是实验性的（可能很长一段时间内都是如此），你就会经常看到这个警告。

嗯，这看起来是省略字符编码信息导致的问题。稍后我们就会讨论这个内容……

这表示，如果我们没有提供一个字符编码，他们会假设一个字符编码。

由此看来，从如何编写HTML来讲，我们已经得到了一个合法的文件，不过看起来还需要对“字符编码”做些处理。下面来看字符编码是什么意思……

No Character encoding declared at document level

No character encoding information was found within the document, either in an HTML meta element or an XML declaration. It is often recommended to declare the character encoding in the document itself, especially if there is a chance that the document will be read from or saved to disk, CD, etc.

Frank：字符编码告诉浏览器页面中使用了哪一种字符。例如，页面可以使用英语、中文、阿拉伯语以及很多其他字符的编码来编写。

Jim：确定如何显示一个字符有什么难的？如果文件中有一个“a”，浏览器就会显示一个“a”。不是吗？

Frank：嗯，如果你在页面中使用了中文呢？这个“字母表”完全不同，绝不仅仅是A～Z这26个字母。

Jim：噢，说的对……不过难道浏览器不能区分吗？那些语言看起来与英语大不相同。

Frank：不能，浏览器只能读取数据。它可能会猜测要使用哪种字符编码，不过，如果它猜错了呢？那么这不仅可能导致页面显示错误，还会带来潜在的漏洞，让黑客有机可趁。有了字符编码，就不需要浏览器去瞎猜了。

Jim：我们建立这个网站已经很久了，为什么现在才谈到这个问题？

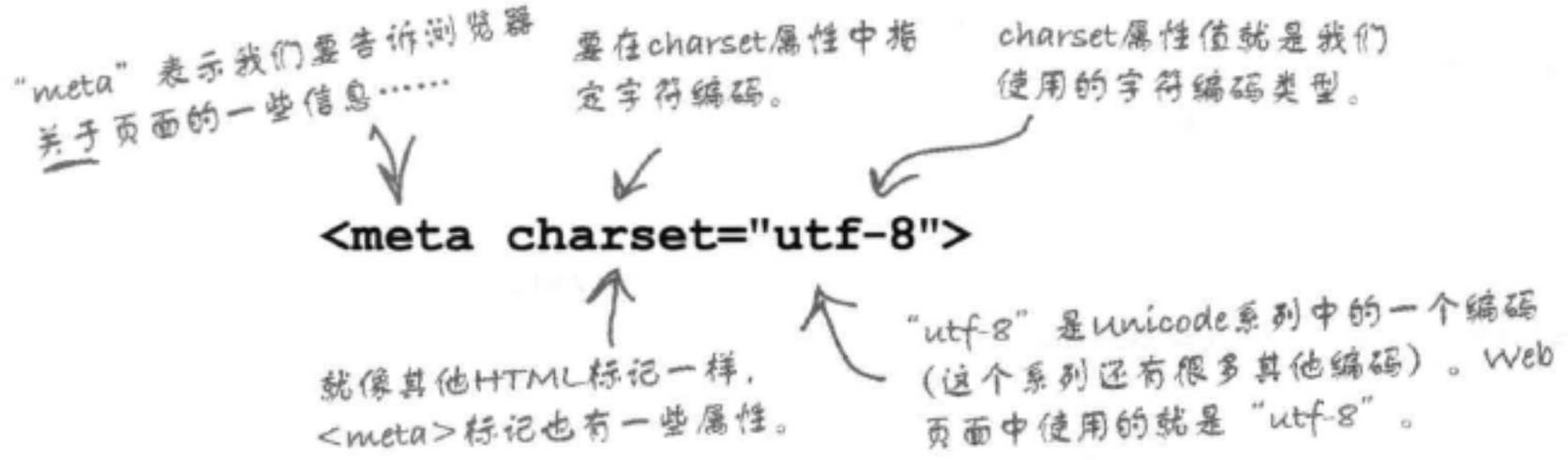
Frank：因为验证工具在说“嘿，如果我要验证你的页面，那么你最好提前告诉我你要使用哪些字符！”想想看，我们总得满足浏览器的这个要求吧。别有压力，只需要为HTML再增加一行，这就是<meta>标记。我应该早点想到的。

Jim：还有没有其他意外？我以为在文件中加入文档类型定义之后，我们的Web页面就合法了呢……

Frank：但愿再没有其他意外了！下面加入<meta>标记，看看结果怎么样。

增加一个<meta>指定字符编码

字符编码为我们提供了一种方法，可以在计算机上表示某种语言中的所有字母、数字和其他符号。你可能知道这样一些编码，如ASCII，甚至莫尔斯码，还有很多其他的编码。幸运的是，如今标准已经统一为Unicode字符编码。采用Unicode，一种编码就可以表示所有语言。不过，由于还存在其他编码，所以我们还需要告诉浏览器我们在使用Unicode（或者你选择的另外一种编码）。要为Web页面指定Unicode，需要在HTML中加一个<meta>标记，如下所示：



there are no Dumb Questions

问：doctype、<meta>标记……噢，写Web页面真的需要这些吗？

答：指定doctype以及用<meta>标记指定字符编码有点像纳税：这是你的义务，必须履行。可以这样来看：在编写Web页面的人当中，你对它们的理解已经超过了98%的人，这很好。不过以后每个人都会在HTML中加上了doctype和<meta>标记，继续前进。所以要确保你的HTML也要加上它们，再来做更有意思的事情。

问：utf-8？

答：你要跟上我们的思路。这就像WD-40,你不用操心为什么它会叫这个名字，直接使用就可以了。正如我们所说，utf-8（有时也写作UTF-8）是Unicode编码系列中的一个编码。utf-8中的u表示Unicode。Unicode是很多常用软件应用和操作系统都支持的一个字符集，这也是Web选择的编码，因为它支持所有语言和多语种文档（即多种语言的文档）。它还与ASCII兼容（ASCII是英语文档常用的一种编码）。如果你有兴趣了解关于Unicode或一般字符编码的更多内容，则可以查看http://www.w3.org/International/O-charset.html，那里提供了关于字符编码的更多信息。

问：我还看见过这样的的<meta>标记：<meta http-equiv=" Content-Type " content=" text/html;charset=utf-8 " >。是不是有时我得使用这个标记？

答：不用，这是HTML 4.01和更早版本中<meta>标记的格式。在HTML5中，可以直接写为<meta charset=" utf-8 " >。

问：是不是由于这个原因，你才在第1章中使用utf-8编码来保存我们的文件？

答：没错。为服务器提供的文件的编码要与<meta>标记中指定的编码一致。

让验证工具（和很多浏览器）接受<meta>标记……

<meta>标记放在<head>元素中（应该记得，<head>包含有关页面的信息）。把这个<meta>标记行增加到你的HTML中。下面首先在“lounge.html”文件中增加这个标记：

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
      <img src="drinks.gif" alt="Drinks">
    <p>
      Join us any evening for refreshing
      <a href="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 server).
    </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="directions.html">detailed directions</a>.
      Come join us!
    </p>
  </body>
</html>
```

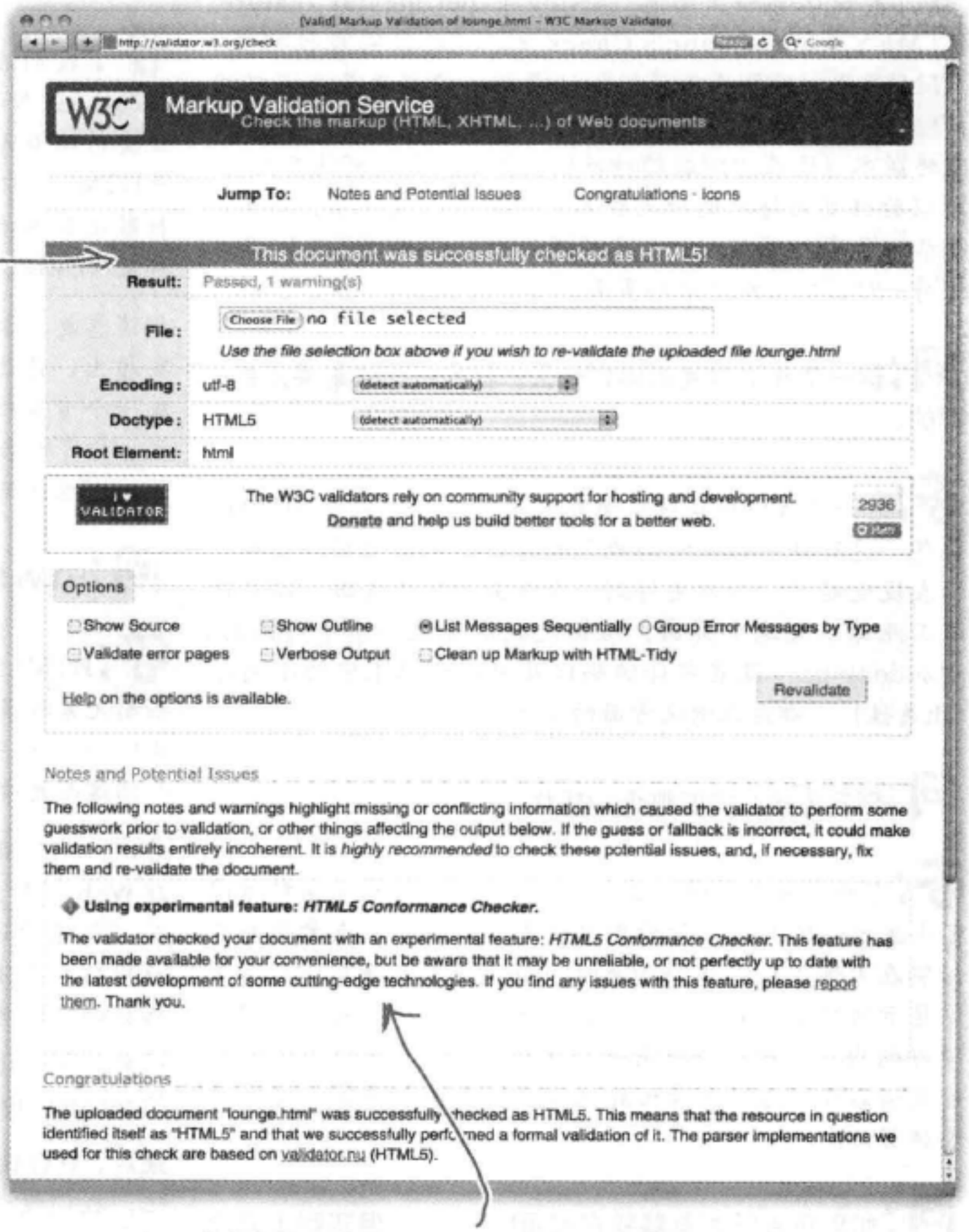
这里是<meta>标记。我们把它增加到<head>元素中，放在<title>元素上面。

这一行要增加到<head>元素中所有其他元素的上面。

是不是想再试一试？这个HTML合法吗？首先，对“lounge.html”文件完成以上修改，保存，然后在浏览器中重新加载。同样的，你不会注意到任何变化，不过浏览器会注意到。现在来看它是否合法……

事不过三？

这一次，我们选择第2个标签页（通过文件上传验证）。你也可以选择对你最适用的方法。如果你想尝试上传方法，则可以把你的“lounge.html” HTML文件上传到W3C验证工具Web页面（http://validator.w3.org）。一旦上传，单击Check（检查）按钮……



“成功通过检查，这是HTML5”！
爱死这个绿条了！

还有一个警告……不过我们不需要担心它（见下面的说明）。

这实际上是同一个警告，指出我们在使用一个“实验性的服务”。不用担心。

there are no Dumb Questions

问：验证工具说它对于HTML5还只是实验性的。这是什么意思？

答：验证工具给出消息“Using experimental feature: HTML5 Conformance Checker”（使用实验性特性：HTML5符合性检查工具），这是指，验证工具正在根据HTML5标准检查你的HTML，不过由于HTML5标准不是最终版本（还在增加新的特性），验证工具很可能会改变，所以验证页面得到的结果也不是固定的。这说明，作为一个尽职尽责的开发人员，你要尽可能与最新的HTML标准保持一致，经常检查你的页面。

问：这一章我们到底能得到什么？我的页面看起来还是一样的。

答：这一章我们稍稍对页面做了些调整，使它符合HTML规范。这有什么好处？与规范越接近，你的页面在实际中的表现就越好。如果要得到一个专业的Web页面，你要使用工业标准写这个页面，这正是我们在这一章中所做的，增加doctype、设置字符编码以及解决HTML中的小问题（alt属性），都是在做这方面的工作。

问：到底为什么需要那个alt属性？

答：有两个主要原因。首先，如果你的图像出于某种原因无法显示（比如说，你的图像服务器宕机了，或者你的连接实在太慢了），这个alt属性（在大多数浏览器上）会帮你显示所指定的alt文本，来取代图像。其次，对于有视力障碍的用户，他们会使用一个屏幕阅读器来阅读页面，屏幕阅读器可以为用户读出alt文本，这样能帮助他们更好地理解页面。

问：如果我告诉浏览器我在使用HTML5，但实际上并没有，会怎么样呢？

答：浏览器会发现你实际上没有写HTML5，并使用它提供的错误处理功能来尝试采取正确的做法。在这方面，你又会遇到老问题，不同的浏览器会采用不同的方式处理页面。要得到可预测的结果，唯一的办法就是告诉浏览器你在使用HTML5，而且没有撒谎，确实是在使用HTML5。

问：我们对HTML5做了一些讨论，不过我想再明确一下：我们写的HTML和HTML5之间有没有差别？

答：我们在使用标准HTML，也就是HTML5。现在HTML5引入了一些新的标记（很快我们就会看到），另外还支持编写Web应用（这本书不作介绍）。不过HTML5还是HTML，你写的所有代码都“符合”HTML5。所以，很抱歉这些术语可能让你有些混乱，不过请相信，所有这些都是HTML，包括HTML5规范提供的所有新特性。

好消息是，你之前学到的所有知识都能很好地用于HTML5，实际上，你会看到，要从一个“非正式的HTML”页面变成一个专业的HTML页面，几乎不需要你做什么。也就是说，你可以告诉你的老板你已经在用HTML5，也许这能让你更快地升职加薪。

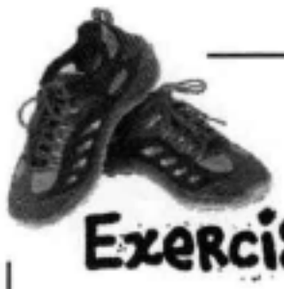
问：HTML5与HTML 4.01相比有哪些大的改进？

答：HTML5的好处有三方面。首先，HTML5中提供了一些新元素和属性，这些新元素和新属性很酷（如<video>元素），另外一些可以帮助你编写更好的页面（本书后面还会介绍这些元素）。

其次，它有很多新特性，允许Web开发人员用HTML5创建Web应用。Web应用就是表现得像应用一样的Web页面（就像你通常在笔记本电脑或移动设备上使用的那些应用），而不只是静态Web页面。如果你对创建Web应用感兴趣，读完这本书之后（不好意思，做个广告），你可以看看O’Reilly出版的另一本好书：《Head First HTML5 Programming》。

最后，HTML5规范比HTML以前的版本健壮得多。还记得吧，我们说过，这个规范现在会记录Web开发人员常犯的错误，还会帮助浏览器了解如何处理这些错误，是不是？这说明有错误的Web页面不会像以往那样导致恐慌，这对用户来说确实是一件好事。

总的来讲，与HTML 4.01相比，HTML5确实有了很大的改进，绝对值得学习。我们会在接下来几章加快速度，继续深入。



Exercise

该轮到你了。为“directions.html”和“elixir.html”增加 <meta>标记。尝试对这两个页面完成验证：它们通过验证了吗？如果没有，请进行修复，使它们能顺利通过验证。

在这里的空白处写出你的验证体验！

Exercise

现在使用验证工具做个小练习。在你刚才成功验证为HTML5的代码（见241页）中，去掉doctype。没关系，先把它去掉，看看验证时会发生什么。再把这个版本的文件提交到验证工具，看看会发生什么。在下面记录你得到的错误。

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
       <img src="drinks.gif" alt="Drinks">
    <p>
       Join us any evening for refreshing
       <a href="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 server).
    </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="directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

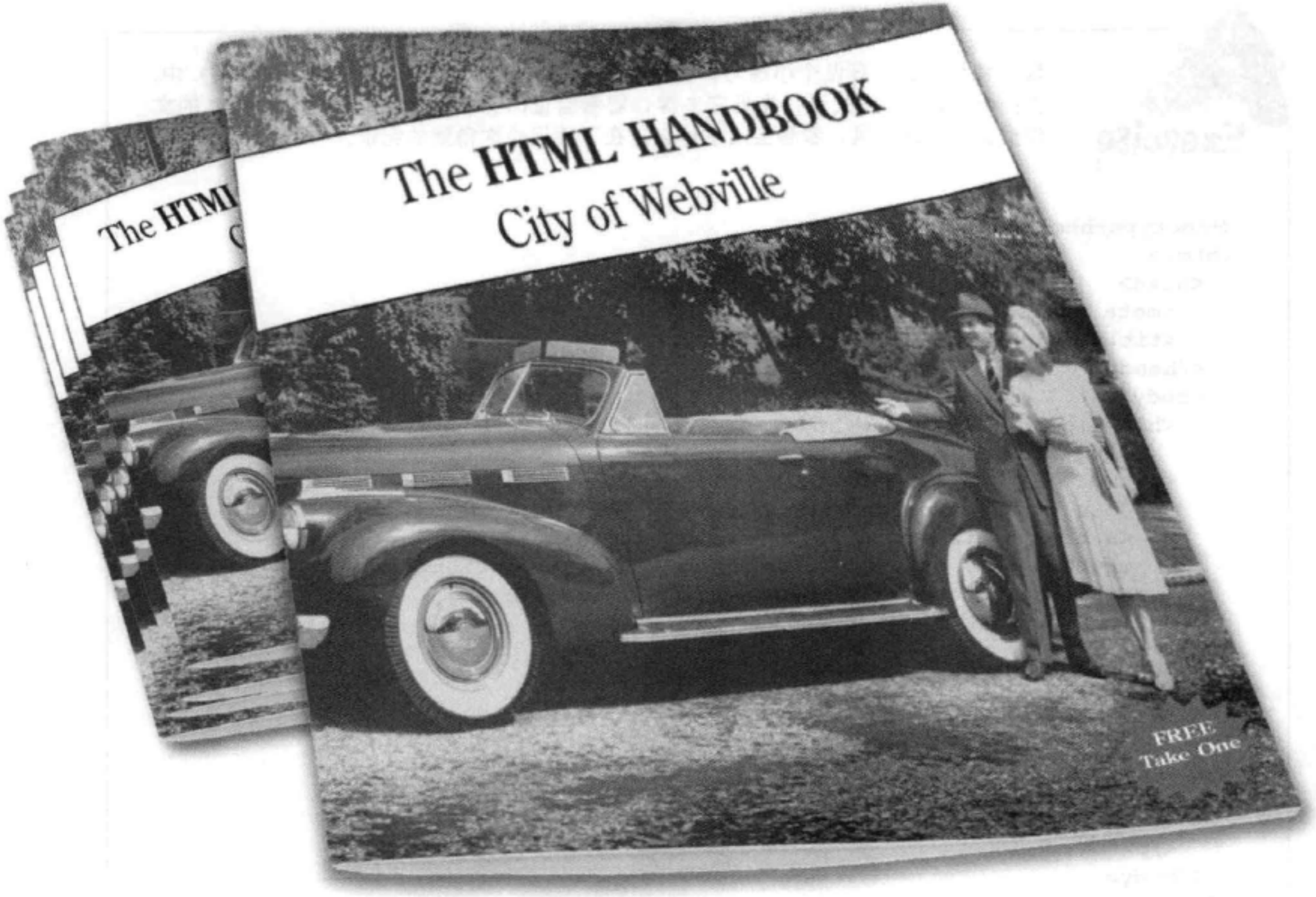
删除doctype！

记录写在这里，你得到多少个错误？

如果没有包含doctype，则从这里可以得出关于HTML类型的哪些信息？

叫所有HTML专业人员来拿手册……

欢迎HTML开发精英的到来，他们知道如何创建专业的页面。要记的东西太多了，所以Web镇提供了一个简明指南，来帮助创建工业标准页面。作为刚刚到访Web镇的你来说，这个指南很有意义。这不是一个详尽的参考资料，只强调了构建页面时比较重要的一些最佳实践。在后面几章游览Web镇时，你还会对这个指南里的知识有所补充，不过，现在先拿一本吧，都是免费的。



Web镇HTML指南

在这个简明指南中，我们把编写合法HTML页面的做法汇集为一组常识性的原则。请依次核对：

一定要以<doctype>开头。

每个页面都要从一个doctype开始。这样才能让浏览器和验证工具顺利开展工作。

任何时候都要使用<!docytype html>，除非你确实在写HTML 4.01或XHTML。

<html>元素：不能没有它。

紧接着doctype，<html>元素必须是Web页面的最顶层元素或根元素。所以，在doctype后面，由<html>标记开始你的页面，</html>标记结束页面，页面中的所有其他内容都嵌套在这个元素中。

记住，要使用<head>和<body>编写更好的HTML。

只有<head>和<body>元素能直接放在<html>元素中。这说明，所有其他元素都必须放在<head>或<body>元素中。没有例外！

在<head>中指定正确的字符编码。

在<head>中包含一个<meta charset="utf-8">标记。浏览器会感谢你的，用户在你的博客上阅读世界各地用户发表的评论时，也会因为你指定了正确的字符编码而感谢你。

Web镇HTML指南（续）

在这个简明指南中，我们把编写合法HTML页面的做法汇集为一组常识性的原则。请依次核对：

没有<title>的<head>算什么？

一定要在<head>元素中包含一个<title>元素。这是雷打不动的原则。如果没有做到，则会得到不符合标准的HTML。只能在<head>元素中放置<title>、<meta>和<style>元素。

嵌套某些元素时要当心。

在这里提供的原则中，嵌套规则相当灵活。不过有一些情况是没有意义的。不要把<a>元素嵌在另一个<a>元素中，因为这样会让访问者很迷惑。另外，不允许在<img>等void元素中嵌套其他内联元素。

检查属性！

有些元素属性是必要的，有些则是可选的。例如，如果<img>元素没有src属性，就没有什么意义，现在你还知道了，alt属性也是必要的。在学习过程中，要逐渐熟悉各个元素的必要和可选属性。

HTML考古

在这本书中，你一直在使用当前HTML标准中的元素和属性。所以，你没有多少机会看到那些过气被淘汰的元素和属性。其中大多数元素都是在HTML 4.01中被淘汰的，不过在老的Web页面中有时还会出现，因此对这些遗留的元素多些了解没有坏处。我们挖出了一个HTML 3.2页面，其中包含现在标准中已经没有的一些元素和属性，另外这个页面还存在现代HTML不推荐的一些错误做法。

```
<html>
<head>
   <title>Webville Forecast</title>
</head>

<body bgcolor="tan" text="black">

   <p>
      The weather report says lots of rain and wind in store for
      <font face="arial">Webville</font> today, so be sure to
      stay inside if you can.
   </p>

   <ul>
      <li>Tuesday: Rain and 60 degrees.
      <li>Wednesday: Rain and 62 degrees.
   </ul>

   <p align=right>
     Bring your umbrella!

   <center><font size="small">This page brought to you buy Lou's
       Diner, a Webville institution for over 50 years.
   </font></center>

</body>
</html>
```

这里是一些控制表现的属性。bgcolor设置页面的背景色，text设置体文本的颜色。

利用<font>元素和它的face属性来改变字体。

可以没有结束标记，如
和</p>。有时你仍然能这么做，不过不推荐这种做法！

属性值缺少引号。现在建议都要加上引号，对于有多个值的属性，引号更是必要的。

这里有两种对齐文本的方法。右对齐一个段落，并居中一段文字。

由<font>元素使用size属性来控制文本大小。

扮演验证工具

下面你会看到一个HTML文件。你的任务是扮演验证工具，找出所有错误。完成这个练习后，对照本章最后给出的答案看看是否找到了所有错误。

完成之后（或者如果你需要提示），可以使用验证工具来检查你的答案对不对。

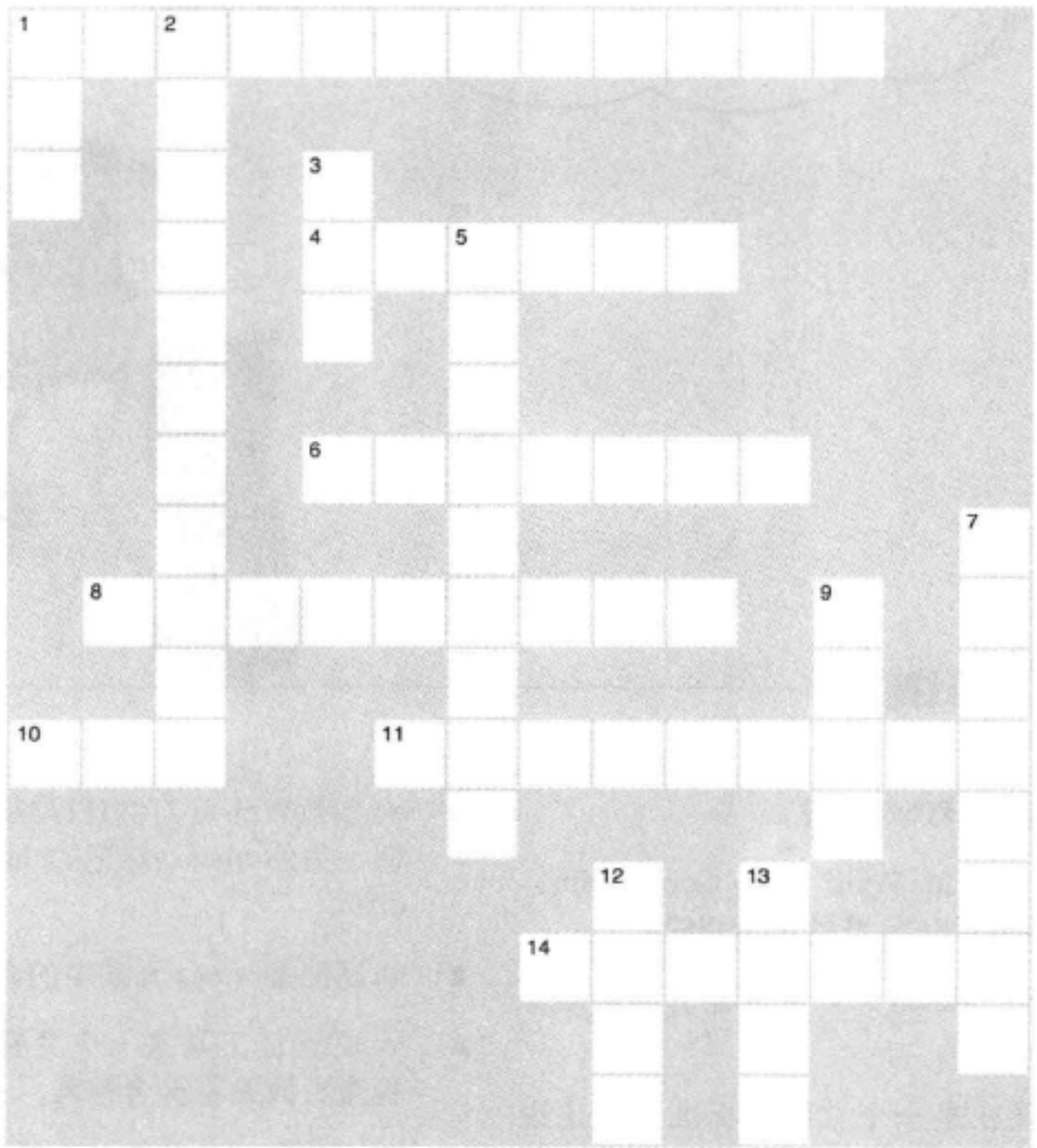
```
<html>
<head>
   <meta charset="utf-9">
</head>
<body>
   <img src="chamberofcommerce.gif">
   <h1>Tips for Enjoying Your Visit in Webville
   <p>
      Here are a few tips to help you better enjoy your stay in Webville.
   </p>
   <ul>
       <li>Always dress in layers and keep an html around your
           head and body.</li>
       <li>Get plenty of rest while you're here, sleep helps all
           those rules sink in.</li>
       <li>Don't miss the work of our local artists right downtown
           in the CSS gallery.
   </ul>
   </p>
   <p>
      Having problems? You can always find answers at
      <a href="http://wickedlysmart.com"><em>WickedlySmart</em></a>.
      Still got problems? Relax, Webville's a friendly place, just ask someone
      for help. And, as a local used to say:
   </p>
   <em><p>
      Don't worry. As long as you hit that wire with the connecting hook
      at precisely 88mph the instant the lightning strikes the tower...
      everything will be fine.
   </em></p>
</body>
</html>
```

BULLET POINTS

- HTML5是当前的HTML标准。
- 万维网协会（World Wide Web Consortium，W3C）是定义HTML标准的标准组织。
- 文档类型定义（doctype）用来告诉浏览器你使用的HTML版本。
- HTML标准现在是一个“活的标准”，这说明这个标准会不断改变，加入新的特性和更新。
- <head>元素中的<meta>标记告诉浏览器关于一个Web页面的额外信息，如内容类型和字符编码。
- <meta>标记的charset属性告诉浏览器Web页面使用的字符编码。
- 大多数Web页面的HTML文件都使用utf-8编码，另外<meta>标记的charset属性值通常也是utf-8。
- alt属性是<img>元素中的必要属性。
- W3C验证工具是一个免费的在线服务，可以检查页面是否符合标准。
- 可以使用这个验证工具确保你的HTML合法，而且元素和属性符合标准。
- 如果遵循标准，则你的页面会更快地显示，而且在不同浏览器中显示时差异会更小，CSS也能更好地工作。

HTML填字游戏

这一章内容真多。现在可以坐下来喝杯饮料了，下面做做这个填字游戏，强化一下学到的知识。所有答案都可以从这一章中找到。



横向

1. 浏览器战争中的受害者。
4. HTML标准是一个_____ 标准。
6. <head>元素中必要的属性。
8. 制订Web标准的人承诺将来的HTML会与老的HTML _____兼容。
10. 老板希望向休闲室页面增加 _____ 之前先完成标准化。
11. 你的HTML满足标准时，就是_____。
14. 这个定义告诉浏览器和验证工具你在创建什么类型的文档。

纵向

1. 提供验证工具的标准组织。
2. Microsoft与Netscape之争。
3. 标准HTML中必要的<img>属性。
5. 这个服务会检查你的HTML是否符合标准。
7. 老的 _____ 与最新的相比复杂得多。
9. 在这里放置有关页面的信息。
12. 在这里放置Web页面内容。
13. Web页面最常用的编码。

扮演验证工具答案

下面你会看到一个HTML文件。你的任务是扮演验证工具，找出所有错误。下面给出答案。

```
<html>                                          缺少doctype。
<head>
   <meta charset="utf-9">                       应该是"utf-8"而不是"utf-9"（这根本不存在）！
</head>                                         <title>应该放在<head>中。
<body>
   <img src="chamberofcommerce.gif">            没有alt属性。
   <h1>Tips for Enjoying Your Visit in Webville   缺少</h1>标记。这会使下面的<p>元素出问题。
   <p>
      Here are a few tips to help you better enjoy your stay in Webville.
   </p>
   <ul>
       <li>Always dress in layers and keep an html around your
           head and body.</li>
       <li>Get plenty of rest while you're here, sleep helps all
           those rules sink in.</li>
       <li>Don't miss the work of our local artists right downtown
           in the CSS gallery.                  缺少</li>标记。这样也能通过验证，不过不推荐这么做！
   </ul>
   </p>                                         多余的</p>，没有与之匹配的<p>。
   <p>
      Having problems? You can always find answers at
      <a href="http://wickedlysmart.com"><em>WickedlySmart</em></a>.
      Still got problems? Relax, Webville's a friendly place, just ask someone
      for help. And, as a local used to say:
   </p>
   <em><p>                                      <em>和<p>标记换了位置。
      Don't worry. As long as you hit that wire with the connecting hook
      at precisely 88mph the instant the lightning strikes the tower...
      everything will be fine.
   </em></p>
</body>
</html>
```

HTML填字游戏答案

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

Exercise Solution

该轮到你了。为“directions.html”和“elixir.html”增加<meta>标记。尝试对这两个页面完成验证：它们通过验证了吗？如果没有，则请进行修复，使它们能顺利通过验证。

答案：要想成功验证“elixir.html”，必须为各个<img>元素增加alt属性。

Exercise

现在使用验证工具做个小练习。在你刚才成功验证为HTML5的代码（见241页）中，去掉doctype。没关系，先把它去掉，看看验证时会发生什么。再把这个版本的文件提交到验证工具，看看会发生什么。在下面记录你得到的错误。

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
      <img src="drinks.gif" alt="Drinks">
    <p>
      Join us any evening for refreshing
      <a href="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 server).
    </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="directions.html">detailed directions</a>.
      Come join us!
    </p>
  </body>
</html>
```

删除doctype!

如果验证时没有doctype，则我们会得到3个错误，还有4个警告。验证工具会假设我们在用HTML 4.01 Transitional写代码（这是HTML 4.01的一个版本，专门设计为在“迁移”到XHTML时使用）。验证工具确实对没有doctype很不满意，所以抱怨了很多次。它还对<meta charset="utf-8">有怨言，因为在HTML5之前，charset不是<meta>标记的一个合法属性。通过这个练习你可以了解到，加上一个doctype可以让验证工具和浏览器更高兴。

记录写在这里，你得到多少个错误？

7 CSS入门

加一点样式

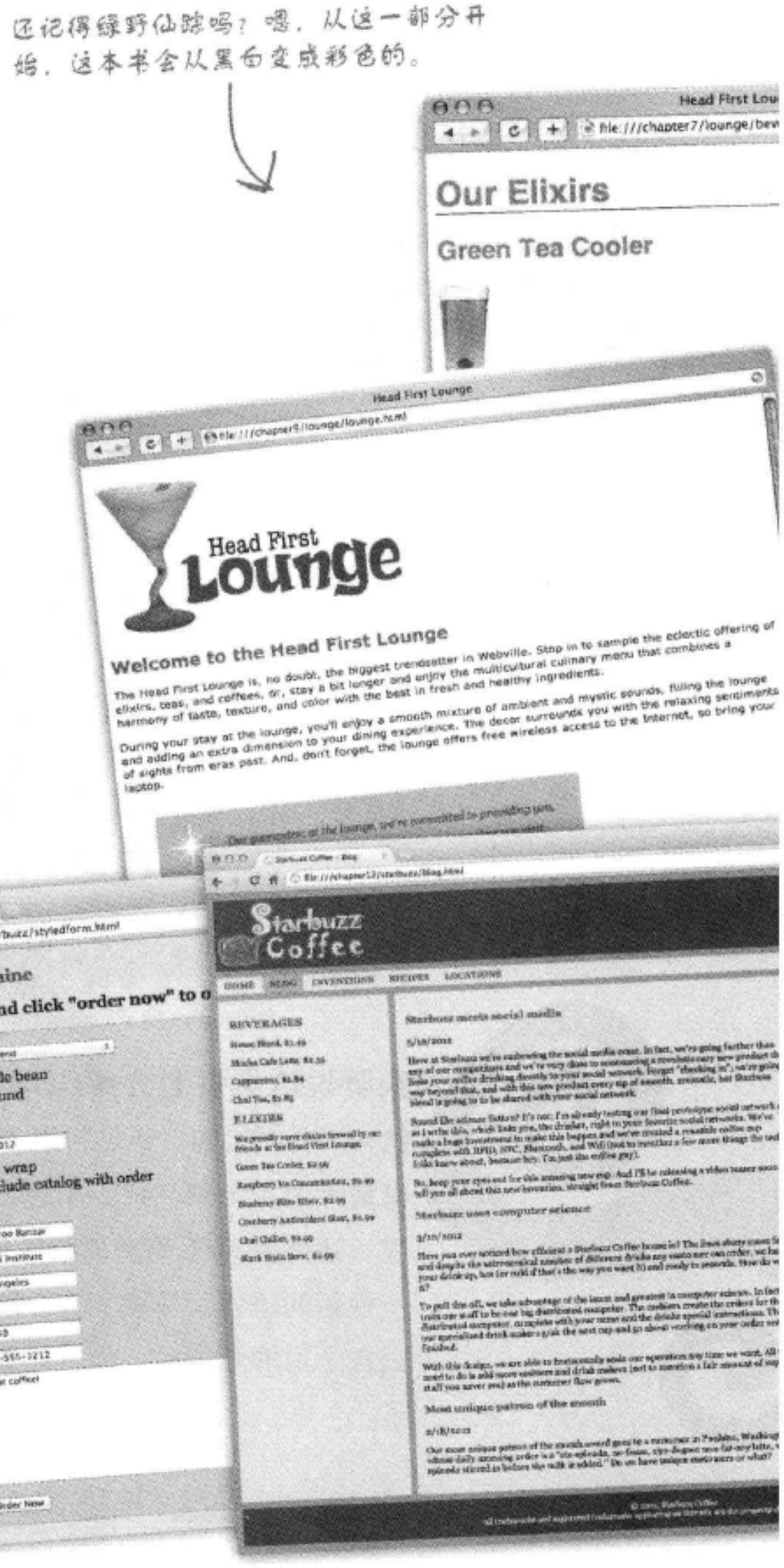
我听说这本书会介绍CSS。到目前为止，你一直在专心学习HTML来创建Web页面结构。不过可以看到，浏览器对样式有很多要求。当然，我们可以请时尚人士来帮忙，不过没有必要。利用CSS，你就能完全掌控页面的表现，通常甚至不用对HTML做任何改变。真的这么容易吗？嗯，你得学习一种新的语言。毕竟，Web镇是一个讲多种语言的小镇。读完这一章对CSS语言学习的介绍后，你就能胸有成竹地站在主干道两边与Web镇的人们交流了。

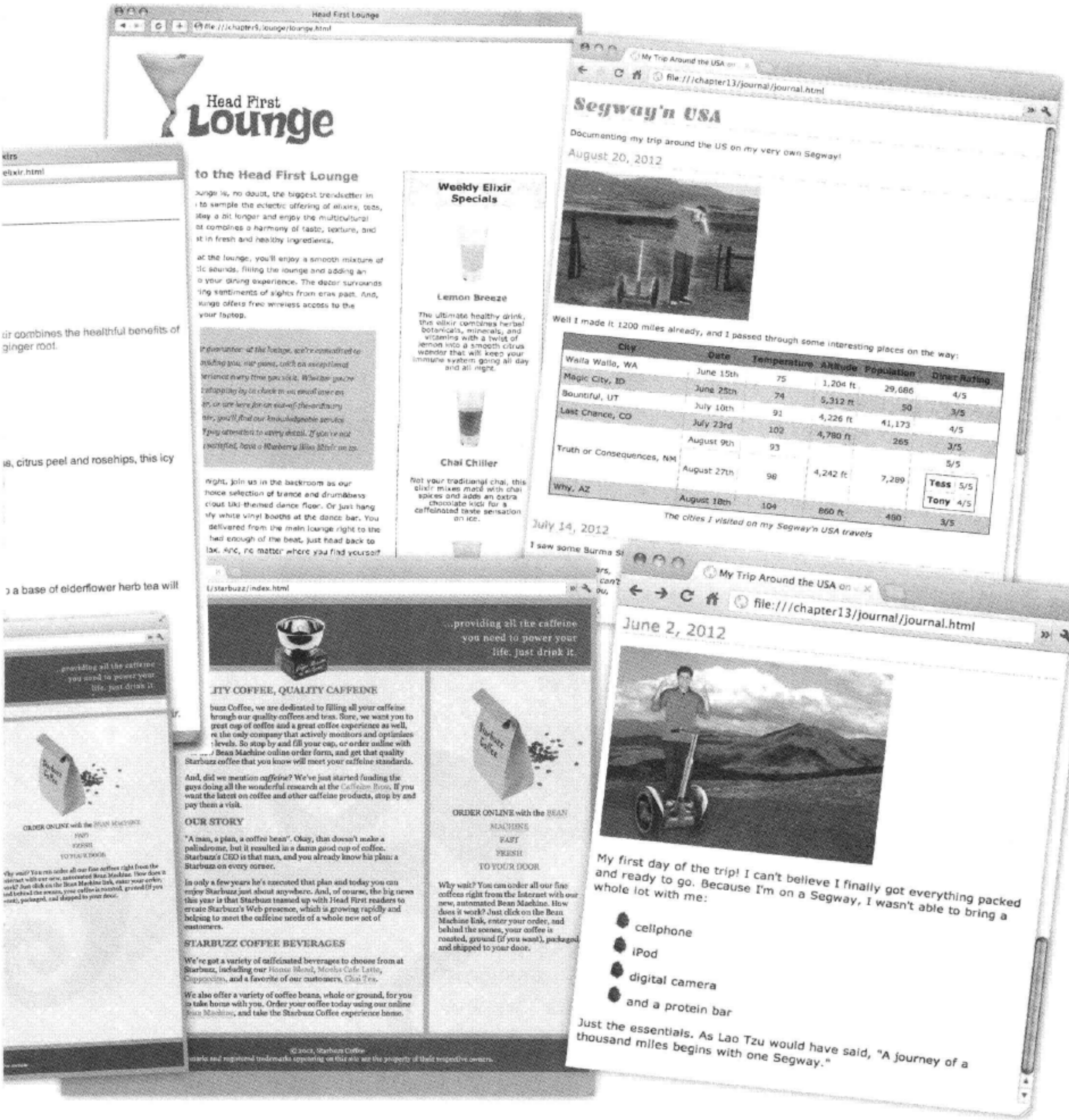
你已经离开堪萨斯

你已经做了不错的热身运动，了解了标记、结构和验证，知道了正确的语法，还明白了什么是嵌套和符合标准，不过现在你会进入更有意思的部分，要为页面增加样式了。不过别担心，你之前做的所有这些HTML工作都不会白费。实际上，你会看到，充分掌握HTML对于学习（和使用）CSS至关重要。另外学习CSS不是一蹴而就的，我们会通过后面几章逐步学习CSS。

为了让你有些初步的认识，这两页上我们散乱地摆放了一些设计，这些设计都将在本书后面用到。是不是与你之前创建的页面大不相同？那么，要创建这些页面你需要做些什么？当然是学习CSS。

下面就开始吧……




Head First Lounge
file:///chapter5/lounge/lounge.html
Head First
Lounge
to the Head First Lounge
Weekly Elixir Specials
Lemon Breeze
Chai Chiller
My Trip Around the USA
file:///chapter13/journal/journal.html
Segway'n USA
Documenting my trip around the US on my very own Segway!
August 20, 2012
Well I made it 1200 miles already, and I passed through some interesting places on the way:
City | Date | Temperature | Altitude | Population | Diner Rating
Walla Walla, WA | June 15th | 75 | 1,204 ft | 29,686 | 4/5
Magic City, ID | June 25th | 74 | 5,312 ft | 50 | 3/5
Bountiful, UT | July 10th | 91 | 4,226 ft | 41,173 | 4/5
Last Chance, CO | July 23rd | 102 | 4,780 ft | 265 | 3/5
Truth or Consequences, NM | August 9th | 93 | 4,242 ft | 7,289 | 5/5
Why, AZ | August 27th | 98 | 860 ft | 480 | 3/5
August 18th | 104
Tess 5/5
Tony 4/5
The cities I visited on my Segway'n USA travels
July 14, 2012
I saw some Burma Sh
ORDER ONLINE with the BEAN MACHINE
FAST
FRESH
TO YOUR DOOR
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.
...providing all the caffeine you need to power your life. just drink it.
OUR STORY
"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.
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.
STARBUZZ COFFEE BEVERAGES
We've got a variety of caffeinated beverages to choose from at Starbuzz, including our House Blend, Mocha Cafe Latte, Cappuccino, and a favorite of our customers, Chai Tea.
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 Bean Machine, and take the Starbuzz Coffee experience home.
© 2012, Starbuzz Coffee
June 2, 2012
My first day of the trip! I can't believe I finally got everything packed and ready to go. Because I'm on a Segway, I wasn't able to bring a whole lot with me:
cellphone
iPod
digital camera
and a protein bar
Just the essentials. As Lao Tzu would have said, "A journey of a thousand miles begins with one Segway."

偶然听到的Web镇“交换空间”节目

没有看过最近的真人秀节目？没关系，我来解释一下：有两个邻居，两所房子，还有1000美元。这两个邻居要交换房子，用这1000美元在48小时内完全重新设计一到两个房间。来听听他们说什么……

当然，在Web镇版的交换空间节目里，所有人谈论的都是CSS设计。如果你听不懂他们在说什么，下面给你一个小小的翻译提示：CSS中的每个语句包括一个场所（如卧室），以及这个场所的一个属性（如窗帘或地毯），还有要应用到这个属性的一个样式（如蓝色，或者1英寸的瓷砖）。

结合HTML和CSS

我们相信，在家装设计领域CSS会有光明的前景，不过下面先回到HTML。HTML没有房间，但它有元素，这些元素会作为将要指定样式的场所。想把你的<p>元素的墙漆成红色？没问题。只不过段落没有墙，所以必须换成段落的background-color属性。可以这样做：

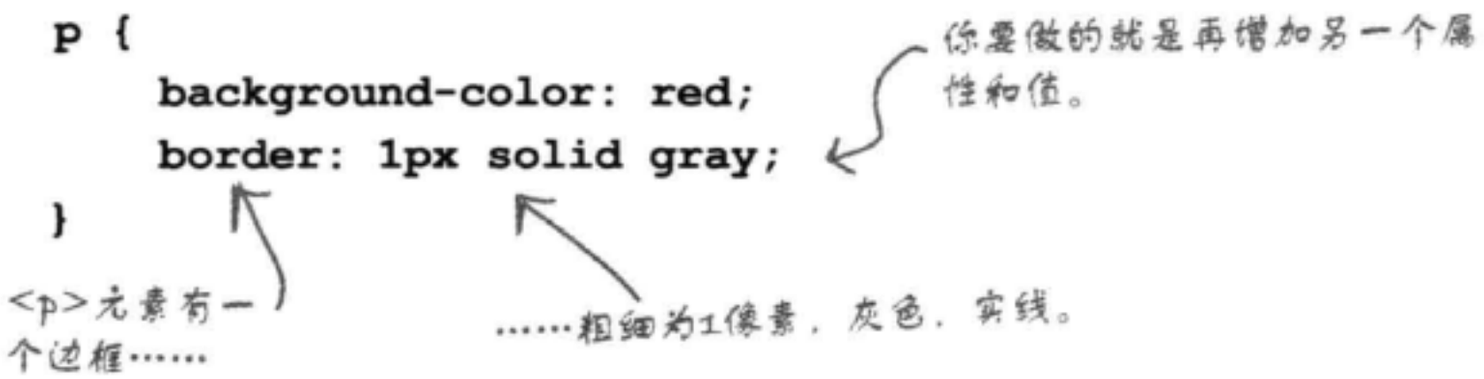
还可以这样写规则：

```
p { background-color: red; }
```

在这里，我们所做的只是去掉了换行。就像HTML一样，可以按你的想法安排CSS的格式。对于更长的规则，通常需要增加一些换行和缩进，让CSS（对你来说）更可读。

想要增加更多样式？

在每个CSS规则中可以根据需要增加多个属性和值。假设你想为段落再加一个边框。可以这样做：



there are no Dumb Questions

问：是不是每个<p>元素都有相同的样式？我能不能（比如说）让两个段落有不同的颜色？

答：目前为止我们使用的CSS规则会为所有段落定义样式，不过CSS表达能力很强。它可以采用很多不同方式为不同的元素指定样式，甚至可以为元素子集指定样式。这一章后面你就会看到如何让两个段落有不同的颜色。

问：我怎么知道可以对一个元素的哪些属性设置样式呢？

答：嗯，元素的很多属性都可以设置样式，太多了，你肯定记不住。通过后面几章的学习，你会非常熟悉较为常用的一些属性。可能还需要找一本好的CSS参考书。网上有很多参考资料，另外O'Reilly的《CSS Pocket Reference》是一本不错的小书，可供参考。

问：再帮我复习一下，为什么我非要用一种不同的语言来定义所有这些样式，而不是使用HTML呢？既然元素都是用HTML写的，用HTML设置样式不是更容易吗？

答：在后面几章你就会看到使用CSS的一些重要优点。不过先简单回答一下你的问题，CSS确实比HTML更适合指定样式信息。通过使用一点点CSS，你可以对HTML建立相当复杂的效果。你还会看到，CSS更有利于处理多个页面的样式。这一章后面会介绍这是如何做到的。

BRAIN POWER

假设在段落中有一个元素。如果改变了段落的背景色，你是不是认为还要改变元素的背景色，才能与段落的背景色一致？

把CSS放入HTML

你现在已经知道一点CSS语法了。你知道如何选择一个元素，然后写一个规则，其中包含属性和属性值。不过，你还需要把这个CSS放到某个HTML中。首先，我们需要一些HTML来放入CSS。在后面几章中，还会再去拜访我们的老朋友——Starbuzz咖啡馆，还有Tony和他的Segway之旅日志网站，当然会让这些网站更有表现力。不过，你认为谁的愿望最迫切，希望最先增加样式？当然是Head First休闲室的那些人。所以，下面给出Head First休闲室主页的HTML。应该记得，上一章我们做过一些修复工作，让它成为正确的HTML（我们真是好人，对不对）。现在来增加一些样式标记，这是为页面增加样式的最容易的方法。

不过，不一定是最好的办法。这一章后面还会再来讨论这个问题，介绍另一种方法。

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge</title>

    <style>

    </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>
```

这是我们感兴趣的东西：<style>元素。

要为HTML直接增加CSS样式，需要在<head>元素中增加开始和结束style标记。

你的CSS规则要放在这里。

为休闲室增加样式

既然在HTML中增加了<style>元素，下面再为休闲室增加一些样式，使你对编写CSS规则有一点具体认识。这个设计可能不会让你赢得“设计大奖”，不过你可以从这里起步。

首先要做的是改变段落中文本的颜色（要与休闲室里那些红色沙发搭配）。为此，可以使用CSS color属性，如下所示：

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge</title>
    <style>

      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 server).
    </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>
```

这个规则将指定段落的字体颜色。

改变字体颜色的属性名为“color”（你可能以为是“font-color”或“text-color”，可惜并不是）。

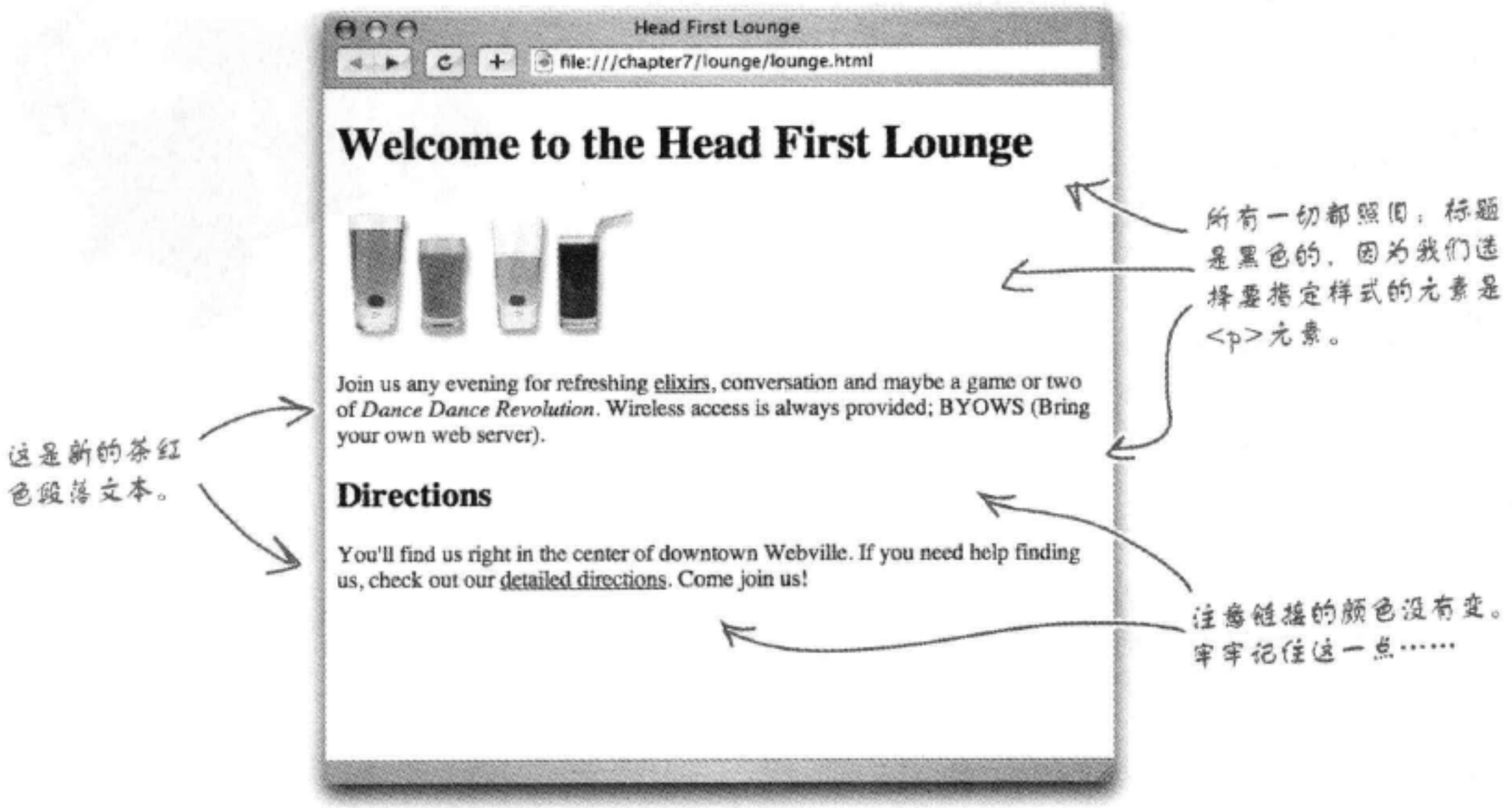
我们选择<p>元素来应用这个样式。

我们要把文本颜色设置为可爱的茶红色，这正好与休闲室沙发的颜色搭配。

p选择器会选择HTML中的所有段落。

试一试：展现样式

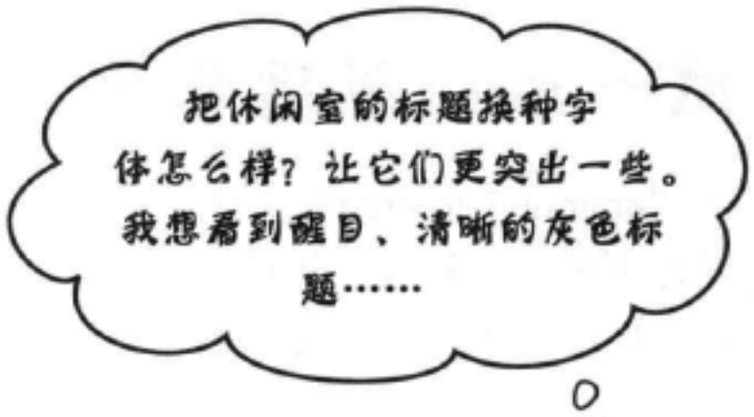
对“chapter7/lounge”文件夹中的“lounge.html”文件完成前面几页的所有修改，然后保存，并在浏览器中重新加载这个页面。你会看到段落的文本颜色已经改为茶红色：



如果不设置color，而是要把<p>元素的background-color设置为maroon会怎么样呢？浏览器显示页面时会有变化吗？

对标题加样式

现在为这些标题加一些样式。把字体稍做改变怎么样？下面就来改变标题的字体和颜色：




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

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

p {
     color: maroon;
}
```

这个规则会选择<h1>元素，将font-family改为sans-serif，将字体颜色改为gray。稍后还会更详细地讨论字体。

这是另一个规则，对<h2>元素做同样的设置。

实际上，由于这些规则是一样的，可以把它们合并起来，就像这样：

要为多个元素编写一个规则，只需要在选择器之间加上逗号，如“h1, h2”。

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

p {
     color: maroon;
}
```

试一试……

把这个新CSS增加到你的“lounge.html”文件中，然后重新加载页面。你会看到，只需要一个规则，可以同时选择<h1>和<h2>标题。

现在页面上的两个标题都指定了新样式，字体为sans-serif，颜色为灰色。

再在欢迎消息下面加一条线

下面再对欢迎标题稍稍做些处理。在下面加一条线怎么样？这样可以在视觉上让主标题更为显眼，效果更好。我们要使用下面这个属性来达到这个目的：

```
border-bottom: 1px solid black;
```

这个属性控制元素下边框的外观。

要设置下边框的样式：粗细为1像素，黑色，实线。

这里有个小麻烦，如果在CSS的“h1, h2”合并规则中增加这个属性和属性值，那么最后两个标题都会加上边框：

```
h1, h2 {
     font-family:   sans-serif;
     color:         gray;
     border-bottom: 1px solid black;
}

p {
     color: maroon;
}
```

这里增加一个属性，这会同时改变<h1>和<h2>元素的下边框。

如果这样做……

……两个标题都会有下边框。

所以，能不能只对<h1>元素设置下边框，而不影响<h2>元素？如何做到？必须把规则再分开吗？要想知道答案，请翻到下一页……

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。

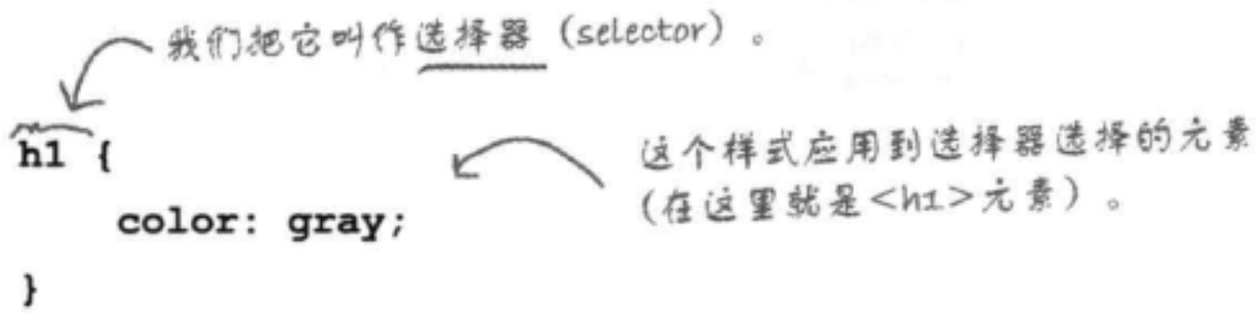
答：不见得。如果把共同的样式合并在一起，倘若它们有改变，你只需要在一个规则中修改。如果把它们分开，就必须修改多个规则，这很容易出错。

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？

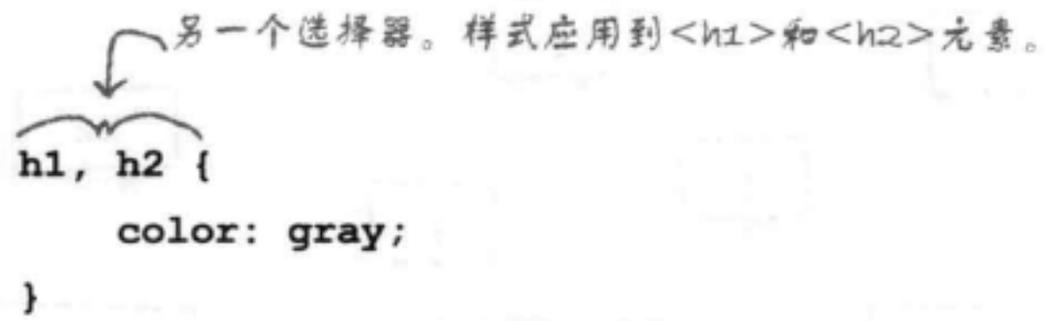
答：问得好。文本确实有一个underline（下划线）样式，我们当然也可以使用这个样式。不过，这两个样式在页面上的效果稍有不同。如果使用border-bottom，这条线会延伸到页面边缘。而使用underline时，下划线只出现在文本下面，不会继续延伸到页面边缘。设置文本下划线的属性名为text-decoration，值为“underline”时表示文本有下划线。你可以试一试，看看有什么不同。

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：

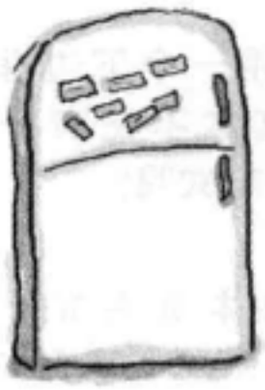


我们也见过选择多个元素的情况，如下所示：



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？

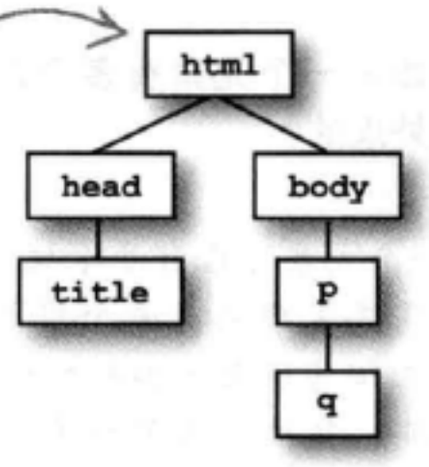
所以，先对休闲室HTML有一个明确的认识，然后再回来深入讨论选择器。

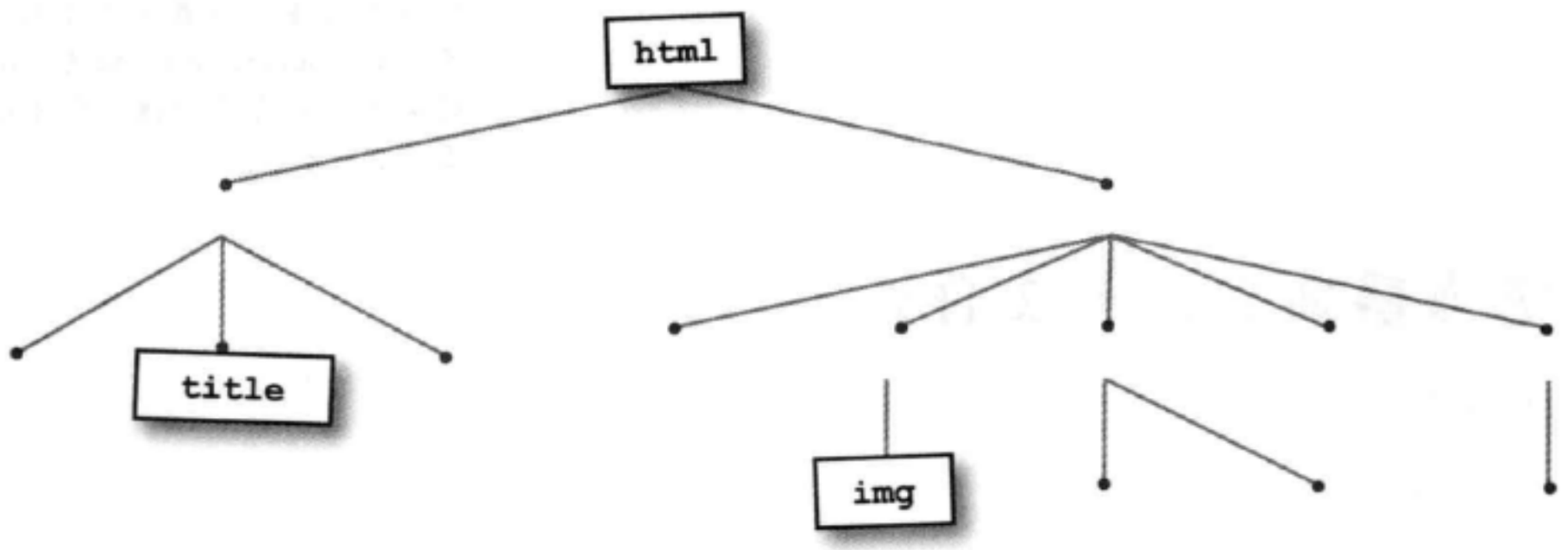


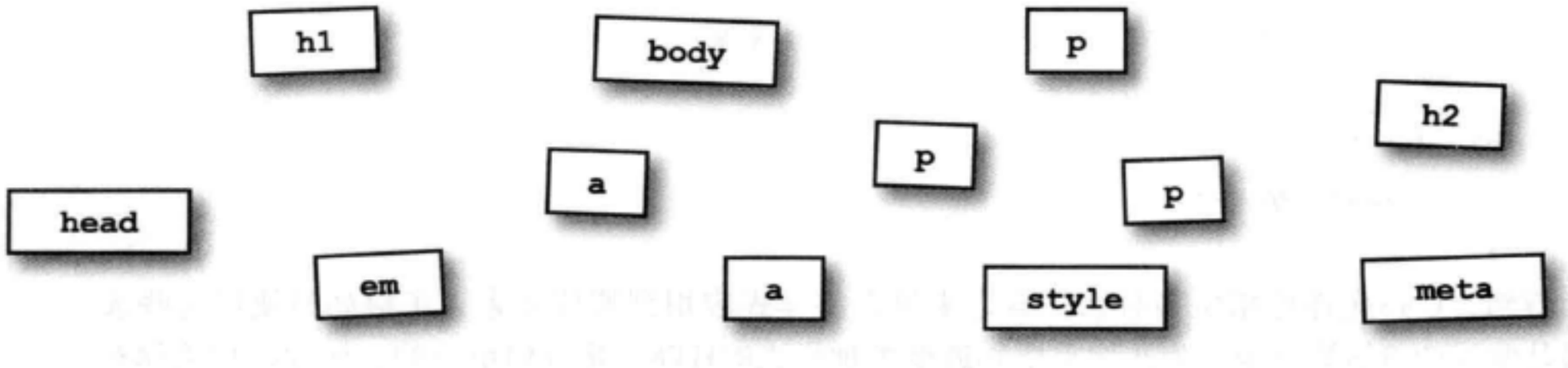
标记磁贴

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。







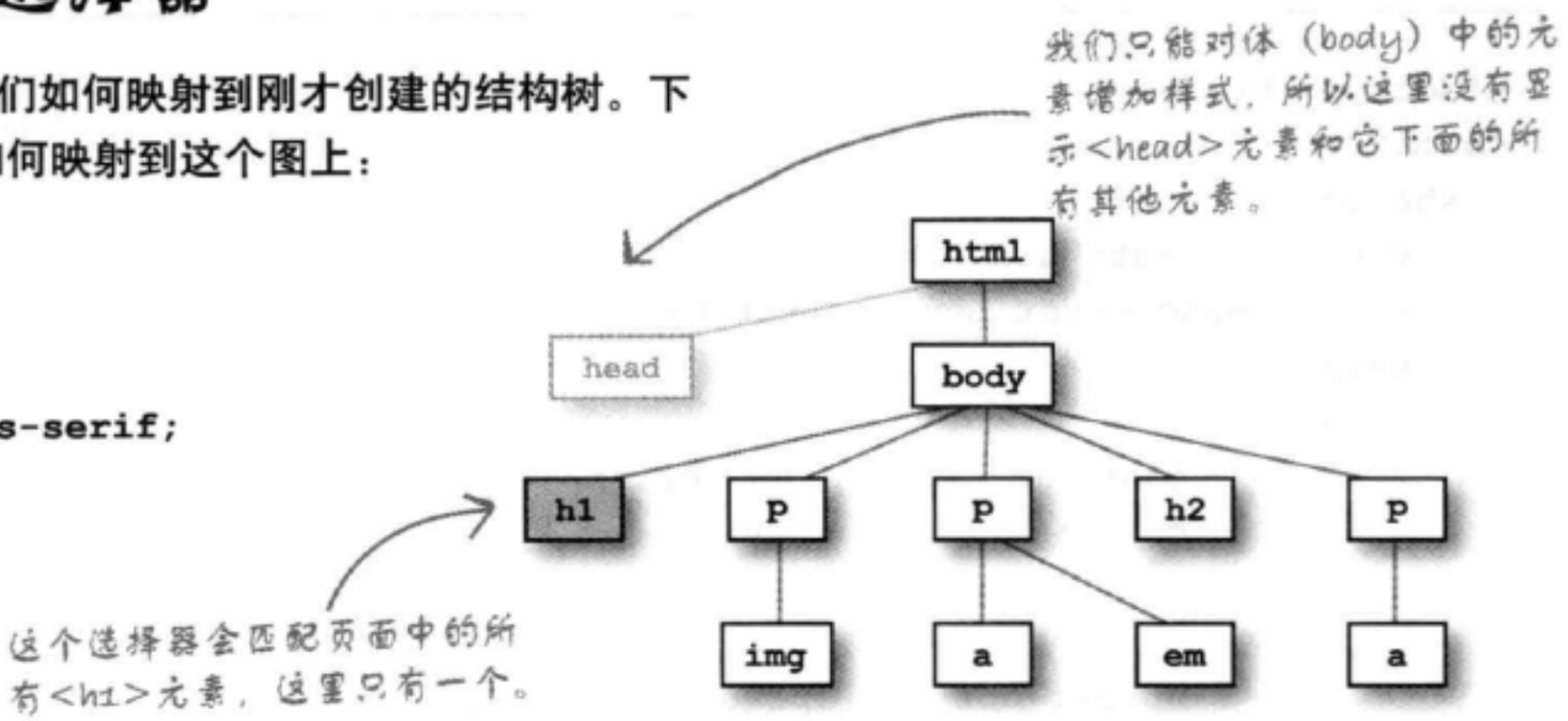
Head First休闲室的HTML。

```
<!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

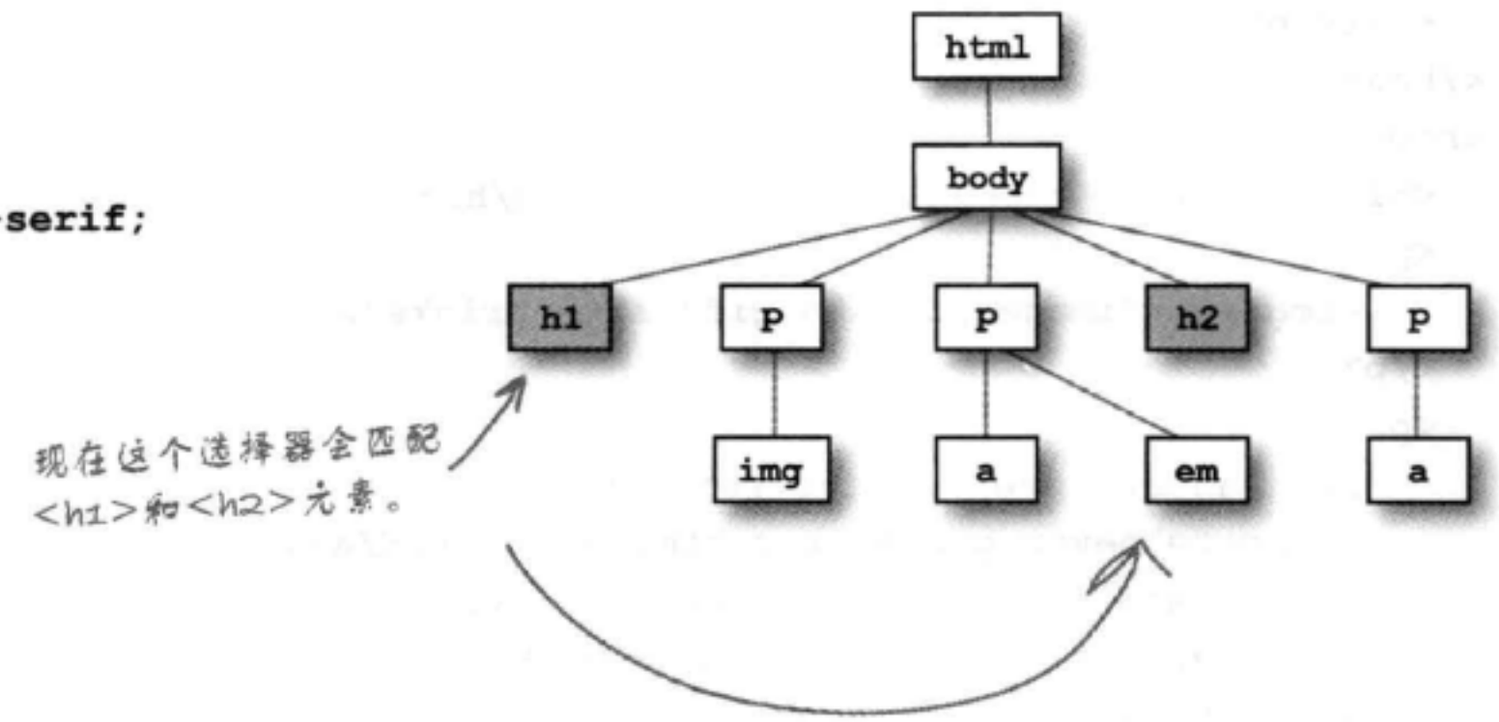
下面来看一些选择器，看看它们如何映射到刚才创建的结构树。下面显示了这个“h1”选择器如何映射到这个图上：

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



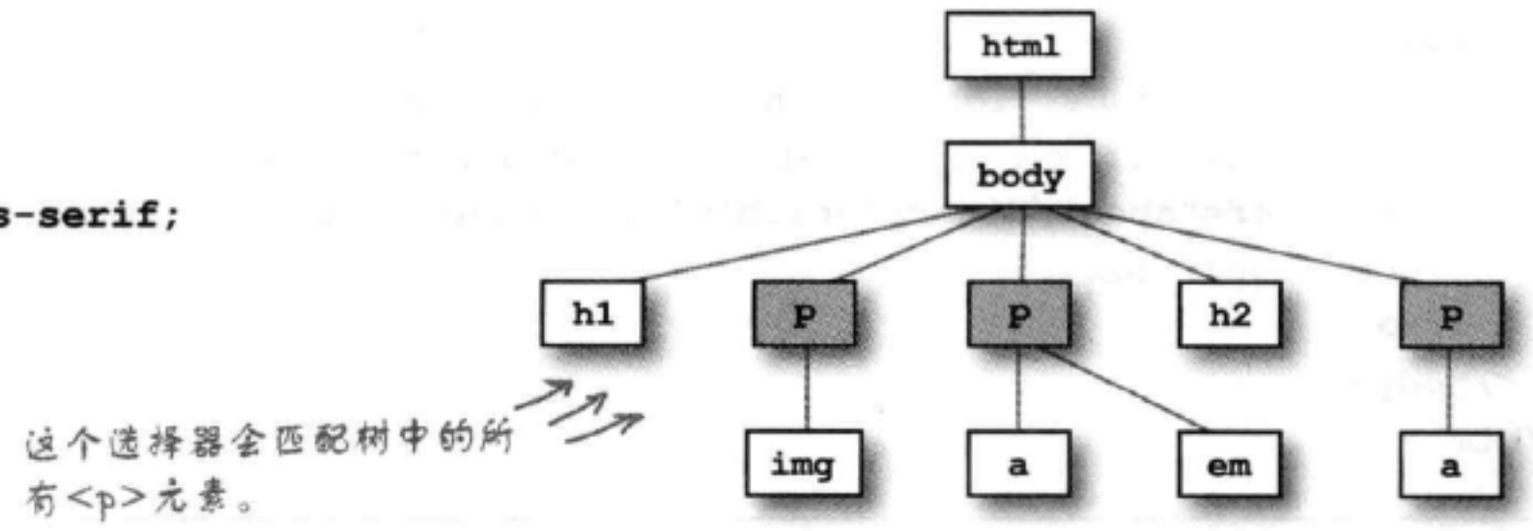
“h1, h2”选择器是这样的：

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



如果使用一个“p”选择器，就像这样：

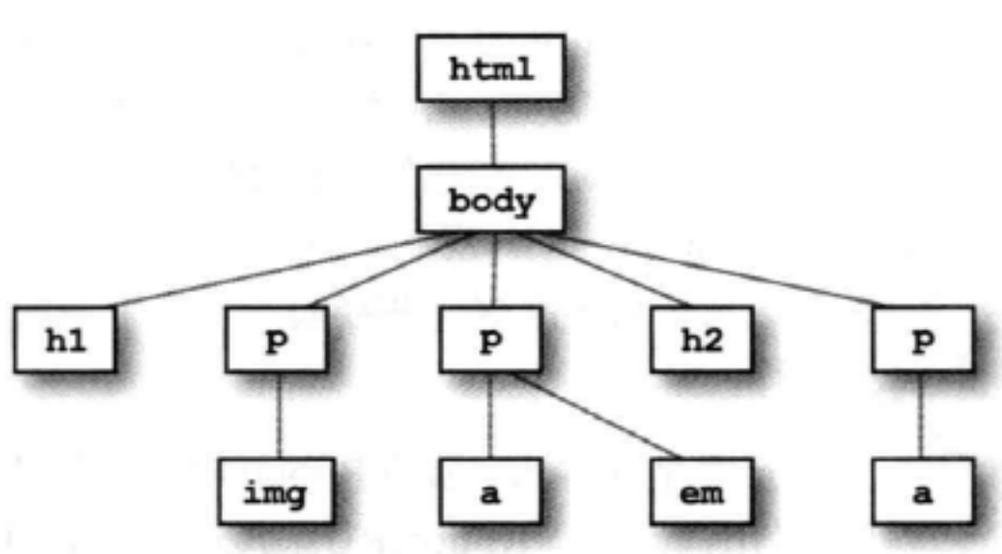
```
p {
    font-family: sans-serif;
}
```



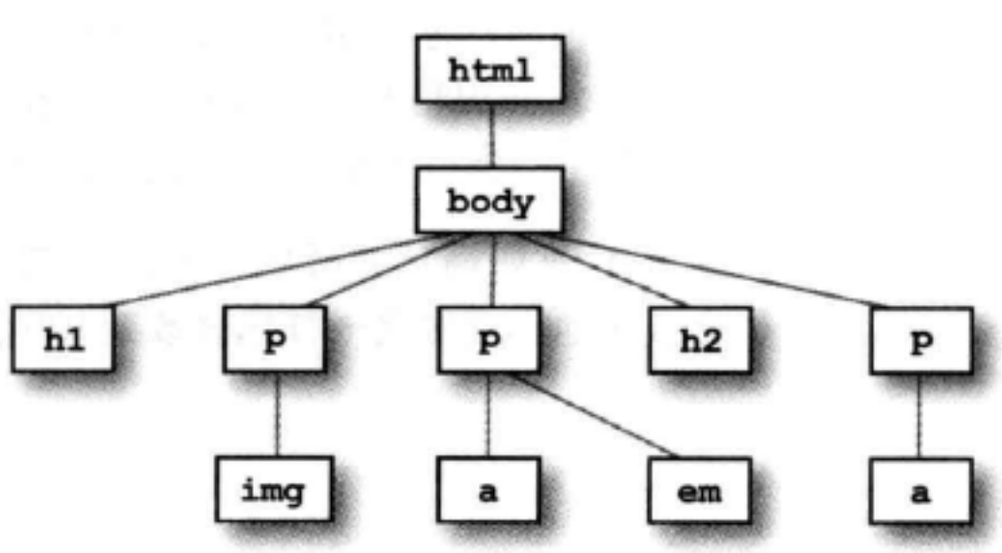
Sharpen your pencil

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？

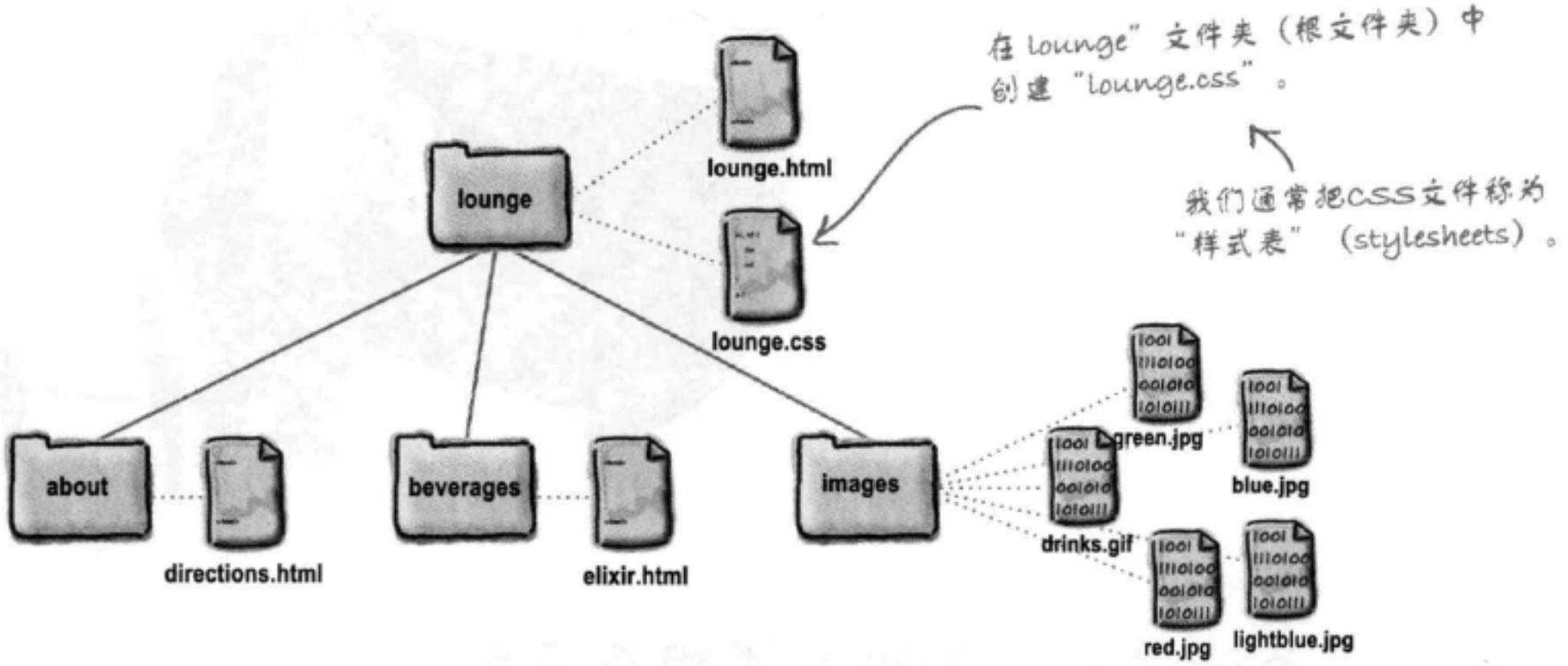
为清凉饮料和路线说明页面加入休闲室页面的样式

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。

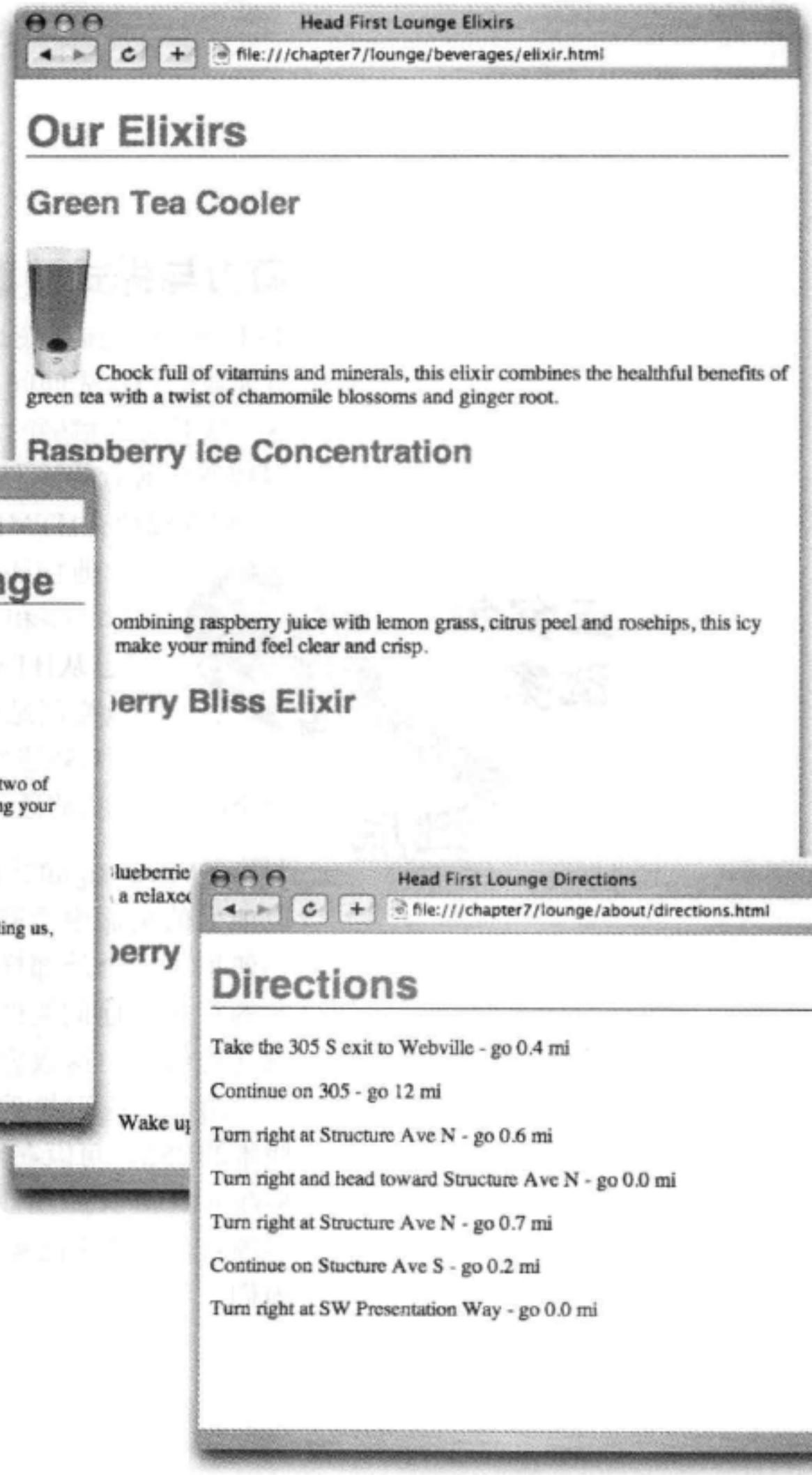
```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge Directions</title>
    <link type="text/css" rel="stylesheet" href="../lounge.css">
  </head>
  <body>
    .
    .
    .
  </body>
</html>
```

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。

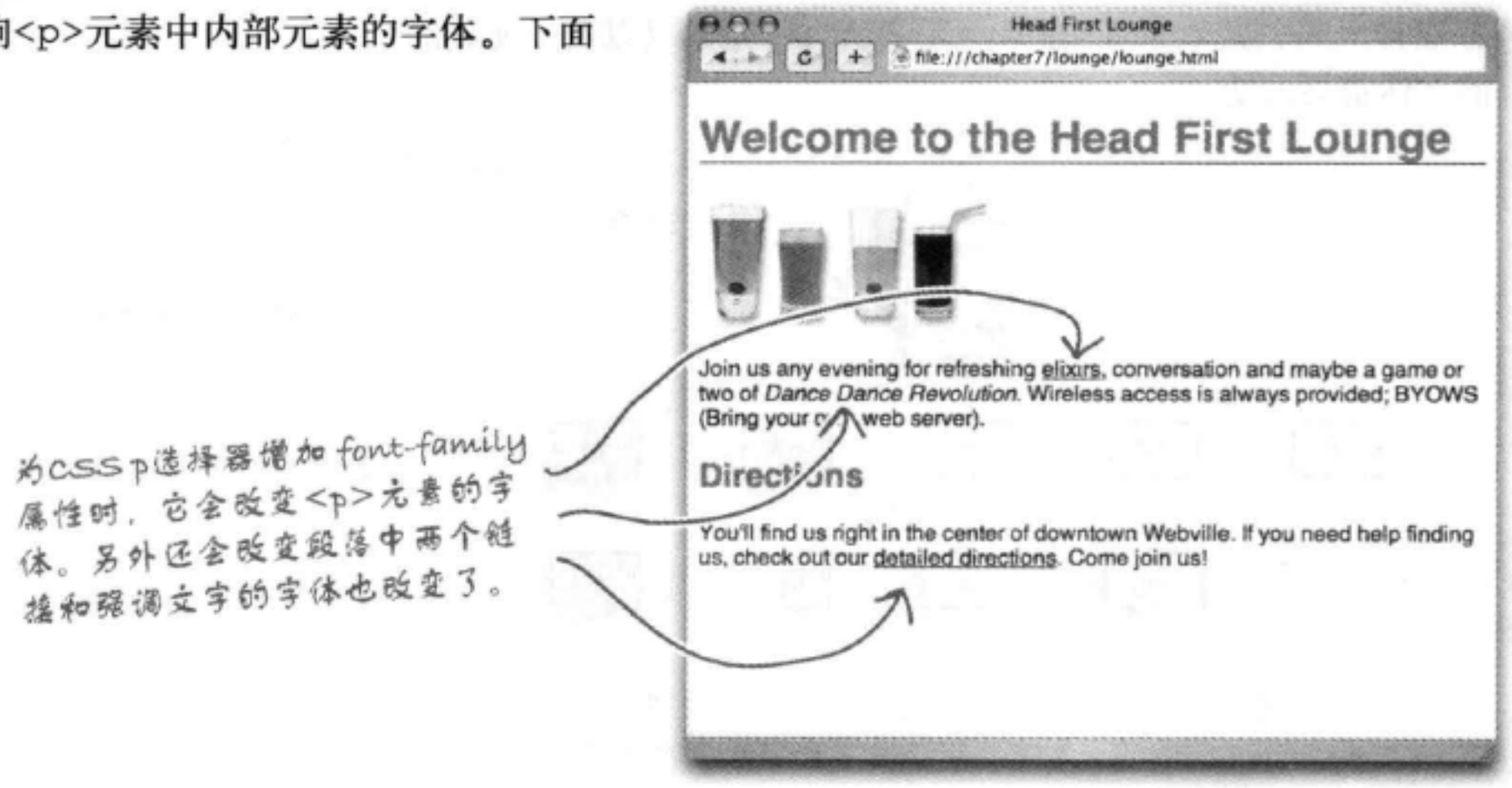
```
h1, h2 {
       font-family:  sans-serif;
       color:        gray;
}

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

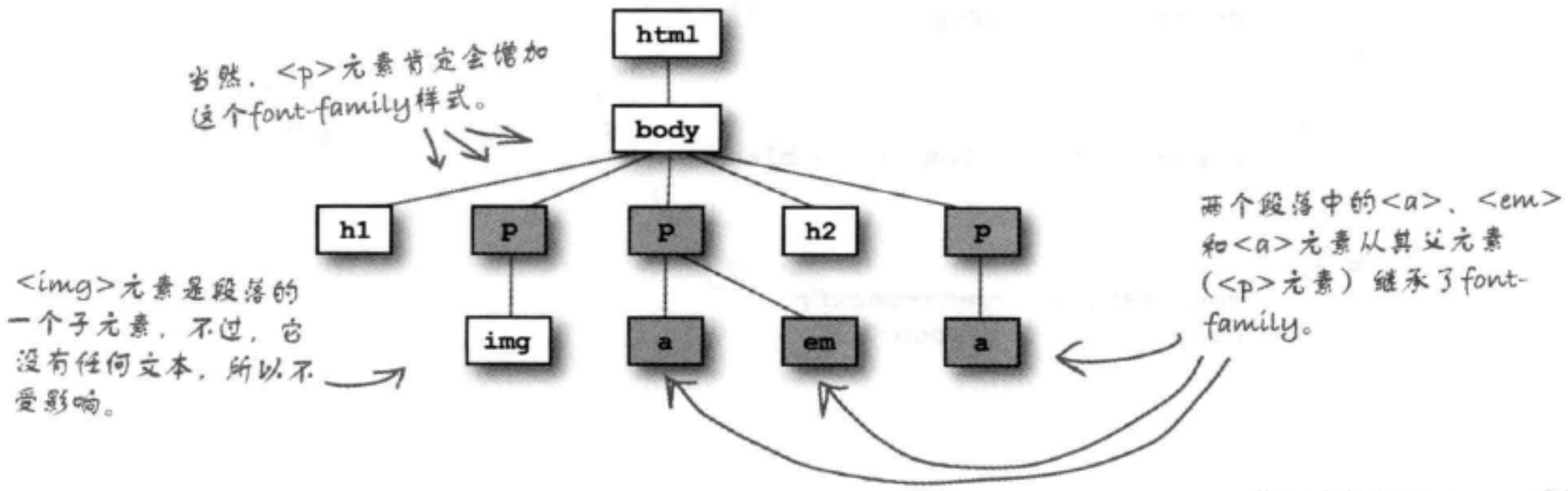
就像你可能会从你的父母那里继承蓝眼睛和金黄色的头发，同样的，元素也可以从它们的父元素继承样式。在这里，<a>和元素就从<p>元素继承了font-family样式，<p>元素就是它们的父元素。改变段落样式也会改变段落中元素的样式，这是有道理的，不是吗？毕竟，如果不是这样，你就只能事必躬亲，为整个网站中每一个段落中的每一个内联元素增加CSS规则……这太可怕了。

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。

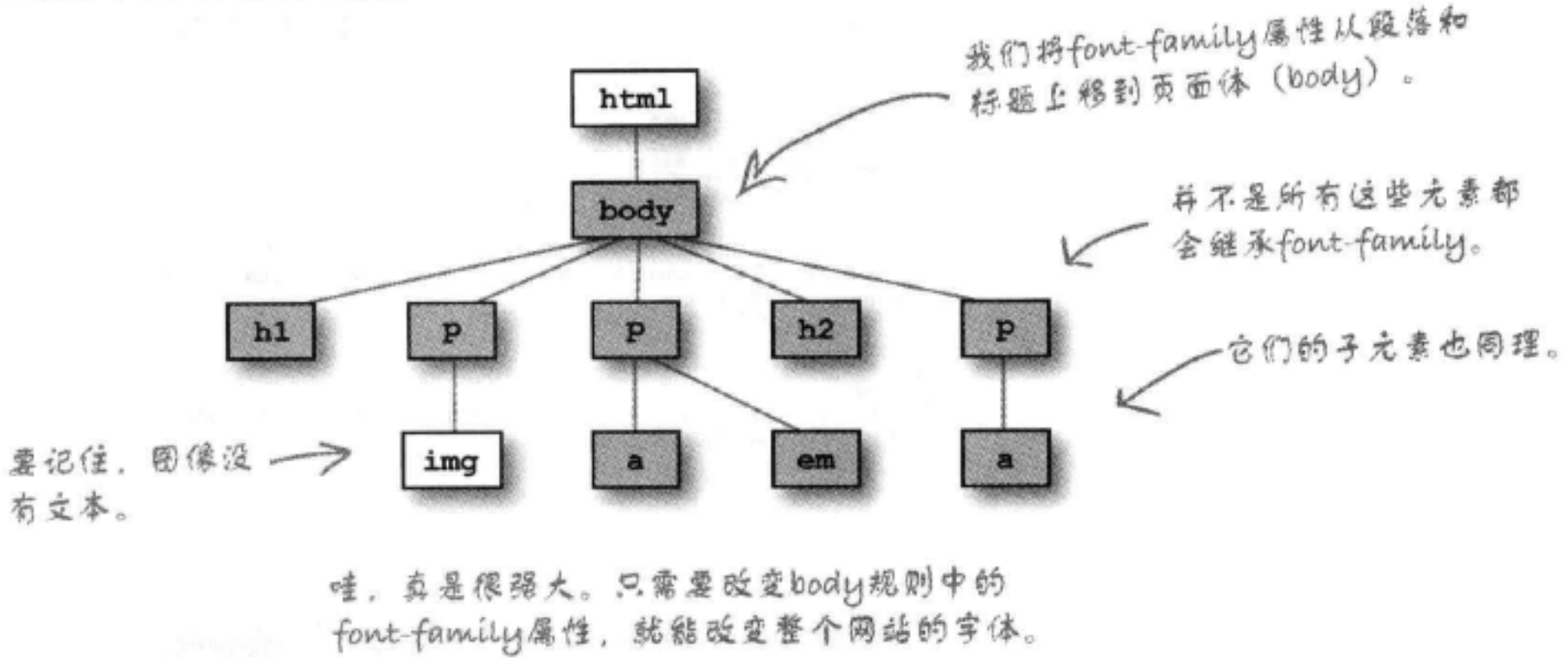
下面给出HTML树，看看继承到底是怎样工作的：

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：

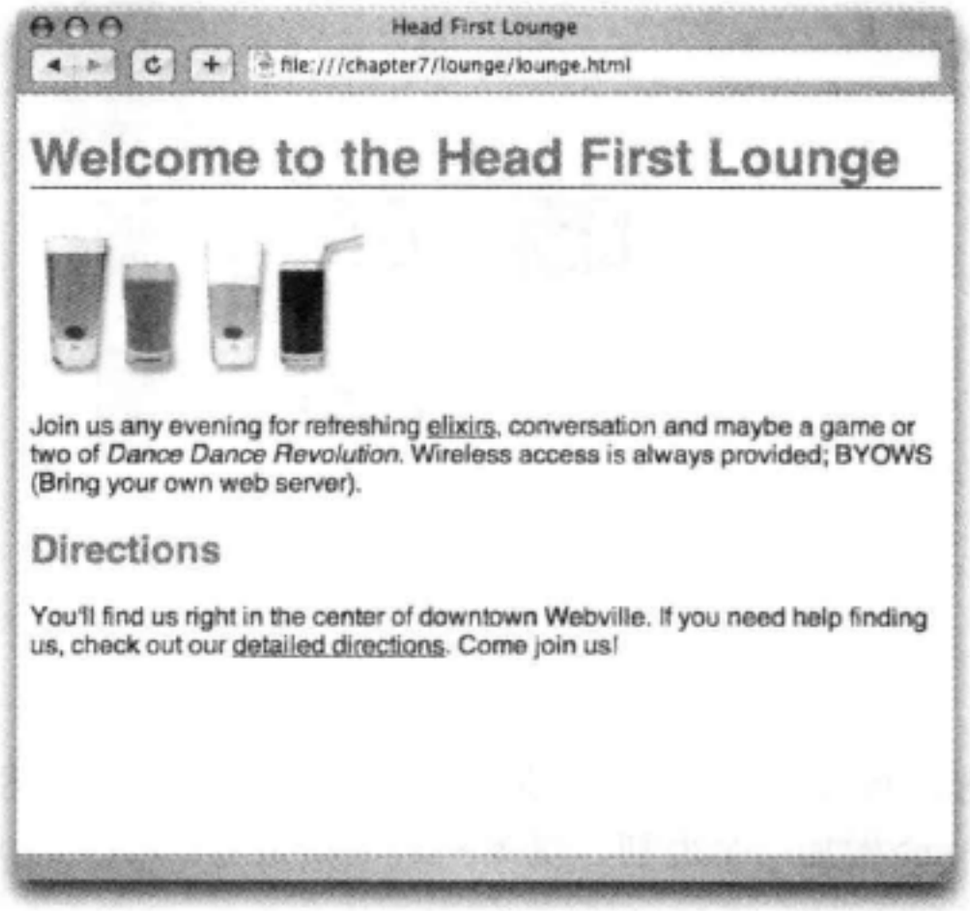
首先，增加一个新规则，选择<body>元素。然后增加font-family属性，值为sans-serif。

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

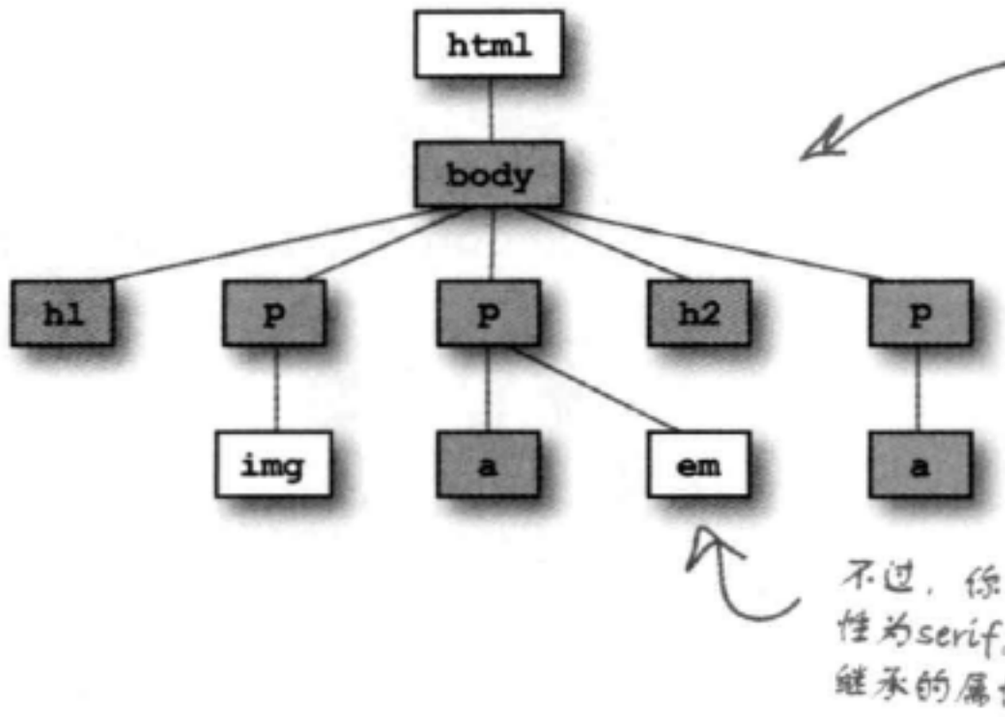
与以往一样，在“lounge.css”样式表中完成这些修改，保存，再重新加载“lounge.html”页面。你不会看到任何改变，因为样式还是一样的，只不过来自一个不同的规则。但是你会感觉你的CSS比以前好，因为现在如果向页面增加新元素，它们会自动继承sans-serif字体。

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。

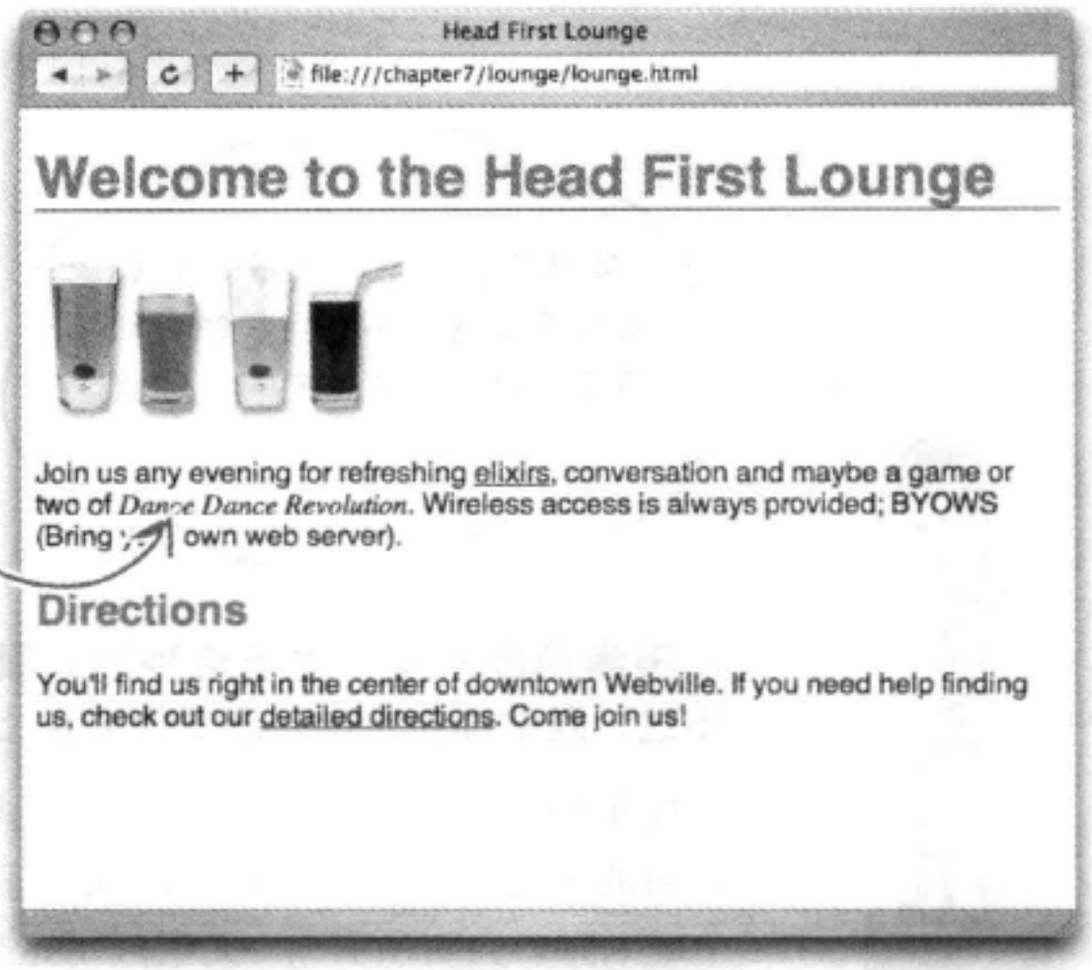
嗯，可以只为提供一个特定的规则来覆盖继承。下面为增加一个规则，覆盖body中指定的font-family：

```
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？ _____

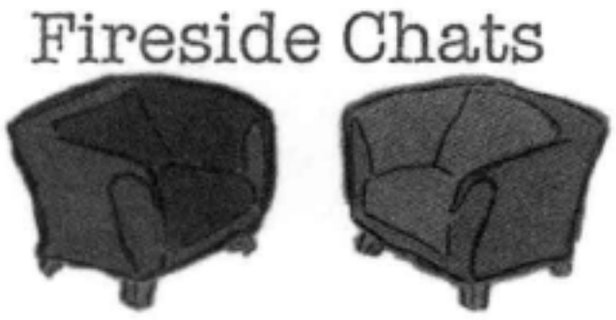
最后，把p.raspberry规则移到最下面。

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？

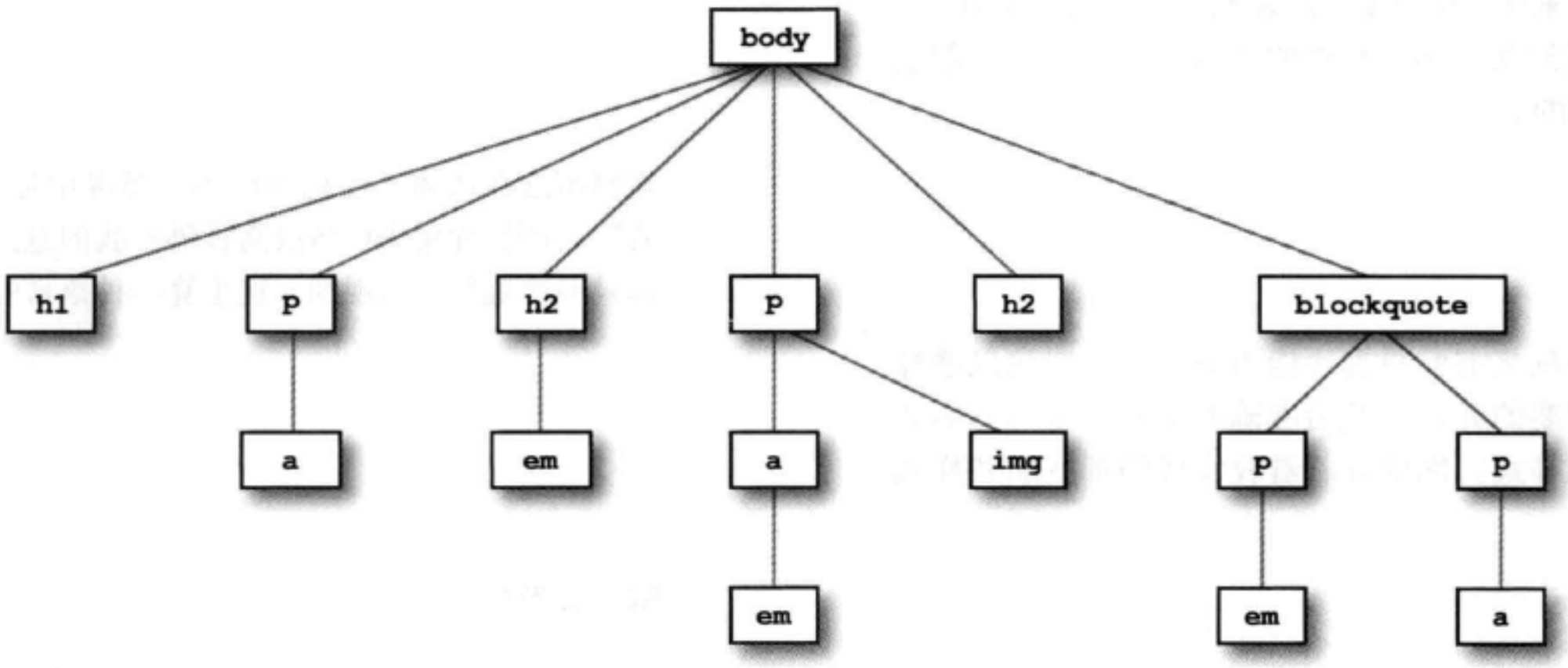
哦，是吗？

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”。

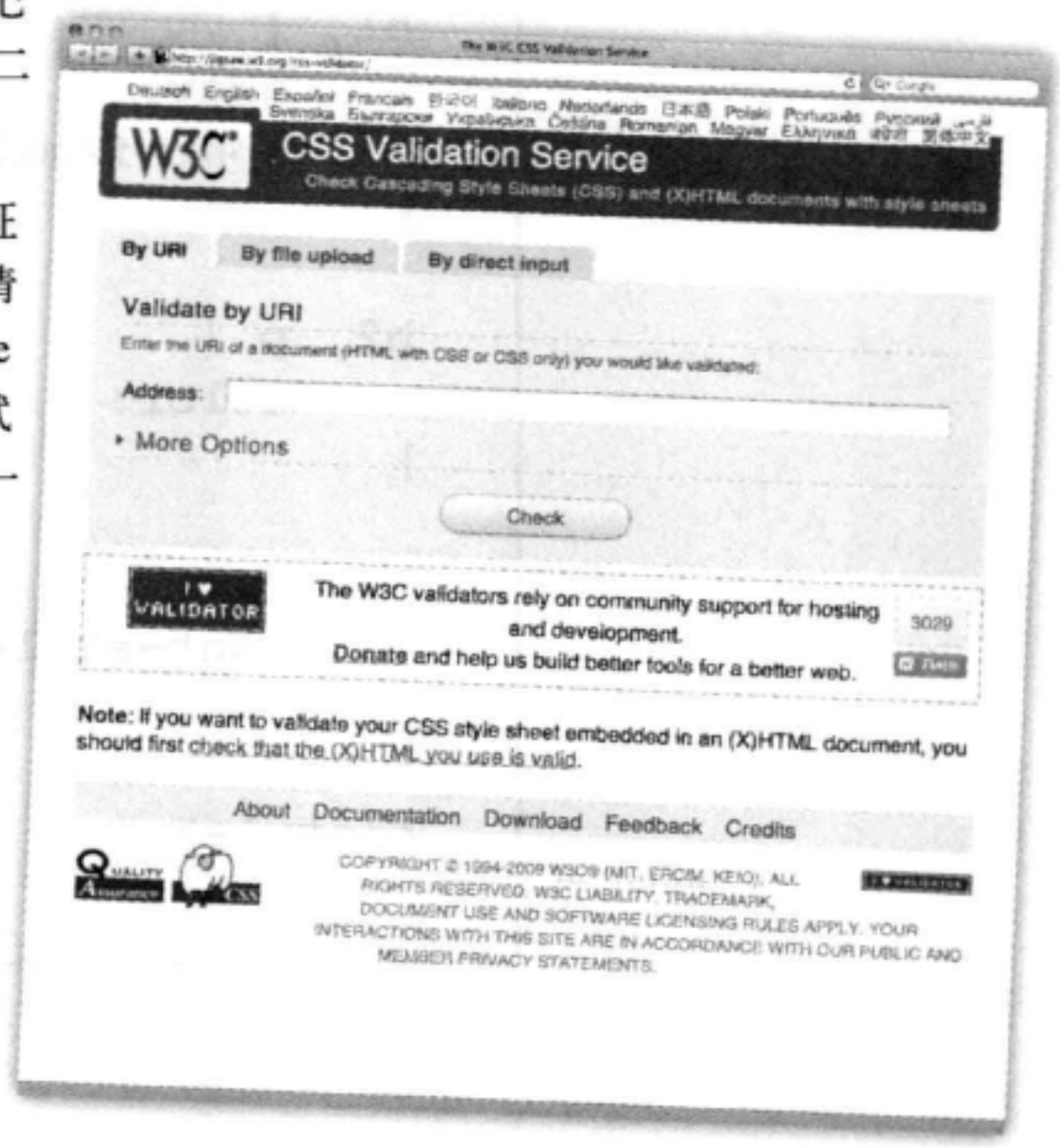
```
<style>

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）。

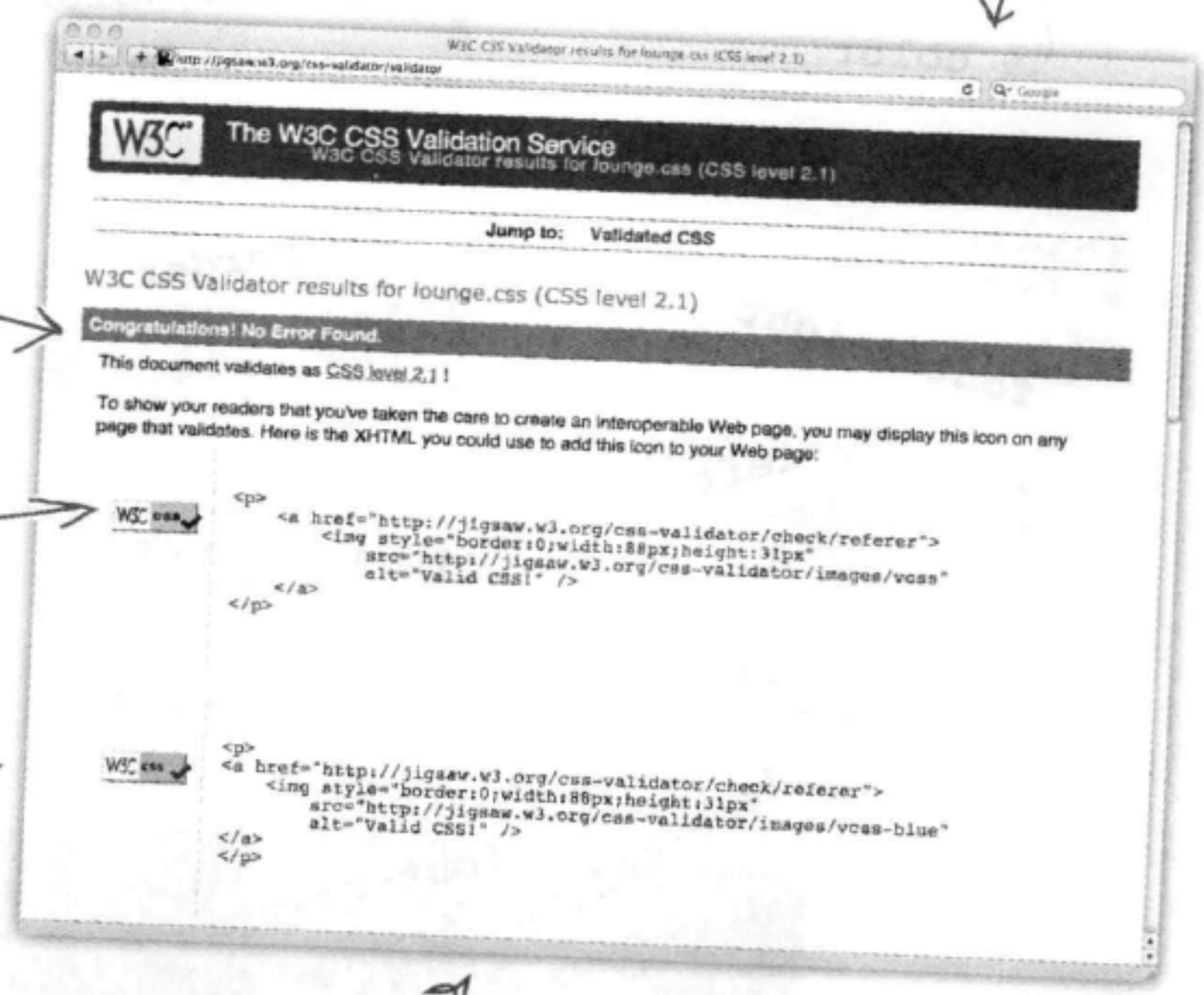
确保休闲室CSS合法

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

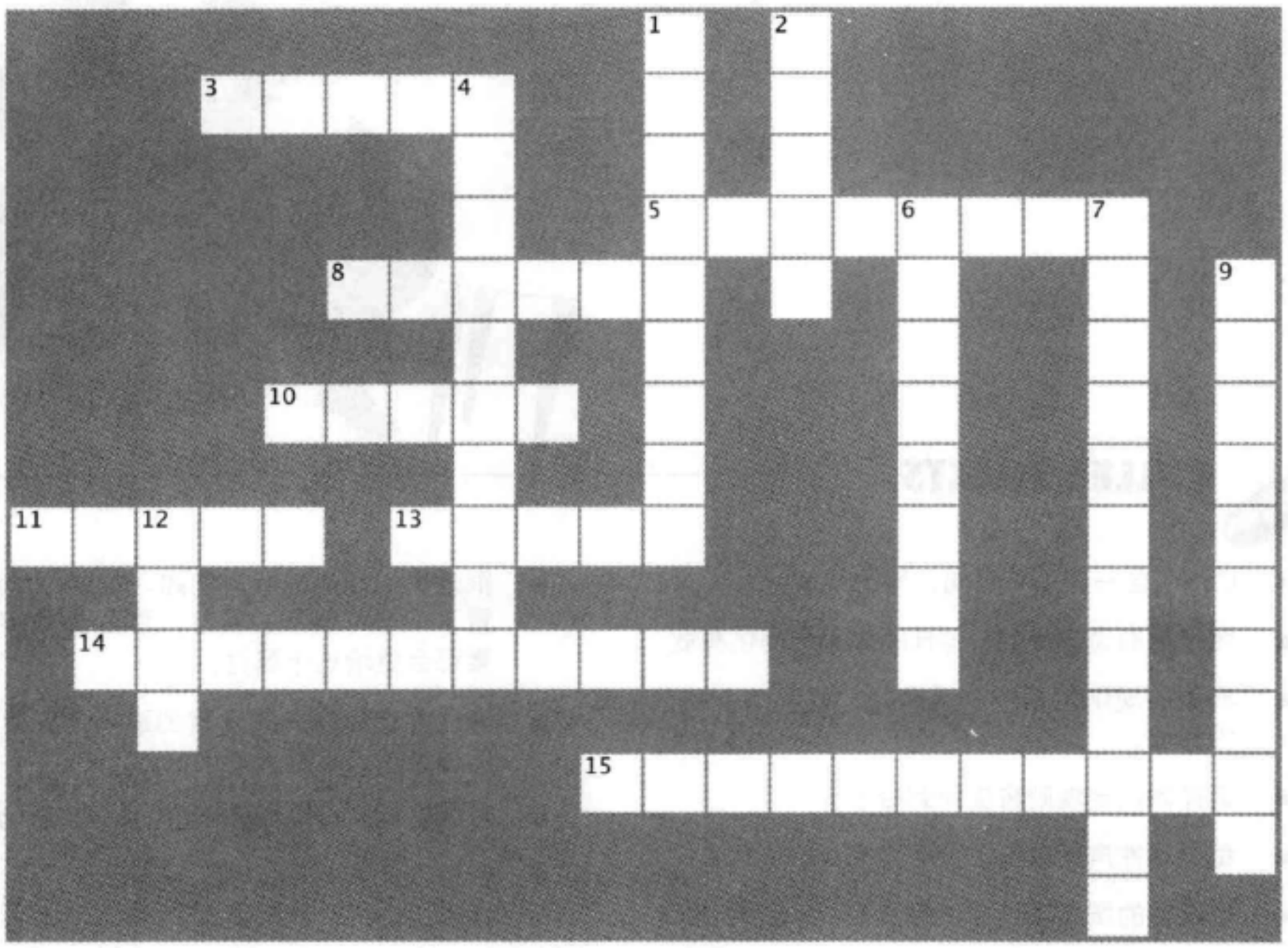
CSS有很多样式属性。这本书后面会看到其中大部分属性，不过现在可以先大体浏览一下，有一个初步的认识，大致了解利用CSS能够控制哪些方面的样式。

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!

