第 1 章

◂ Bootstrap从何而来 ▸

Bootstrap 框架源自于 Twitter，是基于 HTML、CSS 和 JavaScript 构建的、目前最受欢迎的前端框架。Bootstrap 简洁灵活，提供很多响应式设计模板，也支持跨平台操作，是目前 Web 开发和移动 Web 开发人员的首选框架。本章将介绍 Bootstrap 框架的基础内容，帮助读者认识 Bootstrap 框架。

本章主要内容包括：

- 了解响应式设计的概念
- 掌握响应式设计的设计原则
- 了解 Bootstrap 的设计目标

1.1 初识 Bootstrap

本节首先介绍 Bootstrap 框架从何而来，帮助读者了解 Bootstrap 框架的历史及特点。

Bootstrap 框架是由 Twitter 设计师 Mark Otto 和 Jacob Thornton 共同开发出来的，准确地讲它就是一个 HTML+CSS 的前端框架。Bootstrap 框架提供了优雅的 HTML 和 CSS 规范，是由动态 CSS 语言写成，完美地解决了页面响应式设计的难点。因此，Bootstrap 框架推出后颇受欢迎，已经成为 Web 开发人员实际意义上的设计规范，目前许多优秀的开源前端框架均是基于 Bootstrap 源码性能优化而来的。

Bootstrap 框架中包含了丰富的 Web 组件，设计人员可以使用这些组件快速搭建出一个界面美观、功能完备的网站。同时，Bootstrap 框架还自带了多个 jQuery 功能插件，这些功能插件为 Bootstrap 框架中的组件提供了功能支持。设计开发人员可以对 Bootstrap 框架中所有的 CSS 变量进行修改，并根据实际需求来裁剪出满足项目需要的代码。

Bootstrap 框架的最新版本是完全向下兼容的。

1.2 什么是响应式设计

页面可以根据用户的终端设备尺寸或浏览器的窗口尺寸来自动地进行布局调整，这就是响应式布局设计。目前，用户终端设备可谓是种类繁多、琳琅满目。我们细想一下，从台式显示器到笔记本电脑屏幕，从平板电脑再到手机界面，同一类设备但不同厂家的产品屏幕尺寸不尽相同，设计起页面来，难度较大。响应式布局就是为了解决这个问题而诞生的，且目前已经是主流的设计方式了。

