Broadview®
www.broadview.com.cn

CSS高效开发实战

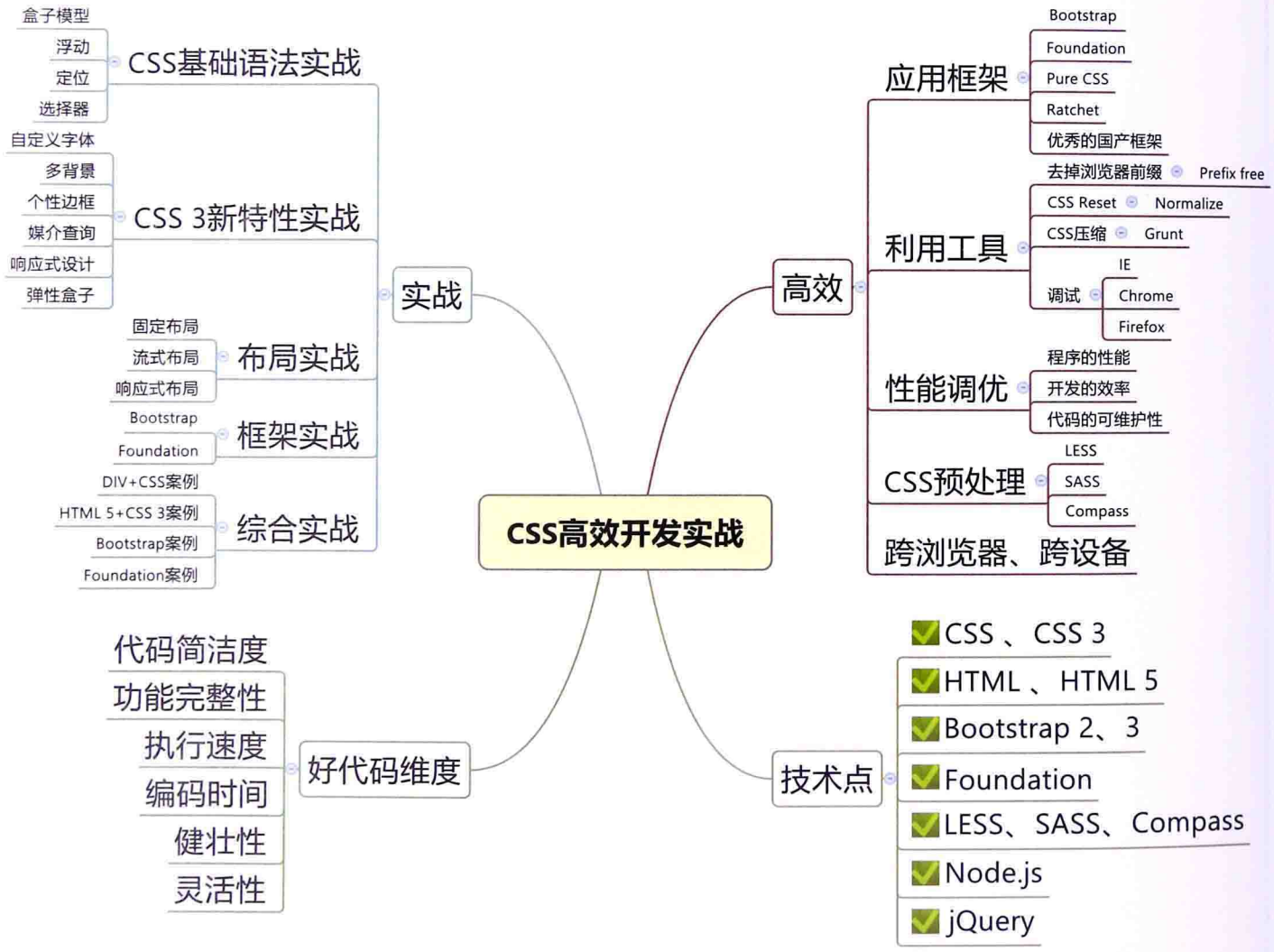
CSS 3 LESS SASS Bootstrap Foundation

—谢郁 编著—

超新、超全、超有用的CSS技术汇总
零难度实践，省时省力跨终端解决方案
学得懂，练得会

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Broadview®
www.broadview.com.cn
博文视点·IT出版旗舰品牌
技术凝聚实力·专业创新出版



博文视点Broadview
新浪微博 weibo.com
@博文视点Broadview

责任编辑：董　英
封面设计：李　玲

上架建议：计算机>程序设计

ISBN 978-7-121-23965-6
9 787121 239656 >
定价：59.00元

CSS高效开发实战

CSS 3 LESS SASS Bootstrap Foundation

—谢郁 编著—

電子工業出版社
Publishing House of Electronics Industry

内 容 简 介

想象一下，一个网页只有 HTML，没有 CSS，那就是素颜和上妆的区别。而一个网页只有 CSS，没用 CSS 3，那就是马车和汽车的区别！汽车代表的是高效、美观，CSS 3 的意图也是如此。移动设备的流行导致了响应式设计的流行，而 CSS 3 正是实现这种设计的精髓。本书围绕的就是如何跨浏览器、跨设备进行高效率的 CSS 开发。

本书分为 3 部分：第 1 部分是 HTML/CSS 基础和 CSS 3 新特性演示，涵盖盒子模型、浮动布局、属性前缀、选择器、字体、边框、背景、颜色、变换、动画、弹性盒子、媒介查询和响应式设计等多个方面；第 2 部分介绍 Bootstrap、Foundation、Pure 等当前流行的 CSS 框架及 LESS、SASS 等生产力工具；第 3 部分是实战案例，包括 DIV+CSS 案例、HTML 5+CSS 3 案例、Bootstrap 案例和 Foundation 案例。

本书内容精练、重点突出、实例丰富、讲解通俗，是广大CSS设计人员和前端开发人员必备的参考书，同时也非常适合大中专院校师生学习阅读，也可作为高等院校计算机及相关培训机构的教材使用。

图书在版编目（CIP）数据

CSS 高效开发实战：CSS 3、LESS、SASS、Bootstrap、Foundation / 谢郁编著. —北京：电子工业出版社，2014.9

ISBN 978-7-121-23965-6

Ⅰ. ①C… Ⅱ. ①谢… Ⅲ. ①网页制作工具 Ⅳ. ①TP393.092

中国版本图书馆 CIP 数据核字（2014）第 174783 号

责任编辑：董　英
印　　刷：北京中新伟业印刷有限公司
装　　订：河北省三河市路通装订厂
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编：100036
开　　本：787×1092　1/16　　印张：22.25　　字数：541 千字
版　　次：2014 年 9 月第 1 版
印　　次：2014 年 9 月第 1 次印刷
定　　价：59.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei. com.cn。

服务热线：（010）88258888。

前　言

有些人想学 CSS，不知如何下手；有些人已经学会了 CSS 的各种属性，却不知如何运用；有些人会平面设计，但不知道如何与网页设计结合；有些人会 HTML，就是学不会 CSS。本书就是为这些人准备的一本指南。正所谓知识来源于实践，本书严格恪守这一原则，对每个 CSS 特性都进行了示例演示，并在最后提供了 4 大项目，让读者接手实际项目时不再无所适从。

诊断自己的 CSS 水平

如果你不知道本书是否能够帮到你，或者说，你不知道是不是要选择本书，可以先看下面的诊断：

- HTML 代码编写学得不错，可是 CSS 网页布局和定位学不好
- 有 CSS 基础，但没有系统地学习过 CSS 3
- 不知道如何给网页布局
- 不会看 CSS 代码的好坏
- 做好的网页在不全屏的情况下会变乱
- 懂一点点的 HTML 语言，却不会网站建设
- 完全不了解如何让 CSS 代码变得高效
- 网页在不同的浏览器下显示不同
- PC 端好好的网页在手机或平板上就废了
- 学过 W3CSchool 中的 CSS 课程，但是不知道如何使用框架提高开发效率
- 想学习国外的一些 CSS 经验，了解但不熟悉 BootStrap

以上 11 条的解决方案都在本书中，正所谓会问才会学习，了解自己的不足，发现实际应用中的问题，就是本书真真正正要做的。

如何更好地学习 CSS

11 个字就能帮助我们更好地学习 CSS。

- **多看、多练**：观摩成功的网页设计，分析并练习网页设计常用的代码。
- **多想、多问**：思考设计实现的原理，提出自己的问题并通过各种渠道来找答案。
- **多总结**：记录前人已经探索出来的 CSS 技巧，总结实战中碰到的问题及解决方案。

只要真正做到勤思考、勤动手、勤总结，CSS 学习一定能一马平川。

本书的编写特点

1．将最有用的 CSS 技术汇总在一起

本书只提供最有用的 CSS 技术实践，包括 CSS 基础中的盒子模型、浮动、定位等难点，包括 CSS 3 中的圆角、动画、个性边框、媒介查询等特色，包括 CSS 的固定布局、流式布局和响应式布局等设计模型，包括最流行的 CSS 框架 Bootstrap、Foundation 的实战和 DIV+CSS、HTML 5+CSS 3 的实战。本书的写作宗旨就是好学、好用、高效，提供更好的 CSS 样式解决方案。

2．最合理的章节安排

本书首先回顾 CSS 的基础难点，然后介绍 CSS 3 的特色属性。在掌握这些之后，再介绍 LESS、SASS 等生产力工具及 Bootstrap、Foundation 等开发框架。每一部分都轻理论重实践，是一本最好的、最新的、最全的 CSS 实战教材。

3．零难度实践

市场上很多 CSS 书，抽象又难以理解，读者看完后还是难以提高。本书前 15 章的示例都力求简单实用，素材简单、代码简单，读者看完就能实践。最后 4 章的项目提供了完整的代码开发流程和设计思路，如何实现及为什么实现都穿插在讲解中，真正让读者能看完 100 页就吸收 100 页。学历是过去，能力是现在，学习能力才是未来！

4．找准实际问题，一针见血

如何既想获得丰富复杂的网页样式，同时又想节省时间和精力？如何让网页一次开发，多种设备通用？如何用更少的代码来完成同样的事情？本书就针对这些实际问题，提供现实中的解决方案。

5．中小示例，项目案例，一个都不能少

根据作者多年的项目经验，本书通过典型的示例与知识点加以整合，让读者对每章的知识点都有整体把握。最后 4 章介绍的项目案例不仅可以让读者在实际应用中更加熟练地掌握前面讲到的知识点，更能让读者了解前端开发中由轮廓到细节的完整实现流程。

本书以 CSS 实战为主，所有代码均通过作者上机调试，力求读者能学得懂、练得会。

本书的内容安排

本书共 3 篇 19 章，内容从 CSS 基础到 CSS 3 特性再到 CSS 实战。

第一篇（第 1 章~第 10 章）揭开 CSS 3 的面纱

回顾了 HTML 和 CSS 的基础知识，重点介绍了 CSS 中盒子模型、属性前缀、浮动、定位等初学者百思不得其解的难点，然后介绍阴影、圆角、渐变、弹性盒子、变换、动画、媒介查询等 CSS 3 的特性。

第二篇（第 11 章~第 15 章）使用 CSS 3 框架进行高效开发

讲述了 Bootstrap、Foundation 等流行 CSS 框架，以及 LESS 和 SASS 等生产力工具的应用，并介绍了一些流行的国内框架，如渴切、Alice 框架等。站在框架这个巨人的肩膀上，我们不仅可以提高开发效率，还能实现多人协同、风格统一。

第三篇（第 16 章~第 19 章）CSS 实战项目

首先介绍最传统的 DIV+CSS 网页的实现，要求读者能编写大量的 CSS 代码，然后介绍了如何用 HTML 5 结合 CSS 3 来开发一个网站，最后用两个实例分别介绍了 Bootstrap 和 Foundation 这一最流行 CSS 框架的实战应用。本篇使用的技术虽然是由旧到新，但是难度却是由深入浅。

本书最后的两个附录介绍了 CSS 下的调试和优化技巧。

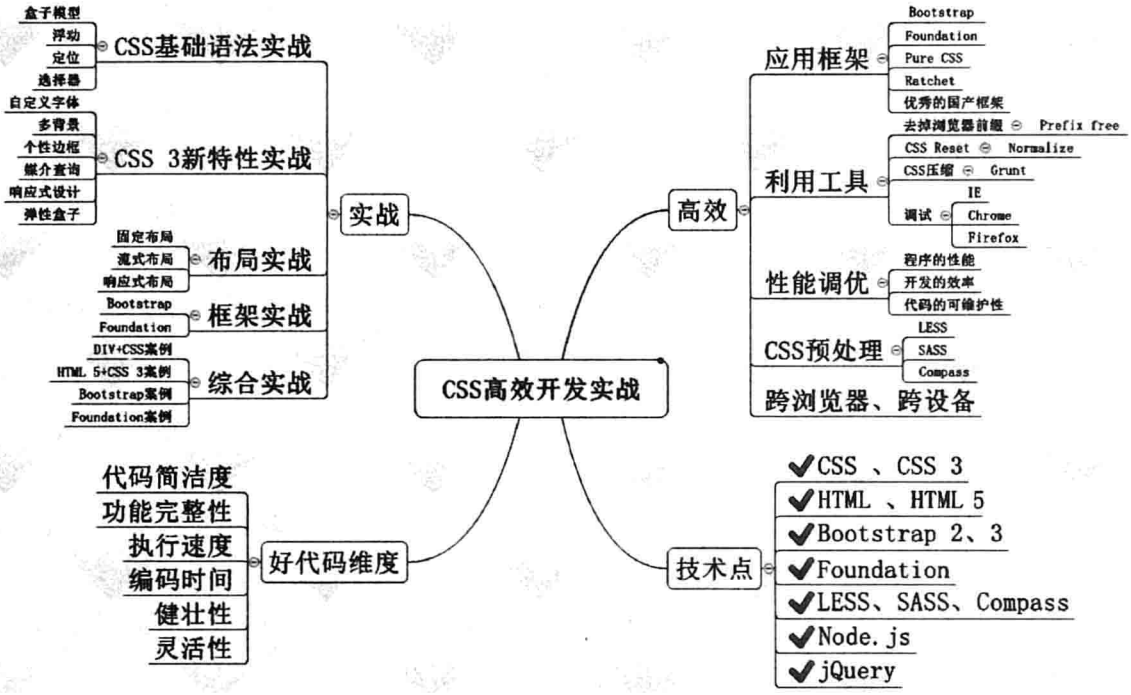
【建议阅读顺序】

CSS 完全小白，建议先练习第 1 章和第 2 章的基础内容，打好基础再继续。有 HTML 和 CSS 基础的人，建议读完第 1 章再转到第 3 章开始学习，对整体知识点有个概览。如果已经接触过 CSS 3，但困扰于如何提高效率和借鉴他人的经验，则可以预览第 1 章后，从第 11 章开始学习，学会用框架和 CSS 预处理等新技术武装自己。

本书面对的读者

- 网页设计入门者
- 网页开发入门者
- 网页美工人员
- 由 CSS 向 CSS 3 转型的开发人员
- 移动设备网页开发者
- 设计师中的 CSS 开发人员和程序员中的 CSS 开发人员
- 大中专院校的学生
- 各种 IT 培训学校的学生
- 网站后台开发人员
- 前端开发入门者
- 网站建设与网页设计的相关威客兼职人员

本书的思维导图



编者推荐

本书写作的目的是要确保 CSS 不会成为开发过程或网站性能的瓶颈，确保读者可以运用一些工具、框架、预处理来提升开发效率和节约人力成本，确保所讲解的内容读者都能活学活用。通过阅读本书，读者能知道如何应对和避免跨浏览器陷阱，如何创建一个优雅、高效、易于维护的响应式网站、如何面对形形色色的设备和大大小小的分辨率。全书包含大量的实战案例和开发技巧，总结了 CSS 开发中的最佳实践（LESS、SASS、Bootstrap、Foundation、Node.js、jQuery），讨论了各种实际问题的解决方案，是一本目前市场上绝无仅有的 CSS 高效开发实战书。

本书的服务

作者能力有限，如果书中有什么疏漏，或者对内容有什么疑问，可通过以下方式与我们沟通。

- QQ 群：296811675，作者在线答疑。
- 扫描封底的微信二维码，时刻参与我们的图书互动。
- @博文视点 Broadview 的微博，了解我们发布的信息和各种前端流行技术。
- 可到博文视点官方网站 http://www.broadview.com.cn/，下载本书所有实例源代码。

很多读者在学习过程中苦于无法交流，小故障无法及时解决，加入我们的服务方阵，我们将为您提供终身免费的服务。

本书主要由谢郁编写，参与编写的人员还有周敏、席新亮、赵荣娇、任建智、李勇、王铁民、 张兴瑜、马新原、薛淑英、殷龙、于健、周洋、董金虎、李兰英。

目　录

第一篇
揭开 CSS 3 的面纱

第 1 章　CSS 3 与现代 Web 标准

本章主要对 Web 技术的发展和新的 CSS 3 标准做一个大体上的介绍，让读者对 CSS 3 能有一个轮廓上的认识。它能做些什么，解决了哪些问题？而具体的用法会在本书后面的章节里进行详细的介绍。

本章主要知识点包括：

- Web 标准的发展
- CSS 3 的特性
- 各浏览器对新特性的支持

1.1　未来 Web 标准的发展

本章笔者主要根据自己的开发经验来谈一谈未来 Web 标准的发展。在相当长的一段时间内，Web 前端开发的三大技术 HTML、CSS、JavaScript，仍然会是主要的技术基础。虽然整体基础保持稳定并向上兼容，但 Web 技术仍然在不断发展以适应人们的需要。个人认为 Web 标准的发展有 4 个趋势：“去 Adobe”化、基础功能集成、客户端执行更多的逻辑和渲染任务、适应移动设备的发展。

1.1.1　“去 Adobe”化

从过去到现在，Web 前端开发除了三大基础技术以外，图片和 Flash 是构造精美页面和内嵌视频所必需的，Adobe 公司开发了一系列强大的图片和 Flash 制作工具，来帮助我们完成这些工作。但是在页面中大量嵌入图片和 Flash，又有很多不足：

- 一方面增大了带宽压力，降低了访问速度。
- 一方面像 Flash 的运行必须依赖浏览器插件，这会消耗大量系统资源。

这两方面大大影响了用户的体验，正因为此，苹果的乔布斯决定不在产品中支持 Flash。而对于开发者来讲，要学习使用更多的工具也会有不小的学习成本。

综上原因，让 Web 技术本身取代过去使用 Adobe 软件才能完成的工作就成了 Web 发展的趋势之一。比如，现在我们可以使用 CSS 3 来制作背景的渐变、按钮的圆角，以及一些简单的动画，而这些原本必须依赖图片和 Flash。

1.1.2 基础功能集成

更多的基础功能被集成进 HTML 5/CSS 3 系统，开发者无须再去不断地重复实现一些基础功能。比如 HTML 5 对视频、音频的内嵌，对邮件、日期选择表单的支持，无须再像过去一样编写上百行的 JavaScript 代码来实现它们。

这些功能在本书的第 3~9 章都会有详细介绍。

1.1.3 客户端执行更多的逻辑和渲染任务

相对于浏览器来说，服务器有更强大的计算能力，因此我们往往将所有的逻辑都放在服务端执行，每次执行都需要网络 I/O 并重新刷新页面。但是随着个人电脑性能的不断提升和浏览器引擎的不断发展，由客户端执行更多的逻辑计算和画面渲染成为了可能，这有两大好处：

- 一方面降低服务器和带宽的压力。
- 一方面大大提高了用户的体验。

1.1.4 适应移动设备的发展

过去 PC 机作为几乎唯一的 Web 浏览终端而存在，随着智能移动设备（手机、平板、Kindle 等）的发展，Web 开发也必须顺应这一潮流。目前新的 HTML/CSS 系统在以下主要设备的浏览器上都能支持：

- 以 iOS 操作系统为主的 iPhone、iPad 智能设备。
- 以 Android 操作系统为主的三星、HTC 等智能设备。

1.2 什么是 CSS 3

CSS，全称是 Cascading Style Sheet，一般译作层叠样式表。它是一种用于展示网页样式和布局的标记语言。目前主流浏览器支持的 CSS 版本到了 CSS 3，而 W3C 组织已经开始着手制订 CSS 4 的一些标准。

CSS 3 是 CSS 2.1 的扩展，它在 CSS 2.1 的基础上增加了很多强大的新功能，但是它已不再像 CSS 2.1 那样是单一的规范。CSS 3 被划分成几个模块，每个模块都是 CSS 的某个子集的独立规范，比如选择器、文本或背景。每个模块都有各自独立的创作者和时间表。这样做的好处是整个 CSS 3 规范的发布不再需要停下来等待某个难产的小条目，其他模块的流程可以继续向前推进。

CSS 3 的扩展主要有两方面：

- 一方面将很多以前需要图片和 Flash 的效果转换为了浏览器自身的图形渲染；
- 一方面支持更多的选择器，让我们可以更轻松地定义样式。

1.3 CSS 3 的特性

应用 CSS 3，我们可以设置很多之前只能使用图片来实现的效果，而使用图片，一方面缺乏灵活性，一旦需要修改，就必须使用又大又慢的图片处理软件重新作图，再重新保存图片；另一方面使用图片增加了传输的数据量和 HTTP 请求数，拖慢了页面加载的速度。

CSS 3 还支持简单的动画，虽然复杂的动画还是必须通过 Flash、JavaScript 来实现，但实际开发中，大多数的动画并不复杂，这大大提高了我们的开发效率。本节就来简要介绍 CSS 3 支持的一些特性。

1.3.1 圆角

在交互设计中，相对于直角，圆角显得更为柔和，尤其应用在拟物图标上更容易被用户接受。苹果公司甚至专门为手机的圆角矩形的设计申请了专利。在 Web 开发中，使用 CSS 3 的 border-radius 属性，可以为一个 HTML 元素设置圆角。

图 1.1 是大名鼎鼎的 Gmail 邮箱，它的按钮和菜单都采用了圆角的设计。

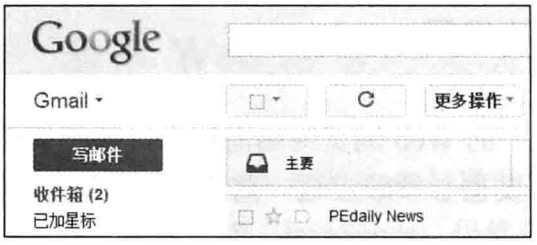


图 1.1 Gmail 邮箱

如果采用传统的方法制作，我们或许会用 PS 画一个圆角背景作为按钮的背景，或者用一个圆形小图片放在按钮的 4 个角上，而这带来的性能影响，对于像 Google、百度这样为海量用户提供基础服务的企业来说是难以接受的。

1.3.2 阴影

应用 CSS 3，可以为元素的边界（box-shadow）和文字（text-shadow）设置阴影。图 1.2 展示了一个网站的选择菜单，由于应用了阴影，显得页面凸出并有一定的立体感，这在交互上给人们一个可以向下点击的暗示。

图 1.2 阴影的应用

1.3.3 渐变

原本 CSS 的 background-image 属性只能指定图像文件作为背景，现在，渐变（gradients）是 CSS 3 为 background-image 属性新增的参数。渐变参数如下。

- linear-gradient：纵向渐变。
- radial-gradient：横向渐变。
- repeating-linear-gradient：重复的纵向渐变。
- repeating-radial-gradient：重复的横向渐变。

图 1.3 展示的是 Bootstrap 2 的按钮，可以发现，它是带有凸起效果的，让人一看就知道这里是可以被点击的，而这就需要用到纵向的背景颜色渐变。

按钮	描述
默认	带渐变的标准灰色按钮
主要	提供额外的视觉感，可在一系列的按钮中指出主要操作
信息	默认样式的替代样式
危险	表示这个动作危险或存在危险

图 1.3 有渐变效果的按钮

1.3.4 变换

在 CSS 3 中，可以使用变换（transform）属性来对元素进行位移、偏转、拉伸、旋转等操作。这让设计师们有了更大的想象空间，能设计出更吸引人的页面。同时 transform 属性还可以配合 JavaScript 使用，制作出炫目的动画效果。变换的基本效果如图 1.4 所示。

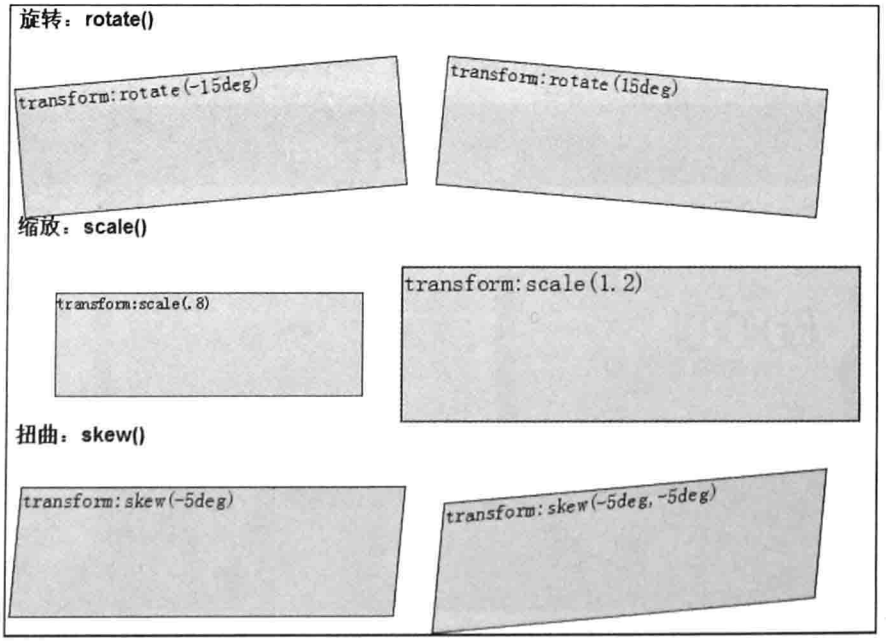


图 1.4 变换的效果

1.3.5 动画

使用 CSS 3 可以实现简单的动画，相比 Flash 和 JavaScript 制作的动画，CSS 3 显然不够强大，但是在构造简单的动画时，使用 CSS 3 大大减轻了开发者的负担，并且性能更佳。

由于纸介质无法让读者直观地感受动画效果，读者可以根据图 1.5，想象一下控制这个 CD 转动起来的效果。

提示：制作动画，需要使用 CSS 3 的@ keyframes 规则，详细内容可见本书第 7 章“变换和动画”。

图 1.5 转动的 CD

1.3.6 媒介查询

随着移动互联网的发展，传统的适应 PC 端的固定式网页已经无法满足人们的要求。针对各种不同尺寸的移动设备，CSS 3 推出了媒介查询（media query）功能，使我们可以根据浏览器窗口或设备尺寸来应用不同的 CSS 样式，从而实现对不同设备的适配。这样我们就不必自己来检测设备的尺寸，大大减少了开发的工作量。

随着开发者和设计者适配不同设备的经验积累和各种技术的增进，一套方法论被逐渐总结出来，我们称之为响应式设计。图 1.6 展示了一个典型的响应式设计案例。

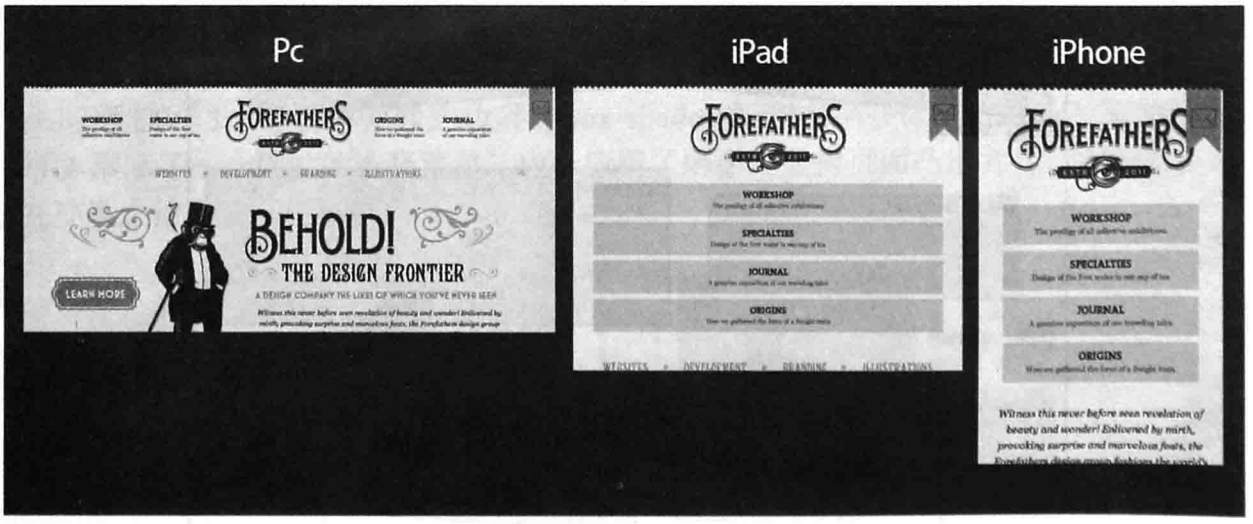


图 1.6 传说中的响应式设计

1.4 检测浏览器是否支持 CSS 3

虽然在理想情况下，CSS 3 对开发者和用户都更为友好，但最让人纠结的事情是，还有很多用户使用老版本的浏览器，或者使用系统自带的 IE，这种情况在中国尤为严重。很可能我们做了一个很华丽的页面，但是一些使用 IE 6 的用户看到的却是一片混乱的布局。

没有关系，在这种情况下可以先检测用户的浏览器对新特性的支持程度，根据支持程度再来执行不同的代码。比如，开发者可以为使用 IE 6 的用户呈现一个提示浏览器升级的画面，如图 1.7 所示。

图 1.7 提示浏览器升级的页面

1.4.1 使用 Modernizr 检测支持 CSS 3

Modernizr 是一个检测浏览器对 HTML 5 和 CSS 3 特性是否支持的 JavaScript 库，它是一个开源项目，托管在 Github，访问地址是 http://github.com/Modernizr/Modernizr。

Modernizr 的功能其实很简单，就是用 JavaScript 检测浏览器对 HTML5/CSS3 的特性支持情况。支持某个属性，就在页面的<html>标签上添加一个相应的 class，不支持某个属性就添加一个带 no-前缀的 class。比如，如果被检测的浏览器支持 video 标签，Modernizr 就会在<html>标签上添加 video 类，否则，添加 no-video 类。

Modernizr 除了添加相应的 class 到 HTML 元素以外，还提供一个全局的 Modernizr JavaScript 对象，该对象提供了不同的属性来表示当前浏览器是否支持某种新特性。例如，下面的代码可以用来判断浏览器是否支持 canvas 和 local storage：

```
$(document).ready(function () {
if (Modernizr.canvas) {
        //这里添加 canvas 代码
    }
if (Modernizr.localstorage) {
        //这里添加本地存储代码
    }
});
```

Modernizr 官方站点：http://modernizr.com。该网站提供了一个自定义工具来选择需要的探测功能，这样可以将下载的脚本最小化。

Modernizr 的用法很简单，仅仅需要在页面中引入库的.js 文件即可，如：

```
<script type="text/javascript" src="modernizr-1.5.js"></script>
```

1.4.2 支持 CSS 3 的浏览器

在页面的开发过程中，我们还需要根据目标人群有的放矢。比如，要做一个讨论 MAC 使用的论坛，目标人群就是 MAC 用户，那么只需要注意 Safari 和 Chrome 有哪些 CSS 3 效果不支持就可以了，无须再搞复杂的适配工作。

表 1.1 是主流浏览器对 CSS 3 特性的支持情况。

表 1.1 CSS 3 的浏览器支持情况

浏览器	Safari		Chrome	火狐	Opera	IE				
Version	5.1	6	25	15	12	6	7	8	9	10
RGBA	Y	Y	Y	Y	Y	N	N	N	Y	Y
HSLA	Y	Y	Y	Y	Y	N	N	N	Y	Y
Box Sizing	Y	Y	Y	Y	Y	N	N	Y	Y	Y
Background Size	Y	Y	Y	Y	Y	N	N	N	Y	Y
Multiple Backgrounds	Y	Y	Y	Y	Y	N	N	N	Y	Y
Border Image	Y	Y	Y	Y	Y	N	N	N	N	N
Border Radius	Y	Y	Y	Y	Y	N	N	N	Y	Y
Box Shadow	Y	Y	Y	Y	Y	N	N	N	Y	Y
Text Shadow	Y	Y	Y	Y	Y	N	N	N	N	Y
Opacity	Y	Y	Y	Y	Y	N	N	N	Y	Y
CSS Animations	Y	Y	Y	Y	Y	N	N	N	N	Y
CSS Columns	Y	Y	Y	Y	Y	N	N	N	N	Y
CSS Gradients	Y	Y	Y	Y	Y	N	N	N	N	Y
CSS Reflections	Y	Y	Y	N	N	N	N	N	N	N
CSS Transforms	Y	Y	Y	Y	Y	N	N	N	Y	Y
CSS Transforms 3D	Y	Y	N	Y	N	N	N	N	N	Y
CSS Transitions	Y	Y	Y	Y	Y	N	N	N	N	Y
CSS FontFace	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
FlexBox	Y	Y	Y	Y	N	N	N	N	N	N
Generated Content	Y	Y	Y	Y	Y	N	N	Y	Y	Y
DataURI	Y	Y	Y	Y	Y	N	N	Y	Y	Y
Pointer Events	Y	Y	Y	Y	N	N	N	N	N	N
Display: table	Y	Y	Y	Y	Y	N	N	Y	Y	Y
Overflow Scrolling	N	N	N	N	N	N	N	N	N	N
Media Queries	Y	Y	Y	Y	Y	N	N	N	Y	Y

1.5 用 CSS 3 实现的优秀网页

目前我们经常上的网站都有 CSS 3 的应用，不过相对来说，国内各大门户网站由于 IE 6 份额巨大，应用得相对较少，而像 Google、Twitter 等国际互联网巨头则已经在其主要应用上大量采用了 CSS 3 效果。图 1.8 是 Twitter 的一张截图，应用了大量圆角、渐变等 CSS 3 特性。

图 1.8 Twitter 网站截图

在开发中，基于 CSS 3 的各种开发框架、插件也是层出不穷，比如本书之后将要介绍的 Bootstrap、Foundation 等框架，以及各种针对单独功能的插件。图 1.9 是一个基于 CSS 3 的进度条插件。

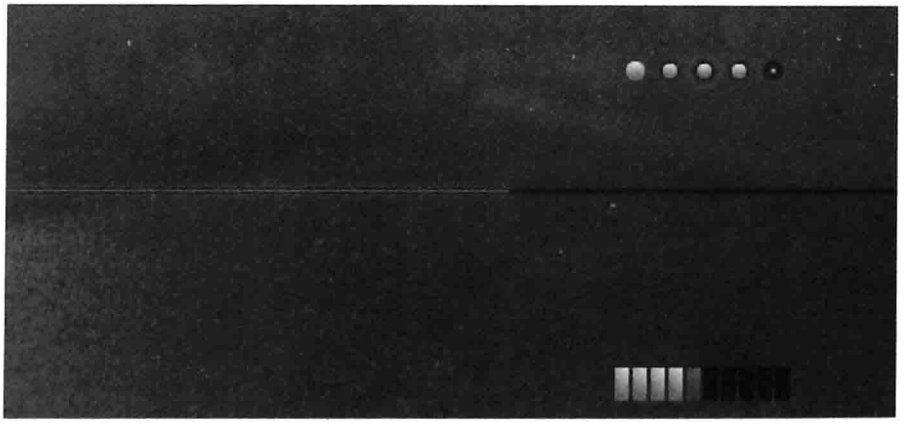


图 1.9 基于 CSS 3 的进度条插件

在国内，虽然限于 IE 6 和 IE 7 用户基数庞大，像新浪、网易等大型门户并不会在其主站上大量应用 CSS 3 效果，但是在后台管理系统上，Bootstrap 已经成为了一种“标准配置”。而对于一些追求显示效果的宣传网站、个性化商品展示页面，CSS 3 也得到了广泛的应用，如图 1.10 所示是一个设计公司的宣传网站，使用了 CSS 3 中的变换（transform）特效，构

建出了鼠标悬停后画面折叠的效果。

图 1.10　一个设计公司的宣传网站

1.6　小结

本章的内容是全书的基础，读者首先要了解 Web 标准发展的趋势，然后了解 CSS 3 都做了什么改动，以及这些改动给网页或者说网站设计带来的变化，最后如果想学习最新的 CSS 技术，那么必须了解如何检查浏览器对 CSS 新特性的支持，了解各大流行的浏览器对新特性的支持情况。

本章的介绍非常简单，主要是期望读者能够对后面的内容有一个整体上的认识，方便读者制定全书的学习计划。

第 2 章
温故知新——HTML、CSS 基础

新的网站制作技术虽然层出不穷，但是技术的发展讲究循序渐进和向上兼容，网站前端的基本技术框架与互联网大潮刚刚兴起时的框架并无不同。因此，在学习新的标准和特性时，对于一些基本的概念仍需要有清晰的理解，这才能帮助我们更好地消化新知识。

本章主要是对 HTML 和 CSS 基础中的一些重点和难点进行分析，包括以下几个要点：

- 盒子模型。
- CSS 的属性前缀。
- 通过 HTML 的条件注释编写 IE 兼容性代码。
- 元素的浮动，以及如何清除浮动。
- 元素的定位，包括绝对定位与相对定位，以及如何解决 fixed 的兼容问题。

2.1 W3C 标准、HTML 语言和 CSS

本节了解什么是 W3C、W3C 的工作流程，以及 HTML 和 CSS 在网页中的作用。

2.1.1 W3C 组织与 W3C 标准

W3C 是万维网联盟（World Wide Web Consortium）的英文简写，万维网联盟创建于 1994 年，是国际上最著名的标准化组织之一，主要致力于实现对 Web 技术的标准化。

为解决 Web 应用中不同平台、技术和开发者带来的不兼容问题，保障 Web 信息的顺利和完整流通，W3C 制定了一系列标准，并负责督促 Web 应用开发者和内容提供者遵循这些标准。标准的内容包括使用语言的规范、开发中使用的导则和解释引擎的行为等。W3C 也制定了包括 XML 和 CSS 等众多影响深远的标准规范，有效促进了 Web 技术的互相兼容，对互联网技术的发展和应用起到了基础性和根本性的支撑作用。

W3C 组织对 Web 标准的制定和审核是非常严格的，一般会经过如图 2.1 所示的 7 个流程。

1. 提交提案

W3C 成员向 W3C 组织投递自己的一个提案，但 W3C 有可能决定不接受这个提案。

2．Notes

如果某个 W3C 成员向 W3C 组织提交了一个提案，而且 W3C 没有拒绝这个提案，那么它就进入了 Note 阶段。Note 的内容由提出方进行编辑修改，W3C 是不管的。Note 发表的时候，表示 W3C 还没有开始进行和这个提案有关的任何工作。

3．成立工作组

Notes 被 W3C 认可后，W3C 会成立一个工作组。工作组包括 W3C 的成员和有兴趣的外界团队和个人。

4．工作草案

草案会在 W3C 的站点上公布，并邀请公共的评论和意见。工作草案一般不会作为参考资料，因为它还会经过大量的修改、更新，而且可能随时被废弃。

5．候选推荐

这个阶段是可选的，依据论题的复杂程度而定。

6．Proposed Recommendations

Proposed Recommendations 是工作组工作的最后一个阶段。它有被继续修改的可能，但在一般情况下，它会马上不做改动地成为新的 W3C 标准。

7．Recommendation

Proposed Recommendations 经过了 W3C 组织成员的检查和 W3C 主席的盖章后，成为 W3C Recommendation。它一般是一个稳定的规范，可以作为参考资料进行学习。

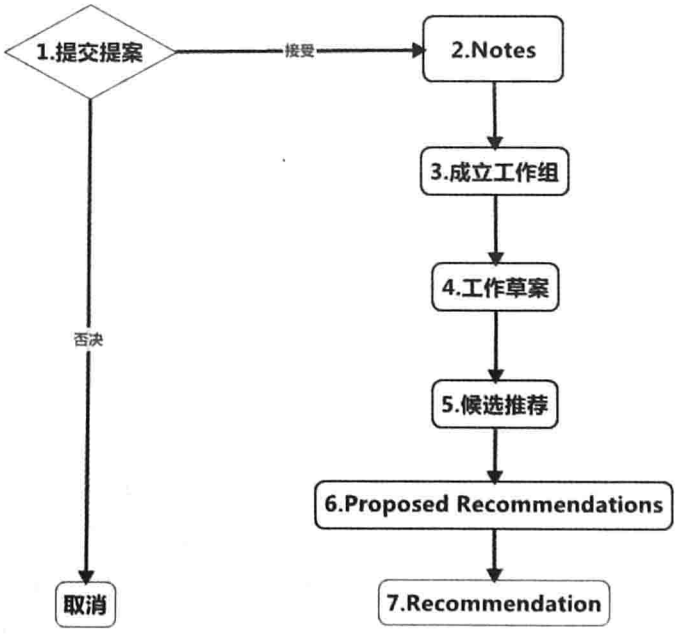


图 2.1　W3C 标准的制定流程

2.1.2　什么是 HTML

HTML 是 HyperText Markup Language 的缩写，汉语一般译作超文本标记语言。HTML

是为显示网页浏览器中的信息而设计的一种标记语言。

HTML 在 1982 年由蒂姆·伯纳斯-李创建，由 IETF 用简化的 SGML（标准通用标记语言）语法进行了扩展，后来成为国际标准，由万维网联盟（W3C）维护。

HTML 文档最常用的扩展名为.html，但是有些旧操作系统（如 DOS）限制扩展名最多为 3 个文字符号，所以.htm 扩展名也允许使用。

HTML 语言本质上和 Office Word 是一致的，都用于编辑文档，它采用不同的标签来表达不同的意义，如下面的例子：

```
<html>
  <body>
    <h1>CSS 高性能实战</h1>
    <h2>前　言</h2>
<p> Cascading Style Sheets，译作层叠样式表，简称为 CSS，是一种用于为 HTML 等结构化文档
    添加样式的标记语言。为了……</p>
  </body>
</html>
```

<h1>标签表示大标题，<h2>标签表示二级标题，<p>标签表示段落。多数完整的 HTML 标签由开头的声明和结尾的闭合组成，闭合标签需要在内部加上/符号，例如上例中的<h1>……</h1>。

注意：有一些 HTML 标签是空标签，只包含属性，没有闭合标签，如<img>标签、<input>标签等。

2.1.3　什么是 CSS

Cascading Style Sheets，译作层叠样式表，简称为 CSS，是一种为 HTML 等结构化文档添加样式的标记语言。目前流行的 CSS 3 标准从 1999 年就开始制定，直到 2011 年才最终发布为 W3C 推荐规范。当前主流浏览器都可以支持绝大部分 CSS 3 标准。最新的 CSS 4 标准从 2011 年开始设计，不过距离完善还有很长的距离，目前只有部分浏览器支持其中极少数的功能。

通俗一点解释，CSS 完成的工作和在 Office Word 里为文档修改样式是一样的，比如在 Word 中可以指定大标题的字号、字体、上下间距、对齐方式，对应到 CSS 中就是修改 HTML 中<h1>标签的样式，例如：

```
h1{
  font-size: 40px;          /*字体大小为 40 像素*/
  color: black;             /*颜色为黑色*/
  line-height:100px;        /*行高为 100 像素*/
  text-align:center;        /*居中对齐*/
}
```

2.1.4　HTML+CSS 之最佳拍档

HTML 用来显示网页的内容，CSS 用来设计网页的样式。那么怎样将 CSS 应用到 HTML

文档中呢？常见的有 3 种方法：使用 style 属性、使用<style>标签、使用<link>标签。下面详细介绍这 3 种方法并举例。

（1）在 HTML 标签内使用 style 属性为该标签指定样式。例如：

```
<html>
  ……
  <h1 style="color: red"> CSS 高性能实战</h1>
  ……
</html>
```

这段代码将大标题的颜色改为了红色。这种定义方式在样式代码较多时阅读和编写都比较困难，而且只能对当前标签生效。一般来说是不推荐在生产代码中使用这种方法的，平时只用于测试，以及用 JavaScript 为元素追加 CSS 属性时使用。

（2）在 HTML 文档内部用<style>……</style>标签包裹 CSS 代码。例如：

```
<html>
  <style>
h1{
  color:red;
}
  </style>
  <h1> CSS 高性能实战</h1>
  ……
</html>
```

这段代码仍然是将大标题的颜色改为红色，只不过将 CSS 定义的位置放在了<style>标签内集中管理，这样降低了 HTML 和 CSS 代码之间的耦合，方便管理。但是有一些 CSS 定义是可以同时应用于多个 HTML 文档的，如果采用这种方式，就必须在不同的 HTML 文档中定义相同的 CSS，因此一般来说，实际生产中也不推荐这种方式。

（3）使用<link>标签引用外部 CSS 样式表。例如：

```
<html>
  <head>
    <link href="http://libs.baidu.com/bootstrap/2.3.2/css/bootstrap.css" rel="stylesheet">
  </head>
  <body>
<h1> CSS 高性能实战</h1>
……
  </body>
</html>
```

使用<link>标签既可以引用本地的 CSS 文件，也可以引用远程的 CSS 文件。这样的话，CSS 样式文档只要编写一份，其他 HTML 文档需要它时就可以将它引入进来，这样既方便管理，又避免了重复劳动。一般在生产环境下，都采用这种方式将 CSS 和 HTML 进行结合。

2.2 不可不知的 CSS 盒子模型

大多数情况下，我们需要对商品进行包装才能出售，商品既需要精美的表层包装来吸

引顾客，也需要牢固的外壳来避免运输途中的碰撞和变质。

一个块级元素，包括内容、外边距、边框、内边距 4 个组成部分，当然，在不设定的情况下，内外边距和边框都是没有的。对于一个块级元素来说，设置内外边距和边框就像为商品套上包装盒一样，这就是“盒子模型”这一说法的由来。CSS 的盒子模型如图 2.2 所示。

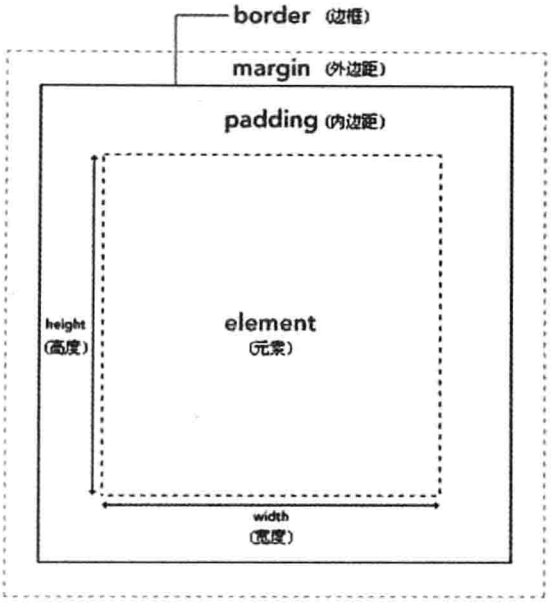


图 2.2　盒子模型

1．内容

元素框的最内部分是实际的内容，直接包围内容的是内边距（padding），它呈现了元素的背景。内边距的边缘是边框，边框以外是外边距（margin），外边距默认是透明的，因此不会遮挡其后的任何元素。

注意：背景应用于由内容和内边距、边框组成的区域。

2．内边距和边框

内边距和边框主要的作用是装饰。在内边距和内容区域，我们可以显示漂亮的背景，还可以控制边框的样式来装点内容。

3．外边距

外边距主要用于布局，目的是控制元素之间的距离。

2.3　跨浏览器的 CSS

由于 W3C 标准的发展，以及浏览器厂商出于商业利益的考量，老版本的浏览器（尤其是 IE 系列）存在着大量和 W3C 标准不一致的情况。一个在 Chrome 浏览器下十分美观的网站，在 IE 6 下很可能惨不忍睹，对此我们不得不进行大量的兼容性设计。

根据开发需求，跨浏览器设计有多种解决方案，如果仅仅是要求在老版本 IE 下能够正

常显示布局，可以采用渐进增强的设计，仅仅使用一套代码，在布局方面采用一些兼容性的措施，保证不出现布局错乱即可。

而如果要保证所有的浏览器显示效果一致的话，一方面不得不放弃一些很酷的选项，比如 HTML 5 中的 canvas、CSS 3 的 transform；另一方面不得不使用更多的图片，降低页面加载的性能；甚至我们可能必须根据用户浏览器的不同编写多套代码。

兼容性问题五花八门，本书只介绍通用的方法，那就是通过条件注释（conditional comment）让不同的浏览器来加载不同的 CSS，示例代码如下：

```
<!--[if !IE]><!--> 除 IE 外都可识别 <!--<![endif]-->
<!--[if IE]> 所有的 IE 可识别 <![endif]-->
<!--[if IE 6]> 仅 IE 6 可识别 <![endif]-->
<!--[if lt IE 6]> IE 6 以及 IE 6 以下版本可识别 <![endif]-->
<!--[if gte IE 6]> IE 6 以及 IE 6 以上版本可识别 <![endif]-->
```

条件注释的控制符见表 2.1。

表 2.1　条件注释的控制符

项目	范例	说明
!	[if !IE]	“非”运算符
lt	[if lt IE 5.5]	小于运算符
lte	[if lte IE 6]	小于等于运算符
gt	[if gt IE 6]	大于运算符
gte	[if gte IE 6]	大于等于运算符
()	[if (lte IE 6)]	用于子表达式，以配合布尔运算符
&	[if (lte IE 9)&(gt IE 6)]	AND运算符
|	[if (gt IE 6)|(!IE)]	OR运算符

2.4　理解浏览器的属性前缀

当一个新的 CSS 属性被开发出来后，由于 W3C 标准的申请和审核流程十分严格和漫长，浏览器厂商往往会暂时绕开这一流程，通过添加前缀的方式让自己的浏览器率先支持新的属性。本节就来介绍这些前缀。

2.4.1　常用的属性前缀

以下是开发中经常会用到的前缀。

- -webkit：　webkit 核心浏览器，包括 Chrome、Safari 等。
- -moz：火狐（Firefox）浏览器。
- -ms：　IE 浏览器。
- -o：　Opera 浏览器。

在实际开发过程中，对于大多数 CSS 3 效果来说，考虑到兼容性，往往需要把所有的前缀属性都写上去，譬如这样：

```
.transform{
  -webkit-transform:rotate(-3deg);        /*带有 Chrome、Safari 浏览器的属性前缀*/
  -moz-transform:rotate(-3deg);           /*带有 Firefox 浏览器的属性前缀*/
  -ms-transform:rotate(-3deg);            /*带有 IE 浏览器的属性前缀*/
  -o-transform:rotate(-3deg);             /*带有 Opera 浏览器的属性前缀*/
  transform:rotate(-3deg);                /*W3C 标准语法，无属性前缀*/
}
```

2.4.2　属性前缀的排序

即使 W3C 标准得到了一致通过和广泛推广，但浏览器厂商为了兼容老的内容，还是不得不继续支持带有前缀的属性；而开发者面对一些使用老版本浏览器的用户时，也不得不继续给代码写上所有的属性前缀。

但是问题随之产生：

W3C 标准属性在某些情况下与带有前缀的属性具有不同的表现形式，那有什么解决方案呢？

这一方面需要依赖开发者的知识和经验，一方面也可以采取通用的办法，就是把标准属性放在最后书写。例如：

```
.border_button {
    -webkit-border-radius: 5px;
    -moz-border-radius: 5px;
    border-radius: 5px;                   //标准属性
}
```

这样即使出现不一致的情况，后书写的符合 W3C 标准的属性，会覆盖前面带有属性前缀的定义，更好地保证显示效果在所有浏览器下的一致性。

2.5　揭开浮动布局的秘密

正常情况下，页面中的块级元素（block）就好像一个个沉在水中的铁块，如果我们将铁块换成木块呢？显然它们会飘起来，浮在水面上，如图 2.3 所示，右边的部分是给 div1~div3 元素设置了 float:left 后的结果。

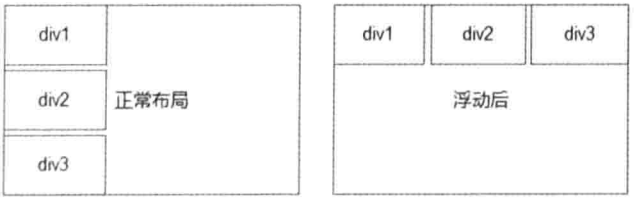


图 2.3　浮动效果对比

2.5.1　浮动导致的布局变动

这里使用浮动（float）这个词语实在是非常形象。当然，这里的浮动和现实中的浮动

并非完全吻合，下面来学习一下 CSS 中的 float 属性。

float 属性有 4 个可选项：none、left、right、inherit。其中 none 为默认值，即不浮动，inherit 表示继承父元素的 float 值。而 left、right 则很好理解，一个是向页面的左侧浮动，一个是向页面的右侧浮动。我们重点需要说明的是设置了浮动后元素的变化情况。

注意：一般不建议使用 inherit，因为 IE 不支持这个选项。

（1）对于块级元素来说，在不设置宽度的情况下，默认的宽度是 100%，一旦设置了浮动，它的宽度就会根据内容进行自动调整，如图 2.4 所示。

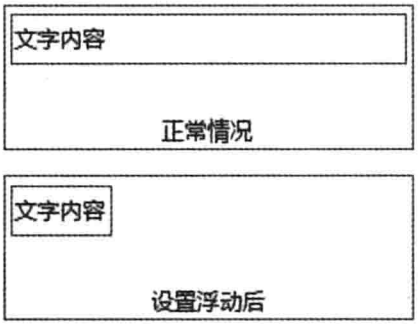


图 2.4　默认宽度下的浮动

图 2.4 的代码如下：

```
<div>文字内容</div>                          <!--未设置浮动-->
<div style="float: left;">文字内容</div>    <!--设置左浮动-->
```

（2）设置了浮动的元素会脱离正常的文档流，我们可以这样理解：设置浮动后，元素不仅在 y 轴上浮了起来，在 z 轴上，也浮了起来。譬如：默认情况下，父元素的高度会根据子元素的内容自动进行调整，而如果我们将子元素设置为浮动，父元素的高度就会变为 0，如图 2.5 所示。

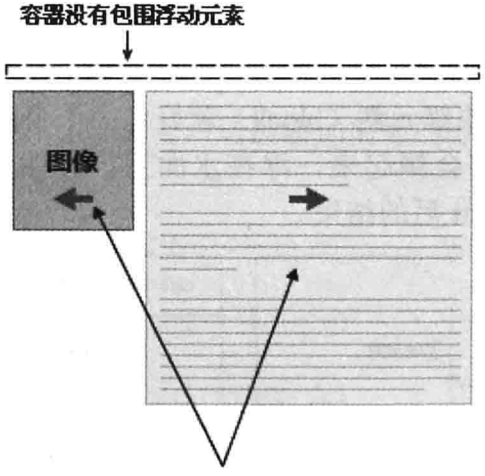


图 2.5　脱离了文档流的浮动

图 2.5 的代码如下：

```
<style>
  .left, .right{
    float: left;
```

```
  }
  .left{
    width:30%;
  }
  .right{
    width:70%;
  }
</style>

<div class="container">
  <div class="left">
    <img src="test.jpg">
  </div>
  <div class="right">
  <p>
  ......//此处省略文字
  </p>
  </div>
</div>
//通过浏览器工具可以发现 container 的高度为 0
```

（3）虽然浮动的元素脱离了文档流，但是里面的内容仍然占据空间，会根据相对位置进行布局，如图 2.6 所示。

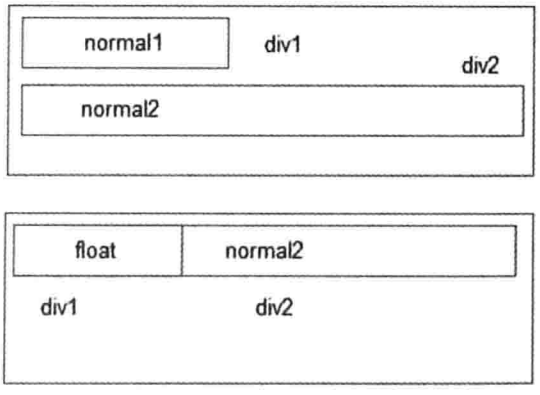


图 2.6　文档内容不会因为浮动形成覆盖

如果将 div1 设置为 float:left，由于 float 元素脱离了文档流，div2 自动向上补一位，但是不同于我们的下意识反应：文字 normal2 并没有被 div1 中的文字覆盖（虽然 div1 是覆盖在 div2 上方的），而是排列在了正常的相对位置上。

图 2.6 的代码如下：

```
//没有设置浮动时的代码
<style>
  .div1, .div2{
    height:20px;
  }
  .div1{
    width:300px;
  }
</style>
<div class="div1">normal1<div>
<div class="div2">normal2<div>
```

```
//为 div1 设置 float:left 后的代码
<style>
  .div1, .div2{
    height:20px;
  }
  .div1{
width:300px;
float: left;
  }
</style>
<div class="div1">float<div>
<div class="div2">normal2<div>
```

2.5.2 清除浮动

2.5.1 讲述了应用浮动后布局出现的种种改变，有时我们需要用到浮动，但又不想由于浮动的某些特性影响布局，这时就需要清除浮动。

清除浮动主要应用的是 CSS 中的 clear 属性，clear 属性定义了元素的哪一侧不允许出现浮动元素。可选项有 left、 right、both。例如：

```
img { float:left; clear:both;}          /*左右两侧都不允许出现浮动元素*/
```

下面介绍两种应用比较广泛的清除浮动的方法。

（1）在需要的地方添加定义了 clear:both 的空标签。

```
html body div.clear,
html body span.clear
{
  background: none;
  border: 0;
  clear: both;                    /*这句是重点，其他都是兼容性代码 */
  display: block;
  float: none;
  font-size: 0;
  margin: 0;
  padding: 0;
  overflow: hidden;
  visibility: hidden;
  width: 0;
  height: 0;
}
/*在需要清除浮动的元素后面添加<div class="clear"></div>即可/
```

这是一个“万能”的清除浮动代码，可以在不同浏览器下兼容。

（2）对父元素使用:after 伪类。

```
.clearfix:after {
  content: "020";
```

```
    display: block;
    height: 0;
    clear: both;
}
.clearfix {
    zoom: 1;
}
.left{
    float:left
}
.right{
    float:right
}

<div class="div1 clearfix">
    <div class="left">Left</div>
    <div class="right">Right</div>
</div>
/*使用调试工具可以看到，父元素的高度不再为 0 了*/
```

注意：由于 IE 6 和 IE 7 不支持:after 伪类，因此需要添加 zoom: 1 兼容代码。

2.6　看穿 CSS 的定位技术

定位是 CSS 基础学习中的一个重点，也是一个难点。CSS 使用 top、left、right、bottom 设置元素的二维（x 轴和 y 轴）偏移量，使用 z-index 设置元素垂直于屏幕的方向，也就是“z 轴”的偏移量。

CSS 使用 position 选项来定义元素的定位属性，该选项有 5 个可选值：static、relative、absolute、fixed、inherit，默认值为 static。inherit 属性表示继承父元素的定位属性，因此，实际上我们只需要掌握 static、relative、absolute、fixed 这 4 种定位属性的特性即可。

2.6.1　相对定位技术与实战

相对定位就是指相对于文档流中的其他已定义的元素位置进行定位，如图 2.7 所示。

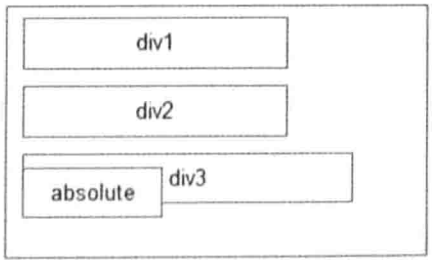


图 2.7　相对定位

<div2>的位置是根据已存在的<div1>的位置向下顺延的，而<div3>的位置则根据<div2>的位置决定，和绝对定位的、已经脱离文档流的 absolute 元素的位置无关。

注意：那些脱离文档流的元素，比如设置了浮动或者绝对定位的元素不会对相对定位产生影响。

relative 和 static 都是相对于文档其他元素进行定位，都属于相对定位的范畴，区别只在于一个可以控制位移，一个不能。

1．static（默认值）

如果使用默认值，在 CSS 中为元素定义 top、left、right、bottom、z-index 都不会生效。换句话说，如果想设置元素的偏移量和 z-index，必须为元素定义 position 属性（static 除外）。

2．relative

relative 的表现和默认值一样，只不过可以通过设置偏移量和 z-index 来控制相对于其正常位置进行的偏移。

说明：所有元素的定位（position）都默认为 static，什么都不写就是相对定位，而使用 position: relative 在不设置 top/left/z-index 等值的情况下和默认值表现是一样的。

2.6.2 绝对定位技术与实战

绝对定位的元素有以下几个特点：

- 块级元素的宽度在未定义时不再为 100%，而是根据内容自动调整。
- 在不定义 z-index 的情况下，absolute 元素会覆盖在其他元素之上。
- 它会脱离正常的文档流，不再占据空间，类似于浮动后的效果。

absolute 和 fixed 都属于绝对定位的范畴，都遵循以上 3 个特点。

1．absolute

absolute 是相对上一个不为 static 的父元素进行绝对定位。换句话说，如果不指定父元素的 position，absolute 将相对于整个 html 文档进行绝对定位，如图 2.8 所示。

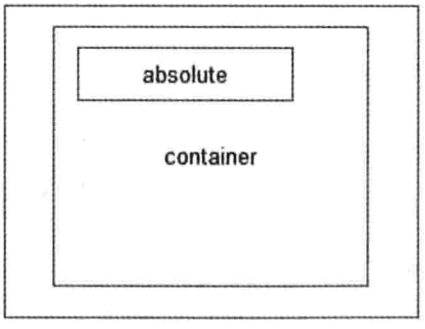


图 2.8　绝对定位

图 2.8 的代码如下：

```
<style>
  .container{
```

```
    margin: auto;
    width: 960px;
    height:300px;
    position: relative;  /*指定父元素的 position*/
  }

  .absolute{
    position: absolute;
    top: 0;
    left: 0;
  }
</style>
<body>
  <div class="container">
   <div class="absolute">绝对定位</div>
  </div>
</body>
```

只有指定了 container 的 position（非 static），子元素才能相对父元素进行绝对定位，否则将相对 html 进行绝对定位，这段代码中，.absolute 元素将出现在.container 容器部分的左上角，如果不指定 container 的 position，.absolute 元素将出现在浏览器页面的左上角。

如果想让 absolute 元素相对 container 进行绝对定位，我们可以为 container 定义 position:relative。

2．fixed

生成绝对定位的元素，相对于浏览器窗口进行定位。也就是说，不论网页如何滚动，该元素始终停留在屏幕的某个位置上。比如我们希望侧边控制栏始终对用户可见，就可以使用 positon: fixed 来进行定位。

IE 6 和 IE 7 不支持 fixed 属性，不过可以通过兼容性方案，用 absolute 来模拟 fixed 效果，代码如下：

```
/* 相当于正常的 position:fixed; top:0;  */
.sl-fixed-top {
    bottom:auto;
    top:0;
    _bottom:auto;
    _top:expression(eval(document.documentElement.scrollTop));
}

/* 相当于正常的 position:fixed;bottom:0px; */
.sl-fixed-bottom {
    bottom:0;
    top:auto;
    _bottom:auto;
    _top:expression(eval(document.documentElement.scrollTop+document.documentElement.cli
```

```
    entHeight-this.offsetHeight-(parseInt(this.currentStyle.marginTop,10)||0)-(parseInt(this.current
    Style.marginBottom,10)||0)));
}

/* 相当于正常的 position:fixed;left:0px; */
.sl-fixed-left {
    left:0;
    _position:absolute;
    right:auto;
    _left:expression(eval(document.documentElement.scrollLeft));
}
/* 相当于正常的 position:fixed;right:0; */
.sl-fixed-right {
    right:0;
    left:auto;
    _right:auto;
    _left:expression(eval(document.documentElement.scrollLeft+document.documentElement.cli
    entWidth-this.offsetWidth)-(parseInt(this.currentStyle.marginLeft,10)||0)-(parseInt(this.current
    Style.marginRight,10)||0));
}
/* Hack for IE 6 */
.sl-fixed-top,.sl-fixed-right,.sl-fixed-bottom,.sl-fixed-left {
    _position:absolute;
}
```

2.7 小结

本章主要对 HTML 和 CSS 基础中的一些重点和难点进行了分析。在学习网页设计与网站搭建之前，必须先对这些基础知识有所了解，这就类似于给房屋打地基，地基没打好，则后面学习的难度会越来越大。本章基本涵盖了 HTML/CSS 基础中最容易碰到问题的技术点，如盒子模型、属性前缀、浮动、绝对定位和相对定位等，一些对基础概念有些模糊的读者，务必在学习新特性和框架实战前阅读本章，打好坚实的基础。尤其是浮动和定位，它们是 CSS 布局当中很重要的两种方法，不管是 CSS 2 还是 CSS 3，只要有 CSS 的地方，它们的地位就无法撼动。

第 3 章
使用 CSS 选择器让样式表更健壮

本章主要介绍 CSS 选择器的知识，选择器可以帮助我们选择特定的某个 HTML 元素，或有共同规律的某组 HTML 元素，从而可以为选中的元素添加样式。合理地使用选择器可以使项目拥有更好的可维护性。

本章主要知识点：

- CSS 的基础选择器。
- CSS 的伪类选择器。
- 选择器的应用案例。

3.1 基础选择器

本节主要介绍最常用的一些选择器，这些选择器在任何浏览器下都支持，是 CSS 学习中必须掌握的基础内容。

3.1.1 标签选择器

标签选择器是最简单的选择器，它的命名只要和对应的 HTML 标签相同即可，例如：

```
h1{
  font-size:30px;
  color:#333;
}
```

这里的 h1 就是标签选择器，在实际项目中，标签选择器一般用于定义全局样式。和 Office 中的 Word 类似，在全局定义中预定义好正文和标题的样式、段落之间的间距、图片的最大宽度和对齐方式等。全局定义只需要定义一次，就可以在本文档的任何地方使用，如果修改，也只需要修改一次。合理地使用标签选择器定义全局样式，可以大大减轻开发和维护的工作量。

下面的示例是节选 Bootstrap 框架中关于标题的全局定义，读者可以研习一下标签选择器在实际项目中的应用。

```
h1,h2,h3,h4,h5,h6 {
  margin: 10px 0;
  font-family: inherit;
  font-weight: bold;
  line-height: 20px;
  color: inherit;
  text-rendering: optimizelegibility;
}

h1,h2,h3 {
  line-height: 40px;
}

h1 {font-size: 38.5px;}
h2 {font-size: 31.5px;}
h3 {font-size: 24.5px;}
h4 {font-size: 17.5px;}
h5 {font-size: 14px;}
h6 {font-size: 11.9px;}
```

3.1.2 类选择器

类选择器也称为 class 选择器，它的语法非常简单，在 class 名称前面加上一个“.”符号。例如：

```
<div class="red content"></div>
.red{
  background:red;
}
.content{
  height:100px;
  width:100%;
}
```

由于一个 HTML 标签可以定义多个 class 属性，因此在实际应用中，类选择器成为了最灵活、应用最广泛的选择器。基本上任何样式定义都可以通过为元素追加 class 属性，然后定义该 class 的样式来完成，可谓是“万金油”方法。

注意：从代码的可读性、可维护性角度来讲，不要滥用类选择器，尽量不要为一个标签添加多于两个的 class 属性。尽量为 class 指定有意义的命名，避免 x1、x2、y1、y2 这样的无意义命名。如果项目比较庞大，可以考虑在命名时进行单词的组合，比如 control-group、control-user 这样的命名。

下面仍然节选一段 Bootstrap 中的代码供读者参考：

```
.icon-heart {
  background-position: -96px 0;
```

```
}
.icon-star {
    background-position: -120px 0;
}
.icon-star-empty {
    background-position: -144px 0;
}
.icon-user {
    background-position: -168px 0;
}
.icon-film {
    background-position: -192px 0;
}
```

注意看这里的命名规则，icon 开头表示这个 class 是在描述一个 icon（小图标），而后缀则表示这个 icon 的意义，比如 icon-heart 一看就知道是一个心形图标的意思，这样即使不看实际效果，也能知道这段 CSS 的作用，大大提高了代码的可读性。

3.1.3　id 选择器

id 选择器的语法是一个“#”号加上 id 的名称，例如：

```
<div id = "user_123"></div>

#user_123{
    width:120px;
    line-height:30px;
    height:30px;
}
```

一个 HTML 元素只能对应一个 id，所以 id 选择器在灵活性上不如 class 选择器，因此在实战中很少会直接在 CSS 文件中为 id 定义样式。

id 选择器在实战中一般有两个用途：

- id 选择器拥有最高的权重，因此可以用于覆盖之前的一些定义。
- 和后台数据对应，从而配合 JavaScript 进行一些逻辑操作。

3.1.4　通配符选择器

通配符的意思就是用一个符号来代替某些字符，例如在 Word 中要搜索以 com 开头的所有单词，可以用“com*”来做搜索关键字，这个*表示任意字符，这个时候可能就会搜到 computer、compact、combo 等以 com 开头的单词。

CSS 从 CSS 2 时代开始就引入了一种简单选择器——通配符选择器（universal selector），它以星号（*）开始，该选择器可以与任意元素匹配。例如，下面的规则可以使文档中的每个元素都显示为红色：

```
* {color:red;}
```

这个声明等价于列出了文档中所有元素的一个分组选择器。

通配符选择器在实际开发中可用于定义全局样式，不过使用标签选择器也能获得类似效果，例如：

```
body{color:red}
html{color:red}
```

注意：通配符选择器的权重是最低的，因此只要有其他的定义，使用通配符选择器进行的定义就会被覆盖。

3.1.5 子元素选择器

子元素选择器用于表示某些特定 HTML 嵌套关系时的样式展现，其语法关键词是一个“>” 符号。例如：

```
//HTML 代码：
<li><a href='#'>www.baidu.com</a></li>
<li><div><a href='#'>www.baidu.com</a></div></li>
//CSS 代码：
li > a{
  color:blue;
}
```

“>”左边是父元素，右边是子元素。上面的代码就表示第一个列表项（<li>）中的链接（<a>）为蓝色，而其他地方的链接则不受影响。

注意：如果两个元素不是严格的“父子关系”，则使用子元素选择器的定义不会生效。例如上面 HTML 代码中的第 2 个列表项的文字就不会被设置为蓝色。

3.1.6 后代元素选择器

后代元素选择器类似于子元素选择器，只不过它的要求不那么严格。它的语法关键词是一个空格，例如：

```
//HTML 代码：
<li><a href='#'>www.baidu.com</a></li>
<li><div><a href='#'>www.baidu.com</a></div></li>
//CSS 代码：
li a{
  color:blue;
}
```

只要链接（<a>）标签是列表项（<li>）的后代元素即可，如上面的 HTML 代码，两个链接的文字都会被设置为蓝色。

注意：读者一定要分清楚后代元素选择器和子元素选择器的区别，后代包括子辈、孙子辈、曾孙子辈等，而子元素只包括子辈。

3.1.7　相邻元素选择器

相邻元素选择器用于选取和某个元素相邻的同级元素，其语法关键词是一个“+”符号，例如：

```
//HTML 代码
<div class="content">
  <h1>测试</h1>
  <p>测试内容</p>
</div>
//CSS 代码：
h1+p{
  font-size: 15px;
}
```

上面的 CSS 代码定义了和 h1 标题（<h1>)相邻的段落（<p>）的样式。

相邻元素选择器的使用有两个条件：

- 二者必须拥有同一个父元素。
- 二者相邻。

相邻元素选择器在实际应用中往往会和其他选择器配合使用，例如：

```
body > .content   h1+p{
  font-size: 15px;
  font-weight: bold;
}
```

这段代码表示 body 的子元素.content 中如果存在后代元素 h1，则最终定义的是和 h1 元素相邻的 p 元素的样式。

一个实际的例子就是标题后第一段的文章导语经常会加粗显示，这里使用相邻元素选择器就非常的合适，如图 3.1 所示。

图 3.1　使用相邻元素选择器的效果

3.1.8 属性选择器

HTML 元素中除了 id、class 等通用的属性以外，有些标签还可以添加其他的属性，比如 title、href、name 等。在 CSS 选择器中，开发者也可以通过判断某些属性是否存在或者通过属性的值来选取 HTML 元素，这时就需要用到属性选择器。属性选择器的语法关键词是一对中括号“[]”，例如：

```
[title] {
  color:red;
}          /*所有拥有 title 属性的元素的文字颜色设为红色*/

a[href][title] {
  color:red;
}          /*同时拥有 href 和 title 属性的 a 标签的文字颜色设为红色*/
```

由第 2 段代码可以发现，属性选择器可以进行链式调用，从而缩小选择范围。

上面这个例子是根据属性是否存在来进行选择，只需要在[]中填入属性名即可。我们还可以通过为属性赋值来选取拥有特定属性值的元素，例如：

```
a[href="http://www.baidu.com"][title="百度"] {
  color: red;
}
```

这样只有 href=http://www.baidu.com 且 title="百度"的链接（<a>）文字才会被设置为红色。

在应用属性选择器时还可以使用通配符来进行模糊匹配，例如：

```
a[src^="https"]  /*选择其 src 属性值以"https"开头的每个<a>元素。*/
a[src$=".pdf"]   /*选择其 src 属性以".pdf"结尾的所有<a>元素。*/
a[src*="abc"]    /*选择其 src 属性中包含"abc"子串的每个<a>元素。*/
```

注意：使用通配符的属性选择器是 CSS 3 新加入的特性，IE 9 以前的浏览器无法兼容。

3.1.9 组选择器

如果要对多个元素定义同样的样式，则可以用组选择器来缩减重复代码。组选择器的语法关键字是一个“,”（英文的逗号），例如：

```
h1, h2, h3,h4,h5,h6{
  font-wight:bold
}
```

这段代码表示从 h1~h6 都采用加粗字体。使用组选择器就可以避免一个一个地定义相同或局部相同的属性，从而使得 CSS 样式表更为整洁易读。

下面节选一段 Bootstrap 中应用组选择器的实例：

```
button,
input,
select,
```

```
textarea {
    margin: 0;
    font-size: 100%;
    vertical-align: middle;
}

button,
input {
    *overflow: visible;
    line-height: normal;
}

button::-moz-focus-inner,
input::-moz-focus-inner {
    padding: 0;
    border: 0;
}

button,
html input[type="button"],
input[type="reset"],
input[type="submit"] {
    cursor: pointer;
    -webkit-appearance: button;
}
```

通过阅读以上代码读者可以发现，组选择器可以跟其他选择器（如这个例子中出现的属性选择器）共同使用。

3.1.10　复合选择器

如果说组选择器相当于一种并集，或者常说的“或”（||）关系的话，那么复合选择器就表示“与”（&）的关系。它的用法很简单，将两个有可能发生“与”关系的选择器连在一起就行了，例如：

```
p.test{                    /*注意中间不要有空格，否则就会被识别成后代选择器了*/
    color:red;
}

<p class="test">hehe</p>
<div>hehe</div>
<div class="test">hehe</div>
<p>hehe</p>
```

只有第 1 个段落（<p>）中的文字会被设置为红色，因为它同时满足了 p 元素和 class="test"两个条件。

注意：应用复合选择器时，标签选择器一定要写在最前面，否则无法识别。

3.2 伪类选择器

本节主要介绍 CSS 中的伪类选择器，谈到伪类选择器就必须明白什么是伪类？伪类表示元素的状态：排序、鼠标是否悬停、是否已被访问过、光标是否指向等。

使用伪类选择器，就可以得到诸如有鼠标悬停的元素、父元素下的第 n 个子元素、已被访问过的链接等使用基本选择器无法进行区分的元素。

CSS 2 中只有:hover、:active、:visited、:link :first-child :lang :link 等有限的几种伪类选择器，CSS 3 增添了大量新的伪类选择器，帮助开发者更为灵活地选取元素。

3.2.1 结构化伪类

结构化伪类这个词乍一看不知道是什么意思，其实就是可以根据文档的结构来选取元素。在 CSS 3 出现前，只有一个:first-child 可以使用，CSS 3 对结构化伪类进行了极大的丰富，让开发者可以根据元素在文档中的结构索引来进行多样的选择。

笔者首先给出一个基本的样例，本节后面的所有介绍都基于此样例。样例代码如下：

```
<style>
ul > li {
   display: inline-block;
   height:24px;
   line-height: 24px;
   width:24px;
   font-size: 15px;
   text-align: center;
   background-color: rgb(226, 129, 129);
   border-radius: 4px;
   margin:5px;
}
</style>
<ul class="test">
   <li>1</li>
   <li>2</li>
   <li>3</li>
   <li>4</li>
   <li>5</li>
   <li>6</li>
   <li>7</li>
   <li>8</li>
   <li>9</li>
   <li>10</li>
```

```
</ul>
<div>
  <ul class="test_one">
    <li>1</li>
    <li>2</li>
    <li>3</li>
    <li>4</li>
    <li>5</li>
    <li>6</li>
    <li>7</li>
    <li>8</li>
    <li>9</li>
    <li>10</li>
  </ul>
</div>
```

这个例子的显示效果如图 3.2 所示。

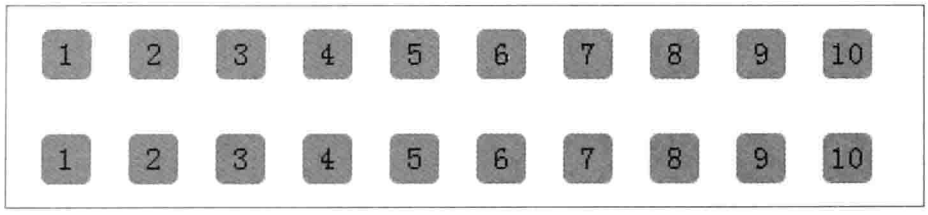


图 3.2　基础样式效果

1．:nth-child(*n*)

“:nth-child(*n*)”选择器中的 *n* 表示一个简单的表达式，它可以是大于等于 0 的整数，比如在基础样例中应用：

```
li:nth-child(2){
  background-color:#333;
  color:white;
}
```

n 取 2，就是取某个父元素内第 2 个<li>元素，即需要同时满足两个条件：

- 是不是第 2 个。
- 是不是<li>元素。

如果两个都满足则生效，应用上述代码后的效果如图 3.3 所示。

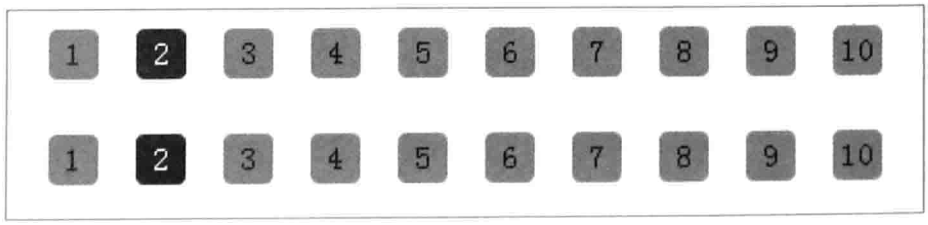


图 3.3　“:nth-child(*n*)”选择器的使用效果

这里可以看到，尽管图案中第 1 行和第 2 行的代码结构不同，但是都符合“某个父元

素中第 2 个<li>元素”的条件，于是样式都发生了相应地改变。

这里的 *n* 不仅仅能指定某个特定值，还可以进行相应的计算，譬如:nth-child(n)，这种用法相当于全选，例如：

```
li:nth-child(n){
  background-color:#333;
  color:white;
}
```

注意：这里的变量只能用字母 *n* 来表示，其他诸如 *x*、*m* 这些是不行的。

代码的效果如图 3.4 所示。

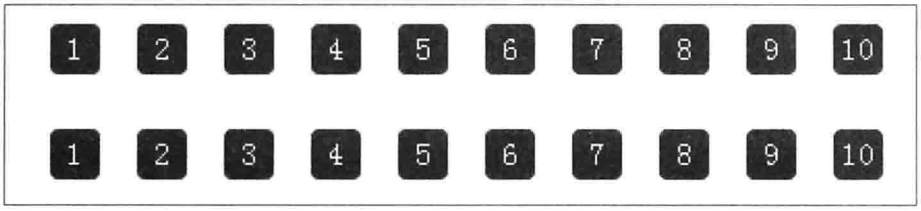


图 3.4　:nth-child(*n*)的效果

:nth-child(2*n*)则表示所有的偶数项，例如：

```
li:nth-child(2n){
  background-color:#333;
  color:white;
}
```

代码效果如图 3.5 所示。

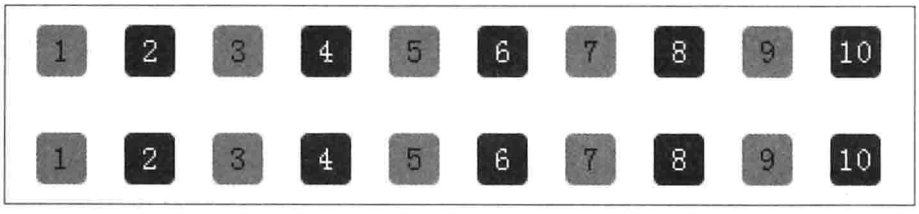


图 3.5　:nth-child(2*n*)的效果

如果这里取 3*n* 的话则会选取 3、6、9 项，如果取 2*n*+1 则会选取所有的奇数项，依次类推即可。

:nth-child(*n*+5)这个选择器是选择从第 5 个元素开始进行全选（这个数字 5 可以自己定义），如：

```
li:nth-child(n+5){
  background-color:#333;
  color:white;
}
```

代码效果如图 3.6 所示。

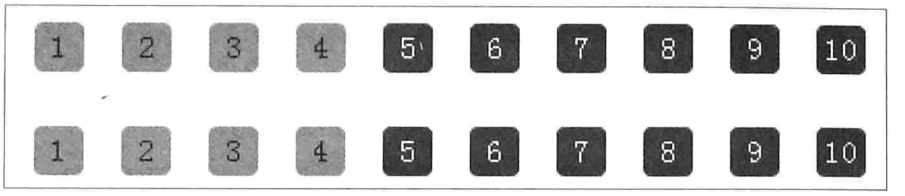


图 3.6　:nth-child(*n*+5)的效果

说明：IE 6~8 和 FF 3-浏览器不支持“:nth-child”选择器。

2．:nth-last-child(*n*)

“:nth-last-child(*n*)”选择器和前面的“:nth-child(*n*)”很相似，只是这里多了一个 last，所以它起的作用就和“:nth-child”选择器获取元素的顺序正好相反，是从最后一个元素开始计算。这里不再举例，读者可以把前面的几个:nth-child(*n*)代码换成:nth-last-child(*n*)。

3．:nth-of-type(*n*)

“:nth-of-type(*n*)”选择器和前面介绍的“:nth-child(*n*)”类似，区别在于，如果使用 p:nth-child(3)这样的条件时，一旦第 3 个元素不为<p>元素，这个选择器就不起作用，而 p:nth-of-type(*n*)则查询的是第 3 个<p>元素

如果把两个列表中的第 3 个元素的<li>都换成 ，对第 1 个列表使用 li:nth-of-type(3)，第 2 个列表使用 li:nth-child(3)：

```
//第 1 个列表
li:nth-of-type(3){
   background:#333;
   color: white;
}
//第 2 个列表
li:nth-child(3){
   background-color:#333;
   color:white;
}
```

最终显示效果如图 3.7 所示。

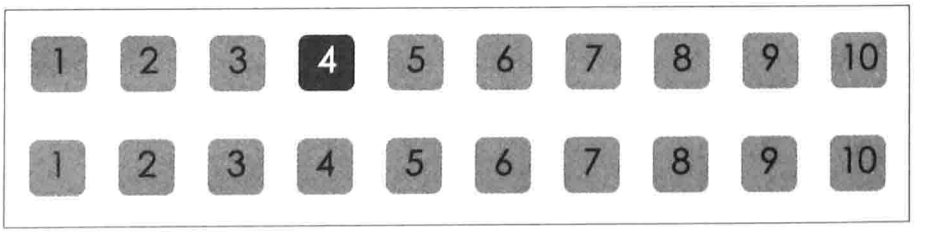


图 3.7　:nth-of-type(3)和 nth-child(3)的对比效果

如果不加标签类型，在使用:nth-of-type(*n*)时就会自动选择所有并列元素的第 *n* 个，如下例：

```
:nth-of-type(1){
   background:red;
}
```

```
<body>                          /*第一层的第一个元素*/
<div>                           /*第二层的第一个元素*/
    <ul class="test_one">       /*第三层的第一个元素*/
      <li>1</li>                /*第四层的第一个元素*/
      <li>2</li>
    </ul>
  </div>
</body>
```

每一层的第 1 个元素的背景都被设为红色，看到的效果就是整个背景变成红色了。如果将 1 改为 2，那么只有
2
的背景变为红色。

4．:nth-last-of-type(*n*)

“:nth-last-of-type(*n*)”选择器和前面的“:nth-of-type(*n*)”的区别只是获取元素的顺序相反，是从最后一个元素开始计算，例如：

```
li:nth-last-of-type(3){
  background:#333;
  color: white;
}
```

最终显示效果如图 3.8 所示。

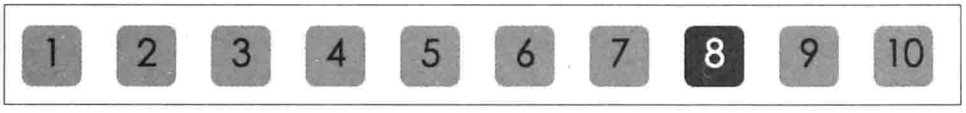


图 3.8　:nth-last-of-type(3)的效果

5．:last-child

“:last-child”选择的是元素的最后一个子元素。比如说，我们需要单独给列表最后一项设置不同的样式，就可以使用这个选择器。

注意：:last-child 是 CSS 3 新增的伪类选择器，而与之对应的:first-child 则是在 CSS 2 就已经加入了。IE 6 不支持:first-child 选择器，IE6~8 不支持:last-child 选择器。

6．:first-of-type 和:last-of-type

:first-of-type 相当于:nth-of-type(1)，:last-of-type 相当于:nth-last-of-type(1)，例如：

```
<style>
  p:last-of-type{
    color:blue;
  }
  p:first-of-type{
    color:red;
  }
</style>

<div>
  <p>今天早上吃稀饭</p>
```

```
  <p>今天中午吃面条</p>
  <p>今天晚上吃米饭</p>
</div>
```

代码执行结果如图 3.9 所示。

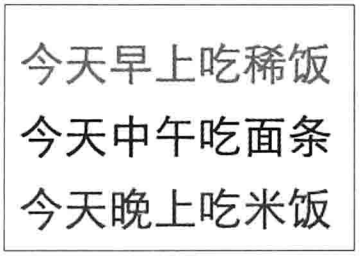


图 3.9　:first-of-type 和:last-of-type

7. :only-child

如果一个父元素只有一个子元素，那么选取这个子元素。如果加了限定条件，例如 p:only-child 则取交集，即如果一个父元素只有一个子元素，且这个子元素为<p>，这个选择器才会生效，如下面这个例子：

```
<style>
  p:only-child{
    color:red;
  }
</style>

<div>
  <p>only-child</p>
</div>
<div>
  <div>only-child</div>
</div>
<div>
  <p>not-only-child</p>
  <p>not-only-child</p>
</div>
```

代码执行结果如图 3.10 所示，只有第一段被 p:only-child 选择器选中，文字颜色为红色。

图 3.10　only-child 示例

8. :only-of-type

基本同:only-child，区别在于如果不指定 type 而直接使用:only-of-type 的话会造成 body 被选择，而:only-child 不会出现这种情况：

```
<style>
  :only-child{
    color:red;
  }
  :only-type-of{
    color:red;
  }
</style>

<div>
  <p>only-child</p>
</div>
<div>
  <p>not-only-child</p>
  <p>not-only-child</p>
</div>
```