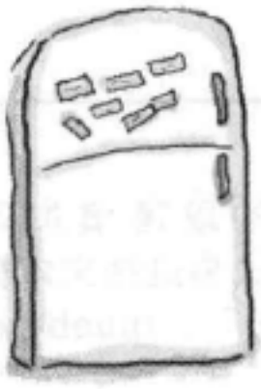


横向

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。

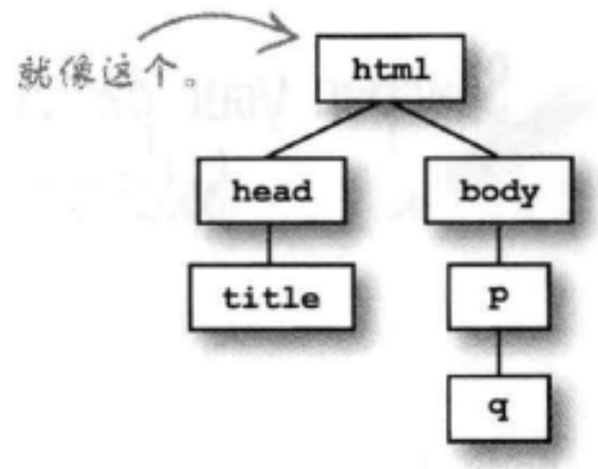
纵向

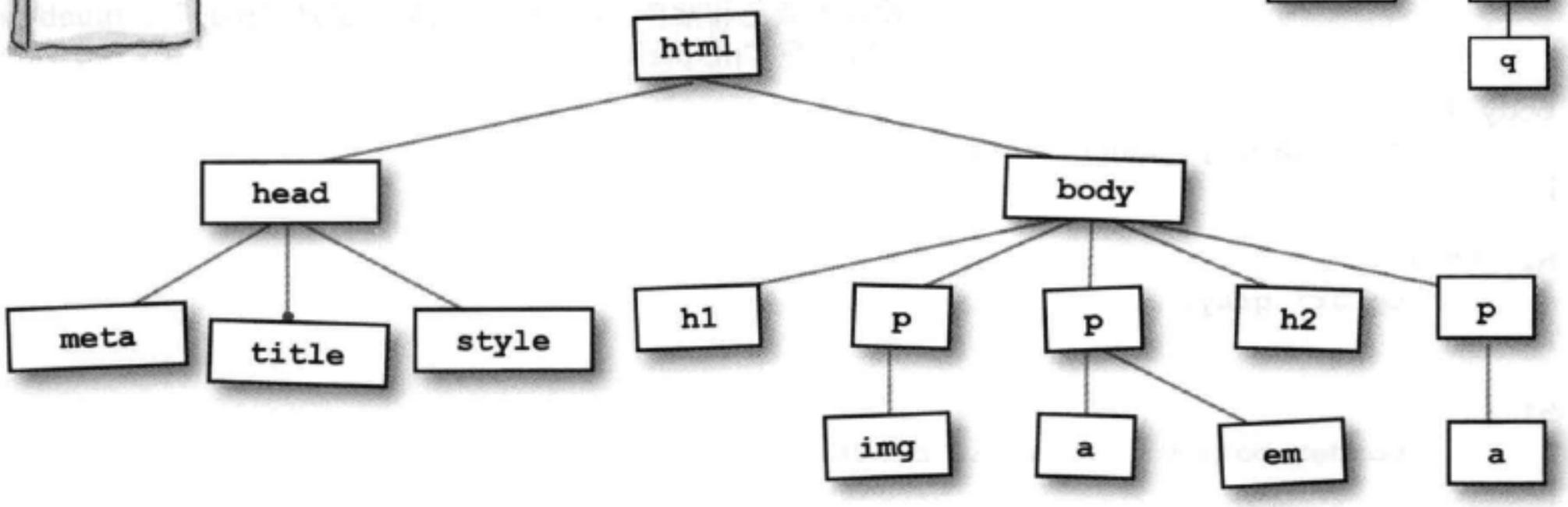
1. 没有上下衬线的字体。
2. 可以把CSS放在HTML文件的这些标记中。
4. 一个外部样式文件称为_____。
6. 利用继承，一个元素上设置的属性可以向下传递到它的_____。
7. 这一次胜出，因为他们使用了外部样式表。
9. 他们迫切希望有一些样式。
12. 使用这个元素来包含一个外部样式表。



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。

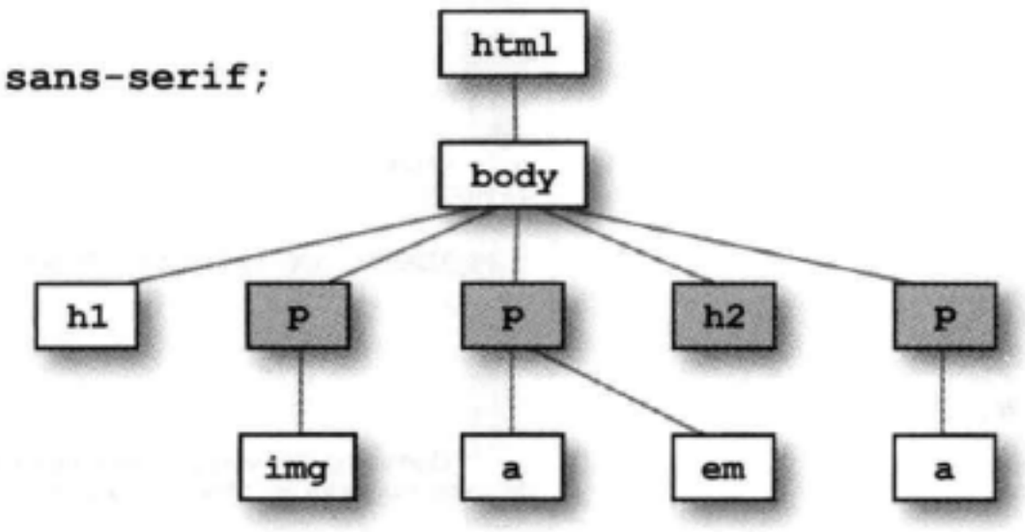




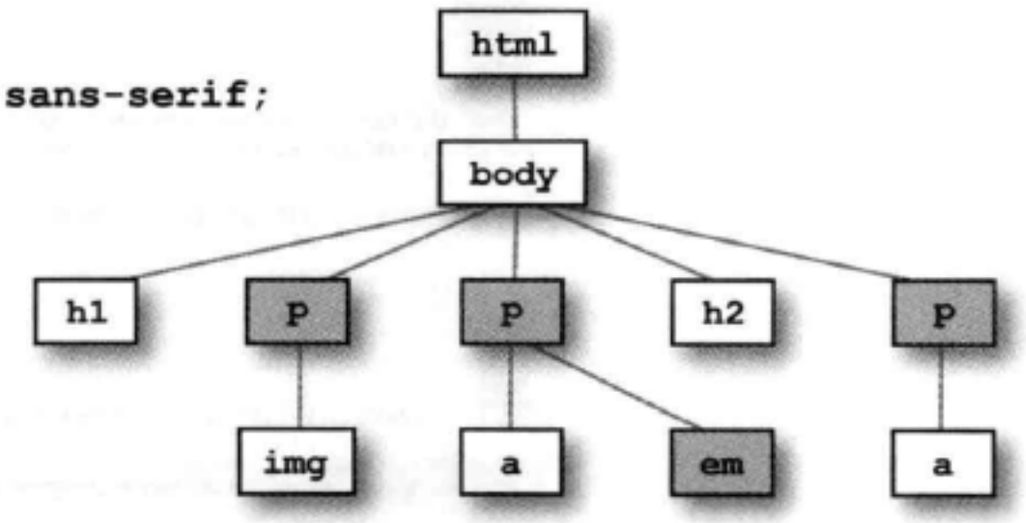
Sharpen your pencil Solution

所选的元素已经涂上颜色。

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



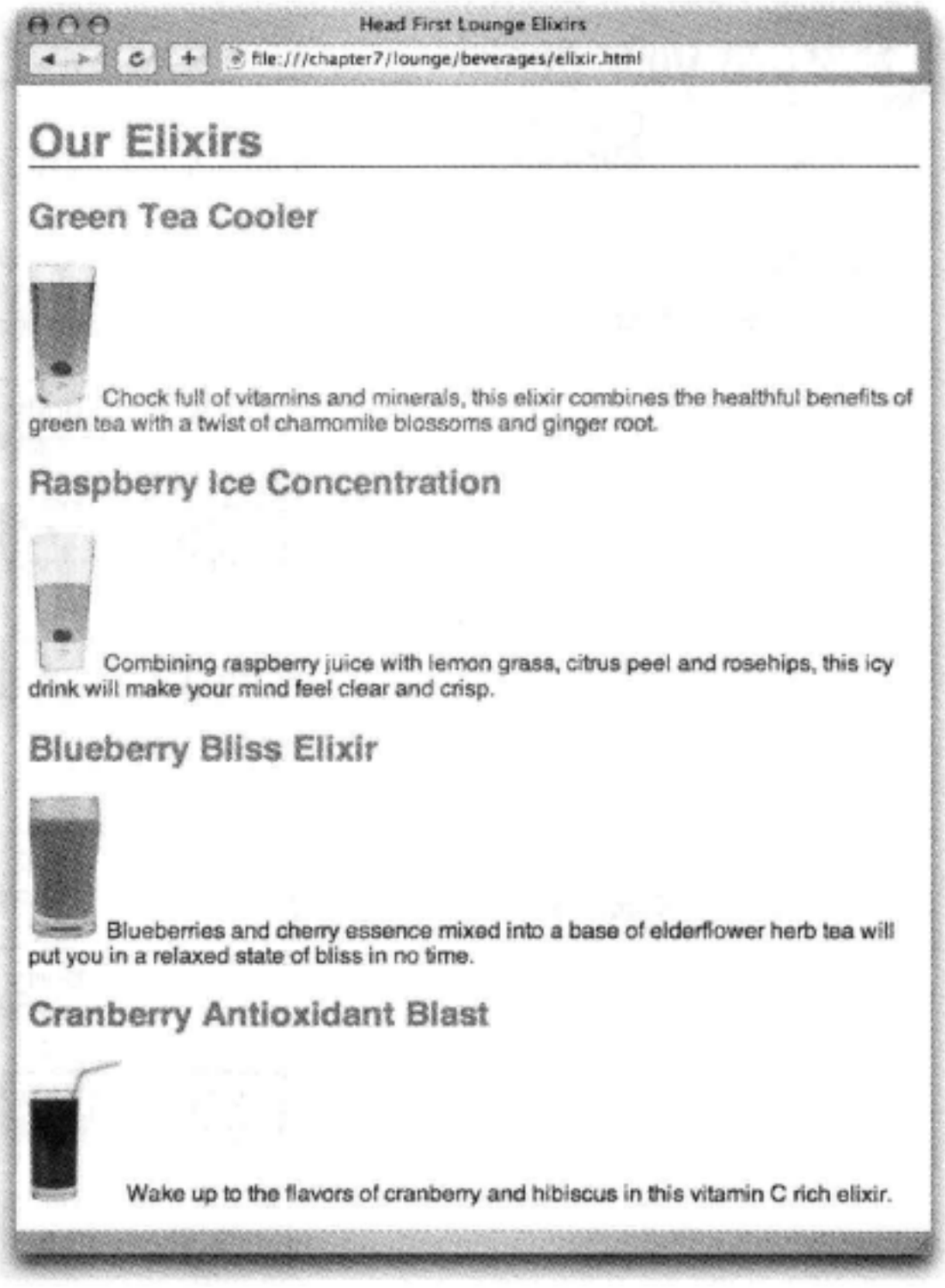

```
p, em {
    font-family: sans-serif;
}
```



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;
}
```



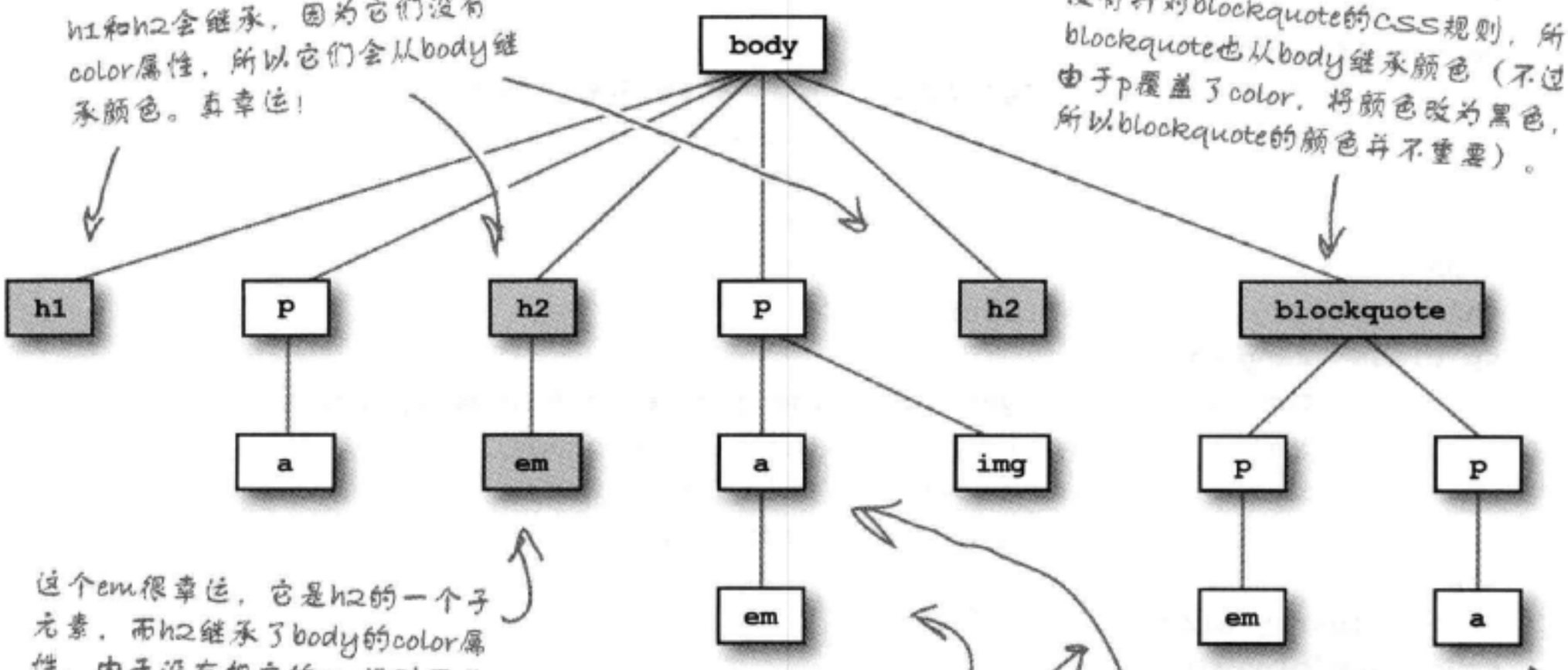
```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Head First Lounge Elixirs</title>
    <link type="text/css" rel="stylesheet" href="../lounge.css">
  </head>
  <body>
    <h1>Our Elixirs</h1>
    <h2>Green Tea Cooler</h2>
    <p class="greentea">
          <img src="../images/green.jpg" 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。

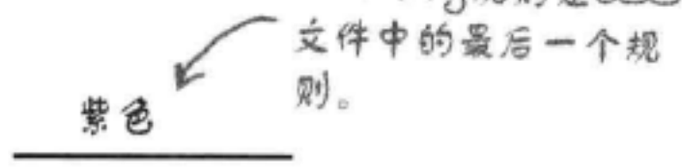
使用了HTML标记而不是元素名。这里应该是em。

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">
```

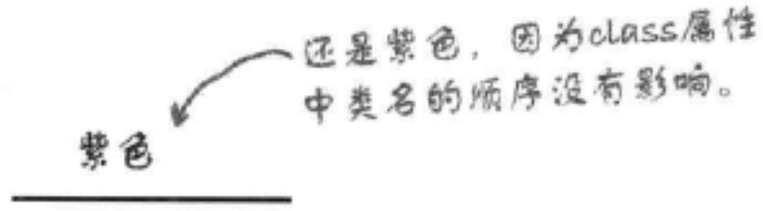
保存并重新加载，现在Green Tea Cooler段落是什么颜色？



接下来，调整HTML中类的顺序：

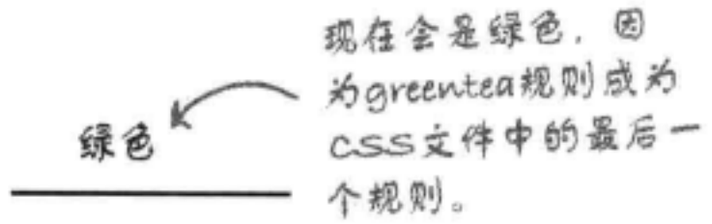
```
<p class="raspberry blueberry greentea">
```

保存并重新加载，现在Green Tea Cooler段落是什么颜色？



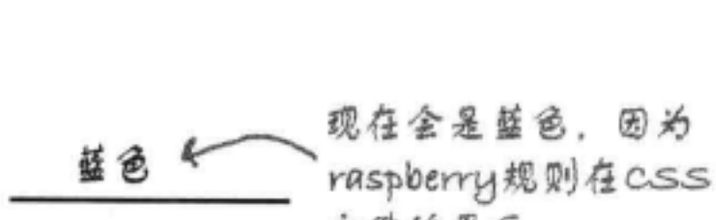
接下来，打开你的CSS文件，把p.greentea规则移到文件最下面。

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。

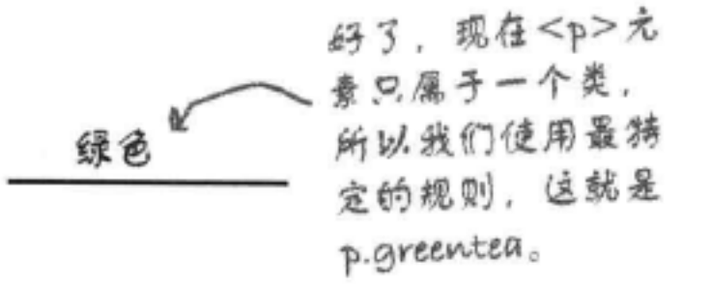
保存并重新加载，现在Green Tea Cooler段落是什么颜色？



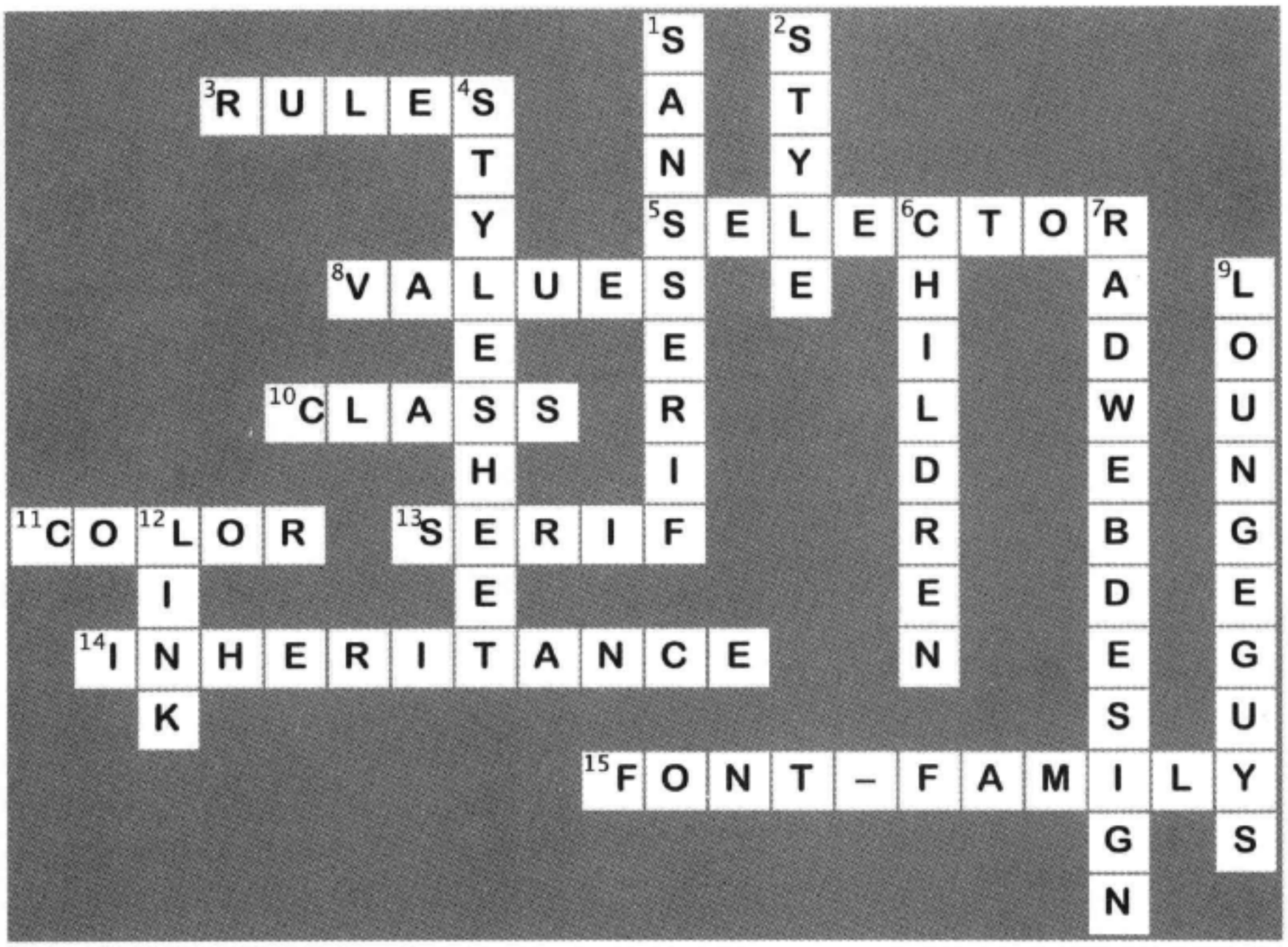
完成之后，把greentea元素重写为原来的样子：

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

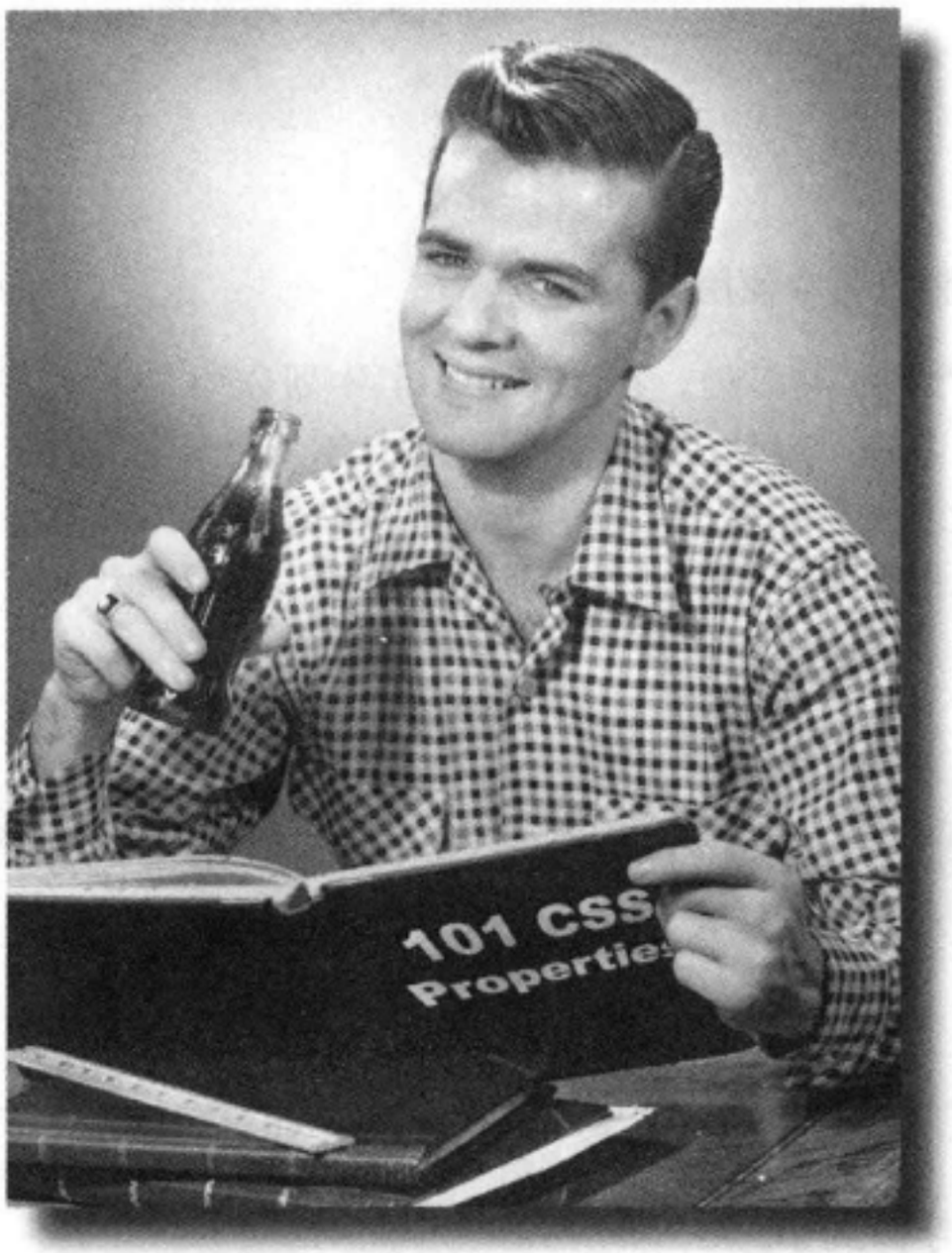
保存并重新加载，现在Green Tea Cooler段落是什么颜色？



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。

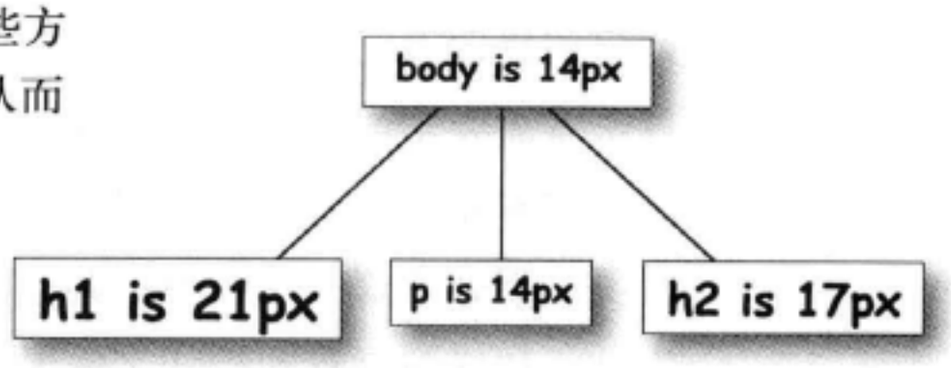
字体会对页面设计产生巨大影响。在CSS中，字体划分为“字体系列”（font family），你可以从中指定希望页面中各个元素使用的字体。大多数计算机上通常只安装了部分字体，所以在选择字体时要特别当心。为了有效地指定字体和最大程度地利用字体，这一章中我们会带着你学习时需要了解的全部知识。

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。

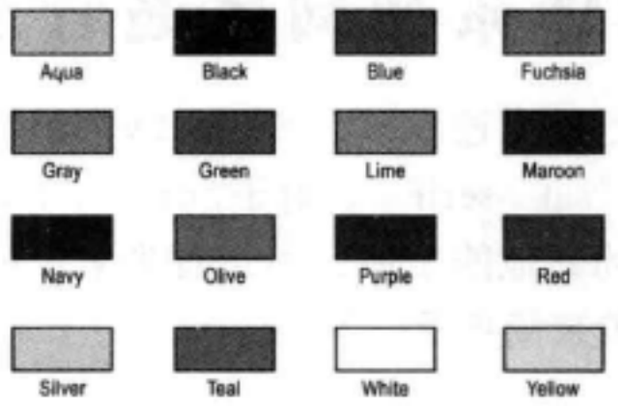



```
body {
      font-size: 14px;
}
```

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

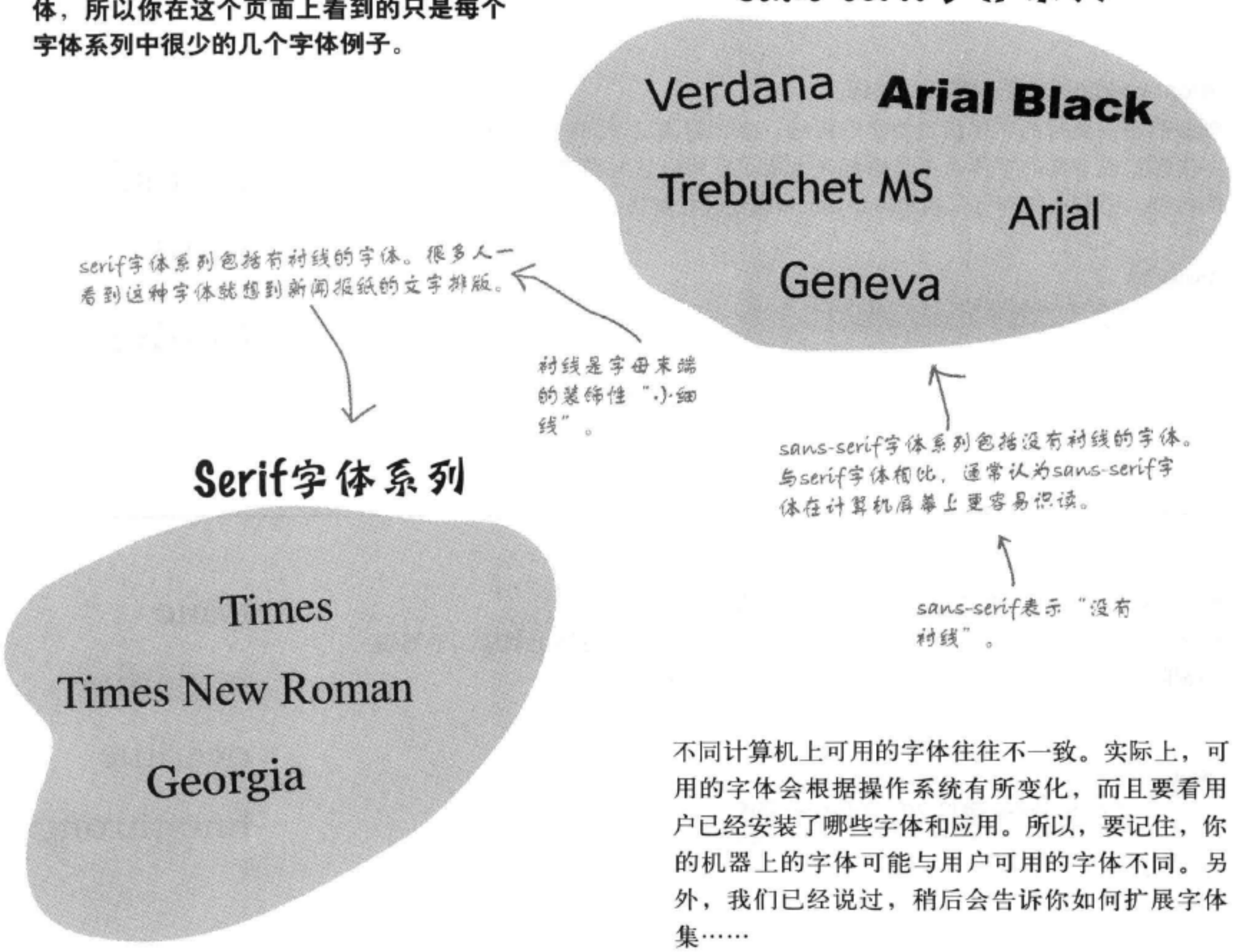
overline

~~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;
}
```

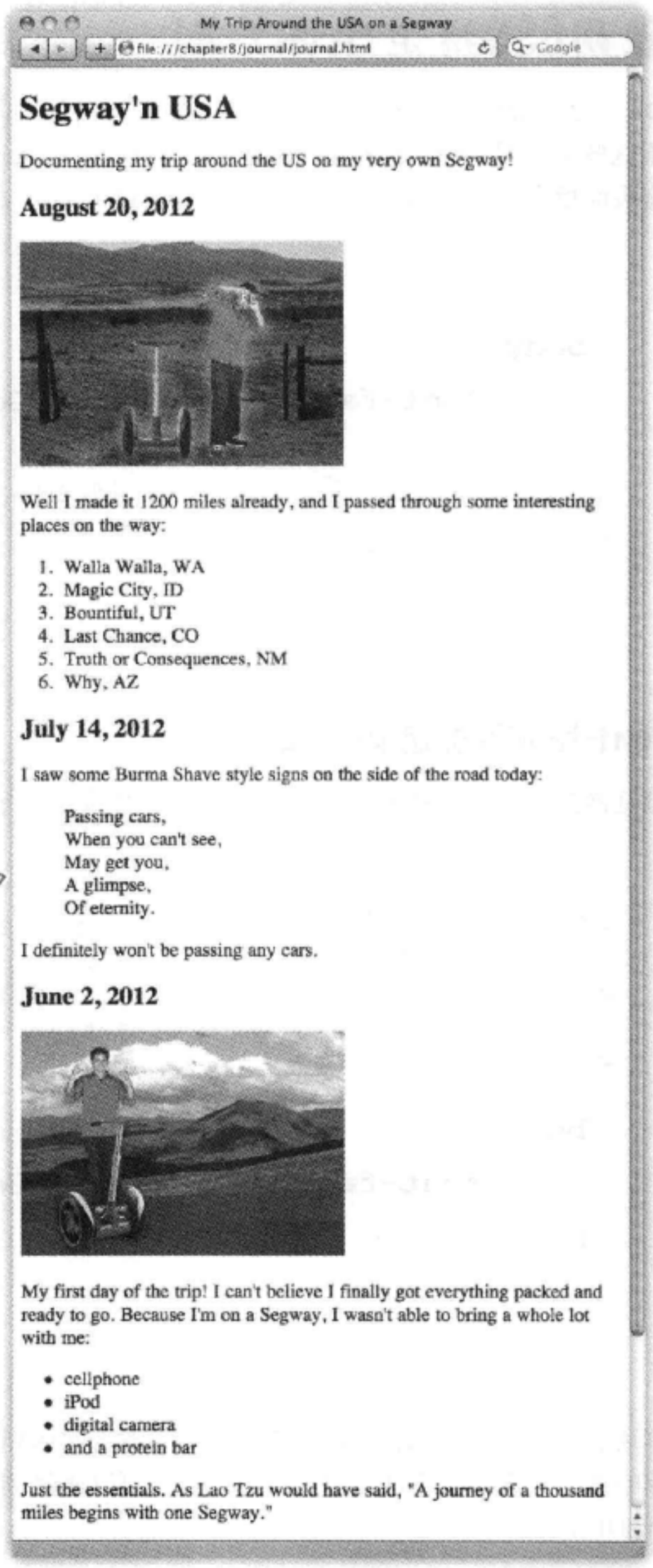
没有必要非得指定4种候选字体。你也可以指定2种、3种等。上一章中，我们只使用了一个字体（就是默认的sans-serif字体），不过我们不建议这么做，因为这样就没办法对所要使用的字体做太多控制。

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……

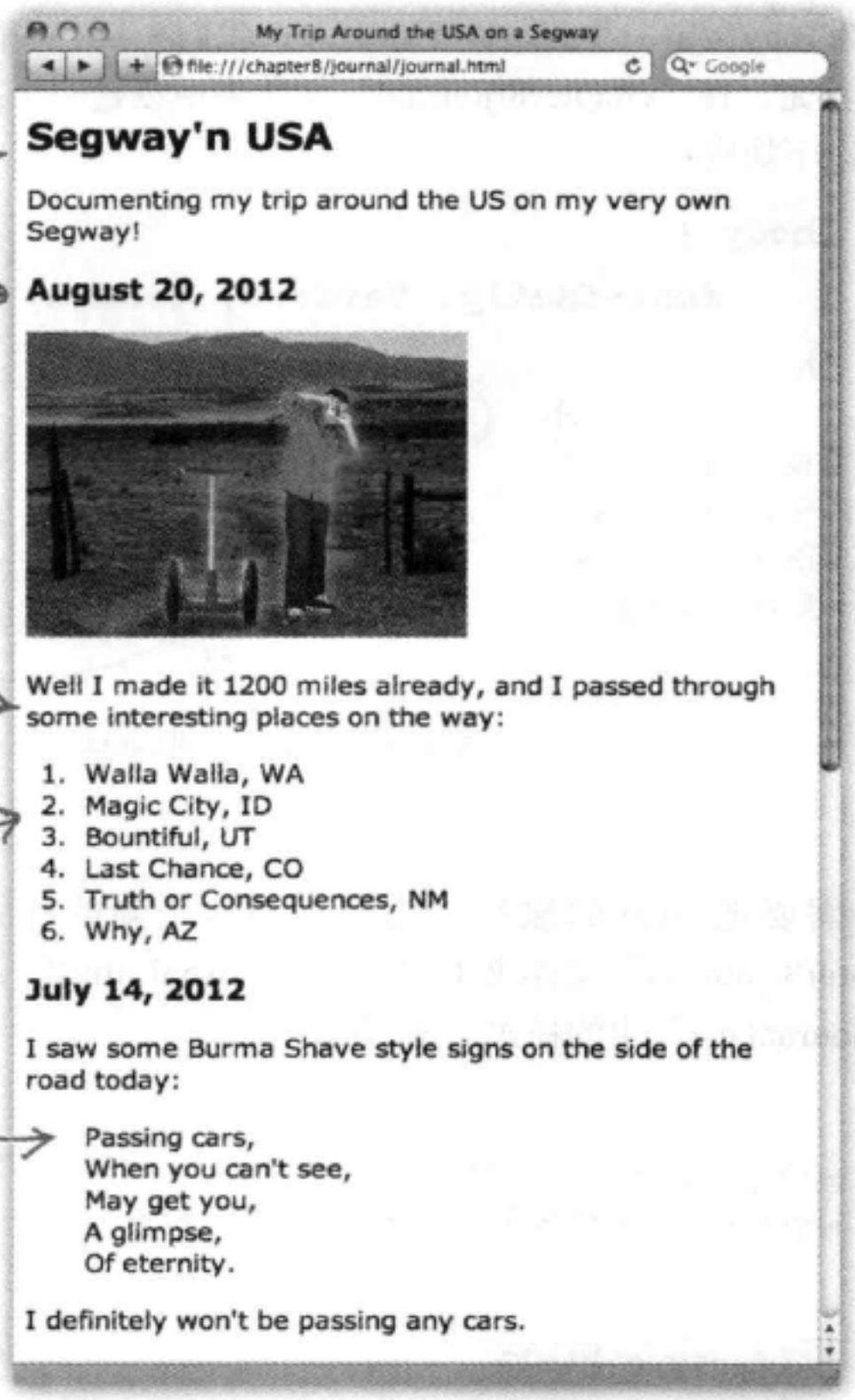
这些字体确实让Tony的Web页面焕然一新。没有字母上下的衬线（serif），标题现在更为清晰，不过在这个页面上看着还是有点太大。

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……

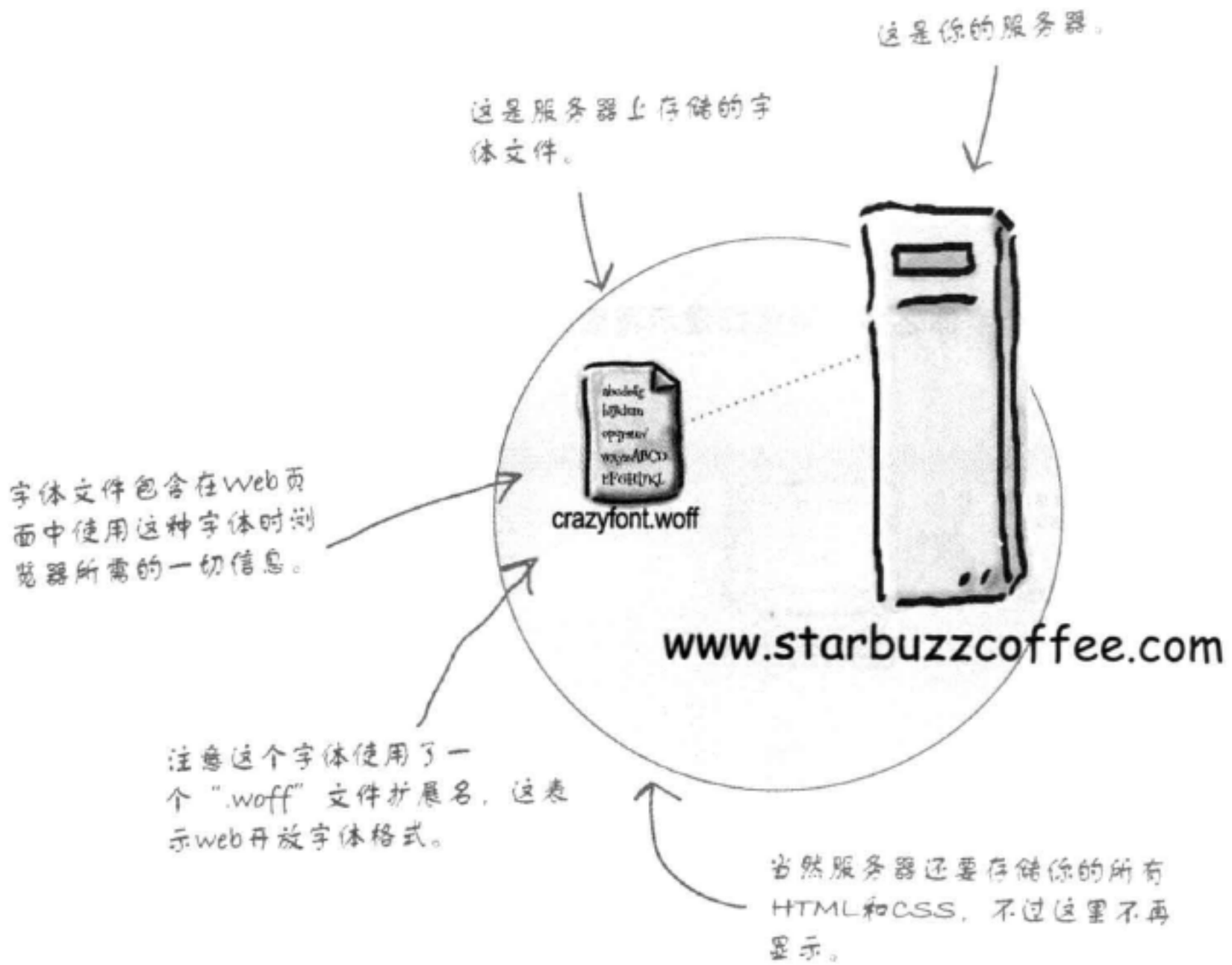
你的建议是可以的，不过很可能只适合一小部分用户。如果你确实必须使用这种“棒极了”的字体，或者排版对你的网站设计很重要，那么你可以使用Web字体（Web Font）向用户的浏览器提供一种字体。

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。

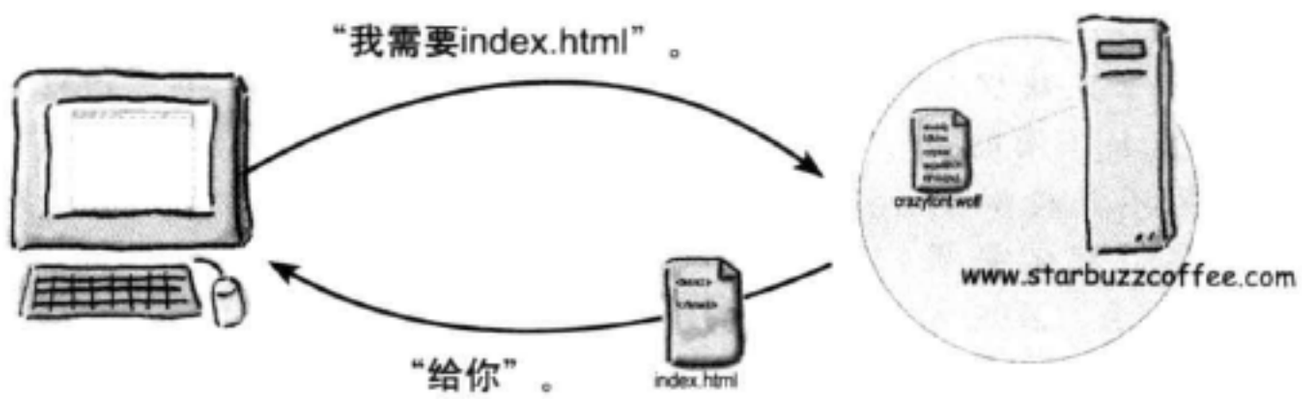
下面来看这是怎么做到的……

Web字体如何工作

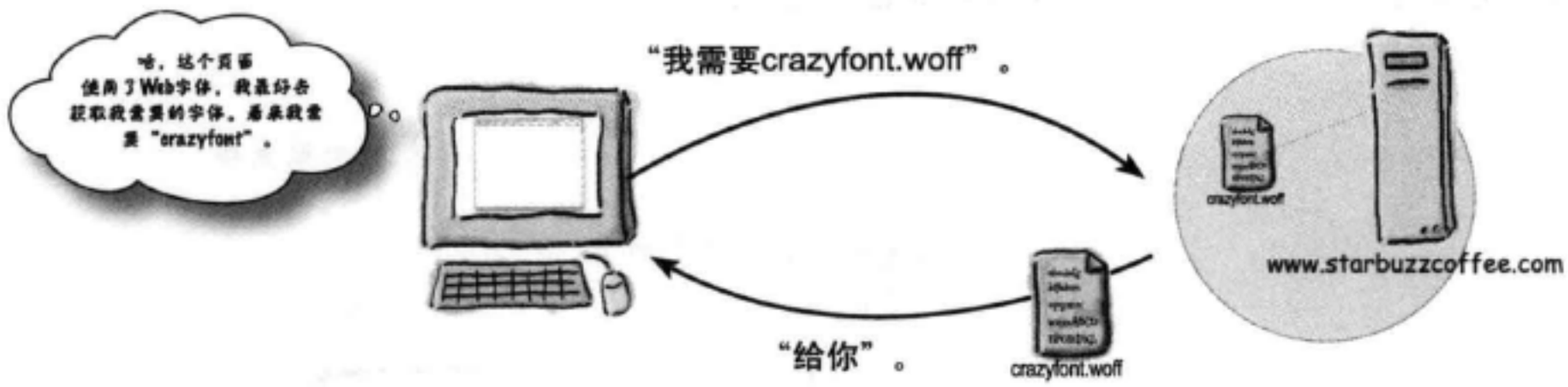
有了Web字体，你可以充分利用现代浏览器的一个新功能，能够向用户直接提供新的字体。一旦提供了新字体，浏览器就能使用Web字体，就像使用所有其他字体一样，甚至可以用CSS指定文本样式。下面详细分析Web字体如何工作：



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。

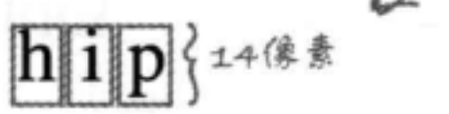
px

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！

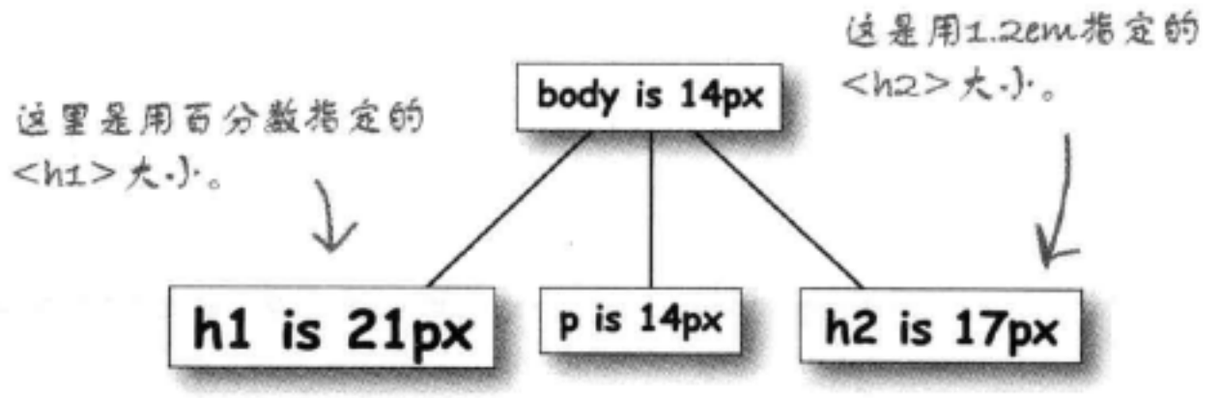
```
font-size: 1.2em;
```

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。

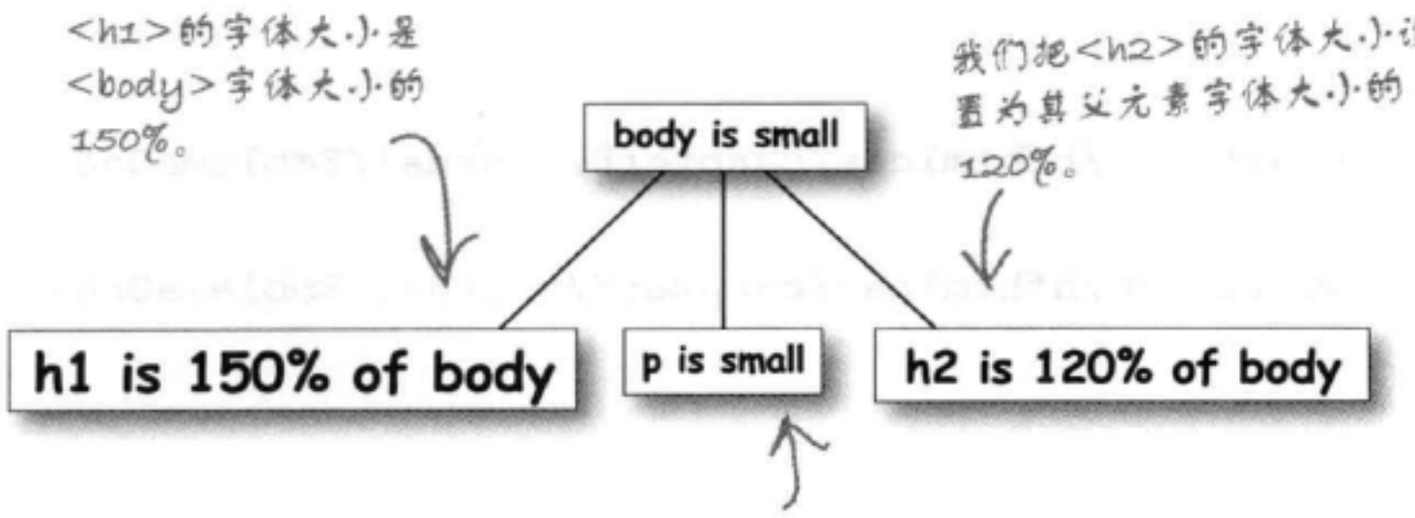
1. **选择一个关键字（我们推荐small或medium），指定它作为body规则中的字体大小。这相当于页面的默认字体大小。**
2. **使用em或百分数，相对于body字体大小指定其他元素的字体大小（选择em还是百分数由你决定，因为实际上这两种方法作用是一样的）。**

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。

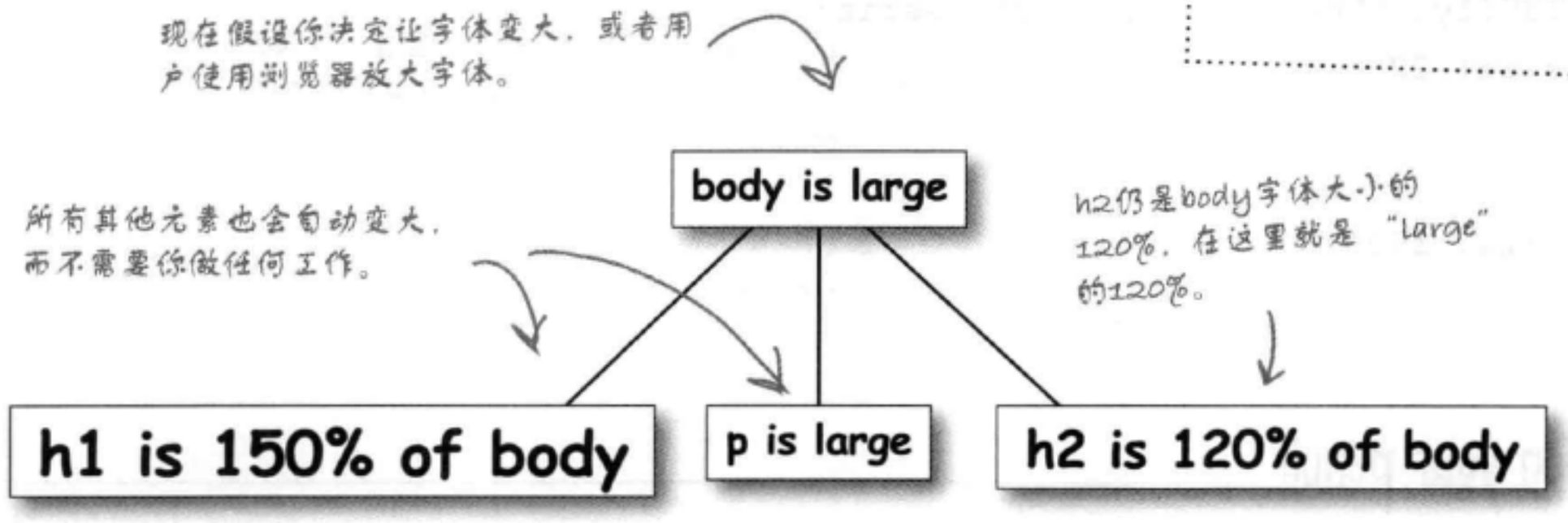
下面来看这是如何工作的。首先，为你的<body>元素设置一个大小。然后，相对于这个大小设置所有其他字体大小，就像这样：

```
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？

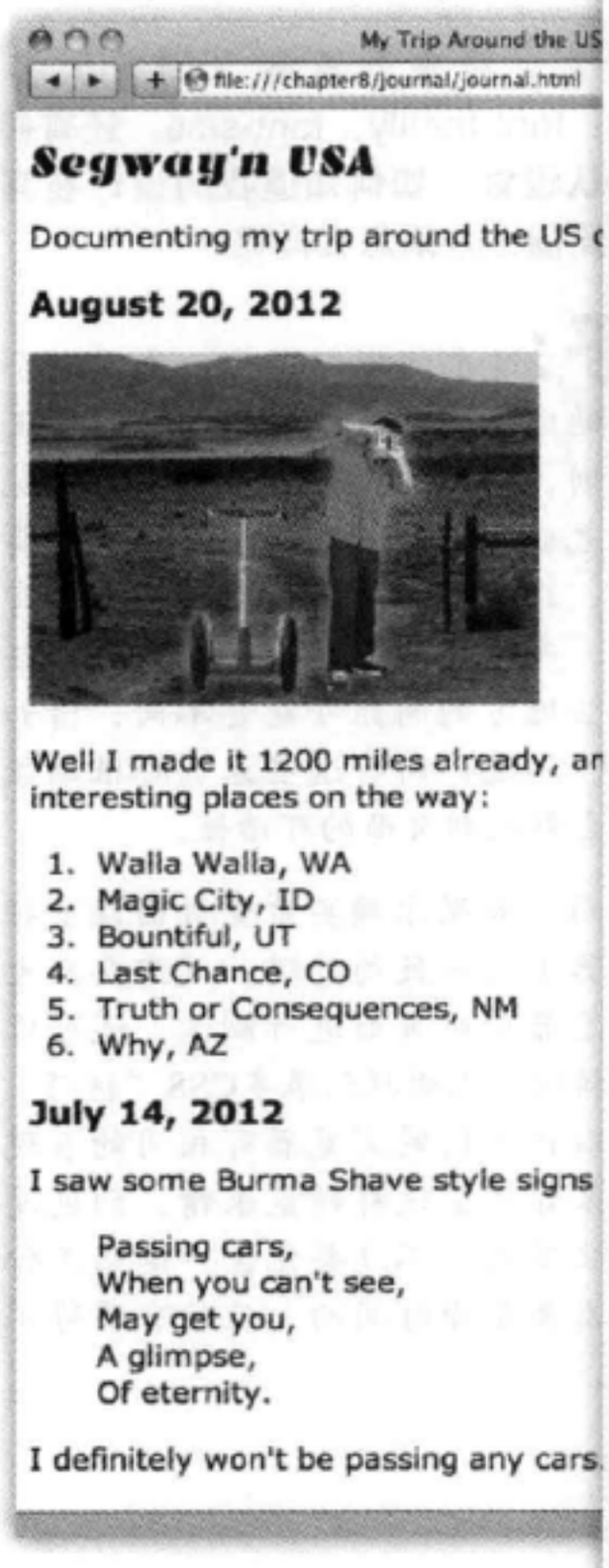
答案：<h1>将是2.2em，<h2>为1.3em。

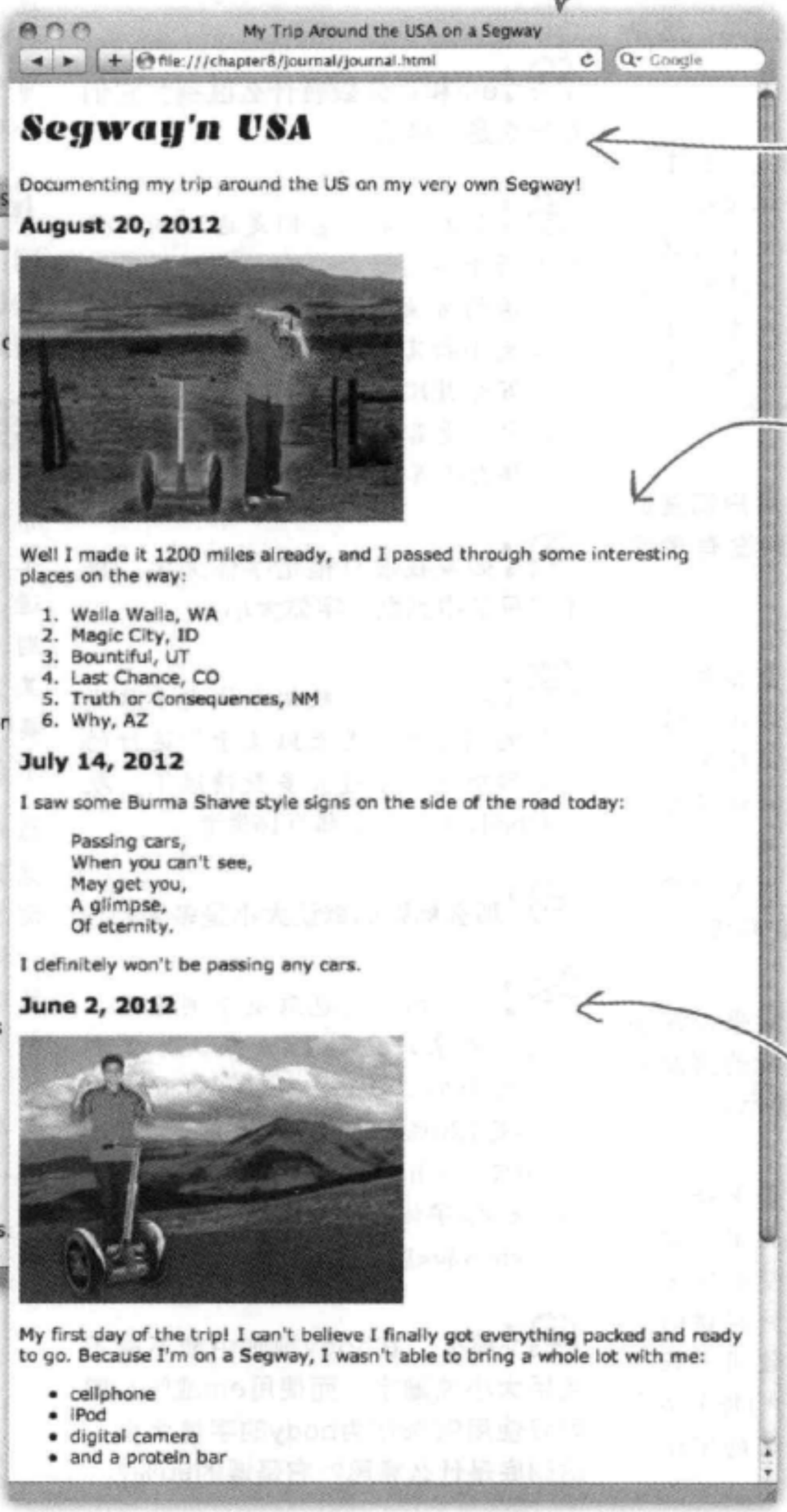
测试字体大小

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：

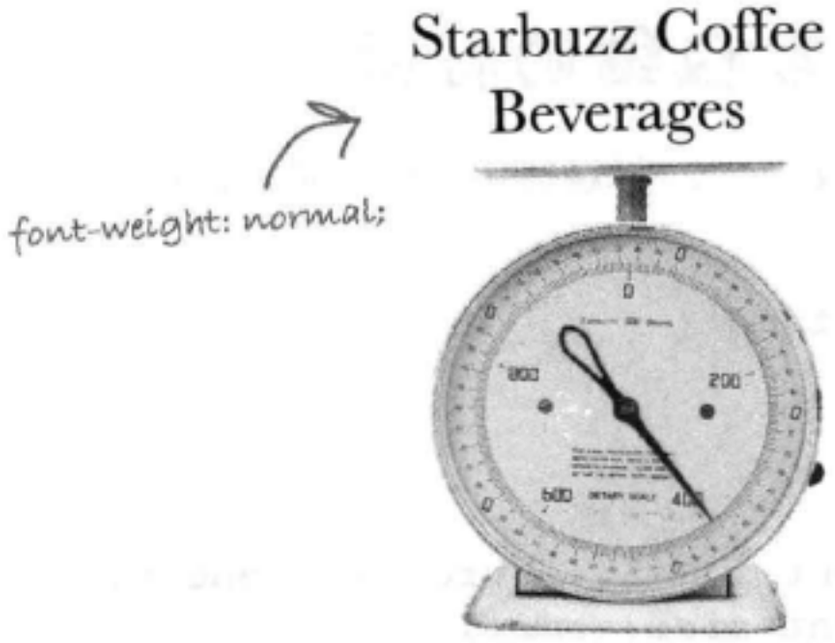
```
font-weight: bold;
```

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。

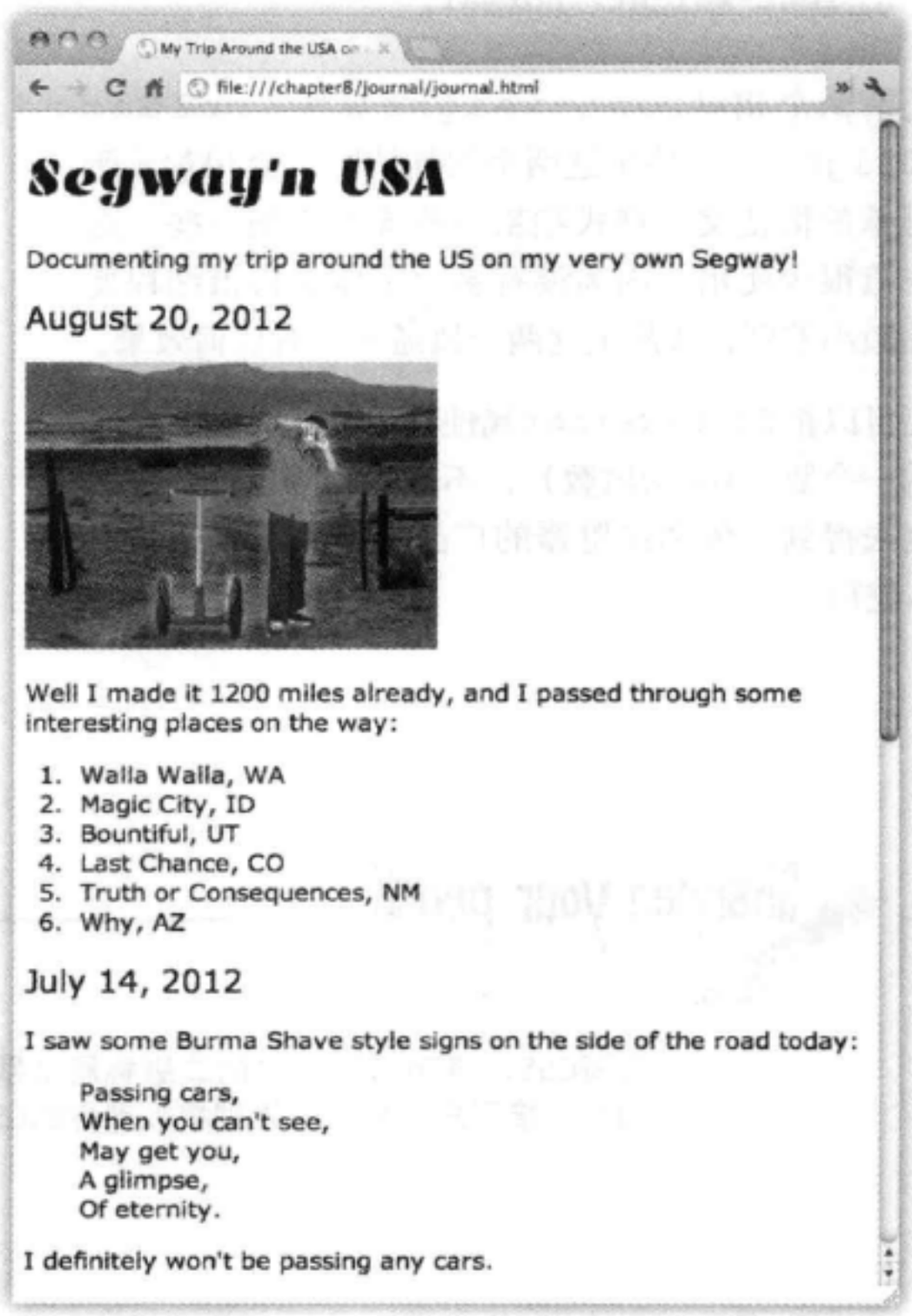
测试正常粗细的标题

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。

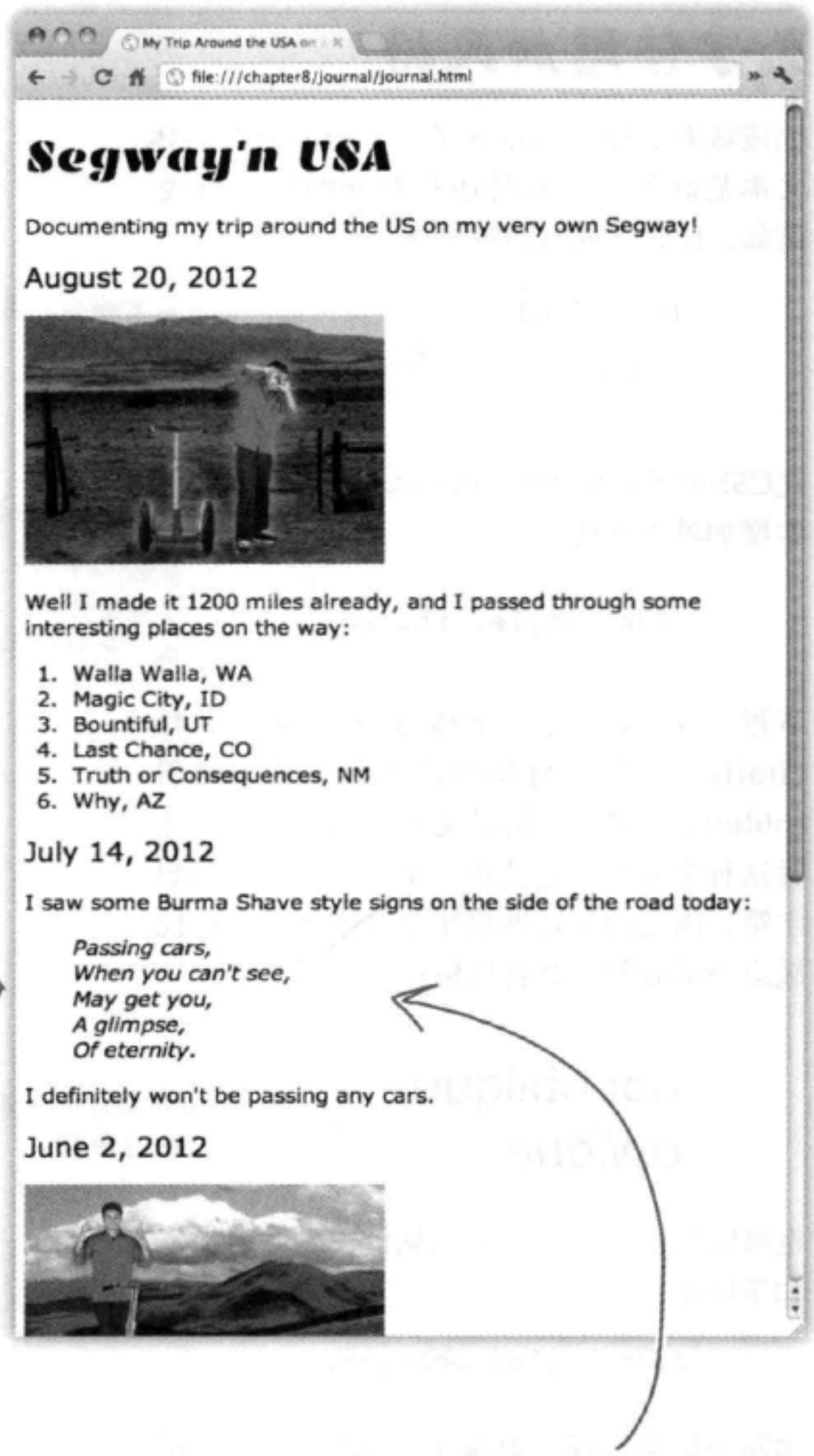
所以完全可以就用斜体，不用担心它们的差别（你可能根本无从控制）。

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。

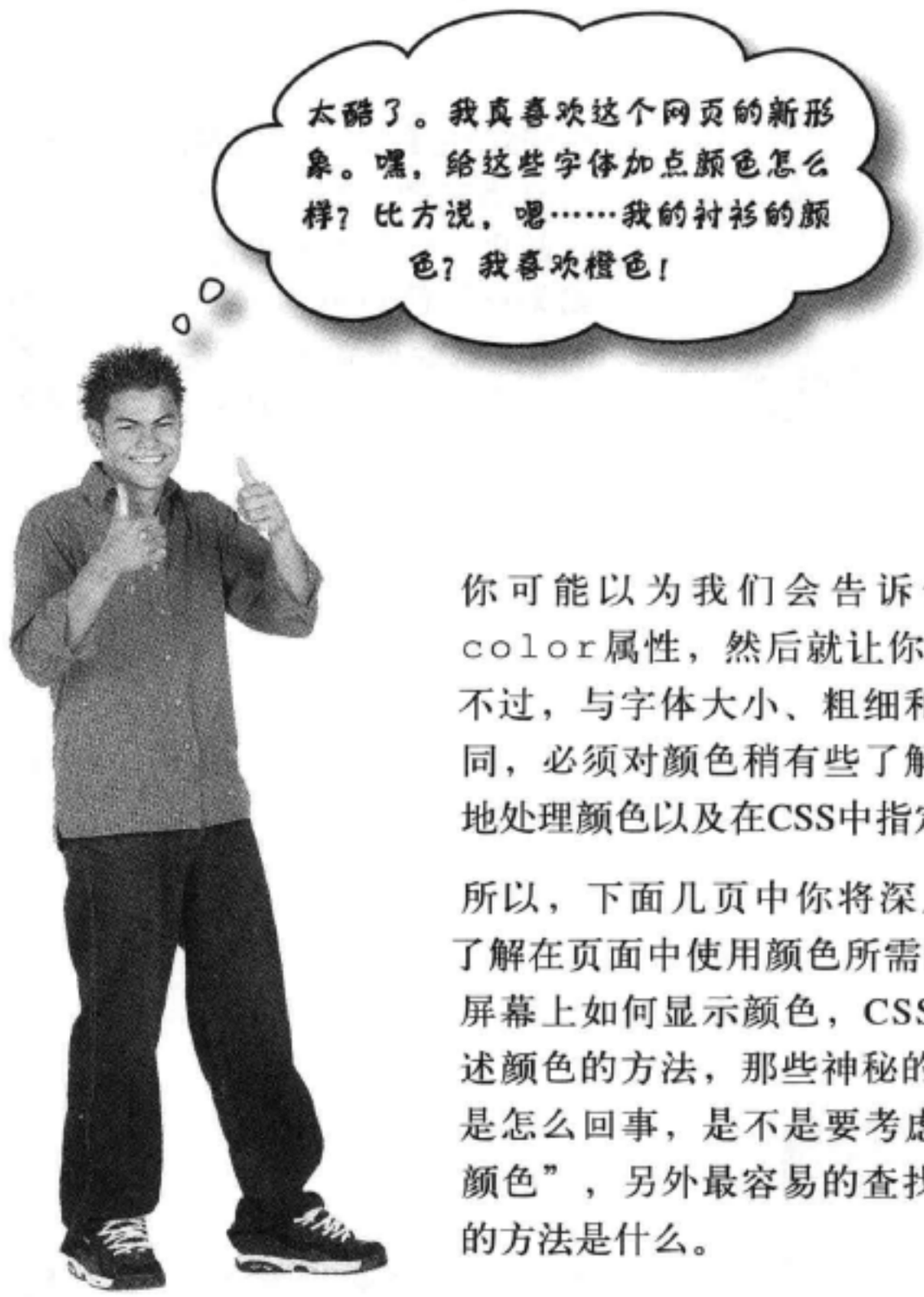
there are no Dumb Questions

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。

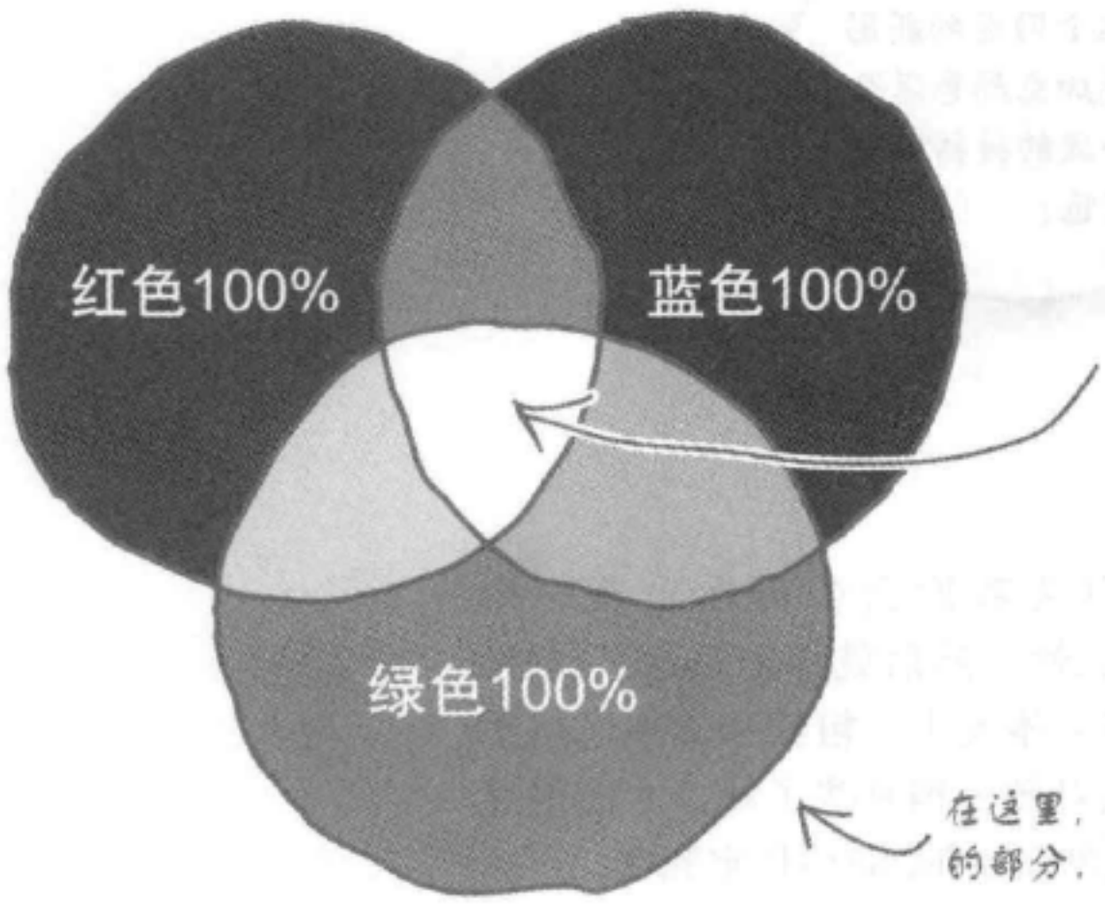


你可能以为我们会告诉你：有一个color属性，然后就让你用这个属性。不过，与字体大小、粗细和文本风格不同，必须对颜色稍有些了解，才能很好地处理颜色以及在CSS中指定颜色。

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。

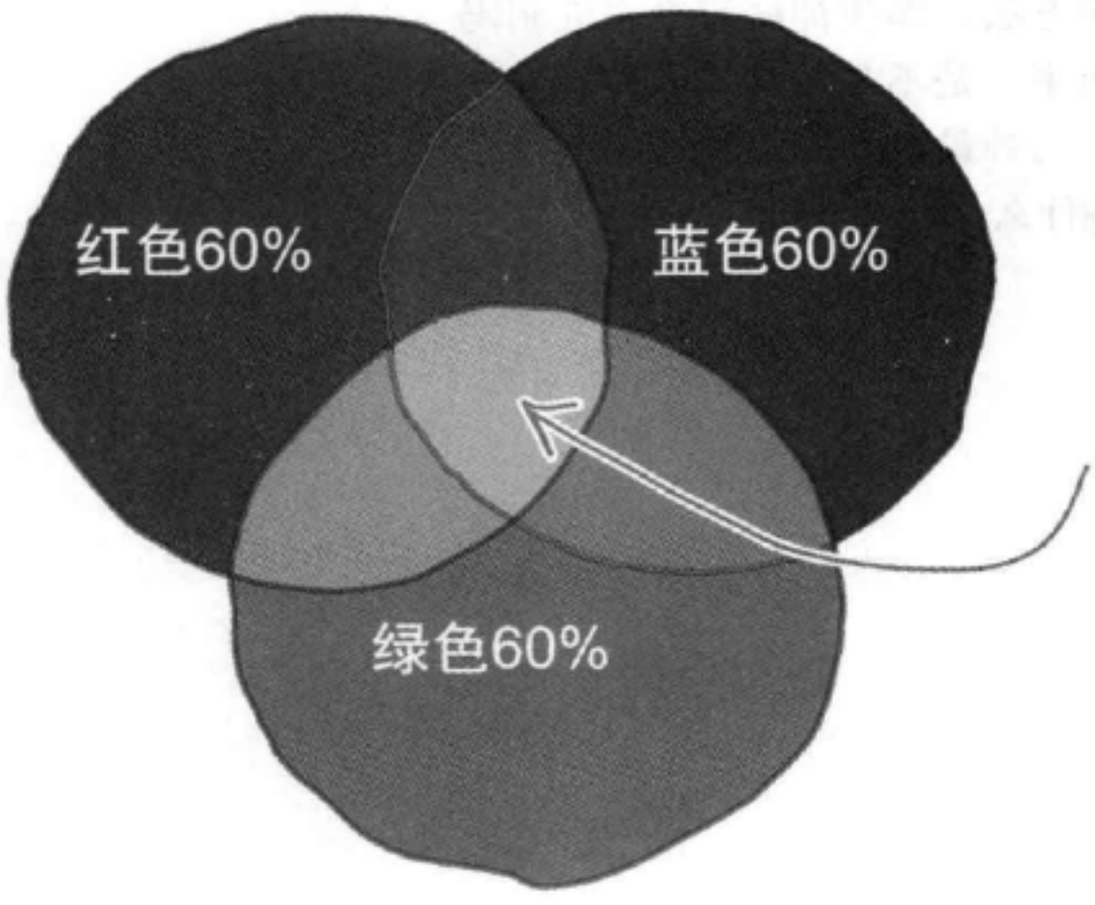
Web颜色如何工作？

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。



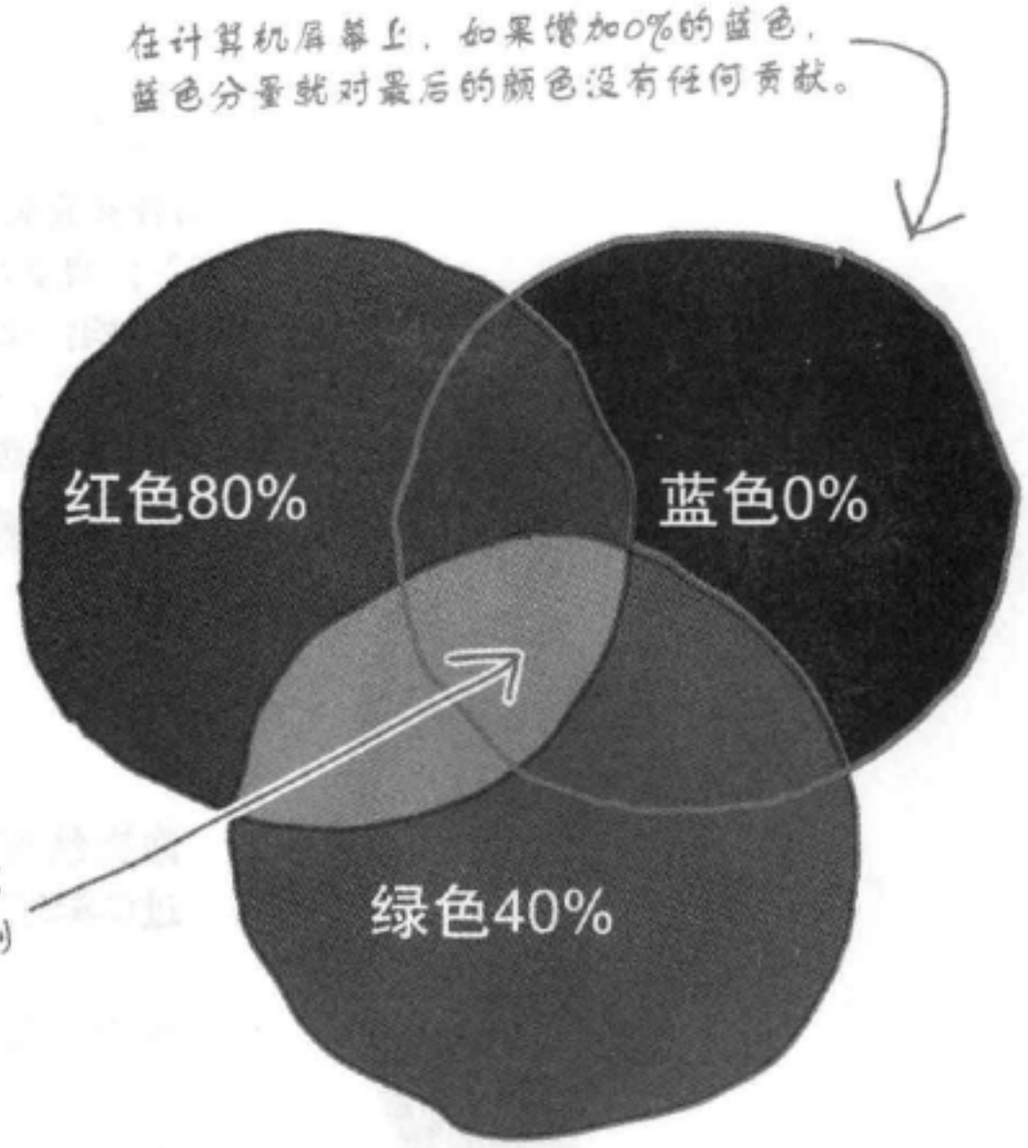
在这里，红色、绿色和蓝色混合在一起。请看中间的部分，可以看到它们是如何叠加的。

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！

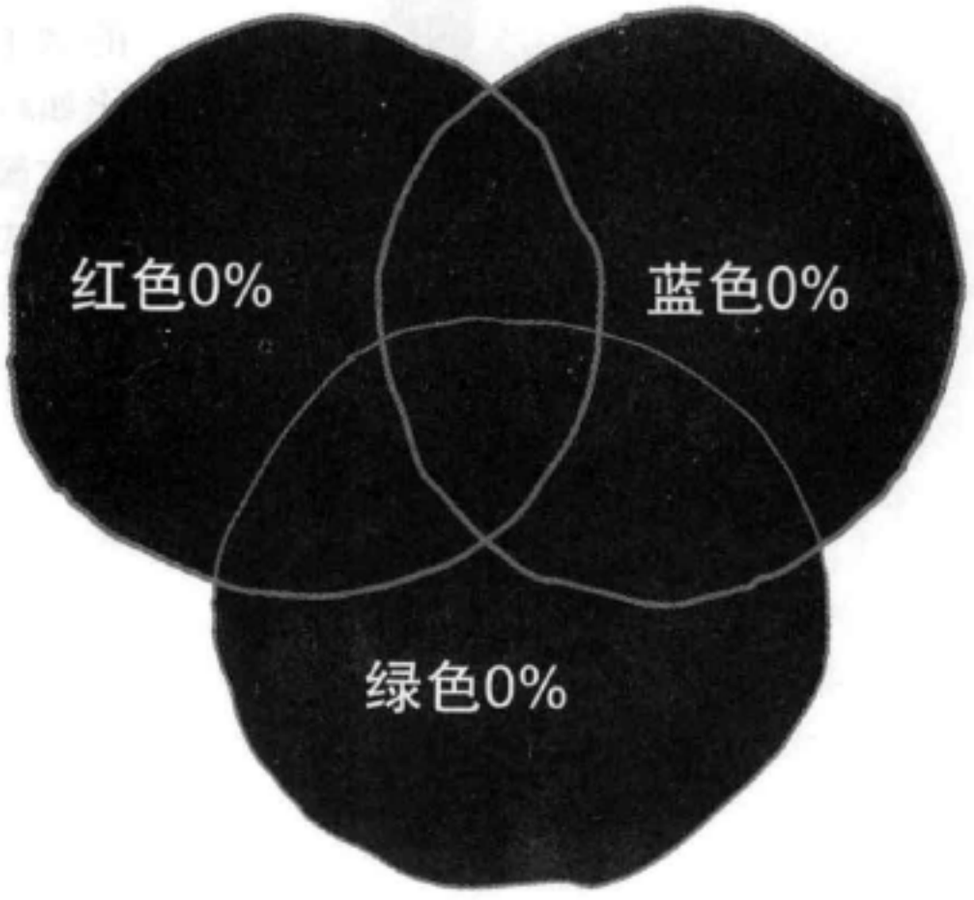


不过，如果每个颜色分量（红、绿和蓝）不是100%，比如只是60%，会得到什么颜色呢？不太白，是吗？换句话说，你会得到一个灰色，因为我们仍然是将这三种颜色等量相加，不过没有那么多光发送到屏幕上。

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;
}
```

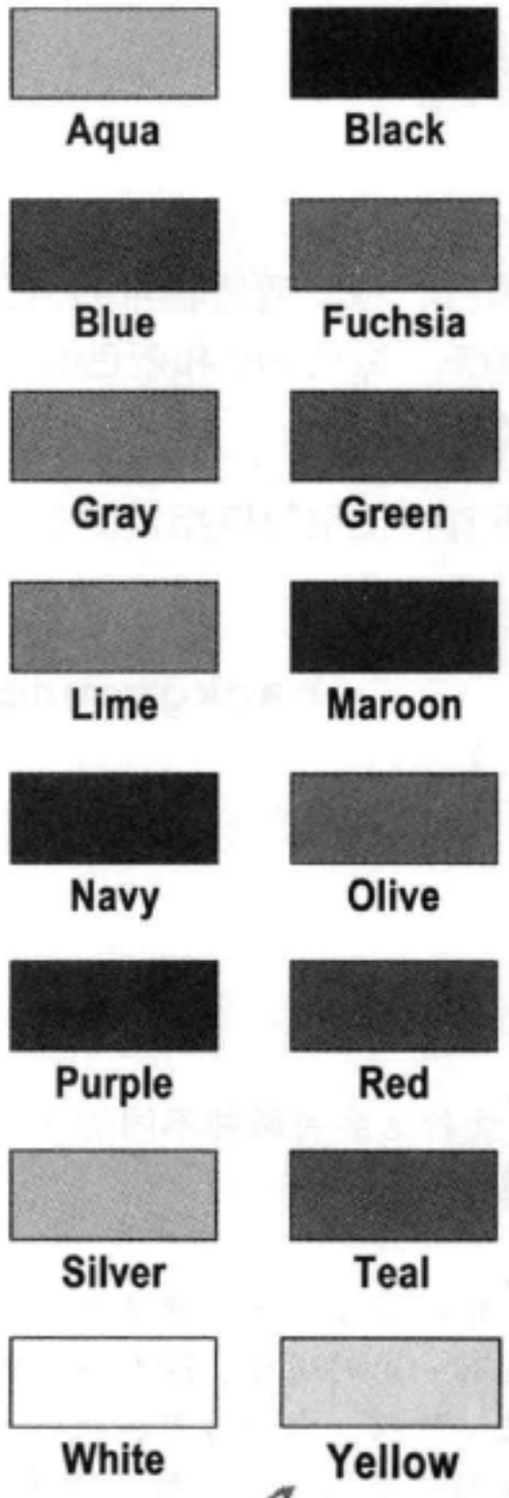
这里是body规则。

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。

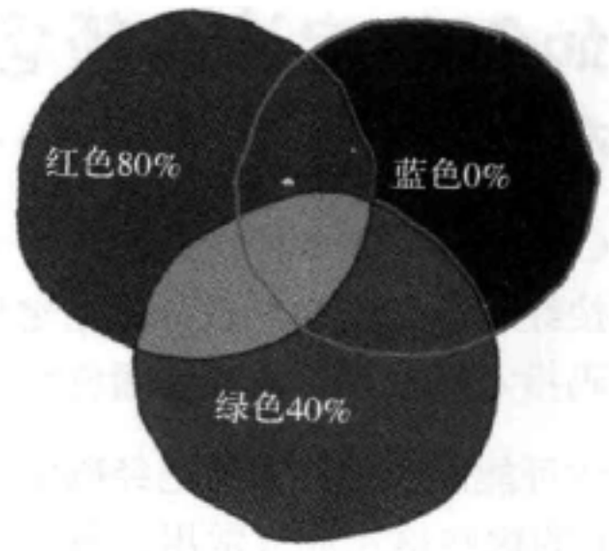
所有浏览器中都肯定有这16种颜色，不过可能只在较新的浏览器中能找到150种扩展颜色。



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。

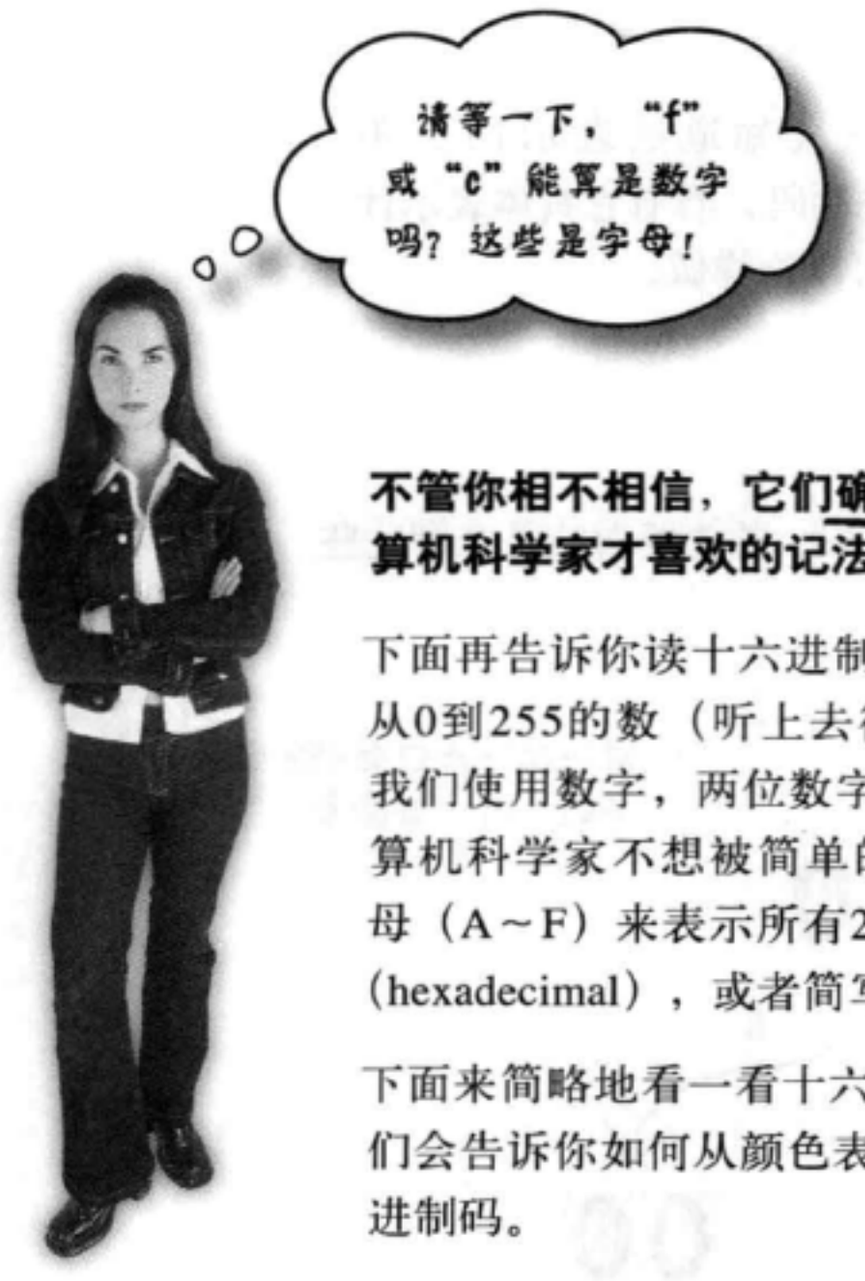
答：使用rgb百分数或数值越来越常见了，不过你说的没错，“十六进制码”仍然使用最广泛，因为人们认为这是一种很方便的指定颜色的方法。

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：



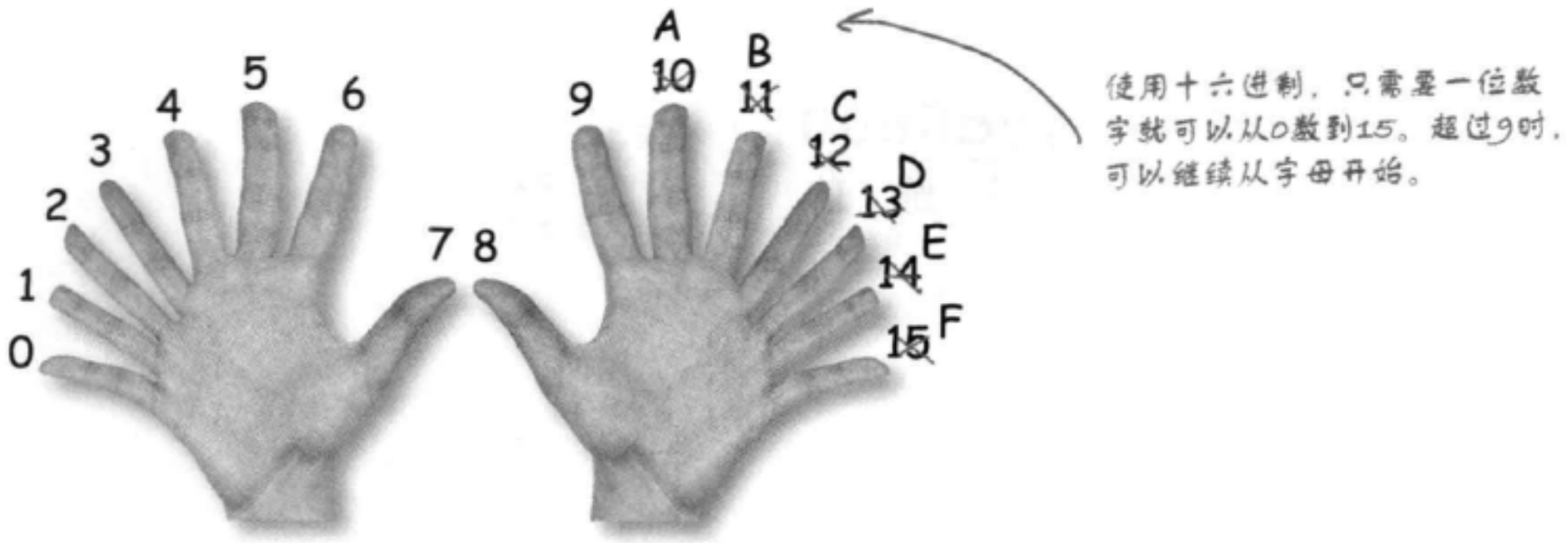
不管你相不相信，它们确实是数字，不过这里采用了只有计算机科学家才喜欢的记法。

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：

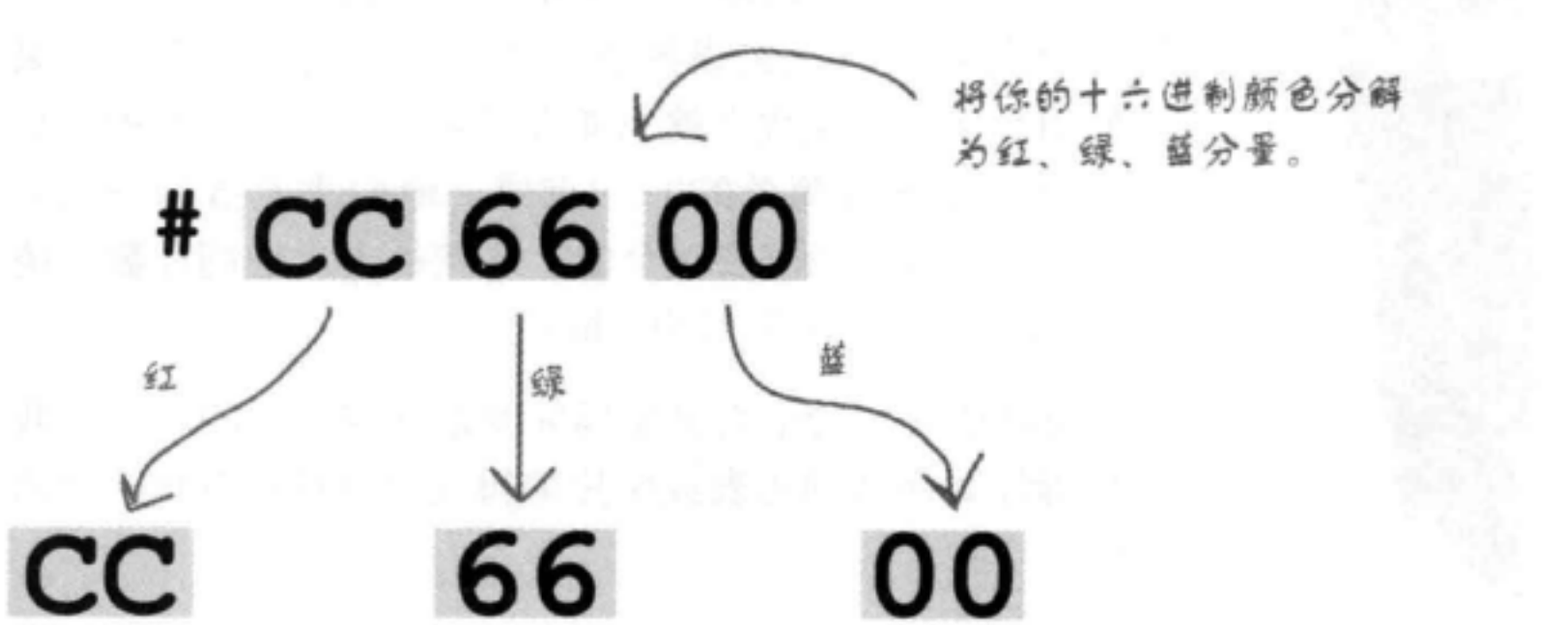


所以，如果你看到一个十六进制数，比如B，就会知道这表示11。不过，BB、E1或FF表示什么呢？下面来分解一个十六进制码，看看它具体表示什么。实际上，对于你可能遇到的十六进制颜色，你都可以这样做。

第1步：

将十六进制颜色分解为它的3个分量。

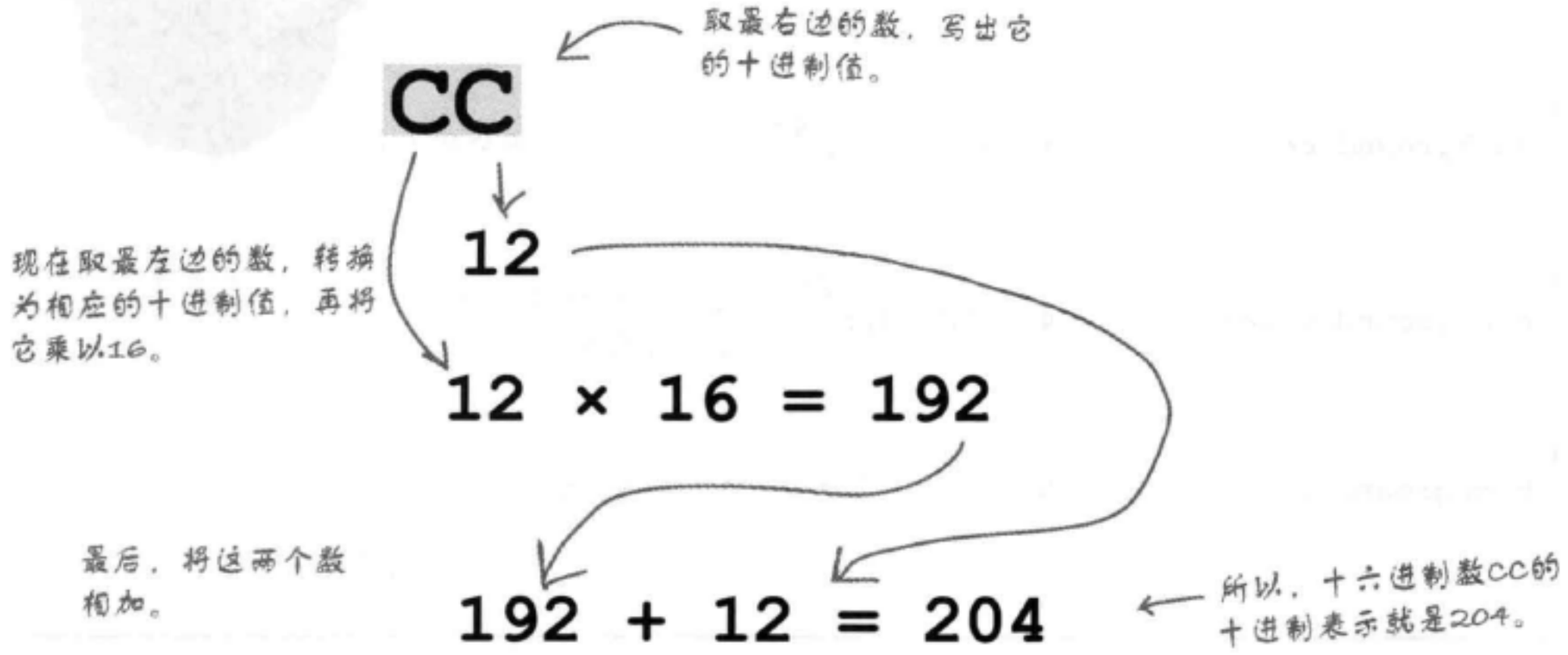
要记住，每个十六进制颜色由红、绿、蓝分量组成。首先要做的是分解这些分量。



第2步：

将每个十六进制数转换为相应的十进制数。

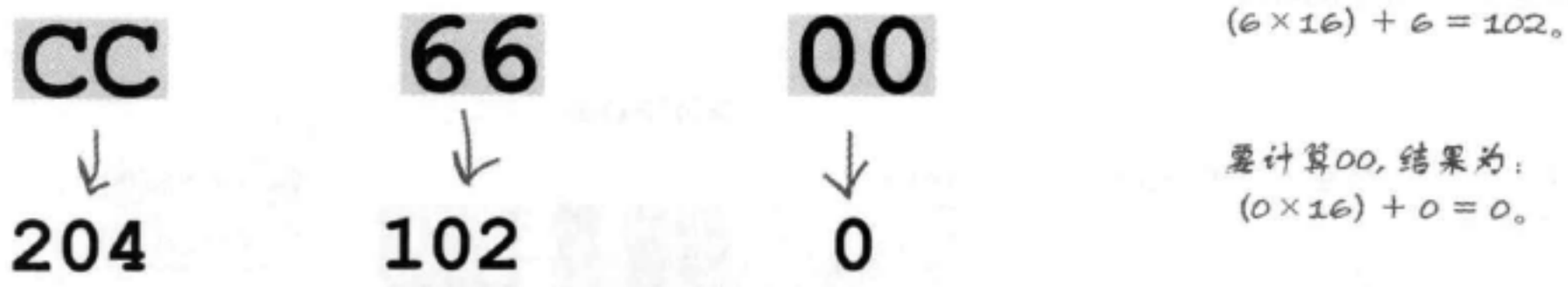
既然已经分解得到了分量，下面可以计算各个分量的值，得到0到255的一个数。先从红色分量的十六进制数开始：



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：



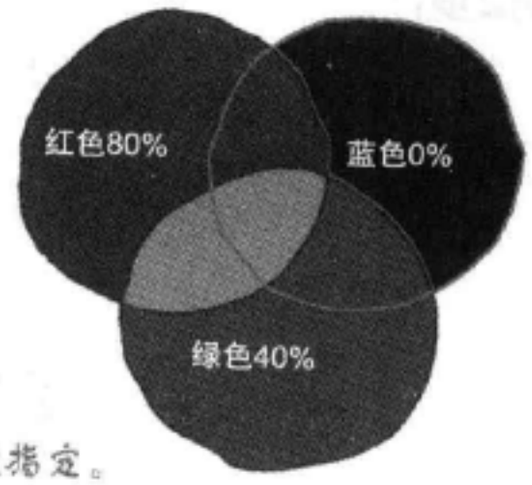
第4步：

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。

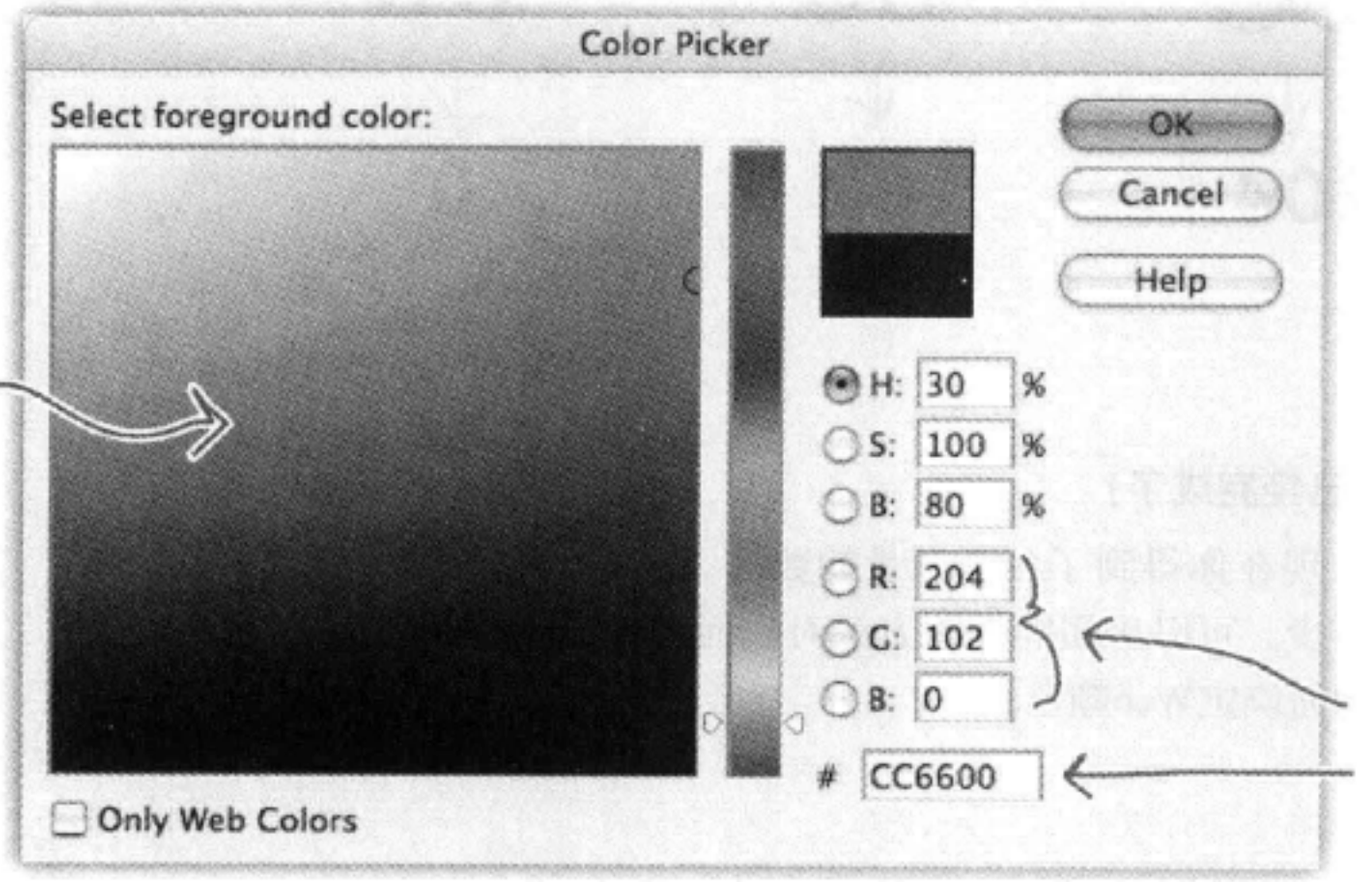
```
body {
      background-color: #cc6600;
}
```

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。

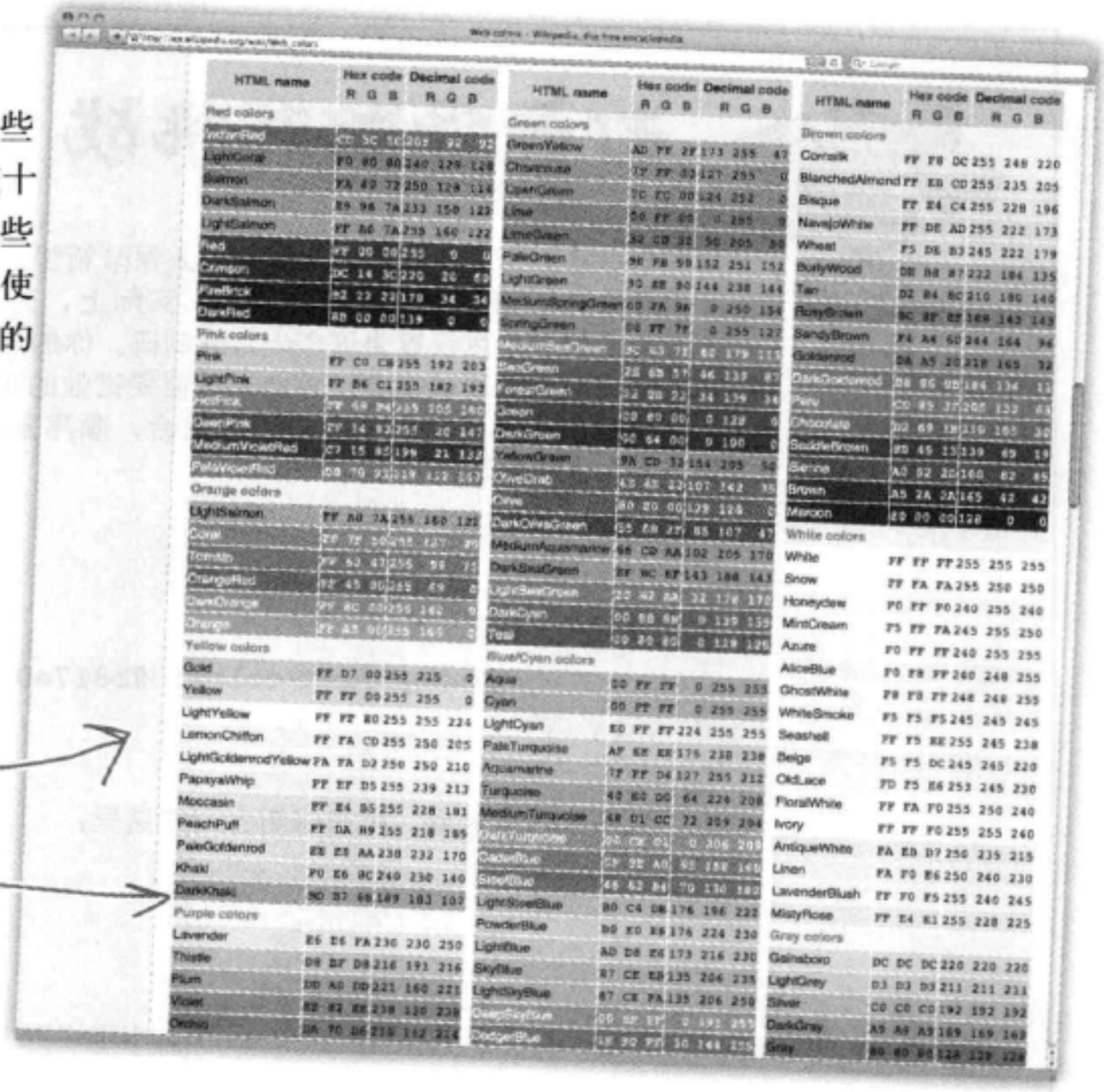
一旦选择了一个颜色，颜色选择器会为你显示这个颜色的rgb值和十六进制码。

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。

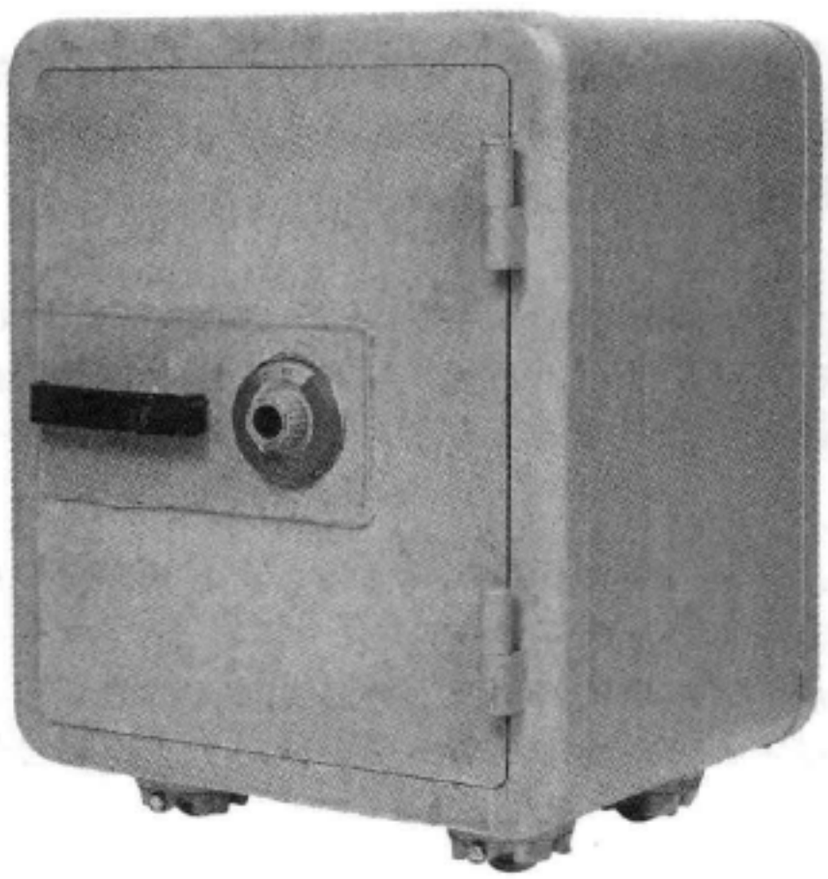
打开保险箱挑战

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%)。

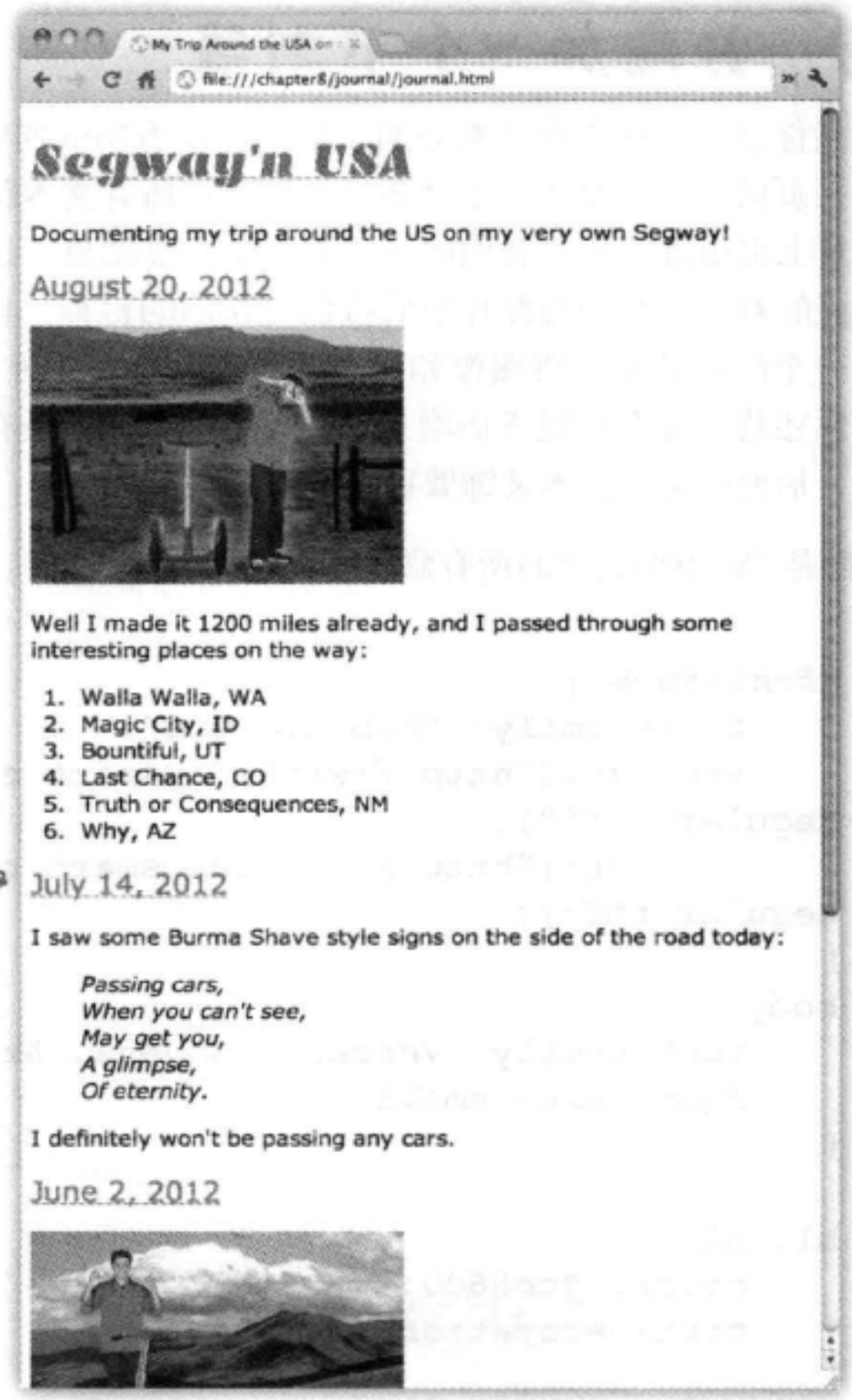
可以这样创建一个下划线。我们使用了text-decoration属性，将它设置为underline。

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。

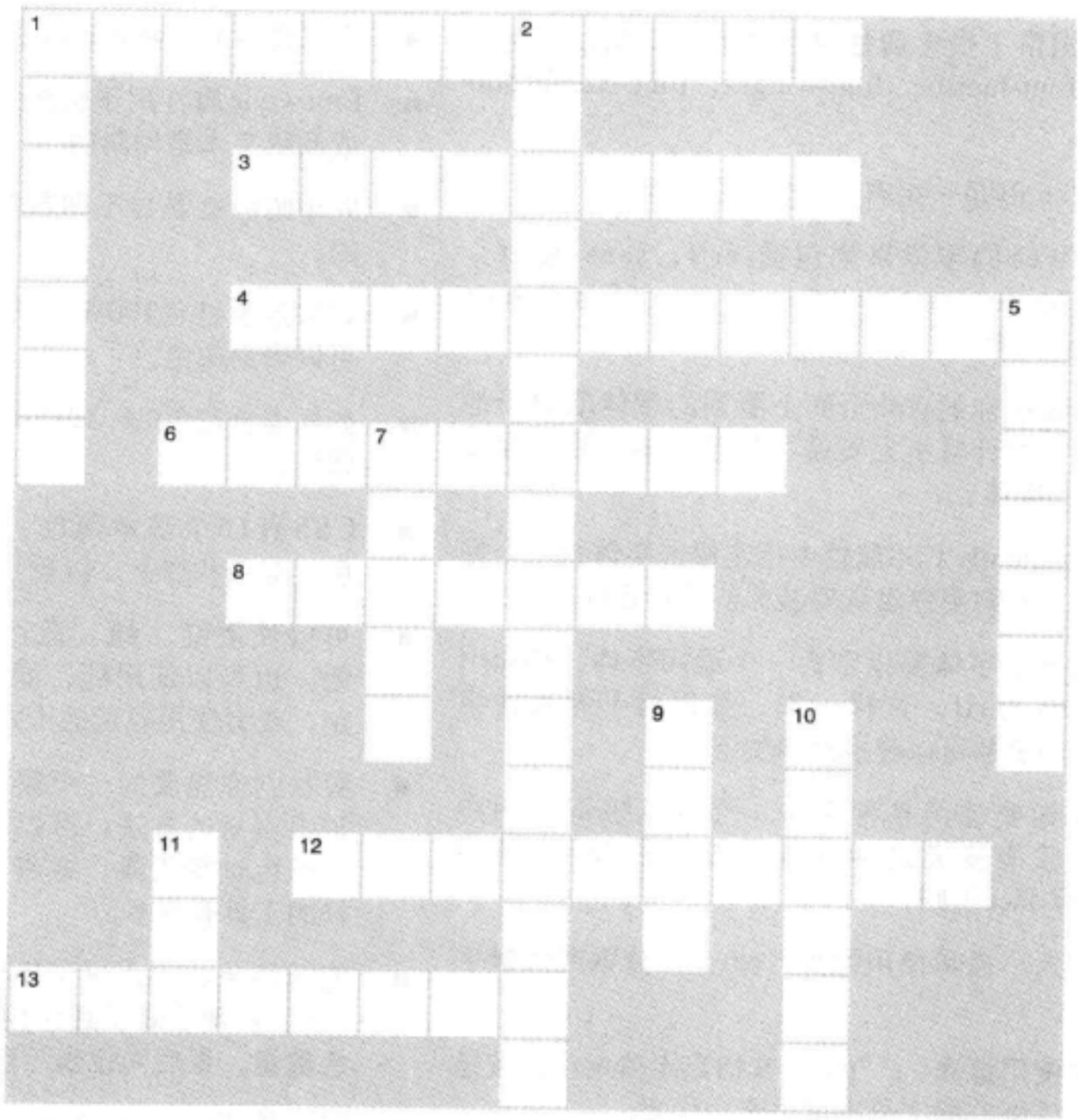
注意这些下边框一直延伸到页面边缘，而不只是在文本下面出现。为什么？下一章会告诉你答案。

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

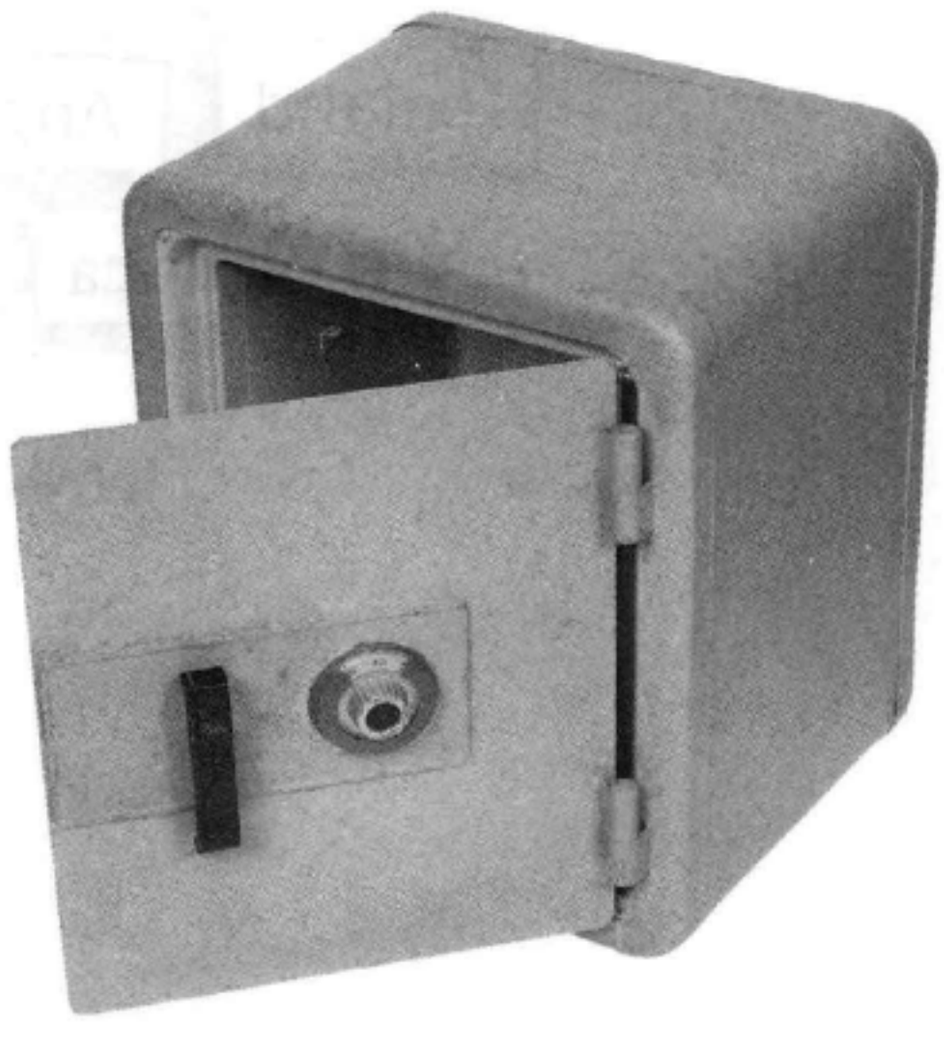
打开保险箱挑战答案

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							

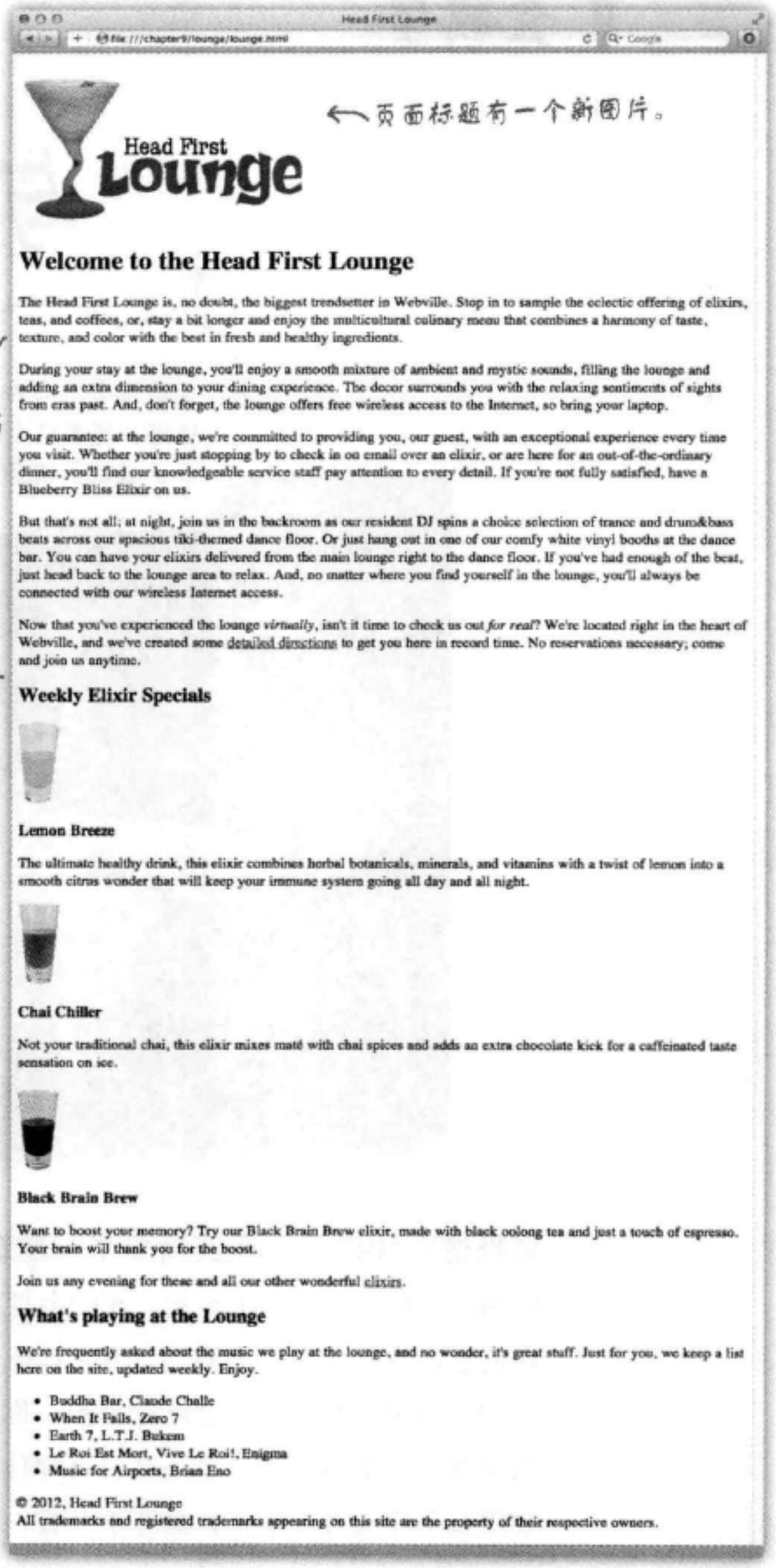
9 盒模型

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）。

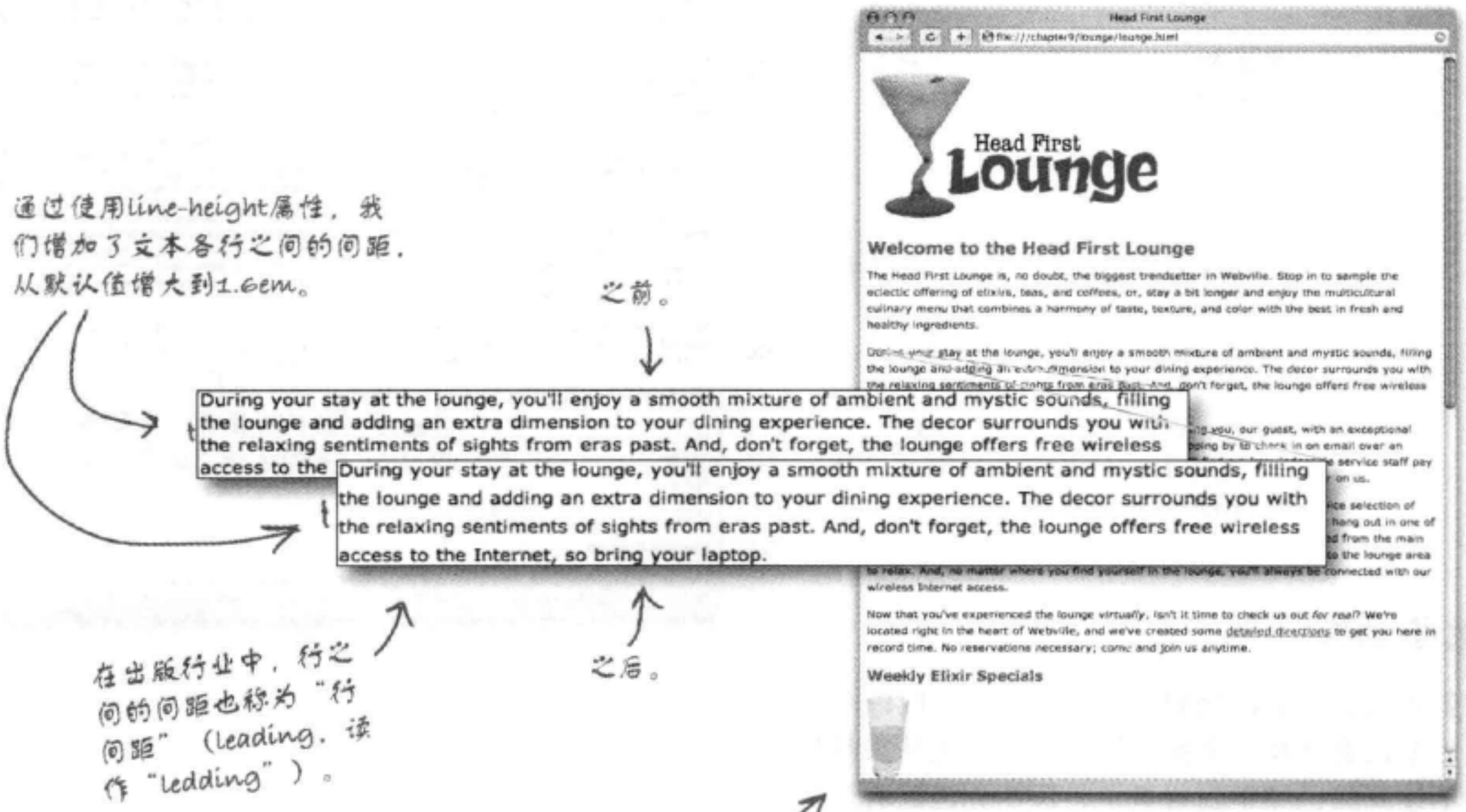
```
body {
        font-size:   small;
        font-family: Verdana, Helvetica, Arial, sans-serif;
        line-height: 1.6em;
}
```

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。



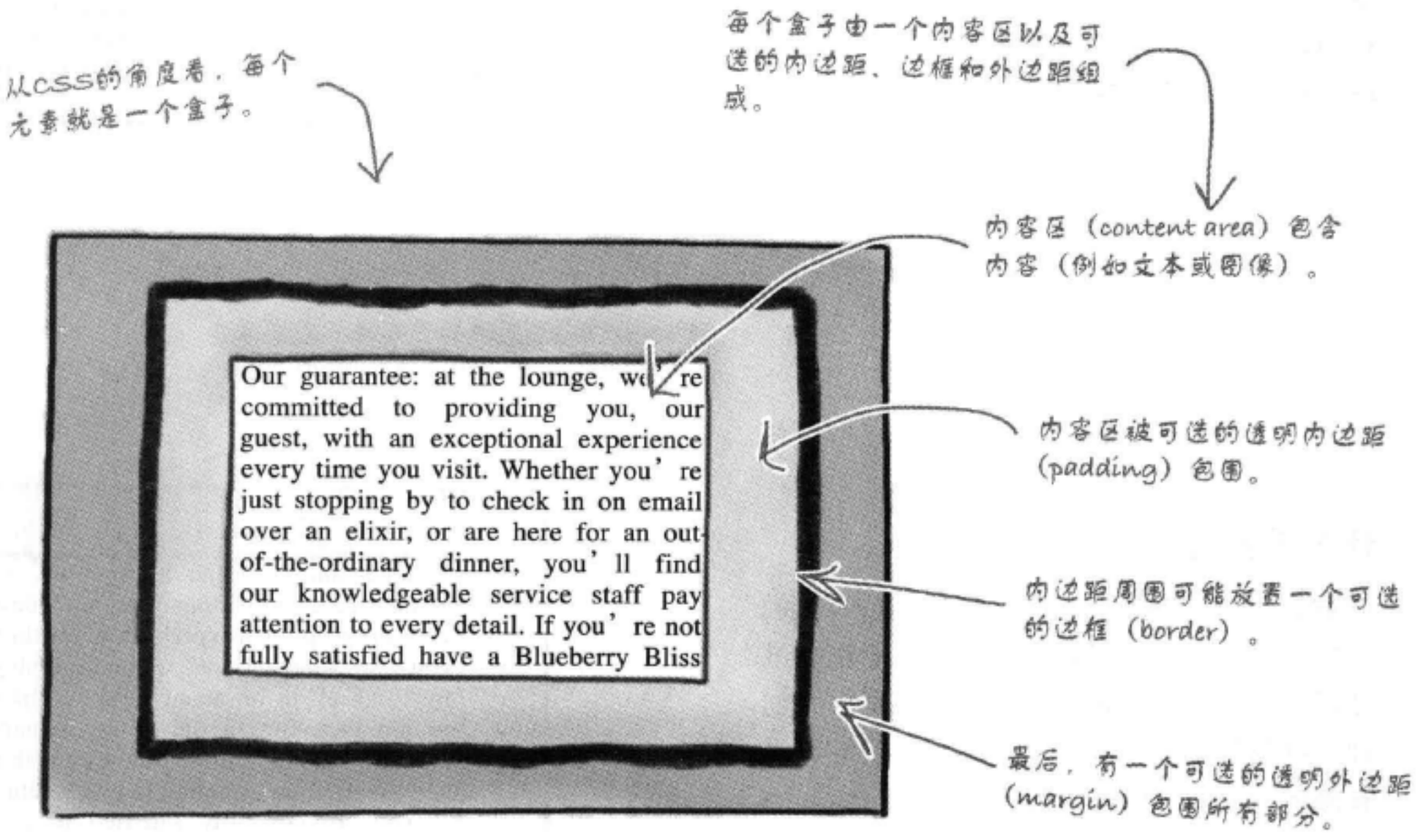
尝试使用几个不同的line-height值，如200%、.5em和20px，看看效果。哪一个最好？哪一个最差？哪一个最可读？完成后，一定要把line-height值再改回1.6em。

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。



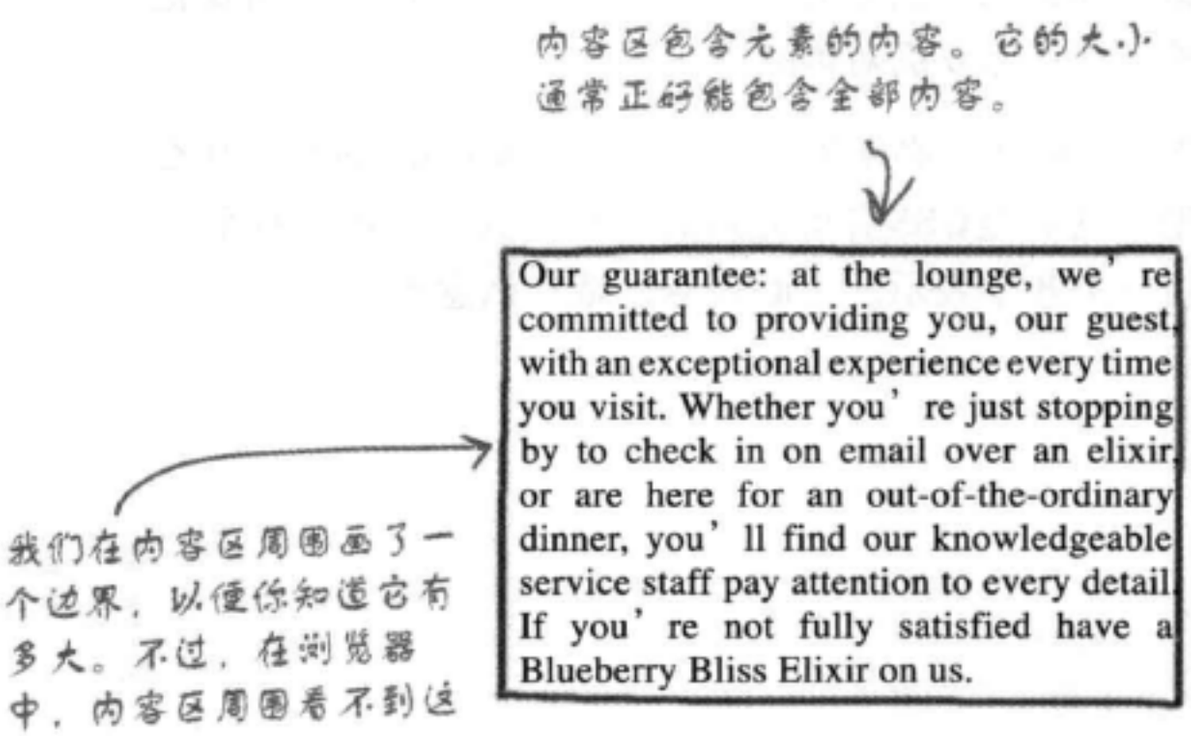
所有元素都被当作盒子：段落、标题、块引用、列表、列表项等。甚至内联元素（如和链接）在CSS看来也是盒子。

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：

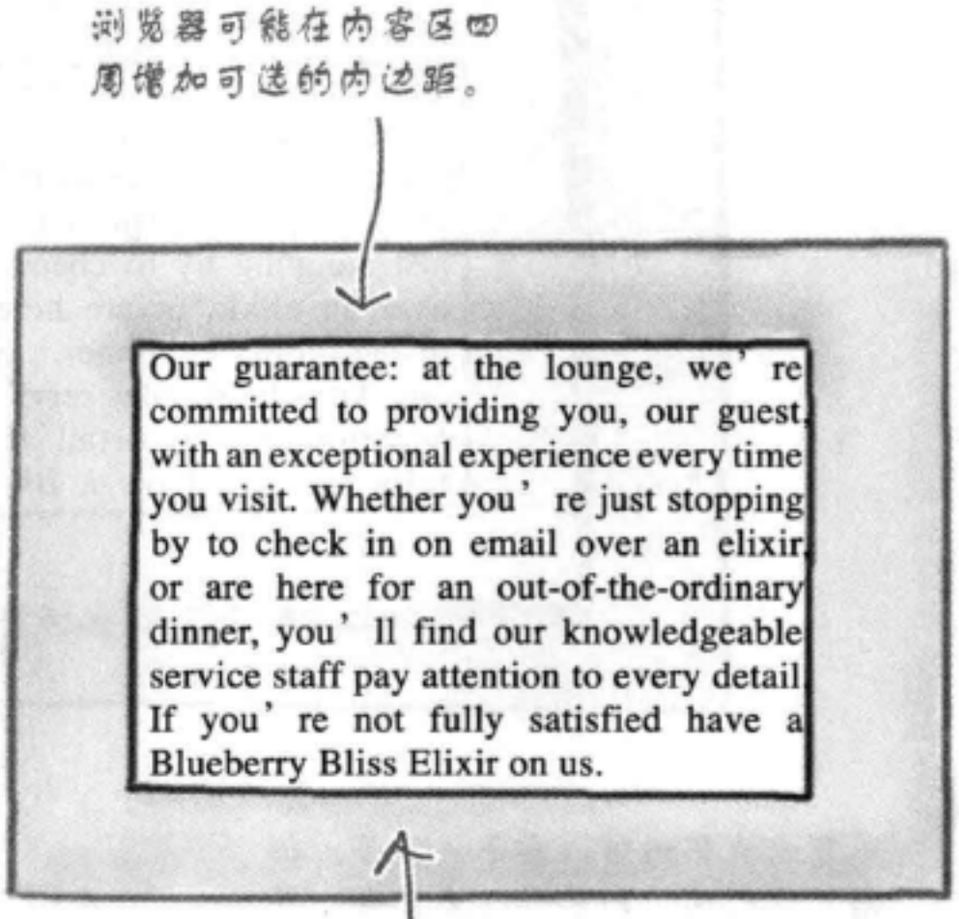
什么是内容区？

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。



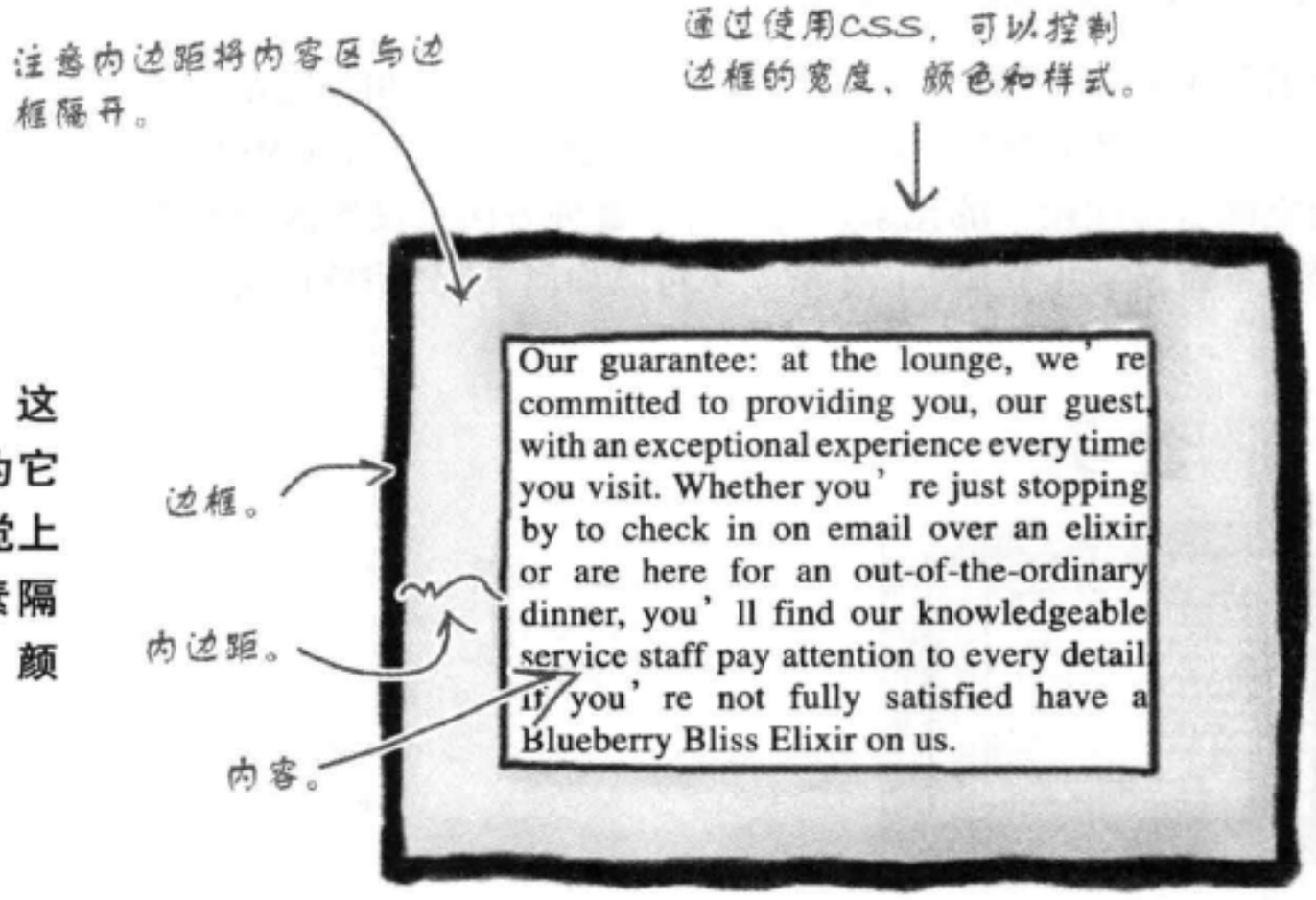
什么是内边距？

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。



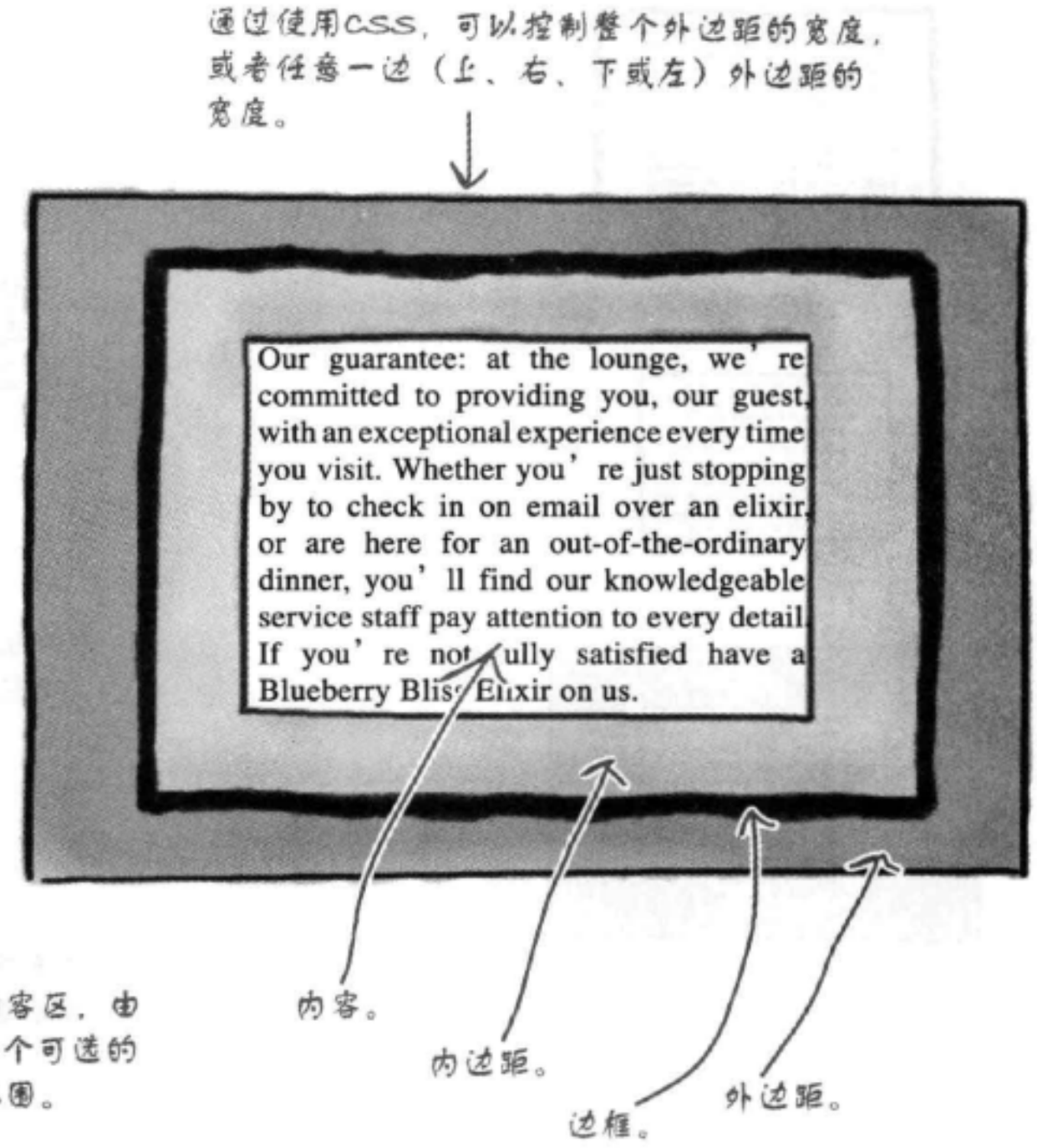
什么是边框?