图 1.1 就是一个直观的响应式布局设计示意图，图中演示了同一个页面在台式显示器、iPad 及 iPhone 三种设备屏幕尺寸上呈现的效果。

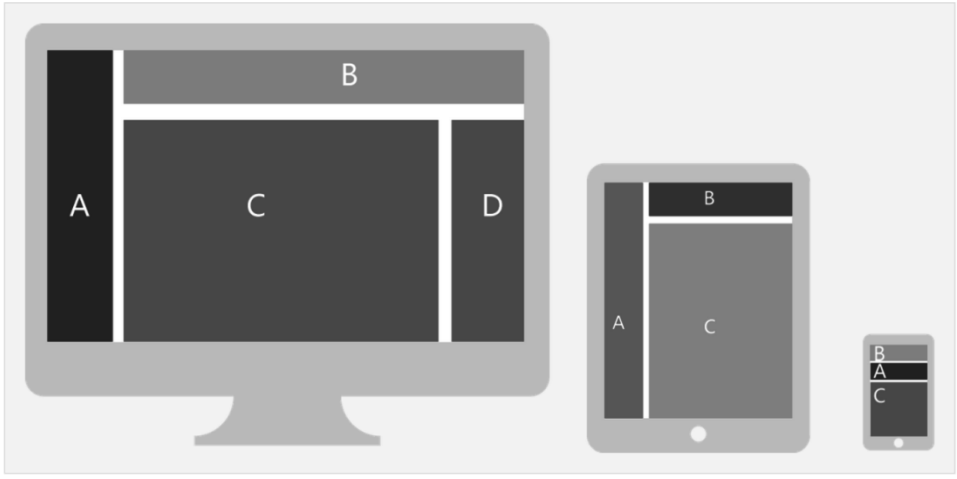

图 1.1　响应式布局设计示意图

下面再来看图 1.2 和图 1.3，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。

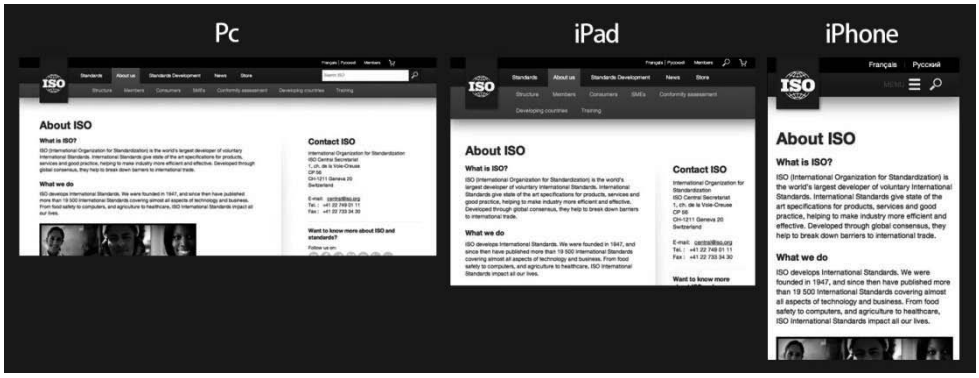

图 1.2　响应式设计案例 1

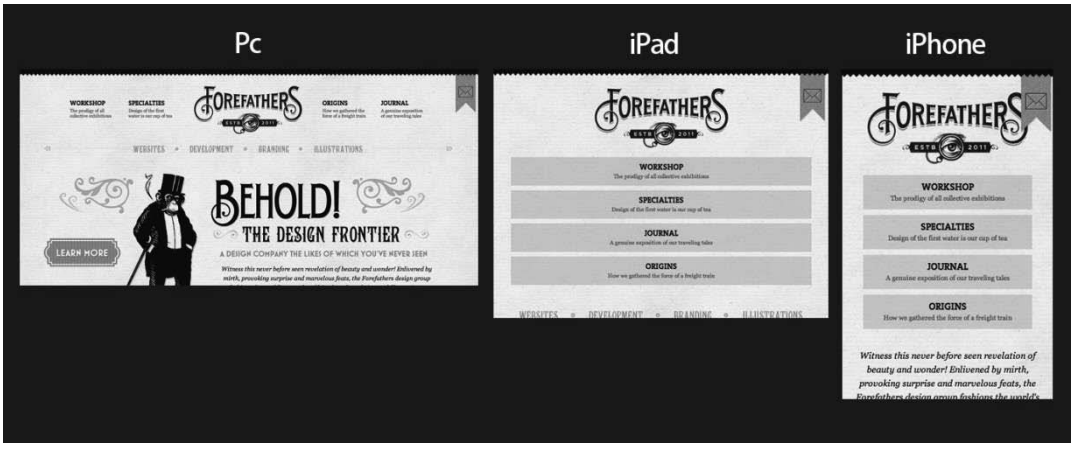

图 1.3　响应式设计案例 2

近年来，移动互联网发展势头非常迅猛，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可能（相对于曾经的 WAP 手机站来说）。响应式设计越来越流行，Bootstrap 框架就是因此应运而生的。可以说，Bootstrap 框架的出现解决了之前一直困扰设计人员的终端设备屏幕尺寸的兼容性问题，是广大前端设计开发人员的福音。