这段代码测试分别使用:only-child 和:only-type-of 时的情况，结果如图 3.11 所示。使用:only-type-of 会对 HTML 文档里所有 body 中的元素生效。当然，如果添加了其他条件限制，使用:only-child 和:only-type-of 是没有区别的。

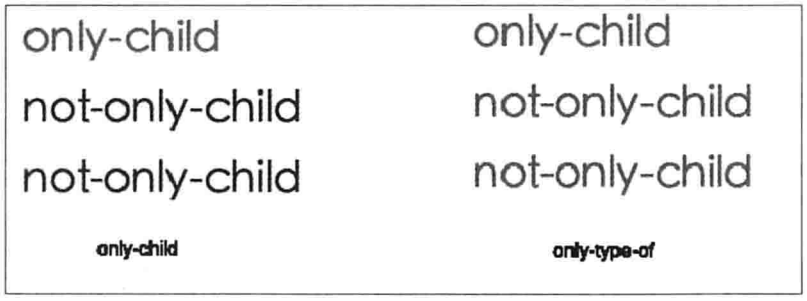


图 3.11　:only-child 和:only-type-of 的区别

9. :root

选择文档的根元素，对于 HTML 文档来说，根元素永远是<html>标签，由于:root 是一个 CSS 3 选择器，不兼容 IE 6~8，因此建议在实际开发中使用标签选择器来代替:root。:root 示例如下：

```
:root{
  background:white;
}
html{
  background:white;
}
/*这两段代码是等效的*/
```

10．:empty

:empty 是用来选择没有任何内容的元素，这里所说的没有内容指的是一点内容都没有，哪怕是一个空格。比如，有 3 个段落，其中一个段落完全是空的，但是段落标签还是会根据 CSS 的设置占据空间，想让这个 p 不显示，那么可以这样来写：

```
p:empty{
  display: none;
}
```

注意：IE 6~8 浏览器不支持:empty 选择器。

3.2.2　目标伪类:target

URL 前面有锚名称#，指向文档内某个具体的元素，例如。那么 这个被链接的元素就是目标元素（target element）。

“:target”选择器可用于选取当前活动的目标元素，例如：

```
<style>
:target{
  border: 2px solid #D4D4D4;
  background-color: #e5eecc;}
</style>
<a href="#a1">跳转至内容 1</a>    /*点击 a1 链接，则内容 1 的背景变为:target 中的设置*/
<a href="#a2">跳转至内容 2</a>    /*点击 a2 链接，则内容 2 的背景变为:target 中的设置，a1 恢
    复原状*/

<p id="a1"><b>内容 1...</b></p>
<p id="a2"><b>内容 2...</b></p>
```

注意：IE 6~8 浏览器不支持:target 选择器。

3.2.3　状态伪类

CSS 3 中新增了状态伪类选择器，用于表示表单元素的状态，虽然使用属性选择器可以达到相同的效果，但是使用状态伪类的语义性更强，更容易理解。不过遗憾的是 IE 6~8 等不支持 CSS 3 的浏览器仍然占有大量的市场份额，因此实战中不推荐使用状态伪类选择器，如果想实现相同的效果可以用属性选择器来代替。

1．:enabled 和:disabled

表单元素可以设置 disbaled 属性表示禁用，:enabled 选择器用于选择所有可用的元素，而:disabled 则用于选择所有已被禁用的元素，例如：

```
/*将被禁用的 input 元素设置为透明*/
input:disabled{
  opacity:0;
```

```
}
```

由于这些选择器不兼容 IE 6~8 浏览器，所以目前不建议使用:disabled 和:enabled，而采用属性选择器来代替：

```
/*使用属性选择器达到和使用:disabled 一样的效果*/
input[disabled"]{
  opacity:0;
}
```

2．:checked

input 表单中的 checkbox 和 radio 都使用 checked 属性表示是否选中，只要 checked 属性存在，使用 checked=false 或 checked=0 都会表示单选/复选框被选中。

:checked 选择器用于选择所有被选中的 checkbox 或者 radio 标签，例如：

```
/*将被选中的输入框设置为透明*/
input:checked{
  opacity:0;
}
```

目前不建议使用:checked 选择器，因为完全可以使用基础选择器中的属性选择器进行替代，而且兼容性上，使用属性选择器可以兼容所有的浏览器：

```
/*使用属性选择器达到和使用:checked 一样的效果*/
input[checked"]{
  opacity:0;
}
```

3．:indeterminate 和:default

:default 状态伪类选择器用来指定当前元素处于非选取状态的单选框或复选框的样式；:indeterminate 状态伪类选择器用来指定当页面打开时，某组中的单选框或复选框元素还没有选取状态时的样式。

注意：这两个选择器只有 Opera 浏览器才支持，因此强烈不建议使用。

3.2.4 否定伪类:not(S)

CSS 3 新加入的否定伪类可以使我们在针对某些特例进行排除时变得更为方便。:not(selector)选择器匹配非指定元素/选择器的每个元素，例如：

```
:not(p){
  background-color: red;
}
```

上面这段代码表示将除了段落（<p>）标签以外的所有 HTML 元素的背景颜色都设置为红色。

:not(S)选择器还可以配合其他选择器一起使用，例如：

```
div :not(.test){
  background: red
}
```

这段代码选择的是 标签的子元素中，class 不为 test 的所有其他元素。

说明：和其他 CSS 3 选择器一样，:not(S)选择器不支持 IE 6~8。

3.3　实战演练——选择器

前两节主要介绍了一些理论知识，本节将结合实际开发中的一些实例，讲解 CSS 选择器中除了最基本的类选择器、id 选择器以外的一些应用。

3.3.1　伪类选择器的实战——新闻聚合类网页

笔者曾经参与过一个新闻聚合类应用的开发，从多个新闻源抓取内容聚合后推荐给用户（类似于手机上流行的“今日头条”应用）。对于标准的新闻网站来说，文字的内容应该包裹在段落（<p>）标签内部，但是由于内容源自不同的网站，格式也各不相同，难免遇到各种不同的样式，比如文章开头或结尾是图片、导语、文章信息等其他内容。

这就造成了一个问题：文章内容与页头、页脚的间距可能是各不相同的。假设如图 3.12 所示是一篇标准的文章样式，其代码如下：

```
<header>雷吉-杰克逊 32 分，雷霆加时险胜灰熊</header>
    <div class="content">
      <p>北京时间 4 月 27 日，…… </p>
    </div>
<footer>.....</footer>
```

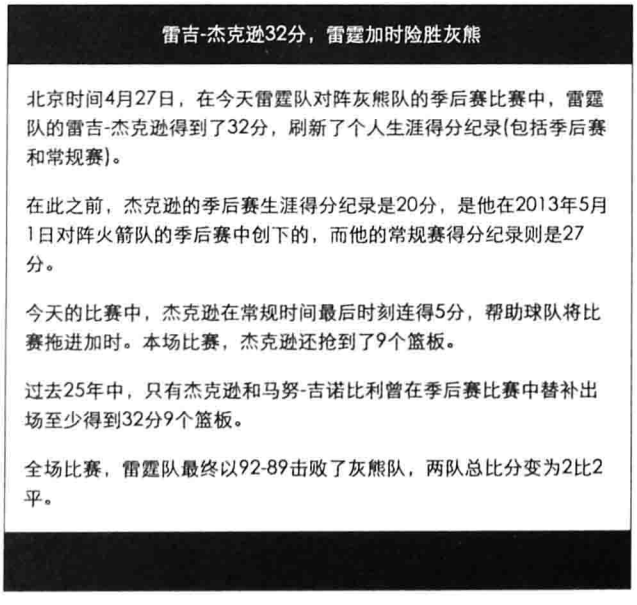


图 3.12　一篇标准的文章样式

而有些文章则可能有时间、来源等内容，如图 3.13 所示，其代码是这样：

```
<header>雷吉-杰克逊 32 分，雷霆加时险胜灰熊</header>
    <div class="content">
```

```
        <span>2014-04-28</span>  <span>来自：虎扑体育</span>
        <p>北京时间 4 月 27 日，…… </p>
    </div>
<footer>……</footer>
```

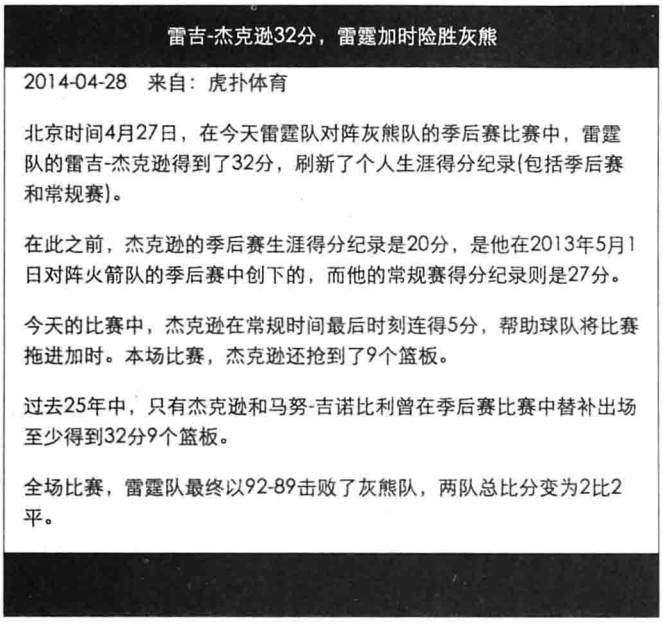


图 3.13　含有时间、来源的文章样式

通过比较图 3.12 和图 3.13 可以发现，文字和页头的间距发生了不一致的情况，通过标签选择器或类选择器等基础选择器很难处理这种情况，这里通过伪类选择器中的 first-child 和 last-child 则可以轻松地解决，代码如下：

```
// CSS 代码
.content>:first-child{
    padding-top: 0;
    margin-top: 1em;
}

.content>:last-child{
    padding-bottom: 0;
    margin-bottom: 1em;
}

header{
    margin-bottom: 1em;
}

footer{
    margin-top: 1em;
}
```

首先要设置 …… 的第一个子元素上方的内外边距为 0，同理设置最后一个子元素下方的内外边距为 0，然后为 header 和 footer 设置统一外边距，这样

即使 …… 内部的内容再不规范，内容和页头、页脚也可以保持一个固定的间距。修改后的示例如图 3.14 所示。

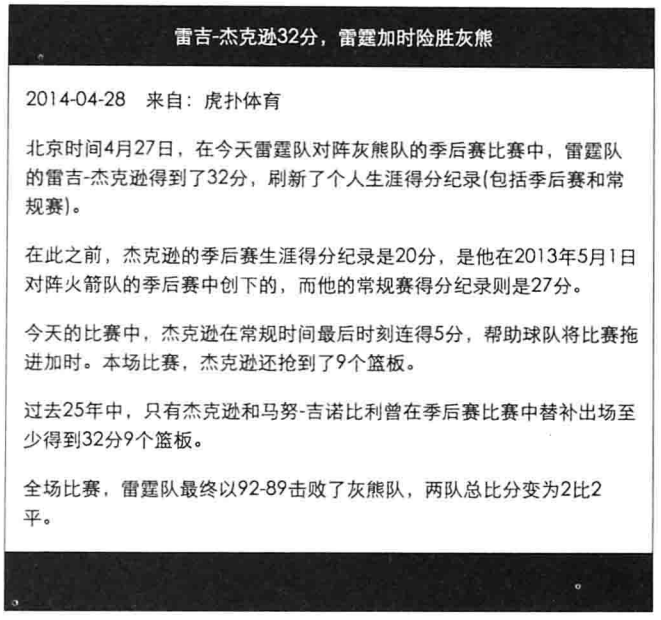


图 3.14　使用 first-child 和 last-child 后的效果

3.3.2　基础选择器的组合实战——新闻聚合类网页

本节选用和 3.3.1 小节同样的应用场景，由于 …… 内部可能采用正常的<p>标签来包裹文字段落，但是也有一些新闻源采用 或者<section>等其他标签来包裹文字，一般来说<p>和 、<section>具有不同的样式，比如行距、缩进、间距等，这样就可能造成样式的不一致，如图 3.15 是使用 代替<p>包裹文字后的效果。读者可以发现此时段落之间没有了段间距。

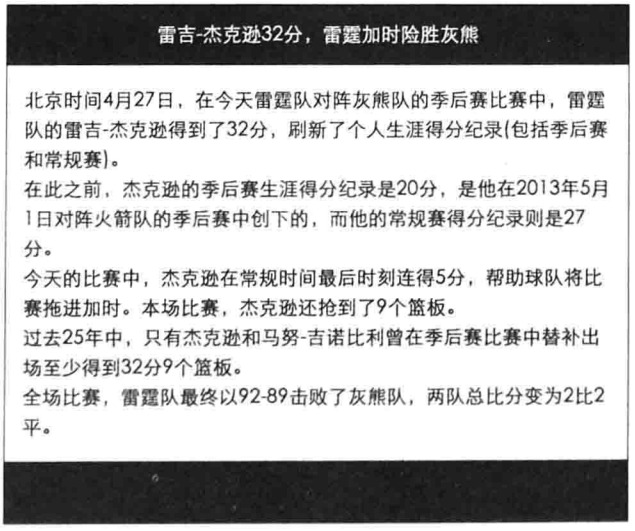


图 3.15　没有段间距的文章

这种情况如果单独去设置 标签的样式使其和<p>标签一致显然是不可取的，一方

面可能有问题的不止 标签一种，另一方面可能会对其他地方的样式产生影响。这里使用子元素选择器搭配通配符选择器则可以很好地解决这个问题，CSS 代码如下：

```
.content > *{
  margin:1em 0;
}
```

通过通配符选择器和子元素选择器的组合，选择所有 …… 的子元素，为它们添加和<p>标签一致的段间距，这样就可以尽可能地让不同文章之间的样式保持一致了。

3.4 小结

本章主要介绍了 CSS 选择器方面的知识，总结本章的要点如下：

- 标签选择器主要用于定义全局样式。
- 单一的类选择器不要滥用，因为比较容易出现命名冲突，或者语义不明的命名。对于复杂页面一般要结合子元素选择器或后代元素选择器一起使用。
- 可以使用通配符选择器进行一些全局样式定义。通配符选择器的权重最低，如果有冲突则一定会被覆盖。
- 注意子元素选择器和后代选择器之间的区别，子元素选择器必须符合严格的父子关系。
- 组选择器可以很好的缩减冗余代码。
- CSS 3 之前的伪类选择器只有:hover、:active、:visited、:link、:first-child、:lang 这 6 种，而在 CSS 3 中对伪类选择器进行了大量的扩展和强化。需要注意这些新增的选择器都无法在 IE 6~8 环境下使用。
- 状态伪类中的:indeterminate 和:default 只支持 Opera 浏览器，因此强烈不建议使用。

合理地使用 CSS 选择器可以更清晰地组织 CSS 代码，增强语义性、可读性，提高代码的复用程度，避免重复劳动。

第 4 章　设计更炫目的字体

对于网页设计来说，应用合适的字体绝对是必不可少的，但是一直以来，艺术字、图标等功能都只能依赖图片来完成。现在利用 CSS 3 的新特性，我们就可以通过设置字体本身来设计炫目且易于修改的艺术字体了。

本章主要介绍 CSS 3 在字体应用方面的新特性，包括以下几个要点：

- 添加和使用自定义字体，应用字体图标。
- 制作字体倒影效果。
- 制作字体阴影。
- 对字体描边。
- 实现排版分栏效果。

4.1　添加和使用自定义字体

在互联网上进行浏览的时候，偶尔会发现某些网站采用了一些平时没有见过的漂亮字体，这是如何做到的呢？本节将介绍如何应用自定义的特殊字体。

4.1.1　传统的字体定义

一般情况下，开发者会使用 CSS 中的 font-family 属性来定义字体，通常会根据优先级定义多个，如果用户的计算机上安装了 font-family 中定义的字体，就会使用指定的字体（优先使用定义时顺序靠前的）。如果没有定义字体或者定义的字体客户端上没有安装，就会显示默认的字体，如：

```
body {
  font: 12px '微软雅黑',"Microsoft JhengHei",'WenQuanYi Micro Hei', 'Helvetica Neue', Verdana,
    Arial, Helvetica, sans-serif;
}
```

上面这段代码在 Windows 7 的机器上时，页面将采用微软雅黑字体来展示中文，而在笔者的 Ubuntu 12.04 机器上则会应用“WenQuanYi Micro Hei”字体，因为那台机器没有安装微软雅黑，但是安装了文泉驿字体。

4.1.2 个性化的字体定义

如果想使用一些个性化的字体，怎么办呢？@font-face 属性可以帮助我们，它可以加载服务器端的字体文件，让客户端显示客户端所没有安装的字体。@font-face 的基本语法如下：

```
@font-face{
  font-family: myFirstFont;
  src: url('Sansation_Light.ttf'), url('Sansation_Light.eot'); /* IE9+ */
}
div{
  font-family:myFirstFont;
}
```

在@font-face 内通过 font-family 指定该字体的名称以便引用，通过 src 选项指定字体文件的 url 来获取服务器上的字体。图 4.1 展示的是造字工坊推出的“情书”字体。

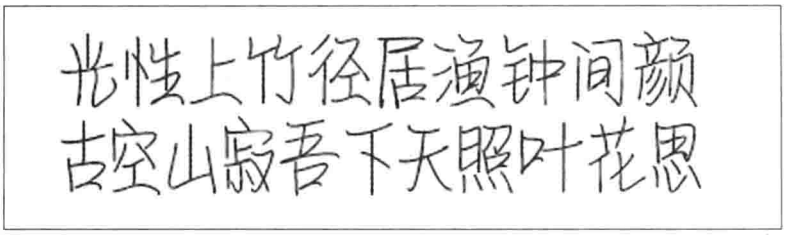


图 4.1 造字工坊的“情书”字体

说明：一般来说，个性化的字体需要给字体的版权方付费后才能使用。

4.1.3 个性化的字体图标

使用@font-face 属性，还可以添加“字体图标”。什么是字体图标呢？其实就是一种特殊的字体，过去都是使用 Photoshop 等作图工具来制作图标，考虑性能还要把多个图标合并成一张图片，如果想修改图标的大小和颜色就非常麻烦。而使用字体图标，我们就可以像定义字体一样随意地改变图标大小、颜色等特性，非常方便而且性能更佳。

图 4.2 展示的是目前比较流行的字体图标插件 Font Awesome 的部分图标。

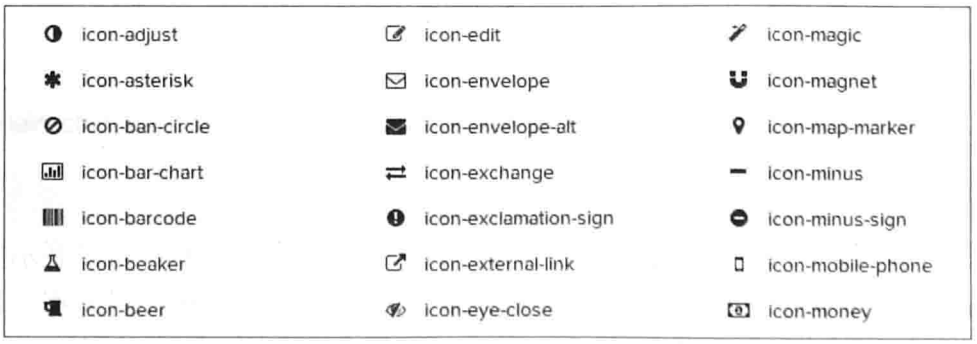


图 4.2 Font Awesome 图标集

Font Awesome 的使用非常简单，根据官方文档的说明引入相应的 CSS 和字体文件，然

后只要在需要的地方使用“”这样的形式即可，例如：

```
<p><i class="fa fa-camera-retro"></i>我是一个照相机</p>
```

其效果如图 4.3 所示。

图 4.3　Font Awesome 图标的应用

不过在作图方面，Photoshop 的功能毕竟比 CSS 要强大得多，一些复杂的图标，还是需要使用图片的方式。

4.2　使用反射让文字倒映

反射（box-reflect）属性其实并不能算是文字的属性，它也可以用在图片等其他方面。不过实际运用中，文字的倒映用反射比较多。

box-reflect 属性目前只有 webkit 核心浏览器支持，不过由于其效果对于布局没有任何影响，我们使用它并不会造成其他浏览器出现问题。因此在渐进增强的网页设计中可以放心大胆地使用这一特性，例如图 4.4 所示的文字的反射效果。

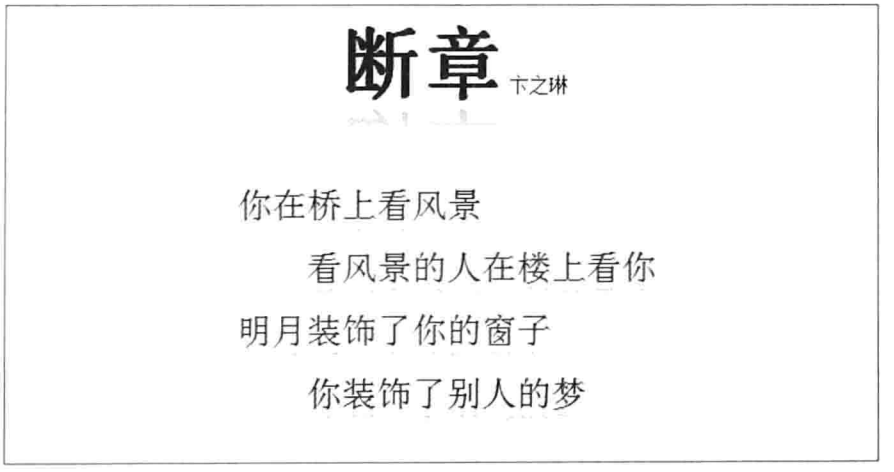


图 4.4　文字的反射效果

4.2.1　反射的基本语法

反射的基本语法如下：

```
.box-reflect: {<方向> <间距> <渐变效果>}
```

（1）方向：可以选择 above、below、left、right。

（2）间距：表示倒影和元素本身之间的额外距离。

注意：padding 同样会影响倒影之间的间距。

（3）渐变效果，有以下几种效果。

- none：无遮罩图像。
- <url>：使用绝对或相对地址指定遮罩图像。
- <linear-gradient>：使用线性渐变创建遮罩图像。
- <radial-gradient>：使用径向（放射性）渐变创建遮罩图像。
- <repeating-linear-gradient>：使用重复的线性渐变创建遮罩图像。
- <repeating-radial-gradient>：使用重复的径向（放射性）渐变创建遮罩图像。

注意：倒影是不占据空间的。

代码示例如下：

```
box-reflect: below 1px linear-gradient(transparent,transparent 50%,rgba(0,0,0,.3));
```

4.2.2 变幻多端的反射效果实例

来看下面的代码，并比较图 4.5 中两组“Hello World！”的异同：

```
//CSS 代码
<style>
.text-reflect{
    float:left;
    -webkit-box-reflect: below 10px -webkit-linear-gradient(transparent,transparent 50%,rgba
     (255,255,255,.5));
}
.text-reflect-base{
       float:left;
     -webkit-box-reflect:below 10px;
}
</style>

//HTML 代码
<div class='text-reflect-base'>Hello World!</div>
<div class='text-reflect'>Hello World!</div>
```

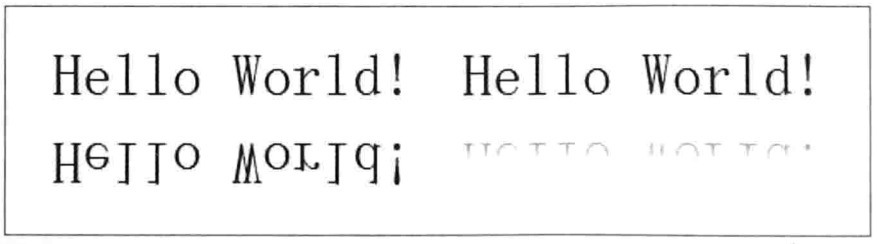


图 4.5 两组 Hello World!的效果对比

读者肯定会发现，如果不设置渐变效果的话，呈现的只是一个除了倒过来显示的文字外，其他什么都没变化的图像。其实对于倒影效果，读者只需要掌握 linear-gradient 的用法即可，其他几个属性应用范围都非常窄。

linear-gradient 共有 3 个参数：

- 第 1 个参数表示线性渐变的方向，top 是从上到下、left 是从左到右，如果定义成 left top，那就是从左上角到右下角，不过这个参数可以省略，默认是 top。
- 第 2 个和第 3 个参数分别是起点颜色和终点颜色。还可以在它们之间插入更多的参数，表示多种颜色的渐变。

注意：例子中的 transparent 是 rgba(0,0,0,0)的别名。

4.3　字体阴影——光晕、浮雕、投影效果

使用字体阴影（text-shadow），可以很容易地制作出光晕、浮雕、投影等效果，读者可以参看图 4.6 和图 4.7 的效果。

图 4.6 对应的代码：

```
h1{
  text-shadow: 0 0 5px #FF0000;
color:white;
  font-size:60px;
}
```

偏移量设为 0，就可以构建出光晕的效果。

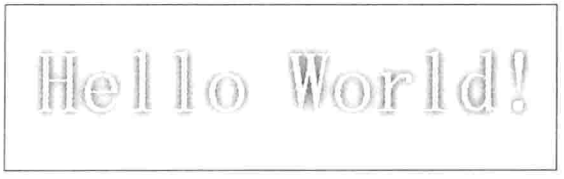


图 4.6　光晕效果

图 4.7 对应的代码：

```
h1 {
  color:white;
  text-shadow:2px 2px 4px #000000;
  font-size:70px;
}
```

垂直偏移量和水平偏移量设置相同的值，就可以构建出凸起的浮雕效果。

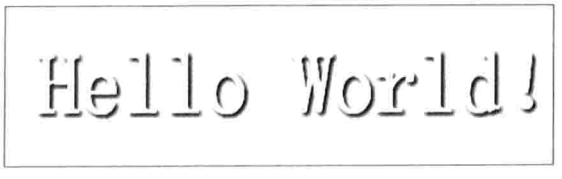


图 4.7　浮雕效果

字体阴影的基本语法如下：

```
text-shadow: h-shadow v-shadow blur color;
```

text-shadow 有 3 个长度参数，第 1 个表示水平偏移，第 2 个表示垂直偏移，第 3 个表

示模糊程度（可选）。颜色值可以写在最后，也可以写在最前。text-shadow 每个选项参数的详细解释参见表 4.1。

说明：text-shadow 属性向文本添加一个或多个阴影。该属性是逗号分隔的阴影列表，每个阴影有 2 个或 3 个长度值和 1 个可选的颜色值进行规定。省略的长度是 0。

表 4.1　text-shadow的选项参数

值	描述
h-shadow	必需，水平阴影的位置，允许负值
v-shadow	必需，垂直阴影的位置，允许负值
blur	可选，模糊的距离
color	可选，阴影的颜色

4.4　字体描边

字体描边（text-stroke）属性目前只有 webkit 核心浏览器支持，因此应用不算广泛。字体描边的语法示例如下：

```
<style>
h1{
        -webkit-text-stroke: 1.0px #000;
        text-stroke: 1.0px #000;
        color:white;
        font-size:50px;
}
</style>

<h1>啊！美丽的祖国！</h1>
```

text-stroke 的语法非常简单，一个长度值表示描边的宽度，一个颜色值表示描边的颜色。上述代码的效果如图 4.8 所示。

图 4.8　字体描边效果

不过在实际开发中，我们常常使用字体阴影（text-shadow）效果来制作字体描边效果，原理很简单，在每个方向上都添加模糊值为 0 的阴影即可。下面的代码就是使用 text-shadow 制作的字体描边，其效果基本类似图 4.8（颜色略有差别）。

```
.text-border{
    text-shadow:#000 1px 0 0,#000 0 1px 0,#000 -1px 0 0,#000 0 -1px 0;
    font-size:50px;
    font-weight:bold;
```

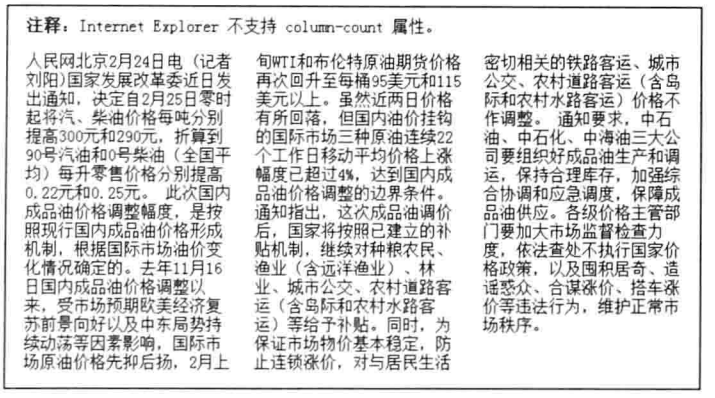
```
    color:white;
}
<div class="text-border">啊！美丽的祖国！</div>
```

虽然 text-shadow 语法上复杂了很多，但是胜在在滤镜的配合下可以通吃几乎所有浏览器。在 text-stroke 推广之前，建议开发者使用这个方法。

注意：IE 6~9 浏览器可以使用特有的滤镜来实现该效果。

4.5 字体分栏——让网页像报纸一样分栏排版

我们可以使用 CSS 3 的多列效果来为网页制作出类似报纸分栏的效果，如图 4.9 所示。当然，传统的方式一样能构造出类似的效果，比如构造多个左浮动的 盒模型，设置它们之间的间距来实现同样的显示效果，但是从代码复杂度和可维护性角度来讲传统方法并不够好。



注释： Internet Explorer 不支持 column-count 属性。

人民网北京2月24日电（记者刘阳）国家发展改革委近日发出通知，决定自2月25日零时起将汽、柴油价格每吨分别提高300元和290元，折算到90号汽油和0号柴油（全国平均）每升零售价格分别提高0.22元和0.25元。 此次国内成品油价格调整幅度，是按照现行国内成品油价格形成机制，根据国际市场油价变化情况确定的。去年11月16日国内成品油价格调整以来，受市场预期欧美经济复苏前景向好以及中东局势持续动荡等因素影响，国际市场原油价格先抑后扬，2月上旬WTI和布伦特原油期货价格再次回升至每桶95美元和115美元以上。虽然近两日价格有所回落，但国内油价挂钩的国际市场三种原油连续22个工作日移动平均价格上涨幅度已超过4%，达到国内成品油价格调整的边界条件。通知指出，这次成品油调价后，国家将按照已建立的补贴机制，继续对种粮农民、渔业（含远洋渔业）、林业、城市公交、农村道路客运（含岛际和农村水路客运）等给予补贴。同时，为保证市场物价基本稳定，防止连锁涨价，对与居民生活密切相关的铁路客运、城市公交、农村道路客运（含岛际和农村水路客运）价格不作调整。 通知要求，中石油、中石化、中海油三大公司要组织好成品油生产和调运，保持合理库存，加强综合协调和应急调度，保障成品油供应。各级价格主管部门要加大市场监督检查力度，依法查处不执行国家价格政策，以及囤积居奇、造谣惑众、合谋涨价、搭车涨价等违法行为，维护正常市场秩序。

图 4.9 报纸分栏效果

使用 CSS 3 的多列效果，需要用到以下 3 个属性。

- column-count：定义分列的数量
- column-gap：定义每一列中间的宽度
- column-rule：定义分栏中间的样式

column-count 和 column-gap 都非常好理解，这里重点讲解 column-rule 属性。如果我们需要在列之间添加一些样式，就需要用到它了。column-rule 的语法如下：

```
column-rule:样式宽度 样式类型 样式颜色;。
```

例如：

```
column-rule:2px outset red;
```

宽度和颜色好理解，样式都包括哪些呢？请看表 4.2。

表 4.2 column-rule-style的可选项

值	描述
none	没有定义规则
hidden	定义隐藏规则

续表

值	描述
dotted	定义点状规则
dashed	定义虚线规则
solid	定义实线规则
double	定义双线规则
groove	定义3D grooved规则，该效果取决于宽度和颜色值
ridge	定义3D ridged规则，该效果取决于宽度和颜色值
inset	定义3D inset规则，该效果取决于宽度和颜色值
outset	定义3D outset规则，该效果取决于宽度和颜色值

图 4.10 就是在每列的间隔处添加了 1 像素宽的虚线后的效果。其代码如下：

```
div{
    column-count:3;
    -webkit-column-count:3;
    column-gap:30px;
    -webkiy-column-gap:30px;
    column-rule:1px dashed black;
    -webkit-column-rule:1px dashed black;
}

.container{
    margin:auto;
    width:700px;
}
</style>
<div class="container">人民网北京 2 月 24 日电……</div>
```

注意：column-rule 中装饰线的宽度不会影响 column-gap 中定义的宽度。

人民网北京2月24日电（记者刘阳）国家发展改革委近日发出通知，决定自2月25日零时起将汽、柴油价格每吨分别提高300元和290元，折算到90号汽油和0号柴油（全国平均）每升零售价格分别提高0.22元和0.25元。 此次国内成品油价格调整幅度，是按照现行国内成品油价格形成机制，根据国际市场油价变化情况确定的。去年11月16日国内成品油价格调整以来，受市场预期欧美经济复苏前景向好以及中东局势持续动荡等因素影响，国际市场原油价格先抑后扬，2月上旬WTI和布伦特原油期货价格再次回升至每桶95美元和115美元以上。虽然近两日价格有所回落，但国内油价挂钩的国际市场三种原油连续22个工作日移动平均价格上涨幅度已超过4%，达到国内成品油价格调整的边界条件。 通知指出，这次成品油调价后，国家将按照已建立的补贴机制，继续对种粮农民、渔业（含远洋渔业）、林业、城市公交、农村道路客运（含岛际和农村水路客运）等给予补贴。同时，为保证市场物价基本稳定，防止连锁涨价，对与居民生活密切相关的铁路客运、城市公交、农村道路客运（含岛际和农村水路客运）价格不作调整。 通知要求，中石油、中石化、中海油三大公司要组织好成品油生产和调运，保持合理库存，加强综合协调和应急调度，保障成品油供应。各级价格主管部门要加大市场监督检查力度，依法查处不执行国家价格政策，以及囤积居奇、造谣惑众、合谋涨价、搭车涨价等违法行为，维护正常市场秩序。

图 4.10　虚线效果

注意：IE 10 和 Opera 支持多列属性。Firefox 需要前缀-moz-。Chrome 和 Safari 需要前缀-webkit-。IE 9 以及更早的版本不支持多列属性。

4.6 实战演练——处理字体溢出和破字

在网站的首页或边栏上输出新闻条目时，经常会遇到文字溢出或截断的问题。由于多数情况下，结构的整齐性优先度更高，我们就要求在文字完整的情况下进行截断处理，并加上一些提示，这种情况一般可以使用 CSS 的 text-overflow 属性。

首先来看一个 CSDN 论坛的例子，如图 4.11 所示，读者肯定会注意到最后一个论坛标兵用户的名称过长了。

图 4.11 文字溢出

这里就用到了 text-overflow:ellipsis，打开浏览器的开发工具会发现如下代码：

```
example dd .user_name {
  display: block;
  white-space: nowrap;          /*文本不换行*/
  text-overflow: ellipsis;      /*重点*/
  overflow: hidden;             /*不允许出现滚动条*/
  width: 100px;
}
```

text-overflow 的参数选项见表 4.3。

表 4.3 text-overflow的参数选项

值	描述
clip	修剪文本
ellipsis	显示省略号来代表被修剪的文本
string	使用给定的字符串来代表被修剪的文本

text-overflow 属性目前支持所有主流浏览器，为了兼容老版本的 Opera 浏览器，可以加上 Opera 浏览器的私有属性-o-text-overflow:ellipsis。

注意：老版本的火狐浏览器不支持 text-overflow 属性。如果追求完全适配的话，可以选择使用 JavaScript 来进行控制。

4.7 小结

本章主要介绍了 CSS 3 对文字的一些特殊处理效果，如使用服务端自定义字体、字体图标，以及如何应对实战中经常遇到的字体溢出情况。

笔者认为，CSS 3 在字体方面的新特性是带有一定的不可替代性的，而其他很多 CSS 3 的新特性其实都可以通过某些方案来变通解决。如大段文字的阴影，实际开发中我们是很难采用图片来代替的，而且字体的特效也不会造成布局的混乱，即使有些浏览器不支持，我们仍然可以比较放心地使用这些特效，达到渐进增强的效果。

对于开发者而言，这些新特性更是一个好消息，可以大大减少图片的使用量，提升网站的性能（尤其在移动设备的页面展示中）。并且在开发过程中，不断切换编程工具和 Photoshop 也是一件很痛苦的事情，这不但意味着思维模式的转换，也意味着电脑会变得运行不流畅，因为绘图工具可是十分消耗资源的。

第 5 章　背景和颜色

颜色搭配和背景的设计是网站吸引用户的关键要素，当然这主要考察的是设计师的功底，不过最终的实现还是需要应用 HTML 和 CSS。本章主要介绍 CSS 3 新增的对背景和颜色的支持特性，帮助读者更好地实现设计构想，制作更有吸引力、交互更好的网站。

本章的主要知识点有：

- 使用 background-size 属性来设置背景图片的尺寸
- 实现多背景
- 使用 origin 和 clip 属性
- RGBA 和 HSLA 颜色模式
- 设置透明度
- 实现渐变背景

5.1　设定背景图的大小

在 CSS 3 出现之前，背景图片的尺寸是由图片的实际尺寸决定的。如果同样的图片要在多个不同的地方作为背景的话，就必须用制图工具做成不同的尺寸，这一方面加大了开发者的工作量，另一方面也占用了更多的磁盘空间和网络空间。在 CSS 3 中，开发者可以使用 background-size 属性来规定背景图片的尺寸，这就可以在不同的环境中重复使用背景图片了。例如下面的代码：

```
div{
    background:url(img_flwr.gif);
    background-size:80px 60px;
    background-repeat:no-repeat;
}
```

最基本的用法当然是直接使用长度单位或者百分比来指定背景的尺寸，其中第 1 个值是宽度，第 2 个值是高度。如果只设置一个值，则高度默认是 auto。

background-size 还有两个可选项：cover 和 contain。这两个选项都不会造成图像比例失真。其中 cover 相当于宽度等于元素宽度、高度设为 auto 的情况；而 contain 则相当于高度等于元素高度、宽度设为 auto 的情况，下面举例说明。

首先，设置一个高度和宽度均为 300 像素的容器，然后将一张 1600⊠1200 尺寸的图片设置为图片的背景：

```
<style>
.container{
   background:url(naicha.jpg) no-repeat;
   border: 2px solid black;
   margin:auto;
   width:300px;
   height:300px;
}
</style>
<div class="container"></div>
```

效果如图 5.1 所示，由于背景取决于背景图片的尺寸，但背景图片太大，导致实际只显示了原图的左上角的部分。

图 5.1　原始图片背景

下一步加上 background-size，效果如图 5.2 所示。

```
<style>
.container{
   background:url(naicha.jpg) no-repeat;
   background-size: 100% auto;      /*设置宽度 100%，高度自动*/
   /*使用 background-size: 100% auto; 等效于使用 background-size: contain; */
   -webkit-background-size: 100% auto;
   border: 2px solid black;
   margin:auto;
   width:300px;
   height:300px;
}
</style>
<div class="container"></div>
```

现在读者可以发现图片的全貌展现出来了，宽度等于容器宽度，高度则根据原图比例生成，最终得到和原图比例一致的背景图片，使用 background-size: contain;等效于使用 background-size: 100% auto;。

如果想占满容器的高度，则只需设置 background-size: auto 100%;或者 background-size: cover;即可，效果如图 5.3 所示。

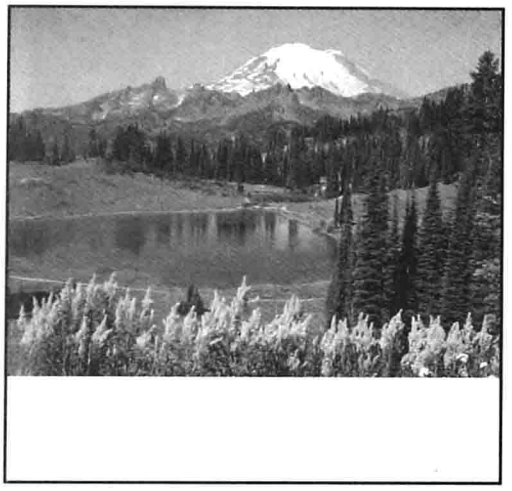


图 5.2　background-size: contain 效果

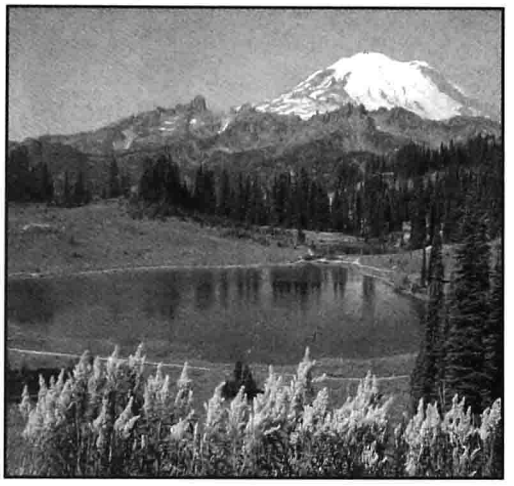


图 5.3　background-size: cover 效果

注意：background-size 一定要在指定图片后设定，否则不会生效。

5.2　利用图层叠加实现多背景

CSS 3 允许设置多个背景图片，每个背景图片占一层，层的上下按照在 CSS 中书写的顺序来定，最先写的背景在最上层，每层图片定义使用英文逗号隔开。

例如下面的代码：

```
background:url(http://dotnet.aspx.cc/Book/Images/CSS1_s.jpg) 0 0 no-repeat,
  url(http://dotnet.aspx.cc/Book/Images/CSS2_s.jpg) 200px 0 no-repeat,
  url("http://dotnet.aspx.cc/Book/Images/ASPNET20Book1_s.jpg") 400px 201px no-repeat;
```

根据代码可以看到，每张图片都可以设置各自的位移、大小、是否重复，如果出现重叠，那么写在前面的图片会覆盖在最上层。

多背景的效果如图 5.4 所示。

图 5.4 使用多张图片构造背景

多背景的意义在于，可以脱离 Photoshop 等图形工具构建一些简单的图层叠加效果，不过从性能上考虑，加载多张图片会增加 HTTP 请求，并且多张图片的总大小肯定是大于使用图片处理工具制作的单张图片的大小，从性能角度考虑笔者并不推荐这种做法。

5.3 使用图片背景的 origin 和 clip 属性

有一定基础的读者可能会知道，背景的显示范围是在元素的内边距（padding）之内的。可是如果想改变背景的显示范围，比如只在内容部分显示背景呢？ CSS 3 新增的 background-origin 和 background-clip 属性可以帮助开发者实现这一想法。

5.3.1 background-origin 属性

先来看一下 background-origin 属性，它指定了背景图片的起始位置。正常情况下，背景图片是从内边距的左上角开始延展的，而 background-origin 属性则可以指定从边框左上角（border-box）、内边距左上角（padding-box）、内容左上角（content-box）开始延展。例如，如图 5.5 所示的效果。

图 5.5 应用 background-origin 的效果

图 5.5 的代码如下：

```
//CSS 代码
<style>
.back{
   background:url(naicha.jpg) no-repeat;
   background-size: cover;                    /*这里使用了上一节讲到的 background-size*/
   height:150px;
   width:200px;
   padding: 20px;
   border: 10px dashed #333;                  /*为了清晰，这里使用了 10px 宽的边框*/
   background-origin: border-box;             /*注意这句代码，关键*/
   margin:30px;
   float:left;
}
.back1{
```

```
    @back                    /*由于大部分代码同.back，这里用@back 指代，增加可读性*/
    background-origin: padding-box;
}

.back2{
  @back
    background-origin: content-box;
}
</style>
//HTML 代码
<div class="back"></div>
<div class="back1"></div>
<div class="back2"></div>
```

由于图片开始延展的起始位置有变化，而且这个例子里对 background-size 进行了设置，因此我们看到的图像是越来越小的。

5.3.2　background-clip 属性

background-clip 控制的不仅仅是背景图片，而且控制整个元素背景的显示范围，根据图 5.6，读者应该可以很快地发现，3 幅图的背景是没有变化的，只是显示的区域不同。第 1 幅即使在边框部分也显示背景，第 2 幅在外边距内显示背景，第 3 幅则只在内容区域显示背景。

background-clip 和 background-origin 的可选参数是相同的，都是 border-box、padding-box、content-box。

图 5.6　应用 background-clip 的效果

图 5.6 的代码如下：

```
<style>
.back{
  background:url(naicha.jpg) no-repeat;
  background-size: cover;          /*这里使用了上一节讲到的 background-size*/
  height:150px;
  width:200px;
  padding: 20px;
  border: 10px dashed #333;        /*为了清晰，这里使用了 10px 宽的边框*/
```

```
  background-origin: border-box;    /*3 段代码统一采用相同的背景图片延展区域*/
  background-clip: border-box;    /*注意这句代码，关键*/
  margin:30px;
  float:left;
}

.back1{
  @back                    /*由于大部分代码同.back，这里用@back 指代，增加可读性*/
  background-clip: padding-box;
}

.back2{
 @back
  background-clip: content-box;
}
</style>
<div class="back"></div>
<div class="back1"></div>
<div class="back2"></div>
```

读者可以发现代码和图 5.5 的代码十分类似，可以对比图 5.5 和图 5.6，同时结合代码体会 background-origin 和 background-clip 的区别。

5.4 颜色模式

CSS 最基础的颜色模式是和计算机系统一致的 RGB 颜色，随着 CSS 标准和浏览器技术的发展，CSS 可以支持更多的颜色模式，包括在 RGB 颜色模式基础上增加透明通道的 RGBA 模式和工业界常用的 HSLA 颜色模式等。本节就来介绍 RGBA 模式和 HSLA 模式。

5.4.1 RGBA 模式

说到 RGBA 就不得不提 RGB 色彩模式。RGB 是计算机通用的一种颜色模式，它是 Red、Green、Blue 3 个单词的缩写，红、蓝、绿是光的三原色，通过这 3 种颜色，理论上可以调和出除了黑色以外的其他任何颜色。

RGB 色彩模式使用 RGB 模型为图像中每一个像素的 RGB 分量分配一个 0~255 范围内的强度值。这样只需要使用红蓝绿 3 种颜色，让它们按照不同的比例混合，就可以在屏幕上重现 16 777 216（256 *256 *256）种颜色了。

注意：光的三原色和美术中颜料的三原色（黄、品红、青，一般记为红、黄、蓝）是不同的。

RGBA 实际就是在 RGB 基础上添加了一个 Alpha 透明通道，可以表现颜色的透明度。

Alpha 通道一般用作透明度参数。如果 Alpha 通道数值为 0%，那它就是完全透明的（也就是看不见的），而数值为 100%则意味着完全不透明。在 0%~100%之间的值则使得像素可以透过背景显示出来，就像透过玻璃（半透明），这种效果是简单的二元透明性（透明或不透明）做不到的。它使数码合成变得更容易。Alpha 通道值可以用百分比、整数或者像 RGB 参数那样用 0~1 的实数表示。

在 CSS 中使用：

```
rgba(255, 255, 255,0.5)
```

这样的格式来表示 RGBA，其中最后一个参数表示 Alpha 通道，表示透明度。RGBA 支持绝大多数浏览器。下面是一个不同透明度的 RGBA 背景色示例：

```
div.rgbaL1 { background:hsla(83, 172, 150, 0.2); height:20px; }
div.rgbaL2 { background:hsla(83, 172, 150, 0.4); height:20px; }
div.rgbaL3 { background:hsla(83, 172, 150, 0.6); height:20px; }
div.rgbaL4 { background:hsla(83, 172, 150, 0.8); height:20px; }
div.rgbaL5 { background:hsla(83, 172, 150, 1.0); height:20px; }
```

效果如图 5.7 所示。

图 5.7　不同透明度的背景对比

注意：IE 6 及之前的浏览器是不支持 RGBA 的，不过可以使用 IE 特有的滤镜来添加透明效果。

5.4.2　HSLA 模式

和 RGB、RGBA 一样，HSLA 也是一种色彩模式，不过在浏览器支持上不如 RGB 和 RGBA。因此实际工作中，这种颜色表示法应用得较少。

HSLA 4 个字母分别代表色调（Hue）、饱和度（Saturation）、亮度（Light）、透明通道（Alpha）。以 CSS 中的 red 颜色为例，用 HSLA 色彩模式表示如下：

```
background: hsla(0, 100%, 100%, 1);
/*等效于下面的代码*/
background: red;
```

HSLA 颜色模式中的 4 个参数分别表示色调、饱和度、亮度、透明度，具体解释如下：

- Hue（色调）：0（或 360）表示红色，120 表示绿色，240 表示蓝色，可取其他数值来表示其他颜色。
- Saturation（饱和度）：　取值为 0%~100%之间的值。
- Lightness（亮度）：取值为 0%~100%之间的值。
- alpha（透明度）：取值在 0~1 之间。

注意：IE 8 及以下的浏览器不支持 HSLA 颜色表示法。

像 RGB 这样依赖于三原色颜色调和的表示方法，虽然计算机可以很好地识别，但对于开发者来说，更多的只能凭借经验来运用颜色，无法精确地调整颜色。而 HSLA 则是根据色调、饱和度、亮度、透明度这几个人类很好理解的数值来确定颜色。因此，在实际开发中可以先使用 HSLA 来调整颜色，再通过转换工具将其转化为 RGBA 颜色，最后应用到实际项目中。

读者可以通过 http://serennu.com/colour/hsltorgb.php 这个在线转换网站查看 HSLA 色彩模式和 RGBA 色彩模式之间的转换。

5.5 透明颜色

上一节介绍了 RGBA 和 HSLA 颜色模式，只要浏览器支持，在设置颜色的时候，都可以直接设置透明度参数。

但是怎么设置图片的透明度呢？如果使用的是 16 进制的 RGB 颜色呢？（类似#333333 这样的）。CSS 3 中加入了 opacity 属性来帮助开发者解决这个问题。

opacity 接受小于等于 1 的小数作为参数，不接受百分比，这一点需要注意。opacity 默认值为 1。opacity 还接受 inherit，用于继承父元素的透明度。由于 opacity 的浏览器支持度非常好，因此应用非常广泛。IE 8 及更低版本的 IE 浏览器不支持该属性，不过可以使用滤镜来代替。兼容代码如下：

```
.image{
  filter:Alpha(Opacity=20);          /*IE 兼容代码，应用 IE 中特有的滤镜效果*/
  opacity:0.2
}
```

如下面这段代码设置了图片的透明度，效果如图 5.8 所示。

```
<style>
  .opicity{
    filter:Alpha(Opacity=40);
    opacity:0.4;
  }
</style>

<img src="test.jpg">
<img src="test.jpg" class="opicity">
```

图 5.8 设置了透明度的图片对比

5.6　语法糖——currentColor 属性

currentColor 实际是 CSS 中的一个语法糖，IE 9 之前的浏览器不支持该属性，因此实际应用非常少，这里只做一个简单的介绍，不推荐使用。

有后端开发经验的读者可能会想到，后端代码编写中常常会将当前登录用户命名为 current_user。currentColor 和这个类似，意思是和当前文字颜色相同。

如果定义了 color 属性，那么在定义 background、border，以及子元素所有和颜色有关的属性时，就都可以使用 currentColor 属性了。

说明：语法糖指那些对语言的功能并没有影响，却更简短或更方便的用法。

5.7　渐变——放弃图片的首选良方

CSS 3 加入了渐变效果，大大降低了网页对图片的依赖，最明显的就是现在的网站凸起效果的按钮几乎都不再采用图片，而采用更容易修改、带宽占用更小的渐变效果来实现。渐变应用的关键词是 gradient，翻译过来就是梯度的意思。

渐变效果可以分为线性渐变（linear-gradient）和放射渐变（radial-gradient）两种，下面将详细介绍这两种渐变的效果。

5.7.1　线性渐变

线性渐变可以设置 3 个参数值：方向、起始颜色、结束颜色。最简单的模式只需要定义起始颜色和结束颜色，起点、终点和方向默认自元素的顶部到底部。下面举例说明：

```
.test{
  background: linear-gradient(red, blue);
}
```

上述代码的效果如图 5.9 所示。

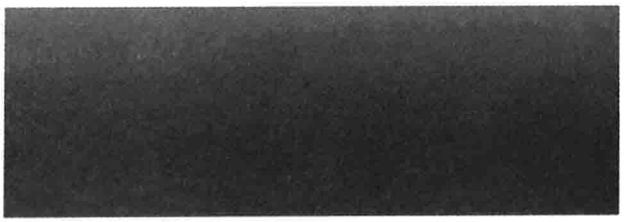


图 5.9　最简单的线性渐变效果

如果要在一些旧版本的浏览器（除 IE）下可以正常显示如图 5.9 的效果，则需要添加兼容代码：

```
.test {
  background: -webkit-linear-gradient(red, blue);          /* webkit 核心浏览器兼容代码*/
```

```
  background: -o-linear-gradient(red, blue);          /* Opera 浏览器兼容代码*/
  background: -moz-linear-gradient(red, blue);        /* Firefox 浏览器兼容代码*/
  background: linear-gradient(red, blue);             /* 标准语法要放在最后*/
}
```

线性渐变可以指定渐变的方向，如下例：

```
.test {
  background: -webkit-linear-gradient(left, red, blue);    /* webkit 核心浏览器兼容代码*/
  background: -o-linear-gradient(left, red, blue);         /* Opera 浏览器兼容代码*/
  background: -moz-linear-gradient(left, red, blue);       /* Firefox 浏览器兼容代码*/
  background: linear-gradient(to right, red, blue);        /* 标准语法要放在最后*/
}
```

上述代码的效果如图 5.10 所示，设置了 left/to right 参数后，渐变方向从自上而下变成了自左向右。

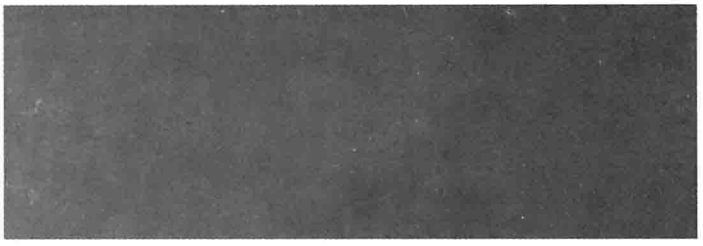


图 5.10 指定起点

注意：标准写法的渐变方向格式如上例中的“to right”，在火狐和 Opera 浏览器下则使用 right，而对于 webkit 核心浏览器则使用起点位置 left 来表示。

渐变方向还可以使用角度来表示，0deg、90deg、180deg 和 270deg 分别对应 to top、to right、to bottom 和 to left，例如：

```
.test {
  background: -webkit-linear-gradient(45deg, red, blue);    /* webkit 核心浏览器兼容代码*/
  background: -o-linear-gradient(45deg, red, blue);         /* Opera 浏览器兼容代码*/
  background: -moz-linear-gradient(45deg, red, blue);       /* Firefox 浏览器兼容代码*/
  background: linear-gradient(45deg, red, blue);            /* 标准语法 */
}
```

效果如图 5.11 所示。

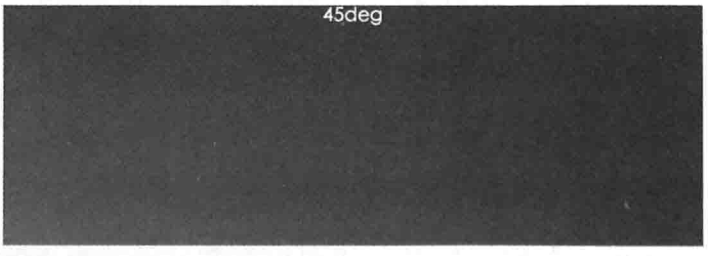


图 5.11 指定渐变方向为 45°

线性渐变不止支持两种颜色的渐变，还可以添加任意种颜色，比如可以使用线性渐变构造一个彩虹效果，如图 5.12 所示。

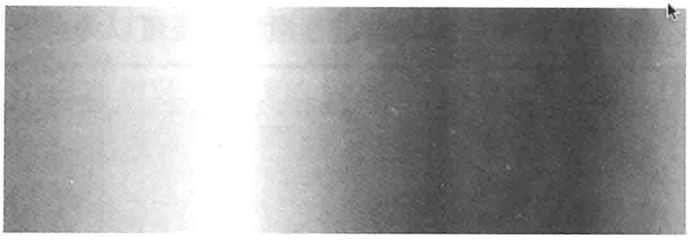


图 5.12　彩虹色

图 5.12 所示的彩虹色效果代码如下：

```
.test {
   background: -webkit-linear-gradient(left,red,orange,yellow,green,blue,indigo,violet);
   background: -o-linear-gradient(left,red,orange,yellow,green,blue,indigo,violet);
   background: -moz-linear-gradient(left,red,orange,yellow,green,blue,indigo,violet);
   background: linear-gradient(to right, red,orange,yellow,green,blue,indigo,violet);
}
```

5.7.2　放射渐变

线性渐变是从一个方向向相对方向的色彩渐变效果，放射渐变则用于表现从中心向外发散的色彩渐变效果。

放射渐变的关键词是 radial-gradient，它的基本效果只需要定义渐变的颜色即可，语法规则和线性渐变是一致的，例如：

```
.test{
   height:200px;
   width:200px;
   border-radius:50%;
   background: -webkit-radial-gradient(red, green, blue);          /* webkit 核心浏览器兼容代码*/
   background: -o-radial-gradient(red, green, blue);               /* Opera 浏览器兼容代码*/
   background: -moz-radial-gradient(red, green, blue);             /* Firefox 浏览器兼容代码*/
   background: radial-gradient(red, green, blue);
}
```

其效果如图 5.13 所示。

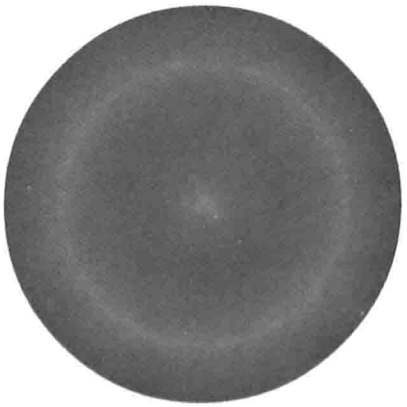


图 5.13　放射渐变

如果只指定颜色，那么 3 种颜色将均匀进行渐变，还可以为渐变加上渐变位置来改变不同颜色所占据的比例，例如：

```
.test{
  height:200px;
  width:200px;
  border-radius:50%;
  background: -webkit-radial-gradient(red 5%, green 15%, blue 60%); /* webkit 核心浏览器兼容*/
  background: -o-radial-gradient(red 5%, green 15%, blue 60%);     /* Opera 浏览器兼容代码*/
  background: -moz-radial-gradient(red 5%, green 15%, blue 60%); /* Firefox 浏览器兼容代码*/
  background: radial-gradient(red 5%, green 15%, blue 60%);        /*标准语法*/
}
```

最终效果如图 5.14 所示。还可以指定中心点的位置，效果如图 5.15 所示。

图 5.14　指定渐变位置的放射渐变

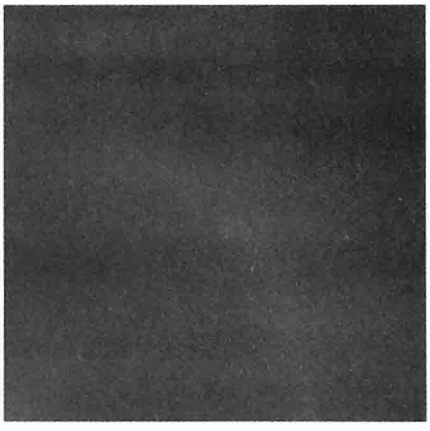


图 5.15　指定中心位置的放射渐变

图 5.15 的实现代码如下：

```
.test{
  height:200px;
  width:200px;
  border-radius:50%;
  background: -webkit-radial-gradient(buttom left, red 5%, green 15%, blue 60%);
  background: -o-radial-gradient(buttom left, red 5%, green 15%, blue 60%);
  background: -moz-radial-gradient(buttom left, red 5%, green 15%, blue 60%);
  background: radial-gradient(buttom left, red 5%, green 15%, blue 60%);
}
```

上述代码加上 buttom left 选项，表示发散的中心点位于元素的左下角。还可以使用百分比来指定横纵坐标。

除了之前介绍的颜色、渐变位置、中心点位置这 3 个常用的选项外，radial-gradient 还有其他一些选项参数可以添加。

- 形状：有 circle（圆形、默认）和 ellipse（椭圆）两个选项。
- 尺寸：参见表 5.1。

表 5.1　放射渐变范围的可选参数

closest-side	指定径向渐变的半径长度为从圆心到离圆心最近的边
closest-corner	指定径向渐变的半径长度为从圆心到离圆心最近的角
farthest-side	指定径向渐变的半径长度为从圆心到离圆心最远的边
farthest-corner	指定径向渐变的半径长度为从圆心到离圆心最远的角
contain	包含，指定径向渐变的半径长度为从圆心到离圆心最近的点
cover	覆盖，指定径向渐变的半径长度为从圆心到离圆心最远的点

5.8　实战演练——渐变效果

看完上一节繁多的渐变语法介绍，那么一般在哪些场景下使用渐变背景呢？本节就为读者介绍渐变在实际项目中的一些应用。

5.8.1　带有立体凸起效果的按钮

自上而下的线性渐变应用最广泛的地方就是构造带有凸起效果的按钮了，比如 Bootstrap 2 的按钮。

图 5.16 可以看到按钮的颜色有自上而下逐渐加深的一个过程。图 5.17 是笔者曾经制作的一个下拉工具按钮，也应用了背景的线性渐变来实现带有立体感的效果。

图 5.16　Bootstrap 2 带有凸起效果的按钮

图 5.17　下拉工具按钮

图 5.17 的实现代码如下：

```
.gray-dropdown{
  ……/*其他 CSS 代码*/
  background:  -webkit-gradient(linear,  50%  0%,  50%  100%,  color-stop(0%,  #ffffff),
    color-stop(100%, #e9e9e9));                    /*兼容老版本的 webkit 浏览器*/
  background: -webkit-linear-gradient(#ffffff, #e9e9e9);   /*兼容较新版本的 webkit 浏览器*/
  background: -moz-linear-gradient(#ffffff, #e9e9e9);
  background: -o-linear-gradient(#ffffff, #e9e9e9);
  background: linear-gradient(#ffffff, #e9e9e9);
  /*这里应用了一个 IE 兼容的插件，用滤镜实现渐变效果*/
  -pie-background: linear-gradient(#ffffff, #e9e9e9);
}
```

添加了兼容性代码，再加上应用 PIE 这样的 IE 滤镜工具就实现了所有浏览器下样式的一致，达到了让按钮带有凸起立体感的效果。

5.8.2 构造尺寸更灵活的背景

相比传统的图片背景来说，使用 CSS 构造背景色不仅可以降低网络传输的开销，更由于其尺寸的可控性受到开发者的青睐。

如果设计师设计了一张背景图片作为标题背景，如图 5.18 所示。对于用电脑浏览网页的用户来说，标题基本不存在折行现象，布局也基本是固定宽度的，因此直接使用设计师给出的背景图即可。但是这个页面如果是在手机上显示，标题根据长度不同可能占 1 行，也可能占 3 行，如果采用图片就必须根据不同的情况放不同的背景图，实现起来很复杂。

图 5.18　设计师给出的背景

好在这张背景图实际是从左到右由深紫蓝色渐变至较浅的蓝色再渐变至深蓝色，那么直接采用多色彩值的线性渐变定义样式即可：

```
.header{
  background-image: -webkit-linear-gradient(left,#241a38,#012c57,#031a40);
  background-image: -o-linear-gradient(left,#241a38,#012c57,#031a40);
  background-image: -moz-linear-gradient(left,#241a38,#012c57,#031a40);
  background-image: linear-gradient(left,#241a38,#012c57,#031a40);
}
```

采用 CSS 方案代替图片，如果标题折行，则自动撑大标题区域。不管几行都可以完美适配，具有非常好的灵活性，而且大大降低了实现的复杂度，同时也节省了图片加载造成的网络流量，可谓一举多得。

5.8.3 使用放射渐变制作光影效果

阴影效果通常用来表现光线投射在物体上的感觉，如果想制作一个如图 5.19 所示的文字光影效果，就可以使用背景的线性渐变进行构建。

图 5.19　文字光影效果

图 5.19 中可以看到有类似光束照射文字的效果，很好地突出了文字。这实现起来很简单，只需要对文字居中，对背景元素设置从中心发散的放射渐变即可，代码如下：

```
// HTML 代码
<div class="box">赞</div>

// CSS 代码：
.box{
    width: 200px;
    height: 200px;
    font-size: 80px;
    line-height: 200px;
    text-align: center;
background: -webkit-radial-gradient(#feb3ad, #fd695d);
background: -o-radial-gradient(#feb3ad, #fd695d);
background: -moz-radial-gradient(#feb3ad, #fd695d);
background: radial-gradient(#feb3ad, #fd695d);
}
```

这样，一个最基本的光影背景效果就完成了，可以通过调节颜色来设置光影的亮度，越接近白色越亮。通过调整中心色彩占据的百分比来调节光晕效果的范围，如图 5.20 所示。

图 5.20　调节光晕效果

相比图片来说，开发者可以通过直接调整 CSS 代码中的参数获得效果的改变，灵活性和开发速度都大大提高了。

5.9　小结

本章主要介绍了 CSS 3 中增加的一些背景设置功能，包括使用 background-size 属性来设置背景图片的尺寸、多背景、背景透明度设置、origin 和 clip 选项、RGBA 和 HSLA 颜色模式的支持、渐变背景色等。

CSS 3 新增的 background-origin 和 background-clip 可以用来改变背景的显示范围，不同点在于，background-origin 采用的是缩放图片的方式而 background-clip 采用的是切割图片的方式。

RGBA 颜色模式是 IE 6 之后的浏览器开始支持的颜色模式，支持透明度的设置；HSLA 是 IE 8 以后的新式浏览器开始支持的颜色模式，采用色调、饱和度、亮度、透明度 4 个值来表示颜色。

CSS 3 加入了颜色的渐变效果，包括线性渐变和放射渐变（也称径向渐变），可以在很多场景下代替图片效果，比如带有立体效果的按钮、彩虹效果等。使用 CSS 新特性代替图片可以降低网络传输的压力，节省带宽，提升网站的整体性能。

第 6 章　更个性的边框

CSS 3 新增了 3 种边框（border）属性：

- border-radius：圆角边框。
- box-shadow：边框阴影。
- border-image：图片边框。

目前 Internet Explorer 9+ 支持 border-radius 和 box-shadow 属性。Firefox、Chrome 以及 Safari 支持所有新的边框属性。本章的内容将详细介绍这 3 种新属性，以及可滚屏元素的 resize 属性。

注意：对于 border-image，Safari 5 以及更老的版本需要前缀-webkit-。Opera 支持 border-radius 和 box-shadow 属性，但是对于 border-image 需要前缀-o-。

6.1　圆角边框

圆角边框是 CSS 3 新增的属性，在圆角边框出现之前，前端开发工程师或者采用整块的圆角图片作为背景，或者采用小的圆角图片分别放在元素的四角，非常麻烦，而且灵活性也非常差，加入更多的图片也大大降低了网站的整体性能。圆角边框的出现改变了这一现状，大大降低了开发和维护的难度。

6.1.1　圆角边框的基本用法

圆角边框（border-radius）最基本的用法是设置 4 个相同弧度的圆角，例如：

```
div{
  border:1px solid;
  border-radius:5px;
  -moz-border-radius:5px;          /*支持旧版本的 Firefox 浏览器*/
  -webkit-border-radius:5px;       /*支持旧版本的 webkit 核心浏览器*/
}
```

代码显示效果如图 6.1 所示。每个角实际是一个 1/4 圆，如果以像素作为单位，圆角的单位像素代表圆角位置圆的半径。

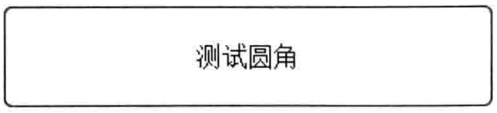


图 6.1　圆角边框效果

6.1.2　使用百分比作为单位

border-radius 也可以使用百分比作为单位，比如对一个正方形设置 border:50%，那么就会形成一个圆形，不过使用百分比和使用像素并不能等效，例如下面的代码：

```
.round1{
        height:120px;
        width:120px;
        border:1px solid;
        border-radius:100%;
        -moz-border-radius:50%;
        margin:20px 0 0 30px;
        float:left;
 }                              /*图 6.2 左边图形的 CSS 代码*/
.round2{
        height:120px;
        width:240px;
        border:1px solid;
        border-radius:20%;
        -moz-border-radius:50%;
        margin:20px 0 0 30px;
        float:left;
 }                              /*图 6.2 右边图形的 CSS 代码*/

<div class="round1"></div>
<div class="round2"></div>
```

这段代码的显示效果如图 6.2 所示。

图 6.2　使用百分比作为单位的圆角效果

从图 6.2 可以看出，如果使用百分比作为单位，圆角不再是一个 1/4 圆形，而是一个和宽高比相匹配的椭圆。

注意：百分比大于 50%之后，形状就不会再变化了，因为圆角的半径不能超过长/宽的一半。

6.1.3　设置不同弧度的圆角

我们还可以为元素的 4 个角设置不同弧度的圆角：

```
border-top-left-radius: 30px;          /*左上角*/
border-top-right-radius: 30px;         /*右上角*/
border-bottom-left-radius: 30px;       /*左下角*/
border-bottom-right-radius: 30px;      /*右下角*/
```

根据例子，我们可以很容易地总结出命名的规律，这样就可以实现如图 6.3 的效果。

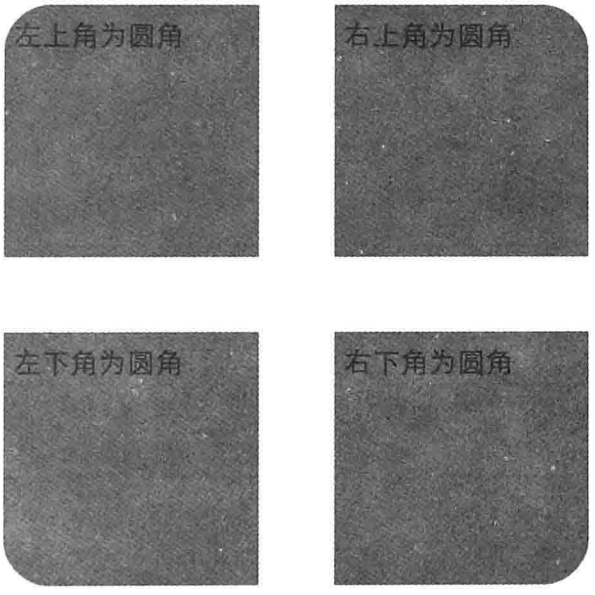


图 6.3　设置不同弧度的圆角效果

上例中的每一句代码都要单独编写兼容代码，例如我们要为左上角和右下角设置不同的圆角，代码就要这样书写：

```
border-top-left-radius: 30px ;
moz-border-top-left-radius: 30px;
webkit-border-top-left-radius: 30px;
border-bottom-right-radius: 20px;
moz-border-bottom-right-radius: 20px;
webkit-border-bottom-right-radius: 20px;
```

是不是很麻烦？别急，在本书的第 14 章会介绍解决方案，这里主要是了解基本的设定和原理。

6.2　边框阴影

严格来说，边框阴影（box-shadow）和边框并没有什么关系，即使不设置边框也可以设置 box-shadow，不过它们都描绘了元素的边界样式，所以笔者把它们归为一类。利用边框阴影，可以制作出光晕、浮雕等原来只有依赖 PS 图片才能达成的效果，如图 6.4 所示。

图 6.4　光晕、浮雕等效果

注意：IE 6~8 不支持 box-shadow，IE9+、Firefox 4、Chrome、Opera 以及 Safari 5.1.1 支持 box-shadow 属性。

首先通过表 6.1 来看一下 box-shadow 属性都有哪些参数。

表 6.1　box-shadow的参数

值	描述
h-shadow	必需，水平阴影的位置，允许负值
v-shadow	必需，垂直阴影的位置，允许负值
blur	可选，模糊距离
spread	可选，阴影的尺寸
color	可选，阴影的颜色，请参阅 CSS 颜色值
inset	可选，将外部阴影（outset）改为内部阴影

参数的组合顺序是这样的：

```
box-shadow{color 水平偏移 垂直偏移 模糊距离 阴影尺寸 inset}
```

例如下面的代码：

```
box-shadow: red 10px 10px 5px 10px inset;
 /*阴影颜色为红色，水平偏移 10px，垂直偏移 10px，模糊距离 5px，阴影尺寸 5px，内阴影*/
box-shadow:10px 10px 30px;
box-shadow: 0 0 10px;
```

阴影的难点在于理解内外阴影、偏移量、模糊距离、阴影尺寸这 4 个概念。

6.2.1　内外阴影

首先根据图 6.5 来看看内外阴影的区别。

图 6.5　内外阴影的区别

图 6.5 的代码是：

```
.shadow{
    box-shadow: 0 0 30px 10px;
    height:100px;
    width:100px;
    margin:30px;
    float:left;
}

.shadow_inset{
    box-shadow: 0 0 30px 10px inset;
    height:100px;
    width:100px;
    margin:30px;
    float:left;
}
```

6.2.2　偏移量

理解了内阴影和外阴影的区别后，来看看偏移量的概念，在图 6.5 的基础上添加水平和垂直 10px 的偏移量，效果如图 6.6 所示。

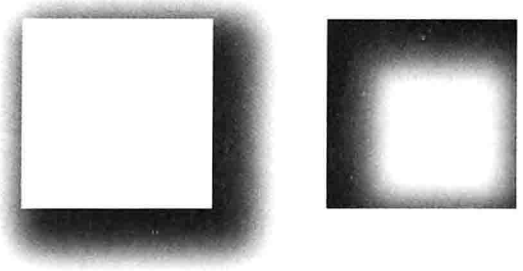


图 6.6　添加水平和垂直 10px 的偏移量

我们发现类似于 CSS 中用于定位的 top、left 属性，对于外阴影，相当于把阴影从图形的正后方，向右下进行了偏移。而对于内阴影，相当于图形内部没有被阴影覆盖的部分向右下进行了偏移。图 6.6 的代码是：

```
.shadow{
    box-shadow: 10px 10px 30px 10px;
    height:100px;
    width:100px;
    margin:30px;
    float:left;
}

.shadow_inset{
    box-shadow: 10px 10px 30px 10px inset;
    height:100px;
```

```
    width:100px;
    margin:30px;
    float:left;
}
```

6.2.3 阴影尺寸

现在，我们将除阴影尺寸外的值都设置为 0，效果如图 6.7 所示。

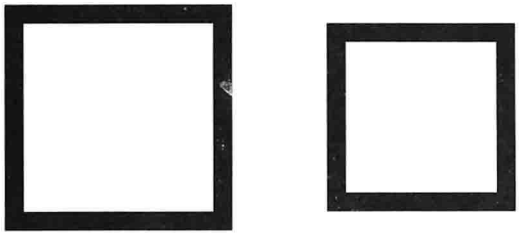


图 6.7 阴影尺寸外的值都为 0

从图中可以很直观地发现，阴影尺寸就是指阴影外延出去总的长度。图 6.7 的代码是：

```
.shadow{
    box-shadow: 0 0 0 10px;
    height:100px;
    width:100px;
    margin:30px;
    float:left;
}

.shadow_inset{
    box-shadow: 0 0 0 10px inset;
    height:100px;
    width:100px;
    margin:30px;
    float:left;
}
```

6.2.4 模糊距离

我们把除了模糊距离外的其他值都设为 0，把模糊距离设为 100px，和边长相等，则效果如图 6.8 所示。

从图中可以发现，模糊距离和阴影尺寸是不同的，模糊距离设为 100px 后正好填充了内阴影图形的内部，阴影尺寸等于外延长

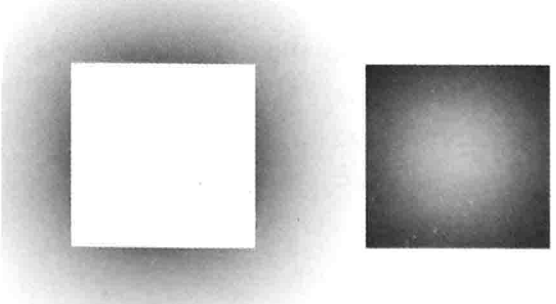


图 6.8 模糊距离设为 100px

度，而模糊距离的外延长度则是设定值的 1/2。图 6.8 的代码如下：

```
.shadow{
  box-shadow: 0 0 100px;
  height:100px;
  width:100px;
  margin:30px;
  float:left;
}

.shadow_inset{
  box-shadow: 0 0 100px inset;
  height:100px;
  width:100px;
  margin:30px;
  float:left;
}
```

注意：和边框不同，阴影是不占据空间的。

6.3　图片边框——让图片环绕在元素周围

图片边框（border-image）可以让图片环绕在元素周围。不过，一方面由于 IE 全系不支持该属性（包括最新的 IE 10），一方面仍然需要加载图片，相比传统的图片背景的方式性能提升并不太大，所以这个新属性在实际环境中应用还很少。表 6.2 列出了 border-image 的参数选项。

表 6.2　border-image的参数选项

值	描述
border-image-source	用在边框的图片的路径
border-image-slice	图片边框向内偏移
border-image-width	图片边框的宽度
border-image-outset	边框图像区域超出边框的量
border-image-repeat	图像边框是否应平铺（repeat）、铺满（round）或拉伸（stretch）

Firefox、Chrome 以及 Safari 6 都支持 border-image 属性。Opera 支持替代的-o-border-image 属性，因此在编写兼容性代码时需要添加-o-前缀，以适应 Opera 浏览器。Safari 5 支持替代的-webkit-border-image 属性。

注意：图片边框是不占据空间的。

1. border-image-source

语法：

```
border-image-source:url(image url);          /*image url 可以是相对地址也可以是绝对地址*/
```

border-image-source 跟 background-imgae 属性相似，也通过 url()调用背景图片，图片的路径可以是相对地址也可以是绝对地址，当然如果不想使用背景图片，也可以把值设置为 none，即：border-image:none，其默认值就是 none。

2．border-image-slice

语法：

```
border-image-slice: [ <number> | <percentage>]
```

其取值支持数字和百分比，其中数字是没有单位的，专指像素（px），如果加上了单位反而是错误的写法。另外还可以使用百分比值来作为单位，百分比的值是相对于边框背景图片而言的，例如边框图片的大小是 300px*240px，我们取百分比为 25% 30% 15% 20%，那么它们实际对应的效果就是剪切了图片的 60px 90px 36px 60px 的四边大小。

那么 border-image-slice 起到了什么作用呢？请看图 6.9。它根据我们设置的值将图片切割成一个九宫格，其中四个角作为图片边框的四角，中间部分则根据 border-image-repeat 的取值选择拉伸还是重复显示。

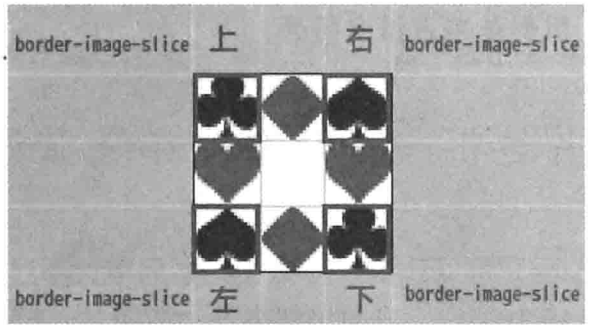


图 6.9　border-image-slice 效果

3．border-image-width

border-image-width 属性定义了图片边框的宽度。

4．border-image-outset

border-image-outset 属性定义了图片边框外延的距离，默认为 0，即图片边框默认在元素内部。

5．border-image-repeat

border-image-repeat 属性定义了 4 条边的显示，它有 3 个选项：repeat、round、stretch。根据图 6.10 来讲解一下它们之间的区别。

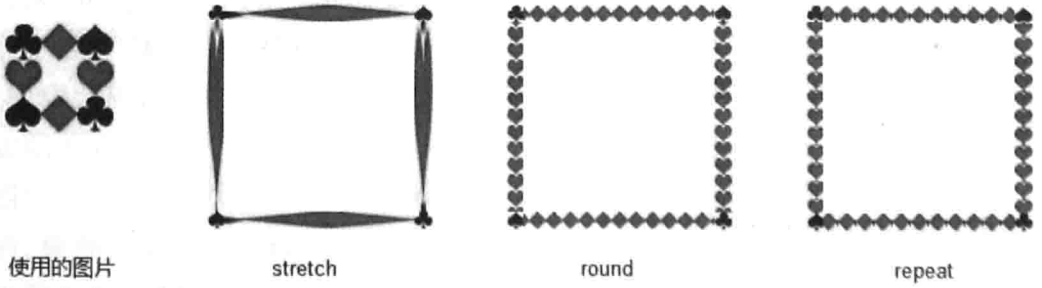


图 6.10　border-image-repeat 属性 3 个选项的区别

根据图 6.10 可以很清楚地看出，stretch 实际就是将除了 4 个角以外的其他部分进行拉伸，以适应元素的边长。而 round 和 repeat 看起来好像差不多，实际上还是有区别的：round 会压缩或伸展 border-image 的背景图片以其刚好适应 border-width 的宽度，从而正好在边框区域内显示，而 repeat 就不一样了，它不管什么适合不适合，直接居中重复。在图中可以看到，使用 round 的重复片段更完整，而 repeat 会存在直接被切割成半个的图形片段。

6.4　通过 resize 属性来改变输入框的大小

对于像<textarea>这样可以设置 overflow 属性的元素，CSS 3 提供了一个叫做 resize 的方法，让用户可以通过拖拽来改变框体的大小。

如图 6.11 所示，用户通过拖拽文本输入框右下角的图案就可以改变框体大小。resize 的可选项见表 6.3。

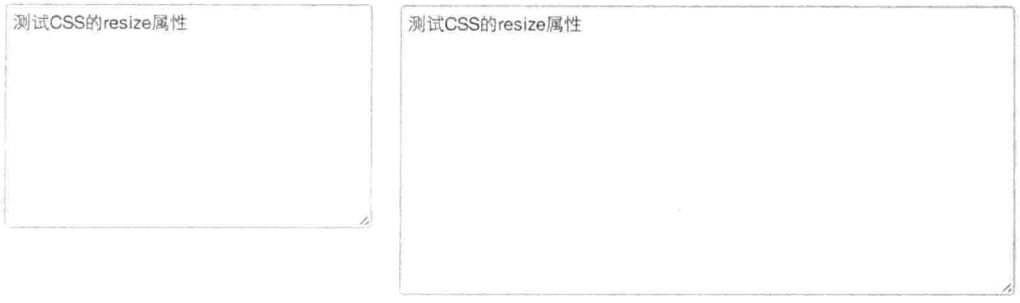


图 6.11　拖拽文本输入框的大小

表 6.3　resize属性的选项参数

值	描述
none	用户无法调整元素的尺寸
both	用户可调整元素的高度和宽度
horizontal	用户可调整元素的宽度
vertical	用户可调整元素的高度

注意：目前只有 webkit 核心浏览器才支持 resize 属性，并且只支持等比例调整。

resize 属性默认是打开的，如果想关闭 resize，有两种方法可供选择。

（1）通过 resize 属性禁止对元素进行缩放：

```
textarea{ resize:none;}
```

（2）限制文本框的最大以及最小宽、高：

```
textarea{
  max-height:100px;
  min-height:100px;
  height:100px;
  max-width:200px;
```

```
    min-width:200px;
    width:200px;
}
```

注意：此方法不能去掉右下角可拖动样式。

6.5 实战演练——CSS 3 边框效果

CSS 3 的边框圆角和边框阴影属于浏览器支持度比较好的特效，只有 IE 6~IE 8 不支持，而且其效果可以采用 IE 的滤镜效果进行一定程度上的模拟，因此应用的普及度很高，下面就来看一些 CSS 3 边框效果的实用案例。

6.5.1 边框圆角在 Bootstrap 和淘宝网中的应用

边框圆角在实际案例中有大量的应用，比如大名鼎鼎的 Bootstrap 框架，如图 6.12 所示，Bootstrap 的按钮、表单、区域划分等常用元素都采用了边框圆角，从心理学角度来说，圆体有圆润、平滑的感觉，比较贴合用户心理，较为锐利的角度则有可能会给用户带来刺痛感和抵触情绪。

再比如实际生活中还有苹果官网、Gmail 邮箱、淘宝网等大量的圆角应用案例。以淘宝网的局部区域为例，如图 6.13 所示。

图 6.12 Bootstrap 中大量应用边框圆角

图 6.13 淘宝对圆角边框的运用

可以发现按钮、icon 的边框都运用了圆角，头像部分则通过使用 50%的边框圆角，使原来正方形的头像呈现为圆形。

6.5.2 边框阴影在苹果官网中的应用

边框阴影效果最常见的用例是构造一个模拟凸起的区域，使得这个区域看上去有一定

的立体感，比如苹果中文官网的局部设计，如图 6.14 所示。

图 6.14　边框阴影在苹果官网的应用

由图 6.14 可以看到，苹果官网采用了浅灰色的底色，内容区域内部为白色，如果只是简单采用深灰色的边框来对不同区域进行划分，就会显得很不清晰，而应用边框阴影构造出凸起效果后就解决了这个问题。

苹果官网采用的阴影样式为：

```
box-shadow: 0 1px 3px rgba(0,0,0,.35);
```

在实际应用中，一般可以采用外阴影来制作凸起效果，内阴影来制作内凹效果，例如图 6.15。左图效果类似苹果官网效果，右图的相框并非图片，而是完全通过 CSS 的边框内阴影制作的效果。

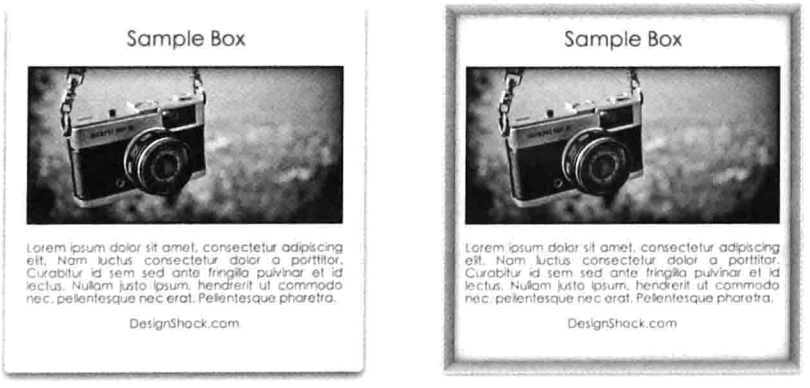


图 6.15　边框阴影效果实例

6.6　小结

本章主要对 CSS 3 中边框的新特性进行了详细的解析，我们可以利用这些效果来取代之前只能通过图片才能达成的效果。对网站优化有所了解的读者可能会知道，网站性能的

瓶颈之一就在于网络通信。图片会增加 HTTP 连接数，并且占用大量的带宽，而采用 CSS 3 特性取代图片，对网站的前端性能是一个质的飞跃。

本章主要介绍了 CSS 3 中新的边框效果，主要包括边框圆角、边框阴影、图片边框 3 大块内容。主要有以下几个要点：

- 可以为边框的 4 个角设置不同的圆角。
- 可以采用百分比设置圆角，其中值大于等于 50%时，正方形元素将呈现为圆形。
- 一般可以边框外阴影展现凸起效果，内阴影展现内凹效果。
- 图片边框目前由于浏览器兼容性以及对图片的依赖性，目前并不推荐使用。
- textarea 等可以设置 overflow 属性的元素，在 webkit 核心浏览器下可以通过拖拽改变其大小，当然也可以通过 resize:none 禁用这种拖拽。