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。

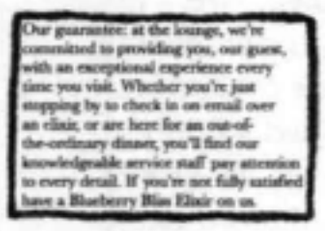


对盒子能做哪些设置

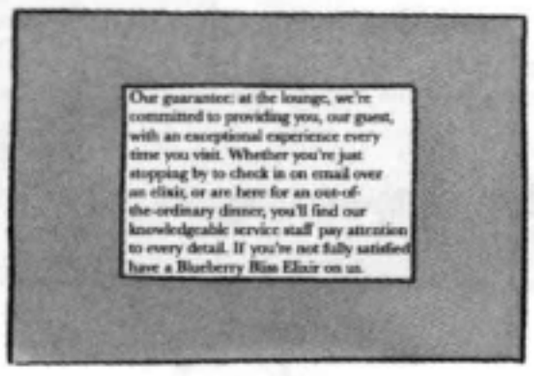
盒模型看起来可能很简单，只有内容、内边距、边框和外边距。不过，如果把它们都结合在一起，对于一个可能有内部空间（内边距）和外围空间（外边距）的元素，就会有无数种方法来设置这个元素的布局。下面来看几个盒子，了解一下可以如何来改变你的元素。

可以指定盒子的样式，包括内边距、边框和外边距。

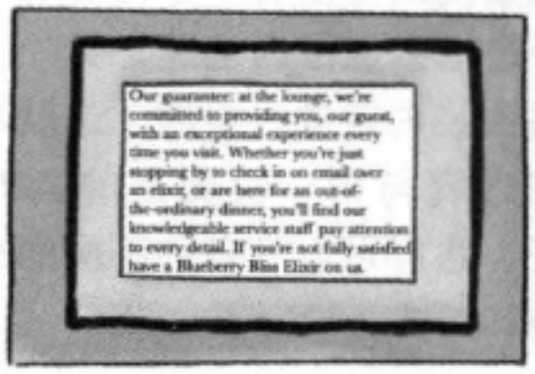
或者只有内边距和一个边框。



或者只有边框。

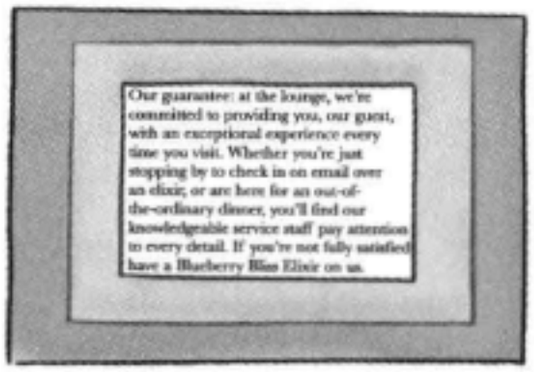


或者有外边距，但没有边框和内边距。

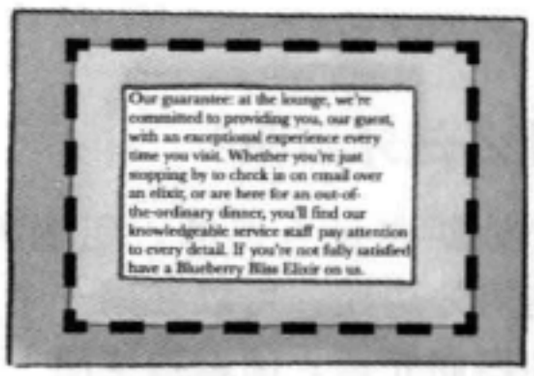


可以有实线边框，边框可以是粗线或细线。

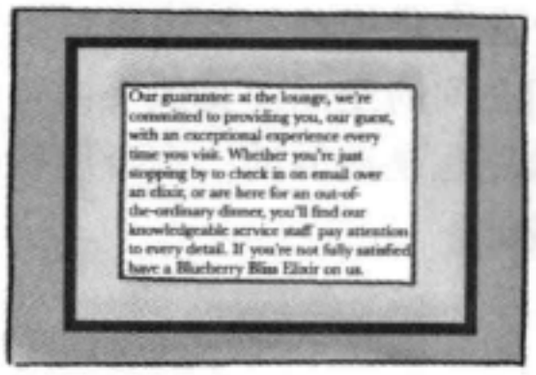
或者根本没有边框。



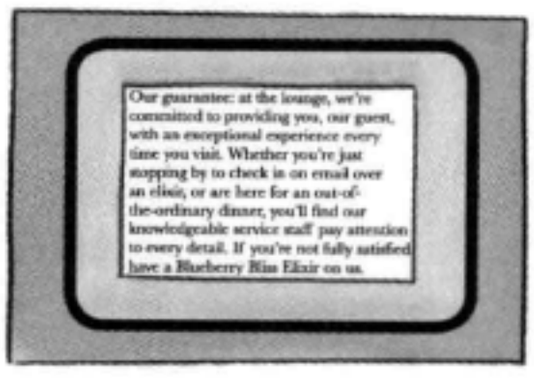
或者可以从8种不同样式的边框中选择，如虚线边框。



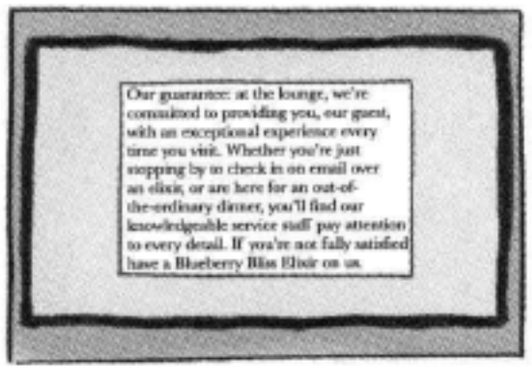
或者对边框指定颜色。



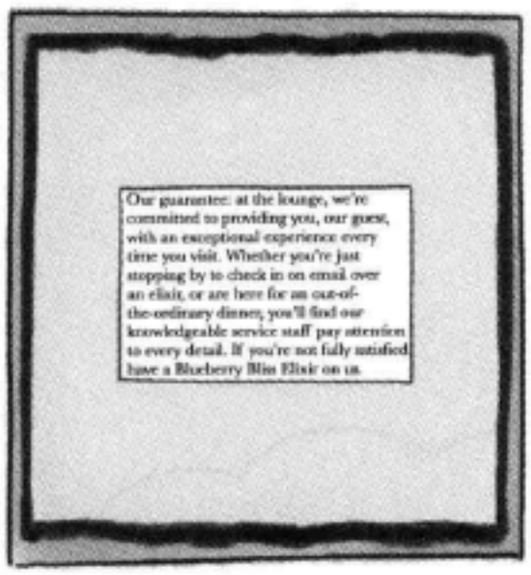
或者甚至对边框创建圆角。



内边距



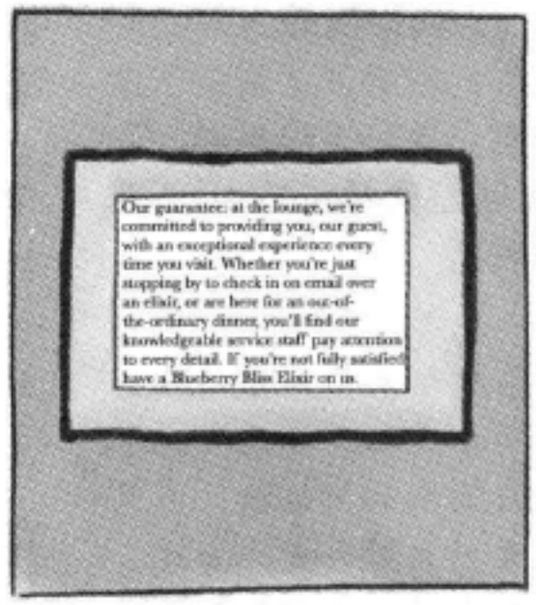
利用CSS，可以控制内容区任意一边的内边距。这里有很大的左右内边距。



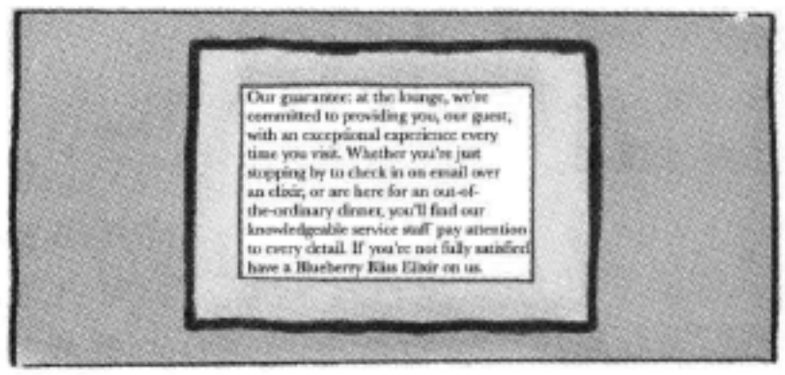
这里有很大的上下内边距。

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。

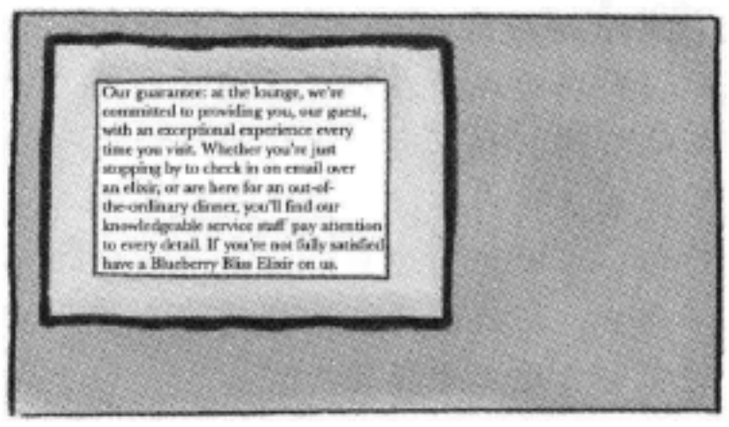


这里有很大的左右外边距。



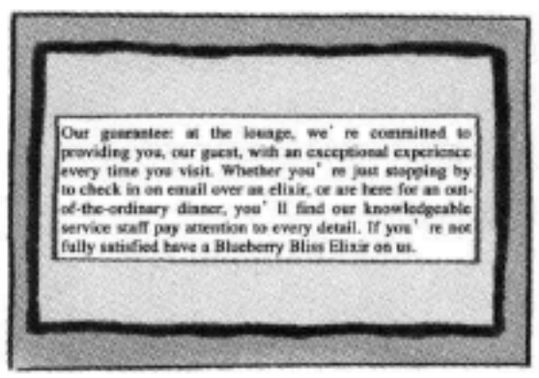
这里利用上边和左边的内边距将内容移到右下角。

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？

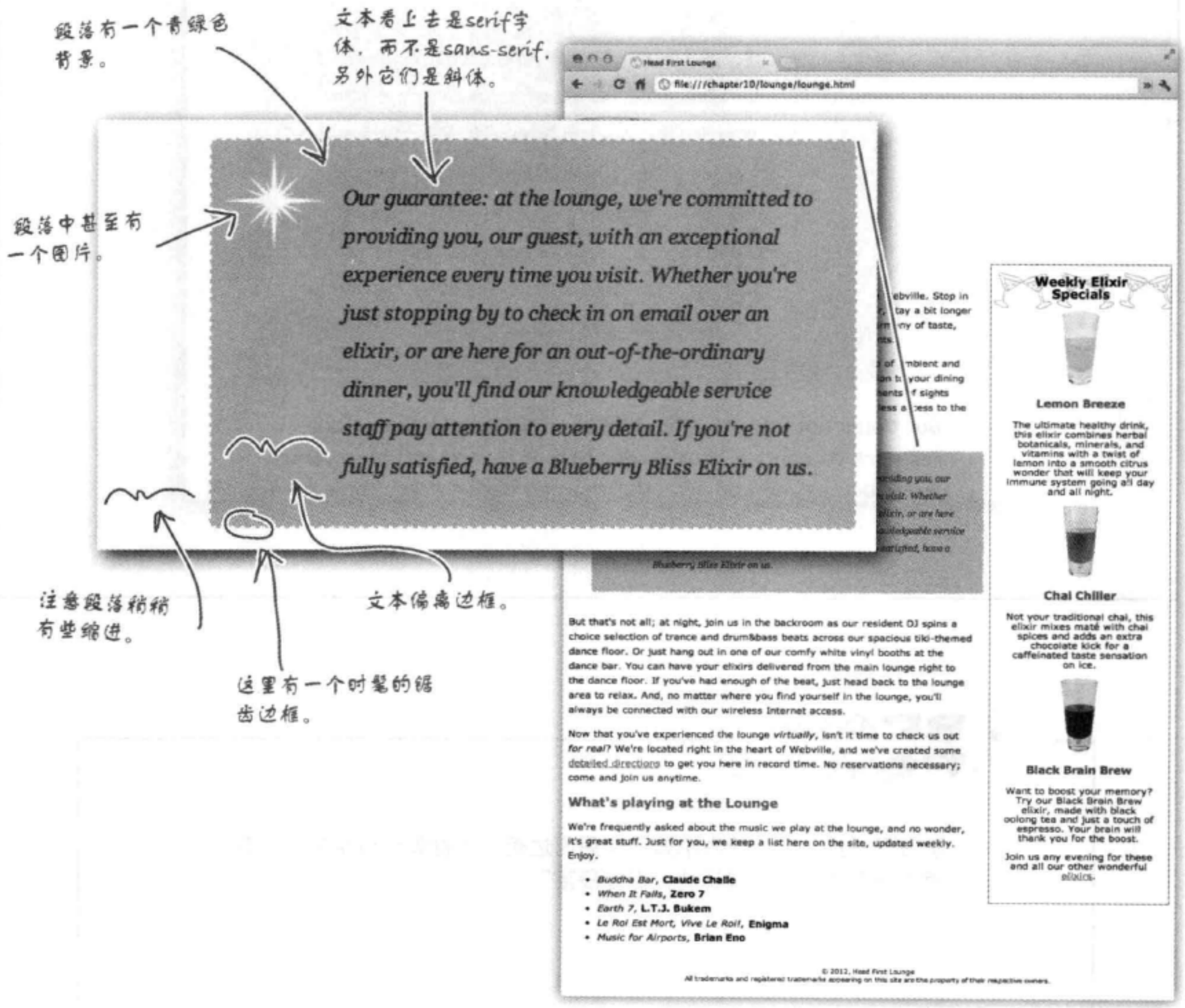
答：基本正确。内边距和外边距都是用来提供更多可见的空间，不能对内边距和外边距指定颜色，也不能加任何装饰。不过，由于它们是透明的，所以他们会呈现背景颜色或背景图像。内边距和外边距之间有一个区别：元素的背景颜色（或背景图像）会延伸到内边距下面，但不会延伸到外边距。稍后你会看到具体是怎样的。

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.

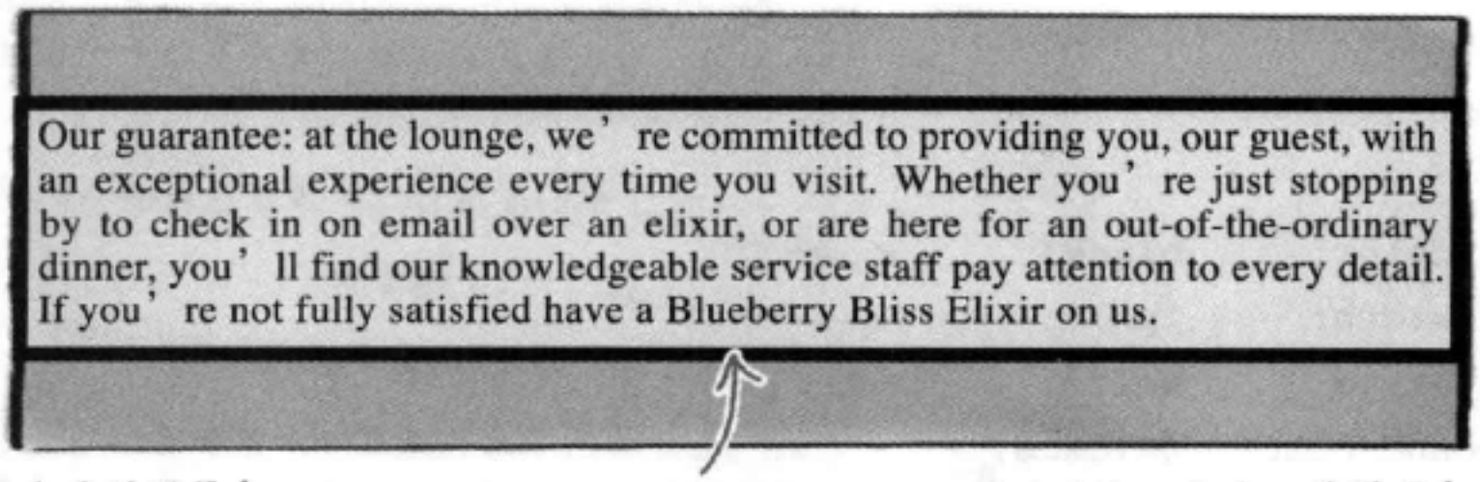
But that's not all; at night, join us in the backroom as our resident DJ spins a choice selection of trance and drum&bass beats across our spacious tiki-themed dance floor. Or just hang out in one of our comfy white vinyl booths at the dance bar. You can have your elixirs delivered from the main

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。

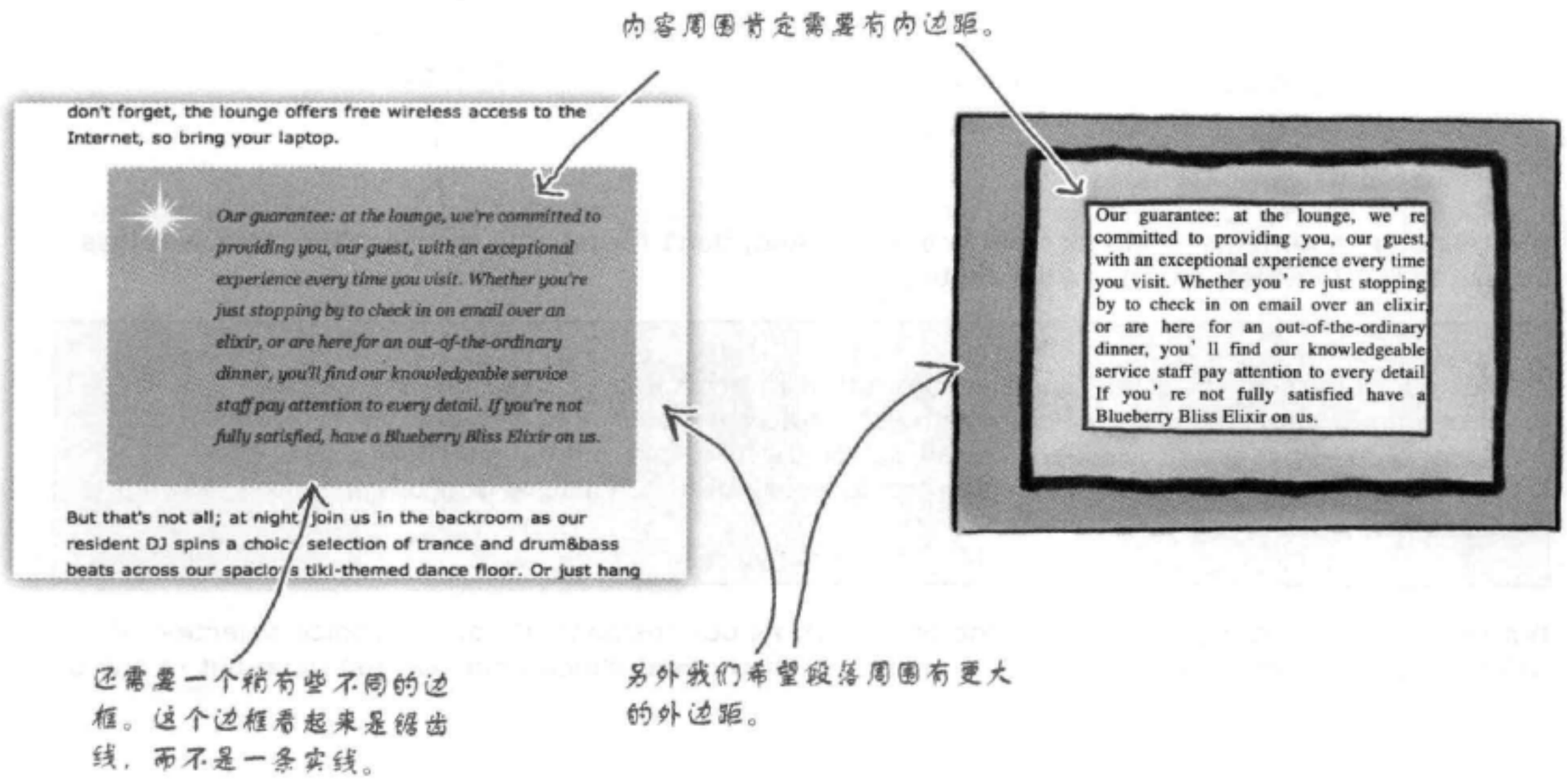


但是左右外边距很小。

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

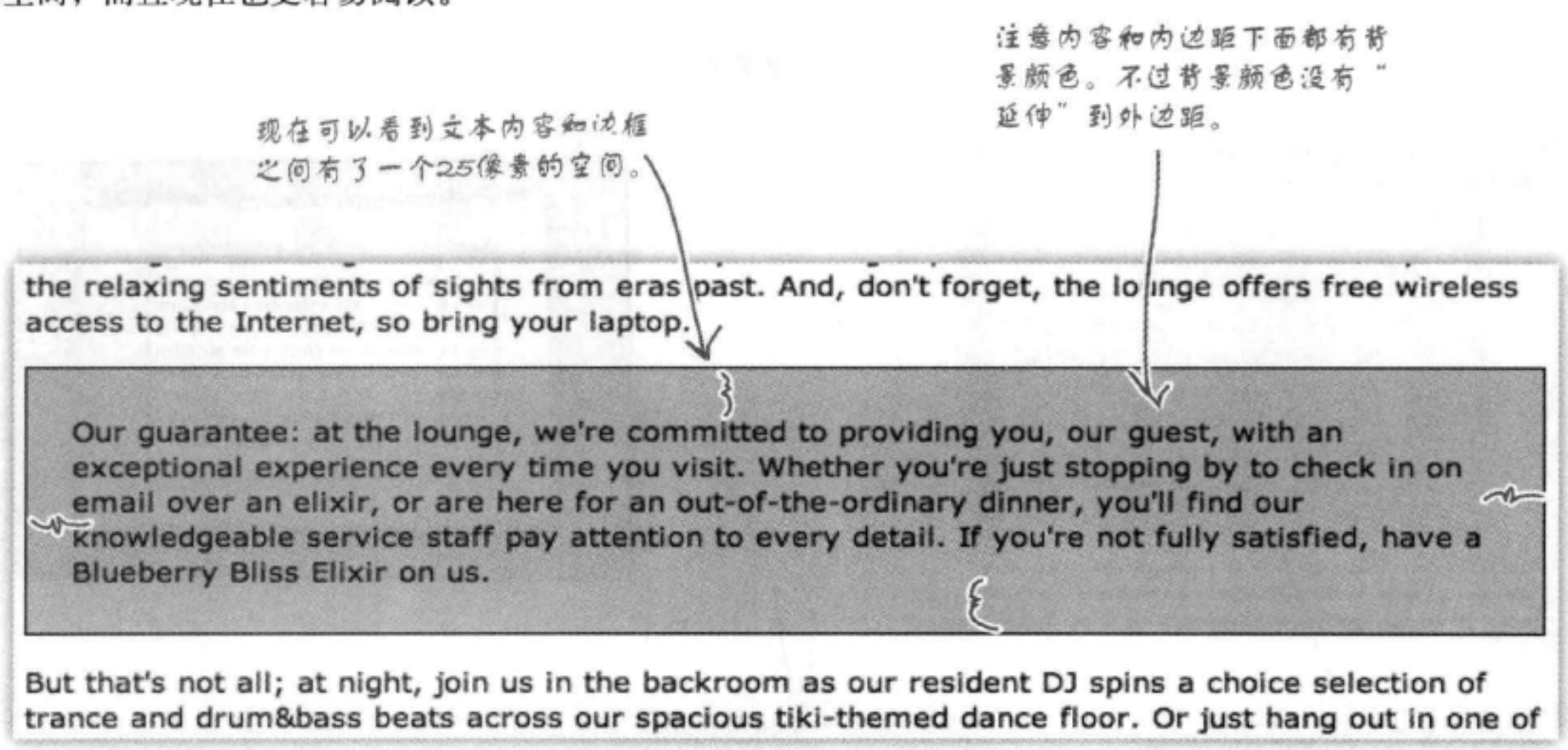
下面先来增加内边距。CSS有一个padding属性，可以用来对内容四周设置相同的内边距。可以将这个属性设置为一个像素数，或者也可以设置为一个百分数。我们将使用像素，把内边距设置为25像素。

```
.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

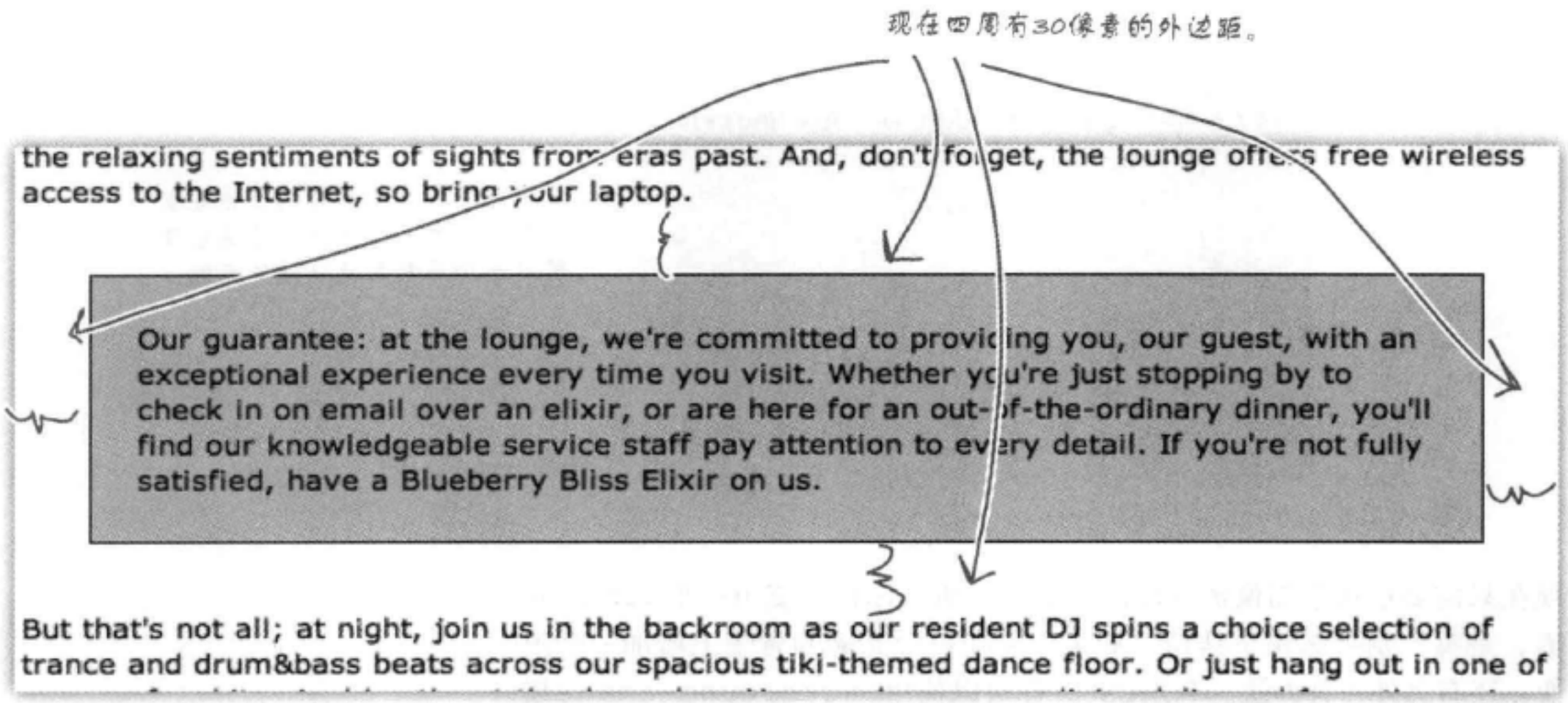
使用CSS很容易增加外边距。类似于内边距，可以将外边距指定为一个百分数或者按像素指定。这里将在整个保证段落周围增加一个30像素的外边距。可以这样做：

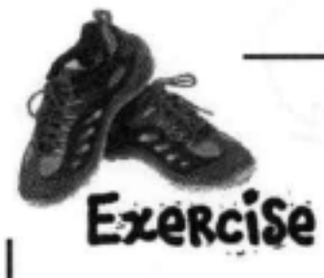
```
.guarantee {
      border-color:     black;
      border-width:     1px;
      border-style:     solid;
      background-color: #a7cece;
      padding:          25px;
      margin:           30px;
}
```

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。





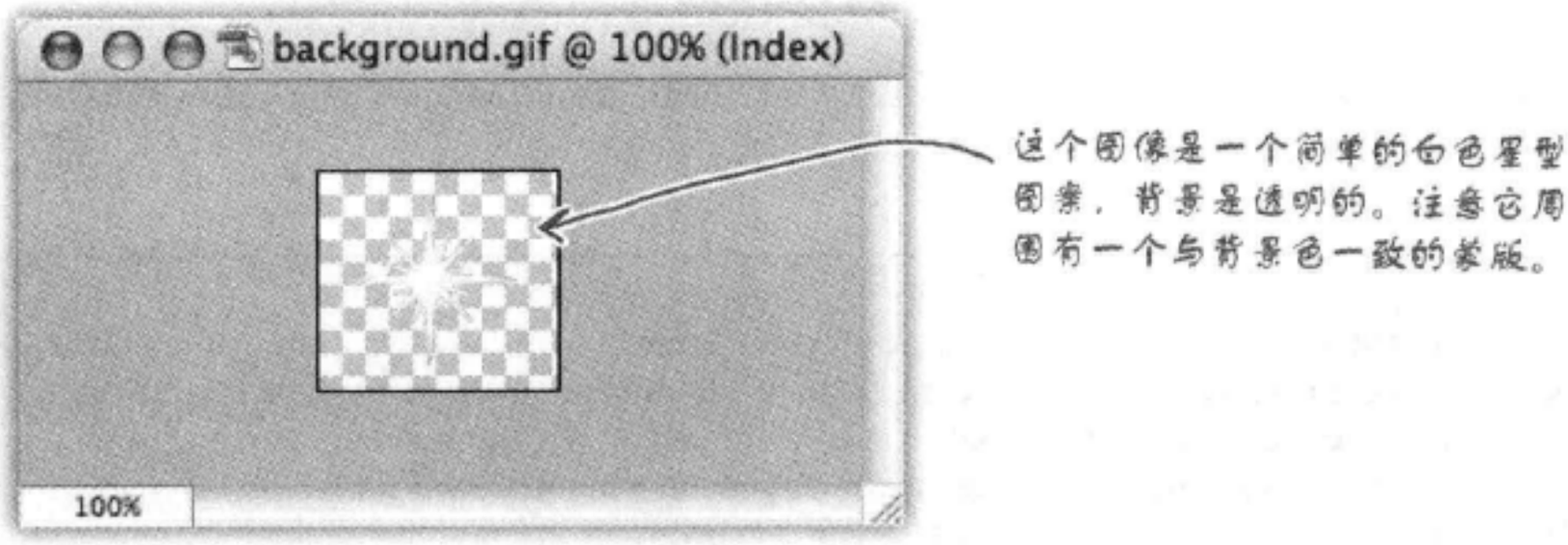
Exercise

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：

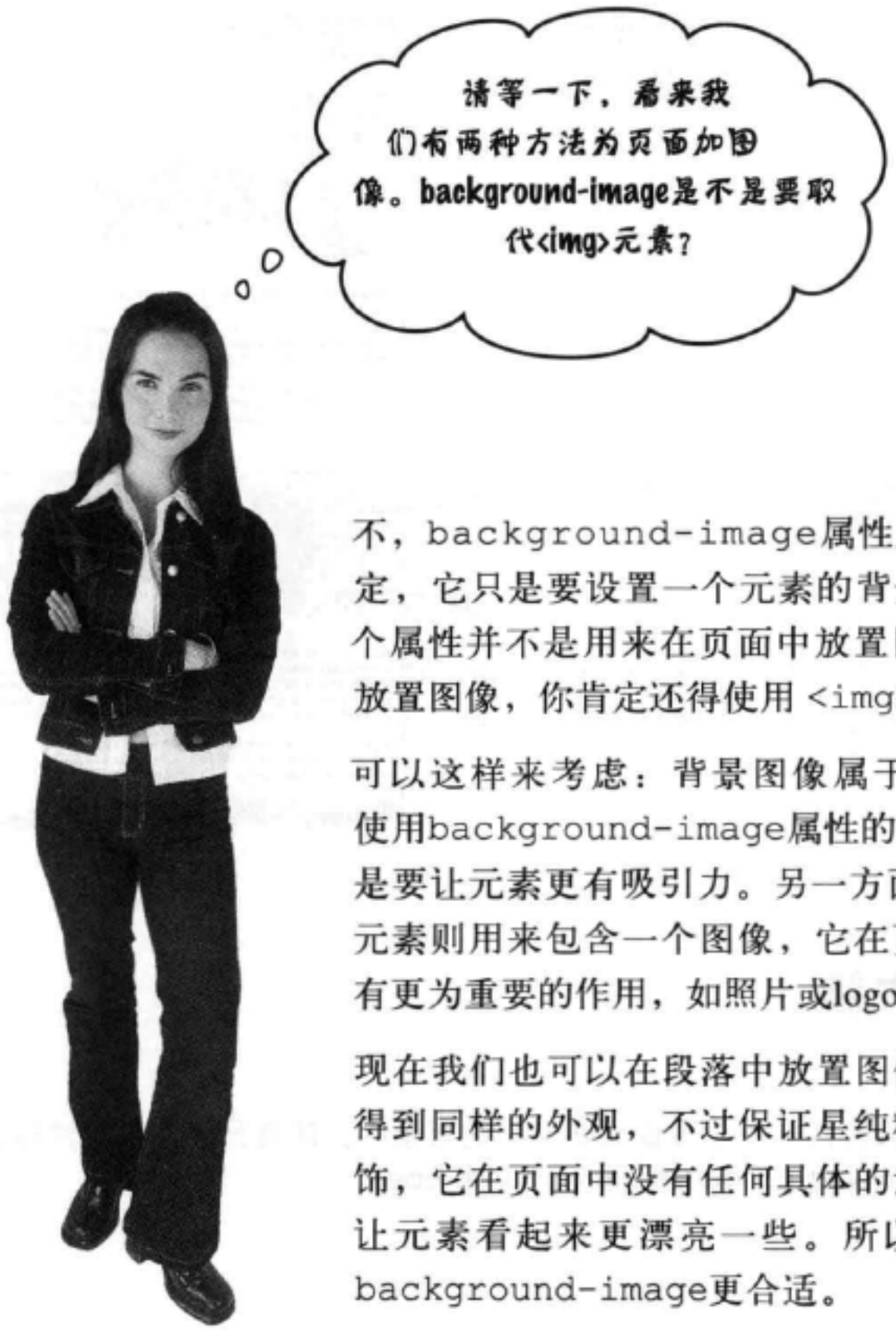


现在只需要把这个图像放入段落元素中，所以你打算使用一个<img>元素，是吗？别那么快下结论。如果你要在一个元素的背景上增加一个图像，还有另外一种办法。利用CSS，你可以使用background-image属性为任何元素增加一个背景图像。下面来试一试，看看能不能奏效：

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;
}
```

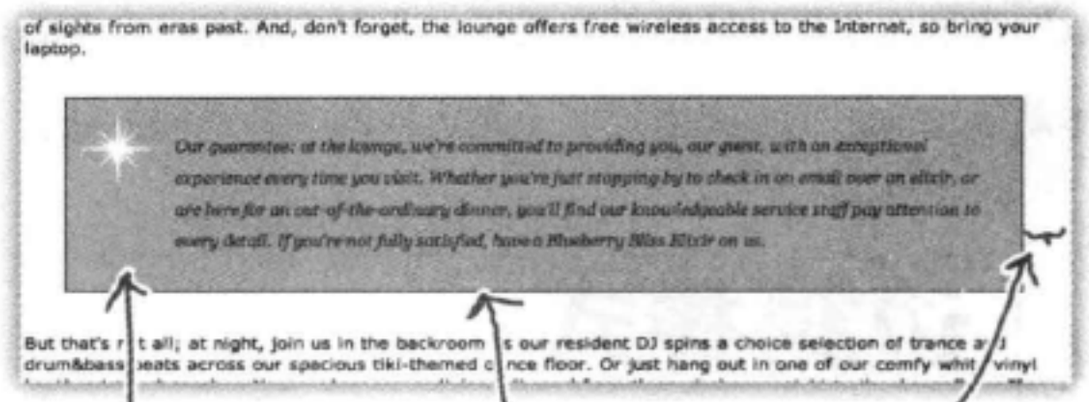
这里使用padding-left属性在左边增加内边距……

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。

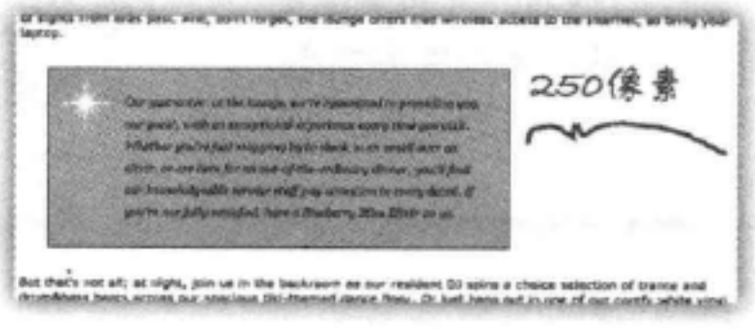
看出规律了吧？首先有一个属性来控制所有4个边，另外如果你想单独对每个边进行设置，实际上对于每个边还分别有一个属性。

```
.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

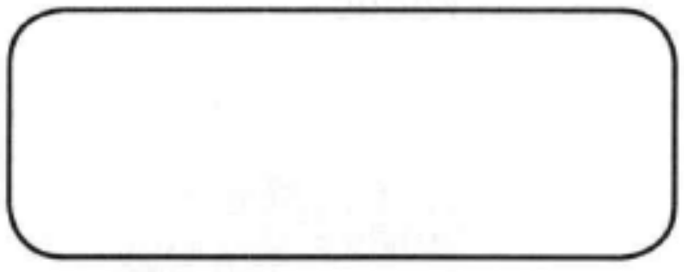
就像外边距和内边距一样，还可以指定任意一边（上、右、下和左）的边框样式、宽度和颜色。

```
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。

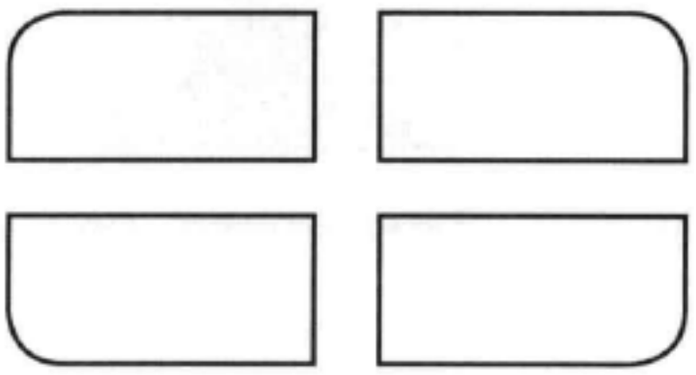
指定边框圆角

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。

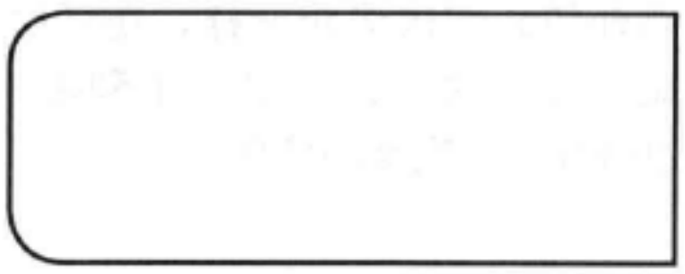
```
border-radius: 15px;
```



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。

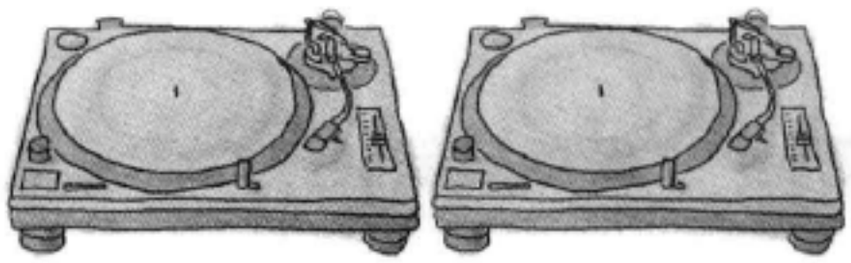
问：是不是总得先从类开始，然后如果知道这个元素是唯一的，再改成id？

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

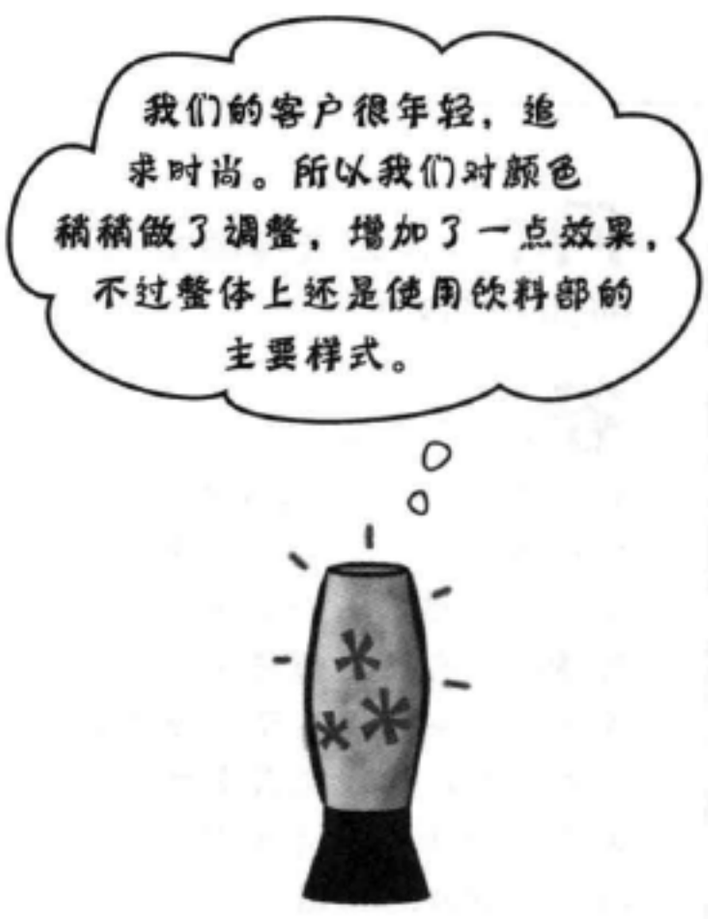


结束这一章之前，下面做个有意思的事情，我们要混合一些样式表。到目前为止，你一直都只是使用一个样式表。嗯，有人说过不能用多个样式表吗？你完全可以指定一组样式表用于任何HTML。不过，你可能想知道为什么会有人希望这么做。这有很多充分的理由。下面先说第一个原因……

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!

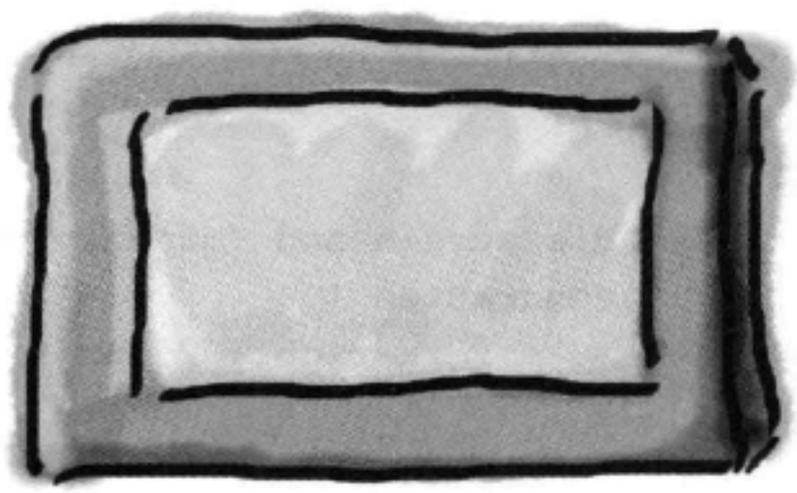
IE8及以前版本不支持媒体查询。

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？

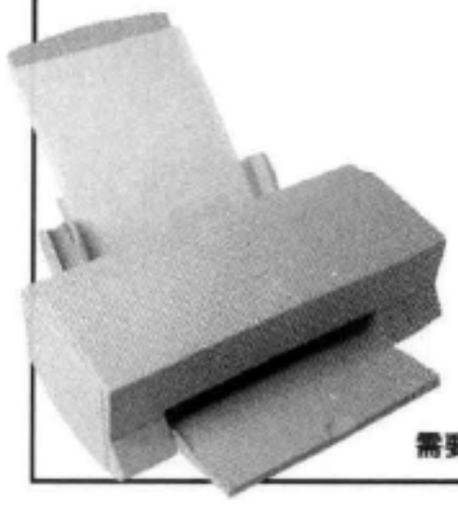
答：这两个都行。不过要注意，如果把所有规则都放在一个文件中，再使用@media规则将它们分开，你的CSS会变得非常庞大。通过为不同的媒体类型使用不同的<link>元素，就能按照媒体类型在不同文件中组织CSS。所以，如果你的CSS文件相当庞大，我们就建议使用<link>元素来指定不同的样式表。

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？

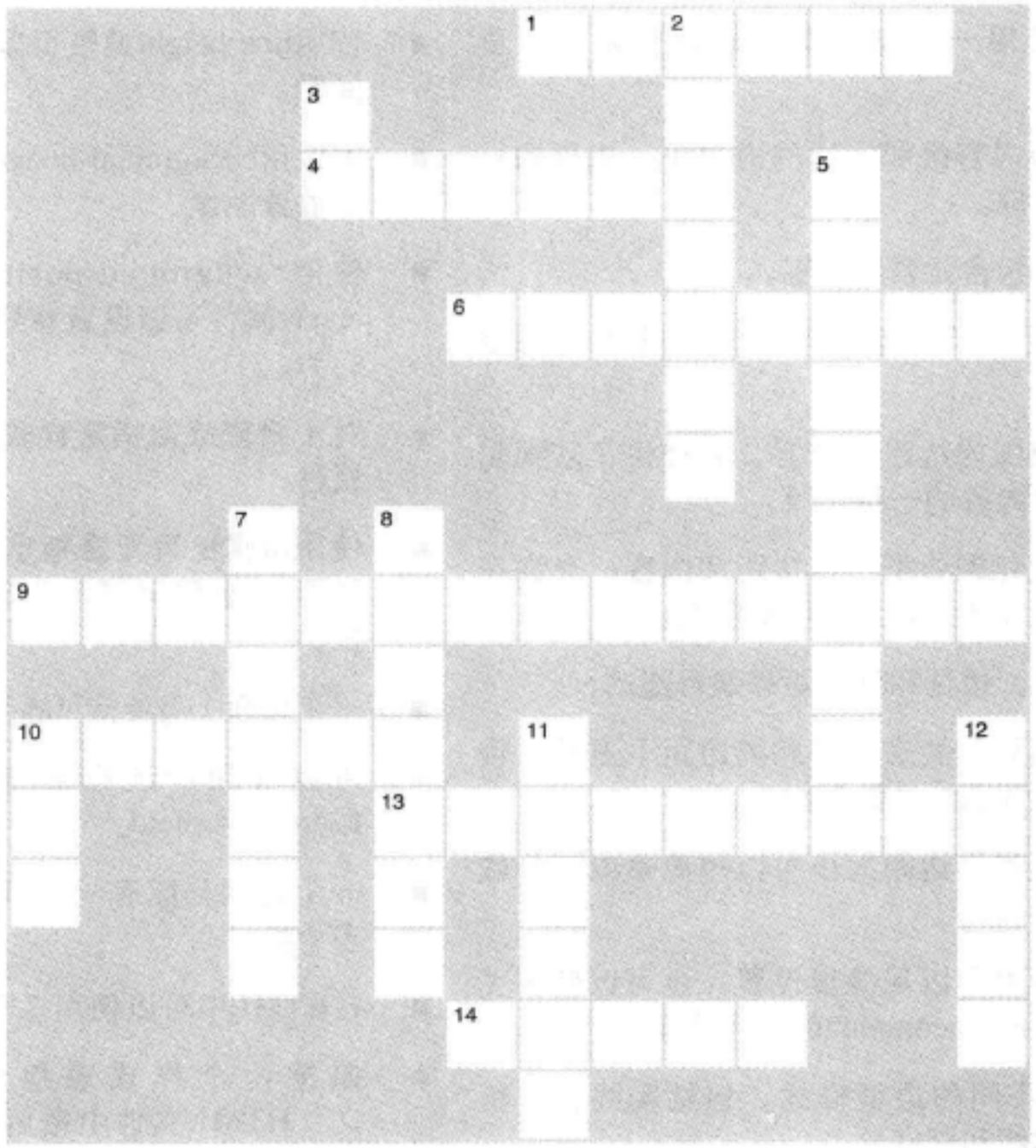
一定要使用一个现代浏览器！如果你在使用IE，这就意味着必须使用IE9以上的版本。

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

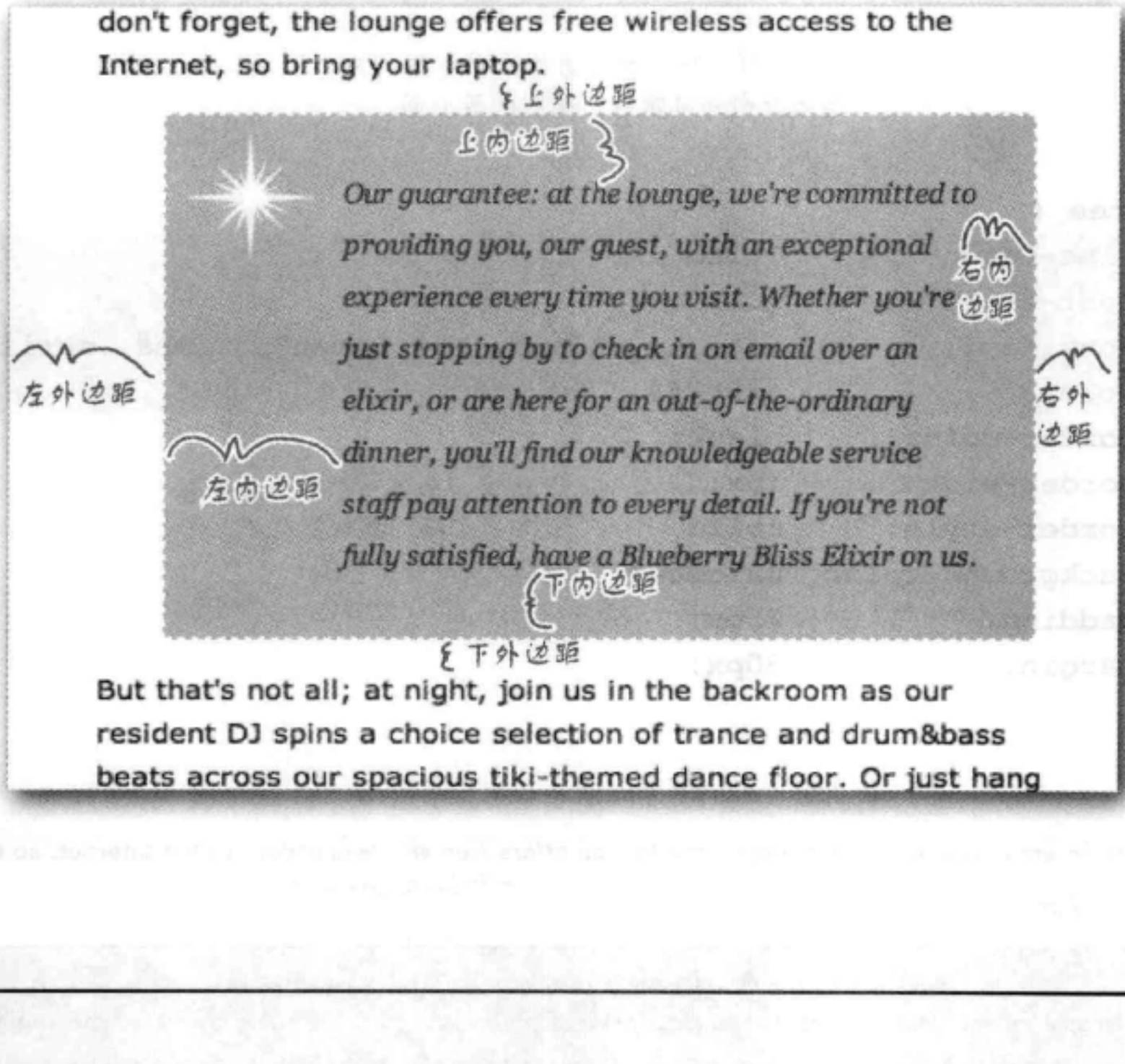
1. 默认地，背景图像会做这件事。
4. 要创建一个“锯齿”边框，要使用_____边框样式。
6. 内边距、边框和外边距都是 _____。
9. 如果你还不太满意，你会拿哪种饮料？
10. 在内边距和外边距之间。
13. 我们把 _____ 类改为一个id。
14. 要为不同的设备使用不同的样式，需要使用 _____ 查询。

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。

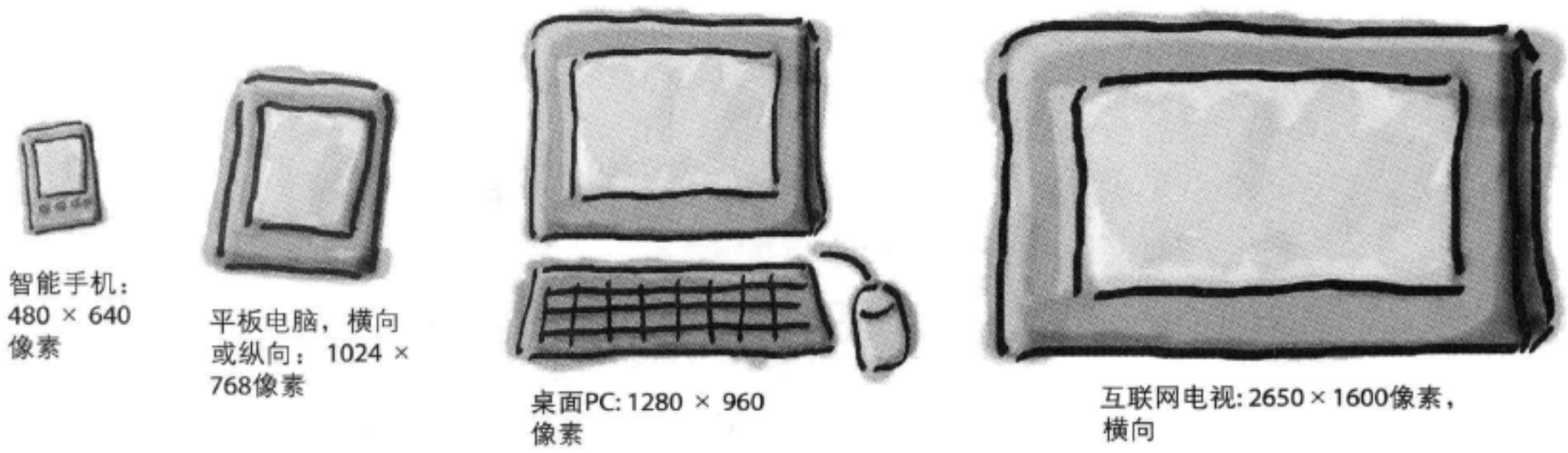
But that's not all; at night, join us in the backroom as our resident DJ spins a choice selection of trance and drum&bass beats across our spacious tiki-themed dance floor. Or just hang out in one of our comfy white vinyl

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

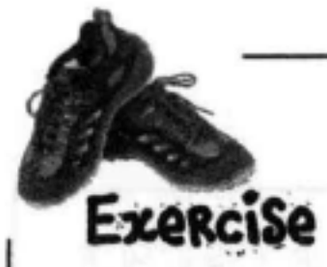
Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention 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：

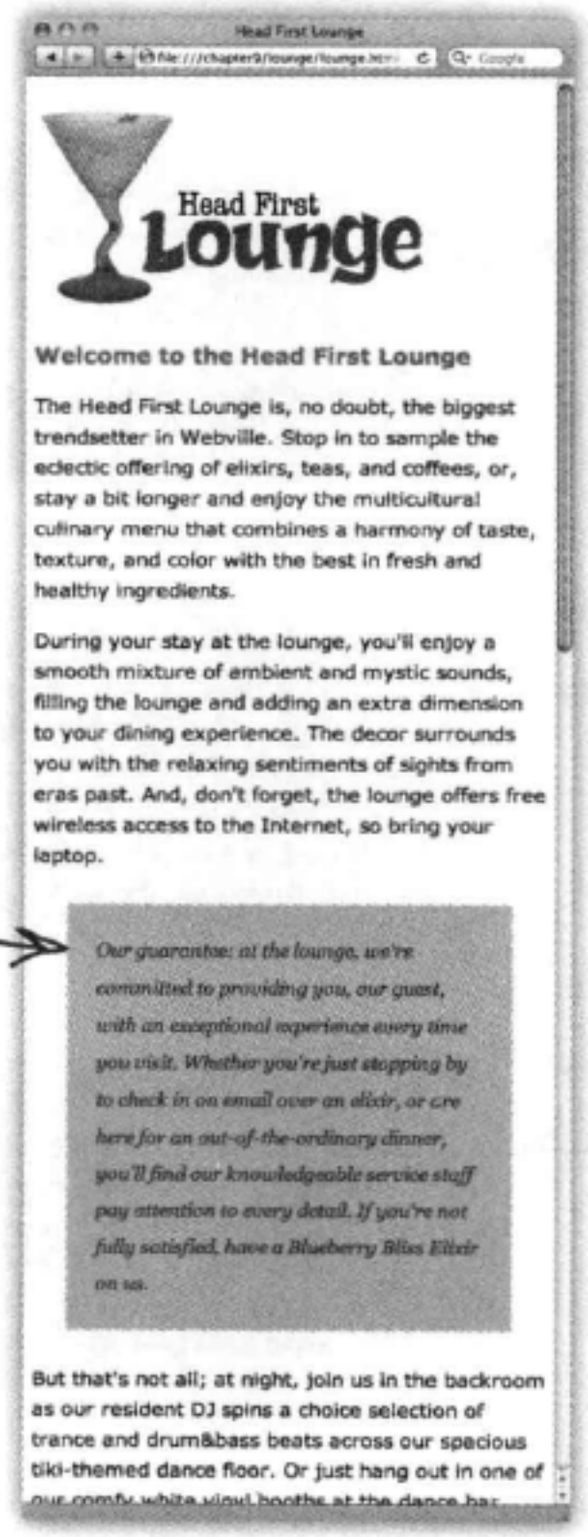
```
<link type="text/css" rel="stylesheet" href="lounge.css"
      media="screen and (min-width: 481px)">
<link type="text/css" href="lounge-mobile.css" rel="stylesheet"
      media="screen and (max-width: 480px)">
<link type="text/css" href="lounge-print.css" rel="stylesheet" media="print">
```

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。

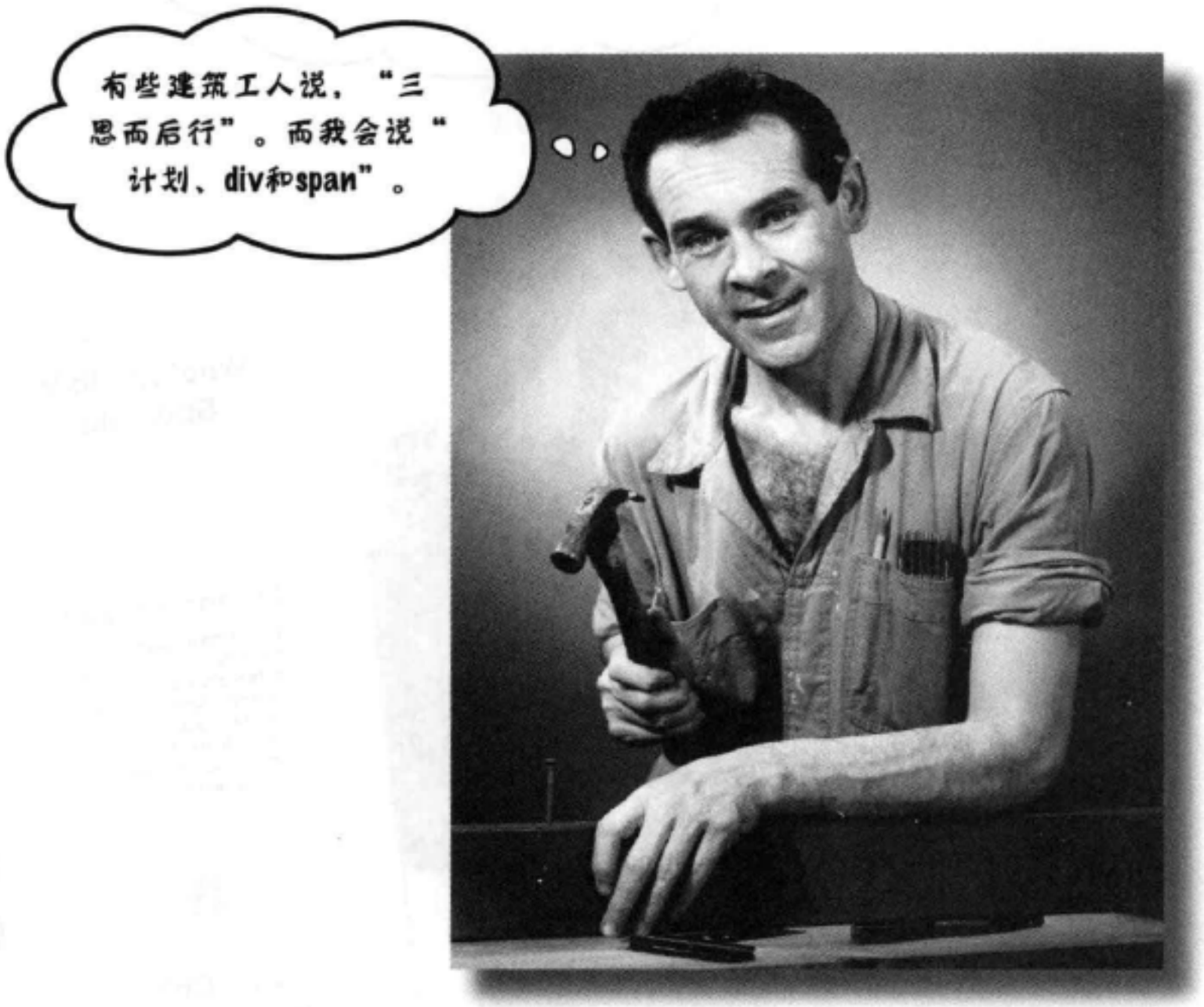
这个版本更适合移动浏览器，因为如果使用为更宽屏幕设计的CSS，保证段落会变得太窄。通过删除背景图像和额外的外边距和内边距，段落将更容易阅读。再说了，最重要的还是内容，不是吗？



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。

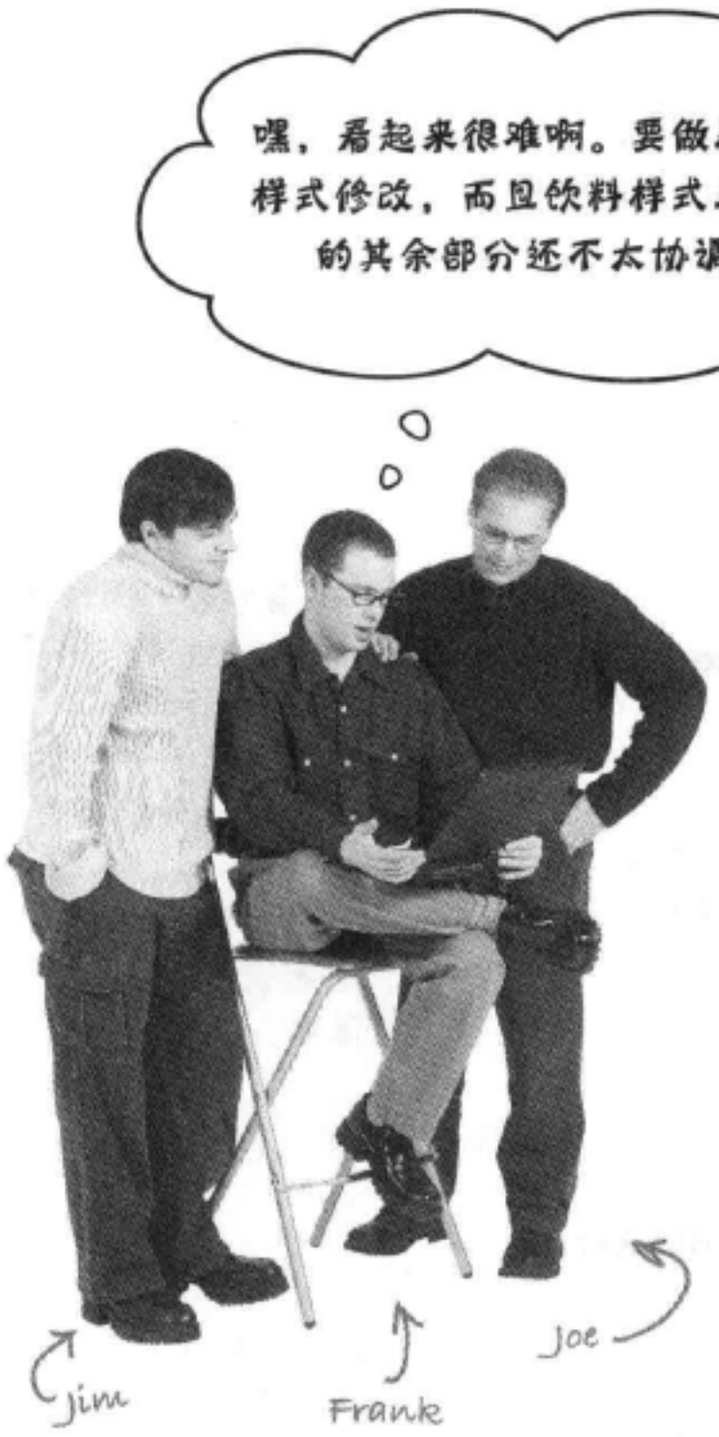
每种饮料有一个<p>元素，其中有一个图像。

……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……

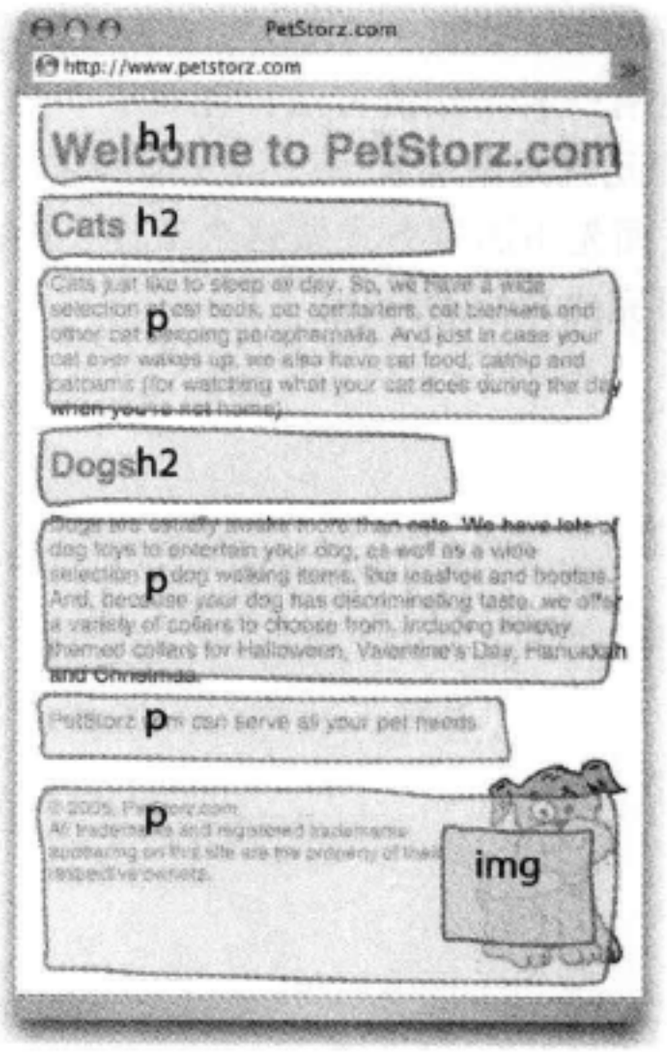
下面来研究如何将一个页面划分为逻辑区

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。

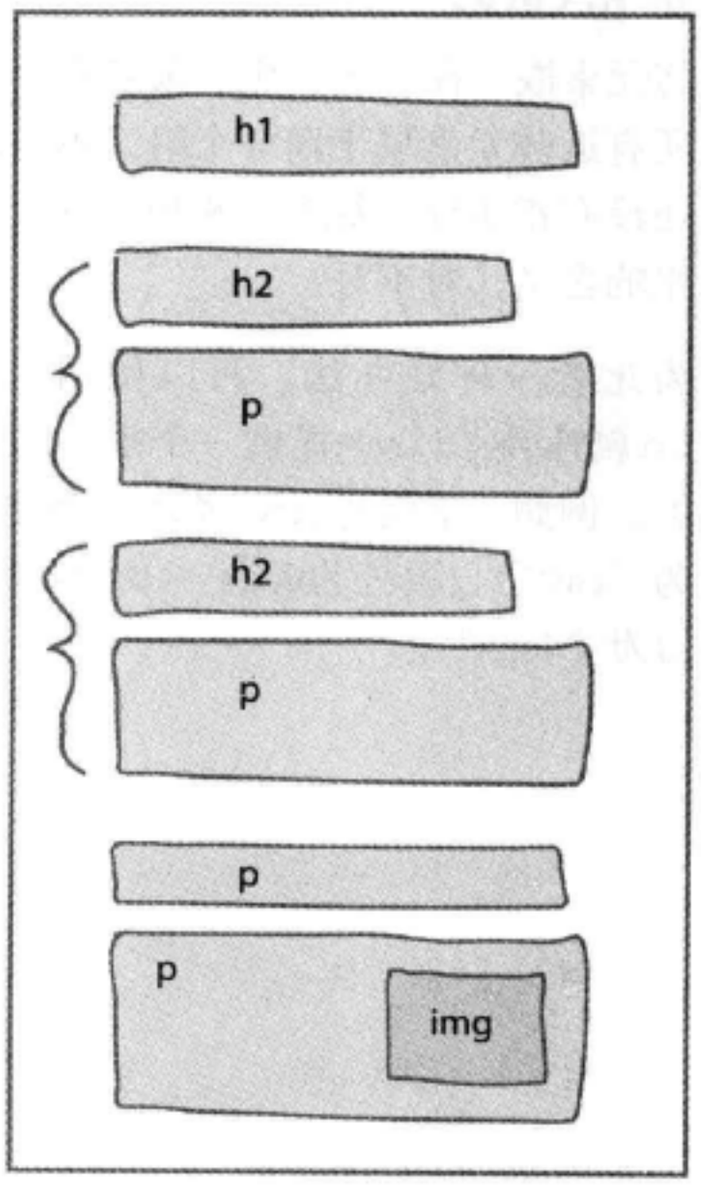


找出逻辑区

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。

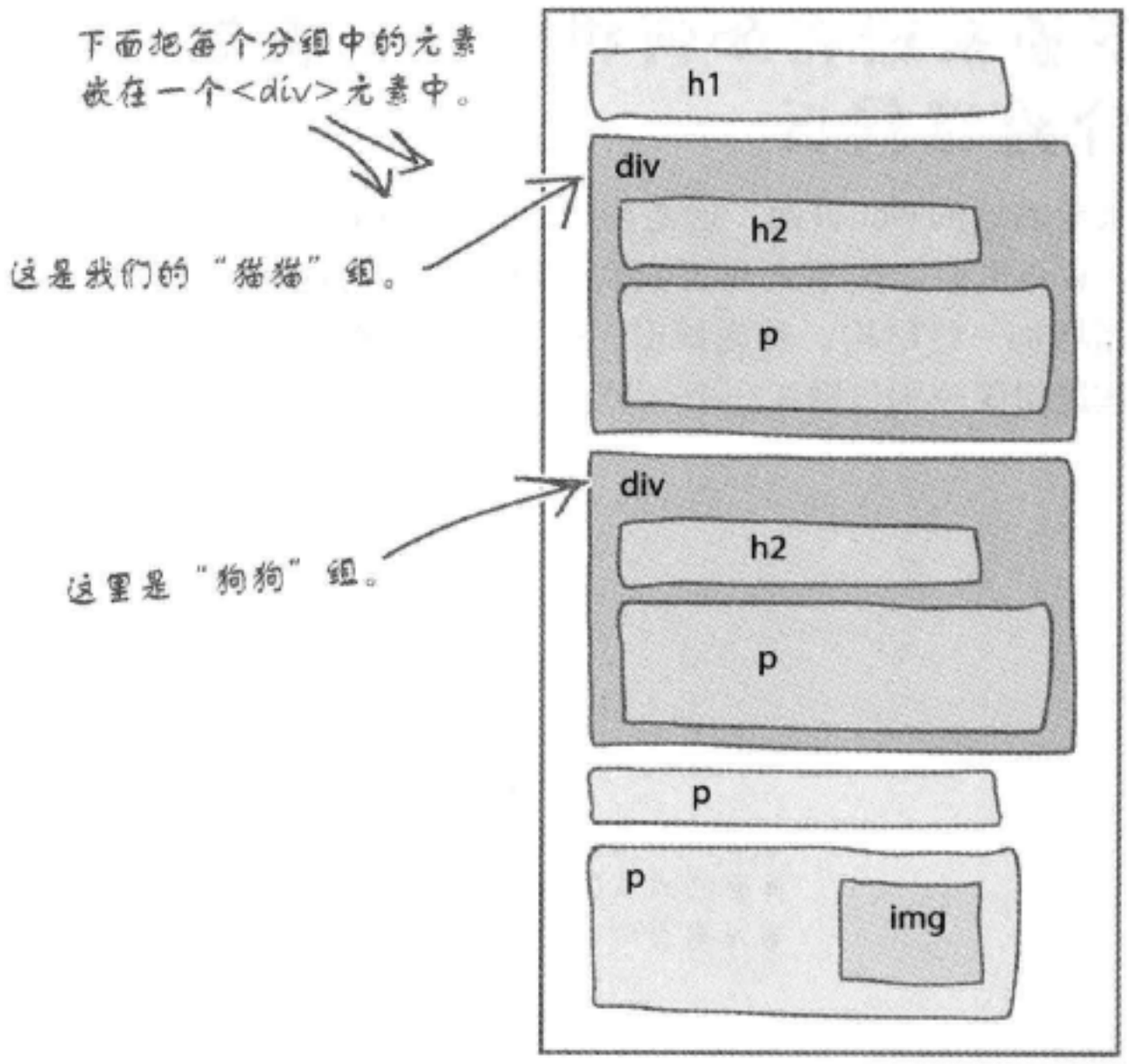
PetStorz页面有两个主要的内容区，一个对应“猫”，另一个对应“狗”。还有另外一些区域，不过稍后再来讨论那些区域。

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

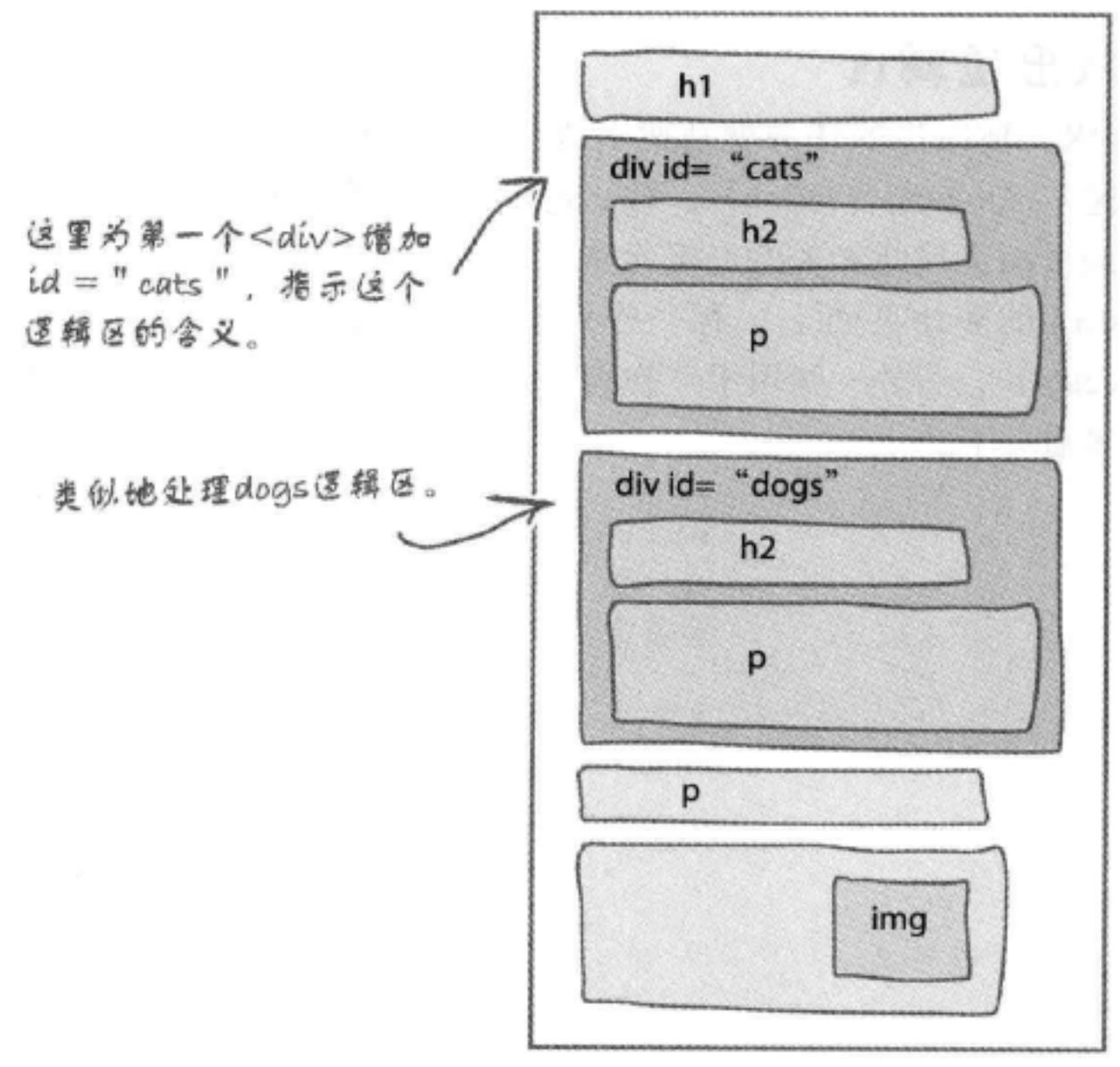
既然已经知道各个逻辑区包括哪些元素，下面可以增加一些HTML来标记这个结构。这有一种常用的方法：在属于一个逻辑区的元素周围放置 开始和结束标记。下面先用图解方式做这个工作，后面几页再完成真正的标记。



标出

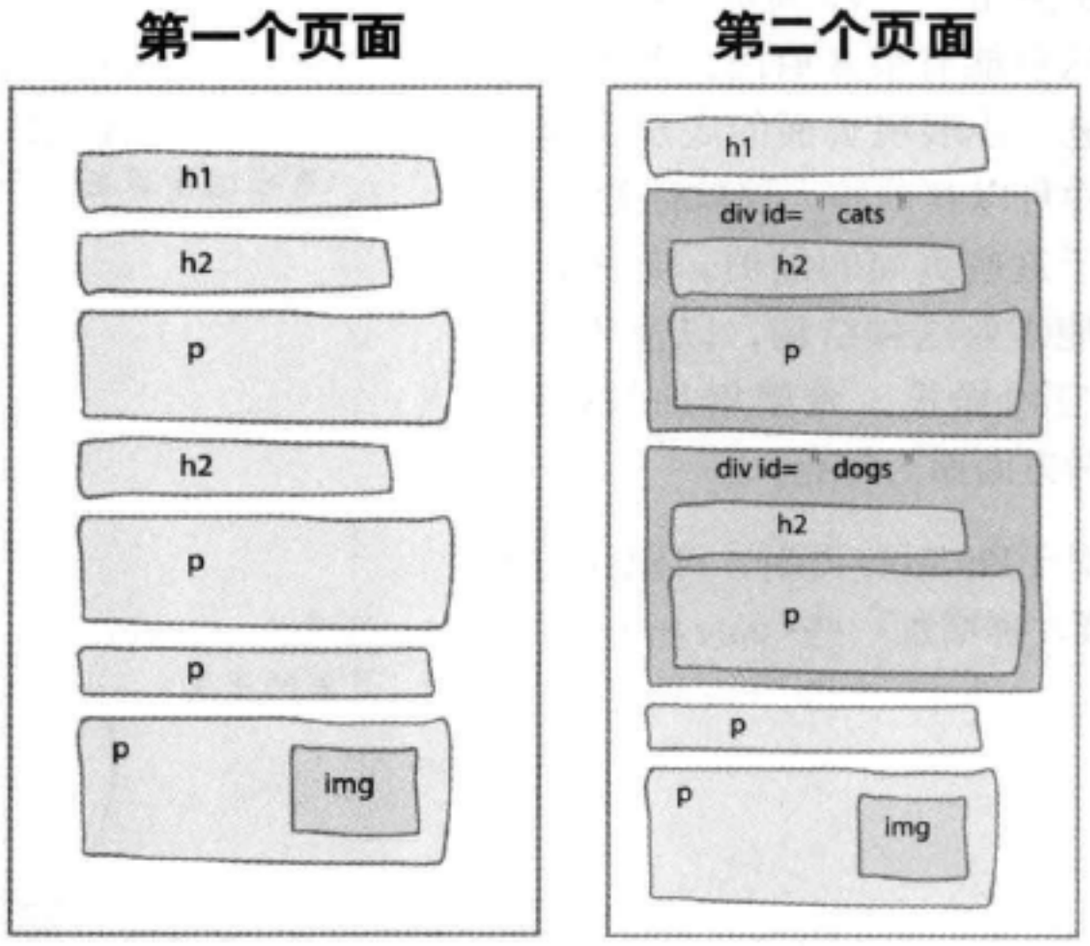
将元素嵌入在 中，就是在指示所有这些元素属于同一个组。不过你还没有指定任何标签，来说明这个分组的含义，对不对？

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。

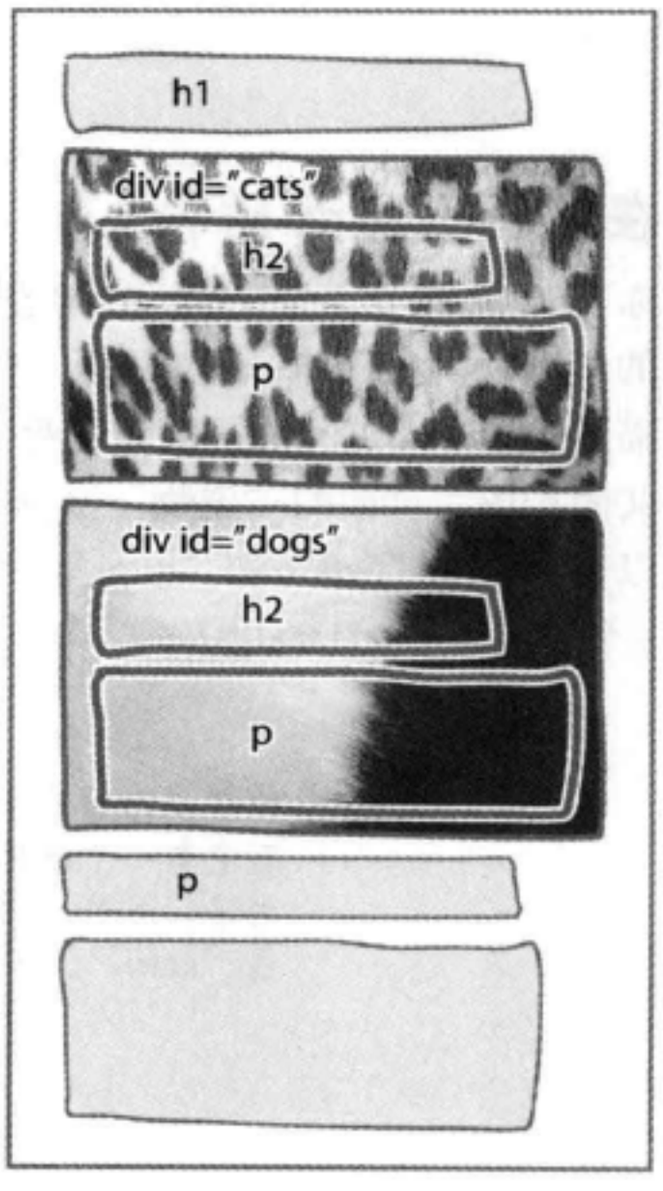
```
#cats {
   background-image: url(leopard.jpg);
}

#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。

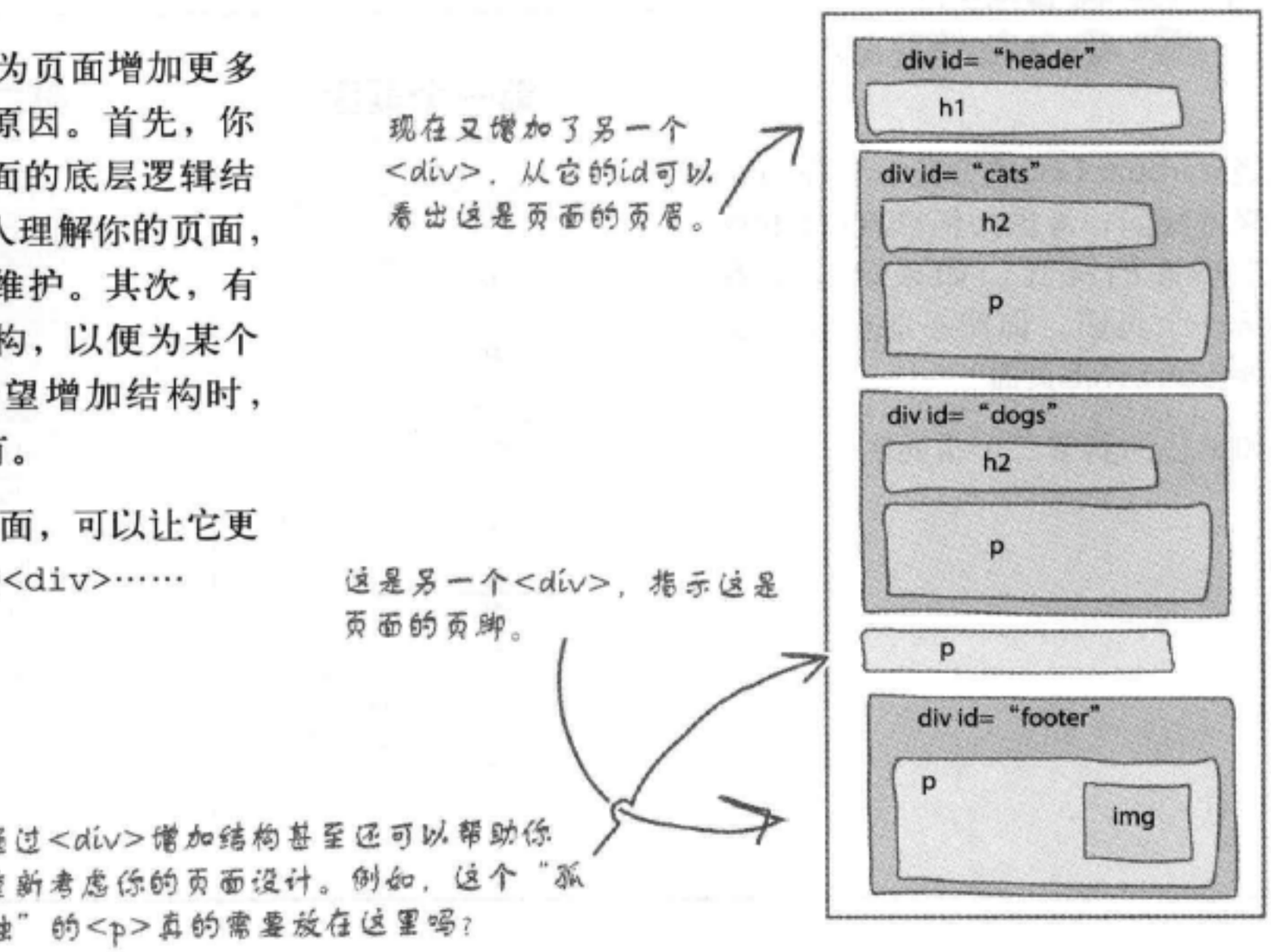
就像所有子元素一样， 中的元素也会从 继承一些属性（如font-size、color等）。



展现更多结构

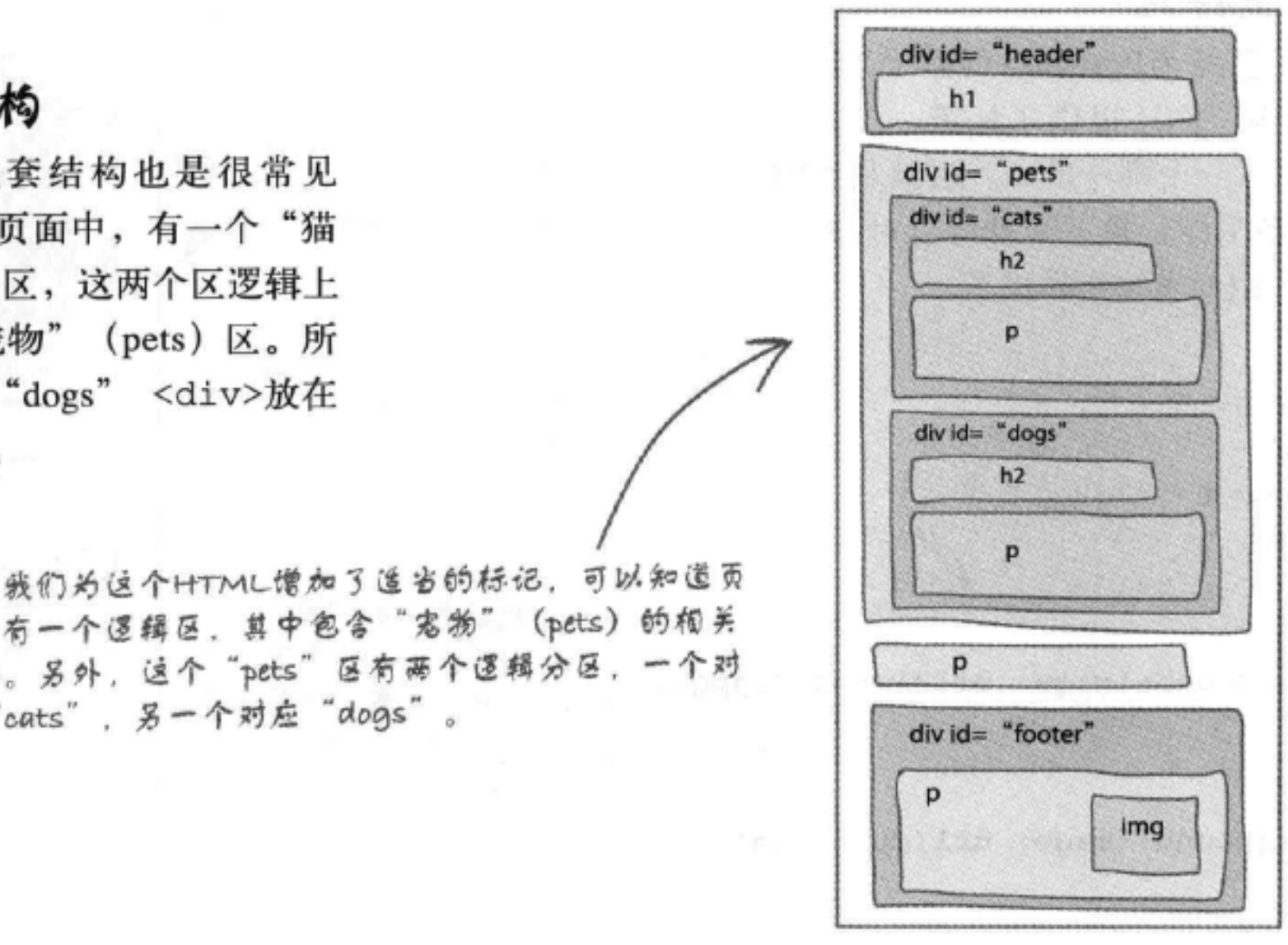
你可能希望用 为页面增加更多结构，这可能有很多原因。首先，你可能想进一步展现页面的底层逻辑结构，这样可以帮助别人理解你的页面，还有助于这些页面的维护。其次，有时你可能需要这种结构，以便为某个逻辑区应用样式。希望增加结构时，通常这两方面原因都有。

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 。

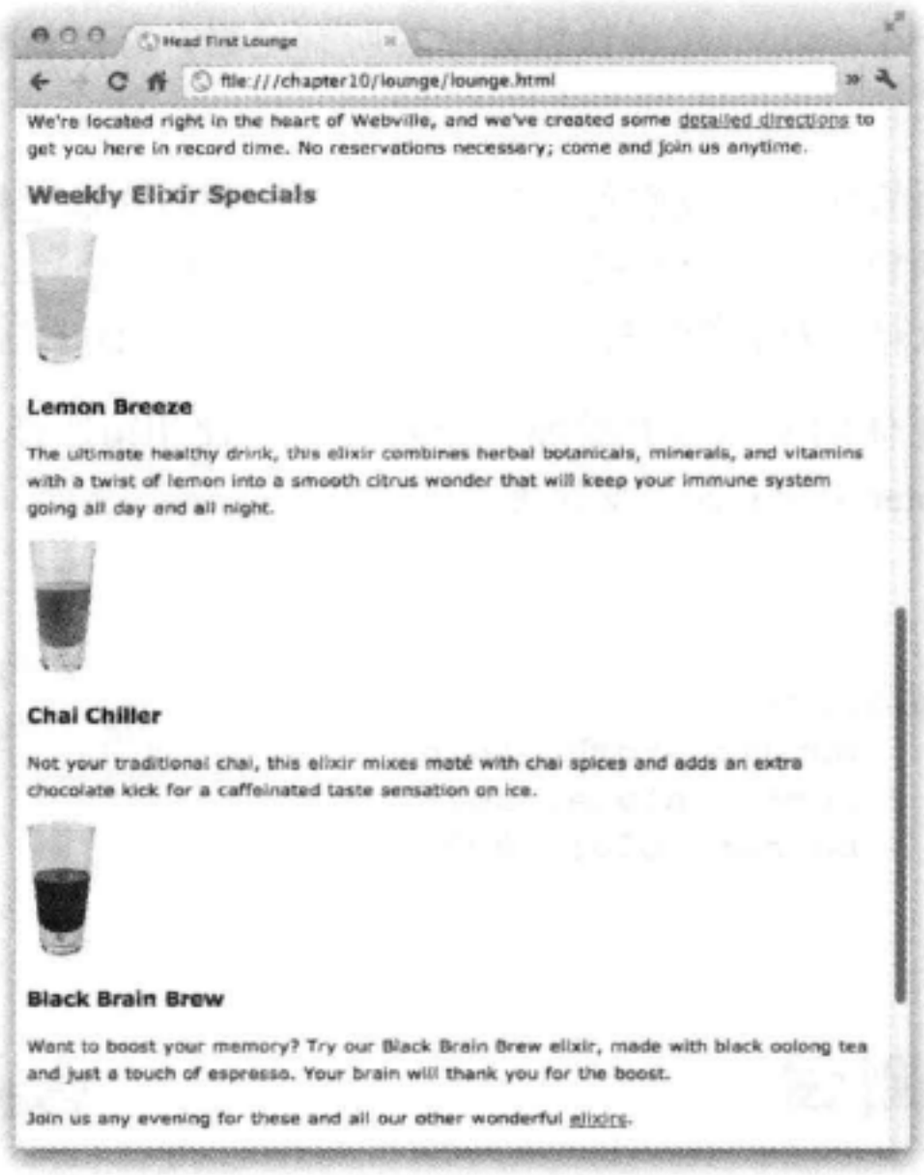
```
<div id="elixirs">
    <h2>Weekly Elixir Specials</h2>

    <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。

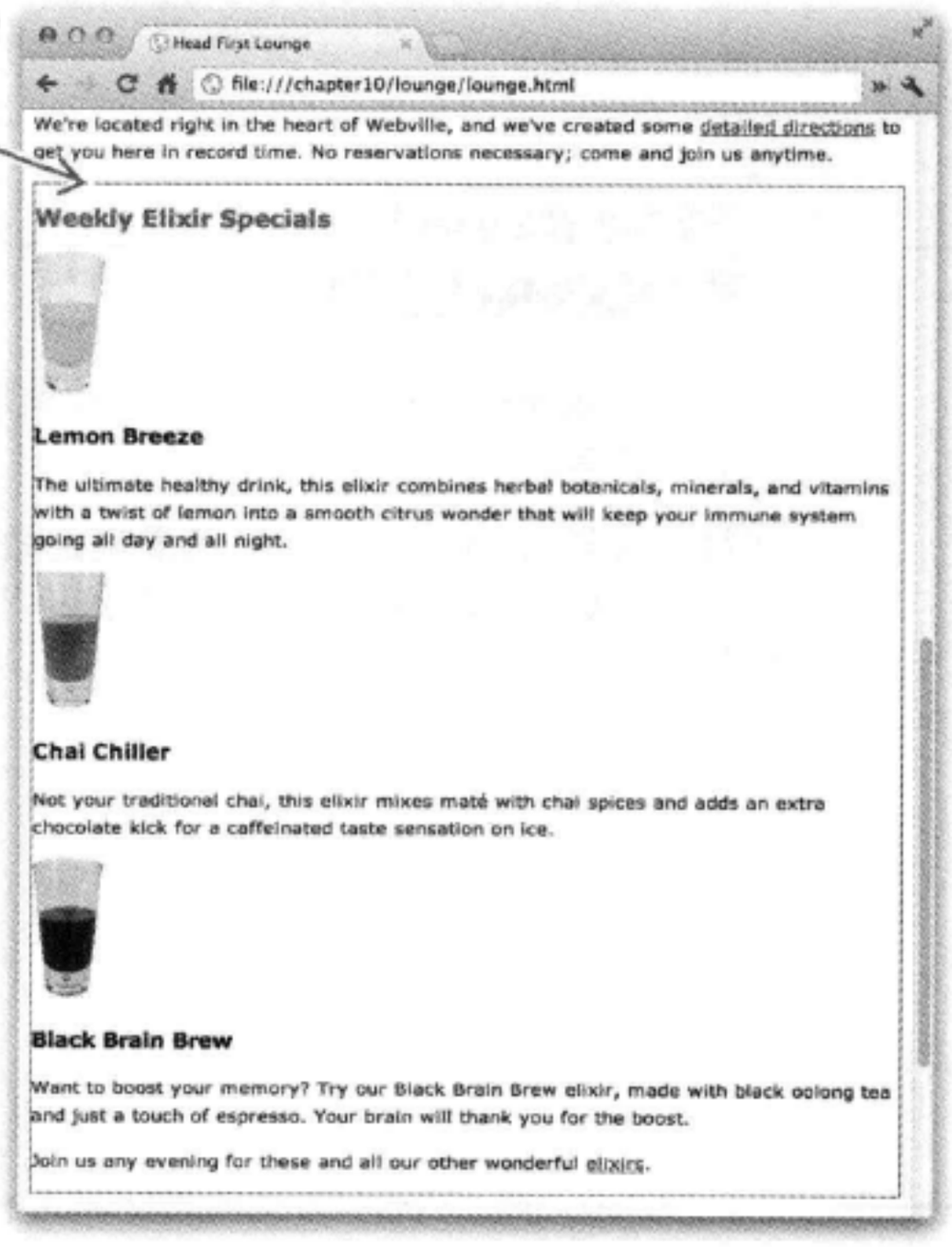
你需要在休闲室CSS中增加一个新规则，使用id选择这个 元素。打开“chapter10/lounge”文件夹中的“lounge.css”文件，在最后增加以下规则：

```
#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。

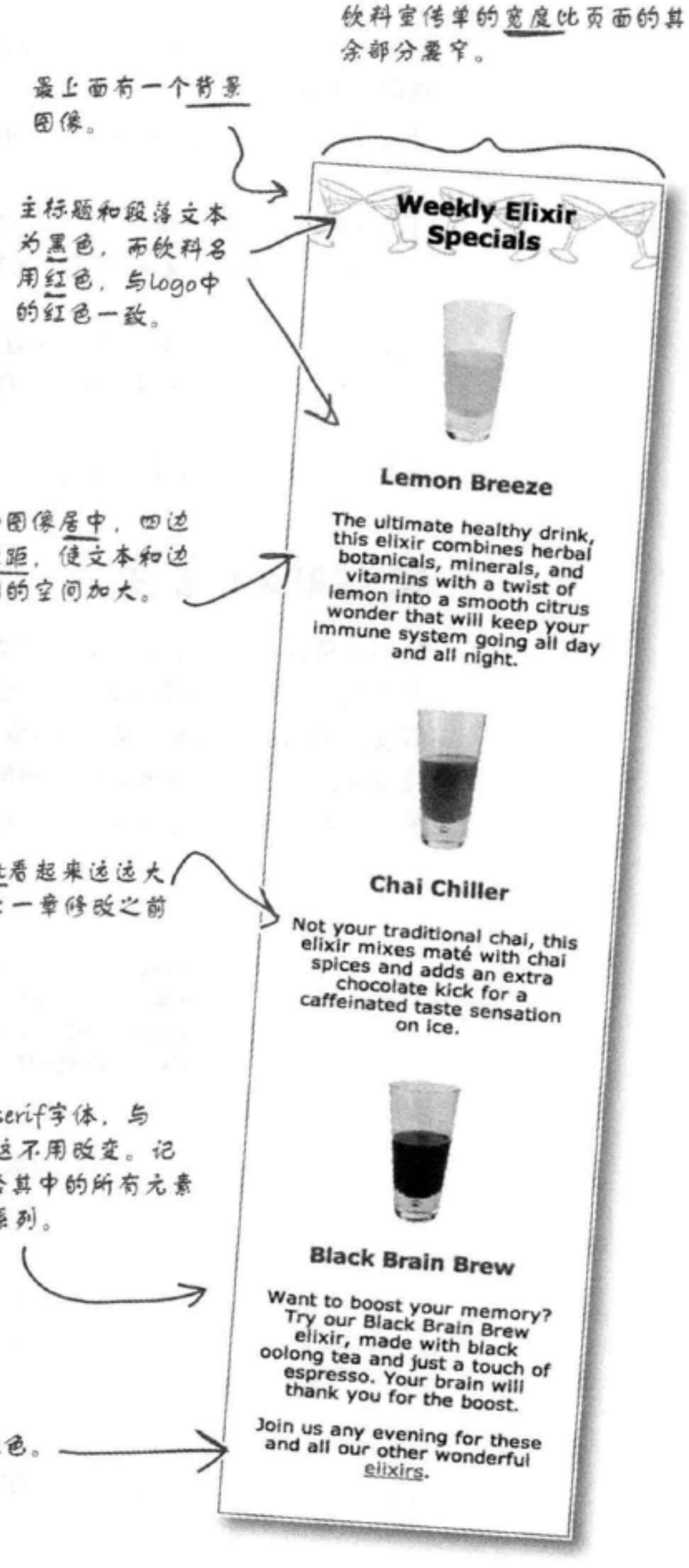
为这个 增加了一个可见的边框，不过还没有内边距和外边距，这些也需要增加。

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。

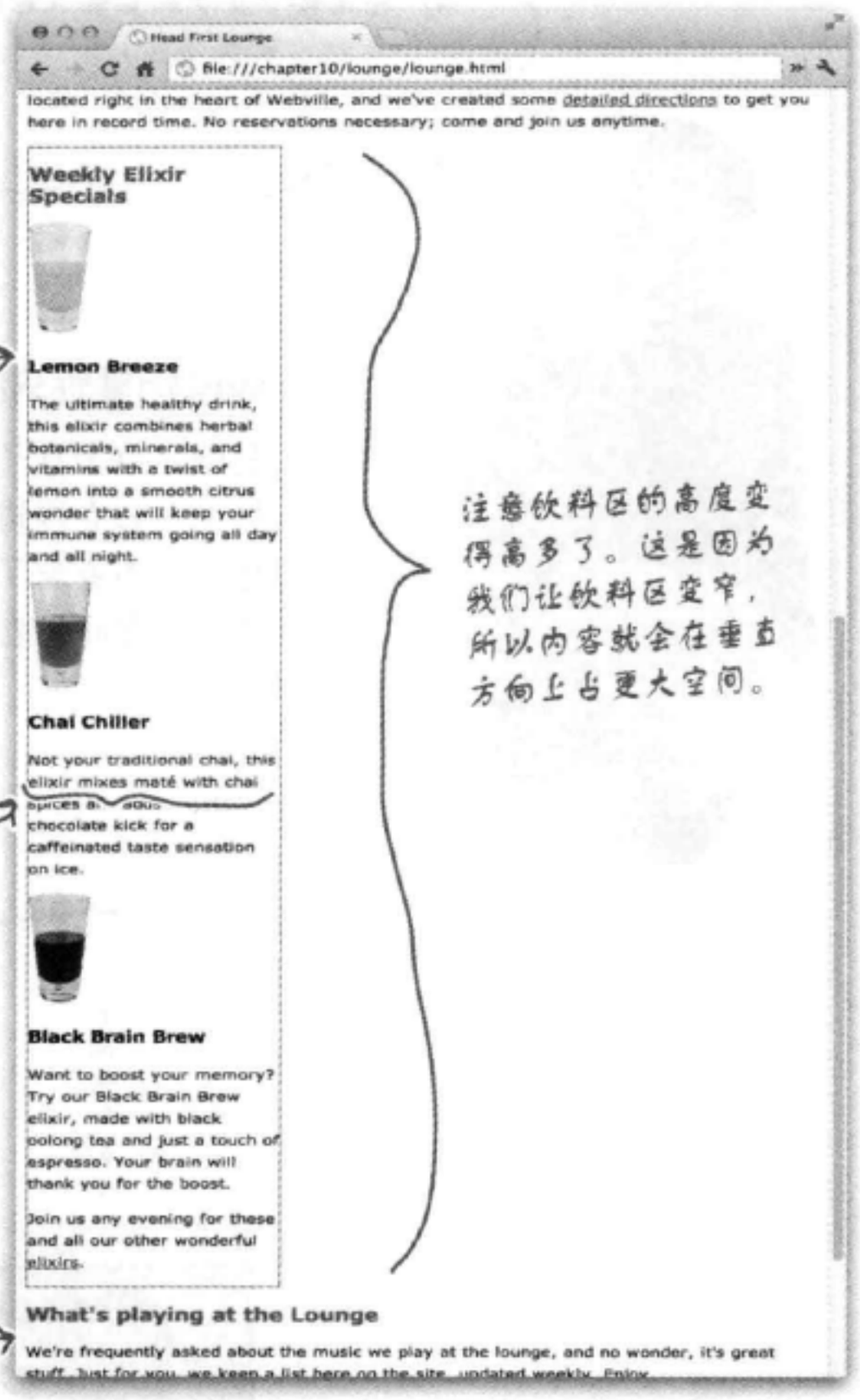
测试宽度

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……



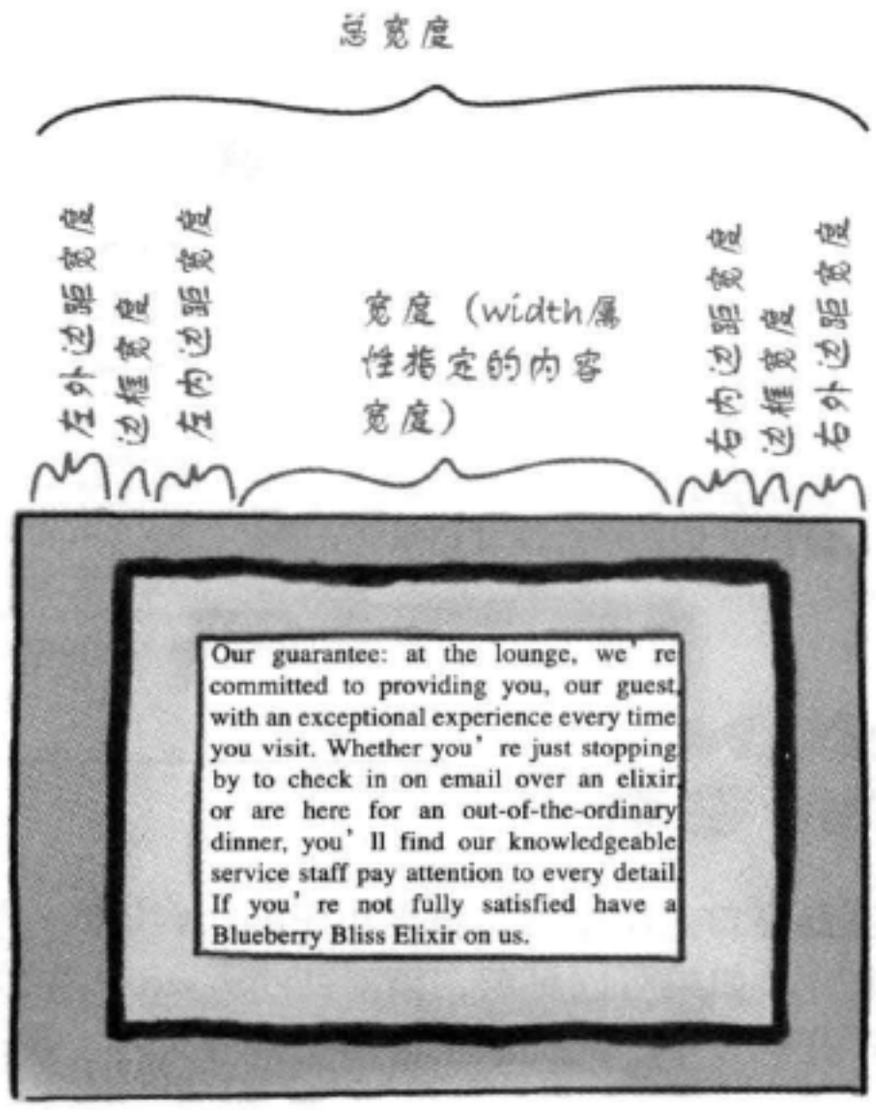
注意饮料区的高度变得高多了。这是因为我们让饮料区变窄，所以内容就会在垂直方向上占更大空间。

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。

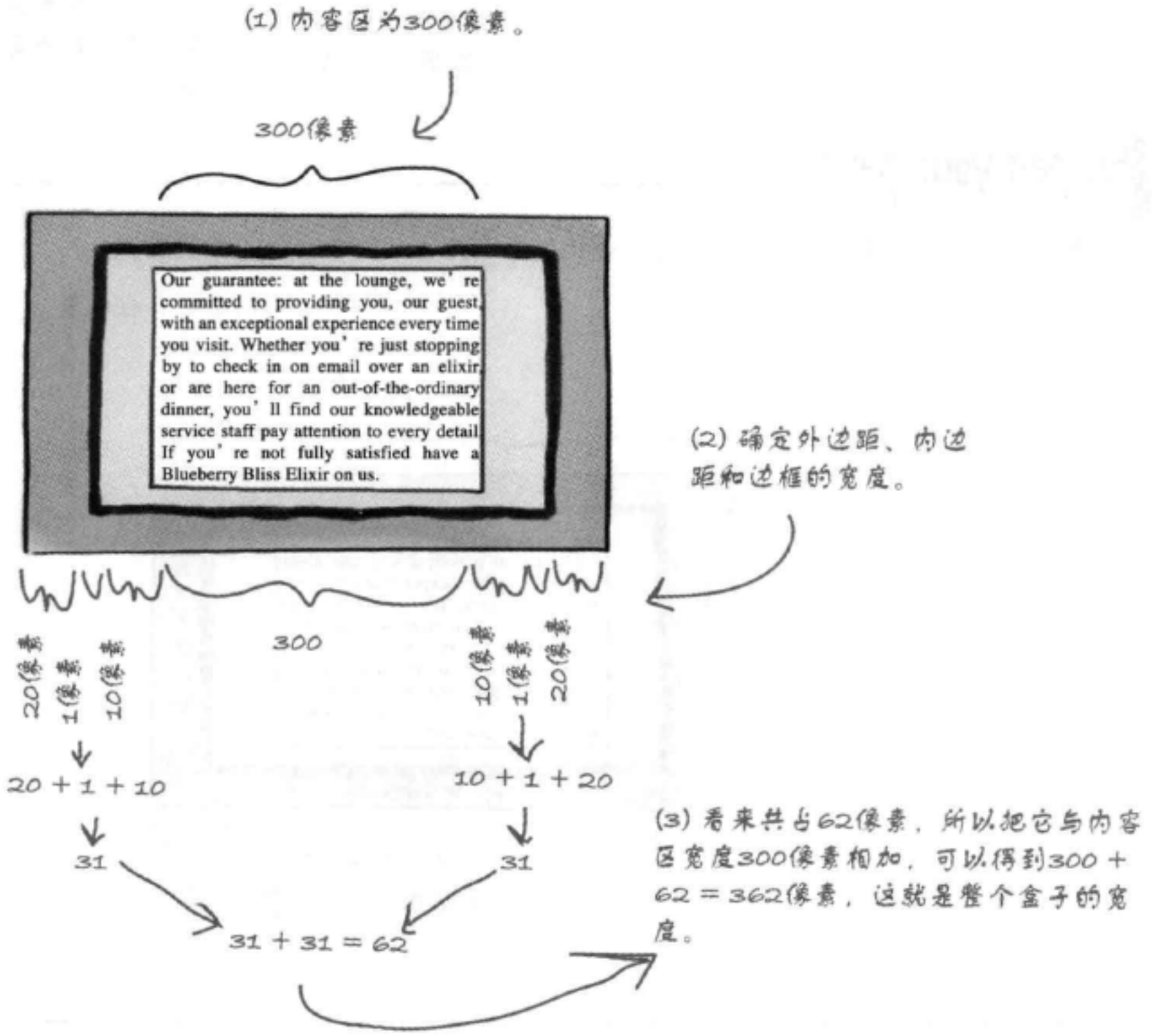
要确定整个盒子的宽度，需要将内容区的宽度加上左和右内边距、左和右外边距以及边框的宽度。不要忘了，边框宽度要加两次，因为左边和右边都有边框。



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）。

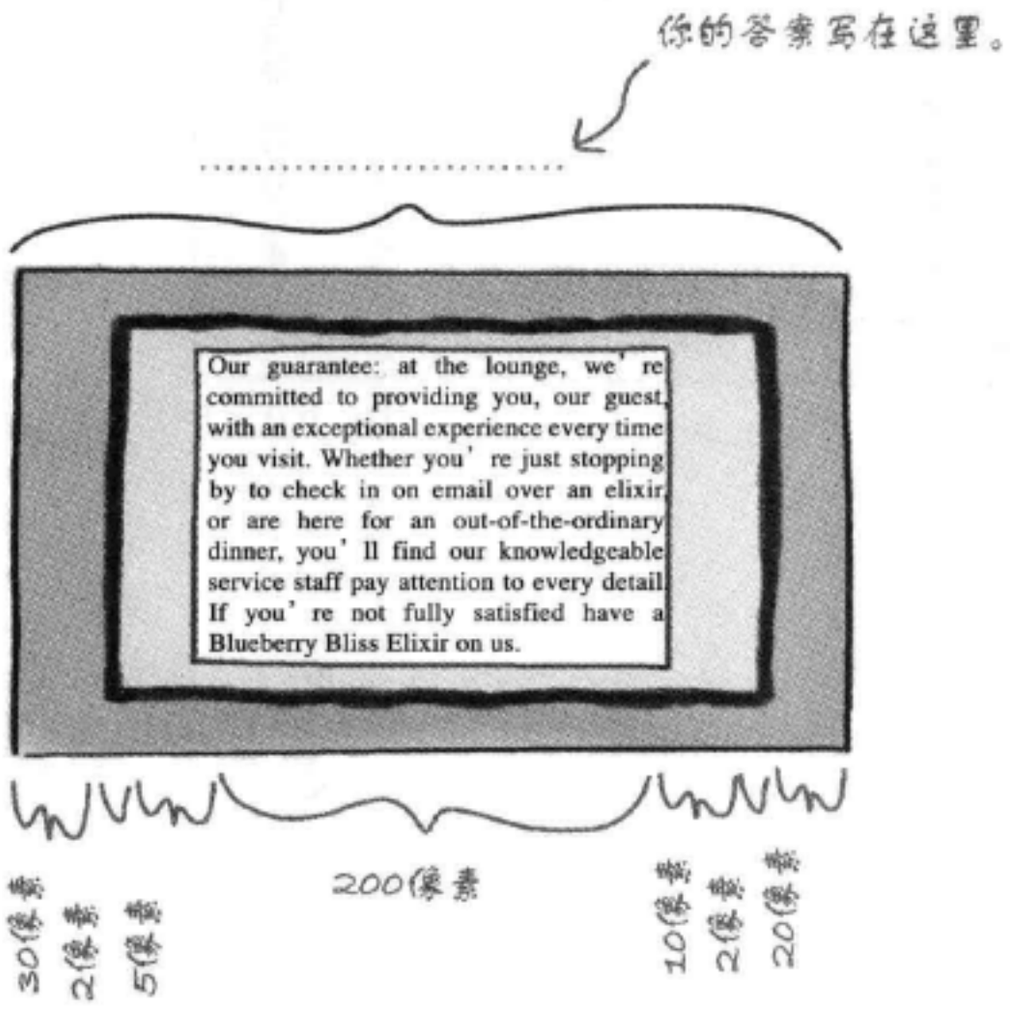
问：高度呢？

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。

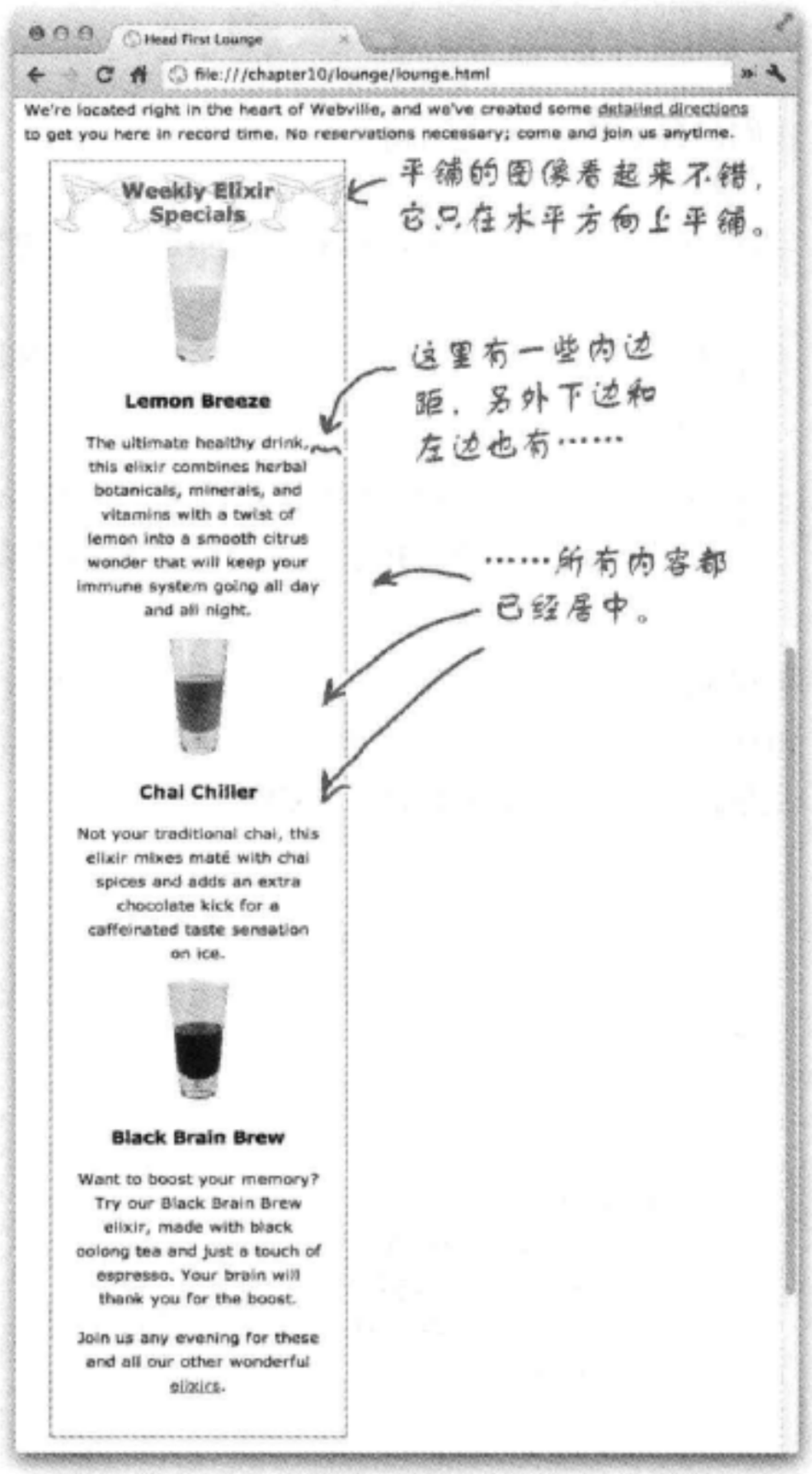
我们在左边增加了一些外边距，使elixirs相对于页面其他内容稍有些缩进。稍后会看到这很有用……

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。

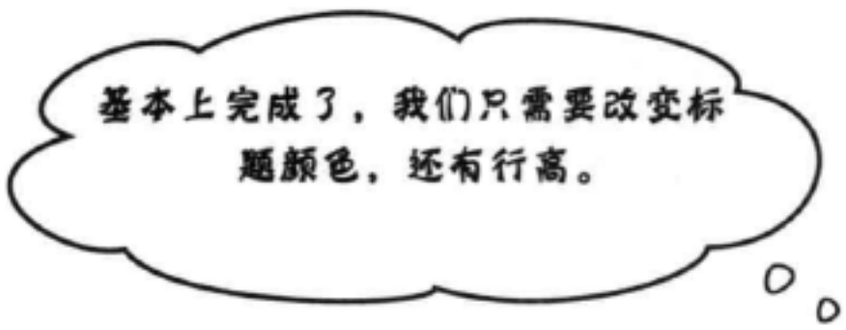
```
border-width:   thin;