1.3 响应式设计四大原则

本节介绍一下响应式设计所需要遵循的一些基本原则，归纳起来一共有 4 个方面，具体如下：

- 移动优先还是 PC 优先
- 内容流
- 位图还是矢量图
- 相对单位还是固定单位

1.3.1　移动优先还是 PC 优先

随着移动互联网的发展，很多小型创业企业甚至没有了自己的网站，只有一个 APP 应用。在这个时代，网站项目是从小屏幕入手过渡到大屏幕（移动优先），还是从大屏幕入手过渡到小屏幕（PC 优先），成为企业考虑的首要问题。

传统的大企业改造型网站，大部分是从大屏幕逐步过渡到小屏幕，而且在过渡到小屏幕时会碰到一些额外的限制，比如没法在第一页面显示更多的内容，要更简洁，具体到要放哪些标签都是需要决策的内容。

在新兴的创业公司中，通常情况下都会从两方面同时着手，所以具体哪个优先还是要看哪种方式最适合。

1.3.2 内容流

随着移动屏幕尺寸越来越小，内容所占的垂直空间越来越多。也就是说，内容会向下方延伸，这被称为“内容流”。早先的 Web 设计师习惯了使用像素和点来设计页面，可能会觉得这有点难以掌握。不过好在它很简单，多多练习就习惯了。图 1.4 展示了两种设计状态下页面内容变宽后的效果。

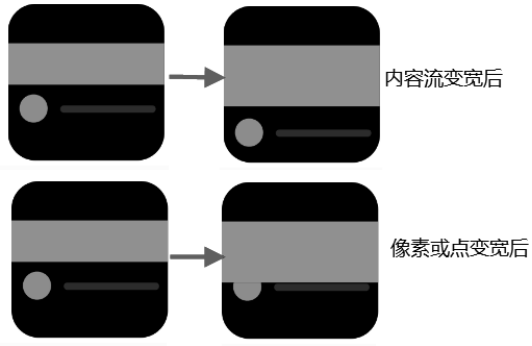

图 1.4 内容流对比

1.3.3 位图还是矢量图

我们知道，当一张图片被放大后就会出现比较“虚”的情况，这种图是位图，而放大后不变“虚”的则是矢量图。先来了解一下两者的概念。

矢量图使用线段和曲线描述图像，所以称为矢量，同时图形也包含了色彩和位置信息。

位图使用像素（一格一格的小点）来描述图像，计算机屏幕其实就是一张包含大量像素点的网格。在位图中，图像由每一个网格中的像素点的位置和色彩值来决定，每一点的色彩是固定的，所以放大后观看图像时，每一个小点看上去就像是一个马赛克，这就是我们常说的“虚”。

在响应式设计中，图标或图像都会涉及这个问题。如果我们的图标有很多细节，并且应用了很多华丽的效果，就用位图；否则，考虑使用矢量图。如果是位图，使用 jpg、png 或 gif；矢量图则最好使用 SVG 或图标字体。位图和矢量图两者各有利弊。矢量图通常比较小，很适合移动端来展示，但部分比较老的浏览器可能不支持矢量图。另外，有些图标有很多曲线，可能导致它的大小比位图还大，所以需要根据实际情况明智取舍。

1.3.4 相对单位还是固定单位

对于设计师而言，我们的设计对象可能是桌面屏幕，也可能是移动端屏幕，或者介于两者之间的任意屏幕类型。不同的终端像素密度也会不同，所以我们需要使用灵活可变且能够适应各种设备的单位。

传统的设计单位有 px、pt、cm 等，但它们都是固定单位，没法实现跨平台展示。那么，在这种情况下，百分比等相对单位就到了发挥作用的时候了。使用百分比时，我们所说的宽度 50%是