第 7 章　变换和动画

页面上的动画效果曾经是 Flash 一统天下，不过随着 HTML 5 和 CSS 3 出现以及浏览器的发展，Flash 用户众多的优势正慢慢失去，关键原因就是 Flash 必须安装插件、学习门槛高、系统资源消耗大。目前简单的动画使用 CSS 3 加一些 JavaScript 就可以实现，而复杂的动画则可以应用 HTML 5 的<canvas>标签来实现。

对于大多数非游戏类的站点来说，需要的都是简单动画，因此 CSS 配合 JavaScript 的动画基本可以满足大多数网页的需求，CSS 3 的新特性可以让动画的选择更为丰富。

本章主要内容有：

- CSS 3 的变换（transform）属性。
- CSS 3 的渐变（transition）效果。
- 使用关键帧（@keyframes）制作不依赖 JavaScript 的动画效果。

7.1　CSS 3 的变换类型

CSS 3 的变换（transform）属性主要有两个作用：一是构建一些 CSS 2 中难以构造的图形，可以把一些过去由 Photoshop 做的工作交给浏览器渲染本身；二是配合 JavaScript 制作更为丰富的动画。

transform 的兼容性如下：

- IE 10、Firefox、Opera 支持 transform 属性。
- IE 9 支持替代的-ms-transform 属性（仅适用于 2D 转换）
- Safari 和 Chrome 支持替代的-webkit-transform 属性（3D 和 2D 转换）
- Opera 只支持 2D 转换

7.1.1　rotate 旋转变换

transform 属性有很多选项，rotate 是比较常用的一种，rotate 英文是轮转、回转的意思，在 CSS 3 中用于元素的旋转，如图 7.1 所示。

图 7.1 展示了最简单的 2D 旋转，其实现代码如下：

```
div{
  transform:rotate(7deg);
```

```
  -ms-transform:rotate(7deg);       /* IE 9 */
  -moz-transform:rotate(7deg);      /* Firefox */
  -webkit-transform:rotate(7deg);   /* Safari 和 Chrome */
  -o-transform:rotate(7deg);        /* Opera */
}
```

图 7.1　旋转图片

为 rotate 传入一个角度值作为参数，元素将沿中轴线顺时针偏转这个角度值。如果使用 rotateX(角度值)则表示沿 x 轴方向上旋转，同样地，使用 rotateY(角度值)表示沿 y 轴方向上旋转，这里以 rotateX 为例：

```
div{
  transform:rotateX(60deg);
  -ms-transform:rotateX(60deg);      /* IE 9 */
  -moz-transform:rotateX(60deg);     /* Firefox */
  -webkit-transform:rotateX(60deg); /* Safari 和 Chrome */
  -o-transform:rotateX(60deg);       /* Opera */
}
```

rotateX 的 3D 视图如图 7.2 所示。

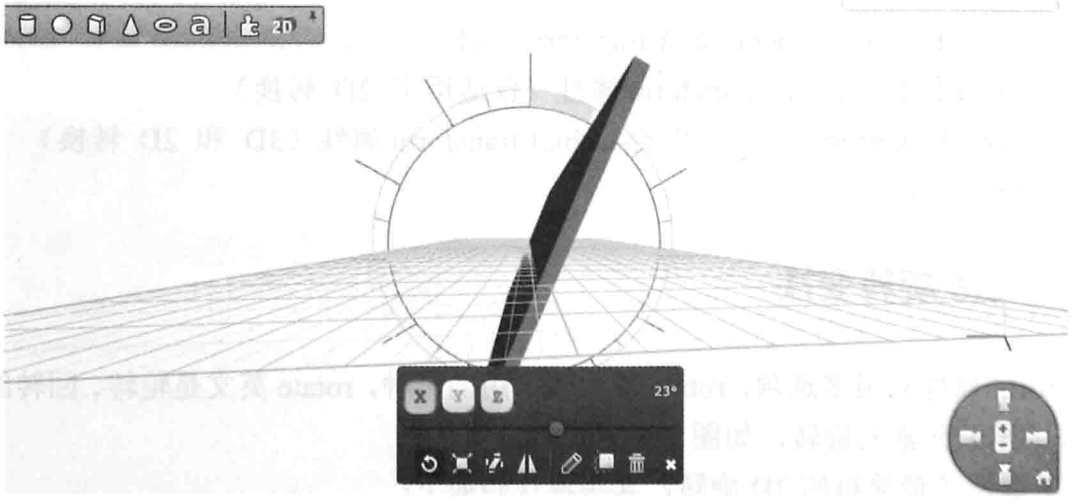


图 7.2　rotateX 的 3D 视图

如果将 rotateX 用在网页上，则效果如图 7.3 所示。

图 7.3　网页上的 rotateX 效果

从图中可以发现，在平面的感官上，对于空白的区域来说，应用 rotateX 只是让图形高度变低了，上下边框变细了，但是一旦在元素中加入图片或文字，就很容易发现其中的差异。

注意：rotateZ 等效于 rotate。

如果需要在其他向量上应用旋转，可以使用 rotate3d(x, y, z, deg)，编辑 x、y、z 的值构建三维向量，最后一个参数是偏转的角度，例如：

```
.trans_3d{
  transform:rotate3d(1,-1, 0, 60deg);
  -moz-transform:rotate3d(1,-1, 0, 60deg);
  -ms-transform:rotate3d(1,-1, 0, 60deg);
  -o-transform:rotate3d(1,-1, 0, 60deg);
  -webkit-transform:rotate3d(1,-1, 0, 60deg);
}
```

上述代码的最终效果如图 7.4 所示。

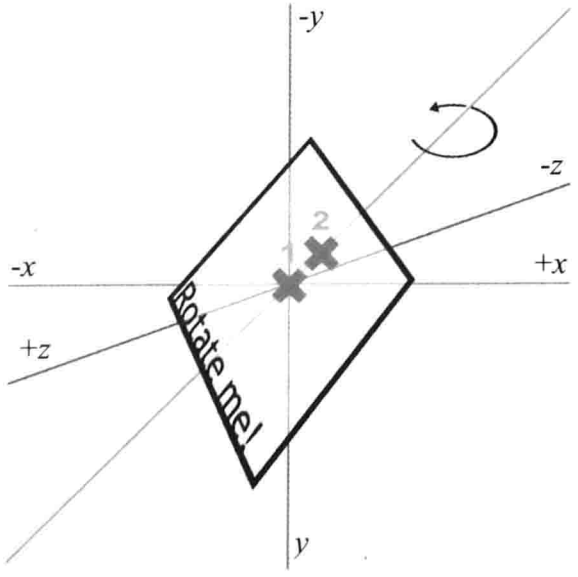


图 7.4　rotate3d 效果

7.1.2 skew 扭曲变换

transform 的 skew 属性用于设置元素的扭曲变形，例如：

```
transform:skew(30deg,10deg);
```

这段代码表示将原图形在 x 轴方向上偏转 30°，在 y 轴方向上偏转 10 度，如图 7.5 所示。

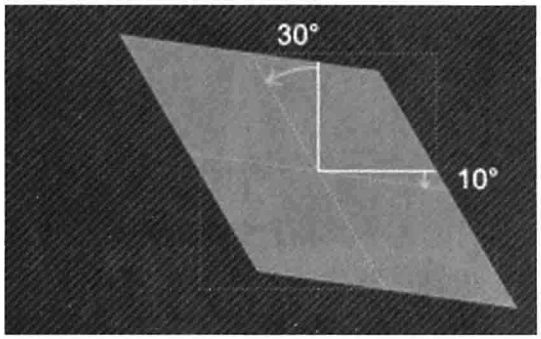


图 7.5 skew 变换

可以使用 skewX()、skewY()表示单一方向上的 skew 变换。

注意：Skew 没有 3D 和 skewZ 选项。

7.1.3 scale 比例缩放

scale 在英文中是尺寸的意思，在 transform 中用于尺寸缩放的控制，例如：

```
transform:scale(1.1, 1.1)
```

这段代码就表示在原尺寸基础上横向、纵向都放大到原来的 1.1 倍，如图 7.6 所示。左侧是原图，右侧是应用 scale 缩放后的样式。

图 7.6 scale(1.1, 1.1)效果

注意：可以使用 scaleX()、scaleY、scaleZ 做单一方向上的缩放，不过元素内的图片、文字等也会被拉伸，造成失真的情况。

7.1.4 translate 位移变换

transform 属性的 translate 参数定义了元素的位移，例如：

```
transform：translate(100px, 20px);
```

表示元素在 x 轴方向上向右位移 100 像素，在 y 轴方向上向下位移 20 像素，如图 7.7 所示。

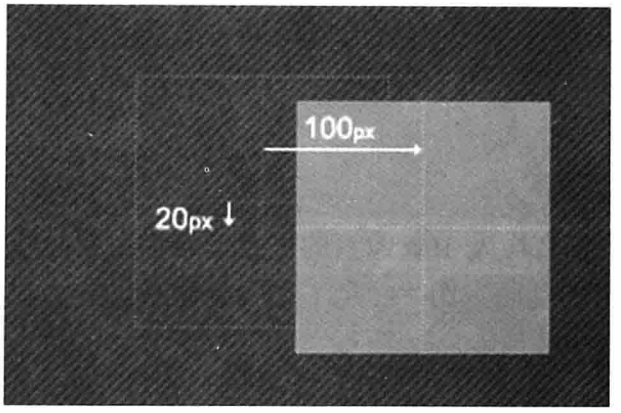


图 7.7 translate 效果图示

可以使用 translateX()、translateY()、translateZ()来表示单一方向上的位移。

7.1.5 transform 小结

transform 属性用于元素的变换操作，常用的选项有以下几个。

- 旋转：使用 rotate 选项，支持 3D。
- 扭曲：使用 skew 选项，不支持 3D。
- 位移：使用 translate 选项，支持 3D。
- 缩放：使用 scale 选项，支持 3D。

7.2 使用 transition 制作交互动画

transition，中文译作过渡，在 CSS 中表示属性渐进变化的效果。例如改变某个元素的宽度，从初始的 100 像素到 300 像素，jQuery 代码如下：

```
$(element).css('width', '300px');
```

执行结果就是该元素瞬间从 100 像素宽变为 300 像素宽，感觉非常突兀，在某些场景下，可能需要的是平滑渐进的动画效果，在 CSS 3 出现之前，可以使用 JavaScript 来实现，不过在 CSS 3 出现后，应用 CSS 3 的 transition 属性制作动画中的过渡效果是更好的选择。不仅相比 JavaScript 实现更容易，而且可以应用浏览器的硬件加速效果，实现更高的性能。

transition 允许 CSS 的属性值在一定的时间区间内平滑地过渡。这种效果可以在鼠标单击、获得焦点、被点击或对元素进行任何改变时触发，并圆滑地以动画效果改变 CSS 的属性值。比如下面的例子：

```
.content {
  height: 100px;
```

```
  width:100px
  -webkit-transition: height 0.6s;
  -moz-transition: height 0.6s;
  -o-transition: height 0.6s;
  transition: height 0.6s;
}
.content:hover{
  height:300px;
}
```

这个例子表现了一个宽高均为 100 像素的正方形元素在鼠标移上后 0.6 秒内，渐变为一个宽 300 像素的长方形的动画。图片经历了如图 7.8 所示的一个渐进的伸展过程，而不是宽度从 100 像素突变至 300 像素。

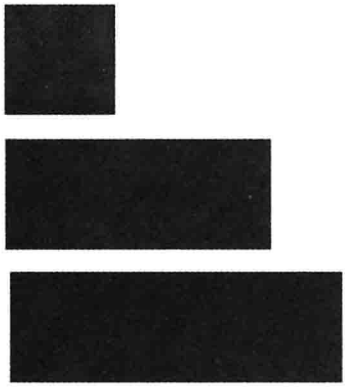


图 7.8　渐进的伸展过程

代码中的 transition: height 0.6s; 意思是高度如果发生改变，那么在 0.6 秒内完成平滑地过渡。如果鼠标移到元素上触发 hover 效果，就可以展现一个平滑过渡的动画。

如果需要同时改变多个属性，可以用逗号隔开，例如：

```
transition:background 2s, width 2s;
```

这样，如果颜色和宽度同时发生改变时，都会被加上过渡动画。

transition 还可以包含设置渐进动画的函数，可以选择的函数有 6 种。

- ease：（逐渐变慢）默认值，ease 函数等同于贝塞尔曲线(0.25, 0.1, 0.25, 1.0)。
- linear：（匀速），linear 函数等同于贝塞尔曲线(0.0, 0.0, 1.0, 1.0)。
- ease-in：(加速)，ease-in 函数等同于贝塞尔曲线(0.42, 0, 1.0, 1.0)。
- ease-out：（减速），ease-out 函数等同于贝塞尔曲线(0, 0, 0.58, 1.0)。
- ease-in-out：（加速然后减速），ease-in-out 函数等同于贝塞尔曲线(0.42, 0, 0.58, 1.0)。
- cubic-bezier：允许开发者自定义一个时间曲线。

transition 还可以设置动画的延迟时间，表示延迟多少秒后执行动画，一个添加了所有属性的 transition 定义如下：

```
transition: all .5s ease-in-out 1s;          /*all 表示任意属性的变化都应用过渡动画效果*/
```

4 个参数依次表示属性、过渡时间、过渡函数、延迟时间。

7.3　使用@keyframes 制作动画

根据之前的介绍，读者可能会问了：制作动画是不是必须依赖 JavaScript 呢？只使用 CSS 可不可以？答案是可以。CSS 3 的@keyframes（关键帧）可以帮助开发者们做到这一点。

7.3.1　@keyframes 的基本语法

@keyframes 的基本语法如下：

```
@keyframes spin {
  from {
    -webkit-transform: rotateY(0);
  }
  to {
    -webkit-transform: rotateY(-360deg);
  }
}

@keyframes spin {
  0% {
    -webkit-transform: rotateY(0);
  }
  50% {
    -webkit-transform: rotateY(-180deg);
  }
  100% {
    -webkit-transform: rotateY(-360deg);
  }
}
```

@keyframes 声明后紧跟动画的名称，花括号内部就是一些不同时间段的样式规则。@keyframes 必须配合元素中定义的 animation 属性，它用于定义动画，语法规则如下：

```
animation: spin 8s infinite linear;
```

第 1 个参数是动画名称。第 2 个参数是动画执行一次的时间。第 3 个参数是动画循环的次数，如果使用 infinite 则为无限循环。第 4 个参数是动画的速度函数，和 transition 中的速度函数一致。

此外还可以设置动画开始前的延迟时间、动画是否轮流反向播放等。完整的参数示例如下：

```
animation: name 8s linear 2s infinite alternate;  /* alternate 表示动画正向循环完成后再反向循环*/
```

现在看到的动画是不可控的，在页面上一直循环，无法停止和启动。实际上，我们可以为元素添加 animation-play-state 属性，例如：

```
animation-play-state: paused;
-webkit-animation-play-state: paused;
```

参数 paused 用于暂停动画，running 用于开启动画。这样就可以很容易地使用 JavaScript 来控制动画的启动和暂停了。

7.3.2 用@keyframes 制作循环动画

下面是一个使用@keyframes 制作动画的完整例子，因为是动画效果，这里暂且不给效果图，读者如果不明白，可以通过源代码来查看结果。

```
<!DOCTYPE html>
<html>
  <head>
    <style>
      div{
        width:100px;
        height:100px;
        background:red;
        position:relative;
        animation:myfirst 5s infinite;
        -webkit-animation:myfirst 5s infinite alternate;
      }

      @keyframes myfirst{
        0%   {background:red; left:0px; top:0px;}
        25%  {background:yellow; left:200px; top:0px;}
        50%  {background:blue; left:200px; top:200px;}
        75%  {background:green; left:0px; top:200px;}
        100% {background:red; left:0px; top:0px;}
      }

      @-webkit-keyframes myfirst {/* Safari and Chrome */
        0%   {background:red; left:0px; top:0px;}
        25%  {background:yellow; left:200px; top:0px;}
        50%  {background:blue; left:200px; top:200px;}
        75%  {background:green; left:0px; top:200px;}
        100% {background:red; left:0px; top:0px;}
      }
    </style>
  </head>
  <body>
    <div></div>
  </body>
</html>
```

这段代码实现了一个动画，效果是一个方块沿正方形轨迹移动并改变颜色，一次循环完成后，会反向循环，再正向循环，如此反复。

7.3.3 @keyframes 小结

总结@keyframes 的用法如下：

- 动画是使元素从一种样式逐渐变化为另一种样式。
- keyframes 可以改变任意多的样式、任意多的次数。
- 使用百分比来规定变化发生的时间，或用关键词 from 和 to，等同于 0%和 100%。
- 0%是动画的开始，100%是动画的完成。
- 为了得到最佳的浏览器支持，应该始终定义 0%和 100%选择器。

注意：IE 10、Firefox 以及 Opera 支持@keyframes 规则和 animation 属性。Chrome 和 Safari 需要前缀-webkit-。IE 9 以及更早的 IE 版本，不支持@keyframe 规则或 animation 属性。

7.4 实战演练——结合变换制作 3D 旋转卡片

本节将通过一个旋转卡片的实例介绍将变换效果组合成动画的过程，完成后的效果如图 7.9 所示，卡片会沿 y 轴进行立体旋转。

图 7.9 旋转卡片

首先要构造一个平面：

```
.rect {
   position: absolute;
   top: 0px;
   left: 0px;
   width: 200px;
   height: 200px;
   border-radius: 5px;
   border: 1px solid #ccc;
   font-size: 125pt;
   text-align: center;
   line-height: 200px;
   background-color: #bbb;
   opacity: 0.5;
}
<div class= "rect" >6</div>
```

上一步完成后就是一个正方形的平面，现在给它加上 animation 属性：

```
.rect {
```

```
  /*……刚才完成的 style……*/
  animation: rotate 8s infinite linear;          /*每个循环 8 秒完成，匀速运动，无限循环*/
  -webkit-animation: rotate 8s infinite linear;  /*weblit 核心浏览器兼容代码*/
}
```

最后一步就是在样式中使用@keyframes 让动画跑起来。

```
@-webkit-keyframes rotate {
  from { -webkit-transform: rotateY(0); }
  to { -webkit-transform: rotateY(-360deg); }
}
@keyframes rotate {
  from { -webkit-transform: rotateY(0); }
  to { -webkit-transform: rotateY(-360deg); }
}
```

7.5 可参考的 CSS 动画资源

由于很难在纸介质上描述动画效果，本节将给读者介绍一些优秀的 CSS 3 动画资源，这样读者就可以通过互联网查看这些动画类库的效果，并结合之前的理论知识来分析它们的具体实现。

7.5.1 Hover.css——鼠标 hover 动画

Hover.css 集成了丰富的简单鼠标 hover 动画效果，比较适合按钮、提示等应用场景。由于效果简单实用，比较适合初学者跟进学习，项目主页地址是 http://ianlunn.github.io/Hover/，项目源代码托管在 Github（目前最流行的代码托管网站），可以访问 https://github.com/IanLunn/Hover 进行查看或下载，其主页效果如图 7.10 所示。

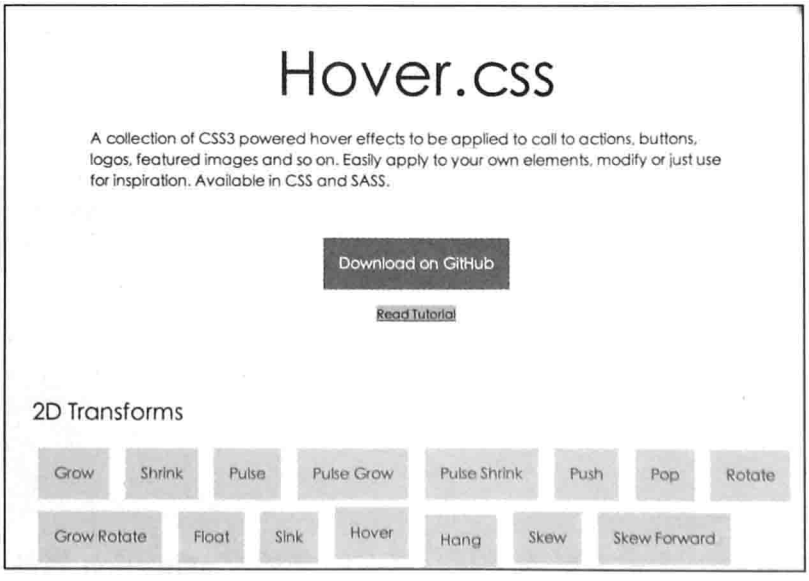


图 7.10 Hover.css 项目主页

7.5.2　iHover——hover 动画类库

iHover 是一个相比 Hover.css 更复杂的 hover 动画类库，项目主页地址是：http://gudh.github.io/ihover/dist/index.html，图 7.11 是项目主页的截图。

图 7.11　iHover 项目主页

iHover 包含了多达数十种美观实用的 hover 效果，不过看似惊艳的效果其实质就是设置元素常态和 hover 时的 CSS 效果，然后通过设置 transition 渐变达成连续的动画效果。

图 7.12 展示了 iHover 部分动画效果执行前后的对比。

图 7.12　iHover 部分效果对比

iHover 的是一个开源项目，托管在 Github 上，可以访问 https://github.com/gudh/ihover 来查看或下载源代码。

7.5.3 CSS 3 和 JavaScript 的结合

由于 CSS 本身只能提供 hover、visited 等有限的几种触发条件，对于一些更为复杂的应用场景，就必须借助于 JavaScript 了。

CSS 3 和 JavaScript 结合可以更为有效地控制动画的触发、结束、重启。Move.js 是一个封装了简单 CSS 3 动画效果的 JavaScript 插件，项目主页地址是 http://visionmedia.github.io/move.js/，如图 7.13 所示。

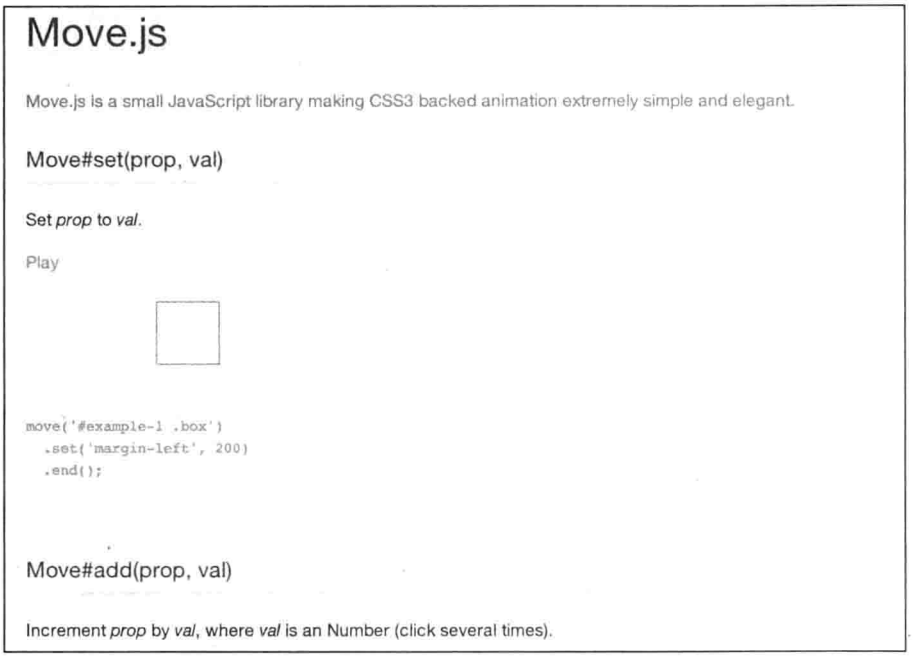


图 7.13 Move.js 项目主页

可以单击项目主页上的“play”链接查看相应的动画效果，Move.js 同样托管在 Github 上，可以访问 https://github.com/visionmedia/move.js 查看或下载源代码。

7.6 小结

本章主要介绍了如何应用 CSS 3 构造页面元素的变换和动画效果，主要包括了以下内容：

- 元素的变换，应用 transform 属性可以对元素进行旋转、扭曲、移位、缩放。
- 元素样式改变的过渡效果，应用 transition 属性可以为元素状态的改变添加过渡效果，可以应用 ease（逐渐变慢）默认值、linear（匀速）、ease-in (加速)，ease-out（减速）、ease-in-out（加速然后减速）这 5 个内置的函数来设置过渡的速度，还可以使用 cubic-bezier 自定义速度曲线。
- 使用@keyframes（关键帧）和 animation 设置动画循环，为元素设置 animation 属性，指定动画的名称、速度函数、运行时间、循环次数等要素，使用@keyframes 描述关键帧的样式，使用 animation-play-state 属性暂停/开启动画。

第 8 章　媒介查询和响应式设计

移动互联网发展势头迅猛，正成为人们日常生活中不可缺少的组成部分。越来越多的用户在手机、平板上阅读新闻、观看视频、玩游戏、购物，这些原本只能在 PC 上完成的事情现在用移动设备完成得更轻松，而且随时随地可以完成。

为了适应移动互联网大潮，传统的网页技术也在不断发展，目前，响应式设计是最为流行的方式。本章将主要介绍响应式设计最最基础的技术：CSS 3 的媒介查询（media query）。

本章主要知识点如下：

- 认识媒介类型。
- 掌握媒介查询。
- 了解移动设备的差异。

8.1　媒介类型=各种浏览终端

这里提到的媒介，不同于新闻传播学对媒介的定义，而是指我们浏览内容所使用的各种电子设备。在 CSS 2 标准中就已经可以根据不同的媒介类型（Media Type）来设置不同的输出样式了。@media 规则使开发者有能力在相同的样式表中，针对不同的媒介来使用不同的样式规则。

下面这个例子中的样式告诉浏览器在显示器上显示 14 像素的 Verdana 字体，但是假如页面需要被打印，将使用 10 像素的 Times 字体。

```
<html>
  <head>
    <style>
        @media screen{
          p.test {font-family:verdana,sans-serif; font-size:14px}
        }
        @media print{
          p.test {font-family:times,serif; font-size:10px}
        }
        @media screen,print{
          p.test {font-weight:bold}
        }
</style>
```

```
  </head>
  <body>....</body>
</html>
```

表 8.1 是 Media Type 的详细类型，不过使用比较多的是 screen 和 print 类型，用于区分打印和屏幕显示。

表 8.1　Media Type的类型汇总

设备	指代
all	匹配所有设备
braille	匹配触觉反馈设备
embossed	凸点字符印刷设备
handheld	手持设备（尤其是小屏幕，有限带宽，PSP、NDS这种规格一般可以叫作handheld）
print	打印机设备
projection	投影仪设备
screen	彩色计算机显示器设备
speech	语音合成器设备
tty	栅格设备（终端或电传打字机）
tv	电视设备

注意：现在的 Android、iPhone 都不是 handheld 设备，它们都是 screen 设备。所以，不要试图用 handheld 来识别 iPhone、iPad 或 Android 等设备。

通过表 8.1 可以看出，@media 只能做一个大概的区分，而现在桌面和移动设备拥有不同的分辨率，即使是同样类型的设备，也可能需要做出不同的适配。所以仅仅依靠 Media Type 已经无法满足时代的要求了。为了顺应这种需求，CSS 3 引入了 Media Query（媒体查询）。

注意：虽然@media 在 CSS 2 时代就已经引入，但是只有 Opera 支持 handheld 属性。

8.2　认识响应式网页设计

响应式网页设计就是一个网站能够兼容多个终端——而不是为每个终端做一个特定的版本。页面有能力去自动响应用户的设备环境。

响应式设计是一项不折不扣的“技术驱动”型设计模式，虽然对于设计师来说，把握响应式设计中的交互模式、色彩运用是一件颇费思量的事情，但是响应式设计本身是来源于移动互联网技术的兴起和新的 CSS 3 技术，没有这些，一切好的想法都是镜花水月。

下面就来简单认识一下响应式设计并了解响应式设计需要遵循的一些模式。首先来看图 8.1 和图 8.2，这是两个典型的响应式设计案例，读者可以直观地感受一下。

图 8.1　响应式设计案例 1

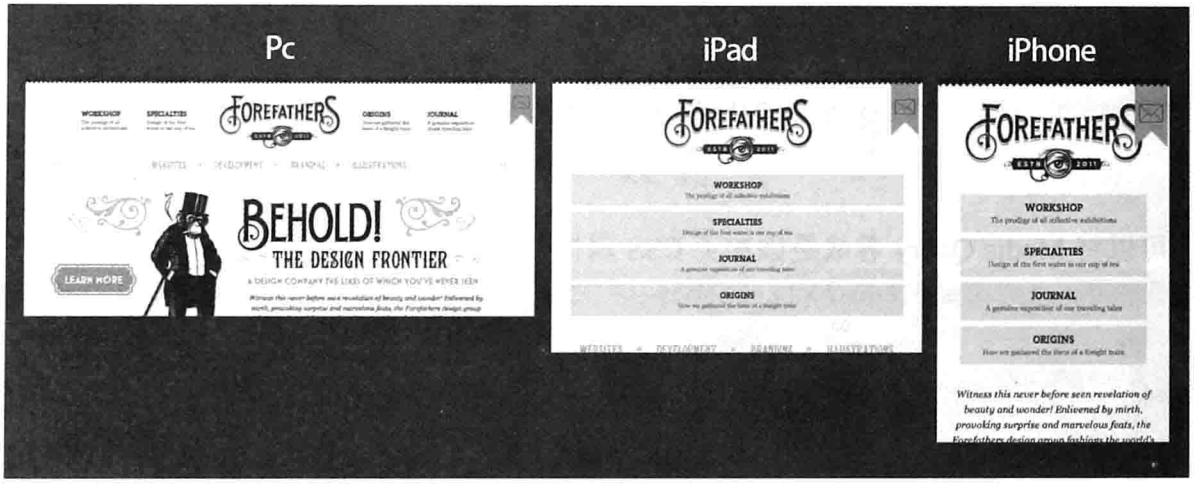


图 8.2　响应式设计案例 2

读者可能已经发现，响应式设计并不是同样内容的等比例缩小，也不像之前流行的 WAP 网站一样，和 PC 端差异巨大。响应式设计在设计风格和色彩搭配上保持了很大的一致性，又根据移动设备的特点对布局进行了调整。

响应式设计一般需要遵循 4 个原则，如图 8.3 所示。

图 8.3　响应式设计原则

1．小屏幕只显示高优先级内容

在响应式网页设计中，切换到小屏幕移动设备时，有时候需要对页面内容进行删减，按照优先级显示内容，只显示高优先级内容是原则之一。在屏幕较小的移动设备上应该优先考虑主要内容并移掉那些小的栏目。在顶部显示高优先级内容，即把最重要的内容放置在顶部。导航是否一定要出现在页头或者重新布局在页尾，都要依网站具体规划去考虑。

2．提供清晰和友好的手指操作链接

尤其在手机设备上，可点击操作的区块不宜过小，引导要清晰强烈，不忽略任何一款设备。

3．体验的一致性

要让用户在不同的设备上仍保持对同一页面相同的视觉和感觉，这也遵循交互设计体验一致性的原则。读者可以参考 Oliver Russell 网站，一个设计非常灵活的网站，在不同的屏幕分辨率下可以保持一致的视觉和感受。

4．考虑操作移动设备的习惯

大部分用户习惯于右手点击操作，左手负责握住设备。右侧的导航列表既方便右手的点击，又可以避免被握着设备的左手不小心触碰到。

8.3 媒介查询的基本语法

可以将 Media Query 看成是添加了 CSS 属性判断的 Media Type，其基本语法如下：

```
@media screen  and (max-width: 600px) {
.class {
  background: #ccc;
  }
}
```

这段代码定义了小于 600 像素的窗口所应用的样式。前半部分@media screen 和 Media Type 的语法是一样的，一般来说选择 screen 或 only screen，因为所有现代的智能手机、平板电脑、PC 在类型上都是 screen。后半部分使用 and 作为条件添加符号，可以使用多个 and 添加多个条件：

```
@media screen and (min-width: 600px) and (max-width: 900px) {
  .class {
    background: #333;
  }
}
```

这段代码表示宽度在 600~900 像素之间的窗口应用该样式。width 作为条件是最常用、最基本的，根据我们的需求，还可以限定更多的条件来更精确地对设备进行适配。比如通过 orientation 来判断设备翻转、通过 device-aspect-ratio 来判断屏幕的纵横比等。表 8.2 列出了可以使用的一些判断条件。

表 8.2　适配设备的判断条件

媒体特性	说明/值	可用媒体类型	接受min/max
width	窗体宽度	视觉屏幕/触摸设备	是
heigth	窗体高度	视觉屏幕/触摸设备	是
device-width	屏幕宽度	视觉屏幕/触摸设备	是

续表

媒体特性	说明/值	可用媒体类型	接受min/max
device-heigth	屏幕高度	视觉屏幕/触摸设备	是
orientation	设备手持方向（portrait横向/landscape竖向）	位图介质类型	否
aspect-ratio	浏览器、纸张长宽比	位图介质类型	是
device-aspect-ratio	设备屏幕长宽比	位图介质类型	是
color	颜色模式（例如旧的显示器为256色）整数	视觉媒体	是
color-index	颜色模式列表整数	视觉媒体	是
monochrome	整数	视觉媒体	是
resolution	解析度	位图介质类型	是
scan	progressive逐行扫描/interlace隔行扫描	电视类	否
grid	整数，返回0或1	栅格设备	否

8.4　设备

既然响应式设计主要针对的是移动互联网时代形形色色的各种终端设备，那么作为网页的开发者或者设计者，当然需要对它们有所了解，图 8.4 展示了一些常见的终端。

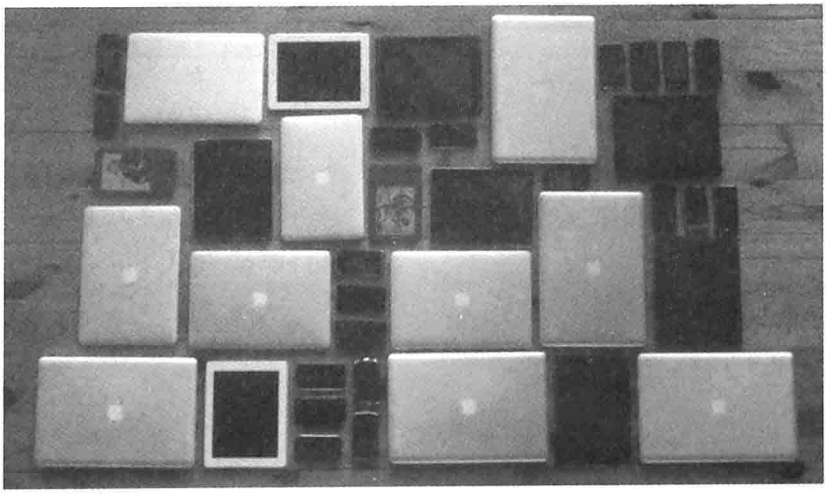


图 8.4　形形色色的终端设备

8.4.1　常见设备的宽度和高度

要实现响应式设计，必须要了解常见设备的宽度和高度（单位均为像素）：

- 老式的 4:3 屏幕的 PC 最小的分辨率是 1024×768（800×600 的机器基本已经被淘汰）。
- 普通的 14 英寸笔记本分辨率是 1366×768。
- iPad mini 和 iPad2 分辨率是 1024×768。
- iPad3 以上分辨率是 2048×1536（视网膜屏，开发中看作是 1024×768）。
- 7 英寸安卓平板的分辨率：从 800×480 到最新 Nexus7 的 1920×1200。

- iPhone4/4s 分辨率为 640×720（视网膜屏，开发中看作是 320×480）。
- 其他安卓 3.5 英寸智能机分辨率多为 320×480。
- 目前流行的 5 英寸+大屏手机分辨率：从 400×640 到 1920×1080。

注意：视网膜设备引入了双倍像素密度，不过浏览器的用户代理仍然报告原来的设备宽度，以免破坏网页。安卓的超高分屏幕也采用了类似的处理方式。

因为浏览器会根据设备的 DPI（每英寸像素数），将高分屏实际分辨率换算成普通中分屏的分辨率（和 CSS 中定义的像素等价），所以就可以根据像素数大致将设备的尺寸归为图 8.5 所示的 3 个边界。

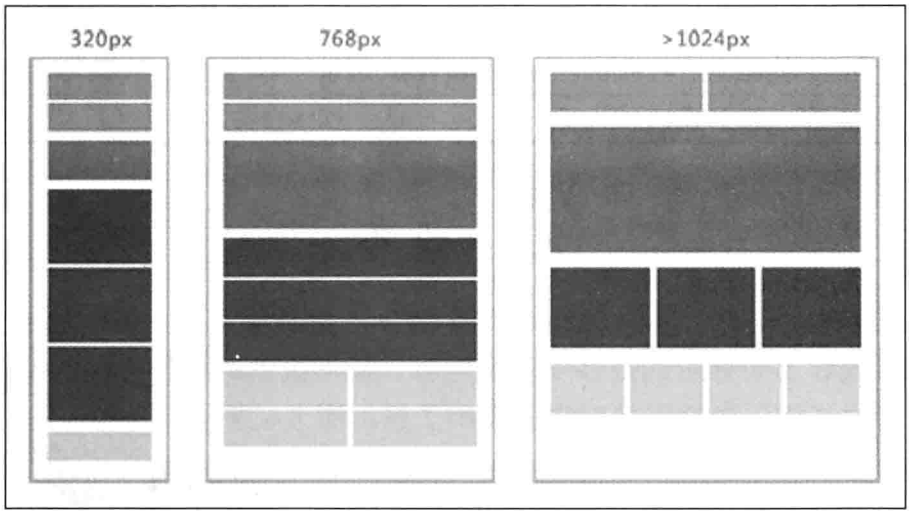


图 8.5　大致归类的设备尺寸

320 像素及以下的分辨率可以归为小屏幕手机，大屏手机的宽度则介于 320px~768px 之间。而 768px 和 1024px 之间则是平板电脑的主要分辨率，1024 及以上则主要是 PC 设备的分辨率。

8.4.2　检测设备翻转

大部分的手机和平板都是支持转屏的，因此在做响应式设计时，转屏是一个必须被考虑进去的因素。在开发中，我们需要做两个工作：

- 为横竖屏准备不同的样式。
- 使用 JavaScript 在屏幕切换时做出判断。

下面的例子分别为竖屏和横屏引入了两个不同的 CSS：

```
<link rel="stylesheet" media="screen and (orientation:portrait)" href="portrait.css" />
<link rel="stylesheet" media="screen and (orientation:landscape)" href="landscape.css" />
```

监听设备翻转的 JavaScript 代码：

```
window.addEventListener("orientationchange",function(obj){
.....//这里编写触发屏幕转换时的函数
});
```

这是不依赖任何第三方 JavaScript 库时的常规做法，在实际开发中，我们往往使用第

三方库提供的方法，一般语义会更简洁并有更多的兼容性优化，不过这就需要读者根据自己选择的库或者框架去阅读它们的文档了。

8.5　实战演练——应用媒介查询制作响应式导航栏

下面本书将构建一个简单的导航样例来了解一下媒介查询的具体应用。

注意：一般在实际应用中，只有简单页面才会手写媒介查询，复杂页面往往会采用各种响应式的框架来简化和规范开发。

（1）我们需要添加几个<meta>标签。

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
/*使用 viewport meta 标签在手机浏览器上控制布局*/
<meta name="apple-mobile-web-app-capable" content="yes" />
/*通过快捷方式打开时全屏显示*/
<meta name="apple-mobile-web-app-status-bar-style" content="blank" />
/*隐藏状态栏*/
<meta name="format-detection" content="telephone=no" />
/* iPhone 会将看起来像电话号码的数字添加电话连接，应当关闭*/
```

（2）为了让 IE 9 以下浏览器能够支持响应式设计，可以加上一个兼容性的 JavaScript 库，目前比较流行的有 media-queries.js 或者 respond.js。

```
<!--[if lt IE 9]>
<script
src="http://css3-mediaqueries-js.googlecode.com/svn/trunk/css3-mediaqueries.js"></script>
<![endif]-->
```

（3）构建 html 结构并为其编写样式。

```
<!DOCTYPE>
<html>
  <head>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="apple-mobile-web-app-capable" content="yes" />
    <meta name="apple-mobile-web-app-status-bar-style" content="blank" />
<meta name="format-detection" content="telephone=no" />
/*这部分是我们刚才提到的 meta 标签*/
<style>
      body{margin: 0}
      .container{width:80%;margin:auto; }
      .header{background-color: #333;}
      li a{color:white;}
 /*这部分是公共的样式，比如一些颜色的定义等*/
      @media screen and (max-width:320px){
        .logo{height: 40px}
        .header{height:40px;}
```

```
        li{
          line-height: 50px;
          padding:0 15px 0 15px;
          display: block;
          background-color: #333;
          text-align: center;
          border-top:1px solid white;
        }
          .logo{display:block;}
      }
  /*这里定义了窗体宽度在 320px 以下的样式*/
      @media screen and (min-width:320px) and (max-width: 765px){
        .logo{height: 50px}
        .header{height:50px;}
        li{
          line-height: 50px;
          padding:0 15px 0 15px;
          display: block;
          background-color: #333;
          text-align: center;
          border-top:1px solid white;
        }
        .logo{display:block;}
      }
  /*这里定义了窗体宽度 320px 到 765px 的样式*/
      @media screen and (min-width:765px){
        .logo{height: 60px}
        .header{height:60px;}
         li{display: block; line-height: 60px; float:left; padding:0 15px 0 15px;}
        .logo{display:block; float:left;}
      }
  /*这里定义了窗体宽度 765px 以上的样式*/
    </style>
  </head>
  <body>
    <div class="header">
      <div class="container">
        <img class="logo" src="http://l.ruby-china.org/assets/text_logo-3609989243456a4c620bf2342986b638.png"/>
        <li><a href="#">热门帖子</a></li>
        <li><a href="#">精华帖子</a></li>
        <li><a href="#">最新原创</a></li>
        <li><a href="#">文档翻译</a></li>
      </div>
</div>
/*这里是导航栏的 HTML 结构*/
```

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

这样，一个简单的响应式导航栏就完成了。在 PC 上的显示效果如图 8.6 所示，在手机上的效果如图 8.7 所示。

图 8.6　PC 上的导航栏

图 8.7　手机上的导航栏

8.6　小结

我们正在跑步进入移动互联网时代，所以针对移动环境下 Web 开发的响应式设计成为发展的一个趋势。本章主要介绍了响应式设计的媒介查询（Media Query）技术，总结了响应式开发过程中可能会遇到的几个问题：

- 如何应用媒介查询。
- 响应式设计的基本规则。
- 如何解决兼容性问题。
- 不同设备的尺寸总结。
- 设备翻转的处理方法。

实际开发中，我们很少不依赖框架直接编写响应式代码，但原理是共通的，后面的章节会详细地介绍响应式框架在实战中的运用。

第 9 章
更简便的布局——弹性盒子

弹性盒子（box-flex）是 CSS 3 中新加入的一种布局模式，相比传统的使用浮动的布局模式来说，它更为简单易用，而且不存在浮动元素脱离正常文档流后需要在某些地方清除浮动的问题。比较遗憾的是，目前这个属性只有部分浏览器支持，再加上有成熟的替代方案，因此在 PC 端开发中应用得还比较少。不过在移动端，webkit 核心浏览器几乎一统天下，在移动端 Web 界面的制作上，使用弹性盒子是非常不错的选择。

本章主要知识点如下：

- 认识弹性盒子。
- 掌握弹性盒子的语法。
- 使用弹性盒子操作元素。

9.1 认识弹性盒子

在弹性盒子出现之前，要实现一个两侧自适应、中间定宽的三栏布局（图 9.1）会怎么做？可行的解决方案非常多，而且各有各的优势和不足。比如使用 table 布局，自适应性很好，但是灵活性、语义性都不好，代码量又偏大；使用绝对定位，自适应性又不太好；使用浮动的话，中间的主要内容会被最后加载，用户体验存在问题。

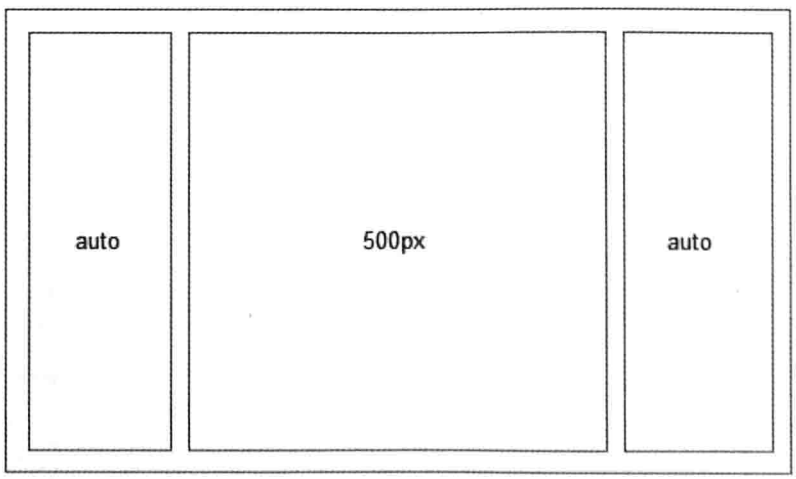


图 9.1　两侧自适应、中间定宽的三栏布局

使用弹性盒子的话，开发者只需要牢记一种方法就可以灵活地进行页面布局，而且除了兼容性问题外，弹性盒子相比其他任何一种方法都更简单易用，更容易理解。

注意：弹性盒子最早从 2009 年开始引入，但是在 2011 年进行了一次重新修订，语法出现了一些变化。不过目前各大浏览器厂商仍然采用较早的标准，下面的介绍采用目前可用的标准。

9.2　弹性盒子的语法

弹性盒子的基本语法非常简单，首先需要对父元素进行一个声明：

```
.father{display:box;}
```

有了这个声明，才能证明父元素的空间是可以被子元素分配的。假设要在这个父元素内部进行一个三栏布局，比例是 1:3:1，代码如下：

```
son1:{box-flex:1;}
son2:{box-flex:3;}
son3:{box-flex:1;}
```

这样一个 1:3:1 的自适应三栏布局就完成了。如果想采用中间固定宽度，两侧成比例自适应的布局，同样非常简单：

```
son1:{box-flex:1;}
son2:{width:500px;}
son3:{box-flex:1;}
```

只要为 son2 指定宽度即可，剩余的空间则会按比例分配给 son1 和 son3。

最后来看一个完整的实例：

```
<html>
<style>
  .container{
    margin:auto;
    display: -webkit-box;
    width:80%;              /*内容部分居中，宽度为窗口的 80%*/
    height:500px;
  }
  .left{-webkit-box-flex:3;background-color: red;}
  .main{width:600px;background-color: green;}  /*中间部分为固定的 600px 宽度*/
  .right{-webkit-box-flex:4;background: blue;}  /*左边栏和右边栏的比例为 3:4，占据剩余的空间*/
</style>
  <div class="container">
    <div class="left"></div>
    <div class="main"></div>
    <div class="right"></div>
  <div>
</html>
```

上面这段代码创建了一个内容占窗体居中 80%，中间部分固定 600px，左边栏和右边栏比例为 3：4 的布局。实例的效果如图 9.2 所示。

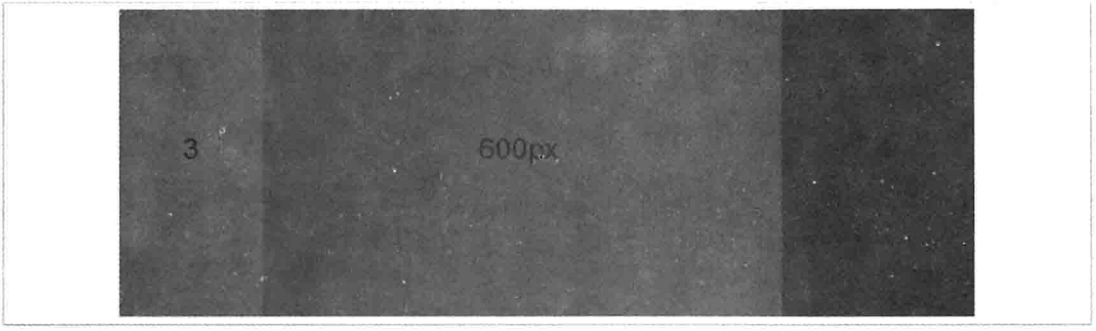


图 9.2　使用弹性盒子实现的布局

浏览器支持：目前 webkit 核心浏览器、火狐浏览器、IE 10 都支持带各自前缀的 box-flex 属性。Opera 和 IE 9 及之前版本的 IE 浏览器不支持弹性盒子。我们编写的兼容性代码应该类似这样：

```
.father{
   display:-weblit-box;
   display:-moz-box;
   display:-ms-box;          /*IE10 的兼容代码，-ms-前缀的较少见，需要注意*/
   display:box;
}
.son_content {
-webkit-box-flex: 4;
-moz-box-flex: 4;
-ms-box-flex: 4;
box-flex:4;
}
```

此外，因为弹性盒子使用一种不同的布局逻辑，一些属性会在弹性容器内部无效。

- 多列模块中的 column-*属性对弹性子元素无效。
- float 和 clear 对弹性子元素无效，使用 float 会导致 display 属性计算为 block。
- vertical-align 对弹性子元素的对齐无效。

9.3　操作元素

上一节主要介绍了弹性盒子的基本用法，当面对更复杂的需求（如需要垂直分配空间、需要页面先加载中间部分、需要控制剩余空间的使用）时，该怎么办呢？没关系，弹性盒子仍然能一一应对自如。

注意：本节介绍的这些属性都是应用于父元素上的。

9.3.1　控制子元素的方向

根据之前的例子，读者可能发现弹性盒子在默认情况下实现的是横向布局，可是如果需要纵向的布局怎么办呢？

可以在父元素中定义 box-orient 属性来确定子元素排列的方向。可选的值有 horizontal、vertical、inline-axis、block-axis、inherit。其中，inline-axis 是默认值，且 horizontal 与 inline-axis 的效果值一样的，让子元素横排；而 vertical 与 block-axis 的效果也是一致的，让元素竖排。inherit 和用在其他属性中是一样的，表示继承父元素。

图 9.3 分别展示了横向和纵向的 1:2:1 布局。父元素的代码如下：

```
.container1{
      margin:30px;
      display: -webkit-box;
      -moz-box-orient:inline-axis;
-webkit-box-orient:inline-axis;
-o-box-orient:inline-axis;
 box-orient:inline-axis;
      width:300px;
      height:300px;
  }
.container2{
      margin:30px;
      display: -webkit-box;
      -moz-box-orient:block-axis;
-webkit-box-orient:block-axis;
-o-box-orient:block-axis;
box-orient:block-axis;
    width:300px;
    height:300px;
  }
```

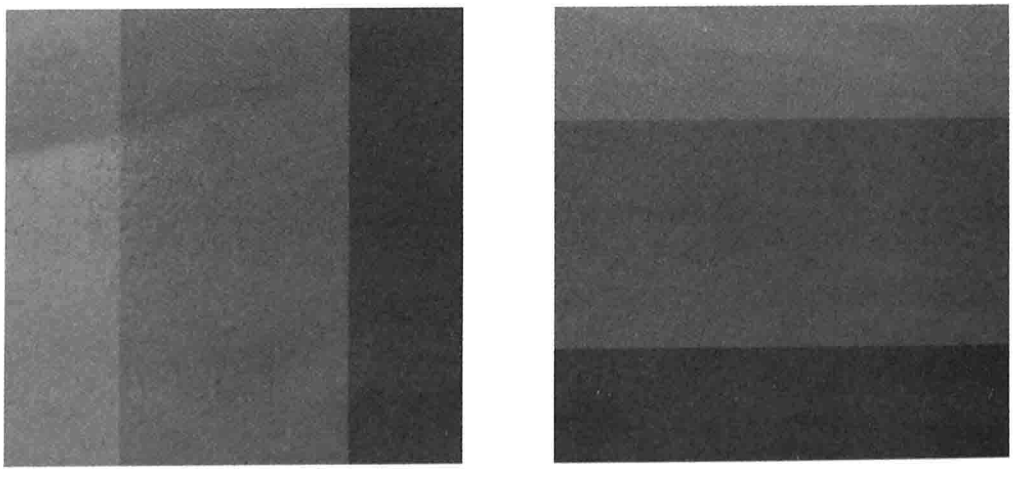


图 9.3　横向和纵向的 1:2:1 布局

9.3.2 控制元素对齐

box-align 与 box-pack 都能决定盒子内部的剩余空间怎么使用，在效果上类似我们平常说的“对齐”。

1. box-align

box-align 决定了垂直方向上的对齐表现，可选参数有 start、end、center、baseline、stretch。其中 stretch 是默认值，表示拉伸，也就是如果不规定子元素高度的话，子元素和父元素的高度一致。start 表示顶边对齐，end 为底部对齐，center 为居中对齐，baseline 表示基线（起始文字的底边位置线）对齐。具体表现如图 9.4 所示。

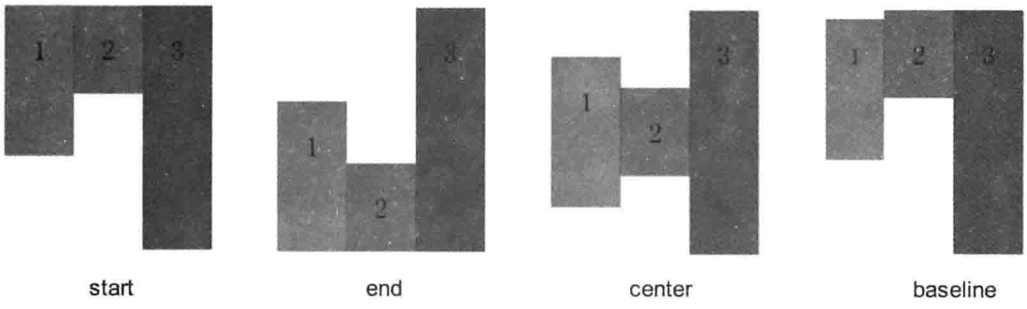


图 9.4 box-align 各参数的效果

图 9.4 对应的代码如下：

```
<style>
  body{font-size: 30px;}
  .container{
    margin:30px;
    display: -webkit-box;
    width:300px;
    height:300px;
-webkit-box-align:stretch;     /*在这里修改成 start、end、center、baseline 进行尝试*/
/*注意，实际使用中要添加兼容代码，这里为了清晰，只兼容 webkit 核心浏览器*/
  }
  .left{ padding:1em;height:100px;background-color: red;font-size: 25px;}
  .main{ padding:1em;background-color: green;}     /*不规定高度的话，在 box-align:strtch 时等于
    父元素高度*/
  .right{ padding:1em;height:200px;background: blue;}          /*规定高度生效*/
</style>
<div class="container">
  <div class="left">1</div>
  <div class="main">2</div>
  <div class="right">3</div>
</div>
```

2. box-pack

box-pack 决定了父标签水平遗留空间的使用，其可选值有 start、end、center、justify。

默认值为 start，具体的表现类似于 text-align 的 left、right、center、justify 属性。

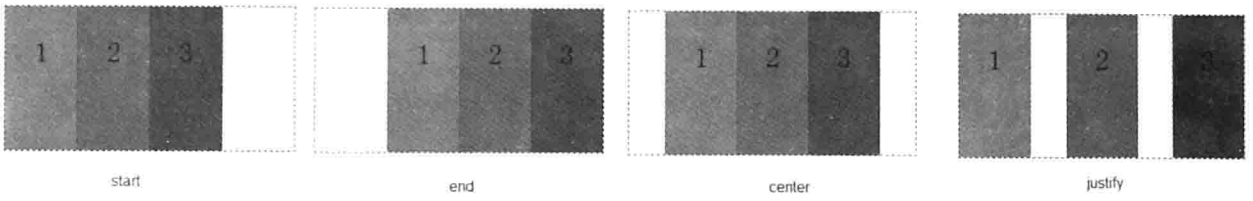


图 9.5　box-pack 各参数的效果

图 9.5 对应的代码如下：

```
<style>
  body{font-size: 30px;}
  .container{
    margin:30px;
    display: -webkit-box;
    width:300px;
    height:150px;
    border:1px dashed #333;
-webkit-box-pack:justify; /*在这里修改成 start、end、center 进行尝试*/
/*注意，实际使用中要添加兼容代码，这里为了清晰，只兼容 webkit 核心浏览器*/
  }
  .left{ padding:1em;background-color: red;}
  .main{ padding:1em;background-color: green;}
  .right{ padding:1em;background: blue;}/*这里子元素不能使用比例，否则将占满父元素的宽度*/
</style>
<div class="container">
  <div class="left">1</div>
  <div class="main">2</div>
  <div class="right">3</div>
</div>
```

9.3.3　控制元素显示顺序

假设有这样一个使用浮动（float）的三栏布局：

```
<style>
.left{float:left;}
.right{float:right;}
.center{width:600px}
</style>

<div class="left"> ..... </div>
<div class="right">..... </div>
<div class="center"> ..... </div>
```

由于 HTML 元素的加载是按顺序进行的，如果页面的内容比较多，这个布局的显示顺序就变为两侧先加载出来，展示主要内容的中间部分最后才加载。

应用弹性盒子能解决这个问题，可以使用 box-direction 这个属性来确定子元素的排列顺序，可选值有 normal、reverse、inherit。其中 normal 是默认值，表示按照正常顺序排列。所谓正常顺序，就是从左往右，由上至下，先出现的元素，就在上面或是左边。而 reverse 表示反转，原本从左往右应该是 1-2-3 的，使用了 reverse 后则变为 3-2-1，如图 9.6 所示。

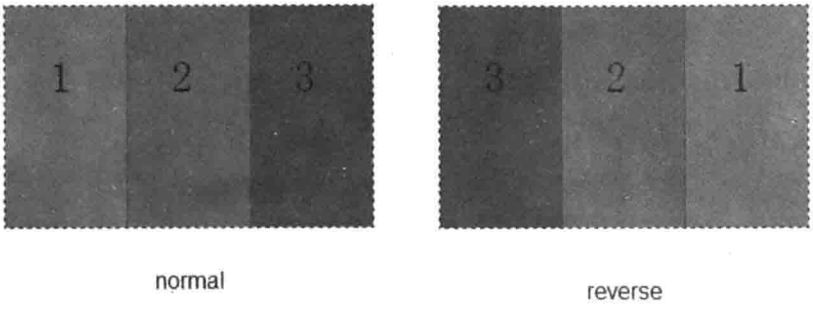


图 9.6　normal 和 reverse 显示顺序对比

图 9.6 的 CSS 代码如下：

```
<style>
  body{font-size: 30px;}
  .container{
     margin:30px;
     display: -webkit-box;
     width:250px;
     height:150px;
     border:1px dashed #333;
     -webkit-box-direction:reverse; /*这里修改为 nornal，观察改变*/
  }
  .left{ -webkit-box-flex:1;padding:1em;background-color: red;}
  .main{-webkit-box-flex:1;padding:1em;background-color: green;}
  .right{-webkit-box-flex:1;padding:1em;background: blue;}
</style>
<div class="container">
  <div class="left">1</div>
  <div class="main">2</div>
  <div class="right">3</div>
</div>
```

9.4　实战演练——用弹性盒子设计阅读 APP

布局对于任何网站都是慎之又慎的事，不像圆角、阴影等效果，即使不生效也只是影响视觉效果，布局一旦出现问题则会直接影响用户的正常使用。弹性盒子一方面不被老版本的 IE 所支持，另一方面由于经历过语法的修改，使一些现代浏览器也存在兼容性问题，因此目前还很难找到一个可以令人信服的现实中的实用案例。

这里，笔者自己制作了一个可以在实战中应用的小例子，仅供读者参考。应用环境是

一个 iOS 程序中的 webview（即苹果手机应用中调用的网页）。iOS 应用统一调用的是系统自带的 Safari 浏览器，这样就可以避免桌面或安卓手机环境下浏览器种类繁多，无法兼容的问题了。

对于阅读类应用来说，每一篇文章后面一般都有顶/踩的按钮，这里笔者就使用弹性盒子来实现这样的阅读 APP 效果。

（1）首先构建基本的 HTML 结构：

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0,
    user-scalable=no">
  </head>
  <body>
   <h1>……</h1>
   <p>……</p>
   <div class="vote">
      <a class="digg" href="#">顶</a>
      <a class="bury" href="#">踩</a>
    </div>
  <body>
</html>
```

（2）然后编写 CSS 代码：

```
.vote a{
  display: block;
  padding: 15px 0 15px 0;
  text-align: center;
  margin: 5px 15px;
  color: #fff;
  text-decoration: none;
  width:0;
  -webkit-box-flex:1;          /*子元素占据的比例一致*/
}

.vote{
  width:100%;
  display: -webkit-box;        /*对父元素设置弹性盒子模型声明*/
  -webkit-box-pack:justify;    /*设置子元素两端对齐*/
}

.digg{
  background: #B01B20;         /*设置“顶”按钮的颜色*/
}
```

```
.bury{
  background: #012753;          /*设置“踩”按钮的颜色*/
}
```

（3）完成后在模拟器中运行程序，可以看到如图 9.7 所示的效果。

有读者肯定会有疑问了，如果使用 float：left 对元素进行设置，或者使用行内元素设置按钮不是也可以达成一样的效果？

的确是这样的，不过使用弹性盒子的好处在于如果需要在同一行内新增按钮非常的方便。比如要在顶/踩按钮中间添加一个“中立”的按钮，如果采用其他方式，则不得不重新计算元素的尺寸，进行比较大的修改。而使用弹性盒子则只需要添加一个<a>链接，然后设置其颜色即可：

```
// HTML 代码
……
<div class="vote">
  <a class="digg" href="#">顶</a>
  <a class="mid" href="#">中立</a>   /*在原来的基础上添加链接*/
  <a class="bury" href="#">踩</a>
</div>
……

// CSS 代码
.mid{ background: #9AA7B3; }          /*设置“中立”按钮的颜色*/
```

最终的效果如图 9.8 所示。

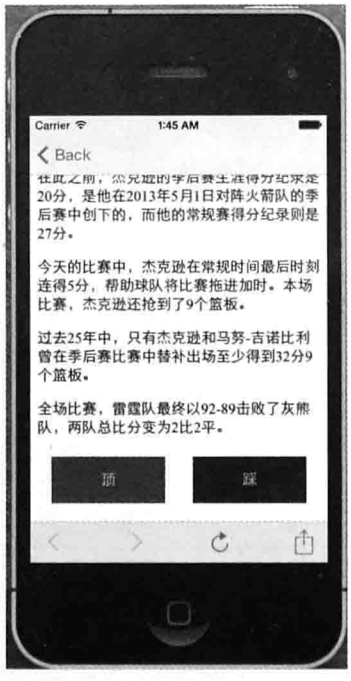


图 9.7　顶/踩按钮效果 1

图 9.8　顶/踩按钮效果 2

由这个例子可以发现，弹性盒子布局有很强的灵活性，不过限于兼容性的问题，目前在实际应用中还很少用到，不过对于一些手机端页面，尤其是针对种类较少的 iOS 设备来

说，应用弹性盒子不失为一个很好的尝试。

9.5　小结

本章主要讲解了关于弹性盒子的知识，这里做一个总结：

- 弹性盒子不像其他一些只影响效果的 CSS 3 属性，它主要应用在布局上，因此需要谨慎使用。
- 弹性盒子不仅可以制作水平的布局，也可以制作垂直的布局。
- 弹性盒子可以使用固定宽度，也可以使用比例，并且可以混合使用定宽和比例。
- 弹性盒子功能强大，但是由于兼容性问题，目前不适合在传统 Web 开发中使用，不过在移动开发方面有很好的前景。
- 使用弹性盒子需要对父元素进行声明，并且可以通过调整父元素的属性来控制子元素的方向、对齐、顺序等。

第 10 章　CSS 常用工具

在实际工作中，开发者或设计者常常需要应用一些工具来提高效率，减少一些不必要的重复劳动。

本章将介绍一些常用的生产力工具：

- 避免编写属性前缀的 Prefix free 插件。
- 让不同浏览器表现一致性的 Normalize。
- 应用 Grunt 自动合并压缩 CSS 文件。

10.1　使用 Prefix free 处理 CSS 3 跨浏览器兼容

随着 IE 浏览器开始全面拥抱 CSS 3 和 HTML 5 标准，现在有越来越多的网站开始在网站中加入 CSS 3 新特性，实现更给力的效果。但由于 CSS 3 和 HTML 5 的 W3C 规范都尚未定稿，各大浏览器对标准的支持还是有所差异的。所以在编写 CSS 3 代码时，还需要针对不同的浏览器编写一堆的 CSS 3 代码前缀。Prefix 就是一个能将 CSS 3 代码自动生成跨浏览器 CSS 代码的在线应用。

目前，大部分浏览器都有针对 CSS 3 特性实现的特定前缀，类似-moz-border-radius、-webkit-border-radius。这样经常会出现为了保证兼容性，不得不对一个效果写 4、5 行代码的情况，此时可以采用 Prefix free 来帮助我们。Prefix free 的项目主页如图 10.1 所示。

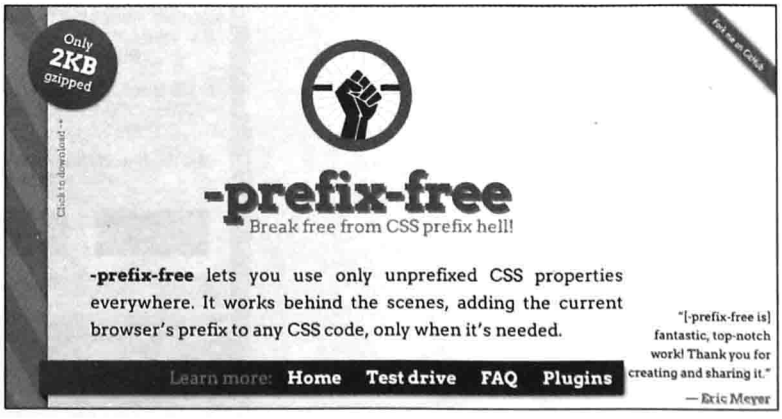


图 10.1　Prefix free 项目主页

Prefix free 的用法很简单，只需要在 CSS 后方引入这个 prefixfree.js 文件就可以了，可以下载后放在项目中，也可以使用 Github 提供的远程链接：

```
<script src="http://leaverou.github.com/prefixfree/prefixfree.min.js"></script>
```

这样在项目中写 CSS 就不再需要写那么多冗长的带有属性前缀的 CSS 3 定义了。

譬如，原先实现一个背景渐变需要 4 行代码来保证兼容性：

```
.test {
   background: -webkit-linear-gradient(red, blue);          /* webkit 核心浏览器兼容代码*/
   background: -o-linear-gradient(red, blue);               /* Opera 浏览器兼容代码*/
   background: -moz-linear-gradient(red, blue);             /* Firefox 浏览器兼容代码*/
   background: linear-gradient(red, blue);                  /*标准语法*/
}
```

使用了 Prefix free 后只需要编写标准语法的 CSS 即可：

```
.test {
   background: linear-gradient(red, blue);                  /*标准语法*/
}
```

10.2　应用 Normalize 统一不同浏览器下的样式

每种浏览器都有自身默认的一套样式，这样即使在 CSS 失效或是没有正常加载时，仍然可以保证一定的可读性。但是在开发时也会造成这样的问题：如果某个样式没有被开发者自定义的样式覆盖，那么就会保持浏览器默认的样式，但是不同浏览器默认样式是不同的。这样就会出现在这个浏览器下显示成这个样子，在那个浏览器下又是那个样子的情况。

传统的做法是写一个 CSS Reset，将默认样式全部覆盖掉，例如下面的代码：

```
body,div,dl,dt,dd,ul,ol,li,h1,h2,h3,h4,h5,h6,pre,
form,fieldset,input,textarea,p,blockquote,th,td {
   padding: 0;
   margin: 0;
}
table {
   border-collapse: collapse;
   border-spacing: 0;
}
fieldset,img {
   border: 0;
}
address,caption,cite,code,dfn,em,strong,th,var {
   font-weight: normal;
   font-style: normal;
}
ol,ul {
   list-style: none;
}
caption,th {
   text-align: left;
```

```
}
    h1,h2,h3,h4,h5,h6 {
  font-weight: normal;
  font-size: 100%;
}
q:before,q:after {
  content:'';
}
abbr,acronym {
  border: 0;
}
```

这种方式简单粗暴，把浏览器可能的自带样式全都覆盖掉了，其缺点在于编写 CSS 项目时不得不重新定义很多本可以无须定义的样式。

Normalize 本质上也是一个 Reset 方案，只不过比起传统的 Reset 更注重通用的方案，重置掉该重置的样式，保留有用的默认样式，同时修复一些 Bug。并且它还对 HTML 5 标签做了一定的约定。目前流行的 Bootstrap 3、Foundation 等框架都采用了 Normalize 进行 HTML 默认样式的 Reset。

Normalize 的用法和其他 Reset 没有任何区别，可以在它的官方网站 http://necolas.github.io/normalize.css/ 进行下载，只要将下载好的文件在 CSS 文件之前引入即可，或者也可以将 Normalize 的内容复制出来放在所有 CSS 代码的最前边。

10.3 应用 Grunt 进行 CSS 压缩

绝大多数情况下，项目完成后需要将 CSS 文件进行合并和压缩后再部署到生产环境上，这样一方面减少网站的 HTTP 请求数量，提高加载速度；另一方面降低网站的流量开销，降低成本。

如果只有单个 CSS 文件，那么直接应用在线的 CSS 压缩工具就可以很方便地完成，这种工具用搜索引擎就可以很方便地找到。不过大多数情况下，都需要将多个 CSS 文件合并压缩为一个文件，如果每次手工粘贴到一起再压缩的话实在是非常麻烦，而且很难应对频繁地修改，稍不留意就会造成生产环境和开发环境的不一致。

对于这个问题，可以采用 Grunt 来进行 JS/CSS 自动化的压缩、合并。Grunt 的官网是 http://gruntjs.com/，如图 10.2 所示。它是一个基于 Node.js 的任务运行工具，用于执行各种需要自动化的任务。

这里采用一个实例进行说明，假设将项目的 CSS 文件全部放在项目目录下名为 css 的文件夹中，现在要求将它们压缩合并成一个名为 main-min.css 的文件，放在 css-min 文件夹下。

（1）首先要保证机器上安装了 Node.js。不同系统下安装方法不同，这里不展开，请读者自行通过搜索引擎查询安装方法。

图 10.2　Grunt 官方网站

（2）在项目目录下建立名为 package.json 的文件，用于配置需要安装的 npm（Node.js 的包管理系统）包，package.json 的内容如下：

```
{
  "name": "CSS_ZIP",
  "description": "CSS 压缩",
  "engines": {
    "node": ">=0.8.0"
  },
  "dependencies": {
    "grunt" : "~0.4.2",
    "grunt-contrib-cssmin" : "x"
  }
}
```

必须填写的是 dependencies 部分，需要用 json 格式填上 npm 包的名称和对应的版本号，版本号可以用 x 来代替。这里我们用到了 Grunt 和 Grunt 的一个用于合并压缩 CSS 的插件 grunt-contrib-cssmin。

（3）完成后从命令行进入项目目录下，执行命令：

```
$ npm install
```

该命令会新建一个名为 node_module 的文件夹（如果不存在的话），将 Grunt 以及 Grunt 插件安装在该文件夹下。

（4）编写 Grunt 配置文件 Gruntfile.js。在项目目录下新建文件 Gruntfile.js 作为 Grunt 的配置文件，内容如下：

```
module.exports = function(grunt) {
  // 配置
    grunt.initConfig({
        cssmin: {
            options: {
                keepSpecialComments: 0
            },
            compress: {
```

```
                files: {
                    'css-min/main-min.css': [
                        "css/*.css"
                    ]
                }
            }
        }
    });
    // 载入 cssmin 插件，用于合并和压缩 CSS
    grunt.loadNpmTasks('grunt-contrib-cssmin');
    // 注册任务
    grunt.registerTask('default', ['cssmin']);
}
```

（5）上一步完成后，在命令行中执行：

```
$ Grunt
```

这时，就可以看到后台自动新建了一个名为 css-min 的文件夹，文件夹中是一个名为 main-min.css 的文件，内容是 css 文件夹中所有.css 文件合并压缩后的结果，最终的目录结构如图 10.3 所示。

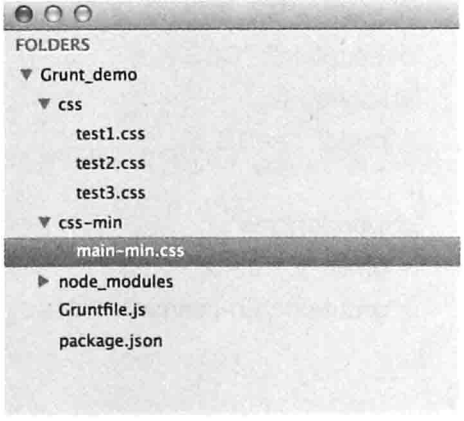


图 10.3　最终的 css 目录结构

以后如果 CSS 有任何修改，只要执行 Grunt 命令就可以了，非常方便。

当然，Grunt 是一款非常强大的工具，自动化的压缩合并 CSS 只是其功能的一部分，Grunt 有很多插件可以帮助开发者完成大多数需要自动化的场景，如图 10.4 所示，可以看到有多达 2500 多个 Grunt 插件可供选择。

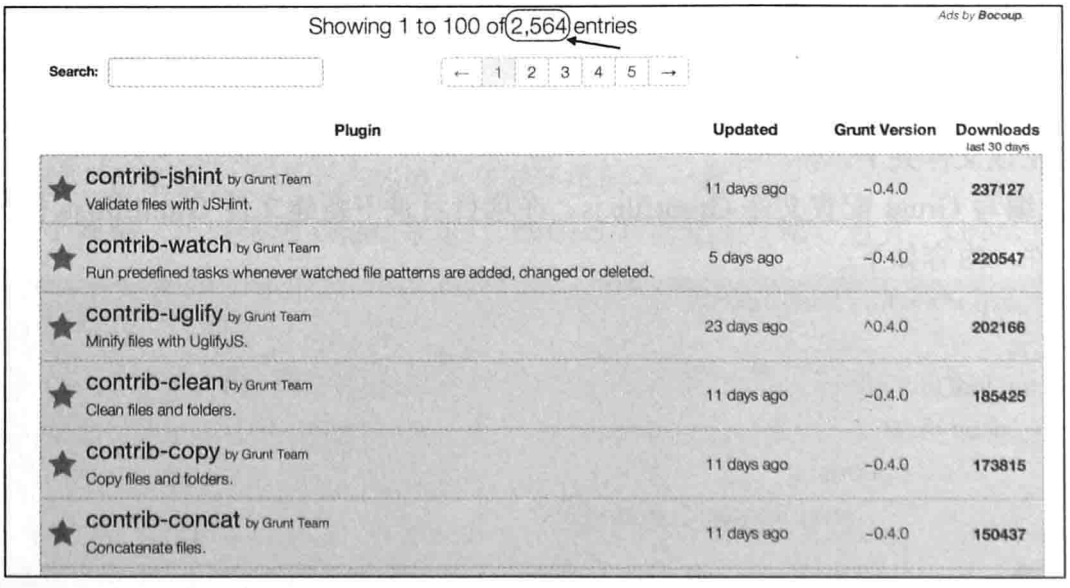


图 10.4　Grunt 插件列表

10.4 小结

本章主要介绍了CSS常用的一些生产力工具，使用工具可以提高编程效率，降低维护成本，帮助开发者将精力放在更有价值的地方。

- Prefix free可以帮助开发者省去编写各种CSS 3属性前缀的工作，只需要在页面中引入prefixfree.js即可。
- Normalize是一个CSS Reset工具，相比传统的Reset，它保留了有用的默认样式，并对一些HTML 5标签进行了一些一致化修正。
- Grunt是基于Node.js的自动化任务运行工具，在完备的插件系统帮助下可以执行各种自动化任务，常用的包括CSS/JS的自动合并压缩、LESS/SASS的自动编译等。

除了本书介绍的这些工具之外，在CSS的编写方面，还有很多不错的生产力工具，合理地应用它们可以减少重复机械的劳动，让工作更为高效。

第二篇

使用 CSS 3 框架进行高效开发

第 11 章　流行的 CSS 布局设计

页面的布局历经二十多年的发展，已经被前人总结出了大量的实用设计方法和辅助工具，本章将为读者介绍当前流行的布局设计，以及用于辅助布局的栅格系统。

本章包括以下主要内容：

- 常用的固定布局设计与 960gs 栅格系统。
- 流式布局与流式栅格系统。
- 响应式设计与媒介查询的应用。

栅格系统是一种网页的排版布局规范，它使元素的宽度都遵循某种约定的规则，这样开发过程中就无须过多地考虑元素的尺寸，同时也方便团队的协作。固定布局、流式布局、响应式布局都有自己的栅格系统。

布局是页面制作中需要考虑的首要因素，一旦开始阶段的布局方案没有选择好，之后再进行转型是非常困难的，开发者应当根据自身需求和能力选择合适的布局方案。

11.1　固定布局

固定布局，顾名思义就是各个部分都采用固定宽度的页面布局，如果缩放页面到窗口宽度小于页面宽度时，就会导致部分内容不可见，必须通过滚动条的拖动才可以浏览全部内容。

虽然移动互联网来势汹汹，响应式设计、流式布局开始逐渐流行，但是在很多应用场景下，固定布局仍是最合适的。例如 B/S 结构的企业应用、海报宣传性质的页面等。而固定布局的稳定、简单、成熟也是前端技术选型中重要的考量。

在开发流程的表达上，固定布局也是成熟而稳定的，从产品经理的草图到设计师的 PSD 设计稿，再到前端页面，全都是静态的，思路的传递简单明晰、成本低廉。相对来说，响应式界面不仅在 HTML/CSS 编写上更为复杂，对产品经理和设计师的能力素质、沟通表达都有更高的要求，需要更多的沟通成本。

开发者不应当盲目追求概念，更需要根据团队的情况、产品的需求、成本的考量来综合考虑技术的选型。因此笔者认为即使移动风潮来势汹汹，固定布局在很多场合下仍然不失为合适的选择。

11.1.1　960 的秘密

大多数信息内容都较复杂，以内容组织为主的网站都采用了居中的 950/960 像素的 元素来包裹页面的主体内容。例如前两年的 Yahoo!、淘宝、新浪、搜狐都是 950 像素；MySpace、优酷、AOL 是 960 像素（目前这些网站大都经历了改版，比如淘宝就已经是响应式的了）。

为什么这些网站都不约而同地选择了这样的布局方式呢？一种说法是在早期电脑屏幕大多为 1024 x 768 的分辨率时，打开 Firefox 的默认状态，窗体的大小约为 974 x 650。减掉左右边框的宽度，宽度大致就在 960 像素左右了。

那为什么不是 961 像素或者是 957 像素这样的值呢？除了整数好记以外，还有一些数学上的原因。

960 可以分解为 2 的 6 次方乘以 3 和 5，这使得 960 可以分割成以下宽度的整数倍：

```
2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 30, 32, 40,
48, 60, 64, 80, 96, 120, 160, 192, 240, 320, 480
```

共 26 种（26 = 7 * 2 * 2 - 2，减去 2 是去掉 1 和 960 自身），标记为：

```
N(960) = N(2^6 * 3 * 5) = 26
```

同理可以得到：

```
N(480) = N(2^5 * 3 * 5) = 6 * 2 * 2 - 2 = 22
N(750) = N(2 * 3 * 5^3) = 2 * 2 * 4 - 2 = 14
N(800) = N(2^5 * 5^2) = 6 * 3 - 2 = 16
N(1000) = N(2^3 * 5^3) = 4 * 4 - 2 = 14
N(1024) = N(2^10) = 11 - 2 = 9
N(1920) = N(2^7 * 3 * 5) = 8 * 2 * 2 - 2 = 30
```

根据数学归纳可以得到一个有趣的结论：要使得 N(width)最大，width 的取值必须是 …, 480, 960, 1920, …

N 越大，可组合的宽度值就越多。对栅格系统来说，这意味着越灵活！

目前绝大多数显示器都支持 1024 x 768 及其以上的分辨率，480 太窄，1920 则太宽（太宽也不利于阅读），因此 960 就成为网页栅格系统中的最佳宽度了。

至于 950 像素和 960 像素其实并没有实质上的区别，只是对左右外边距的设置不同，如果算上外边距的值，加起来也是 960 像素。

11.1.2　定义列宽

如图 11.1 所示，一个固定列宽的栅格系统主要由 3 个部分组成：列（Column）、槽（Gutter）、外边距（Margin）。列的宽度决定了容器内部的宽度，槽的宽度决定了列与列之间的固定间距，外边距则表示 container 边界和实际内容之间的间距。

要构建栅格系统，首先要确定 container 的宽度和列数。

将 Flowline 的总宽度标记为 W，Column 的宽度标记为 c，Gutter 宽度标记为 g，Margin 的宽度标记为 m，Column 的个数标记为 N。由图 11.1 可以得到以下公式：

```
W = c * N + g * (N - 1) + 2 * m
```

一般来说，Gutter 的宽度是 Margin 的两倍，上面的公式就可以简化为：

```
C= c+g
W = C * N
```

以 960 像素宽度的 Flowline 为例，如果要构建 16 列的栅格，一列加上左右外边距的宽度为 960/16，即 60 像素。以 10 像素作为槽的宽度，那么一列的实际容器宽度就是 50 像素。N 列的宽度就是 60*N-10，Margin 的宽度就是 5 像素。如果去掉 Margin 的宽度，就是 950 像素，很多 950 像素居中设置的网站实际就是除去了 Margin 宽度后得出的。

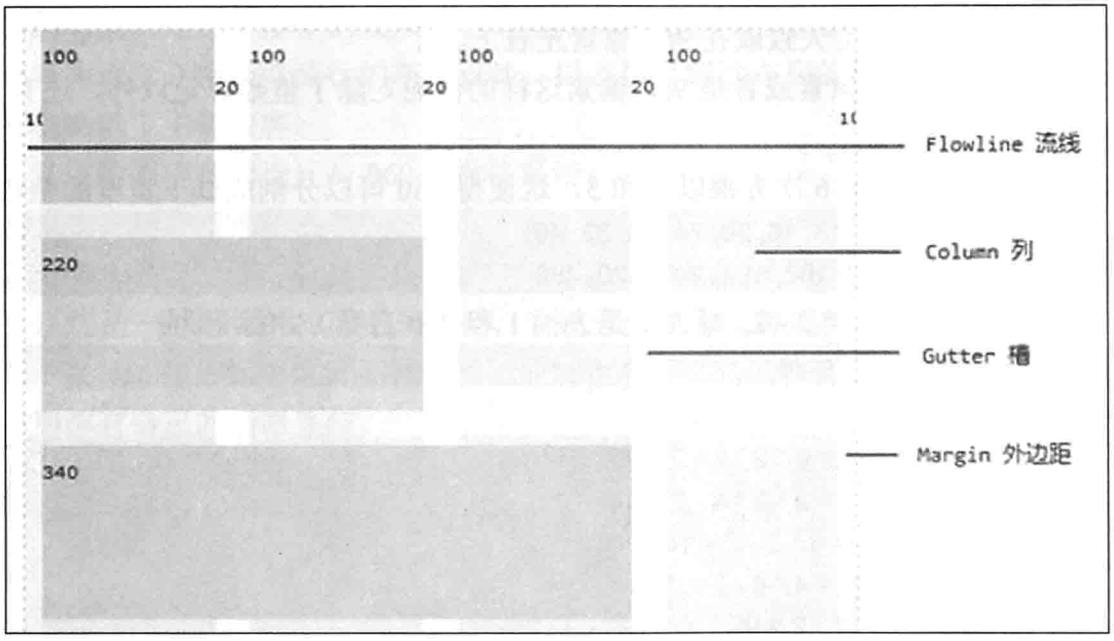


图 11.1　固定栅格系统示意图

11.1.3　运用 CSS 实现固定列宽的栅格

大多数情况下，开发者要么不使用栅格系统，要么使用现成的栅格系统。但一方面了解栅格系统的实现有助于更好地运用它，另一方面对于一些特殊的要求也可以自己构建符合要求的栅格系统。

以较为著名的 960gs 的源代码为例进行说明：

```
/* 定义容器
----------------------------------------------------------------------------------------------*/
.container_12 {
   margin-left: auto;
   margin-right: auto;
   width: 960px;
}

/* `栅格的全局设置
----------------------------------------------------------------------------------------------*/
.grid_1,
```

```
.grid_2,
.grid_3,
.grid_4,
.grid_5,
.grid_6,
.grid_7,
.grid_8,
.grid_9,
.grid_10,
.grid_11,
.grid_12 {
    display: inline;
    float: left;
    margin-left: 10px;
    margin-right: 10px;
}

/* 分别设置栅格的宽度
----------------------------------------------------------------------------------------------------*/
.container_12 .grid_1 {
    width: 60px;
}

.container_12 .grid_2 {
    width: 140px;
}

.container_12 .grid_3 {
    width: 220px;
}

.container_12 .grid_4 {
    width: 300px;
}

.container_12 .grid_5 {
    width: 380px;
}

.container_12 .grid_6 {
    width: 460px;
}

.container_12 .grid_7 {
    width: 540px;
```

```
}

.container_12 .grid_8 {
  width: 620px;
}

.container_12 .grid_9 {
  width: 700px;
}

.container_12 .grid_10 {
  width: 780px;
}

.container_12 .grid_11 {
  width: 860px;
}

.container_12 .grid_12 {
  width: 940px;
}
```

这段示例代码并没有包含 960gs 栅格框架的全部内容，但是已经涵盖了固定列宽的栅格系统的主要内容，有以下几点：

- 设置容器，960gs 的容器采用了 960 像素宽度居中的设置，需要设置 width: 960px; 并且左右两侧的外边距设置为 auto，也可以简单地设置为 margin : auto;。
- 为栅格命名，栅格命名一般采用 grid（格子）或者 column（列）作为前缀进行表示，数字作为后缀表示栅格的列数。
- 设置栅格的全局属性，左浮动（float: left;）属性是必需的，此外还需要设置栅格之间的间隔宽度。960gs 设置左右两侧的外边距均为 10 像素，也可以像 Bootstrap 2 一样只设置一侧的边距。
- 设置列宽，960gs 采用的方式是 12 列栅格的宽度恰好填满容器。那么 grid_1 的整体宽度就是 60 像素，加上左右两侧的外边距为 80 像素。grid_n 的宽度就是(80*n-20)像素。

注意：有一些栅格系统，比如 Bootstrap 2，它的 12 个单列栅格宽度之和超过了外层容器的宽度，需要在容器内多增强一层，设置 margin 的负值，例如.row{margin-left: -30px;}。

对于大多数项目需求来说，采用目前已有的 12 列/16 列/24 列栅格系统是足够的，如果的确需要自己构建一套，只要做到以上 4 点，就可以轻松地搭建自定义的栅格系统。

注意：对于像 960gs 这样完善的栅格系统，还会提供诸如偏移量、清除浮动等辅助的工具类。

11.1.4　实战演练——运用 960gs 实现固定布局的新闻页面

960gs 全称 960 Grid System，是目前运用较多的一个固定布局栅格系统，它以 960 像素居中的容器为基础，设计了相对应的 12/16/24 列栅格系统。开发者可以根据自身需求进行选择，列数越多，搭配也越多，使用起来也越灵活，但设计的复杂度也会相应地提高。图 11.2 给出的是 12 列 960gs 栅格系统。