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”。

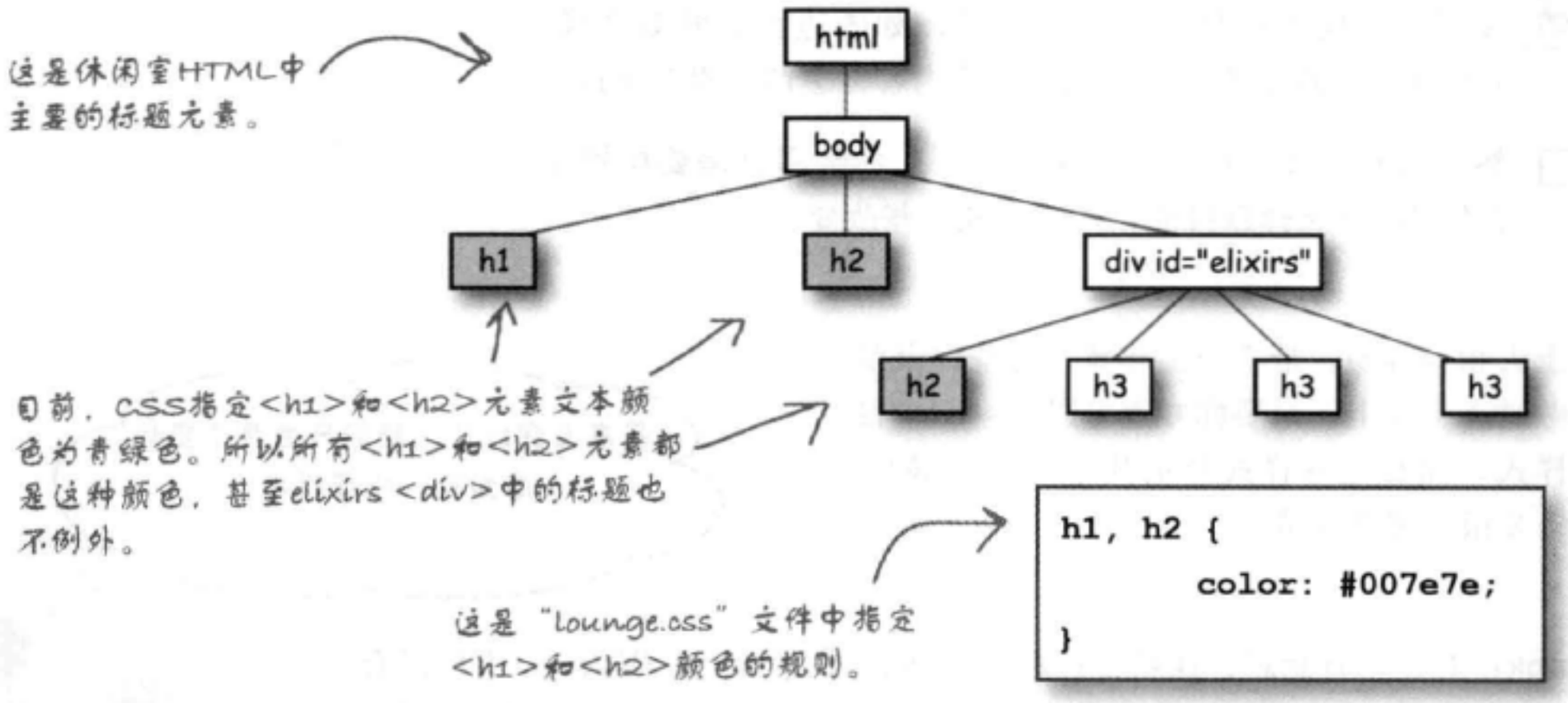
Joe：我不明白。

Frank：没关系，我们一步一步解释……

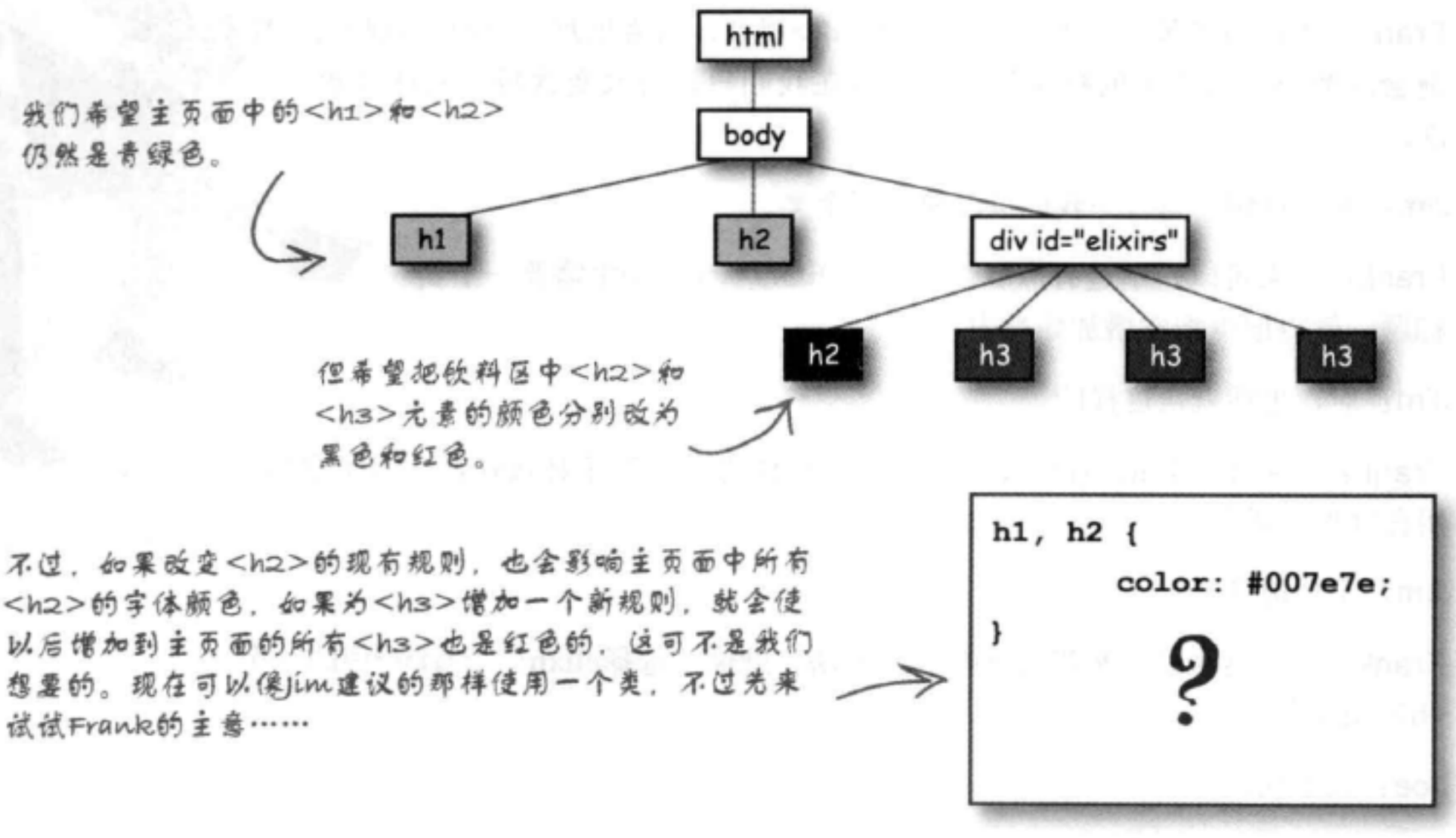
我们的目标是什么？

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

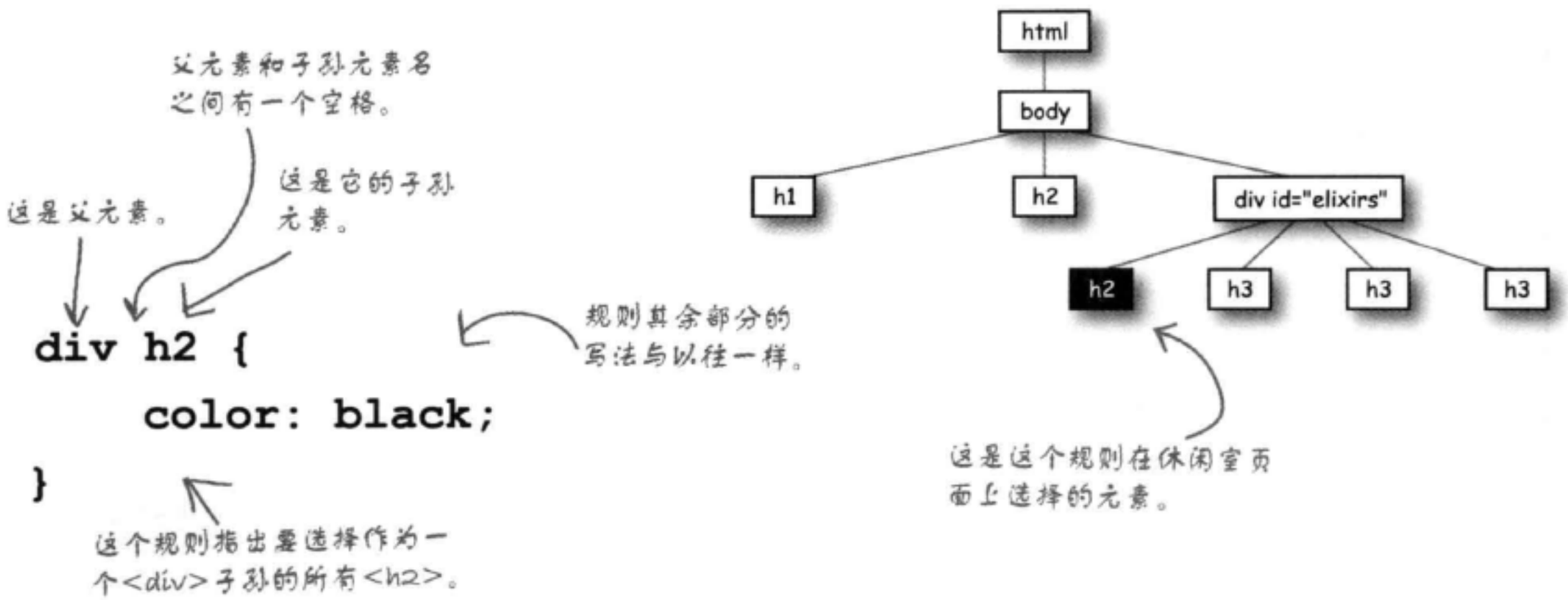


我们想要有什么

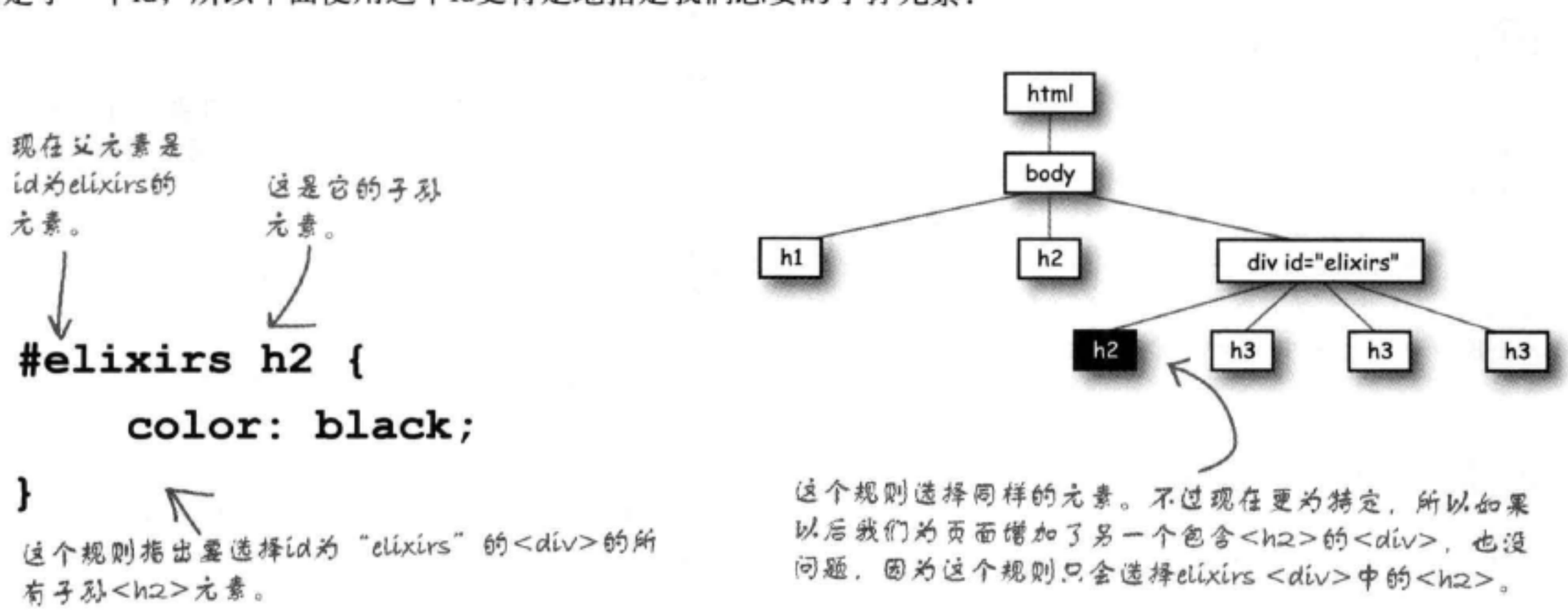


我们需要一种选择子孙的方法

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。



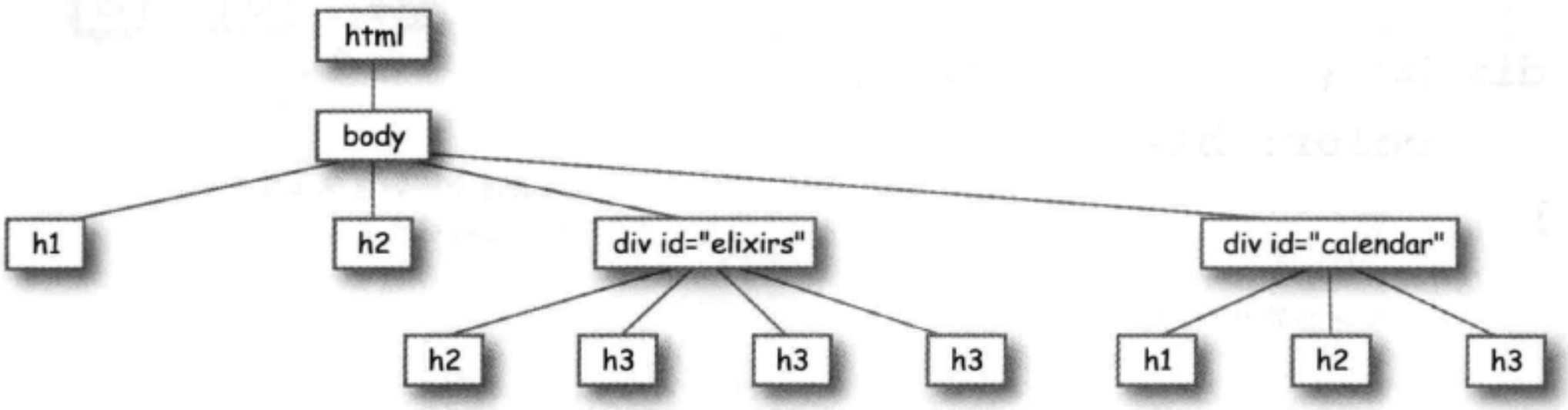
现在这个规则还有一个问题，如果有人在“lounge.html”文件中创建了另一个 ，她就会得到黑色的<h2>文本，尽管她可能并不想这样。不过，我们已经为elixirs 指定了一个id，所以下面使用这个id更特定地指定我们想要的子孙元素：



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;
}
```

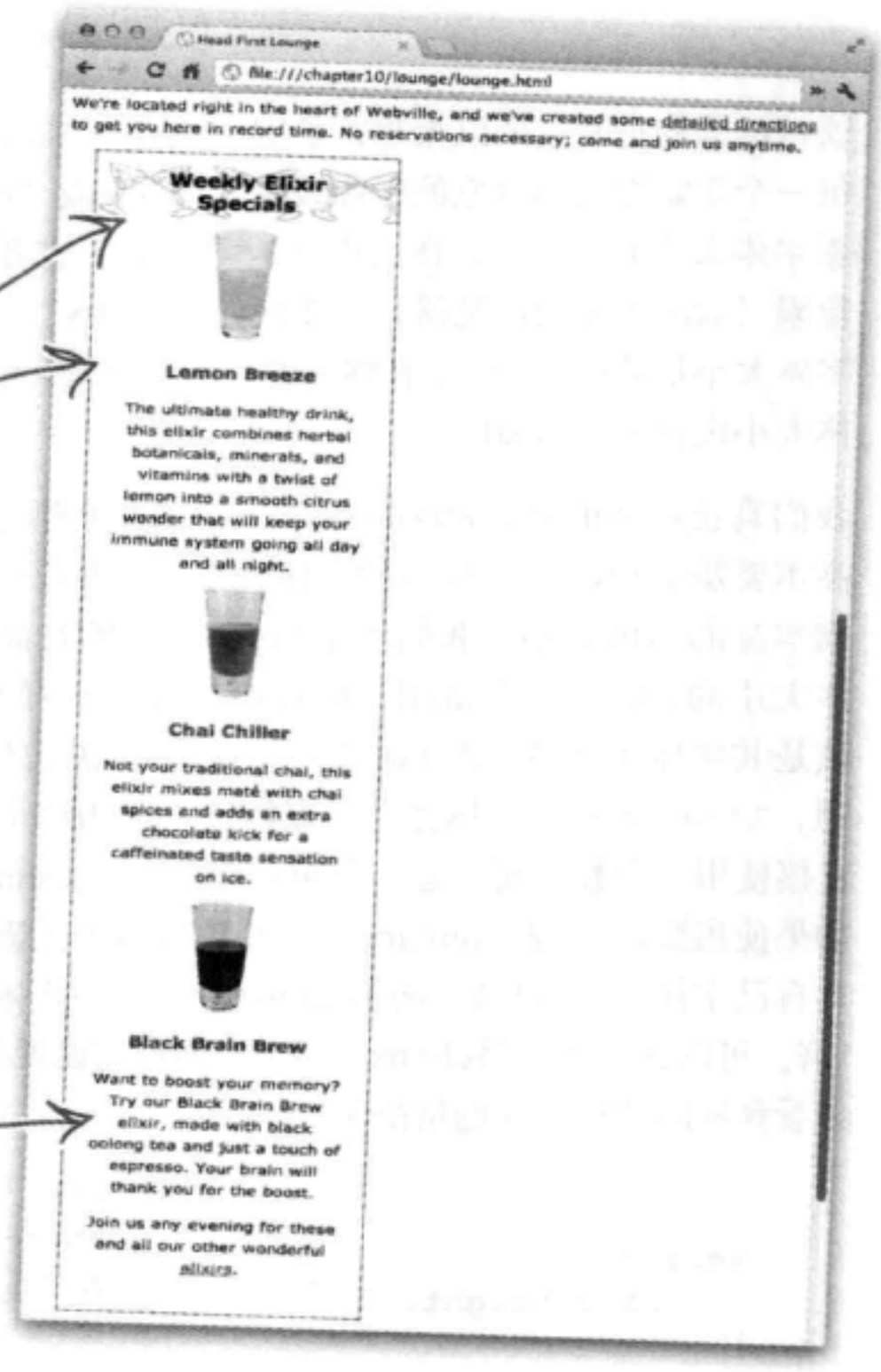
在这里使用子孙选择器来指定只选择elixirs 中的<h2>和<h3>元素。我们使用十六进制码将<h2>设置为黑色，将<h3>设置为红色。

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。

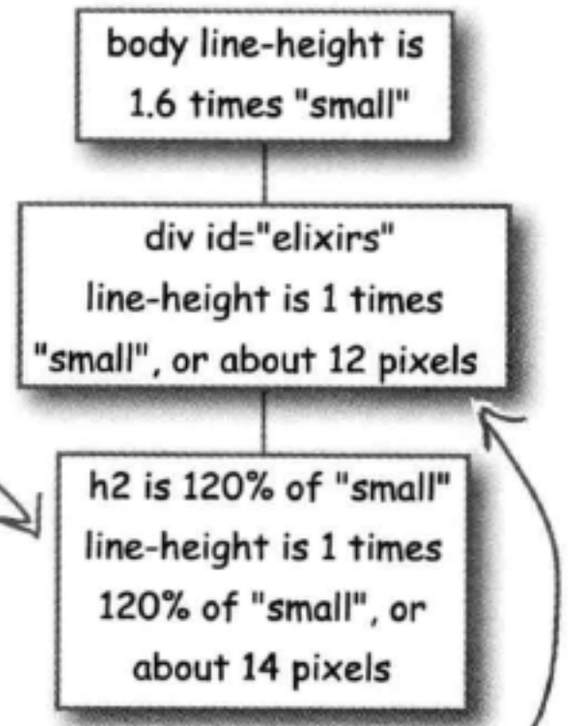
<h2>的line-height设置为elixirs字体大小的1倍，也就是“small”或大约12像素。

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%）。



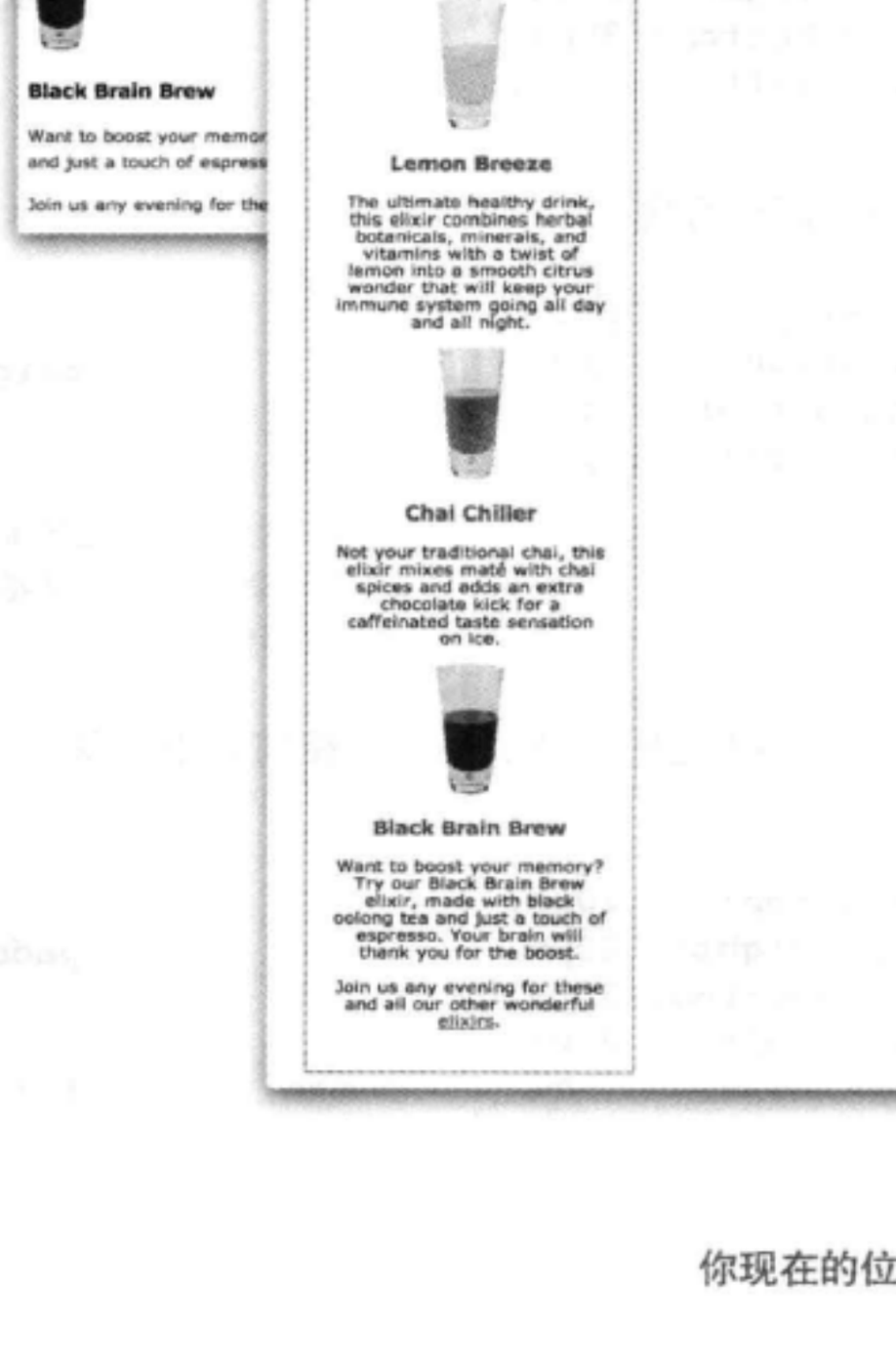
p元素的font-size是“small”（p继承了elixirs 的font-size），所以它的line-height为12像素，这正是我们想要的。

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）。

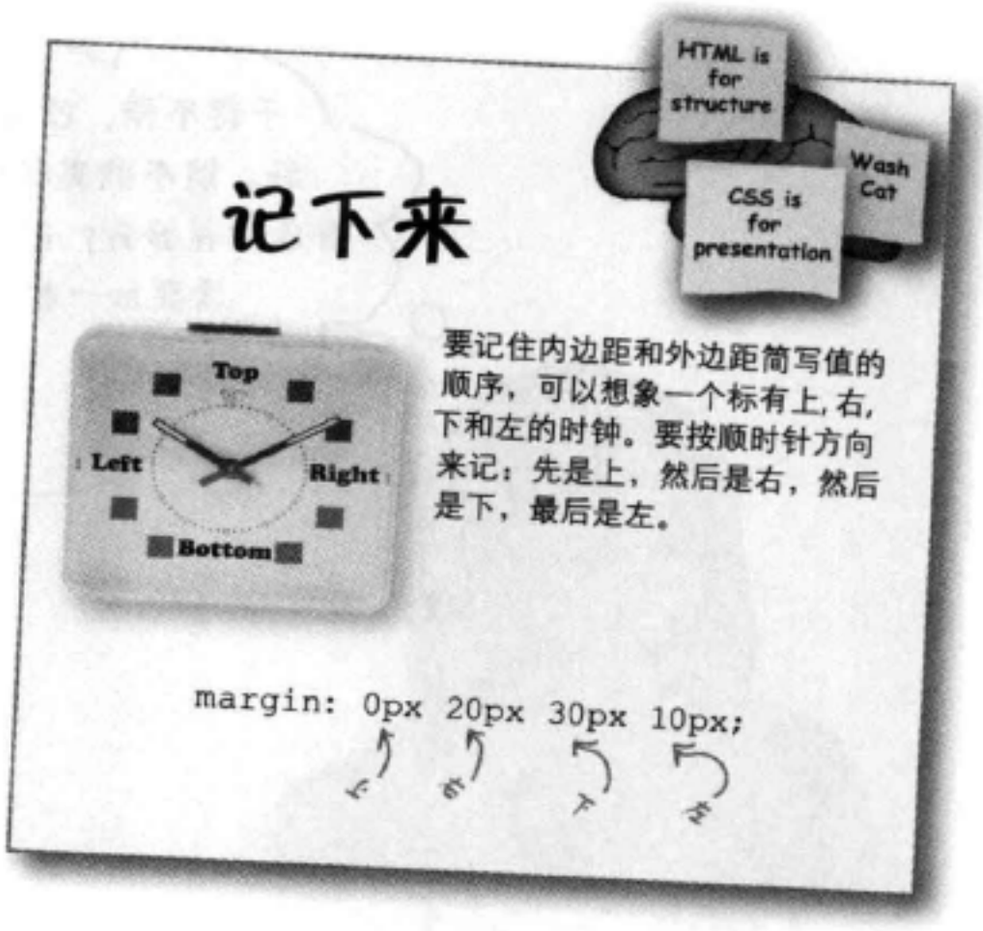
there are no Dumb Questions

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。

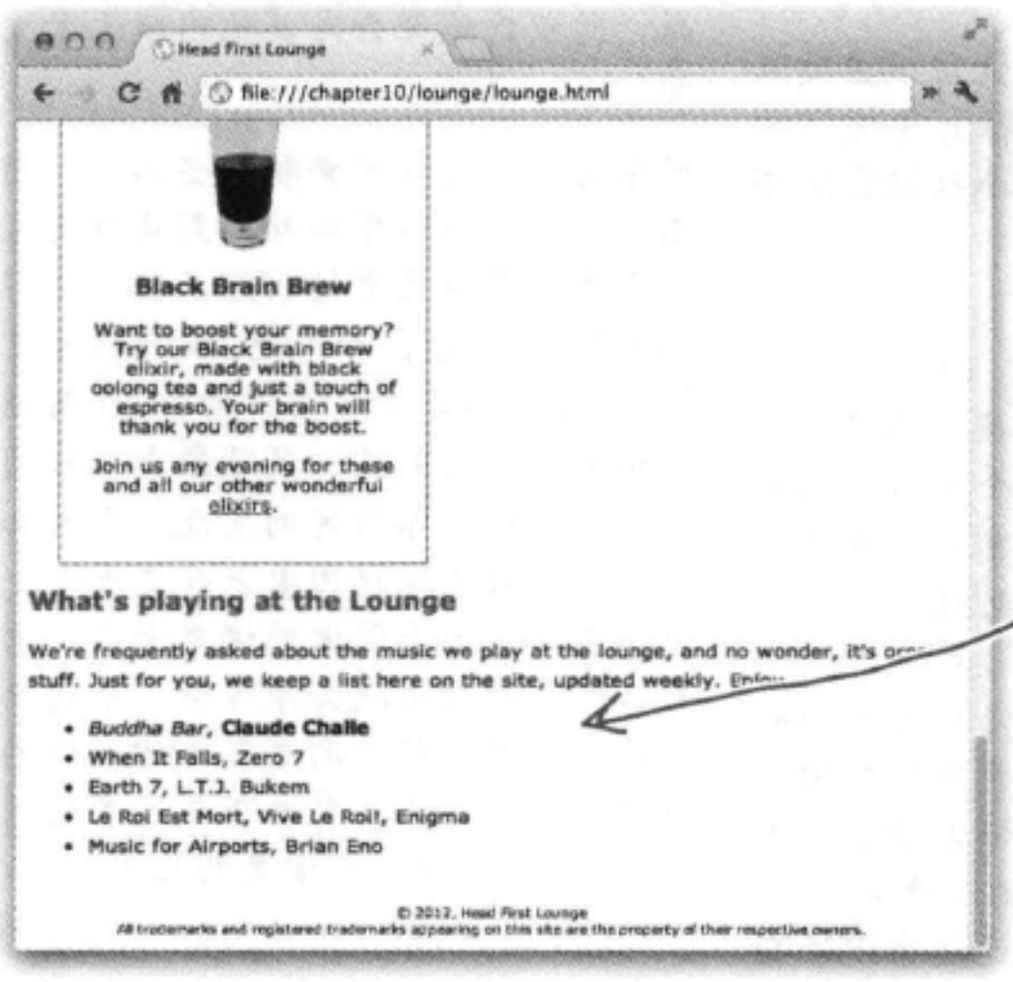
对于CD，要设置字体风格为斜体。

```
.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？

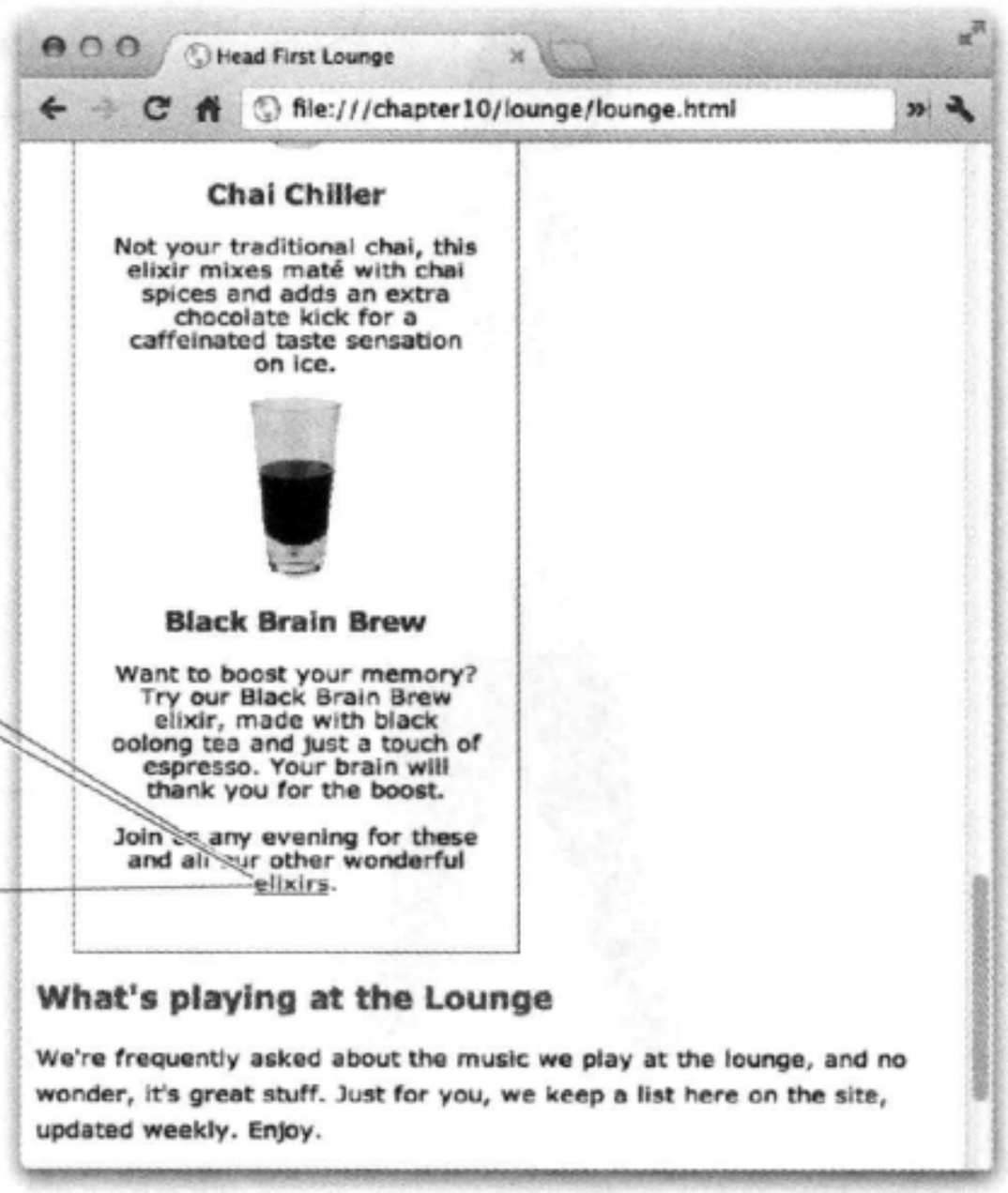
答：可以设置内联元素（如、和）的宽度，不过在对这些元素定位之前（这个内容将在下一章学习），你可能注意不到宽度改变的效果。另外还可以对这些元素增加外边距、内边距以及边框。内联元素上的外边距和内边距与块元素稍有不同，如果一个内联元素四周都增加外边距，只能看到左边和右边会增加空间。你也可以对内联元素的上边和下边增加内边距，不过这个内边距不会影响包围它的其他内联元素的间距，所以内边距会与其他内联元素重叠。

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：

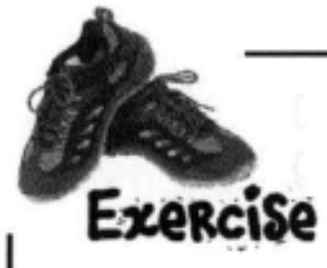
哈，这里变化很大。这里结合了一个伪类和一个子孙选择器。第一个选择器要选择嵌套在一个id为“elixirs”的元素中的所有未访问过的<a>元素。所以我们只是对elixirs中的链接指定样式。

```
#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。

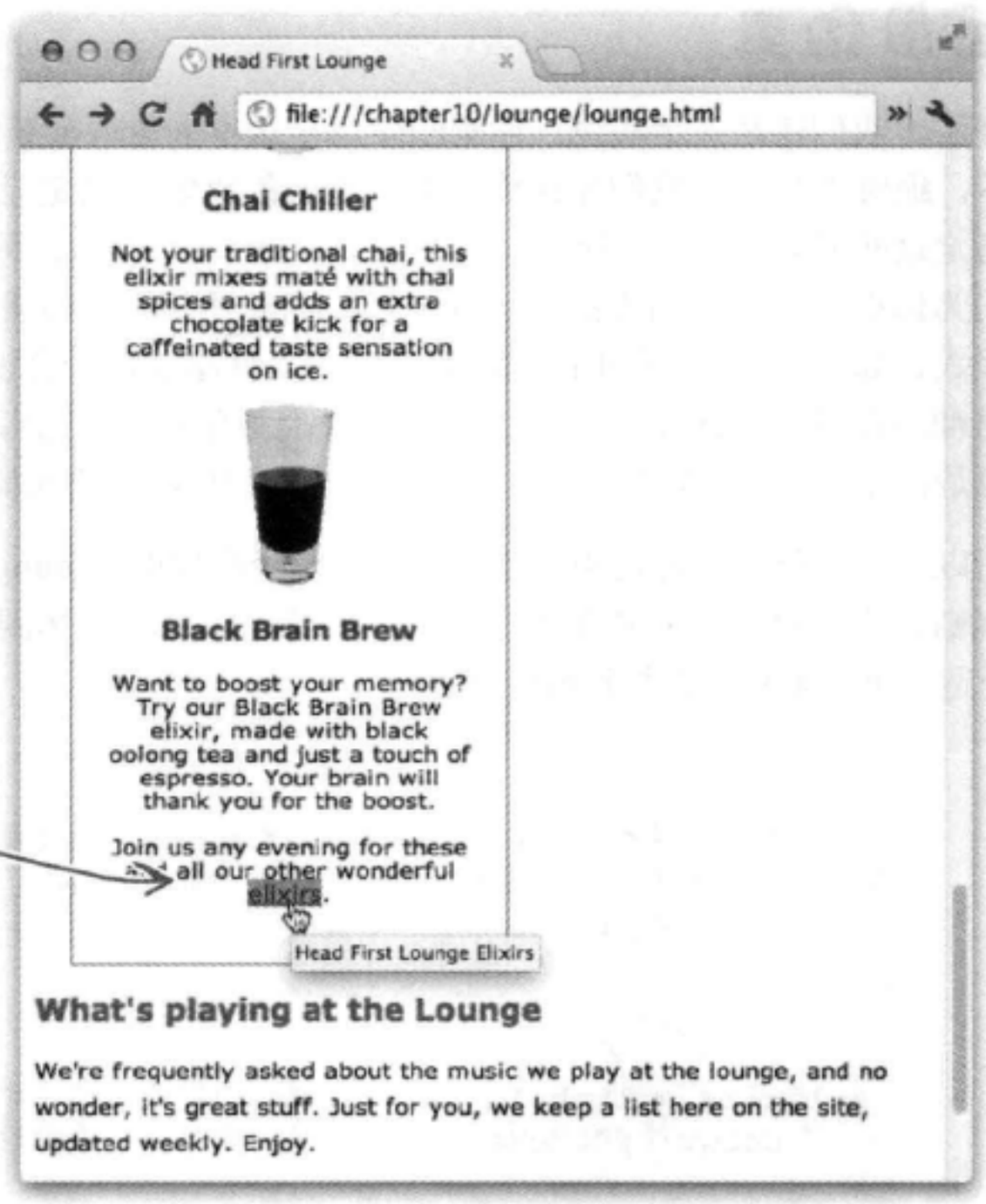
现在来看一个有意思的规则。用户悬停在链接上时，我们要把背景改为红色。这样一来，鼠标移过链接时链接会非常醒目。可以试一下！



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

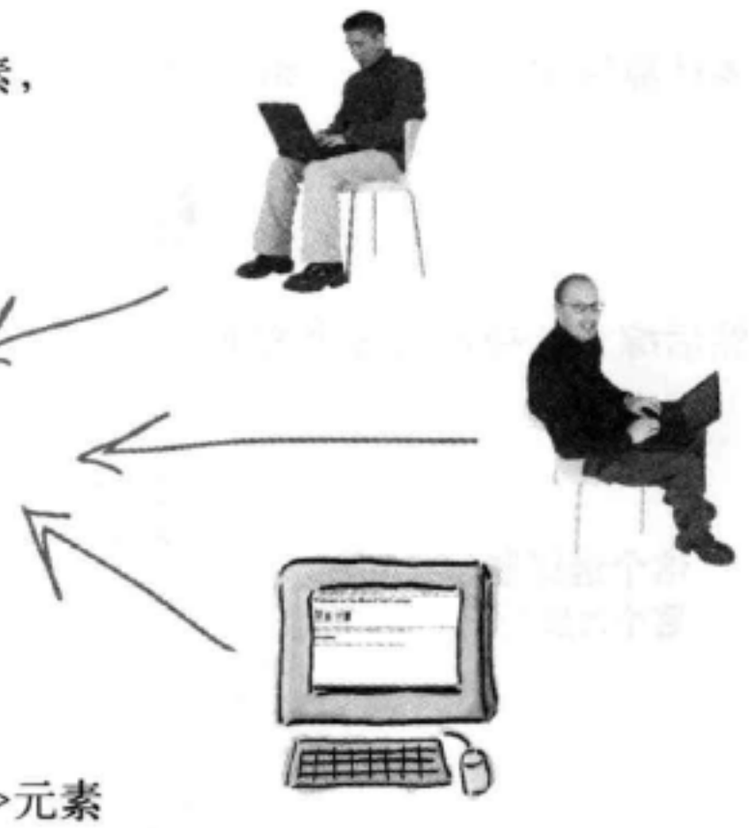
我们来复习一下，作为页面作者，我们可以对我们的HTML使用多个样式表，用户也可能会提供自己的样式，另外浏览器也有默认样式。而且不仅如此，还可以有多个选择器应用到同一个元素。那么如何确定一个元素究竟应用哪些样式呢？

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。

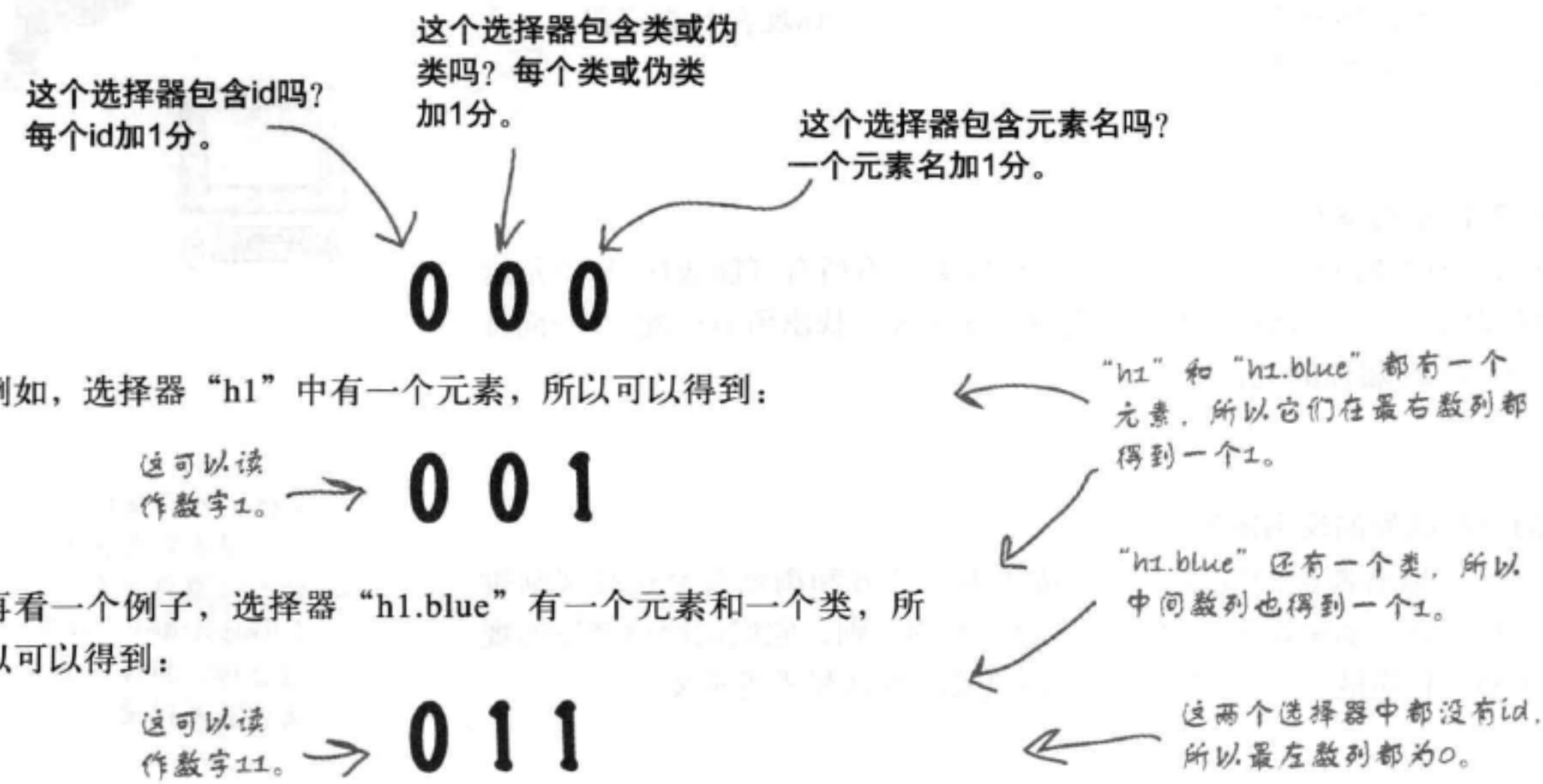
就是这样！得到的有序列表中的第一个规则就是胜出者，它的font-size属性就是你要使用的属性。现在来看如何确定一个选择器究竟有多特定。

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。

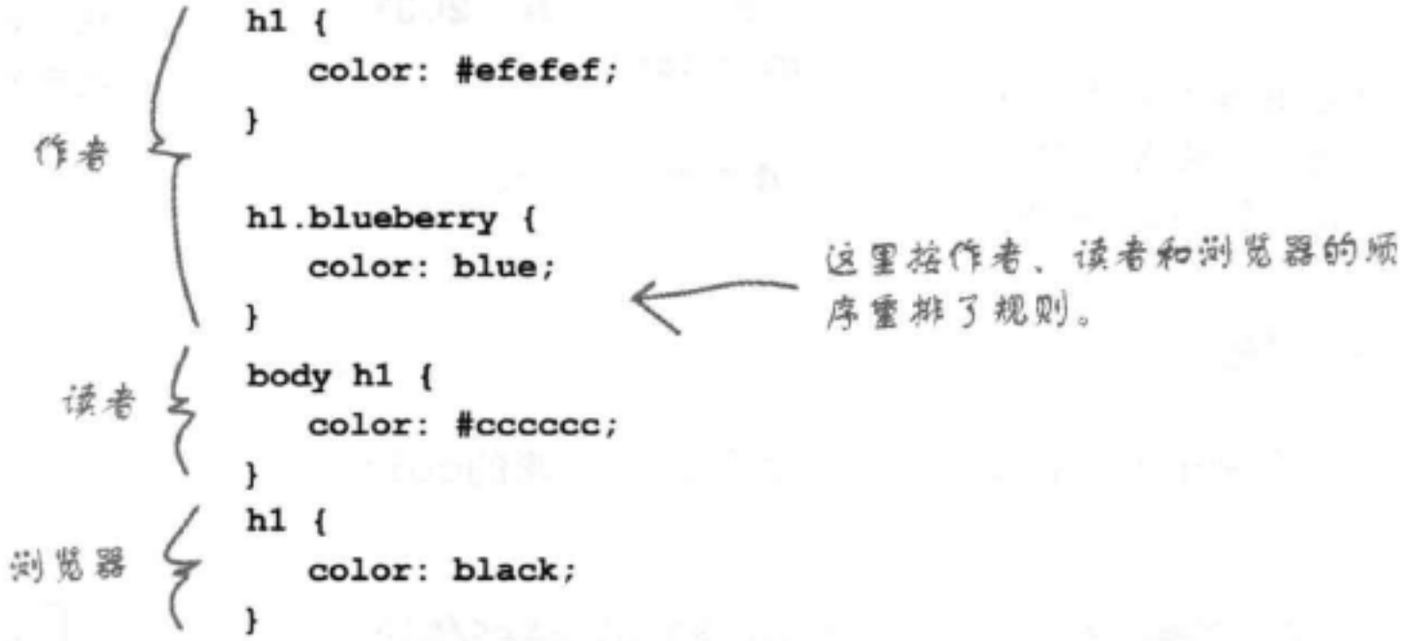
第2步：

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。

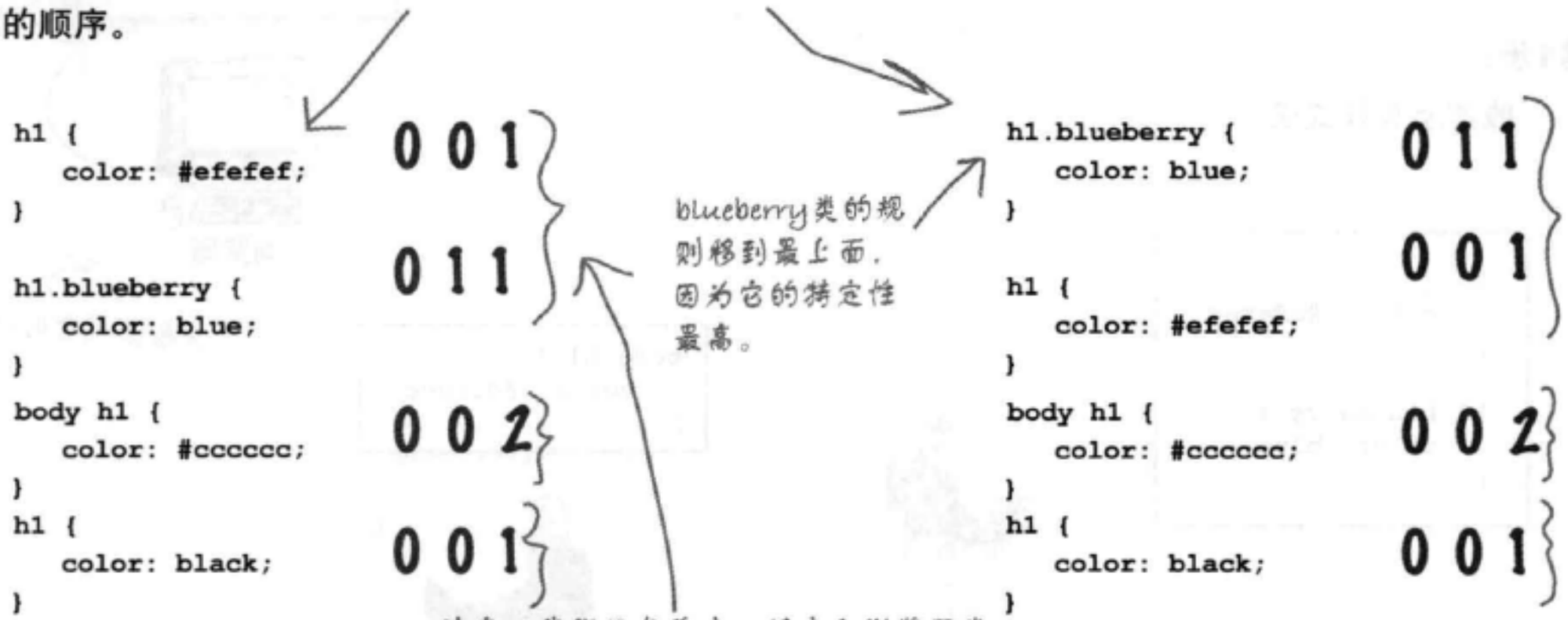
第3步：

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。

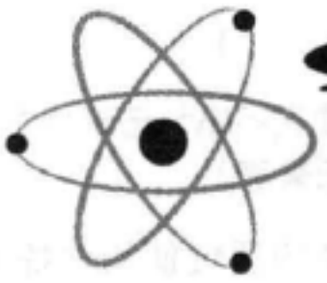
我明白了。嘿，不过如果这个属性不能继承呢？或者如果我在祖先元素的规则中也找不到这样一个值呢？那该怎么办？

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 {...}
```

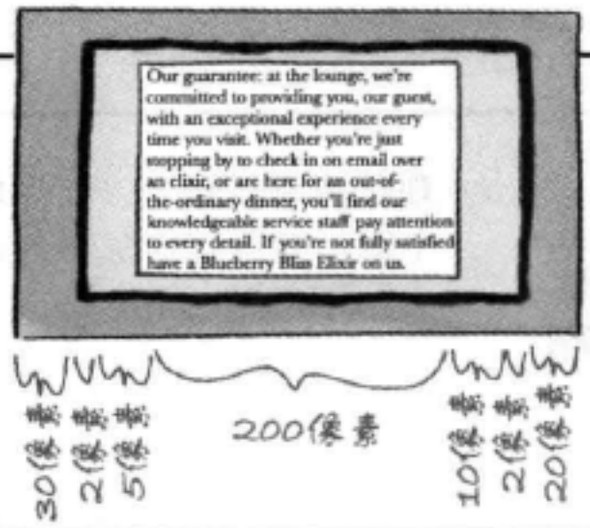
- 会选择嵌款在 元素中的所有<h2>（包括子元素、孙子元素等）。
- 可以对相关的属性使用快捷方式。例如，padding-top、padding-right、padding-bottom和padding-left都与内边距有关，可以用一个快捷规则来指定：padding。
- 内边距、外边距、边框、背景和字体属性都可以用快捷方式指定。
- 内联元素与 元素类似；它用于将相关的内联元素和文本归组在一起。
- 类似于 ，可以将元素增加到类（或者为元素指定唯一的id），对它们增加样式。
- 有些元素有不同的状态，<a>元素就是这样一个例子。<a>元素的主要状态包括未访问、已访问和悬停。
- 可以用伪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

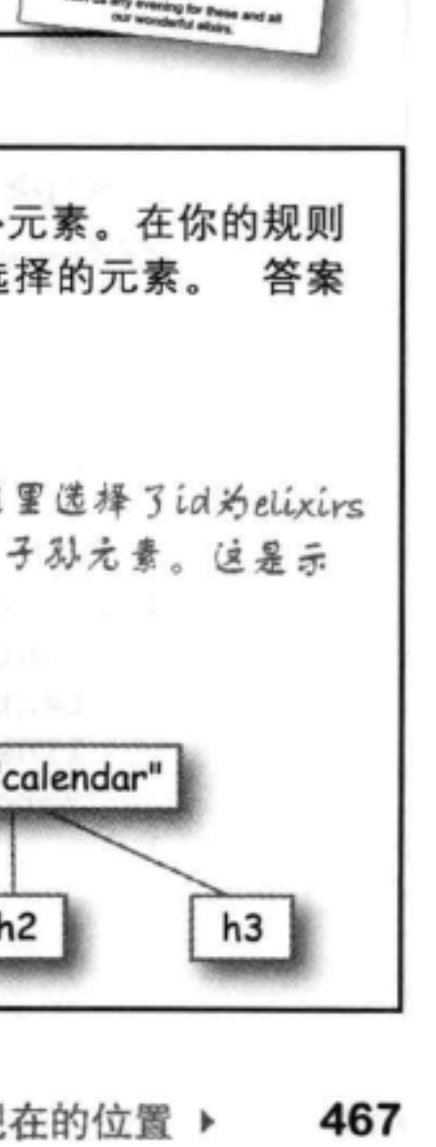
Sharpen your pencil Solution

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

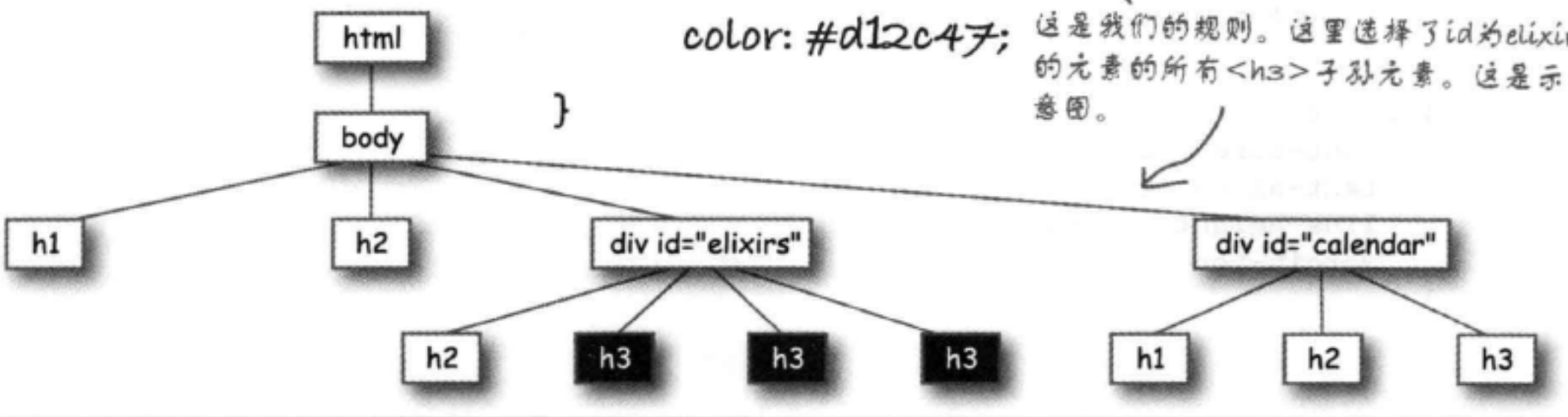


Sharpen your pencil Solution

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。

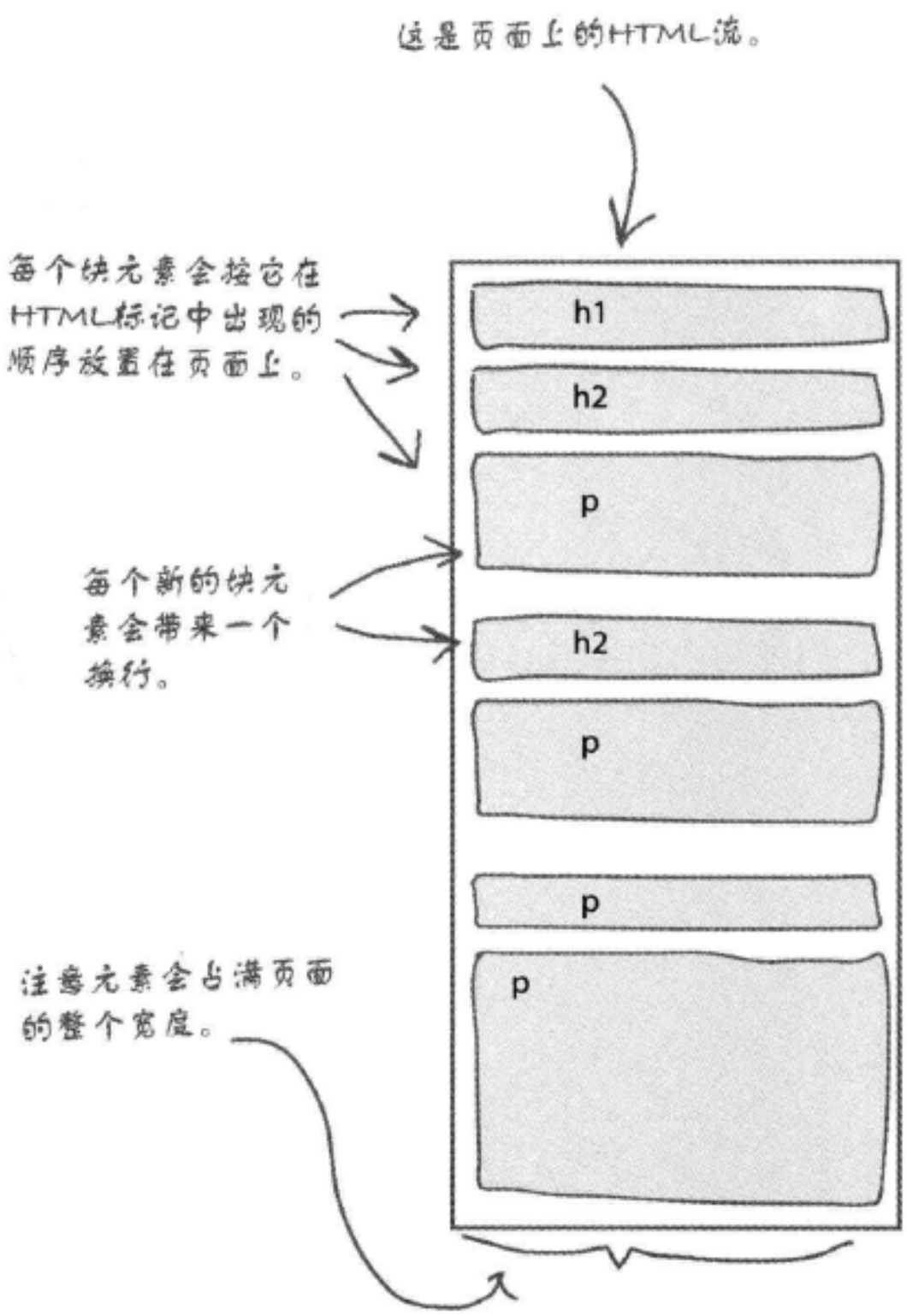
使用流

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。

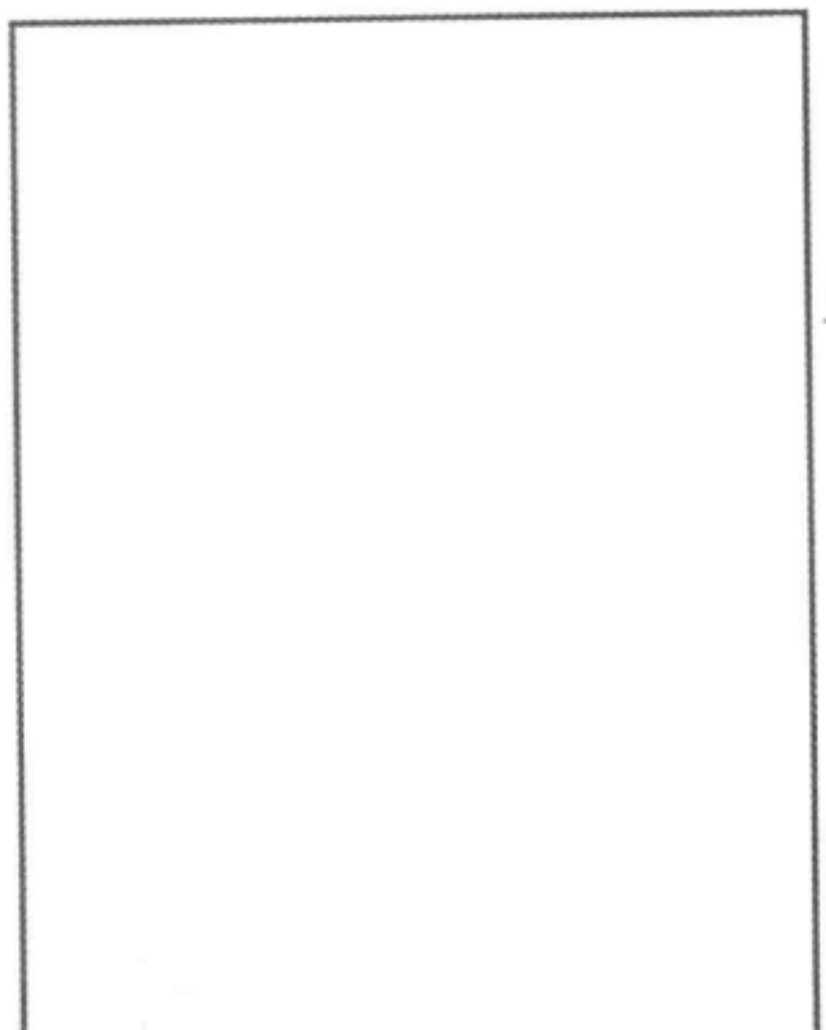
流实际上就是浏览器在页面上摆放HTML元素所用的方法。浏览器从HTML文件最上面开始，从上到下沿着元素流逐个显示所遇到的各个元素。现在先来考虑块元素，它会在每个块元素之间加一个换行。所以首先会显示文档中的第一个元素，然后是一个换行，然后是第二个元素，接下来又是一个换行，如此继续，从文件最上面一直到文件末尾逐个显示。这就是流。

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>
```



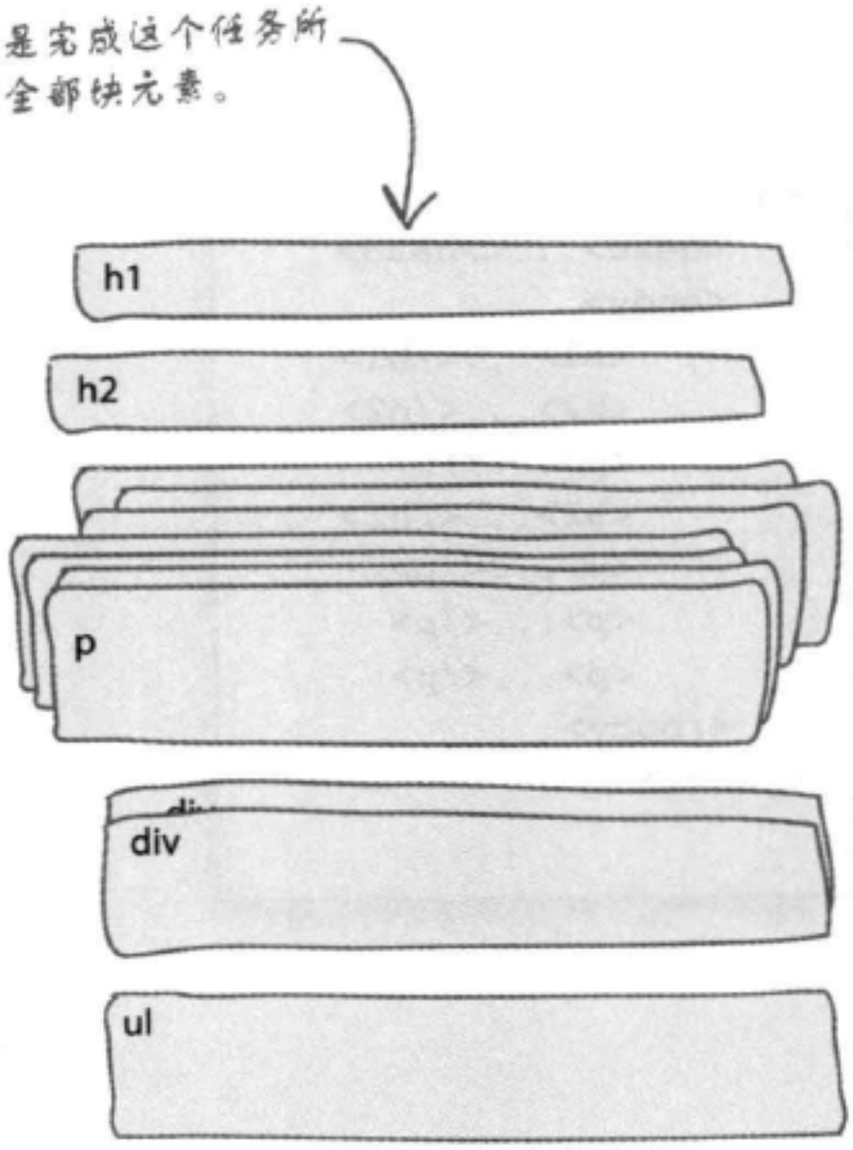
这是你的页面，在这里画出“lounge.html”中的块元素流。



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？

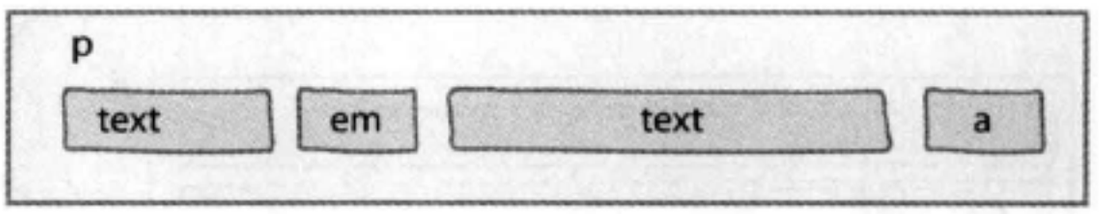
内联元素在水平方向上会相互挨着，总体上会从左上方流向右下方。具体是这样的：

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>
```

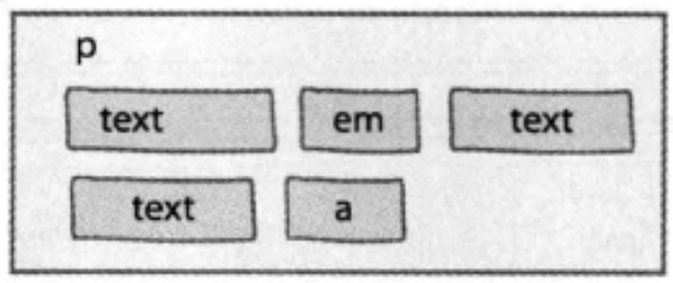
如果将这个<p>元素的内联内容流入页面，会从左上方开始。

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）。



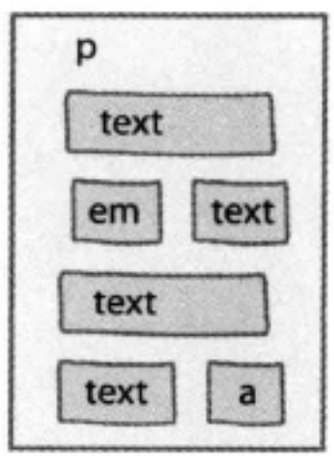
这里有足够的空间，在水平方向上可以放下所有内联元素。注意，文本是内联元素的一种特殊情况。浏览器会把它分解为适当大小的内联元素，以适应给定的空间。

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。



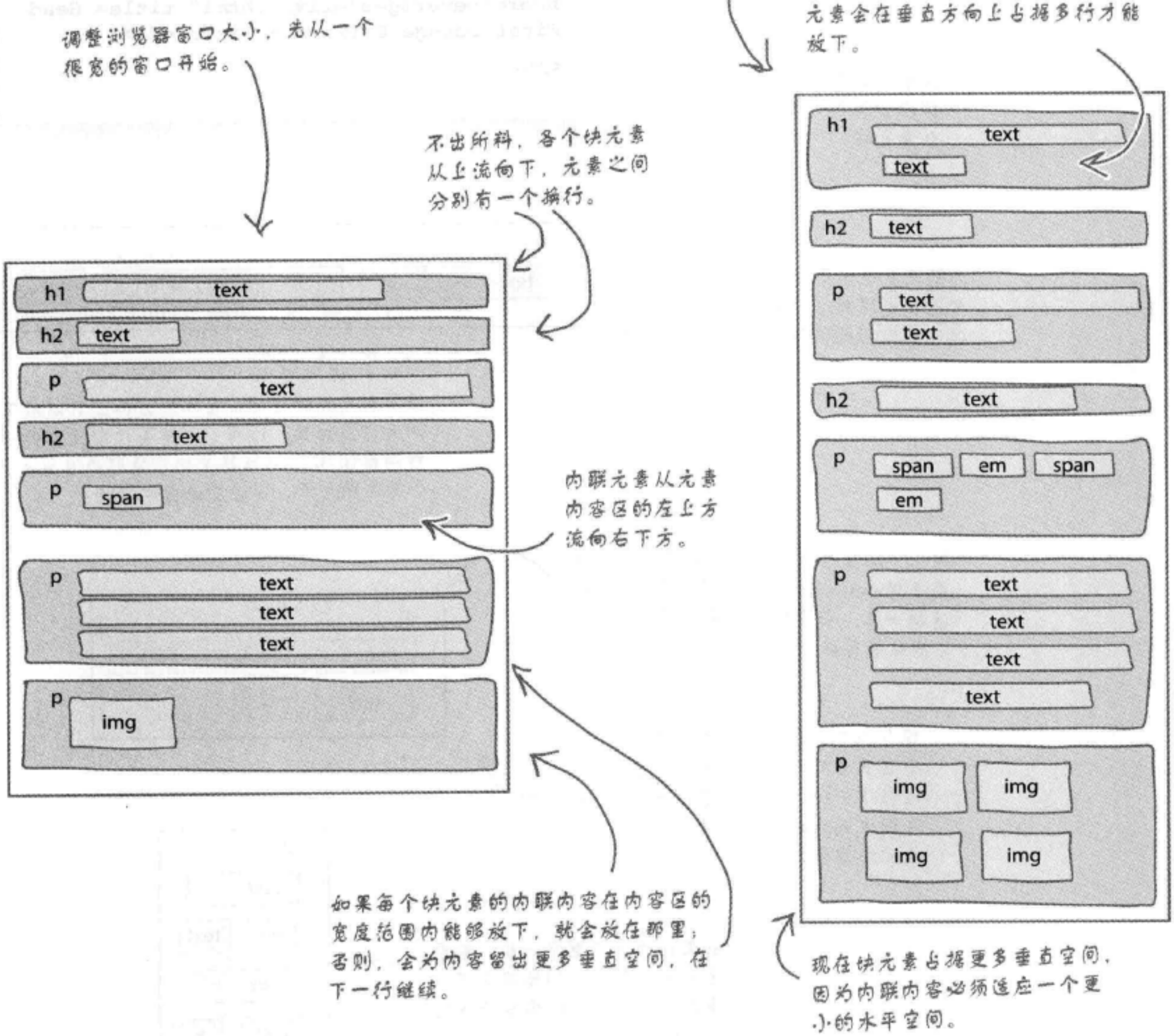
现在内容会从左向右流，直到没有更多空间为止，剩下的内容会放在下一行。注意浏览器必须用不同方式分解文本，使它能刚好适合内容区大小。

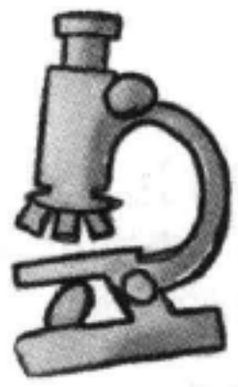
如果让内容区更窄一些，看看会有什么结果。浏览器会根据需要使用多行，将内容流入这个空间。



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。



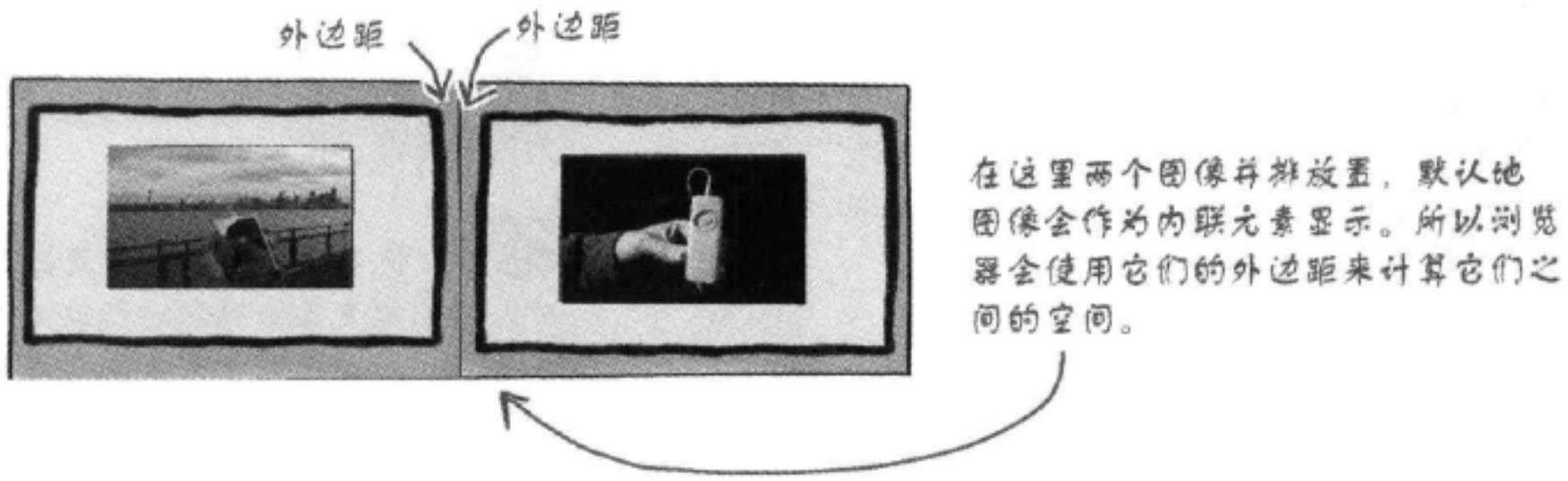


关于流和盒模型还需要知道……

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。

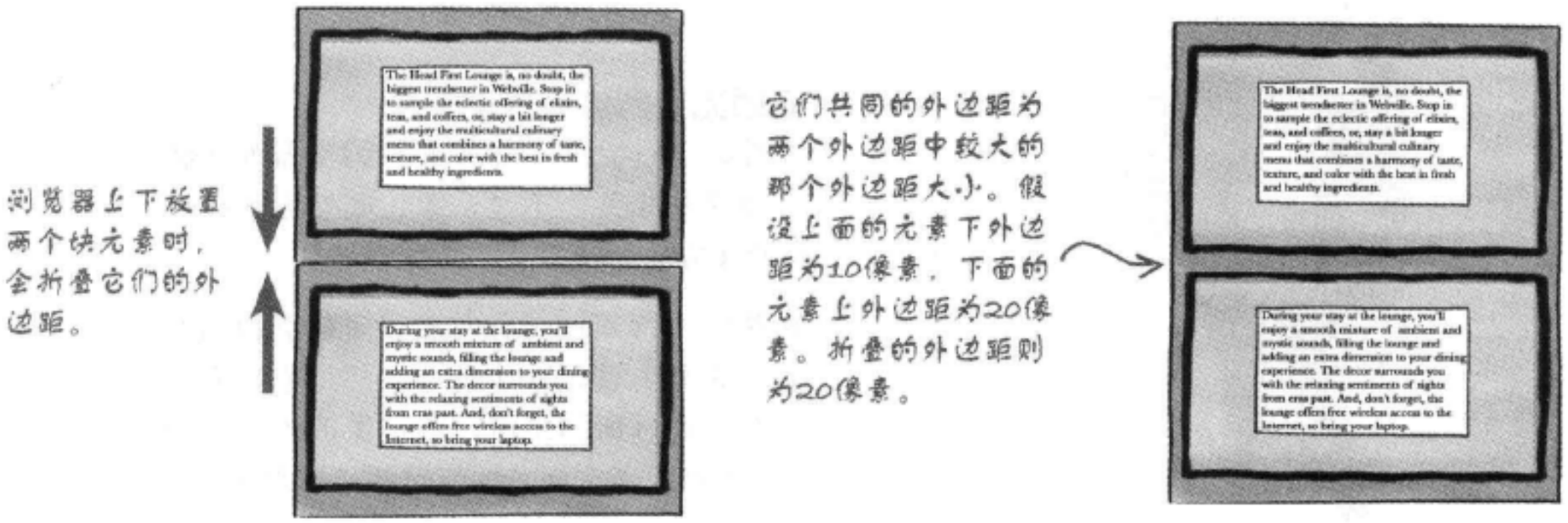
浏览器并排放置两个内联元素时……

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。

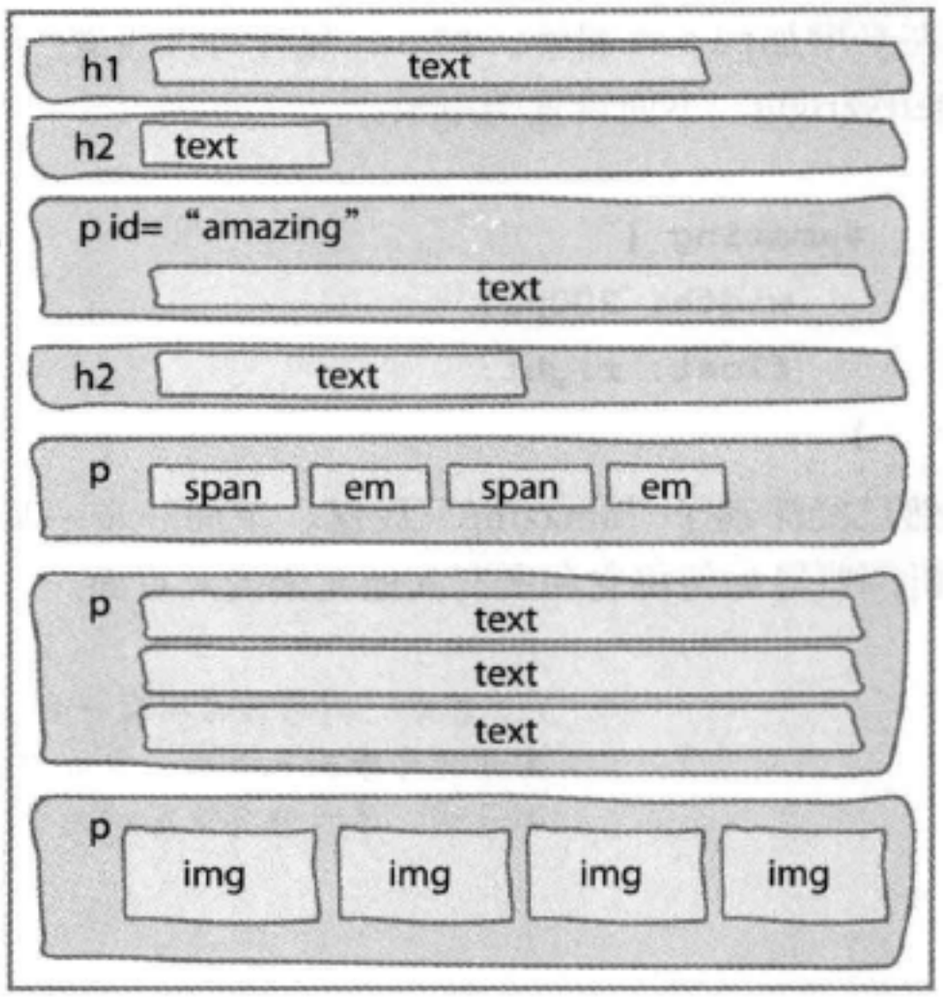
先给出一个简单的答案：float属性首先尽可能远地向左或向右（根据float的值）浮动一个元素。然后它下面的所有内容会绕流这个元素（所谓绕流，就是像流体一样绕着这个元素流动）。当然，还有更多细节问题，下面就来仔细分析……

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。

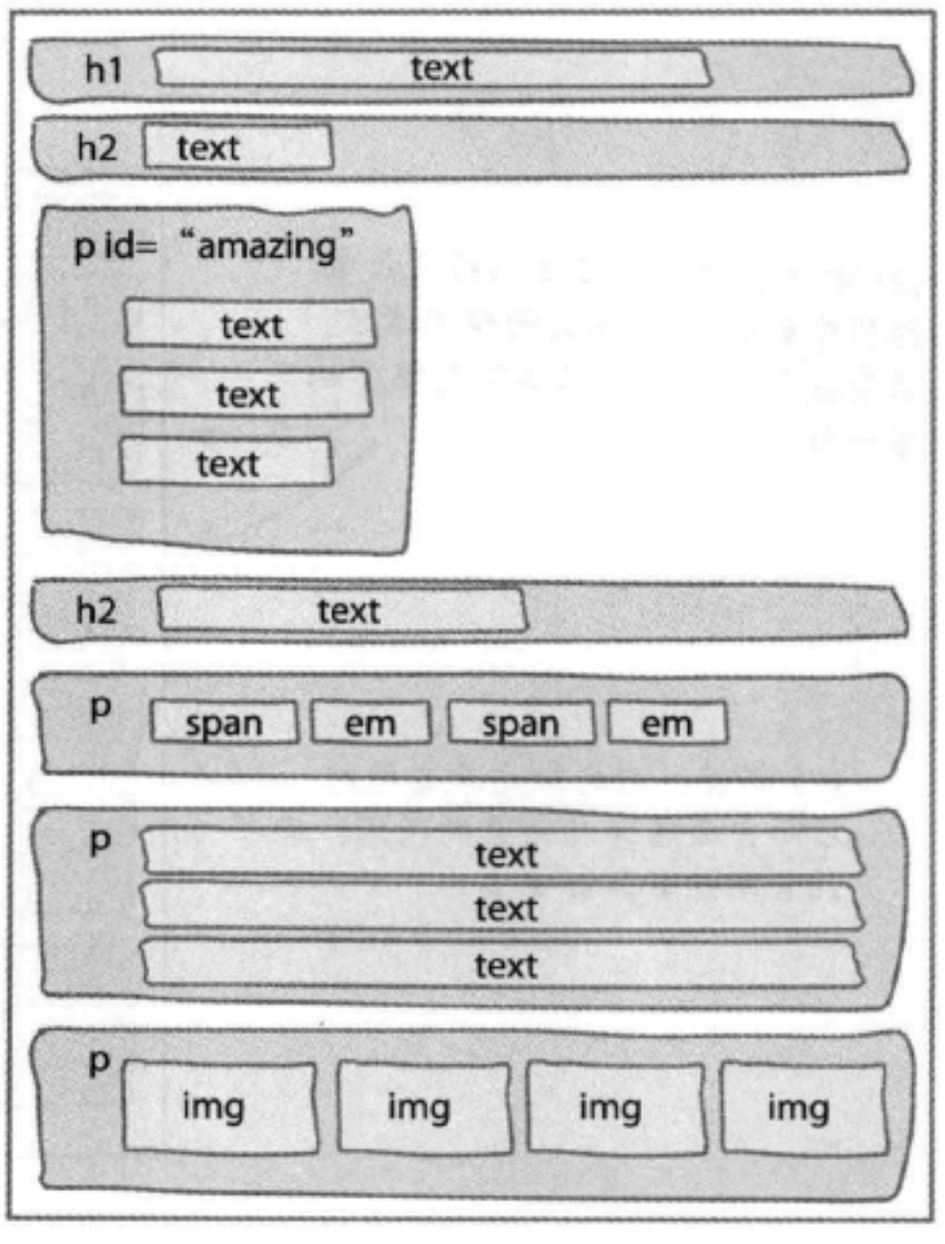


现在指定一个宽度

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。

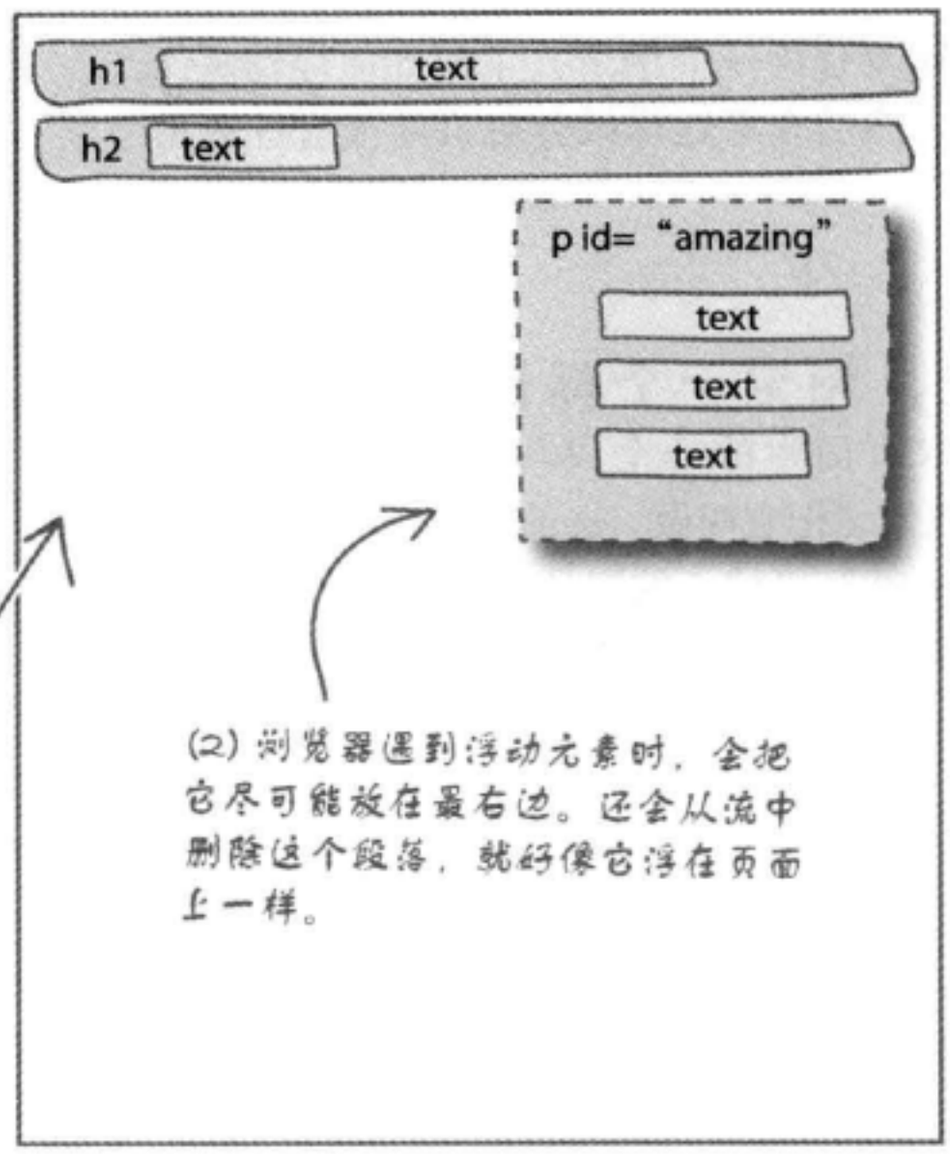


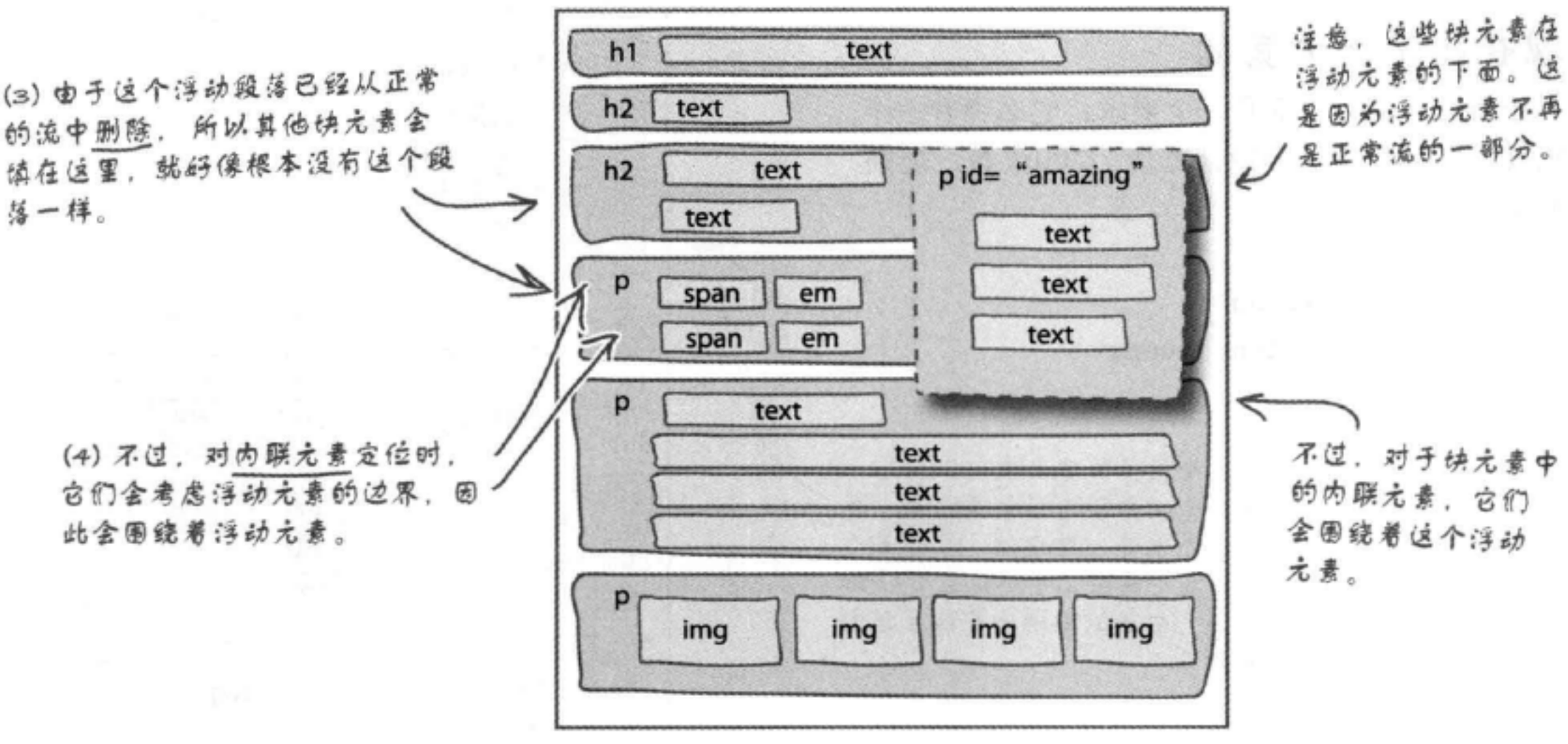
现在让它浮动

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。

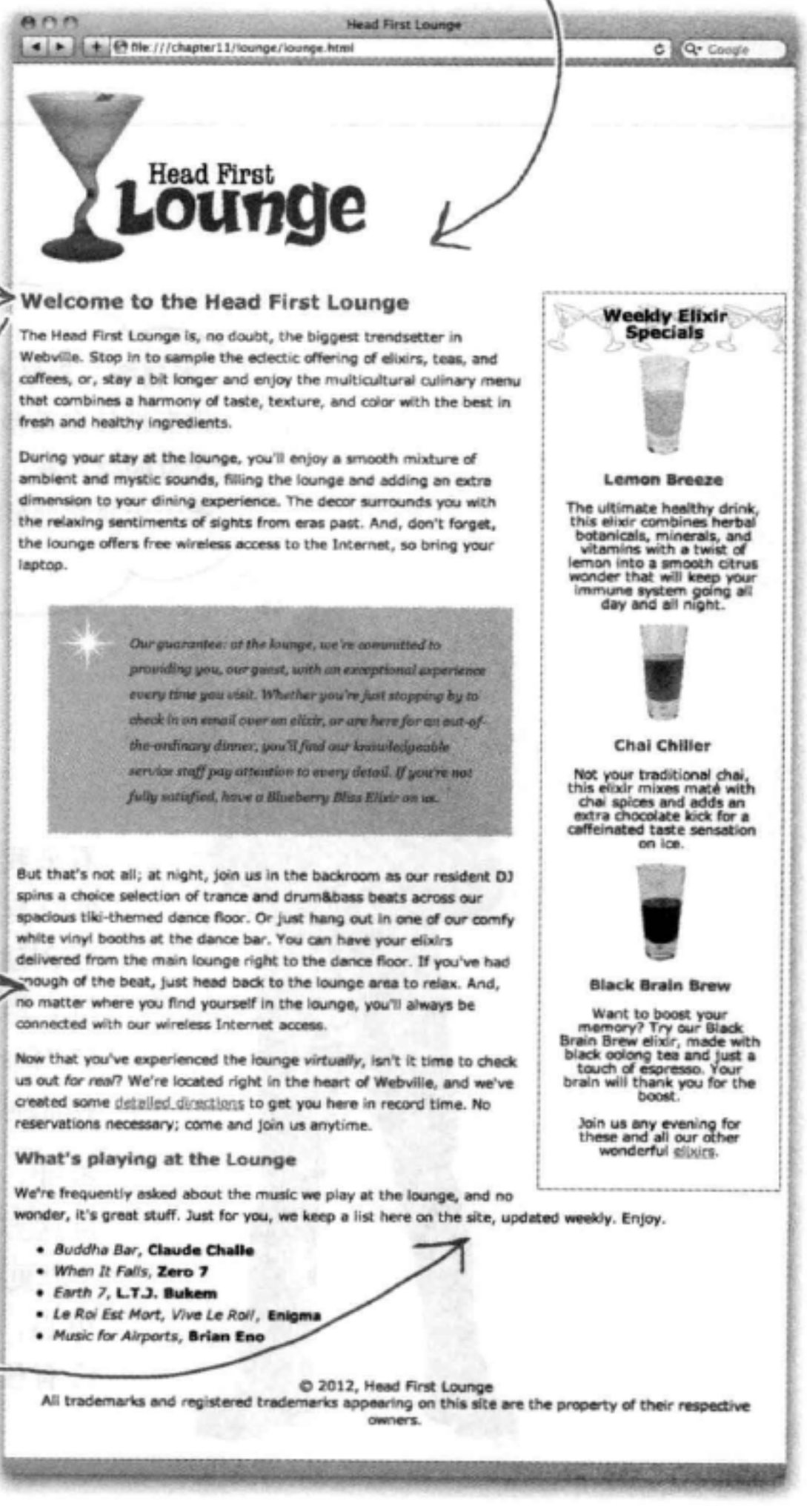
记住，除了将elixirs 设置为向右浮动，我们还把elixirs 上移，让它紧接着放在页面顶部的logo下面。

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。

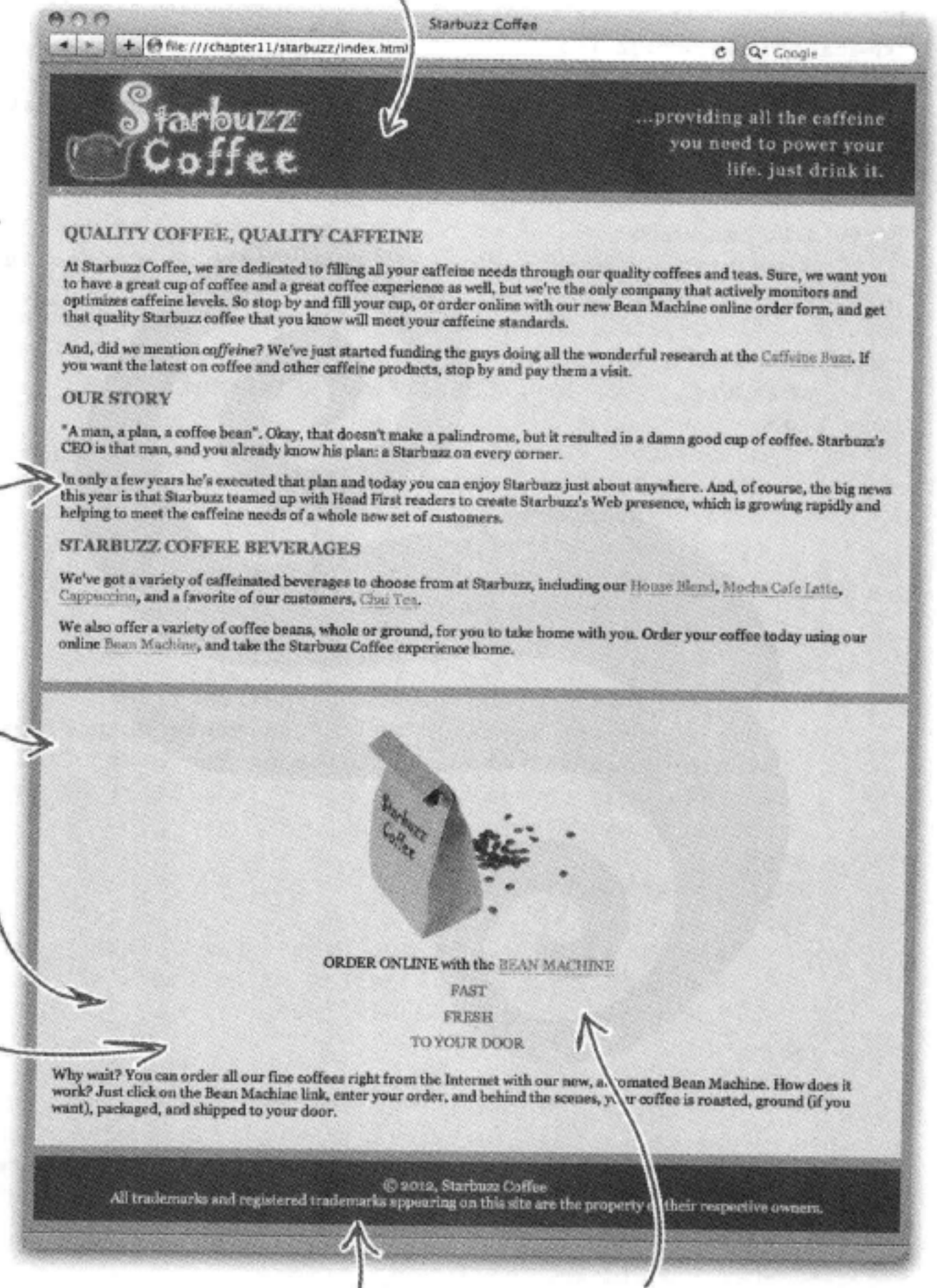
现在有一个页眉，它有一个新的漂亮的Starbuzz logo，还有公司的宗旨声明。这实际上是一个GIF图像。

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……

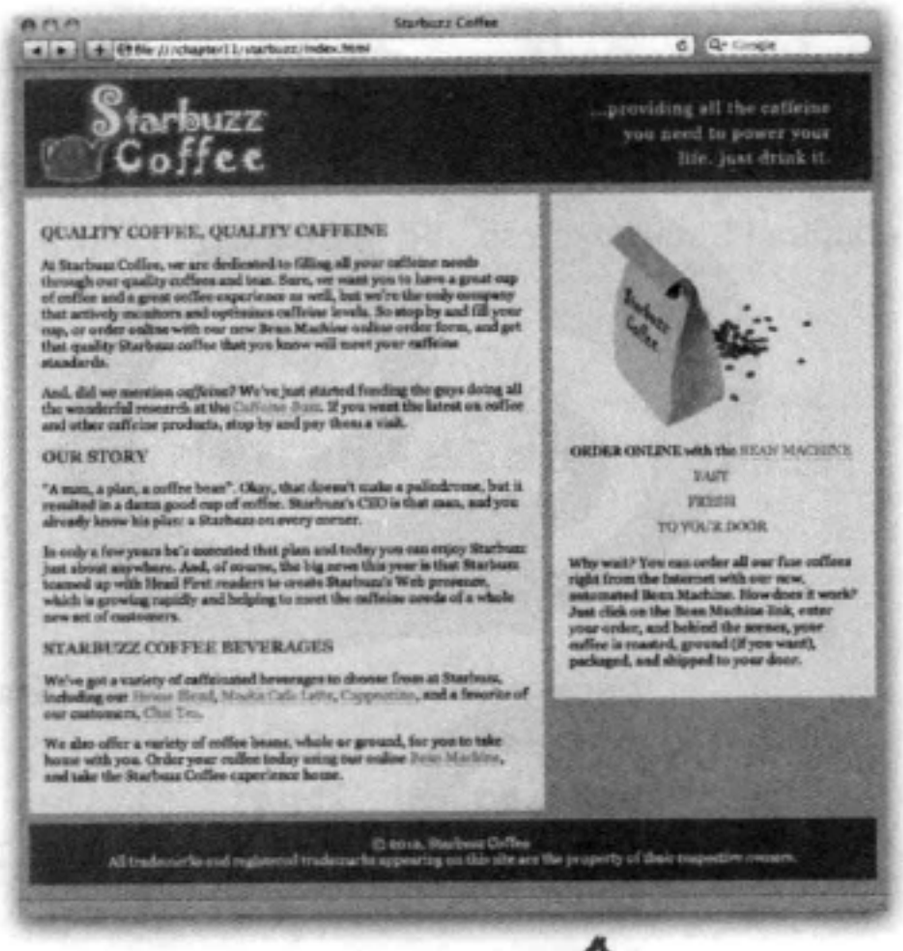
……通过使用一个虚线下边框而不是下划线，来得到漂亮的虚线下划线效果。这是对内联元素使用边框属性的一个很好的例子。

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。

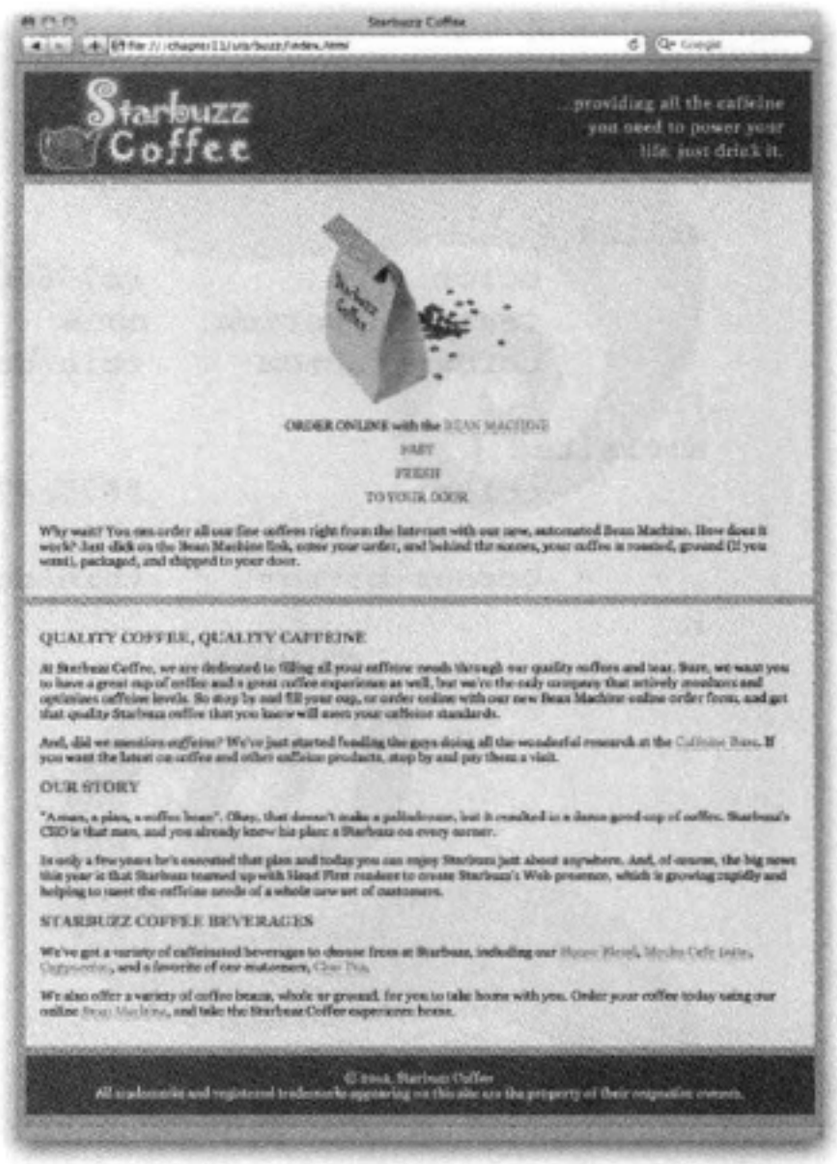


这里我们得到了一个漂亮的两栏页面，两栏明显是分开的。

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

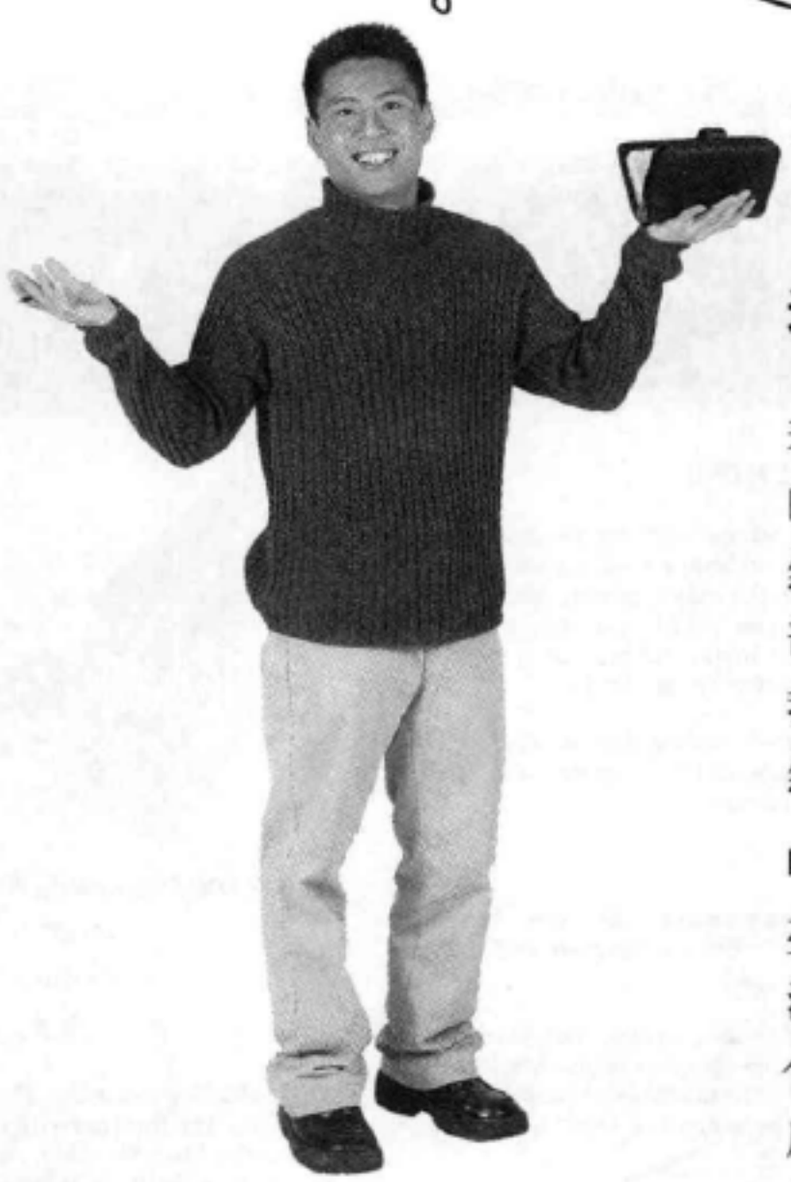
下面把这个边栏的宽度设置为280像素，为了浮动边栏，需要为“chapter11/starbuzz.css”增加float属性，如下：

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。

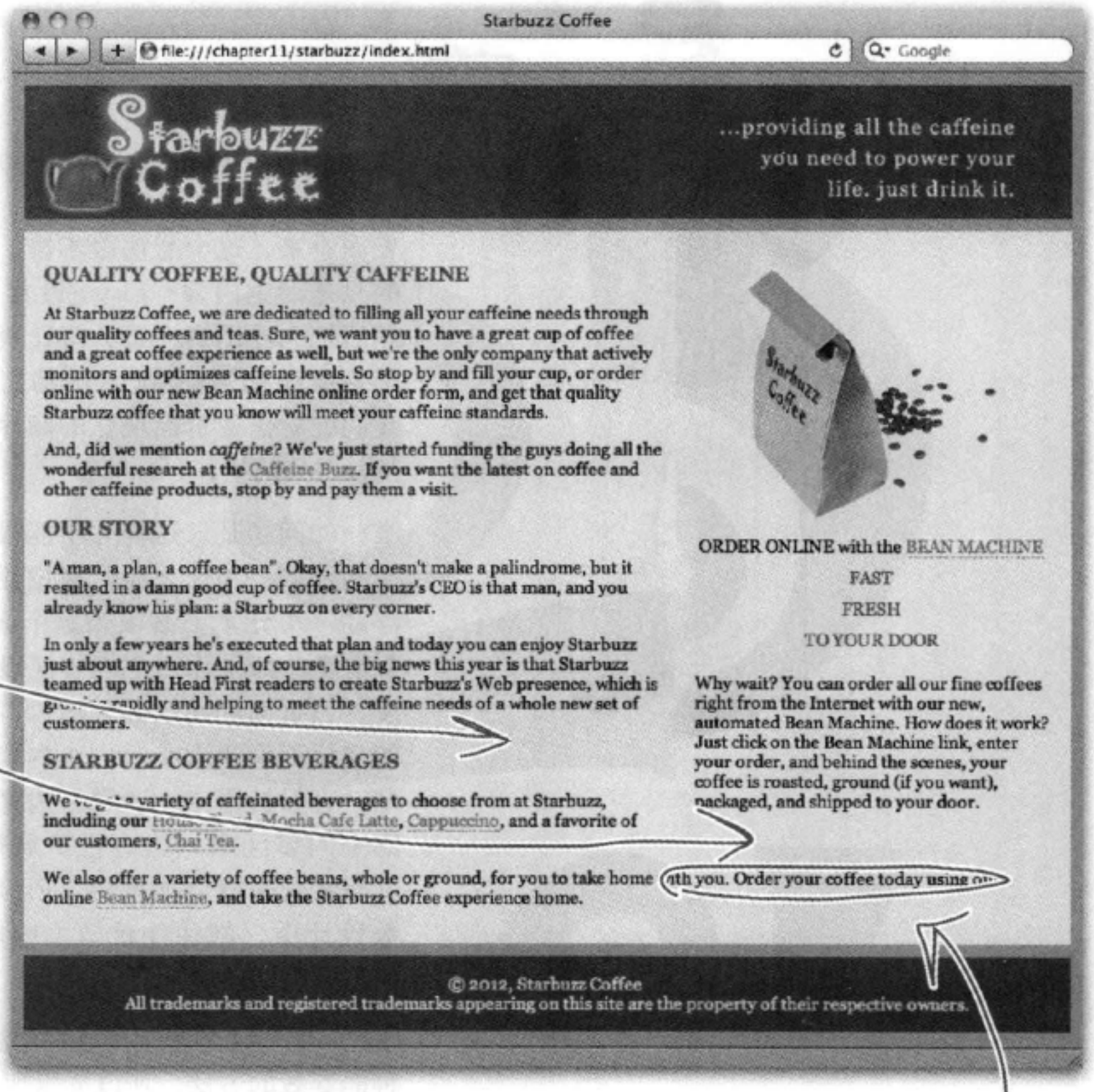
唯一的问题是，这样并不能得到一个漂亮的页面。为什么呢？要记住，如果要浮动一个元素，必须为这个元素设置一个宽度，如果你对主内容区设置一个宽度，当页面的其余部分随着浏览器宽度的变化而调整大小时，主内容区的宽度却一直保持固定。通常，边栏会设计得比主内容区窄，当边栏扩展得太宽时，页面看起来就会很糟糕。所以，在大多数设计中，都往往希望主内容区扩展，而不是边栏。

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。

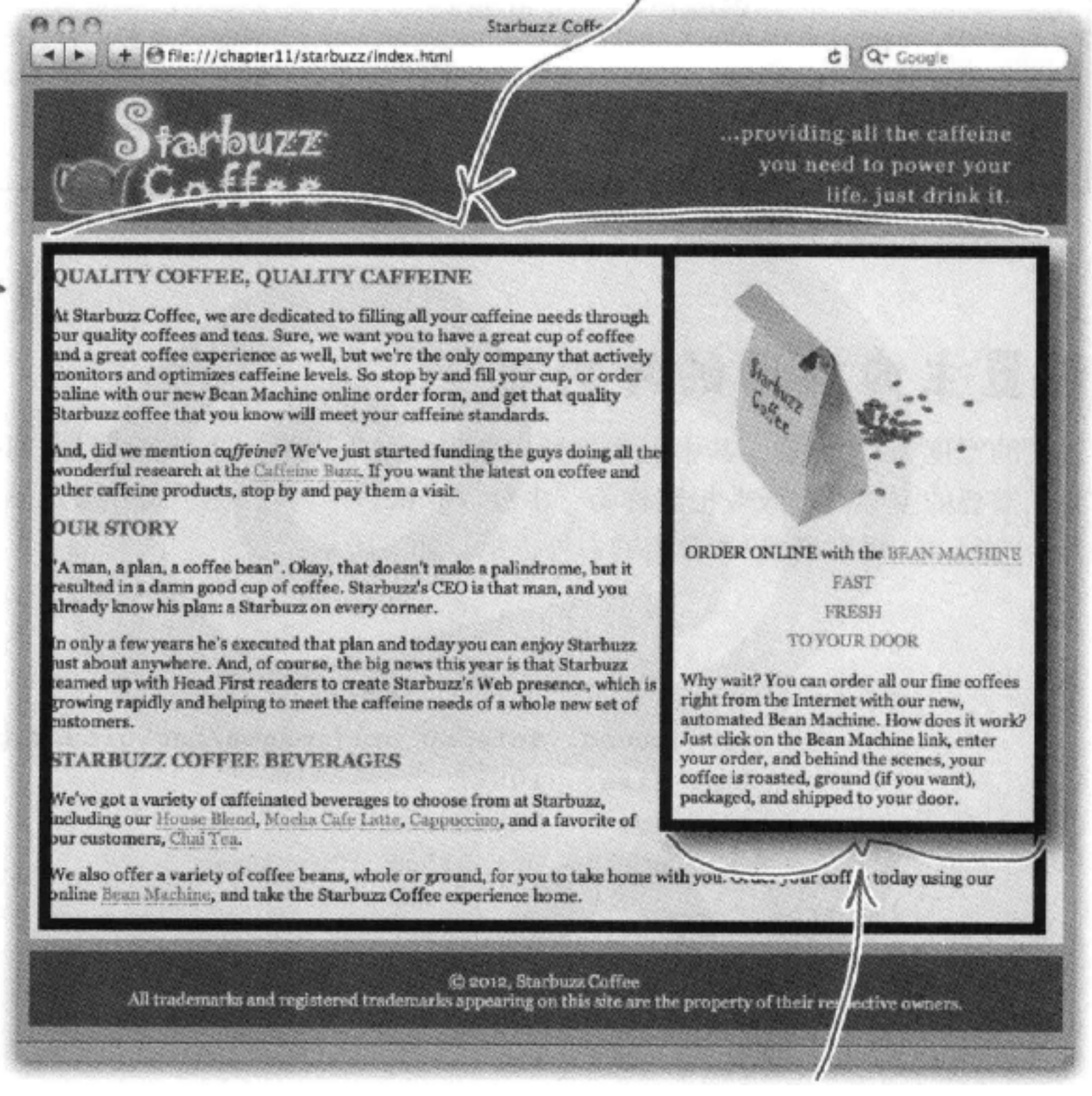
文本围绕在边栏下面，这也使得页面看起来不像有两栏。嗯，休闲室就是这么做的，也许只能这样吧？

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。

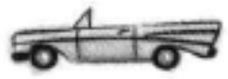
设置主内容区的外边距

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：

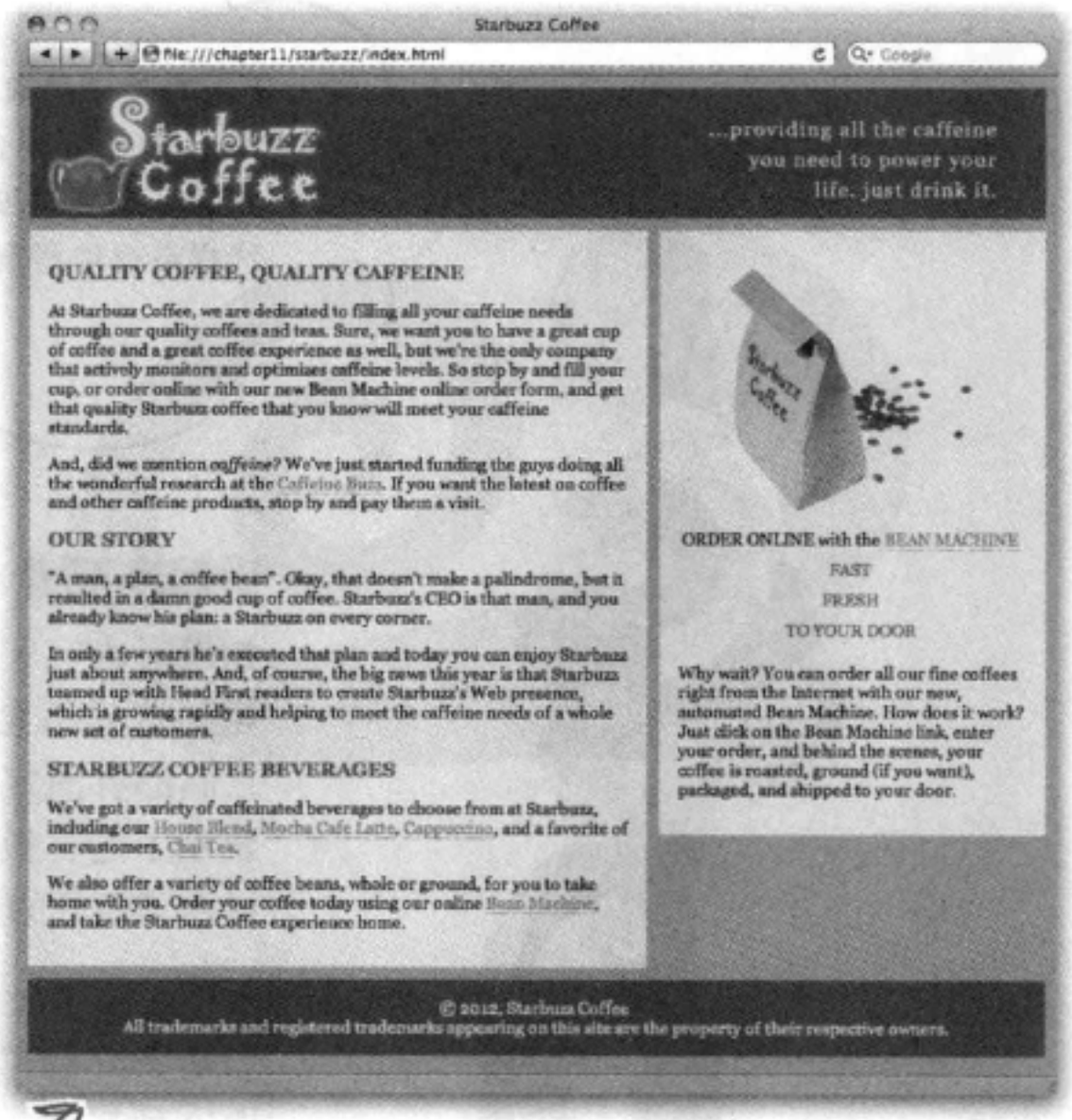
```
#main {
        background: #efe5d0 url(images/background.gif) top left;
        font-size:  105%;
        padding:    15px;
        margin:     0px 330px 10px 10px;
}
```

我们把右外边距改为330像素，与边栏的大小一致。

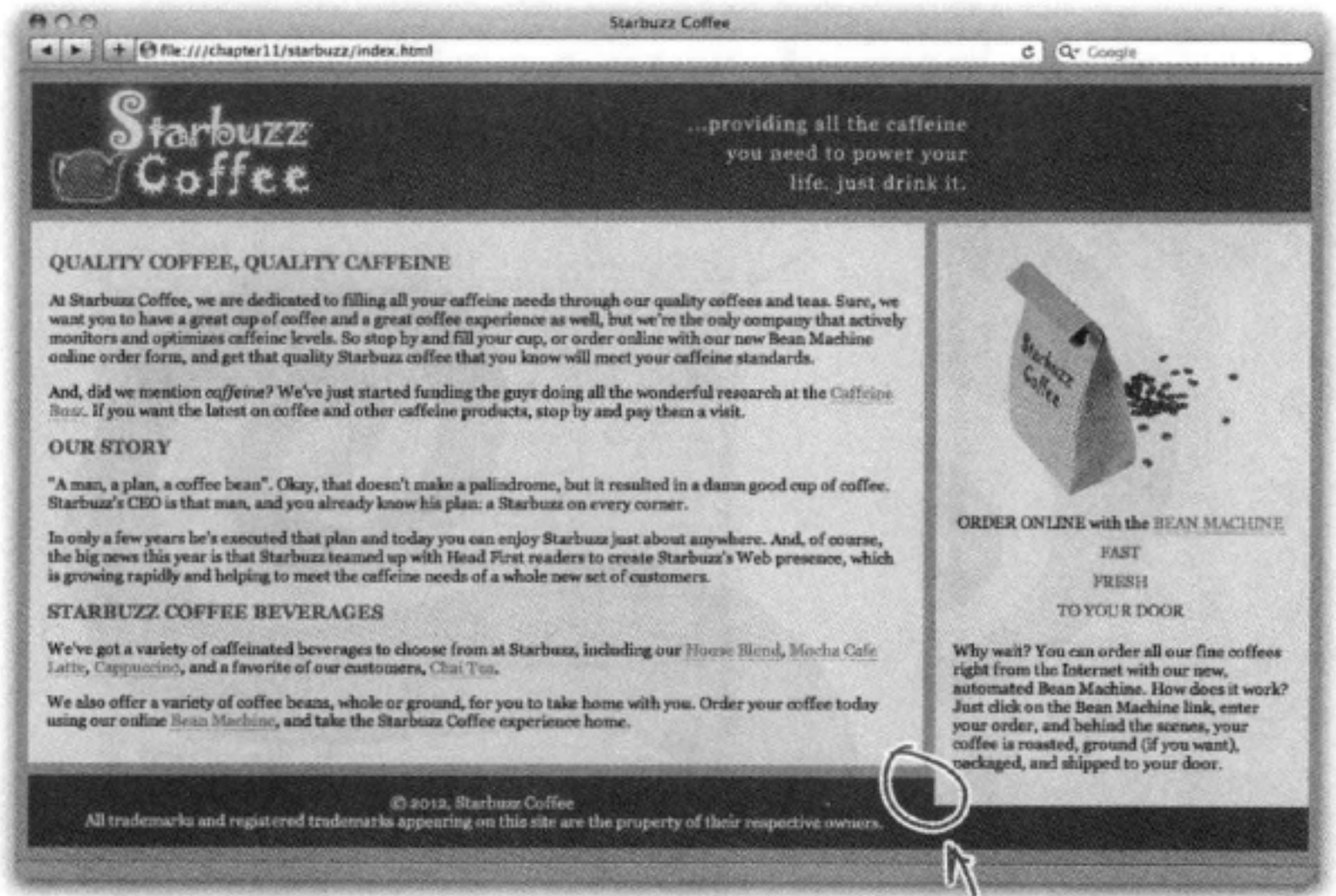
试一试



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

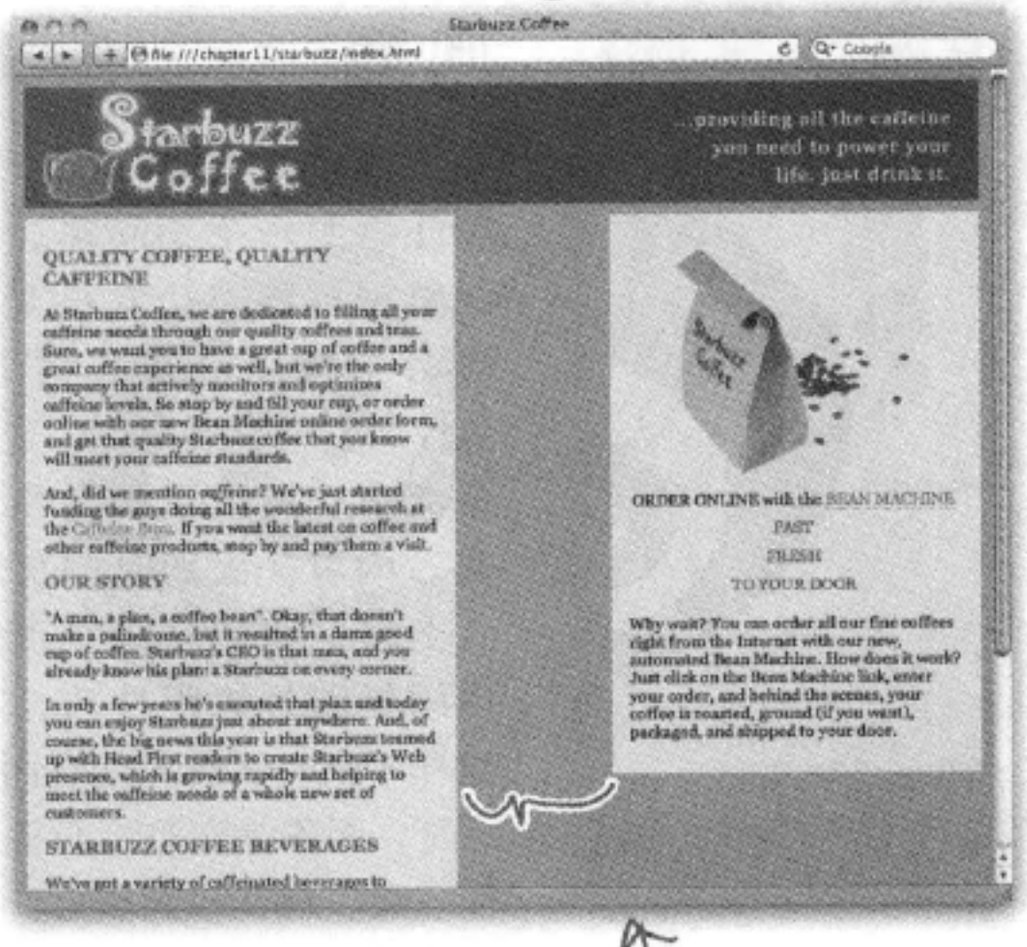
测试这个页面时，你可能已经注意到一个小问题。如果调整浏览器，让它很宽，页脚会上移出现在边栏下面。为什么？嗯，要记住，边栏不在流中，所以页脚会将它忽略，内容区太短时，页脚就会上移。我们可以对页脚使用同样的外边距技巧，不过这样一来，页脚只能在内容区下面，而不是在整个页面的下面。唉，现在该怎么办？

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。

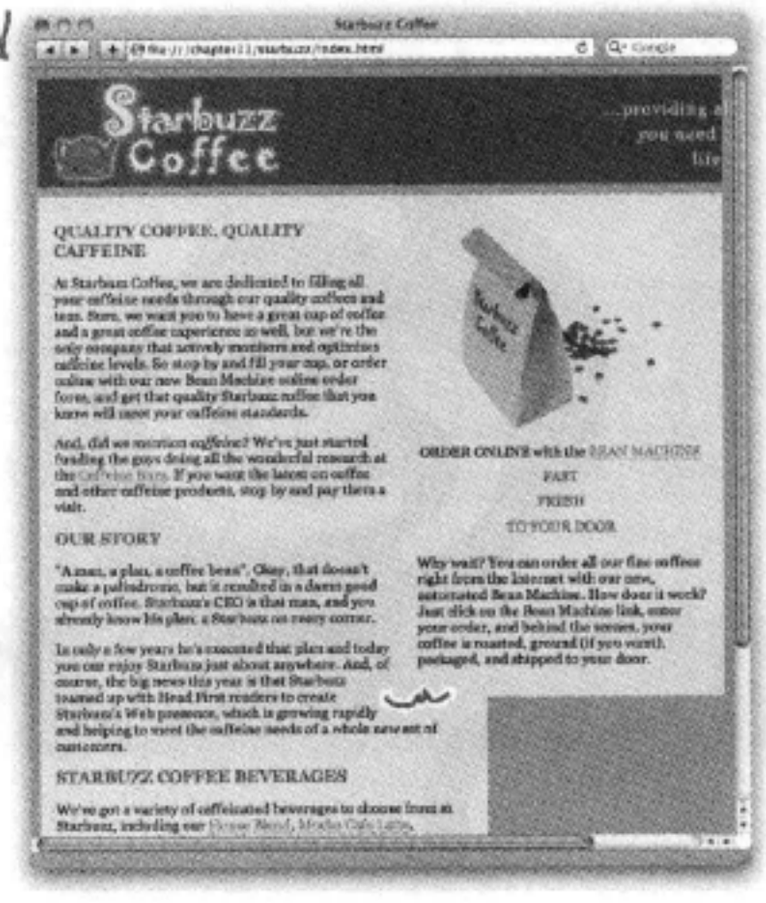
如果对内容区和边栏都设置宽度，会有一个问题，这样不允许页面正确地扩展和收缩，因为它们都有固定的宽度。查看下面的截屏图，它显示了这样做的效果（或者更确切地讲，这里展示了不好的效果）。