表示宽度占屏幕大小（或者叫视区，即所打开浏览器窗口的大小）的一半，如图 1.5 所示。

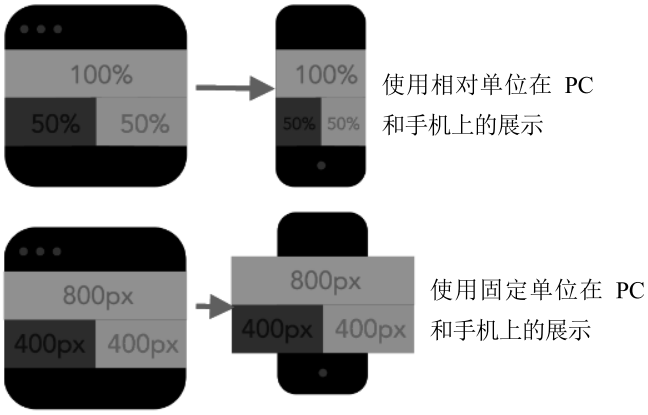

图 1.5　固定单位和相对单位对比

1.4 Bootstrap 设计目标

本节介绍一下 Bootstrap 设计目标，包括目标对象、浏览器支持和响应式设计等内容。

1.4.1　优先针对移动设备

其实自 Bootstrap 3 版本开始，Bootstrap 框架包含了贯穿于整个库的移动设备优先的样式。Bootstrap 4 版本默认的 CSS 本身就是对移动设备优先进行友好支持。

Bootstrap 框架的设计目标从优先支持桌面设备转变到优先支持移动设备，这实际上是一个非常及时的、适应 Web 设计发展方向的转变。因为，现在越来越多的用户在使用移动设备，而且使用的频次也越来越高。

1.4.2　浏览器支持

Bootstrap 框架几乎支持目前市面上所有的主流浏览器。下面列举几款浏览器：

- Internet Explorer（IE）
- Google Chrome
- Firefox
- Opera
- Safari
- Microsoft Edge

1.4.3 响应式设计

Bootstrap 框架的响应式 CSS 能够自适应于台式机、平板电脑和手机等多种设备终端。下面我们简单归纳一下 Bootstrap 框架的特点：

- 为开发人员创建接口提供了一个简洁统一的解决方案。
- 包含了功能强大的内置组件，易于定制。
- 提供了基于 Web 开发的定制。
- 最重要的是框架是开源的，设计人员可以通过修改框架源码满足特定需求。

1.5 本章小结

本章主要介绍了 Bootstrap 框架的基础知识、其与响应式布局的关系以及 Bootstrap 框架的设计目标，希望能够提高读者使用 Bootstrap 框架进行 Web 开发的兴趣。

第 2 章

◂ Bootstrap开发环境 ▸

从本章开始，我们就带领读者真正进入 Bootstrap 开发的实战阶段了，要实战就要了解 Bootstrap 的开发环境、框架结构、布局方式，以及如何调用 Bootstrap 样式和组件等方面的基本内容。本章的目的就是从 Bootstrap 项目的整体大局上入手，让读者对它的使用有个预先判断。

本章主要内容包括：

- 下载 Bootstrap 的开发包
- 初次在项目中使用 Bootstrap
- 在项目中调用 Bootstrap 的各种组成元素
- 进行第一次 Bootstrap 项目实战

2.1 Bootstrap 开发环境概述

本节先介绍 Bootstrap 框架的开发环境，包括如何下载 Bootstrap 开发包，如何在网站中使用 Bootstrap 框架，如何调用 Bootstrap 样式、组件和 JS 特效等方面的内容。对于全书的内容来讲，本节的内容是基础部分。