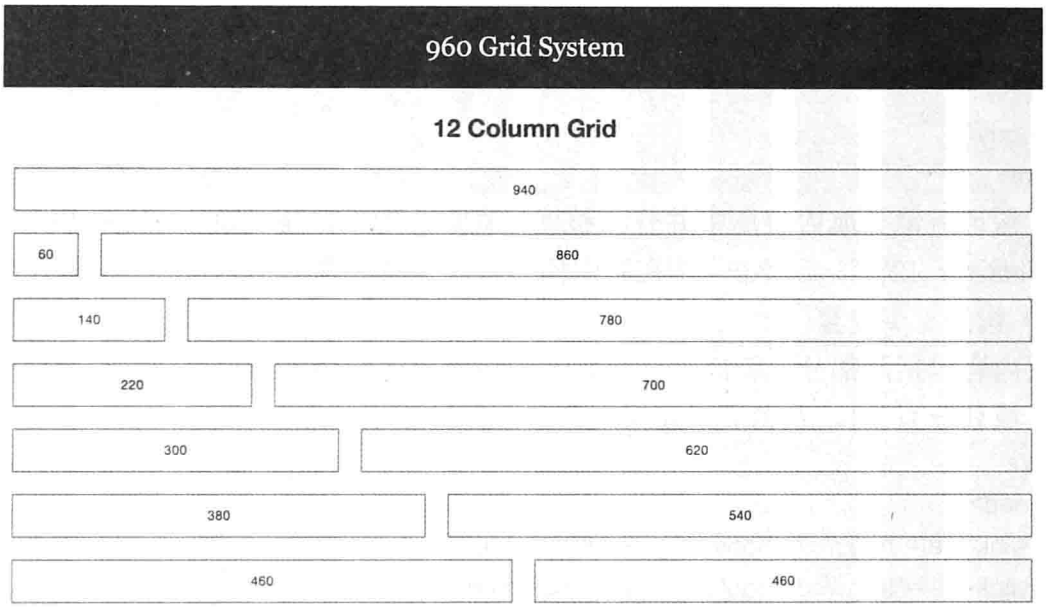


图 11.2　12 列 960gs 栅格系统

下面用一个新闻页面的实例来说明如何使用 960gs 来实现一个固定布局的页面。

（1）构建基本结构并引入 960gs 的 CSS 文件：

```
<html>
  <head>
    <link href="http://cdn.bootcss.com/960gs/0/960.css" rel="stylesheet">
    <link href="http://cdn.bootcss.com/foundation/5.1.1/css/foundation.css" rel="stylesheet">
  </head>
  <body>
  </body>
</html>
```

这里为了方便，直接使用 CDN 链接引入了远程的 960gs 文件，为了快速构建样式，还同时引入了 Foundation 框架。

（2）创建居中的首部导航：

```
<html>
  <head>
    <link href="http://cdn.bootcss.com/960gs/0/960.css" rel="stylesheet">
    <link href="http://cdn.bootcss.com/foundation/5.1.1/css/foundation.css" rel="stylesheet">
  </head>
  <body>
    <header style="margin-top:10px;">
```

```
        <div class="container_12">
          <ul class="sub-nav">
            <li><h4><a href="#">今日要闻</a></h4></li>
            <li class="active"><h4><a href="#">体育新闻</a></h4></li>
            <li><h4><a href="#">国际时讯</a></h4></li>
            <li><h4><a href="#">经济快报</a></h4></li>
            <li><h4><a href="#">军情解码</a></h4></li>
          </ul>
        </div>
</header>
  </body>
</html>
```

为了保证首部导航内容居中并且和整体一样居中对齐，需要在<header>内部加入 ..... ，在其中构建首部导航内容。

（3）构建主要内容：

主要内容包括两部分：新闻主体内容、侧边的相关新闻推荐与广告。这里将居中的内容主体区域划分为 2:1，左侧显示新闻主体，右侧显示相关推荐。

```
<html>
  <head>
    <link href="http://cdn.bootcss.com/960gs/0/960.css" rel="stylesheet">
    <link href="http://cdn.bootcss.com/foundation/5.1.1/css/foundation.css" rel="stylesheet">
  </head>
  <body>
    <header>
      <div class="container_12">
        <ul class="sub-nav">
          <li><h4><a href="#">今日要闻</a></h4></li>
          <li class="active"><h4><a href="#">体育新闻</a></h4></li>
          <li><h4><a href="#">国际时讯</a></h4></li>
          <li><h4><a href="#">经济快报</a></h4></li>
          <li><h4><a href="#">军情解码</a></h4></li>
        </ul>
      </div>
    </header>
    <div class="container_12">
      <div class="grid_8">
        <h3>詹姆斯鼻梁骨折，赛前决定是否出战公牛</h3><hr>
        <p>热火今天宣布，经过今天在迈阿密的重新评估之后，前锋勒布朗-詹姆斯的伤势确诊为鼻梁骨折。</p>
        <p>热火下一场比赛是在北京时间 2 月 24 日，对手是公牛，詹姆斯将赛前决定是否出战本场比赛。</p>
        <p>ESPN 的记者 Brian Windhorst 透露，一位消息人士说詹姆斯应该会上场。热火方面将先等詹姆斯不再感到酸痛，随后再决定对他进行何种治疗，以及詹姆斯是否需要带上面具。</p>
        <p>詹姆斯是在昨天热火与雷霆第四节比赛还剩 5 分 56 秒的时候，在一次突破中被伊巴卡
```

```
    打到面部时受的伤，随后他回到了更衣室接受治疗并且退出了该场比赛。</p>
      <p>在退场前，詹姆斯在这场比赛中得到了 33 分，帮助热火 103-81 击败雷霆。</p>
    </div>
    <div class='grid_4'>
      <ol class="side-nav prefix_1">
        <li><h4><a href="#">24 小时新闻排行榜</a></h4></li>
        <li><a href="#">官方 MVP 榜：詹姆斯反超杜兰特重回第一</a></li>
        <li><a href="#">詹姆斯鼻梁骨折，赛前决定是否出战公牛</a></li>
        <li><a href="#">保罗-乔治谈格兰杰：我爱你，兄弟</a></li>
        <li><a href="#">格兰杰无意留在 76 人，希望被买断？ </a></li>
        <li><a href="#">快船报价香珀特只为防止其去雷霆？ </a></li>
        <li><a href="#">保罗-加索尔：期待成为自由球员</a></li>
      </ol>
    </div>
  </div>
 </body>
</html>
```

代码首先使用 …… 包裹主体内容，使其以 960 像素宽度居中布置。然后使用 …… 和 …… 将主体部分划分为 2:1。之后就可以向两部分分别填入内容了。

注意：由于新闻内容和推荐内容之间只有 20 像素的间距，视觉上看起来很拥挤，这时就可以使用 960 提供的偏移类进行调整。注意示例代码中的<ol class="side-nav prefix_1">，这里的 prefix_1 表示元素向右偏移一列的宽度。

完成后的效果如图 11.3 所示。

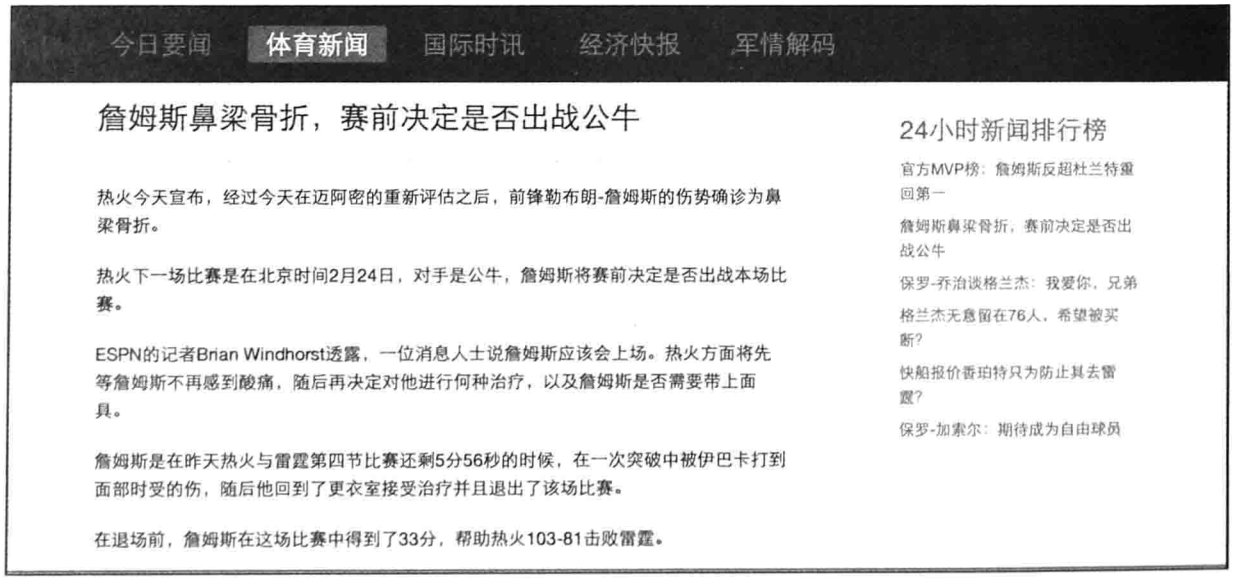


图 11.3　运用 960gs 实现固定布局的新闻页面

这样，一个主体内容居中的新闻页面布局就基本完成了，剩下的工作就是进行细节雕琢了。

11.2 流式布局

在响应式布局成熟之前，某些需要元素随窗口大小变化的需求往往采用 Table 进行布局，但是 Table 布局相对来说又有着不够灵活、结构复杂、语义性差的问题，而流式布局在某种程度上可以帮助开发者们解决这个难题。

11.2.1 计算列百分比

对于流式布局来说，我们可以通过直接定义模块和模块间距百分比的方式来实现，不过和固定布局一样，对于复杂页面来说，把工作交给更规范、更快捷的栅格系统是更好的选择。

和固定布局不同的是，流式布局无须再去考虑 container 的宽度选择，在栅格计算上，直接将 container 的宽度值设为 100%进行计算，计算方法则和固定栅格是一致的。

如果需要构建一个 12 列的流式栅格系统，列+槽的宽度为：

```
100%/12 = 8.333%
```

如果列宽和槽宽的比为 3:1 的话，那么槽的宽度为：

```
8.333%/4 = 2.083%
```

N 列的宽度为：

```
8.333% * N – 2.083%
```

以 Bootstrap 2 的流式布局代码为例：

```
.row-fluid [class*="span"] {
  display: block;
  float: left;
  width: 100%;
  min-height: 30px;
  margin-left: 2.127659574468085%;
  *margin-left: 2.074468085106383%;
  -webkit-box-sizing: border-box;
     -moz-box-sizing: border-box;
          box-sizing: border-box;
}

.row-fluid [class*="span"]:first-child {
  margin-left: 0;                          /*设置第一个子元素的左外边距为 0*/
}

.row-fluid .controls-row [class*="span"] + [class*="span"] {
  margin-left: 2.127659574468085%;
}

.row-fluid .span12 {
```

```
    width: 100%;
    *width: 99.94680851063829%;          /*老版本的 IE 浏览器兼容代码*/
}

.row-fluid .span11 {
    width: 91.48936170212765%;
    *width: 91.43617021276594%;
}

.row-fluid .span10 {
    width: 82.97872340425532%;
    *width: 82.92553191489361%;
}

.row-fluid .span9 {
    width: 74.46808510638297%;
    *width: 74.41489361702126%;
}

.row-fluid .span8 {
    width: 65.95744680851064%;
    *width: 65.90425531914893%;
}

.row-fluid .span7 {
    width: 57.44680851063829%;
    *width: 57.39361702127659%;
}

.row-fluid .span6 {
    width: 48.93617021276595%;
    *width: 48.88297872340425%;
}

.row-fluid .span5 {
    width: 40.42553191489362%;
    *width: 40.37234042553192%;
}

.row-fluid .span4 {
    width: 31.914893617021278%;
    *width: 31.861702127659576%;
}

.row-fluid .span3 {
```

```
  width: 23.404255319148934%;
  *width: 23.351063829787233%;
}

.row-fluid .span2 {
  width: 14.893617021276595%;
  *width: 14.840425531914894%;
}

.row-fluid .span1 {
  width: 6.382978723404255%;
  *width: 6.329787234042553%;
}
```

可以看到 Bootstrap 计算值和笔者得出的计算结果并不一致，这是由于 Bootstrap 对第一个子元素的 margin-left 设置为 0 导致的。

注意：观察 Bootstrap 的代码可以发现类似*width: 23.351063829787233%;这样的属性定义，原因是 IE 6/7 下宽度 100%时是包含了外层滚动条的宽度的，因此需要针对性地做出兼容性设置。

11.2.2 使图片更加灵活

在流式布局页面中引入图片时会发现图片大小是固定的，怎样让图片随着窗口的大小调整显示大小呢？

我们只需要为图片元素添加 max-width: 100%; 和 height: auto;属性，就可以让图片按比例缩放不超过其父元素的尺寸。如果想让图片和父元素一直等宽的话，则将 max-width: 100%;改为 width: 100%;即可，例如下面的代码：

```
<html>
  <head>
    <style>
      .respond_img{
        height:auto;
        width: 100%;
      }
    </style>
  </head>
    <div style="width:30%">
      <img class="respond_img" src="http://imge.gmw.cn/mytest.jpg">
    </div>
  <body>
</html>
```

如果不添加.respond_img 类，图片会一直保持原始大小，添加后，图片则始终保持页面宽度的 30%，且图片的宽高比例不变，不会导致失真。

11.2.3 定义最大/最小宽度

本节一开始提出的例子中展示了没有定义最大/最小宽度会产生的问题，如图 11.4 所示，窗口宽度太窄或者太宽都可能导致流式布局页面出现错乱，或者不适合阅读的情况。

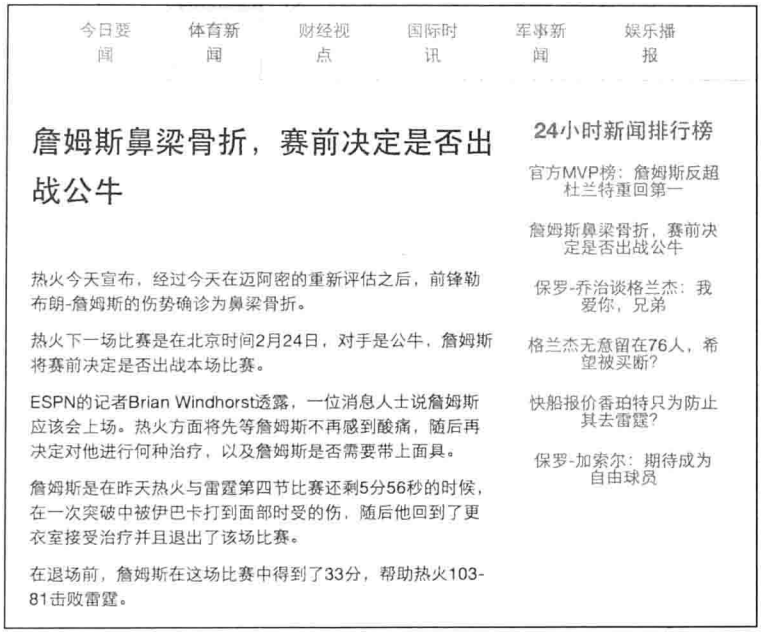


图 11.4 采用流式布局的新闻页面

可以看到，由于采用了流式布局，首部导航随着窗口宽度的缩小自动进行了折行，这样就显得很不美观。同样的，如果屏幕过大，整个页面会被拉得非常宽，也会影响美观和使用体验。

在流式布局的基础上，应用最大/最小宽度则可以在一定程度上解决这个问题。设置最大/最小宽度后，如果窗口大于最大宽度或小于最小宽度时，页面将呈现和固定布局一样的表现。例如图 11.5，可以发现当页面缩小到最小宽度以下后，文字没有折行，页面下方则出现了滚动条，和固定布局有着相同的呈现。

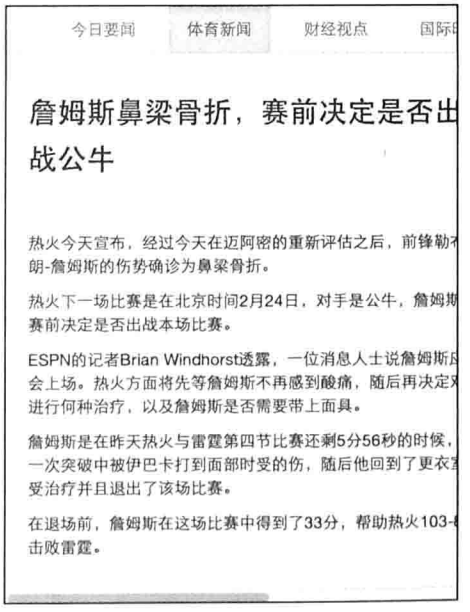


图 11.5 应用最大/最小宽度的流式布局

合理地定义最大/最小宽度的流式布局页面，比较适合制作同时适应平板和 PC 的页面，也适合制作不同尺寸 PC 屏幕的页面。

11.2.4 实战演练——实现一个流式布局的新闻页面

和固定布局一样，这里仍然采用同样的新闻页面，以帮助读者做对比。本节采用了

Bootstrap 2 提供的流式栅格进行布局，构造一个同时满足主流平板和 PC 的流式布局页面。

（1）引入 Bootstrap，构建页面基本结构：

```
<html>
  <head>
    <link href="http://libs.baidu.com/bootstrap/2.3.2/css/bootstrap.css" rel="stylesheet">
  </head>
  <body>
  </body>
</html>
```

（2）构建顶部导航，设置最大/最小宽度：

```
<html>
  <head>
    <link href="http://libs.baidu.com/bootstrap/2.3.2/css/bootstrap.css" rel="stylesheet">
    <style>
      .nav li{
        text-align: center;
      }
      .align-fluid{
        max-width: 1000px;
        min-width: 600px;
        margin: auto;
      }
    </style>
  </head>
  <body>
    <div class="navbar">
      <div class="navbar-inner">
        <div class="container-fluid align-fluid">
          <div class="row-fluid">
            <ul class="nav" style="float:none;">
              <li class="span2"><a href="#">今日要闻</a></li>
              <li class="span2 active"><a href="#">体育新闻</a></li>
              <li class="span2"><a href="#">财经视点</a></li>
              <li class="span2"><a href="#">国际时讯</a></li>
              <li class="span2"><a href="#">军事新闻</a></li>
              <li class="span2"><a href="#">娱乐播报</a></li>
            </ul>
          </div>
        </div>
      </div>
</div>
  </body>
</html>
```

在 Bootstrap 2 中应用流式布局，需要添加：

```
<div class="container-fluid ">
  <div class="row-fluid">
  ……
  </div>
</div>
```

这里每个导航标题占据两列宽度，标题需要居中，因此添加了额外的 CSS 代码：

```
<style>
    .nav li{
        text-align: center;   /*设置标题文字居中*/
    }
</style>
```

由于该页面设定的需求是同时适配主流平板和 PC，主流平板的宽度一般是 640 像素，那么这里笔者选择了设置最大宽度为 1000 像素，最小像素为 600 像素，并让主要内容居中显示。

```
<style>
  .align-fluid{
     max-width: 1000px;   /*设置最大宽度*/
     min-width: 600px;    /*设置最小宽度*/
     margin: auto;        /*设置居中*/
  }
</style>
```

（3）构建主要内容：

```
<div class="container-fluid align-fluid" style="margin-top:20px;">
  <div class="row-fluid">
    <div class="span8">
      <h3>詹姆斯鼻梁骨折，赛前决定是否出战公牛</h3><hr>
      <p>热火今天宣布，经过今天在迈阿密的重新评估之后，前锋勒布朗-詹姆斯的伤势确诊为鼻梁骨折。</p>
      <p>热火下一场比赛是在北京时间 2 月 24 日，对手是公牛，詹姆斯将赛前决定是否出战本场比赛。</p>
      <p>ESPN 的记者 Brian Windhorst 透露，一位消息人士说詹姆斯应该会上场。热火方面将先等詹姆斯不再感到酸痛，随后再决定对他进行何种治疗，以及詹姆斯是否需要带上面具。</p>
      <p>詹姆斯是在昨天热火与雷霆第四节比赛还剩 5 分 56 秒的时候，在一次突破中被伊巴卡打到面部时受的伤，随后他回到了更衣室接受治疗并且退出了该场比赛。</p>
      <p>在退场前，詹姆斯在这场比赛中得到了 33 分，帮助热火 103-81 击败雷霆。</p>
      </div>
    <div class="span4">
      <ul class="nav nav-pills nav-stacked">
        <li><h4><a href="#">24 小时新闻排行榜</a></h4></li>
        <li><a href="#">官方 MVP 榜：詹姆斯反超杜兰特重回第一</a></li>
        <li><a href="#">詹姆斯鼻梁骨折，赛前决定是否出战公牛</a></li>
        <li><a href="#">保罗-乔治谈格兰杰：我爱你，兄弟</a></li>
        <li><a href="#">格兰杰无意留在 76 人，希望被买断？ </a></li>
        <li><a href="#">快船报价香珀特只为防止其去雷霆？ </a></li>
```

```
        <li><a href="#">保罗-加索尔：期待成为自由球员</a></li>
      </ul>
    </div>
  </div>
</div>
```

和之前固定布局的例子一样，这里也将新闻主体和推荐内容左右分开，宽度比例为 2:1。

```
<div class="container-fluid align-fluid" style="margin-top:20px;">
  <div class="row-fluid">
<div class="span8">
  ……新闻主体内容
</div>
<div class="span4">
  ……推荐内容
</div>
  </div>
</div>
```

最终的显示效果如图 11.6 和图 11.7 所示，其中图 11.6 为窗口宽度为 1000 像素时的效果，图 11.7 为窗口宽度为 600 像素时的效果。读者可以对比两个图的效果，观察其中的区别。

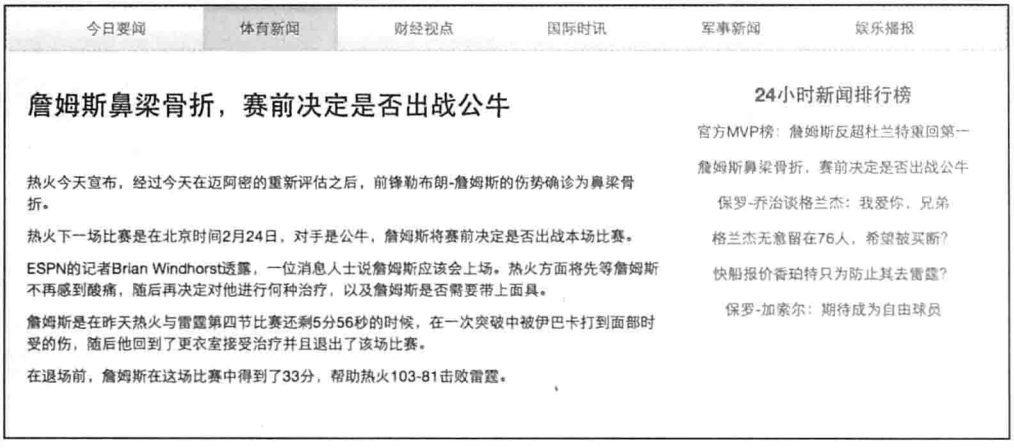


图 11.6　1000 像素时的效果

图 11.7　600 像素时的效果

11.3　响应式布局

页面可以根据用户的设备尺寸或浏览器的窗口尺寸来自动地进行布局的调整，这就是响应式布局。在这个移动互联兴起的时代，响应式布局占据着越来越重要的地位。图 11.8 是一个直观的响应式布局设计示意图。

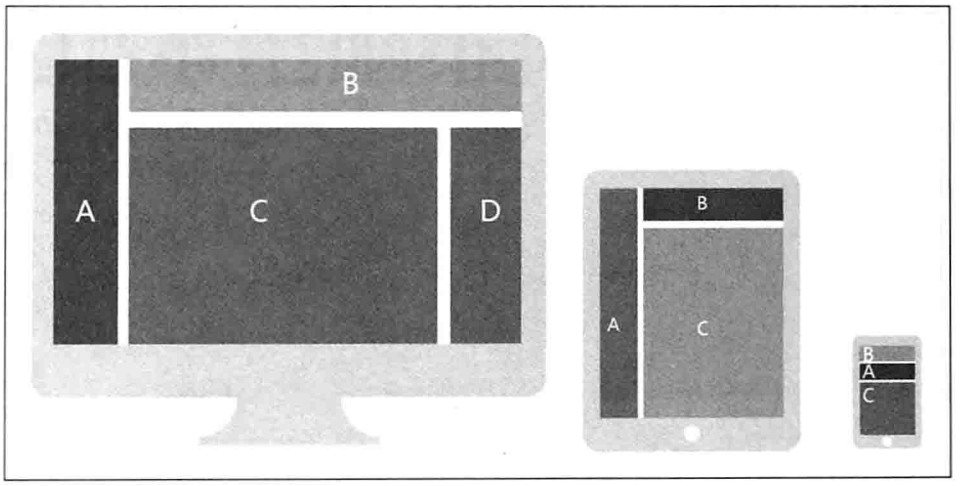


图 11.8　响应式布局设计示意图

近年来，移动互联网发展势头迅猛，尤其是高性能智能手机和平板的普及，使得在移动设备上浏览绚丽的页面成为了可能（相对于曾经的 WAP 手机站来说）。响应式设计越来越流行，预计在不久的将来，大部分的网站都会拥抱移动，响应式页面会成为主流选择。

11.3.1　使用媒介查询

要实现页面的响应式从技术上要如何实现呢？CSS 3 提供了媒介查询的功能帮助我们实现这个目标。在本书第 8 章中对媒介查询的原理已经有了详细地介绍，这里不再赘述。本小节主要了解一些主流响应式框架对媒介查询的应用。

这里以著名的响应式开发框架 Foundation 5 的部分源代码作为示例：

```
@media only screen {     /*在不被覆盖的情况下适用于所有场景（仅作用于屏幕显示器，不包括打
    印机等设备）*/
 ......
  .column,
  .columns {
    position: relative;
    padding-left: 0.9375rem;
    padding-right: 0.9375rem;
    float: left; }

  .small-1 {
    position: relative;
```

```
    width: 8.33333%; }  /*Foundation 放弃了对旧版本 IE 的兼容，故无须兼容代码*/

  .small-2 {
    position: relative;
    width: 16.66667%; }

  .small-3 {
    position: relative;
    width: 25%; }
    ......
    ......
  .small-12 {
    position: relative;
width: 100%; }
......
......
}

@media only screen and (min-width: 40.063em) {  /*适用于中等大小窗口*/
  ......
  .column,
  .columns {
    position: relative;
    padding-left: 0.9375rem;
    padding-right: 0.9375rem;
    float: left; }

  .medium-1 {
    position: relative;
    width: 8.33333%; }

  .medium-2 {
    position: relative;
    width: 16.66667%; }
  ......
  ......
  .medium-12 {
    position: relative;
width: 100%; }
  ......
}
```

为了更为简明，这里只保留了很少一部分代码，可以发现，响应式设计的栅格系统实际就是在流式栅格系统的基础上添加了媒介查询，使之更加灵活而已。

如果只使用 small-n 系列栅格，那么实际上和使用流式布局的栅格系统没有任何区别。

响应式栅格系统实际就是告诉浏览器当前宽度下应当使用的样式。譬如下面的代码：

```
<div class="small-12 medium-2" >.....</div>
```

如果是小窗口，那么该 元素就占据 100%的宽度，如果是中等大小窗口，则只占据 16.66667%的宽度。

11.3.2 实战演练——实现一个响应式布局的新闻页面

本小节仍然采用之前相同的内容作为案例，不同的是，这里将采用著名的响应式开发框架 Foundation 来构建这个新闻页面，让它可以同时适应手机、平板、PC 等不同终端的浏览。

（1）引入框架，构建页面基本结构：

```
<html>
  <head>
    <meta charset="utf-8">
    <link href="http://cdn.bootcss.com/foundation/5.1.1/css/foundation.css" rel="stylesheet">
    <link href="http://cdn.bootcss.com/foundation/5.1.1/css/normalize.min.css" rel="stylesheet">
  </head>
  <body>
    ..... <!--这里编辑内容-->.....
    <script src="http://cdn.bootcss.com/jquery/2.0.3/jquery.min.js"></script>
    <script src="http://cdn.bootcss.com/foundation/5.1.1/js/foundation.min.js"></script>
    <script>$(document).foundation();</script>
  </body>
</html>
```

注意：由于这个实例要实现小屏幕下导航收起展开的效果，因此需要引入 Foundation 的 JavaScript 插件以及其需要依赖的 jQuery，如果只是单纯的做页面的响应式，可以无须引用这些 JavaScript 组件。

（2）构建首部导航：

```
<html>
  <head>
    <meta charset="utf-8">
    <link href="http://cdn.bootcss.com/foundation/5.1.1/css/foundation.css" rel="stylesheet">
    <link href="http://cdn.bootcss.com/foundation/5.1.1/css/normalize.min.css" rel="stylesheet">
  </head>
  <body>
    <header style="background:#333">
      <nav class="top-bar row" data-topbar>
        <ul class="title-area">
          <li class="name">
            <h1><a href="#">新闻频道</a></h1>
          </li>
```

```
            <li class="toggle-topbar menu-icon"><a href="#">Menu</a></li>
          </ul>
          <section class="top-bar-section">
            <ul class="left">
              <li><a href="#">今日要闻</a></li>
              <li class="active"><a href="#">体育新闻</a></li>
              <li><a href="#">国际时讯</a></li>
              <li><a href="#">经济快报</a></li>
              <li><a href="#">军情解码</a></li>
            </ul>
          </section>
        </nav>
</header>
    <script src="http://cdn.bootcss.com/jquery/2.0.3/jquery.min.js"></script>
    <script src="http://cdn.bootcss.com/foundation/5.1.1/js/foundation.min.js"></script>
    <script>$(document).foundation();</script>
  </body>
</html>
```

读者可能会注意到，首部导航这里并没有用到响应式的栅格系统，这里实际上是在框架内部通过媒介查询实现的，而导航的展开和收起则是通过 JavaScript 来进行控制，当窗口宽度小于某个阈值时就会将导航内容全部收起。如图 11.9 所示。

图 11.9　收起导航内容的效果

这里就体现了响应式设计的原则，在小屏幕设备上，让最重要的内容直面用户，而将一些较为次要的内容延后、收起或隐藏。

（3）构建主体内容：

布局不变，仍然是将页面主体划分为 2:1 两部分，左侧主体内容，右侧推荐内容。不同的是，由于需要在手机下进行浏览，本身屏幕就非常窄了，又有 1/3 的部分用于显示次要的推荐内容，非常影响阅读体验和页面的美观，因此在小屏幕下，我们要让推荐内容出现在主体内容的后方。

```
<div class="row" style="margin-top:20px;">
  <div class="small-12 medium-8 large-8 columns">
    <h3>詹姆斯鼻梁骨折，赛前决定是否出战公牛</h3><hr>
    <p>热火今天宣布，经过今天在迈阿密的重新评估之后，前锋勒布朗-詹姆斯的伤势确诊为鼻梁骨折。</p>
    <p>热火下一场比赛是在北京时间 2 月 24 日，对手是公牛，詹姆斯将赛前决定是否出战本场比赛。</p>
    <p>ESPN 的记者 Brian Windhorst 透露，一位消息人士说詹姆斯应该会上场。热火方面将先等詹姆斯不再感到酸痛，随后再决定对他进行何种治疗，以及詹姆斯是否需要带上面具。</p>
```

```
    <p>詹姆斯是在昨天热火与雷霆第四节比赛还剩 5 分 56 秒的时候，在一次突破中被伊巴卡打到面部时受的伤，随后他回到了更衣室接受治疗并且退出了该场比赛。</p>
    <p>在退场前，詹姆斯在这场比赛中得到了 33 分，帮助热火 103-81 击败雷霆。</p>
  </div>
  <div class="small-12 medium-4 large-4 columns">
    <ul class="side-nav prefix_1">
      <li><h4><a href="#">24 小时新闻排行榜</a></h4></li>
      <li><a href="#">官方 MVP 榜：詹姆斯反超杜兰特重回第一</a></li>
      <li><a href="#">詹姆斯鼻梁骨折，赛前决定是否出战公牛</a></li>
      <li><a href="#">保罗-乔治谈格兰杰：我爱你，兄弟</a></li>
      <li><a href="#">格兰杰无意留在 76 人，希望被买断？</a></li>
      <li><a href="#">快船报价香珀特只为防止其去雷霆？</a></li>
      <li><a href="#">保罗-加索尔：期待成为自由球员</a></li>
    </ul>
  </div>
</div>
```

将实际的代码简化后，将看到如下的结构：

```
<div class="row">
  <div class="small-12 medium-8 large-8 columns">
    .....<!-- 主要内容 -->.....
  </div>
  <div class="small-12 medium-4 large-4 columns">
    .....<!-- 推荐内容 -->.....
  </div>
</div>
```

为主体添加.row 类用于保证内容的居中，内容宽度为 Foundation 的默认值 62.5rem。栅格的设置根据 HTML 的语义是很容易理解的，以左侧主要新闻内容部分为例，.small-12 表示在小窗口下占据 12 列的宽度，即占据整行，.medium-8 和.large-8 则表示在中等大小窗口和大窗口下占据 8 列的宽度。

页面完成后最终的效果如图 11.10、图 11.11、图 11.12、图 11.13、图 11.14 所示，分别展示了响应式新闻页面在 PC 屏幕、平板和手机下的浏览效果。

图 11.10　PC 窗口下的页面样式

图 11.11　平板下的页面样式

图 11.12　手机下的页面样式

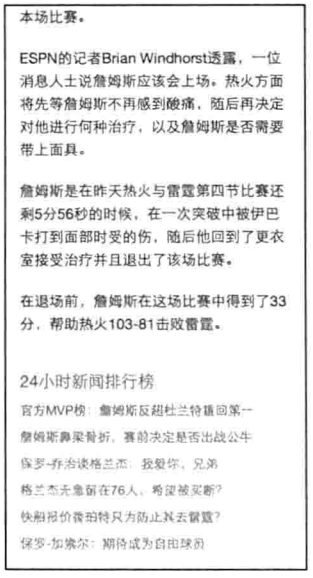


图 11.13　推荐内容放在了正文之后显示

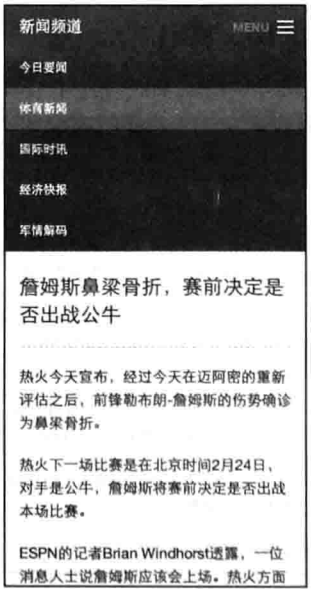


图 11.14　手机中导航展开后的样式

11.4　小结

本章介绍了当前应用最广泛的 3 种页面布局方式：固定布局、流式布局、响应式布局，以及与之配合的栅格系统。

从结构上来说，固定布局最简单，开发成本最低，但相对来说灵活性比较差，难以适应不同屏幕设备的浏览需要。

流式布局一般用于终端窗口大小变化不太大的情况，比如同时适应 PC 和平板的需求，也比较简单，但是需要注意最大/最小宽度的合理设置以及设置图片随父元素的缩放大小。

响应式布局是目前发展的一个趋势，可以帮助开发者们做到同时支持超大屏幕显示器和手机。当然，开发难度也要相对更大一些。不过 Bootstrap 3、Foundation 等开发框架的出现可以大大帮助我们简化响应式页面的开发。

本章的重点是栅格系统，固定布局、流式布局、响应式布局目前都有多款对应的优秀栅格系统，而像 Bootstrap 这类整体式的前端解决方案更是将它们都进行了集成，开发者们可以根据需要选择自己需要的栅格系统。

读者需要注意的是，栅格系统更适合构建那些复杂的、需要一致性的页面，一些简单的，或者需要设计灵活性非常大的页面则更适合直接使用像素单位进行更精确的定义。

第 12 章　Bootstrap 框架实战

本章主要介绍目前最流行的前端一体化框架 Bootstrap 的基本信息、使用场景和使用方法。对于互联网工具来说，借助官方或者翻译好的使用手册无疑是工作中查询效率最高的手段，因此本章会更多通过笔者的使用经验，筛选出重点，着重介绍一些实战中经常出现的问题和相应的解决方案。

本章主要内容包括：

- Bootstrap 能做什么、解决了什么问题。
- 如何使用 Bootstrap。
- 如何对 Bootstrap 进行个性化定制。

12.1　认识 Bootstrap

Bootstrap 是当前应用最广泛、最为开发者所熟知的前端框架，它缘何出现？发展的历程是什么？实现了哪些功能？为何如此流行？本节将揭开谜题。

12.1.1　初识 Bootstrap

Bootstrap 是 Twitter 公司于 2011 年 8 月开源的整体式前端框架，它由 Twitter 的设计师 Mark Otto 和 Jacob Thornton 合作开发。经过短短几个月的时间就红遍全球，大量 Bootstrap 风格的网站出现在互联网的信息浪潮之中，而应用更为广泛的是它的后台管理界面。笔者近两年接触的所有互联网项目的后台均采用了 Bootstrap 进行构建。

Bootstrap 的官方网站地址是 http://getbootstrap.com/，界面如图 12.1 所示。可以在官网下载最新的版本和详细的使用说明文档。目前国内也有不错的 Bootstrap 汉化文档，地址是 http://www.bootcss.com/。

图 12.1　Bootstrap 官网

12.1.2　Bootstrap 为何如此流行

1．功能强大和样式美观的强强联合

Bootstrap 包含了绝大多数的常用页面组件和动态效果，并且它是由专业的网页设计师精心制作的，足够的美观精致，即使是一个没有专业网页设计师的团队也可以利用 Bootstrap 快速地构建简洁美观的页面，而在 Bootstrap 出现之前，快速和美观往往是互斥的。

一些大型互联网公司（如 Google、雅虎、新浪、百度等）都会有强大的内部通用样式库和 JavaScript 组件库，但它们一方面是不开源的，另一方面大部分库都带有这些公司的特定风格和烙印，即使开源，应用面也并不广泛。

2．简单易用，文档丰富

Bootstrap 使用起来非常简单，并且有非常详尽的文档（如图 12.2 所示），甚至可以不用查看代码，只需将文档当做“黑盒”来使用，就可以构建出相当漂亮的页面效果，而且样式类的语义性非常好，根据英文单词的意义很容易记忆。

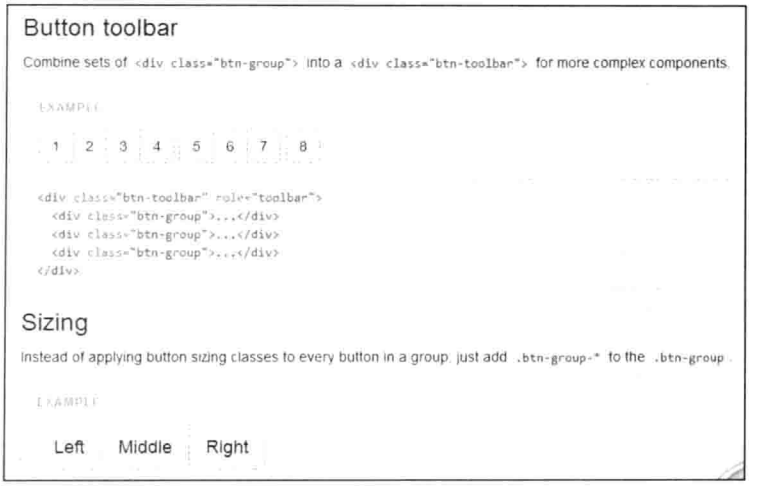


图 12.2　Bootstrap 的文档样例

3．高度可定制性

Bootstrap 的一大优点是它极佳的可定制性，一方面可以有选择性的只下载自己需要的组件，另一方面在下载前可以调配参数（如图 12.3 所示）来匹配自己的项目。由于 Bootstrap 是完全开源的，使用者也可以根据自己的需要来更改代码。

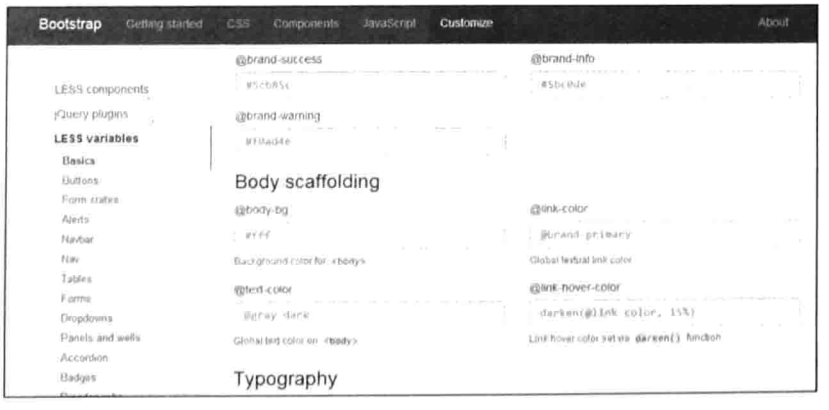


图 12.3　定制化选择界面

4．丰富的生态圈

由于 Bootstrap 如此优秀，在 Web 开发领域出现了很多基于 Bootstrap 的插件，一些集成的 CMS 也开始应用 Bootstrap。例如图标字体插件 Font Awesome、富文本编辑器插件 bootstrap-wysihtml5、Rails 插件 bootstrap-sass 等，还有很多基于 Bootstrap 的“皮肤”插件，弥补了 Bootstrap 流行后同质化严重的问题，例如基于 Window Metro 风格的 Flat UI、基于 Google 风格的 Google Bootstrap。

国内外都有 Bootstrap 的免费 CDN 服务，这更推动了 Bootstrap 的流行，由于国内无法使用 Google 的 CDN，建议使用百度 CDN 服务：

未压缩版本：

```
<script src="http://libs.baidu.com/bootstrap/2.0.4/js/bootstrap.js"></script>
<link href="http://libs.baidu.com/bootstrap/2.0.4/css/bootstrap.css" rel="stylesheet">
```

压缩后的版本：

```
<script src="http://libs.baidu.com/bootstrap/2.0.4/js/bootstrap.min.js"></script>
<link href="http://libs.baidu.com/bootstrap/2.0.4/css/bootstrap.min.css" rel="stylesheet">
```

5．布局兼容性良好

虽然 Bootstrap 采用了很多 CSS 3 的效果，但是在布局上可以兼容到 IE 7。使用 Bootstrap 可以很大程度上避免在 IE 下的布局错乱。当然，在较老版本的 IE 浏览器下，效果会打一些折扣。

12.1.3 Bootstrap 的版本发展

目前 Bootstrap 的版本已经发展到了 Bootstrap 3。不过这并不意味着 Bootstrap 2 已退出了历史舞台。Bootstrap 3 在样式上采用了扁平化的风格，Bootstrap 2 在按钮、工具栏等位置更多地采用了立体的效果。它们之间并没有孰优孰劣的区别，更多的是设计风格的不同。使用者应该根据自己项目的实际需求来决定使用哪一个版本。

图 12.4 是 Bootstrap 2 和 Bootstrap 3 的按钮样式对比，可以发现 Bootstrap 2 的按钮有明显的凸起效果。

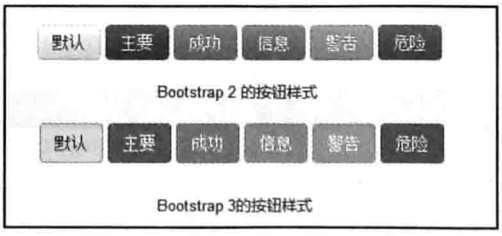


图 12.4　Bootstrap 2 和 Bootstrap 3 中的按钮对比效果

12.2　Bootstrap 入门

当我们拿到一个开源框架时，如何引入这个框架，如何调用其中的组件，这些都是我

们最关心的问题。本节以这些问题为切入点，为读者介绍 Bootstrap 入门的简单知识。

12.2.1　在自己的项目中引入 Bootstrap

Bootstrap 的源代码是使用 CSS 的预编译语言 Less 编写的，关于 Less，本书后面会详细介绍，不过应用 Bootstrap 需要使用的是编译好的 CSS 文件。

官方网站提供两个下载入口，一个是首页的 Download Bootstrap 按钮，另一个是首部导航栏的 Customize，这里可以提供定制化的下载。一般情况下，建议直接全部下载，在开发基本完成后，再考虑根据实际的使用情况进行定制化下载，以缩减前端代码。

在项目中引入 Bootstrap 的方法很简单，和引入其他 CSS 或 JavaScript 文件一样，使用<script>标签引入 JavaScript 文件，使用<link>标签引入 CSS 文件。不过需要注意的是 Bootstrap 的 JavaScript 效果都是基于 jQuery 的，因此如果需要使用 Bootstrap 的 JavaScript 动态效果的话，就必须先引入 jQuery：

```
<head>
  <link href=" bootstrap.css" rel="stylesheet">
</head>
<body>
  Html code.....
  ..................
  <script src=" jQuery.js"></script>          /*jQuery 应该放在前面优先加载*/
  <script src=" bootstrap.js"></script>
</body>
```

注意：JavaScript 文件放在文档尾部有助于提高加载速度。

引入 Bootstrap 还可以使用第三方的 CDN 服务，Bootstrap 2 版本可以使用百度的 CDN 服务，网址是 http://developer.baidu.com/wiki/index.php?title=docs/cplat/libs；Bootstrap 3 版本则建议使用 Bootstrap 中文网提供的 CDN，网址是 http://open.bootcss.com/；当然如果是做国外的项目，首选则是 Google 的 CDN 服务了。

12.2.2　添加 Bootstrap 的 class 实现基本样式

以编写一个表格为例，如果不使用 Bootstrap 或者其他类似的框架，有以下两步：

（1）第一步肯定是构思设计表格的样式，宽度、高度、行高、对齐方式、边框等很多地方。如果一开始的设想与实际效果并不符合，还需要后面不断地调试。

（2）第二步需要编写相应的 HTML/CSS 代码，边写，边调试，还要边思考如何给 id 或者 class 命名，最后可能还需要上司或者同事进行审核。

如果决定使用 Bootstrap，那么只需要引入 Bootstrap，然后在<table>标签中添加一个 class="table"就可以获得一个 Bootstrap 设定好的表格样式，示例代码如下：

```
<table class="table">    /*只需要添加 class="table"即可*/
  <tr>
```

```
<th>姓名</th>
<th>年龄</th>
<th>职业</th>
  </tr>
  <tr>
<td>张三</td>
<td>18</td>
<td>程序员</td>
  </tr>
  ……
</table>
```

效果如图 12.5 所示。

姓名	年龄	职业
张三	18	程序员
李四	20	运营专员

图 12.5　应用 Bootstrap 的表格样式

当然，Bootstrap 不会死板的只提供一种样式，对于表格来说，还可以添加 table-striped 类来添加斑马纹，添加 table-bordered 来为表格加上边框和圆角。例如：

```
<table class="table table-striped table-bordered">
<th>姓名</th>
<th>年龄</th>
<th>职业</th>
  </tr>
  <tr>
<td>张三</td>
<td>18</td>
<td>程序员</td>
  </tr>
  ……
</table>
```

代码效果如图 12.6 所示。

姓名	年龄	职业
张三	18	程序员
李四	20	运营专员

图 12.6　带斑马纹和圆角边框的表格

12.2.3　调用 Bootstrap 的通用组件

除了添加 class 的方式外，在布局方面，只要符合约定的一些 class 命名和嵌套结构，我们就可以轻松地构建出一些通用组件，以导航条为例：

```
<div class="navbar">
  <div class="navbar-inner">
<a class="brand" href="#">Title</a>
<ul class="nav">
        <li class="active"><a href="#">首页</a></li>
        <li><a href="#">Link</a></li>
        <li><a href="#">Link</a></li>
</ul>
  </div>
</div>
```

只要符合 div.navbar > div.navbar-inner > ul.nav >li 这样的 HTML 文档结构，就可以构建出一个顶部导航条，效果如图 12.7 所示。

图 12.7　顶部导航条

12.2.4　添加 JavaScript 动态效果

对于 Bootstrap 中 JavaScript 效果的添加，一方面需要根据文档编写特定的 HTML 结构，另一方面需要调用 JavaScript 插件。下面以标签页切换效果为例来讲解。

首先编写 HTML 文档：

```
<ul class="nav nav-tabs" id="myTab">
  <li class="active"><a href="#home" data-toggle="tab">首页</a></li>
  <li><a href="#profile" data-toggle="tab">Profile</a></li>
  <li><a href="#messages" data-toggle="tab">Messages</a></li>
  <li><a href="#settings" data-toggle="tab">Settings</a></li>
</ul>
/*href 属性的值要和后面 tab-pine 中的 id 值对应*/

<div class="tab-content">
  <div class="tab-pane active" id="home">.....</div>          /*tab 标签对应的内容*/
  <div class="tab-pane" id="profile">.....</div>
  <div class="tab-pane" id="messages">.....</div>
  <div class="tab-pane" id="settings">.....</div>
</div>
```

JavaScript 插件的调用一般有两种方式，一种是采用 Bootstrap 自带的触发规则，在标签中添加 data-toggle="tab"这样的属性来实现，这种方式的好处是无须编写任何 JavaScript 代码就可以实现功能；另一种则类似普通 jQuery 插件的调用方式，例如：

```
$('#myTab a').click(function (e) {
  e.preventDefault();
  $(this).tab('show');
})
```

最终实现的效果如图 12.8，单击标签页就可以切换内容。

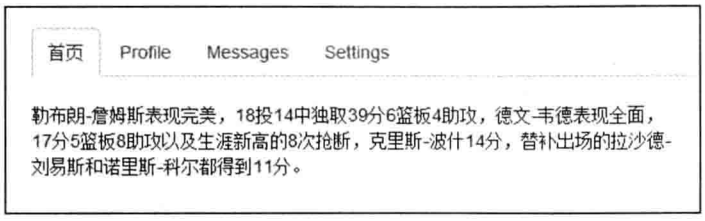


图 12.8　标签页效果

12.3　Bootstrap 的栅格系统

上一章的布局设计介绍了栅格系统、栅格系统的优缺点、应用场景等内容，Bootstrap 作为最流行的前端框架，也为开发者们准备了一套灵活、强大的栅格系统。随着移动设备的发展，为了更好地拥抱移动端设备， Bootstrap 3 对栅格系统做了较大的调整，因此本节会通过比较的方式来介绍 Bootstrap 2 和 Bootstrap 3 的栅格系统，看看移动互联网带来了哪些变化，同时介绍如何定制栅格系统。

12.3.1　固定布局的栅格系统

1．基础用法

Bootstrap 2 默认不自带响应式固定布局的 12 列栅格系统。栅格的宽度为 60 像素，分隔宽度为 20 像素，总宽度为 940 像素（不计算最左侧的分隔宽度）。

绝大多数网站的内容都是居中布置的，Bootstrap 2 对此提供了一个让网站内容居中的“容器”。可以通过添加 class="container" 来使用。为了配合 Bootstrap2 内置的 12 列固定布局栅格，“容器”的宽度默认为 940 像素。

固定栅格使用起来非常简单，如下面的例子：

```
<div class="container">
  <div class="row">
<div class="span3">
.....<--! 构建左侧导航 -->
</div>
<div class="span9">
.....<--! 构建主要内容-->
</div>
  </div>
</div>
```

注意：在使用栅格时需要在外边包裹 class="row"的元素。

如果外层的栅格数=内层的栅格个数总和时，必须使用 class="row"，否则会造成折行，以 span12 和 container 为例，它们都不包含最左侧的外边距的宽度，而 span3+span9 则会多

出来一个 20 像素的分隔宽度，而添加 会自动加上这 20 像素，即 的宽度为 960 像素。

Bootstrap 3 中不再提供非响应式的固定布局，如果需要禁用响应式布局，请执行以下操作：

- 不要添加 viewport<meta>。
- 通过为.container 设置一个 width 值从而覆盖框架的默认 width 设置，例如 width: 970px !important;。
- 如果使用了导航条，需要移除所有导航条的折叠和展开行为。
- 对于栅格布局，额外增加.col-xs-* class 或替换掉.col-md-*和.col-lg-*，针对超小屏幕设备的栅格系统能够在所有分辨率的环境下展开。

2．设置偏移量

栅格系统的意义之一就是统一网站的布局宽度和间距，便于信息地组织和代码地维护，由于阅读是从左至右进行的，因此在编写代码中经常需要设置 margin-left 属性来指定元素的偏移量，而使用了栅格系统后，margin-left 的值往往是固定的几个值，因此 Bootstrap 提供了 offset 类来帮助开发者无须测量也无须编写任何 CSS 代码就可以方便地设置偏移量。

每个类都给列的左边距增加了指定单位的列。例如，.offset3 将元素右移了 3 个列的宽度：

```
<div class="row">
  <div class="span4">...</div>
  <div class="span3 offset3">...</div>
</div>
```

12.3.2　流式布局的栅格系统

固定布局的栅格系统采用固定宽度的栅格宽度、分隔宽度、偏移量，而对于一些需要自适应宽度的场景，使用百分比作为宽度是更好的选择，这种布局方式一般称作流式布局。

将固定布局中外层的.row 类替换为.row-fluid 类就可以实现流式布局的栅格系统了。应用于每一列的类不用改变，这样能方便地在流式与固定栅格之间切换，例如：

```
<div class="row-fluid">
  <div class="span3">...</div>
  <div class="span9">...</div>
</div>
```

12.3.3　响应式布局的栅格系统

Bootstrap 2 和 Bootstrap 3 在响应式方面其实差别不大，比较大的区别在于 Bootstrap 2 更偏重于传统的 PC 端网页，默认不支持响应式，需要额外引入；而 Bootstrap 3 则恰好相反，默认自带响应式，不再提供非响应式的固定布局。因此从灵活性角度考虑，反而是 Bootstrap 2 更好。

1. Bootstrap 2 中的响应式布局

先来看看 Bootstrap 2 提供的响应式方案。在 Bootstrap 2 中，必须额外引入响应式布局的代码，在下载解压 Bootstrap 后，会有一个名为 bootstrap-responsive.css 的文件。需要在 HTML 文档中引入这个文件，此外，还需要添加响应式必需的<meta>标签，代码示例如下：

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="assets/css/bootstrap-responsive.css" rel="stylesheet">
```

在本书第 8 章中详细介绍了媒介查询和响应式设计的内容，手写响应式代码由于各人风格、水平的差异，同样的效果下写出的代码可能大相径庭，而 Bootstrap 等框架对响应式的封装不仅仅提升了开发效率，也避免了多人协作和维护项目的问题。

Bootstrap 2 根据市场上常用的设备宽度分隔了 5 个区间：

- 1200 像素以上的大屏幕设备，如台式机、13 寸以上的笔记本。
- 980~1200 像素之间的设备，如上网本、老式的窄屏显示器。
- 768~980 像素之间的主流平板。
- 480~768 像素的大屏手机或 6、7 寸的平板。
- 480 像素以下的小屏手机。

不同的区间中，栅格和 container 会使用不同的值来适应比例的变化，具体的数值请参看表 12.1。

表 12.1　Bootstrap 2 的响应式布局区间

类型	布局宽度	列宽	间隙宽度
大屏幕	大于等于1200px	70px	30px
默认	大于等于980px	60px	20px
平板	大于等于768px	42px	20px
手机到平板	小于等于767px	流式列，无固定宽度	
手机	小于等于480px	流式列，无固定宽度	

2. Bootstrap 3 中的响应式布局

除了默认引入响应式以外，在功能方面 Bootstrap 3 相比 Bootstrap 2 有两点比较大的变化：

- 拥抱大屏幕，移除了小屏手机和大屏手机（480~768 像素）这个媒介查询区间，768 像素以下的统一归为小屏幕设备。
- 设计了表现不同的栅格类，对栅格类的命名规则也做了很大的修改，更复杂但使用也更灵活，能适应更多的场景。

在 Bootstrap 2 中，栅格全部采用 span*作为前缀。而在 Bootstrap 3 中采用了 col-type-* 这样命名的前缀，其中 type 可以取 xs（超小屏）、sm（小屏）、md（中屏）、lg（大屏）4 个值。

通过表 12.2 可以详细查看 Bootstrap 的栅格系统是如何在多种屏幕设备上工作的。

表 12.2　Bootstrap 3 的响应式布局区间

	超小屏幕设备 手机 (<768px)	小屏幕设备 平板 (≥768px)	中等屏幕设备 桌面 (≥992px)	大屏幕设备 桌面 (≥1200px)
栅格系统行为	总是水平排列	开始是堆叠在一起的，超过这些阈值将变为水平排列		
最大.container宽度	None (自动)	750px	970px	1170px
class前缀	.col-xs-	.col-sm-	.col-md-	.col-lg-
列数	12			
最大列宽	自动	60px	78px	95px
槽宽	30px (每列左右均有15px)			
可嵌套	Yes			
Offsets	N/A	Yes		
列排序	N/A	Yes		

Bootstrap 中的响应式栅格用例如下：

```
<head>
  <script src="http://libs.baidu.com/jquery/1.9.0/jquery.js"></script>
  <script src="http://cdn.bootcss.com/twitter-bootstrap/3.0.2/js/bootstrap.js"></script>
  <link href="http://cdn.bootcss.com/twitter-bootstrap/3.0.2/css/bootstrap.css" rel="stylesheet">
</head>          < --! 引入 Bootstrap 3-->
<body style="margin:20px">
  <div class="container">
    <div class="row">
      <div class="col-xs-12 col-sm-3 col-md-5 col-lg-4">  < --! 左侧边栏-->
        <h1>体育新闻</h1>
        <h1>娱乐新闻</h1>
        <h1>经济新闻</h1>
      </div>
      <div class="col-xs-12 col-sm-9 col-md-7 col-lg-8">  < --! 右侧边栏-->
        <p>“魔兽” 德怀特-霍华德贡献 25 分 7 个篮板，其中罚球 24 罚 17 中，成功破解了掘金队的“砍兽”战术</p>
      </div>
    </div>
  </div>
</body>
```

根据表 12.2 中的介绍来看这个示例，可以发现在窗口尺寸大于 1200px 时，左侧边栏占据 4 列宽度，右侧边栏占据 8 列宽度；尺寸在 992px~1200px 之间时，左侧边栏占据 5 列宽度，右侧边栏占据 7 列宽度；而当尺寸在 768px~992px 之间时，左侧边栏占据 3 列宽度，右侧边栏占据 9 列宽度。小于 768px 时，则左右侧边栏都占据 100%宽度，堆叠起来。

12.4 使用 Bootstrap 的基本样式

Bootstrap 的基本样式包括表格、表单、按钮、图片、工具类等几大类，基本的使用方式都是通过添加特定名称的 class 来实现的。

12.4.1 字体排版

1. 标题

Bootstrap 重新定义了<h1>～<h6>标签的样式，Bootstrap 2 采用了加粗字体和固定的行高（line-height），而 Bootstrap 3 则采用了半加粗的字体，所有标题的行高都采用了 1.1 倍字体尺寸。两者对比示例如图 12.9。

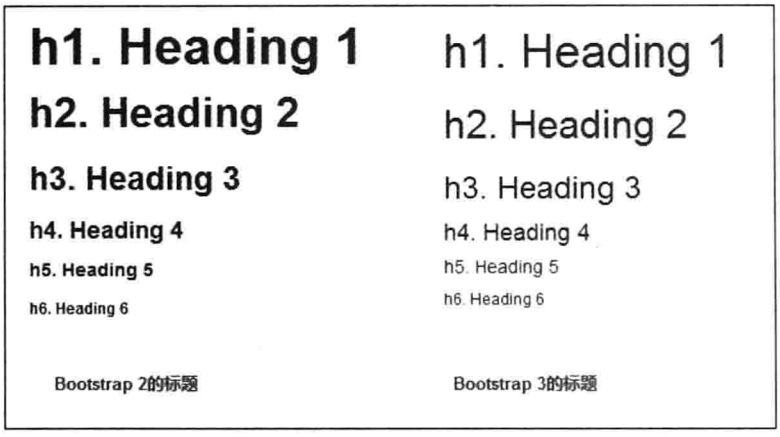


图 12.9 <h1>～<h6>标签的对比

不过依笔者的经验，大多数网站对于标题都会采用自己设计的样式，Bootstrap 默认的标题样式一般只会在做后台管理或者内部工具的时候才会使用，主要是解决各个浏览器下默认样式不一致的问题，而且相比浏览器默认的样式还是要美观不少。

可以在标题中插入<small>标签或者.small 元素来设置副标题，例如：

```
<h1>h1. Bootstrap heading  <small>Secondary text</small></h1>
```

效果如图 12.10 所示。

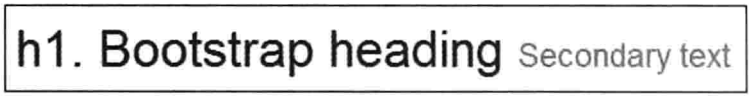


图 12.10 <small>标签效果

2. 全局设置

Bootstrap 2 中定义的全局字体是 14px，行高（line-height）是 20px。这些样式能应用到<body>和所有的段落上。另外，Bootstrap 2 对段落（<p>）还定义了 1/2 行高（默认为 10px）的底部外边距（margin）属性。

Bootstrap 3 中的全局设置实际的效果和 Bootstrap 2 保持了一致，不过为了提高可用性和适应性，将行高（line-height）从固定高度改为了字体的倍数，默认为字体尺寸的 1.428 倍。

对于段落（<p>），可以通过添加 class="lead"进行突出显示。Bootstrap 默认的全局字体和段落效果如图 12.11 所示。

“魔兽”德怀特-霍华德贡献25分7个篮板，其中罚球24罚17中，成功破解了掘金队的“砍兽”战术

火箭其他球员数据，钱德勒-帕森斯拿到20分6个篮板3次助攻；詹姆斯-哈登17分9次助攻5个篮板；林书豪拿到16分7次助攻4个篮板；多纳塔斯-莫蒂埃尤纳斯拿到12分3个篮板。

掘金这边，泰-劳森拿到28分17次助攻的大号两双数据；JJ-希克森拿到14分8个篮板；肯尼思-法里德8分10个篮板。替补方面，威尔森-钱德勒拿到14分3个篮板；季莫费-莫兹戈夫拿到9分10个篮板。

图 12.11　Bootstrap 默认的全局字体和段落效果

其中第一段采用了段落的突出显示，请读者注意段落间距和字间距。如果认为默认设置不合适，可以修改 Bootstrap 代码或者自己编写代码进行覆盖。

12.4.2　表格

1．基本用法

构建 Bootstrap 的表格基础样式是通过为<table>标签添加 class="table"来实现的。Bootstrap 2 和 Bootstrap 3 在用法上完全一致，示例代码如下：

```
<table class="table">
  <tr>
    <th>#</th>
    <th>姓</th>
    <th>名</th>
    <th>昵称</th>
  </tr>
  <tr>
    <td>1</td>
    <td>李</td>
    <td>四</td>
    <td>拉里</td>
  </tr>
  ……
</table>
```

Bootstrap 的表格默认样式是不带边框和分隔的，如图 12.12 所示。

#	姓	名	昵称
1	李	四	拉里
2	王	五	博德
3	赵	四	詹姆斯

图 12.12　Bootstrap 的表格默认样式

Bootstrap 3 和 Bootstrap 2 的表格区别不大，比较明显的是在 Bootstrap 3 中，表格标题的分隔线做了加粗处理，如图 12.13 所示。

#	姓	名	昵称
1	李	四	拉里
2	王	五	博德
3	赵	四	詹姆斯

图 12.13　在 Bootstrap 3 的表格标题分割线

2．表格的附加样式

添加了表格基础样式后，Bootstrap 还提供 4 种附加样式。

- .table-bordered：为表格加上边框，如图 12.14 所示。

#	姓	名	昵称
1	李	四	拉里
2	王	五	博德
3	赵	四	詹姆斯

图 12.14　Bootstrap 2 下的 table-bordered 样式

- .table-striped：为表格加上斑马线效果，如图 12.15 所示。

#	姓	名	昵称
1	李	四	拉里
2	王	五	博德
3	赵	四	詹姆斯

图 12.15　Bootstrap 的 table-striped 样式

- .table-hover：鼠标悬停在表格行上时展现不同的颜色，如图 12.16 所示。

#	姓	名	昵称
1	李	四	拉里
2	王	五	博德
3	赵	四	詹姆斯

图 12.16　Bootstrap 的 table-hover 样式

- .table-condensed：更为紧凑的表格样式，如图 12.17 所示。

#	姓	名	昵称
1	李	四	拉里
2	王	五	博德
3	赵	四	詹姆斯

图 12.17　Bootstrap 的 table-condensed 样式

注意：Bootstrap 3 取消了 table-bordered 状态下的圆角效果。

注意：表格的斑马线是通过:nth-child CSS 选择器实现的，而在本书第 3 章介绍过，这是一个 CSS 3 新加入的选择器，因此无法被 IE 9 以前的浏览器支持。

注意：鼠标悬停在颜色不同的那一行。

这 4 种样式是可以叠加使用的，比如下面的示例代码：

```
<table class="table  table-bordered  table-striped">
  <tr>
    <th>#</th>
    <th>姓</th>
    <th>名</th>
    <th>昵称</th>
  </tr>
  <tr>
    <td>1</td>
    <td>李</td>
    <td>四</td>
    <td>拉里</td>
  </tr>
  ..........
</table>
```

实际效果如图 12.18 所示，表格加上了斑马纹和边框效果。

#	姓	名	昵称
1	李	四	拉里
2	王	五	博德
3	赵	四	詹姆斯

图 12.18 加上了斑马纹和边框效果的表格

3. 为表格行或单元格添加状态标识

实际需求中常常要将某些表格行或者单元格进行特殊的标记，Bootstrap 也提供了这一支持，通过为表格中的<tr>添加相应的 class 即可实现。

- .active：鼠标悬停在行或单元格上时所设置的颜色。
- .success：标识成功或积极的动作。
- .warning：标识警告或需要用户注意。
- .danger：标识危险或潜在的会带来负面影响的动作。

例如下面的代码：

```
<!—为表格行添加标示 -->
<tr class="active">...</tr>
<tr class="success">...</tr>
<tr class="warning">...</tr>
<tr class="danger">...</tr>
```

```
<!—为单元格(`td`或者`th`)添加标示 -->
<tr>
  <td class="active">...</td>
  <td class="success">...</td>
  <td class="warning">...</td>
  <td class="danger">...</td>
</tr>
```

4．响应式表格

响应式表格是 Bootstrap 3 中新加入的特性，可以让较宽的表格在小屏幕设备上（小于768px）出现水平滚动条。当屏幕大于 768px 宽度时，水平滚动条消失，效果如图 12.19 所示。

#	表格标题	表格标题	表格标题	表格标题	表格标
1	表格内容	表格内容	表格内容	表格内容	表格内
2	表格内容	表格内容	表格内容	表格内容	表格内
3	表格内容	表格内容	表格内容	表格内容	表格内

图 12.19　小屏幕设备上的滚动条

使用方法很简单，只需要把表格包裹在一个 class="table-responsive"的元素中即可，例如：

```
<div class="table-responsive">
  <table class="table"> ... </table>
</div>
```

12.4.3　表单

表单有很多类型，如文本输入框、下拉菜单、单选框、复选框、提交按钮等，Bootstrap 提供了一整套风格统一、简洁美观的表单样式，只要引入 Bootstrap，无须做任何配置，表单的样式就会生效，如图 12.20 所示。

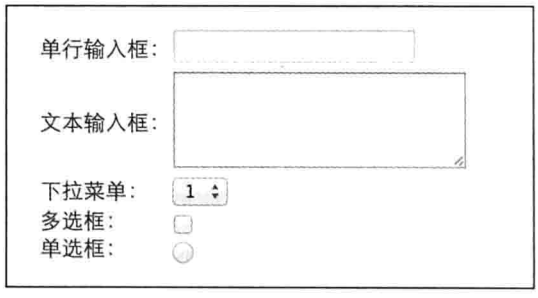


图 12.20　Bootstrap 3 下表单默认样式

注意：在 Bootstrap 3 中，只有正确设置了.form-control 类的 input 和 textarea 元素才能被赋予正确的样式，如图 12.21 所示。

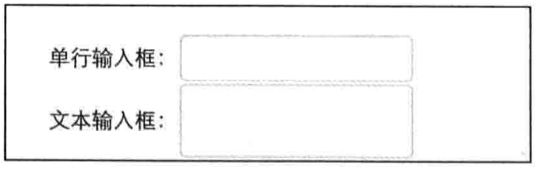


图 12.21　Bootstrap 3 添加.form-control 类后输入框样式

Bootstrap 更为便捷的是，只需要简单地配置，就可以组合出诸如搜索框、按钮下拉菜

单、表单对齐等实战中经常用到的效果。

表单对齐是表单应用中最基本、最常用的技巧，解决方案多种多样，既可以通过调整元素的内外边距，也可以使用表格，而 Bootstrap 则提供了一种通用的解决方案，下面以 Bootstrap 3 的表单为例介绍。

如果要出现<label>提示和表单元素换行的情况，类似图 12.22 这样的效果。

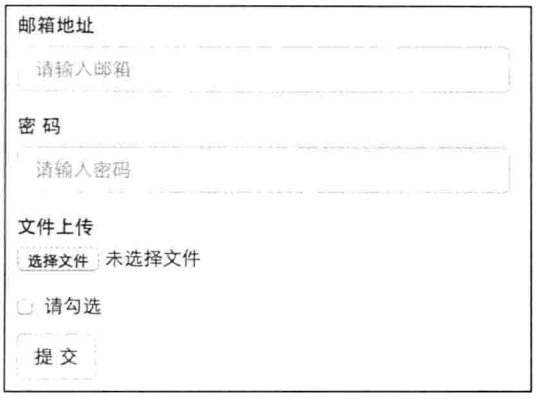


图 12.22　<label>提示和表单元素换行显示

这种情况可以采取如下代码处理：

```
<form role="form">
  <div class="form-group">                    <--! form-group 用于调整表单排列效果-->
    <label for="Email1">邮箱地址</label>
<input type="email" class="form-control" id="Email1" placeholder="Enter email">
    <--! 添加 form-control 类用于将表单设置为 100%宽度，并添加样式上的美化-->
  </div>
  <div class="form-group">
    <label for="Password1">密码</label>
    <input type="password" class="form-control" id="Password1" placeholder="Password">
  </div>
  <div class="form-group">
    <label for="exampleInputFile">文件上传</label>
    <input type="file" id="exampleInputFile">
    <p class="help-block">.....</p>
  </div>
  <div class="checkbox">
    <label>
      <input type="checkbox">请勾选
    </label>
  </div>
  <button type="submit" class="btn btn-default">提交</button>
</form>
```

其中所有设置了.form-control 的<input>、<textarea>和<select>元素都将被默认设置为 width:100%;。将 label 和前面提到的这些控件包裹在.form-group 中可以获得最佳的排列效果。

<label>和表单在同一行也是一种常见的需求，如图 12.23 这样的效果。

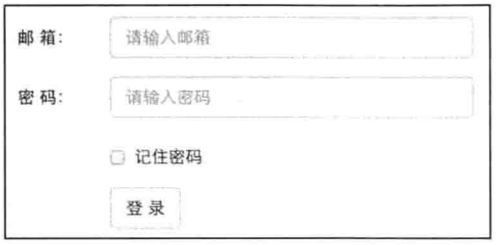


图 12.23　<label>提示和表单元素同行显示

这里需要为<form>添加.form-horizontal 类，并为<label>元素添加 .control-label 类，此外还需要用到之前介绍的栅格系统，例如下面的代码：

```
<form class="form-horizontal" role="form">
  <div class="form-group">
    <label for="inputEmail3" class="col-sm-2 control-label">邮箱</label>
    <div class="col-sm-10">
      <input type="email" class="form-control" id="inputEmail3" placeholder="Email">
    </div>
  </div>
  <div class="form-group">
    <label for="Password3" class="col-sm-2 control-label">密码</label>
    <div class="col-sm-10">
      <input type="password" class="form-control" id="iPassword3" placeholder="Password">
    </div>
  </div>
  <div class="form-group">
    <div class="col-sm-offset-2 col-sm-10">
      <div class="checkbox">
        <label>
          <input type="checkbox"> 记住密码
        </label>
      </div>
    </div>
  </div>
  <div class="form-group">
    <div class="col-sm-offset-2 col-sm-10">
      <button type="submit" class="btn btn-default">登录</button>
    </div>
  </div>
</form>
```

如果需要制作排成一行的表单，并有合适的间距，且可以适应不同尺寸窗口的显示，如图 12.24 这样的效果。

图 12.24　内联表单样式

Bootstrap 的解决方案是为<form>元素添加 form-inline 类：

```
<form class="form-inline" role="form">
  <div class="form-group">
……
  <button type="submit" class="btn btn-default">登录</button>
</form>
```

12.4.4　按钮

1．按钮基本样式

Bootstrap 提供了一组标准的按钮配色和大小调整方案，只需要简单地应用相应的按钮类即可。Bootstrap 3 提供的按钮标准样式如图 12.25 所示。

图 12.25　Bootstrap 3 中的按钮样式

下面是对应图 12.25 的代码

```
<!-- 标准按钮样式 -->
<button type="button" class="btn btn-default">默认</button>

<!-- 表示主要的按钮 -->
<button type="button" class="btn btn-primary">主要</button>

<!-- 表示成功的按钮 -->
<button type="button" class="btn btn-success">成功</button>

<!-- 表示消息提示 -->
<button type="button" class="btn btn-info">信息</button>

<!-- 表示警告 -->
<button type="button" class="btn btn-warning">警告</button>

<!-- 表示危险操作 -->
<button type="button" class="btn btn-danger">危险</button>

<!--使其看起来像一个链接，同时保持按钮的行为 -->
<button type="button" class="btn btn-link">Link</button>
```

按钮类的总结参见表 12.3。

表 12.3　按钮类说明

意义	按钮类	颜色和样式
标准样式	btn btn-default	白色

续表

意义	按钮类	颜色和样式
主要按钮	btn btn-pri	深蓝色
成功	btn btn-success	绿色
消息	btn btn-info	淡蓝色
警告	btn btn-warning	橙黄色
危险操作	btn btn-danger	红色
链接	btn btn-link	和链接的样式一样

注意：按钮类不仅可以用于<button>标签，还可以用于<a>或<input>标签，它们的样式表现是一致的，不过出于浏览器表现一致性的考虑，Bootstrap 推荐使用<button>标签。

2. 调节按钮大小

Bootstrap 还提供了.btn-lg、.btn-sm 和.btn-xs 3 个类对按钮的大小进行标准化的调节，如图 12.26 所示。

图 12.26　调节按钮大小

图 12.26 分别显示了大按钮(.btn-lg)、标准按钮、小按钮(.btn-sm)、超小按钮(.btn-xs)，其相应的代码如下：

```
<p>
  <button type="button" class="btn btn-primary btn-lg">大号按钮</button>
  <button type="button" class="btn btn-default btn-lg">大号按钮</button>
</p>
<p>
  <button type="button" class="btn btn-primary">默认按钮</button>
  <button type="button" class="btn btn-default">默认按钮</button>
</p>
<p>
  <button type="button" class="btn btn-primary btn-sm">小号按钮</button>
  <button type="button" class="btn btn-default btn-sm">小号按钮</button>
</p>
<p>
  <button type="button" class="btn btn-primary btn-xs">更小的按钮</button>
  <button type="button" class="btn btn-default btn-xs">更小的按钮</button>
</p>
```

3. 块级按钮

<button>或者<a>标签默认都是内联元素，而对于移动端的设计来说，一个占据一整行

的大按钮是再正常不过了，如图 12.27 所示，这种情况可以对按钮使用.btn-block 类

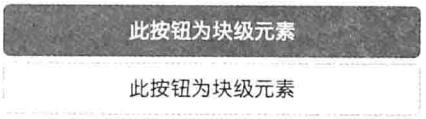


图 12.27　占据一整行的大按钮

图 12.27 对应的代码如下：

```
<button type="button" class="btn btn-primary btn-lg btn-block">此按钮为块级元素</button>
<button type="button" class="btn btn-default btn-lg btn-block">此按钮为块级元素</button>
```

4．为按钮设置不可点击样式

Bootstrap 通过将按钮的背景色做 50%的褪色处理以呈现出无法点击的效果，如图 12.28 所示。

图 12.28　无法点击的按钮效果

实现方法很简单，只需要为按钮添加 disabled 属性，代码如下：

```
<button type="button" class="btn btn-lg btn-primary" disabled="disabled">主要按钮</button>
<button type="button" class="btn btn-default btn-lg" disabled="disabled">按钮</button>
```

对于装饰成按钮样式的<a>标签，则需要添加.disabled 类，例如：

```
<a href="#" class="btn btn-primary btn-lg disabled" role="button">主要链接</a>
<a href="#" class="btn btn-default btn-lg disabled" role="button">链接</a>
```

注意：<a>标签添加了.disabled 类后只是样式发生变化，点击后仍然是可以正常工作的，一般情况下可以使用 JavaScript 来进行<a>标签的禁用。

12.4.5　图片

1．图片类

Bootstrap 的图片类包含了对最常用的圆角、圆形、简洁边框这 3 种图片形状的修正，常用于头像的处理，如图 12.29 所示。

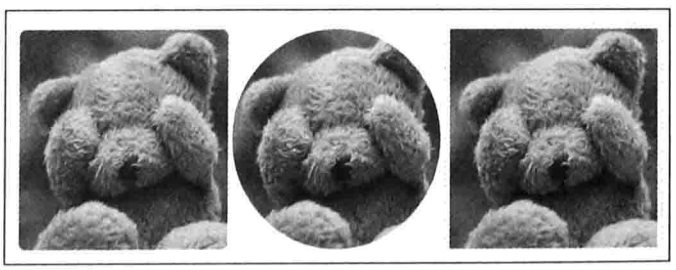


图 12.29　3 种最常用的图片形状

图 12.29 对应的代码如下：

```
<img src="test.png" class="img-rounded" />  <!--img-rounded 类用于构建带有圆角的图片-->
```

```
<img src="test.png" class="img-circle"/>        <!--img-circle 类用于构建圆形的图片-->
<img src="test.png" class="img-thumbnail"/> <!--img-thumbnail 类为图片添加简洁的边框-->
```

2．响应式图片

通过添加.img-responsive 类可以让 Bootstrap 3 中的图片对响应式布局的支持更友好。其实质是为图片赋予 max-width: 100%; 和 height: auto;属性，可以让图片按比例缩放不超过其父元素的尺寸，使用代码如下：

```
<img src="naicha.png" class="img-responsive" alt="Responsive image">
```

12.4.6 响应式工具

在第 8 章介绍响应式设计时，曾经介绍过响应式设计中一种常见的做法就是在小屏幕设备中隐藏或收起一些内容，以保证最主要内容的展现。比起在 CSS 中不同的媒介查询中手写 display:none，Bootstrap 提供了针对响应式布局的辅助类，来控制元素在不同尺寸屏幕下的显示/隐藏，详情见表 12.4 和表 12.5。

表 12.4　Bootstrap 2 中的响应式工具

class	手机767px及以下	平板979px到768px	电脑默认
.visible-phone	显示	隐藏	隐藏
.visible-tablet	隐藏	显示	隐藏
.visible-desktop	隐藏	隐藏	显示
.hidden-phone	隐藏	显示	显示
.hidden-tablet	显示	隐藏	显示
.hidden-desktop	显示	显示	隐藏

表 12.5　Bootstrap 3 中的响应式工具

class	超小屏幕手机（<768px）	小屏幕平板（≥768px）	中等屏幕桌面（≥992px）	大屏幕桌面（≥1200px）
.visible-xs	可见	隐藏	隐藏	隐藏
.visible-sm	隐藏	可见	隐藏	隐藏
.visible-md	隐藏	隐藏	可见	隐藏
.visible-lg	隐藏	隐藏	隐藏	可见
.hidden-xs	隐藏	可见	可见	可见
.hidden-sm	可见	隐藏	可见	可见
.hidden-md	可见	可见	隐藏	可见
.hidden-lg	可见	可见	可见	隐藏

和响应式 class 一样，使用表 12.6 所列的 class 可以针对打印机隐藏或显示某些内容。

表 12.6　针对打印机隐藏或显示

class	浏览器	打印机
.visible-print	隐藏	可见
.hidden-print	可见	隐藏

例如，有一个工具栏，在电脑上浏览时展开显示，在移动设备上浏览器时则收起变成一个按钮，在 Bootstrap 2 下就可以这样写：

```
<div class="toobar visible-desktop">
  .....<!-- 工具栏内容只在宽度 980 像素以上窗口上显示-->
</div>

<button class="toobar-button hidden-desktop"></button>
.       <!-- 工具栏按钮只在宽度 980 像素以下窗口上显示-->
```

同样的场景，Bootstrap 3 则是这样的：

```
<div class="toobar hidden-xs hidden-sm">
  .....<!-- 工具栏内容在小窗口及超小窗口下不显示-->
</div>

<button class="toobar-button visible-md visible-lg "></button>
.       <!-- 工具栏按钮只在中等窗口和大窗口下显示-->
```

12.4.7　工具类

工具类主要提供一些常用的效果，直接在 HTML 元素中添加类显然在开发效率上比在 CSS 文件中添加属性要高得多。Bootstrap 提供了以下工具类。

- 关闭按钮：对按钮元素应用.close 类就可以显示一个关闭按钮，如图 12.30 右上角。

```
<button type="button" class="close" aria-hidden="true">&times;</button>
```

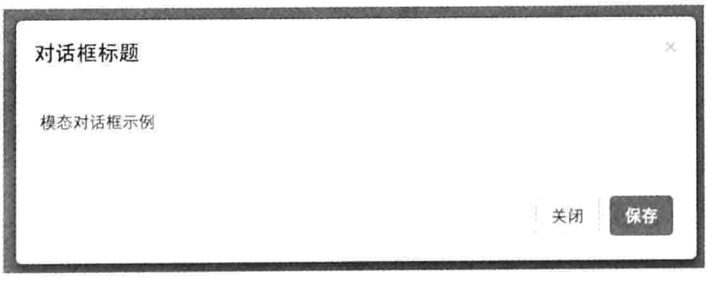


图 12.30　右上角的关闭按钮

- 下拉按钮：对元素应用 caret 类就可以显示一个下拉符号，如图 12.31 所示。

```
<span class="caret"></span>
```

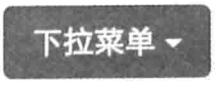


图 12.31　下拉符号

- 左浮动、右浮动：为元素添加.pull-left、.pull-right 类就可以设置左浮动/右浮动。

```
<div class="pull-left">...</div>
<div class="pull-right">...</div>
```

内部的实现方法如下：

```
.pull-left { float: left !important; }
.pull-right { float: right !important; }
```

用!important 实现强制覆盖原样式。

注意：如果是用于对齐导航条上的组件，请务必使用.navbar-left 或.navbar-right。

- 清除浮动：使用.clearfix 类清除任意页面元素的浮动，例如：

```
<div class="clearfix">...</div>
```

- 显示/隐藏：使用.show 类显示，.hidden 类隐藏，例如：

```
<div class="show">...</div>   <!--显示-->
<div class="hidden">...</div>   <!--隐藏-->
```

- 内容区域居中：使用.center-block 类将元素设为块级元素并居中，例如：

```
<div class="center-block">...</div>
```

- 对除了屏幕阅读器的设备隐藏：使用.sr-only 类，例如：

```
<a class="sr-only" href="#content">Skip to main content</a>
```

说明：屏幕阅读器是一种为阅读障碍人群开发的设备，可以识别页面元素并发声阅读。

.sr-only 类最常见的应用场景是针对表单中的<label>元素，<label>对于一般设备来说只是起一个提示作用，但是对于屏幕阅读器来说，只有带有<lable>标签的表单才会被识别，如果既需要被屏幕阅读器识别，而又要在其他设备上隐藏<lable>标签，就要对<lable>应用.sr-only 类。

12.5 使用 Bootstrap 的组件

前面讲解了 Bootstrap 的基本样式以及内置的一些 class，它们的用法基本上是一致的，只需要为页面元素添加相应的 class，就可以轻松地为元素设置一个美观的样式。对于一些较为常用的功能模块，Bootstrap 也进行了相应的封装，在用法上就不仅仅是添加 class 那么简单了，需要编写遵循 Bootstrap 约定的结构。

12.5.1 下拉菜单

不同于表单中的 select 标签，下拉菜单中的选择一般对应网页中的链接或者 Ajax 模块，完整的下拉菜单功能由触发下拉的按钮和下拉列表两部分组成，如图 12.32 所示。

图 12.32 下拉菜单

图 12.32 的代码如下：

```
<div class="dropdown">
  <button class="btn btn-primary dropdown-toggle" type="button" id="dropdownMenu1" data-toggle=
    "dropdown">
    下拉菜单
    <span class="caret"></span>  <!-- 下拉按钮的向下箭头-->
  </button>
  <ul class="dropdown-menu" >
    <li><a href="#">选项一</a></li>
    <li><a href="#">选项二</a></li>
    <li><a href="#">选项三</a></li>
    <li class="divider"></li>
    <li><a href="#">选项四</a></li>
  </ul>
</div>
```

分析这段代码的结构：

- 按钮和下拉选择都要包裹在 …… 内
- 按钮必须添加 data-toggle="dropdown"触发器
- 放置下拉选项的无序列表需要添加.dropdown-menu 类
- 添加一个空的<li class="divider">
标签来分隔列表项

注意：官方网站给出的示例代码会添加很多 role="……"的属性，这些不是必须的，但在实际应用中最好加上以提升可访问性。

12.5.2　按钮组

按钮组用于把一组按钮放在同一行里。基本的用法很简单，只需要将一组按钮放在 …… 中即可，例如：

```
<div class="btn-group">
  <button type="button" class="btn btn-default">Left</button>
  <button type="button" class="btn btn-default">Middle</button>
  <button type="button" class="btn btn-default">Right</button>
</div>
```

效果如图 12.33 所示。

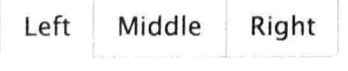


图 12.33　按钮组

注意：当为按钮组中的元素应用工具提示或弹出框时，必须指定 container: 'body'选项，这样可以避免不必要的副作用（例如工具提示或弹出框触发时，会让页面元素变宽或失去圆角）。

按钮组支持垂直排列、两端对齐、嵌套。

（1）垂直排列：

```
<div class="btn-group-vertical"> ... </div>
```

应用. btn-group-vertical 类可以让按钮组垂直排列。

（2）两端对齐（按钮组拉伸至 100%宽度如图 12.34 所示）：

```
<div class="btn-group btn-group-justified">
  <a class="btn btn-default">Left</a>
  <a class="btn btn-default">Middle</a>
  .....
</div>
```

图 12.34　两端对齐的按钮组

注意：两端对齐的用法只适用<a>元素，因为<button>元素不能应用这些样式并将其所包含的内容两端对齐。

（3）嵌套，可以在按钮组内继续内嵌按钮组，例如：

```
<div class="btn-group">
  <button type="button" class="btn btn-default">1</button>
  <button type="button" class="btn btn-default">2</button>

  <div class="btn-group">
    <button type="button" class="btn btn-default dropdown-toggle" data-toggle="dropdown">
      Dropdown
      <span class="caret"></span>
    </button>
    <ul class="dropdown-menu">
      <li><a href="#">Dropdown link</a></li>
      <li><a href="#">Dropdown link</a></li>
    </ul>
  </div>
</div>
```

效果如图 12.35 所示。

图 12.35　嵌套后的按钮组

12.5.3　input 控件组

input 输入框经常会和其他元素配合使用，最常见的 input 控件组肯定非搜索框莫属了，Bootstrap 的 input 控件组包含了多数常见的分组类型。

Bootstrap 中控件组的共同点是都需要包裹在 ..... 内部，

下面将逐个解析控件组的各种不同组合。

（1）首先是最常见的搜索框，就是按钮+input 表单的组合：

```
<div class="input-group">
  <input type="text" class="form-control">
  <span class="input-group-btn">
    <button class="btn btn-default" type="button">Search</button>
  </span>
</div>
```

效果如图 12.36 所示。

图 12.36　搜索框

其实质就是 input 表单+按钮，需要注意的是要在按钮外包裹一层……。如果需要带下拉菜单的按钮，则只需要将按钮换成下拉菜单即可。

（2）如果和 input 配合的不是可单击的按钮，只是用于说明的文字或图片的话，则可以如下应用：