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。

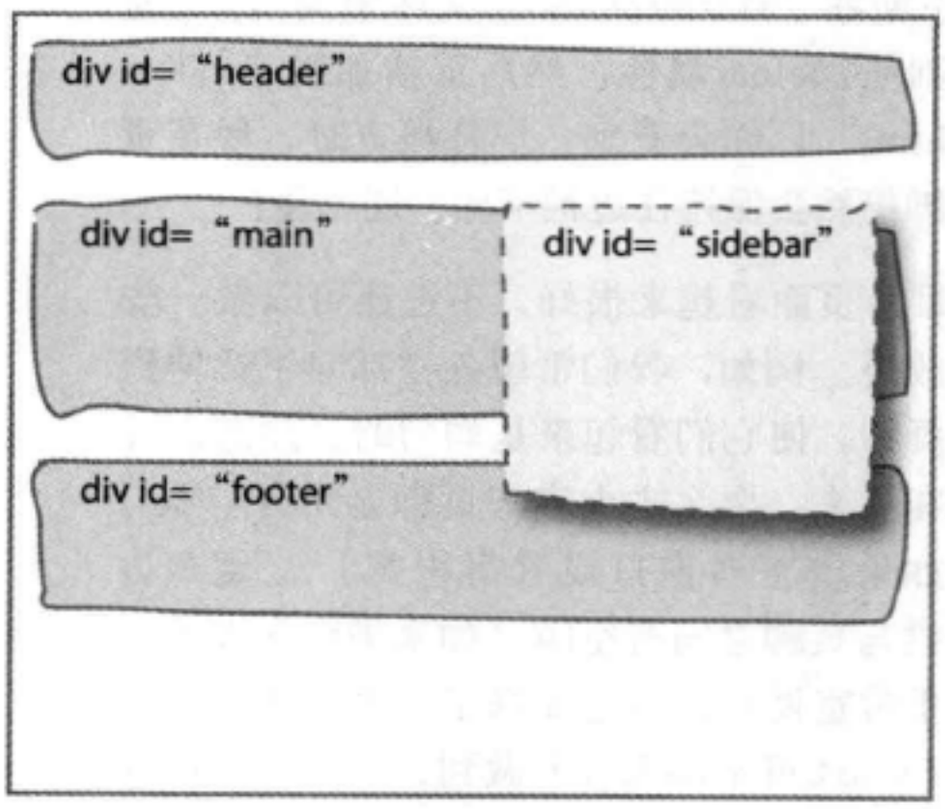


解决重叠问题

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）。

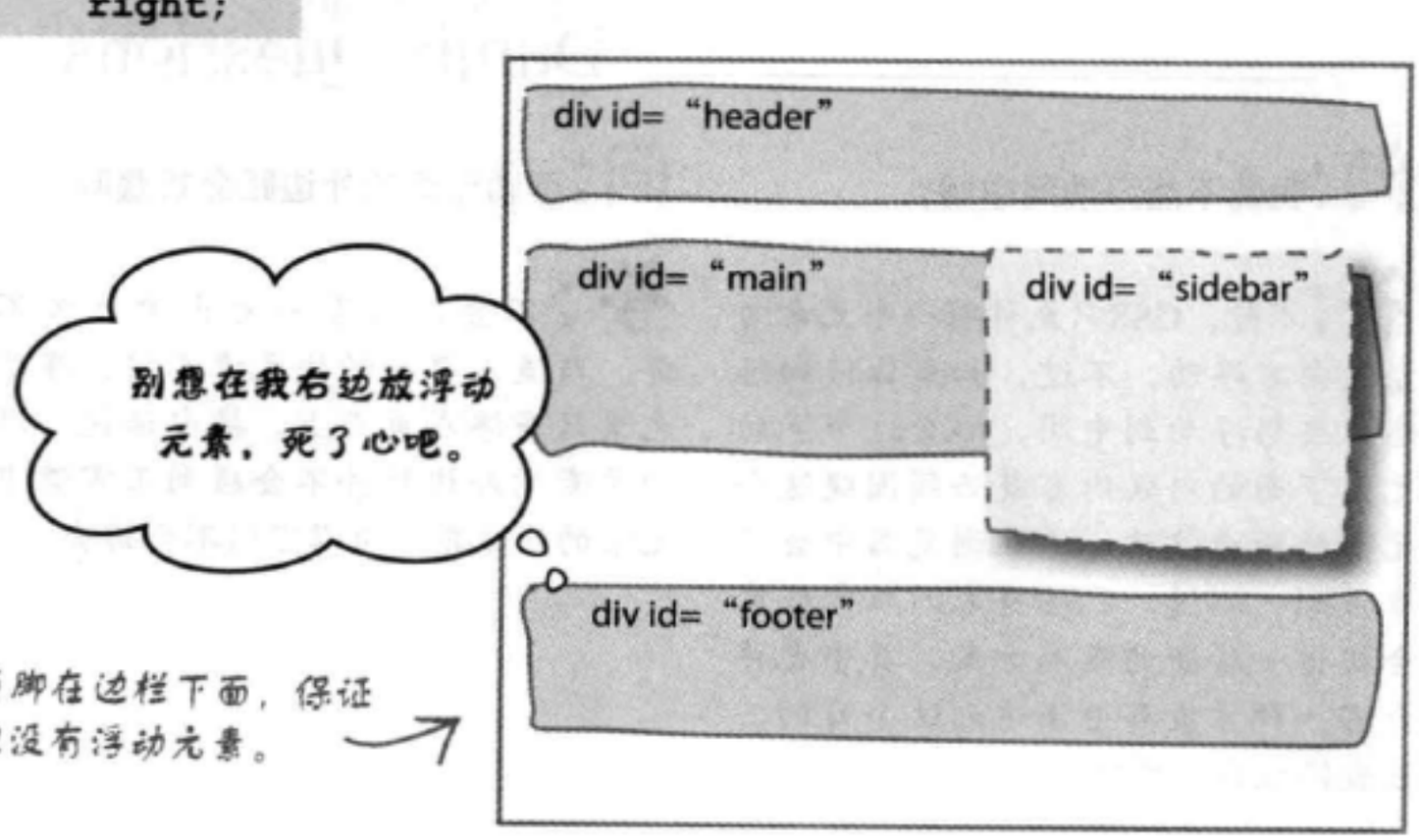


可以使用元素的CSS clear属性来提出请求：当元素流入页面时，在这个元素左边、右边或两边不允许有浮动内容。下面来试一下……

```
#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。



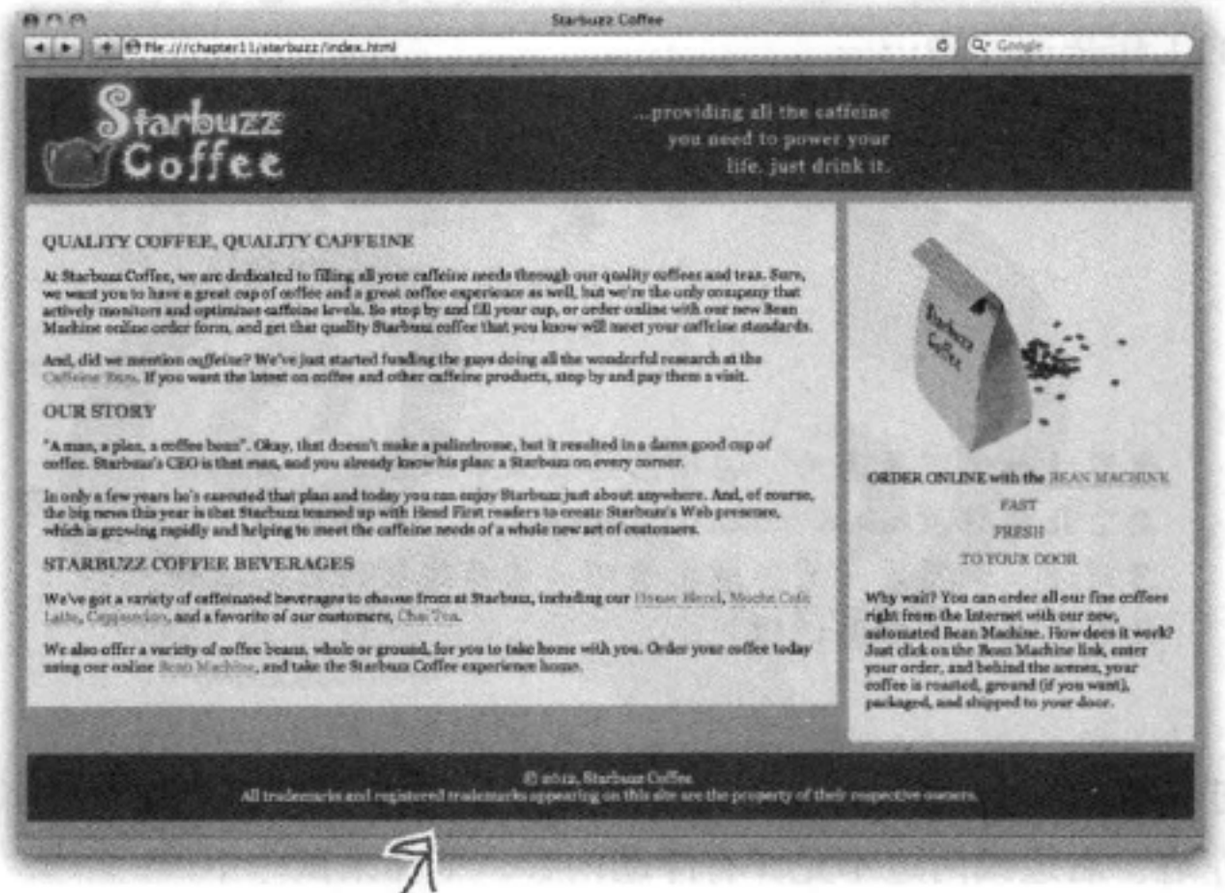
现在页脚在边栏下面，保证它右边没有浮动元素。

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。

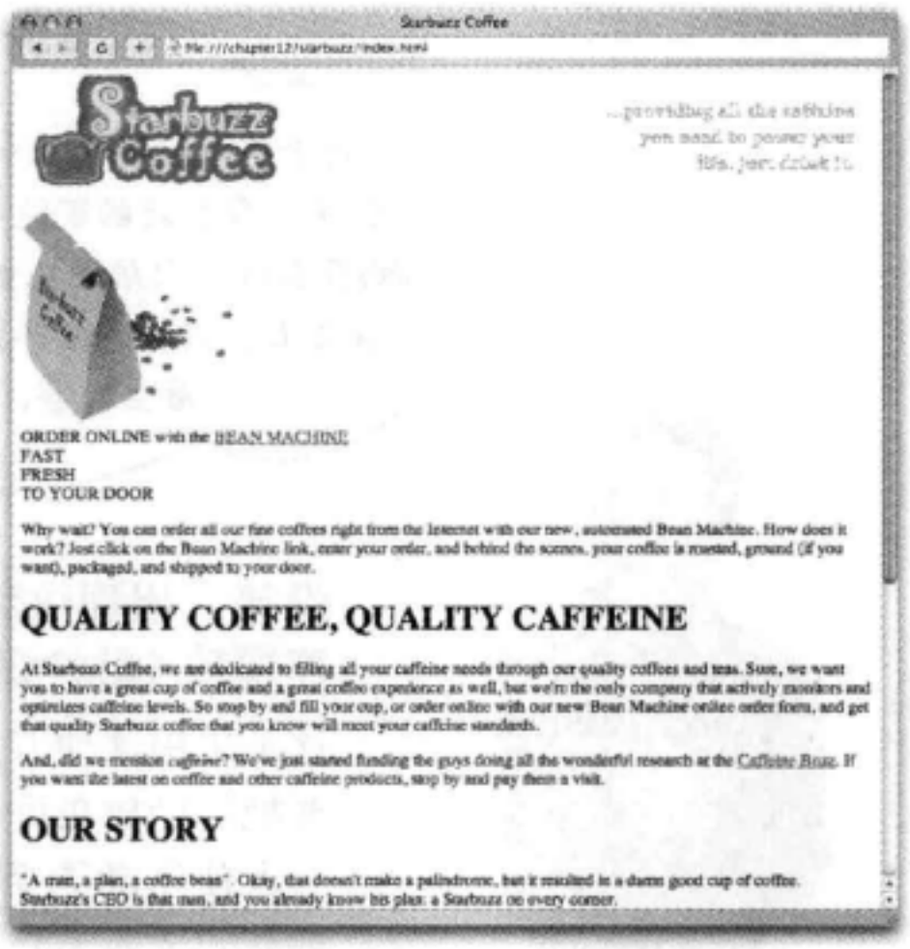
这是采用这种方式设计页面的缺点之一，由于我们需要把边栏放在页眉下面，而且要在主内容之前，如果有人使用功能受限的浏览器（PDA、小的移动设备、屏幕阅读器等），他们看到的页面就会采用我们写页面时使用的元素顺序，边栏在最前面。不过，大多数人更希望先看到主内容，然后才是某种形式的边栏或导航。

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。

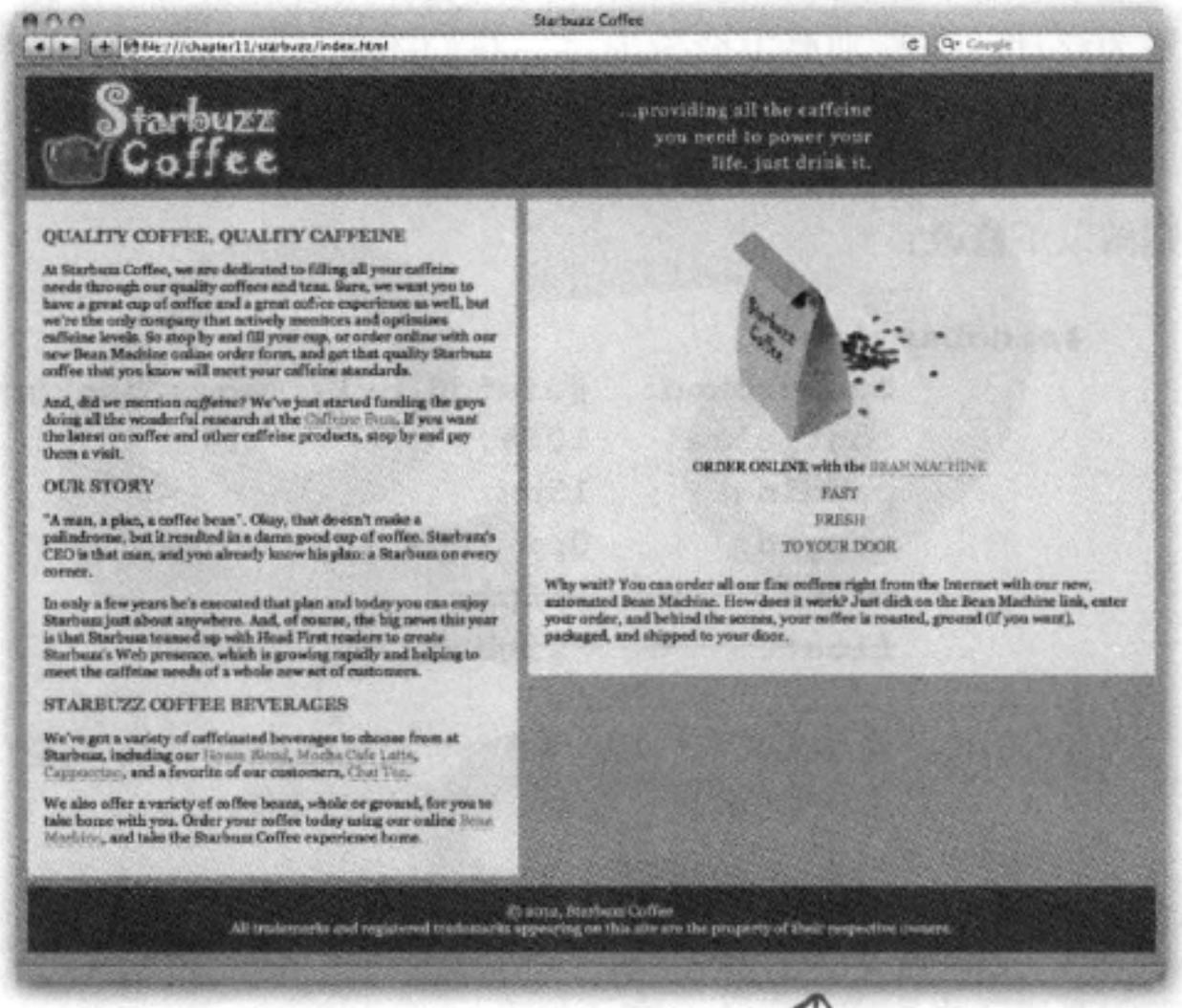
现在只需要调整页脚，让它清空左边的所有浮动元素，而不是右边。

```
#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。

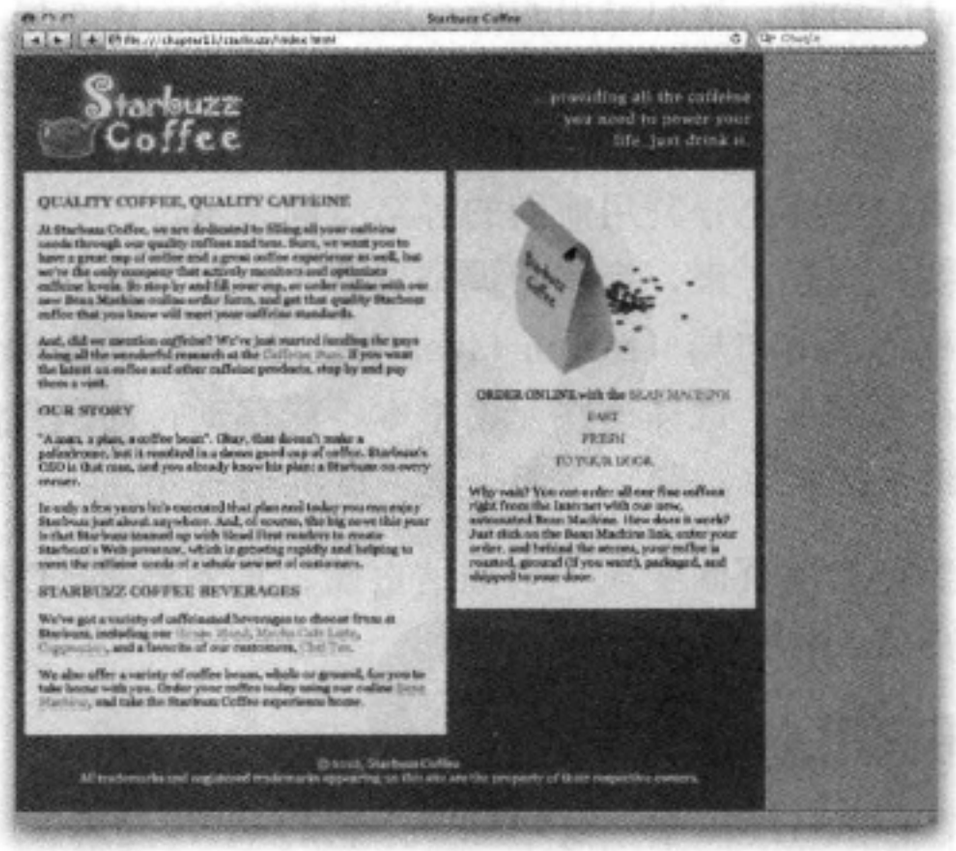
由于这是第一次指定这个 的样式，所以下面增加一些内边距，并让它有自己的背景颜色。你会看到这将有助于把整个页面联系在一起。

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：

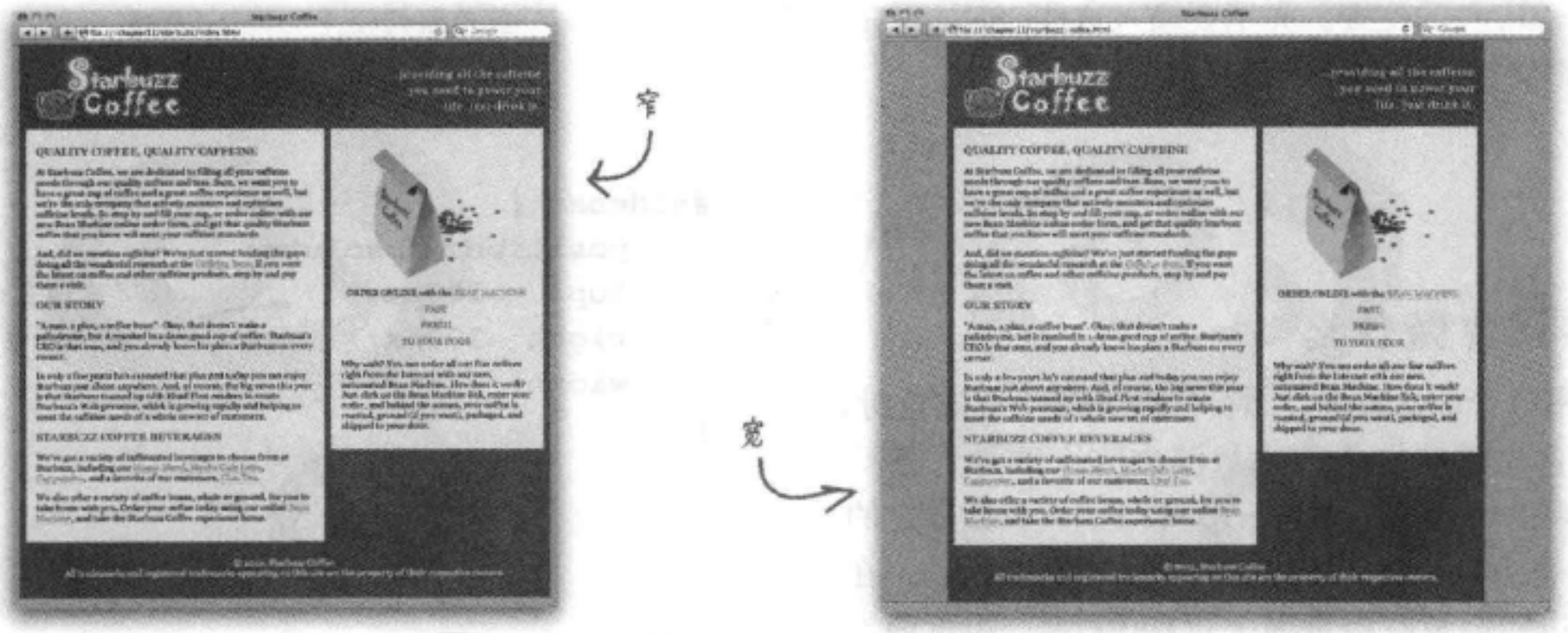
```
#allcontent {
        width: 800px;
        padding-top:    5px;
        padding-bottom: 5px;
        background-color: #675c47;
        margin-left:  auto;
        margin-right: auto;
}
```

“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。

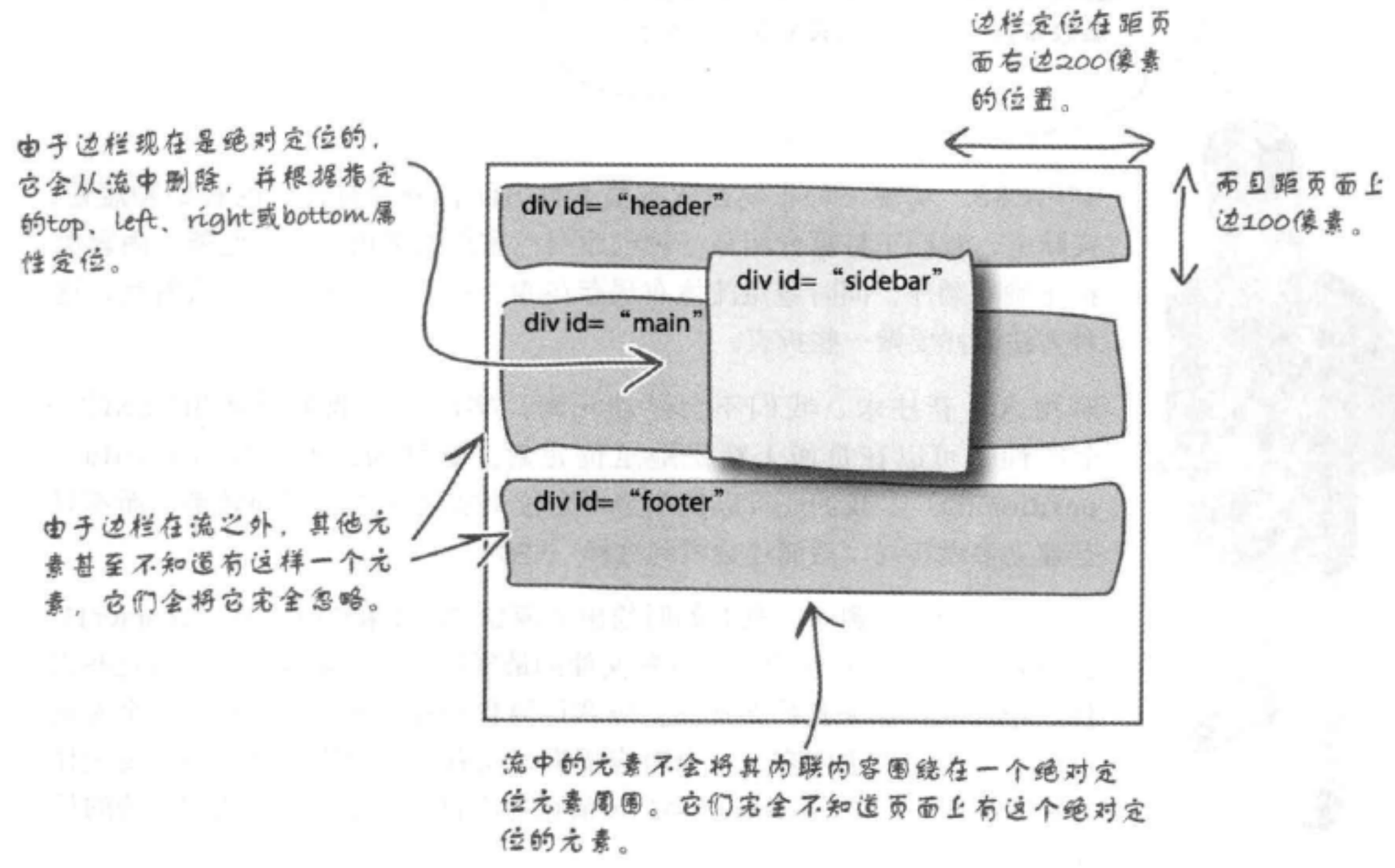
```
#sidebar {
    position: absolute;
    top:    100px;
    right: 200px;
    width: 280px;
}
```

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

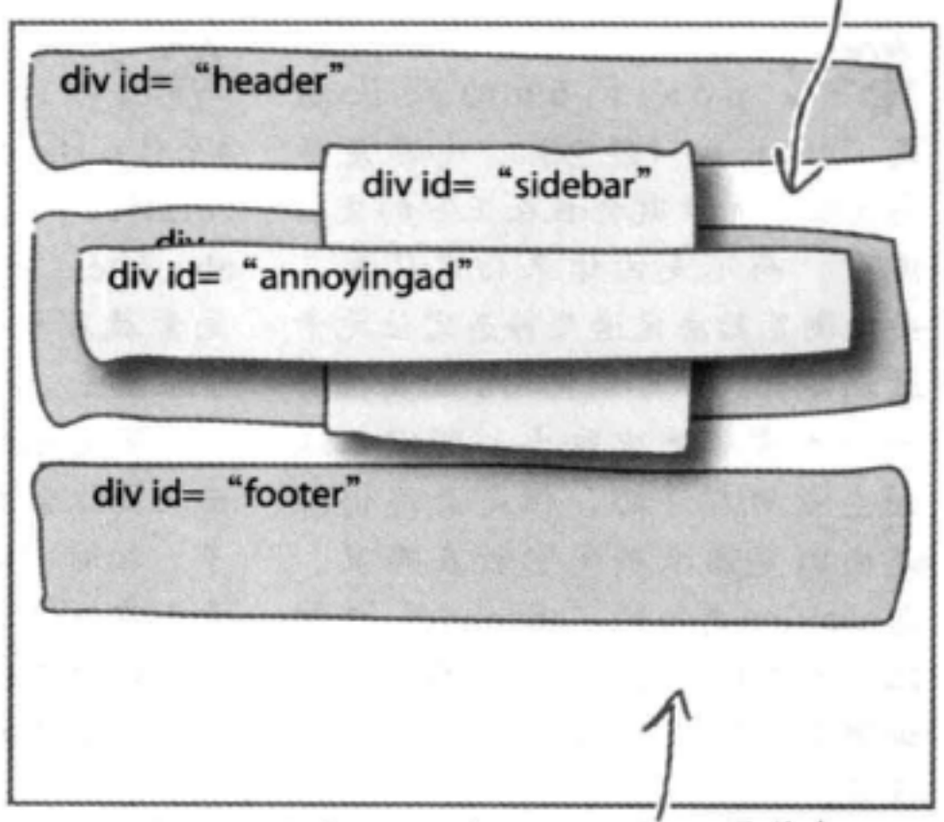
下面来看另一个例子。假设我们有另一个 ，id为“annoyingad”。可以这样指定它的位置：

```
#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。



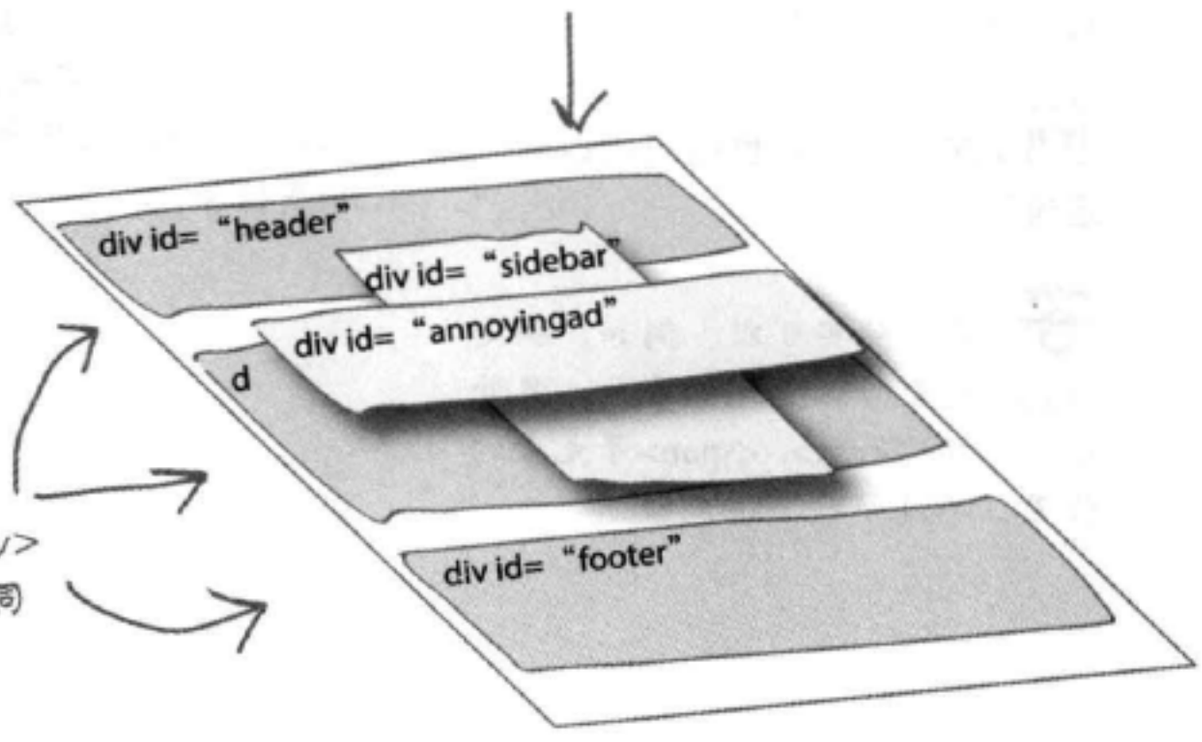
注意annoyingad 覆盖在sidebar 之上。

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。

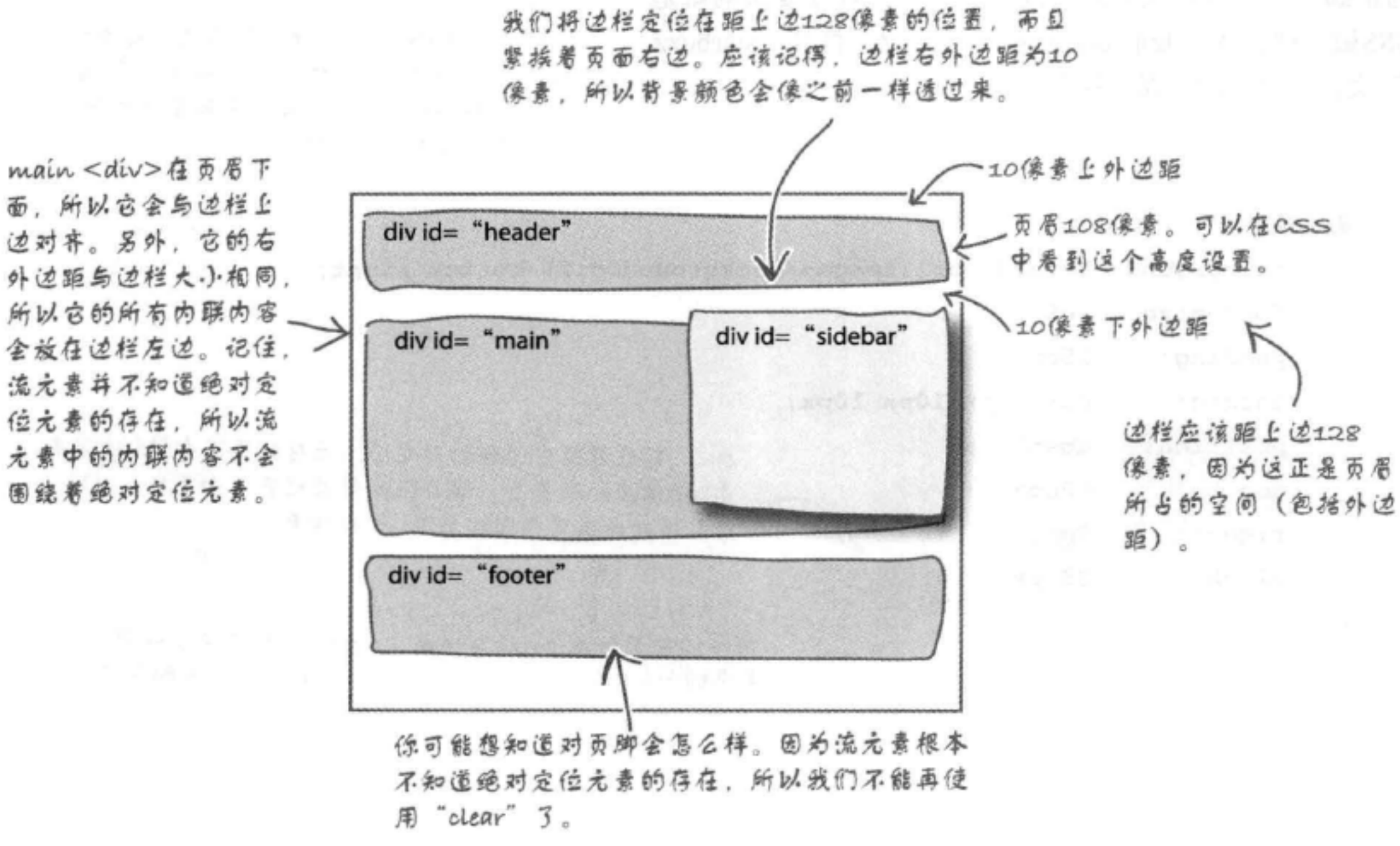
现在只需要调整main

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;
}
```

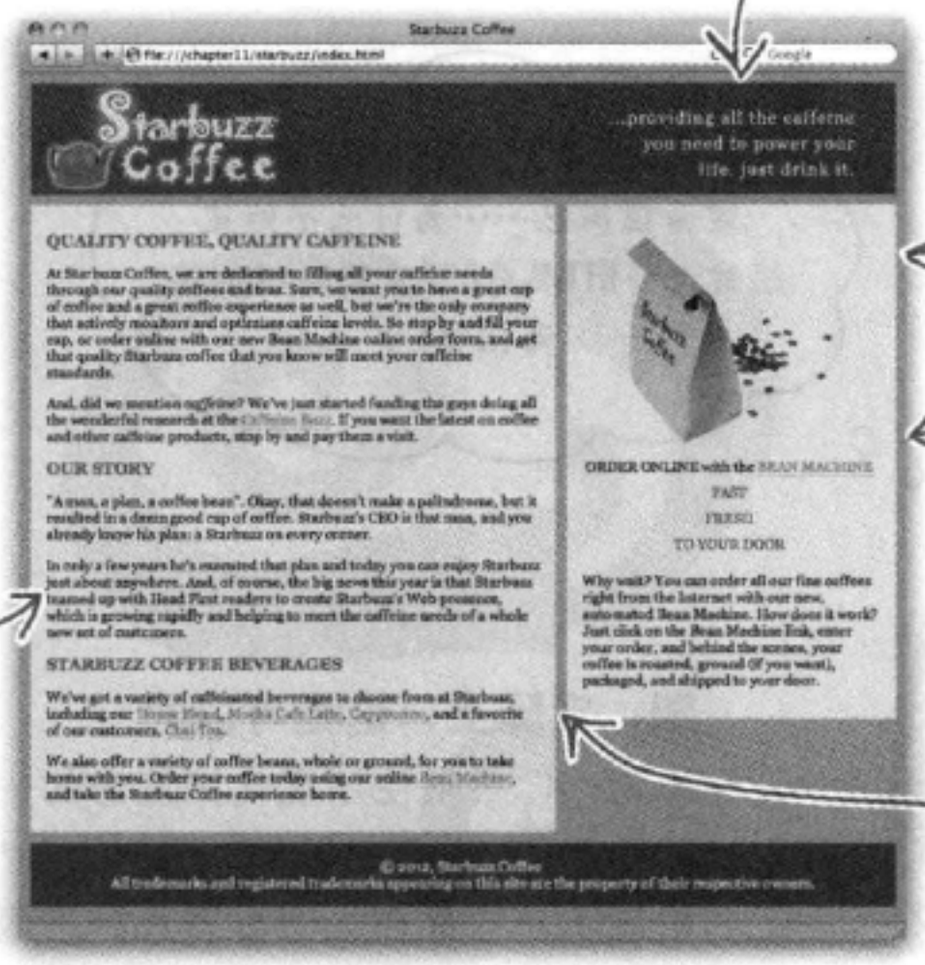
通过为main 指定一个大的外边距，会留出一些空间来放置边栏。这与浮动版本中使用的技术是一样的。唯一的区别是sidebar 放在外边距空间上。

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。

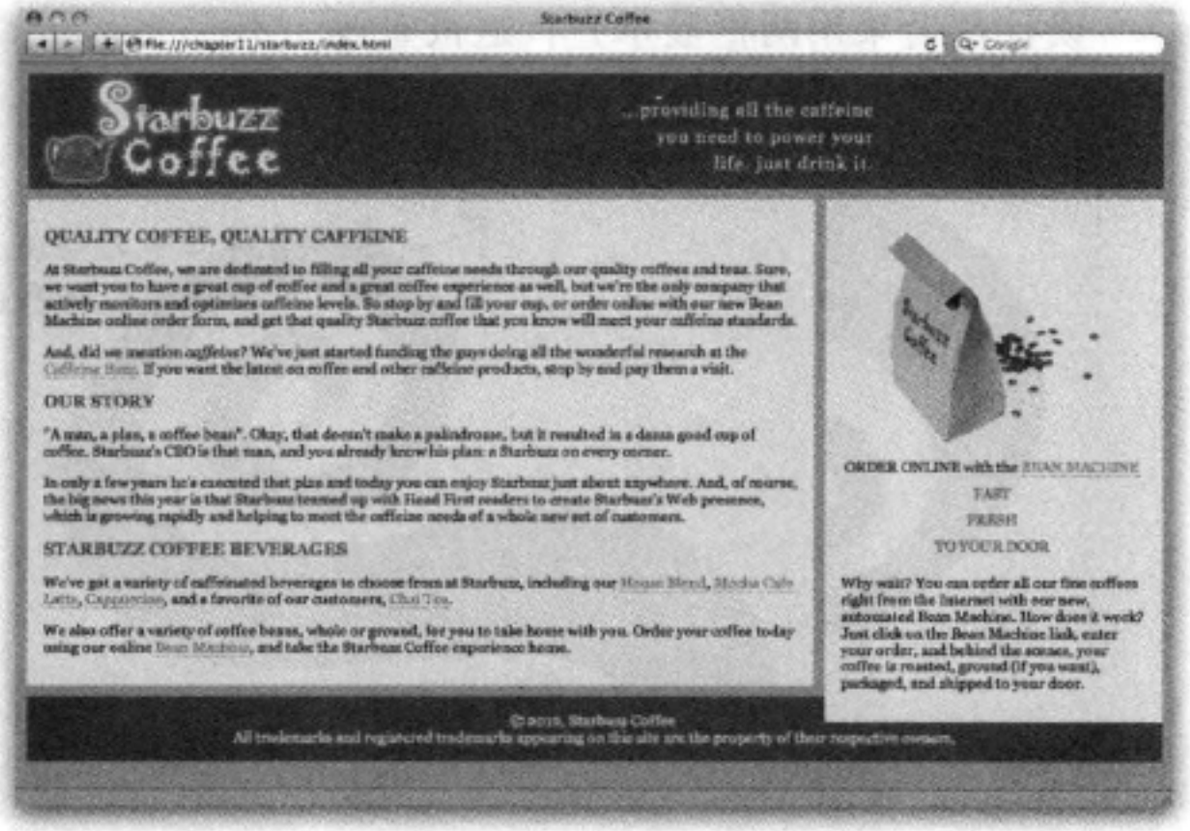
调整浏览器大小时，边栏总是在距上边128像素的位置，而且紧挨着页面右边。

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。

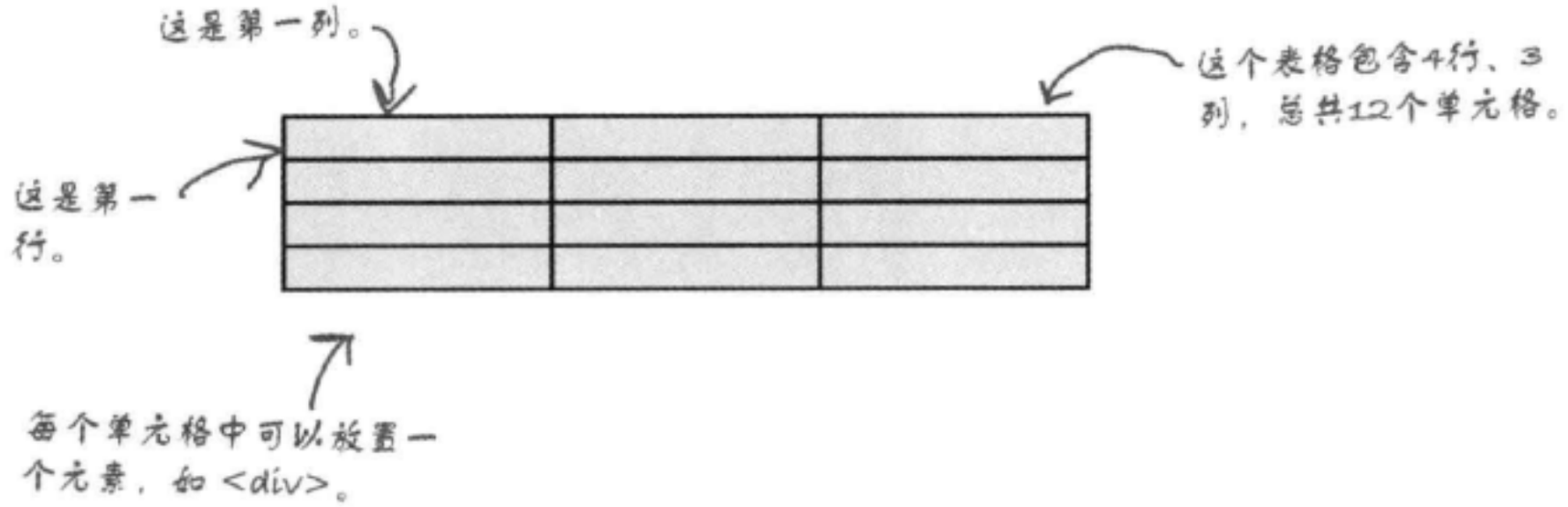
目前，所有现代浏览器都支持这个特性。

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。

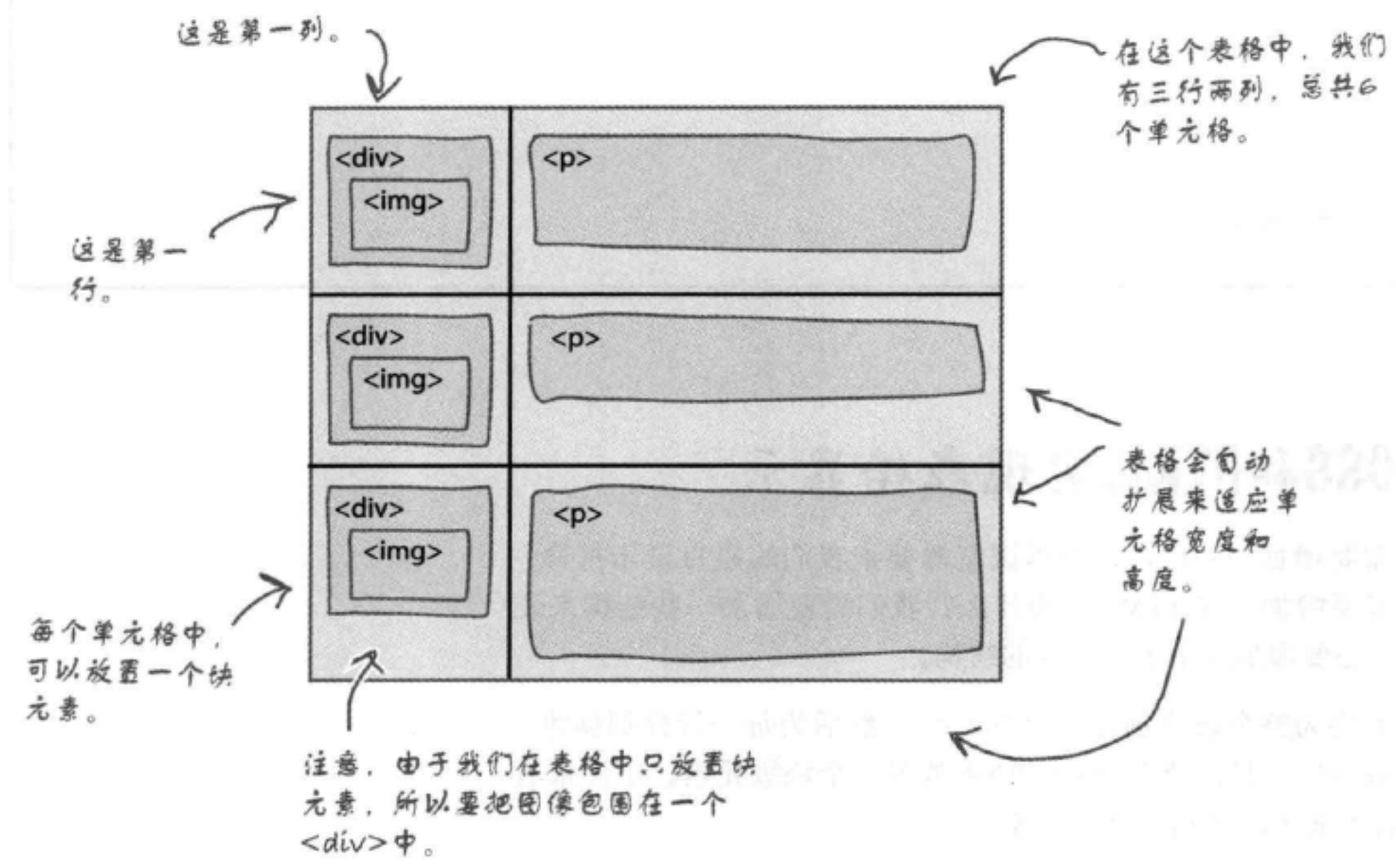
与所有其他布局解决方案一样，甚至表格显示也有自己的优缺点。

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

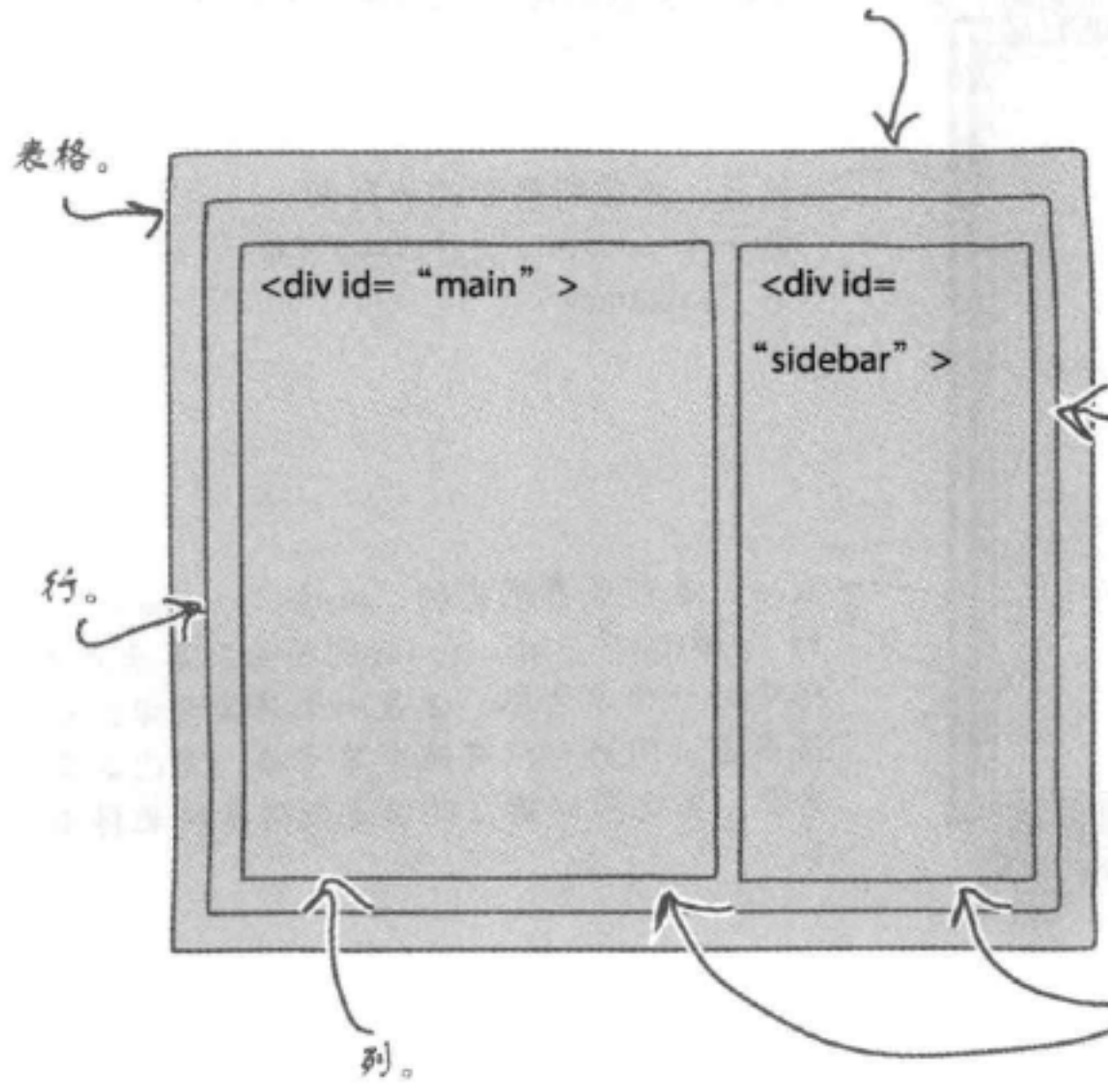
不难想到，我们需要增加一些CSS，告诉浏览器要把我们的栏目显示得像一个表格，不过还需要增加一些HTML。为什么？我们需要增加一些结构来表示表格的行和列，还要增加这个表格本身的结构。

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

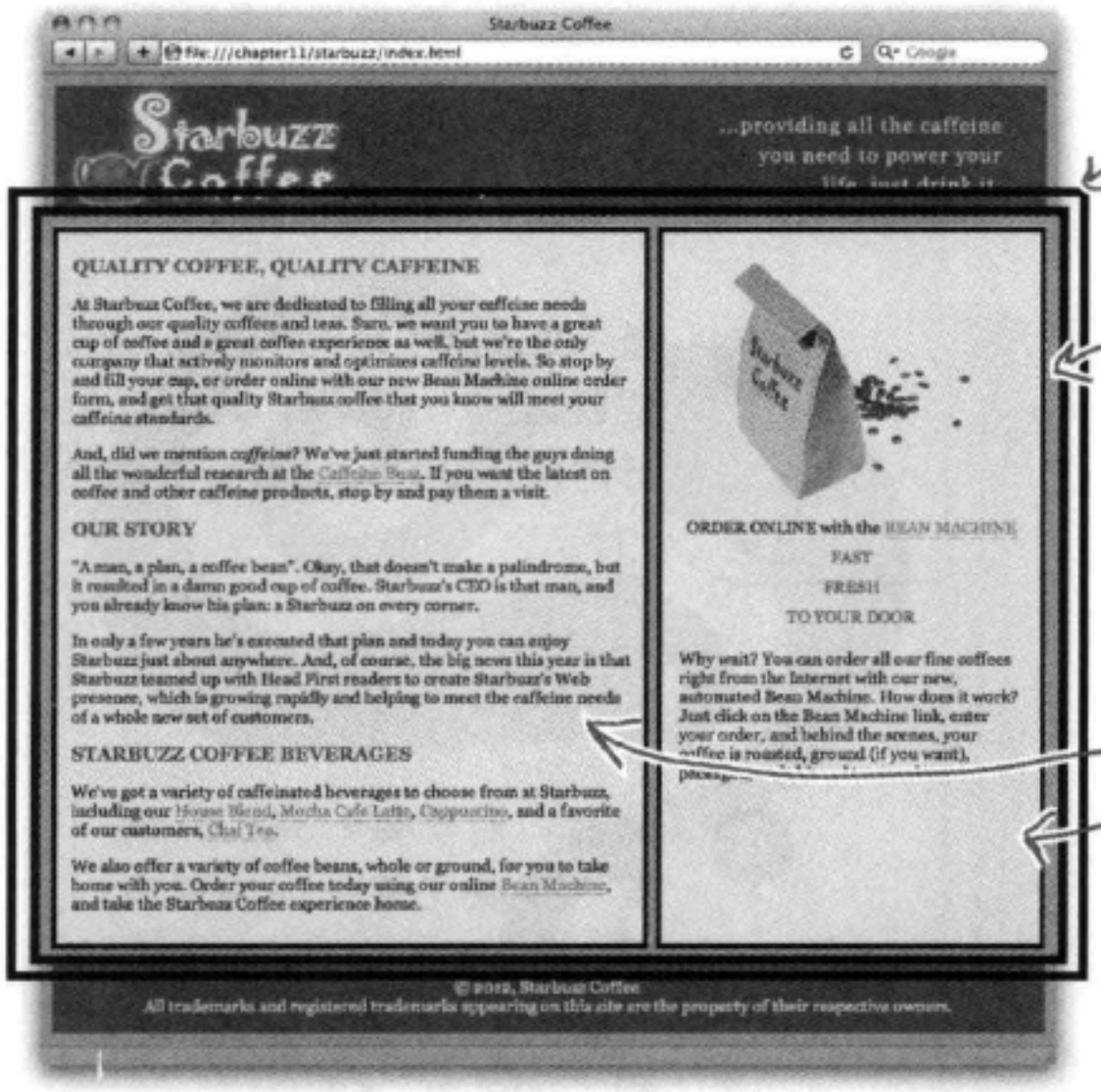
现在该轮到你了：在下面写出建立Starbuzz表格结构所需的HTML。

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。

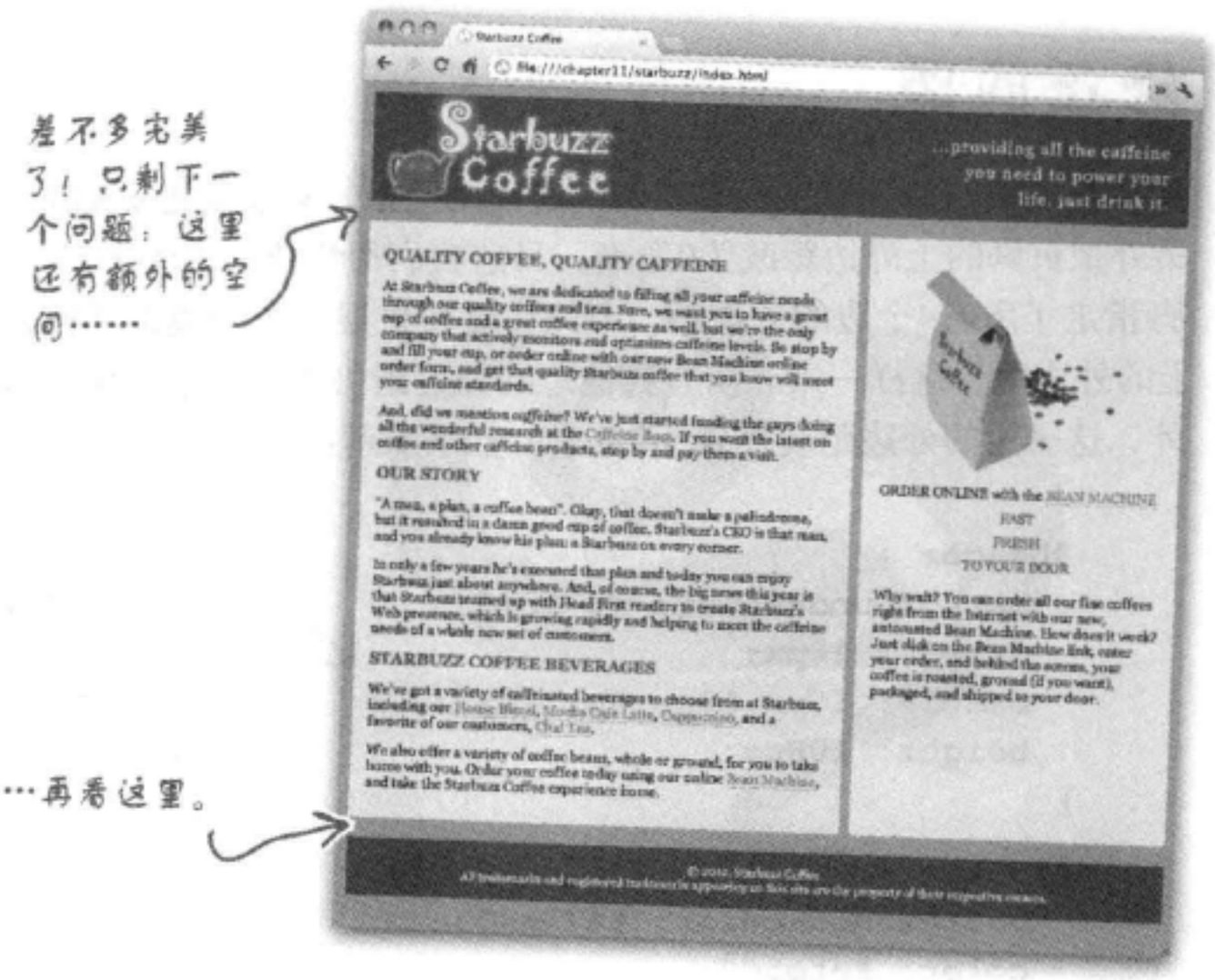
可以删除“main”和“sidebar”的外边距。

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……

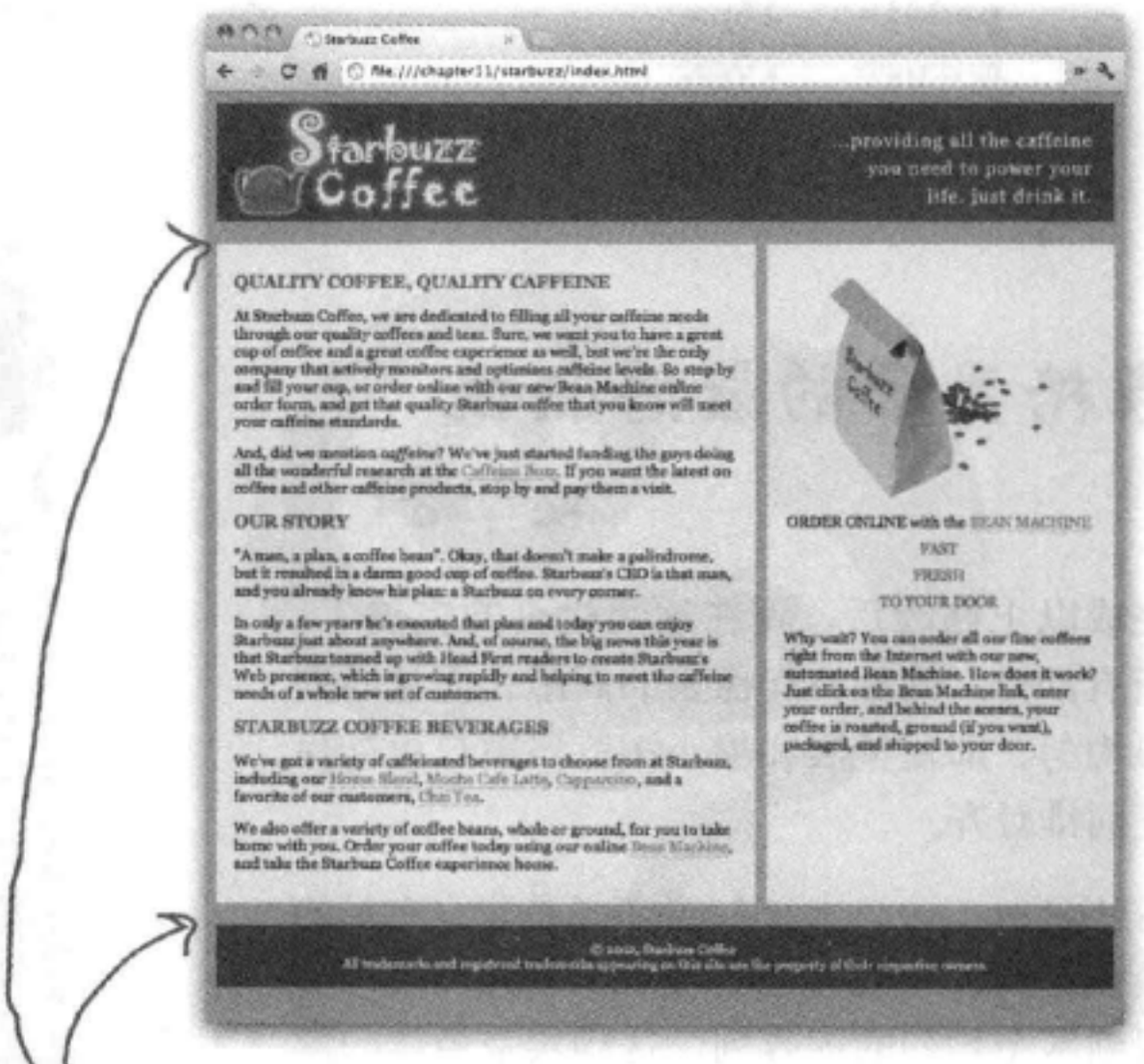


这些间距是怎么回事？

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。

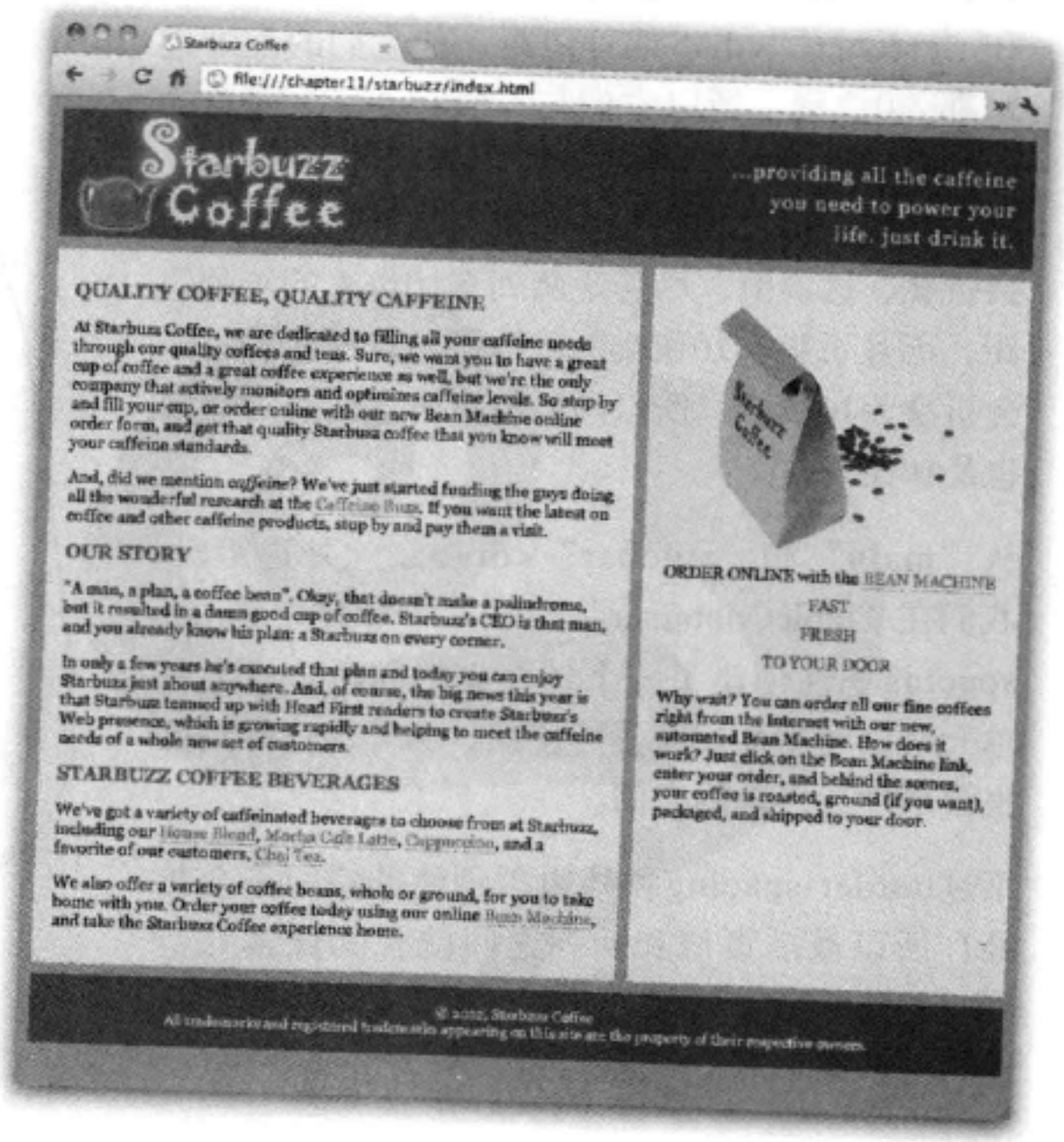
类似的，对于页脚，现在另外3个边的外边距仍为10像素，但上外边距为0像素。

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。

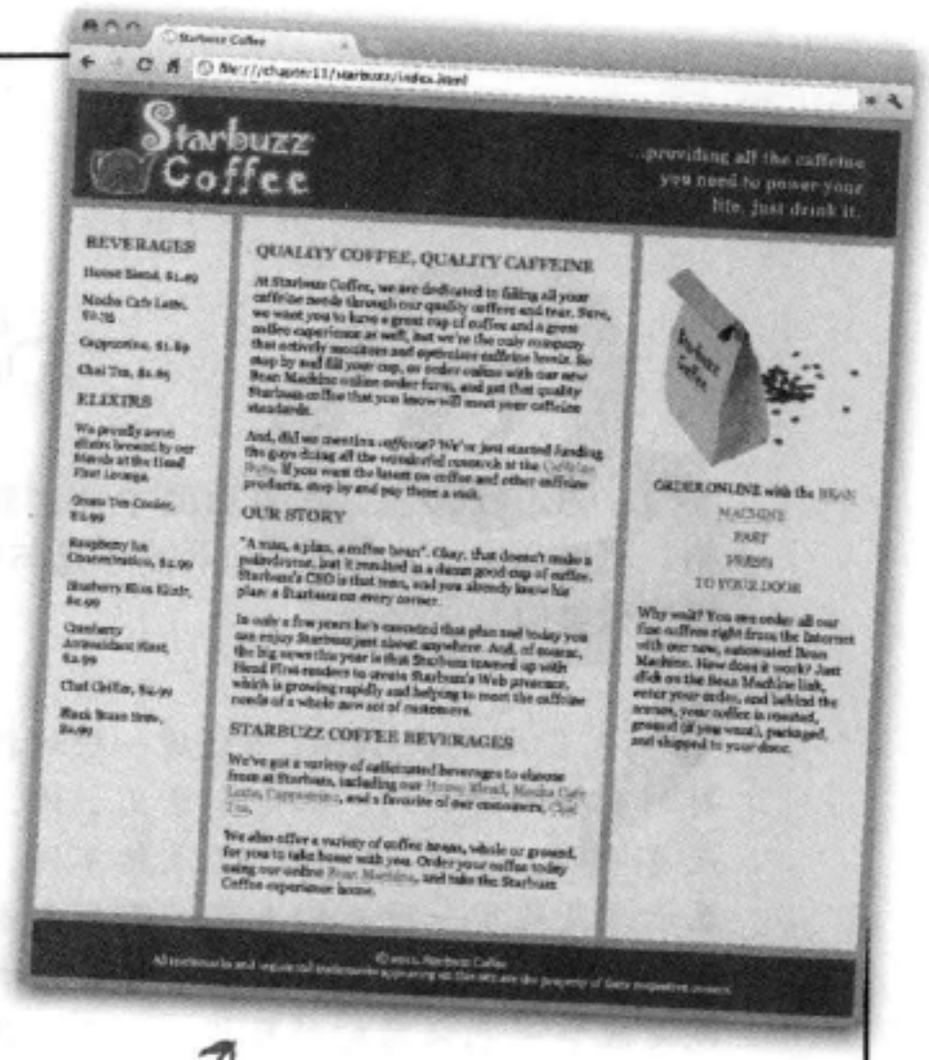
不过display: table并不总是建立布局的最佳工具，但在这里，要在Starbuzz页面中得到两个均匀的内容列，这确实是最佳的解决方案。

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。

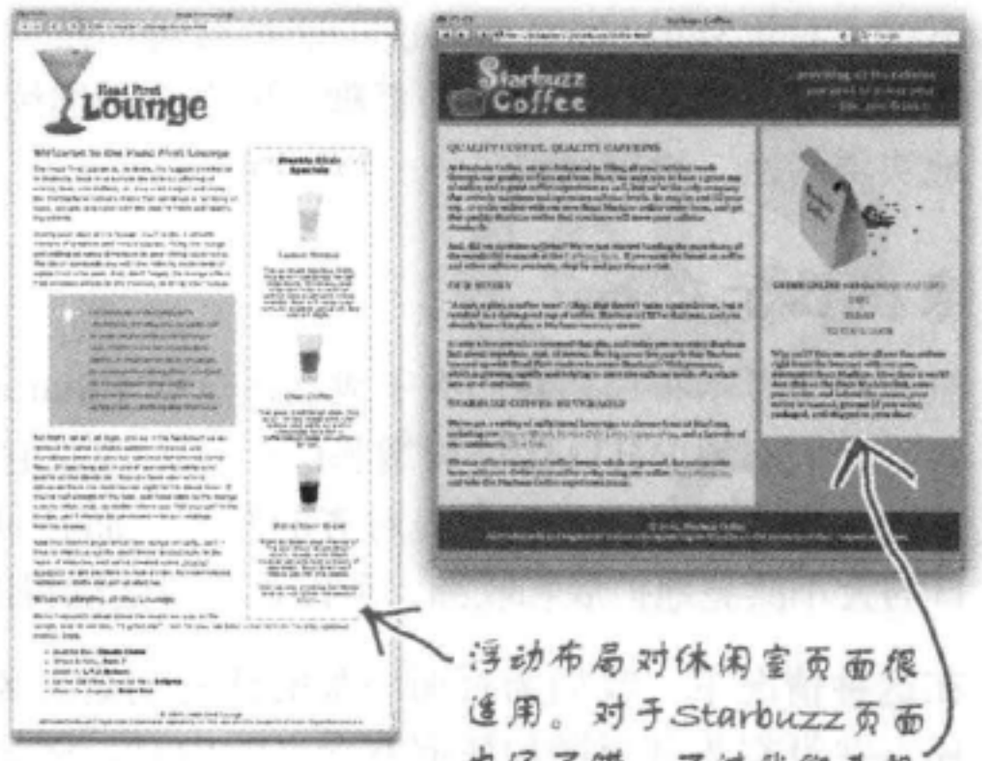
下面复习一下前面的内容。

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。

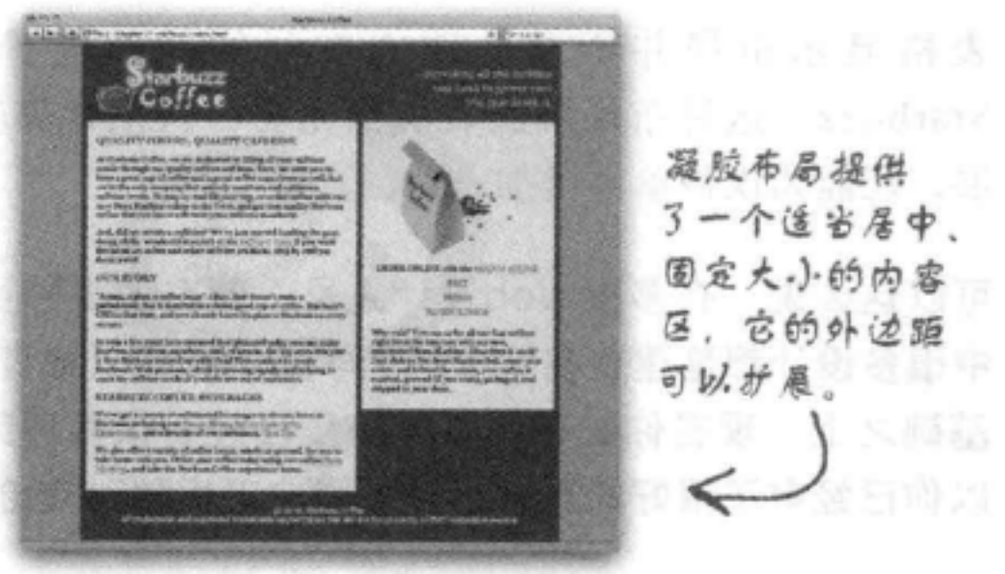
接下来我们使用float使sidebar 浮在Starbuzz页面中，并使用clear确保这个浮动边栏不会与页脚重叠。

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。

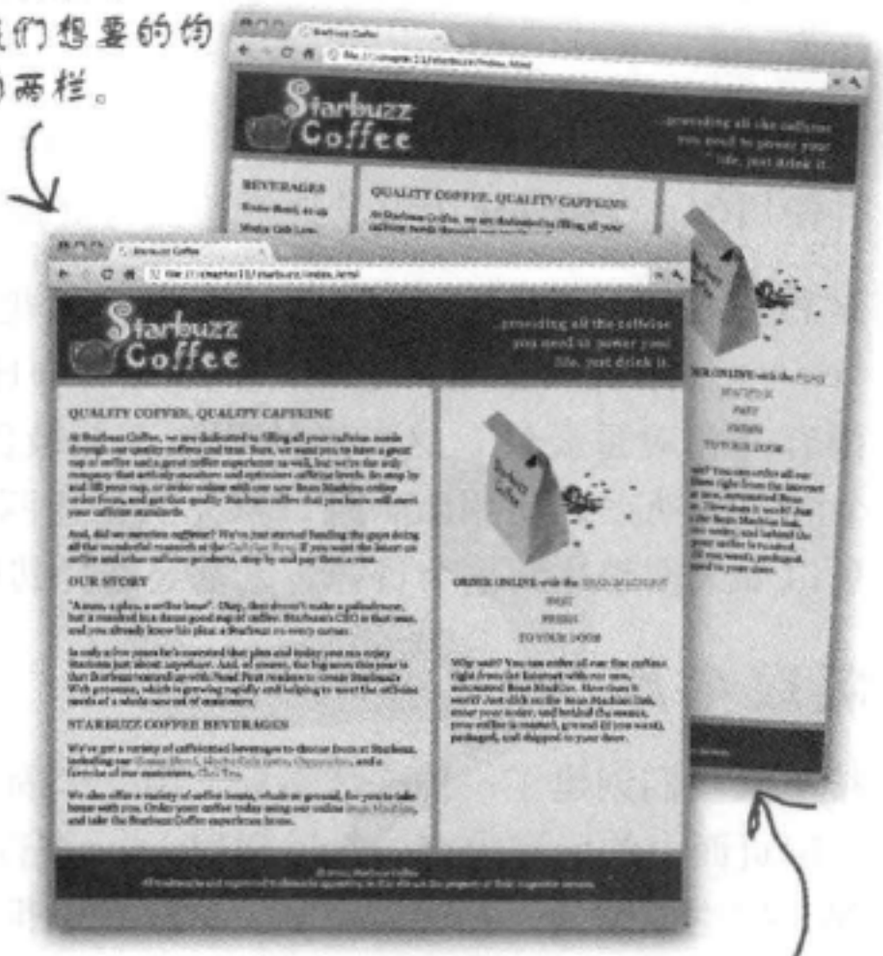
利用绝对定位，可以提供一个漂亮的流体主内容区，还有一个固定的边栏。

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。

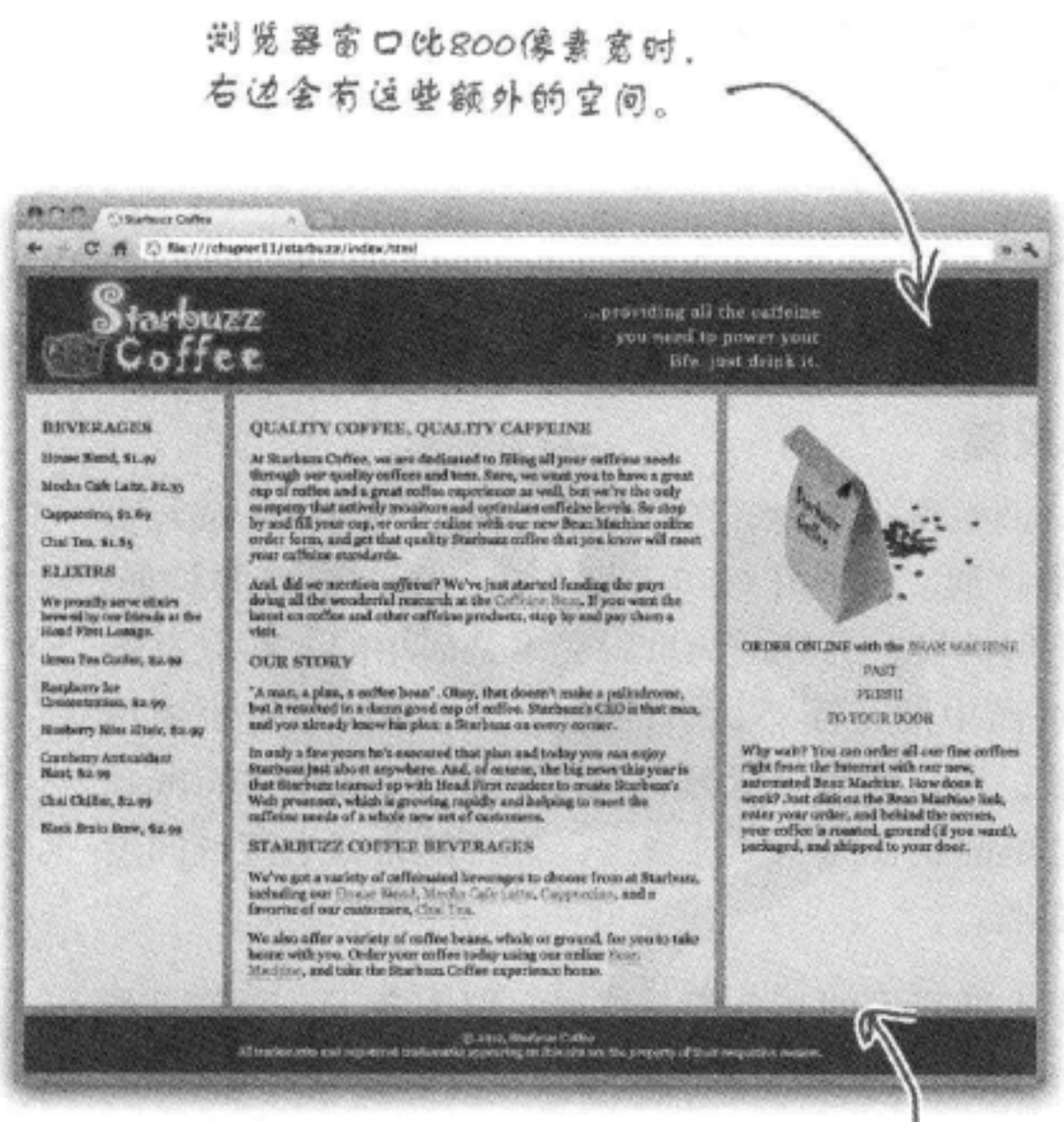
将浏览器窗口调整得很宽时，除了页眉以外，这个Starbuzz页面都能很好地扩展。归功于CSS表格布局，随着窗口的扩展，各列也会成比例扩展。另外因为页脚文本是居中的，所以页脚看起来总在页面中间，而不论页面是宽还是窄。不过页眉没能很好地扩展。可以看到背景颜色确实扩展了，但是Starbuzz口号看起来总是固定在同一个位置上，你可能希望它应该向后移到窗口的右边。

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

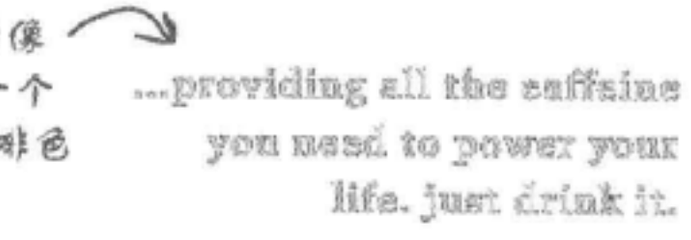
下面来看看页面的变化。打开你的浏览器窗口，先让窗口比页眉图像宽，然后缩小窗口，让它比页眉图像窄。你会看到页眉的表现还不尽如人意。



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）。

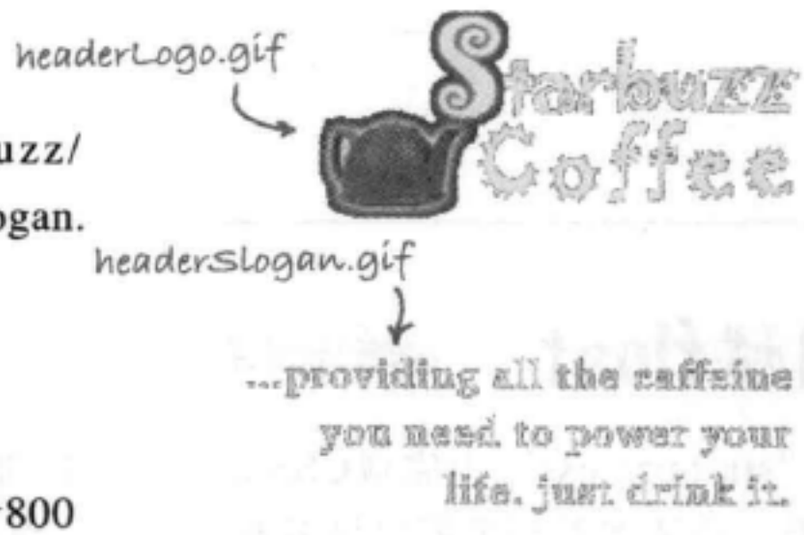


用float修正页眉图像

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”。



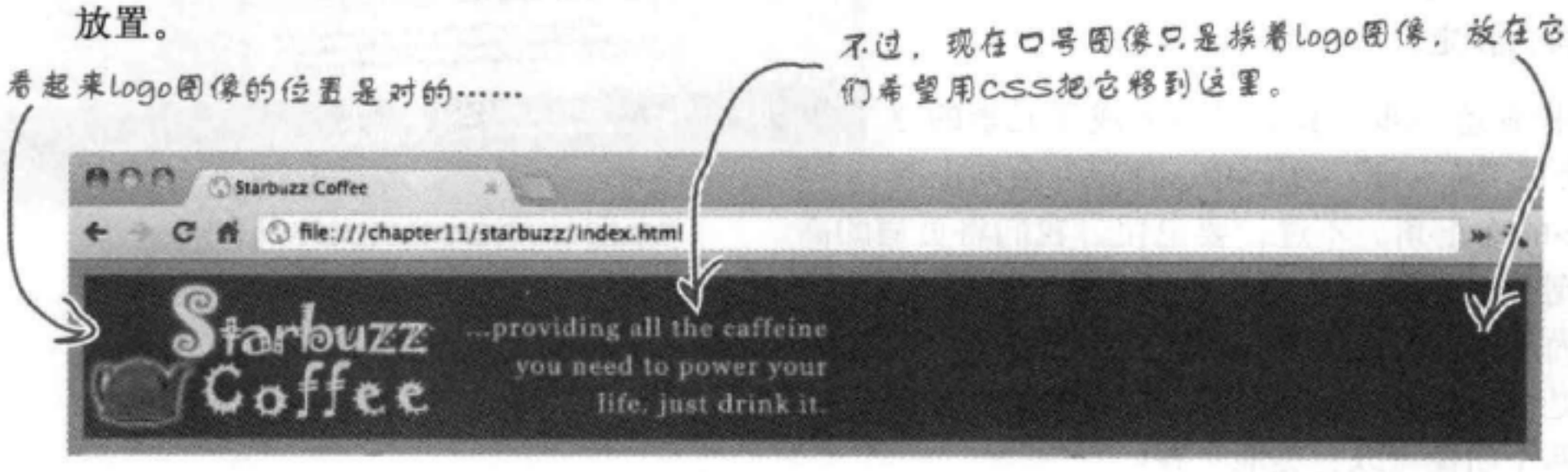
❷ 更新HTML来使用这些图像

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。



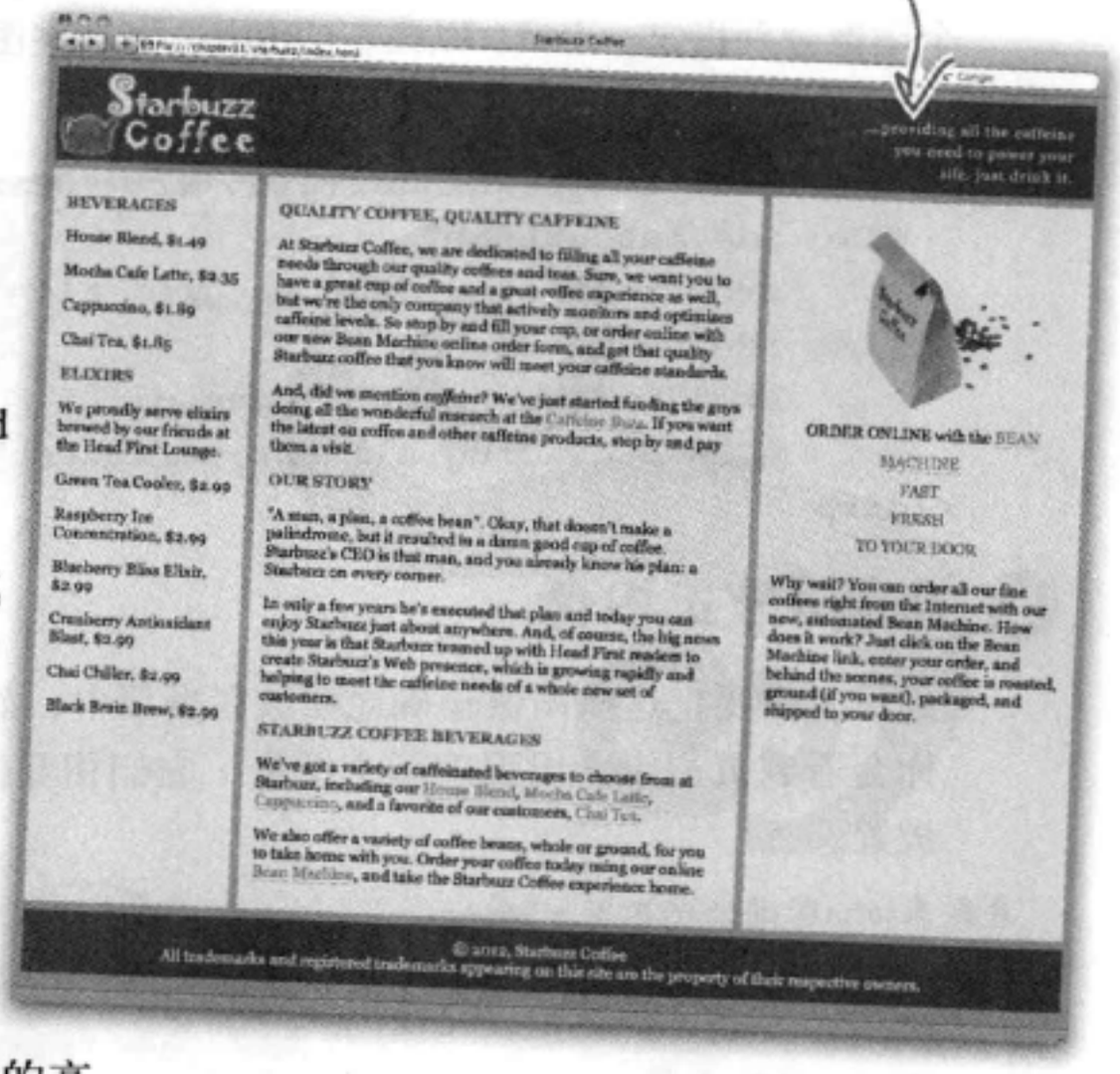
这个CSS很容易，你可能闭上眼睛都能写出来，毕竟这一章你已经有了很多布局经验。编写CSS来修正页眉中的图像。你已经知道要用float，完成下面的填空，补全规则的其余部分，将图像放在正确的位置上。学习后面的内容之前，先对照这一章最后给出的答案检查你的结果。

```
_____________________ {
    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”？

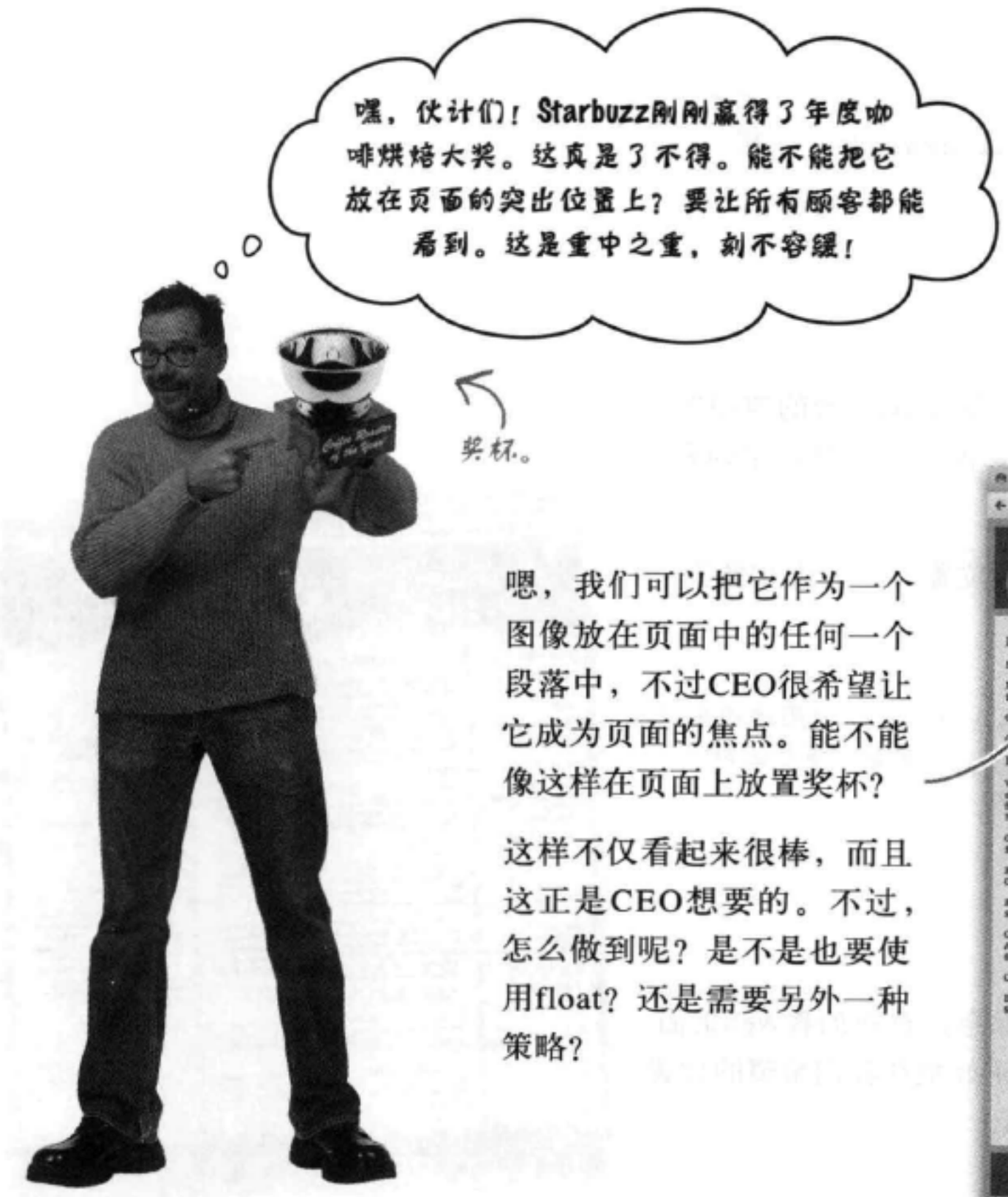
答：因为我们要浮动的图像恰好与页眉中的另一个图像高度相同，都是108像素，所以对于页面中的其他内容来说，再没有空间可以上移和围绕这个浮动元素。两个图像所占的垂直空间相同，所以页面中的其他元素都能稳定在它们自己的位置上。

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。

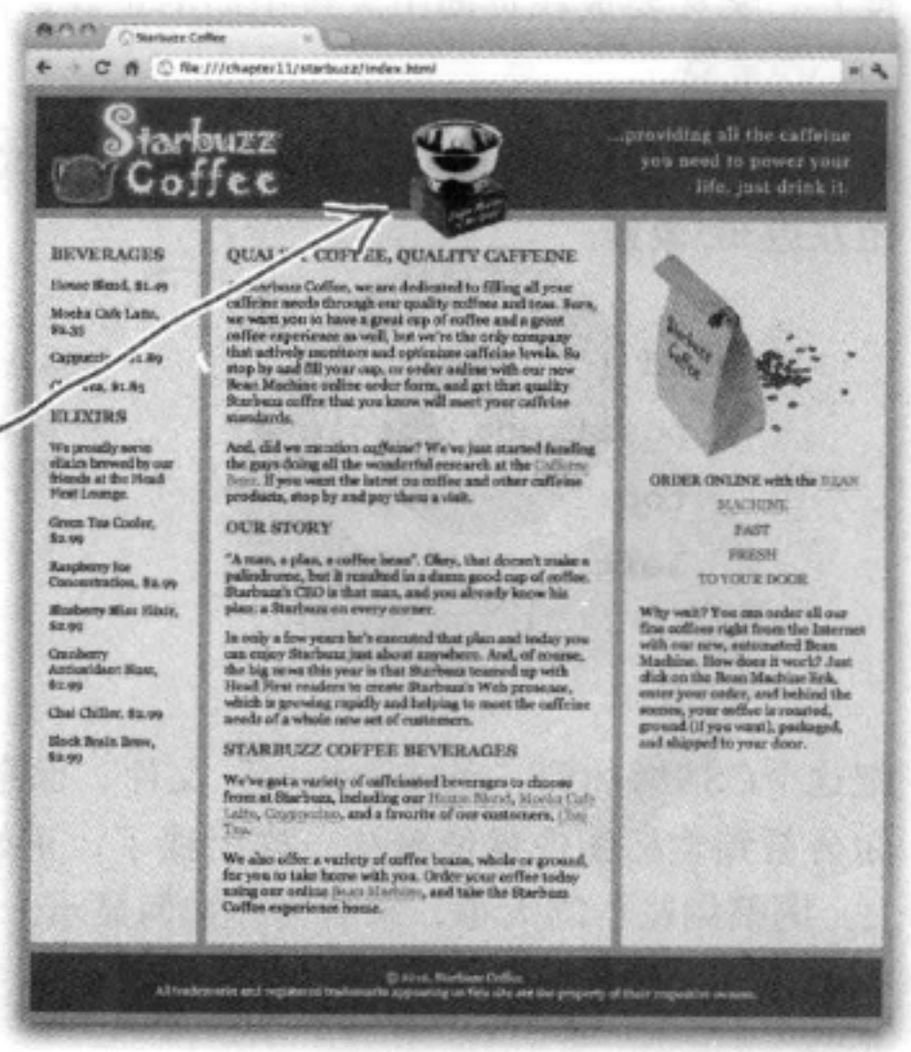
问：能不能使用前面讨论的另外一种布局策略来定位页眉图像？

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。

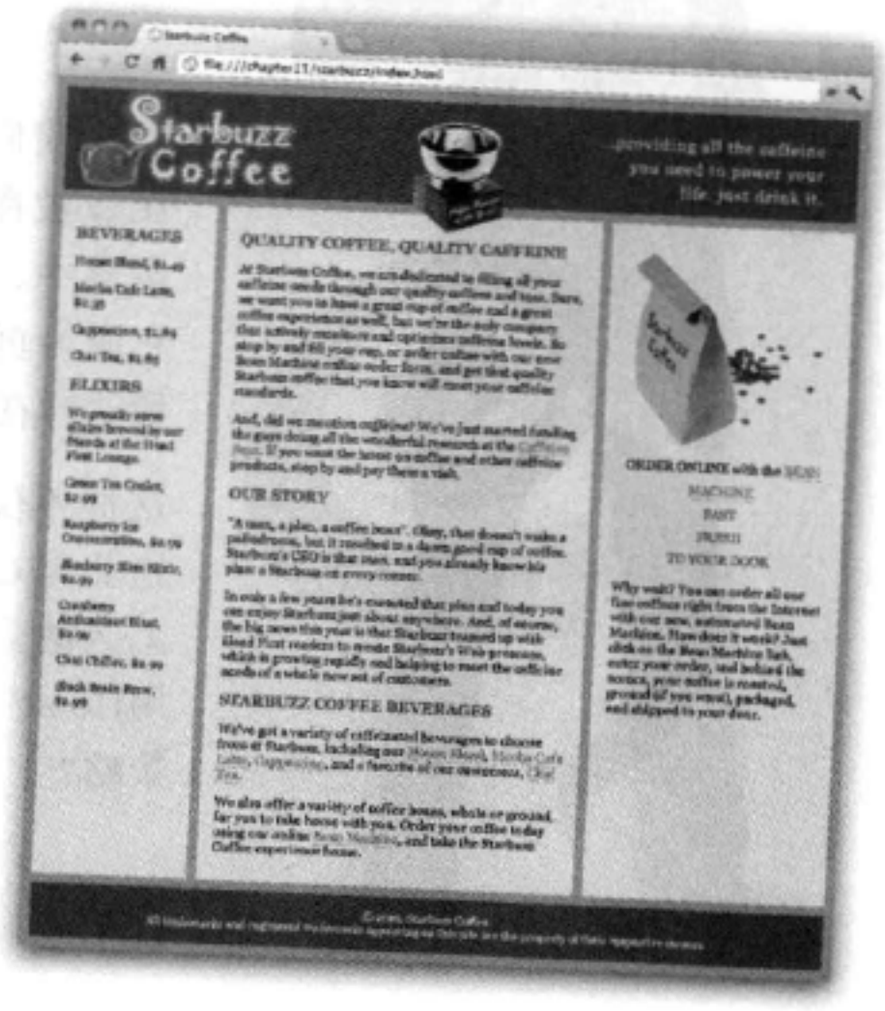
指定奖杯位置

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？

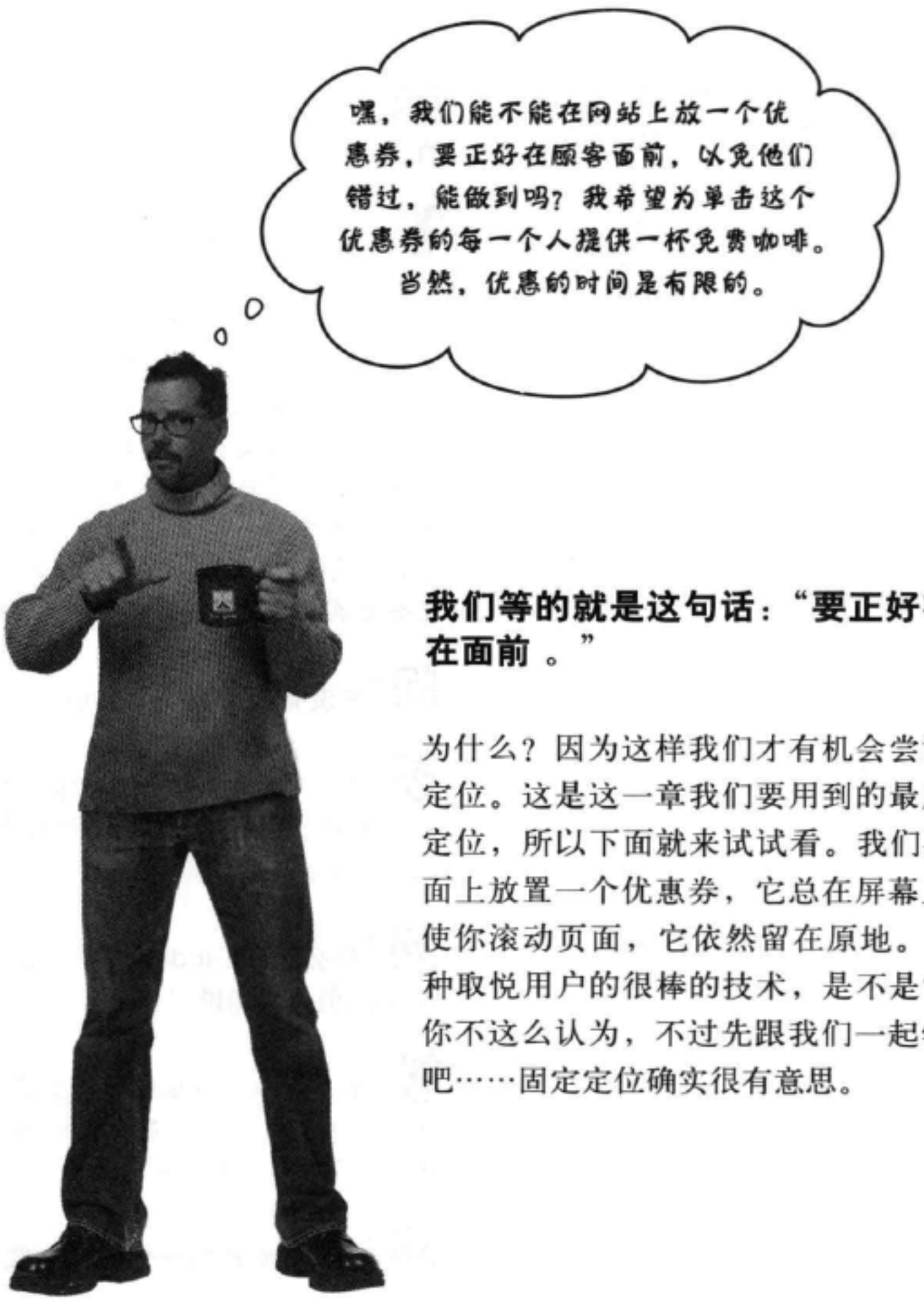
答：有，不过这是一个非常大的数，而实际来讲，你永远也不会用到那么大的z-index值。

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。

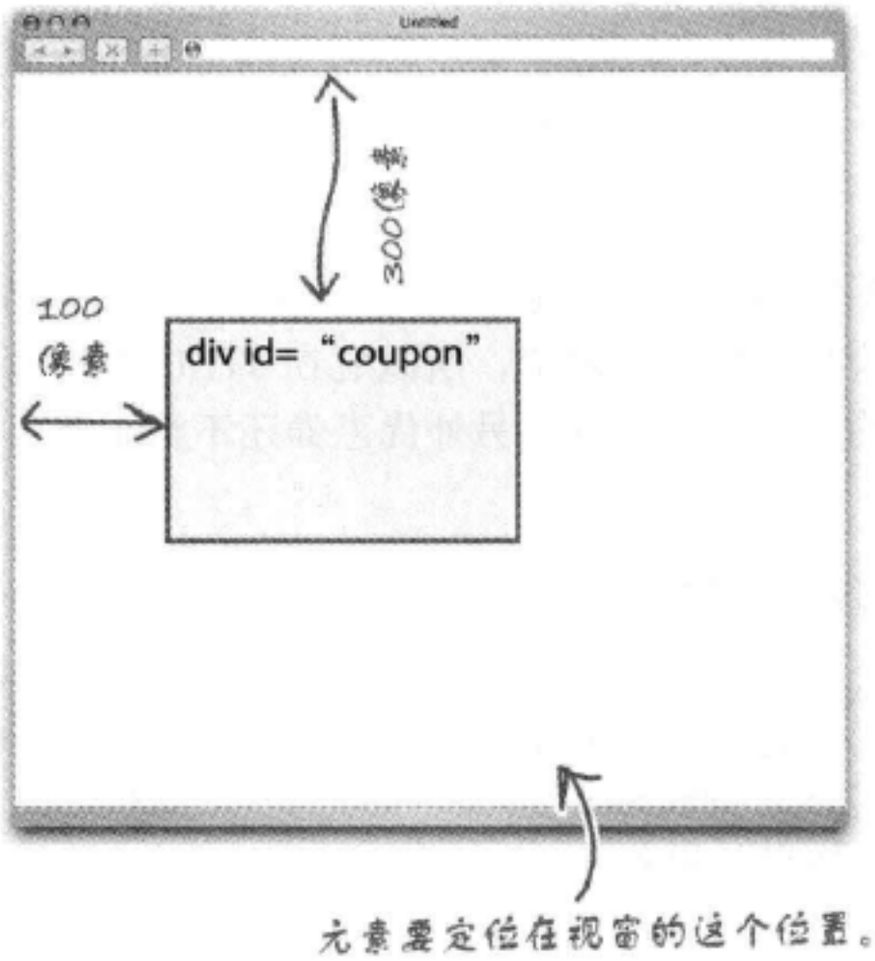
所以，假设有一个 ，id为“coupon”。可以把这个 固定在距视窗上方300像素，距左边100像素的一个位置上，就像这样：

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;
}
```

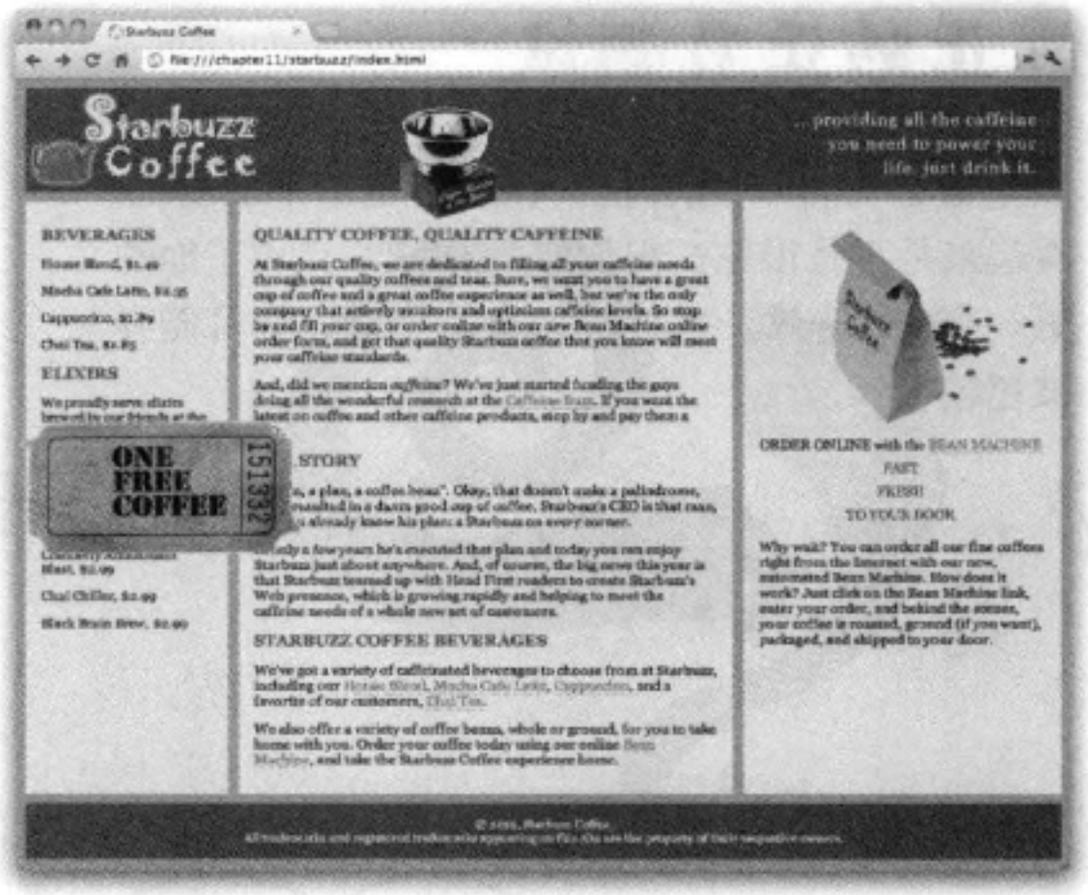
可以设置优惠券的固定位置，距视窗上边350像素，它的左边紧挨着视窗边界放置。所以需要指定距左边0像素。

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。



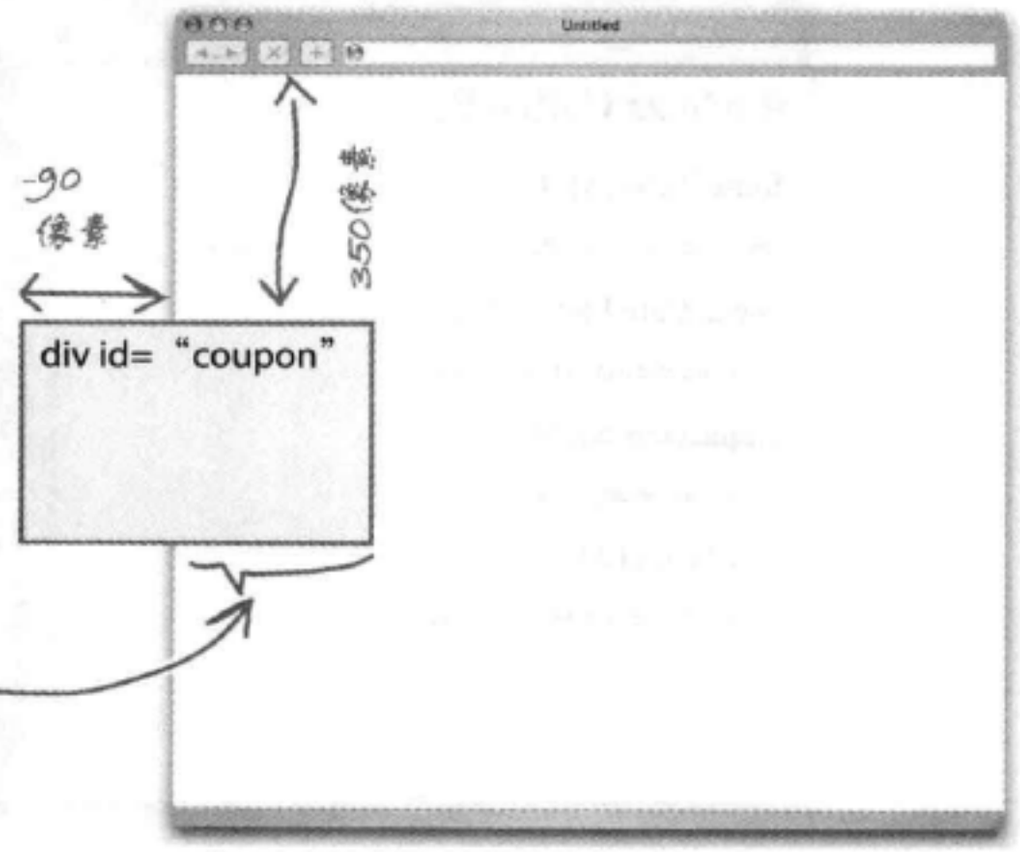
这看起来很不错，不过如果优惠券能偏到左边，它会显得更时髦，这样一来，它看起来就像是从视窗一边飞出的。为了达到这个目的，可以进入我们的照片编辑软件，去掉图像的左边来创建这种效果。或者也可以使用一个负偏移量，使图像的左边定位在比视窗边界更左的位置。好吧，就这么做。

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。

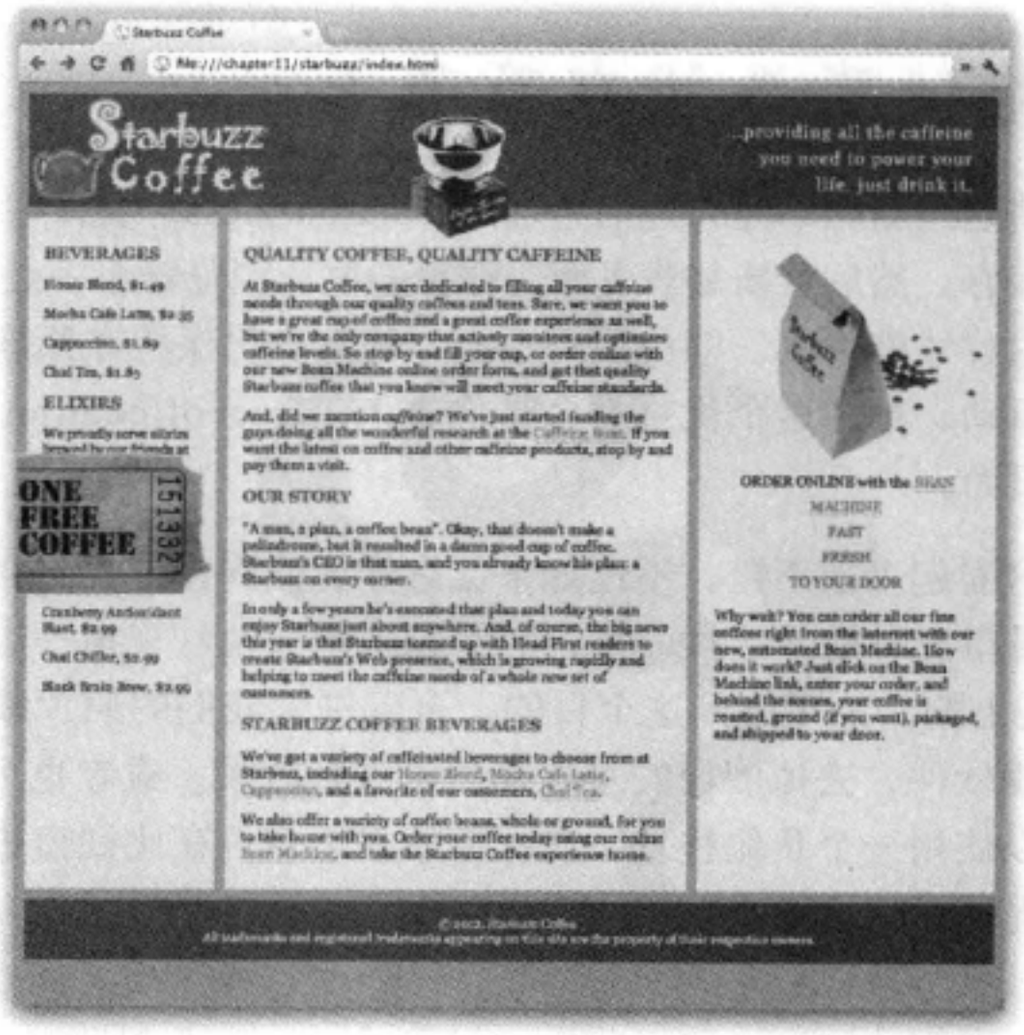


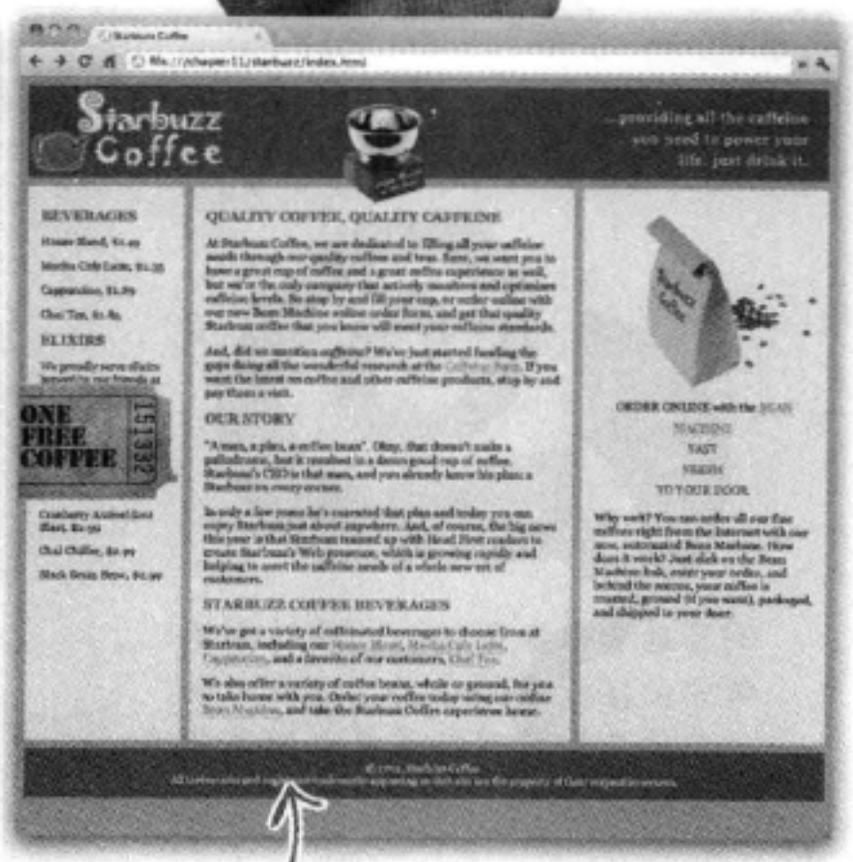
浏览器会很高兴地为你把这个图像定位在视窗的左边，这样一来，你只能看到仍留在屏幕上的那部分图像。

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。

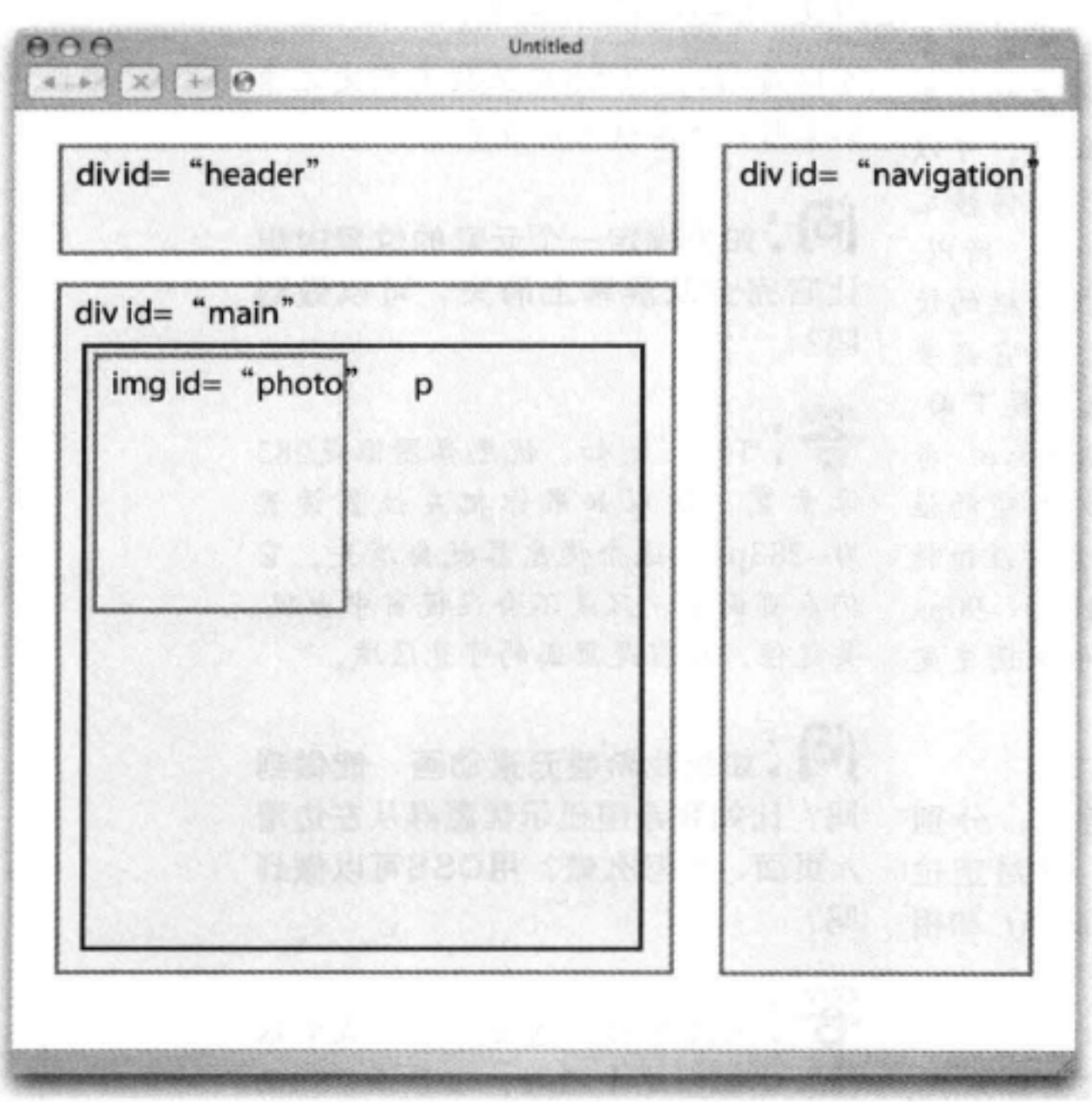




哇！真是天壤之别！

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？

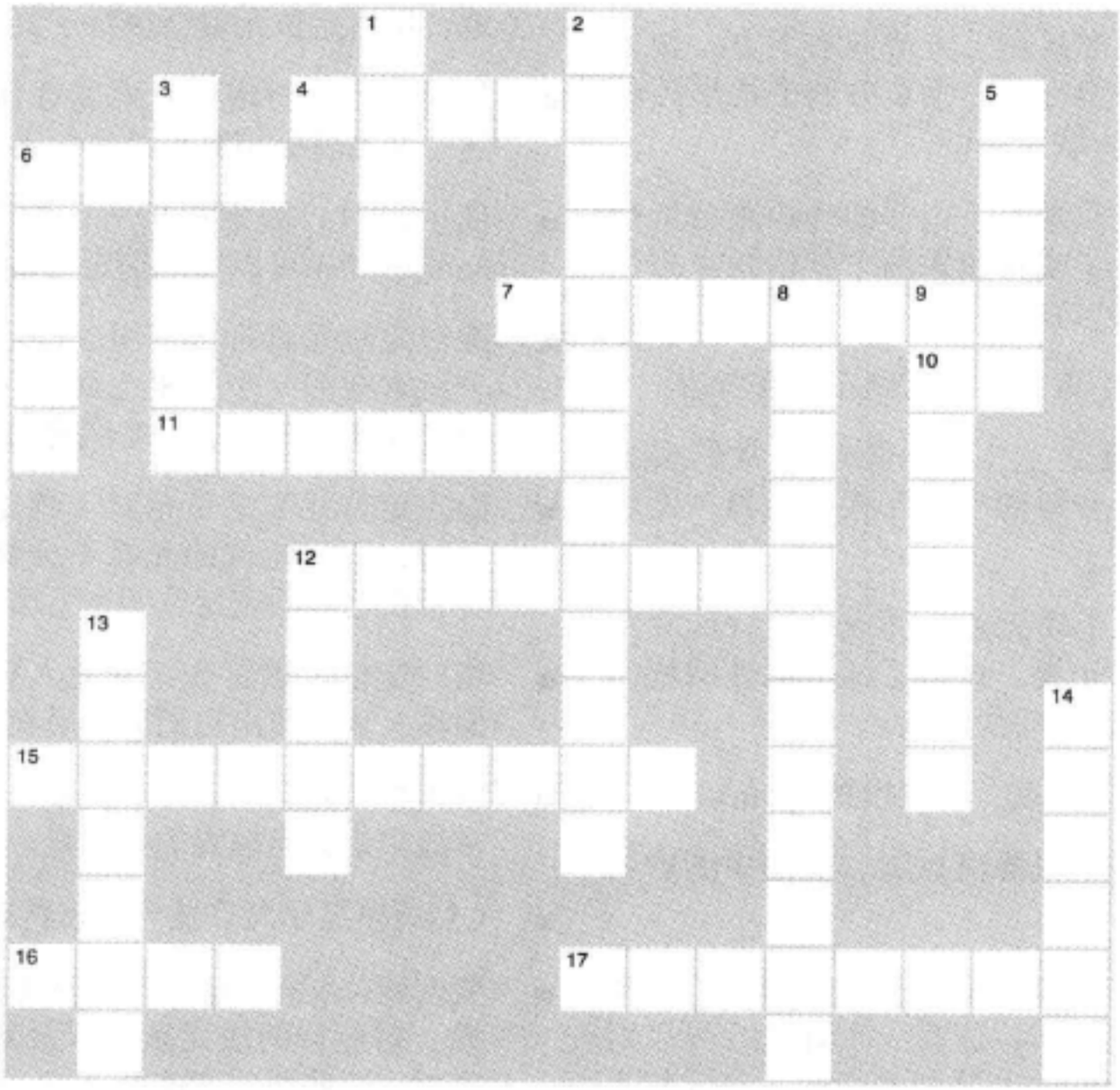
答：说实在的，确实可以，我们很高兴你能问这个问题。这涉及CSS动画，超出了这本书的范围，不过要知道，CSS3为元素引入了基本动画，提供了变换和过渡特性，这让我们Web开发人员欣喜不已。这些特性功能还很有限，不过，利用CSS动画确实可以完成一些很酷的效果。如果你想要的效果用CSS做不到，就要用到JavaScript了，这又是另一个内容。我们会在附录中对CSS变换和过渡做一个简要的介绍，吊吊你的胃口。

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。

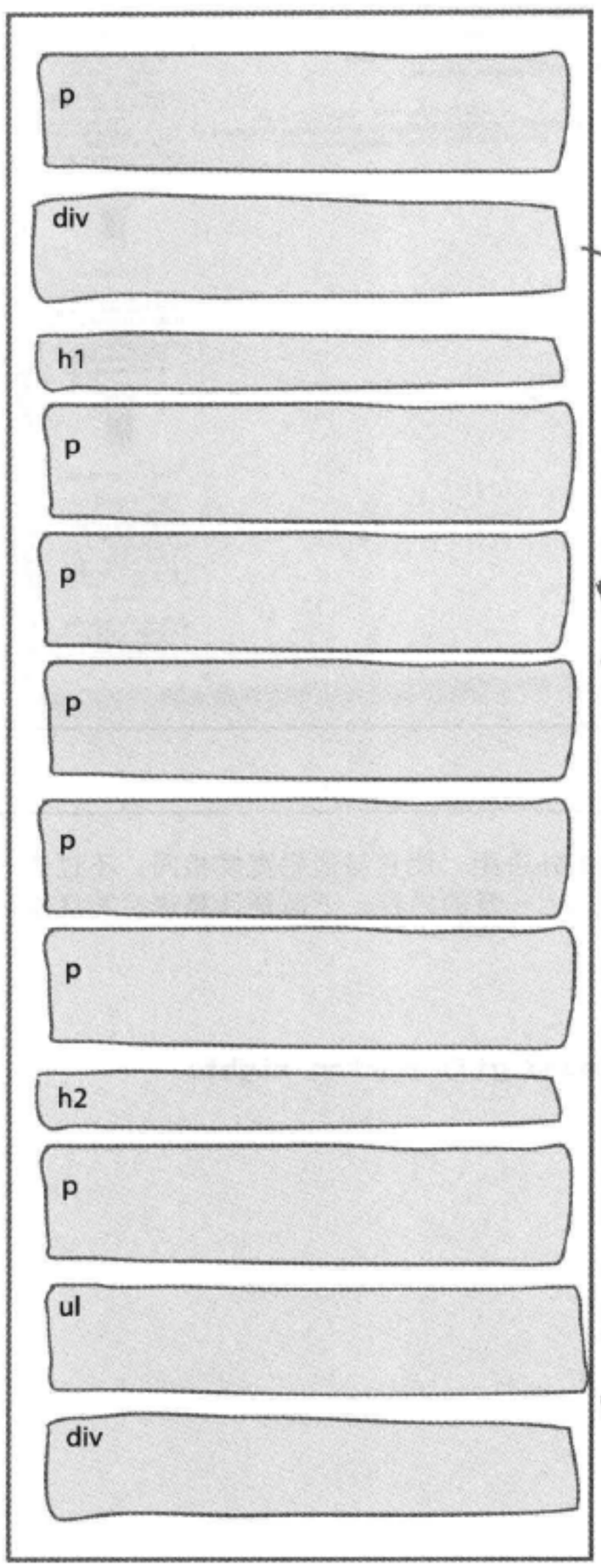
纵向

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

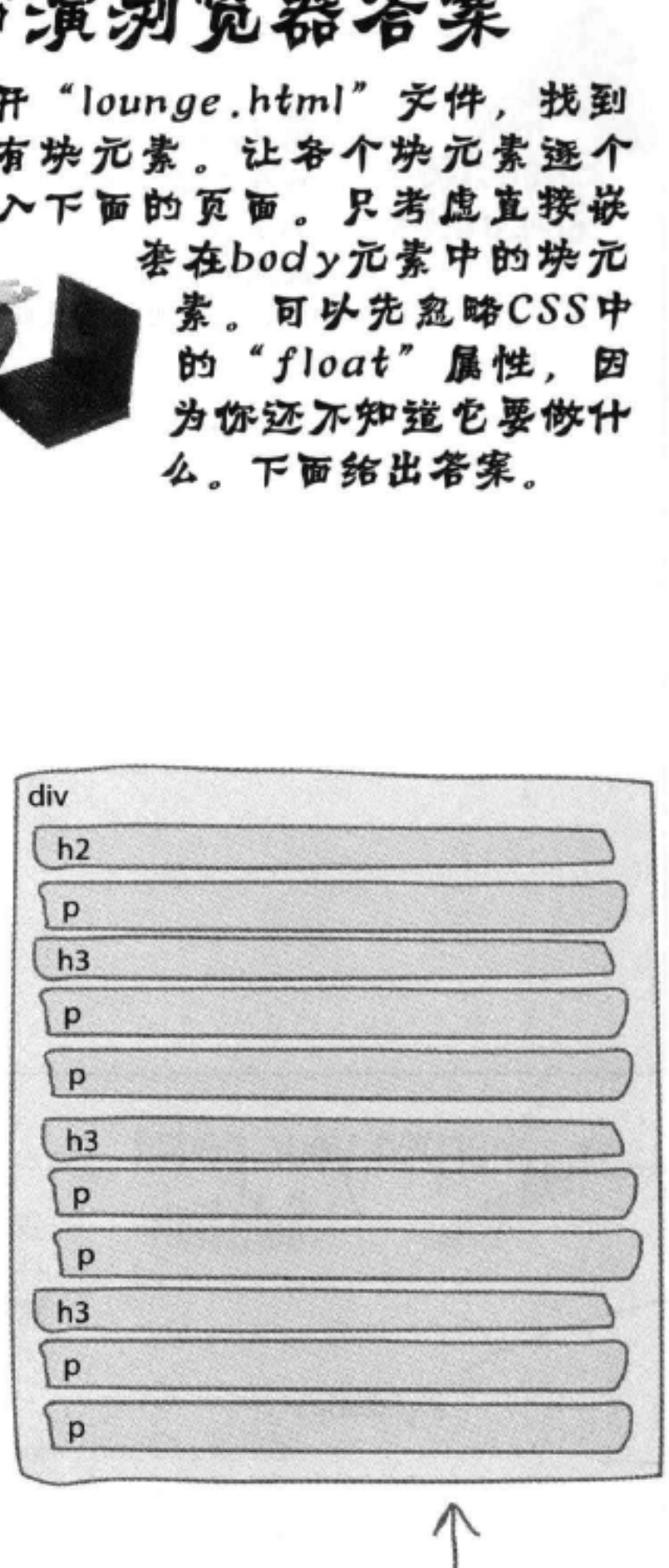
打开“lounge.html”文件，找到所有块元素。让各个块元素逐个流入下面的页面。只考虑直接嵌套在body元素中的块元素。可以先忽略CSS中的“float”属性，因为你还不知道它要做什么。下面给出答案。

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 。

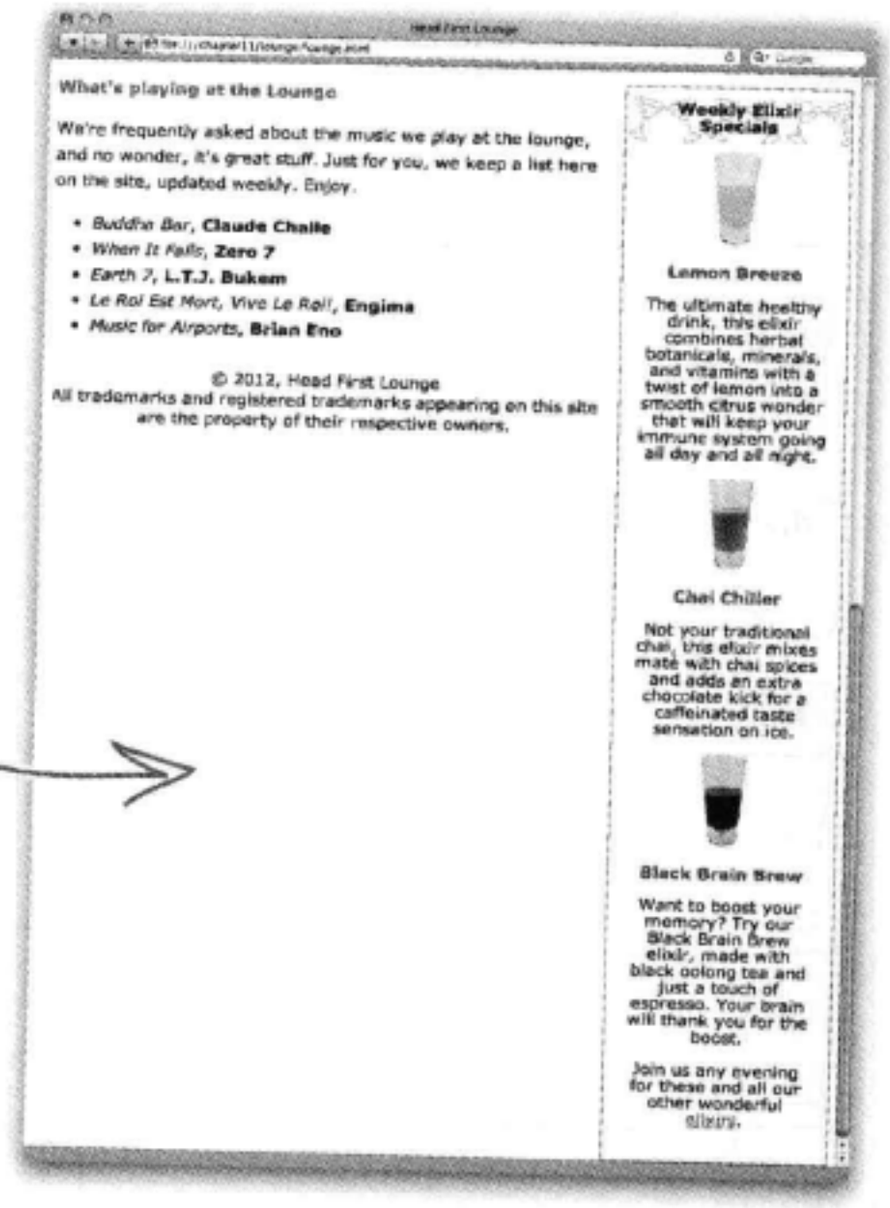


我们没有要求你画出嵌套的块元素流，不过如果你想更进一步，这里给出了elixirs 中的元素流。

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。

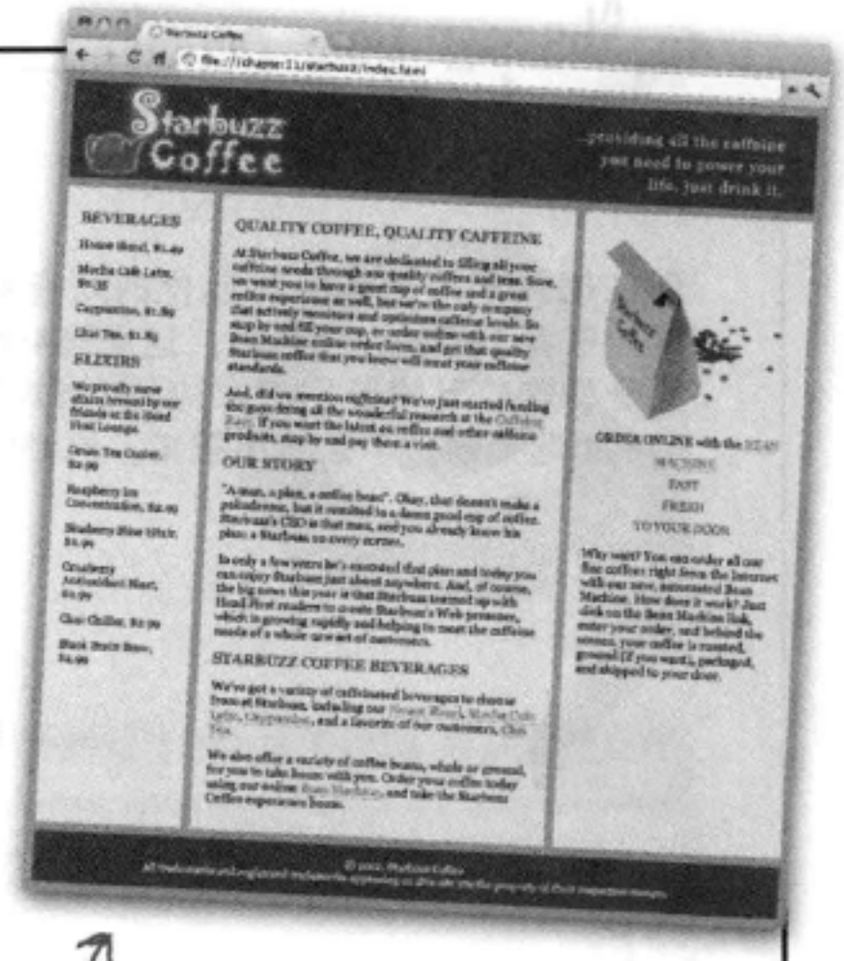
```
#sidebar {
        background: #efe5d0 url(images/background.gif) bottom right;
        font-size:  105%;
        padding:    15px;
        margin:     0px 10px 10px 10px;
        width:      280px;
        float:       right;
}
```

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">
      ...
```

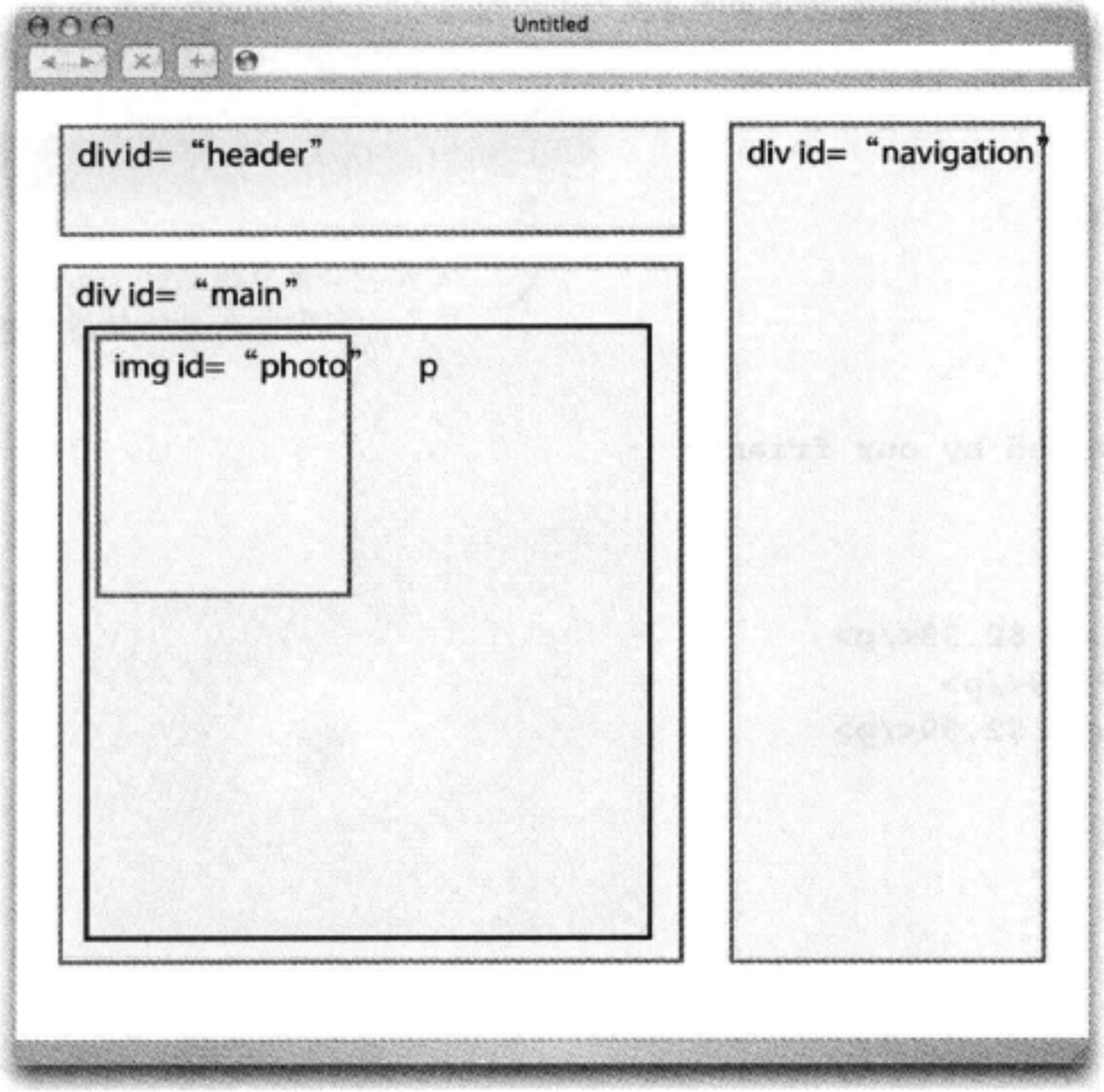
新的CSS……你需要完成这个CSS！

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___;
}
```

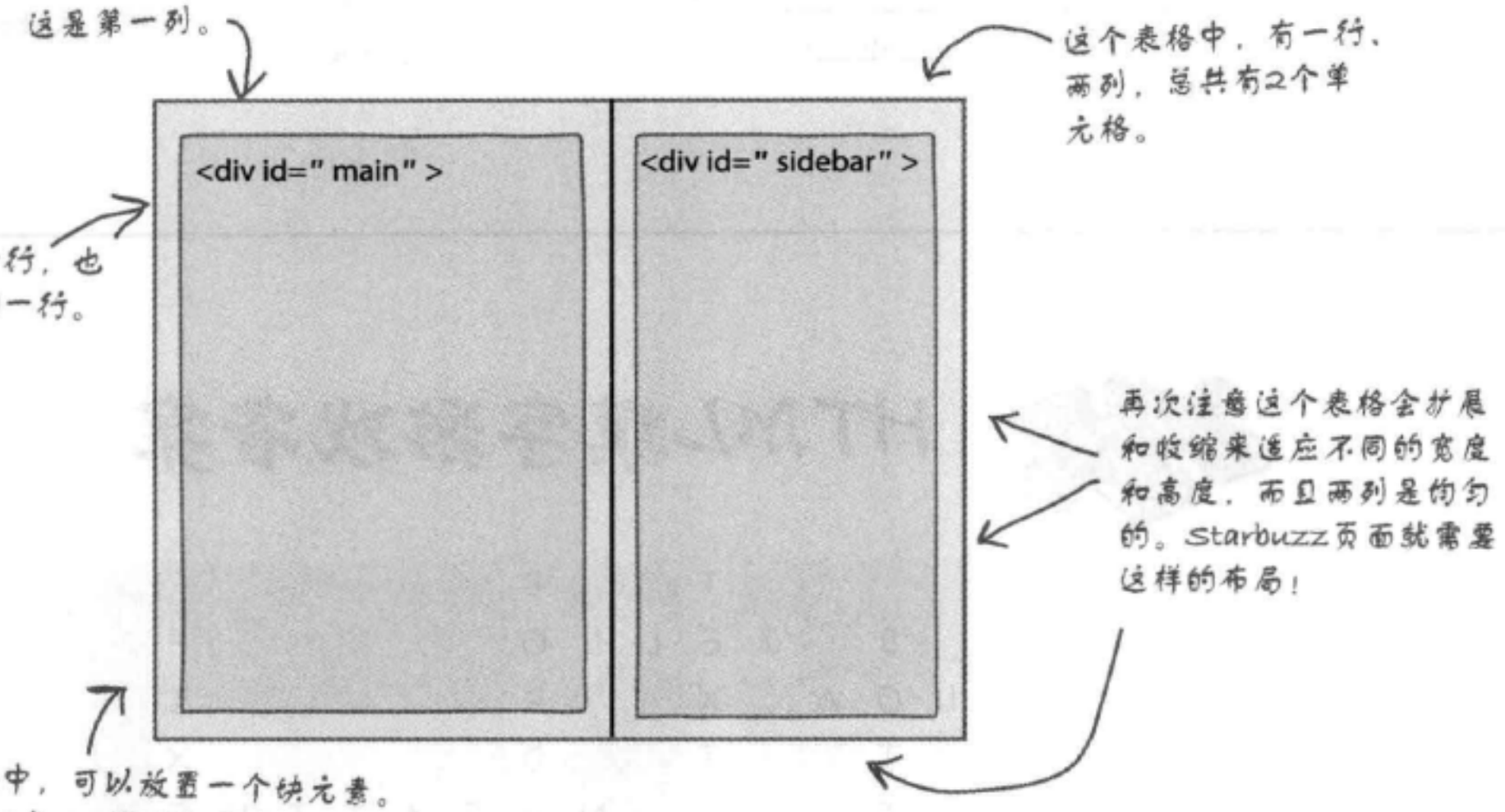
如果你愿意，这里还可以使用#headerSlogan作为选择器。

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？

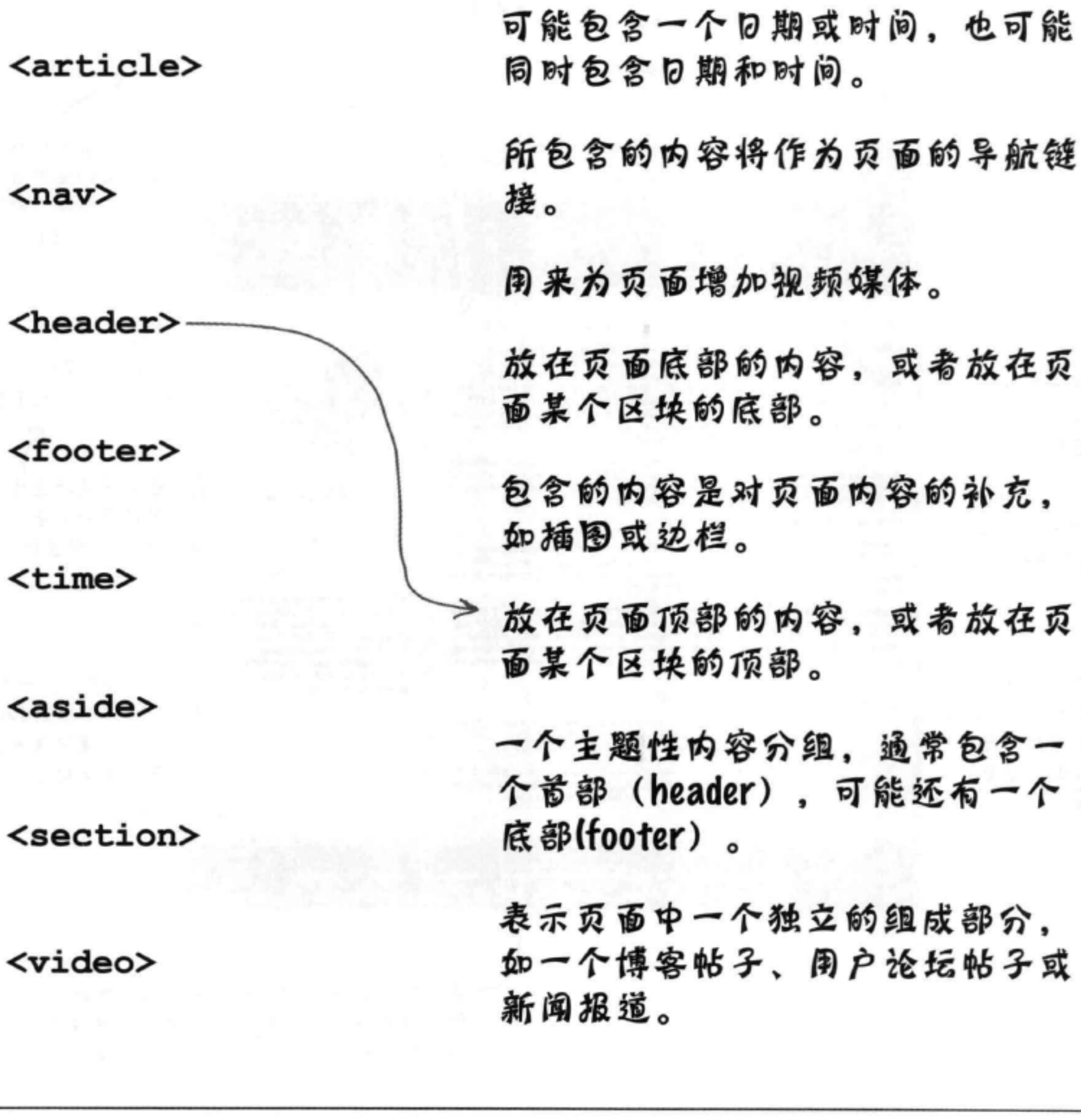
新的HTML5标记正是考虑到这一点，首先明确了人们如何利用 建立页面结构，相应地提供了更特定、更适合某些结构的标记。可以看到，浏览器（或搜索引擎，或屏幕阅读器）看到页面中的id="navigation"时，它们根本不知道这个 将用于导航。还不如把它叫做id="goobledygoop"。

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）：

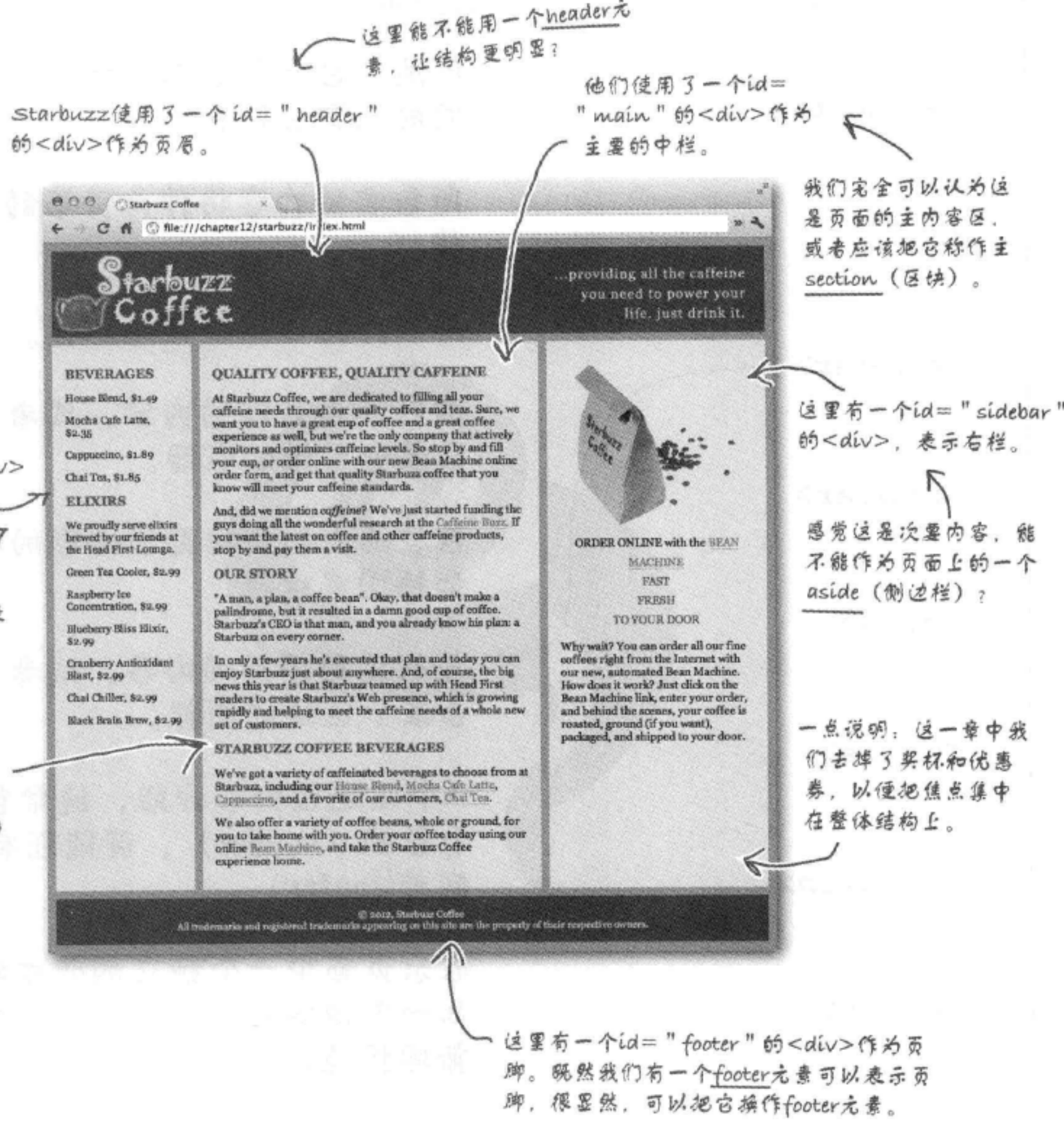
WHO DOES WHAT?