```
<div class="input-group">
  <input type="text" class="form-control">
  <span class="input-group-addon">输入完成后回车</span>
</div>
```

效果如图 12.37 所示，只需将提示文字放到……内即可。

图 12.37　搜索框后不是按钮

12.5.4　导航

Bootstrap 的导航主要分为胶囊式导航、面包屑导航、头部导航 3 类，可以满足大多数的开发需求。

1．胶囊式导航

胶囊式导航一般用于平级的选项列表，如图 12.38 所示。

图 12.38　横向的胶囊导航

胶囊导航实质是一个无序列表，只需要给 ul 元素添加.nav 和.nav-pill 类即可，例如：

```
<ul class="nav nav-pills">
  <li class="active"><a href="#">首页</a></li>  <!—active 表示已选中的选项-->
  <li><a href="#">简介</a></li>
  <li><a href="#">详情</a></li>
</ul>
```

如果需要纵向的胶囊导航，只需要为 ul 元素追加.nav-stacked 类即可。

```
<ul class="nav nav-pills nav-stacked">
  <li class="active"><a href="#">首页</a></li>  <!—active 表示已选中的选项-->
  <li><a href="#">简介</a></li>
  <li><a href="#">详情</a></li>
</ul>
```

纵向的胶囊导航如图 12.39 所示。

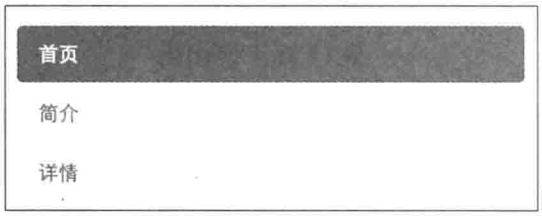


图 12.39　纵向的胶囊导航

2．面包屑导航

面包屑导航一般用于有层级关系的选项，如图 12.40 所示。

图 12.40　面包屑导航

面包屑导航同样采用了列表结构，这里需要为 ul 或 ol 元素添加.breadcrumb 类来实现，例如：

```
<ol class="breadcrumb">
  <li><a href="#">首页</a></li>
  <li><a href="#">资料库</a></li>
  <li class="active">数据</li>
</ol>
```

3．头部导航

绝大多数网站首页的页头部分都会放置一个针对主要内容的导航，让用户可以快速了解网站的内容和结构。Bootstrap 的头部导航如图 12.41 所示。

图 12.41　头部导航

头部导航的基本结构如下：

```
<nav class="navbar navbar-default" >
```

```
<div class="navbar-header">          <!--这里设置网站的标题-->
  <a class="navbar-brand" href="#">网站 Logo</a>
</div>
<div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
  <!--这里设置网站的链接、表单等其他元素-->
  <ul class="nav navbar-nav">
    <li class="active"><a href="#">链接</a></li>
    <li><a href="#">链接</a></li>
  </ul>
  <ul class="nav navbar-nav navbar-right">
    <li><a href="#">链接</a></li>
  </ul>
</div>
</nav>
```

具体分析头部导航主要分为两层结构，第一层是最外面的<nav class="navbar navbar-default" >……</nav> ，这一层用于设置导航的基本样式，如果将.navbar-default 类替换为. navbar-inverse 类，则显示为反色的导航（黑底白字），如图 12.42 所示。

图 12.42　反色的头部导航

第二层有两个并列的元素： …… 内部用于设置标题内容； …… 内部则用于编写具体的导航链接、搜索表单、下拉菜单等具体的导航内容。

Bootstrap 3 提供了在小窗口下导航收起/展开的功能，如图 12.43 所示。

图 12.43　头部导航小窗口下收起/展开

图 12.43 需要在 中进行设置展开/收起的按钮，例如：

```
<nav class="navbar navbar-inverse" role="navigation">
  <div class="navbar-header">
```

```
    <button class="navbar-toggle" data-toggle="collapse" data-target="#bs-example">
      <span class="sr-only">Toggle navigation</span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
    </button>
    <a class="navbar-brand" href="#">网站 Logo</a>
  </div>

  <!-- Collect the nav links, forms, and other content for toggling -->
  <div class="collapse navbar-collapse" id="bs-example">
    <ul class="nav navbar-nav">
      <li class="active"><a href="#">链接</a></li>
      <li><a href="#">链接</a></li>
      <li class="dropdown">
        <a href="#" class="dropdown-toggle" data-toggle="dropdown">下拉菜单</a>
        <ul class="dropdown-menu">
          <li><a href="#">Action</a></li>
          <li><a href="#">Another action</a></li>
          <li><a href="#">Something else here</a></li>
          <li class="divider"></li>
          <li><a href="#">Separated link</a></li>
          <li class="divider"></li>
          <li><a href="#">One more separated link</a></li>
        </ul>
      </li>
    </ul>
    <form class="navbar-form navbar-left" role="search">
      <div class="form-group">
        <input type="text" class="form-control" placeholder="Search">
      </div>
      <button type="submit" class="btn btn-default">搜索</button>
    </form>
    <ul class="nav navbar-nav navbar-right">
      <li><a href="#">链接</a></li>
      <li class="dropdown">
        <a href="#" class="dropdown-toggle" data-toggle="dropdown">下拉菜单 </a>
        <ul class="dropdown-menu">
          <li><a href="#">Action</a></li>
          <li><a href="#">Another action</a></li>
          <li><a href="#">Something else here</a></li>
          <li class="divider"></li>
          <li><a href="#">Separated link</a></li>
        </ul>
      </li>
```

```
    </ul>
  </div>
</nav>
```

添加.navbar-fixed-top 可以让导航条固定在顶部，不会随页面滚动而消失。为了防止固定在顶部后遮挡正常内容，需要设置：body { padding-top: 70px; }，其中具体的值取决于导航条的高度。

注意：这个响应式的导航依赖 Bootstrap 的 collapse（折叠）插件。为导航条加上 role="navigation"可以提高访问的兼容性。

12.5.5　列表组

列表组不仅仅是对列表项进行美化，还可以支持任意内容的列表化展示，图 12.44 是未经修饰的无序列表和应用了列表组的列表的对比。

图 12.44　未经修饰的无序列表和列表组

对于列表来说，列表组结构如下：

```
<ul class="list-group" >
  <li class="list-group-item"><a href="#">选项一</a></li>
  <li class="list-group-item"><a href="#">选项二</a></li>
  <li class="list-group-item"><a href="#">选项三</a></li>
  <li class="list-group-item"><a href="#">选项四</a></li>
</ul>
```

需要为 ul 或 ol 元素添加.list-group 类，同时需要为列表项添加.list-group-item 类。

注意：在列表组中使用有序列表时不会显示序号。

列表组不仅可以应用于列表，还可以将其他需要列表的元素展现为列表的样子，如图 12.45 所示。

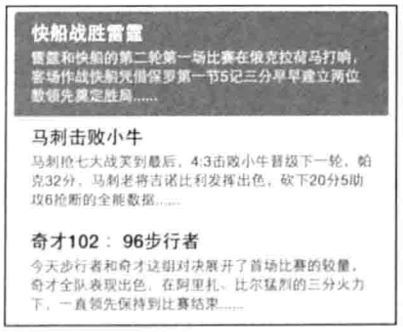


图 12.45　非列表但是展现为列表的样子

图 12.45 的代码如下：

```
<div class="list-group">
  <a href="#" class="list-group-item active">
    <h4 class="list-group-item-heading">快船战胜雷霆</h4>
    <p class="list-group-item-text">...</p>
  </a>
  ......
</div>
```

为列表组添加徽章也十分容易，Bootstrap 会自动将徽章放置在右边，如图 12.46 所示。

图 12.46　为列表组添加徽章

为列表组添加徽章的代码：

```
<ul class="list-group">
  <li class="list-group-item">
    <span class="badge">14</span>    <!—即使将徽章放在前面，最终还是会居右放置-->
    中国队金牌
  </li>
</ul>
```

12.5.6　分页

几乎所有的列表页面都需要分页，Bootstrap 提供了一个较为美观的分页样式，如图 12.47 所示。

图 12.47　分页

图 12.47 的代码结构比较简单，只需给无序列表的 ul 元素添加 pagination 类即可，例如：

```
<ul class="pagination">
  <li class= "disabled"><a href="#">&laquo;</a></li>  <!-- disabled 类表示不可点击项-->
  <li class="active"><a href="#">1</a></li>       <!-- active 类表示已选择项-->
  <li><a href="#">2</a></li>
  <li><a href="#">3</a></li>
  <li><a href="#">4</a></li>
  <li><a href="#">5</a></li>
  <li><a href="#">&raquo;</a></li>
</ul>
```

可以通过添加.pagination-lg 类或.pagination-sm 类来获得比标准尺寸更大或更小的分页，例如：

```
<ul class="pagination pagination-lg">...</ul>
<ul class="pagination">...</ul>
<ul class="pagination pagination-sm">...</ul>
```

效果如图 12.48 所示。

图 12.48　不同大小的分页栏

如果仅仅想使用上一页/下一页的功能怎么办？Bootstrap 也内置了该功能，为无序列表的 ul 添加.paper 类即可，例如：

```
<ul class="pager">
  <li><a href="#">上一页</a></li>
  <li><a href="#">下一页</a></li>
</ul>
```

效果如图 12.49 所示。

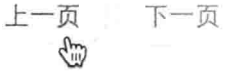


图 12.49　翻页效果

注意：翻页元素默认居中对齐，如果为列表元素添加.previous 和.next 类，可以将上一页/下一页按钮设置为两端对齐。

12.5.7　标签与 Badge

标签一般用于对内容进行标记，常用在内容审核后台，如图 12.50 所示。

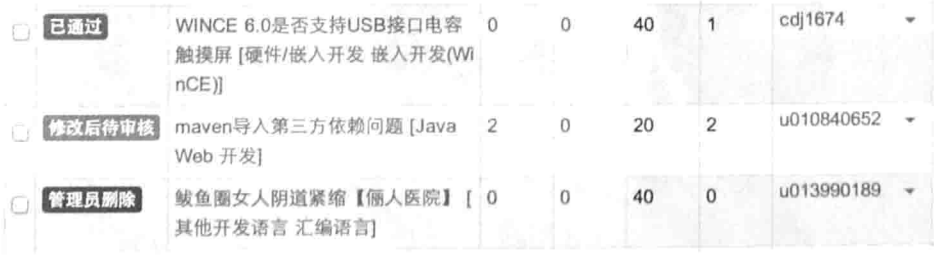


图 12.50　内容审核后台

Bootstrap 内置了 6 种常用的标签类，分别为 default（默认）、primary（主要）、success（成功）、info（消息）、warning（告警）、danger（危险操作）。它们分别对应不同的颜色，例如：

```
<span class="label label-default">Default</span>
<span class="label label-primary">Primary</span>
<span class="label label-success">Success</span>
<span class="label label-info">Info</span>
<span class="label label-warning">Warning</span>
<span class="label label-danger">Danger</span>
```

上面的代码效果如图 12.51 所示。

图 12.51　6 种标签类

除了标签之外，还有一种提示信息很常用，很多网站都有消息系统来提示用户有未读的新闻、私信等内容，Bootstrap 中称之为 badge（徽章），如图 12.52 所示。

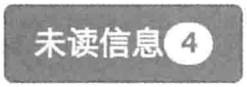


图 12.52　徽章

图 12.52 的代码如下：

```
<button class="btn btn-primary" type="button">
  未读信息
  <span class="badge">4</span>
</button>
```

badge 的应用很简单，只需要给行内元素添加.badge 类即可。

注意：当没有新的或未读的条目时，没有内容的徽章会消失（通过 CSS 的:empty 选择器实现），在 IE 9 版本以下的 IE 浏览器徽章不会自动消失，因为不支持:empty 选择器。

12.5.8　缩略图

在 Bootstrap 中，配合栅格系统可以很容易地构建带链接的缩略图，并让缩略图支持响应式设计，如图 12.53 所示。

图 12.53　缩略图

示例代码如下：

```
<div class="row">
  <div class="col-xs-2">
    <a href="#" class="thumbnail">
      <img src="http://pic16.nipic.com/20110826/8064486_224449386194_2.jpg">
    </a>
  </div>
  <div class="col-xs-2">
    <a href="#" class="thumbnail">
      <img src="http://pic16.nipic.com/20110826/8064486_224449386194_2.jpg">
    </a>
  </div>
  <div class="col-xs-2">
    <a href="#" class="thumbnail">
      <img src="http://pic16.nipic.com/20110826/8064486_224449386194_2.jpg">
    </a>
  </div>
</div>
```

构建缩略图整体上还是依赖 Bootstrap 的栅格系统，通过栅格系统来控制缩略图占据的宽度比例，保证缩略图集的响应式。这里需要给图片链接加上. thumbnail 类添加边框样式并调节图片间距。

如果将上例中的<a href="#" class="thumbnail">标签改为 ，就可以在图片下方追加内容，如图 12.54 所示。

图 12.54　为缩略图追加内容

图 12.54 的代码如下：

```
<div class="row">
  <div class="col-xs-2">
    <div class="thumbnail">
      <img src="http://pic16.nipic.com/20110826/8064486_224449386194_2.jpg">
      <div class="caption">
        <p>耳机很好，不错的低端耳机，一般玩电脑足够了。材质...</p>
        <p><a href="#" class="btn btn-primary" role="button">查看详情</a></p>
```

```
        </div>
      </div>
    </div>
    ……
</div>
```

12.5.9 面板

很多时候需要将某些内容放到一个容器里，此时可以使用 Bootstrap 的面板组件，一个最简单的面板如图 12.55 所示。

图 12.55　面板基础样式

可以看到，面板的作用就是加上了容器的边框并设置了内容和容器间的边距，对应最简单面板样式的代码如下：

```
<div class="panel panel-default">
  <div class="panel-body">基础面板示例</div>
</div>
```

我们还可以为面板添加 header 和 footer，如图 12.56 所示。

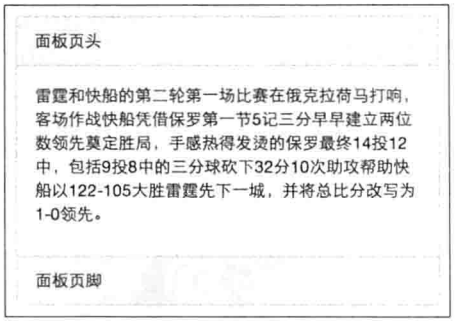


图 12.56　添加了 header 和 footer 的面板

带有 header 和 footor 的面板代码如下：

```
<div class="panel panel-default">
  <div class="panel-heading">面板页头</div>
  <div class="panel-body">面板内容省略...</div>
  <div class="panel-footer">面板页脚</div>
</div>
```

面板代码的配色和之前介绍的标签是一致的，都是对应诸如 success、warning、danger 等状况的颜色，从代码中可以很容易看出来：

```
<div class="panel panel-primary">...</div>
<div class="panel panel-success">...</div>
<div class="panel panel-info">...</div>
<div class="panel panel-warning">...</div>
<div class="panel panel-danger">...</div>
```

最终效果如图 12.57 所示。

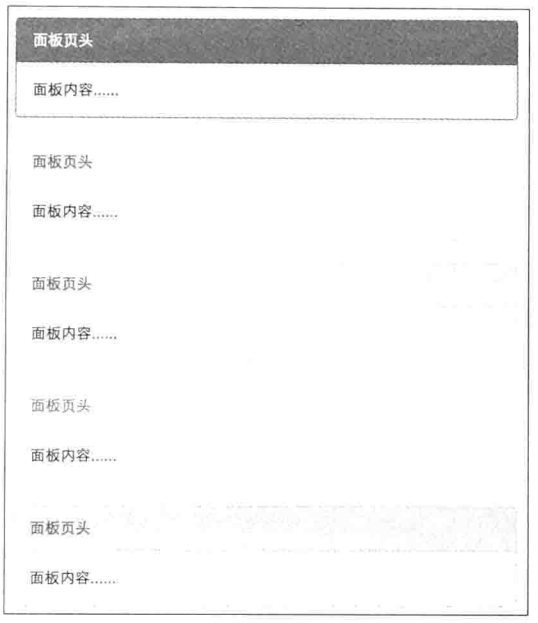


图 12.57　不同配色的面板

12.5.10　进度条

进度条常用于文件的上传/下载、内容的加载等场景，Bootstrap 提供了多种进度条样式供选择。

注意：Bootstrap 以及其他前端组件只解决进度条的样式问题，追踪进度仍需依赖服务端程序。

Bootstrap 中，一个标准的进度条如图 12.58 所示。

图 12.58　标准的进度条

最简单的实现代码如下：

```
<div class="progress">
    <div class="progress-bar" role="progressbar"   style="width: 60%;"></div>
</div>
```

为外层的 div 元素添加.progress 类，为内层的 div 元素添加.progress-bar 类，并控制内层 div 的宽度百分比，这样就得到一个基础的进度条了。

进度条的颜色既可以自己根据需要用自定义的颜色进行覆盖，也可以调用 Bootstrap 内置的类来覆盖。

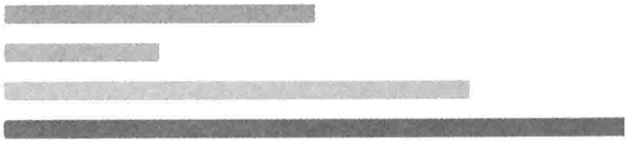


图 12.59　控制进度条色彩

为内层的 div 元素添加.progress-bar-success 就可以获得如图 12.59 中第一个进度条的颜色，其命名规律和 Bootstrap 的标签类是一致的。

```
<div class="progress">
  <div class="progress-bar progress-bar-success" style="width: 40%"></div>
</div>
<div class="progress">
  <div class="progress-bar progress-bar-info" style="width: 20%"></div>
</div>
<div class="progress">
  <div class="progress-bar progress-bar-warning" style="width: 60%"></div>
</div>
<div class="progress">
  <div class="progress-bar progress-bar-danger" style="width: 80%"></div>
</div>
```

我们还可以为进度条添加条纹效果，如图 12.60 所示。

图 12.60　条纹效果的进度条

实现条纹效果需要为外层的 div 添加. progress-striped 类，例如：

```
<div class="progress progress-striped">
  <div class="progress-bar progress-bar-info"style="width: 20%"></div>
</div>
```

说明：如果需要条纹带有运动效果，则为外层的 div 追加.active 类。

12.6　Bootstrap 中的 JavaScript 特效

Bootstrap 的流行很大程度上得益于它大大降低了页面开发的学习成本，很多时候，JavaScript 效果是一些非专业人士或美工出身的站长最头疼的问题，从零开始学习一门真正的编程语言，到能够在实战中实现足够好的效果，这个时间和学习成本是相当高的。而 Bootstrap 实现了从 UI 到 JavaScript 代码的一体化，只要在 HTML 代码中加入框架约定的触发器，就可以实现大多数的效果。即使用到官方没有提供的效果，开源社区的开发者也往往能提供现成的解决方案。

下面就来看看 Bootstrap 提供的几种常用效果吧。

12.6.1　模态对话框

如果想让用户在当前页面完成某种稍显复杂的操作，譬如登录、注册，或者是阅读一

段用户说明，模态对话框（如图 12.61 所示）是一个不错的选择。用户在操作或阅读完毕后可以很方便地返回原页面，免去了页面跳转带来的等待。

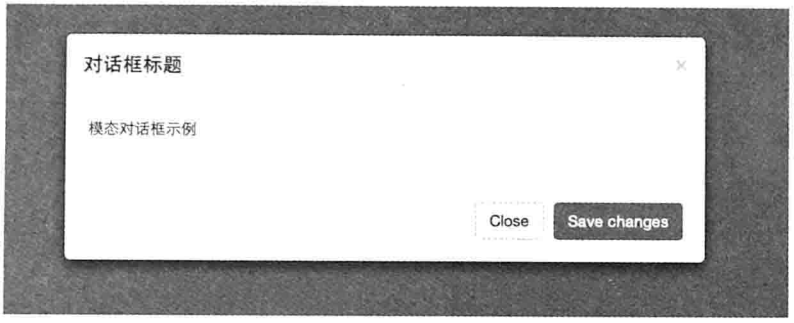


图 12.61　模态对话框

构造模态对话框的代码如下：

```
<button class="btn btn-primary btn-lg" data-toggle="modal" data-target="#myModal">
    点击触发模态对话框
</button>

<div class="modal fade" id="myModal" tabindex="-1" role="dialog" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal" >&times;</button>
        <h4 class="modal-title" id="myModalLabel">对话框标题</h4>
      </div>
      <div class="modal-body">模态对话框示例</div>
      <div class="modal-footer">
        <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div>
  </div>
</div>
```

根据代码分析，完整的模态对话框功能主要分为两个部分：触发按钮和对话框。

触发按钮可以是一个 button，也可以是一个链接，只需要加入两个元素：

- data-toggle="modal"触发器。
- data-target="#myModal"，用于和相应的对话框 id 进行对应。

对话框部分主要分 3 层。

- 第一层： …… ，这一层使用 class="modal" 设置样式并设置模态对话框的触发类，提供 id 和触发按钮的 data-target 属性的值进行对应，还可以添加其他的配置属性。
- 第二层： …… ，设置一个居中的对话框。
- 第三层： …… ，设置具体的内容。

注意：不要在一个模态框上重叠另一个模态框。

开发中，选项参数可以通过 data 属性或 JavaScript 进行传递。对于 data 属性，需要将选项名称放到 data-之后，例如 data-backdrop=""，具体的参数可以参考官方文档。

12.6.2 标签页切换

如果有多个分类的内容，又不想全部直接展现在页面上，使用标签页进行切换是一个不错的选择，如图 12.62 所示。

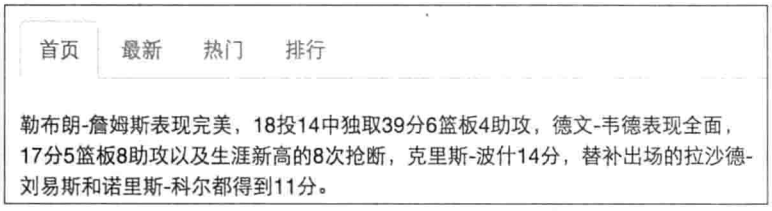


图 12.62 标签页切换

标签页切换的代码如下：

```
<ul class="nav nav-tabs">
  <li class="active"><a href="#home" data-toggle="tab">首页</a></li>
  <li><a href="#profile" data-toggle="tab">最新</a></li>
  <li><a href="#messages" data-toggle="tab">热门</a></li>
  <li><a href="#settings" data-toggle="tab">排行</a></li>
</ul>

<!-- Tab panes -->
<div class="tab-content">
  <div class="tab-pane active" id="home">
    <p>内容省略...</p>
  </div>
  <div class="tab-pane" id="profile">
    <p>内容省略...</p>
  </div>
  <div class="tab-pane" id="messages">热门</div>
  <div class="tab-pane" id="settings">排行</div>
</div>
```

标签页切换由两部分组成：标签页部分和与标签页对应的内容部分。

标签页部分本质是一个列表，为列表的 ul/ol 属性添加.nav 和.nav-tabs 类，使其展现为标签页的样式，列表项中的<a>链接需要加上 data-toggle="tab"这个触发器，并且 href 的值要和对应内容部分的 id 进行对应。

内容部分需要包裹在 …… 内部，保证除了应该显示的内容外，其他是隐藏的。内容的各个单项需要包裹在 …… 内部，并且要为 标签设置一个 id，用于与标签页的 href 属性的值对应。

12.6.3　Tooltip

Tooltip 插件的效果是鼠标悬停在目标元素上时，显示额外的提示，如图 12.63 所示。

图 12.63　鼠标悬停时的提示

示例代码如下：

```
<a href="#" data-toggle="tooltip" data-placement="right" title="Tooltip on right" class="btn btn-primary">工具提示</a>
```

其中 data-toggle="tooltip"是插件触发器，title 的内容是提示文字，data-placement 属性用于指定提示出现的位置。

要使该插件生效，需要在页面底部添加 JavaScript 代码完成初始化：

```
$(element).tooltip();
```

开发者可以为 tooltip()函数添加参数，或者在标签内添加“data-参数名”进行配置，比如上面例子中的 data-placement="right"。

12.6.4　弹出框

Tooltip 采用的是 hover 进行触发，多用于简单的提示，弹出框则通过点击触发，一般用于显示更多的内容，如图 12.64 所示。

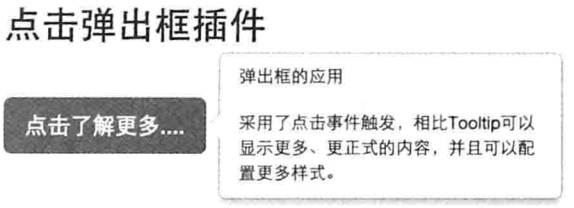


图 12.64　弹出框插件

应用弹出框插件的代码结构和 Tooltip 差不多，图 12.64 的实现代码如下：

```
<a href="javascript:void(0);" class="btn btn-lg btn-danger" data-toggle="popover" data-content="采用了点击事件触发，相比 Tooltip 可以显示更多、更正式的内容，并且可以配置更多样式。" data-original-title="弹出框的应用">点击了解更多....</a>
```

需要添加 data-toggle="popover"触发器进行触发，主要有两个配置项：data-content 配置弹出框内容，data-original-title 配置弹出框的标题。

弹出框对 Tooltip 存在依赖，因此插件中必须包含有 Tooltip。和 Tooltip 一样，也需要添加初始化 JavaScript 代码：

```
$( element ).popover(options)
```

和 Tooltip 一样，可以为 popover ()函数添加参数，或者在标签内添加“data-参数名”进行配置，比如上面 Tooltip 例子中的 data-placement="right"。

12.6.5 提示信息

一般来说，任务执行成功或失败后，用户需要得到一个提示信息，这个信息可以出现在页面跳转后的新页面，也可以是 Ajax 执行成功后的回调。但它们都有一个共同的特点：需要在阅读完毕后消失。Bootstrap 内置了警告框插件，使用户可以单击关闭按钮关掉提示信息。

示例代码如下：

```
<div class="alert alert-danger fade in">警告，服务器挂了！
  <a class="close" data-dismiss="alert" href="#" >X</a>
</div>
```

构造提示信息需要两个部分：提示信息和关闭信息按钮。提示信息这里使用了 Bootstrap 内置的 alert 类，关闭按钮则是在和提示信息文字并列的位置构造一个链接，为该链接添加 data-dismiss="alert"这个触发器来触发关闭事件，效果如图 12.65 所示。

图 12.65 提示信息

Bootstrap 为警告框的关闭动作暴露了事件，允许进行监听，可以再编写关闭警告框后执行的动作。

- close.bs.alert：当 close 函数被调用之后，此事件被立即触发。
- closed.bs.alert：当警告框被关闭之后（CSS 过渡效果执行完毕），此事件被触发。

示例代码：

```
$('#my-alert').bind('closed.bs.alert', function () { // do something… })
```

12.6.6 按钮

一般情况下，按钮用于完成动作的触发或页面的跳转，除此之外，Bootstrap 还通过 jQuery 插件对按钮的功能进行了一些扩展，比如按钮的状态设置、模拟 checkbox、radio 效果的按钮组等。

1. 按钮的 Loading 状态

按钮的 Loading 状态，一般用于需要 Ajax 请求的场景，防止多次点击，通过添加 data-loading-text="Loading..."，并编写 JavaScript 代码绑定事件来实现，例如：

```
<button id='loading-btn' data-loading-text="Loading..." class="btn btn-primary"> 提交 </button>
<script>
$('#loading-btn').click(function () {
    var btn = $(this)
    btn.button('loading')
});
</script>
```

效果如图 12.66 所示。

图 12.66　按钮的 Loading 状态

2．按钮组的状态设置

Bootstrap 支持按钮组的选中状态设置，类似于复选/单选表单。首先来看一个未经任何设置的按钮组，如图 12.67 所示。

图 12.67　未经任何设置的按钮组

图 12.67 中的代码如下：

```
<div class="btn-group">
    <button type="button" class="btn btn-default">左转弯</button>
    <button type="button" class="btn btn-default">直行</button>
    <button type="button" class="btn btn-default">右转弯</button>
</div>
```

默认状态下，按钮组中的按钮如果被单击，它会显示一个被选中的状态，但是这个状态是无法保持的，只要单击其他位置，该状态就会消失。

如果要让按钮的状态可以保持，需要为外层的 添加属性 data-toggle="buttons"，并在每一个按钮内部添加单选/复选表单，例如：

```
<div class="btn-group" data-toggle="buttons">
  <button type="button" class="btn btn-default">
    <input type="checkbox">左转弯
  </button>
  <button type="button" class="btn btn-default">
    <input type="checkbox">直行
  </button>
  <button type="button" class="btn btn-default">
    <input type="checkbox">右转弯
  </button>
</div>
```

代码效果如图 12.68（中）所示，如果将其中的<input type="checkbox">替换为<input type="radio">，则会将复选效果变为单选效果，如图 12.68（下）所示。

图 12.68　让按钮的状态可以保持

12.6.7 折叠

折叠用于内容的展开/收起，其功能同标签页类似，两者展开方向是一样的，都是向下展开内容，但是标签页的标题项是左右排列，而折叠（俗称手风琴效果）的标题项是上下排列的。而且折叠还可以同时展开多个项目的内容，而标签页只能同时展开一个。折叠效果如图 12.69 所示。

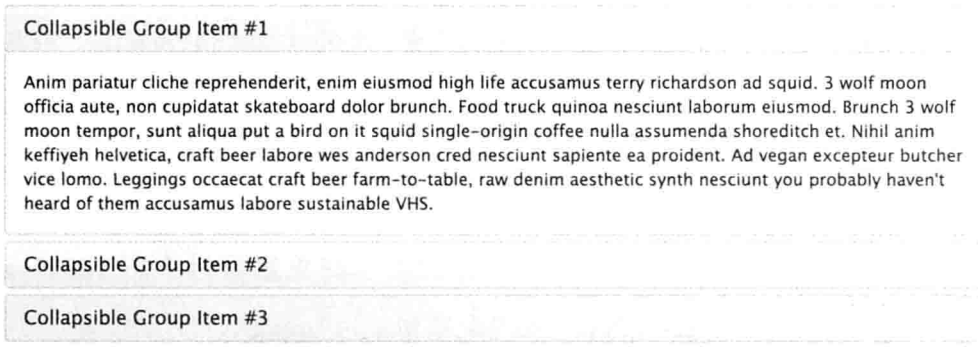


图 12.69 折叠

如果只是构建单个元素的展开收起，那么结构将非常简单：

```
<button class="btn btn-default" data-toggle="collapse" data-target="#demo">
  折叠标题
</button>
<div id="demo" class="collapse in">折叠内容</div>
```

只需要为标题容器添加 data-toggle="collapse"触发器并将 data-target 的值和折叠内容容器的 id 进行对应即可。

如果要构造如图 12.69 那样的折叠组，那么代码如下：

```
<div class="panel-group" id="accordion">
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">
        <a data-toggle="collapse" data-parent="#accordion" href="#collapseOne">
          折叠的标题部分
        </a>
      </h4>
    </div>
    <div id="collapseOne" class="panel-collapse collapse in">
      <div class="panel-body">
        折叠的内容部分
      </div>
    </div>
  </div>
  ……
<div>
```

折叠插件首先需要构建一个折叠组 …… ，所有的内容都

要放在这个组里。

组里的每一个项目实质上是一个面板，面板的结构我们在前面介绍过。不同点在于 panel-body（面板的内容）要包裹在 …… 内部。panel-heading（面板的标题部分）中要将标题文字放在链接<a data-toggle="collapse" data-parent="#accordion" href="#collapseOne">……<a>内部。该链接必须要有 data-toggle="collapse"这个触发器，data-parent="#accordion"用于和折叠组的 id 进行对应，href="#collapseOne"用于和面板内容外层 div 元素的 id 对应。

12.6.8　幻灯片

Bootstrap 集成了一个幻灯片组件，可以完成图片或内容的切换和自动播放，如图 12.70 所示。

图 12.70　幻灯片

根据图片的幻灯片页面结构由 3 部分组成：控制器、内容部分、标示符。控制器负责控制幻灯的翻页，标示符告诉我们页码，内容部分负责展现内容。具体的代码如下：

```
<div id="carousel-example-generic" class="carousel slide" data-ride="carousel">
  <!--标示符 -->
  <ol class="carousel-indicators">
    <li data-target="#carousel-example-generic" data-slide-to="0" class="active"></li>
    <li data-target="#carousel-example-generic" data-slide-to="1"></li>
    <li data-target="#carousel-example-generic" data-slide-to="2"></li>
  </ol>

  <!--包裹幻灯内容 -->
  <div class="carousel-inner">
    <div class="item active">
      <img src="..." alt="...">
      <div class="carousel-caption">
        ...
      </div>
    </div>
```

```
    ...
  </div>

  <!-- 控制器-->
  <a class="left carousel-control" href="#carousel-example-generic" data-slide="prev">
    <span class="glyphicon glyphicon-chevron-left"></span>
  </a>
  <a class="right carousel-control" href="#carousel-example-generic" data-slide="next">
    <span class="glyphicon glyphicon-chevron-right"></span>
  </a>
</div>
```

首先，所有内容都需要包裹在 …… 内部，如果需要开启轮播，则需要加入 data-ride="carousel"触发器。

标示符部分是一个列表，需要为 ol/ul 项添加.carousel-indicators 类。

内容部分需要整体包裹在 …… 内部，每一页的内容则需要包裹在 …… 内部。

控制器部分实际就是一个<a>链接，需要为这个链接加上. carousel-control 类，并添加一个.left 或.right 类指明向前翻页还是向后翻页。翻页的图标可以使用 Bootstrap 内置的图标：

```
<span class="glyphicon glyphicon-chevron-left"></span>
<span class="glyphicon glyphicon-chevron-right"></span>
```

也可以自定义图标的样式。

注意：Bootstrap 的幻灯片插件是基于 CSS 3 实现的动画效果，但是 IE 9 及 IE 9 以下的浏览器不支持这些必要的 CSS 属性，因此在 IE 下会丢失过渡动画效果。

和其他插件的参数配置一样，可以通过 data 属性或 JavaScript 传递选项参数。对于 data 属性，将选项名称放到 data-之后，例如 data-interval=""。

12.7 定制 Bootstrap

Bootstrap 提供两种定制化途径：一种是通过官方网站提供的定制化参数设置页面进行设置，设置完成后下载；另一种是下载 LESS 源代码，通过修改 LESS 变量然后自己编译生成 CSS。本节将分别介绍这两种定制化方式。

12.7.1 在官方网站进行 Bootstrap 的定制

首先进入官网，单击 Customize 选项，进入 Bootstrap 的定制页面，如图 12.71 所示。

向下滚动页面，首先会发现选择 CSS 组件的部分，如图 12.72 所示。在这里，可以只选择需要的组件下载，避免整体使用 Bootstrap 造成的 CSS 文件过大的问题。

图 12.71　官网定制页面

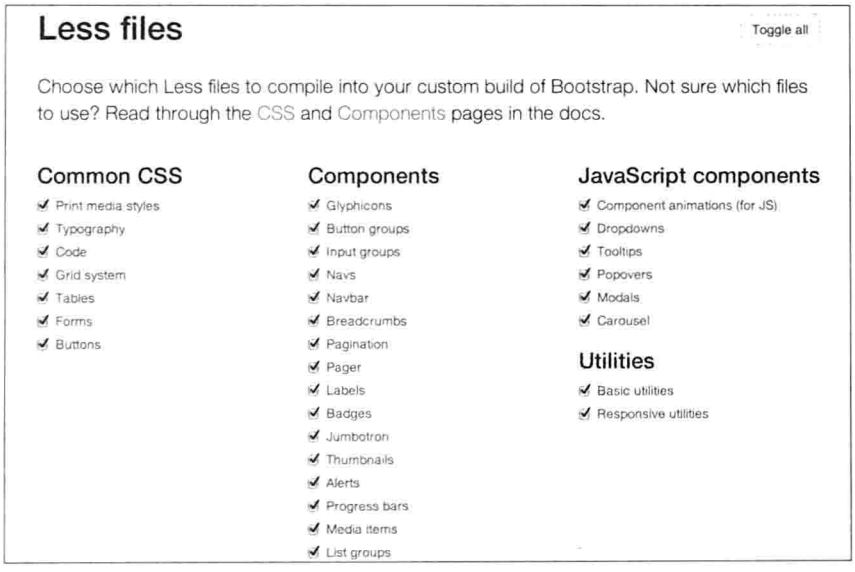


图 12.72　CSS 组件定制

继续向下滚动页面，如图 12.73 所示，这是定制 jQuery 组件的部分。对于大多数项目而言，其实用不到这么多的效果，可以在这里选择自己需要的组件；又或者你压根不想使用 Bootstrap，但又觉得其中的一些 JS 效果正好用得上，也可以在这里直接下载对应的 jQuery 插件。

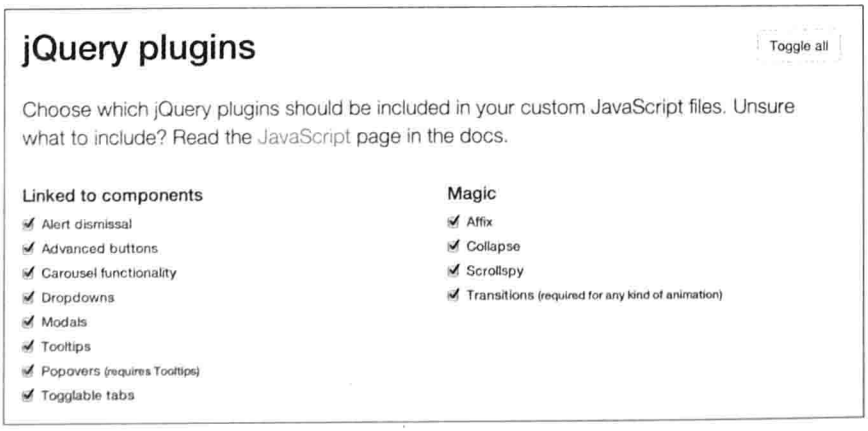


图 12.73　jQuery 组件定制

最后就是全局变量的设置了，如图 12.74 所示。在这里，可以对 Bootstrap 的全局样式进行调整，获得想要的配色效果，还可以制作自己的 Bootstrap 皮肤。

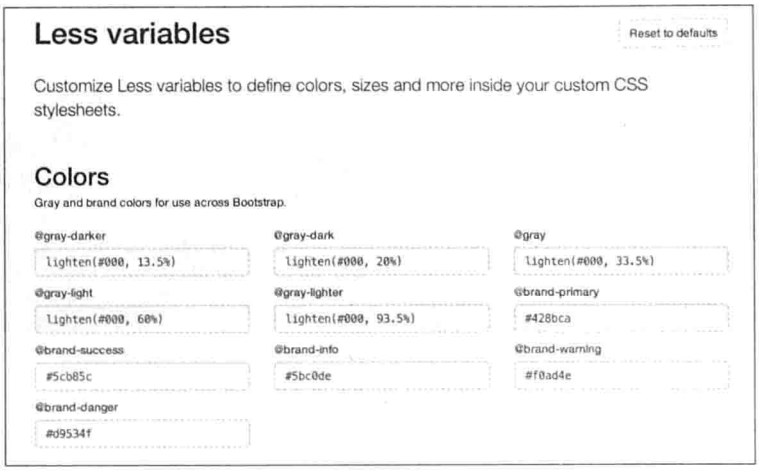


图 12.74　全局变量设置

根据自己的要求定制完成后，单击页面最后的下载按钮就完成定制了。

官方网站提供的定制化从使用角度来说还是很方便的，但是要求你必须已经对定制方案胸有成竹，因为网站无法保存之前的定制化信息，定制化下载后发现有些地方需要修改，就必须重复上面的过程，这无疑是非常麻烦的。

12.7.2　修改源代码定制 Bootstrap

首先需要做一些准备工作，由于 Bootstrap 使用了 Grunt 作为自动化打包测试工具，而 Grunt 是依赖于 Node.js 的，所以需要先在电脑上安装 Node.js。

（1）在官方下载页面（图 12.75），单击 Download source 按钮，下载源代码。

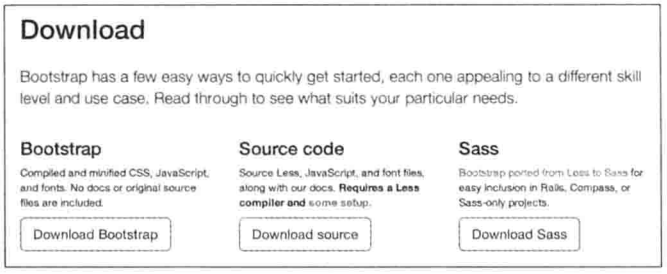


图 12.75　Node.js 官方下载页面

（2）下载完成后解压，解压后的文件夹如图 12.76 所示。

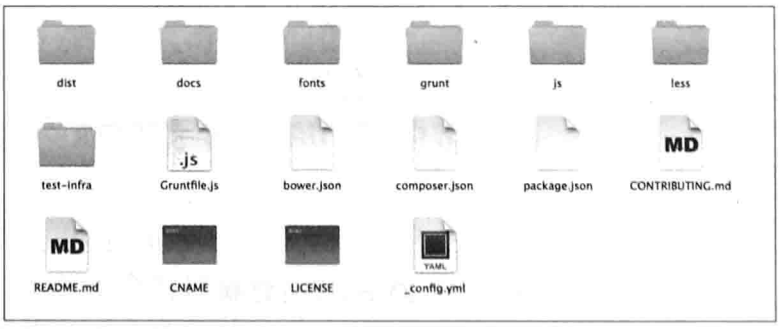


图 12.76　Bootstrap 源代码

其中 less 文件夹中包含了 Bootstrap 中所有样式组件的 less 源代码，js 文件夹中包含了

所有 jQuery 插件，fonts 文件夹用于保存字体文件，dist 文件夹用于保存编译后的 CSS、JavaScript 和字体文件。

（3）需要在命令行中进入该文件夹，执行命令：

```
npm install
```

该命令用于安装自动化管理 Bootstrap 的 Grunt 插件，安装完成后会生成 node_modules 文件夹，所有依赖的 Node.js 插件都安装在这里。

注意：由于默认的安装源服务器在国外，可能由于网络原因安装失败，可以在命令行中使用：npm config set registry http://registry.cnpmjs.org 这个命令，将下载源改为国内。

（4）要完成 CSS 的定制化，需要进入 less 文件夹（图 12.77），修改 variables.less 和 bootstrap.less 文件。其中 variables.less 是全局变量的配置文件，bootstrap.less 是加载项的配置文件。

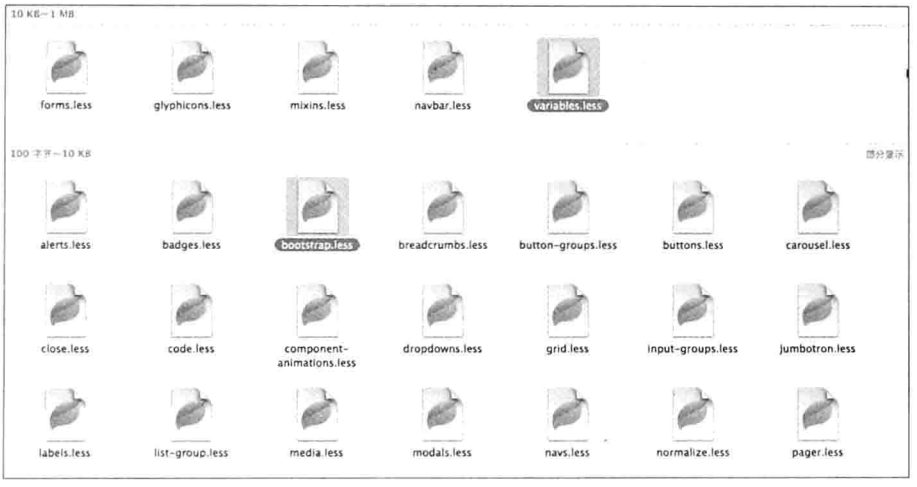


图 12.77　less 文件夹中的组件

（5）打开 bootstrap.less 文件，可以看到如图 12.78 所示的结构。一般来说，我们可以配置组件部分，也就是图中注释“// Components”后面的代码，将其中一些用不到的注释去掉。而对于核心组件，建议保留原貌。

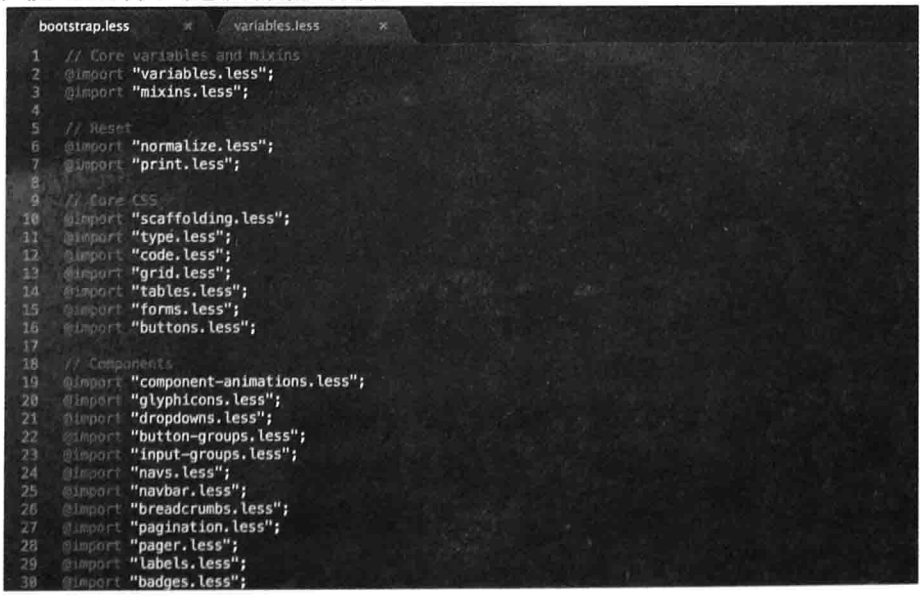


图 12.78　bootstrap.less 文件

（6）如果只是想单独使用某个组件，那只需要将 js 文件夹中对应的组件应用到项目中即可。如果需要应用多个组件，并将其打包处理，则通过修改 Gruntfile.js 配置文件（图 12.79）移除那些用不上的 JavaScript 插件，或者添加新的插件。

bootstrap.less | variables.less | Gruntfile.js

```
81      csslint: {
82        options: {
83          csslintrc: 'less/.csslintrc'
84        },
85        src: [
86          'dist/css/bootstrap.css',
87          'dist/css/bootstrap-theme.css',
88          'docs/assets/css/docs.css',
89          'docs/examples/**/*.css'
90        ]
91      },
92
93      concat: {
94        options: {
95          banner: '<%= banner %>\n<%= jqueryCheck %>',
96          stripBanners: false
97        },
98        bootstrap: {
99          src: [
100           'js/transition.js',
101           'js/alert.js',
102           'js/button.js',
103           'js/carousel.js',
104           'js/collapse.js',
105           'js/dropdown.js',
106           'js/modal.js',
107           'js/tooltip.js',
108           'js/popover.js',
109           'js/scrollspy.js',
110           'js/tab.js',
111           'js/affix.js'
112         ],
113         dest: 'dist/js/<%= pkg.name %>.js'
114       }
115     },
116
```

图 12.79　Gruntfile.js 配置文件

（7）自定义完成后，只要在命令行执行 grunt dist 命令，就可以自动将文件打包到 dist 文件夹中，同时生成开发版本和应用于生产环境的最小化版本。

除了用于打包的 grunt dist 命令外，还有几个常用的命令可能会用到：

- grunt:：运行所有的测试用例并且编译 css 文件到 dist 目录
- grunt test：只运行测试用例
- grunt watch：监控 less 文件修改的命令，当该命令运行时，只要修改 less 文件都会自动地调用编译命令生成最新的 css/js 文件。

总的说来，两种定制化方式各有优势，直接在官网定制适合少量的、明确的修改，简单方便。如果需要深度定制，下载源代码修改配置的方式则更为可取。

12.8　其他 Bootstrap 资源

Bootstrap 在整体式的前端框架中算是比较早出现的，很多后进者在某些特性上可以说已经超越了 Bootstrap，比如更小巧的 Pure、对移动设备更友好的 Foundation，以及一些对浏览器兼容更好的国内框架。

Bootstrap 仍然受到开发者青睐的原因不仅仅是它自身的优秀，还在于开源社区基于

Bootstrap 开发出的一系列插件、皮肤、模板，这已经形成了良好的生态系统。如果不喜欢 Bootstrap 的配色和样式，那么有成百上千的皮肤可供选择；如果觉得官方的 JavaScript 插件不够丰富，那么 Google 搜一下，各种基于 Bootstrap 的插件可供选择；如果实在懒得下工夫，整体的 Bootstrap 风格网站源代码拿来直接部署就可以。

这里为读者介绍一些 Bootstrap 的相关资源：

- Bootstrap 中文网（图 12.80），地址 http://www.bootcss.com/，不仅有 Bootstrap 官方文档的中文翻译，还提供了很多相关资源的链接，以及常用前端资源的 CDN。

图 12.80　Bootstrap 中文网

- Bootswatch（图 12.81），地址：http://bootswatch.com/，提供各种风格的 Bootstrap 主题下载，还有一些付费的整体方案。

图 12.81　Bootswatch 首页

- Bootsnipp（图 12.82），地址：http://bootsnipp.com/，这里可以找到各种 Bootstrap 的设计元素、JavaScript 插件。

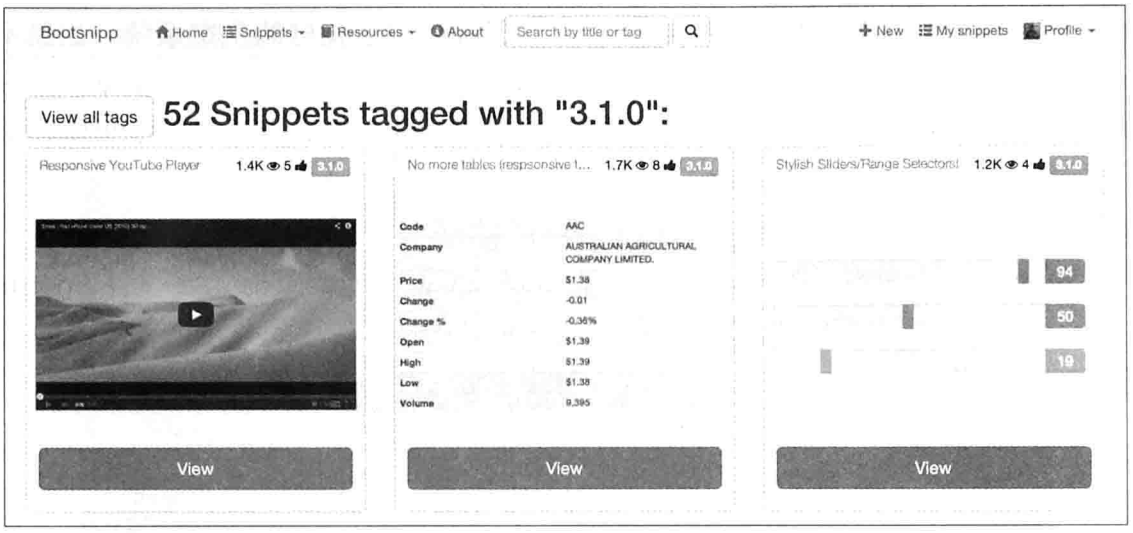


图 12.82　Bootsnipp 的资源列表

12.9　小结

本章主要介绍当前最流行的前端框架 Bootstrap，包括它的应用场景、主要功能模块、定制化、扩展资源等内容。

Bootstrap 主要包括以下几大模块：

- 基础 CSS 样式，包括标题、段落、表单、表格、按钮、栅格等基础样式，只要向元素添加指定的类就可以获得 Bootstrap 定义好的样式。
- 功能组件，包括图标、下拉菜单、导航、分页、按钮组等组合好的模块，需要应用 Bootstrap 指定的 HTML 结构和类。
- jQuery 插件，包括模态对话框、滚动监听、标签页切换、弹出框、警告框、轮播等常用的 jQuery 插件。大多数组件无须编写 JavaScript 代码，只需在 HTML 标签中应用指定的触发器即可。

Bootstrap 提供两种定制化的方式：一种是直接在官方网站的定制化页面修改配置后下载，另一种是下载源代码，修改配置文件并重新编译。直接在官方网站定制虽然简单易行，但只适合修改内容比较少而且比较明确的情况，因为修改的状态是无法保存的。下载源代码的方式更适合较为深度的定制，但是要使用 Bootstrap 的自动化打包工具需要先安装 Node.js 环境，对初学者来说有一定的难度。

第 13 章　Foundation 框架实战

本章介绍目前流行的移动优先的响应式前端框架 Foundation，包括它的基本信息、使用场景和使用方法。与 Bootstrap 相比，Foundation 更为轻量，对移动方面的优化更好，缺点是完全不兼容 IE 9 以下的 IE 浏览器。

本章主要包括以下内容：

- Foundation 框架的简介和安装。
- Foundation 包含的主要组件以及用法。
- Foundation 的定制化。

13.1 认识 Foundation

Foundation 是著名的产品设计咨询公司 ZURB 开发的一款移动优先的前端框架，源代码采用 SASS 编写，目前最新版本是 Foundation 5。官方网站地址：http://foundation.zurb.com/，如图 13.1 所示。

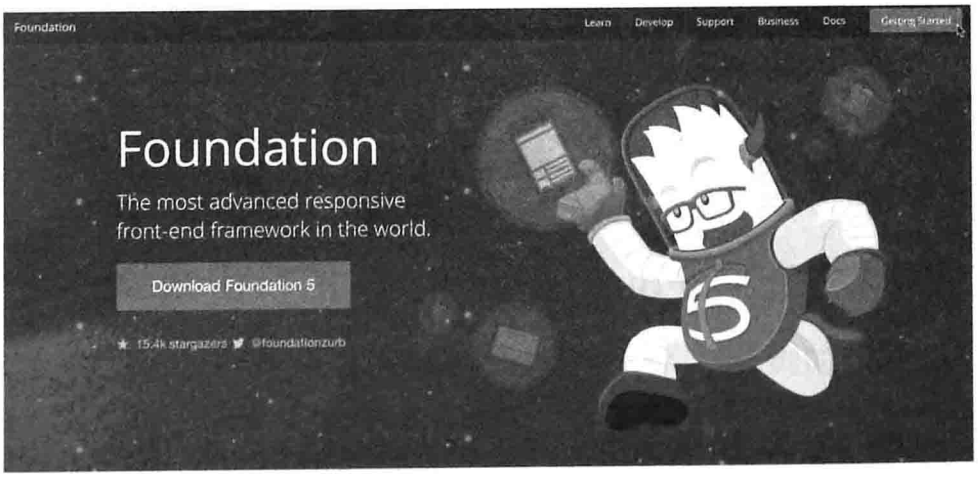


图 13.1　Foundation 首页

相比同样开始拥抱移动的 Bootstrap 3，笔者将 Bootstrap 3 和 Foundation 进行了对比：

（1）Foundation 4 同时支持 Zepto 和 jQuery 作为 js 类库。与 jQuery 相比，Zepto 专注于移动端设计，由于移动端 webkit 核心浏览器基本一统天下，因此移除了大量的兼容代码，gzip 压缩后只有 9.2kb，在传输效率和执行效率上都优于 jQuery。不过由于 jQuery 2 的出现和移动端性能、网络条件的提升，Foundation 5 取消了对 Zepto 的支持。

（2）js 方面移除了兼容代码，在 css 方面也激进地放弃了 IE 8，而 Bootstrap 不仅可以较好地兼容 IE 8，在一些诸如 bsie 的扩展下甚至可以兼容到 IE 6。

（3）Bootstrap 由于基于桌面设计的历史原因和兼容性方面的考虑，基本上使用固定的像素值作为长度单位，宽度方面虽然可以使用响应式的栅格或流式布局来处理，但是整体的尺寸就很难做到兼顾了。而 Foundation 除了在圆角等细节上少量使用 px，多数使用 em、rem 等相对值。在媒介查询方面也采用了 em 作为单位，更为灵活，但这也造成了 IE 8 及以下的浏览器完全无法兼容。

说明：rem 是 CSS 3 新加入的长度单位。

在实际应用领域，Foundation 非常适合于构建移动优先或是纯移动端的响应式 Web 应用，而对于一些考虑到 IE 浏览器兼容的应用就不合适了。Foundation 5 的浏览器兼容性如图 13.2 所示。

Browser/OS	The Grid	Layout/UI	JS
Chrome	✓	✓	✓
Firefox	✓	✓	✓
Safari	✓	✓	✓
IE10	✓	✓	✓
IE9	✓	✓	✓
IE8	✗	✗	✗
IE7	✗	✗	✗
iOS (iPhone)	✓	✓	✓
iOS (iPad)	✓	✓	✓
Android 2. 4 (Phone)	✓	✓	✓
Android 2, 4 (Tablet)	✓	✓	✓
Windows Phone 7+	✓	✓	✓
Surface	✓	✓	✓

图 13.2　Foundation 5 的浏览器兼容性

13.2　Foundation 的安装和使用

Foundation 提供 3 种安装和使用方式：

- 下载编译后的 CSS 和 JavaScript 文件，在 HTML 文档中引入。比较适合采用原生 CSS 进行编码的开发者。
- 对于本身就使用 SASS 进行编码的开发者，Foundation 提供了基于 Compass 的开发环境（SASS 和 Compass 参见第 14 章）。
- 对于 Ruby on Rails 开发者，Foundation 专门为 Rails 提供了插件，一行命令就可以为 Rails 应用生成 Foundation 开发环境。

官方网站上对这 3 种方式的介绍，如图 13.3 所示。

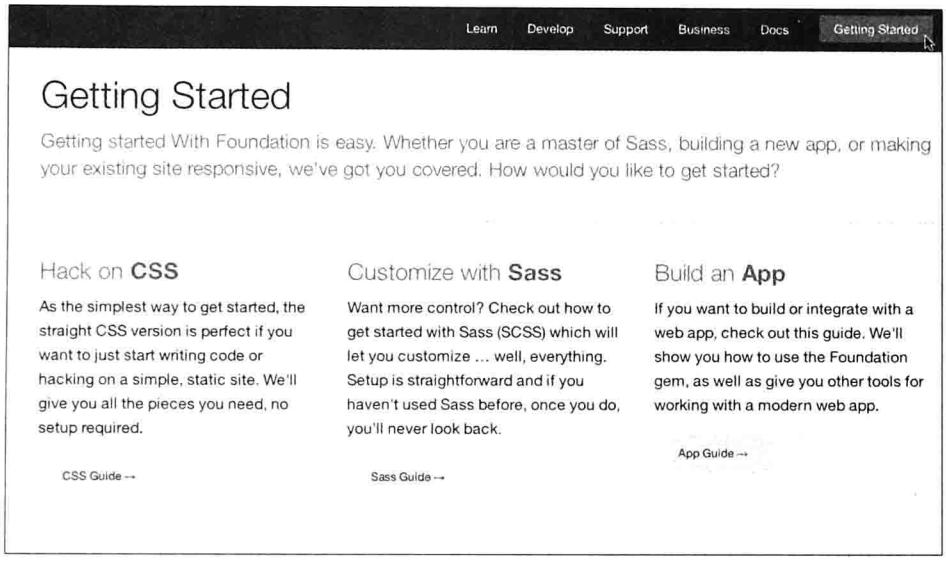


图 13.3　3 种方式的介绍

13.2.1　传统方式的下载安装

在官方网站下载安装包，如图 13.4 所示，选择 Default CSS 进行下载即可。

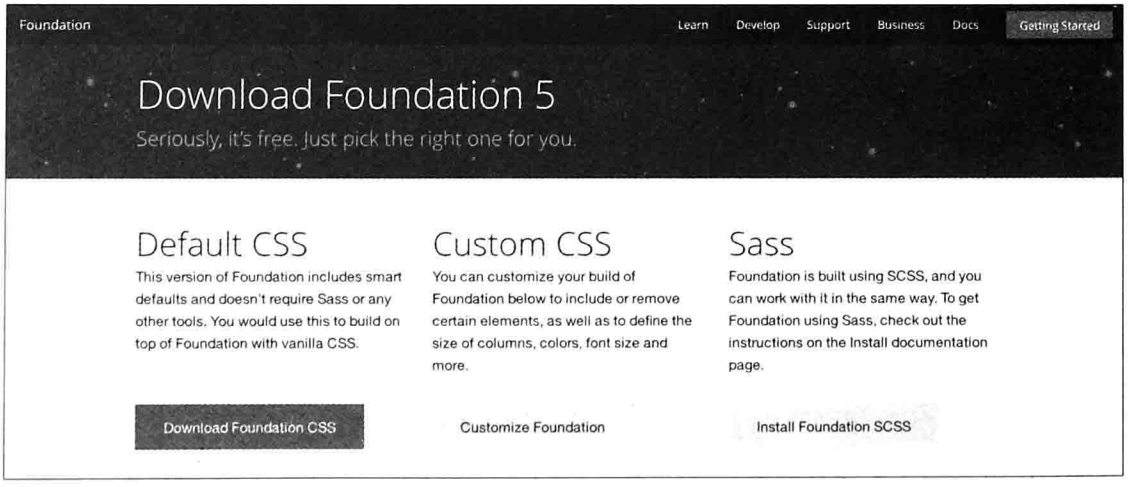


图 13.4　Foundation 下载页面

下载完成后解压缩，解压后的文件夹中有一个示例 HTML 文件可以进行参考。

首先需要在<head>标签内引入 Foundation 的 CSS 文件、用于浏览器嗅探和向下兼容的 modernizr.js，以及响应式框架必要的<meta>标签：

```
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<link rel="stylesheet" href="css/foundation.css" />
<script src="js/modernizr.js"></script>
```

如果需要用到内置的一些 JavaScript 动态效果，就需要在<body>标签底部引入 jQuery 和 Foundation 的 JavaScript 插件，并进行调用：

```
<script src="js/jquery.js"></script>
<script src="js/foundation.min.js"></script>
<script>
```

```
  $(document).foundation();
</script>
```

完整的代码如下所示：

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="css/foundation.css" />
    <script src="js/modernizr.js"></script>
  </head>
  <body>
    .....<!--在这里编写 HTML 代码-->
    <script src="js/jquery.js"></script>
    <script src="js/foundation.min.js"></script>
    <script>
      $(document).foundation();
    </script>
  </body>
</html>
```

采用传统安装方式的好处在于不依赖任何其他环境，如果需要定制化，相对就麻烦些，要么在官网的定制化页面进行设置并下载定制化版本，要么自己修改 CSS 文件。

13.2.2 使用 Compass 进行 Foundation 开发

第 14 章会介绍 SASS 和 Compass 工具，Foundation 也正是用它们开发的，对于使用 SASS&Compass 或需要定制化的开发者来说，Foundation 提供的基于 Grunt 和 Compass 的集成环境非常方便。

注意：Grunt 是一种前端自动化工具，可以帮助开发者进行编译、压缩、测试等工作。

要使用 Foundation 提供的 Compass 开发环境，首先需要保证计算机上安装有以下 3 个软件：

- git
- Node.js
- ruby 1.9+

确保以上 3 个软件安装后，下一步需要安装 Grunt、Compass 和 Foundation，只要在终端中输入下列命令即可完成安装：

```
$ npm install -g bower grunt-cli
$ gem install compass
$ gem install foundation
```

安装完成后，使用命令：

```
foundation new My_app
```

就会自动生成一个基于 Grunt 和 Compass 的 Foundation 开发环境，可以直接使用 compass compile、compass watch 等 compass 的命令。

使用 compass 进行开发的好处除了第 10 章介绍过的之外，对于 Foundation 本身的意义在于非常方便定制化，可以在生成的_setting.scss 文件中对全局变量进行定制，还可以对引入的组件进行控制，也可以很方便地引入和移除组件。

13.2.3　在 Rails 应用中引入 Foundation

基于 Ruby 语言的 Rails 是 Web 开发领域一个较为流行的开发框架（Ruby 在中国属于比较小众的开发语言），由于 SASS 和 Ruby 的血缘关系，Rails 默认集成了 SASS。

安装过程也很简单，在 Rails 框架的 Gemfile 文件中引入：

```
gem 'foundation-rails'
```

然后在命令行工具中执行 bundle install 命令进行安装，安装完成后执行：

```
rails g foundation:install
```

这样就完成了 Foundation 在 Ruby on Rails 应用中的集成。

13.3　使用 Foundation 栅格系统

Foundation 的栅格系统总体上类似于 Bootstrap 3，通过不同的栅格类来指定页面元素在不同尺寸窗口下的表现以实现响应式，同时 Foundation 还加入了块网格的概念，用于解决一块区域内元素平铺的问题。

13.3.1　基本栅格系统

Foundation 的栅格系统和 Bootstrap 3 的类似，也是通过不同的栅格类来区分不同尺寸窗口下的表现形式。下面是 Foundation 栅格系统的一个典型示例：

```
<div class="row">    <!--确保不会折行-->
  <div class="small-4 medium-4 large-4 columns">...</div>
  <div class="small-4 medium-4 large-4 columns">...</div>
  <div class="small-6 medium-4 large-4 columns">...</div>
</div>
```

在使用方面，Foundation 和 Bootstrap 一样，首先需要在外层包裹一层 ，来确保内部的栅格不会折行。

对于每一个栅格元素，都需要添加一个 columns 类，columns 类表明了这个 HTML 元素是一个栅格列。small、medium、large 表示在不同尺寸窗口下的表现，后缀的数字则表示占据的列数，例如 ... 就表示该元素在小尺寸窗口下，占据 6 列宽度，在中尺寸和大尺寸窗口下占据 4 列宽度。

如果单独使用 large/medium/small 类，Foundation 的表现和 Bootstrap 3 也是一致的。small

类采用流式布局，在任何尺寸的窗口下都根据比例占据宽度。而 large 和 medium 类则会在窗口尺寸小于阈值的情况下堆叠起来，占据一整行的宽度。

需要注意的是，对于大、中、小尺寸的定义，Foundation 和 Bootstrap 有较大的区别。Bootstrap 3 采用固定的像素值来进行媒介查询范围定义，而 Foundation 则采用相对值 em，默认值如下：

```
@media only screen and (min-width: 40.063em) {.....}
@media only screen and (min-width: 64.063em) {.....}
@media only screen and (min-width: 90.063em) {.....}
```

这样只要基础字体大小变化，媒介查询的范围也会随之变化，从而在比例上保持一致性。

13.3.2 块网格（Block Grid）

块网格用于解决在一块区域内平铺元素的问题，块网格的效果如图 13.5 所示。

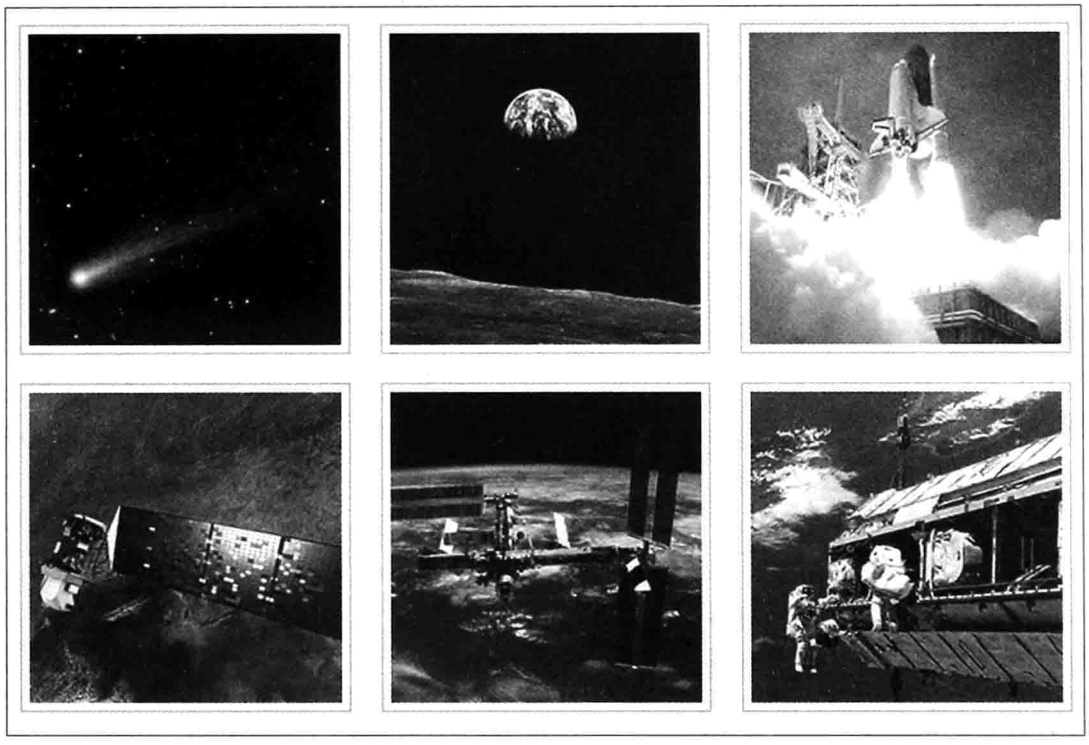


图 13.5 块网格示例

块网格的基础用法很简单，如下：

```
<ul class="small-block-grid-3">
  <li><!-- 内容 --></li>
  <li><!-- 内容 --></li>
  <li><!-- 内容 --></li>
  <li><!-- 内容 --></li>
</ul>
```

使用块网格需要一个无序列表结构，并为<ul>元素添加块网格类 small-block-grid-n，n 表示每一行的元素数量。如果只使用 small-block-grid-n，在任何大小的窗口下都是以每行 n 个<li>元素进行平铺，不过<li>元素的大小会根据外层容器的大小进行自适应的变化。

如果在不同大小的窗口下每行的排列数量不同，可以使用 small-block-grid-n、medium-block-grid-n、large-block-grid-n 来分别表示在小窗口、中等窗口、大窗口下的表现，如下例所示：

```
<ul class="small-block-grid-2 medium-block-grid-3 large-block-grid-4">
  <li><!-- 内容 --></li>
  <li><!-- 内容 --></li>
  <li><!-- 内容 --></li>
  <li><!-- 内容 --></li>
</ul>
```

块网格的基本实现原理是采用 CSS 中的 box-sizing:border-box;，这样将边框、内边距都算在宽度里，然后用百分比表示<li>元素的宽度，就可以精确地等分外层容器的宽度，实现平铺元素了。

13.4 Foundation 基本样式

同 Bootstrap 一样，Foundation 也为基础的 HTML 元素定制了一套样式，以配合整体的 UI 风格，包括标题、段落、列表、按钮等，同时实现了图片、视频等媒体素材的响应式。

13.4.1 标题和段落

1. 标题样式

Foundation 重置了 h1~h6 元素的样式，以配合整体的 UI 设计，如图 13.6 所示。

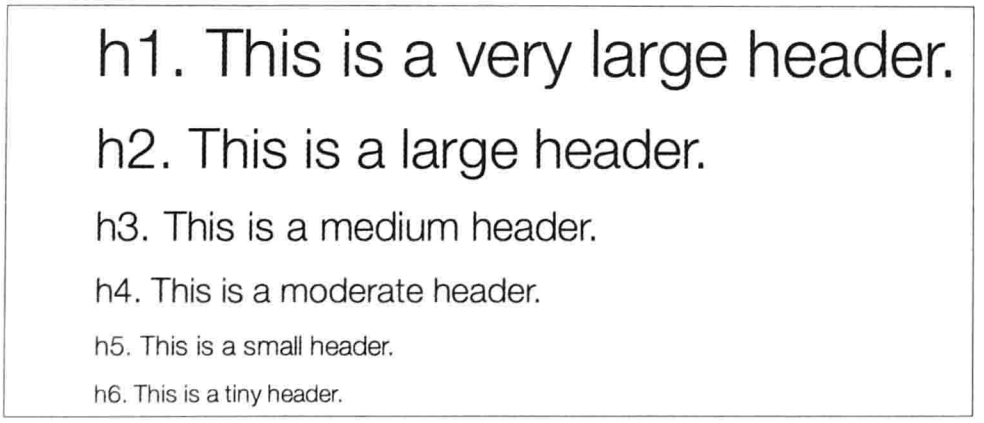


图 13.6 Foundation 中的默认标题样式

对于子标题，Foundation 通过为标题添加 subheader 类来实现，主要的变化是修改了标题的颜色。

```
<h1 class="subheader">h1. This is a very large header.</h1>
```

图 13.7 中的 h1 元素就是添加了 subheader 类后的样式。

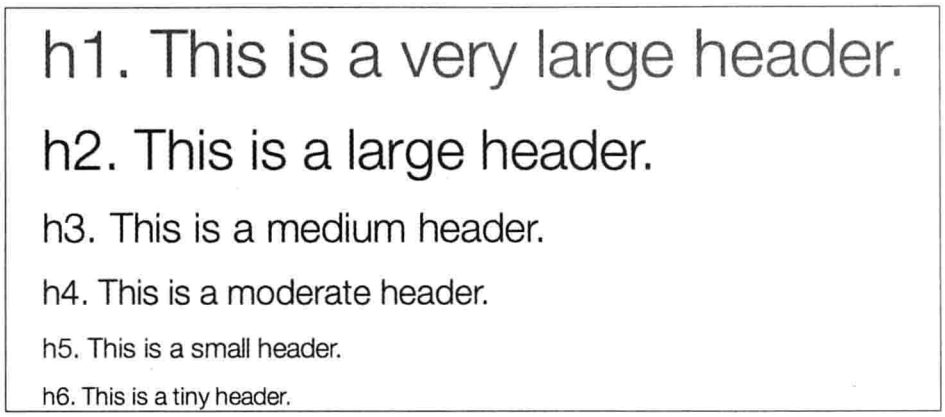


图 13.7　为 h1 添加了 subheader 类后的样式

Foundation 支持在 h1~h6 中使用 small 标签来设置二级标题，例如：

```
<h1>h1. <small>Small segment header.</small></h1>
```

效果如图 13.8 所示。

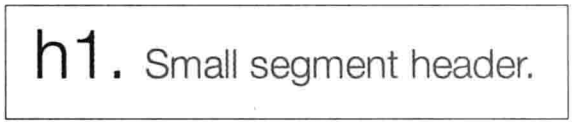


图 13.8　二级标题的样式

2．段落样式

Foundation 为段落元素内的文字设计了更为合适的行高，如图 13.9 所示，左边是 div 元素中的文字，右侧是 p 元素内的文字，哪个更为美观一目了然。

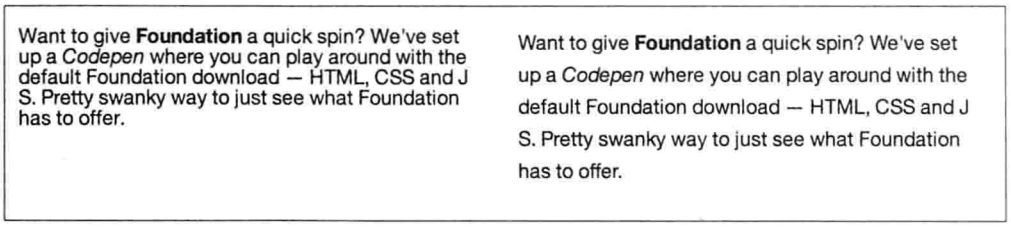


图 13.9　div 内的文字和段落内的文字对比效果

在段落内使用标签可以将文字加粗，如图 13.9 中的 Foundation 单词。使用标签则可以获得倾斜的文字，如图 13.9 所示的 Codepen 单词。

13.4.2　列表

Foundation 可以通过为<ul>元素添加不同的类来实现无序列表的不同样式。例如：

```
<ul class="small-block-grid-4">
    <li>
      <ul class="no-bullet">   /*无装饰*/
        <li>列表项</li>
        <li>列表项</li>
        <li>列表项</li>
      </ul>
```

```
    </li>
    <li>
      <ul class="disc">      /*实心圆形的无序列表装饰*/
        <li>列表项</li>
        <li>列表项</li>
        <li>列表项</li>
      </ul>
    </li>
    <li>
      <ul class="circle">    /*空心圆形的无序列表装饰*/
        <li>列表项</li>
        <li>列表项</li>
        <li>列表项</li>
      </ul>
    </li>
    <li>
      <ul class="square">    /*方形的无序列表装饰*/
        <li>列表项</li>
        <li>列表项</li>
        <li>列表项</li>
      </ul>
    </li>
</ul>
```

代码浏览效果如图 13.10 所示。

列表项	• 列表项	○ 列表项	▪ 列表项
列表项	• 列表项	○ 列表项	▪ 列表项
列表项	• 列表项	○ 列表项	▪ 列表项

图 13.10 无序列表的 4 种样式

让列表在同一行排列也是 Web 开发中常见的问题，Foundation 对此也有很不错的解决方案，例如：

```
<ul class="inline-list">
  <li><a href="#">链接 1</a></li>
  <li><a href="#">链接 2</a></li>
  <li><a href="#">链接 3</a></li>
  <li><a href="#">链接 4</a></li>
  <li><a href="#">链接 5</a></li>
</ul>
```

只需为无序列表的<ul>元素添加.inline-list 类即可，效果如图 13.11 所示。

链接 1 链接 2 链接 3 链接 4 链接 5

图 13.11 同一行排列的列表

一般来说在同一行内的列表元素都没有装饰符号，Foundation 很贴心地把它去掉了。

13.4.3 按钮

Foundation 的按钮在设计上相比 Bootstrap 更偏重移动，更大的按钮保证了在移动设备上的可用性，其用法和 Bootstrap 类似，通过为<button>或<a>元素添加 class="button"从而获得基础的按钮样式，如下：

```
<button class="button">默认的按钮样式</button>
```

上述代码效果如图 13.12 所示，左侧是按钮正常状态样式，右侧是鼠标移上后的样式。

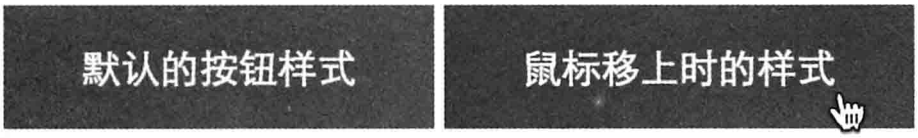


图 13.12 Foundation 的基础按钮样式

Foundation 提供控制按钮大小的类：tiny、small、large 类，代码如下：

```
    <button class="button tiny">确 认</button>      <!-- 最小的按钮-->
    <button class="button small">确 认</button>     <!-- 小按钮-->
    <button class="button">确 认</button>
<button class="button large">确 认</button>     <!--大按钮-->
```

代码的效果如图 13.13 所示。

图 13.13 使用内置的类控制按钮的大小

和 Bootstrap 一样，Foundation 也支持添加类来改变按钮的配色，从而表达一些常用的意义。

```
    <button class="button secondary">确 认</button>   <!--表示次要-->
    <button class="button success">确 认</button>     <!--表示成功-->
    <button class="button">确 认</button>
<button class="button alert">确 认</button>          <!--表示警告-->
```

效果如图 13.14 所示。

图 13.14 按钮的配色

与 Bootstrap 相比，Foundation 对按钮的风格设置更为灵活，除了默认风格外还可以通过添加.radius 类和.round 类为按钮添加不同弧度的圆角。

```
<button class="button radius">圆角按钮</button>   <!--固定 3 像素的圆角-->
```

```
<button class="button round">圆弧按钮</button>    <!--半圆形的圆角-->
```

效果如图 13.15 所示。

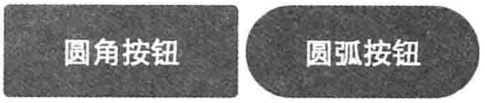


图 13.15　圆角风格的按钮

Foundation 还可以通过.expand 类来实现占据整行的大按钮，类似于 Bootstrap 中的.btn-block 类，如图 13.16 所示。

```
<button class="button expand">整行按钮</button>
```

图 13.16　整行按钮

Foundation 同样提供了按钮组功能，并且可以进行一定程度上的定制。基础样式非常简单，只需要将按钮或按钮样式的链接放入一个无序列表中，并为<ul>元素添加.button-group 类即可。

```
<ul class="button-group">
  <li><a href="#" class="button">按钮 1</a></li>
  <li><a href="#" class="button">按钮 2</a></li>
  <li><a href="#" class="button">按钮 3</a></li>
</ul>
```

效果如图 13.17 所示。

图 13.17　按钮组基础样式

和单个的按钮类似，可以为按钮组的<ul>元素添加.round、.radius 类，为按钮组添加不同弧度的圆角。例如：

```
<ul class="button-group round">
  <li><a href="#" class="button">按钮 1</a></li>
  <li><a href="#" class="button">按钮 2</a></li>
  <li><a href="#" class="button">按钮 3</a></li>
</ul><br>
<ul class="button-group radius">
  <li><a href="#" class="button">按钮 1</a></li>
  <li><a href="#" class="button">按钮 2</a></li>
  <li><a href="#" class="button">按钮 3</a></li>
</ul>
```

效果如图 13.18 所示，.round 类表示将按钮组边缘设置为圆弧形，.radius 类则添加了 3 个像素的圆角。

图 13.18　添加了.round 类和.radius 类的按钮组样式

13.4.4　面板

面板是一个简单、有效的基础组件，它可以轻松地勾勒出页面的某些部分，对这一部分进行强调或淡化，并在其中添加内容。尤其在网页原型构建时，面板可以帮助设计者轻松地划分页面。面板的本质是一个设计好的 元素，它为开发者准备好了合适的边框、底色、内外边距等属性，从而免去了调整这些属性的时间。对于不擅长美工设计的程序员来说这是一种福音。

面板样式使用.panel 类进行设置，例如：

```
<div class="panel">
  <h4>这是一个常规的面板样式</h4>
  <p>轻松的构造一个视觉上更为优先的区域，让内容看起来更为醒目</p>
</div>
```

效果如图 13.19 所示。

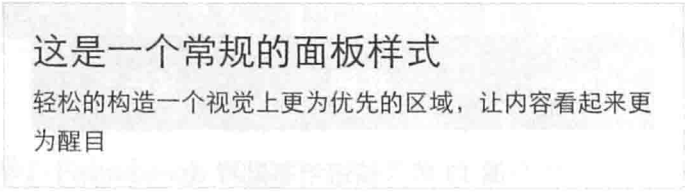


图 13.19　面板的基本样式

面板也支持圆角，可以通过添加.radius 类来实现，对于需要强调的部分，则可以添加.callout 类来实现，例如：

```
<div class="panel callout radius">
  <h4>这是一个 callout 面板</h4>
  <p>视觉上更为引人注目，适用于一些需要着重显示的区域</p>
</div>
```

效果如图 13.20 所示。

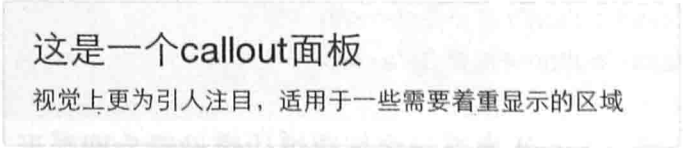


图 13.20　添加了 callout 类和圆角的面板

13.4.5　缩略图

一般来说，缩略图往往附带着链接，而没有任何修饰的缩略图往往并不美观，Foundation 提供了一个看起来还不错的方案。它的 HTML 结构很简单，就是一个添加了.th 类的<a>链接内包裹一个<img>标签，例如：

```
<a class="th" href="../img/demos/demo2.png">
  <img src="../img/demos/demo2-th.png">
</a>
```

上述代码的效果如图 13.21 左侧所示。缩略图也支持添加圆角，只需要为<a>标签添加.radius 类即可，效果如图 13.21 右侧所示。

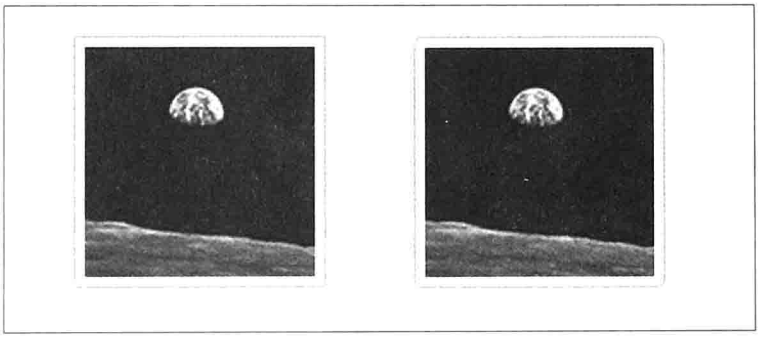


图 13.21　缩略图

13.4.6　视频

一般来说，在页面中嵌入的视频都是固定尺寸的，Foundation 则可以很容易地帮助开发者让视频窗口实现响应式。这需要为<iframe>元素添加视频链接，并在外层包裹一层 ，然后为 元素添加.flex-video 类，例如：

```
<div class=" small-10 large-6 medium-6 column">
  <div class="flex-video">
    <iframe width="420" height="315" src="http://v.test.com/test.mp4" frameborder="0"
    ></iframe>
  </div>
</div>
```

效果如图 13.22 所示。

图 13.22　响应式视频窗口

这样，视频窗口的大小就可以随着外层设置了响应式的 的宽度来成比例地缩放了。

13.4.7 可见性

所有需要兼容不同设备的设计都需要考虑在小尺寸设备下突出重要的内容、收起或者干脆不显示一些次要内容这个原则，因此便捷地设置元素的可见性就成了响应式框架必备的功能。

Foundation 在可见性方面的命名十分容易记忆，例如：

```
<p class="show-for-small-only">只在小窗口下显示</p>
<p class="show-for-medium-up">在中等及以上大小的窗口下显示</p>
<p class="show-for-medium-only">只在中等窗口下显示</p>
< p class="show-for-large-up">在大型及以上大小的窗口下显示</p>
< p class="show-for-large-only">只在大窗口下显示</p>
< p class="show-for-xlarge-up">在超大型及以上大小的窗口下显示</p>
< p class="show-for-xlarge-only">只在超大窗口下显示</p>
```

从代码中可以发现一个规律，.show-for-尺寸-only 表示只在这个尺寸范围内显示，.show-for-尺寸-up 表示在该尺寸及该尺寸以上时都显示。同样地，将 show 换成 hide 就表示隐藏，例如：

```
<p class="hide-for-small-only">只在小窗口下隐藏</p>
<p class="hide-for-medium-up">在中等及以上大小的窗口下隐藏</p>
<p class="hide-for-medium-only">只在中等窗口下隐藏</p>
< p class="hide-for-large-up">在大型及以上大小的窗口下隐藏</p>
< p class="hide-for-large-only">只在大窗口下隐藏</p>
< p class="hide-for-xlarge-up">在超大型及以上大小的窗口下隐藏</p>
< p class="hide-for-xlarge-only">只在超大窗口下隐藏</p>
```

除了根据窗口大小进行媒介查询来显示\隐藏元素，Foundation 还可以通过引入的 Modernizr 插件来检测终端是否为触屏设备，这样就可以根据终端的类型来设置可见性，例如：

```
<p class="panel">
  <strong class="show-for-touch">在触屏设备上显示</strong>
  <strong class="hide-for-touch">在触屏设备上隐藏</strong>
</p>
```

而对于触屏设备来说，横屏和竖屏时也有可能需要实现不同的可见性设置，Foundation 也为我们准备好了，例如：

```
<p class="panel">
  <strong class="show-for-landscape">横屏状态下显示</strong>
  <strong class="show-for-portrait">竖屏状态下显示</strong>
</p>
```

13.5 导航系统

Foundation 提供了多种简约风格的导航系统，包括头部导航、面包屑导航、侧边栏导

航、子导航等，可以满足绝大部分应用场景的需求，而且其导航结构较为简单，可以方便地混入自定义代码。

13.5.1　面包屑导航

面包屑导航通常在页面顶部水平出现，一般会位于标题或页头的下方，它允许用户返回之前任何一个页面的链接。Foundation 实现这个功能非常轻松，只需要为无序列表添加.breadcrumbs 类即可。

```
<ul class="breadcrumbs">
  <li><a href="#">首页</a></li>
  <li><a href="#">特性</a></li>
  <li class="unavailable"><a href="#">基因拼接</a></li>
  <li class="current"><a href="#">克隆</a></li>
</ul>
```

为<li>元素添加.unavailable 类表示当前所在的页面链接，用.current 类表示刚才所在的链接，示例代码效果如图 13.23 所示。

首页 / 特性 / 基因拼接 / 克隆

图 13.23　面包屑导航

除了使用无序列表外，为<nav>元素添加.breadcrumbs 类也可以实现同样的效果，例如：

```
<nav class="breadcrumbs">
  <a href="#">首页</a>
  <a href="#">特性</a>
  <a class="unavailable" href="#">基因拼接</a>
  <a class="current" href="#">克隆</a>
</nav>
```

13.5.2　侧边栏导航

Foundation 提供的侧边栏导航非常简单，仅仅为一个无序列表的<ul>元素添加一个.side-nav 类就完成了，如图 13.24 所示。

链接一
链接二
链接三
链接四

图 13.24　侧边栏导航

```
<ul class="side-nav">
  <li class="active"><a href="#">链接一</a></li>    <!-- 为<li>元素添加.active 类表示已选中-->
  <li><a href="#">链接二</a></li>
  <li><a href="#">链接三</a></li>
  <li><a href="#">链接四</a></li>
</ul>
```

Foundation 提供的侧边栏导航样式非常简单，虽然直接使用显得有些简陋，但也意味着自由发挥的空间更大，开发者可以很方便地为其添加样式。

13.5.3 头部导航

在 Foundation 中构建一个基本的头部导航非常容易，首先需要构建最外层的结构：

```
<nav class="top-bar">
  ……
</nav>
```

这个时候，呈现在网页上的就只有顶部的一片黑色区域。

一般来说，最左侧会放置网站的名称，这部分内容放在一个无序列表中，需要给<ul>添加.title-area 类，给<li>元素添加.name 类，并且无序列表内部必须是一个<h1>元素，例如：

```
<nav class="top-bar">
  <ul class="title-area">
<li class="name">
  <h1><a href="/home">我的博客</a></h1>
</li>
  </ul>
</nav>
```

效果如图 13.25 所示。

图 13.25　头部导航 1

设置好网站的名称后，重头戏就是设置导航列表，以博客为例，可能需要设置博客列表、博客精选等导航链接，这部分一般居左放置，而登录、搜索框、RSS 订阅等内容则需要居右放置。

整个导航列表部分用<section class="top-bar-section"> …… </section>进行包裹，内部是一组或多组无序列表，如果居左，则为<ul>元素添加.left 类，如果居右则添加.right 类。实现代码如下：

```
<nav class="top-bar">
  <ul class="title-area">
<li class="name">
  <h1><a href="/home">我的博客</a></h1>
</li>
  </ul>
  <section class="top-bar-section">
    <ul class="left">
      <li><a href="#">列表</a></li>
      <li><a href="#">精选</a></li>
    </ul>
<ul class="right">
  <li><input type="text" class="nav-search" placeholder="搜索"></li>
      <li><a href="#">RSS</a></li>
      <li><a href="#">登录</a></li>
    </ul>
```

```
  </section>
</nav>
```

最终效果如图 13.26 所示，这样一个完整的博客头部导航就完成了。

图 13.26　头部导航 2

但是当我们缩小窗口时就发现问题了，如图 13.27 所示，为什么除了标题以外其他的东西都不见了，是出现 Bug 了吗？

图 13.27　头部导航 3

当然不是，小窗口下无法显示这么多的内容，熟悉原生手机应用的读者应该知道，手机应用的目录都是收缩起来的，需要的时候点击才能展开，Foundation 也采用了同样的处理方式。

现在需要添加一些代码：

```
<nav class="top-bar" data-topbar>  <!-- 在 nav 标签上添加一个名为 data-topbar 的触发器-->
  <ul class="title-area">
<li class="name">
  <h1><a href="/home">我的博客</a></h1>
</li>
    <li class="toggle-topbar menu-icon"><a href="#"><span>Menu</span></a></li>
<!-- 添加这一行代码，以显示菜单按钮-->
  </ul>
  <section class="top-bar-section">
    <ul class="left">
      <li><a href="#">列表</a></li>
      <li><a href="#">精选</a></li>
    </ul>
<ul class="right">
  <li><input type="text" class="nav-search" placeholder="搜索"></li>
      <li><a href="#">RSS</a></li>
      <li><a href="#">登录</a></li>
    </ul>
  </section>
</nav>
```

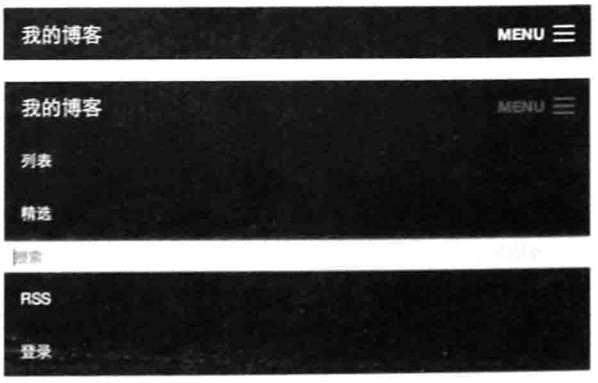


图 13.28　头部导航 4

上述代码为<nav>标签添加了一个 JavaScript 触发器，在标题列表中添加了一行用于显示菜单按钮的代码。最终的效果如图 13.28 所示，上面是导航收起时的效果，下面是导航展开时的效果。

这样，一个功能完整的头部导航就完成了。

13.5.4 子导航

子导航往往位于头部导航条下方，显示本版块的一些信息。Foundation 中子导航选择了定义列表（<dl>）的标签语义，结构同样很简单，为<dl>元素添加.sub-nav 类，对于已被选中的选项，添加.active 类。示例代码如下：

```
<dl class="sub-nav">
  <dt>过滤器</dt>
  <dd class="active"><a href="#">全部</a></dd>
  <dd><a href="#">进行中</a></dd>
  <dd><a href="#">待定</a></dd>
  <dd><a href="#">暂停</a></dd>
</dl>
```

示例代码效果如图 13.29 所示。

图 13.29 子导航样式

13.6 Foundation 中的 JavaScript 特效

Foundation 也内置了很多常用的 JavaScript 特效，比如幻灯片、长页面滚动、弹出层提示等，这些效果在一些组件里也有用到，比如上一节中头部导航的展开/收起功能。本节主要介绍它们的具体使用方法。

13.6.1 幻灯片

Foundation 的幻灯片模块简单、功能强大、反应敏捷，不仅可以在电脑上单击滑动翻页，还可以让用户使用触控设备时进行滑动。

幻灯片应用需要为无序列表添加名为 data-orbit 的触发器，如下例：

```
<ul class="example-orbit" data-orbit>
  <li>
    <img src="../assets/img/examples/satelite-orbit.jpg" alt="slide 1" />
  </li>
  <li>
    <img src="../assets/img/examples/andromeda-orbit.jpg" alt="slide 2" />
  </li>
  <li>
    <img src="../assets/img/examples/launch-orbit.jpg" alt="slide 3" />
  </li>
</ul>
```

这样就可以得到一个简单的图片幻灯播放效果了，如图 13.30 所示。

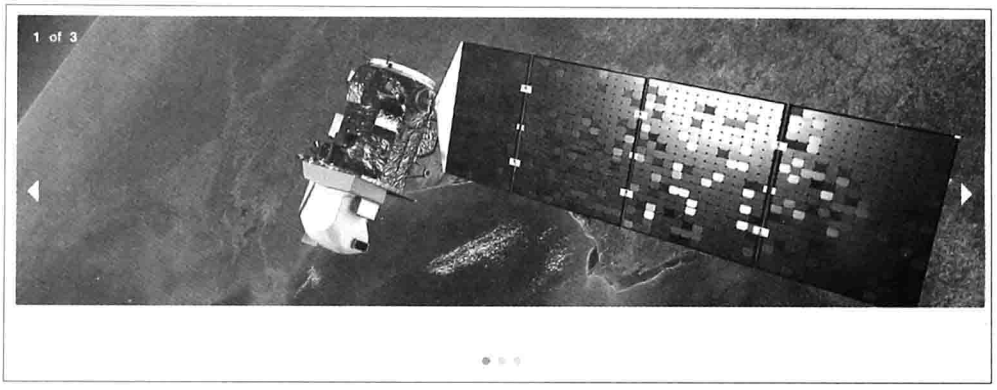


图 13.30　图片幻灯播放效果

如果需要为图片添加说明，就需要在列表项内添加 ..... ,例如：

```
<ul class="example-orbit" data-orbit>
  <li>
    <img src="../assets/img/examples/satelite-orbit.jpg" alt="slide 1" />
    <div class="orbit-caption">
      Caption One.
    </div>
  </li>
  <li>
    <img src="../assets/img/examples/andromeda-orbit.jpg" alt="slide 2" />
    <div class="orbit-caption">
      Caption Two.
    </div>
  </li>
  <li>
    <img src="../assets/img/examples/launch-orbit.jpg" alt="slide 3" />
    <div class="orbit-caption">
      Caption Three.
    </div>
  </li>
</ul>
```

加上文字说明后的效果如图 13.31 所示。

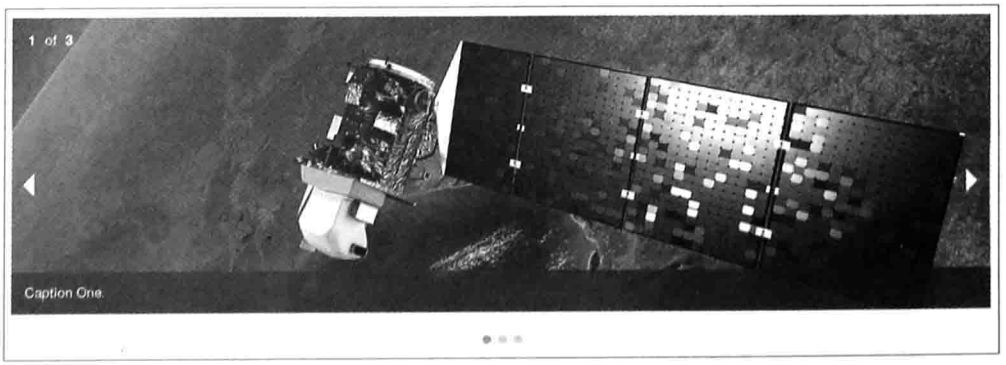


图 13.31　加上文字说明的图片幻灯播放效果

注意：幻灯片不仅可以播放图片，也可以播放文字内容，只需要将列表中的图片替换掉就可以了。

13.6.2 Clearing lightboxes

幻灯效果实现了图片的轮播，但图片的尺寸是固定的，用户如果要查看放大的图像就无法满足需求了，Foundation 内置的 Clearing lightboxes 插件就可以满足这个需求：

```
<ul class="clearing-thumbs" data-clearing>
  <li><a href="path/to/your/img"><img src="path/to/your/img-th"></a></li>
  <li><a href="path/to/your/img"><img src="path/to/your/img-th"></a></li>
  <li><a href="path/to/your/img"><img src="path/to/your/img-th"></a></li>
</ul>
```

它的基本结构是一个无序列表，需要为<ul>元素添加一个.clearing-thumbs 类以及一个名为 data-clearing 的触发器。

在列表项内部，<a>标签中的链接地址为高清大图的地址，<img>标签中的地址为缩略图的地址。显示在页面上的是一组缩略图，点击缩略图后，就会出现一个弹出层覆盖当前页面，来展示大图，如图 13.32 所示。

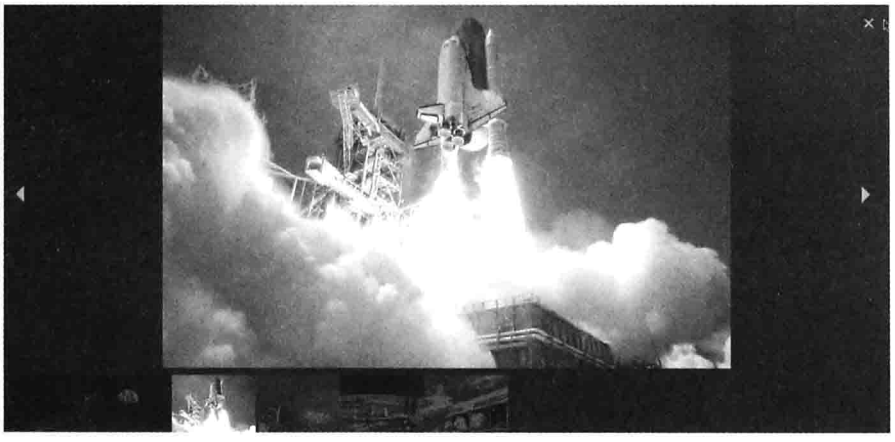


图 13.32　为缩略图显示高清大图

如果需要为图片添加说明，则需要在<img>标签中添加 data-caption= "图片说明"，例如：

```
<ul class="clearing-thumbs" data-clearing>
<li>
  <a href="path/to/your/img">
    <img data-caption="浩瀚的星空" src="path/to/your/img-th ">
  </a>
</li>
<li>
  <a href="path/to/your/img">
    <img data-caption="航天飞机发射" src="path/to/your/img-th ">
  </a>
</li>
</ul>
```

其效果如图 13.33 所示。

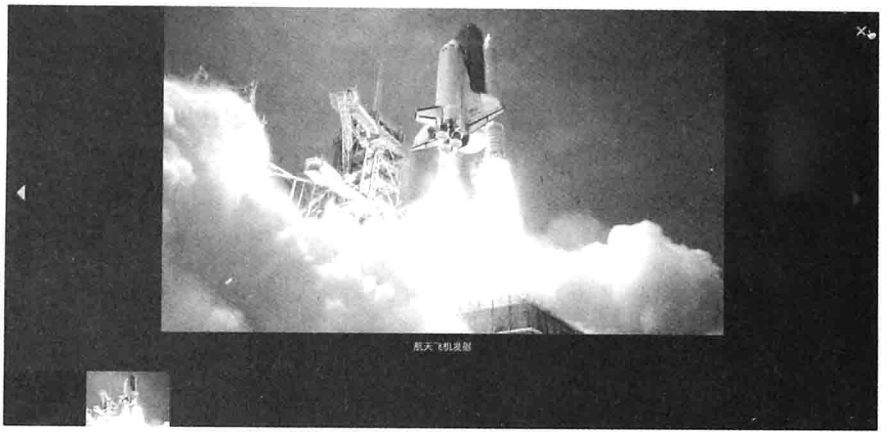


图 13.33　添加了说明的高清大图

注意：点击左右箭头可以翻页，点击右上角的叉符号可以关闭弹出层，返回原页面。弹出层是响应式的，在小屏幕下只显示大图和关闭按钮，没有轮播效果。

13.6.3　弹出层显示

弹出表单或者视频播放等内容是一个很常见的需求，Foundation 自带的 JavaScript 控件也自带了弹出层的功能。用法如下：

```
<button data-reveal-id="myModal">点击弹出</button>
<div id="myModal" class="reveal-modal" data-reveal>
  弹出层内容……
</div>
```

该效果由两部分代码组成：一个是按钮或链接，用于点击触发弹出；另一个是包裹弹出层内容的 标签。其中 标签需要带有一个 id、一个.reveal-modal 类、一个名为 data-reveal 的触发器。按钮或链接需要加上 data-reveal-id=id 的值。这样一个基于 Foundation 的弹出层效果就制作好了，效果如图 13.34 所示。

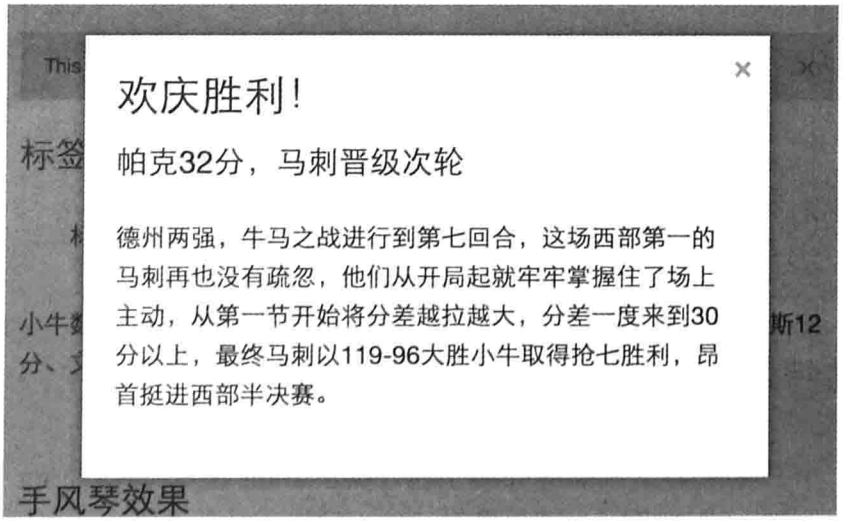


图 13.34　弹出层样式

13.6.4 长页面滚动效果

对于展示型页面来说，长页面配合头部导航的切换是一种很常见的做法，最简便的方式当然是使用锚点定位，但是这样画面的切换非常突兀，使用页面滚动的方式则更符合视觉效果。对此 Foundation 提供了一个整体的解决方案：

```
<div data-magellan-expedition="fixed">
  <dl class="sub-nav">
    <dd data-magellan-arrival="build"><a href="#build">Build with HTML</a></dd>
    <dd data-magellan-arrival="js"><a href="#js">Arrival 2</a></dd>
  </dl>
</div>
<h1 data-magellan-destination="build">Build</h3>
<p>About Build</p>
<h1 data-magellan-destination="js">Js</h3>
<p>About Js</p>
```

首先需要构建一个使用 position:fixed 固定在上方的导航条，需要在导航外层包裹一层 ..... 。

然后为定位区域开头的元素添加 data-magellan-destination="区域名称"，然后为对应的导航添加 data-magellan-arrival="同样的区域名称"，这样就可以将导航和对应的区域进行绑定，点击导航项，页面就可以滚动到相应的区域，同时导航项显示为选中状态。如果采用鼠标滚动到某一区域，对应的导航项也会显示为选中状态，如图 13.35 所示。当页面处于“成长”区域时，导航上的“成长”项处于选中状态。

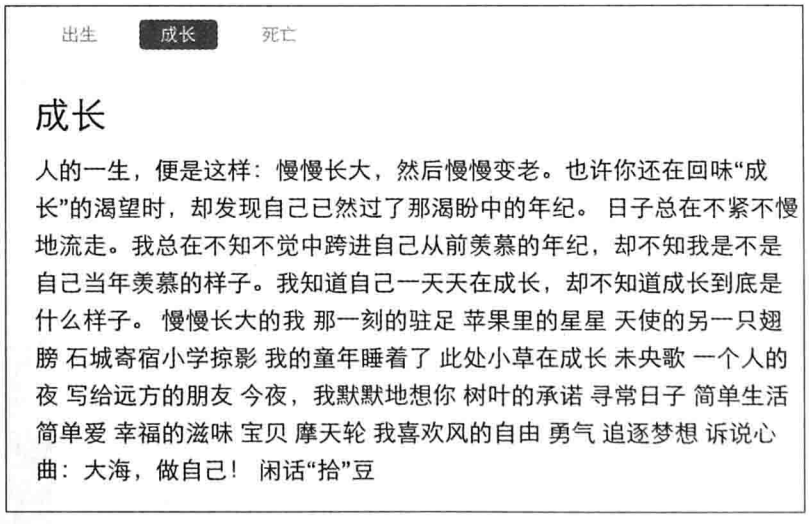


图 13.35 长页面滚动效果

13.6.5 其他特效

限于篇幅，本书无法为读者详细介绍全部 Foundation 中的 JavaScript 特效，更多的细节需要查找官方文档，这里可以为读者简单地介绍 Foundation 的 JavaScript 效果列表，方便读者查询。

除了前面详细介绍的几种 JavaScript 效果以外，Foundation 还对以下几种效果进行了集成：

- 下拉按钮，如图 13.36 所示。

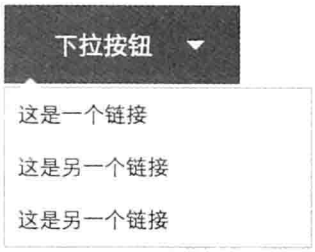


图 13.36　下拉按钮

- 侧向展开/收起的头部导航，如图 13.37 所示。

图 13.37　侧向展开/收起的头部导航

- 表单验证，如图 13.38 所示。

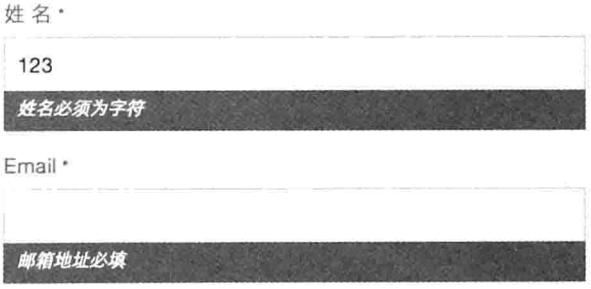


图 13.38　表单验证

- 提示信息，如图 13.39 所示。

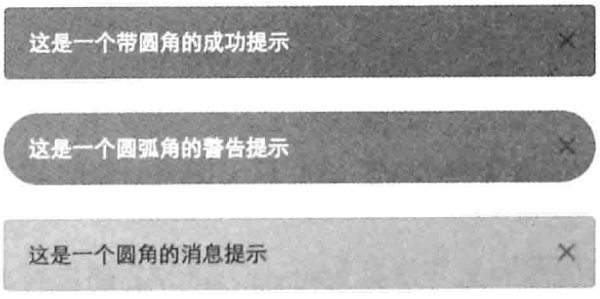


图 13.39　提示信息

- 鼠标移上时的注释说明，如图 13.40 所示。

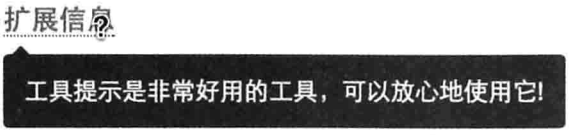


图 13.40　注释说明

- 页面第一次访问时出现的介绍窗口，如图 13.41 所示。

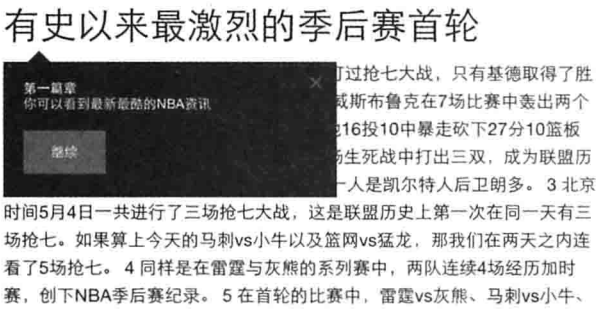


图 13.41　首次访问出现的窗口

- 手风琴展开/收起效果，如图 13.42 所示。

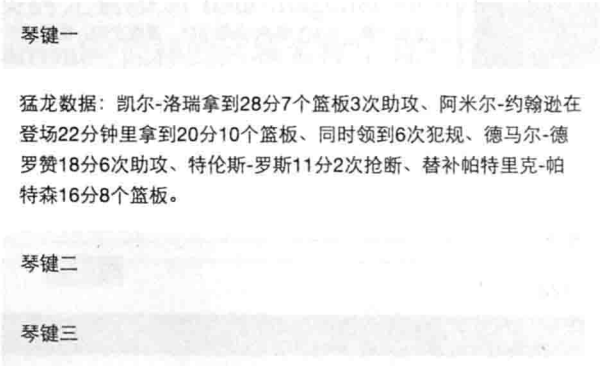


图 13.42　手风琴效果

- Tab 切换内容，如图 13.43 所示。

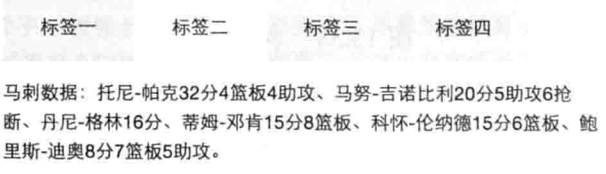


图 13.43　Tab 切换

13.7　定制 Foundation

与 Bootstrap 相比，Foundation 的订制更为方便，它不仅和 Bootstrap 一样提供在下载前在网站上设置自定义参数，还可以通过修改 SASS 配置文件来进行更加灵活的定制，甚至

可以自由地选择要加入的模块。

13.7.1　在官方网站进行定制

如果开发者采用下载编译后的 CSS 进行开发的方式，那么就可以根据自己的需求在官网下载页面的 Custom CSS 版块进行定制，如图 13.44 所示，选择第 2 项。

图 13.44　Foundation 下载页面

进入定制化页面后，可以选择需要的模块并配置可选的参数，如图 13.45 所示。

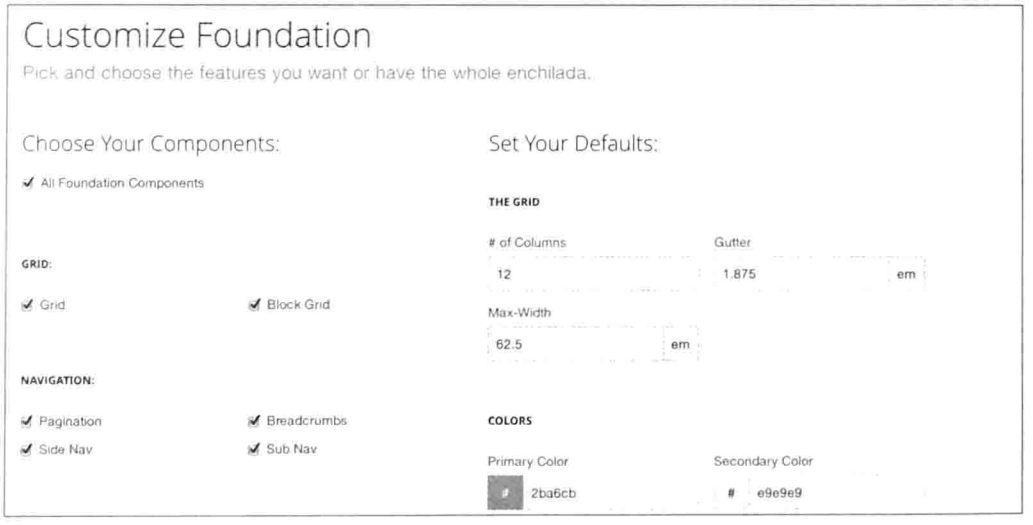


图 13.45　Custom CSS 版块

不过显而易见的是，如果有计划外的需求，就不得不重新进行选择和下载，十分不方便。

13.7.2　通过配置文件进行定制

如果采用 Foundation 提供的 SASS&Compass 集成开发环境的话，通过配置文件进行定制是最好的选择。图 13.46 左侧的目录树是生成的文件结构，右侧打开的就是 Foundation 的配置文件。

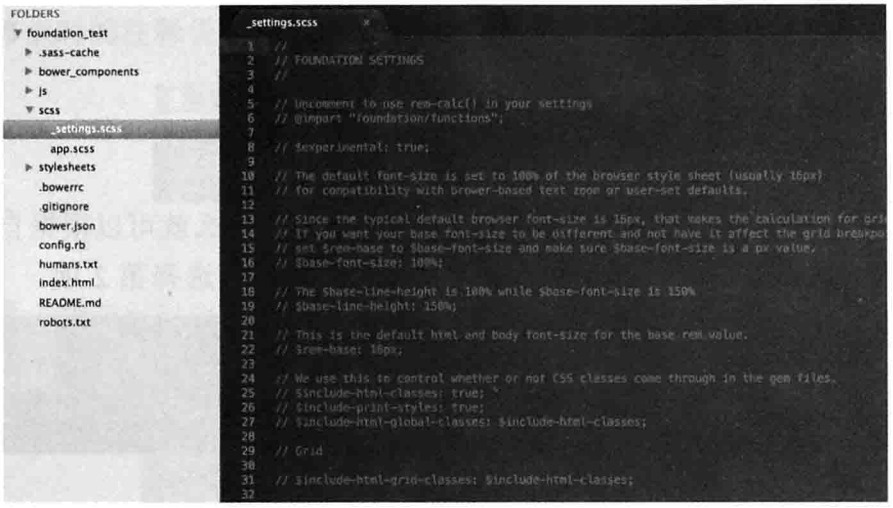


图 13.46　配置文件

整个配置文件都处于被注释的状态，所有的值都取默认值，如果需要修改某一个样式的配置数值，只需要将那一行的注释取消并将值设置成我们想要的就可以了。

使用集成开发环境也可以订制模块，只挑选需要的模块进行加载，以最小化 CSS 文件，从而优化速度，如图 13.47 所示。

图 13.47　挑选需要的模块

图 13.47 中第 2 行@import "foundation";表示加载 Foundation 的全部模块。如果注释掉这一句，然后选择图中被注释掉的单个模块，就可以做到只加载需要的模块，从而避免加载整个 Foundation 造成体积过大的问题。

13.8　小结

本章主要讲解了 Foundation 框架，它从性质上来说和 Bootstrap 是一样的，都是一套完整的前端框架，可以让开发者只编写少量的 CSS 和 JavaScript 就可以完成美观的页面。

对比 Bootstrap，Foundation 牺牲了 IE 9 以下版本的兼容换来了更好的移动响应式页面效果，在 UI 方面也更偏向移动设备。在特性方面，Foundation 提供了块网格、价格表、响应式视频窗口等独有的工具。

不过 Foundation 目前还没有形成自己的生态链条，很少有开发者基于 Foundation 的规范制作皮肤、教程等扩展，再加上对低版本浏览器的兼容性问题，虽然在很多地方 Foundation 表现得更为优秀，但目前还是难以撼动 Bootstrap 的地位。

Foundation 整合的主要样式效果可以分为以下几类：

- 基础样式，包括标题、段落、列表等基础元素的样式。
- 栅格系统，包括响应式栅格和块网格。
- 导航系统，包括头部导航、侧边栏、子导航、分页等。
- 提示系统，包括模态弹出层、文字提示、解释说明等。
- 表单系统，包括表单样式和错误提示。
- 按钮系统，包括按钮样式、按钮组、下拉按钮等。
- 内容系统，包括表格、价格表、切换显示、下拉菜单等。
- 响应式多媒体，包括响应式的图片、幻灯、视频窗口等。

Foundation 整合的 JavaScript 效果主要包括以下内容：

- 幻灯片。
- 大图展示效果。
- 弹出层效果。
- 长页面滚动切换效果。
- 下拉按钮。
- 侧向展开/收起的头部导航。
- 表单验证。
- 提示信息。
- 鼠标移上时的注释说明。
- 页面第一次访问时出现的介绍窗口。
- 手风琴展开/收起效果。
- Tab 切换内容。

本章旨在帮助读者了解 Foundation 框架，了解其安装和使用方式，并对其中较重要、较有特色的组件进行了示例演示，但限于篇幅，无法像官方手册一样对每一个组件都给出详细的介绍和示例。有需要的读者可以访问 Foundation 的官方网站 http://foundation.zurb.com/，获取更详细的信息。

第 14 章　LESS 和 SASS

CSS 并不能算是一门真正意义上的“编程”语言，它本身无法完成像其他编程语言一样的嵌套、继承、设置变量等工作，因此很难实现 DRY（Don’t repeat yourself）的原则，CSS 文件中常常充斥着大量重复的定义，不但编写时很难组织，代码量庞大，而且随着项目规模的扩大，维护会越来越困难。

为了解决 CSS 的不足，开发者想到了编写一种对 CSS 进行预处理的“中间语言”，可以实现一些“编程”语言才有的功能，然后自动编译成 CSS 供浏览器识别，这样既一定程度上弥补了 CSS 的缺陷，也无须设计一种新语言来代替 CSS 以供浏览器识别。于是 CSS 预处理语言（CSS Preprocessor）就应运而生了。

本章主要的知识点如下：

- 了解 CSS 的缺陷。
- 学习两种目前最流行的 CSS 预处理语言 LESS 和 SASS。
- 掌握 Compass 的使用方法。

14.1　CSS 的缺陷

CSS 作为一种标记语言可以很好地完成页面样式的定义，但是一些标记语言固有的缺陷也限制了编写 CSS 的效率。要提高效率，一方面依赖工具，比如编辑器的自动补全或像 Dreamweaver 这类的图形化工具；一方面也依赖于语言本身的改进，但是由于 CSS 和浏览器以及互联网上的历史数据紧密耦合，很难像 Python、Ruby 这类主要运行于服务端的语言一样可轻易进行语法升级，这样就使得一些缺陷不得不一直继承了下来。本节就来了解这些传承了很久的缺陷。

14.1.1　无法定义变量

CSS 在定义颜色时一般采用十六进制的 RGB 模式，例如#333333 这样的格式，但是开发者很难记住编号，而默认支持的 red、blue、black 等颜色一方面不够丰富，另一方面实际工程中用的并不多。那么为什么不能把色彩的 RGB 编号赋予一个变量呢？这是一个很实际的需求，但是很遗憾，CSS 并不能做到。

再比如我们在进行一些 CSS 3 效果的定义时，经常会需要针对不同的浏览器写很多行

带有属性前缀的定义，为什么不能用类似宏命令的形式用一句话来指代这好几行的定义呢？

14.1.2 重复代码

CSS 的继承机制是根据 HTML 的层级关系来决定的，如果 HTML 文档中存在父子元素的关系，那么子元素可以继承父元素的部分属性，比如字体、背景等。而很多情况下，几个元素拥有类似的定义，却没有父子关系，那么就只能分别进行定义，结果就造成了大量的重复代码。在编码上和维护上都对开发者们造成了一定的负担。

14.1.3 计算问题

CSS 没有变量，也就谈不上计算。CSS 一般都是自己算好固定值填写上去，不过假设有很多值都是相同的，或者是以某个值为基准进行计算的，显然使用带有变量的算式表达更为合理，既方便编写和修改，也可以大大避免犯错的概率。

假设我们设定了一个 normal-length=20px，所有的长度都以它为基准：

```
.a {width:20px;}
.b {width:19px;}
.c {width:40px;}
```

一旦我们决定修改 normal-length 的值，譬如改为 19px；那么.a、.b、.c 3 处定义都必须重新计算并修改。遇到这种情况我们一定会想，能不能像下面这样定义呢：

```
$nl :20px;
.a {width: $nl;}
.b {width: $nl-1px;}
.c {width: 2$nl;}
```

这样只需修改 nl 的值就完成了。

14.1.4 作用域和命名空间

CSS 通过子元素选择器或后代元素选择器可以实现作用域和命名空间，例如下面的代码：

```
#main div li {
  list-sytle:none;
}
#main .container{
  margin:auto;
  width:960px;
}
#main a{
  text-decoration:none;
}
.....
```

上面这个例子，所有的样式都在 id="main"这个元素的包裹内才会生效，这样在编写时每一句定义都需要添加#main 语句。在编写时问题或许还不大，但是在需要添加命名空间时，就非常麻烦了，比如要给下面的定义统一添加#main 的命名空间：

```
div li {
   list-sytle:none;
}
.container{
   margin:auto;
   width:960px;
}
a{
   text-decoration:none;
}
....../*如果这里有几十个定义，需要添加命名空间就很难处理了*/
```

14.1.5 CSS 缺陷总结

CSS 作为一种简单易学的标记语言走过了很长的旅程，并且一直在不断地完善。但是由于其不能进行逻辑运算，无法方便地继承和模块化，在实际开发时使得开发者不得不花费很多笨功夫。由于兼容性问题，目前要设计新的语言来代替 CSS 是不太现实的。而本章将要介绍的 CSS 预处理语言则利用其他编程语言，巧妙地解决了 CSS 存在的不足。

14.2 LESS 其实更多

为了解决 CSS 的种种不足，开发者们采取了很多手段，目前来说，使用 CSS 预处理语言是一种比较有效的方式，而 LESS 则正是目前应用最为广泛的一种。

14.2.1 LESS 介绍

LESS 诞生于 2009 年，是使用 JavaScript 语言编写的一种 CSS 预处理语言，它为 CSS 赋予了编程语言的特性，如变量、继承、运算、函数等。LESS 既可以在客户端上运行（支持 IE 6+、webkit 核心浏览器、Firefox），也可以借助 Node.js 或 Rhino 在服务端运行。

LESS 是笔者最为推荐的一款 CSS 预处理语言，或许功能上它并不是最强的，但却是安装最简单的开发工具，而且互联网上的资料也是最多的，尤其是中文互联网的汉化是所有预处理语言中最好的。对于国内的开发者来讲，LESS 的学习成本最低，再加上它的源码采用的是 JavaScript 这款大多数前端工程师都熟悉的脚本语言，对应的工具开发也有开发者社区强有力的支持。LESS 官方网站如图 14.1 所示。

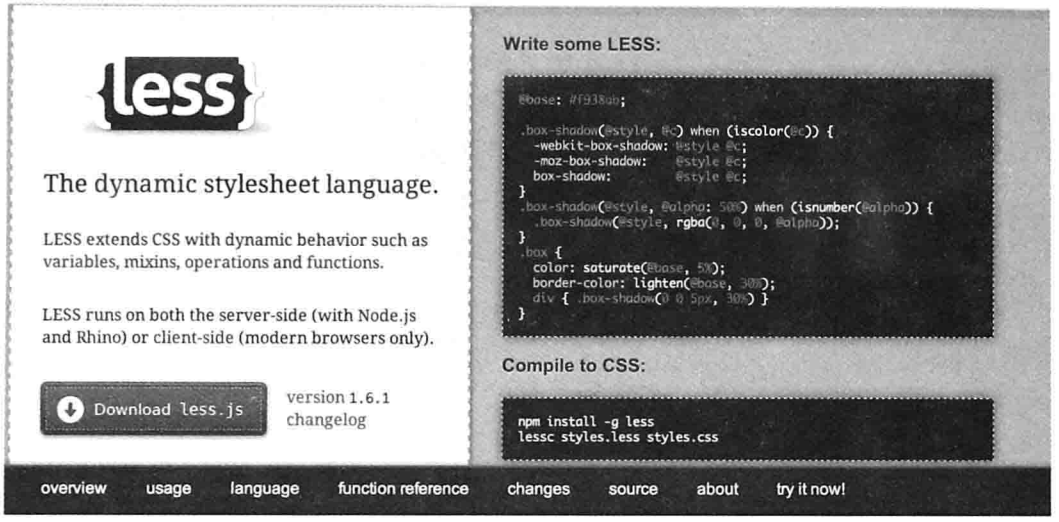


图 14.1　LESS 官方网站

14.2.2　LESS 使用基础

编写好的 LESS 文件只有编译为 CSS 文件后才能被浏览器识别，有很多 LESS 编译工具可供选择。下面是一些目前较为流行的 LESS 编译工具。

（1）Koala，全平台的 LESSCSS/SASS 编译工具，下载地址：http://koala-app.com/index-zh.html。

（2） Codekit，MAC 下自动编译 Less/Sass/Stylus/CoffeeScript/Jade/Haml 的工具，含语法检查、图片优化、自动刷新等附加功能，下载地址：http://incident57.com/codekit/。

（3）WinLess，Windows 下的 LESS 编译软件，下载地址 http://winless.org/。

（4）SimpleLess，全平台 LESS 编译软件，适用于 Windows、Linux 和 MAC 操作系统。下载地址：http://wearekiss.com/simpless。

除了使用编译工具，LESS 还提供客户端调试方式，步骤如下所示。

（1）引入.less 样式文件：

```
<link rel="stylesheet/less" type="text/css" href="http://localhost/styles.less">
```

注意：要设置 rel 值为“stylesheet/less”。客户端调试方式下需要引入 http 链接的.less 样式文件，使用本地的.less 文件会报错。

（2）下载 less.js，在<head>中引入：

```
<script src="less.js" type="text/javascript"></script>
```

注意：less 样式文件一定要在引入 less.js 前先引入。

完成这两个步骤后，刷新网页就可以看到修改 less 文件后发生的变化了。

14.2.3　使用变量和操作符

在 LESS 中，使用@关键字进行变量的定义。如下面的这个示例：

```
/*LESS 代码*/
@color: #4D926F;
#header {
    color: @color;
}
h2 {
    color: @color;
}
/*编译后的 CSS 代码*/
#header {
    color: #4D926F;
}
h2 {
    color: #4D926F;
}
```

这样，只需要记住@color 这个有实际意义的变量名称，就无须在编码时去粘贴复制难记的 RGB 颜色代码，并且对于整体的配色调整，只需要修改一个地方就可以完成全局的更新，极大地方便了后期的升级和维护。

14.2.4 使用 Mixin 混入

Mixin 这个概念在很多编程语言中都有，一般译作混合、混入。类似 C 语言的宏（macro），是可以重用的代码块。

Mixin 在这里的具体作用就是把一些通用的定义抽取出来，以后就无须编写重复的代码了。对于 CSS，尤其是 CSS 3 被引入之后，由于各大浏览器厂商推行各自的标准，导致开发者为了浏览器兼容性必须编写大量带有属性前缀的代码。Mixin 用在这里正是再合适不过了，比如下面这个示例：

```
/*LESS 源码：*/
.rounded-corners (@radius: 5px) {
    -webkit-border-radius: @radius;
    -moz-border-radius: @radius;
    -ms-border-radius: @radius;
    -o-border-radius: @radius;
    border-radius: @radius;
}

#header {
    .rounded-corners;
}
#footer {
    .rounded-corners(10px);
}
/*编译后的 CSS 代码：*/
```

```
#header {
    -webkit-border-radius: 5px;
    -moz-border-radius: 5px;
    -ms-border-radius: 5px;
    -o-border-radius: 5px;
    border-radius: 5px;
}
#footer {
    -webkit-border-radius: 10px;
    -moz-border-radius: 10px;
    -ms-border-radius: 10px;
    -o-border-radius: 10px;
    border-radius: 10px;
}
```

Mixin 的语法关键字是一个.符号，可以将其联想记忆为 CSS 选择器中的类。通过这个示例可以看到，对于像圆角这类需要属性前缀的 CSS 3 属性，以及其他类似的通用模块都可以采用 Mixin，从而实现一次定义，无限使用，既可以大大缩减无谓的重复定义，又提高了代码的可读性和可维护性。

14.2.5　内嵌规则

LESS 可以用嵌套的方式编写层叠样式，之前我们在讨论 CSS 的缺陷时曾经举了这样一个例子：

```
#main div li {
  list-sytle:none;
}
#main .container{
  margin:auto;
  width:960px;
}
#main a{
  text-decoration:none;
}
.....
```

如果使用 LESS 进行编写：

```
#main{
  div li {
    list-sytle:none;
  }
  .container{
    margin:auto;
    width:960px;
  }
```

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

这样代码更为简洁，并且更易于维护。

14.2.6 运算

任何数字、颜色或者变量都可以参与运算，例如：

```
@base: 5%;
@filler: @base * 2;
@other: @base + @filler;
color: #888 / 4;
background-color: @base-color + #111;
height: 100% / 2 + @filler;
```

LESS 的运算能够分辨出颜色和单位，例如：

```
@length: 1px + 7;
```

LESS 会输出 8px。

可以使用括号来改变运算的优先级：

```
width: (@var + 5) * 2;
```

可以在复合属性中进行运算，例如：

```
border: (@width * 2) solid black;
```

14.2.7 LESS 总结

LESS 是当前最流行的 CSS 预处理语言，它是采用前端工程师熟悉的 JavaScript 语言来编写的。简单易学、文档丰富、拥有多种图形化的编译工具是 LESS 的优势所在。

使用 LESS，可以较好地解决 CSS 编写中暴露出的不能进行变量定义、无法计算、重复代码过多、难以进行嵌套和命名空间的设置等显著问题。

14.3 使用 SASS

SASS 是最早的 CSS 预处理语言，有比 LESS 更为强大的功能，不过其一开始的缩进式语法并不能被大众接受，不过由于其强大的功能和 Ruby on Rails 的大力推动，还是有很多开发者选择了 SASS。

14.3.1 SASS 介绍

SASS 是采用 Ruby 语言编写的一款 CSS 预处理语言，它诞生于 2007 年，是最早的成

熟 CSS 预处理语言。最初它是为了配合 HAML（一种缩进式 HTML 预编译器）而设计的，因此有着和 HAML 一样的缩进式风格。SASS 官网如图 14.2 所示。

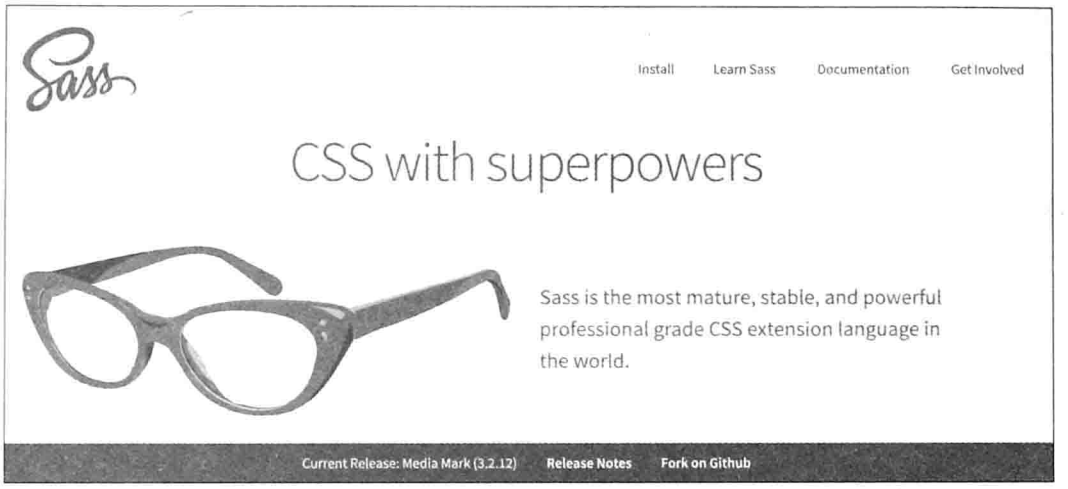


图 14.2　SASS 官方网站

虽然缩进式风格可以有效缩减代码量，强制规范编码风格，但它一方面并不为大多数程序员接受，另一方面无法兼容已有的 CSS 代码。这也是 SASS 虽然出现得最早，但是远不如 LESS 普及的原因。

SASS 从第三代开始，放弃了缩进式的风格，并且完全向下兼容普通的 CSS 代码，这一代的 SASS 也被称为 SCSS。

相对 LESS 来说，SASS 功能上更强大，再加上有基于 SCSS 的著名类库 Compass，以及 Ruby on Rails 框架（一款非常著名的 Web 全栈式开发框架）的默认支持，SASS 目前地使用也非常广泛。

14.3.2　SASS 安装和使用

SASS 是采用 Ruby 语言编写的，不过使用 SASS 不需要懂 Ruby，但是想使用 SASS 就必须先安装 Ruby，因此相对 LESS 在安装上要麻烦一些。

第一步：安装 Ruby。

第二步：Ruby 安装完成后，就可以使用 gem 系列命令安装 Ruby 的插件和扩展，在终端输入 gem install sass 进行安装，安装完成后就可以使用 sass 命令了。在终端下输入 sass –help 可以查看 sass 命令的选项。

14.3.3　使用变量

SASS 的变量关键字和 PHP 一样，都是以一个美元符号$开头，例如：

```
$blue : #1875e7;
div {
  color : $blue;
};
```

SASS 还支持变量内嵌在字符串中，由于 SASS 是 Ruby 语言编写的，这里的语法也和 Ruby 一样，使用#{}进行包裹。例如：

```
$side : left;
.rounded {
border-#{$side}-radius: 5px;
}
```

编译后的 CSS 代码就是：

```
.rounded {
border-left-radius: 5px;
}
```

14.3.4 计算

和 LESS 一样，SASS 允许直接在代码中编写算式，并且支持变量和函数，例如：

```
$width: 1000px;
body {
  margin: (14px/2);
  padding: pi()px;
  top: 50px + 100px;
  right: $width * 10%;
}
```

转换为 CSS 就是：

```
body {
  margin: 7px;
  padding: 3.14159px;
  top: 150px;
  right: 100px;
}
```

14.3.5 使用@import 导入

@import 可以用来插入外部文件，例如：

```
@import "path/filename.scss";
```

如果插入的是 CSS 文件，则等同于 CSS 的 import 命令。

```
@import "foo.css";
```

注意：@import 的顺序和先后有关，权重较高的文件应当放在靠后的位置引入。

14.3.6 使用@extend 继承

SASS 支持编程语言中的继承概念，具体到这里就是可以通过@extend 关键字来继承一个已有的定义。比如下面这个示例：

```
.class1 {
  border: 1px solid #ddd;
}
.class2 {
  @extend .class1;
  font-size:120%;
}
```

class2 完全继承了 class1 中的定义，拥有一样的边框，同时 class2 中自己定义的字体大小为 120%。

14.3.7　使用@mixin 混入

使用@mixin 命令，可以定义一个代码块，示例如下：

```
@mixin float_left {
  float: left;
  margin-left: 10px;
}
```

使用@include 命令，调用这个 Mixin：

```
div {
  @include float_left;
}
```

和 LESS 一样，SASS 的 Mixin 也支持参数。

14.3.8　使用@function 定义函数

SASS 允许用户编写自己的函数，例如：

```
@function double($n) {
    @return $n * 2;
}
#sidebar {
    width: double(5px);
}
```

14.3.9　控制语句

编程语言都有程序控制语句，来控制代码的运行方向。SASS 中也有@if、@else、@while 等控制语句。

（1）@if 可以用来判断：

```
p {
      @if 1 + 1 == 2 { border: 1px solid; }
      @if 5 < 3 { border: 2px dotted; }
}
```

配套的还有@else 命令：

```
@if lightness($color) > 30% {
        background-color: #000;
} @else {
        background-color: #fff;
}
```

（2）for 循环：

```
@for $i from 1 to 10 {
.border-#{$i} {
        border: #{$i}px solid blue;
}
}
```

（3）while 循环：

```
$i: 6;
@while $i > 0 {
.item-#{$i} { width: 2em * $i; }
$i: $i - 2;
}
```

（4）each 命令，作用与 for 类似：

```
@each $member in a, b, c, d {
.#{$member} {
background-image: url("/image/#{$member}.jpg");
}
}
```

14.3.10 SASS 总结

SASS 是最早的 CSS 预编译语言，在功能上和 LESS 相比更为强大，如支持分支和循环操作、支持自定义函数。但是由于 Ruby 在前端领域不如 JavaScript 流行，以及 SASS 一开始采取的缩进式语法等原因，其在流行度上不如 LESS。

以笔者的经验来看，SASS 结合 Compass 是相比 LESS 更为强大的工具，但是一方面实际需求可能用不到一些更强的功能，另一方面学习成本也更高。开发者应该根据自身团队的情况选择适合自己的技术。

14.4 使用 SASS 的扩展库 Compass

前面介绍了 SASS 中的 Mixin 和 Function，但是还有很多样式都是通用的，比如 reset（重置 CSS 样式），比如 CSS 3 中带有属性前缀的兼容代码，再比如链接的颜色、下划线等设置。如果开发者决定使用 SASS 或者 LESS，定义这些 Mixin 和 Function 也是一件不小的工作，基于软件工程中“不要重复造轮子”的名言，Compass 应运而生。

Compass 是一个基于 SASS 的类库（官网如图 14.3 所示），项目主页地址是 http://compass-style.org/，它帮助我们预定义好了很多常用的 Mixin 和 Function。Compass 由以下几个模块组成。

- CSS 3：将 CSS 3 的带有属性前缀的兼容代码组合成 Mixin
- Reset：用于清除浏览器自带样式，保证不同浏览器下显示的一致性
- Utilities：对一些常用的 CSS 样式进行简化
- Helpers：提供一些常用的函数
- Layout：提供栅格系统和一些简单的布局样式

图 14.3　Compass 官方网站

注意：目前 LESS 中还没有像 Compass 一样成熟且有影响力的类库，如果需要的话可以使用 Comless 和 VeryLess 这两个类似的库。

14.4.1　CSS 3 模块

Compass 的 CSS 3 模块实际就是将开发中经常用到的 CSS 3 属性用语义性很强的 Mixin 进行了封装，节约了开发者自己定义的时间。CSS 3 模块也是 Compass 应用最为广泛的模块。要使用 CSS 3 模块，首先要在 SASS 文件头部引入：

```
@import "compass/css3"
```

下面举两个例子为读者介绍 Compass 的 CSS 3 模块的用法。

1．元素透明度的设置

第 1 个例子是元素透明度的设置。IE 9 以前的 IE 浏览器不支持 opacity 属性的，而同样的效果需要用到 IE 独有的滤镜来实现，但绝大多数开发者记不住滤镜的写法。此时使用 Compass 就非常简单了：

```
#opacity-50 {
  @include opacity(0.5);
}
```

简单的一行定义就解决了这个问题。

下面是 Compass 中对 opacity 这个 Mixin 的实现：

```
@mixin opacity($opacity) {
 @if $legacy-support-for-ie6 or $legacy-support-for-ie7 or $legacy-support-for-ie8 {
   filter:      unquote("progid:DXImageTransform.Microsoft.Alpha(Opacity=#{round($opacity      *
100)})");
 }
 opacity: $opacity;
}
```

实际上它是将正常的 opacity 属性和 IE 的滤镜写在了一个 Mixin 中，在使用中，只需要用@include 关键字对 Mixin 进行调用就可以。

2．圆角的设置

第 2 个例子是圆角的设置。Compass 不仅简单地支持 4 个角都是圆角，还支持指定哪一个角为圆角，例如：

```
/*分别为 4 个角设置不同的圆角弧度*/
#border-radius {
  @include border-radius(25px);                    /*4 个角全是圆角*/
}

#border-radius-top-left {
  @include border-top-left-radius(25px);           /*左上角为圆角*/
}

#border-radius-top-right {
  @include border-top-right-radius(25px);          /*右上角是圆角*/
}

#border-radius-bottom-left {
  @include border-bottom-left-radius(25px);        /*左下角是圆角*/
}

#border-radius-top {
  @include border-top-radius(25px);                /*上方两个角是圆角*/
}

#border-radius-bottom {
  @include border-bottom-radius(25px);             /*下方两个角是圆角*/
}

#border-radius-combo {                                          /*设置 4 个弧度不同的圆角*/
  @include border-corner-radius(top, left, 40px);               /*左上角为 40px 的圆角*/
  @include border-corner-radius(top, right, 5px);               /*右上角为 5px 的圆角*/
  @include border-corner-radius(bottom, left, 15px);            /*左下为 15px 的圆角*/
  @include border-corner-radius(bottom, right, 30px);           /*右下角为 30px 的圆角*/
}
```

上述代码除了最后 4 个角都不同的“高难度动作”外，其他的设置都是一行代码就搞定。而对比#border-radius-bottom 编译后等效的 CSS 代码：

```
#border-radius-bottom {
  -moz-border-radius-bottomleft: 25px;
  -webkit-border-bottom-left-radius: 25px;
  border-bottom-left-radius: 25px;
  -moz-border-radius-bottomright: 25px;
  -webkit-border-bottom-right-radius: 25px;
  border-bottom-right-radius: 25px;
}
```

可以看出，在普通 CSS 中一共要写 6 行代码，而使用 Compass 一行就搞定了，这个对比充分显示了 Compass 的优越性。

Compass 目前一共定义了 20 种常用的 CSS 3 效果，限于篇幅，本书无法一一为读者详细介绍，读者可以访问 Compass 的官方网站进行查阅。

14.4.2　Reset 模块

它的功能类似 Bootstrap 3 中集成的 normalize.css，用于清除浏览器的默认样式，尽量保持网站在不同浏览器下的一致性表现。

要使用 Reset 模块首先要在 SASS 文件头部引入：

```
@import "compass/reset/utilities"
```

这样就可以实现清除浏览器的默认样式了。

不同于其他样式重置方案的是，Compass 不仅仅提供全局的样式重置，还可以针对某些指定元素进行重置。例如使用 reset-box-model 进行盒子模型的重置：

```
.box{
  @include reset-box-model;
}
```

这样可以去除盒子模型的内外边距和边框。

再比如用 reset-display 对元素的 display 进行重置：

```
.inline{
  @include reset-display;
}
```

这样可以恢复元素原先的 display 值。

14.4.3　Utilities 模块

和 Utility 这个单词的英文意义“实用”、“常用”一样，这个模块是对开发中经常用到的样式进行封装。主要包括以下几个分类。

- Links：常用链接样式。
- Lists：常用列表样式。

- Text：文本样式的辅助方法。
- Color：常用颜色。
- General：其他常用样式。
- Sprites：“图片精灵”，即对 background-position 的封装。
- Tables：表格样式的辅助方法。

下面举两个例子进行说明。

链接在平时不显示下划线，而在鼠标移上时显示下划线，这是页面制作中常见的一种设计。下划线的显示由 text-decoration 属性控制，这个属性既不好记，又不好拼，开发者遇到这个问题经常不得不借助搜索或字典，而 Compass 中的处理则十分简单好记：

```
a{
  @include hover-link;
}
```

编译后的 CSS 代码：

```
a {
  text-decoration: none;
}
a:hover {
  text-decoration: underline;
}
```

再举一个常见的需求：在文字长度超出框体时，截断文字并显示省略号。相信很多开发者都会遇到这个问题，但设计时往往需要查询搜索引擎或者手册才能完成。这里使用 Compass 提供的 ellipsis，它是一个包装好的 Mixin，同样一行代码搞定：

```
p{
  @include ellipsis;
}
```

编译后的 CSS 代码：

```
p{
  white-space: nowrap;
  overflow: hidden;
  -ms-text-overflow: ellipsis;
  -o-text-overflow: ellipsis;
  text-overflow: ellipsis;
}
```

限于篇幅，本书无法为读者一一进行介绍，有需要的读者可以在 Compass 的官方网站查询 Utilities 模块的完整版。

14.4.4 Helpers 模块

前面介绍的几个 Compass 模块主要都是应用 SASS 的 Mixin 特性，对常用的一些样式和兼容性代码进行封装。Helpers 模块则主要使用了 SASS 的函数特性，将一些操作封装成函数以提升开发效率。

Helpers函数列表可以在官方文档进行查询，下面会通过一个典型的例子进行介绍。

三角函数是Helpers模块中的重要组成部分，以大家最熟悉的正弦函数为例，在Compass中用sin()表示正弦函数，用法如下：

```
.test_deg{
  width: 200px * sin(30deg);
}
/*如果采用角度制需要加上 deg 作为单位，如果没有单位则默认为弧度制*/
.test{
  width: 200px * sin(pi()/2);
}
/*这里的 pi()也是 Compass 内置的一个函数，表示π，sin(π/2)=1 */
```

转换为CSS后就是：

```
.test_deg{
  width: 100px;
}
.test{
  width: 200px ;
}
```

14.4.5 Compass总结

Compass是基于SASS的一个扩展库，它可以帮助我们进一步地简化和抽象代码，并提升代码的兼容性，但前提是熟悉Compass的各种命名。

Compass的优势之一在于可以选择性地引入自己需要的模块，以笔者的经验来看，应用最多的是CSS 3和Utilities模块，可以有效地缩减代码，提高兼容性。相对来说，Layout模块则很少得到应用，因为开发者往往会采用Bootstrap等更丰富的框架来设置页面的布局。

LESS也有类似的库可以选用，例如VeryLess和Comless，但是目前尚不太成熟。

14.5 小结

本章主要通过分析CSS存在的诸如无法设置变量、无法进行计算、冗余代码过多、不易设置命名空间等缺陷，引出了目前流行的CSS预处理语言概念，并介绍了两种当前最流行、最完善的CSS预处理语言：LESS和SASS。

Compass是基于SASS的扩展类库，封装了开发中常用的样式集合、CSS 3兼容代码、函数等，Compass的强大功能也是众多开发者选择SASS的原因。

以LESS和SASS为代表的CSS预处理语言在功能上是接近的，都可以实现变量、运算、混入（Mixin）、函数、继承、嵌套等功能，大大简化了CSS编写和维护的难度。不过需要注意的是，CSS预处理语言最终是需要编译成CSS来执行的，因此在实际性能上它们和CSS是等价的。

第 15 章　其他 CSS 框架简介

虽然 Bootstrap 框架目前备受推崇，围绕它的生态系统也十分丰富，但并非任何情况下 Bootstrap 都适用，譬如纯粹的手机页面、展示型页面、要求兼容 IE 6 的建站项目等，对于这些需求，也有很多不错的 CSS 框架工具可供我们选择，本章就将介绍一些同样非常实用的 CSS 框架。

本章主要知识点如下：

- Pure CSS 框架。
- Ratchet 框架。
- 一些优秀的国产框架。

15.1　轻量级框架代表——Pure CSS

Pure 是雅虎推出的一款超轻量级 CSS 框架，不包含任何 JavaScript 组件和依赖，它的特点就是轻量，经过 gzip 压缩后只有 4.4KB，但已经包含了前端开发中最常用的栅格、按钮、表单、表格、菜单等样式组件，并且支持响应式。

Pure 可以很灵活的和其他 JavaScript 插件进行组合而不受限于固定的结构，其官方网站是 http://purecss.io/，如图 15.1 所示。

图 15.1　Pure 官方网站

Pure 同样支持分组件下载，可以在官方网站的定制化页面下载某个单独的模块。总的来说，Pure 代表了一类自由、轻量的前端框架，它们只提供栅格、常用的组件样式，非常适合于构造需要展现个性的页面和一些简单场景。比如在一个简单的表单填写或表格展示的页面一下子引入了 200 多 KB 的 Bootstrap，显然是没必要的。

和 Pure 类似的还有 YAML、Blueprintcss 等框架，这类轻量级框架对于快速构建复杂页面来说肯定是不如Bootstrap这类框架面面俱到，但比不依赖框架从头写还是会高效得多，同时可以避免 Bootstrap 的 UI 严重同质化的问题，在已有的项目中引入也不会像 Bootstrap 一样造成大面积的冲突。

15.2　手机页面 UI 框架——Ratchet 框架

Ratchet（如图 15.2 所示）是 Bootstrap 开发团队专门针对移动端开发的框架，不同于 Bootstrap、Foundation 这类同时兼顾 PC、平板、手机的响应式框架，它主要针对开发手机应用内的 webview 页面，以及像 PhoneGap 这类应用 Web 前端技术制作移动 APP 的场景。

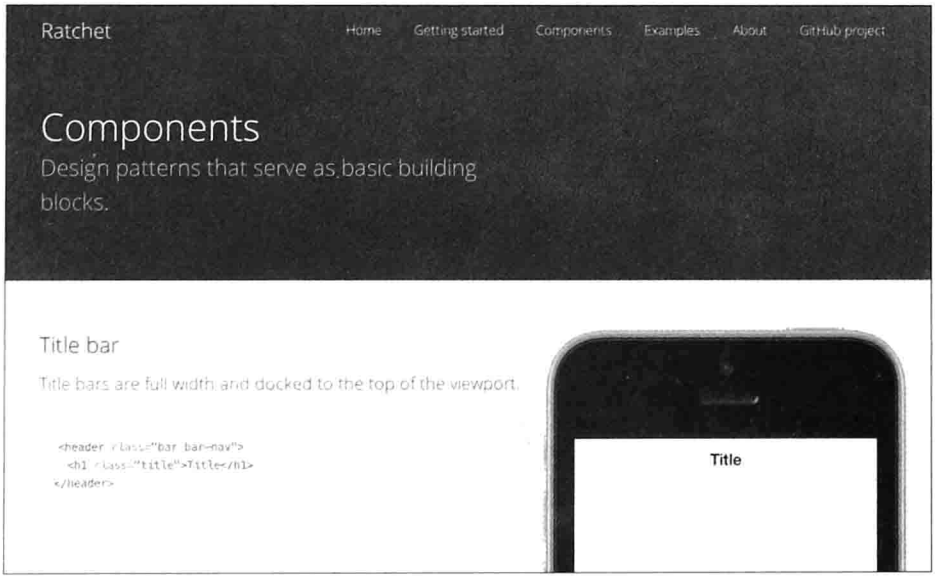


图 15.2　Ratchet 官方网站首页

Ratchet 模拟了和 iOS/Android 原生 UI 几乎一模一样的按钮、图标、进度条、滑块等样式。可以让 Web 页面构成的应用看起来十分类似原生应用，更符合用户的习惯，也更容易让 iOS 应用通过 App Store 的审核，如图 15.3 所示。

Ratchet 不仅仅可以用于开发，也可以用于产品原型设计。使用 Push.js 把页面链接起来并在 iPhone 上预览，就可以在移动产品设计、规划阶段进行高保真的模拟。

和 Ratchet 类似的还有 ionic（如图 15.4 所示），ionic 也是一款主要针对手机应用内页面的框架，相比 Ratchet，它和 Google 的前端 MVC 框架 Angular.js 结合得更紧密，可以通过 JavaScript 完成数据的双向绑定。

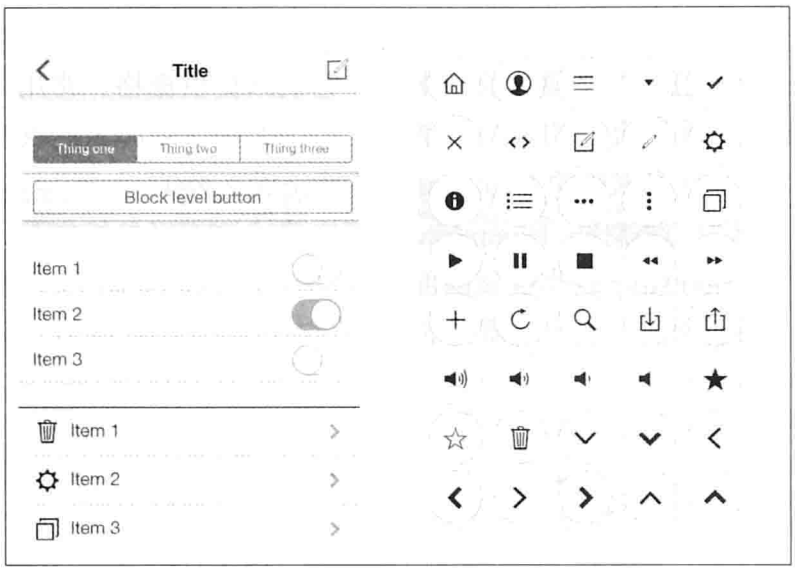


图 15.3　Ratchet 的设计元素节选

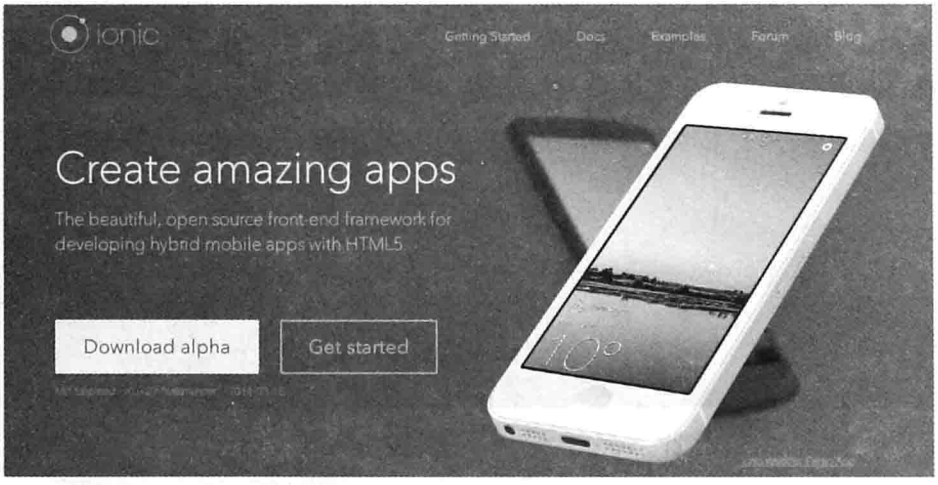


图 15.4　ionic 官网

像 Ratchet 和 ionic 这类框架已经脱离了传统页面开发的范畴，更多的用于 Web APP 和 Hybrid APP（混合动力应用，指同时使用原生接口/控件和 HTML/CSS/JavaScript 构建应用）的开发和原型构建。

15.3　优秀的国产 CSS 框架

前面介绍的 CSS 开发框架都出自国外的开发者之手，其实国内也有很多很不错的前端开发框架，有些地方，譬如浏览器兼容性等问题比国外做得更好，更适合中国国情。

15.3.1　阿里巴巴的 Alice 框架

Alice 是阿里巴巴前端团队共享的一套前端解决方案，包括基础框架、通用样式库、常用的 JavaScript 插件、前端开发规范和兼容性解决方案。官方网址是 http://aliceui.org/，如

图 15.5 所示。

图 15.5　Alice 官方网站

它的通用样式库是淘宝风格的，也没有人根据它来开发皮肤，很难适应多变的开发需求，JavaScript 部分的功能也和 Bootstrap 是重合的。但是它的兼容性处理部分笔者认为非常好，总结了绝大多数 IE6、7 下可能出现的兼容性问题并给出了完善的解决方案。这部分内容在官方页面上不好找到，可以访问它的 Github 地址：https://github.com/sofish/Alice。

此外 Alice 还总结了一些前端开发的规范，诸如命名规则、模块化组织等工作中的一些经验，非常值得借鉴。

15.3.2　网易的 NEC

NEC 是网易前端 CSS 开源项目代号，它包括规范、框架、代码库等内容。官方网址是。http://nec.netease.com/，如图 15.6 所示。NEC 并不像 Bootstrap 一样拥有自己的一套 UI 体系，它更像是一个代码库，或者说是经验总结，包含了常见的兼容性问题、页面布局方法、动画效果、代码格式规范等内容。

图 15.6　NEC 官方首页

NEC 相对来说更适合那些不依赖现成框架的场景，它有很多很好的总结可以帮助开发者少走弯路，而且非常适合拿来学习前端知识。如果需要一个 CSS 学习参照物的话，NEC

是一个非常好的选择。

15.3.3 百度的 GMU 框架

GMU 是百度开发的基于 zepto 的 mobile UI 组件库，主要为 Web App、pad 端简单易用的 UI 组件，目前在百度内部已经得到应用。

GMU 是百度开发的基于 zepto(一种类似 jQuery 的轻量级 JavaScript 类库)的 Mobile UI 组件库，主要目的是为 Web App 提供简单易用的 UI 组件。官方网站是 http://gmu.baidu.com/，如图 15.7 所示。

图 15.7 百度 GMU 官网

GMU 目前包括图片轮播、进度条、导航、弹出框、加载更多、搜索建议等组件，每个组件都根据不同的应用情景做了单独地处理，如图 15.8 所示。

图 15.8 GMU 包含的组件

15.3.4　渴切

前面几个框架都是脱胎于互联网大公司的内部项目，不过对于有着各种奇葩需求的外包网站建设来说，渴切或许更为实用。渴切官方网站是 http://download.keqie.com/，如图 15.9 所示。

图 15.9　渴切官网

渴切虽然也有类似 Bootstrap 的一套 UI 组件，但是笔者认为这方面它和 Bootstrap 还是有很长一段距离的。对 IE 兼容问题和一些国内建站常见问题的处理上，渴切有自己的优势。比如它总结了很多常见的效果，例如网站变灰、对联广告、特殊符号大全、IE 滤镜、滚动条样式调整等，还有像邮件模板等现成的样式。

它还总结了一些非常实用的代码片段，例如文字超出省略、网页不能另存、禁止 frame、关闭输入法、禁止复制粘贴等常用代码片段，如图 15.10 所示。

图 15.10　渴切总结的一些常用效果

15.3.5　用于中文排版的 Typo.css

除了各种通用的开发框架以外，国内也有一些优秀的针对特定用途的工具，比如

Typo.css。

Typo.css 是一款主要针对中文字体排版的样式组件，不仅可以保证排版的美观，还可以让文字在不同浏览器下保持一致。Typo.css 项目主页地址是 http://typo.sofish.de/，如图 15.11 所示。

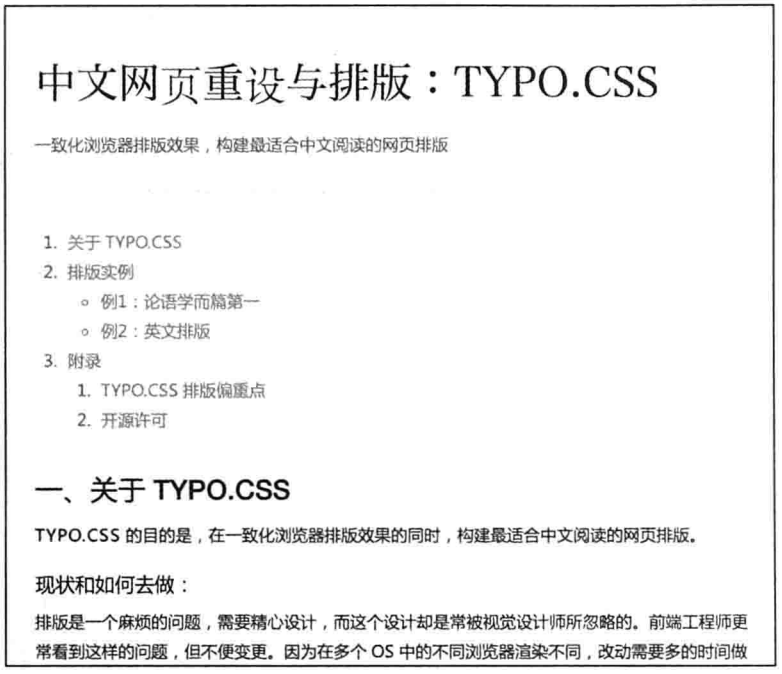


图 15.11　Typo.css 项目主页

大多数 CSS 框架对于排版都只有简单地定义，而如果应用国外的 CSS 框架，那么默认的排版完全针对的是英文等文字，对于中文来说效果并不理想。Typo.css 可以帮助开发者解决这个问题，如图 15.12 所示。

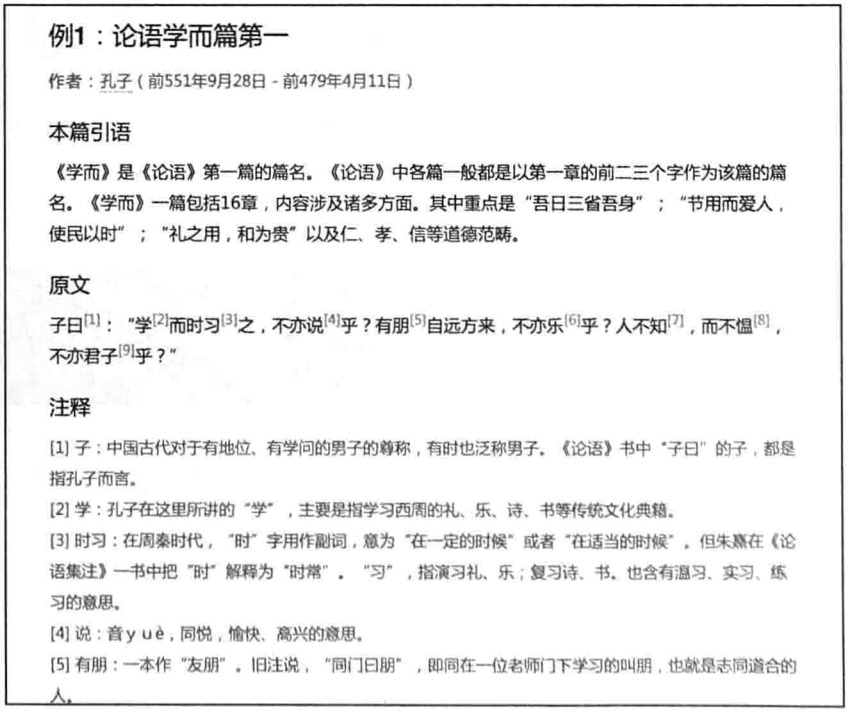


图 15.12　Typo.css 效果展示

15.4 小结

本章主要介绍了除 Bootstrap 和 Foundation 以外的其他优秀 CSS 框架，包括以 Pure 为代表的轻量级框架，以 Ratchet 为代表的手机页面 UI 框架，以及国内的一些优秀框架。

目前应用较多的轻量级框架有 Pure、YAML、Blueprintcss 等，它们基本是由一套栅格系统加上一套简单的样式集组成，非常小，一般只有几 KB，适合构建一些简单页面，或以其为基础的自定义样式，毕竟像 Bootstrap 这类框架太大，自定义的样式很容易与之冲突。

在移动应用开发领域，虽然应用 HTML 制作的 Web View 性能不如原生 UI，但是有着易开发、成本低、易扩展的优势，而且随着移动设备性能的提高，性能越来越不是问题。对于应用内部的页面来说，完全没有必要应用响应式来增加复杂度，反而更需要 UI 和原生的尽量保持一致，让 Hybrid APP（混合应用）更接近于原生应用。而以 Ratchet、ionic 为代表的框架则正是为了满足这种需要而诞生的。

中国的互联网环境有其独特性，譬如大量的传统企事业单位和政府部门统一使用正版的 Window XP 系统导致的 IE 6、7 兼容性问题、中文排版的问题、各种建站项目的奇葩需求等。对于这些问题，那些久负盛名的国外框架并没有给出解决方案，不过国内有很多优秀的框架可以帮助我们，譬如淘宝的 Alice、网易的 NEC、渴切等。

第三篇

CSS 实战项目

第 16 章
传统 DIV+CSS 设计的视频网站

随着 4G 和各种移动端智能设备的流行，各种视频网站进入了白热化的用户争夺战。传统视频网站除了要维护自己的 PC 端视频网站外，也相继推出了移动端的 APP，估计每个用户的平板上都下载了多个视频 APP。本章使用的是传统网站的 DIV+CSS 布局，之所以还用这些技术，一是要让读者掌握最基础的 CSS 技术的使用；二是让读者了解传统网站制作流程与使用 CSS 框架的制作流程有什么不同。

本章主要的知识点有：

- 视频网站效果图的拆分。
- 网站的布局规划。
- HTML 框架搭建。
- 传统的 CSS 编写。

16.1 网站的页面效果图分析

本节主要对网站效果图进行分析，包括页面头部和页脚分析、首页主体内容分析和内页主体内容分析。图 16.1 是一个视频网站首页，图 16.2 是视频介绍的页面图。

图 16.1　首页

图 16.2　视频介绍

注意：首页的 HTML 页面名为 index.html，除首页外的其他页面统称为内页。

16.1.1 页面头部和页脚分析

页面的头部如图 16.3 所示，包括网站全局导航、标志/搜索以及网站子导航，分别对应图中①②③。网站全局导航分为左右两栏，分别由若干文字链接组成。标志/搜索包括网站标志和搜索，其中网站标志由背景图片和指向首页的链接组成。搜索是 1 个搜索文本框和 1 个搜索按钮。网站子导航由若干文字链接组成。

页脚如图 16.4 所示，包括网站版权和底部导航，分别对应图中①②。其中网站版权是一段文字，底部导航是若干文字链接。

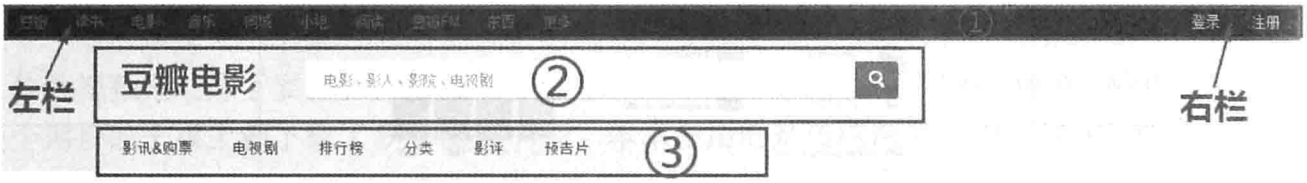


图 16.3 页面头部

图 16.4 页脚

16.1.2 首页主体内容分析

首页的主体内容如图 16.5 所示，包括广告、主要内容以及侧栏，分别对应图中①②③。

在布局上，①位于主体内容的上半部分，②和③位于主体内容的下半部分。其中②位于左栏，③位于右栏。

首页广告是 1 张带有链接的图片。首页主要内容包括正在热映、近期热门、热门豆列以及最受欢迎的影评，共 4 个版块，分别对应图中的 ABCD。侧栏包括影院搜索、影片分类、本周口碑榜、豆瓣视频 TOP250 以及合作联系，共 5 个版块，分别对应图中 EFGHI。

每个版块都是由版块头部和版块内容组成。

- 正在热映的版块头部包括版块标题、面包屑导航以及滚动按钮。版块内容包括 4 个视频海报，其中每个视频海报包括海报图片、视频名、网友评分以及选座购票按钮。
- 近期热门的版块头部包括版块标题和面包屑导航。版块内容包括 5 个视频海报，其中每个视频海报包括海报图片和视频名。
- 热门豆列的版块头部包括版块标题。版块内容包括 4 个网友留言，其中每个留言包括网友头像、留言标题以及留言摘要。
- 最受欢迎的影评的版块头部包括版块标题和面包屑导航。版块内容包括两条视频影评，其中每条视频影评包括视频海报图片、影评标题、视频评分以及影评摘要。
- 影院搜索没有版块头部，只有版块内容。包括 1 个 select 选择列表、1 个搜索文本

框和 1 个搜索按钮。

- 影片分类的版块头部包括版块标题和面包屑导航。版块内容是若干文字链接，这些文字链接是常用的影片分类关键词。
- 本周口碑榜的版块头部包括版块标题和面包屑导航。版块内容是 10 条文字链接，并且每条文字链接前面都有序号，这些序号从 1 到 10 依次排列，这些文字链接是口碑榜前 10 个热门视频的名字。
- 豆瓣视频的版块头部包括版块标题和面包屑导航。版块内容包括 4 条视频海报，其中每条视频海报包括海报图片和视频名。
- 合作联系的版块头部包括版块标题。版块内容包括两条文字链接、1 个联系邮箱和视频客户端下载链接。

图 16.5　首页的主体内容

通过对首页所有版块的分析可以得出，每个版块的版块头部的标题、面包屑导航的 HTML 结构和 CSS 样式都相同。正在热映、近期热门和豆瓣视频 TOP250 这 3 个版块中都有视频海报列表。这些列表的结构和样式都有相似性，比如都可以用 ul 列表结构，每个海报都是向左浮动，每个视频海报都包括海报图片和视频名，每个视频名都有链接等。热门

豆列和最受欢迎的影评这两个版块中的评论都是左右结构，每条评论的左侧都是图片，右侧都包括标题和内容摘要。正在热映和最受欢迎的影评中都有评分星星，评分星星的样式相同。

根据上面的分析，将提炼出两种图文列表的结构和样式。正在热映、近期热门和豆瓣视频 TOP250 这 3 个版块用到了一种，热门豆列和最受欢迎的影评这两个版块用到了另一种。评分的星星也将作为网站的公共部分，供有需要的版块公用。

16.1.3 内页主体内容分析

视频介绍页的主体内容如图 16.6 所示，包括主要内容和侧栏，分别对应图中①②。在布局上，①位于左栏，②位于右栏。

图 16.6 内页的主体内容

主要内容包括视频信息/评分、对视频评分/写评论、剧情简介、预告片和图片、短评和影评 6 个版块，分别对应图中 ABCDEF。侧栏包括购票选座和广告，共两个版块，分别对应图中 GH。

视频信息/评分包括视频名、影片信息和视频评分，其中，影片信息包括视频海报图片和导演/编剧等与影片有关的信息。对视频评分/写评论是网友对视频评价的操作区域，包括

“想看/看过/评分/推荐/写短评/写影评/分享”等诸多网友可以发表观点和看法的操作入口。剧情简介由标题和介绍剧情的几段文字组成。预告片和图片由标题和一组图片列表组成。短评包括标题、短评列表、查看全部短评链接和网友点评链接。影评包括标题、影评列表、查看全部影评链接和网友点评链接。

购票选座包括购票按钮和豆瓣票价。侧栏的广告是 1 张广告图片。

16.2　网站的布局规划

本节主要介绍视频网站的页面布局规划、页面图片切割，以及图片的导出。这些工作是制作本章案例前的必要步骤。

16.2.1　页面布局规划

根据前面对网站效果图的分析，为了后面写出清晰简洁的 HTML 代码，笔者对页面的整体结构进行了提炼，得到了页面的大致布局，图 16.7 是首页和视频介绍页的页面布局图。

图 16.7　页面布局图

16.2.2　切割首页及导出图片

首页需要切割的图片有：网站标志 lg_movie_a12_2.png、搜索文本框和按钮 nav_mv_bg.png、正在热映中的左右滚动按钮 slide_swithc_2.png、选座购票按钮 buyBtn.gif、评分星星 midstars.gif、影院搜索中的展开 select 列表图标 city_select_arrow.png 和搜索按钮 focus_search.png、豆瓣视频客户端背景图 client_bg.gif。

临时图片有：广告图片 ad1.jpg、视频海报图片 p1.jpg、p2.jpg、p3.jpg、p4.jpg、p5.jpg、p6.jpg、p7.jpg、p8.jpg、p9.jpg、p10.jpg、p11.jpg、p12.jpg、p13.jpg、p14.jpg、p15.jpg、网友头像 u1.jpg、u2.jpg、u3.jpg、 u4.jpg。

注意：影院搜索中的 select 列表不是用 HTML 中的 select 标签呈现的。由于 select 标签不能用 CSS 定义高度和展开图标的样式，为了实现效果图中的样子，网页制作者要用 div 和 ul 模拟 select 列表的功能，因此效果图中 select 列表的展开图标需要切出来以便在后面编写样式时使用。

如图 16.8 所示是首页在 Photoshop 中的所有切片。

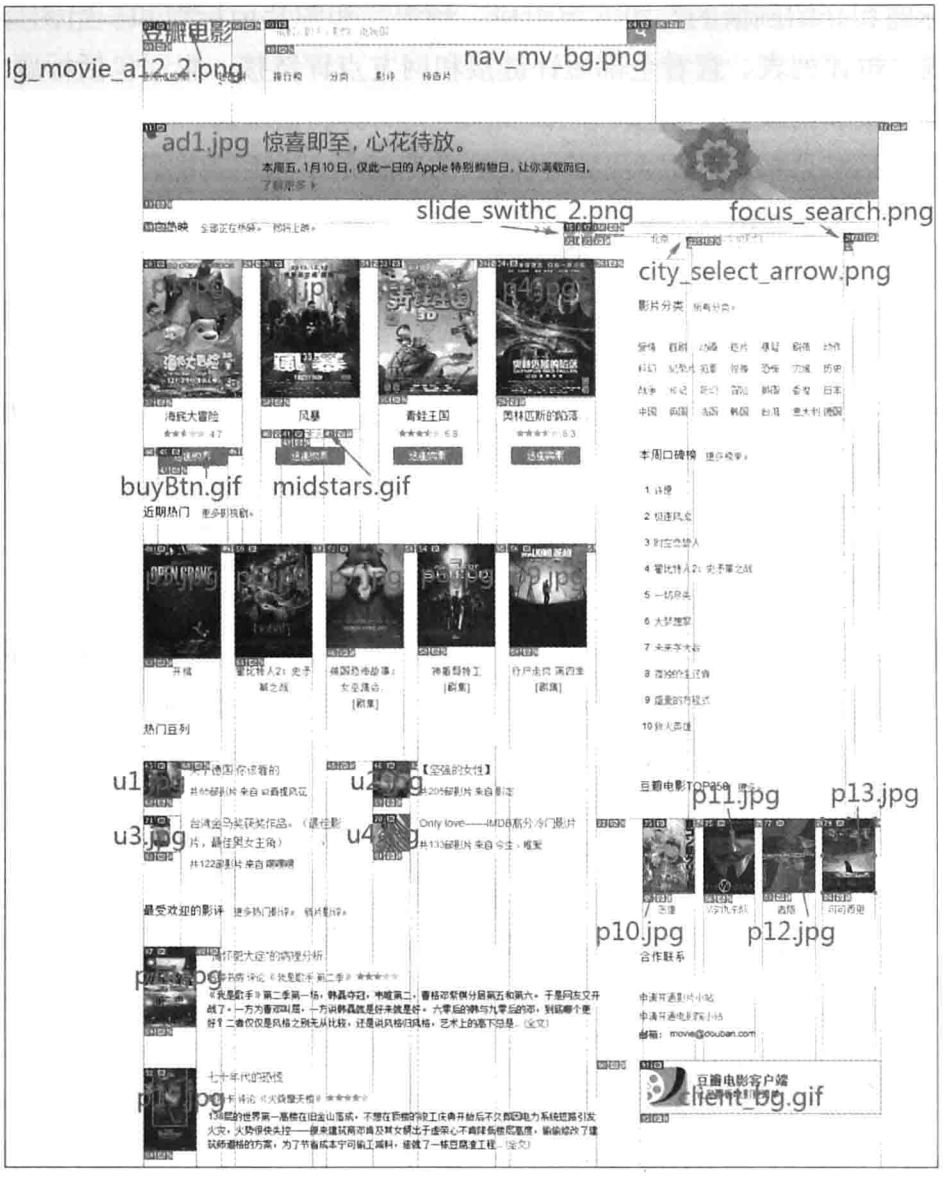


图 16.8　首页在 Photoshop 中的所有切片

16.2.3　切割内页及导出图片

内页需要切割的图片有：评分星星 bigstars.gif 和 smlstars.gif、想看/看过按钮背景 btn.gif、写短评/写影评/分享链接的图标 short-comment.gif、add-review.gif、a1.png、对视频评分星星图标 nst.gif、播放器图标 play_btn.png、影评展开图标 review-expand.png。临时图片有：广告图片 ad2.jpg、视频信息中的海报图片 i_p.jpg、预告片和图片中的影片截图 i_p1.jpg、i_p2.jpg、i_p3.jpg、i_p4.jpg、i_p5.jpg。