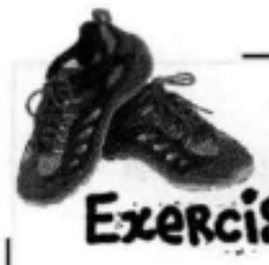
没错，我们当然可以直接介绍那些新的HTML5元素，不过，由你自己找出来不是更有趣吗？下面在左边给出了新元素（这些肯定不是全部的新元素，不过你会发现比较重要的新元素都在这里），请将各个元素与右边的相应描述连线：



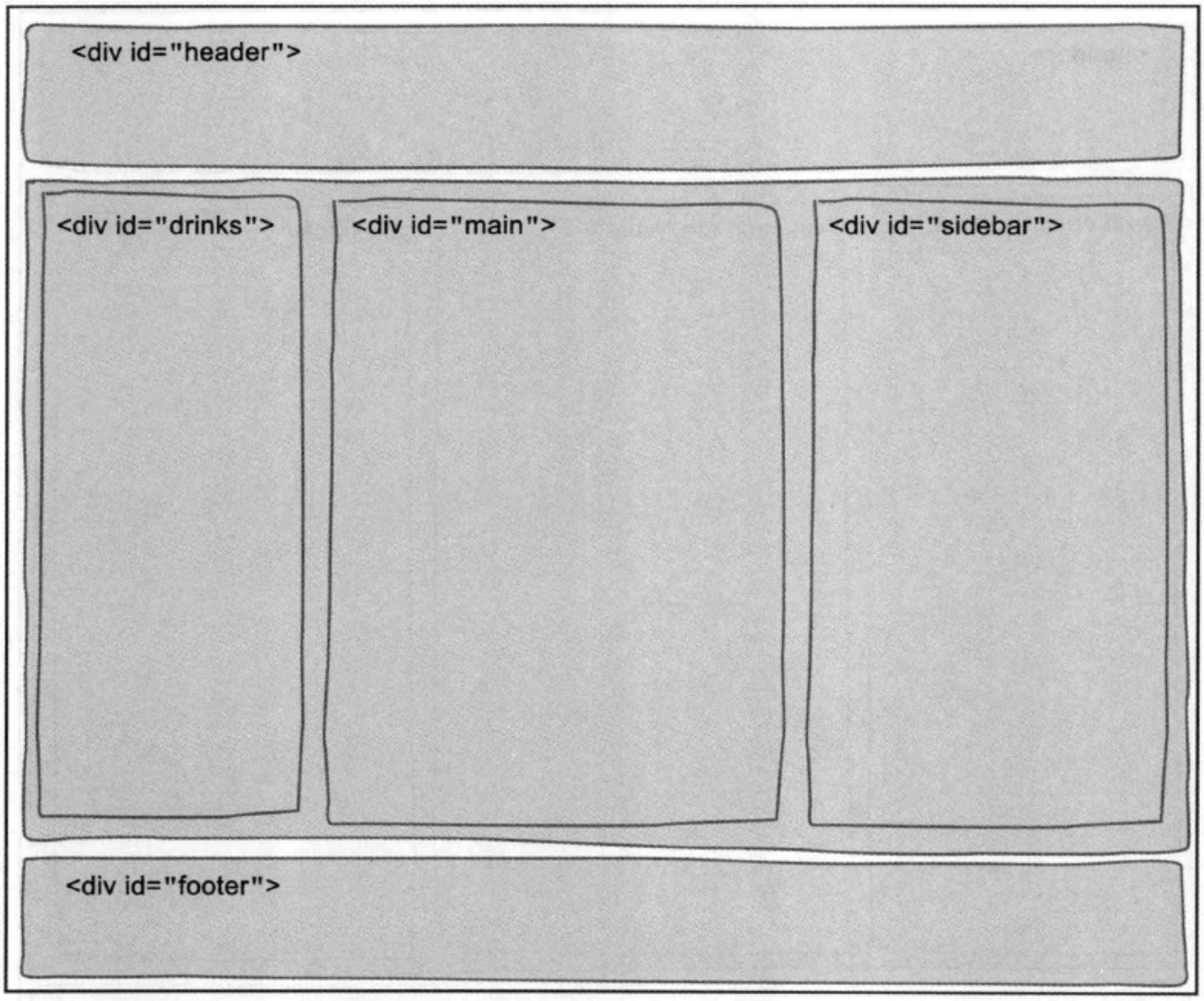
现代Starbuzz

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：





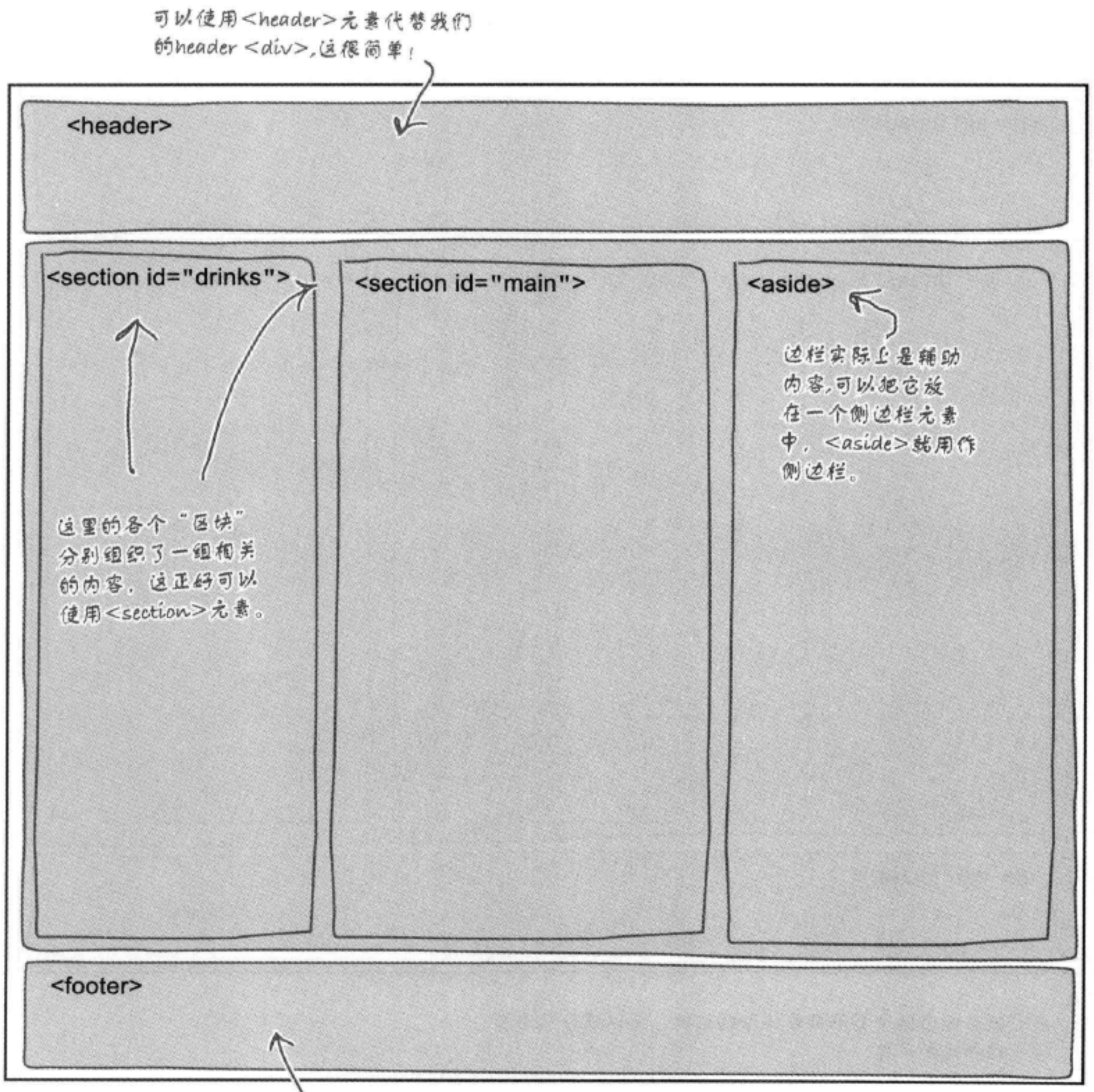
到目前为止，你对新的HTML5元素已经有所了解，利用你掌握的全部知识，看看能不能修改Starbuzz页面来利用这些新元素。只需要在这一页上简单标出来。



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>。

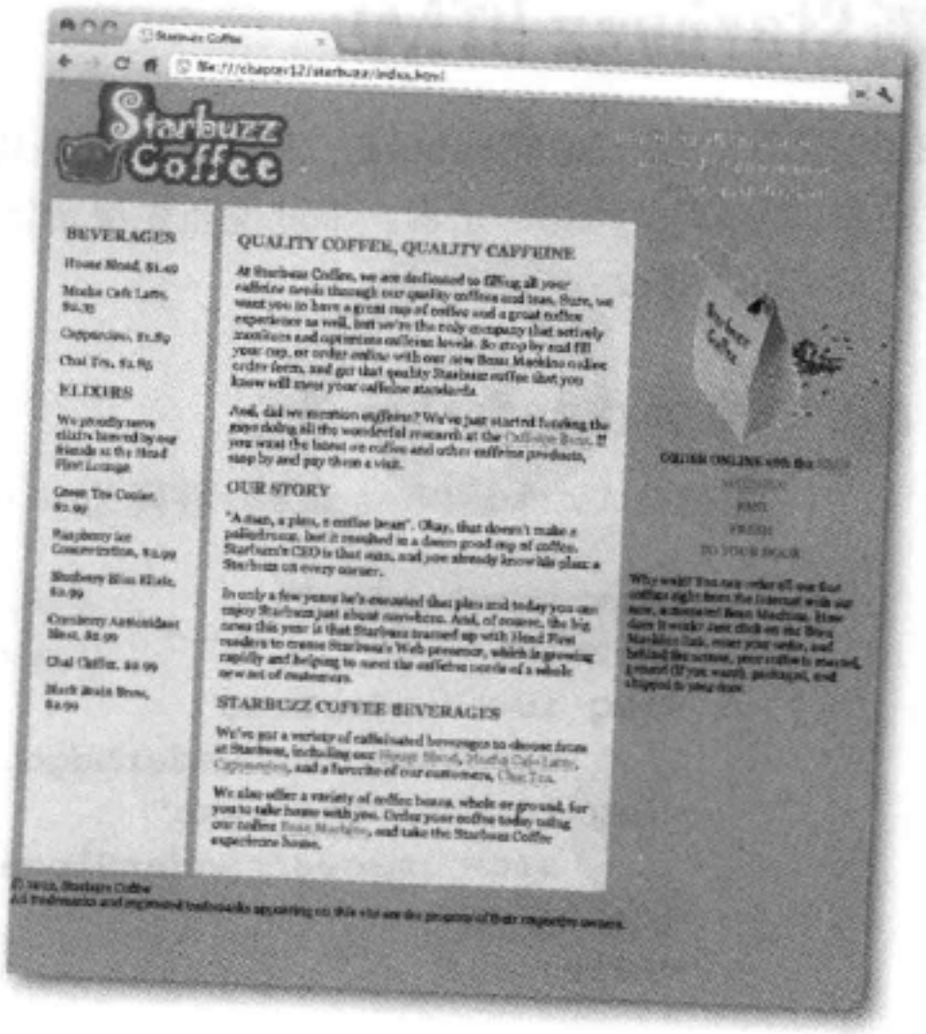
现在把"sidebar" 改为一个<aside>元素：

```
<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。

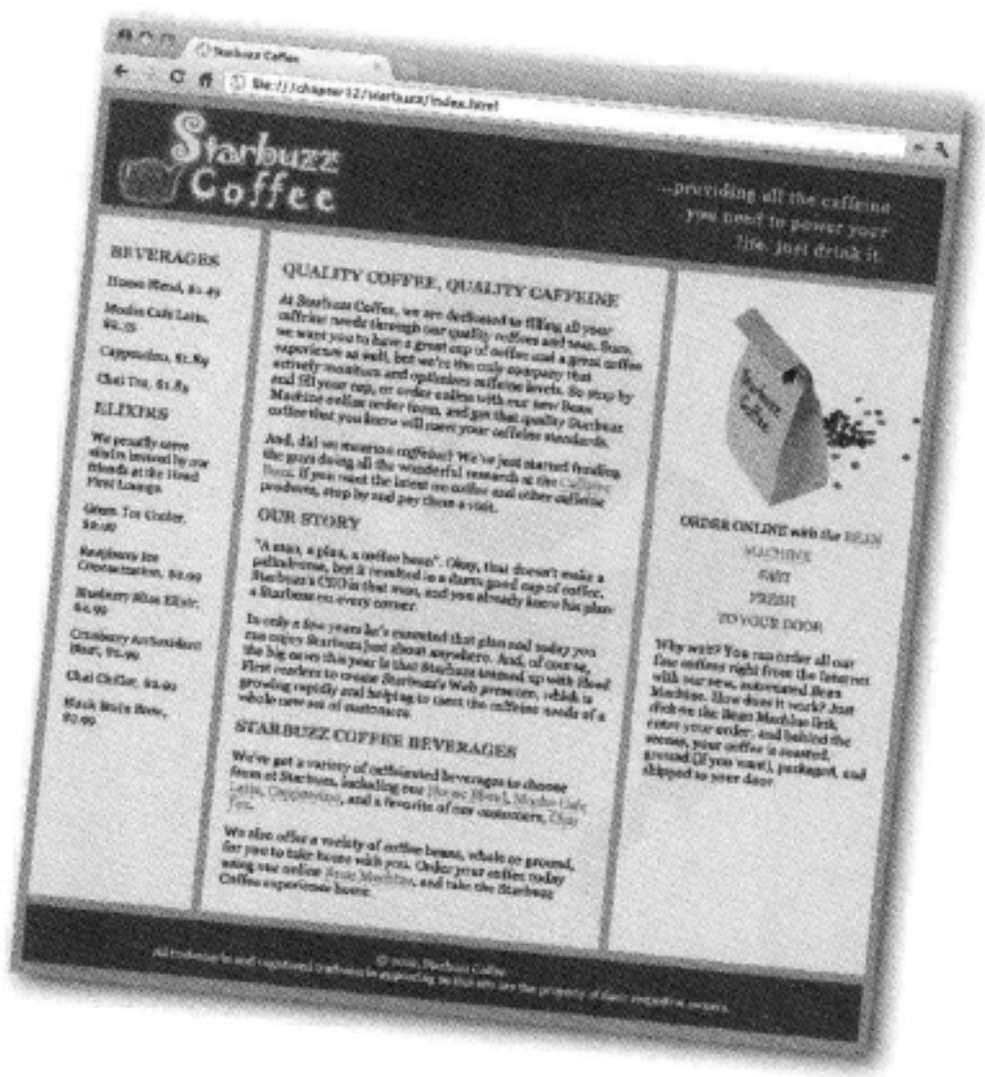
这里也需要修改，原来针对一个id为"sidebar"的元素，现在要改为选择一个aside元素。

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。

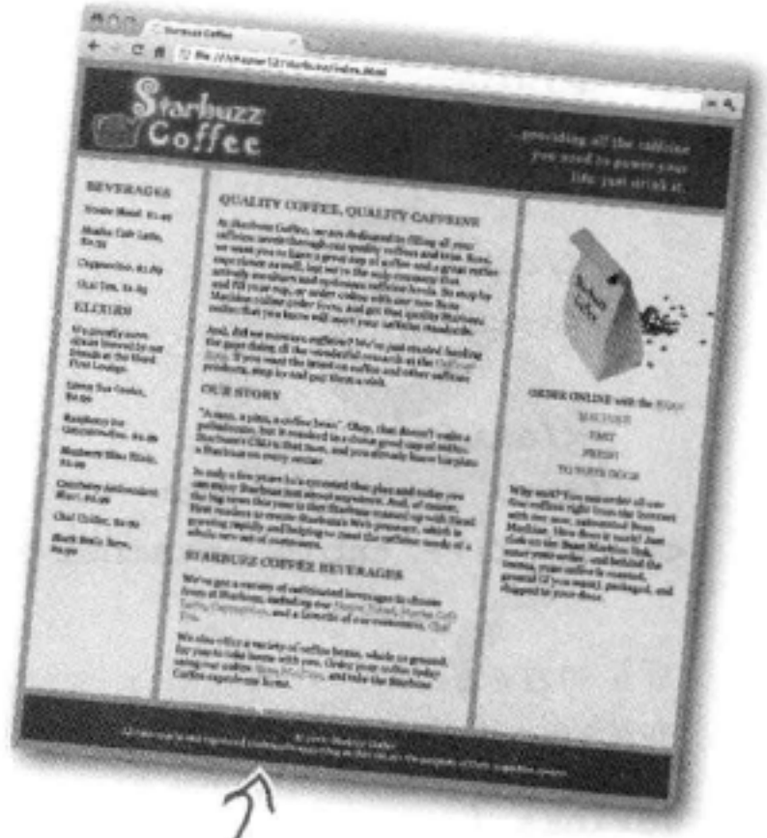
下面是我们的答案：

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。



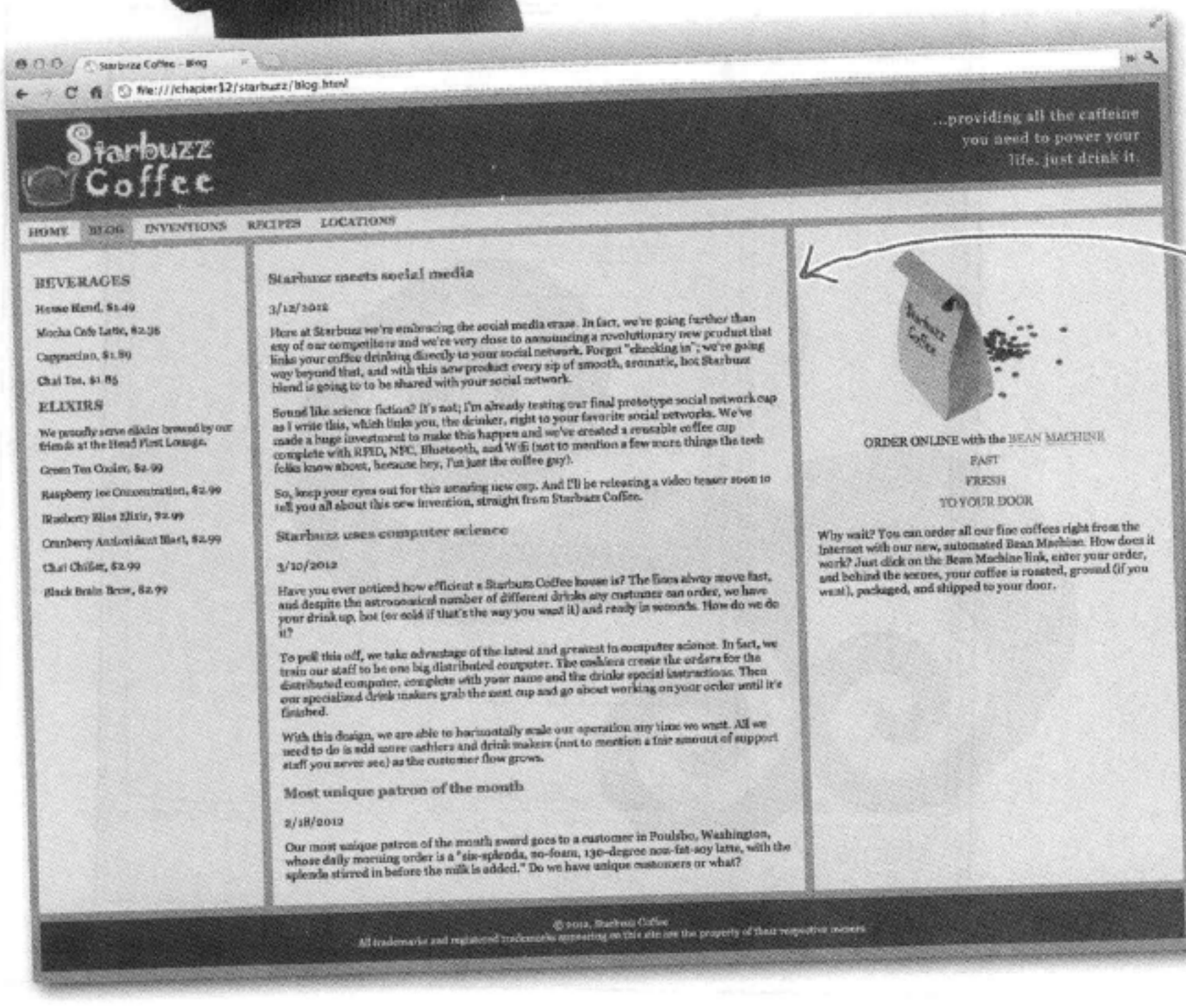
这是现在的页面！看起来完全一样，不过你已经知道现在换成了新的HTML5元素，是不是感觉好多了？

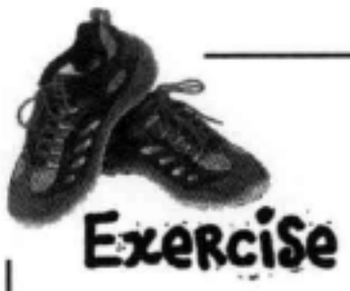
为这两个区块更新的CSS。

```
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：



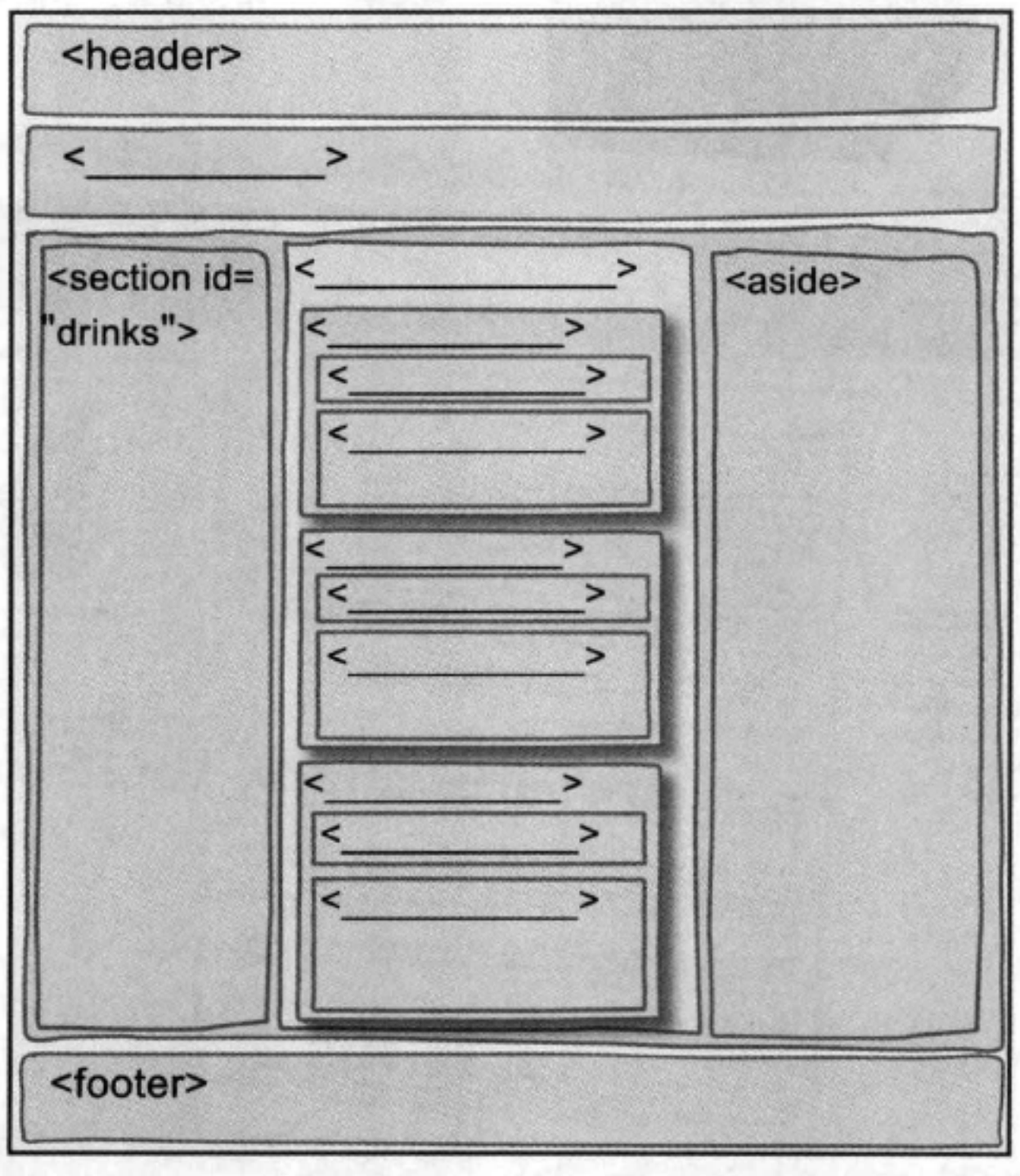


你的任务是选择你认为最适合建立这个新博客的元素。在下面的图表中填空，标出你会选择哪些元素。需要说明，每个博客帖子都有一个标题，另外至少有一个文本段落。

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>

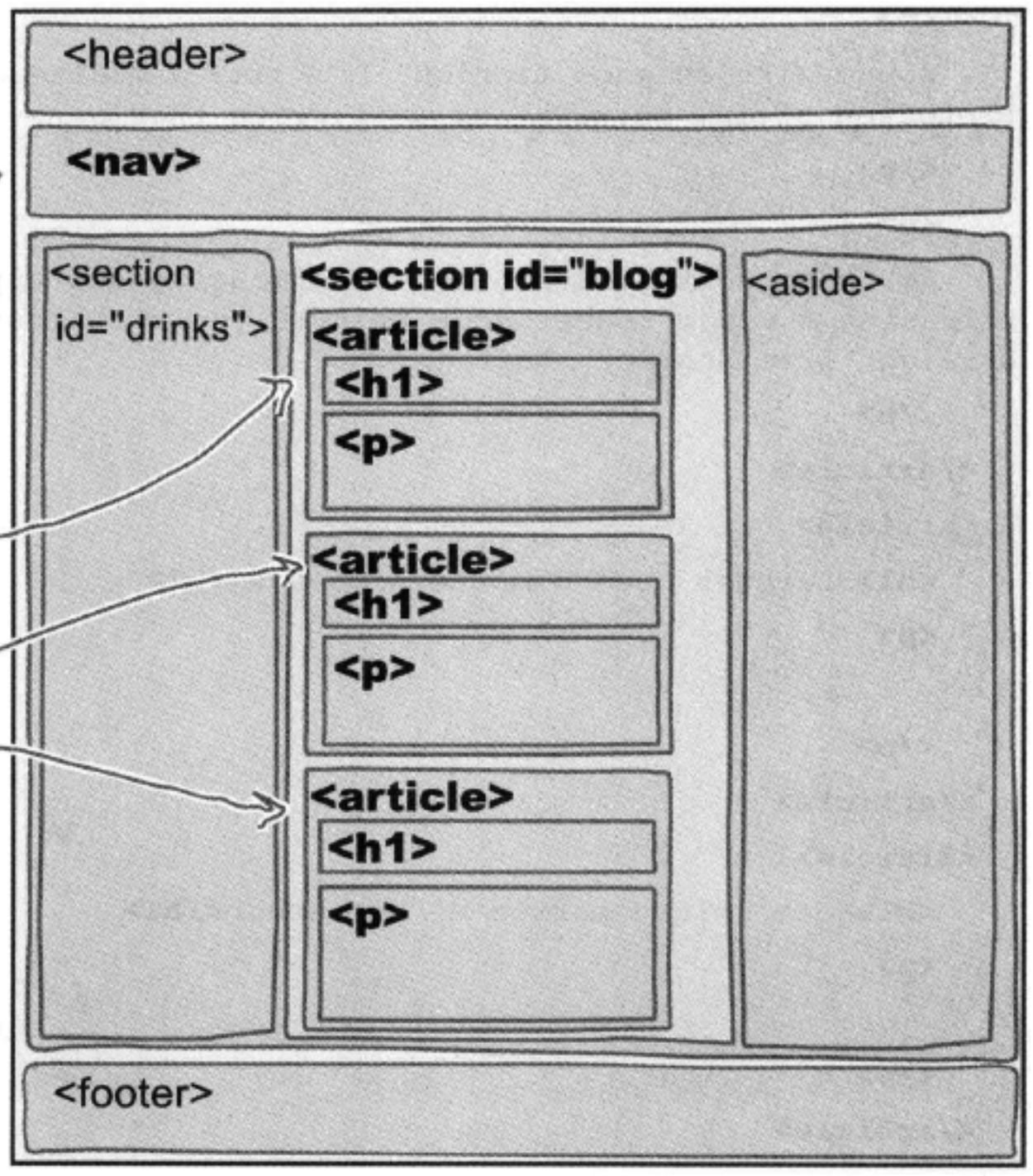
<h1>
<p>

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;
}
```

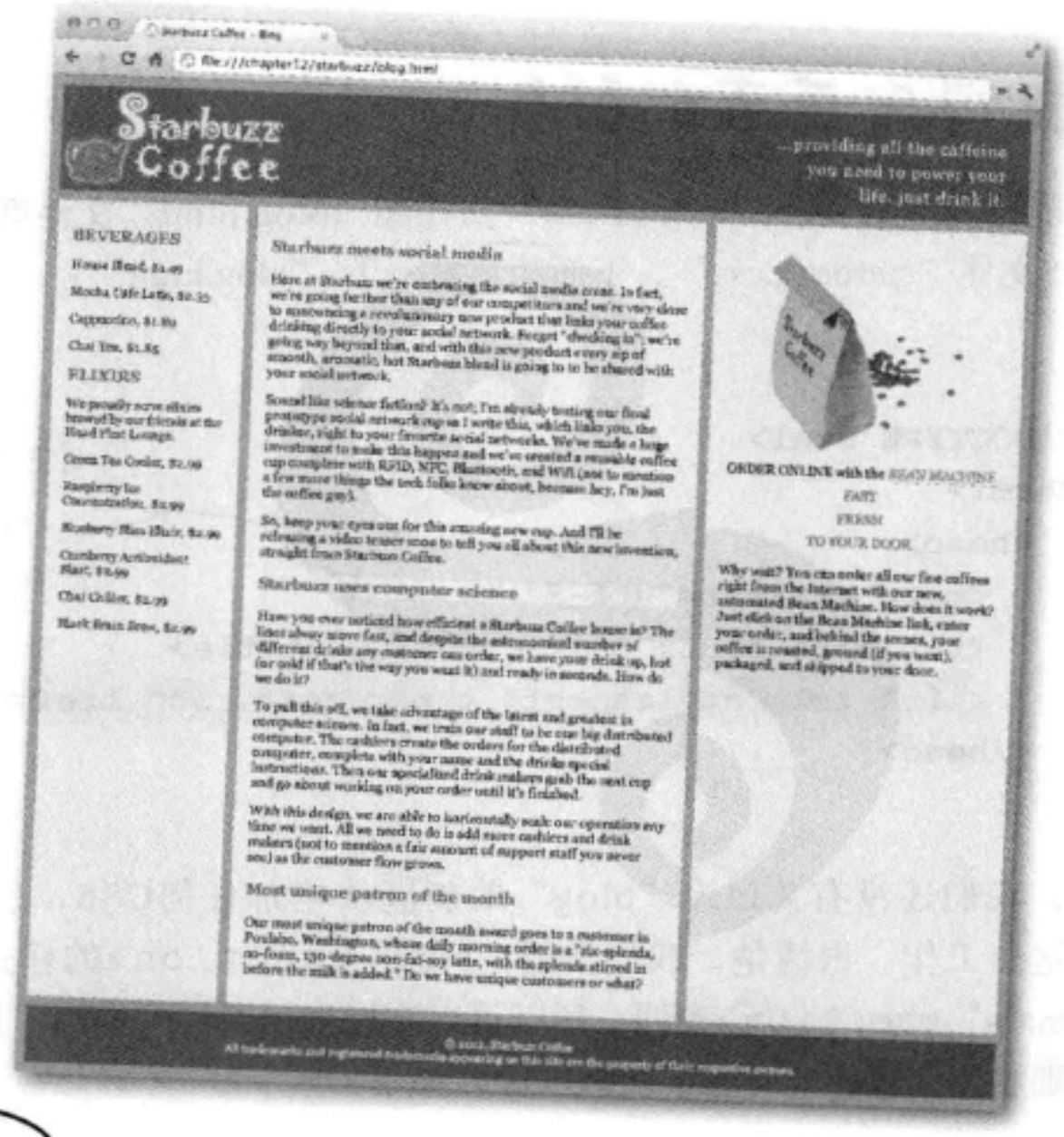
通过使用两个选择器（用逗号分隔），可以对这两个<section>元素使用相同的规则。这说明，所选择的两个元素都要应用这些属性。

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？

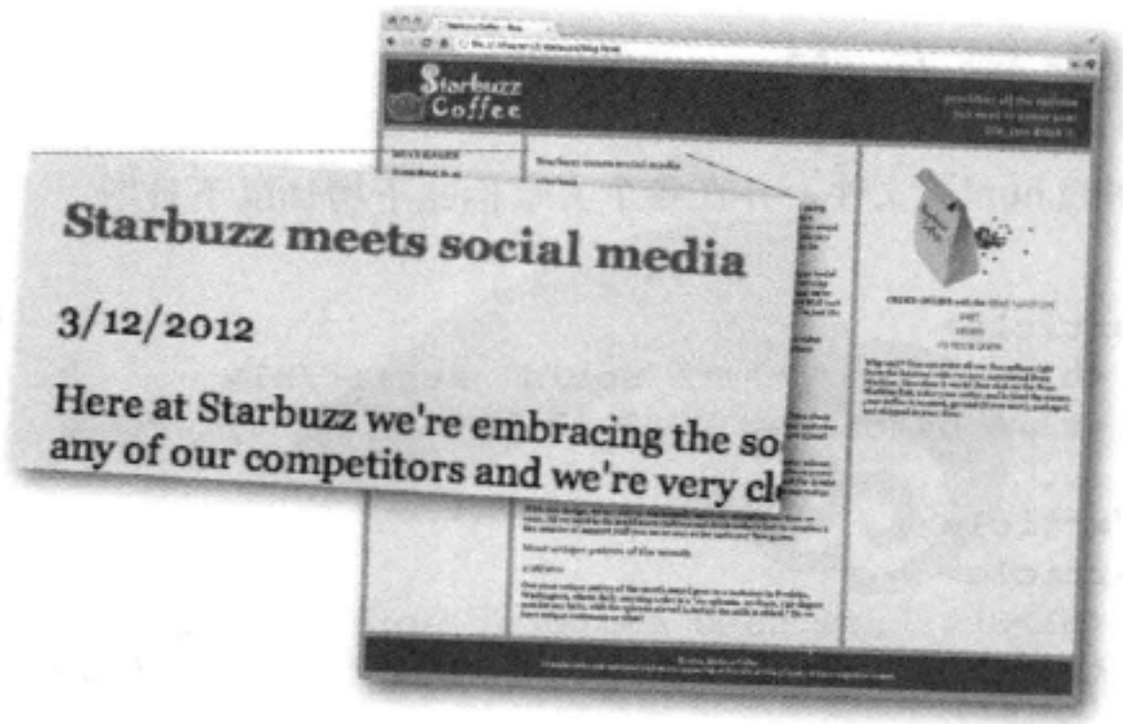
是的，这确实让人很困惑。可以明确告诉你，关于这个问题并没有透彻明了的答案。实际上，使用<article>和<section>的方法有很多。不过，通常可以这样来考虑：使用<section>可以把相关的内容分组在一起，另一方面，要用<article>包含独立的内容，如一个新闻报道、一个博客帖子或者一个简短的报告。

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)

为博客增加<time>元素

编辑“blog.html”文件，并在各个文章标题下增加以下日期：

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

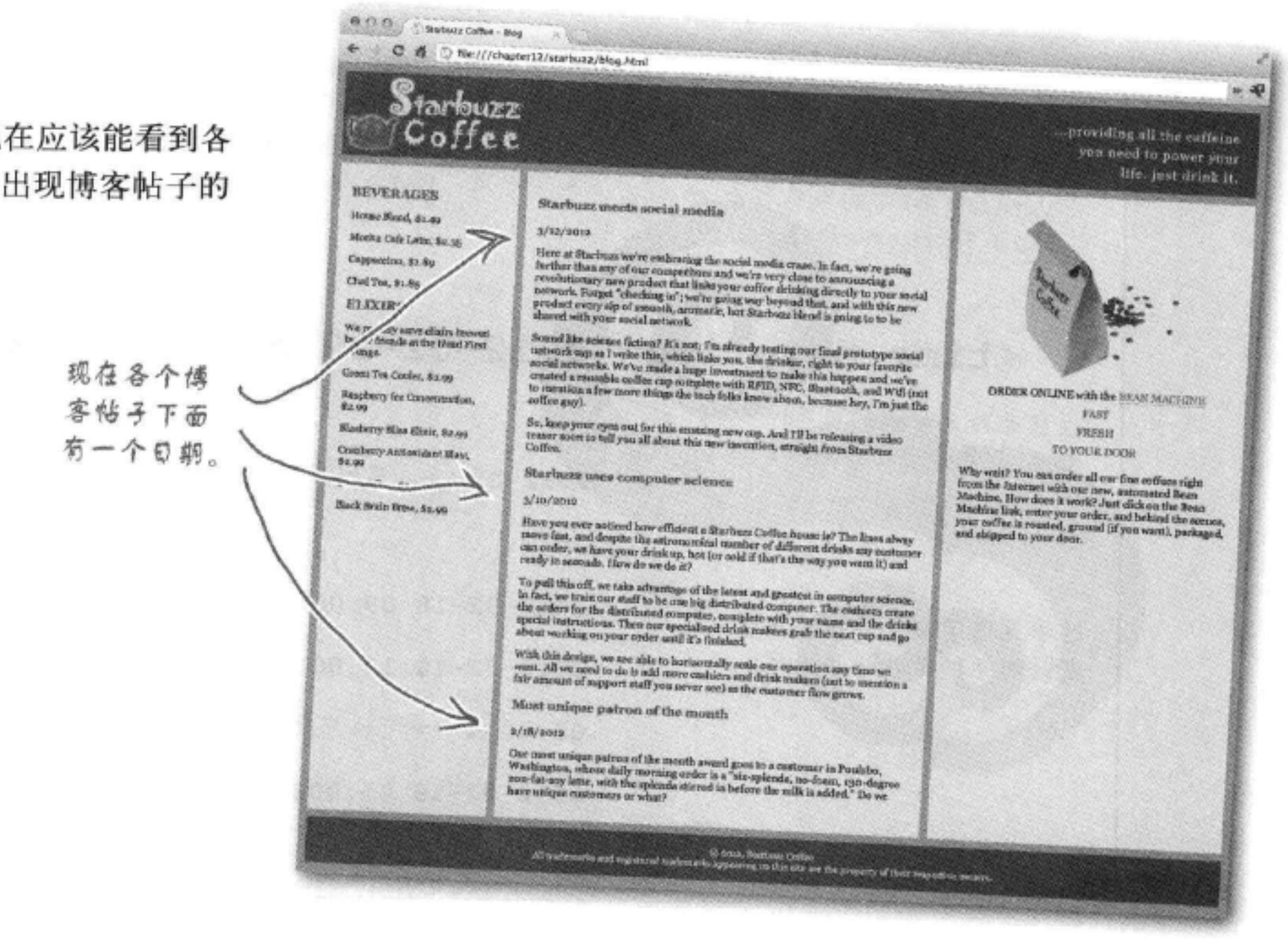
在各个标题下增加了一个<time>元素。

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

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

测试博客

再来测试这个博客，现在应该能看到各个博客帖子标题下面会出现博客帖子的日期。



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

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

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

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

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

如何增加更多<header>元素

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

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

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

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

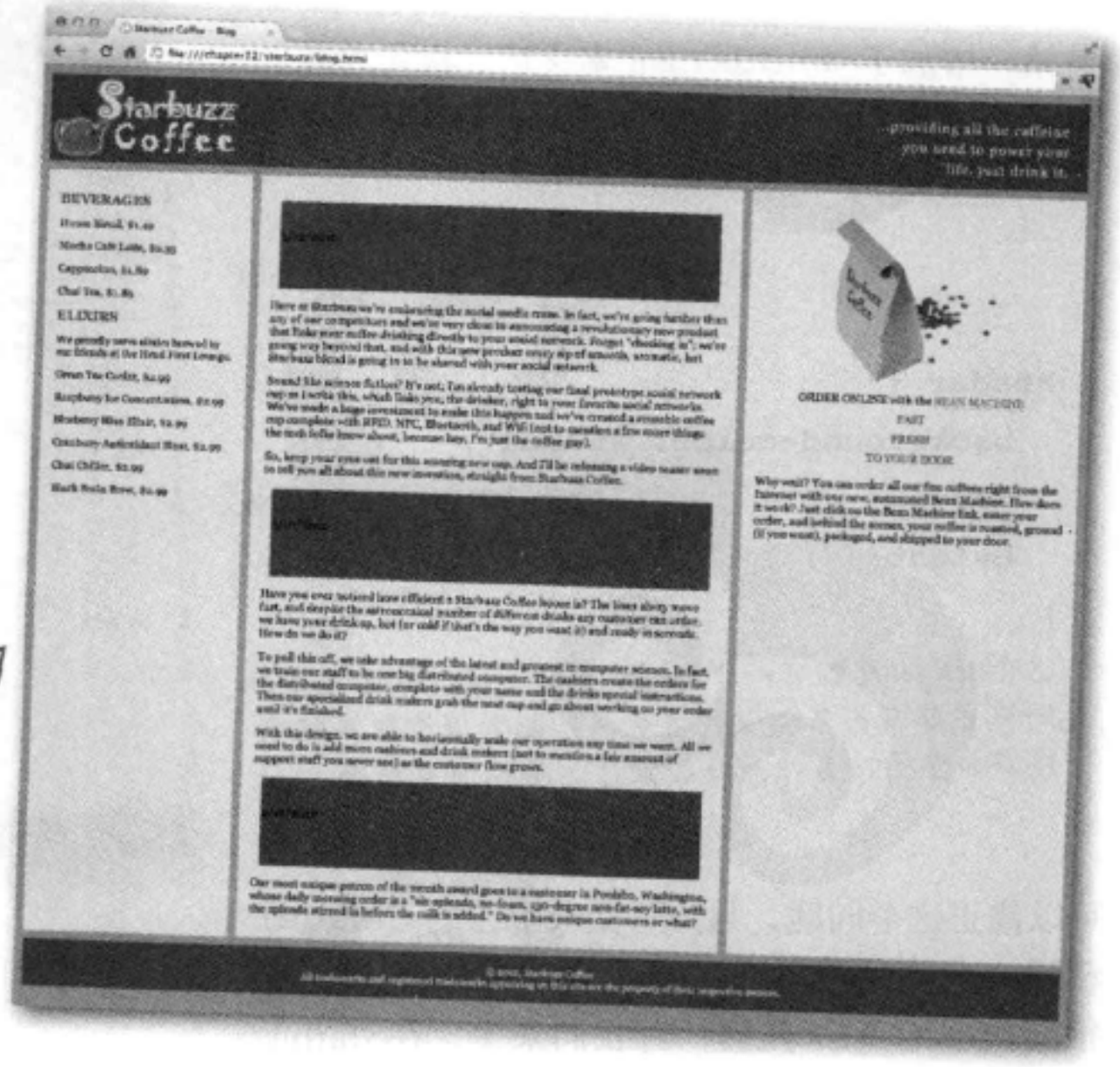
把<header>元素放在这里，包围标题和时间元素。

确保为博客区块中的每个文章都增加一个<header>。

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

测试header

为Starbuzz博客增加<header>元素，下面来做个测试。



嗯，注意到了吗？加载页面时，文章的首部看起来不太对劲。格式现在全乱了……

Sharpen your pencil

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

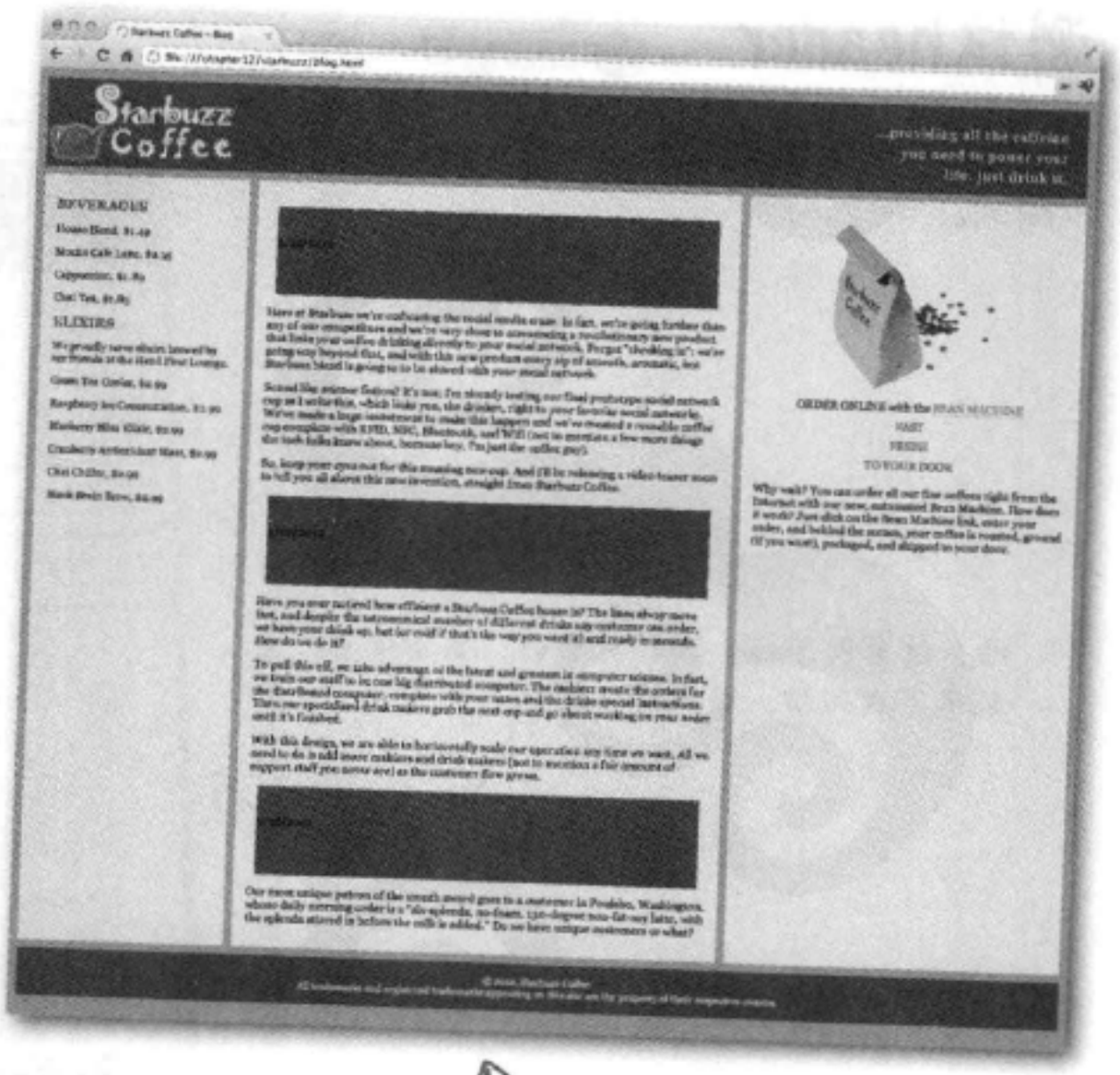
提示：查看你的CSS，看看有没有其他<header>规则可能影响你刚增加的这些新的文章首部。

这些首部到底怎么了？

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

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

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



为首部指定样式的规则对于主页眉很合适，但是对于文章中的首部看起来很糟糕。

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

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

为页面中的第一个<header>元素增加类“top”。

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

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

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

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

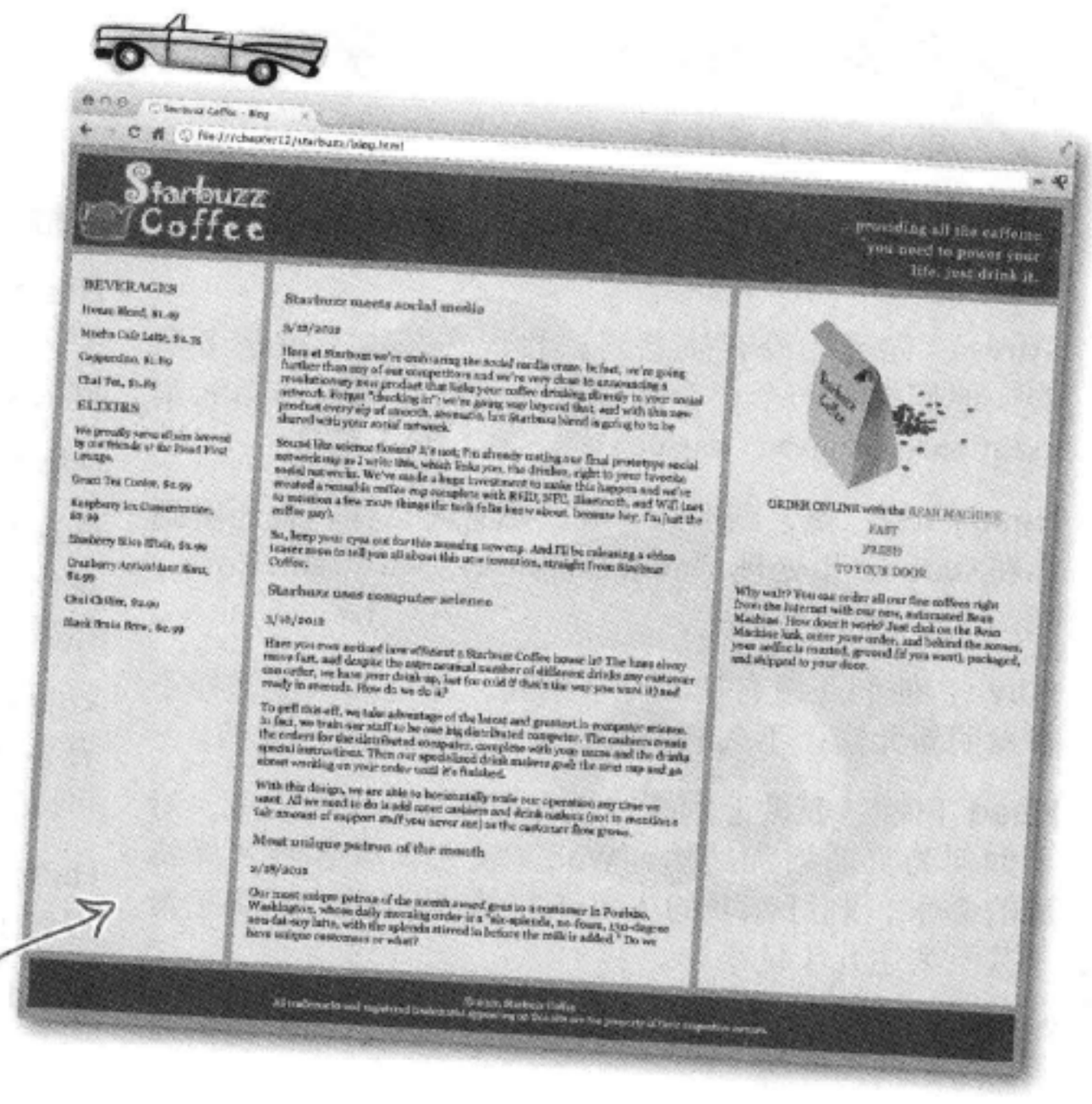
我们在CSS中为header规则增加了.top类选择器。

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

页眉的最终测试

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

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



现在文章中的首部总算有正确的格式了！

there are no Dumb Questions

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

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

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

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

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

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

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

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

对 的简短采访

** 感觉有些失落……**

Head First：你好， ，听说你最近心情不太好。怎么了？

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

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

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

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

：这倒是真的，我还没有看到那种只用来创建通用结构的新元素。

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

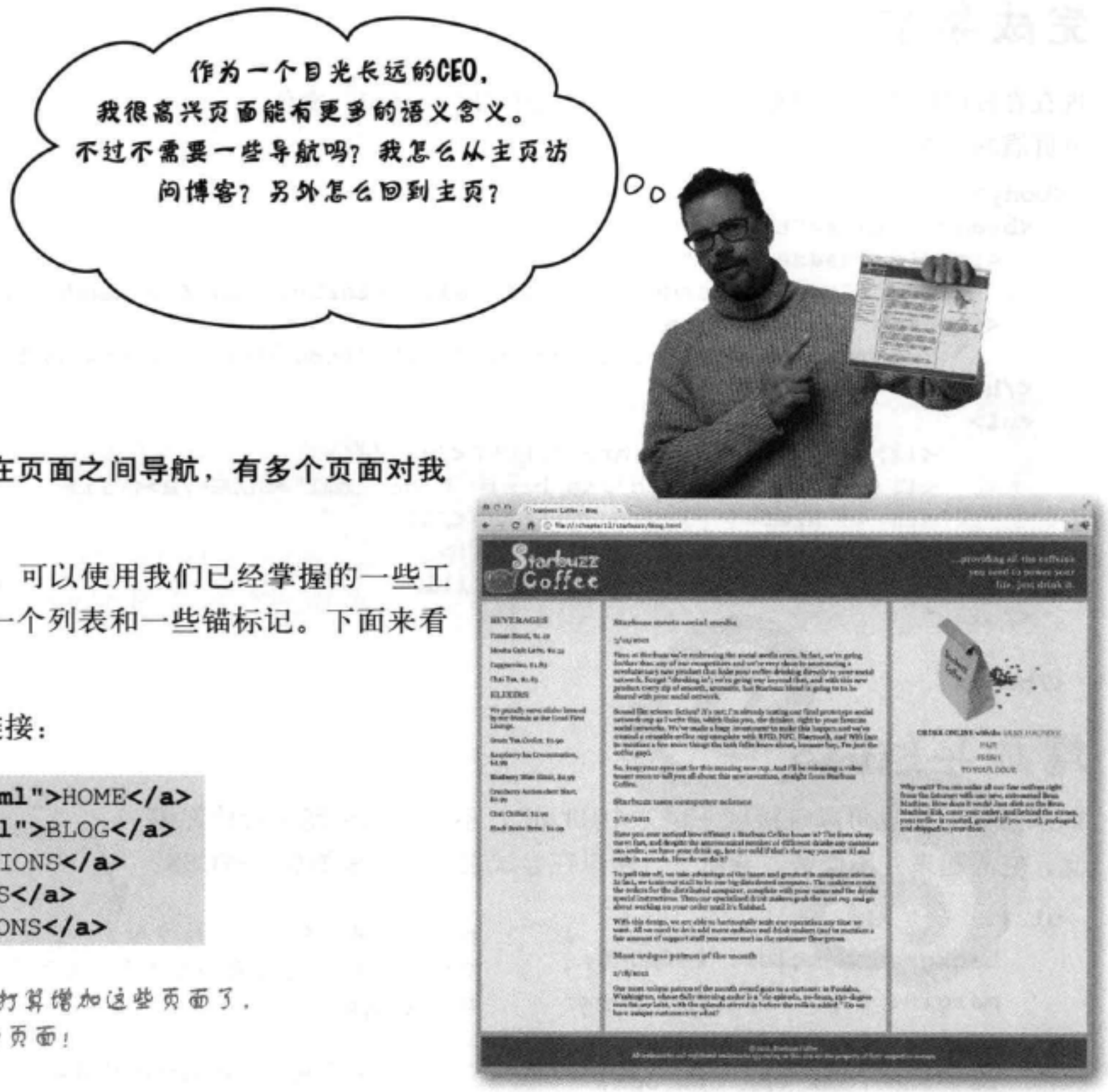
：不会吧！

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

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

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

Head First：喂？ ？



没错！如果访问者不能在页面之间导航，有多个页面对我们来说就毫无用处。

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

首先，为导航创建一组链接：

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

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

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

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

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

另外要注意，我们使用了一个类来标识被选中的一项。

完成导航

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

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

增加导航CSS

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

一定要把这个CSS增加到starbuzz.css文件的最后。

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

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

还要注意我们删除了列表项的项目符号。

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

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

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

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

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

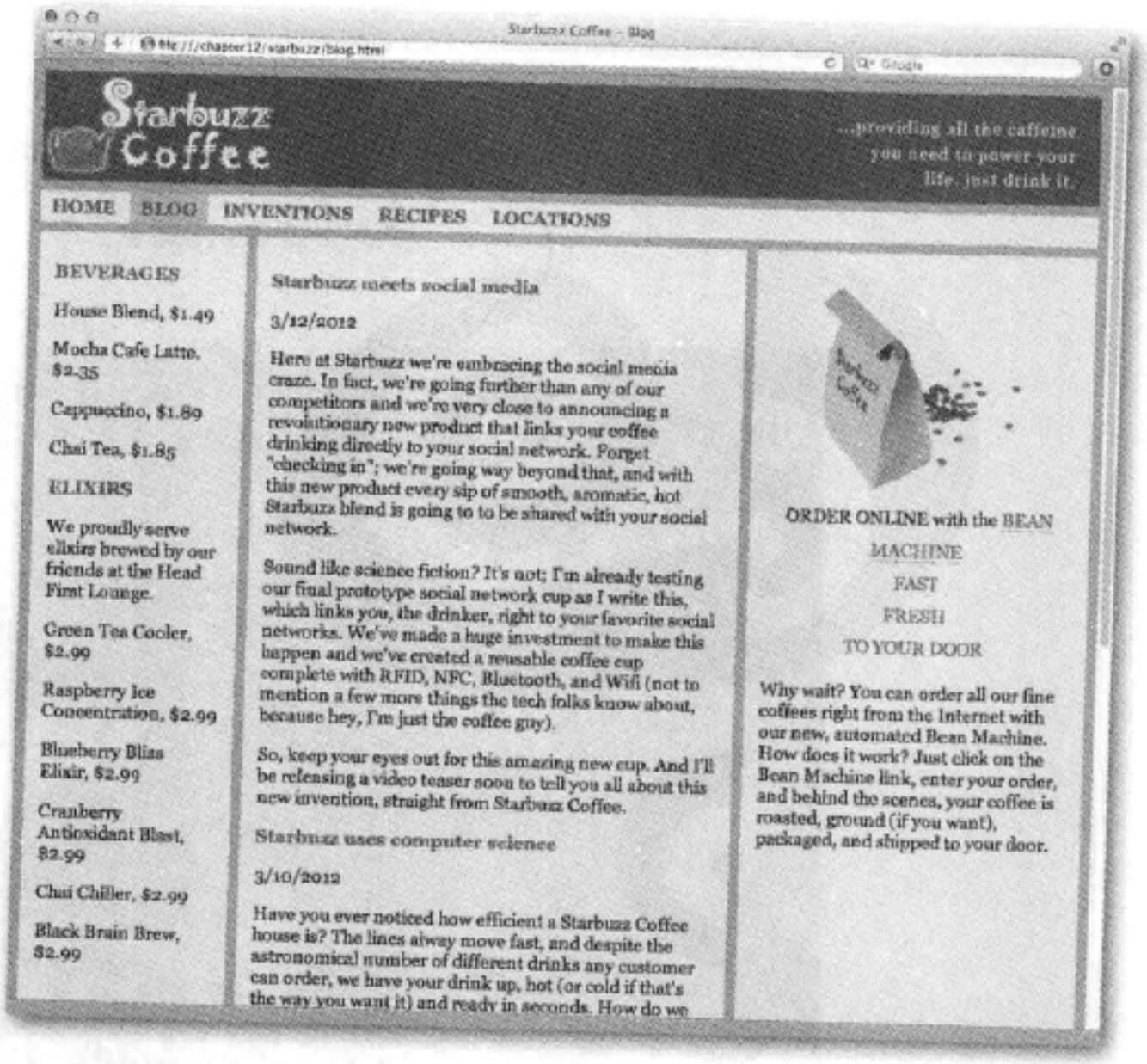
最后，为“selected”类的<li>元素设置背景，这样一来，如果某个导航项正好对应我们所在的页面，它看起来会与其他导航项有所不同。

谁需要GPS？测试导航

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

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

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



增加<nav>元素……

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

这是<nav>开始标记，我们把整个导航列表包围在一个<nav>元素中。

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

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

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

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

让CSS更特定……

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

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

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

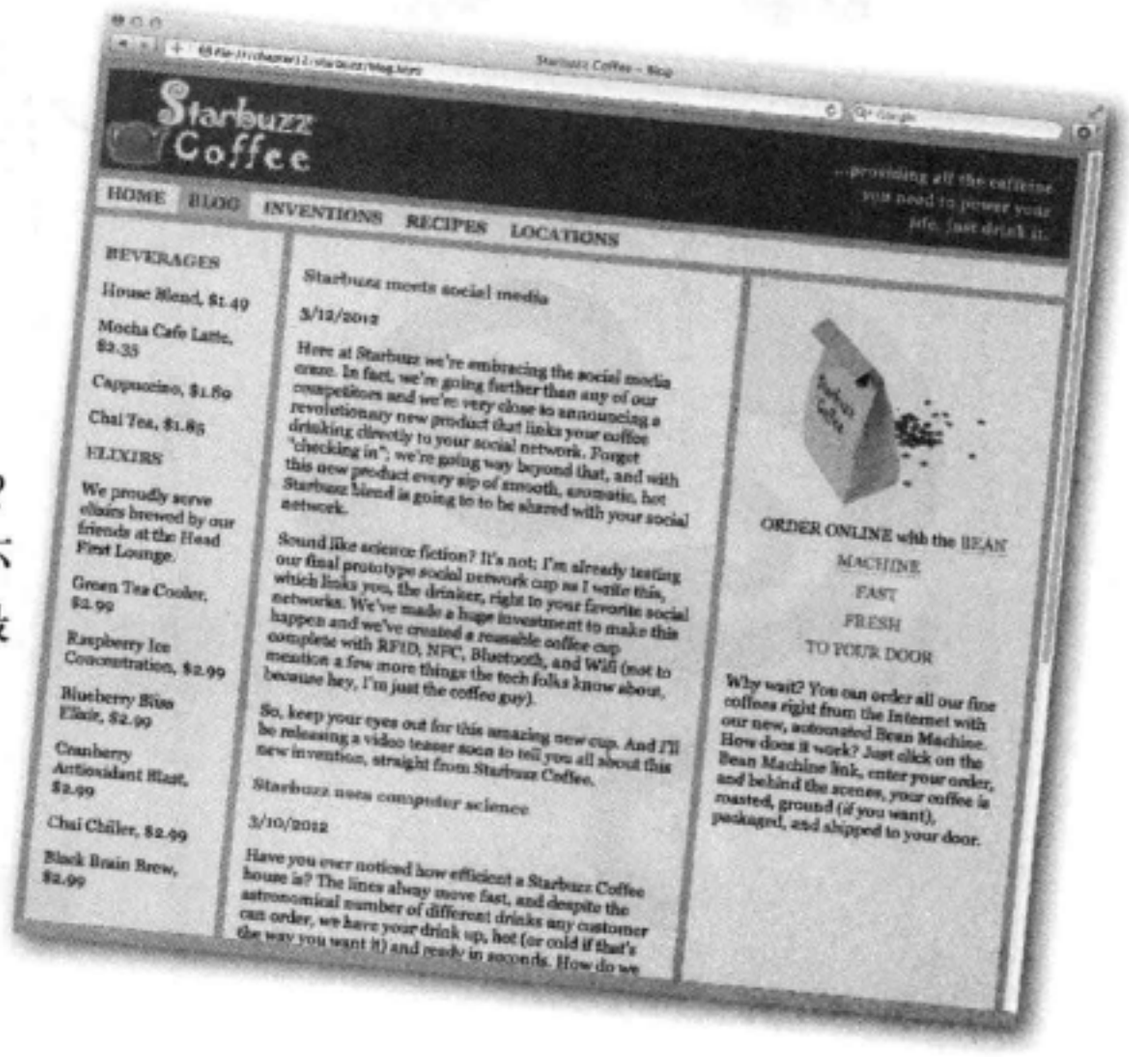
这里增加了一个属性，将<ul>元素的外边距设置为0，使它能紧贴着放在<nav>元素中（否则默认情况下，如果没有明确设置为0，<ul>元素会有一个外边距，这会让<ul>稍稍偏离）。

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

注意，两个规则都增加了“nav”，这个规则中有两个选择器！

哇！看看这里的导航！

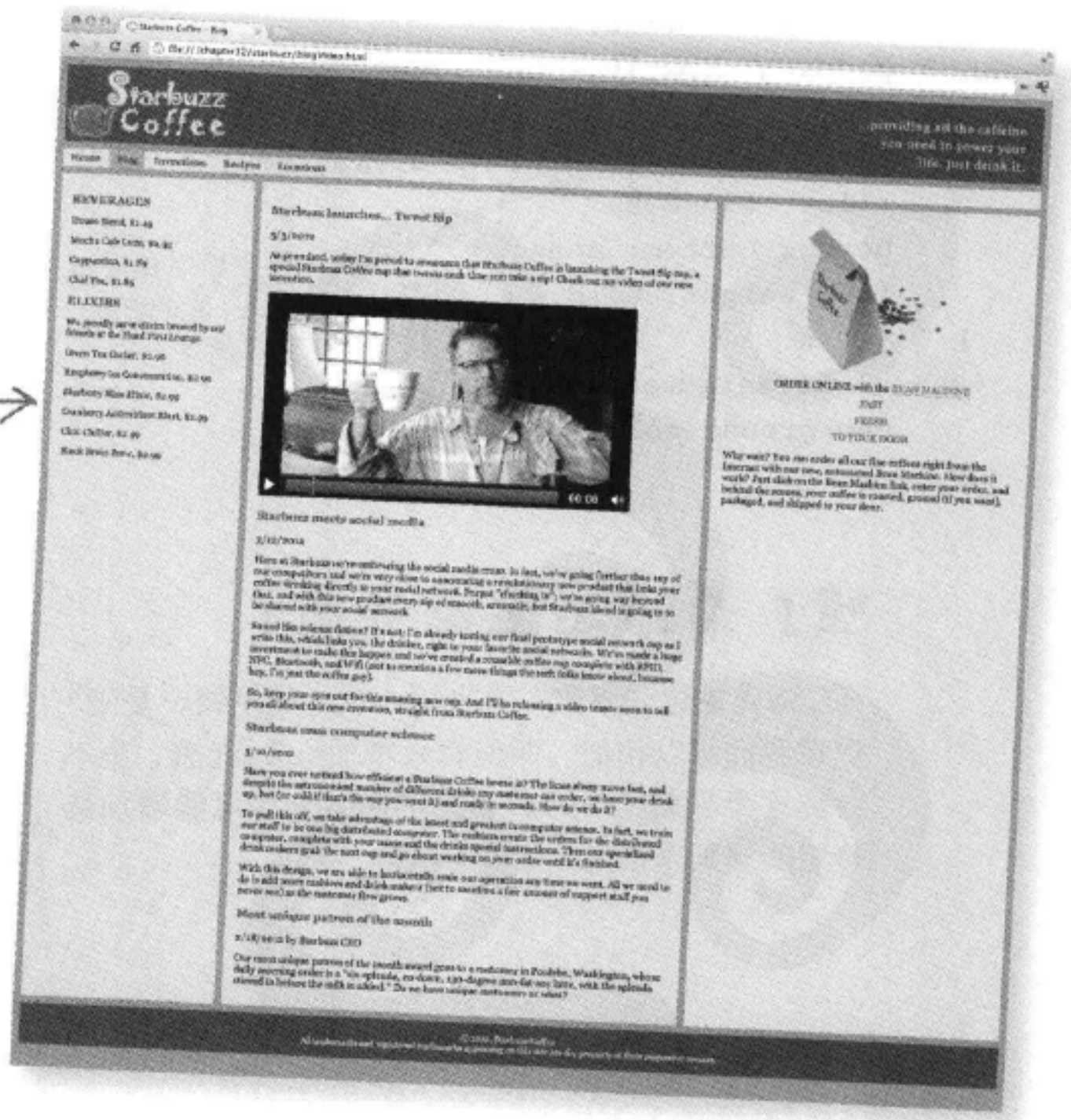
在CSS完成这些修改，再来试一试。还不错，是吧？现在我们可以相信，即使将来增加<ul>元素，也不会受这个导航CSS的影响。记住，要尽可能增加最特定的规则来指定元素的样式。



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

他想把视频放在页面里，就像这样……

这是Starbuzz博客页面，加入了我们所有最新的改进……



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

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

Frank：等一下，Flash不是更好些吗？它已经发展了很长一段时间了。

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

Frank：我明白了。那么怎么利用一个元素来实现视频呢？

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

Frank：听上去很容易。简直是小菜一碟。

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

Frank：编码？

Jim：就是对一个视频片段中的视频和音频编码所用的格式。

Frank：这很麻烦吗？

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

Frank：听起来不错，开始干吧！

创建新博客条目

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

把它增加到 "blog" <section>的最上面……

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

</article>
```

要在这里增加视频，放在博客条目中的段落下面。

现在加入<video>元素

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

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

这里是video开始标记，它有很多属性……

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

稍后还会介绍这里可以放置什么内容……

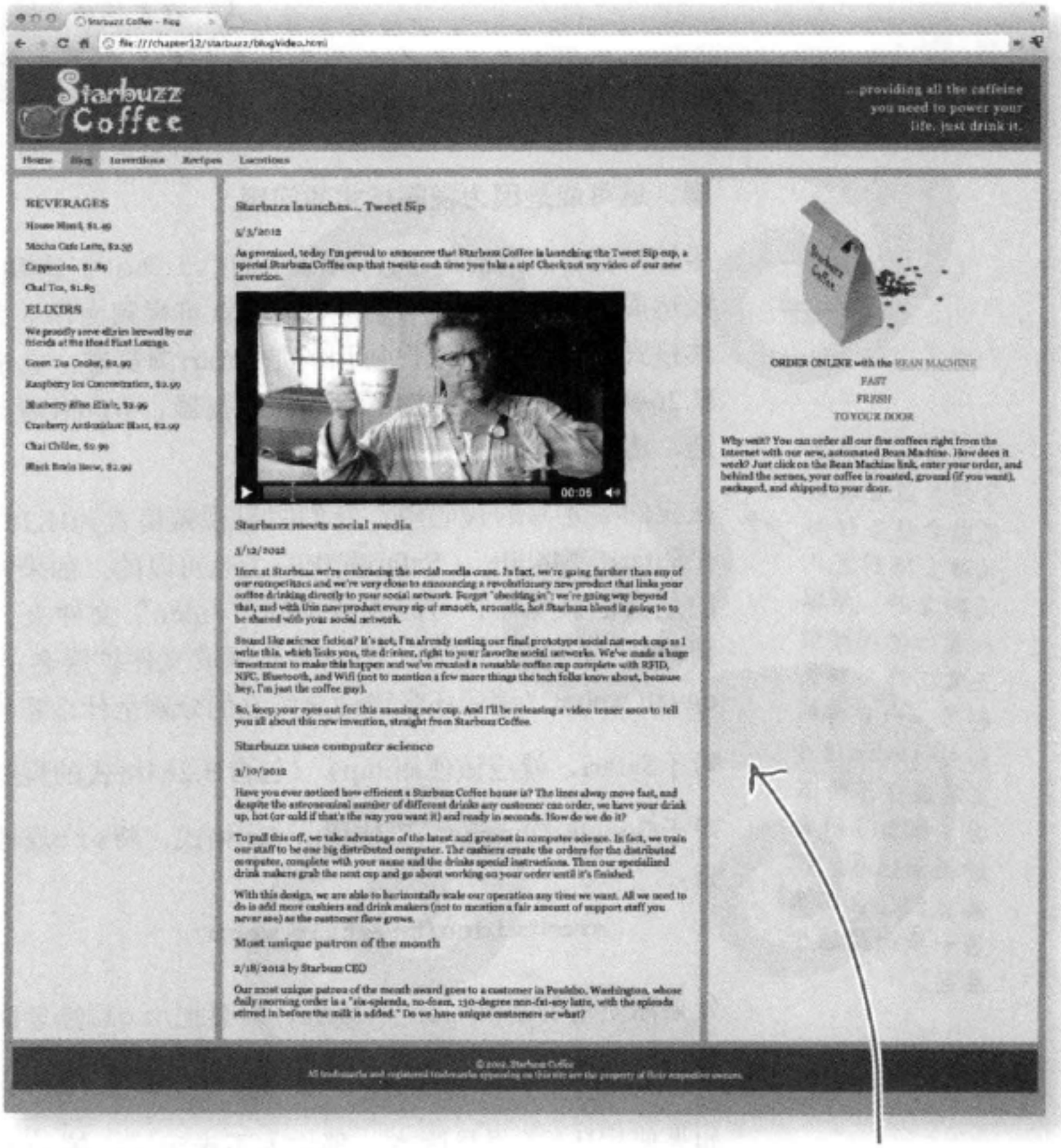
这里是video结束标记。

灯光，摄像，开拍……

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

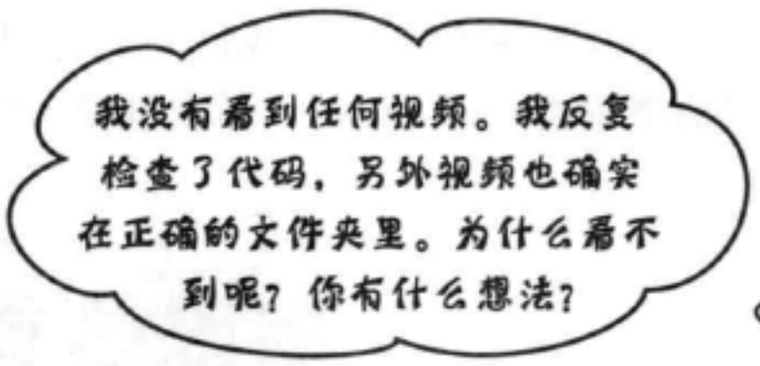
我们的视频现在就在页面中，宽度和高度正是我们刚才指定的。

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



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

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



嗯，这可能是因为视频格式的问题。

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

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

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

对于Safari，就应该使用.mp4（包含H.264格式的视频）。

对于Google Chrome，要使用.webm格式，将src属性替换为：

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

如果你使用的是Firefox或Opera，就要把src属性替换为：

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

先来试一试，稍后我们还会回来详细说明。

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

<video>元素如何工作？

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

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

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

如果有autoplay属性，一旦页面加载视频就会开始播放。

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

页面中视频的宽度和高度。

视频的源位置。

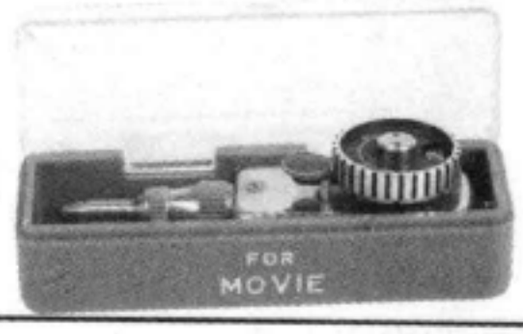
如果愿意，可以提供一个可选的海报图像，视频未播放时就会显示这个图像。

当然，还可以为元素增加一个id，以方便对它应用某些样式。

Web镇的视频礼节：autoplay属性

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

仔细检查video属性……



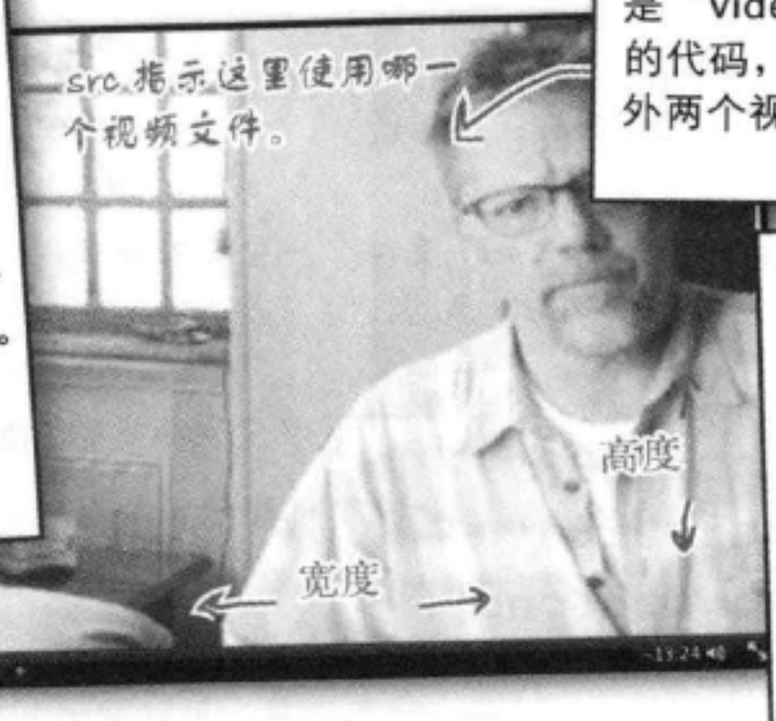
下面再来更仔细地研究一些比较重要的的video属性：

controls

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

src

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



preload

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

autoplay

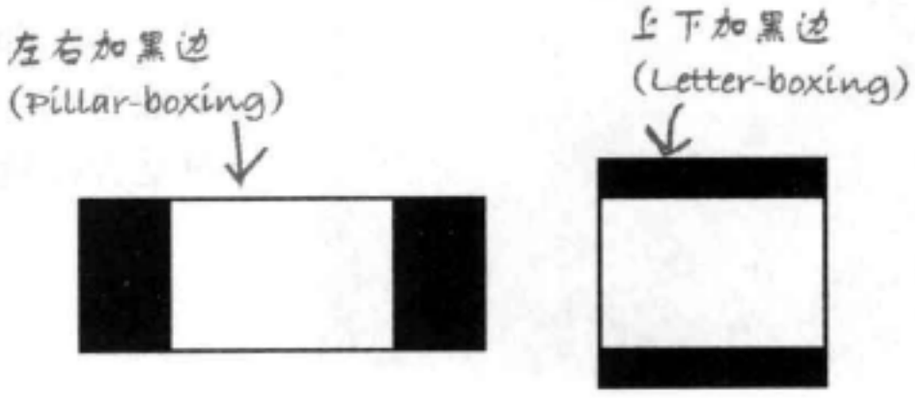
autoplay布尔属性告诉浏览器：一旦有了足够的数据就开始播放视频。你可能会看到，我们的演示视频几乎立即开始播放。

poster

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

width, height

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



loop

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

是的。对于HTML视频，每个浏览器中提供的控件都不同。

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

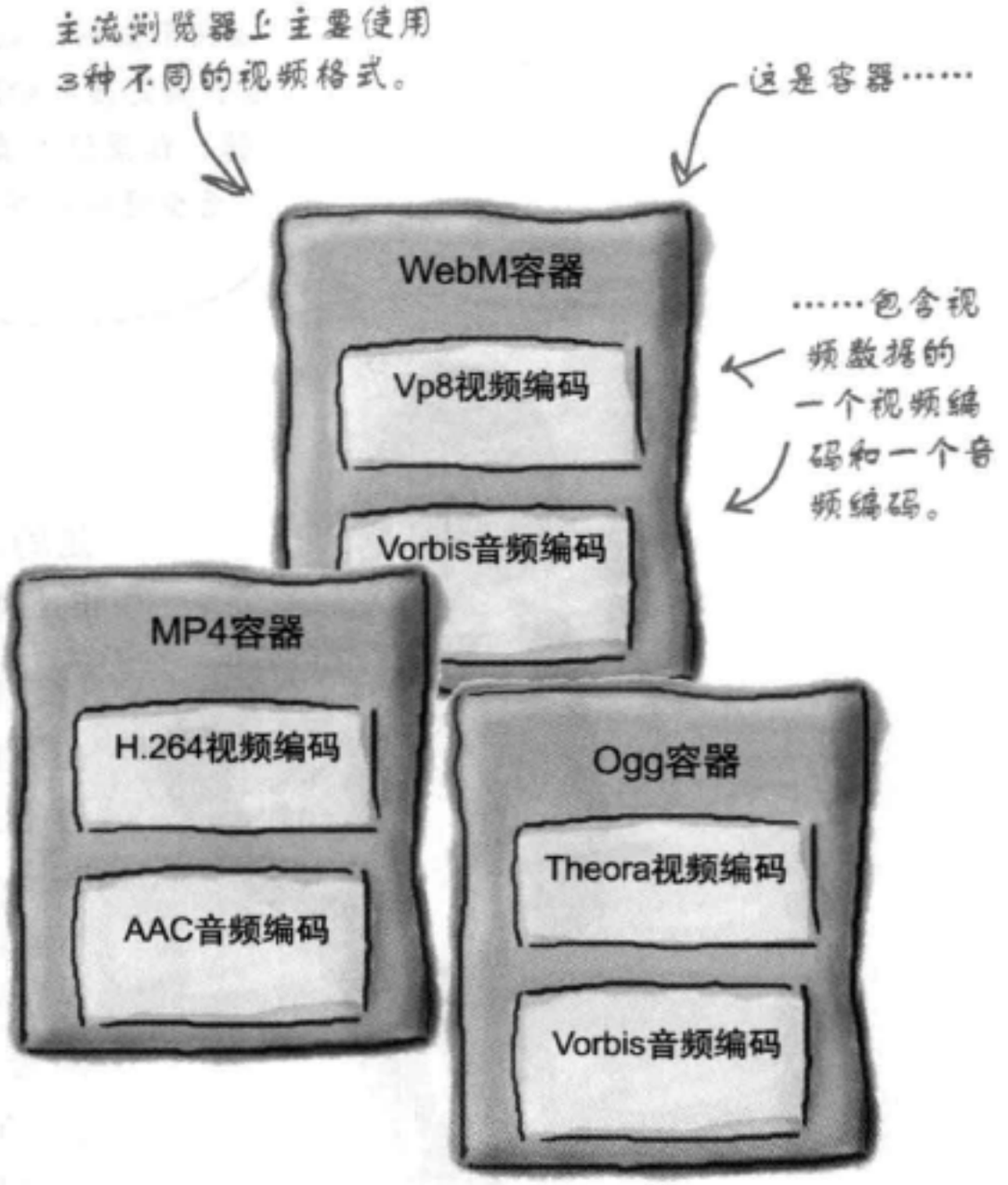
我们在《Head First HTML5 Programming》中就实现了自己的定制控件。加入我们吧，JavaScript真的很有趣！

关于视频格式需要知道什么

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

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

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



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

HTML5规范允许采用任何视频格式。具体支持哪些格式由浏览器实现来确定。

视频格式竞争对手

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

MP4容器，包含H.264视频和AAC音频

H.264由MPEG-LA公司授权。

目前有多种H.264，每一种分别称为一个“profile”（等级或类）。

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

WebM容器，包含VP8视频和Vorbis音频

WebM由Google设计，用来处理VP8编码视频。

WebM/VP8得到了Firefox、Chrome和Opera的支持。

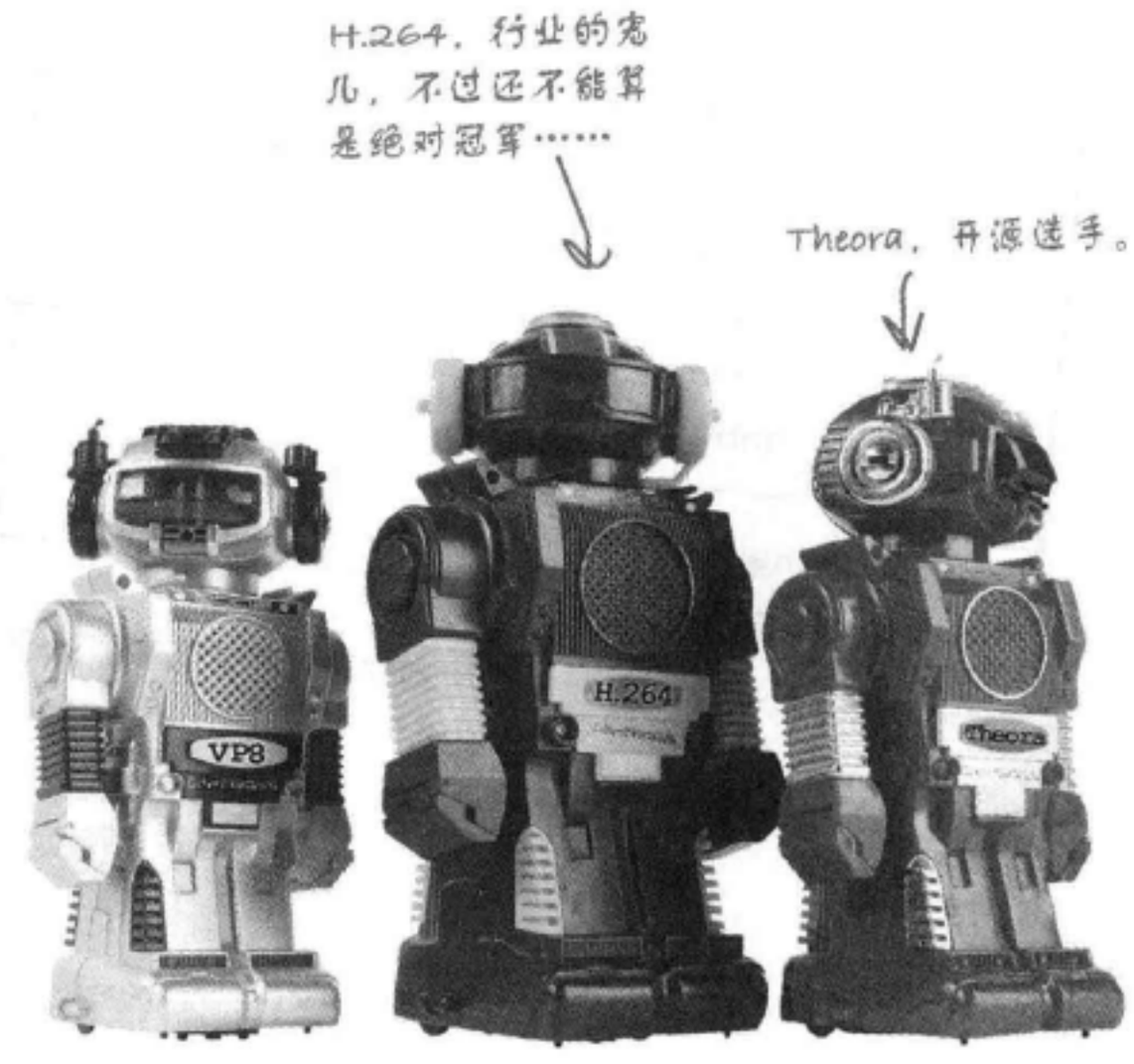
WebM格式的视频扩展名为.webm。

Ogg容器，包含Theora视频和Vorbis音频

Theora是一个开源编解码器。

采用Theora编码的视频通常包含在Ogg文件中，文件扩展名为.ogv。

Ogg/Theora得到了Firefox、Chrome和Opera的支持。



CASE FILE: VIDEO

TOP SECRET

下一项任务：视频侦察

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

iOS和Android以及其他设备。

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

如何处理所有这些格式……

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

注意我们从<video>标记删除了src属性……

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

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

如果浏览器不支持视频，就会显示这个文本。

浏览器从上向下查找，直到找到它能播放的一种格式。

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

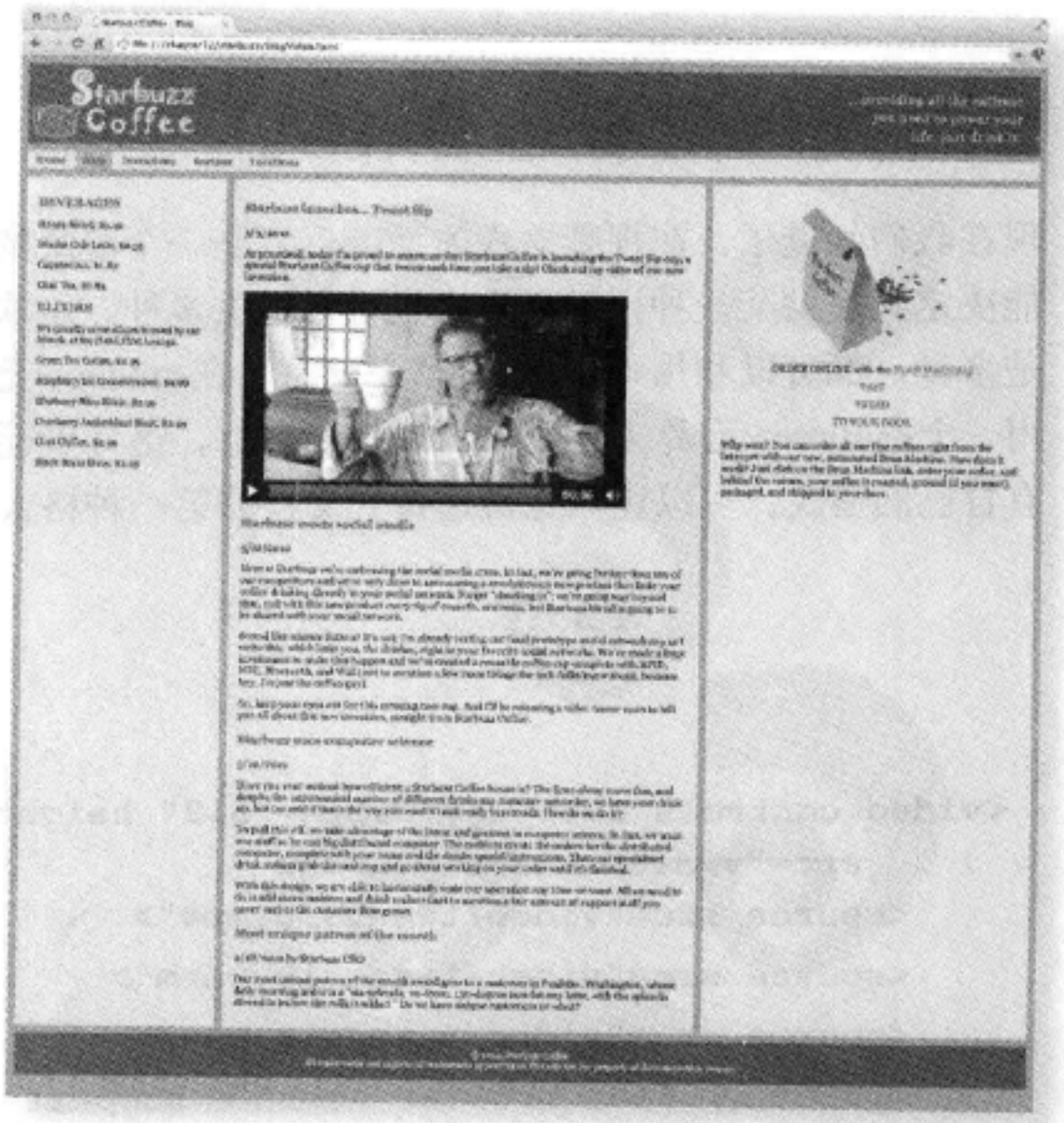
BULLET POINTS

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

再来一次：灯光，摄像，开拍……

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

现在不同浏览器上应该都能看到视频正常播放！



如何更具体地指定视频格式

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

src中使用的文件实际上是具体视频（和音频以及一些元数据）的容器。

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

视频编解码器。

音频编解码器。

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

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

这是视频文件的MIME类型。指定了容器格式。

注意codecs参数的双引号。这说明type属性两边需要使用单引号。

接下来，可以更新<source>元素，让它包含我们的3种视频的类型信息，

更新和测试

如下更新<source>元素，对页面做个测试：

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

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

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

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

there are no Dumb Questions

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

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

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

问：如果我想对我自己的视频编码，如何做起呢？

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

问：我能全屏播放我的视频吗？真奇怪API中居然没有这样的一个属性。

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

没问题。

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

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

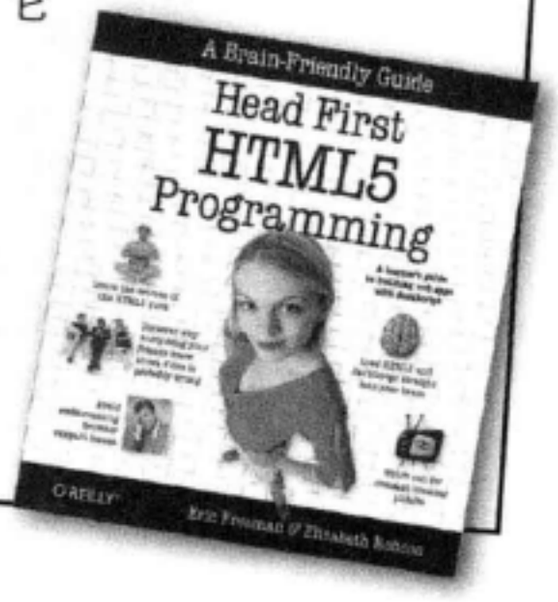
```
<video poster="video.jpg" controls>
        <source src="video.mp4">
        <source src="video.webm">
        <source src="video.ogv">
        <object>...</object>
</video>
```

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

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

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

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



元素汤

<progress>

需要显示任务的完成进度吗？比如完成了90%？可以使用这个元素。

<section>

可以使用这个元素定义文档的主要区块。

<header>

有首部的区块和整个文档的页眉可以使用这个元素。

<footer>

这个元素定义一个区块的底部或整个文档的页脚。

这个元素用于表示放在主内容旁边的内容，比如边栏或引用。

<aside>

想在页面中加入一个视频？你肯定需要这个元素。

<video>

<meter>

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

这个元素用于突出显示某些文本。就像记号笔一样棒！

<mark>

使用这个元素把网站中用于导航的所有链接组织在一起。

<nav>

可以用它在页面中包含声音内容。

<audio>

用来标记类似新闻报道或博客帖子等独立的内容。

<article>

time元素是一个时间、一个日期或者一个日期时间（如1月21日凌晨2点）。

<time>

这个元素用来在页面中显示用JavaScript绘制的图像和动画。

<canvas>

这个元素用来定义类似照片、图表甚至代码清单等独立的内容。

<figure>

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

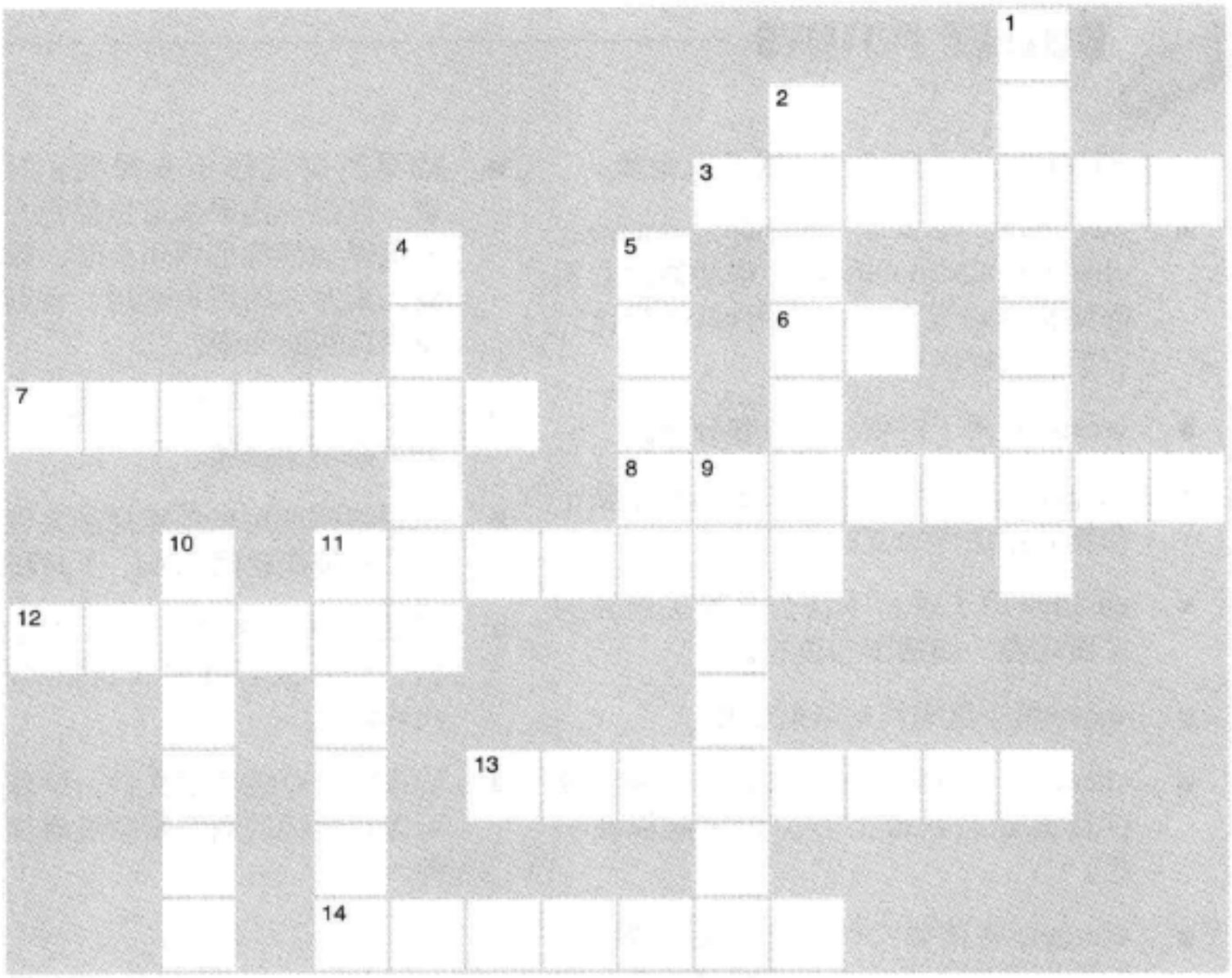
记住，HTML的乐趣一半来自试验！所以你可以自己创建几个文件，来试着用用这些新元素吧。

BULLET POINTS

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

HTML填字游戏

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



横向

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

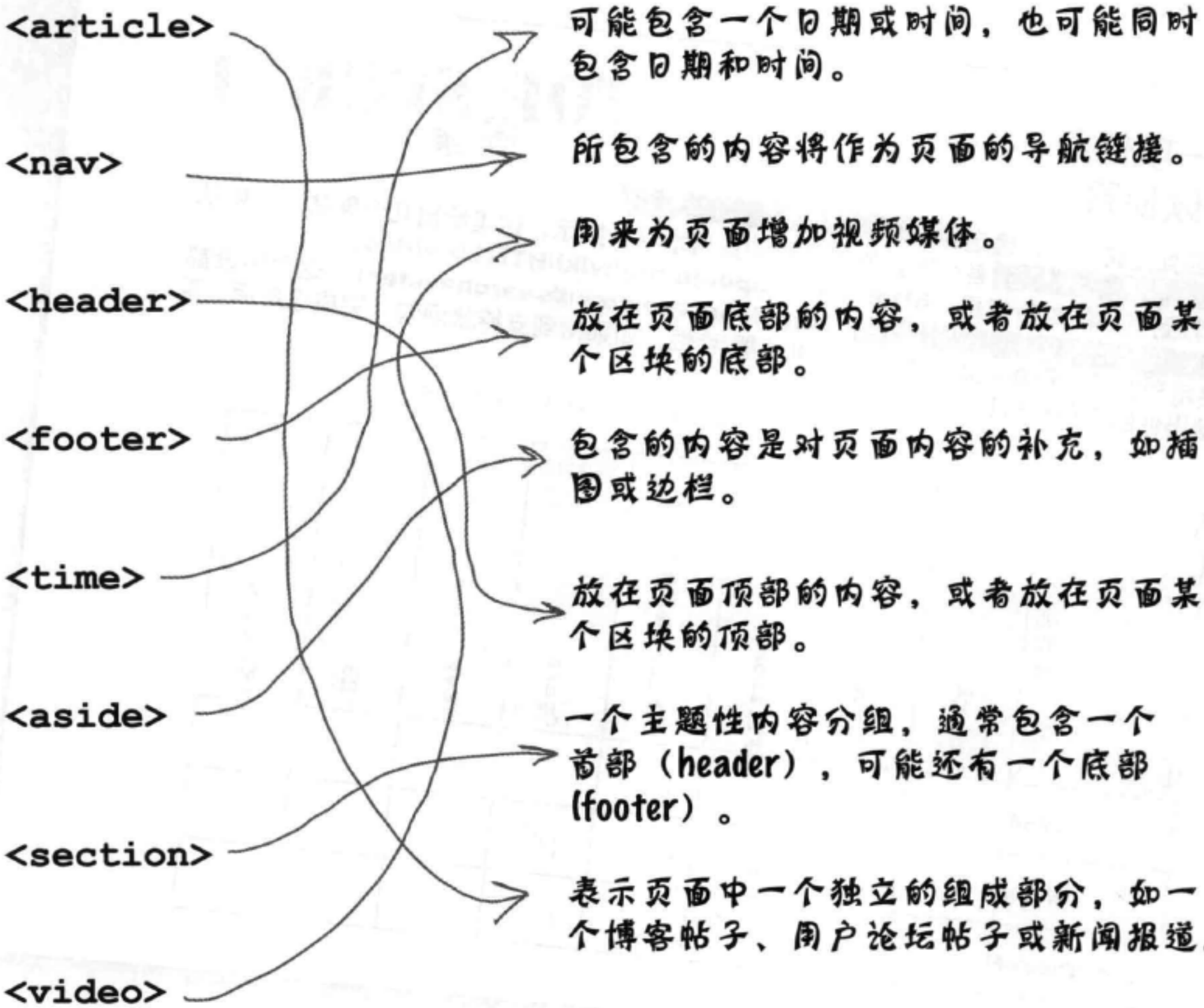
纵向

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

WHO DOES WHAT?

答案

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



CASE FILE: VIDEO

TOP SECRET

答案

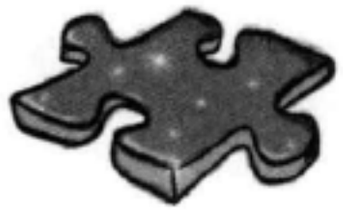
下一项任务：

视频侦察

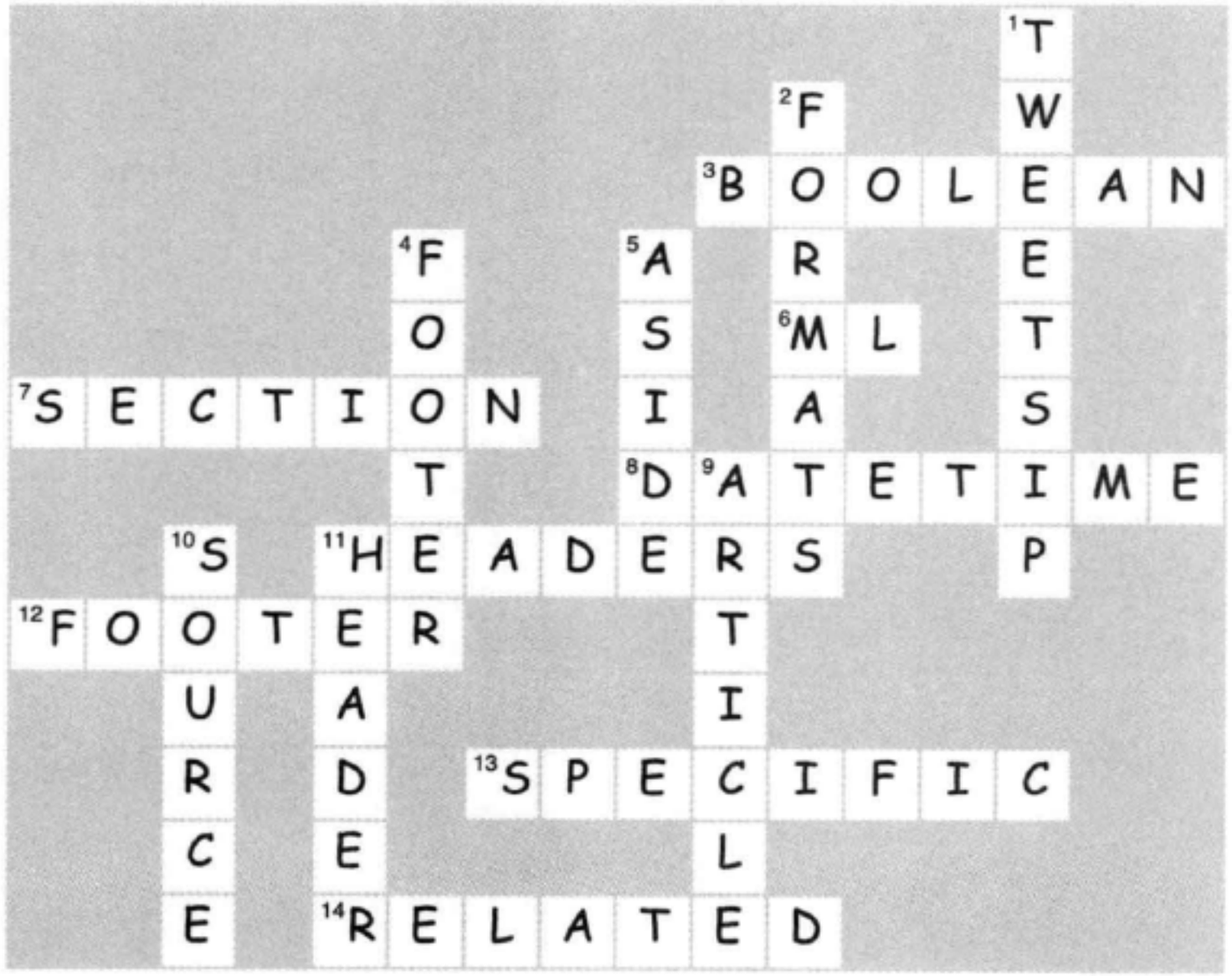
请四处走访一下，确定[redacted]各个浏览器对视频的支持水平（提示，这里给出几个网站，可以从中找到有关的最新信息：**http://en.wikipedia.org/wiki/HTML5_video**，[redacted]，**http://caniuse.com/#search=video**）。这里假设都使用浏览器的最新版本。对于每组浏览器/特性，如果得到支持就画勾，完成任务后，要给出你的任务报告！

iOS和Android以及其他设备。

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



HTML填字游戏答案


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

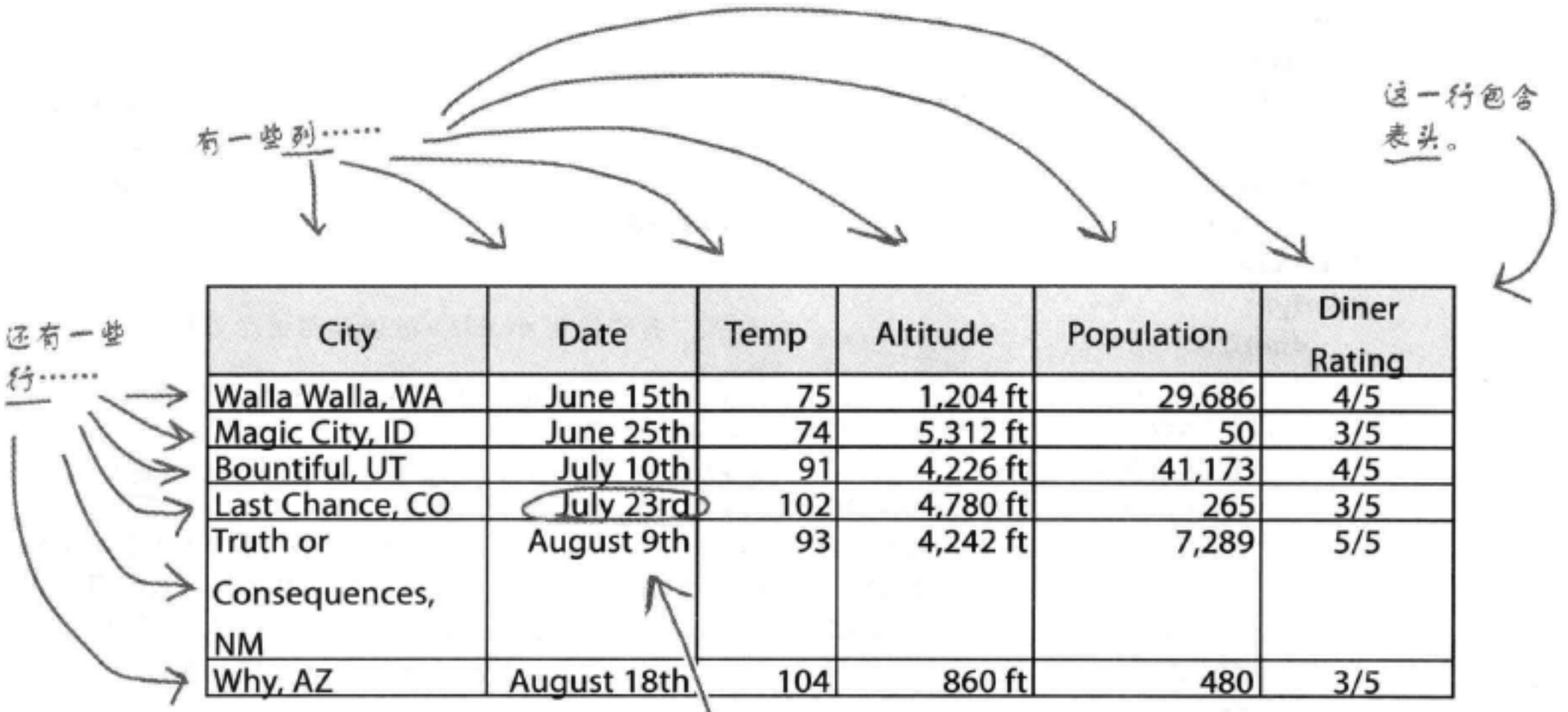
建立表格

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

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

如何用HTML建立表格？

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



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

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

用HTML创建一个表格

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

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

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

我们使用<table>标记开始这个表格。

这里是第一行，用一个<tr>开始。

每个<th>元素分别是某一列的表头。

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

每个<tr>元素构成表格中的一行。

这是第二行的开始，对应城市Walla Walla。

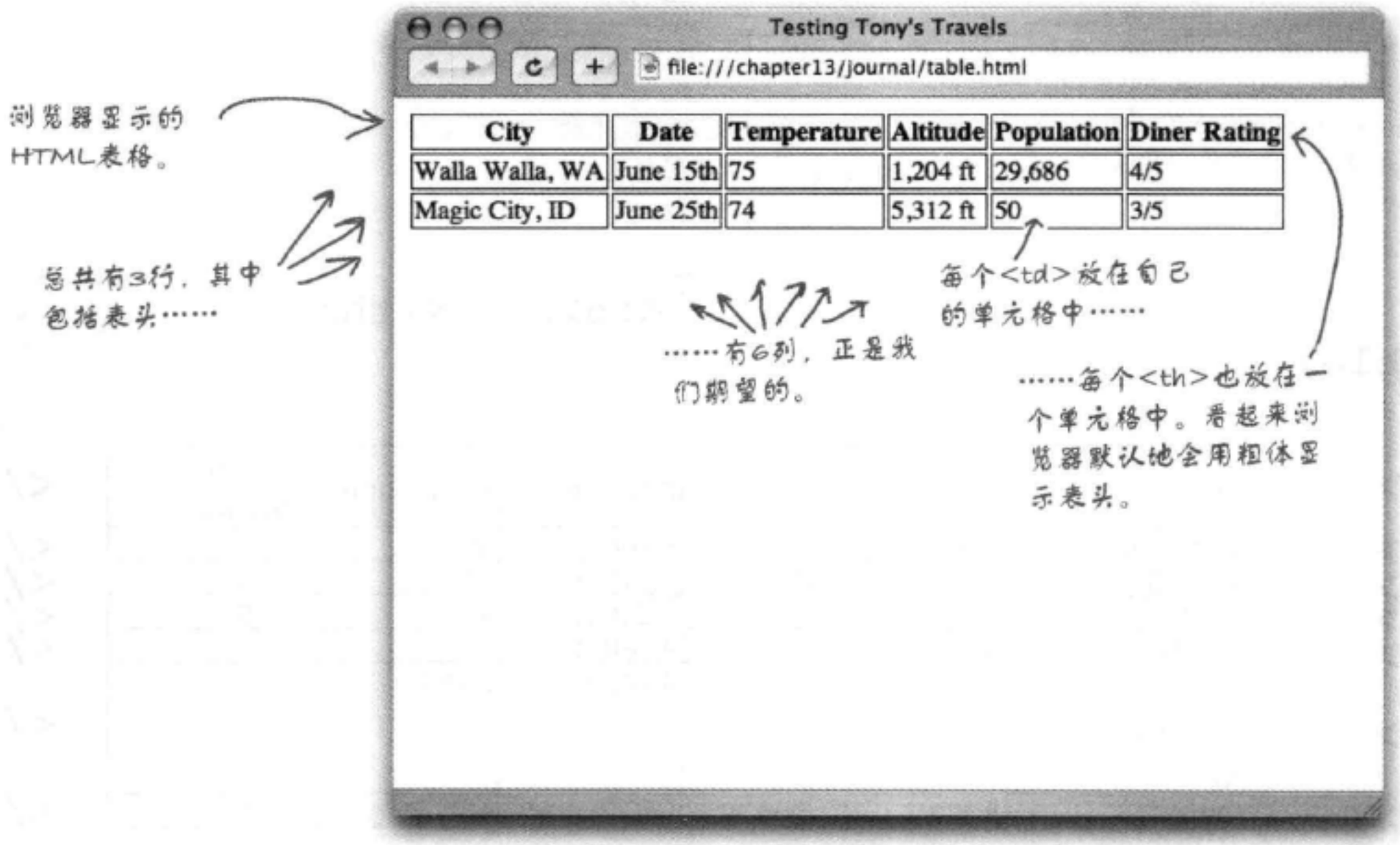
每个<td>元素包含表格中的一个单元格，每个单元格分别构成一个单独的列。

所有这些<td>构成了一行。

这里是第3行。同样的，<td>元素分别包含一个表格数据。

浏览器创建了什么？

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



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

表格剖析

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

<table>标记开始整个表格。如果想要一个表格，都要从这个标记开始。

<th>元素包含表格表头中的一个单元格。它必须放在一个表行中。

</tr>标记结束一个表行。

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

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

<td>元素包含表格中的一个数据单元格。它必须嵌套在一个表行中。

</table>标记结束表格。

there are no Dumb Questions

问：为什么没有一个表列元素？感觉好像很重要。

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

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

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

问：如果我希望表格表头放在表格的左侧，而不是放在上方，能行吗？怎么做到呢？

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

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

答：还是要用CSS，别走捷径，别贪小利。

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

问：表格不就是关于表现的吗？表现与结构有什么关系？

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

问：HTML表格与CSS表格显示有什么关系？

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

问：可以使用CSS表格显示为HTML表格指定样式吗？

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

表格提供了一种在HTML中指定表格数据的方法。

表格由行中的数据单元格组成。列隐含地定义在行中。

表格中的列数就是行中数据单元格的个数。

一般来讲，表格不用来提供表现，那是CSS的工作。

扮演浏览器

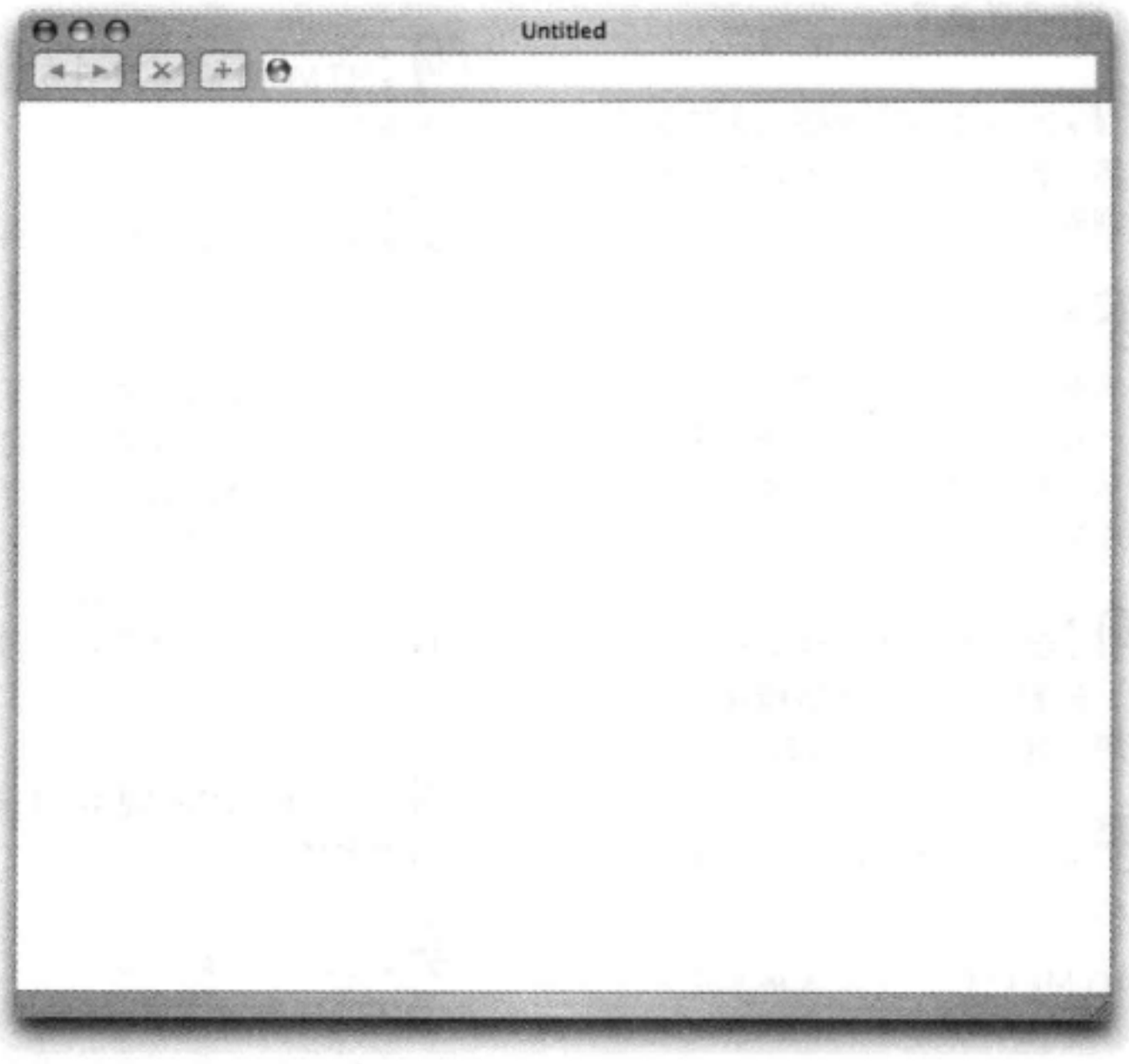
在左边可以看到一个表格的HTML。你的任务是扮演显示这个表格的浏览器。完成这个练习之后，对照这一章最后的答案，检查你做对了没有。

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

这是表格的HTML。

哎呀！看来有人得学学如何调整HTML的格式。

在这里画出表格。



增加一个标题

通过增加一个标题，可以让表格立即有所改观。

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

标题在浏览器中显示。默认地，大多数浏览器会把标题显示在表格上方。

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

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

其余的表格行放在这里。

测试……开始考虑样式

为表格增加标题，保存并重新加载页面。

标题在表格上方。放在下面可能看起来会更好点。

Tony's Table

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

The cities I visited on my Segway'n USA travels

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

我们确实需要为表格数据单元格增加一些内边距，以便于阅读……

……为Tony的网站加一点点橙色，可以把所有内容联系起来。

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

开始指定样式之前，先把表格放在Tony的页面上

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

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

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

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

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

这是原来的城市列表。将它删除，因为我们要把它换成表格。

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

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

现在为表格增加样式

将下面突出显示的新样式增加到“journal.css”样式表的最下面。

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

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

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

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

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

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

我们希望把标题移到表格的下方。

还要修改表格数据单元格的边框，设置为一个更细的灰色虚线边框。

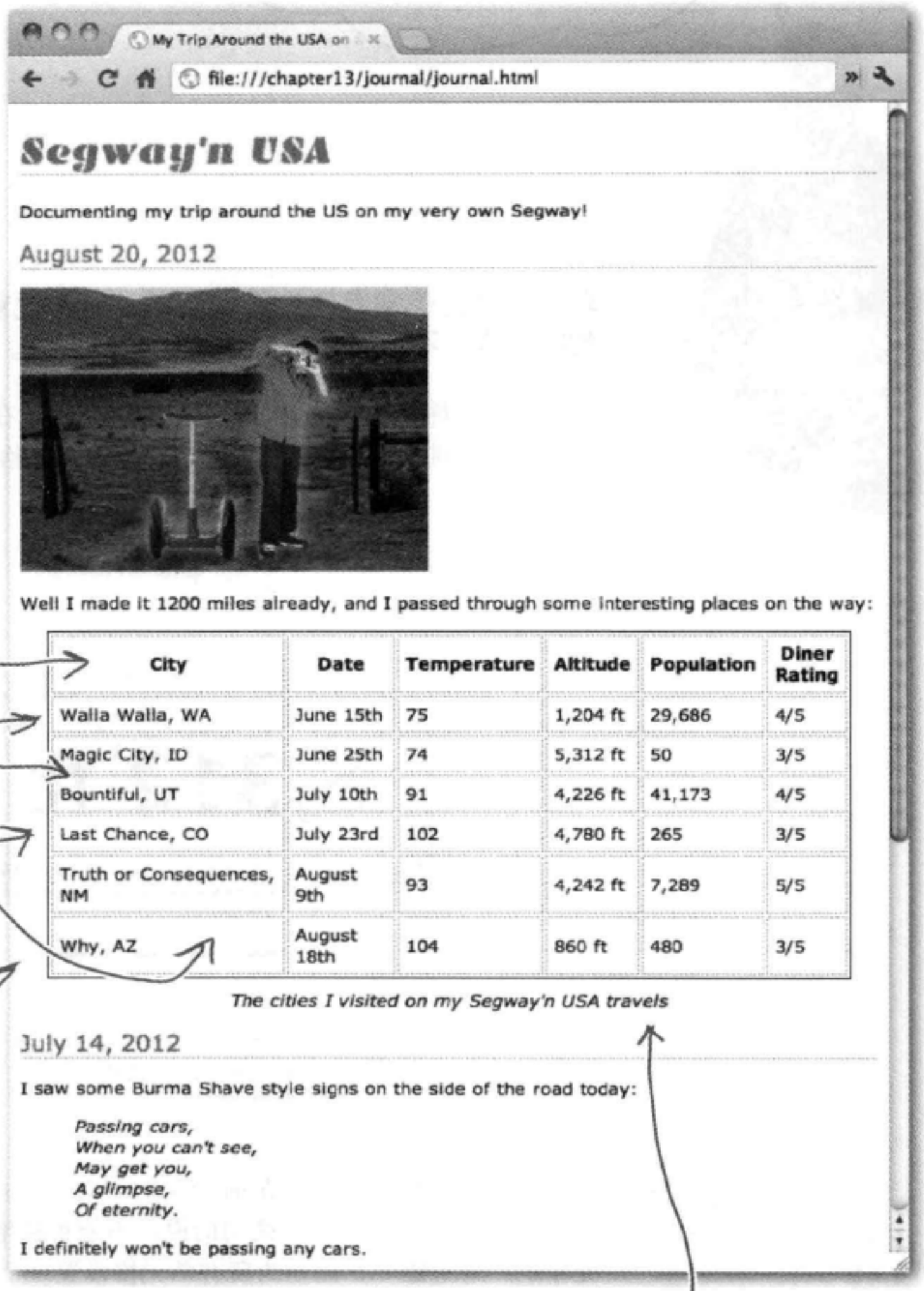
另外为数据单元格增加一些内边距，使数据内容与边框之间有一些空间。

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

测试增加样式后的表格

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

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



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

现在有一个深色虚线边框。

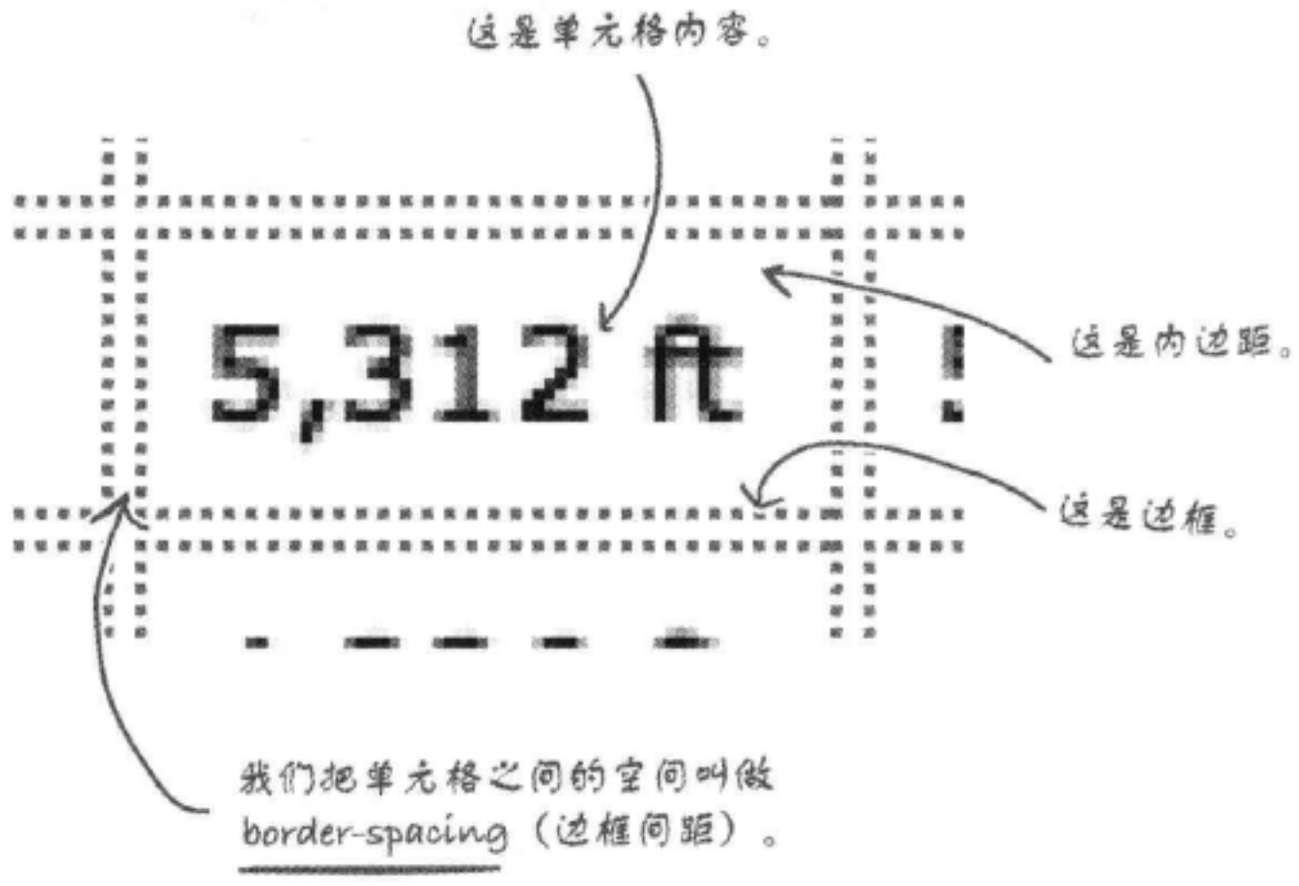
表格上有一些外边距，另外每个表格单元格有一些内边距。

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

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

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

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

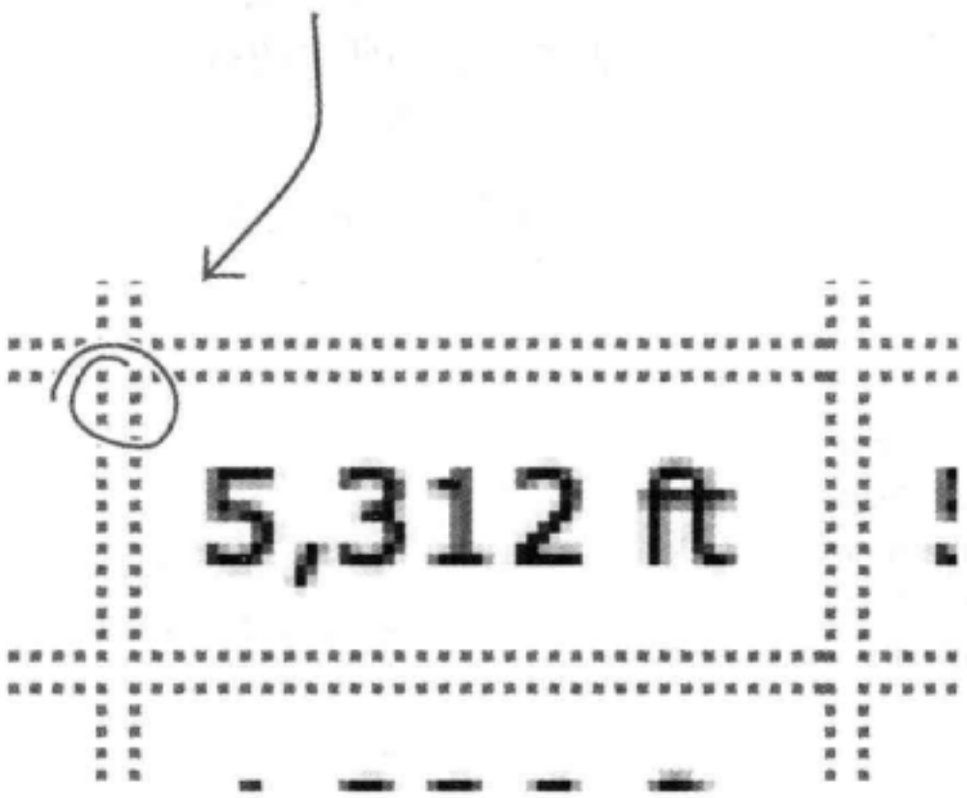


这就像Starbuzz的CSS表格显示布局中使用的border-spacing属性。

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

Sharpen your pencil

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



there are no Dumb Questions

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

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

问：嗯，有没有办法在垂直方向和水平方向上设置不同的边框间距？好像这很有用。

答：当然可以。可以像这样指定边框间距：

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

这会设置10像素的水平边框间距，30像素的垂直边框间距。

问：看起来border-spacing属性在我的浏览器中不起作用。

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

折叠边框

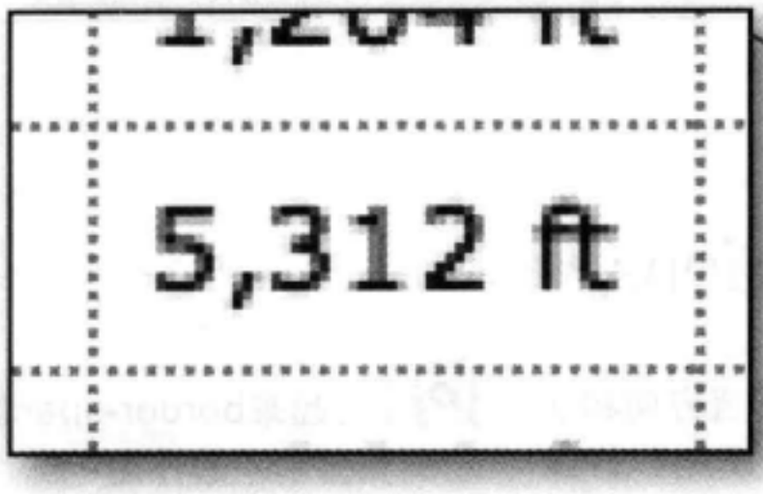
除了border-spacing属性，还有一种办法可以解决这个边框难题。你可以使用一个名为border-collapse的CSS属性来折叠边框，使单元格之间根本没有边框间距。这样一来，浏览器就会忽略表格上设置的所有边框间距。另外还会把紧挨着的两个边框合并成一个边框。这样两个边框就会“折叠”为一个边框。

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

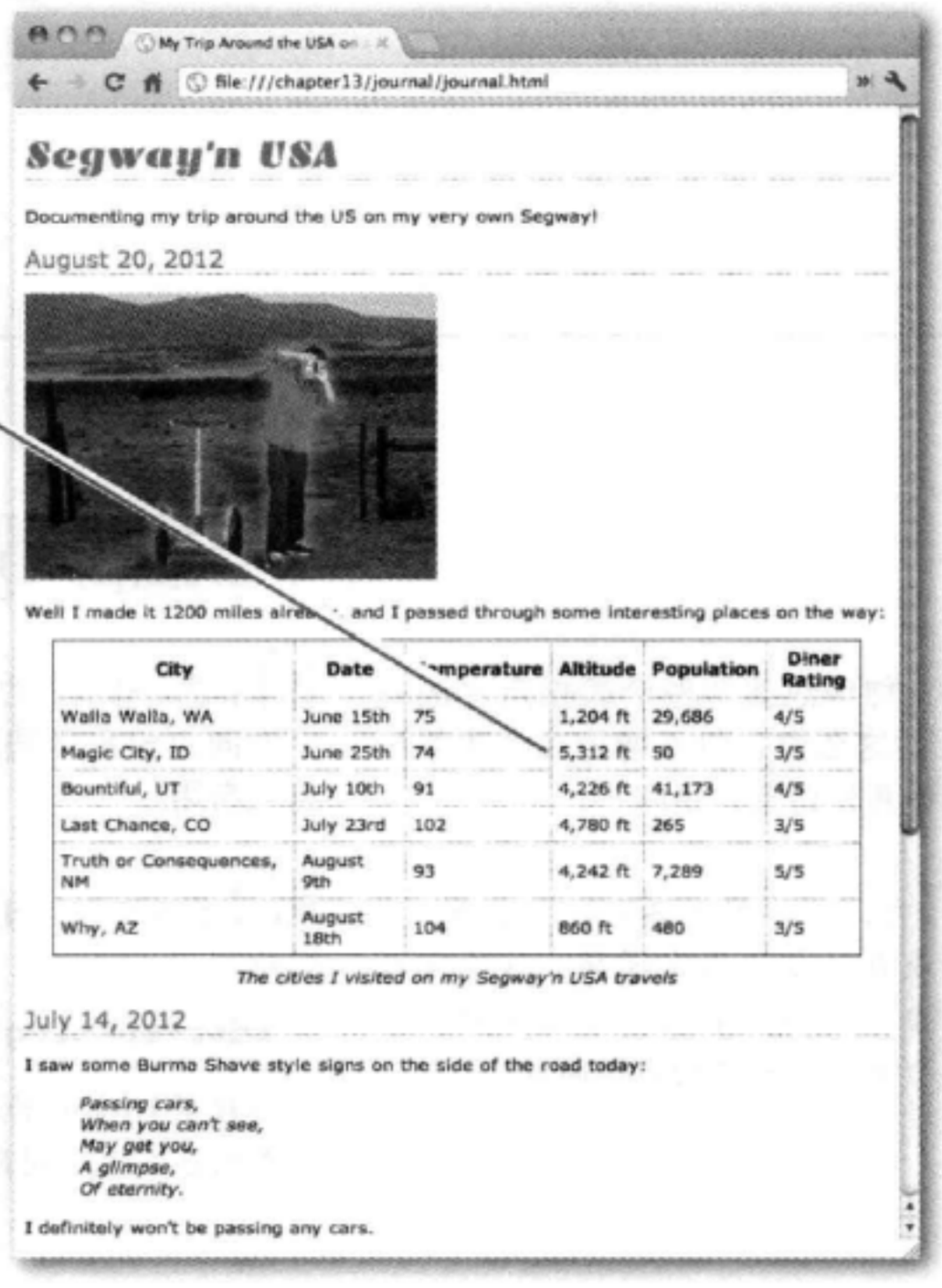
```
table {
    margin-left: 20px;
    margin-right: 20px;
    border: thin solid black;
    caption-side: bottom;
    border-collapse: collapse;
}
```

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

保存文件并重新加载页面，然后检查边框的变化。



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

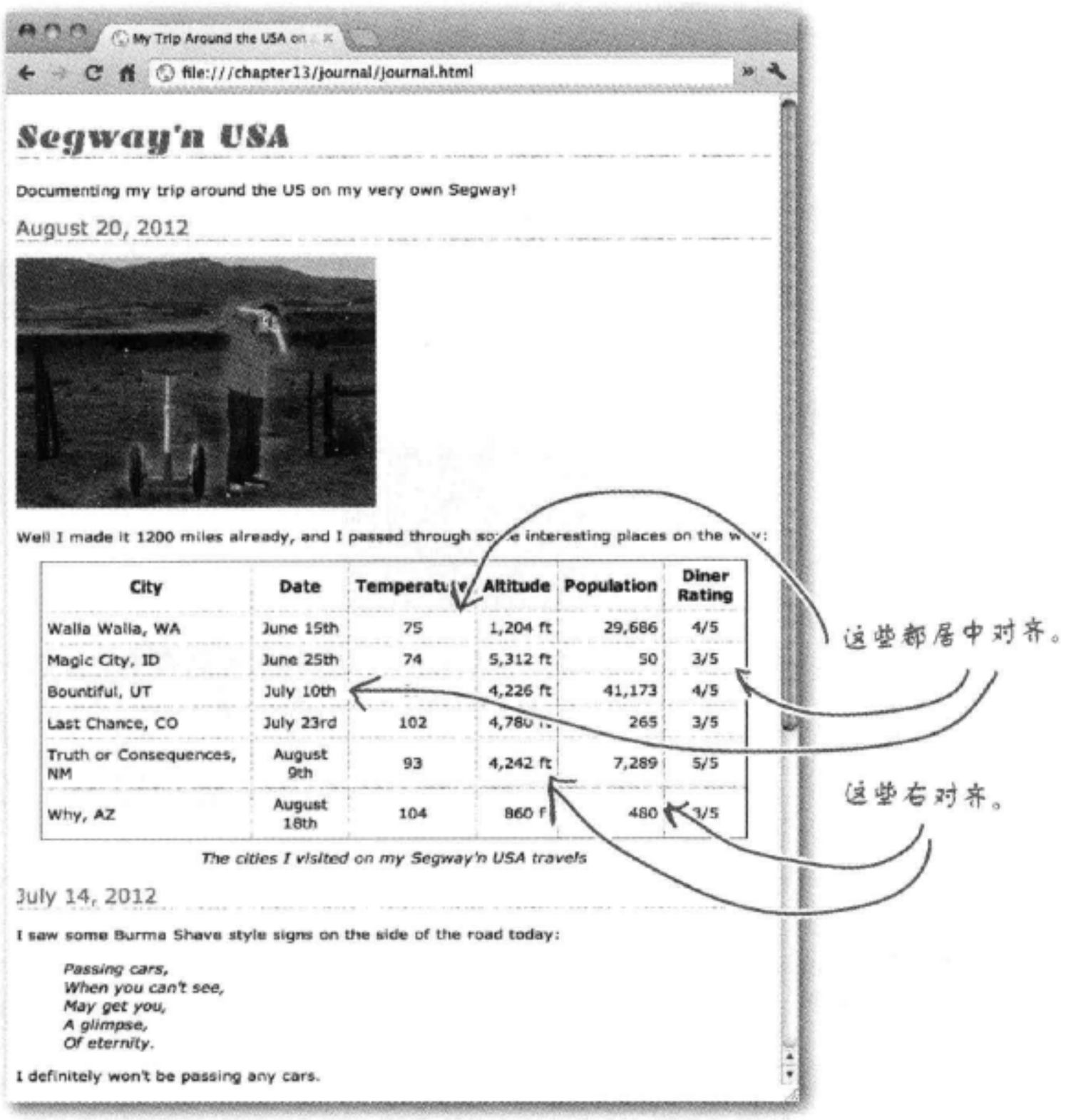


Sharpen your pencil

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

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

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

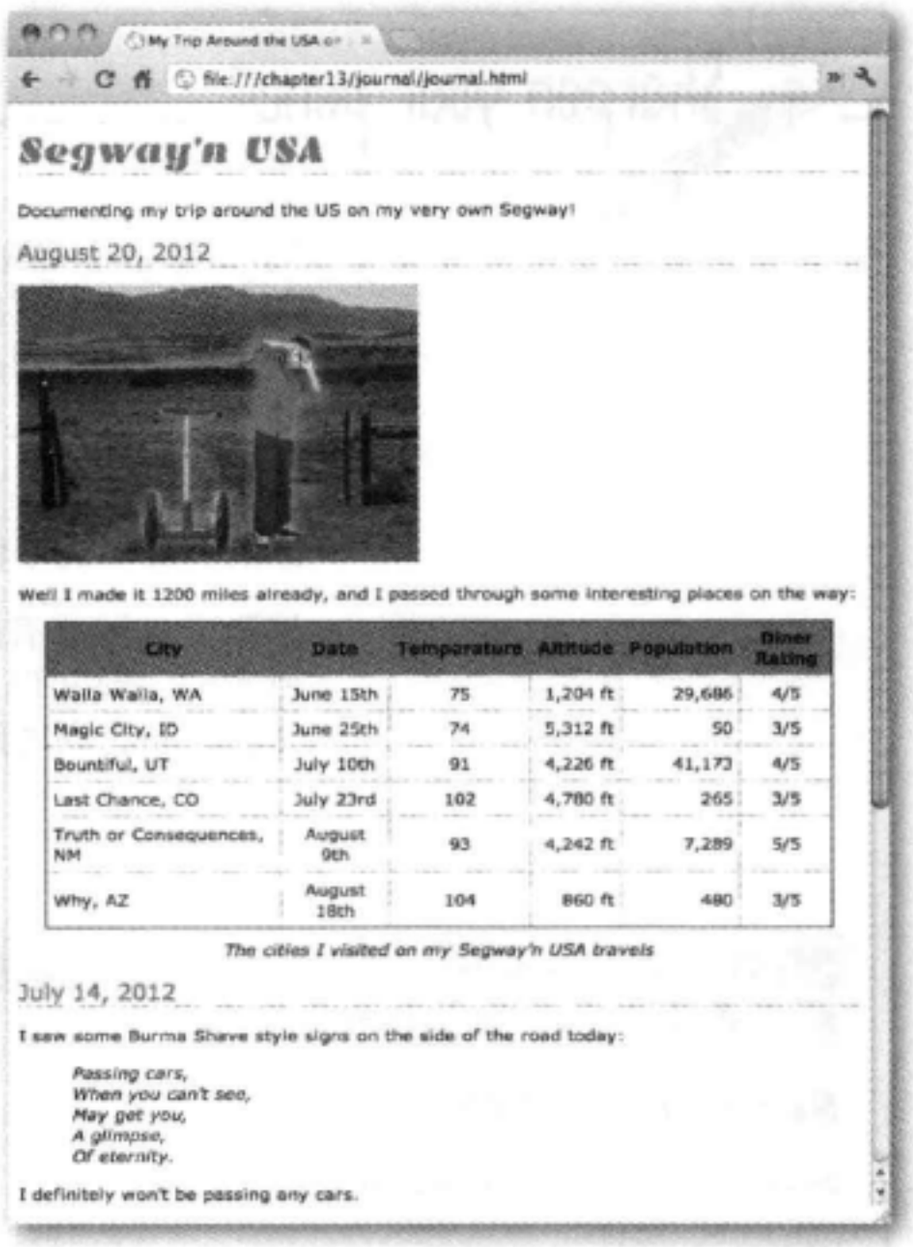


来点颜色怎么样？

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

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

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



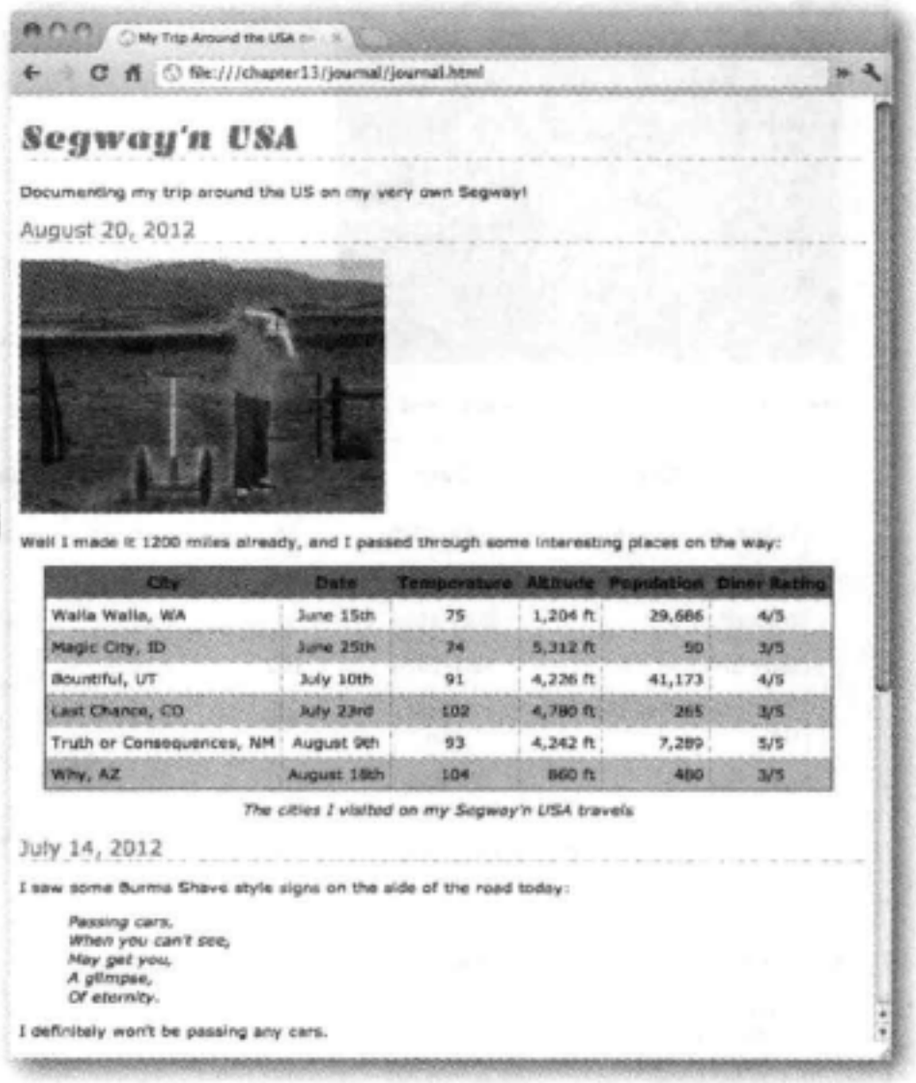
表行里加点颜色怎么样？

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

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

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

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

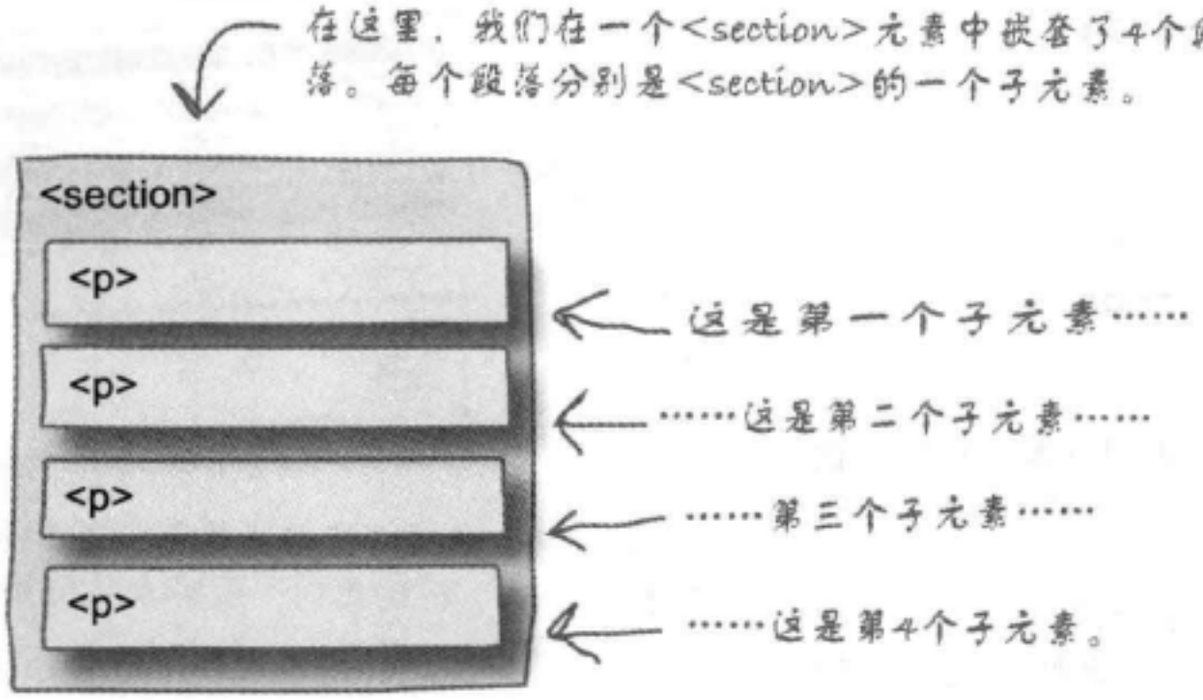


该轮到你了。把class“cellcolor”增加到“journal.css”的CSS中，然后在HTML中，对应要着色的各行，为相应的各个<tr>开始标记增加class="cellcolor"，让这些行有交替的颜色。继续学习后面的内容之前，先检查你的答案。

一些正式CSS

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

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



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

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

第2段和第4段有红色背景……

……第1段和第3段有绿色背景。

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

选择偶数<p>。

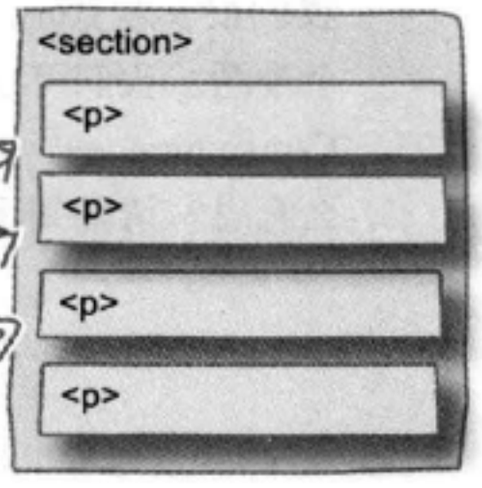
```
p:nth-child(2n) {
  background-color: red;
}
```

选择奇数<p>。

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

如果n=0，则2n=0（无段落），2n+1为1，这就是第1个段落。

如果n=1，则2n=2，第2个段落，另外2n+1=3，这是第3个段落。



正式练习

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

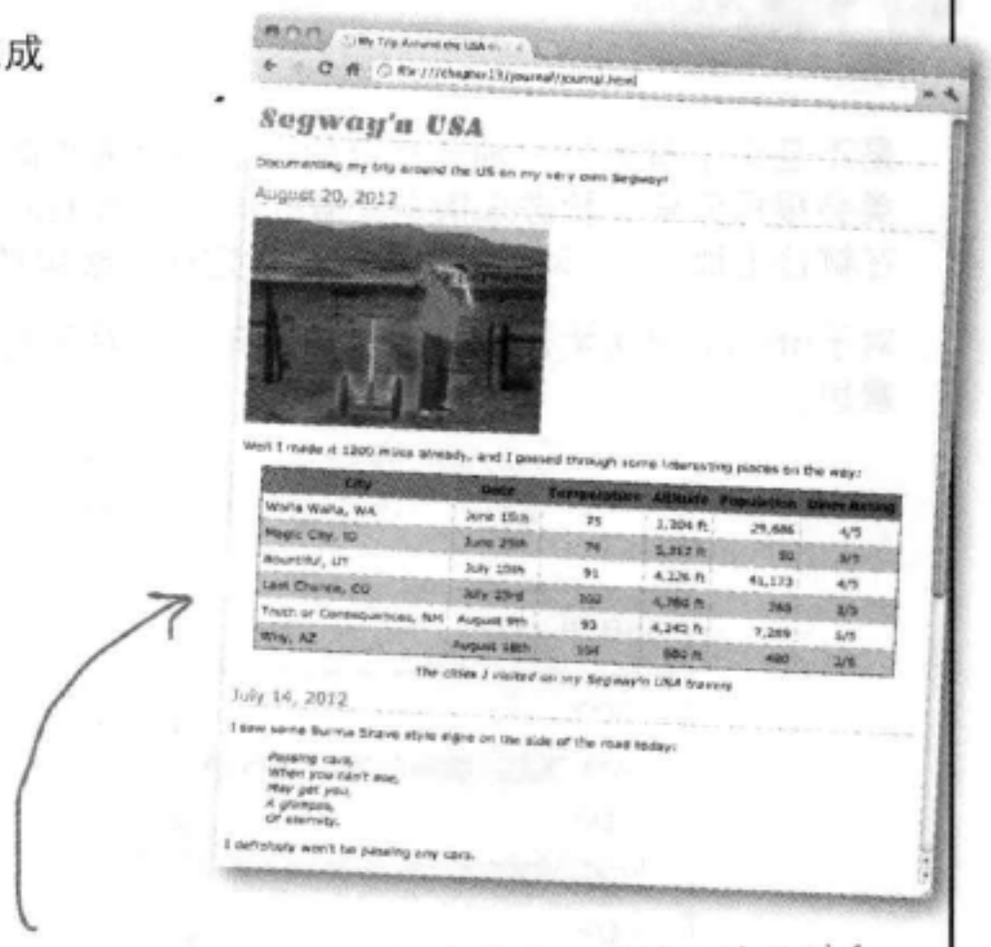
在这里写出你的伪类选择器。

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

将.cellcolor类注释掉，如下所示：

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

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

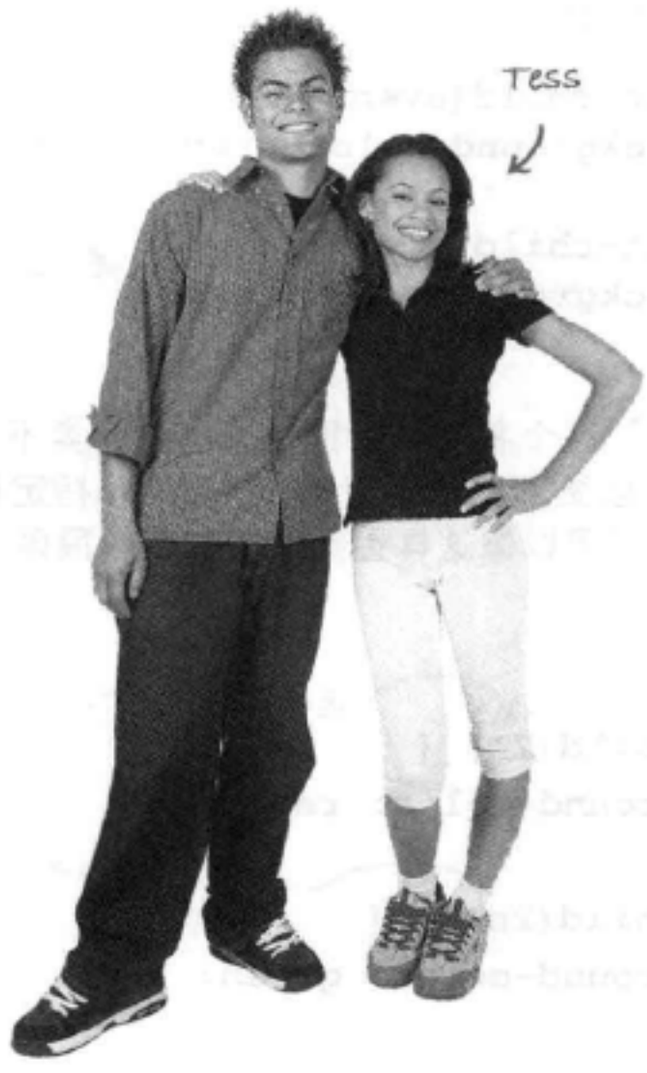


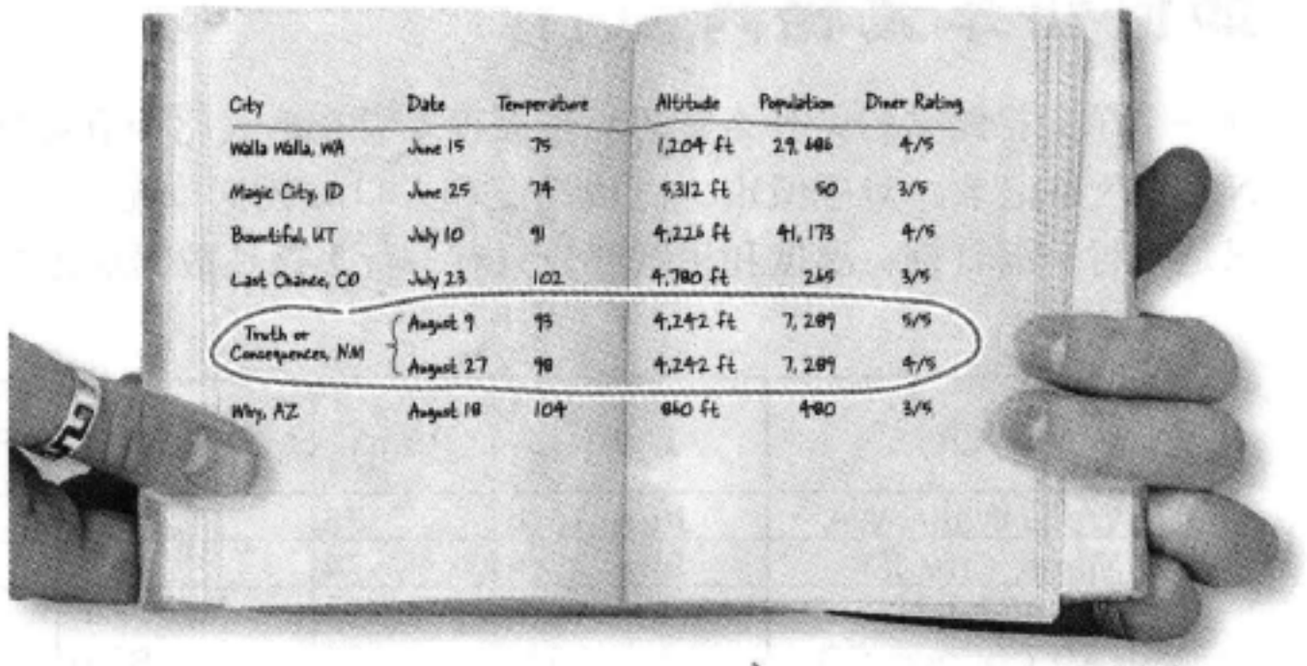
1、3、5和7行都有淡橙色背景。不过th的规则会覆盖“奇数”行的规则，所以表头仍然是深橙色。

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

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

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





再来看Tony的表格

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

这些都是Tony在Truth or Consequences的旅行记录。

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

这些表格数据单元格跨两行。

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

如何让单元格跨多行

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

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

这些单元格跨两行。

而日期、温度和用餐评分单元格分别只占一行。

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

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

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


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

这里是包含新墨西哥州数据的两个表行。

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

由于rowspan设置为2，所以不再需要城市。

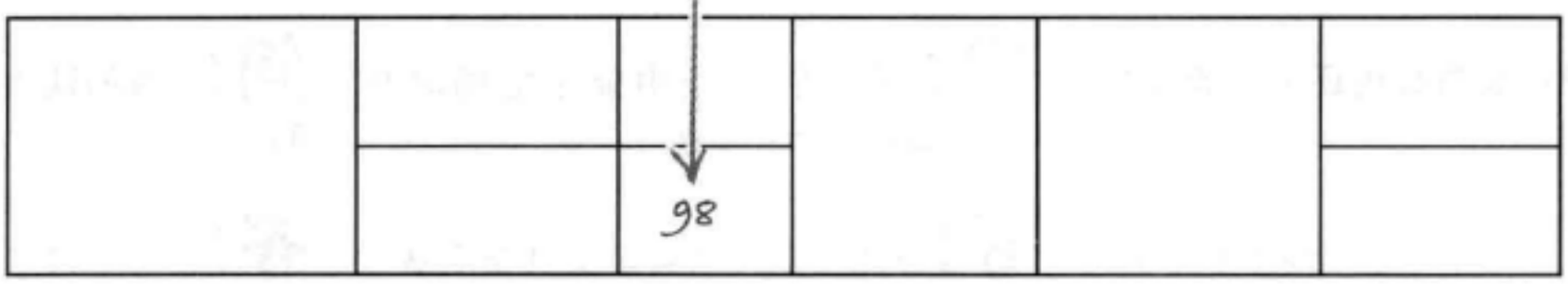
在第二行中，只指定我们需要的列（日期、温度和新的用餐评分）。

海拔高度和人口也一样。

WHO DOES WHAT?

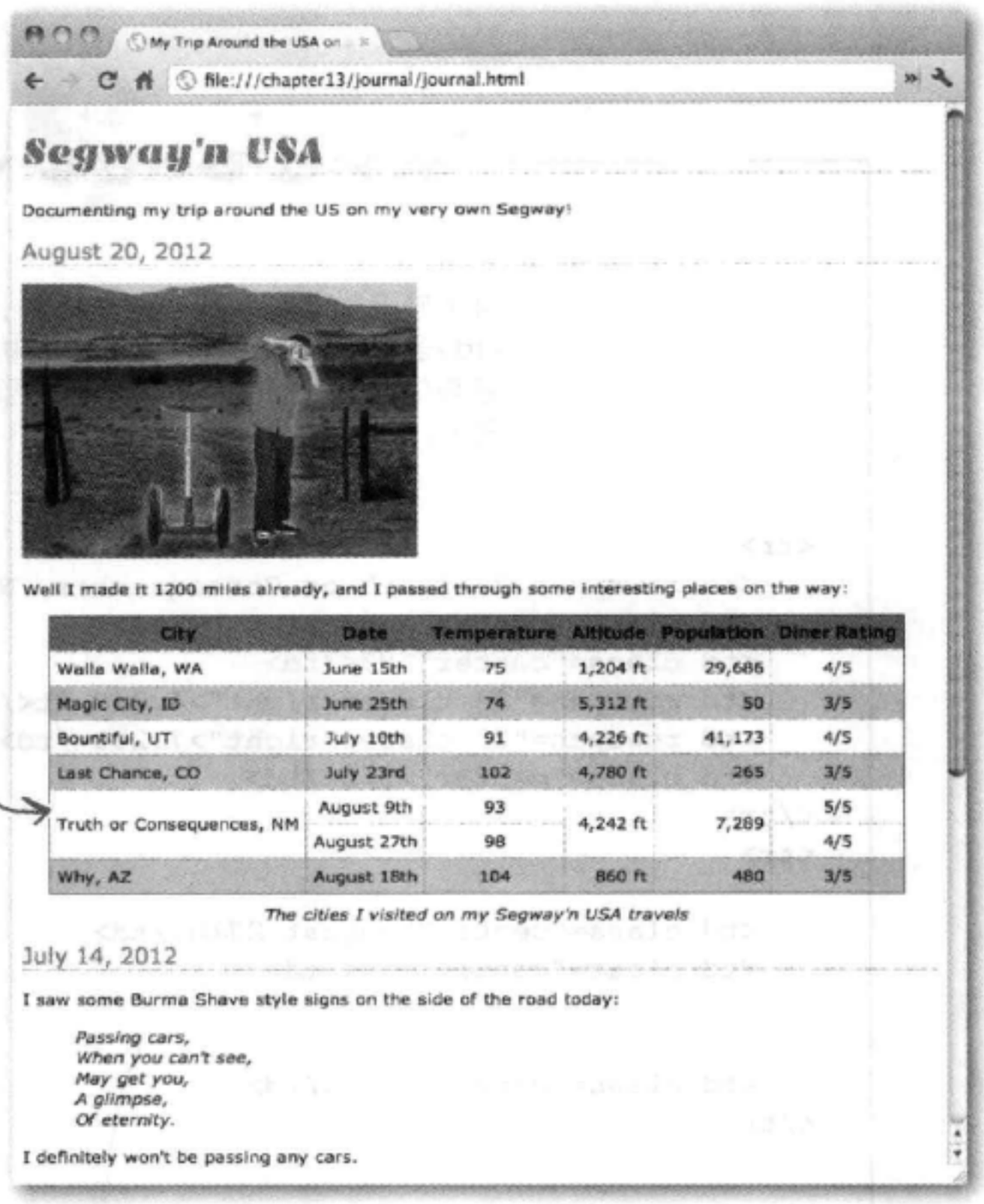
为了确保你真正掌握了这些内容，请在下面找出正确的<td>表格单元格数据，填写表格中的各个单元格。我们已经帮你填了一个空。学习后面的内容之前，请先检查你的答案。

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

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


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



测试表格

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



现在我们得到了一个漂亮的表格，其中不再有冗余的信息，看起来真的很棒！

there are no Dumb Questions

问：你说过表格数据还可以跨多列，是吗？

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

问：同一个<td>中能不能同时有colspan和rowspan？

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

问：你真的认为这些跨行看起来更好吗？

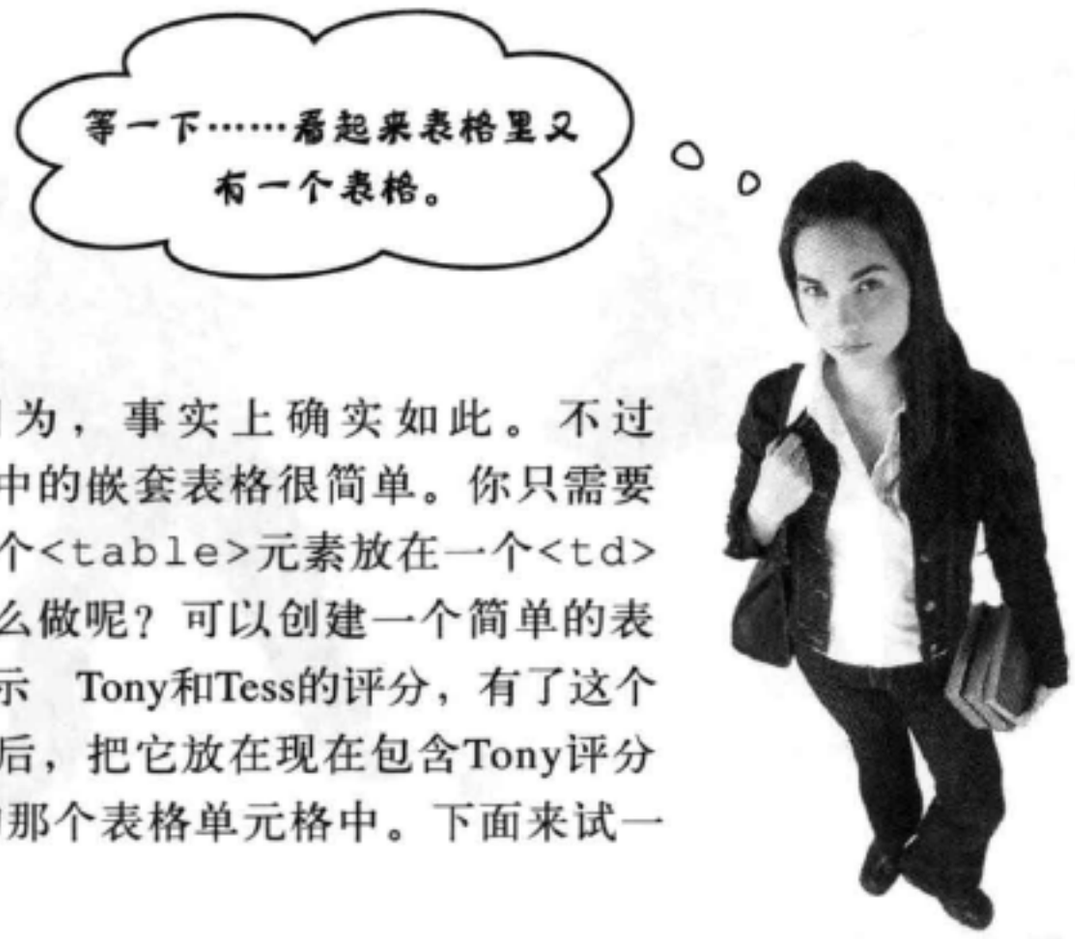
答：嗯，它们可以减少表格中的信息量，这通常是件好事。另外，如果看看现实世界中的表格，你会发现跨行和跨列相当常见，所以能够在HTML中做到跨行和跨列确实很棒。不过，如果你更喜欢以前的表格，当然也可以修改你的HTML，改回为原先的版本。

天堂也有麻烦？

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

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

为什么不把他们俩的评分都放在表格里？这样一来，我们就能得到更准确的信息。



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

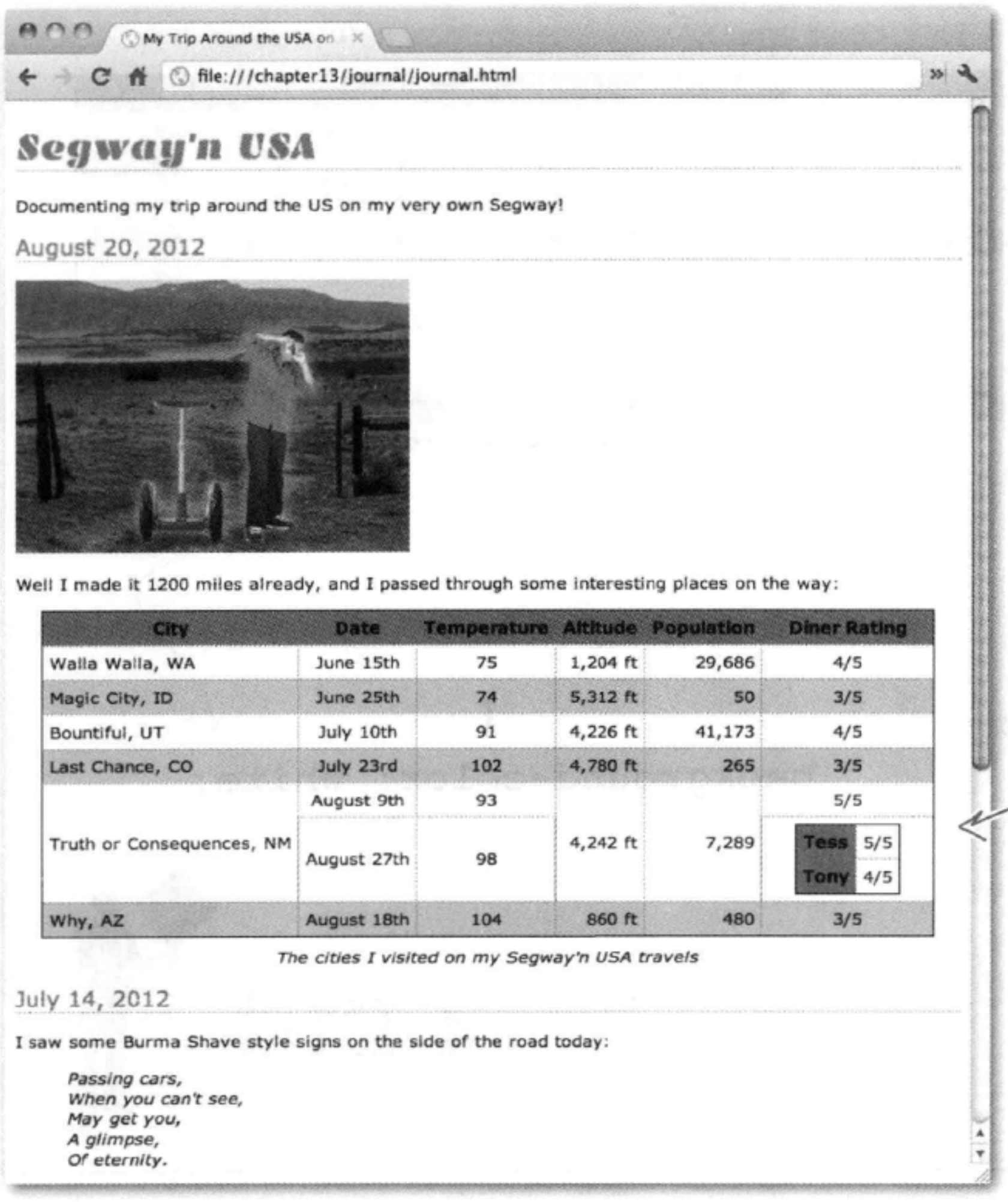
```
<tr>
   <td rowspan="2">Truth or Consequences, NM</td>
   <td class="center">August 9th</td>
   <td class="center">93</td>
   <td rowspan="2" class="right">4,242 ft</td>
   <td rowspan="2" class="right">7,289</td>
   <td class="center">5/5</td>
</tr>
<tr>
   <td class="center">August 27th</td>
 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

首先，删除原来Tony的评分……

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

测试嵌套表格

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



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

BRAIN BARBELL

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

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

The cities I visited on my Segway'n USA travels

我们想把嵌套表格表头的背景颜色改为白色。

确定选择器，只选择嵌套表格的表头元素。

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

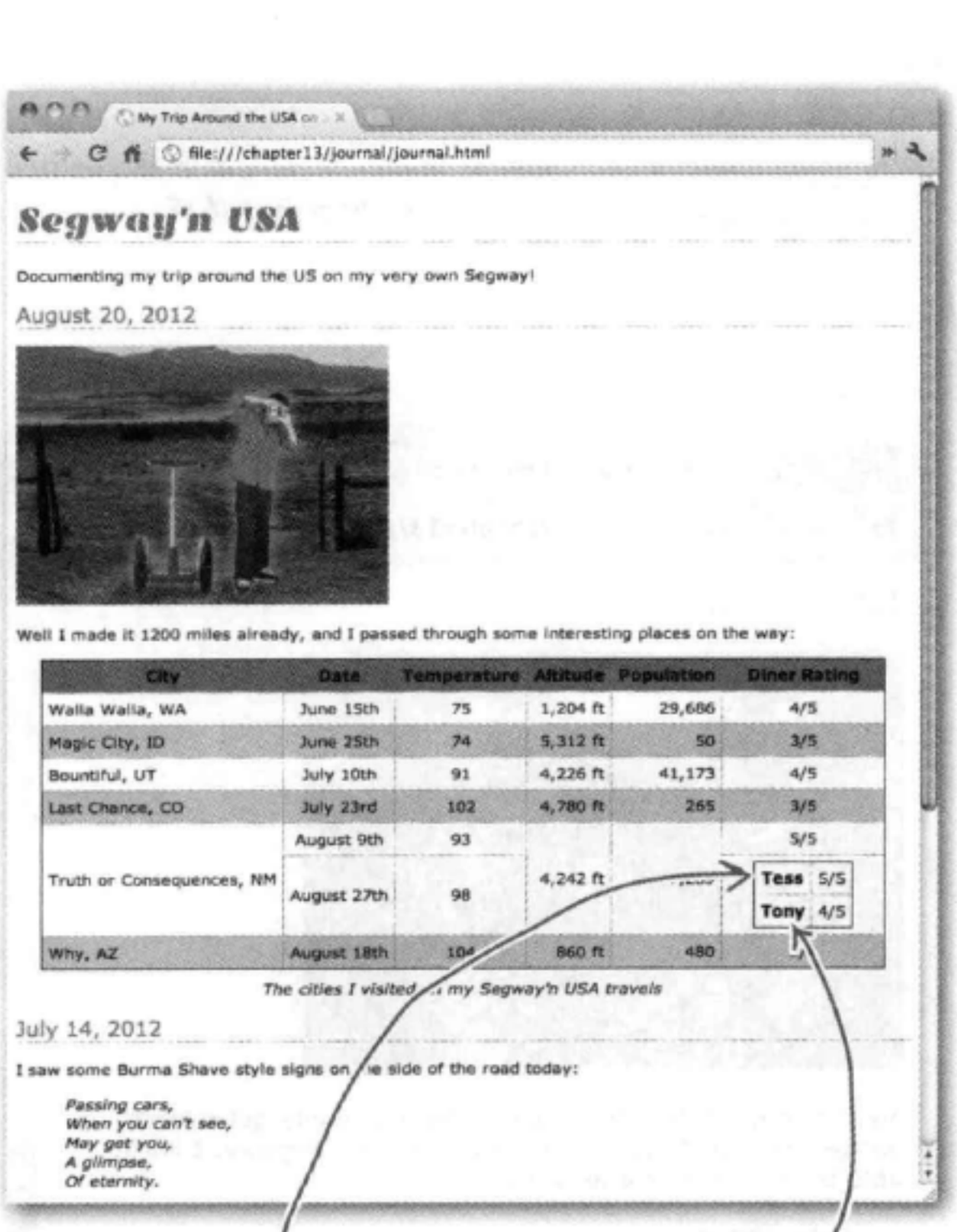
停！完成这个练习之前，先别看下一页！

覆盖嵌套表格表头的CSS

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

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

现在保存对“journal.css”文件的修改，并重新加载页面。



there are no Dumb Questions

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

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

这个方案也是可行的吧？

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

BRAIN POWER

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

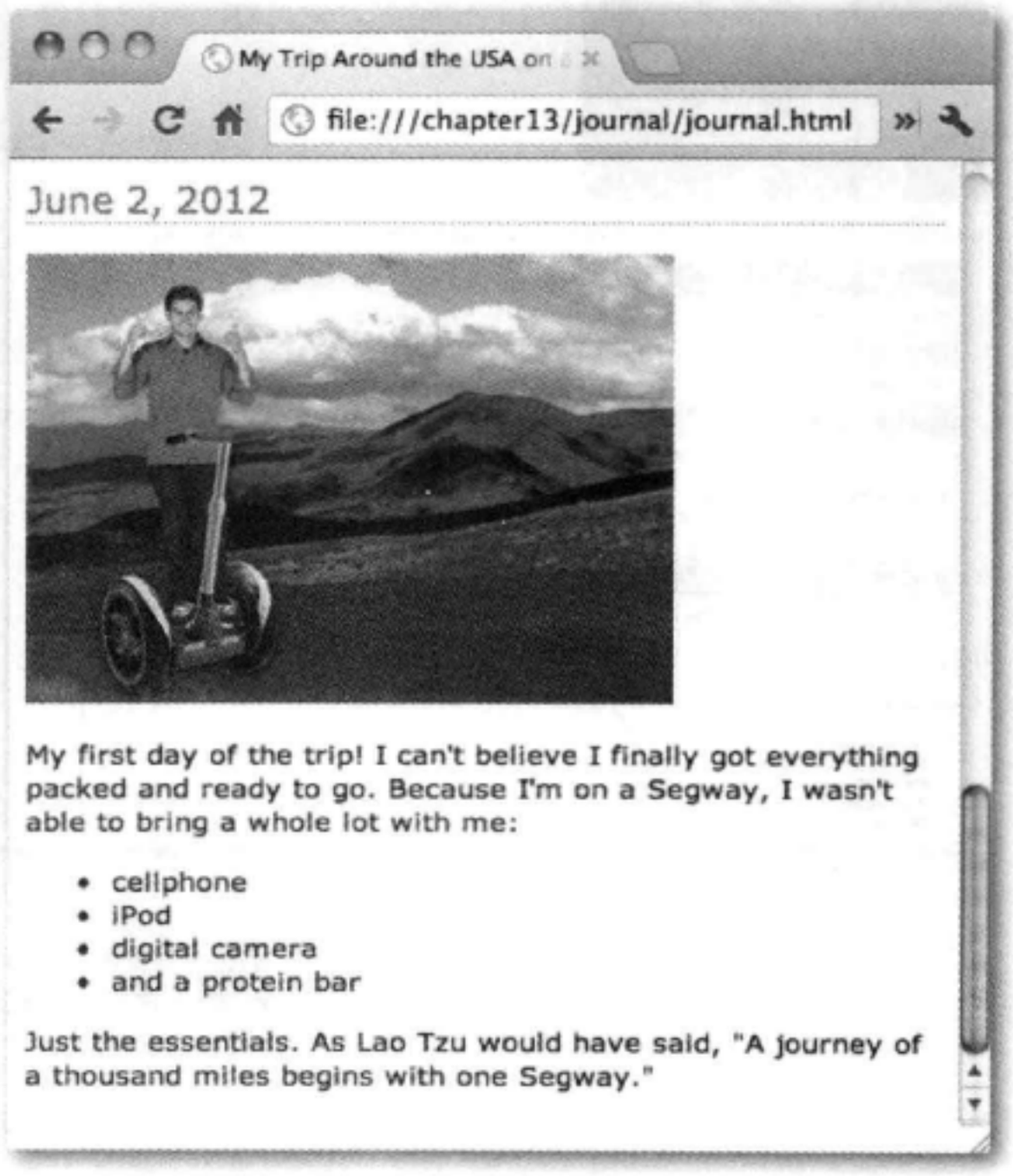
对Tony网站的最后润色

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

我们只查看6月2日日志的HTML片段。

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

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

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

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



现在的清单是这样的。

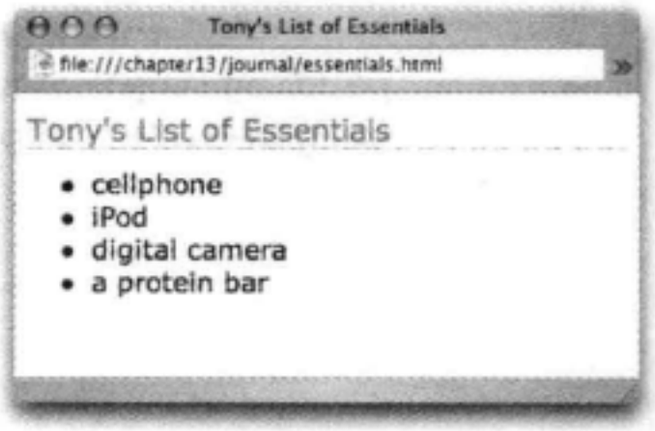
为列表增加一些样式

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

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

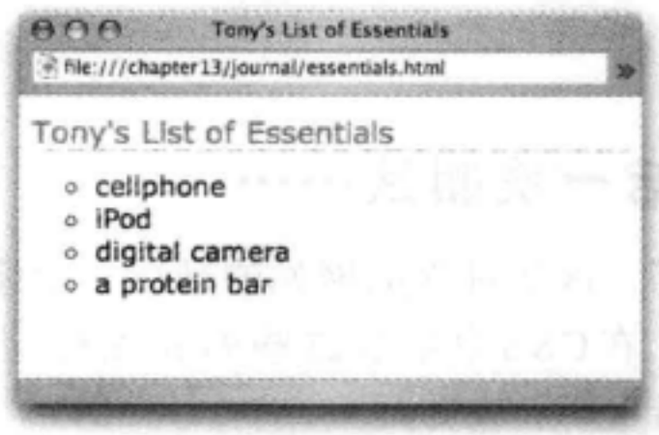
```
li {
    list-style-type: disc;
}
```

disc是默认的列表标记类型。



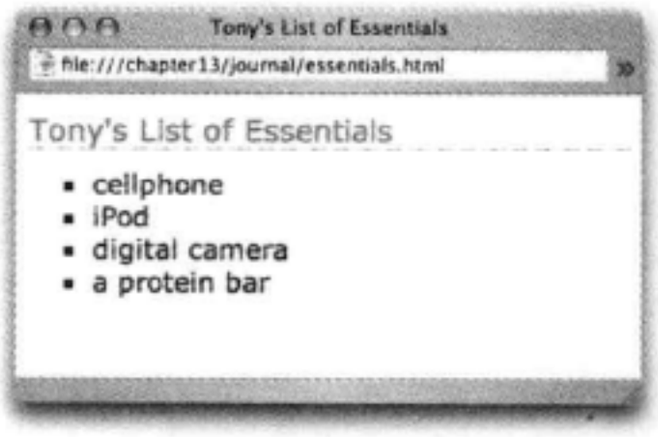

```
li {
    list-style-type: circle;
}
```

circle属性值提供一个简单的圆形标记。



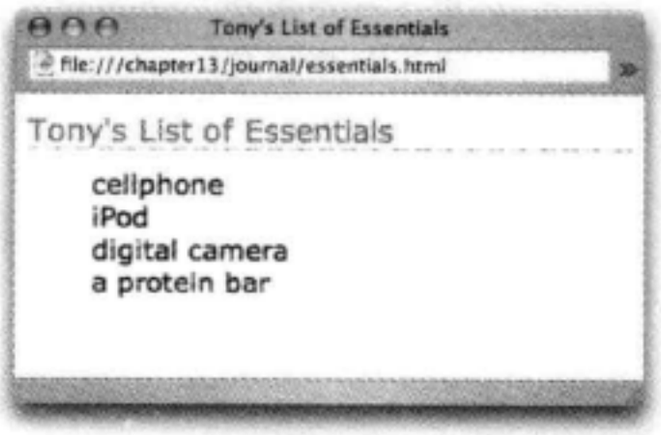

```
li {
    list-style-type: square;
}
```

square属性值提供一个方块标记。




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

值为none时，会删除所有列表标记。



如果需要一个定制标记呢？

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

这里是list-style-image属性，我们把它设置为一个URL。

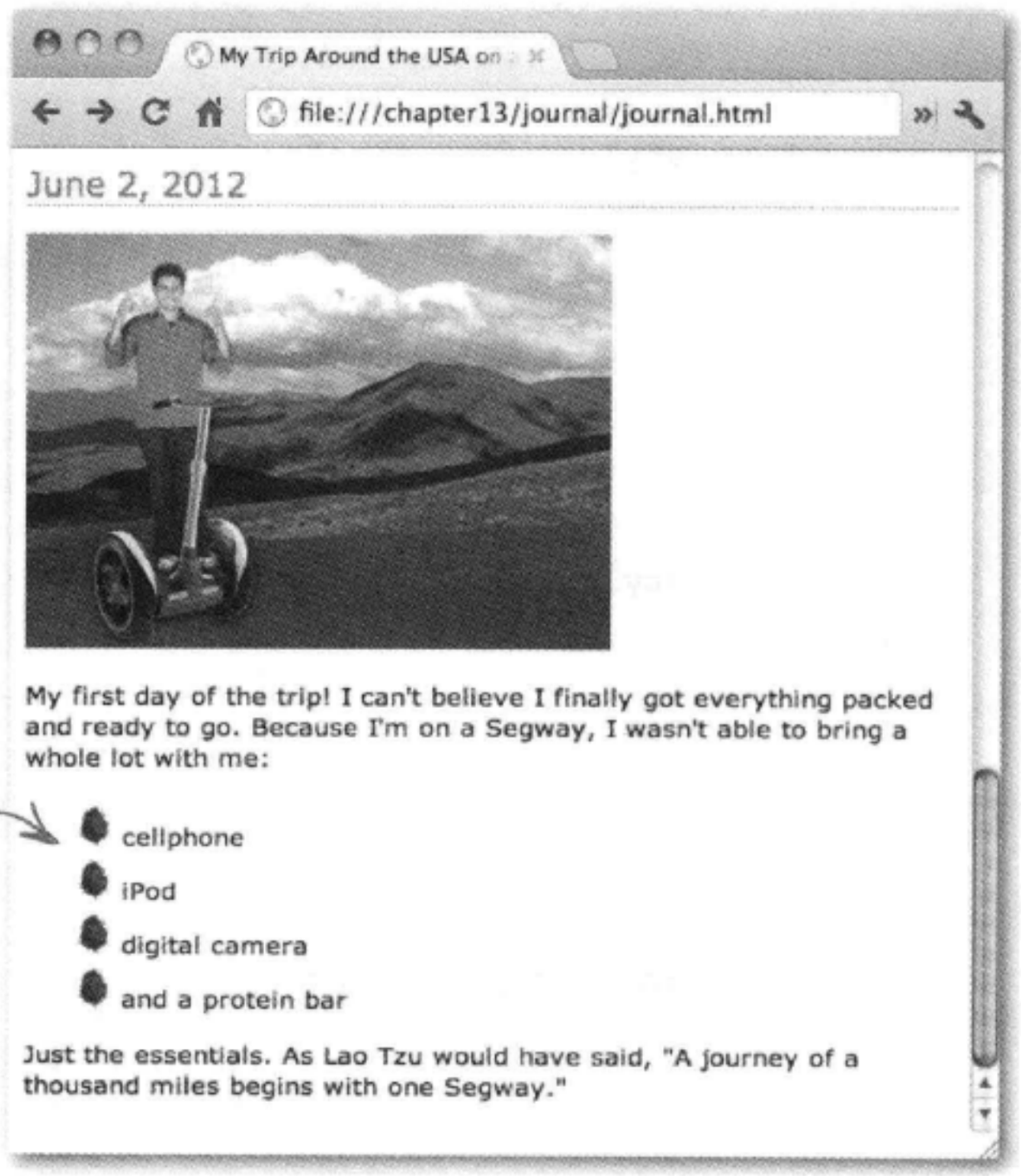
```
li {
    list-style-image: url(images/backpack.gif);
    padding-top: 5px;
    margin-left: 20px;
}
```

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

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

最后一次测试……

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



来看这里的列表，现在列表标记换成了一个图像，而且增加了一些外边距和内边距。

there are no Dumb Questions

问：有序列表呢？我能改变有序列表的样式吗？

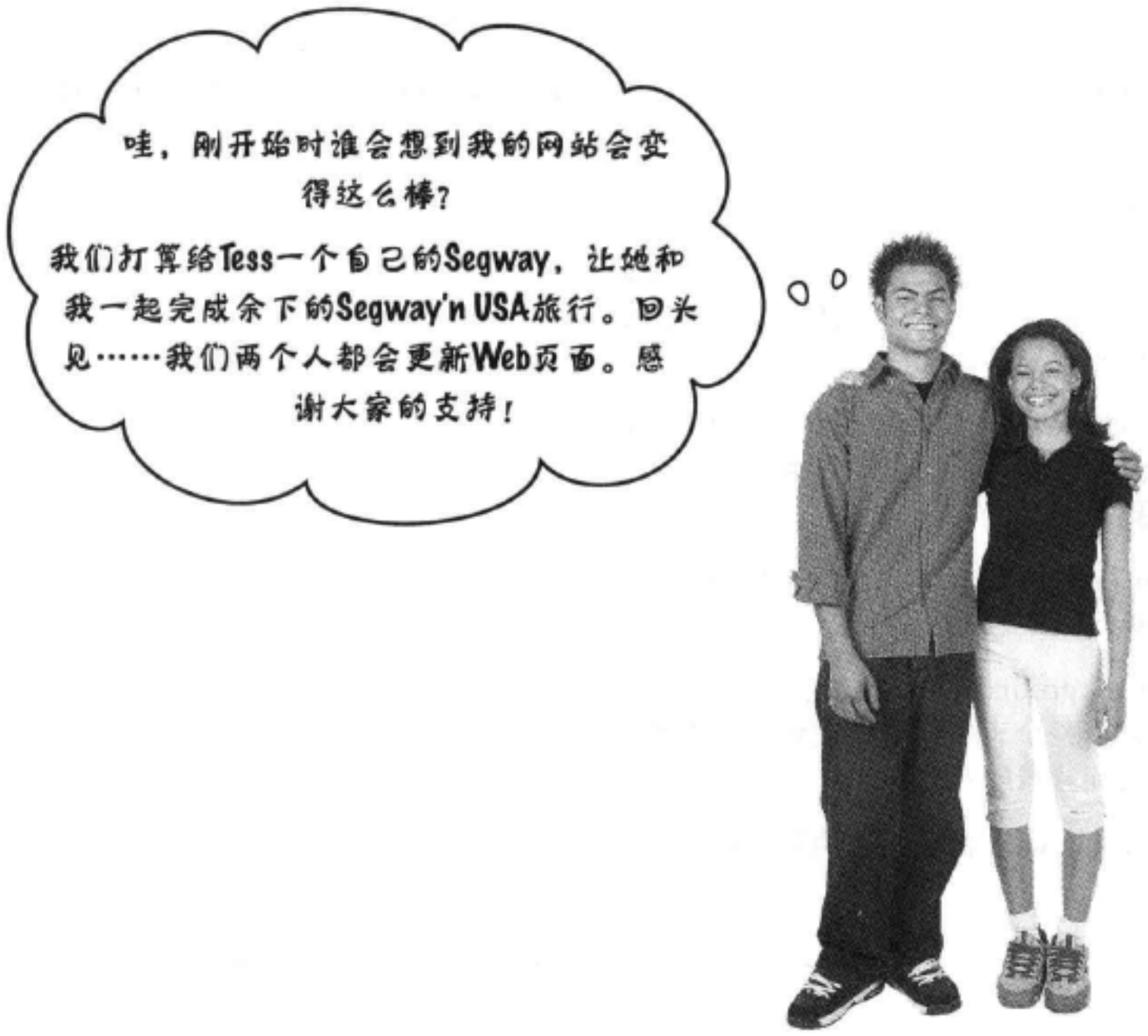
答：不论是有序列表还是无序列表，指定样式的方法都是一样的。当然，有序列表会用一个数字或字母序列作为列表标记，而不是使用项目符号。利用CSS，你可以用list-style-type属性控制一个有序列表的标记是十进制数字、罗马数字还是字母表字符（如a、b、c）。常用的值包括decimal（十进制数）、upper-alpha（大写字母）、lower-alpha（小写字母）、upper-roman（大写罗马数字）和lower-roman（小写罗马数字）。可以找一本CSS参考书来了解更多选择（有很多这样的参考书）。

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

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

问：你确定没错吗？好像反了吧。

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

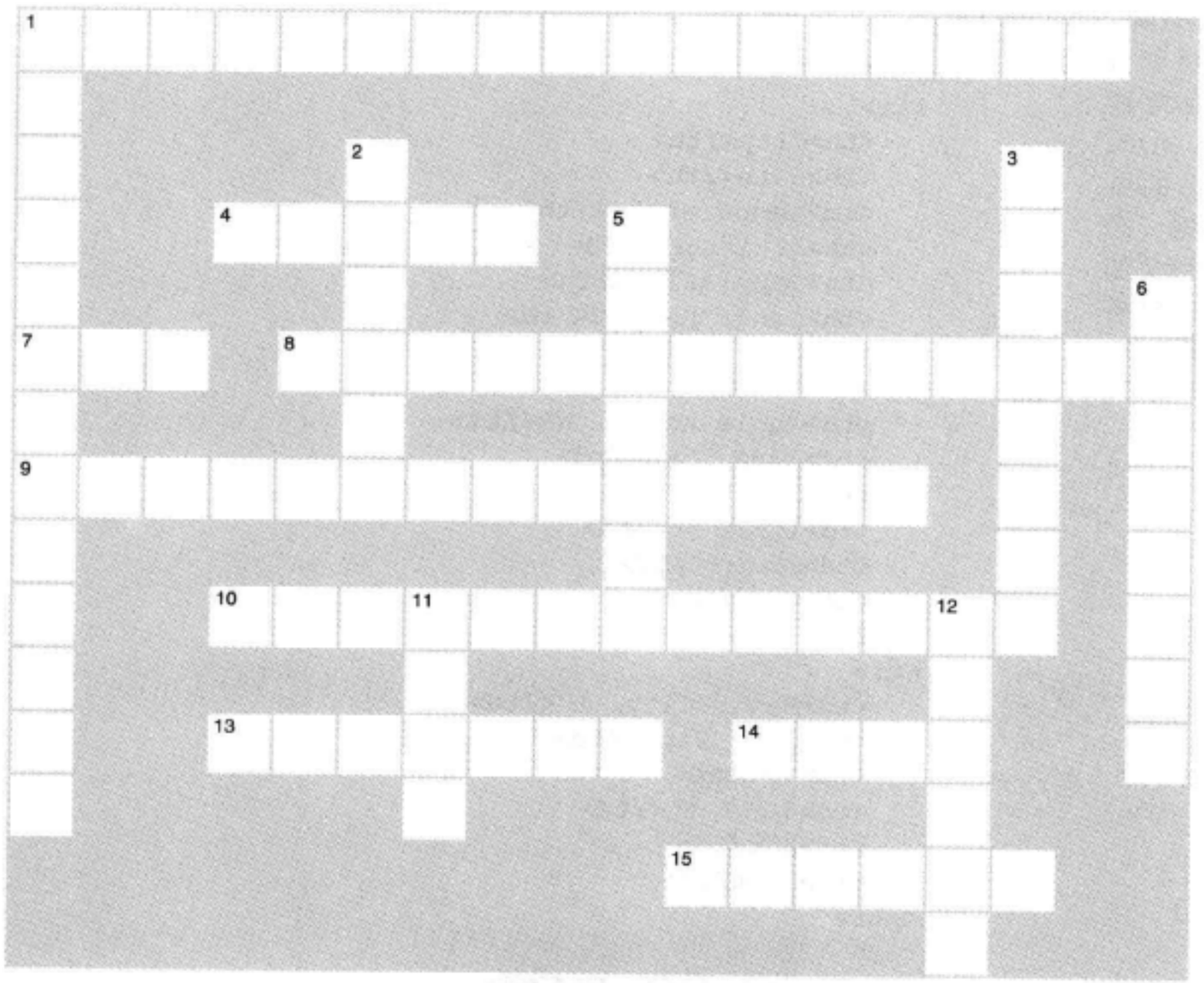


BULLET POINTS

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

HTML填字游戏

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



横向

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

纵向

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

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

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

继续看下一页。


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

Testing Tony's Table

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

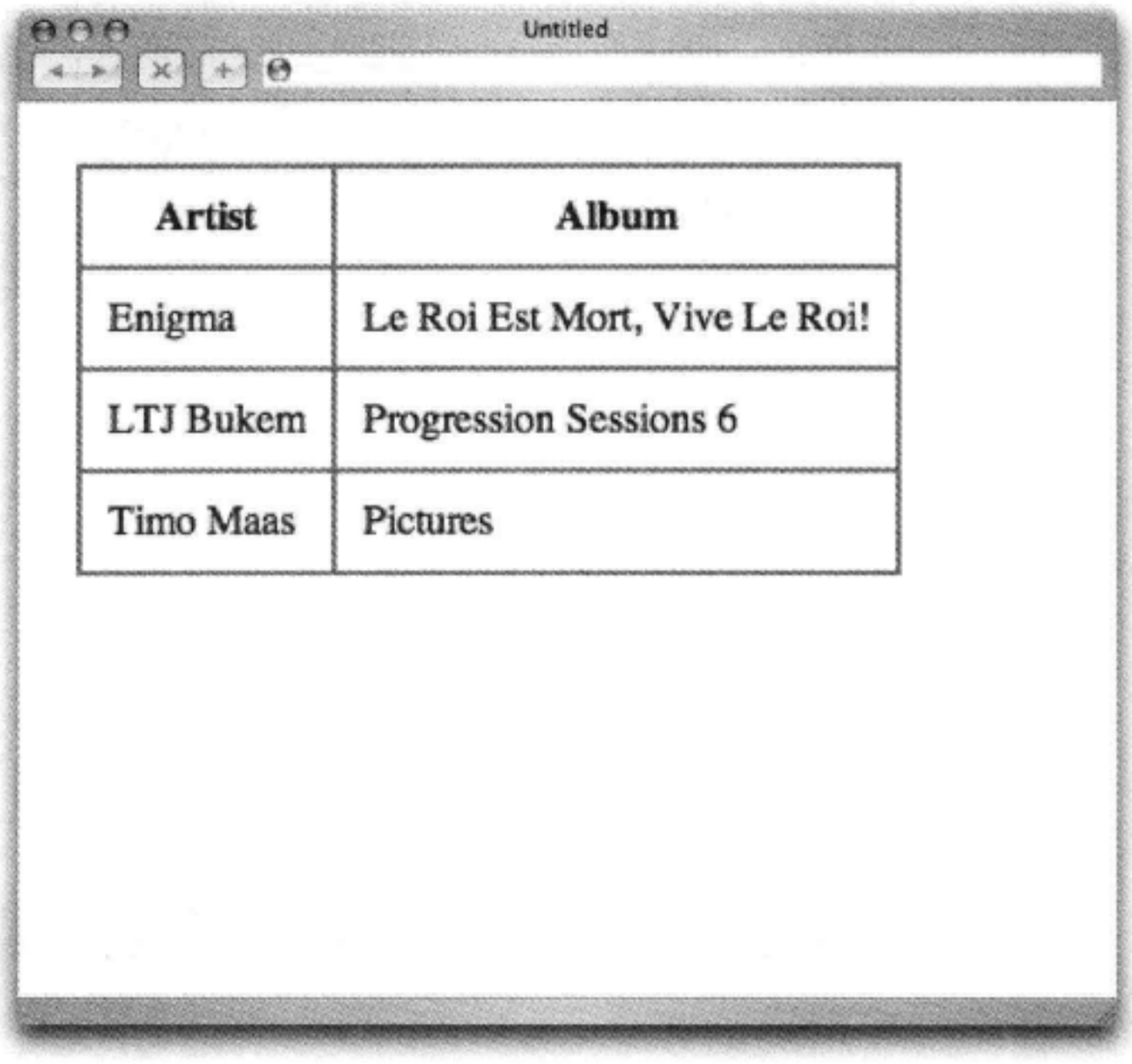
City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5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

扮演浏览器答案

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

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

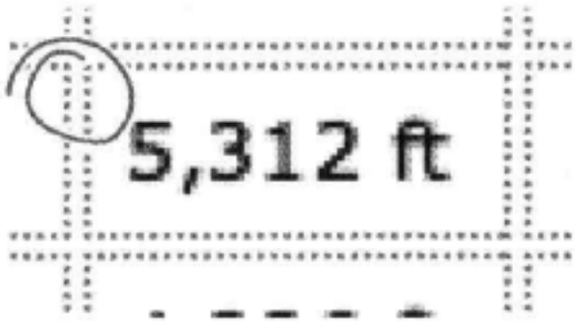
我们调整了这个HTML的格式，这样能让人更容易阅读。



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

Sharpen your pencil Solution

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



可以使用border-spacing设置间距为0，这样两条线就会紧挨着。

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

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

Sharpen your pencil Solution

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

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

这里有两个类，一个对应居中对齐，另一个对应右对齐。

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

这里只需要把各个<td>增加到适当的类！

Exercise Solution

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

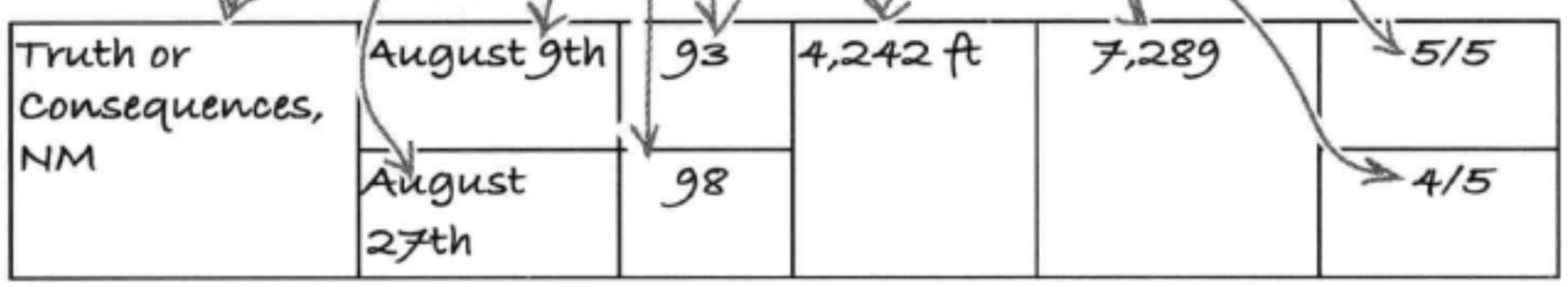
```
<tr class="cellcolor">
  <td>Magic City, ID</td>
  ...
</tr>
```

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

WHO DOES WHAT? 答案

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

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

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


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



正式练习答案

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

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

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

BRAIN BARBELL 答案

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

我们使用了一个子孙选择器，只选择嵌套表格表头。可以这样做：

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

(1) 先选择外表……

(2) 再选择内表……

(3) 然后选择表头。

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

确定选择器只选择嵌套表格的表头元素。

HTML填字游戏答案

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

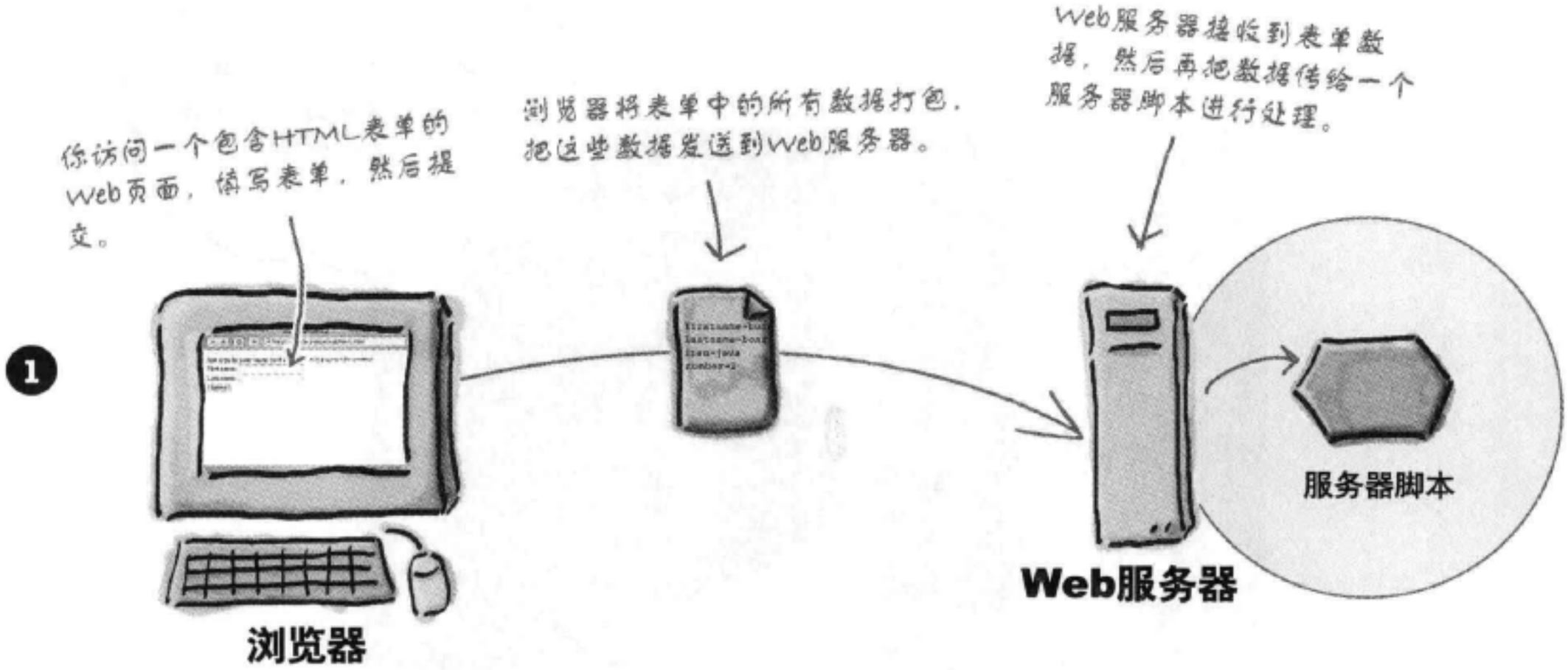
14 HTML表单

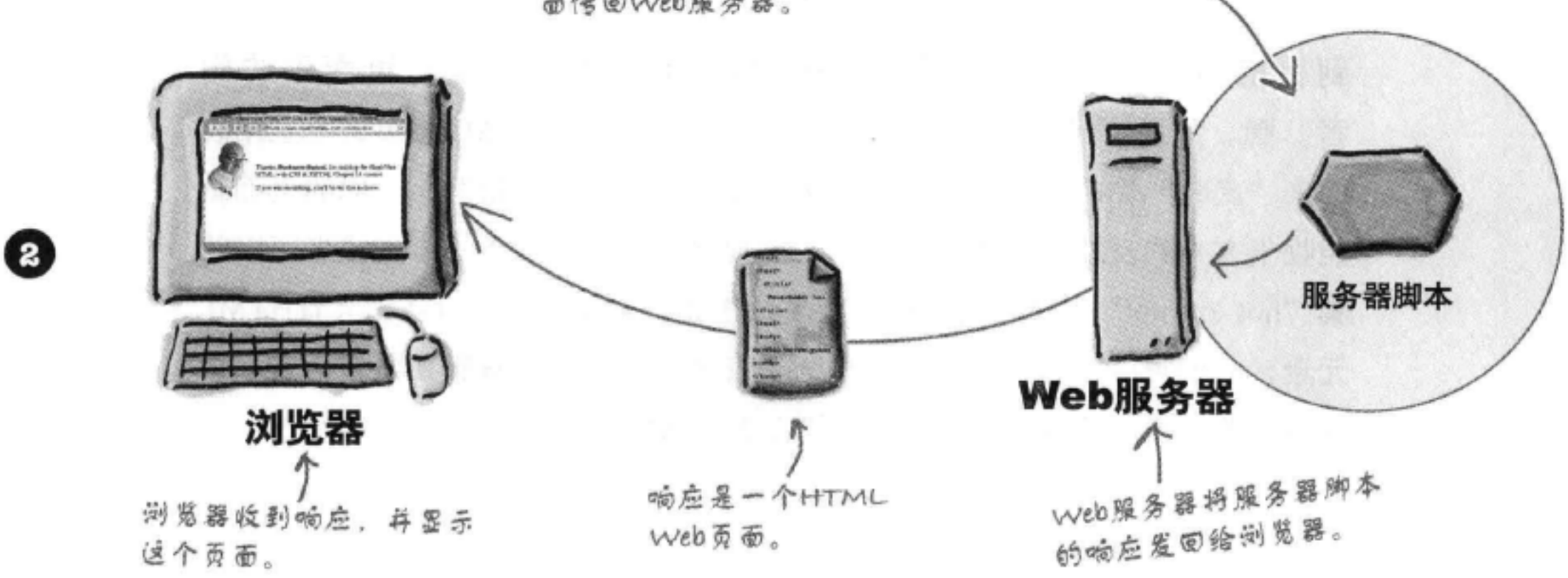
实现交互

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

表单如何工作

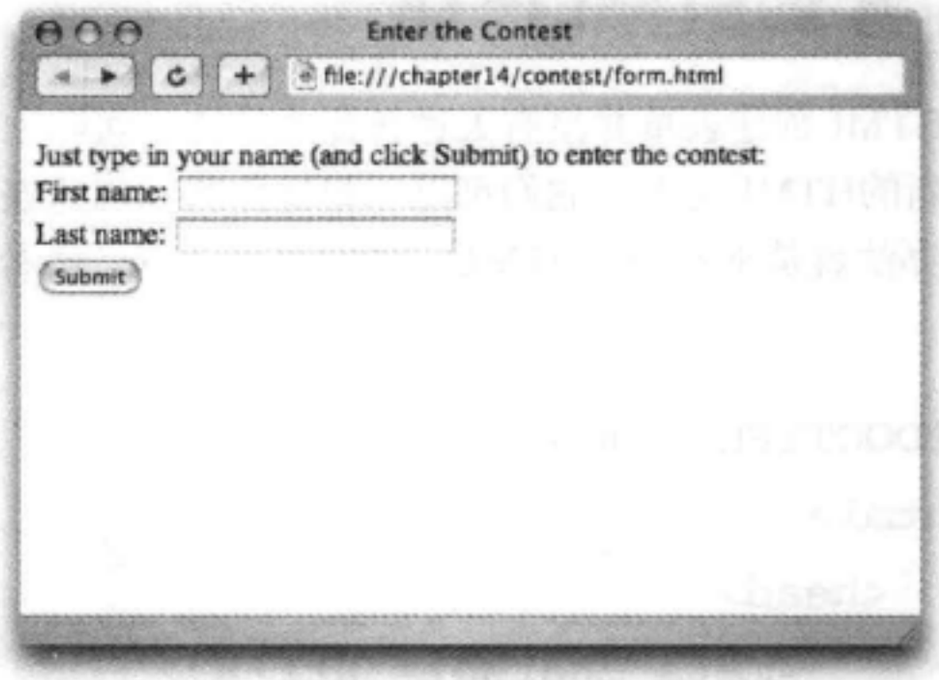
只要你接触过Web，就应该知道表单是什么。不过，你可能没有好好想过表单与HTML到底有什么关系。表单实际上就是一个包含输入域的Web页面，允许你输入信息。提交表单时，这些信息会打包并发送到一个Web服务器，由一个服务器脚本处理。处理完成时，你会得到什么？当然会得到另一个Web页面作为响应。下面来仔细分析这个过程：





表单在浏览器中如何工作

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

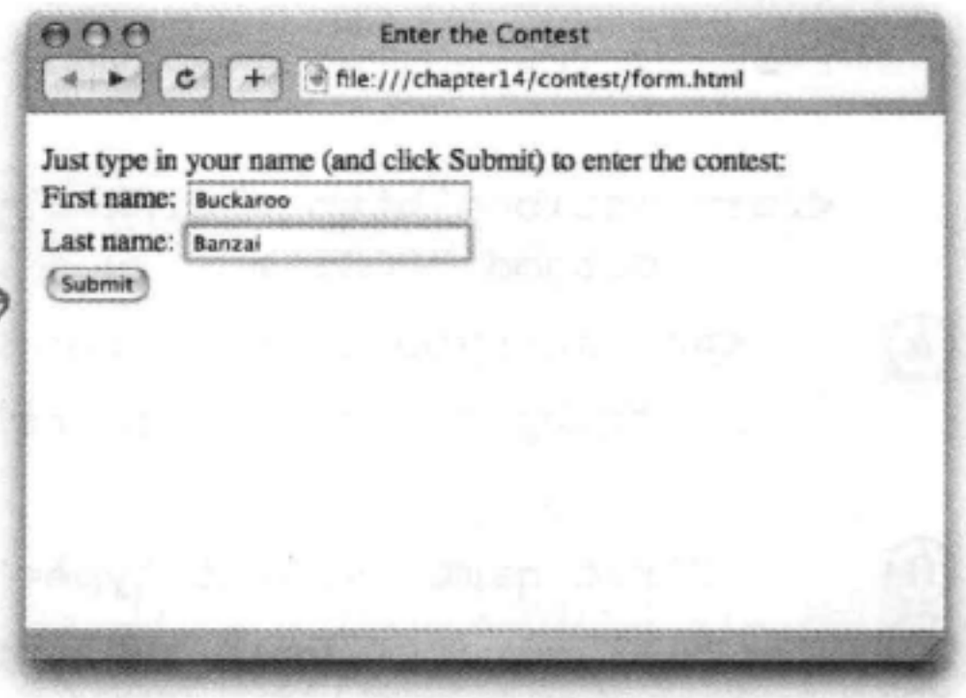


浏览器加载页面

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

你输入数据

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

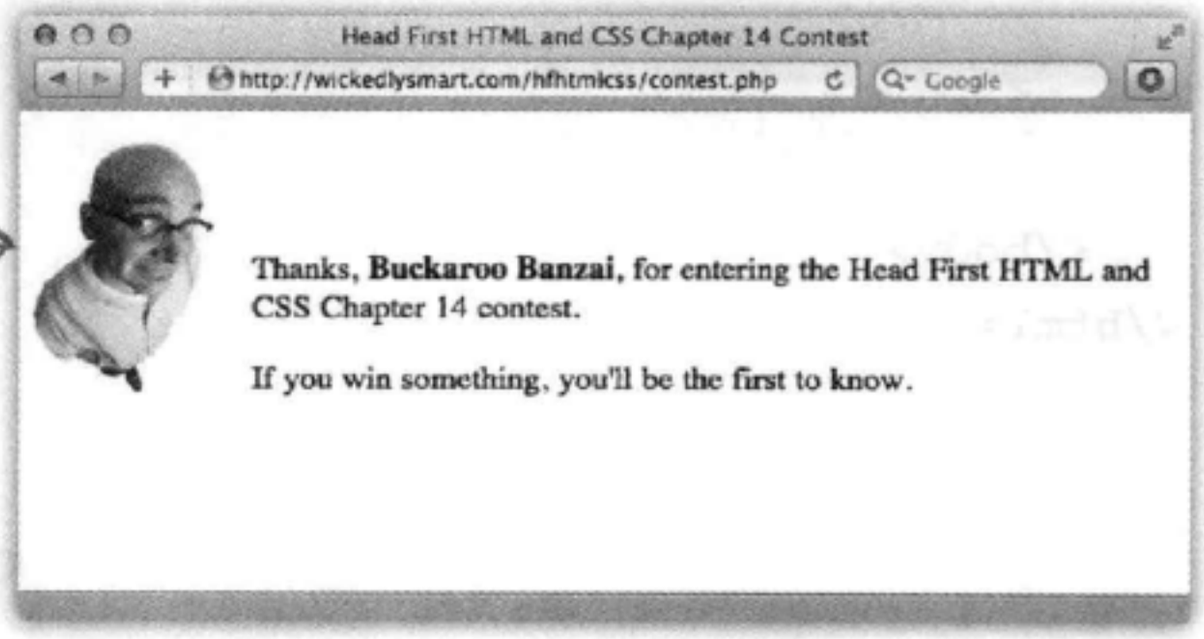


你提交表单

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

服务器响应

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



你写的HTML代码

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

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

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

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

    </body>
</html>
```

现在这些对你来说已经很熟悉了。

这里是表单。

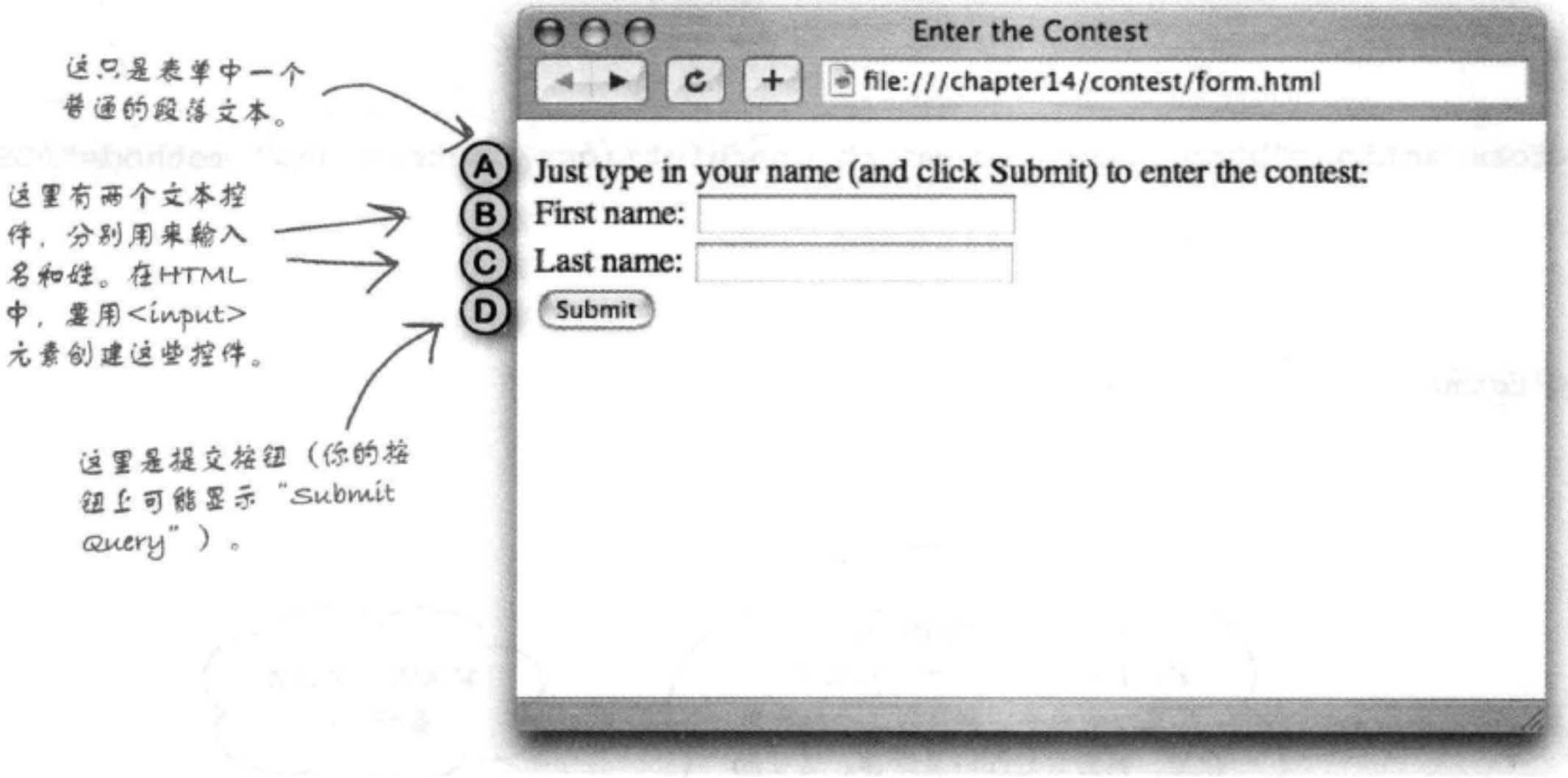
这里是<form>元素本身……

……还有一大堆嵌套在其中的元素。

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

浏览器创建的页面

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



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

<form>元素如何工作

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

这是开始标记。表单中的所有内容都包含在这里。

action属性包含Web服务器的URL……

……脚本所在的文件夹……

……将处理表单数据的服务器脚本的文件名。

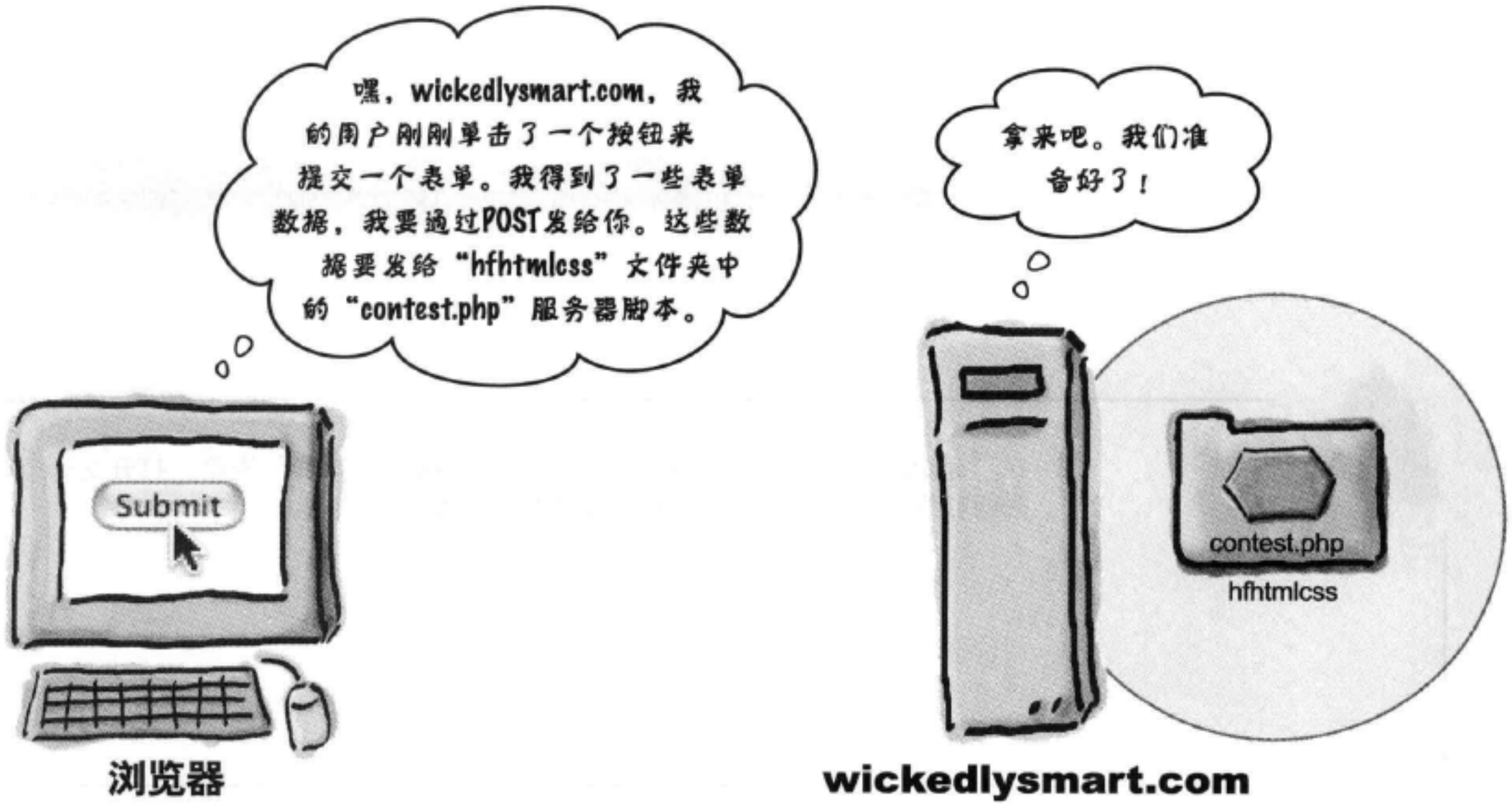
method属性确定表单数据如何发送到服务器。我们将使用最常用的方法：POST。这一章后面还会介绍发送数据的其他方法，另外会解释为什么使用（或不使用）POST。

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

表单的所有内容都放在这里……

```
</form>
```

……结束标记结束这个表单。



问得好。

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

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

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

表单里可以有什么？

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

文本输入

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

Name: Buckaroo Banzai

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

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

使用type属性指示你希望得到一个“文本”输入。

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

这个<input>元素是一个void元素，所以后面没有内容。

注意它们都使用相同的HTML元素，只是type属性值不同。

提交输入

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

Submit

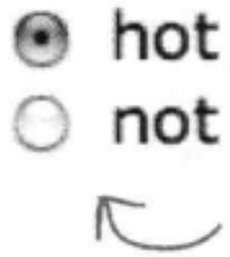
这个按钮的标签默认为“Submit”（提交），或者也可能是“Submit Query”（提交查询），不过你可以改变这个标签（后面会告诉你怎么做）。

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

提交按钮要指定“submit”作为<input>元素的type属性值。

单选钮输入

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



单选钮只允许从一组选项中选择其一。

每个选项使用一个radio <input>。

与一组给定选项关联的单选钮必须有相同的名字……

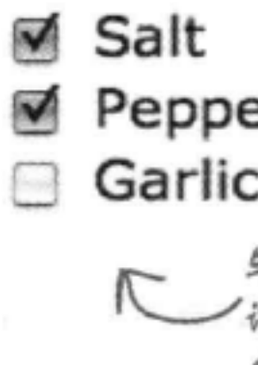
……不过每个选项可以有不同的值。

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

这里也是一样，我们仍使用<input>元素，只是有不同的type值。

复选框输入

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



与单选钮不同，复选框允许在一组选项中选择0个或多个选项。

类似于单选钮，对各个选项使用相同的checkbox <input>元素。

相关的复选框也共用一个名字。

每个复选框有一个不同的值。

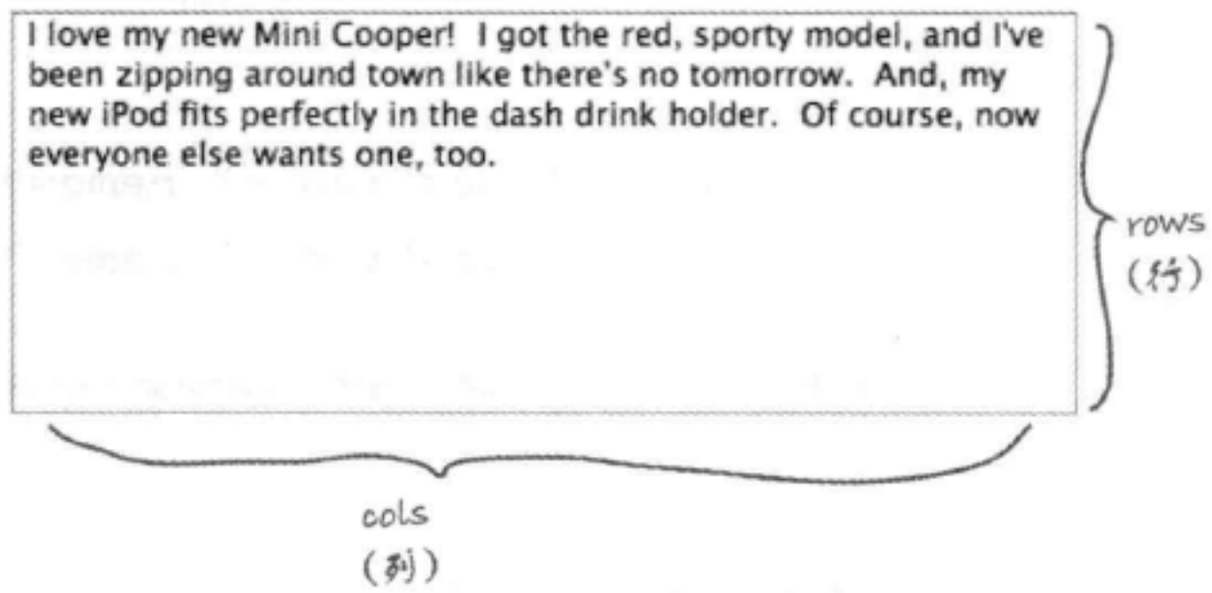
```
<input type="checkbox" name="spice" value="Salt">
<input type="checkbox" name="spice" value="Pepper">
<input type="checkbox" name="spice" value="Garlic">
```

表单里可以有什么（第2部分）

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

文本区

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



<textarea>元素不是一个空元素，所以它有开始和结束标记。

使用name属性为元素指定一个唯一的名字。

cols属性告诉浏览器文本区宽度为多少个字符。

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

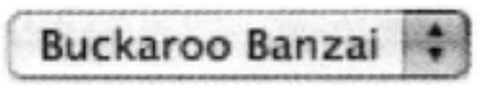
rows属性告诉浏览器文本区高度为多少个字符。

开始和结束标记之间的所有文本会成为浏览器文本区控件中的初始文本。

还可以使用CSS指定文本区的宽度和高度。

select

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



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

<select>元素包围所有菜单选项，把它们组合为一个菜单。

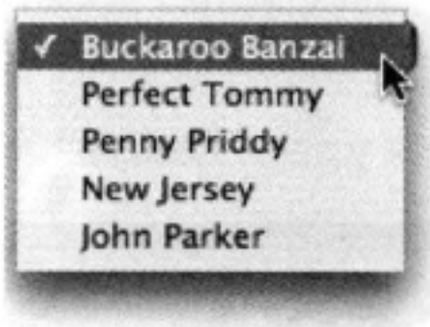
就像其他表单元素一样，要用name属性为select元素指定一个唯一的名字。

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

option

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

单击菜单后，会下拉列出菜单项。

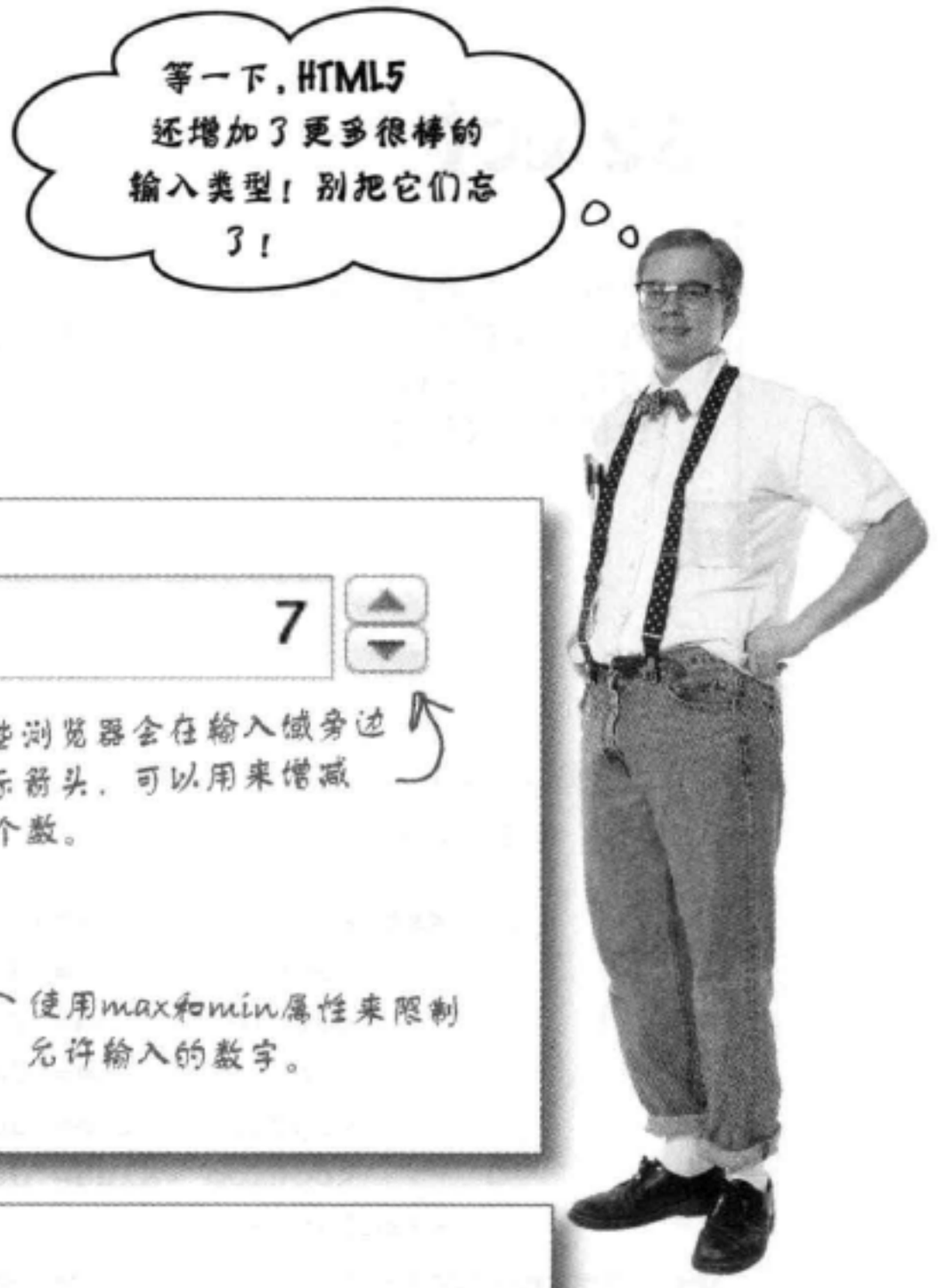



```
<select name="characters">
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny">Penny Priddy</option>
    <option value="Jersey">New Jersey</option>
    <option value="John">John Parker</option>
</select>
```

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

哇，还有更多元素可以放在表单里！

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



数字输入

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

7

有些浏览器会在输入域旁边显示箭头，可以用来增减这个数。

type为“number”表示你只希望输入数字，而不是文本。

使用max和min属性来限制允许输入的数字。

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

范围输入

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

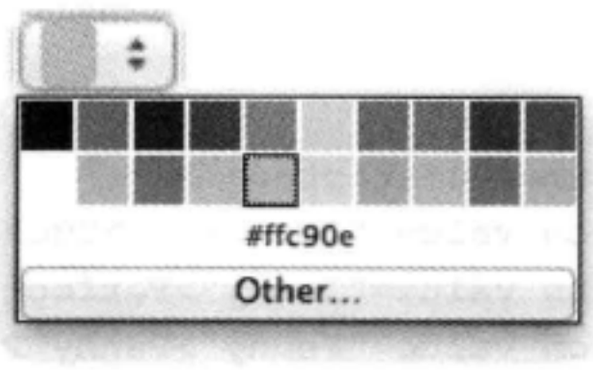
number和range都有一个可选的step属性，可以用来指定值的间隔数（步长）。

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

颜色输入

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

如果浏览器不支持颜色输入元素，你会得到一个常规的文本输入控件。




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

日期输入

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


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

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

email输入

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

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

Email: Buckaroo Banzai

tel输入

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

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

Phone: 555-1212

url输入

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

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

URL: http://banzai.com

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

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

Watch it!