如图 16.9 所示是视频介绍页在 Photoshop 中的所有切片：

注意：首页和视频介绍页的评分星星图片 midstars.gif、bigstars.gif 和 smlstars.gif 在网页效果图上展示的只是其中一个级别，由各个级别评分星星组成的完整图片，在实际项目中可由设计师提供，本章案例中的 midstars.gif、bigstars.gif 和 smlstars.gif 图片参见源代码。

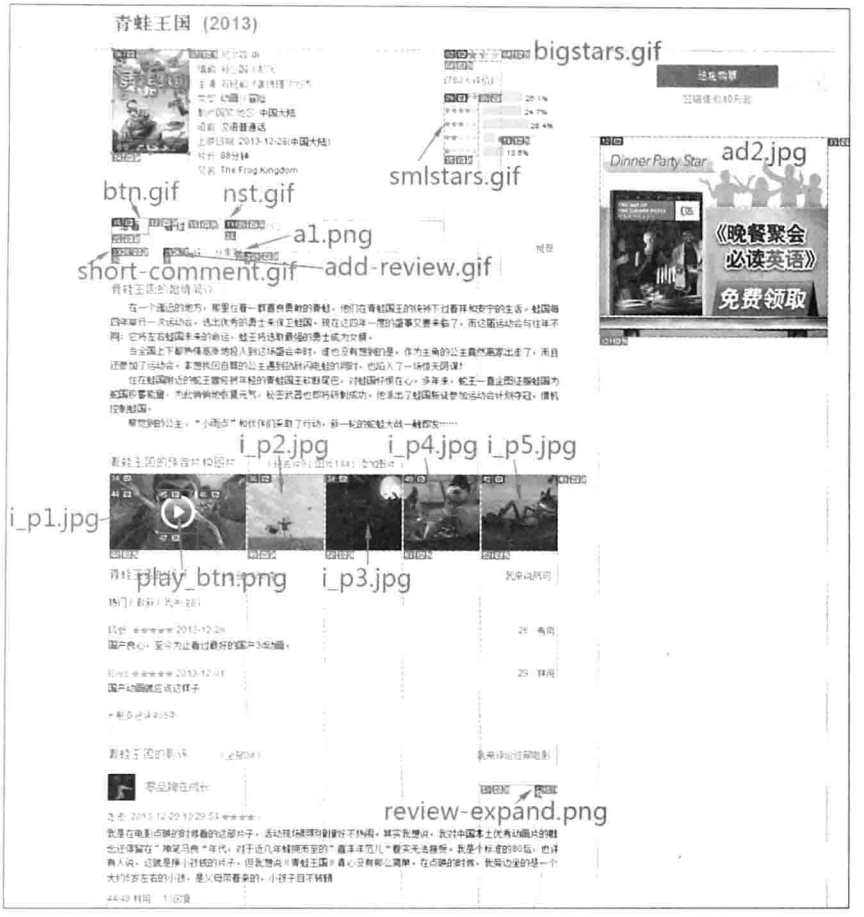


图 16.9　内页在 Photoshop 中的所有切片

16.3　网站 HTML 框架的编写

本节详细讲解页面头部、页脚、页面公共部分、页面框架和每个页面的 HTML 代码的编写。语义和结构良好的 HTML 代码不仅在制作网站时省时省力，更有利于提高网站排名，因此 HTML 的编写虽然简单但很重要。

16.3.1　页面 HTML 框架搭建

首页和内页的 HTML 框架相同，包括 3 部分：页面头部、主体内容和页脚，id 分别为 hd、bd 和 ft。HTML 框架的代码编写如下：

```
<div id="hd"></div>                          <!--页面头部-->
<div id="bd" class="w950 cf"></div>          <!--页面主体内容-->
<div id="ft" class="w950"></div>             <!--页脚-->
```

16.3.2　页面头部和页脚的 HTML

根据前面对首页和视频介绍页面头部的分析，网站导航由 a 链接包含相关导航文字组

成。网站标志上有指向网站首页的链接，因此用 h1 标签包含 a 标签组成。搜索包含在 form 表单标签中，搜索文本框和搜索按钮都用 input 标签，type 分别是 text 和 submit。

编写网站头部的 HTML 代码如下：

```
<div id="hd">
    <div class="nav">
     <div class="login-set fr"><a href="#">登录</a><a href="#">注册</a></div>
     <div class="left-nav"><a href="#">豆瓣</a><a href="#">读书</a><a href="#">视频</a><a
    href="#">音乐</a><a href="#">同城</a><a href="#">小组</a><a href="#">阅读</a><a
    href="#">豆瓣 FM</a><a href="#">东西</a><a href="#" class="more">更多</a></div>
    </div>
    <div class="logo-search">
     <div class="w950">
          <h1><a href="index.html">豆瓣视频</a></h1>
          <form name="form" method="get" action="">
          <input type="text" name="searchBox" class="searchBox" value="视频、影人、影院、电
    视剧" /><input type="submit" name="searchBtn" class="searchBtn" value="搜索" />
          </form>
     </div>
    </div>
    <div class="sub-nav w950"><a href="#">影讯&购票</a><a href="#">电视剧</a><a href="#">
    排行榜</a><a href="#">分类</a><a href="#">影评</a><a href="#">预告片</a></div>
</div>
```

最外层的 div 是首页头部的最外层容器，id 是 hd。第 2 个 div 是网站全局导航，包括右侧的登录/注册和左侧的 10 个链接。class="logo-search"的 div 是标志/搜索，最后一个 div 是网站子导航。

根据前面对网站页脚的分析，底部导航由 8 个 a 标签包含相关导航文字组成，网站版权是一段文字。编写这部分的 HTML 代码如下：

```
<div id="ft" class="w950">
    <span class="fr"><a href="#">关于豆瓣</a><a href="#">在豆瓣工作</a><a href="#">联系我
    们</a><a href="#">免责声明</a><a href="#">帮助中心</a><a href="#">开发者</a><a
    href="#">手机视频</a><a href="#">豆瓣广告</a></span>
    <span>©2005-2014 douban.com all rights reserved</span>
</div>
```

第 1 个 div 是页脚的最外层容器，id 是 ft，第 1 个 span 是底部导航，第 2 个 span 是网站版权信息。

16.3.3 页面公共部分的 HTML

本章案例中，页面的公共部分包括页面头部和页脚，如图 16.10 所示。页面头部和页脚的 HTML 代码在前面已经介绍过了，这里不再赘述。

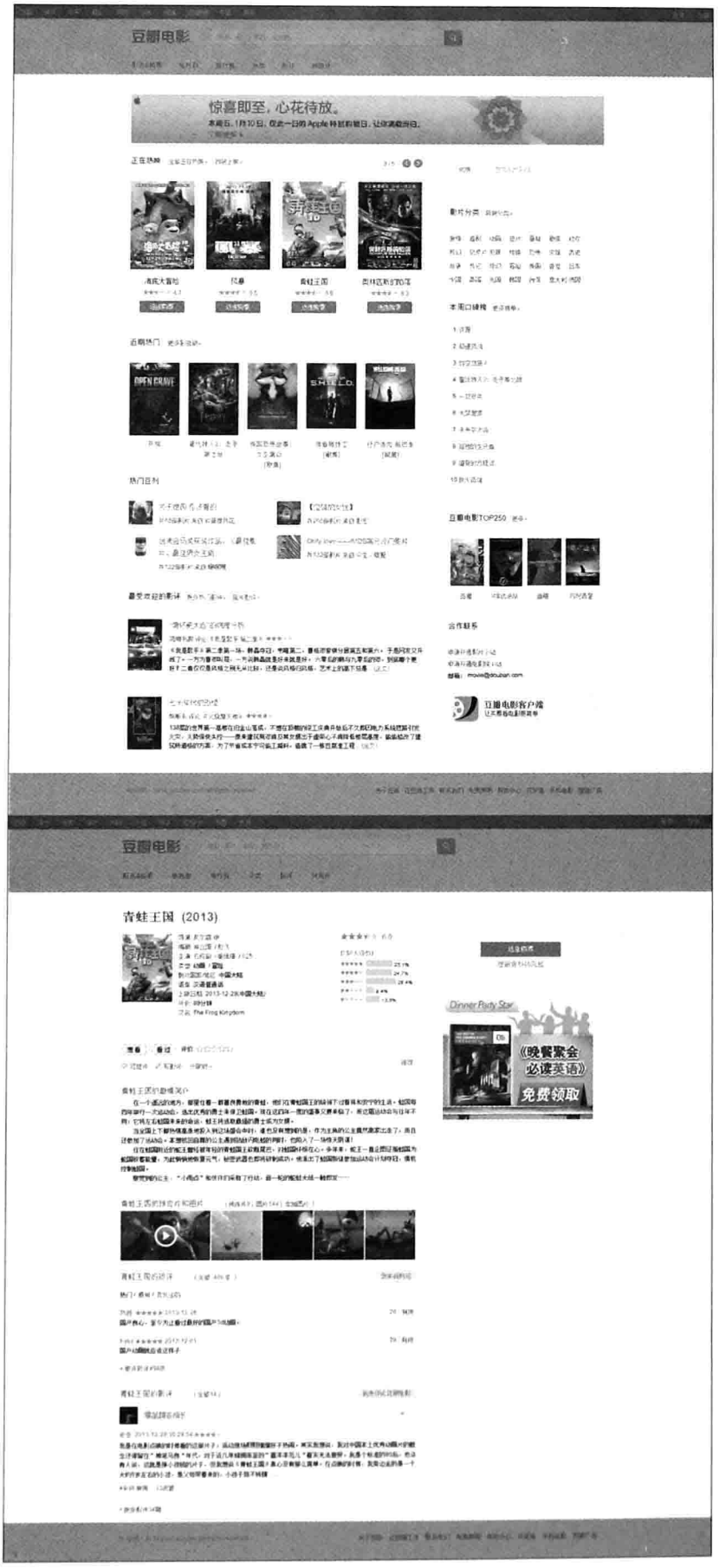


图 16.10　网站所有页面的公共部分

16.3.4 首页主体内容的 HTML

根据前面对首页主体内容的分析，编写首页主体内容的 HTML 代码如下：

```
<div id="bd" class="w950 cf">
<div class="ad"><a href="#"><img src="temp/ad1.jpg" /></a></div>
<div class="con fl">
    <div class="now">
     <div class="hd borderbom">
          <div class="scroll-btn fr"><a href="#" class="next">向后</a><a href="#" class="prev">
    向前</a><span class="num">3 / 5</span></div>
          <h2> 正 在 热 映 <span><a  href="#"> 全 部 正 在 热 映 »</a><a  href="#"> 即 将 上 映
    »</a></span></h2>
     </div>
     <div class="bd">
          <ul class="piclist cf">
          <li>
               <a href="#"><img src="temp/p1.jpg" /></a>
               <h3><a href="#">海底大冒险</a></h3>
               <div class="score"><span class="star star5"></span>4.7</div>
               <input type="button" name="buy" class="buyBtn" value="选座购票" />
          </li>
          ……
          </ul>
     </div>
    </div>
    <div class="hot">
     <div  class="hd  borderbom"><h2> 近 期 热 门 <span><a  href="#"> 更 多 影 视 剧
    »</a></span></h2></div>
     <div class="bd">
          <ul class="piclist cf">
               <li>
                    <a href="#"><img src="temp/p5.jpg" /></a>
                    <h3><a href="#">开馆</a></h3>
               </li>
               ……
          </ul>
     </div>
    </div>
    <div class="user">
     <div class="hd borderbom"><h2>热门豆列</h2></div>
     <div class="bd">
          <ul class="pic-text cf">
               <li>
                    <a href="#"><img src="temp/u1.jpg" /></a>
```

```
                    <h3><a href="#">关于德国 你该看的</a></h3>
                    <p>共 65 部影片 来自 ≦菩提风花</p>
                </li>
                ……
            </ul>
        </div>
    </div>
    <div class="comment">
        <div class="hd borderbom"><h2>最受欢迎的影评<span><a href="#">更多热门影评»</a><a
    href="#">新片影评»</a></span></h2></div>
        <div class="bd">
            <ul class="pic-text cf">
                <li class="borderbom">
                    <a href="#"><img src="temp/p14.jpg" /></a>
                    <h3><a href="#">“情怀肥大症”的病理分析</a></h3>
                    <div class="score">鸣柳书房 评论《我是歌手 第二季》<span class="star
    star6"></span></div>
                    <p>《我是歌手》第二季第一场，韩磊夺冠，韦唯第二，曹格邓紫棋分居第五
    和第六。于是网友又开战了。一方为曹邓叫屈，一方说韩磊就是好来就是好。六零后的韩与九
    零后的邓，到底哪个更好？二者仅仅是风格之别无从比较，还是说风格归风格，艺术上的高下
    总是...<a href="#">（全文）</a></p>
                </li>
                ……
            </ul>
        </div>
    </div>
</div>
<div class="side fr">
    <div class="search-cinema cf">
        <form name="form" method="get" action="">
        <div class="like-select fl">
            <div id="city">北京</div>
            <ul>
                <li>北京</li>
                ……
            </ul>
        </div>
        <div class="search fl"><input type="text" name="searchBox" class="searchBox" value="搜索
    本地影院" /><input type="submit" name="searchBtn" class="searchBtn" value="搜索" /></div>
        </form>
    </div>
    <div class="class">
        <div class="hd borderbom"><h2>影片分类<span><a href="#">所有分类
    »</a></span></h2></div>
        <div class="bd">
```

```
            <ul class="cf">
                <li><a href="#">爱情</a></li>
                ……
            </ul>
        </div>
    </div>
    <div class="list">
        <div class="hd borderbom"><h2> 本 周 口 碑 榜 <span><a href="#"> 更 多 榜 单
    »</a></span></h2></div>
        <div class="bd">
            <ul>
                <li><span>1</span><a href="#">许愿</a></li>
                ……
            </ul>
        </div>
    </div>
    <div class="top">
        <div class="hd borderbom"><h2> 豆 瓣 视 频 TOP250<span><a href="#"> 更 多
    »</a></span></h2></div>
        <div class="bd">
            <ul class="piclist cf">
                <li>
                    <a href="#"><img src="temp/p10.jpg" /></a>
                    <h3><a href="#">恶童</a></h3>
                </li>
                ……
            </ul>
        </div>
    </div>
    <div class="contact">
        <div class="hd borderbom"><h2>合作联系</h2></div>
        <div class="bd">
            <p><a href="#">申请开通影片小站</a></p>
            <p><a href="#">申请开通视频院小站</a></p>
            <p>邮箱：movie@douban.com</p>
            <div class="client">
                <h3><a href="#">豆瓣视频客户端</a></h3>
                <p>让买票看视频更简单</p>
            </div>
        </div>
    </div>
</div>
</div>
```

读者可以参考本书的源代码来了解运行效果，因为代码比较简单，只是布局上的一些

效果，所以此处不再赘述。

16.3.5　内页主体内容的 HTML

根据前面对内页主体内容的分析，编写内页主体内容的 HTML 代码如下：

```
<div id="bd" class="w950 cf">
<div class="con fl">
    <div class="intro-star">
     <h2>青蛙王国<span>(2013)</span></h2>
     <div class="bd cf">
          <div class="cover fl">
               <img src="temp/i_p.jpg" />
               <ul class="cf">
                    <li>导演：<a href="#">尼尔森・申</a></li>
                    ……
                    <li>又名：<span>The Frog Kingdom</span></li>
               </ul>
          </div>
          <div class="film-star fr">
               <div class="total"><span class="bigstar bigstar7"></span>6.8</div>
               <div class="num">(<a href="#">780 人评价</a>)</div>
               <ul>
                    <li><span class="minstar minstar5"></span><span class="scale" style=
"width:25.1%"></span>25.1%</li>
                    <li><span class="minstar minstar4"></span><span class="scale" style=
"width:24.7%"></span>24.7%</li>
                    <li><span class="minstar minstar3"></span><span class="scale" style=
"width:28.4%"></span>28.4%</li>
                    <li><span class="minstar minstar2"></span><span class="scale" style=
"width:8.4%"></span>8.4%</li>
                    <li><span class="minstar minstar1"></span><span class="scale" style=
"width:13.5%"></span>13.5%</li>
               </ul>
          </div>
     </div>
    </div>
    <div class="mystar">
     <div class="recommend"><a href="#">推荐</a></div>
     <div class="op1"><a href="#">想看</a><a href="#">看过</a><span class="givestar">评价:
    <em class="graystar"></em><em class="graystar"></em><em class="graystar"></em><em
    class="graystar"></em><em class="graystar"></em></span></div>
     <div class="op2"><a href="#" class="brief">写短评</a><a href="#" class="long">写影评
    </a><a href="#" class="share">分享到</a></div>
    </div>
```

```
<div class="synopsis">
 <h3>青蛙王国的剧情简介·····</h3>
 <div class="bd">
      <p>在一个遥远的地方，……也陷入了一场惊天阴谋！</p>
      ……
 </div>
</div>
<div class="pic">
 <h3> 青蛙王国的预告片和图片……<span> (<a href="#">预告片 9</a>|<a href="#">图片
144</a>|<a href="#">添加图片</a>)</span></h3>
 <div class="bd">
      <ul class="piclist cf">
           <li><a href="#"><img src="temp/i_p1.jpg" width="178" height="100" /></a><span
class="play"></span></li>
           <li><a href="#"><img src="temp/i_p2.jpg" /></a></li>
           ……
      </ul>
 </div>
</div>
<div class="filmbrief">
 <h3><a href="#" class="isay fr">我来说两句</a>青蛙王国的短评·····<span>(<a href="#">
全部 468 条</a>)</span></h3>
 <div class="bd">
      <dl>
           <dt>热门 / <a href="#">最新</a> / <a href="#">我关注的</a></dt>
           <dd>
                <div class="name"><a href="#"> 风 逝 </a><span class="star star10">
</span><em>2013-12-28</em></div>
                <p>国产良心，至今为止看过最好的国产 3d 动画。</p>
           </dd>
           ……
      </dl>
      <div>> <a href="#" class="more">更多短评 468 条</a></div>
 </div>
</div>
<div class="filmlong">
 <h3><a href="#" class="isay fr">我来评论这部视频</a>青蛙王国的影评·····<span> (<a
href="#">全部 34</a> )</span></h3>
 <div class="bd">
      <div class="comm">
           <dl>
                <dt><a href="#"><img src="temp/u1.jpg" width="36" height="36" /></a>
<h3><a href="#">零品牌在成长</a></h3></dt>
                <dd>
                     <div class="name"><a href="#">走走</a><em>2013-12-29 10:29:54
```

```
    </em><span class="star star8"></span></div>
                        <p>我是在视频点映的时候看的这部片子.....</p>
                        <div class="ope">44/48 有用<a href="#">13 回复</a></div>
                    </dd>
                </dl>
            </div>
            <div>> <a href="#" class="more">更多影评 34 篇</a></div>
        </div>
    </div>
</div>
<div class="side fr">
    <div class="buy">
     <a href="#">选座购票</a>
     <p>豆瓣售价 40 元起</p>
    </div>
    <div class="ad"><a href="#"><img src="temp/ad2.jpg" /></a></div>
</div>
</div>
```

16.3.6　首页 HTML 代码总览

前面对网站首页各个模块的 HTML 代码进行了逐一编写，如图 16.11 所示是这些模块组成的首页的 HTML 框架图，主要说明了层的嵌套关系。

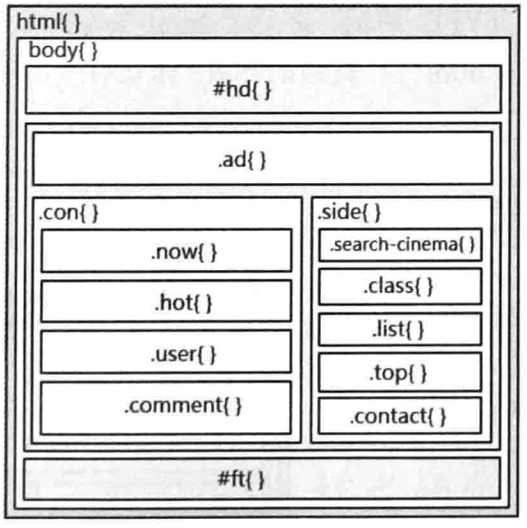


图 16.11　首页 HTML 框架图

在这些 HTML 代码的基础上增加页面的<!DOCTYPE>声明及 html 头部元素，就是首页的完整 HTML 代码。完整的首页 HTML 代码如下：

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

```
    <title>首页</title>
    <link href="css/style.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="hd">……</div>
<div id="bd" class="w950 cf">
    <div class="ad">……</div>
    <div class="con fl">
     <div class="now">……</div>
     <div class="hot">……</div>
     <div class="user">……</div>
     <div class="comment">……</div>
    </div>
    <div class="side fr">
     <div class="search-cinema cf">……</div>
     <div class="class">……</div>
     <div class="list">……</div>
     <div class="top">……</div>
     <div class="contact">……</div>
    </div>
</div>
<div id="ft" class="w950">……</div>
</body>
</html>
```

第 1 行是页面的<!DOCTYPE>声明，之后是 html 头部元素。从第 2 行到最后 1 行是页面的 html 标签，对应图中的 html{}。页面的 body 标签对应图中的 body{}。其他的内容读者可以依次对号入座。

16.3.7 内页 HTML 代码总览

前面对内页各个模块的 HTML 代码进行了逐一编写，图 16.12 是这些模块组成的内页的 HTML 框架图，说明了层的嵌套关系。

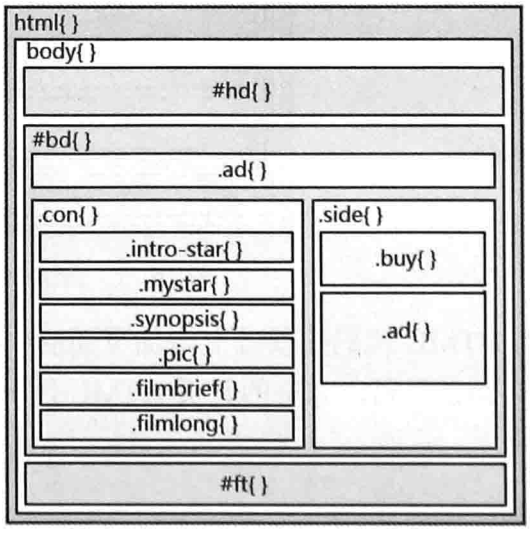


图 16.12　内页的 HTML 框架图

在这些HTML代码的基础上增加页面的<!DOCTYPE>声明及 html 头部元素，就是内页的完整 HTML 代码。完整的内页的 HTML 代码如下：

```
<!DOCTYPE HTML>
<html>
```

```
<head>
    <meta charset="utf-8">
    <title>关于我们</title>
    <link href="css/style.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="hd">……</div>
<div id="bd" class="w950 cf">
    <div class="con fl">
     <div class="intro-star">……</div>
     <div class="mystar">……</div>
     <div class="synopsis">……</div>
     <div class="pic">……</div>
     <div class="filmbrief">……</div>
     <div class="filmlong">……</div>
    </div>
    <div class="side fr">
     <div class="buy">……</div>
     <div class="ad">……</div>
    </div>
</div>
<div id="ft" class="w950">……</div>
</body>
<!--[if IE 6]>
<script src="js/DD_belatedPNG_0.0.8a-min.js"></script>
<script>
    DD_belatedPNG.fix('*');
</script>
<![endif]-->
</html>
```

第 1 行是页面的<!DOCTYPE>声明，接着是 html 头部元素。第 2 行到最后 1 行是页面的 html 标签，对应图中的 html{}。页面的 body 标签对应图中的 body{}。其他的对应关系读者可以根据从上到下、从左到右的原则依次分析。

16.4　网站 CSS 样式的编写

本节主要讲解视频网站的 CSS，包括页面头部和页脚、首页和内页的 CSS。

16.4.1　页面公共部分的 CSS

页面公共部分包括 CSS 重置、页面中的公用字体、字体颜色、公用模块的样式，以及

页面头部和页脚。

页面头部和页脚的 CSS 将在 16.4.3 节中编写。CSS 重置代码、页面公用样式的 CSS 代码编写如下：

```
/*css 重置代码*/
body,div,dl,dt,dd,ul,ol,li,h1,h2,h3,h4,h5,h6,pre,code,form,fieldset,legend,input,button,textarea,
p,blockquote,th,td{margin:0; padding:0;}/*以上元素的内外边距都设置为 0*/
……
*.cf{zoom:1;} /* IE 6/7 浏览器 (触发 hasLayout) */
/*公用 CSS 代码 开始*/
/*定义页面中用到的通用样式和标签*/
body{font-family:Helvetica,Arial,sans-serif;font-size:12px;color:#121212;line-height:20px;}
a{color:#3576AB;}
a:hover{color:#3576AB;text-decoration:none;}
.fl{float:left;display:inline;}
.fr{float:right;display:inline;}
.w950{width:950px;margin:0 auto;}
.borderbom{border-bottom:1px solid #EAEAEA;}
input,label{vertical-align:middle;}
/*主体内容最外层容器的样式 */
#bd{padding:40px 0 0;}
/*主体内容中的主要内容和侧栏的样式*/
#bd .con{width:590px;overflow:hidden;}
#bd .side{width:310px;overflow:hidden;}
/*页面中广告的公共样式*/
#bd .ad{margin:0 0 10px;}
#bd .ad img{display:block;}
/*页面中每个版块的头部的公共样式*/
.hd{height:50px;line-height:50px;}
.hd h2{font-size:14px;font-weight:normal;}
.hd span{margin:0 0 0 15px;}
.hd span a{margin:0 15px 0 0;font-size:12px;}
/*页面中每个版块的主要内容的公共样式*/
.bd{padding:18px 0;}
/*网站所有评分中星星的样式*/
.star{background:url("../images/midstars.gif") no-repeat 0 0;display:inline-block;
*zoom:1;*display:inline;width:55px;height:11px;overflow:hidden;margin:0 5px;}
.star10{background-position:0 0}
.star9{background-position:0 -11px}
.star8{background-position:0 -22px}
.star7{background-position:0 -33px}
.star6{background-position:0 -44px}
.star5{background-position:0 -55px}
.star4{background-position:0 -66px}
.star3{background-position:0 -77px}
```

```
.star2{background-position:0 -88px}
.star1{background-position:0 -99px}
.star0{background-position:0 -110px}
.bigstar{background:url("../images/bigstars.gif") no-repeat 0 0;display:inline-block;
*zoom:1;*display:inline;width:75px;height:14px;overflow:hidden;margin:0 5px;}
.bigstar10{background-position:0 0}
.bigstar9{background-position:0 -14px}
.bigstar8{background-position:0 -28px}
.bigstar7{background-position:0 -42px}
.bigstar6{background-position:0 -56px}
.bigstar5{background-position:0 -70px}
.bigstar4{background-position:0 -84px}
.bigstar3{background-position:0 -98px}
.bigstar2{background-position:0 -112px}
.bigstar1{background-position:0 -136px}
.bigstar0{background-position:0 -140px}
.minstar{background:url("../images/smlstars.gif") no-repeat 0 0;display:inline-block;
*zoom:1;*display:inline;width:45px;height:9px;overflow:hidden;margin:0 5px;}
.minstar5{background-position:0 0}
.minstar4{background-position:0 -9px}
.minstar3{background-position:0 -18px}
.minstar2{background-position:0 -27px}
.minstar1{background-position:0 -36px}
.minstar0{background-position:0 -45px}
/*网站中两种图文列表的公共样式*/
.piclist li{float:left;display:inline;text-align:center;}
.piclist li img{display:block;}
.piclist li h3{font-weight:normal;margin:12px 0 10px;font-size:12px;}
.piclist li .score{color:#F54D06;text-align:center;}
.pic-text li{float:left;}
.pic-text li img{float:left;display:inline;margin:0 10px 0 0;}
.pic-text li h3{font-size:14px;margin:0 0 10px 0;font-weight:normal;}
.pic-text li .score{color:#666;}
/*播放器按钮的样式，定义了播放器按钮的背景图*/
.play{background:url("../images/play_btn.png") no-repeat 50% 50%;}
/*公用 CSS 代码 结束*/
```

其中，前 5 行是 CSS 重置代码，从第 6 行开始是公用 CSS 代码。

16.4.2　页面框架的 CSS

前面分析了页面的布局图并且编写了页面框架的 HTML 代码，根据这两部分编写页面框架的 CSS 代码如下：

```
#hd{background:#EFF2F5;}
#bd{padding:40px 0 0;}
```

```
#ft{color:#999;border-top:1px dashed #ddd;line-height:30px;padding:0 0 30px;
font-family:Arial;}
```

第 1 行是页面头部最外层容器的样式，第 2 行是页面主体内容最外层容器的样式，第 3 行是页脚最外层容器的样式。

16.4.3 页面头部和页脚的 CSS

前面分析了页面头部并且编写了这部分的 HTML 代码，页面头部的 CSS 代码如下：

```
#hd{……}
#hd .nav{background:#535751;height:28px;line-height:28px;}
#hd .nav a{color:#D6D6D4;margin:0 10px;}
#hd .logo-search{border-bottom:1px solid #E3EBEC;padding:20px 0;}
#hd .logo-search h1{background:url("../images/lg_movie_a12_2.png") no-repeat left top;
width:115px;height:27px;overflow:hidden;float:left;margin:0 42px 0 0;}
#hd .logo-search h1 a{display:block;width:100%;height:100%;text-indent:-999em;}
#hd .logo-search input{border:0;}
#hd .logo-search .searchBox{background:url("../images/nav_mv_bg.png") no-repeat left top;
width:470px;height:31px;line-height:30px;padding:0 10px 2px;font-size:12px;color:#BABABA;}
#hd .logo-search .searchBtn{background:url("../images/nav_mv_bg.png") no-repeat 0 -40px;
width:37px;height:33px; text-indent:-999em;cursor:pointer;}
#hd .sub-nav{height:38px;line-height:38px;}
#hd .sub-nav a{margin:0 36px 0 0;}
```

第 1 行是页面头部的最外层容器的样式。第 2~3 行是全局导航的样式。中间部分分别是标志/搜索的最外层容器的样式、网站标志的样式、搜索文本框和搜索按钮的样式。最后两行是网站子导航的样式。

根据前面对页脚和页脚 HTML 代码的分析，编写页脚 CSS 代码如下：

```
#ft{……}
#ft a{margin:0 5px;}
```

第 1 行是页脚最外层容器的样式，第 2 行是页脚中所有链接的样式。

16.4.4 首页主体内容的 CSS

根据对首页主体内容和这部分HTML代码的分析，编写首页主体内容的CSS代码如下：

```
/*“选座购票”按钮的样式*/
.buyBtn{background:url("../images/buyBtn.gif") no-repeat 0 0;width:90px;height:24px;
border:0;color:#fff;font-size:12px;text-align:center;line-height:24px;margin:10px 0 0;cursor:pointer;}
/*正在热映中的滚动按钮的样式*/
.now .scroll-btn{width:100px;height:18px;line-height:18px;overflow:hidden;margin:20px 0 0;}
.now .scroll-btn a{float:right;background:url("../images/slide_swithc_2.png") no-repeat left top;
width:18px;height:18px; margin:0 0 0 5px; text-indent:-999em;}
.now .scroll-btn .prev{background-position:0 0}
.now .scroll-btn .prev:hover{background-position:0 -18px}
```

```
.now .scroll-btn .next{background-position:-18px 0}
.now .scroll-btn .next:hover{background-position:-18px -18px}
.now .scroll-btn .num{margin:0 10px 0 0;color:#666;float:right;}
/*正在热映版块中视频海报列表的样式*/
.now .piclist{width:700px;}
.now .piclist li{margin:0 24px 0 0;width:128px;}
.now .piclist li img{width:128px;height:180px;}
.now .piclist li h3{font-size:14px;}
.now .piclist li h3 a{color:#333;}
/*近期热门版块中视频海报列表的样式*/
.hot .piclist{width:700px;}
.hot .piclist li{margin:0 18px 0 0;width:100px;}
.hot .piclist li img{width:100px;height:148px;}
/*热门豆列版块中用户评论列表的样式*/
.user .pic-text li{width:295px;margin:0 0 20px 0;line-height:24px;}
.user .pic-text li h3,.user .pic-text li p{margin:0 0 0 60px;}
.user .pic-text li p{color:#666;}
/*最受欢迎的影评的样式*/
.comment .bd{padding-top:0;}
.comment .pic-text li{padding:25px 0;}
/*影院搜索版块的样式*/
.search-cinema{background:#F5F3D2;padding:10px;}
.search-cinema .like-select{width:67px;margin:0 5px 0 0;color:#666;line-height:24px;}
.search-cinema .like-select #city{width:60px;height:24px;
background:url("../images/city_select_arrow.png") no-repeat 50px center #fff;
border:1px solid #E0E0E0;padding:0 0 0 5px;cursor:pointer;}
.search-cinema .like-select ul{padding:0 5px;background:#fff;border:1px solid #E0E0E0;
width:55px;display:none;}
.search-cinema .search{float:left;border:1px solid #E0E0E0;width:202px;height:24px;
background:#fff;}
.search-cinema .search .searchBox{border:0;font-size:12px;color:#BFBFBF;width:170px;
height:22px;line-height:22px;padding:0 5px;}
.search-cinema .search .searchBtn{border:0; text-indent:-999em;
background:url("../images/focus_search.png") -14px 0;width:12px;height:12px;
cursor:pointer;margin:0 10px 0 0;}
/*影片分类中每个关键词所在的 li 标签的样式*/
.class ul li{float:left;width:40px;line-height:28px;}
/*本周口碑榜的样式*/
.list ul li{border-bottom:1px solid #EAEAEA;height:33px;line-height:33px;}
.list ul li span{display:inline-block;*zoom:1;*display:inline;width:20px;text-align:center;}
#bd .list .bd{padding-top:5px;}
/*豆瓣视频 TOP250 版块的样式*/
.top .piclist{width:700px;}
.top .piclist li{margin:0 11px 0 0;width:68px;}
.top .piclist li img{width:68px;height:98px;}
```

```
/*合作联系版块的样式*/
.contact{line-height:24px;}
.contact .client{background:url("../images/client_bg.gif") no-repeat 0 0;width:290px;
height:40px;padding:15px 10px;margin:30px 0 0;}
.contact .client h3{font-size:16px;margin:0 0 0 60px;}
.contact .client h3 a{color:#214E92;}
.contact .client p{color:#4D4D4D;margin:0 0 0 60px;}
```

16.4.5 内页主体内容的 CSS

根据对内页主体内容和这部分HTML代码的分析，编写内页主体内容的CSS代码如下：

```
/*视频信息/评分的样式*/
intro-star h2{color:#494949;font-size:25px;line-height:36px;}
intro-star h2 span{color:#888;}
intro-star .cover{width:310px;}
intro-star .cover img{float:left;}
intro-star .cover ul{margin:0 0 0 112px;color:#666;}
intro-star .cover ul li span{color:#111;}
intro-star .film-star{width:150px;}
intro-star .film-star .total{font-size:14px;color:#F54D06;}
intro-star .film-star ul li{font-size:10px;height:14px;line-height:14px;margin:0 0 4px 0;}
intro-star .film-star ul .scale{background:#F5CBAD;
display:inline-block;*zoom:1;*display:inline;height:14px;margin:0 3px 0 0;}
.intro-star .film-star .num{margin:10px 0 5px 5px;}
/*对视频评分/写评论的样式*/
.mystar{margin:25px 0 0;}
.mystar .recommend{width:40px;height:19px;text-align:center;line-height:19px;float:right;
background:#F2F8F2;border:1px solid #E3F1ED;}
.mystar .recommend a{color:#4F946E;}
.mystar .op1 a{background:url("../images/btn.gif") no-repeat 0 0;
display:inline-block;*zoom:1;*display:inline;width:49px;height:22px;line-height:21px; margin:0 10px 0 0;
    text-align:center;color:#000;}
.mystar .op1 .givestar{vertical-align:middle;}
.graystar{background:url("../images/nst.gif") no-repeat 0 center;
display:inline-block;*zoom:1;*display:inline;width:15px;height:15px;vertical-align:middle;cursor:poi
    nter;}
.mystar .op2{margin:5px 0 30px 0;}
.mystar .op2 a{margin:0 10px 0 0;padding:0 0 0 17px;}
.mystar .op2 .brief{background:url("../images/short-comment.gif") no-repeat 0 50%;}
.mystar .op2 .long{background:url("../images/add-review.gif") no-repeat 0 50%;}
.mystar .op2 .share{background:url("../images/a1.png") no-repeat 100% 50%;padding:0 10px 0 0;}
/*剧情简介的样式*/
.synopsis h3{font-weight:normal;color:#007722;font-size:15px;}
.synopsis .bd{padding:5px 0 30px;}
.synopsis .bd p{text-indent:2em;}
```

```
/*预告片和图片的样式*/
.pic h3{font-weight:normal;color:#007722;font-size:15px;}
.pic h3 span{font-size:12px;color:#666;}
.pic h3 span a{margin:0 5px;}
.pic .bd{padding:5px 0 30px;}
.pic .bd p{text-indent:2em;}
.pic .bd .piclist li{position:relative;margin:0 2px 0 0;}
.pic .bd .piclist .play{width:178px;height:100px;position:absolute;top:0;left:0;cursor:pointer;}
/*短评的样式*/
.filmbrief h3{font-weight:normal;color:#007722;font-size:15px;}
.filmbrief h3 .isay{height:26px;text-align:center;line-height:26px;background:#FAE9DA;
color:#CA6445;padding:0 10px;font-size:12px;}
.filmbrief h3 span{font-size:12px;color:#666;}
.filmbrief h3 span a{margin:0 5px;}
.filmbrief .bd{padding:5px 0 30px;}
.filmbrief .bd dl{margin:0 0 8px 0;}
.filmbrief .bd dd,.filmbrief .bd dt{border-bottom:1px dashed #ddd;padding:8px 0;}
.filmbrief .name em{color:#666;}
/*影评的样式*/
.filmlong h3{font-weight:normal;color:#007722;font-size:15px;}
.filmlong h3 .isay{height:26px;text-align:center;line-height:26px;background:#FAE9DA;
color:#CA6445;padding:0 10px;font-size:12px;}
.filmlong h3 span{font-size:12px;color:#666;}
.filmlong h3 span a{margin:0 5px;}
.filmlong .bd{padding:10px 0 30px;}
.filmlong .bd .comm dt{background:#F0F3F5;height:36px;line-height:36px;}
.filmlong .bd .comm dt img{float:left;margin:0 15px 0 0;display:inline;}
.filmlong .bd .comm dt h3{background:url("../images/review-expand.png") no-repeat right center;
    margin:0 20px 0 0;}
.filmlong .bd .comm dd{padding:10px 0 0;}
.filmlong .name em{color:#666;margin:0 0 0 5px;}
.filmlong .ope{padding:10px 0 20px;color:#666;}
.filmlong .ope a{margin:0 0 0 20px;}
/*选座购票的样式*/
.buy{background:#F0F3F5;padding:20px 0 10px;margin:0 0 40px 0;}
.buy a{display:block;background:#268DCD;color:#fff;width:160px;height:30px;
line-height:30px;text-align:center;margin:0 auto;}
.buy p{text-align:center;color:#999;line-height:30px;}
```

最后 3 行是选座购票的样式，其中倒数第 2 行通过 display:block 将 a 标签由行内元素转换成块元素，并通过 width:160px;height:30px 设置了 a 标签的宽度和高度。

16.4.6　网站 CSS 代码总览

前面讲解了页面头部、页脚、页面主体内容、CSS 重置和页面公用的 CSS 代码，这些代码共同组成了网站页面的完整 CSS 代码，如下所示：

```
@charset "utf-8";
/*css reset*/
……
/*global css*/
……
/*module css*/
/*index.html*/
.buyBtn {……}
.now .scroll-btn {……}
……
/*page.html*/
.intro-star h2 {……}
.intro-star h2 span {……}
……
```

注意：省略的代码在前面每个小节中都有讲解。

16.5 小结

本章没有使用 CSS 3 或 HTML 5 中的新内容，完全是按照传统的 HTML+DIV+CSS 的网站设计方式来完成一个视频网站的主要布局。完成这么一个网站设计，读者可能需要花费一周时间，从时间成本来说，肯定不如使用 Bootstrap 等框架来得高效，但却能让我们了解 CSS 设计最基本的原理和方法。

第 17 章
使用 HTML 5+CSS 3 开发搜房网

苹果不支持 Flash，促进了 HTML 5 的发展，目前很多 APP 应用和跨设备的网站开始逐渐使用 HTML 5 来开发，包括淘宝、大众点评等个性化网站中也有很多网页应用了 HTML 5 的一些新特性。HTML 5 本身提供了大量语义化的标签可以实现网站的布局，借助于 CSS 3 的新特性，可以完成很多原本需要额外处理的特效。

本章主要知识点包括：

- 学习 HTML 5。
- 掌握 CSS 3 新特性在实际网站开发中的应用。
- 了解一个 HTML 5+CSS 3 网站的设计流程。
- 利用 jQuery 提高开发效率。

17.1 网站前期策划

本章将演示如何使用 HTML 5+CSS 3 实现一个与房产租售有关的搜房网，该网站包含一些房屋售卖和出租的信息，同时发布一些联系信息和房屋买卖方面的知识。本章将大量应用 HTML 5 中的语义性标签来实现网站的布局，同时使用 CSS 3 和 jQuery 来为网站添加有趣的特效。

17.1.1 理解 HTML 5 的语义性元素

如果使用 HTML 5 提供的布局元素，整个文档代码将具有更好的可理解性，便于被其他的应用程序分析和识别。

HTML 5 提供了 3 种类型的语义性元素，分别如下。

- 结构性语义元素：用来处理网页的语义性布局，比如定义网页的页眉、页脚、导航、内容显示部分等，通过 CSS 的样式控制，可以达到布局的效果。
- 语义性块级元素：用来定义具有块级（block）效果的语义元素，比如显示图示、边栏以及会话信息。
- 语义性内联元素：用来定义具有内联（inline）效果的语义元素，比如度量衡、进度条等。

结构化语义元素用来定义网页的结构，这与使用 DIV 进行布局非常类似，相关的 HTML 5 元素如下所示。

- header 元素：在页面上显示页眉信息，可以是标题栏、LOGO 栏等。
- footer 元素：在页面上显示页脚信息，比如电子邮件签名、版权信息等。
- nav 元素：在网页上显示一组链接。
- section 元素：显示网页上的一个块，有自己的标题、页脚以及内容区。
- article 元素：显示页面的主体内容，比如博客文章、新闻内容等。

语义性的块元素是指具有块级结构的 HTML 5 页面元素，在 HTML 5 中具有语义性的块元素如下。

- aside：与主内容无关但是又可以独立的块内容，一般用于显示广告、引用或侧边栏等。
- figure：用来组合多个元素，一般用来在网页上显示图像以及其标题，它用来表示网页上的一块独立的内容，即便将其移除后也不会对网页上其他内容造成影响。
- dialog：用来显示谈话或会话信息，在内部使用 dt 和 dd 表示对话的内容。

块级元素总是会占据整行来显示，而内联元素则不会占据整行，通常用来在块级元素内部显示一些强调或特殊意义的信息。在 HTML 5 中提供了如下的几个具有语义性的内联元素。

- m：定义一段需要突出显示的文本内容。
- mark：定义需要进行标记的文本。
- time：用来显示日期时间，该元素有一个 datetime 属性用来标识能被电脑所识别的时间。
- meter：表达特定范围内的数值。可用于薪水、百分比、分数等。
- progress：用来表示进度。

相较于 DIV+CSS 布局，页面上可以见到的基本上就是 元素，而使用 HTML 5 的语义性元素后，就可以看到页面内容的语义性提示，例如使用 aside 和 article 构建非主要内容的提示：

```
<aside class="grid_6">
  <div class="prefix_1">
  <article>
      <div class="box">
        <h2>如何找到我们</h2>
        <h3><a href="#">我们的工作理念</a></h3>
        <p>自以为聪明的人往往是没有上场的。世界上最聪明的人是最老实的人。
                        因为只有老实人才能经得起事实和历史的考验。</p>
        <a href="#" class="button">Read More</a>
      </div>
      <!-- /.box -->
    </article>
</aside>
```

aside 在 HTML 5 中用来装载非正文类的内容，因此对于侧边栏或与主要内容相关的部分，可以使用 aside 进行声明。在 aside 内部放置了 article 标签，article 标签用来定义文章，可以是一篇新的文章或来自博客的文本内容，或其他的外部内容。article 标签的内容独立于文档的其余部分。在 article 标签的内部，可以看到各种格式化用的 HTML 标签，这些语义性标签和 CSS 样式搭配使用，也可以创建出非常有创意的网页。

17.1.2　搜房网网站结构

搜房网是笔者为了研究 HTML 5+CSS 3 使用一套模板搭建的一套简单的网站，重点在于演示 HTML 5 中新的语义性标签在布局中的应用。目前房产租售一般是使用动态网站的形式便于维护和推广，但是对于一些高端的房产租售，很多公司仍然会搭建专业的网站平台，以供潜在的客户进行查阅。搜房网整个网站由 7 个页面组成，其结构如图 17.1 所示。

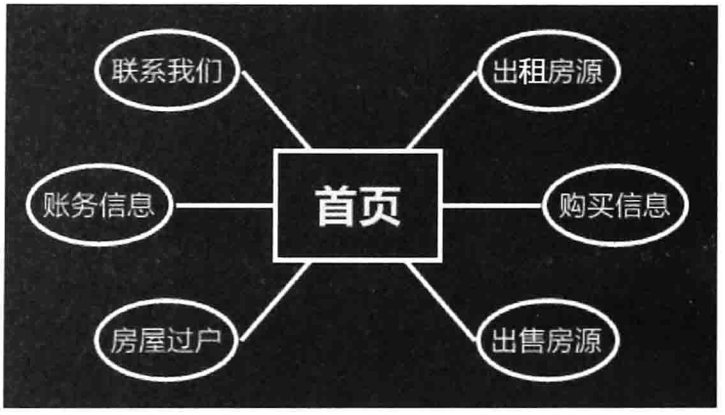


图 17.1　搜房网页面结构

该网站使用 Dreamweaver 作为开发工具，大量应用 CSS 样式来管理页面的结构，在页面的布局上面，没有使用传统的 DIV+CSS 布局方式，而是使用大量 HTML 5 的语义标签。在内容布局上，该网站提供了大的分类样式，但是并不包含具体的页面内容，希望通过对该网站的学习，读者可以了解到如何使用 HTML 5+CSS 3 来创建自己的网站。

17.1.3　搜房网整站预览

网站的首页包含一些热门推荐的房源信息，同时提供最新装修的房产列表，网站定位于展示一些高端的房产资讯，比如一些国外的热门别墅之类的。

网站的首页如图 17.2 所示。可以看到，首页包含 Logo、网站导航栏、使用圆角矩形的图片宣传栏，单击右侧不同的房源会在图像显示区显示不同的图像。热门房产区域下面是网站的介绍信息，它包含欢迎信息以及最新的房源列表，如图 17.3 所示。

在出售房源部分，保留了与图 17.2 相似的宣传页结构，在页面底部显示了与出售房源相关的一些信息，比如贷款事项和优质房源简介等，如图 17.4 所示。

在购买信息页面，包含了关于房产购买的一些事项，以及购买的链接，如图 17.5 所示。出租房源、房屋过户和账务信息基本上保持了与图 17.4 和图 17.5 所示的页面布局，对于联

系我们页面，包含了一个 HTML 表单允许用户留言，同时它还包含了网站的联系方式等信息，其布局结构如图 17.6 所示。

图 17.2　房产网站首页

图 17.3　首页欢迎和最新房产信息

图 17.4　出售房源页面

图 17.5　房屋购买需求页面

图 17.6　联系我们页面

很明显，这些页面的结构基本上风格统一。都是使用黑色的底色、白色的文字和绿色的修饰色，这使得页面高贵大方，又适宜阅读，同时页面上的图片与背景形成了鲜明的对比，使得图片容易让人印象深刻。接下来将分析如何使用 HTML 5 和 CSS 3 来实现这种类型的网页。

17.2　搜房网的首页设计

网站的首页是整个网站的门户，从技术上来说，完成了首页，基本上就算完成了静态网站搭建一半的工作，因为网站首页要实现网站的导航栏、搭建页面的布局结构、准备网站的图片、创建整个网站将要使用的 CSS 样式等，后续的页面除了进行局部的更改外，可以复用首页实现的多种效果，以便与首页保持统一的页面风格。

17.2.1　首页的布局

整个网站在结构上分为 3 个部分。

- 页头部分：包含网站的 Logo、导航栏和宣传页区域。
- 内容部分：包含首页的内容介绍，比如显示欢迎信息和热门的房源介绍。
- 页脚部分：包含网站的链接和网站的版权信息，如果进行了网站备案则包含网站的备案信息。

网站的这 3 大部分通过 HTML 5 的语义标签 header、section 和 footer 进行区分，它们都属于结构化语义元素，用来定义网站的结构，下面分别介绍这 3 个部分的组成结构。

1．页头部分

页面头部包含了导航栏、Logo 和宣传广告栏，在<header>标签内部使用了 div 进行布局，导航栏部分使用 nav 语义性标签表示，它标示出该标签内的内容属于页面的导航区域，header 部分的代码实现如下：

```
<!-- 页面头部分的内容 -->
<header>
  <div class="container_16">
      <!--网站 Logo 部分-->
   <div class="logo">
     <h1><a href="index.html"><strong>房产网站</strong></a></h1>
   </div>
     <!--网站的导航栏-->
    <nav>
      <!—导航 ul 区域，将在导航部分详细讨论-->
    </nav>
    <!-- 淡入淡出的图片显示区，使用 jQuery 控制 -->
    <div id="faded">
      <div class="rap">
        <a href="#"><img src="images/big-img1.jpg" alt="" width="571" height="398"></a>
        <a href="#"><img src="images/big-img2.jpg" alt="" width="571" height="398"></a>
        <a href="#"><img src="images/big-img3.jpg" alt="" width="571" height="398"></a>
      </div>
    <!-- 宣传文字区域，单击可以淡入淡出图片区图片的显示-->
      <ul class="pagination">
        <li>
            <a href="#" rel="0">
            <img src="images/f_thumb1.png" alt="">
            <span class="left">
              …
            </span>
            <span class="right">
              <!—省略了文字内容-->
          </span>
          </a>
        </li>
        <li>
            <a href="#" rel="1">
            <img src="images/f_thumb2.png" alt="">
            <span class="left">
              <!—省略了文字内容-->
            </span>
            <span class="right">
             …
            </span>
```

```
            </a>
          </li>
          <li>
              <a href="#" rel="2">
              <img src="images/f_thumb3.png" alt="">
              <span class="left">
                <!—省略了文字内容-->
              </span>
              <span class="right">
                <!—省略了文字内容-->
              </span>
            </a>
          </li>
        </ul>
        <img src="images/extra-banner.png" alt="" class="extra-banner">
    </div>
    </div>
</header>
```

在代码中，位于<header>标签下面的是一个 div 元素，它显示了页面的 Logo 信息。接下来使用了<nav>标签来创建导航栏，导航栏使用<ul>和<li>标签来实现，通过 CSS 来实现水平显示的导航效果。位于导航栏后面的是分页的宣传内容展示区，首先在 标签内放置了 3 幅用于淡入淡出的图片，然后放置了<ul>和<li>元素对文字内容进行布局。

可以看到，因为有了 HTML 5 的语义性元素的布局，整个代码变得清晰易懂，容易分析，相较于传统的 DIV+CSS 的布局设计来说，更容易使人阅读，并且也便于第三方的软件工具对页面的布局进行分析。

2．内容部分

内容部分使用 section 标签，section 显示网页上的一个块，有自己的标题、页脚以及内容区，从布局结构上来说它是一个容器，在容器内部可以包含多个 article 或其他的语义性结构元素。在网站的内容区使用 section 和 article 构建主页的内容显示，以 section 作为主要容器，内部通过多个 article 来划分不同的文章内容，代码如下：

```
<section id="content">
  <div class="container_16">
   <div class="clearfix">
        <section id="mainContent" class="grid_10">
          <article>
              <!--省略部分文字内容-->
              <a href="#" class="button">了解更多</a>
        </article>
        <article class="last">
            <h2>最新装修的房屋列表</h2>
             <ul class="img-list clearfix">
            <li><a href="#"><img src="images/thumb1.jpg" alt=""></a></li>
```

```
                <!--省略部分相似内容-->
            </ul>
            <a href="#" class="button">了解更多</a>
          </article>
        </section>
        <aside class="grid_6">
        <div class="prefix_1">
            <article>
            <!-- .box -->
            <div class="box">
              <h2>5步成为优秀的房产经纪人</h2>
              <dl class="accordion">
                <dt><img src="images/icon1.gif" alt=""><a href="#">了解用户需求</a></dt>
                <!--省略部分相似内容-->
              </dl>
            </div>
            <!-- /.box -->
          </article>
          <article class="last">
            <h2>我们的办公地点</h2>
            <p><img src="images/map.jpg" alt=""></p>
            <div class="wrapper">
                <ul class="list1 grid_3 alpha">
                <li><a href="#">北美</a></li>
                <li><a href="#">欧州</a></li>
                <li><a href="#">拉丁美州</a></li>
              </ul>
              <ul class="list1 grid_2 omega">
                <li><a href="#">亚州</a></li>
                <li><a href="#">澳州</a></li>
                <li><a href="#">南美州</a></li>
              </ul>
            </div>
          </article>
        </div>
      </aside>
    </div>
  </div>
</section>
```

整个内容结构部分被一个<section>标签包含，在它内部，又包含了<section>标签指示主要内容区和辅助的<aside>区指示侧边栏内容区，在主要内容部分和<aside>侧边栏内容部分均包含了<article>标签来显示文章内容，在<article>标签内部包含了普通的HTML格式化标签来显示格式化的内容。可以看到，<section>是一个包罗万象的标签，在里面还可以嵌套<section>标签，<section>内部的文字内容区通过<article>进行区分，对于侧边栏用<aside>

进行划分，它使得程序员一眼就能理解网站的布局结构。

3．页脚部分

页脚部分主要显示页面底部的导航栏和版权信息，它使用<footer>进行标识，代码如下：

```
<footer>
  <div class="container_16">
   <nav>
     <ul>
        <!—省略导航部分的代码-->
      </ul>
   </nav>
    <p class="copy"><span>房产网站</span>
           &copy;  2010    <a href="privacypolicy.html">隐私宣言
  </a></p>
  </div>
</footer>
```

可以看到，通过<footer>标签，标明这块区域将作为页面的页脚进行显示，在页脚部分也包含了一个用来进行导航的<nav>标签，同时包含一些版权文字信息，构成在页面底部显示的页脚栏。

通过对首页布局设计代码的了解，相信读者应该理解了 HTML 5 中结构性语义元素的使用，但是如果不应用 CSS 3 样式，页面的显示会比较混乱，下面将通过导航栏、内容显示布局等模块讨论如何让元素显示成想要的效果。

17.2.2 设计导航栏

在 HTML 页面上，导航栏使用<ul>和<li>标签，这基本上已经成为了导航栏设计的标准元素，重点在于 CSS 的设计，只有良好地运用 CSS 才能让导航变得美观吸引人。在首页中，导航栏定义在<header>标签内部的<nav>元素中，代码如下：

```
<!--网站的导航栏-->
<nav>
  <ul>
    <li><a href="index.html" class="current">首页</a></li>
    <li><a href="selling.html">出售房源</a></li>
    <li><a href="buying.html">购买信息</a></li>
    <li><a href="renting.html">出租房源</a></li>
    <li><a href="moving.html">房屋过户</a></li>
    <li><a href="finance.html">账务信息</a></li>
    <li><a href="contacts.html">联信我们</a></li>
  </ul>
</nav>
```

可以看到，对于当前显示的页面，会使用 class=current 类，表示要应用特别的显示样式，在网站的 style.css 文件中，对<nav>标签和其内部的子元素分别应用了样式，以便使之

显示水平的导航栏效果。导航栏的 CSS 代码如下：

```
/* 导航栏显示样式 */
header nav {
    position:absolute;      /*绝对定位方式*/
    right:25px;             /*导航栏居右对齐*/
    top:97px;               /*距离顶部 97px*/
}
header nav ul li {          /*导航顶的样式*/
    float:left;             /*于容器中向左浮动*/
    padding-left:6px;       /*左内边距 6px*/
}
header nav ul li a {
    float:left;             /*链接向左浮动*/
    color:#fff;
    text-decoration:none;
    width:80px;
    text-align:center;
    line-height:31px;
    font-size:14px;         /*链接文本*/
}
/*鼠标经过时与类 current 具有相同的样式*/
header nav ul li a:hover,
header nav ul li a.current {
    /*指定链接背景*/
    background:url(../images/nav-bg.gif) 0 0 repeat-x;
    /*使用圆角矩形*/
    border-radius:5px;
    -moz-border-radius:5px;
    -webkit-border-radius:5px;
}
```

nav 容器指定绝对定位，right 指定居右显示，位于右侧的 25px 靠齐；top 指定位于顶部的距离，因为是绝对定位，因此它总是固定在顶部 97px 的位置，并且与右侧保持 25px 的距离。<li>标签内的链接和<li>都会向左浮动，以便导航能够水平进行显示。其中链接部分指定了各种字体、对齐、行高等属性用于控制链接的显示方式。

鼠标经过（a:hover）与 CSS 类 current 使用相同的样式，类 current 将被用于当前选中的链接，CSS 属性 background 指定链接的背景色，同时通过 border-radius 指定圆角外边框，因此使得连接的显示具有圆角矩形的效果，如图 17.7 所示。

图 17.7　具有圆角矩形的链接

在过去要实现这种效果，必须要嵌入多层 CSS，并要准备具有圆角效果的图片。使用

了 CSS 3 以后，可以看到 3 行代码就轻松地实现了圆角显示。

17.2.3 设计宣传广告栏

网站的宣传栏位于名为 faded 的 div 元素内部，它包含了用于淡入淡出显示的图片区域，位于名为 rap 的 div 元素内，以及一个文字栏 pagination 中。在页面上的显示效果如图 17.8 所示。

图 17.8 宣传广告栏显示效果

宣传广告栏的一个特色就是当单击图文混排区中的文字介绍时，会自动淡入淡出显示圆角矩形区的宣传图案，这是通过 jQuery 代码来实现的。宣传广告栏中图片的 CSS 样式定义如下：

```
/* 广告宣传栏容器显示*/
#faded {
    position:absolute;
    left:0;
    top:161px;              /*在距离顶部 161px 的位置显示*/
    padding-bottom:20px;
}
/*宣传图片显示样式*/
#faded .rap {
    /*背景图片*/
    background:url(../images/img-wrapper-bg.jpg) no-repeat 50% 0 #d92400;
    border:1px solid #e46b00;       /*边框*/
    width:589px;                    /*图片宽度和高度*/
    height:416px;
    border-radius:8px;              /*圆角矩形*/
    -moz-border-radius:8px;
    -webkit-border-radius:8px;
    /*阴影设置区*/
```

```
    box-shadow:-2px 8px 5px rgba(0,0,0,.6);
    -moz-box-shadow:-2px 8px 5px rgba(0,0,0,.6);
    -webkit-box-shadow:-2px 8px 5px rgba(0,0,0,.6);
    z-index:10;                    /*指定 z 轴*/
    overflow:hidden;               /*溢出区隐藏显示*/
}
#faded .rap img {
    margin:9px 0 0 9px;            /*图片外间距*/
}
```

在代码中，id 值为 faded 的 div 是整个外部容器，它使用绝对对齐方式，在距离顶部 161px 的位置显示。faded 内部包含一个 class 为 rap 的 div 元素，用来放置多幅图片，rap 内部放置了 3 个<img>标签，分别对应不同的图像。#faded .rap 用于定义图片容器的样式，可以看到使用 background 指定了红色的图片背景，同时使用 border_radius 和 box_shadow 指定了边框的圆角样式和阴影，将该 div 的 z-index 指定为 10，使其永远显示在其他 div 的前面。最后当容器内部的图片超过容器的大小时，将被隐藏，使得容器内部的图片总是显示在圆角 div 的内部。

可以看到，通过 box_radius 和 box_shadow 使得网站的宣传图片具有了圆角和阴影的显示效果，宣传文字栏是使用<ul>和<li>来实现的，它使用了很多 CSS 3 中的新样式，比如圆角、阴影还有转换 transition 的控制，以下代码演示了关于宣传文字栏的样式设置。

```
/*宣传内容区的 CSS 样式定义*/
#faded ul.pagination {
    position:absolute;             /*绝对定位方式*/
    left:537px;                    /*距离左边 537px 的位置显示*/
    top:10px;                      /*距离容器顶部 10px*/
    /*ul 的背景图片*/
    background:url(../images/pagination-splash.gif) no-repeat 0 0 #2a2a2a;
    border:1px solid #3a3a3a;      /*边框*/
    border-radius:8px;             /*圆角矩形*/
    -moz-border-radius:8px;
    -webkit-border-radius:8px;
    /*显示边框阴影*/
    box-shadow:-2px 8px 5px rgba(0,0,0,.4);
    -moz-box-shadow:-2px 8px 5px rgba(0,0,0,.4);
    -webkit-box-shadow:-2px 8px 5px rgba(0,0,0,.4);
    z-index:9;                     /*z 轴为 9，显示在图片栏的下面*/
    padding:25px 0 25px 0;         /*内部间距*/
}
/*内容区单个文字项*/
#faded ul.pagination li {
    width:429px;
    position:relative;
    /*在背景处显示底部分隔栏*/
    background:url(../images/line-bot.gif) no-repeat 77px 100%;
```

```
    padding-bottom:1px;
    height:1%;
}
#faded ul.pagination li:last-child {
    background:none;                    /*最消最后一个元素的背景显示，取消最后的分隔线*/
}
/*每个项内部都是一个链接，在这里指定链接的样式*/
#faded ul.pagination li a {
    display:block;                      /*在页面内用块级显示*/
    padding:16px 40px 14px 77px;        /*指定页面内间距*/
    overflow:hidden;                    /*溢出方式显示为隐藏*/
    color:#7f7f7f;
    text-decoration:none;               /*取消链接的下划线*/
    font-size:9pt;                      /*字段为 9pt*/
    line-height:28px;
    height:1%;
    cursor:pointer;                     /*鼠标指针的显示方式*/
    /*控制链接第 0.3 秒淡出显示。*/
    -moz-transition:all 0.3s ease-out;
    /* FF3.7+ */
    -o-transition:all 0.3s ease-out;
    /* Opera 10.5 */
    -webkit-transition:all 0.3s ease-out;
    /* Saf3.2+,Chrome */
}
#faded ul.pagination li a:hover,#faded ul.pagination li.current a {
    background-color:#1d1d1d;    /*鼠标经过时，或者被选中时，调整当前颜色*/
    color:#fff;
}
```

可以看到，在宣传文字的样式控制中，使用了很多 CSS 3 的新特性，包含圆角、阴影边框，还包含了 transition 转换效果的设置，下面是 CSS 代码的定义内容：

（1）样式类为 pagination 的 ul 使用绝对定位方式，left 指定左侧 527px 位置处显示列表内容，background 用于指定背景的图片 。整个 ul 使用 border-radius 显示了 8px 的圆角，使用 box-shadow 显示整个 ul 的阴影区域。指定 z-index 为 9，它比图像区的 rap 的 z-index 属性要小，因此如果与之重叠会显示在图片的后面。

（2）ul.pagination li 用来设置每个列表项的样式，它使用相对对齐方式，background 指定了一个图片，这个图片用来在列表项的底部显示一条分隔线，由于在最后一个列表项下面显示分隔线很明显会影响显示，因此使用 CSS 选择器 li:last-child 为列表项中的最后一个元素指定 background:none，不显示背景分隔线。

（3）每个列表项内部是一个<a>标签组成的链接，#faded ul.pagination li a 样式指定了链接的显示方式，每个链接使用 display:block 指定块级显示方式，text-decoration:none 指定取消链接下划线的显示，这里使用了 CSS 3 的转换样式 transition，与 border-radius 类似，前缀-moz 和-webkit 用来指定不同的浏览器的兼容性方式。以-moz-transition:all 0.3s ease-out

这句为例，all 表示当<a>标签的任何属性发生更改时，将在 0.3 秒的时间内淡出显示，因此只要<a>链接的属性方式更改，就会出现淡出的效果。

（4）由于为<a>标签定义了属性改变时的转换效果，接下来定义了 a:hover 样式改变<a>的 background-color 和 color 属性，因此鼠标经过时，就具有了转换的效果。

网站宣传栏的另一个有趣的特性是它除了用 CSS 3 的转换效果外，还会自动进行幻灯播放，这是通过 jQuery 的插件 jquery.faded.js 来实现的，在 index.html 页面中包含了 jquery.faded.js 文件，同时在 script.js 中包含了用于设置 id 为 fade 的 div 自动播放方式，代码如下：

```
$(function(){
    // 幻灯播放
    $("#faded").faded({
     speed: 500,              //指定播放速度
     crossfade: true,         //是否循环播放
     autoplay: 5000,          //自动播放时间
     autorestart: 3000,       //自动重新播放时间
     autopagination:false     //自动翻页
    });
})
```

该函数要求 的命名与<ul>的命名符合规范，在设置了这行代码后，在页面加载时调用 faded 方法就可以实现自动播放的效果，当鼠标悬停时会暂停在当前播放位置。

17.2.4　CSS 布局

网站的主要内容部分位于 id 为 content 的 section 区域中，实际上 section 主要是起一个语义性的作用，在 section 标签的内部创建了一个 class 类为 container_16 的容器，这个容器将用来作为 section 中元素的布局容器。

该容器的 CSS 样式永久在 grid.css 文件中，这个文件实际上是由 Variable Grid System 在线的布局生成器来产生的 CSS 样式，通过下面的网址可以查看 Variable Grid System，可以学习这个在线生成器的使用方法：

```
http://grids.heroku.com/
```

container_16 的 CSS 实现如下：

```
.container_16 {
    margin-left: auto;
    margin-right: auto;      /*水平和垂直居中显示*/
    width: 976px;            /*宽度 976px*/
}
```

以 container_16 内部包含了一个 class 为 clearfix 的 div，这个元素主要用于清除浮动，在该 div 内部才开始 mainContent 这个 section、aside 的布局实现。

在 div 内部包含一个 section，用来对首页的主要内容部分进行布局，一个 aside，用来对侧边栏的提示区进行布局。

可以看到，id 为 mainContent 的 section 元素，其 class 指定为 grid_10，该 CSS 类指定

了 section 的布局样式和宽度，统一定义在 grid.css 文件中，在 mainContent 内部包含了 2 个 article 元素，第 1 个 article 用来显示文章内容，第 2 个 article 用来显示文章的链接。它们分别在 style.css 中指定了其显示的样式。代码如下：

```
.grid_1,
/*省略 gridxxx*/
.grid_16 {
    display:inline;                          /*使用内联显示方式*/
    float: left;                             /*向左浮动显示*/
    position: relative;                      /*相对显示方式*/
    margin-left: 6px;                        /*左右外边距*/
    margin-right: 6px;
}
/*grid10 的宽度设置*/
.container_16 .grid_10 {
    width:598px;
}
/* 主要内容区的 article 的样式，定义在 style.css 文件*/
#mainContent article {                       /*第 1 个 article 的显示样式*/
    padding:0 0 32px 0;                      /*文章内边距的显示样式*/
    margin-bottom:30px;                      /*底部外边距*/
    border-bottom:1px dashed #323232; /*显示底部边框*/
}
#mainContent article.last {                  /*第 2 个 article 的显示样式*/
    padding-bottom:0;
    margin-bottom:0;                         /*底部内外边距都为 0*/
    border:none;                             /*不显示页面边框*/
}
```

可以看到，实际上这里的 CSS 已经非常简单了，grid_10 指定浮动的样式为 left，用于进行向左浮动，同时元素为内联显示，并且指定其宽度为 598px。几个 article 元素只需要设置内外边距和显示方式即可，grid_10 已经实现了向左浮动的显示方式。

在首页的“最新装修的房屋列表”菜单项中，包含了一个使用<ul>和<li>显示的图片列表，可以看到它们水平上下排列，并且具有圆角矩形边框，当鼠标经过时，边框会高亮并放大显示，如图 17.9 所示。

图 17.9　最新房源图片列表

此页的 HTML 定义代码如下：

```
<!--最新房源列表-->
<ul class="img-list clearfix">
    <!--房源图片列表-->
    <li><a href="#"><img src="images/thumb1.jpg" alt=""></a></li>
    ....
</ul>
```

ul 的 class 属性即指向 img-list，也指向 clearfix 类，clearfix 主要用于清除浮动，防止<ul>下面的内容跑到<ul>浮动占用区域的位置，形成重叠的效果。img-list 用来为图片布局，其样式设置来指定圆角和鼠标经过时的显示样式，详细代码如下：

```
/* 图像列表 */
.img-list {
    padding-bottom:9px;    /*底部间距为 9px*/
}
/*图像列表项样式*/
.img-list li {
    float:left;                /*向左浮动*/
    padding:0 9px 9px 0;    /*内部右和左 9px 间距*/
    width:109px;              /*每个图片宽 109px*/
    height:93px;               /*每个图片高度 93px*/
}
/*列表内部的每个链接的样式*/
.img-list li a {
    float:left;                /*向左浮动*/
    padding:4px;              /*内部间距 4px*/
    background:#fff;          /*链接背景色*/
    position:relative;        /*相对对齐方式*/
    z-index:1;
    border-radius:6px;       /*圆角矩形*/
    -moz-border-radius:6px;
    -webkit-border-radius:6px;
    behavior:url(js/PIE.htc); /*IE 兼容模式*/

    -webkit-transition-duration:0.5s;
     -moz-transition-duration:0.5s;
    -o-transition-duration:0.5s;
}
/*列表项链接的样式*/
.img-list li a:hover {
    z-index:2;                 /*指定 z 轴*/
    background:#ce2300;    /*指定背景*/
    /*使用 transform 指定缩放 1.5 倍*/
    -webkit-transform:scale(1.5);
     -moz-transform:scale(1.5);
```

```
    -o-transform:scale(1.5);
}
```

img-list 这个 class 本身只设置整个列表项的底部间距为 9px。但是它指定了内部的 li 样式为向左浮动，同时指定了每个列表项的宽度和高度以及内间距。为列表内部的图片链接<a> 指定样式，这里指定浮动为向左浮动，间距为 4px，定位方式为相对定位，同时指定了 border-radius 为 6px 表示圆角为 6 个像素，同时指定了 behavior 为 PIE.htc，表示要让 IE 的 6、7、8 版本也支持这种样式。最后指定了 transition-duration，指定转换的时间数，只要<a>的属性发生变化，将会按照 0.5 秒时间内进行转换变换。定义 a:hover 样式，当鼠标经过时，指定 z-index 为 2，要大过于其他的链接，以便图像放大时可以显示在最前面不被其他的图像所遮档。接下来指定了 CSS 3 样式 transform 的 scale 将图像放大 1.5 倍。

注意：transform 的属性包括：rotate()、skew()、scale()、translate()等方法，分别实现旋转，倾斜、缩放和定位操作。

经过上述设置之后，房源列表会水平浮动显示，每个图像呈现为圆角矩形，当鼠标经过时，图像会放大显示，鼠标离开又恢复图像原大小。

在首页的右侧，包含了一个可折叠的提示信息，用来提供房产经纪人必须具备的一些条件，如图 17.10 所示。

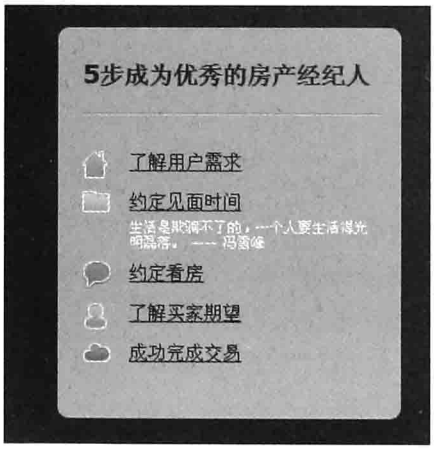


图 17.10　可折叠的侧边栏

整个侧边栏使用 HTML 5 的 aside 作为容器，aside 用来显示与主要内容相关的侧边栏，每个列表项使用<dl>和<dt>以及<dd>标签，这组标签用来定义项目列表，其中<dl>标签定义一个定义列表；<dt>标签定义列表中的项目；<dd>标签描述列表中的项目。侧边栏的布局代码如下：

```
<aside class="grid_6">
    <div class="prefix_1">
    <article>
     <!--侧边栏方块位于一个 div 中，该 div 显示为圆角矩形-->
      <div class="box">
        <h2>5 步成为优秀的房产经纪人</h2>
         <!--通过 accordion 设置内部的列表样式-->
        <dl class="accordion">
```

```
            <dt><img src="images/icon1.gif" alt=""><a href="#">了解用户需求</a></dt>
            <dd>...</dd>
            <dt><img src="images/icon2.gif" alt=""><a href="#">约定见面时间</a></dt>
            <dd>...</dd>
            <dt><img src="images/icon3.gif" alt=""><a href="#">约定看房</a></dt>
            <dd>...</dd>
            <dt><img src="images/icon4.gif" alt=""><a href="#">了解买家期望</a></dt>
            <dd>...</dd>
            <dt><img src="images/icon5.gif" alt=""><a href="#">成功完成交易</a></dt>
            <dd>...</dd>
          </dl>
        </div>
        <!-- /.box -->
      </article>
      <article class="last">
        <!-省略地址内容-->
      </article>
    </div>
  </aside>
```

id 为 box 的 div 元素主要用来显示圆角的外边框，可折叠的项目定义在<dl>标签中，它应用了样式 accordion 来设置列表项的样式，以保持可折叠的效果。具体实现请读者参考 style.css 的样式定义代码。

通过首页的实现，可以看到各种 HTML 5 标签的灵活运用，比如 section、article、dl、dt、dd 等元素，理解了这些元素的应用方式，可以在规划自己的页面时，考虑如何使用这些语义性元素来增强页面的可理解性。

17.3　搜房网的内容页设计

当首页设计完成后，网页的其他页面除非有较大的风格变换，否则一般的商业网站都会保持统一的页面格式。首页之外的内容页基本上保留与首页相同的导航、Logo、类似的显示字体等，但是在局部会有适用于不同页面的变化，本节将讨论首页之外的其他页面的设计。

17.3.1　出售房源页面

出售房源位于 selling.html 页面，该页面的布局基本上与 index.html 相似，在导航栏部分，指定 li 内容为“出售房源”的 class=current，这样可以高亮显示当前出售房源页面，如下：

```
<li><a href="selling.html" class="current">出售房源</a></li>
```

出售房源在布局时，在 mainContent 这个 section 中，直接由两个 article 元素组成，可以看到，在这两个 article 内部使用了 figure 和 hgroup 进行布局。mainContent 的布局实现如下：

```
<section id="content">
  <div class="container_16">
   <div class="clearfix">
        <section id="mainContent" class="grid_10">
          <article>
            <h2>优质房源</h2>
            <div class="wrapper">
                <!--figure 标签规定独立的流内容（图像、图表、照片、代码等）。-->
              <figure><img src="images/2page-img1.jpg" alt=""></figure>
              <h3>...</h3>
              <p>...</p>
              <a href="#" class="button">了解更多</a>
            </div>
          </article>
          <article class="last">
              <h2>抵押贷款中心</h2>
            <!--对网页标题进行组合显示-->
            <hgroup>
              <h5></h5>
              <h4></h4>
            </hgroup>
            <div class="img-box">
                <!--figure 标签规定独立的流内容（图像、图表、照片、代码等）。-->
              <figure><img src="images/2page-img2.jpg" alt=""></figure>
                <p>..</p>
            </div>
            <p>...</p>
            <a href="#" class="button">了解更多</a>
          </article>
</section>
```

可以看到名为 content 的 section 标签内部包含一个 container_16 的 div，该 div 与 index.html 中介绍的一样，是个内容布局容器。在一个使用 clearfix 类进行清除浮动的内部包含了 id 为 mainContent 的 section 元素，可以看到基本上布局与首页类似，在 section 内部有 2 个 article，不同之处在于在对子元素布局时，使用 figure 来显示图像，对于多个标题组合，使用了 hgroup 元素，这里演示了 HTML 5 的语义性元素在页面中的具体应用。

在 style.css 内部，为 hgroup 和 figure 分别定义了显示样式，figure 主要用来包含图像，hgroup 则是组合 HTML 中的<h1>..<h6>这样的样式。这两个 HTML 5 元素的样式如下：

```
/*图片部分样式设置*/
figure {
    margin:0 24px 0 0;    /*外边距*/
    float:left;           /*向左浮动*/
    padding:4px;
    background:#fff;
```

```
    position:relative;      /*相对对齐方式*/
    /*使用 CSS 3 定义圆角*/
    border-radius:6px;
    -moz-border-radius:6px;
    -webkit-border-radius:6px;
    /*IE 兼容模式*/
    behavior:url(js/PIE.htc);
}
 /*标题组样式设置*/
hgroup h5 {
    margin-bottom:4px;
}
hgroup h4 {
    margin-bottom:18px;
}
```

figure 用来显示图像，在 CSS 代码中，它指定了 margin 属性用来设置图像的外边距，在进行图文混排时经常使用这个属性来保持与文字内容的间距。图像使用 float:left 进行左浮动显示，并且使用相对对齐的方式。在 CSS 代码中可以看到 figure 使用 border-radius 指定了图像进行圆角显示，并且通过 behavior 设置了 IE 兼容性模式。hgroup 中的 h5 和 h4 分别定义了 margin-bottom 样式，实际上 h5 和 h4 在 CSS 中已经重新定义了显示的风格，因此在这里会继承样式设置，并应用 margin-bottom 的定义。

经过上述设置，可以看到经典的图文混排结构，只是这里使用了 HTML 5 新的语义性标签，图文混排的效果如图 17.11 所示。

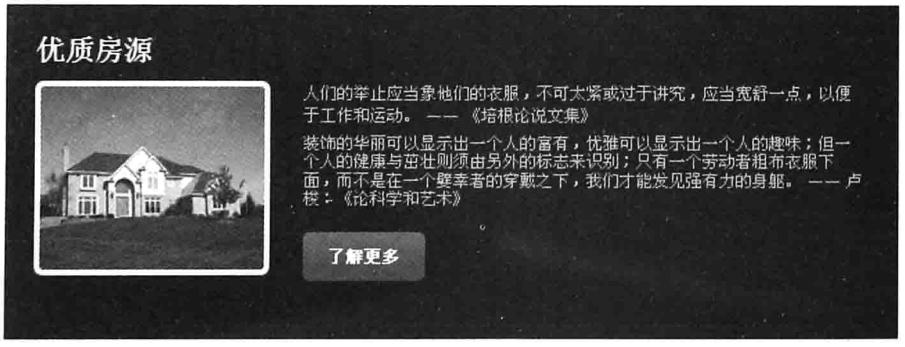


图 17.11　出售房源的图文混排结构

17.3.2　购买房源页面

购买信息页面的布局与首页和出售页的布局比较相似，细微之处在于右侧的市场调研，它包含了两幅水平排列的图片和图文混排结构，这个结构中的图片也使用了 figure 元素进行标识，如图 17.12 所示。

图 17.12　市场调研侧边栏

整个侧边栏放在 aside 元素内部，市场调研的图文混排结构被放在 article 元素内部，定义代码如下：

```
<aside class="grid_6">
<div class="prefix_1">
<article>
    <h2>市场调研</h2>
    <!--显示水平排列的图片-->
    <div class="wrapper">
        <figure class="alt">
            <img src="images/3page-img6.jpg" alt="">
        </figure>
        <figure class="last">
            <img src="images/3page-img7.jpg" alt="">
        </figure>
    </div>
    <!--显示文字内容-->
    <h4>....</h4>
    <p>.....</p>
    <a href="#" class="button">了解更多</a>
  </article>
 </div>
</aside>
```

可以看到，图片包含在一个 class 为 wrapper 的 div 元素中，它们都包含在 aside 元素内部，表示作为网站侧边栏进行显示，由于 figure 类已经定义了 float:left 的向左对齐方式，因此图片会自动向左浮动，class 为 wrapper 的 div 定义了容器的基本样式，CSS 定义代码如下：

```
/*图片容器样式*/
.wrapper {
    width:100%;          /*宽度 100%*/
```

```
    overflow:hidden;        /*溢出部分隐藏显示*/
}
/*alt 类的样式定义*/
figure.alt {
    margin-right:5px;
}
/*图片的 last 类的设置*/
figure.last {
    margin-right:0;
}
```

wrapper 样式指定宽度占据 aside 的 100%宽度，对于子元素超过其宽度的部分隐藏显示， figure 指定了图片的圆角显示，并且在 wrapper 元素中向左浮动，实现两幅图片水平排列的效果，figure 的 alt 样式类指定右边距 5px，它指定了与另一幅图片之间的间距，而 last 样式类指定右边距 0px，表示与 wrapper 容器之间的间距可为 0px。

17.3.3　出租房源页面

出租房源页面包含了房屋的租赁信息，位于 renting.html 页面，在导航栏中指定 class=current 为该页面，代码如下：

```
<li><a href="renting.html" class="current">出租房源</a></li>
```

这表示将导航栏的当前项设置为该页面，页面的其他部分与首页比较类似，比如宣传广告栏保持了与首页相同的风格，在布局方面不同之处在于资产管理部分，它包含了 hgroup 组成的标题组，并且通过<ul>和<li>构建了 3 栏式的导航项结构，如图 17.13 所示。

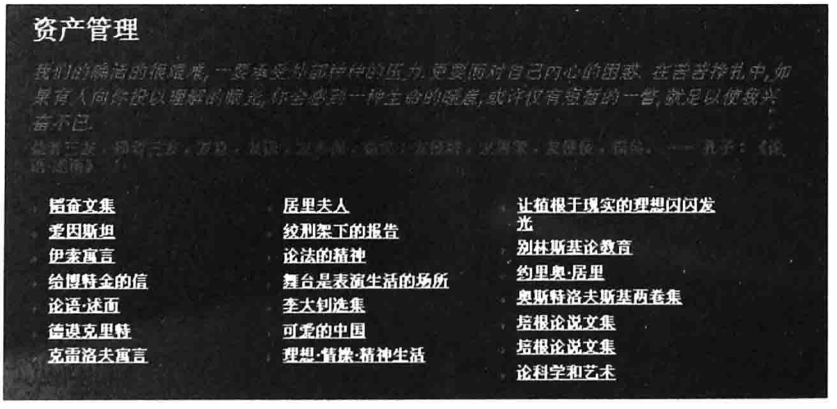


图 17.13　资产管理的导航结构

由图中可以看到，导航栏 3 栏水平显示，它具有自定义的项目符号和显示样式。以下代码显示了资产管理的布局，可以看到它的结构非常简单，由 hgroup 和常规的 ul 与 li 组成。

```
article class="last">
  <h2>资产管理</h2>
  <hgroup>
    <h5>标题文本内容</h5>
```

```
    <h4>幅标题文本内容</h4>
  </hgroup>
  <div class="wrapper p2">
    <ul class="list1 grid_3 alpha">
     <li><a href="#">...</a></li>
       <li>省略其他列表项内容</li>
    </ul>
    <ul class="list1 grid_3">
     <li><a href="#">...</a></li>
       <li>省略其他列表项内容</li>
    </ul>
    <ul class="list1 grid_3 omega">
     <li><a href="#">...</a></li>
       <li>省略其他列表项内容</li>
    </ul>
  </div>
  <!--正文段落信息-->
  <p>...</p>
  <p>...</p>
  <a href="#" class="button">了解更多</a>
</article>
```

整个布局内容包含在 article 元素中，可以看到，article 元素可以不拘一格地使用，只要是为了呈现与文章相关的页面都可以使用 article 元素，当为 article 指定 class=last 时，表示这是位于主内容容器中的后一段落。<h2>包含的资产管理用来指定内容，<hgroup>标签内的文本内容是副标题，它是由两个标题元素组成的组合。class 为 wrapper 的 div 元素包含了 3 个水平排列的 ul 元素，通过 CSS 来控制其显示的样式。控制这个布局的 CSS 样式如下：

```
/*列表容器样式*/
.wrapper {
    width:100%;              /*宽度 100%*/
    overflow:hidden;         /*溢出部分隐藏显示*/
}
/*指定底部边距为 18px*/
.p2 {
    margin-bottom:18px;
}
/*列表项的样式*/
.list1 li {
     /*列表项的背景*/
    background:url(../images/arrow1.gif) no-repeat 0 7px;
    padding:0 0 6px 15px;
    font-size:9pt;
    zoom:1;
}
```

```
/*列表项链接的样式*/
.list1 li a {
    color:#fff;
    font-weight:bold;
}
/*每个列表项布局的样式*/
.grid_3{
    display:inline;          /*使用内联显示方式*/
    float: left;             /*向左浮动显示*/
    position: relative;      /*相对显示方式*/
    margin-left: 6px;        /*左右外边距*/
    margin-right: 6px;
}
.omega {
    margin-right: 0;
}
```

div 的 wapper 类指定宽度为 100%，且溢出设置显示为隐藏，p2 类指定 margin-bottom 为 18px，表示下外边距显示为 18px 像素。list1 的列表项 li 指定了箭头背景，让列表项显示箭头，每个列表项内部是一个链接，通过指定.list1 li a 的样式来指定每个链接的字体和颜色。每个 ul 元素都使用了 grid_3 样式，因此每一个<ul>标签会使用内联显示方式，float:left 表示每个<ul>标签将向左浮动显示，并显示相对对齐方式。

可以看到，三栏式的 ul 元素的显示核心在于浮动式布局方式的使用，而整个网页布局中，大量浮动式布局的应用和 CSS 盒子模型样式的使用，使得网页的效果既丰富，又容易理解，并且保持了良好的搜索引擎优化特性。

17.3.4　房产过户页面

房产过户页面 moving.html 包含了买房过户的一些信息，在页面的布局上中规中矩，与前面介绍的几个页面的布局基本相似。下面就从全局的角度对整个页面的布局进行一次回顾，由于页面的导航和 Logo 以及宣传图片部分已经进行过详细的讨论，下面看一下房产过户内容部分的布局，如图 17.14 所示。