并不是所有浏览器都完全支持这些输入类型。

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

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

这是一个咖啡下拉菜单。

选择全豆咖啡还是研磨咖啡（只能选择其一）。

多少包，以及何时到货。

礼品包装，另外是否包括一个商品目录（可以不选择，也可以选择一个或都选）。

送货地址，包括6个文本框。

表单应该是这样的。

Starbuzz Coffee

Choose your beans:

House Blend
Shade Grown Bolivia Supremo
Organic Guatemala
Kenya

Type:

- ○ Whole bean
- ◉ Ground

Number of bags:

Must arrive by date:

Extras:

- ☐ Gift wrap
- ☑ Include catalog with order

Ship to:

Name:

Address:

City:

State:

Zip:

Phone:

Customer Comments:

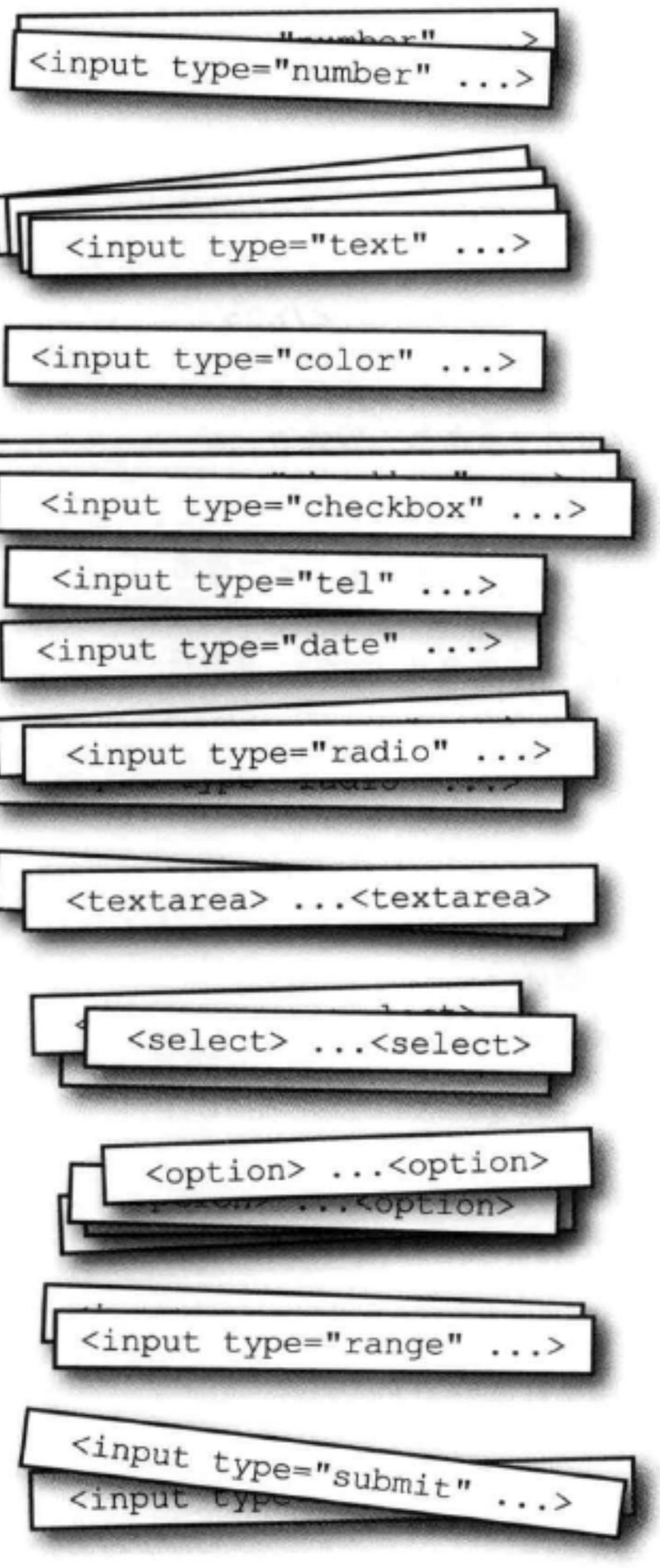
Order Now

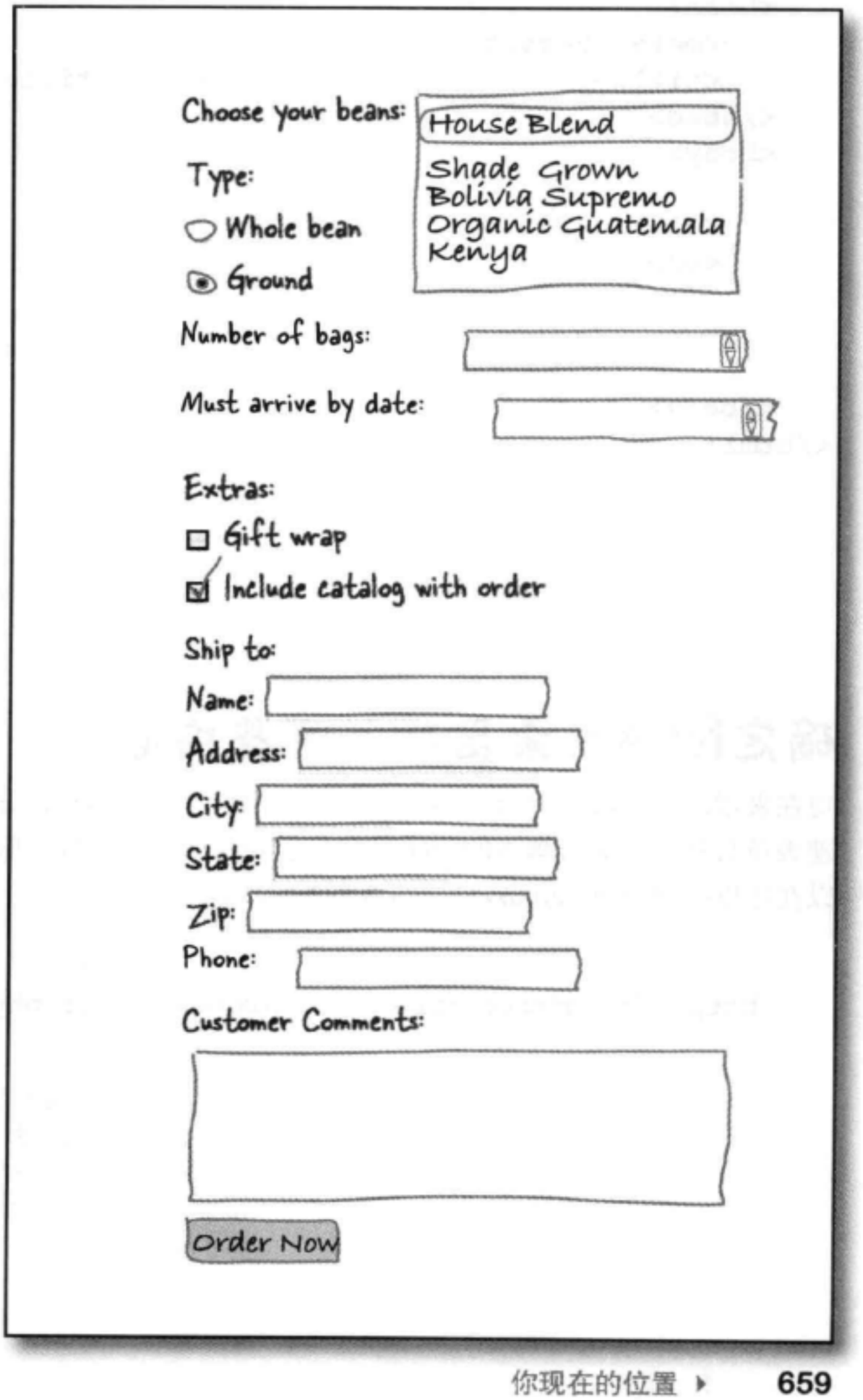
填写客户意见的文本框。

这是一个提交按钮。

标记磁贴

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





准备建立Bean Machine表单

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

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

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

      <!-- 表单要放在这里。 -->

   </body>
</html>
```

目前为止，我们只有一个标题来标识这个页面，另外还有一些及相关的说明。

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

确定form元素里加入哪些内容

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

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

这个URL指向Starbuzz Coffee网站……

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

增加<form>元素

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

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

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

这里是form元素。

action属性包含服务器脚本的URL。

记住我们要用POST方法向服务器提交表单数据。有关的更多内容将在后面介绍。

还要增加form结束标记。

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

这里的类型是“text”，因为它将作为一个文本输入控件。

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

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

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

这里类型是“tel”，因为我们希望这里的值是一个电话号码。

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

表单元素名如何工作

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

表单中的每个输入控件都有一个name属性：

在HTML文件中输入表单元素时，会为它们指定唯一的名字。

前面的text和tel输入元素就指定了不同的名字：

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

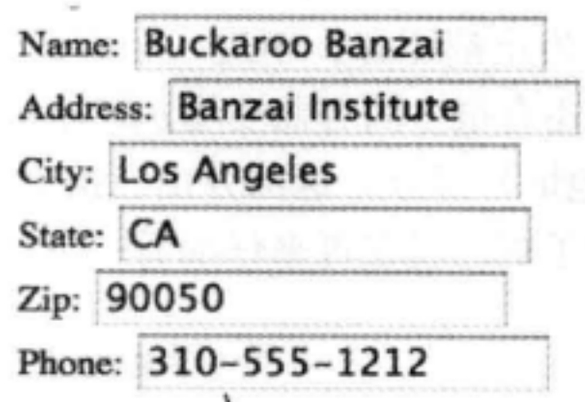
注意，这里有一个元素的名字是"name"（这是完全可以的）。

每个<input>元素都有自己的名字。

提交表单时，浏览器会使用这些唯一的名字打包所有数据：

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

表单中输入的值。

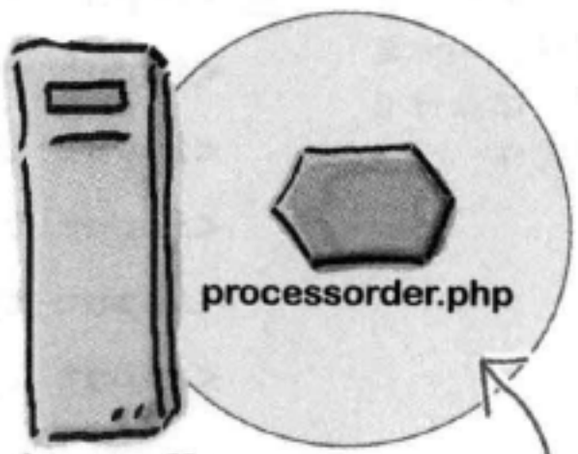


各个表单元素的唯一名字。

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

对于每个唯一的元素名，从你在表单中输入的数据中得到一个相应的值。

浏览器打包这些数据，发送给服务器。



服务器脚本要求这些表单数据有相应的标签，这样它才能分清谁是谁。

there are no Dumb Questions

问：text <input>与<textarea>有什么区别？

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

问：提交按钮上能不能不显示“Submit”，可以显示别的名字吗？

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

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

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

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

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

问：我还是不清楚这些名字怎么与表单数据对应起来。

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

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

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

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

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

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

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

问：复选框呢？它们的工作与单选钮类似吗？

答：对，唯一的区别是，复选框允许你选择多个选项。

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

问：天呐，数据发送到服务器这么复杂，我必须了解所有这些细节吗？

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

将这些<input>元素放在HTML中

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

这只是“form.html”中的表单片段。嘿，节省些篇幅可以少砍些树！

把这个元素直接嵌套在表单中。

先把所有内容放在一个<p>元素中。

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

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

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

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

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

完成这些修改之后，保存你的“form.html”文件，下面来试一试。

别忘了验证你的HTML。表单元素也需要验证！

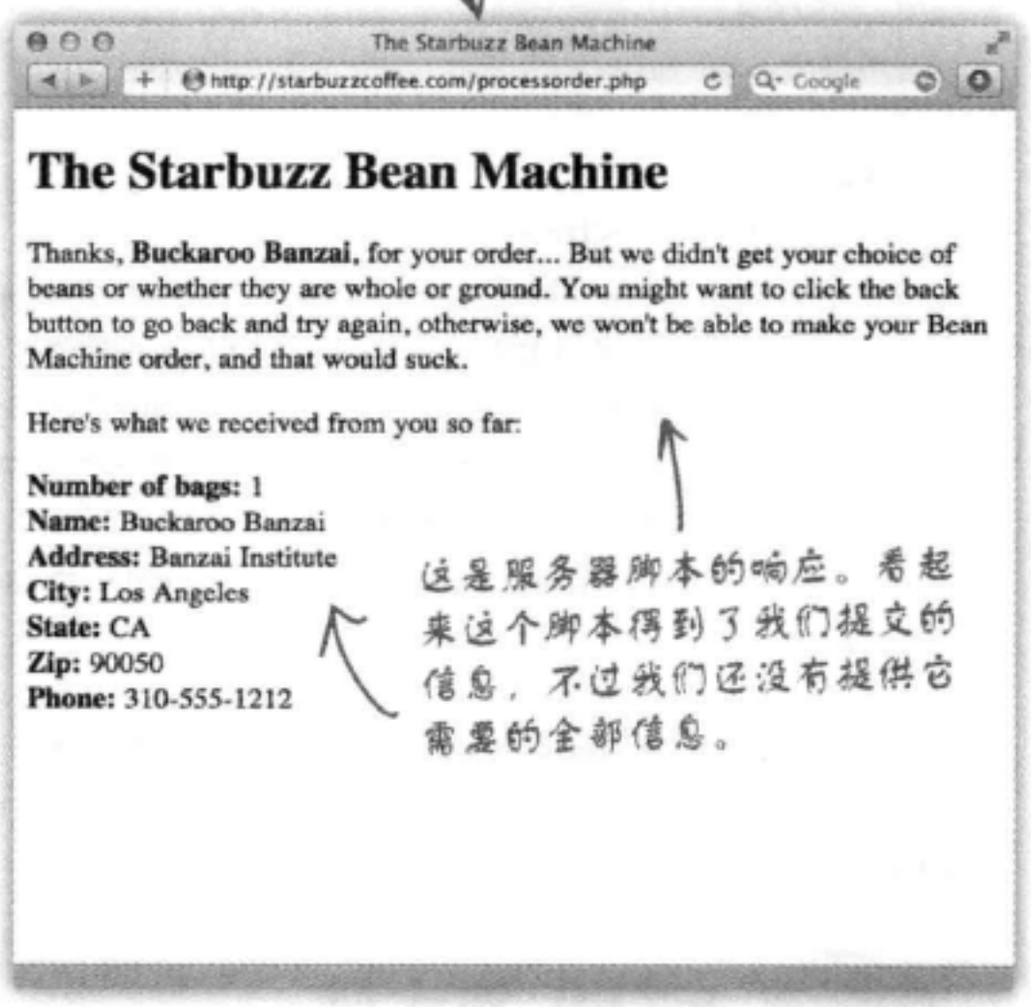
正式的表单测试

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

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

这是我们得到的表单。

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



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

这是提交表单后得到的响应。

为表单增加更多输入元素

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

增加<select>元素

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

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

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

这是全新的<select>元素。它也有一个唯一的名字。

在这个元素中放入各个<option>元素，分别对应一种咖啡选择。

HTML放大镜

下面来仔细查看<option>元素。

每个option有一个值。

这个元素的内容将作为下拉菜单中的标签。

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

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

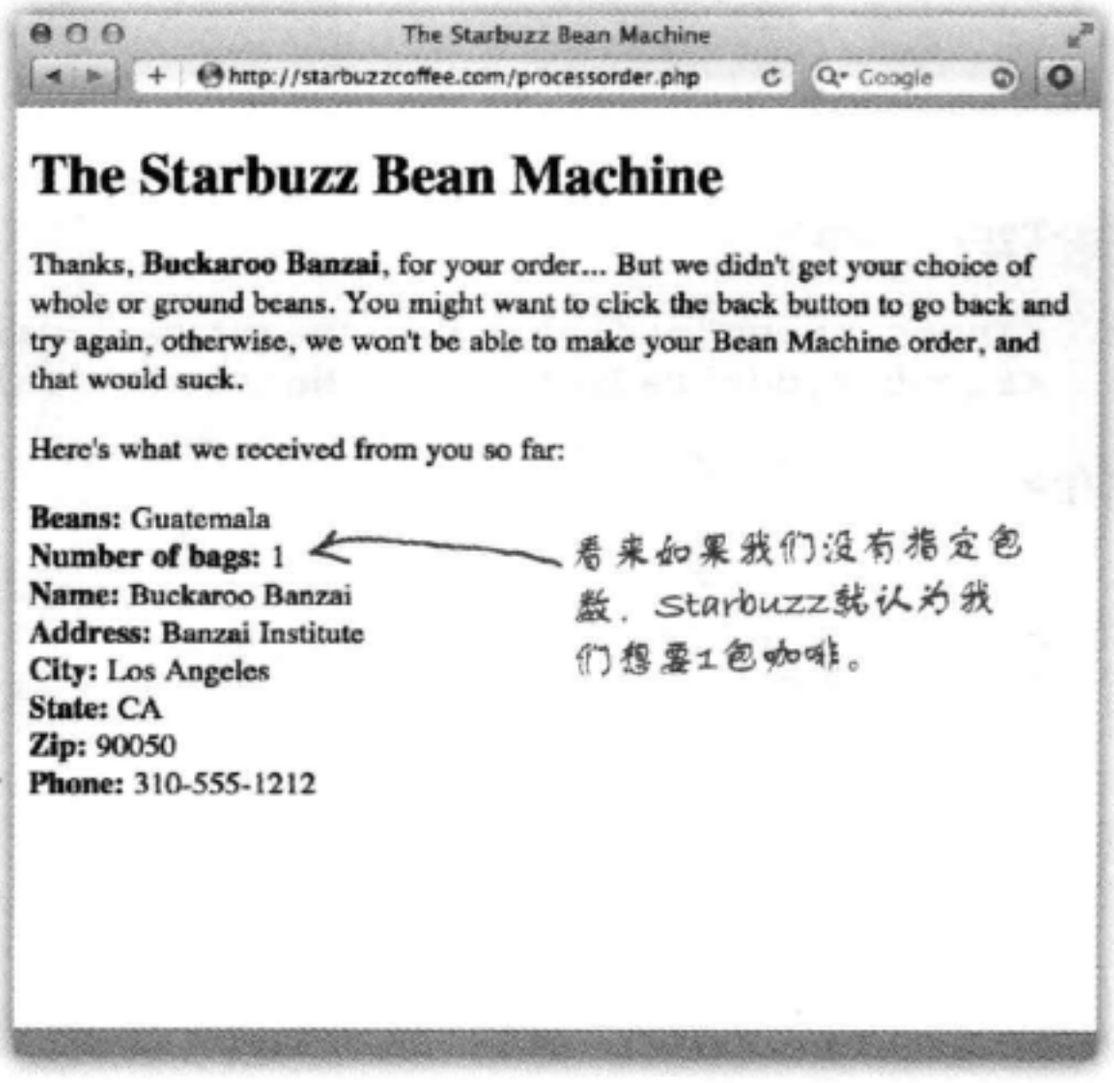
在这个例子中，浏览器会向服务器发送beans = "Guatemala"。

测试<select>元素

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

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

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



这是<select>选择的结果。

看来如果我们没有指定包数，Starbuzz就认为我们想要1包咖啡。

这里是所有text输入控件和tel输入控件的结果。

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

完成这个练习之后，一定要把name属性再改回为“beans”。

允许客户选择全豆咖啡还是研磨咖啡

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

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

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

这里要使用<input>元素，type设置为“radio”。

这是唯一的名字。同组的所有单选钮都有相同的名字。

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

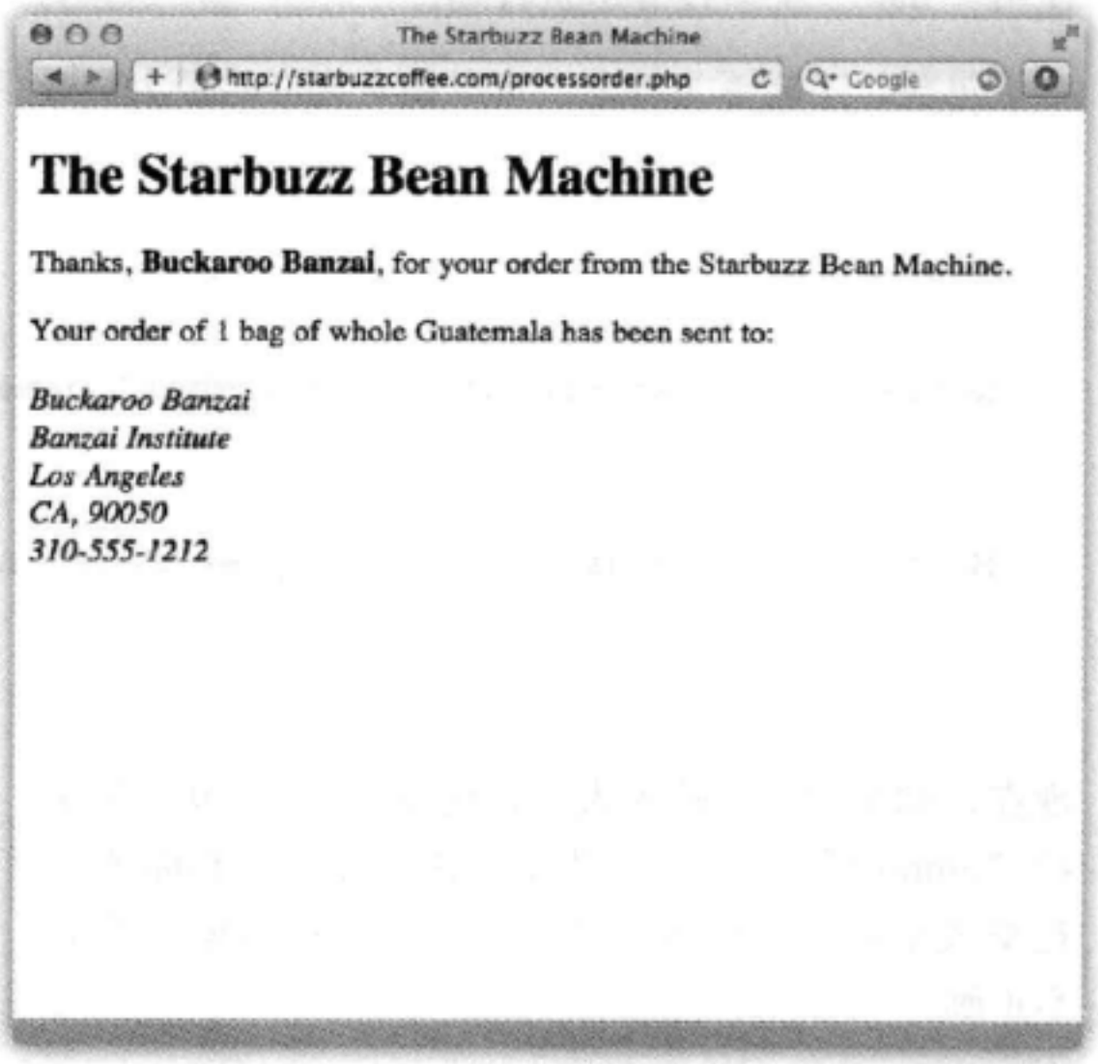
注意我们通常把单选钮的标签放在元素右边。

试试单选按钮

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

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

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



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

Exercise

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

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

记住，布尔属性不需要值。如果有属性checked，这个输入控件就会选中。

使用更多输入类型

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

对于包数，还能更为特定，可以指定允许的最大和最小包数：

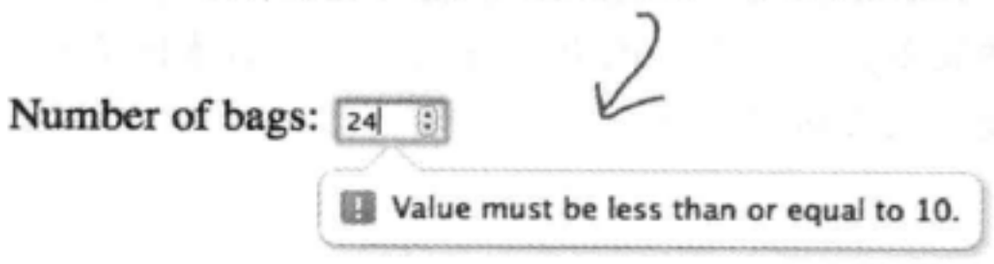
利用“number”类型，并指定最小和最大包数，可以限制输入是一个有效的值（我们不希望客户一次购买10包以上同类咖啡）！

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

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

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

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

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



增加数字和日期输入类型

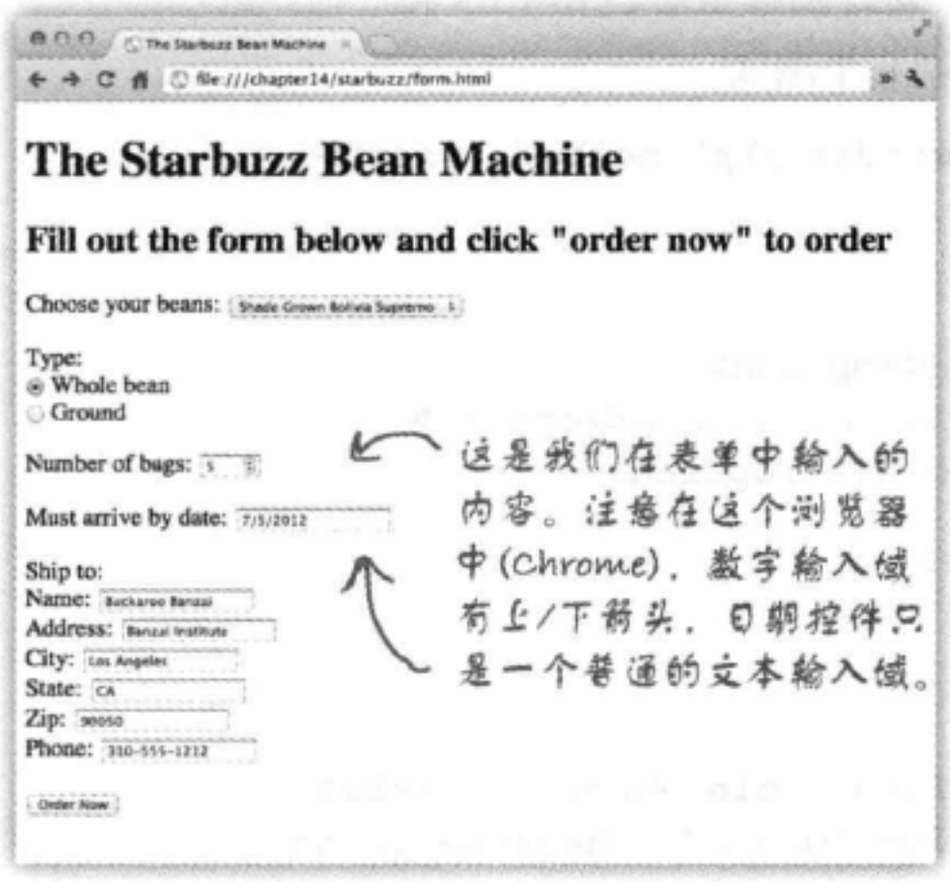
把这两个新的<input>元素增加到你的“form.html”文件，放在咖啡类型<input>下面，而且要在Ship To（送货地址）部分上面，对这些新代码做一个测试。

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

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

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

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

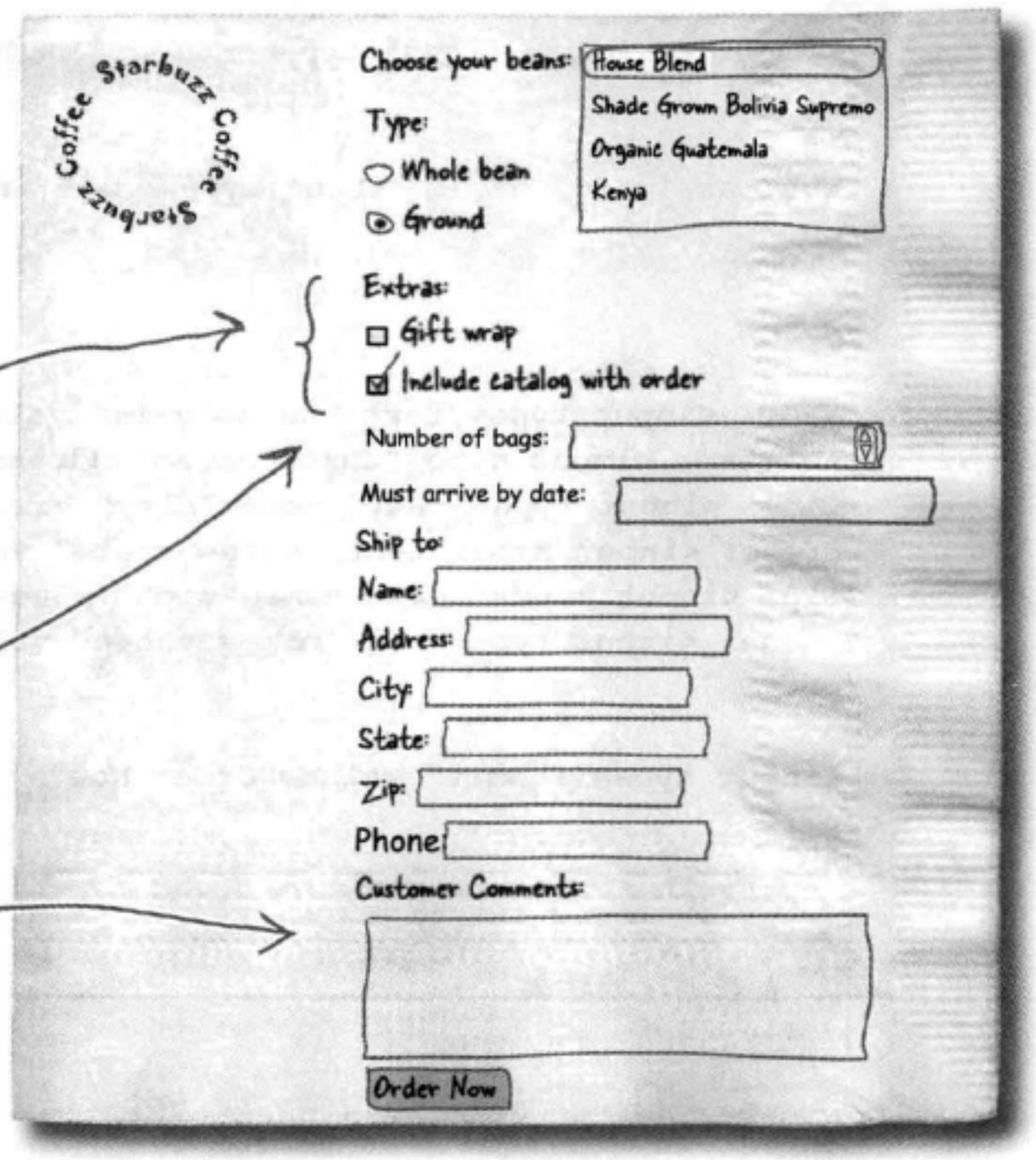
翻开下一页，看看测试的结果……

测试number和date<input>元素



完成表单

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



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

看起来“Include catalog”（包含商品目录）选项要默认选中。

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

增加复选框和文本区

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

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

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

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

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

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

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

……不过有不同的值。

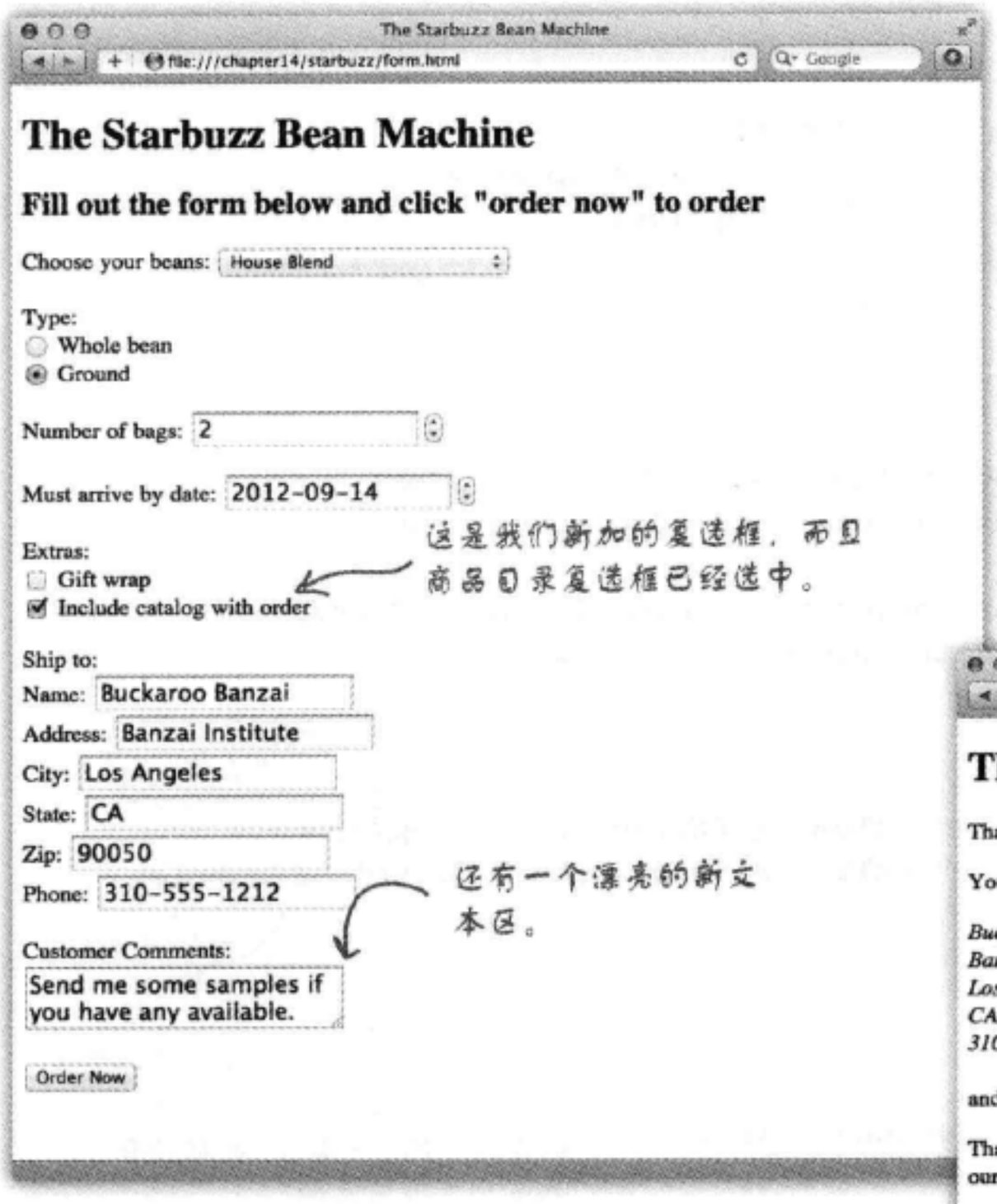
我们使用了checked属性来指定商品目录选项默认是选中的。可以为多个复选框增加checked属性。

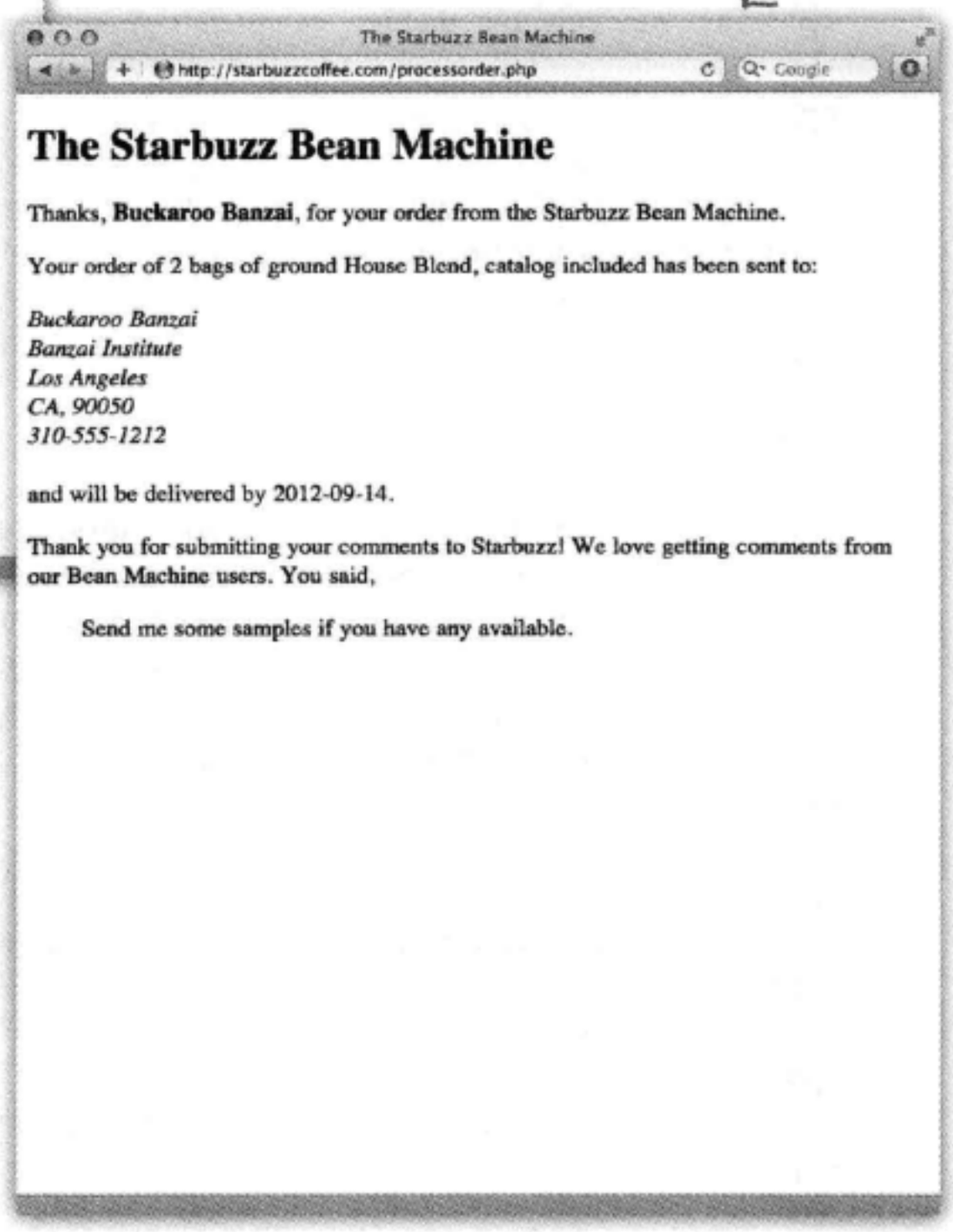
与单选钮一样，我们把这些标签放在复选框的右边。

这里是一个文本区。

最后的表单测试

保存你的修改，重新加载页面，查看这个新表单。 是不是感觉好多了？





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

不管你相不相信，“extras[]”确实是一个完全合法的表单元素名。

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

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

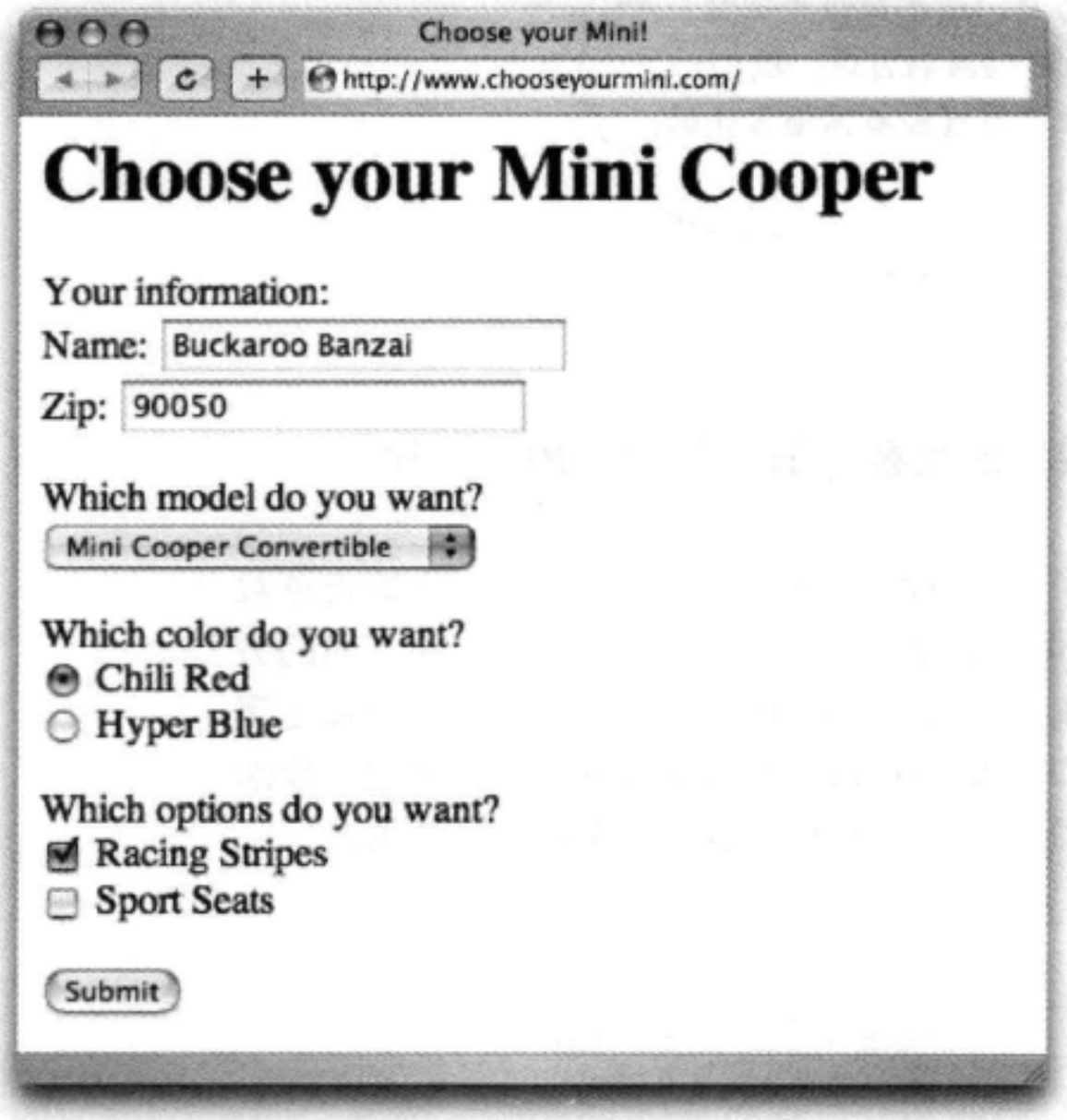
所以，从学习HTML来讲，你完全可以先不考虑它，不过最好还是能记得，万一将来你要写一个使用PHP服务器脚本的表单呢。

扮演浏览器

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

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

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

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

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

</form>
```

这里是表单。



这里是已经填写的表单。

将各部分表单数据与相应的表单名匹配，在这里写出你的答案。

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

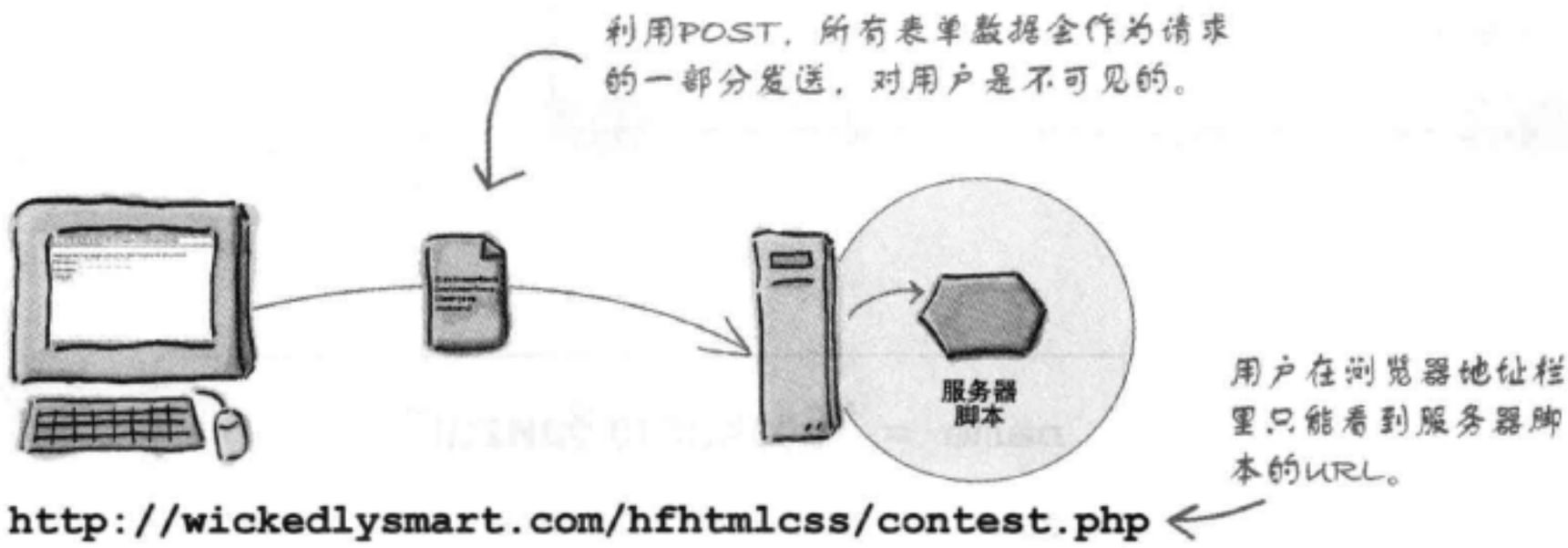
加分题……

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

浏览器使用的主要方法有两种：POST和GET。

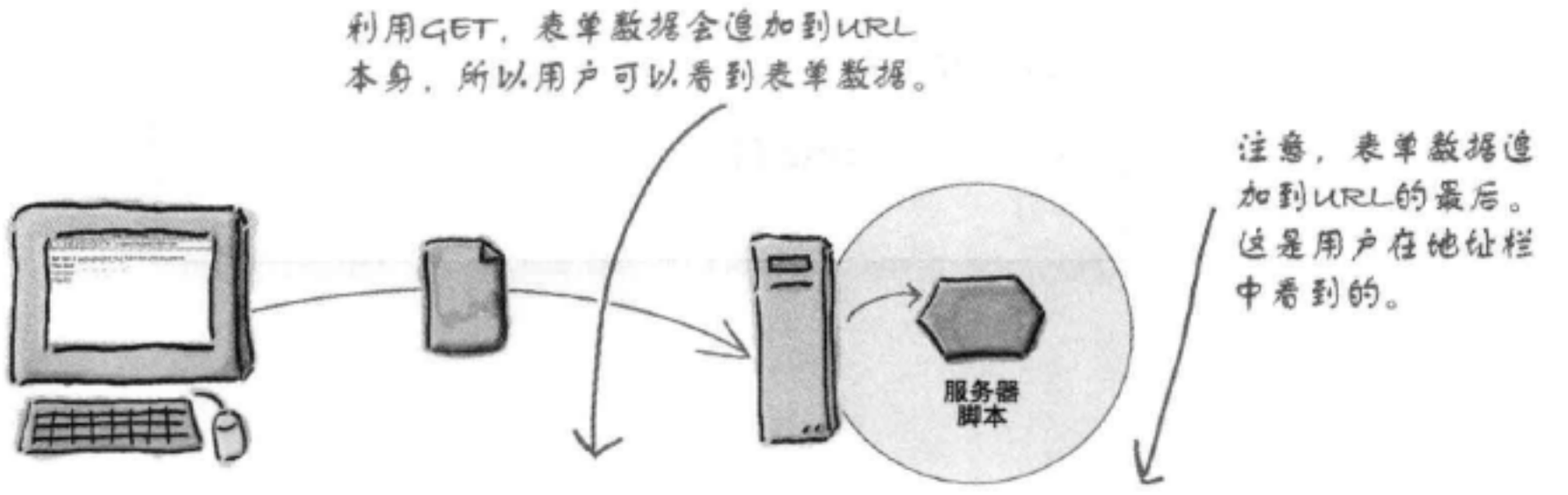
POST和GET完成的任务是一样的，都是将你的表单数据从浏览器发送到服务器，不过采用了两种不同的方式。POST会打包你的表单变量，在后台把它们发送到服务器；GET也会打包你的表单变量，但会把这些数据追加到URL的最后，然后向服务器发送一个请求。

POST



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

GET



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

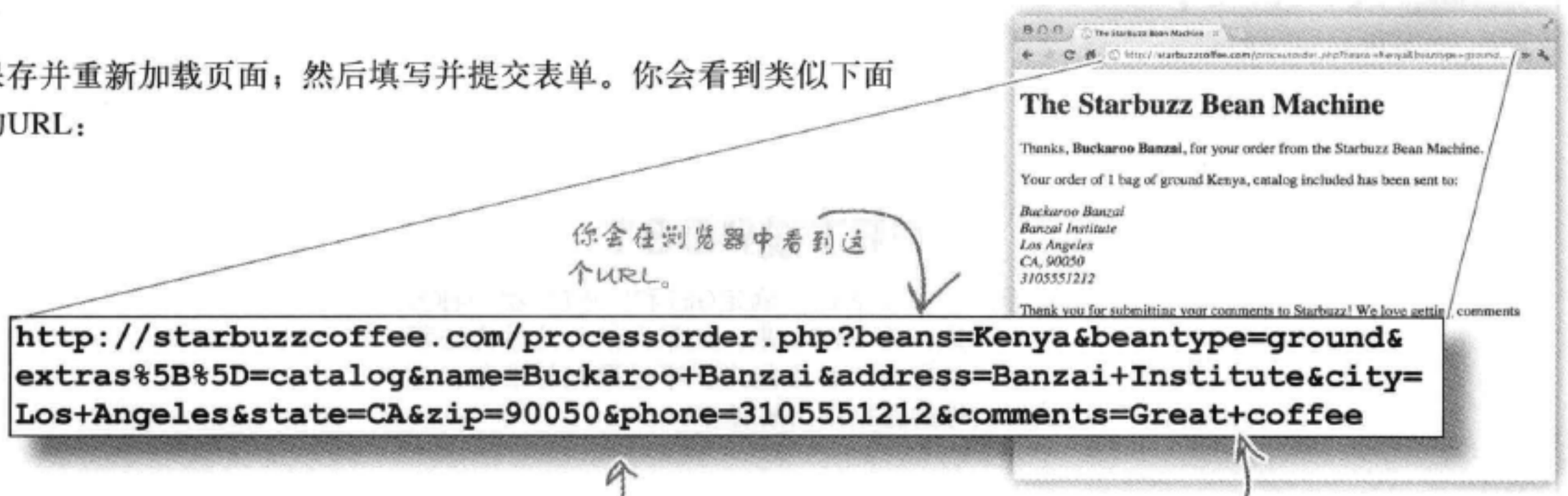
GET的实际使用

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

只需要把方法从“POST”改为“GET”。

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

保存并重新加载页面；然后填写并提交表单。你会看到类似下面的URL：



你会在浏览器中看到这个URL。

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

在这个URL中可以看到每个表单元素名和相应的值。

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

there are no Dumb Questions

问：既然我们在向服务器发送数据，为什么这种方法叫GET？

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

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

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

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

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

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

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

Sharpen your pencil

GET或POST

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

GET **POST** 输入用户名和口令的表单

GET **POST** 订购CD的表单。

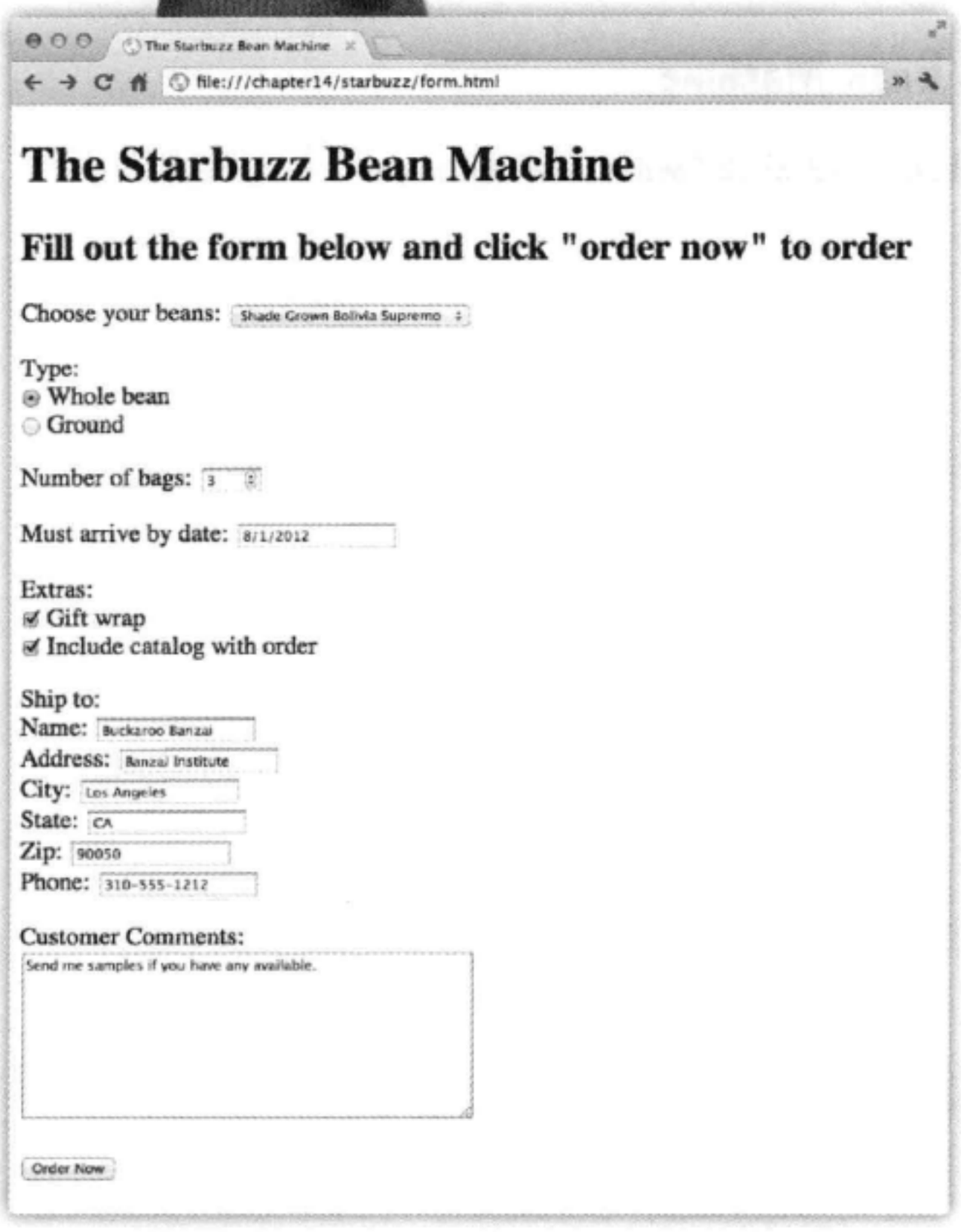
GET **POST** 查看当前时事的表单。

GET **POST** 提交书评的表单。

GET **POST** 按身份证号查看福利的表单。

GET **POST** 发送客户反馈的表单。

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



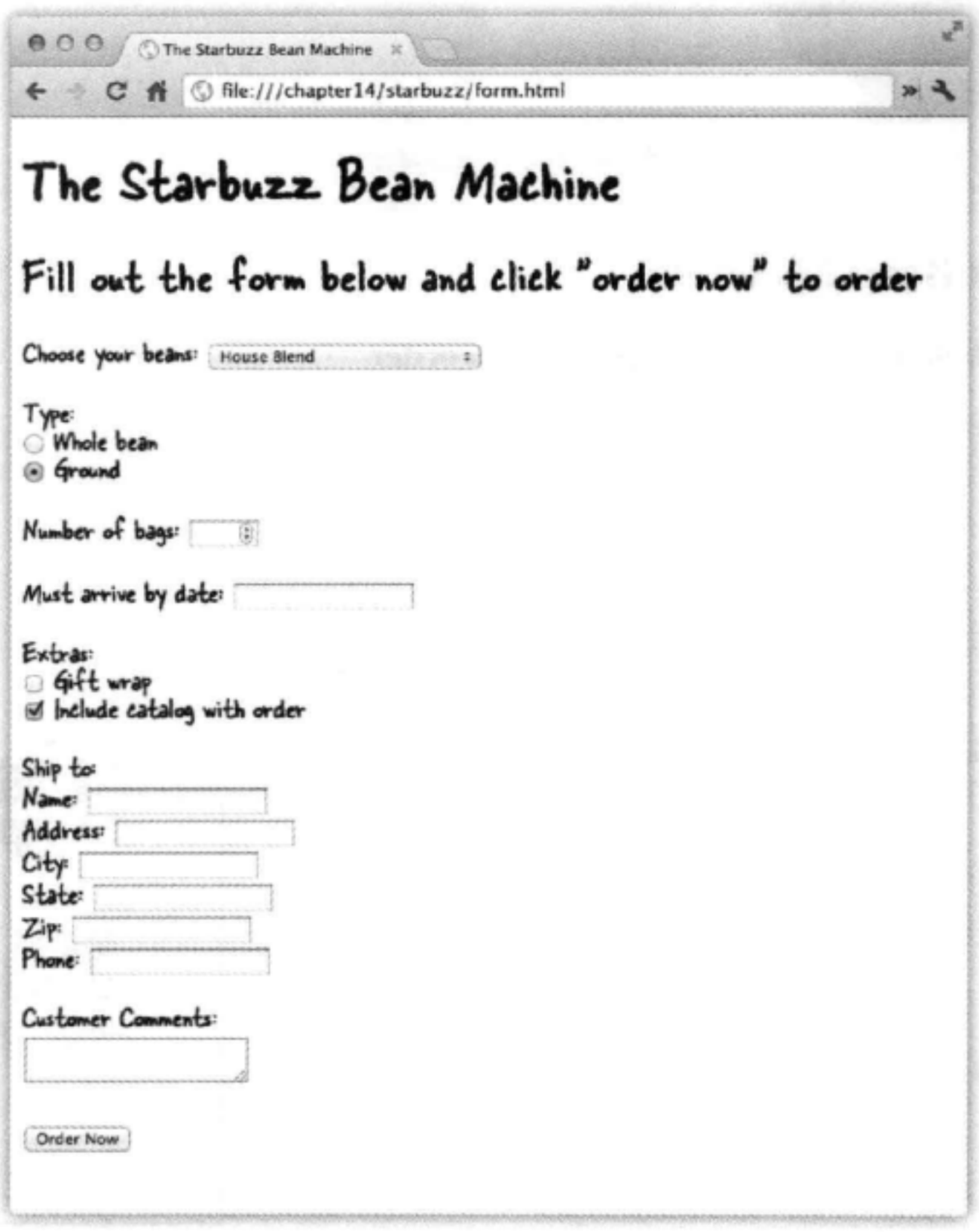
根据你了解的HTML和CSS的全部知识，如何为这个表单增加样式？

Sharpen your pencil

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

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

完成这个练习之前，别偷看下一页。真的！可以把下一页先盖住。

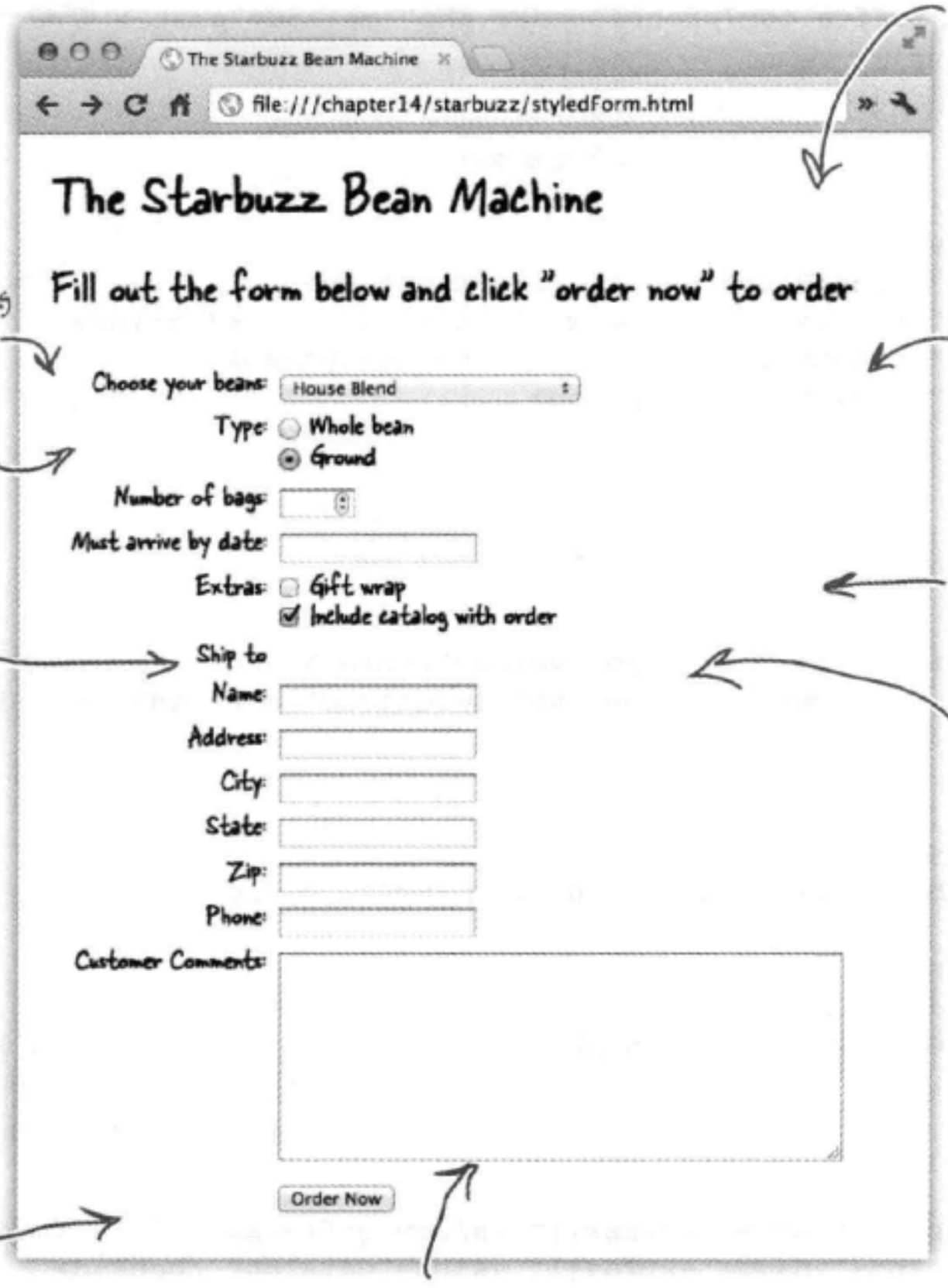


Sharpen your pencil Solution

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

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

这是我们的答案……在学习后面的内容之前，请对照检查你的答案！



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

每个表单元素的标签放在左列。

所有输入元素都放在右列中。

单元格值在垂直方向上都向上对齐。

注意我们把各组复选框和单选钮都归组到一个单元格中。

"Ship to"右边的单元格为空，这里没有控件。

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

各行还需要另外一些块元素。就像前面一样（第11章），我们将使用 元素。

最后，需要一个元素包含所有内容，作为表单本身。为此可以使用form元素！

提交按钮左边的单元格为空。这里没有放置标签。

文本区也更大了！

将表单元素放入HTML结构实现表格显示布局

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

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

这里是<form>元素，我们将使用这个元素实现“表格”显示部分。

我们将使用一个class为“tableRow”的 表示表格中的各行。

每个单元格中的内容嵌套在一个<p>元素中。

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

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

代码未完，见下一页。

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

注意，右列中还有一个空单元格，所以在这里直接放一个空的<p>元素。

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

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

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

用CSS建立表单样式

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

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

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

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

这里使用form利用表格显示布局来表示表格……

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

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

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

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

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

每个“tableRow” 相当于表格显示布局中的一行。

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

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

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

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

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

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

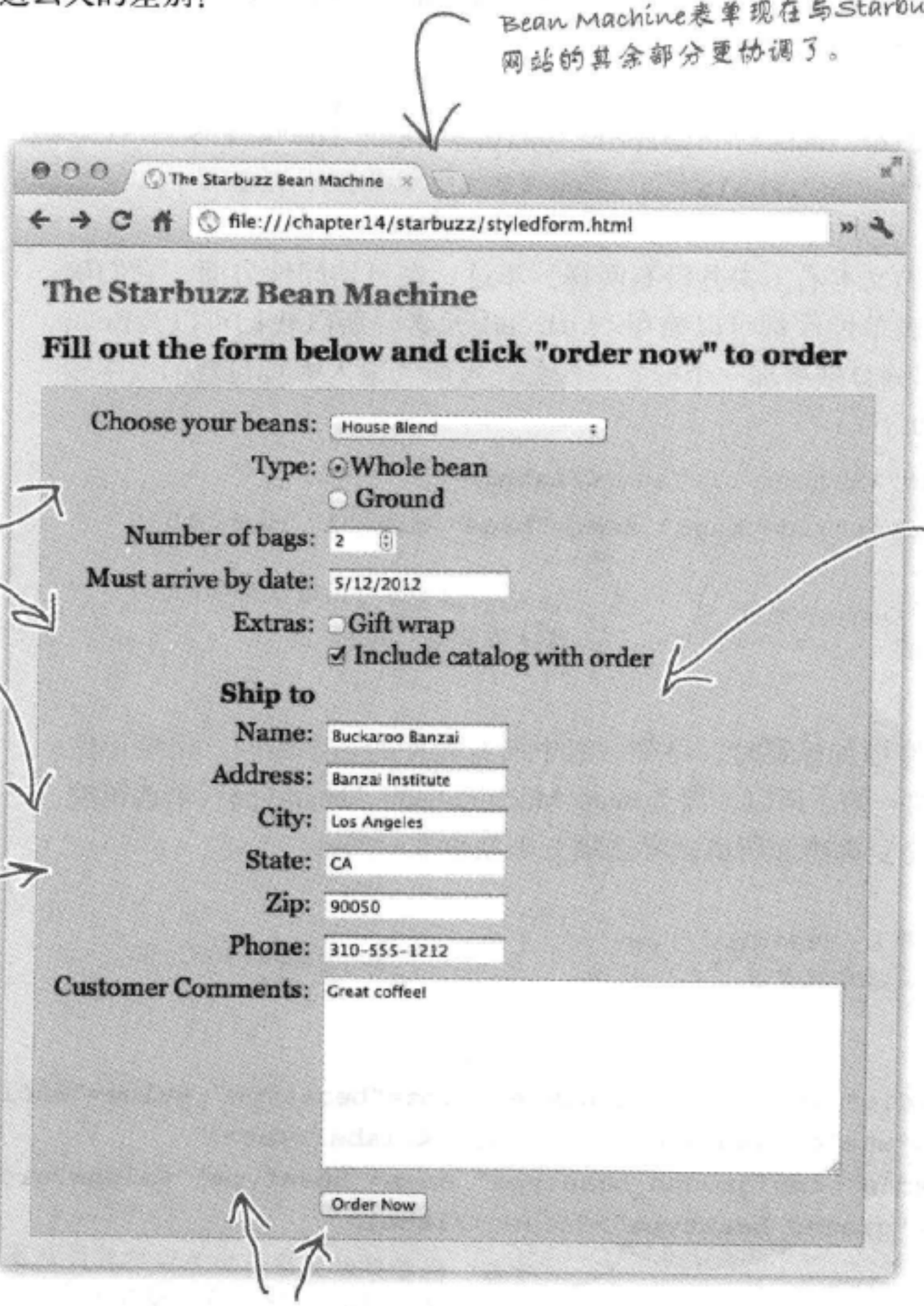
测试增加样式后的表单

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

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

哇，一点点样式竟然带来这么大的差别！

Bean Machine表单现在与Starbuzz网站的其余部分更协调了。



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

正如我们期望的，“Ship to”标题是粗体。

行之间的间距让表单大为改观，另外也更容易阅读。

这里有两列，行中的所有内容都能很好地对齐！

关于可访问性

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

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

要使用<label>元素，首先为表单元素增加一个id属性。

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

然后增加一个<label>，设置其"for"属性为相应的id。

现在这些单选钮旁边的文本就是一个真正的标签（label）。

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

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

我们已经为这个<input>元素增加了id "bags"。

name和id属性可以使用相同的值，在这里就是"bags"。

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

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

不过，每个id必须是唯一的。

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

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

表单中还可以有哪些元素？

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

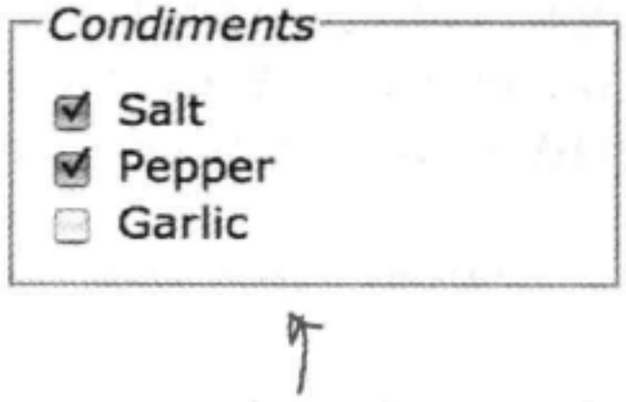
fieldset和legend

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

<fieldset>元素包围一组输入（input）元素。

<legend>为这一组提供一个标签。

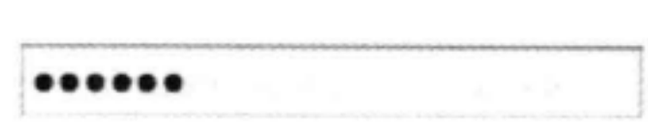
```
<fieldset>
   <legend>Condiments</legend>
      <input type="checkbox" name="spice" value="salt">
         Salt <br>
      <input type="checkbox" name="spice" value="pepper">
         Pepper <br>
      <input type="checkbox" name="spice" value="garlic">
         Garlic
</fieldset>
```



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

passwords

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



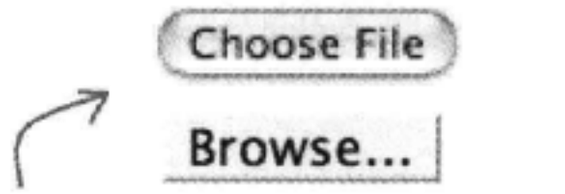
```
<input type="password" name="secret">
```

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

可以放在表单中的其他元素

文件输入

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



这是两个不同浏览器中看到的文件输入元素。

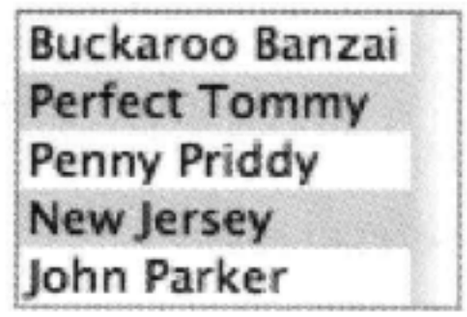
```
<input type="file" name="doc">
```

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

多选菜单

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

利用多选菜单，你可以一次选择多个选项。

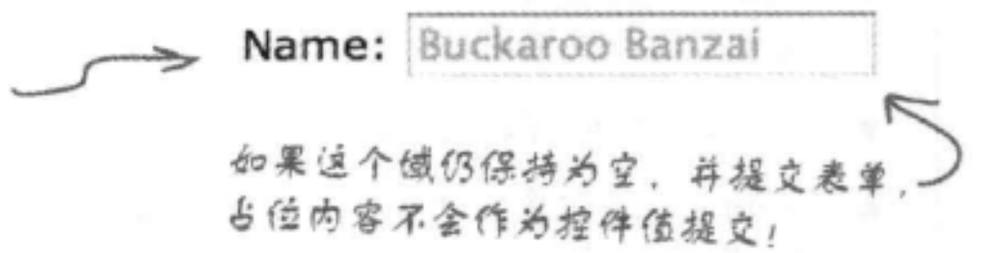



```
<select name="characters" multiple>
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny Priddy">Penny</option>
    <option value="New Jersey">Jersey</option>
    <option value="John Parker">John</option>
</select>
```

只需要增加属性multiple，就把一个单选菜单变成了多选菜单。

Placeholder

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



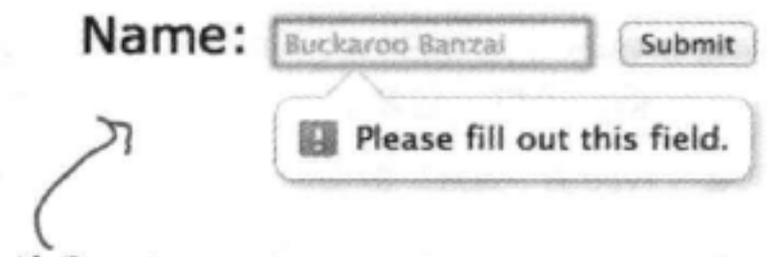

```
<input type="text" placeholder="Buckaroo Banzai">
```

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

Required

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

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



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

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

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

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

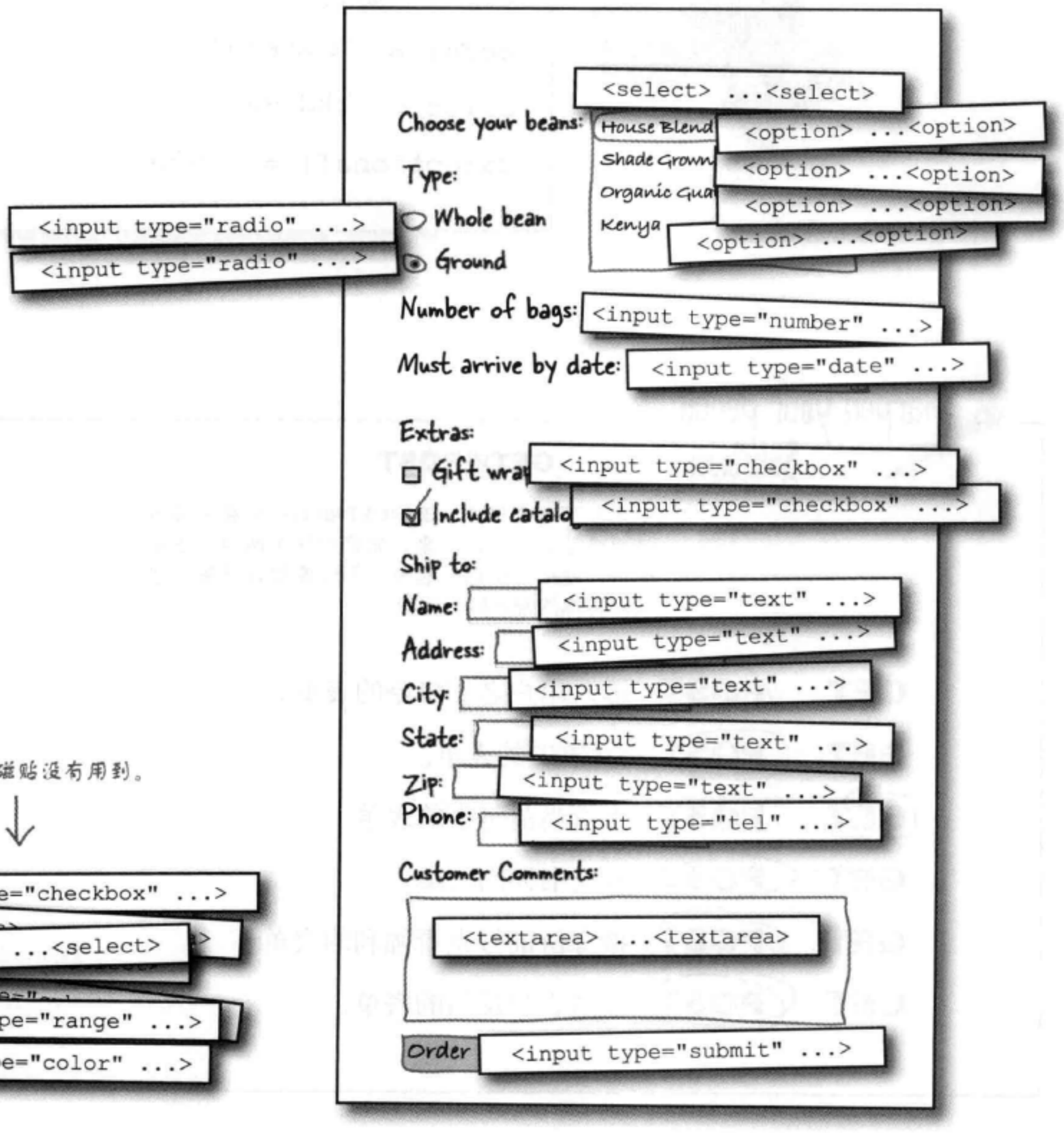
接下来，继续编辑这个文件，为Starbuzz Bean Machine页面中所有必要的表单域（所有“Ship to”域）增加required属性。既然beans和beantype都有默认值，这些域还需要required属性吗？如果从beantype删除checked属性会发生什么情况，还需要required吗？在不同的浏览器上做些实验，看看哪些浏览器支持placeholder和required。

BULLET POINTS

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

标记磁贴答案

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



扮演浏览器答案

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

Sharpen your pencil Solution

GET或POST

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

GET **POST** 输入用户名和口令的表单。

GET **POST** 订购CD的表单。

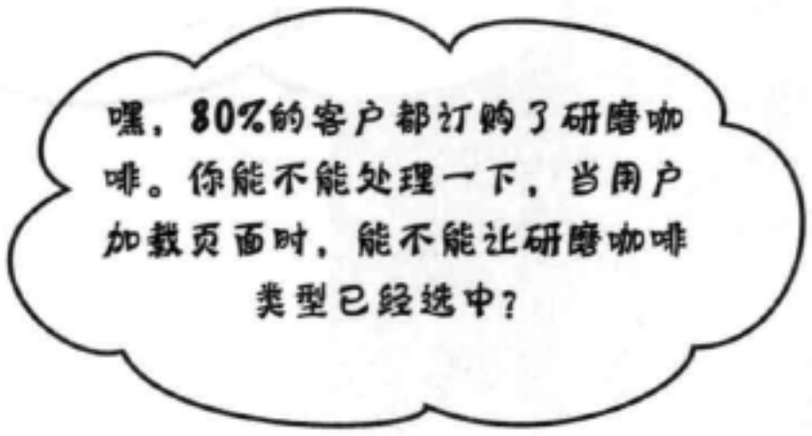
GET POST 查看当前时事的表单。

GET **POST** 提交书评的表单。

GET **POST** 按身份证号查看福利的表单。

GET **POST** 发送客户反馈的表单。

Exercise Solution

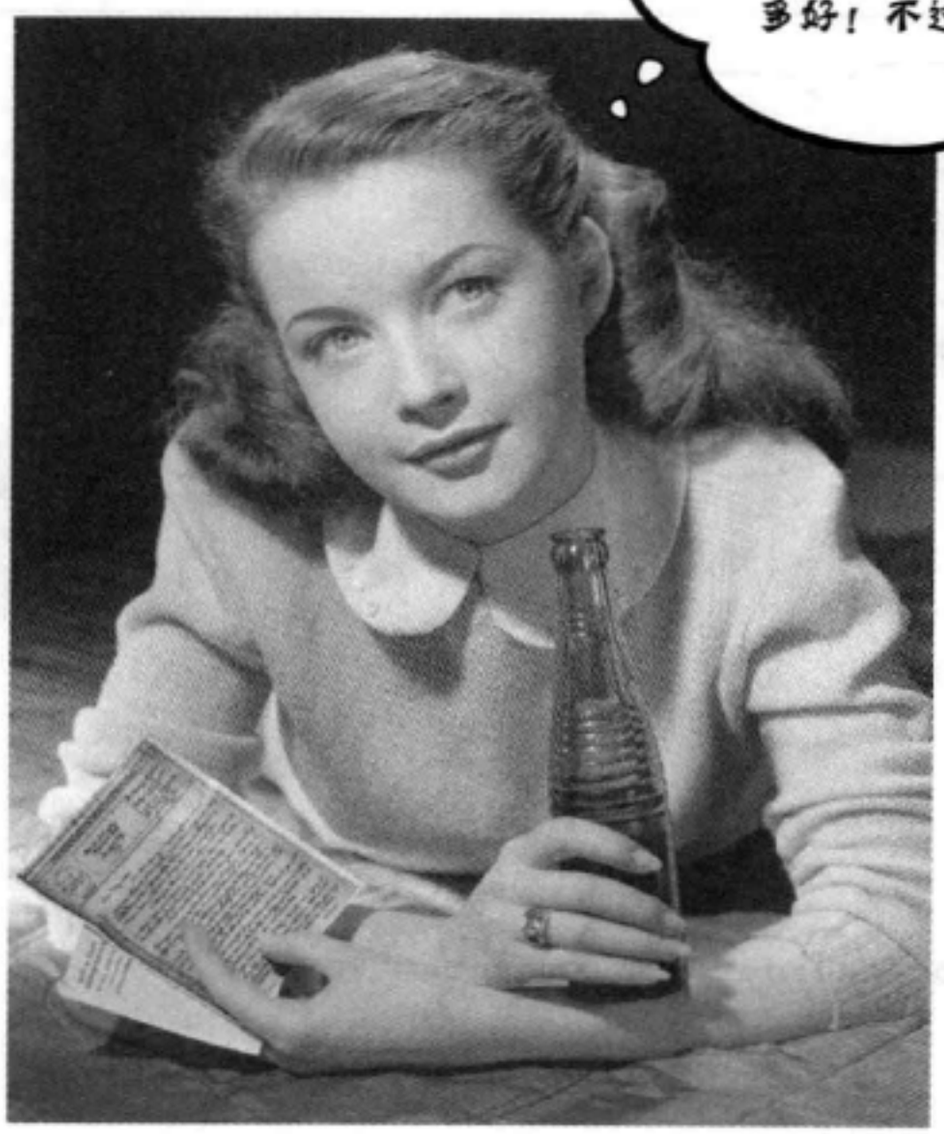


如果为单选钮输入元素增加一个布尔属性“checked”，浏览器显示表单时就会默认地选中这个元素。为“ground” radio <input>元素增加checked属性，再来测试这个页面。下面给出答案。

这只是“form.html”表单中相关的部分。

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

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

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

这就是选中的“ground”单选钮的新属性。



祝贺你！

终于到终点了。

当然了，后面还有一个附录。

还有索引。

还有封底。

然后还有网站……

实际上，这是无止境的。

十大主题(我们没有谈到的)

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

#1 更多CSS选择器

你已经了解了那些最常用的选择器，下面再来介绍一些你可能也想知道的选择器……

伪元素

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

伪元素的语法与伪类相同。

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

这里将段落的第一个字母放大，另外把第一行设置为斜体。

属性选择器

顾名思义，属性选择器就是根据属性值来选择元素。可以像这样使用：

这个选择器会选择HTML中所有包含一个width属性的图像。

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

这个选择器会选择height属性值为300的所有图像。

这个选择器会选择alt属性包含单词“flowers”的所有图像。

按兄弟选择

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

先写前面的元素，再写一个“+”（加号），然后是兄弟元素。

这个选择器会选择所有紧跟在一个<h1>元素后面的段落。

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

结合选择器

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

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

这里会选择作为 blueberry 类元素的子孙的所有段落。

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

❶ 首先为你想要选择的元素定义上下文，如下所示：

```
div#greentea > blockquote
```

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

❷ 然后给出你想选择的元素：

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

上下文

空格

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

❸ 然后指定伪类或伪元素：

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

上下文

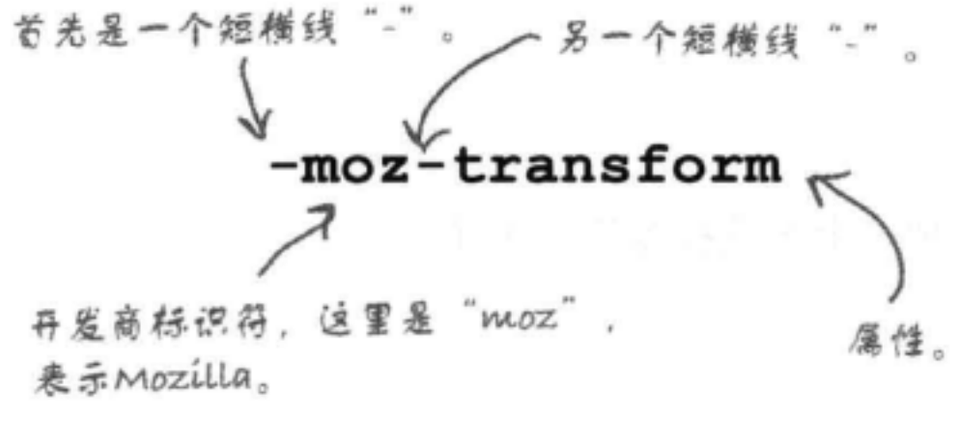
冒号

然后增加一个伪元素first-line，指定只选择这个段落的第一行。

这是一个相当复杂的选择器！你完全可以使用这种方法构造自己的选择器。

#2 开发商特定的CSS属性

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



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

如果打算使用这些属性，通常会创建类似下面的CSS：


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

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

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

#3 CSS变换和过渡

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

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

这是下面的“box” 的基本样式……

这是绝对位置（你是不是很庆幸跟着我们学完了定位那一章）。

下面为这个 指定一个位置和大小……

……而且把它变成红色。

只有当 处于悬停状态时才会应用这个样式规则……没错， 也可以有悬停状态！

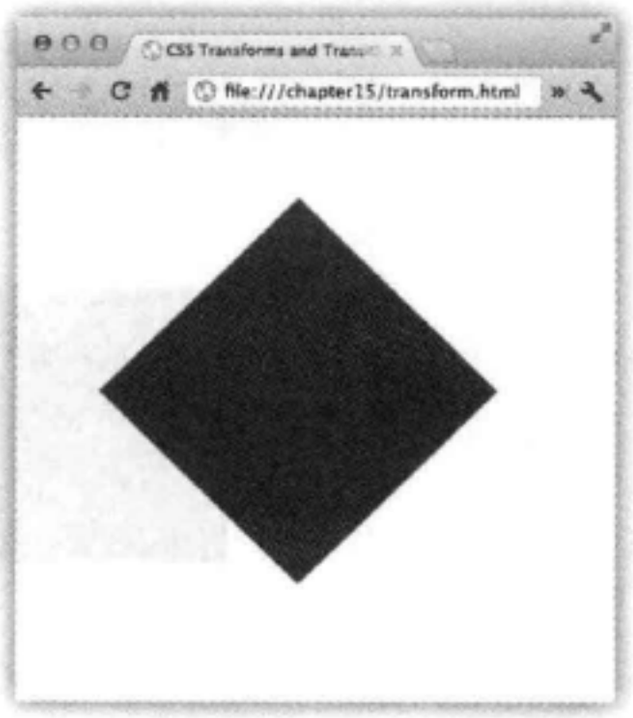
把鼠标停在 上时，会对这个元素完成变换，将它旋转45度。

这个属性只能用于IE9+。

我们还需要浏览器特定的扩展。

这是我们要变换的 。

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



鼠标放在 上，可以看到它旋转了！

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

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

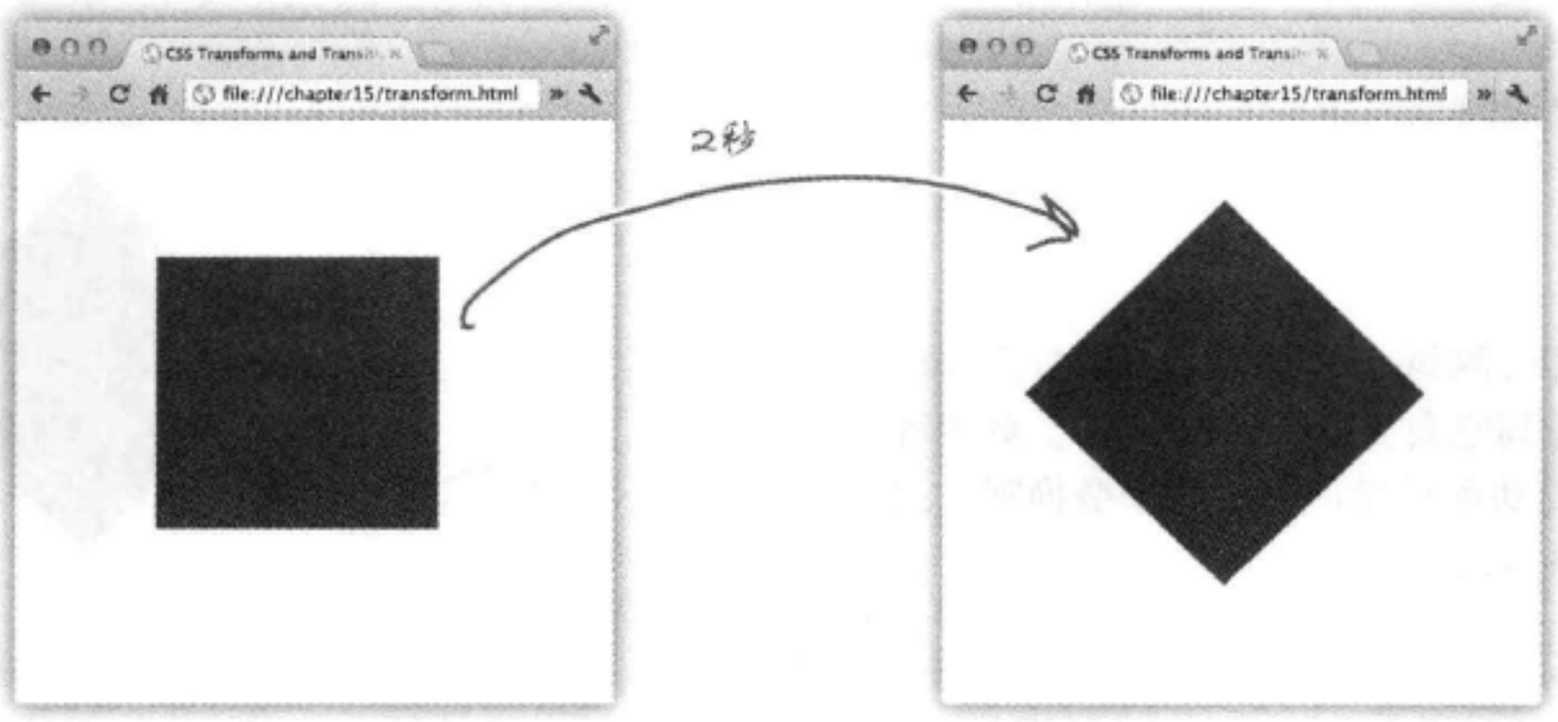
这个transition属性指出：“如果transform属性的值改变，要在指定的时间内从当前transform值过渡到新的transform值。”

transform没有默认值。也就是说，没有变换。

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

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

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



#4 交互性

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

这是一个新的HTML元素<script>，允许你在HTML中放入代码。

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

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

这里的JavaScript会检查用户的bid，确保它不会小于等于0。

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

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

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

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

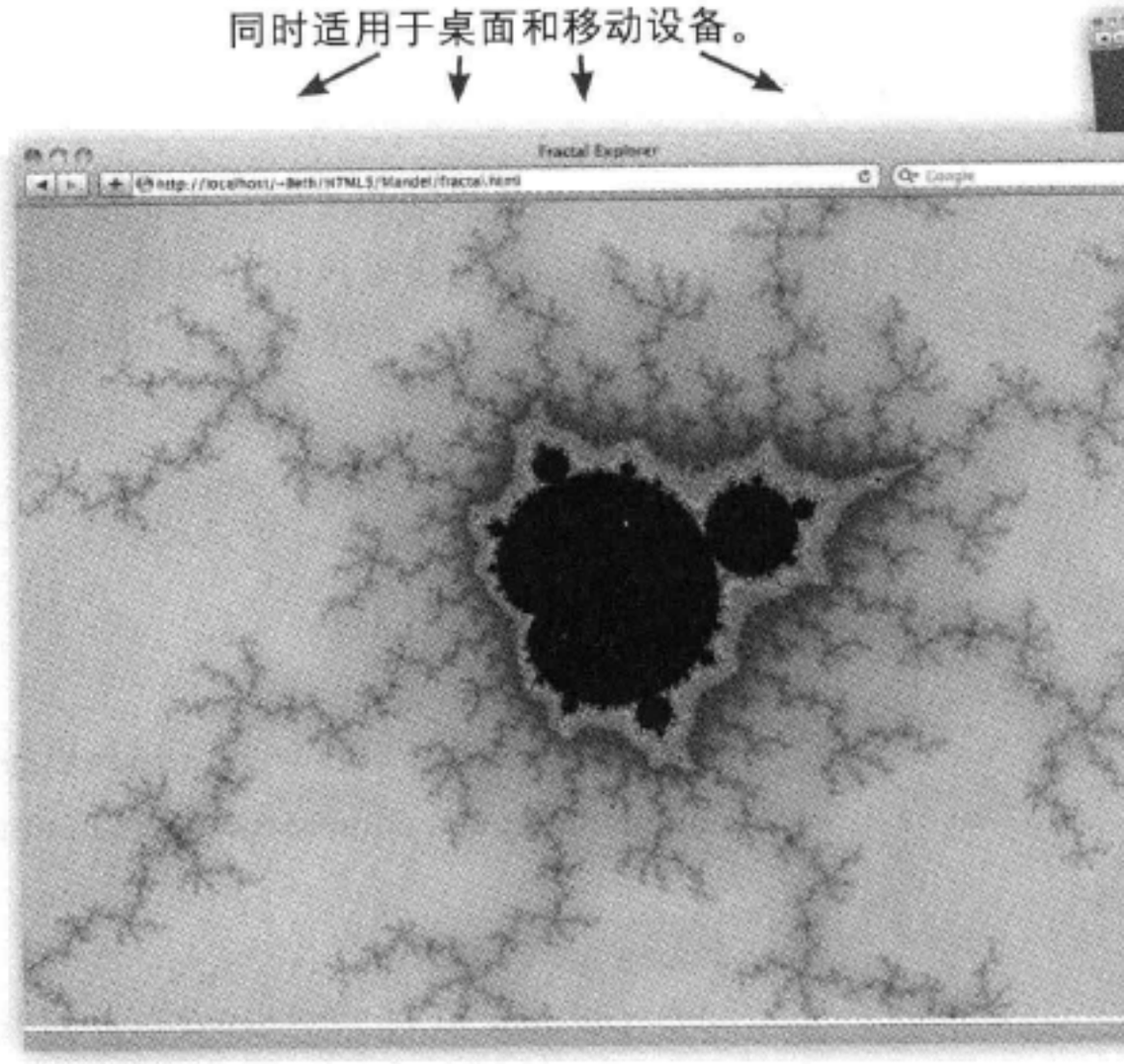
脚本还能做什么？

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

#5 HTML5 API和Web应用

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

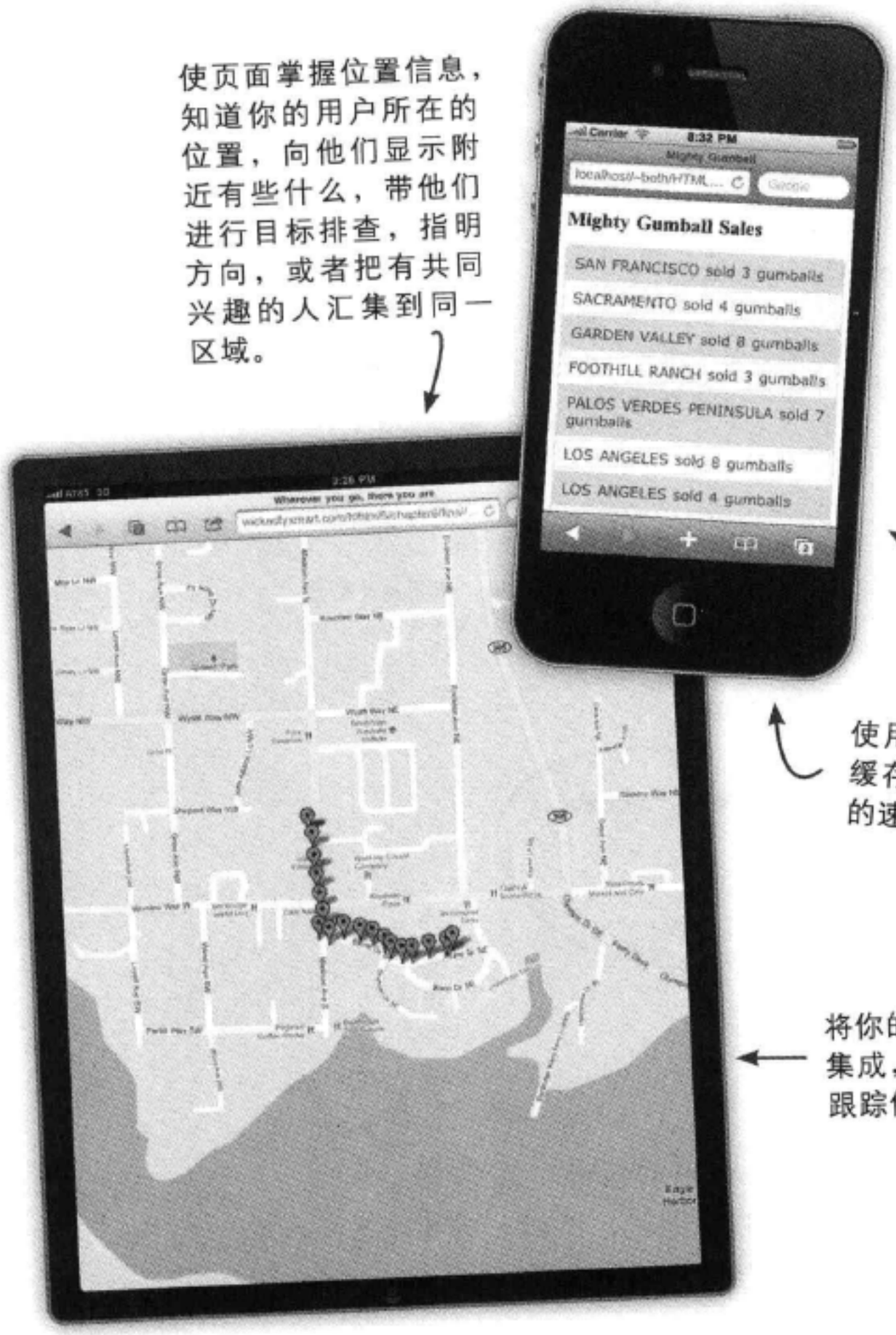
采用新方法与页面交互，这些方法同时适用于桌面和移动设备。



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

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

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



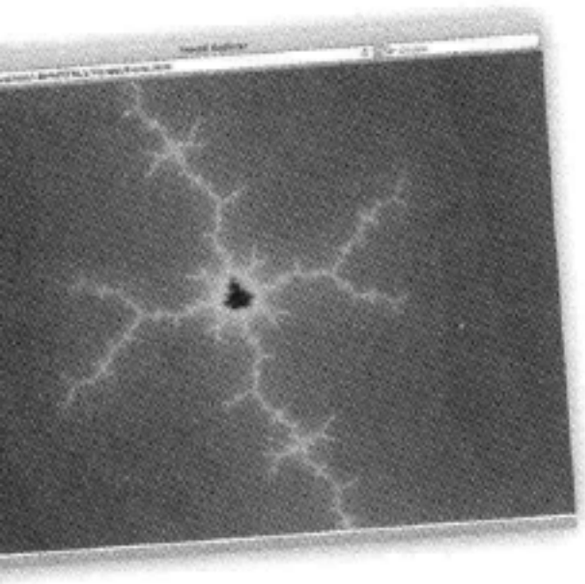
访问任何Web服务，并将数据（近实时地）传回应用。

使用浏览器存储在本地的缓存数据，提高移动应用的速度。

不再需要特殊的插件来播放视频。

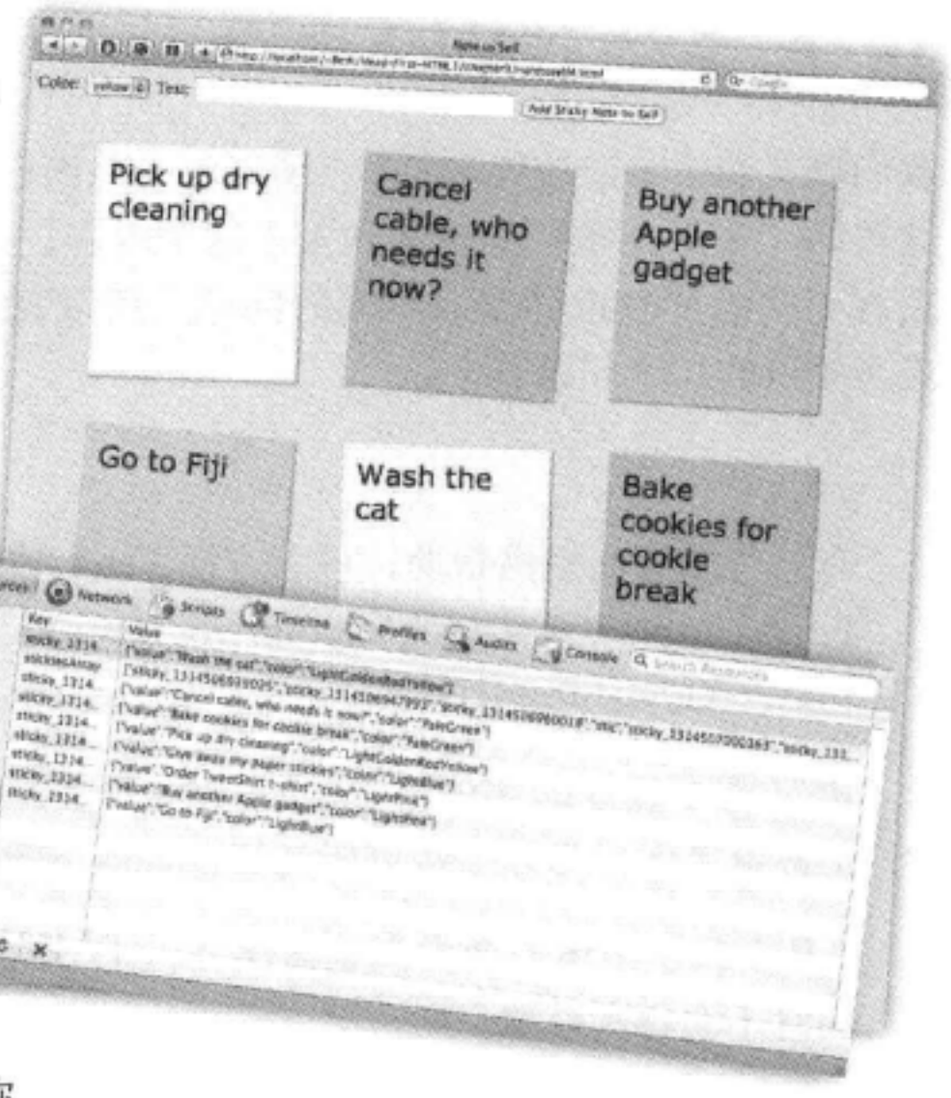
将你的页面与Google Maps集成，甚至允许用户实时地跟踪他们的移动轨迹。

使用HTML和JavaScript创建你自己的视频回放控件。

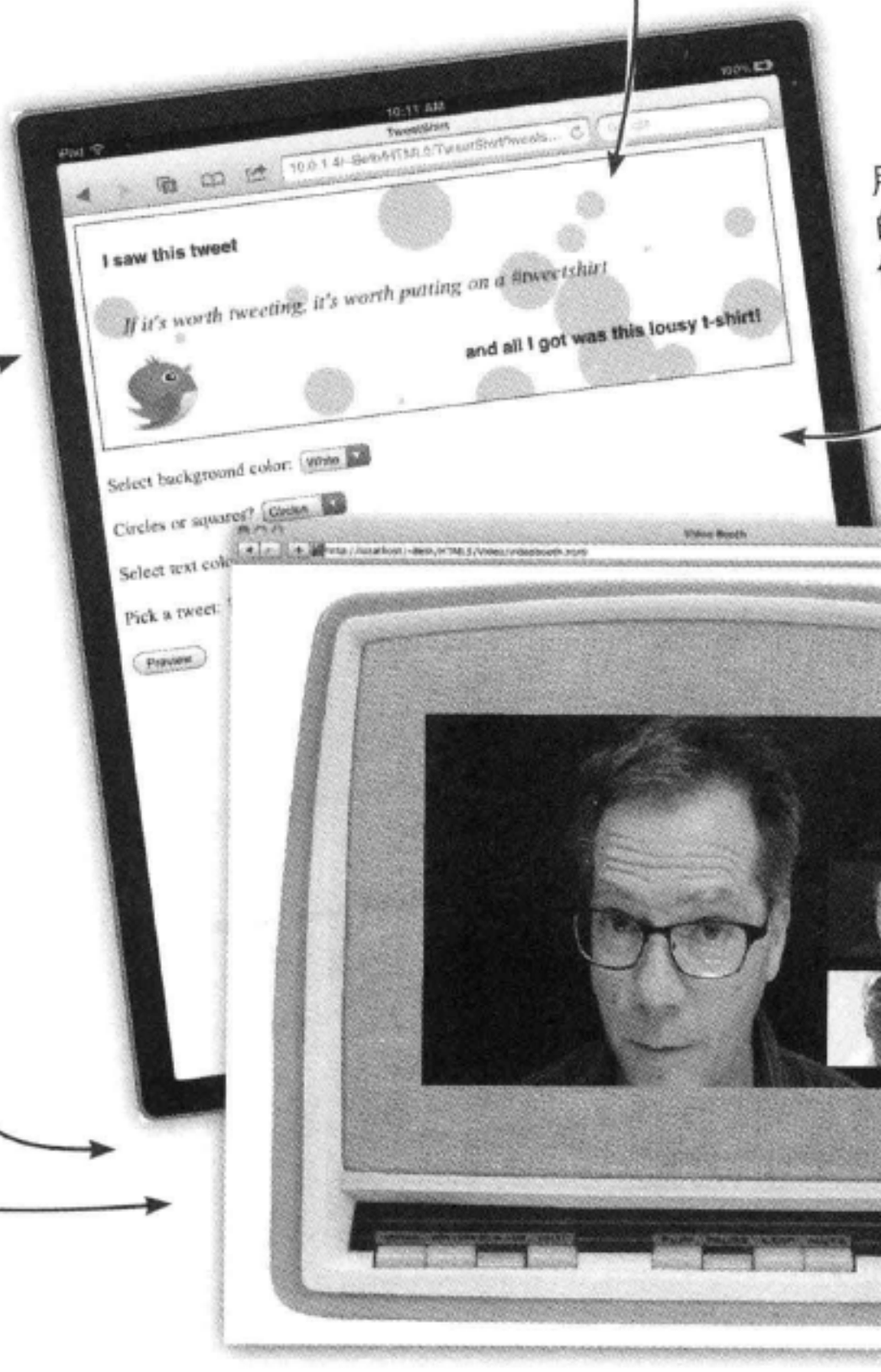


浏览器肯定不再只是应付枯燥的文档了。有了JavaScript，你可以直接在浏览器上绘制像素。

利用浏览器的本地存储。



你可以为用户在本地（浏览器中）存储大量首选项和数据，甚至可以离线访问。



用JavaScript可以让你的表单如虎添翼，提供真正的交互性。

建立完整的视频体验，可以采用新的方式加入视频。

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

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

#6 再来谈谈Web字体

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

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

#7 创建Web页面的工具

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

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

这些工具并非没有缺点：

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

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

可以考虑的一些工具

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

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

#8 XHTML5

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

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

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

相同的doctype!

这是XML，我们需要增加命名空间。

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

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

可以把XML直接嵌在页面中！太酷了。

对于你的XHTML页面，有几个问题需要考虑：

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

良构是指，要结束所有元素，用引号包围属性值，合法地嵌套元素等。

前面说过，关于XML，需要了解、需要注意的内容有很多，愿主与你同在……

#9 服务器端脚本

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

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

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

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

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

#10 音频

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

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

看着很熟悉？没错，音频支持的功能与视频类似（当然要除于视频）。

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

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

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

索引

符号

&amp abbreviation, &amp缩写，112～113

& (ampersand) for entities, &（与字符）用于实体， 113

/* and */ for comments in CSS,/*和*/用于CSS中的注释， 285

<!-- and --> for comments in HTML,<!-- 和 -->用于HTML中的注释， 6

<>(angle brackets),<>（尖括号）， 25

<code> element,<code>元素， 114

: (colon) in CSS rules,CSS规则中的:（冒号）， 259

.. (dotdot) syntax for paths,..(点 点)语法用于路径， 64～65

" " (double quotes)， " " (双引号)
  for parent folders,用于父文件夹 65
  <q> element and,<q>元素和， 86～87

@font-face rule,@font-face规则， 322～325

/ (forward slash)，/（斜线）
  in closing tags,结束标记中， 26
  for paths,用于路径， 64～65

(hash symbol) for id selectors,#用于id选择器， 395

@media rules (CSS),@media规则（CSS）， 401, 405

; (semicolon) in CSS rules,CSS规则中的;（分号）， 259

[ ] (square brackets) in form element names,表单元素名中的[ ]（中括号）， 675

A

<a> elements，<a>元素
  changing styles of,改变样式， 452～453
  creating links from elements in HTML5, 由HTML5中的元素创建链接， 153
  creating links with,用来创建链接， 48～49
  in lounge.html, lounge.html中， 47
  states of, 状态， 466

absolute layouts, 绝对布局， 522

absolute paths, 绝对路径， 136～137, 145, 159

absolute positioning,绝对定位， 504～510, 528～529, 536～537

action attribute, action 属性， 650～651, 661, 692

active link state, active链接状态， 453

alt attribute, alt属性， 173, 211, 242

angle brackets (< >),尖括号（<>）， 25

anti-aliasing text,反锯齿文本， 210

Applications folder, Applications文件夹， 12

application/xhtml+xml MIME type, application/xhtml+xml MIME类型， 708

<article> element,<article>元素， 562～564, 572～573, 595

<aside> element,<aside>元素， 551, 595

attributes，属性
  attribute selectors,属性选择器， 698
  of elements,元素， 29, 51～53
  matching to elements,与元素匹配， 52
  in opening tags,开始标记中， 36

<audio> element,<audio>元素， 710

“auto” margins, “auto” 外边距， 502

autoplay attribute (<video>),autoplay属性(<video>)， 583, 584

B

backgrounds，背景
  background-color property, background-color 属性， 618, 634
  background-image property, background-image 属性， 380～383, 405
  background-position property, background-position 属性， 383, 405
  background-repeat property, background-repeat 属性， 383, 405
  colors (Web pages),颜色（Web页面）， 206, 210

block elements，块元素
  flowing,流， 473～478, 537
  planning pages with,用来规划页面， 115

<blockquote> elements，<blockquote>元素, 90～95, 92

<body> tags,<body>标记， 8, 23～24

bold text,粗体文本， 335～336

borders，边框
  adding to element structure,增加到 元素结构， 424～433

border-bottom property, border-bottom 属性， 265~266, 354
border-collapse property, border-collapse 属性， 616, 634
border-color property, border-color 属性， 387, 389
border-radius property, border-radius 属性， 388, 405, 411
border-spacing (cells), border-spacing（单元格）， 634
border-spacing property, border-spacing 属性， 516, 616, 639
border-style property, border-style 属性， 386
border-width property, border-width 属性， 387
bottom, 265
box model and,盒模型， 369～370, 377～379, 387
displaying in browsers,浏览器中显示， 31
shorthand for,简写， 442～445
specifying corners of,指定边角， 388
bottom property, bottom 属性， 504
box model (CSS),盒模型（CSS）， 367～372

 elements，
元素, 96～99, 115
broken images in browsers,浏览器中破坏的图像， 215
browsers, Web 参见 Web browsers

C

<caption> element,<caption>元素， 634
captions (HTML tables),标题（HTML表格） 609～610
cd (change directory) command,cd(改变目录）命令， 130
cells, table，单元格，表格， 634
character encoding,字符编码， 238～240
character entities,字符实体， 112～113, 115
charset attribute (<meta> tags), charset 属性（<meta>标记），239，249
checkbox controls,复选框控件， 692
checkbox <input> element,checkbox <input>元素， 653, 663, 673～674
checked attribute (forms), checked 属性（表单）， 695
Chrome, 16
class attributes, class 属性， 301
classes，类
adding elements to,增加元素， 286～291
class attributes, class 属性， 392～397
placing elements into ，放置 元素, 421
clear property, clear 属性， 495～497, 537
closing tags,结束标记， 25
codecs, 589, 590～592
collapsing borders,折叠边框， 616
colors，颜色，
adding to HTML tables,增加到HTML表格， 618～620
background (Web pages),背景（Web页面）， 206, 210
“color” type attribute, “color” 类型属性， 692
color <input> element (forms),颜色<input>元素（表单），656
Color Pickers, 206, 348～349
color property (CSS),color属性（CSS）， 262, 313
of headings, changing,标题，改变， 439
naming,命名， 343
Web colors, 参见 Web colors
colspan attribute, colspan 属性， 624, 634
comments，注释
CSS, 285
HTML, 6
container file format,容器文件格式， 589
content area (box model),内容区（盒模型）， 368, 371, 372, 405
Content Delivery Network (CDN),内容分发网络（CDN），591
controls attribute (video),属性（视频）， 584
CSS (Cascading Style Sheets)，CSS（层叠样式表）
box model,盒模型， 367～372
cascade,层叠， 458～463
comments in,注释， 285
comparing languages with HTML,与HTML比较， 294～295
CSS Pocket Reference, 260, 445
CSS table displays，CSS表格显示
creating,创建， 510～520
laying out forms with,用来建立表单布局， 682～685
layouts,布局， 522
errors in,错误， 297
vs. HTML, 34～35
inheriting styles from parent elements,从父元素继承样式，281～285
linking pages to external stylesheets,页面链接到外部样式表， 273～277
properties overview,属性概览， 300
rules,规则 36, 259～260, 301
selectors,选择器， 698～699

<style> element,<style>元素， 29～32, 261～263
style definitions,样式定义， 42
styling forms with, 用来指定表单样式，686～687
styling upgrade project,样式升级项目， 362～365
transforms and transitions,变换与过渡， 701～702
updating for HTML5 elements,更新到HTML5元素， 554
validating,验证， 298～299
vender-specific properties,开发商特定属性， 700
cursive fonts, cursive 字体， 315

D

data transfers (hosting),数据传输（托管）， 125
“date” type attribute, “date”类型属性， 692
date <input> element (forms),date <input>元素（表单）， 657, 671～672
datetime attribute, datetime 属性， 565～566
default.htm files, default.htm 文件， 138～139, 159
definition lists,定义列表， 106
<del> element (HTML),<del>元素（HTML）， 353
descendant selectors,子孙选择器， 437～439, 466
dir command, dir 命令， 131
directions.html, 54
directories (folders),目录（文件夹）， 130
display: table property, display: table 属性， 516
 element (HTML), 元素（HTML）
adding styles to,增加样式， 424～433
defined,定义， 466
dividing pages into sections with,用来将页面划分为区块，417～422
line-height property and, line-height 属性和， 440
new HTML5 elements and,新HTML5属性和， 595
doctype definitions,doctype定义， 225～227, 229～230, 249
domain names,域名， 126～128, 159
Dreamweaver, 6

E

editors, text,编辑器，文本， 16
elements，元素
adding to classes,增加到类， 286～291
basics,基础知识， 25～26, 36
defined,定义， 25
floating,浮动， 479～482, 487～490, 497, 525～529
form,表单， 652～657
height of,高度， 430
linking to with IDs,用ID链接， 151
multiple rules for,多个规则， 267
nesting,嵌套， 107～109
new in HTML5, HTML5中新增， 547～550
selecting by siblings,由兄弟选择， 699
structure,结构， 36
styling based on state,根据状态指定样式， 453～454
elixir.html, 54
 element,元素， 92, 114, 338
<email> element (forms),<email> 元素（表单）， 663
email <input> element (forms),email <input> 元素（表单），657
Embedded OpenType fonts, Embedded OpenType 字体， 325
em units for sizing fonts,指定字体大小的em单位， 329, 334
encoding，编码
character,字符， 238～240
formats (video),格式（视频）， 586～587
entities, character,字体，字符， 112～114
executable content in Web pages,Web页面中的可执行内容，703
external stylesheets,外部样式表， 273～277, 301

F

fantasy fonts, fantasy 字体， 315
<fieldset> element,<fieldset>元素， 689, 692
files，文件
creating in Mac,Mac中创建， 12～13
creating in Windows, Windows中创建， 14～15
extensions,扩展名 15
file <input> element (forms),file <input>元素（表单），690
“file:///” protocol, “file:///”协议， 145
file protocol,协议， 159
organizing in folders,文件夹中组织， 56～59
transferring to server root folder,传输到服务器根文件夹，129～133
Firefox, 16
:first-child pseudo-class,:first-child伪类， 454

:first-letter pseudo-element,:first-letter伪元素， 698
:first-line pseudo-element,:first-line伪元素， 698
fixed position elements,固定位置元素， 537
fixed positioning,固定定位， 506, 531～534, 536
Flash video, Flash视频， 592
floated elements,浮动元素 537
floating，浮动
  elements,元素， 525～529
  float property, float 属性， 472, 478～482, 487～490
  inline elements,内联元素， 497
  layouts,布局， 521, 525～526
flowing block/inline elements,流动块/内联元素， 473～478
flow of elements,元素流， 537
:focus pseudo-class,:focus伪类， 453
folders，文件夹
  organizing files/images in,在其中组织文件/图像， 56～59
  for thumbnails, 缩略图，192
fonts (CSS)，字体（CSS）
  changing weight of,改变粗细， 335～336
  colors, background vs. font,颜色，背景与字体， 349
  families of,字体系列， 355
  @font-face rule,@font-face规则， 322～325
  font-family property, font-family 属性， 279～280
  for Mac/Windows,面向 Mac/Windows， 321
  properties,属性， 312～313
  shorthand for,简写， 444
  sizing,指定大小， 328～334
  styling,指定样式， 337～339
  Web Fonts, Web 字体， 325～327, 706
footers，页脚（底部）
  <footer> element (HTML5),<footer>元素(HTML5), 551, 595
  laying out,布局， 493～496, 499
formats，格式
  image,图像， 167
  video,视频， 586～591
forms, HTML，表单，HTML
  action attribute, action 属性， 650, 661, 665, 692
  adding checkboxes/text area to,增加复选框/文本区， 673～674
  adding fieldsets/legends to,增加fieldset/legend， 689
  adding <input> elements to ，增加<input>元素, 664～666, 671～675
  adding <label> elements for accessibility ，增加<label>元素提高可访问性, 688
  adding password <input> element to,增加password <input>元素， 689
  basics,基础知识， 646～649
  commonly used elements,常用元素， 652～657
  file <input> element,file <input>元素， 690
  <form> element,<form>元素， 649～651, 660～663, 692
  GET vs. POST methods, GET vs. POST方法， 678～680
  laying out with CSS table display,用CSS表格显示布局， 682～685
  multiple choice menus,多选菜单， 690
  name attribute, name属性， 662
  placeholder attribute, placeholder 属性， 691
  required attribute, required 属性， 691
  server scripts, 服务器脚本，646～647, 650～652, 660, 663
  styling with CSS,用CSS指定样式， 686～687
frozen layouts,冻结布局， 501～502, 537
FTP (File Transfer Protocol),FTP（文件传输协议）， 129～132, 159

G

get <filename> command,get <filename>命令， 131
GET method,GET方法， 678～680, 692
GIF image format,GIF图像格式， 167～168, 172, 211
Google Web Fonts, Google Web 字体， 325～327, 706
Guide, HTML, 245～246

H

<h1> element (headings),<h1>元素（标题）， 22
<h2> element (subheadings),<h2>元素（子标题）， 8, 22
<head> element,<head>元素， 8, 23～24, 36
<header> elements，<header>元素, 551, 568～571, 572～573, 595
header images,页眉图像， 523～524
Head First HTML5 Programming, 6, 52, 231, 593, 705
Head First learning principles, Head First学习原则， xxviii
Head First Lounge project, Head First休闲室项目， 4～5
Head First Mobile Web, 403
headings，标题
  changing color of,改变颜色， 439
  levels of,级别， 6

height properties， height 属性
attribute,属性， 174, 584
CSS box model and,CSS盒模型， 366, 371
of elements,元素， 430
property,属性， 570
hex code (colors),十六进制码（颜色）， 32, 345～347, 349, 355
“Hide extensions for known file types” option, “已知文件类型隐藏扩展名”选项， 15
hosting companies, 托管公司， 125, 159
hover state,悬停状态， 453～454
href attribute， href 属性
basics (interview),基础（采访） 53
relative paths in, 相对路径， 59
specifying link destination with, 用来指定链接目标， 48～51
HTML5
APIs and web apps， API和Web应用， 704～705
browser support for,浏览器支持， 553
building blog with new elements,用新元素构建博客， 562～569
vs. HTML4.01, 555～556
new elements in,新元素， 546～550, 594, 595
specification， 规范
doctype, 227
overview, 概览， 231, 242
HTML (Hypertext Markup Language)， HTML（超文本标记语言）
basics,基础知识， 4
comments,注释， 6
creating tables with 参见 tables, HTML
creating web page,创建Web页面， 9～11, 17～22
vs. CSS, 34～35
forms. 参见 forms, HTML
guidelines for well-formed pages,良构页面原则， 245～246
Guide to, 指南， 245～246
.html extension,.html扩展名， 14
<html> tag,<html>标记， 6, 23
HTML & XHTML: The Definitive Guide (O’Reilly), 52
incorporating CSS into,结合CSS， 259～260
language vs. CSS, 语言与CSS， 294～295
legacy elements,遗留元素， 247
living standard, 活标准， 228
marking up page structure, 标记页面结构， 38～41
overview,概览， 2～3
readability of,可读性， 6
saving,保存， 18
structure for table displays, 表格显示结构， 512～514
structuring text with tags,用标记建立文本结构， 21～23
version history of,版本历史， 222～225
HTTP (HyperText Transfer Protocol),HTTP（超文本传输协议）， 135, 159
hypertext links， 参见 links

I

id attribute,id属性， 150～153, 392～397, 405, 418
“Ignore rich text commands in HTML files” option, “忽略HTML文件中的富文本命令”选项， 13
images， 图像
adding logo to myPod application,为myPod应用增加logo， 202～209
broken images in browsers,浏览器中破坏的图像， 215
browser handling of,浏览器处理， 164～166
fixing broken, 修正破坏的图像， 66～68
formats,格式， 167, 211
<img> element,<img> 元素， 55, 170～172, 381
<img> elements， <img> 元素, 211
as links,作为链接， 211
organizing in folders,在文件夹中组织， 57～58
quality of, 质量， 187
sizing/resizing,指定大小/调整大小， 174, 178, 183～184
using as list markers,用作为列表标记， 632～633
!important, 459, 461
index.html file,index.html文件， 11, 18, 138～139, 159, 176
inherited properties,继承属性， 301, 464
inheriting styles, 继承样式， 281～285
inline elements， 内联元素
basics,基础知识， 94～95
flowing,流动， 473～478, 537
<q> and ,<q> 和, 115
setting properties on,设置属性， 450
<input> elements (forms),<input>元素（表单）， 652～653, 656～657, 664～667, 670～674
<ins> element,<ins>元素， 353
Internet Explorer, 16
italic style text,斜体风格文本， 337～339

J

JavaScript, 703
jello layouts,凝胶布局， 502～503, 521, 537
JPEG images, JPEG 图像， 167～168, 172, 211

K

keywords for sizing fonts,设置字体大小的关键字， 330, 334

L

labels，标签
  <label> elements，<label>元素, 688, 692
  labeling with id attribute,用id属性标记 ， 418
  link,链接 148
:last-child pseudo-class,:last-child 伪类， 454
layouts,CSS, 布局，CSS， 521～523
leading (text),前导（文本）， 366
left property, left 属性， 504
legacy HTML elements,遗留HTML元素， 247
<legend> element,<legend>元素， 689
 elements，<li>元素, 101～106
lighter font-weight property, lighter font-weight 属性， 335
linebreaks,换行 95
line-height property, line-height 属性， 365～366, 405, 440
<link> element,<link>元素， 275～277, 301
links，链接
  adding titles to,增加标题， 147～149
  Head First Lounge example, Head First休闲室示例， 44～50
  :link pseudo-class,:link 伪类， 455
  linking into parent folders,链接到父文件夹， 63～65
  linking into subfolders,链接到子文件夹， 60～62
  linking to new windows,链接到新窗口， 155～157
  linking to points in pages,链接到页面中的点， 149～152
  linking within pages,页面中链接， 153
  link labels,链接标签， 148
  link states,链接状态， 453
  to other websites,链接到其他网站， 142～145
  relative, 相对，154
  turning thumbnails into,把缩略图变成， 196～200
liquid layouts,流体布局， 501～502, 537
lists，列表
  list-style-image property, list-style-image 属性， 632
  list-style-position property, list-style-position 属性， 633
  list-style-type property, list-style-type 属性， 631
loop attribute (video), loop 属性（视频）， 584
lossy/lossless formats (images),有损/无损格式（图像）， 167～168
lounge.html, 54, 66

M

Mac OS X
  creating HTML files in,创建HTML文件， 12～13
  FTP applications for,FTP应用， 132
  specifying fonts for,指定字体， 321
margins，外边距
  box model and,盒模型， 369, 371～372, 377～379
  margin-right property, margin-right 属性， 385
  settings for,设置， 405
  shorthand for,简写， 442～445
markers, list, 标记，列表，631～632
matching tags,匹配标记， 25
matte color, setting, 蒙版颜色，设置，206～207
Matte option，蒙版选项
  in Photoshop Elements, Photoshop Elements中， 210
  in “Save for Web” dialog box, “Save for Web”对话框中， 205
max-device-width property, max-device-width 属性， 400, 404, 412
max-width property, max-width 属性， 404, 412
@media rules (CSS),@media规则（CSS）， 401
media attribute, media 属性， 400
media queries,媒体查询， 401～402
<meta> tags,<meta> 标记， 239～240, 249
metacognition,元认知， xxix–xxxi
method attribute (forms), method 属性（表单）， 650, 692
MIME type,MIME类型， 590
min-device-width property, min-device-width 属性， 400, 404, 412
min-width property, min-width 属性， 404, 412

mismatching tags,失配标记， 109～110
mission.html page, mission.html 页面， 33
mkdir (make directory) command, mkdir （建立目录）命令， 131
monospace fonts, monospace 字体， 315
MP4 containers, MP4 容器， 586～587
multiple attribute (forms), multiple 属性（表单）， 690
multiple classes,多个类， 291
multiple custom fonts,多个定制字体， 327
multiple links to stylesheets,多个样式表链接， 463
multiple stylesheets,多个样式表， 399～400
myPod application (images), myPod 应用（图像）， 175～177, 188～191

N

name attribute (form element), name 属性（表单元素）， 662～663
naming，命名
  classes/ids,类/id， 397
  colors,颜色， 343
<nav> element (HTML5),<nav>元素(HTML5), 575～577, 595
navigating multiple pages,导航多个页面， 573～577
negative property values,负属性值， 533～534
nesting，嵌套
   elements， 元素, 420, 466
  elements,元素， 107～109
  lists,列表， 115
  nested tags,嵌套标记， 26
  tables,表格， 634
normal keyword, normal 关键字， 445
Notepad, 14
:nth-child pseudo-class,:nth-child伪类， 619, 634
“number” type attribute, “number” type属性， 692
number <input> element (forms),number <input> 元素（表单）， 656, 671～672

O

oblique style text,倾斜风格文本， 337～339
Ogg container, Ogg 容器 586～587
 element,<ol>元素， 103～106
online color charts,在线颜色表， 349
“Open and Save” tab, “Open and Save”标签页 13
opening tags,开始标记， 25
OpenType fonts, OpenType 字体， 325
Opera, 16
<option> element (forms),<option>元素（表单）， 655, 663, 666, 692
ordered lists,有序列表， 102～105, 115, 633
orientation property, orientation 属性， 400
overriding style inheritance,覆盖样式继承， 284～285

P

<p> elements，<p>元素, 55, 101
<p> tags,<p>标记， 8, 22
padding，内边距
  basics,基础知识， 405
  box model and,盒模型， 368, 371～372, 377～379
  padding-left property, padding-left 属性， 384
  shorthand for,简写， 442～445
paragraphs，段落
  linking,链接 55
  styling independently (box model),单独指定样式（盒模型）， 375～383
parent folders, linking into, 父文件夹，链接到，63～65
password <input> element,password <input> 元素， 689
paths (links)，路径（链接）
  absolute,绝对， 136～137, 145
  planning,计划， 60～65
  relative,相对， 137, 145
percentages，百分数
  positioning elements with,用来定位元素， 506
  sizing fonts with,用来指定字体大小， 328～329, 334
photo images,size of, 照片图像，大小， 211
Photoshop Elements
  finding Web colors with,用来查找Web颜色， 348～349
  Matte color menu in,其中的蒙版颜色菜单， 211
  resizing images with,调整图像大小， 181
pixels，像素
  pixel resolutions,像素分辨率， 179～180
  sizing fonts with,用来指定字体大小， 328
placeholder attribute (forms), placeholder 属性（表单），

691, 692
"Plain text", 13
PNG images,PNG图像， 167～168, 172, 203～205, 211
ports,端口， 145
position property, position 属性， 504～507, 537
poster attribute (video), poster 属性（视频）， 584
POST method, POST 方法， 678～680, 692
<pre> element,<pre>元素， 114
Preferences, TextEdit, 13
preload attribute (<video>), preload属性(<video>)， 584
properties，属性
  CSS, 300
  of fonts,字体， 312～313
  inherited, 继承，301, 464
  list-style (CSS), 634
  shortcuts for,快捷方式， 466
pseudo-classes,伪类， 454～456, 466, 619
pseudo-elements,伪元素， 698
put <filename> command,put <filename>命令， 130
pwd command (FTP),pwd命令（FTP），131

Q

<q> elements，<q> 元素, 86～88, 92, 117

R

radio buttons, 单选钮，692
radio <input> element,radio <input>元素， 653, 663, 668
ragged borders,锯齿边框， 389
"range" type, "range" 类型， 692
range <input> element,range <input>元素， 656
relative font sizing,指定相对字体大小， 328～329
relative links,相对链接， 154
relative paths，相对路径
  absolute paths and, 绝对路径，137
  basics,基础知识， 69～71
  grand challenge solutions,挑战答案， 75～76
  vs. URLs, 145, 159
relative positioning,相对定位， 506, 536, 537
reload button (browsers),重新加载按钮（浏览器），24
required attribute, required 属性， 692
required attribute (forms), required 属性（表单），691
RGB color values，RGB颜色值
  specifying in CSS, CSS中指定，344
  Web colors and,Web颜色， 340～342
right property, right 属性， 504
root folders (server),根文件夹（服务器）， 129～133
rows (HTML tables)，行（HTML表格）
  adding color to,增加颜色， 618～620
  cells spanning multiple,单元格跨多行， 622～624
  columns and,列， 607
rules, CSS，规则，CSS
  cascade and, 层叠，459
  combining, 合并，264
  ordering of, 顺序，293, 459
  overriding inherited styles with,用来覆盖继承的样式， 284～285
  syntax,语法， 259～260
  writing for multiple elements,为多个元素写规则， 264～266

S

Safari, 16
sans-serif fonts, sans-serif 字体， 314, 320
saving，保存
  HTML, 18
  images,图像， 187
  "Save for Web" option, "Save for Web" 选项， 183～184, 187, 204～205
screen readers,屏幕阅读器， 157
<script> element,<script>元素， 703
scripting, server-side,脚本，服务器端， 709
<section> element,<section>元素， 562～564, 572～573, 595
<select> element,<select>元素， 692
selecting，选择
  elements by siblings,按兄弟选择元素， 699
  elements with ids/classes, 用id/类选择元素，395
  <select> element,<select>元素， 655, 663, 665～667

selectors (CSS)，选择器(CSS)
basics,基础知识， 267
class,类， 288
combining,合并， 698～699
descendant,子孙 437～439
serif fonts, serif 字体， 314, 320
server-side scripting,服务器端脚本， 651, 709
SFTP (Secure File Transfer Protocol),SFTP(安全文件传输协议)，132
shortcuts, property,快捷方式，属性， 466
shorthand, CSS,简写，CSS， 442～445
sidebars, laying out,边栏，布局， 488～490, 499
sizing/resizing，指定大小/调整大小
fonts,字体， 312, 328～334
images,图像， 174, 178～184, 183～185
<source> element,<source>元素， 589～591
 element (HTML),元素（HTML）,448～450, 466
special characters,特殊字符， 112～113
specificity, calculating,特定性，计算， 460～461
src attribute (CSS), src属性(CSS)， 68～69, 170, 211, 582, 584
Starbuzz Coffee project， Starbuzz Coffee项目
adding CSS to,增加CSS， 30～33
basic structure with HTML tags,用HTML标记建立基本结构， 21
creating Web page,创建Web页面， 11～12
loading content into browser,在浏览器中加载内容， 17
markup,标记， 38
structure,结构， 10
states of links,链接状态， 453
static positioning,静态定位， 506, 536, 537
 element,元素， 114
styles，样式
guide to applying,应用样式的原则， 292～293
inheriting from parent elements,从父元素继承， 281～285
<style> element,<style>元素， 29～32
<style> element, placing,<style>元素，放置， 36
<style> tags,<style>标记， 261～263, 301
stylesheets, multiple,样式表，多个， 399～400, 405
styling fonts,指定字体样式， 337～339
subfolders, linking into,子文件夹，链接， 60～62
subheadings, HTML,子标题，HTML， 22
submit buttons,提交按钮， 692
submit <input> element (forms),submit <input>元素（表单），652, 663
SVG fonts, SVG 字体， 325

T

<table> element (HTML),<table> 元素（HTML）， 634
tables, HTML，表格，HTML
adding captions to,增加标题， 609～610
adding color to,增加颜色， 618～620
adding styles to,增加样式， 612～616
cells spanning multiple rows,单元格跨多行， 622～624
collapsing borders,折叠边框， 616
creating,创建， 603～607
CSS table displays,CSS表格显示， 537, 607
pasting into Web pages,粘贴到Web页面， 611
styling lists in,指定列表样式， 631～633
tags,HTML， 标记，HTML
basics,基础知识， 25～42
in Head First Lounge project, Head First休闲室项目， 5～6
matching, 匹配，25
mismatching,不匹配， 109～110
nested,嵌套， 26
Starbuzz Coffee project, Starbuzz Coffee 项目， 21
starting and ending,开始和结束， 26
structuring text with,用来建立文本结构， 21～23, 39～41
target attribute, target 属性， 156～157, 159
<td> element,<td>元素， 604～606, 624, 634, 641
tel <input> element (forms),tel <input>元素（表单）， 657, 663
text，文本
anti-aliasing,反锯齿， 210
editors,编辑器， 16
flowing onto Web pages,流入Web页面， 478
fonts, 参见 fonts (CSS)
text-align property, text-align 属性， 431～433, 466, 634
<textarea> element (forms),<textarea>元素（表单）， 654, 663, 673, 692
text-decoration property (CSS), text-decoration属性（CSS），267, 313, 353, 355
TextEdit (Mac), 12～13
text <input> element,text <input>元素， 652, 692
wrapping around list markers,包围在列表标记中， 633
<th> element (HTML),<th> 元素（HTML）, 604～606, 634

thumbnails, 缩略图， 192～196, 211
<time> element,<time> 元素， 114, 565～566, 595
<title> element,<title>元素， 8, 23～24
title attribute (<a> element),title属性(<a>元素)， 147～149
tooltips,工具提示， 153
top property, top属性， 504
<tr> element,<tr>元素， 604～606
transforms and transitions (CSS),变换和过渡(CSS)， 701～702
transparency in images,图像中的透明度， 204～205, 210
TrueType fonts, TrueType 字体， 325
type attribute, type 属性， 51, 652

U

 element,<ul>元素， 103～106, 118～119
underlining text,文本加下划线， 267
Unicode, 112～113, 239
unordered lists,无序列表， 102～105, 633
URLs (Uniform Resource Locators)，URL(统一资源定位符
  basics,基础知识， 134
  defined,定义， 159
  for images,图像， 171～172
  <url> element (forms),<url> 元素（表单）， 663
  url <input> element (forms),url <input>元素（表单）， 657
UTF-8 encoding, UTF-8编码， 18, 239, 249

V

validating，验证
  CSS validator,CSS验证工具， 298～299
  W3C Validator,W3C验证工具， 233～240
value attribute (forms), value 属性（表单）， 663, 692
vendor-specific CSS properties,开发商特定的CSS属性， 700
vertical-align property, vertical-align 属性， 516, 520, 634
<video> element (HTML5)，<video>元素(HTML5)
  attributes,属性， 584
  basics,基础知识， 580～583
  formats,格式， 586～591
:visited pseudo-class,:visited伪类， 455
void elements, void 元素， 98～99, 115, 172

W

W3C Validator, W3C 验证工具， 233～238, 249, 298～299
Web applications, HTML5 for,Web应用，HTML5， 242
Web browsers,Web浏览器
  basics,基础知识， 3
  broken images in,破坏的图像， 215
  built-in default styles,内置默认样式， 28
  choosing,选择， 16
  displaying HTML in, 显示HTML， 3～4
  displaying HTML tables,显示HTML表格， 605
  displaying HTML video,显示HTML视频， 585
  handling of forms by,处理表单， 647
  handling of images by,处理图像， 164～166
  HTML version support,HTML版本支持， 228
  interpreting HTML,解释HTML， 5～6
  loading content into,加载内容， 17～19
  opening pages in,打开页面， 19
  resizing images to fit in, 调整图像大小以适应， 180～186
  selecting, 选择， 16
  supporting HTML5,支持HTML5， 553
  whitespace and,空白符， 36
Web colors，Web颜色
  basics,基础知识， 340～342
  creating,创建， 355
  finding,查找， 348～349
  specifying,指定， 343～347
Web Fonts,Web字体， 325～327, 706
Web forms. 参见 forms, HTML
WebM container, WebM容器 586～587
Web pages，Web页面
  adding executable content to,增加可执行的内容， 703
  applications for creating,创建Web页面的应用， 707
  dividing into sections with element,用 元素划分为区块， 417～422
  how the Web works,Web如何工作， 2～6
  linking to external CSS stylesheets,链接到外部CSS样式表， 273～277
  opening in browsers,浏览器中打开， 19
  setting background color,设置背景颜色， 206
  structure of,结构， 115
Web pages, constructing，Web页面，构建
  adding <blockquote> elements，增加 <blockquote>元素, 90～94
  adding
 element,增加
元素， 96～99

adding <q> element,增加<q> 元素， 86～88
outline, 略图， 81
overview,概览， 79
rough design sketch,概要设计草图， 80
testing page,测试页面， 84
Web servers，Web服务器
basics,基础知识， 3
editing files on,编辑文件， 132
moving files to root folder on, 文件移动到根文件夹，128～131
port 80 and,端口80， 145
requests from browsers,浏览器的请求， 138
root folder,importance of, 根文件夹，重要性， 129
submitting forms to,提交表单， 646～647
Web for further information，提供进一步信息的网站
character encoding,字符编码， 239
CSS3 Media Queries specification,CSS3媒体查询规范，403
domain names,域名， 126～127
FTP applications, FTP应用， 132
hosting companies,托管公司， 125
symbol/foreign character abbreviations, symbol/foreign字符缩写，112～113
W3C validator,W3C验证工具， 233
Web Fonts,Web字体， 327
Weight property (CSS fonts), Weight属性(CSS字体)， 335～336
WHATWG and W3C, WHATWG和W3C 591
whitespace, use of,空白符，使用， 36
width attribute，width属性
images,图像， 174
video,视频， 584
width property，width属性
basics,基础知识， 426～430
borders and,边框， 387
CSS box model and,CSS盒模型， 369, 371
height of columns and,列高， 520
setting for elements,为元素设置， 466
Windows, linking to new, 窗口，链接到新窗口，155～157
Windows, Microsoft
creating HTML files in,创建HTML文件 14～15
FTP applications for, FTP应用， 132
specifying fonts for,指定字体， 321
World Wide Web Consortium (W3C),万维网协会(W3C) 249

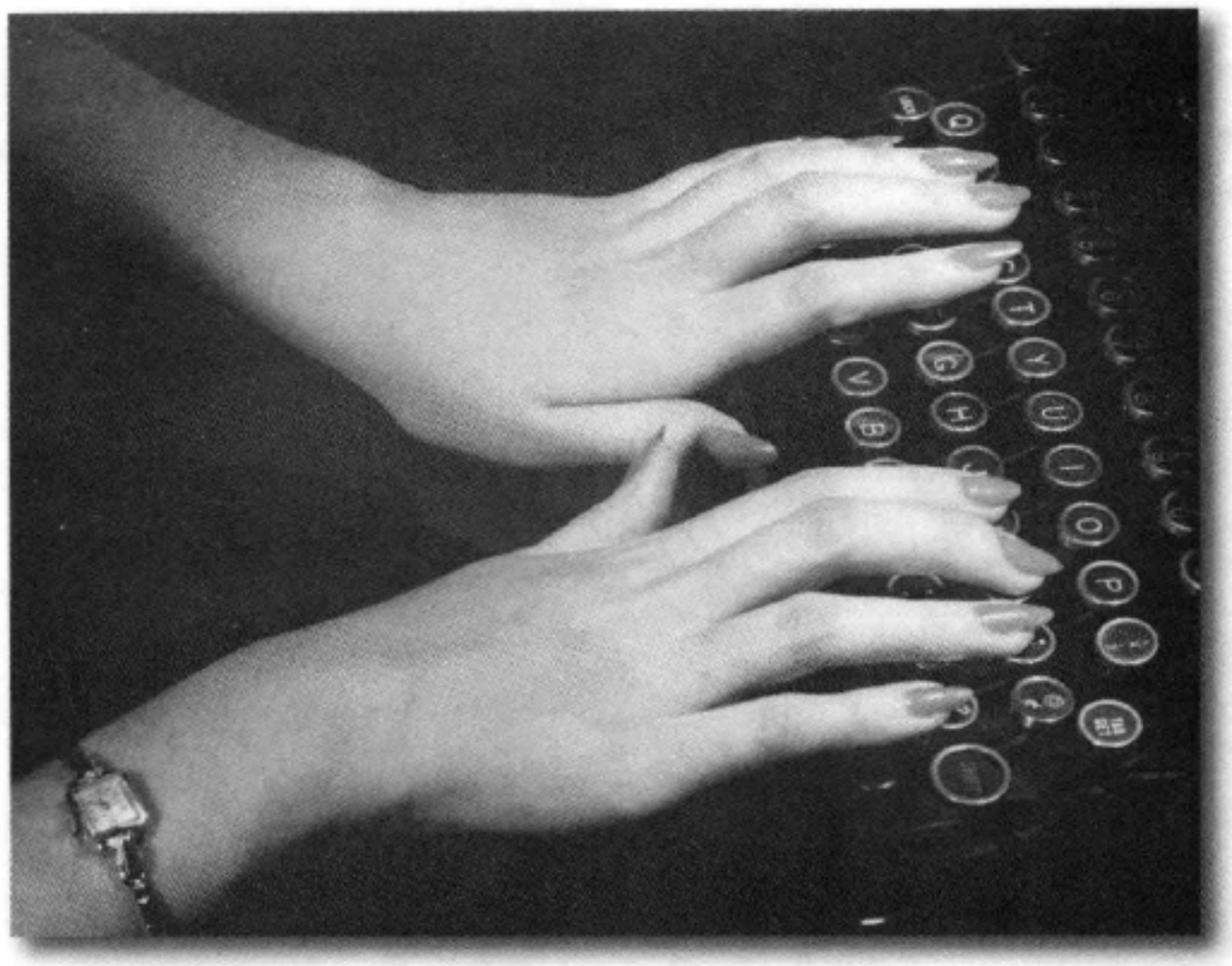
X

XHTML, 99
XHTML5, 708
XML, 223, 225, 708

Z

z-index property, z-index属性， 505～506, 537

末页



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

版面设计和制版在两台Mac Pros和两台MacBook Airs上完成。

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

不是说再见

潜心加入我们的
wickedlysmart.com