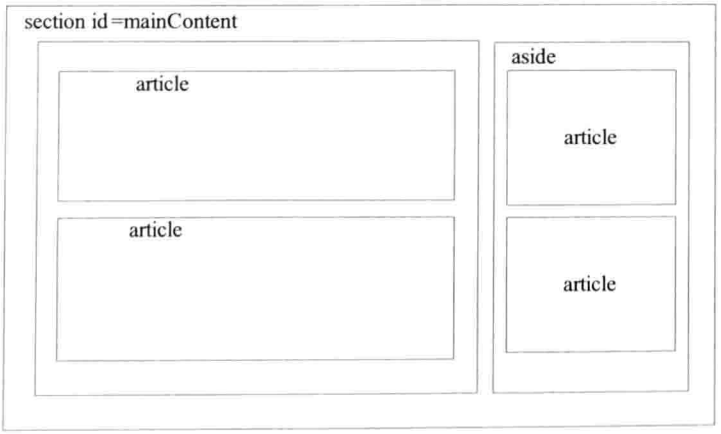


图 17.14　房产过户内容页布局

由图中可以看出，语义性结构元素用来完成页面的布局设计，它构成了整个页面的主体，在语义性结构元素内部通过语义性块元素或传统的 DIV+CSS 布局方式，让整个页面不仅外观显示可以具有创造性，而且代码也具有可读性，因此可以说 HTML 5 的出现会统一过去 HTML 版本中的混乱性，无论从其本身增强的功能还是这些语义性的标签来说，相较于过去的 HTML 版本都有了明显的变化。

将图中的布局方式形成 HTML 代码的话，如下所示，这里出于简化的目的取消了页面中出现的文本。

```
<!-- 页面主要内容布局区域 -->
<section id="content">
  <div class="container_16">
   <div class="clearfix">
        <section id="mainContent" class="grid_10">
         <!--这里是第一段文字内容-->
         <article>
           <h2>关于过户手续</h2>
           <h3>子标题文字介绍 </h3>
           <h4>小标题文字介绍</h4>
           <p>这里用来放置较长的文本</p>
           <a href="#" class="button">了解更多</a>
         </article>
         <!--这里是第二段文字内容-->
         <article class="last">
           <h2>过户事项</h2>
           <div class="img-box">
             <figure><img src="images/5page-img1.jpg" alt=""></figure>
             <p>这里是图文混排的文字内容</p>
           </div>
           <h3>这里是纯标题文字</h3>
           <h4>这里是纯标题文字</h4>
           <a href="#" class="button">了解更多</a>
         </article>
       </section>
       <!--这里是侧边栏的显示，用 aside 标识-->
        <aside class="grid_6">
           <div class="prefix_1">
           <!--侧边栏的第 1 段-->
           <article>
             <h2>过户步骤</h2>
             <h4>简要的文字小标题</h4>
             <p>对过户信息的文字描述</p>
             <p>对过户步骤的文字描述</p>
             <a href="#" class="button">了解更多</a>
           </article>
```

```
            <!--侧边栏的第 2 段-->
            <article class="last">
              <!--创建一个基于 CSS 的盒子样式 -->
              <div class="box">
                <h2>帮助 &amp;amp; 技巧</h2>
                <h3><a href="#">注意过户的骗局</a></h3>
                <p>帮助文本</p>
                <h3><a href="#">房屋买卖技巧</a></h3>
                <p>帮助文本</p>
                <a href="#" class="button">了解更多</a>
              </div>
            </article>
          </div>
        </aside>
      </div>
    </div>
  </section>
```

基本上内容页都以 section 开始，id 都指定为 content，表示页面的内容页开始了，id 为 content 的 section 是与 header 和 footer 处于同级别的容器，用来对整个页面进行布局设计。在确定了全局范围的容器后，对于内容部分的布局，就可以考虑用 section 作为主内容，侧边栏的内容用 aside，这是很显然的考虑，aside 应该总是侧边栏，可以是左侧也可以是右侧。article 就是用来显示文本片段的，因此它可以放在 section 中。

关于布局的 CSS 内容，需要理解 CSS 中的盒子模型和 CSS 选择器的使用，可以看到在完成实例的过程中，大量使用了浮动式布局以及内、外边距。如果读者对这些还不是很理解，可以回过头再重温下第一篇的内容。

17.3.5　联系我们页面

联系我们页面包含了公司的联系方式、一个发送留言的表单，从整体布局来说，它的组织方式与图 17.16 并没有什么相同之处，但是它的联系方式使用了 figure、dl、dt 和 dd 标签，通过 CSS 来控制联系方式的显示，布局代码如下：

```
<article>
  <h2>主要联系方式</h2>
  <div class="wrapper">
    <!--圆角显示的地址图片-->
    <figure><img src="images/7page-img1.jpg" alt=""></figure>
    <!--第 1 个联系方式-->
    <dl class="address">
     <dt>红岭中路<br />
       中信大厦 x 楼 x 座</dt>
      <dd><span>手机：</span>13888888888</dd>
      <dd><span>公司坐机：</span>07558888888</dd>
```

```
        <dd>电子邮件： <a href="#">webmaster@domain.com</a></dd>
      </dl>
      <!--第 2 个联系方式-->
      <dl class="address last">
       <dt>黄埔大道<br />
        富兴商贸大楼西塔</dt>
        <dd><span>手机：</span>13888888888</dd>
        <dd><span>公司坐机：</span>02058888888</dd>
        <dd>电子邮件：<a href="#">webmaster@domain.com</a></dd>
      </dl>
      <a href="#" class="button">了解更多</a>
    </div>
</article>
```

联系方式位于 mainContent 中的第 1 个 article 元素内部，因此它会显示在宣传广告下面。联系方式包含在 class 为 wrapper 的 div 中，wrapper 的样式可以参考前面的代码，它具有 100%的宽度，溢出部分隐藏显示。接下来使用了一个 figure，显示图片相关的信息，figure 的样式会向左浮动，并且具有圆角的样式。在页面上放了两个联系方式，每个都使用 dl 和 dt 进行设置。在 HTML 中，dl 表示一个内容块，dt 指定了内容块的标题，dd 指定具体的内容项，第 1 个 dl 应用了 address 样式，它会指示元素向左浮动，用来与圆角图片保持水平一致。应用于地址显示的 CSS 代码如下：

```
/*地址的显示样式*/
.address {
    float:left;
    margin:0 30px 10px 0;
}
/*地址中的 last 类的显示样式*/
.address.last {
    margin-right:0;
}
/*地址中的 dt 显示样式*/
.address dt {
    margin-bottom:4px;
    text-transform:uppercase;
}
/*地址中的 dd 显示样式*/
.address dd {
    clear:both;
}
/*地址中的 span 显示样式*/
.address dd span {
    float:left;
    padding-right:25px;
}
```

```
/*地址中的 a 显示样式*/
.address dd a {
    color:#c0c0c0;
}
```

address 样式的 float 属性指定为 left，表示在 wrapper 容器中向左浮动，在 address 中的 dd 指定了 clear:both，表示清除尾部的浮动，这使得后面的元素会换行显示，dd 内部的 span 标签也使用了浮动，但是清除浮动后，会使得下面的项换行显示。通过这样的设置，使得地址栏的显示样式即有水平浮动的效果，每个地址又能垂直显示。

17.4　小结

本章讨论了基于 HTML 5 和 CSS 3 的房产管理网站，首先讨论了 HTML 5 提供的语义性元素的作用，介绍了网站的整体结构，并对网站的实现页面进行了预览。接下来讨论了如何使用 HTML 5 的语义性标签设计首页，并讨论了 CSS 3 和 jQuery 为首页带来的新效果。最后介绍了 CSS 在布局中的运用。

笔者希望通过对本章的介绍，让想了解 HTML 5+CSS 3 具体应用的读者有一个清晰的思路，并通过这个案例，具备一定的动手能力。本书后面就介绍如何使用框架来设计网站，从而提高开发和设计人员的工作效率。

第 18 章
使用 Bootstrap 实现论坛后台管理系统

后台管理系统是 Bootstrap 应用最为广泛的地方，很多项目虽然展示给用户的界面看不到一丝 Bootstrap 的影子，但是它们的后台都是由 Bootstrap 快速搭建的。本章将结合一个论坛的后台管理系统页面，帮助读者在实际应用中快速搭建美观实用的前端界面。

本章主要知识点如下：

- 结合前面所学，实践 Bootstrap 框架
- 学习如何搭建后台管理系统
- 掌握前端页面设计的制作流程

18.1 项目开始

开始一个项目，首先需要确定需求。论坛后台管理包含很多方面，譬如帖子审核、用户封杀、管理员和版主设置、首页推荐、数据统计等，限于篇幅，这里选择最有典型性的帖子审核作为示例。

由于政策监管和水军攻击等原因，很多论坛要么被广告帖、垃圾帖占领，要么由于大量发布敏感信息而被网监部门勒令关停，因此对论坛内容的审核是后台必不可少的功能。

那么可以方便的查看帖子内容并对帖子进行删除/通过/恢复操作，就是该页面的核心功能。此外，论坛后台管理还有很多其他功能模块，还需要有清晰的链接结构，方便向其他模块跳转。

明确了需求以后，就可以开始进行页面布局的设计了。本系统基本的设计思路如图 18.1 所示，这里为了更好地说明，使用了整页完成后的截图。

由图 18.1 可以看出，这个页面主要包括 3 个主要模块。

- 首部导航栏：包括内容审核、论坛管理、数据统计这 3 个主要模块的链接，以及搜索、消息通知、管理员登录信息等通用功能。
- 左侧边栏：内容审核分类下的功能导航，包括主贴审核、内容审核、用户管理、审核日志等模块的导航链接。
- 主功能部分：查看帖子内容，并进行通过/删除/恢复操作。

图 18.1　论坛后台管理系统的设计思路

18.2　页面布局

页面的制作和画家作画异曲同工，都是由轮廓到细节，因此页面布局是先引入类库再进行设计。

18.2.1　引入 Bootstrap 3 框架

既然使用 Bootstrap，就必须先引入。为了方便，这里使用 Bootstrap 中文网提供的 CDN 链接引入 Bootstrap 3，使用百度 CDN 链接引入 jQuery。一些需要自定义的 CSS 样式则放在 main.css 文件中。

引入后代码如下：

```
<!DOCTYPE html>
<html>
  <head>
    <title>论坛管理系统</title>
    <script src="http://libs.baidu.com/jquery/1.9.0/jquery.js"></script>
    <script src="http://cdn.bootcss.com/twitter-bootstrap/3.0.2/js/bootstrap.js"></script>
    <link href="http://cdn.bootcss.com/twitter-bootstrap/3.0.2/css/bootstrap.css" rel="stylesheet">
    <link href="main.css" rel="stylesheet">
  </head>
  <body>
    .....
  </body>
</html>
```

这里考虑到该系统可能在移动设备上使用，再加上 Bootstrap 3 默认采用响应式设计，因此需要在头部添加 viewport 的 meta 标签。此外为了防止在某些没有指定编码的浏览器

下出现乱码，需要添加 meta 标签来指定 charset=UTF-8。

添加后代码如下：

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>论坛管理系统</title>
    <script src="http://libs.baidu.com/jquery/1.9.0/jquery.js"></script>
    <script src="http://cdn.bootcss.com/twitter-bootstrap/3.0.2/js/bootstrap.js"></script>
    <link href="http://cdn.bootcss.com/twitter-bootstrap/3.0.2/css/bootstrap.css" rel="stylesheet">
    <link href="main.css" rel="stylesheet">
  </head>
  <body>
    .....
  </body>
</html>
```

18.2.2 编写布局代码

编写布局代码，页头采用 nav 标签，主内容则包裹在 Bootstrap 默认的 container 容器内部。在 container 内部，则分为左侧边栏和右侧的主功能部分，这里笔者采用的比例是 1:5，即 12 列栅格中，左侧占两列，右侧占 10 列。而在小屏幕设备下，则采用堆叠放置。

代码如下：

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>论坛管理系统</title>
    <script src="http://libs.baidu.com/jquery/1.9.0/jquery.js"></script>
    <script src="http://cdn.bootcss.com/twitter-bootstrap/3.0.2/js/bootstrap.js"></script>
    <link href="http://cdn.bootcss.com/twitter-bootstrap/3.0.2/css/bootstrap.css" rel="stylesheet">
    <link href="main.css" rel="stylesheet">
  </head>
  <body>
    <!--页头-->
    <div class="header">
      <nav>

      </nav>
    </div>
    <div class="container">
```

```
        <div class="row">
          <!--左侧目录-->
          <div class="col-xs-12 col-sm-2 col-md-2 col-lg-2">
            ..............
          </div>
          <!--右侧主要内容-->
          <div class="col-xs-12 col-sm-10 col-md-10 col-lg-10">
            ..................
          </div>
        </div>
      </div>
    </body>
</html>
```

18.3　实现导航栏

在需求明确、设计和布局完成后，就可以进行细节的施工了，首先需要实现页面的头部导航功能。

本例中的头部导航主要包括标题、主要功能模块的链接、搜索框、通知、登录信息 5 个部分，主要采用 Bootstrap 中内置的头部导航组件来实现。

18.3.1　构建导航的框架代码

根据之前介绍过的 Bootstrap 头部导航可以进行如下设置：

```
<nav class="navbar navbar-default" role="navigation">
  <div class="navbar-header">
     <!--这里设置标题 -- >
  </div>
  <div class="collapse navbar-collapse" >
    <ul class="nav navbar-nav">
      <!--这里设置导航链接-- >
    </ul>
    <ul class="nav navbar-nav navbar-right">
       <!--这里设置搜索、通知、登录信息-- >
    </ul>
  </div>
</nav>
```

18.3.2　填写标题和导航链接

标题的设置需要在 …… 内添加标题链接，要为该链接

添加.navbar-brand 类。

导航链接只要在<ul class="nav navbar-nav">.....
内添加列表项即可，用.active 类表示当前所处的功能模块。

```
<nav class="navbar navbar-default" role="navigation">
  <div class="navbar-header">
<a class="navbar-brand" href="#">论坛管理系统</a>
  </div>
  <div class="collapse navbar-collapse" >
    <ul class="nav navbar-nav">
      <li class="active"><a href="#">内容审核</a></li>
      <li><a href="#">论坛管理</a></li>
      <li><a href="#">数据统计</a></li>
    </ul>
    <ul class="nav navbar-nav navbar-right">
      <!--这里设置搜索、通知、登录信息-- >
    </ul>
  </div>
</nav>
```

现在的样式如图 18.2 所示。

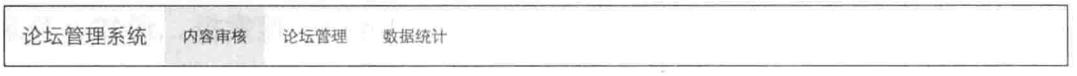


图 18.2　头部导航

18.3.3　添加搜索框和通知系统

对于搜索框，需要为其外层的 form 元素添加.navbar-form 和.navbar-left 类，为了美观，这里没有添加显式的提交按钮，而将提交按钮设为了隐藏，通过回车键提交表单。

通知系统则使用了 Bootstrap 的徽章系统，在值为空时会自动隐藏。

```
<nav class="navbar navbar-default" role="navigation">
  <div class="navbar-header">
<a class="navbar-brand" href="#">论坛管理系统</a>
  </div>
  <div class="collapse navbar-collapse" >
    <ul class="nav navbar-nav">
      <li class="active"><a href="#">内容审核</a></li>
      <li><a href="#">论坛管理</a></li>
      <li><a href="#">数据统计</a></li>
    </ul>
    <ul class="nav navbar-nav navbar-right">
      <form class="navbar-form navbar-left" role="search">
        <div class="form-group">
          <input type="text" class="form-control" placeholder="搜索">
        </div>
```

```
        <button type="submit" class="btn btn-default hidden">Submit</button>
      </form>
      <li><a href="#">未读消息 <span class="badge">5</span></a></li>
    </ul>
  </div>
</nav>
```

这一步完成后导航的样式如图 18.3 所示。

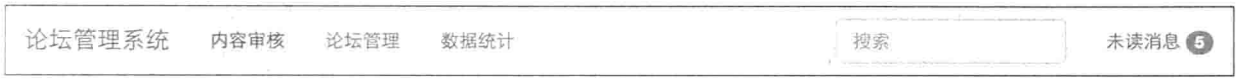


图 18.3　添加了搜索框的导航样式

18.3.4　添加管理员的登录信息

如果管理员未登录，这里应当显示登录的链接（对于后台管理，一般是不开放注册的）；如果管理员已经登录，这里应当显示管理员的用户名，并提供下拉菜单，菜单项涉及查看该管理员的操作日志，以及注销链接。

这里用到了 Bootstrap 内置的下拉菜单组件：

```
<nav class="navbar navbar-default" role="navigation">
  <div class="navbar-header">
<a class="navbar-brand" href="#">论坛管理系统</a>
  </div>
  <div class="collapse navbar-collapse" >
    <ul class="nav navbar-nav">
      <li class="active"><a href="#">内容审核</a></li>
      <li><a href="#">论坛管理</a></li>
      <li><a href="#">数据统计</a></li>
    </ul>
    <ul class="nav navbar-nav navbar-right">
      <form class="navbar-form navbar-left" role="search">
        <div class="form-group">
          <input type="text" class="form-control" placeholder="搜索">
        </div>
        <button type="submit" class="btn btn-default hidden">Submit</button>
      </form>
<li><a href="#">未读消息 <span class="badge">5</span></a></li>
<li class="dropdown">
  <a href="#" class="dropdown-toggle" data-toggle="dropdown">Admin_one
    <b class="caret"></b>
  </a>
  <ul class="dropdown-menu">
    <li><a href="#">我删除的条目</a></li>
    <li><a href="#">我修改的条目</a></li>
    <li><a href="#">我恢复的条目</a></li>
```

```
    <li class="divider"></li>
    <li><a href="#">注销</a></li>
   </ul>
</li>
    </ul>
  </div>
</nav>
```

这一步完成后，一个完整的论坛管理系统的头部导航的前端部分就基本完成了，如图18.4所示，其中消息的数量、表单提交后的处理、用户登录的判断则交由后台程序来处理。

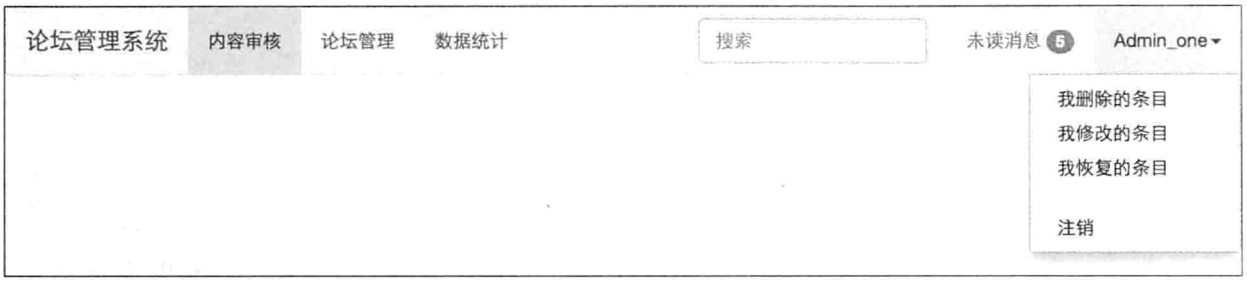


图 18.4　完整的头部导航

18.3.5　构建响应式导航

如果想让导航在小屏幕设备上实现展开和收起，还要进一步的设计：

在 …… 内添加一个指定样式的按钮，用于控制列表的展开和收起，需为该按钮添加 data-toggle="collapse"触发器和 data-target 属性。

为 添加 id 属性，id 的值要和 data-taget 属性的值对应，比如 id="set"，则 data-target ="#set"。

注意：按钮的样式可以自己控制，但是 data-toggle="collapse"这个触发器和 data-target 属性都是必需的。

完整的代码如下：

```
<nav class="navbar navbar-default" role="navigation">
  <div class="navbar-header">
    <!-- 这里是按钮的样式代码-->
    <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#set">
      <span class="sr-only">Toggle navigation</span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
    </button>
    <a class="navbar-brand" href="#">论坛管理系统</a>
  </div>
  <div class="collapse navbar-collapse" id="set">
    <ul class="nav navbar-nav">
```

```
        <li class="active"><a href="#">内容审核</a></li>
        <li><a href="#">论坛管理</a></li>
        <li><a href="#">数据统计</a></li>
      </ul>
      <ul class="nav navbar-nav navbar-right">
        <form class="navbar-form navbar-left" role="search">
          <div class="form-group">
            <input type="text" class="form-control" placeholder="搜索">
          </div>
          <button type="submit" class="btn btn-default hidden">Submit</button>
        </form>
        <li><a href="#">未读消息 <span class="badge">5</span></a></li>
        <li class="dropdown">
          <a href="#" class="dropdown-toggle" data-toggle="dropdown">Admin_one<b
    class="caret"></b></a>
          <ul class="dropdown-menu">
            <li><a href="#">我删除的条目</a></li>
            <li><a href="#">我修改的条目</a></li>
            <li><a href="#">我恢复的条目</a></li>
            <li class="divider"></li>
            <li><a href="#">注销</a></li>
          </ul>
        </li>
      </ul>
    </div>
</nav>
```

最后实现的效果如图 18.5 和图 18.6 所示。

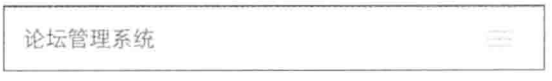


图 18.5　导航未展开时的样式

图 18.6　展开后的导航样式

这样，一个功能完整并附带响应式的头部导航就完成了。

18.4 实现左侧边栏

导航完成后，根据从上到下、从左到右的顺序，现在开始实现左侧边栏的功能，左侧边栏主要是一个大的功能模块中的子模块列表，本质上就是一组链接，这里可以选择 Bootstrap 的胶囊导航，也可以选择列表组。

笔者选择使用列表组进行实现，代码如下：

```
<div class="container">
        <div class="row">
          <div class="col-xs-12 col-sm-2 col-md-2 col-lg-2">
            <div class="list-group">
              <a href="#" class="list-group-item active">主贴审核</a>
              <a href="#" class="list-group-item">回复审核</a>
              <a href="#" class="list-group-item">用户管理</a>
              <a href="#" class="list-group-item">版主审核</a>
              <a href="#" class="list-group-item">主贴审核日志</a>
              <a href="#" class="list-group-item">回复审核日志</a>
              <a href="#" class="list-group-item">用户管理日志</a>
            </div>
          </div>
          <div class="col-xs-12 col-sm-10 col-md-10 col-lg-10">
      <!--右侧主要内容-->
            ..................
          </div>
        </div>
</div>
```

完成后的效果如图 18.7 所示。

图 18.7 左侧边栏部分

18.5 实现主功能部分

完成了头部导航和在侧边栏的工作之后，就该进行主功能展示部分的开发了。一般来

说，一个后台管理系统包括审核、管理、日志等多个模块，这里限于篇幅无法一一赘述，笔者选择主贴审核功能页面作为样例进行展开。

18.5.1　主功能的头部

按从上到下的原则，本小节先来制作主功能的头部。

（1）为了让页面区域的划分更为清晰，笔者将主功能部分包裹在一个面板中。基本框架如下：

```
<div class="col-xs-12 col-sm-10 col-md-10 col-lg-10">
  <div class="panel panel-default">
     <div class="panel-heading">
.....
<!-- 这里放置标题、选项、分页-->
</div>
<div class="panel-body">
  .....
  <!-- n 这里帖子列表-->
</div>
   </div>
</div>
```

根据功能的划分，笔者将标题、选项、分页等内容放在面板的头部，将帖子列表放在面板的内容部分。

（2）向面板头部填充预定的内容：

```
<div class="col-xs-12 col-sm-10 col-md-10 col-lg-10">
  <div class="panel panel-default">
<div class="panel-heading">
    <h4>主贴审核</h4>
    <div class="form-group">
      <span>全选 <input type="checkbox"/></span>
      <button class="btn btn-success">通过</button>
      <button class="btn btn-primary">恢复</button>
      <button class="btn btn-danger">删除</button>
      <ul class="pagination visible-md visible-lg visible-sm" id="page-right">
        <li><a href="#">&laquo;</a></li>
        <li class="active"><a href="#">1</a></li>
        <li><a href="#">2</a></li>
        <li><a href="#">3</a></li>
        <li><a href="#">4</a></li>
        <li><a href="#">5</a></li>
        <li><a href="#">&raquo;</a></li>
      </ul>
    </div>
     </div>
```

```
<div class="panel-body">
  ....................
  <!-- n 这里帖子列表-->
</div>
   </div>
</div>
```

首先是一个标题，表明该页的主要功能是“主贴审核”。

对于一个帖子，可以有 4 种状态：未审核、已通过、已删除、已恢复。一个新帖子默认为未审核状态。

首先选中一个或多个帖子，然后单击按钮进行标记，正常帖子标记为已通过，避免重复审核；散布垃圾或敏感信息的标记为删除，让其无法在页面显示并使其不能被搜索引擎索引；对于误删的帖子则可以恢复，并标记为已恢复。

这里笔者设置了通过、删除、恢复 3 个按钮来进行通过/删除/恢复操作。按钮通过 Bootstrap 内置的.btn-success、.btn-danger、.btn-primary 类设置颜色。鉴于可能出现的大量垃圾信息刷屏，还可以设置一个全选按钮，以降低审核人员的工作量。

（3）分页信息也放在这里，将它们放在同一个 …… 中，使按钮和分页信息对齐呈一排。完成后效果如图 18.8 所示。

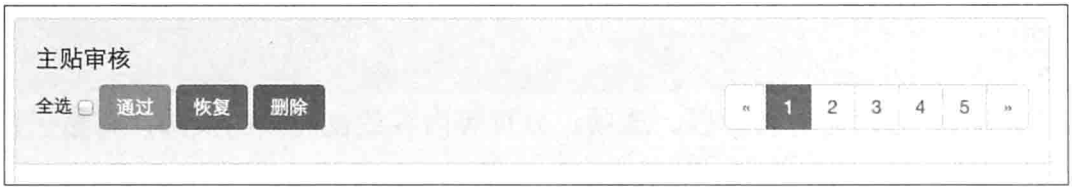


图 18.8　面板头部效果

18.5.2　主功能的帖子列表

面板头部完成后，开始制作帖子列表部分，列表每一行需要显示帖子的审核状态、主题、发帖时间、作者、详情等信息，显然这里使用表格是最合适的选择。

还有一个问题是如果有很多很长的帖子，让帖子的内容铺开在页面上显然不合适，这里需要用到 Bootstrap 的折叠插件，将帖子的详情折叠起来，审到哪一篇帖子时再展开审查。

根据这些需求，我们在面板内容部分加入如下代码：

```
<div class="panel-body">
  <table class="table">
    <tr>
      <th></th>
      <th>审核状态</th>
      <th>标题</th>
      <th>作者</th>
      <th>创建时间</th>
      <th>详情</th>
    </tr>
```

```
<tr>
  <td><input type="checkbox"/></td>
  <td><span class="label label-default">未审核</span></td>
  <td><a href="#">战士输出装备问题探讨</a></td>
  <td><a href="#">庇护祝福</a></td>
  <td>2013 年 12 月 18 日 10:52</td>
  <td>
    <a class="detail-link" data-toggle="collapse" data-parent="#accordion" href="#collapse1">
      <span class="glyphicon glyphicon-chevron-down"></span>
    </a>
  </td>
</tr>
<tr id="collapse1" class="collapse">
  <td colspan="10">
    在 MOP 之前，坦克们在保持仇恨和拉小怪的手段上都有多种选择，而在有效降低承受的伤害方面则没有太多的选择。幸运的是，这种情况已经改变。如果能够“正确使用技能”，我们可以有效地减少承受的伤害，就像 DPS 通过正确使用技能来增加伤害一样。而什么是“正确使用技能”呢？答案基本可以概括为频繁使用[盾牌猛击]、[复仇]来获得怒气，然后使用这些怒气来施放“[盾牌格挡]”以及[盾牌屏障]，从而降低你自己承受的伤害。此外，这个机制也让“命中”和“精准”变成了非常有价值的属性。
  </td>
</tr>
.....
  </table>
</div>
```

这样帖子列表部分就完成了，如图 18.9 和图 18.10 所示。

审核状态	标题	作者	创建时间	详情
未审核	战士输出装备问题探讨	庇护祝福	2013年12月18日 10:52	

在MOP之前，坦克们在保持仇恨和拉小怪的手段上都有多种选择，而在有效降低承受的伤害方面则没有太多的选择。幸运的是，这种情况已经改变。如果能够“正确使用技能”，我们可以有效地减少承受的伤害，就像DPS通过正确使用技能来增加伤害一样。而什么是“正确使用技能”呢？答案基本可以概括为频繁使用[盾牌猛击]、[复仇]来获得怒气，然后使用这些怒气来施放“[盾牌格挡]”以及[盾牌屏障]，从而降低你自己承受的伤害。此外，这个机制也让“命中”和“精准”变成了非常有价值的属性。

未审核	战士防御装备问题探讨	庇护祝福	2013年12月18日 10:52	
已通过	虎式真是太强了，应该削弱	王者祝福	2013年12月18日 10:52	
已删除	终于拿到双刀了，哈哈哈哈！	棺材步	2013年12月18日 10:52	
已恢复	重庆火锅比成都的好吃	智慧祝福	2013年12月18日 10:52	
已通过	虎式真是太强了，应该削弱	王者祝福	2013年12月18日 10:52	
已通过	虎式真是太强了，应该削弱	王者祝福	2013年12月18日 10:52	
已通过	虎式真是太强了，应该削弱	王者祝福	2013年12月18日 10:52	

图 18.9　列表第一项展开时的样式

图 18.10　列表全部收起时的样式

至此，一个论坛后台的主帖审核页面部分就完成了，完整的样式如图 18.11 所示。

图 18.11　完成后的主帖审核页面

虽然对于后台审核这样工作来说，一般都是在电脑上完成，但我们仍然做了一些响应式处理，在小屏幕下的效果如图 18.12 所示。一方面应用框架本身的特性，另一方面也方便审核人员处理突发的情况，毕竟国内对论坛的监管是很严格的，敏感信息处理不及时很容易造成重大损失。

图 18.12　小屏幕下的显示效果

18.6　小结

本章演示了如何从零开始应用 Bootstrap 3 框架完成一个论坛后台管理页面构建的过程，包括头部导航、侧边栏、主体内容 3 大部分，应用到了 Bootstrap 中的按钮、标签、表格、表单等基础样式；列表组、面板、头部导航、分页、徽章等样式组件；以及头部响应式导航、折叠、下拉菜单等 jQuery 组件。

读者从该样例中可以发现，相比传统的手动编写 CSS、JavaScript 代码的项目，这个例子作者没有自己编写一行 CSS 和 JavaScript 代码，完全应用 Bootstrap 就搭建了一个相当美观的后台管理界面。对于像后台管理这类功能性强，个性化要求不高的项目来说此做法非常合适，可以把更多的精力放在业务逻辑而非细节的调整上。

当然由于有大量的皮肤支持，再加上可以较方便地定制，即使做一些需要个性化的页面，Bootstrap 也可以胜任，只是需要做更多的工作罢了。

第 19 章
使用 Foundation 实现论坛首页

论坛这个从互联网开始兴起到现在始终长盛不衰的产品形态，如今碰上了移动互联网的兴起，人们的阅读从 PC 端流向了移动端。移动阅读早已不局限在公交地铁，随着移动设备的不断进步和运营商支持力度地提升，越来越多的人即使在家也会选择用手机或平板进行阅读。比如笔者自己就从来不用电脑阅读小说、论坛等休闲读物。

第 13 章曾介绍了移动优先的响应式框架 Foundation，本章我们就介绍如何应用 Foundation 框架快速完成一个响应式的论坛首页。

本章主要知识点如下：

- 结合前面所学，实践 Foundation 框架。
- 学习如何搭建论坛首页。
- 熟悉前端页面设计的制作流程。

19.1 项目开始

上一章讲解了应用 Bootstrap 开发论坛后台管理系统的实例，本章将向读者介绍应用 Foundation 框架进行响应式的移动论坛首页的开发过程。就框架的用法来说，Foundation 和 Bootstrap 3 区别并不大，开发者同样只需要根据官方文档设置 HTML 结构、添加相应的类、偶尔写两句 jQuery 代码就可以了；但不同于之前后台管理系统的例子，对于移动优先、用户体验优先的页面来说，在设计上要更多的考虑移动端的操作习惯和更合理的响应式布局。

在任何项目开始前，都要先明确需求。本实例是一个论坛的首页，除了标准的头部导航之外，主要有两大块内容：版块列表和热点推荐。

一般来说，可以先用一些产品原型画出草图，这里直接使用成品图代替，如图 19.1 所示。

图 19.1　论坛首页

19.2　页面布局

需求和技术方案确定后，就要着手实施了，和上一章一样，还是需要先勾勒轮廓。

19.2.1　引入 Foundation 需要的包

引入本例需要的包：

```
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>示例论坛首页</title>
    <link rel="stylesheet" href="css/foundation.css" />
    <link rel="stylesheet" href="css/main.css" />
    <script src="js/modernizr.js"></script>
  </head>
  <body>
.....
<!-- 这里编写内容-->
.....

    <script src="js/jquery.js"></script>
    <script src="js/foundation.min.js"></script>
    <script>
      $(document).foundation();
    </script>
```

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

观察这段代码，我们需要做以下工作：

- 在页头设置必要的 meta 标签：

```
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

设置 charset="utf-8"防止某些用户的浏览器下出现乱码，设置 viewport，让页面在手机浏览器下以 100%宽度展示。

- 在头部引入 Foundation 的 CSS 文件，最好还引入 Foundation 推荐的 modernizr.js 来解决一些浏览器兼容问题。
- 设置 title 属性。
- 在页面底部引入 jQuery 和 Foundation 的 JS 文件，并进行初始化：

```
<script>
  $(document).foundation();  <!-- Foundation 插件初始化代码-->
</script>
```

19.2.2 移动优先的布局

这里笔者采用了一个最简单的顺序布局，对于一个移动优先的页面，应当尽量简单，突出最重要的内容：

```
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>示例论坛首页</title>
    <link rel="stylesheet" href="css/foundation.css" />
    <link rel="stylesheet" href="css/main.css" />
    <script src="js/modernizr.js"></script>
  </head>
  <body>
<header style="background:#333">
      <!--这里编写头部导航-->
</header>
<div class=row>
  <div ="panel">
    <!--这里编写版块列表-->
  </div>
   <div ="panel">
    <!--这里编写推荐内容-->
  </div>
</div>
    <script src="js/jquery.js"></script>
```

```
    <script src="js/foundation.min.js"></script>
    <script>
      $(document).foundation();
    </script>
  </body>
</html>
```

整个布局就是一个自上而下的顺序结构，非常简单。 …… 用于保证内容的居中布置。版块列表和推荐内容分别放置在两个面板中，以对页面结构做出清晰地划分。

19.3　实现头部导航栏

布局完成后就可以着手实现了，首先仍然是实现头部导航。

（1）根据 Foundation 的文档，构建头部导航部分的框架：

```
<header style="background:#333">
  <nav class="top-bar row" data-topbar>
    <ul class="title-area">
      <!--这里填写网站标题-->
    </ul>
    <section class="top-bar-section">
      <ul class="left">
        <!--这里填写功能模块居左部分-->
      </ul>
      <ul class="right">
        <!--这里填写功能模块居右部分-->
      </ul>
    </section>
  </nav>
</header>
```

（2）根据需求向框架中填入需要的内容：

```
<header style="background:#333">
  <nav class="top-bar row" data-topbar>
    <ul class="title-area">
      <li class="name">
        <h1><a href="#">上帝之手</a></h1>
      </li>
    </ul>
<section class="top-bar-section">
      <ul class="left">
        <li class="active"><a href="#">精华区</a></li>
        <li><a href="#">排行榜</a></li>
        <li><a href="#">Wiki</a></li>
```

```
        </ul>
        <ul class="right">
          <li><input type="text" class="nav-search" placeholder="搜索"></li>
          <li><a href="#">消息</a></li>
          <li class="has-dropdown">
            <a href="#">庇护祝福</a>
            <ul class="dropdown">
              <li><a href="#">个人中心</a></li>
              <li><a href="#">我的收藏</a></li>
              <li><a href="#">注销</a></li>
            </ul>
          </li>
        </ul>
      </section>
    </nav>
</header>
```

完成后的效果如图 19.2 所示。

图 19.2　头部导航 1

（3）添加小屏幕下的展开/收起按钮。

和 Bootstrap 类似，在成功引入并初始化 Foundation 的 JavaScript 插件后，只需要添加带有对应触发器的按钮，就可以实现小屏幕下头部导航项展开/收起的功能。

这里需要在<ul class="title-area">……
内部添加如下按钮结构：

```
<li class="toggle-topbar menu-icon">
  <a href="#"><span>Menu</span></a>
</li>
```

添加后的代码如下：

```
<ul class="title-area">
    <li class="name">
      <h1><a href="#">上帝之手</a></h1>
    </li>
    <li class="toggle-topbar menu-icon">
<a href="#"><span>Menu</span></a>
    </li>
</ul>
```

完成后的效果如图 19.3 所示。

这样，一个带有响应式功能的头部导航就完成了，其过程和 Bootstrap 十分类似，只是具体的结构和类不同罢了。

图 19.3　小尺寸窗口下的头部导航

19.4　实现响应式版块列表

导航部分完成后，就可以进行主体部分的施工了。首页主要起一个导航的作用，因此版块列表作为相对重要的部分放置在上方。

由于 Foundation 的面板内部无法划分 header 和 body，这里笔者将标题和内容分别放在两个面板中。代码如下：

```
<div class="row">
  <div class="panel index-title">
    <h3>论坛版块</h3>
  </div>
  <div class="panel row" id="forum-index">
    <p class="large-3 medium-4 small-6 column">
      <a href="#"><img src="img/sm.png"> 魔兽世界</a>
    </p>
    <p class="large-3 medium-4 small-6 column">
      <a href="#"><img src="img/sm.png"> 坦克世界</a>
    </p>
    <p class="large-3 medium-4 small-6 column">
      <a href="#"><img src="img/sm.png"> 生活专区</a>
    </p>
    <p class="large-3 medium-4 small-6 column">
      <a href="#"><img src="img/sm.png"> 魔兽争霸</a>
    </p>
    ……
  </div>
</div>
```

这里的关键在于对响应式栅格的应用，由于面板是包裹在 …… 中的，整体内容最大宽度为 62.5rem，如果窗口宽度小于这个值，则为 100%。这里通过在链接外包裹 …… 进行响应式的处理。

Foundation 采用的是 12 列栅格系统，large-3 表明大屏幕下该条目占据 3 个栅格列的宽度，即每行可以有 4 个条目；medium-4 表示中等大小屏幕下该条目占据 4 个栅格列；small-6 表示小屏幕下每个条目占据 6 个栅格列，每行两个条目。

不同窗口大小下的显示效果如图 19.4 所示。

在根据 Foundation 的栅格系统编写完代码后，仍然需要仔细地测试，尤其是几个关键的宽度阈值，比如 640px、320px 等。这里如果不加任何处理，就会发现窗口在 320px 下的显示效果会变成如图 19.5 所示的样子。

图 19.4　帖子列表

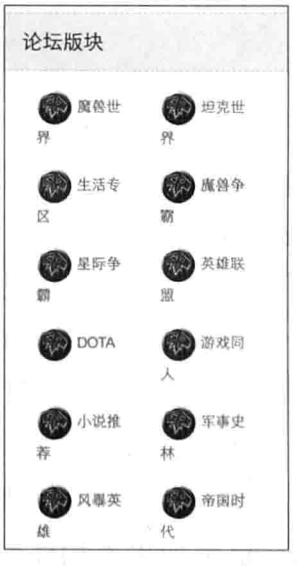


图 19.5　折行的问题

这里笔者通过测试发现，出现阈值是在窗口宽度为 350px 时。这里选择了通过媒介查询的方式，让窗口宽度在 350px 以下时缩小字号来实现：

```
@media only screen and (max-width: 350px) {
```

```
    a{
      font-size: 0.9rem;
    }
  }
```

这样折行的问题就解决了，随着现在手机的发展，分辨率320像素以下的手机基本已经被淘汰了，因此大多数情况下无须再考虑320像素以下的适配情况。

19.5 实现热门帖子推荐

论坛一般会将一些热点的或比较有话题性的帖子推荐到首页，其核心在于吸引人的标题和顶/踩的数量（如果有这个功能的话）。下面就来实现论坛的热门帖子推荐部分。

将标题放在面板中，保持和帖子列表的标题一致，内容则使用表格来展示：

```
<div class="panel index-title">
  <h3>今日头条</h3>
</div>
<table class="large-12 medium-12 small-12">
  <tr>
    <td class="hide-for-small-only"><span class="label">魔兽世界</span></td>
    <td><a href="#"><span class="label alert">精华</span>  战士输出装备问题探讨</a></td>
    <td><a href="#">庇护祝福</a></td>
    <td class="hide-for-small-only">2013 年 12 月 18 日 10:52</td>
  </tr>
  <tr>
    <td class="hide-for-small-only"><span class="label">魔兽世界</span></td>
    <td><a href="#">战士防御装备问题探讨</a></td>
    <td><a href="#">庇护祝福</a></td>
    <td class="hide-for-small-only">2013 年 12 月 18 日 10:52</td>
  </tr>
  .....
</table>
```

如果不固定好表格单元格的宽度，表格本身就是响应式的，但是在小屏幕下显示表格就会出现折很多行的情况，或者像Bootstrap的响应式表格一样，通过显示横向的滚动条来展示完整的内容，但这样用户体验其实并不好。

响应式设计其中很重要一条原则就是在移动设备上尽量展示最重要的内容，具体分析现在这个例子可以发现，对于推荐内容来说，最重要的内容是标题，其次是作者，而该帖子所属版块和更新时间则相对不那么重要。那么完全可以在小屏幕下不展示这些次要内容。

这里可以采用Foundation内置的.hide-for-small-only类，表示仅在小尺寸窗口下隐藏，最终的效果如图19.6所示。

图 19.6　热门帖子推荐

这样，一个论坛的首页就基本完成了，最终效果如图 19.7 所示。

图 19.7　不同尺寸窗口下的论坛首页效果（续）

图 19.7　不同尺寸窗口下的论坛首页效果

19.6　小结

本章主要介绍了如何用 Foundation 框架制作一个简单的响应式论坛首页，一方面是对 Foundation 框架的一次实际应用，另一方面也是对响应式页面设计思想的一次实践。

本例主要应用了 Foundation 框架中的头部导航、面板、栅格系统、表格、显示/隐藏类等组件。展示了如何将各种组件整合成完整页面的过程，其中响应式部分的核心在于理解 Foundation 的栅格系统。

构造响应式页面并不能像做 PC 页面一样完全依赖框架，即使在完全采用框架默认 UI 的情况下，也要注意一些特定的尺寸，如 640px、320px，这些是主流平板/手机的相对分辨率，尤其要保证这些尺寸下的正常显示。

附录 A　网页制作的调试工具及使用

作为网站页面的开发者，各种网页调试工具必不可少。尤其是在遇到浏览器兼容问题时，有一款好用的网页调试工具显得尤为珍贵。为了方便读者学习和调试代码，这里介绍在 3 种最流行的浏览器中的调试方式。合理使用调试工具能快速找到问题来源，提高工作效率。

- Firefox 浏览器中的网页调试工具 Firebug 的使用。
- IE 浏览器中的网页调试工具 IE Developer Toolbar 的使用。
- Chrome 浏览器中的网页调试工具的使用。

A.1　Firefox 浏览器下的页面调试工具及使用

Firebug 是 Firefox 浏览器的一个插件，使用时单击浏览器右下角或右上角的萤火虫图标，如图 A.1 所示。或者按 F12，会自动启用这个插件。

图 A.1　Firebug 的萤火虫图标

如图 A.2 所示是打开后的 Firebug 面板，主要菜单选项有控制台、HTML、CSS、脚本、DOM、网络，共 6 个。

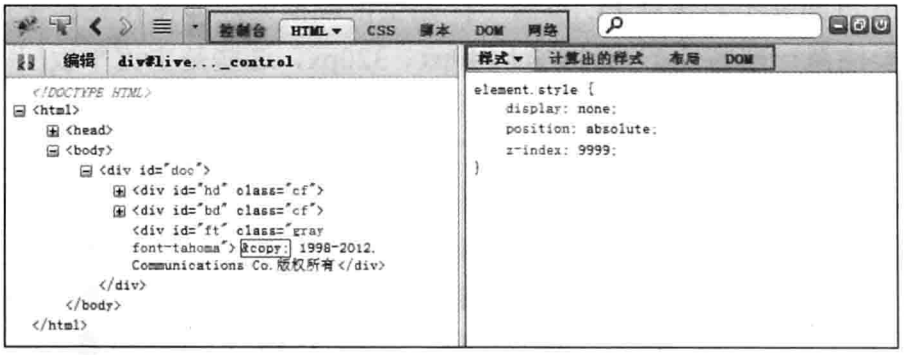


图 A.2　Firebug 面板

HTML 菜单主要用于查看当前页面的 HTML 源代码，单击“HTML”菜单后，Firebug 面板右侧包括 4 个子菜单，分别是：样式、计算出的样式、布局和 DOM。“样式”用于查看每个 HTML 标签对应的 CSS 样式，“计算出的样式”用于查看浏览器自动计算出的选中

的 HTML 标签的样式，这些样式是经过覆盖等过程后最终显示在页面中的样式，“布局”用于显示用户所选中的容器的盒子模型，“DOM”用于 JavaScript 调试。

CSS 菜单用于查看网页加载的所有 CSS 样式表。

如需查看某一部分代码，单击插件上的查看页面元素按钮后，鼠标移到网页上，会出现一个相应的框，当框选中需要查看源代码的元素后点击，在插件区域就显示出当前区域的代码及样式了，可以禁用某些样式，还可以添加新的样式来查找问题所在。

如图 A.3 所示，先单击①位置处的查看页面元素按钮，然后鼠标移动到②的地方，会出现图中所示的蓝色线框，然后点击鼠标，会选中③位置的代码并在④位置出现对该区域生效的所有样式。当鼠标移动到④位置时，会出现一个禁止图标，点击后禁用当前样式，这样便于查找问题所在。

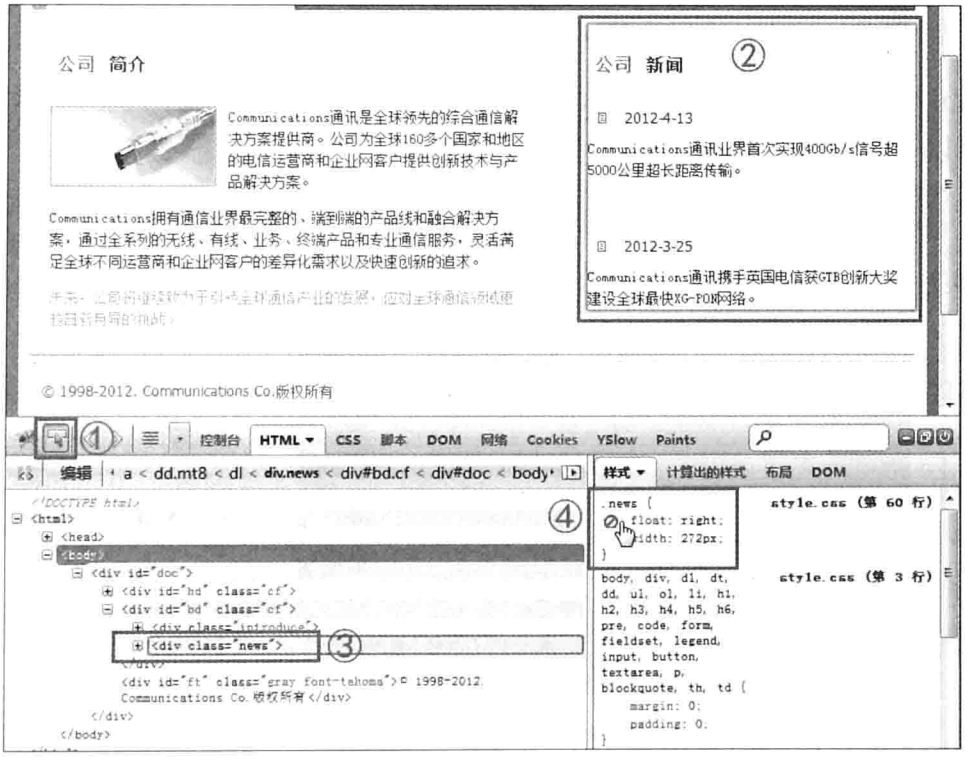


图 A.3　用 Firebug 调试

另外还可以在图 A.4 所示的①位置的任意一行代码后面双击插入新的样式，插入新样式后，浏览器可以实时显示应用新样式后的效果。

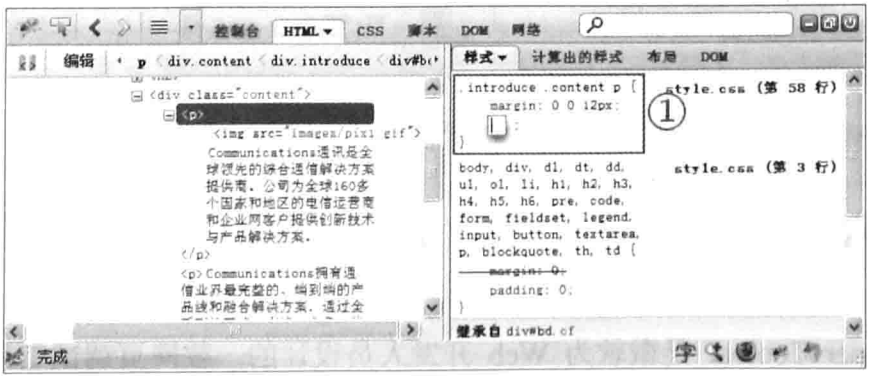


图 A.4　双击插入新的样式

也可以在图 A.5 所示的①位置的任意一行代码的属性对应的值上面单击修改属性值，修改属性值后，浏览器同样可以实时显示应用新属性值后的效果。

图 A.5　单击修改属性值

如图 A.6 所示，在 HTML 菜单下，选中 class 是 content 的容器，单击右侧的“布局”子菜单，可以查看容器 content 的盒子模型。

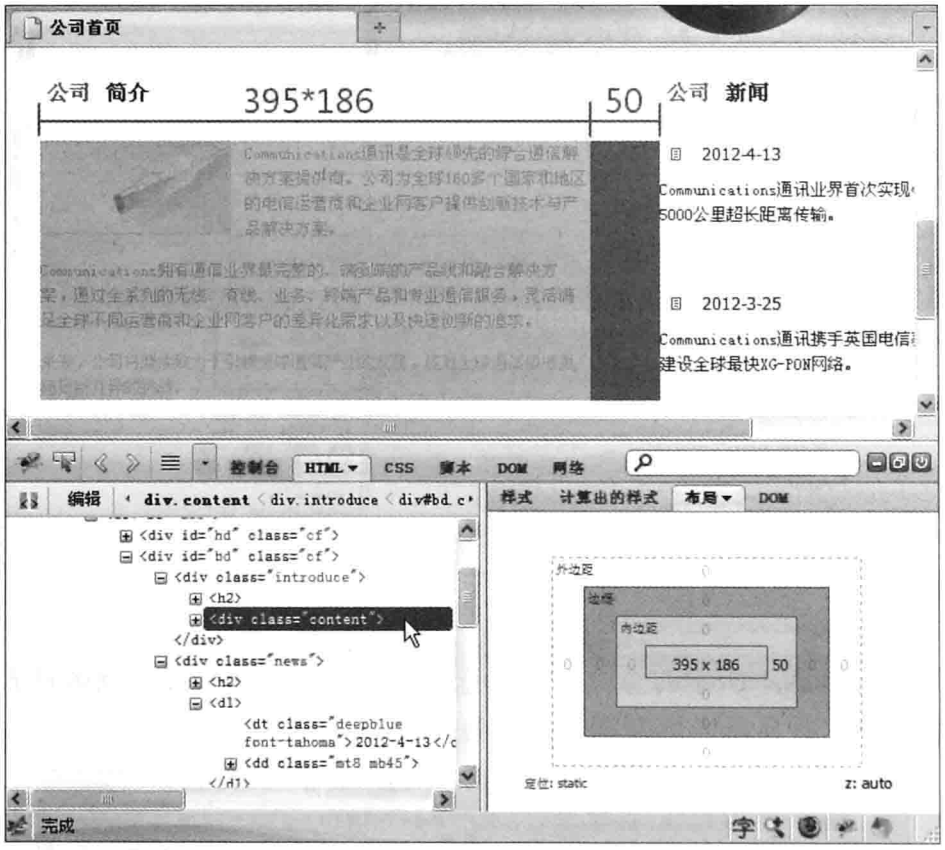


图 A.6　查看容器的盒子模型

A.2　IE 浏览器下的页面调试工具及使用

IE Developer Toolbar 是微软为 Web 开发人员设计的一款网页调试工具，用于 IE 下 HTML 和 CSS 的调试。它的使用方法和火狐的 Firebug 类似。

如果你装的是 IE 8 或 IE 8 以上版本的浏览器，打开浏览器后，按 F12 键，可以打开相似功能的面板，不需要任何安装。如果是 IE7 或以下版本，需要下载 IE Developer Toolbar 并进行安装。

如果是 IE 8 或以上版本，打开浏览器后，按 F12 键或在 IE 头部的工具栏中单击“F12 开发人员工具”。如果是 IE 7 或以下版本，在 IE 头部的工具栏中单击“工具栏”|“浏览器栏”|“IE Developer Toolbar”，如图 A.7 所示，这是 IE8 下的开发人员工具面板。

图 A.7　IE8 的开发人员工具面板

IE 的开发人员工具与 HTML 和 CSS 调试有关的是 HTML 和 CSS 菜单。HTML 菜单的功能与 Firebug 中的 HTML 菜单相似，可以查看页面的 HTML 代码和容器所应用的 CSS 样式。CSS 菜单可以查看网页加载的所有 CSS 样式。

单击 HTML 菜单后，面板的左侧是页面的 HTML 代码，面板右侧包括 4 个子菜单，其中，“样式”可以查看左侧选中容器上所应用的 CSS 样式，“跟踪样式”可以查看左侧选中容器上应用的浏览器计算出的样式，“布局”可以查看左侧选中容器的盒模型。

“样式”中，勾选 CSS 代码前面的多选框，表示应用此样式，取消勾选表示删除该样式。如图 A.8 所示是勾选和取消勾选多选框。单击每行的 CSS 代码可以修改相应的属性或值，如图 A.9 所示。

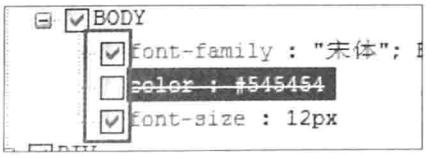


图 A.8　勾选和取消勾选多选框

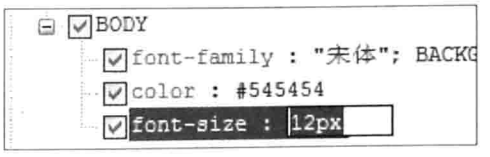


图 A.9　单击每行的 CSS 代码可以修改相应的属性或值

在“属性”子菜单中，单击“添加属性”或“删除属性”按钮，可以为左侧选中的 HTML 容器增加 CSS 样式或删除 CSS 样式。如图 A.10 所示是为页面中的#doc 容器增加样式 background:red 后，页面背景变为红色。

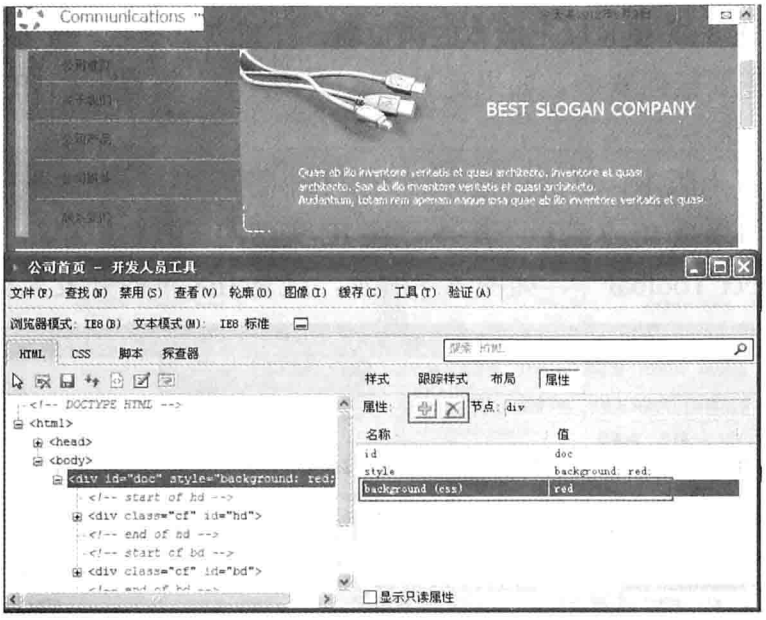


图 A.10　为页面容器#doc 增加样式 background:red

A.3　Chrome 浏览器下的页面调试工具及使用

Chrome 浏览器下的开发者工具，是 Chrome 浏览器自带的一款用于调试网页的工具。打开开发者工具面板的方法有多种，可以直接在页面上右击，然后选择审查元素，或者从 Chrome 的右上角的工具栏中打开，或者按 F12 键打开。如图 A.11 所示是打开的开发者工具面板。在开发者工具面板中，与 HTML 和 CSS 调试有关的菜单主要是 Elements 菜单。这个菜单的作用是查看、编辑页面上的元素，包括 HTML 和 CSS。与 Firebug 面板中的 HTML 菜单的功能和用法非常相似。

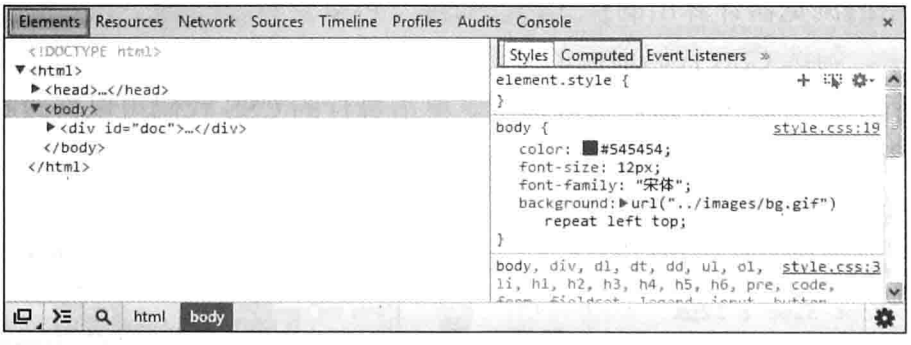


图 A.11　Chrome 的开发者工具面板

单击 Elements 菜单后，面板左侧可以查看页面的 HTML 结构，面板右侧与 HTML 和 CSS 有关的子菜单是 Styles 和 Computed。Styles 子菜单可以对元素的 CSS 进行查看、编辑与修改。Computed 子菜单可以看到左侧面板中选中容器的盒模型和浏览器自动计算出的样式。其中自动计算出的样式是各种设置到该选中容器上的 CSS 值覆盖后的样式。

如图 A.12 所示是企业网站的首页，图中①是页面元素③所对应的 HTML 代码，图中②是页面元素③所对应的 CSS 代码。

图 A.12　企业网站首页中的元素在 Chrome 开发者工具中对应的 HTML 和 CSS

与 Firebug 中的操作相似，可以在图 A.12 所示的②位置的代码后面双击插入新的样式，也可以在 CSS 代码上单击修改属性和值。

如图 A.13 所示是页面元素对应的盒子模型和计算出的样式。

图 A.13　页面元素对应的盒子模型和计算出的样式

附录 B　提升 CSS 的性能和效率

多数情况下，在开发过程中主要需要考虑程序的性能、开发的效率、代码的可维护性这 3 个因素。一般来说，需要根据实际的情况来选择三者的优先级。百度、Google 的首页，性能问题是头等大事，一些需要抢占市场的创新产品则更需要保证开发的效率，对于成熟系统的持续改进，可维护性的重要性就凸显出来了。

- 如何在 CSS 层面提升网站/应用的性能。
- 怎样提升 CSS 开发的效率。
- 如何提高 CSS 代码的可维护性。

B.1　保证 CSS 的性能

CSS 代码中并不包含逻辑和算法，因此无法像后端程序一样通过改进算法、更换语言等方式大幅度地提高运算的速度，但在 CSS 层面上同样有很多技巧可以有效地提升网站的性能。

提升网站性能主要有两种途径：提高 I/O 的效率和提高运算的速度。和网站的其他组成模块一样，提高 CSS 的性能也需要从这两方面着手。

B.1.1　将小图片合并为大图

对于存在多张小图片的页面来说，如果每个小图片都是一张单独图片的话，那么作为一个单独的文件，每增加一张小图片，浏览器在请求页面时就会增加一个 HTTP 连接数

HTTP 连接的建立需要浏览器向服务器发送请求，服务器再返回结果，这是一个双向的过程，而且中间要附带和文件内容无关的请求头信息和响应头信息，增加带宽消耗。这样在小图片较多时会严重影响页面的响应速度。

对于这种情况，可以将小图片合成一张大图片，然后通过 CSS 的 background-position 属性来定位大图片中的小图片。将小图片作为元素的背景显示在页面上，这个技术在国外又被称作 CSS Sprite（国内译作 CSS 精灵）。

以百度新版首页（需要登录百度账号）上的百度电台为例，如图 B.1 所示。每一个图标都是一张图片，选中或鼠标移上后的变色图标又是另一张图片，这个小功能模块至少需要二十几张小图片，通过浏览器工具我们可以拿到百度将小图标合成后的大图，如图 B.2 所示。

图 B.1　百度电台

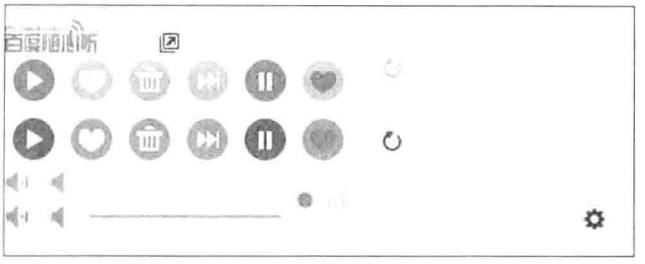


图 B.2　百度电台用到的图标

那么具体实现需要注意些什么呢？来看一下百度电台这部分功能的实现代码：

```
.s-pk-baidu-fm .music-ctrl .pause {
    background-position: 0 -110px;
}
.s-pk-baidu-fm .music-ctrl a {
    background: url('../img/cube_d8237111.png') no-repeat;
    display: block;
    width: 36px;
    height: 36px;
}
```

首先当然是需要使用 background 属性引入这张大图片，然后要确保该元素为块级元素，否则无法指定宽和高，如果像这里使用<a>标签的话，那么就必须使用 display: block;将其强制转换为块级元素。最后使用 background-position 找到大图上该小图标的位置就大功告成了。

过程可以总结为：指定图片→确认块级元素→指定元素宽和高→使用 background-position 定位。

注意：大图片一般不要使用图片合并，单张图片加载过慢反而会影响用户体验，而且还会增加工作量。

B.1.2　合并压缩 CSS 文件

合并压缩 CSS 文件和合并小图标的原理是一样的，合并可以减少 HTTP 连接数，压缩可以减小文件体积，加快传输的过程，减少带宽消耗。

CSS 的文件压缩有很多工具可以使用，一般来说，简单的压缩直接通过在线压缩工具

就可以了，如果经常需要压缩操作则可以使用第 10 章介绍过的 Grunt 工具进行。另外一些开发框架自带压缩合并的功能，比如下面的这个例子：

```
<link href="/assets/application-1893c2c3ab406c707f6c1f29217e73e1.css" media="all" rel="stylesheet"
    type="text/css">
```

这是 Ruby on Rails 框架自带的合并压缩工具处理过的效果，所有的 CSS 文件被合并压缩为这一个文件，文件名中-后面的一串字符是一个哈希校验，如果文件发生改变，这串字符就会发生变化，那么浏览器端之前的缓存就会失效，从服务端获取新的样式表，这样就防止了网站修改了样式但用户感受不到的情况。

B.1.3 使用缩写

使用缩写可以缩减代码量，减小 CSS 文件的体积，从而降低流量的消耗和数据传输所需的时间，不过总的来说性能上提升并不明显，但是从维护性角度上来说，使用缩写的意义更大一些。

下面总结一些常用的缩写规则：

- 边框(border)

边框的属性如下：

```
border-width:1px;
border-style:solid;
border-color:#000;
```

可以缩写为一句：

```
border:1px solid #000;
```

语法是 border:width style color;。

- 背景(Backgrounds)

背景的属性如下：

```
background-color:#f00;
background-image:url(background.gif);
background-repeat:no-repeat;
background-attachment:fixed;
background-position:0 0;
```

可以缩写为一句：

```
background:#f00 url(background.gif) no-repeat fixed 0 0;
```

语法是 background:color image repeat attachment position;。

注意：可以省略其中一个或多个属性值，如果省略，该属性值将用浏览器的默认值。

- 字体(fonts)

字体的属性如下：

```
font-style:italic;
font-variant:small-caps;
font-weight:bold;
```

```
font-size:1em;
line-height:140%;
font-family:"Lucida Grande",sans-serif;
```

可以缩写为：

```
font:italic small-caps bold 1em/140% "Lucida Grande",sans-serif;
```

注意：缩写字体定义，至少要定义 font-size 和 font-family 两个值。

B.1.4　尽可能使用 CSS 3 效果代替图片

由于现代计算机技术的进步，硬件运算能力和浏览器的渲染能力都大大加强，虽然使用 CSS 3 效果会增加浏览器的运算，但是网页应用的瓶颈始终都是在 I/O 方面，越少的网络请求就意味着越快的速度和越好的用户体验。对多数网页来说，图片占据着大部分的网页流量。

对于摄影、图画来说自然需要使用图片，但是按钮、图标、渐变背景等功能性的元素使用图片不仅不方便，还大大提高了网站的带宽成本，拖慢了加载速度。因此在一些无须过度考虑浏览器兼容性的场合，使用 CSS 3 效果代替图片是非常好的选择。例如图 B.3 和图 B.4 的两个背景效果就可以完全使用 CSS 3 效果来实现。

图 B.3　CSS 3 制作的文字光晕效果

图 B.4　CSS 3 制作的文字效果

B.1.5　删除无用的样式

保留一些无用的 CSS 代码虽然在功能和效果上看不出来什么，但是最好还是要把它们移除掉，一方面这些代码会占据更多的网络流量，一方面浏览器同样要对这些无用的代码

进行解析，存在性能上的影响。同时在可维护性角度上讲，遗留的无用样式还可能会在以后的修改中造成样式冲突。因此修改代码后检查并删除无用的 CSS 代码是一个好习惯。

在应用像 Bootstrap 这样的大型 CSS 框架时开发者们往往会引入整个框架，并将其应用到生产环境上，这其实是一个偷懒的做法。在开发阶段由于需求尚不明确，可能随时会用到框架中的某个功能，引入整个框架是个好的选择，但是在生产环境下，框架产生的大量冗余代码确实会对性能产生影响。

目前一个比较好的解决方案是使用这些框架的定制化工具，在配置文件中注释掉没有用到的组件，目前流行的一些框架都有比较完善的定制化工具，关于 Bootstrap 和 Foundation 的定制化可以参见本书的第 12 章和第 13 章。

B.1.7 了解 CSS 的解析规则

浏览器解析 CSS 的过程，实际上是一个将 CSS 选择器和 HTML 元素匹配的过程，提高浏览器解析的效率是性能提高的又一个环节。

优化选择器的原则是尽量避免需要消耗更多匹配时间的选择器。因此我们需要了解 CSS 选择器匹配的原理，例如：

```
#header > a {font-weight:blod;}
```

大多数人的阅读习惯是从左到右，但 CSS 选择器却是从右到左进行规则匹配。浏览器必须遍历页面中所有的 a 元素并且确定其父元素的 id 是否为 header。

如果把例子的子选择器改为后代选择器则会开销更多，例如：

```
#header  a {font-weight:blod;}
```

这样在遍历页面中所有 a 元素后还需向其上级遍历直到根节点。

理解了 CSS 选择器从右到左匹配的机制后，可以知道选择器中最右边的规则往往决定了浏览器继续左移匹配的工作量，因此最右边的选择器又被称为关键选择器。

了解了 CSS 的解析规则之后，就可以根据规则来处理一些性能敏感的前端模块，不过多数情况下，性能问题没有那么敏感，不应该拒绝上下文选择器带来的好处。

B.1.8 尽量避免使用通配符和正则表达式

通配选择器使用*表示，可匹配文档中的每一个元素。如下例规则将所有元素的字体大小设置为 20px：

```
* { font-size:20px;}
```

通配选择器作用于所有的元素，如规则最右边为通配符：

```
.selected * {color: red;}
```

浏览器匹配文档中所有的元素后分别向上逐级匹配 class 为 selected 的元素，直到文档的根节点，因此其匹配开销是非常大的，通常比开销最小的 ID 选择器高出 1~3 个数量级，所以应避免使用关键选择器是通配选择器的规则。

CSS3 添加了复杂的属性选择器，可以通过类似正则表达式的方式对元素的属性值进行匹配。正则表达式匹配会比基于类别的匹配会慢很多。大部分情况下应尽量避免使用*=、

|=、^=、$=、和~=语法的属性选择器。

B.2　提高 CSS 开发的效率

在开发工作中，开发效率也是必须考虑的因素，在很多时候其重要性甚至在性能之上。有时开发效率会和程序的执行效率产生冲突，最显著的就是在框架的使用上，不论前端还是后端，应用框架大都没有自己实现的程序性能好。

在大多数情况下，我们需要做的是在保证开发效率的前提下再考虑尽量提高执行效率，因为绝大多数网站的访问量是不足以碰到性能瓶颈的。

除了提高自身的熟练程度以外，提高开发效率主要在于合理的使用工具。

B.2.1　合理选择开发框架

提高效率的最佳方式当然是直接复用，使用开发框架可以直接复用已经封装好的效果，如果全盘采用框架本身 UI 的话，甚至可以做到不写一行 CSS 就完成页面的制作。因此根据自身的需求合理选择开发框架十分的重要。

现在对几种常用的需求进行框架选择的分析。

1．单一的产品展示类网站

一个新产品出现，一般企业都会做一个专题页面对产品进行介绍，这类页面一般以追求比较炫酷的视觉效果为主。对于前端开发来说，页面本身的结构是比较简单的，主要需要根据网站兼容性的要求，做一些吸引人的动画效果，由于 IE 6~8 是完全不兼容 CSS 3 的，因此主要的工作在于一些 JavaScript 或者 Flash 的动画上。一个典型的产品展示页面如图 B.5 所示。

图 B.5　一个典型的产品展示页面

这类页面结构一般比较简单，而且更多体现的是个性化，因此可以考虑不依赖框架，

或者只使用像 960gs 这样没有任何 UI 的简单框架进行开发，如果引入像 Bootstrap 这类重型框架，一方面很多功能用不上，反而增大 CSS 文件的体积，影响性能；另一方面 Bootstrap 这类框架对 UI 的一些预定义可能会和自定义的 UI 产生冲突，处理冲突反而会增大工作量。

2．复杂功能网站

像新浪、淘宝这类“历史悠久”的网站由于出现的早，那时候前端并没有现在这么多的开源框架以供使用。不过如果在现在这个时代，要做一个功能较复杂的网站可能就不需要花那么多功夫自己写了。

这里以一个项目管理工具“风车”为例，如图 B.6 所示，UI 看起来和 Bootstrap 完全没有关系对不对？不过通过浏览器工具打开 CSS 代码一看就可以发现，里面的 class 命名规则和 Bootstrap 是一致的。

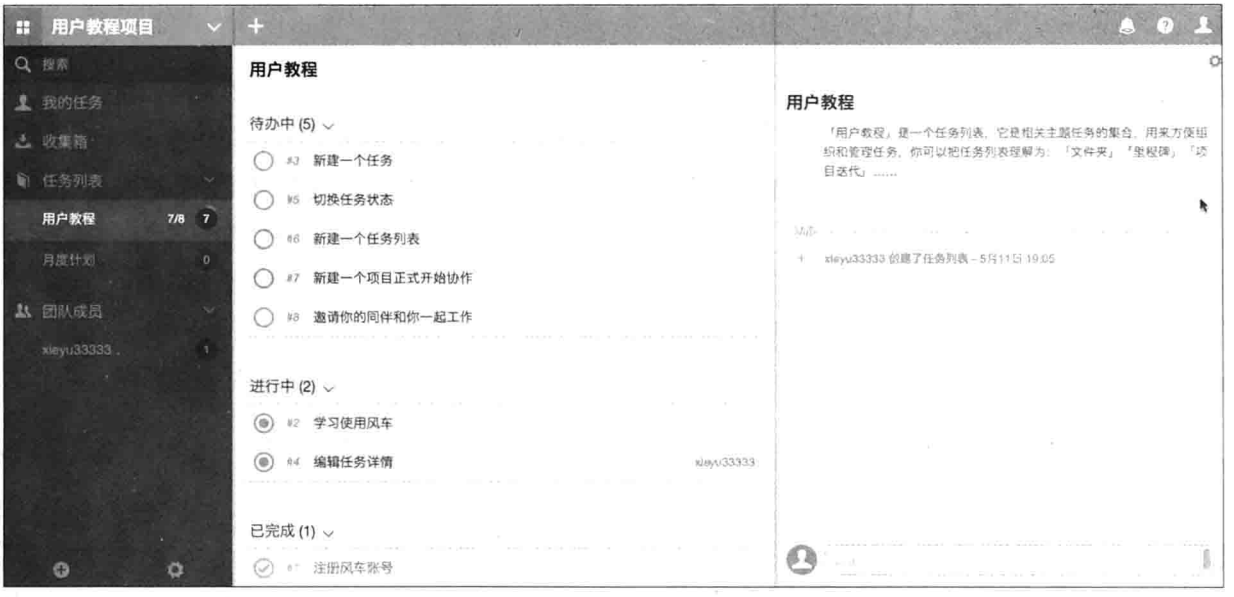


图 B.6　风车项目管理工具

实际上“风车”采用的是一个 Bootstrap 的皮肤，Bootstrap 皮肤的使用方法完全和原版的 Bootstrap 一样，只不过重新设计了 UI 元素，这样只要使用者熟悉 Bootstrap 的使用，就可以通过不同的皮肤制作出 UI 风格完全不同的网站。

Bootstrap 非常适合于复杂页面的制作，一方面大多数的功能都可以用得上，另一方面各色各样 Bootstrap 皮肤的存在使得开发者无须担心缺乏个性化的问题，而且还可以根据实际情况自己开发基于 Bootstrap 的皮肤。

3．简单功能网站

对于一些简单功能的网站，不想采用 Bootstrap 这类重型框架，但又希望快速完成功能的话，笔者推荐采用 Pure 等轻量级的框架（如图 B.7 所示），既可以帮助开发者提高效率，又无须引入过多的功能。

4．后台管理类网站

这类站点一方面无须考虑 UI 的个性化，一方面也可以较少的考虑浏览器的兼容性（可以建议或强制使用新式浏览器），因此目前来说，使用 Bootstrap 是最合适的，一方面 Bootstrap

拥有相当全面的功能，涵盖了几乎所有的常用组件，另一方面有最多的相关资料和基于其结构设计的皮肤。如图 B.8 所示是使用 Bootstrap 3 制作的后台管理界面。

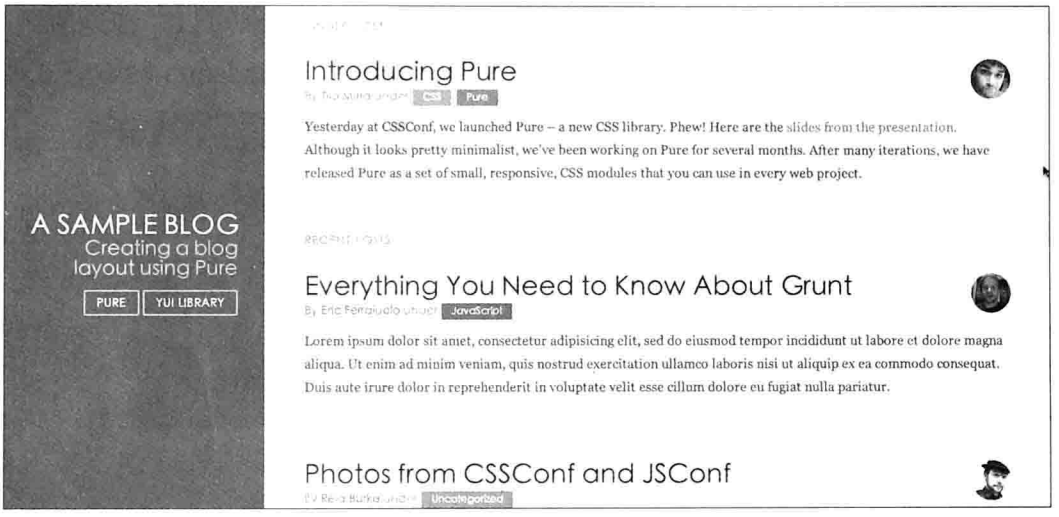


图 B.7　使用 Pure 开发的博客系统

图 B.8　使用 Bootstrap 3 制作的后台管理界面

5. 手机专用页面

对于专门为手机开发的 WebAPP 或者 Web 和原生 UI 混合的 APP，可以选用像 Ratchet（如图 B.9 所示）、ionic、GMU 等专门针对手机的开发框架，里面有很多模拟原生 UI 的组件。如果不追求模拟原生 UI，手机页面也比较简单的话，也可以选择不依赖任何框架。

图 B.9　使用 Ratchet 制作的手机页面

6. 需要兼容 PC、平板、手机的页面

对于这种需求（如图 B.10 所示），如果需要框架内置的 UI 和 JavaScript 控件可以采用 Bootstrap 3、Foundation 等框架。如果只是需要处理响应式，那么可以单纯地应用响应式栅格系统，比如 Simple Grid。或者也可以单独下载 Bootstrap 或 Foundation 的响应式栅格组件。

图 B.10　Simple Grid 首页

图 B.2.2　使用 CSS 预处理语言

在本书的第 11 章介绍了 LESS 和 SCSS 等 CSS 预处理语言，熟练使用 CSS 预处理语言可以有效地提高开发的效率，并且可以方便地进行模块化的组织。如图 B.11 所示是 Koala 图形化预处理语言编译工具。

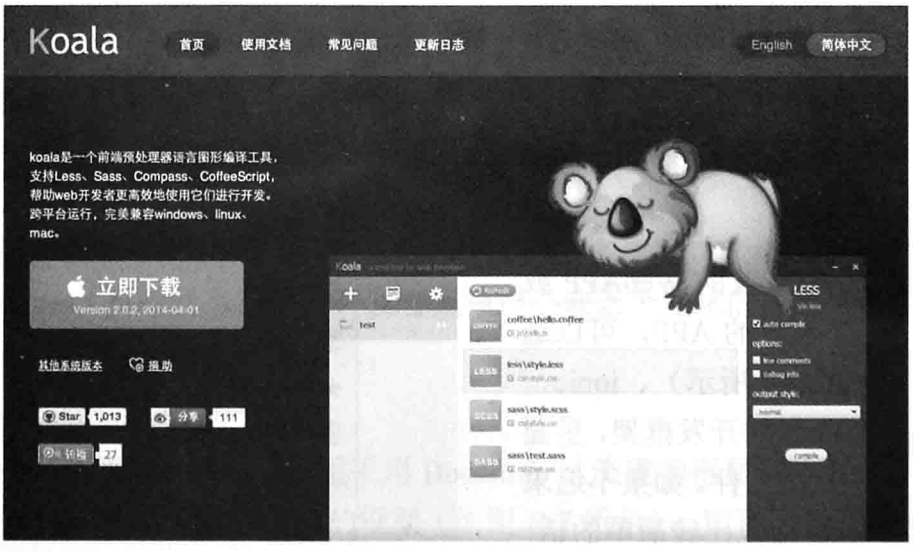


图 B.11　Koala 图形化预处理语言编译工具

一般来说，笔者建议初学 CSS 预处理语言的读者选择 LESS，一方面常用的功能 LESS 都有，另一方面 LESS 的资料更多，各种平台下的辅助工具也更多，在 Windows、MAC、Linux 下都有不错的可视化客户端可供选择。

而 SASS 功能上比 LESS 强一些，并且有 Compass 工具辅助，可以进一步缩减代码量，但是需要 Ruby 环境，而且相对资料比较少，图形化工具的选择也不多，大多数开发者还是通过命令行进行 SASS 的编译。不过如果是 Ruby on Rails 的开发者建议使用 SASS，因为框架内自带了 SASS 编译功能。

图 B.2.3　使用编辑工具的特性和插件提高工作效率

开发者们根据自己的喜好和习惯使用各种不同的代码编辑工具，比如 Textmate、Sublime、Notepad++这类简单的代码编辑器，还有 WebStorm 这类 IDE 工具，以及 Coda、Dreamweaver 这样专门用来做页面的开发工具。

对于代码编辑器来说，一般只有常用语言的代码高亮功能，但是可以通过添加各种插件来实现自动补全、代码简写等功能，而且还可以通过插件来支持一些普及度不高的语言的代码高亮，比如 LESS 和 SASS。

还有像 Zen coding 这样的插件可以大大提高编码速度，比如图 B.12 中的例子。

```
1  div#page>div.logo+ul#navigation>li*5>a
2
3  <div id="page">
4          <div class="logo"></div>
5          <ul id="navigation">
6                  <li><a href=""></a></li>
7                  <li><a href=""></a></li>
8                  <li><a href=""></a></li>
9                  <li><a href=""></a></li>
10                 <li><a href=""></a></li>
11         </ul>
12 </div>
```

图 B.12　Zen coding

如图 B.12，白字部分是 Zen coding 语法，下面的代码是生成的 HTML。

像 WebStorm 和 Dreamweaver 本身的功能就很强大，学习挖掘它们的功能，将其熟练应用在自己的工作中也是提高效率的好办法。

图 B.3　CSS 代码的可维护性

这个世界上不存在一次成型的完美代码，总会出现 bug 或者需求变更，因此代码的可读性、可维护性也非常重要。这样不论是以后修改、复用还是重构都更为方便。

图 B.3.1　合理的语法规范

大多数语言对于同样的实现都存在不同的写法，在团队内部统一采用一种清晰的语法规范有助于提高代码的可读性和可维护性。

不同的开发团队可能有不同的语法规范，不过有一些规范是受到普遍认可的，本节将为读者介绍一些 CSS 代码编写中需要注意的地方。

1. 语义化的命名

选择器要根据其功能的具体含义进行命名，避免出现 div1、div2 这样无意义的命名，这样在阅读代码时就能一目了然的知道这段代码的作用，同时也更方便记忆。

具体的命名规范，不同的团队不尽相同，但是遵循语义这一点是公认的。

2. 命名时避免使用下画线连接单词

有时候一个单词无法做到清晰的表意，可能需要多个单词连接，在 CSS 中推荐使用中横线连接单词，不推荐使用下画线。

```
/*推荐*/
main-title{
  font-size: 40px;
}

/*不推荐*/
main_title{
  font-size: 40px;
}
maintitle{
  font-size: 40px;
}
```

不推荐使用下划线基于两点原因：

- 浏览器兼容问题（比如使用_tips 的选择器命名，在 IE 6 中是无效的）。
- 能良好区分 JavaScript 变量命名，JavaScript 变量名不支持中横线。

3. 避免滥用 class

为每一个需要添加样式的元素定义一个 class 不是一个好主意，首先为这些 class 起名字本身就是一个伤脑筋的工作，其次在后续维护中也很难记住名目繁多的 class 名。

解决方法是采用子元素选择器和后代元素选择器等，利用上下文来定义样式，这样既无须为命名伤脑筋，还可以缩减一点代码量。比如图 B.13 中的例子。

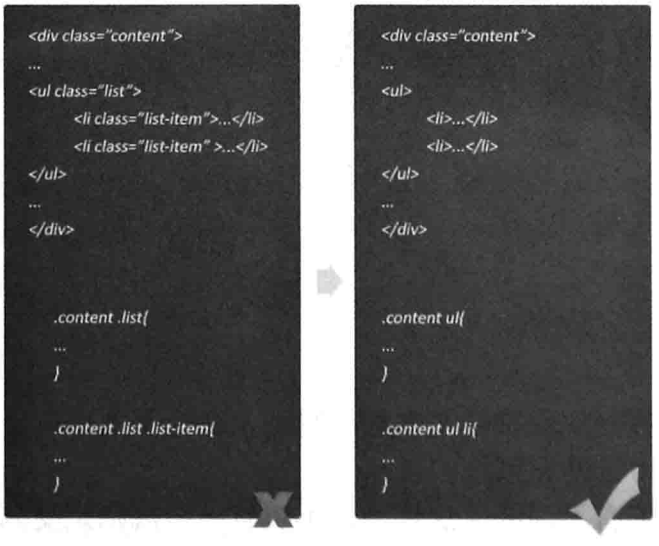


图 B.13　class 对比

4．合并重复的样式

比如元素 A、元素 B、元素 C，除了颜色以外其他的样式都一样，那么不应为 A、B、C 分别定义样式，而是将它们之间相同的部分抽取出来定义一个 class，通过为元素添加多个 class 来实现，例如：

```
/*不推荐*/
.a{
   width:20px;
   height:20px;
   color:red;
}
.b{
   width:20px;
   height:20px;
   color:black;
}
.c{
   width:20px;
   height:20px;
   color:blue;
}

<div class="a"></a>
<div class="b"></a>
<div class="c"></a>

/*推荐*/
.commen{
   width:20px;
   height:20px;
}
.a{
   color:red;
}
.b{
   color:black;
}
.c{
   color:blue;
}

<div class="commen a"></a>
<div class="commen b"></a>
<div class="commen c"></a>
```

B.3.2 使用 Grunt 等自动化工具

本书在第 10 章曾经介绍了自动化工具 Grunt，它可以帮助我们自动化地进行编译、压缩、合并等工作。使用自动化工具不仅可以提高效率，而且规避了手动操作可能带来的失误风险。

比如在图 B.1.5 中提到的按需加载框架中的组件，如果没有自动化的工具，基本是不可能在工作中实现的。可以想象一下手动把十几个 CSS 文件拷贝到一起然后压缩的情景，而使用 Grunt，只需要根据需要改几行配置，执行一个命令就大功告成了。

总而言之，在代码的维护过程中，尽量使用自动化工具，避免繁琐的手动流程可以有效地降低出错的概率。

B.4 CSS 高效开发经验总结

前面主要从性能、效率、可维护性 3 个方面介绍了如何高效开发 CSS 的一些经验。

从性能角度讲，有两个方面可以提高性能：网络 I/O 方面和浏览器对 CSS 的解析效率方面。

从提升网络 I/O 方面的效能来说，主要有降低 HTTP 连接数和降低文件大小两种手段：

- 合并小图片和合并 CSS/JS 文件可以有效降低 HTTP 连接数，效果明显。
- 在可以不考虑 IE6-8 的场合，使用 CSS 3 效果代替图片同样可以起到减少 HTTP 连接的作用。
- 使用缩写、删除无用样式可以缩减 CSS 文件的大小，降低流量消耗。

从提高 CSS 解析效率角度来说：

- 慎用通配符选择器和正则表达选择器。
- 了解 CSS 的解析过程，在性能敏感时使用合适的选择器。

在提高开发效率方面，主要在于选取合适的框架和工具：

- 合理选用开发框架可以让很多代码不必亲自动手。
- 使用 LESS 或 SASS 等预处理语言可以有效缩减代码量，同时方便模块化组织代码。
- 深入了解开发工具的使用，选用一些提高生产力的编辑器插件。

在可维护性方面，需要注意合理的编码规范和自动化工具的使用：

- 规范方面需要注意命名的语义性和代码结构的合理性，不要滥用 class，不要过多的重复代码。
- 尽量把代码的压缩、合并、发布等动作自动化，通过配置文件进行管理，减少出错的可能性。