2.1.1 下载 Bootstrap 开发包

Bootstrap 的官方网站地址是 https://getbootstrap.com/，界面如图 2.1 所示。可以在官网下载最新的版本和详细的使用说明文档。

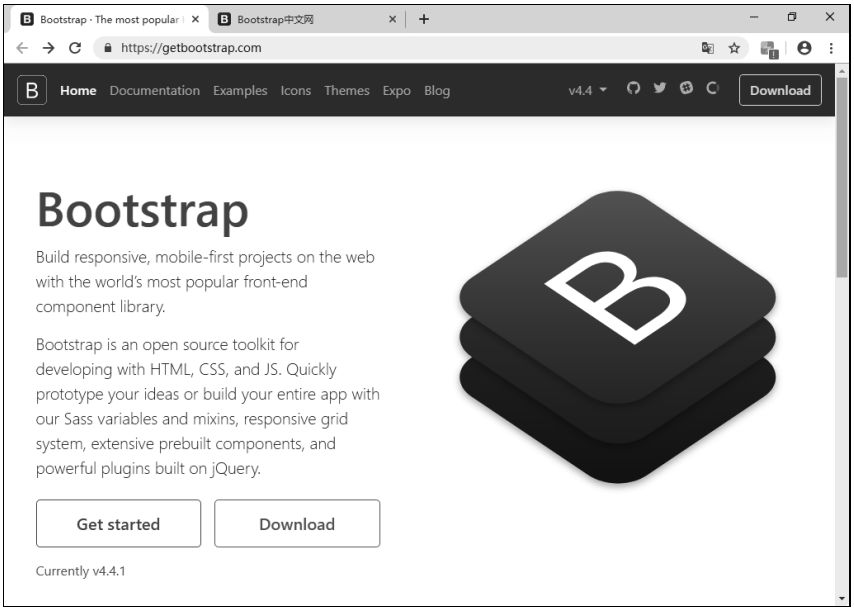

图 2.1　Bootstrap 官网

目前，国内也有不错的 Bootstrap 中文网站，例如“Bootstrap 中文网”，可以通过访问网址 https://www.bootcss.com/进行浏览，如图 2.2 所示。

图 2.2　Bootstrap 中文网

在图 2.1 所示的 Bootstrap 官网中，可以单击 Download 按钮转到下载页面，如图 2.3 所示。

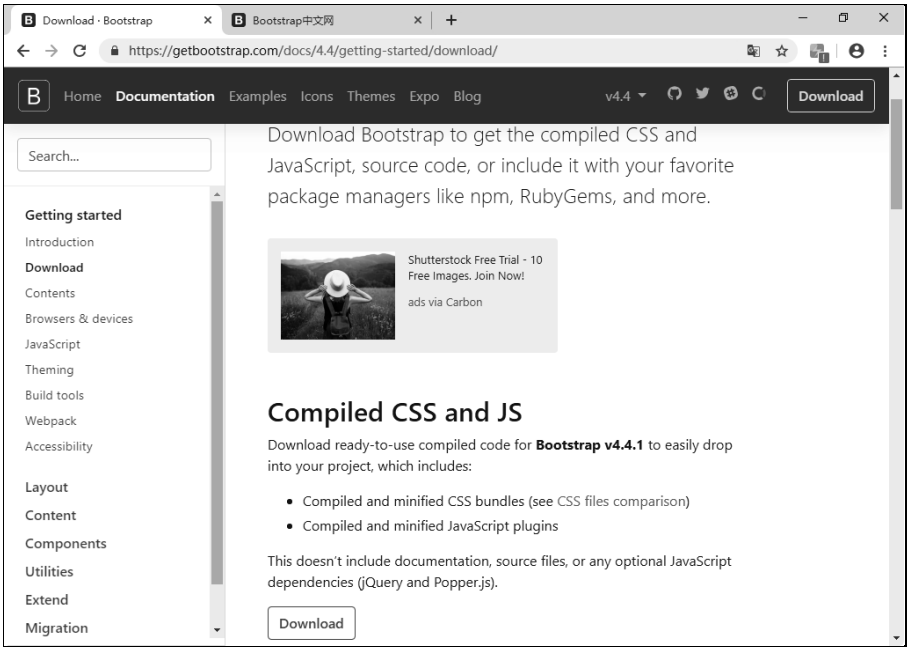

图 2.3　下载 Bootstrap 框架

直接单击 Download 按钮会下载一个压缩包，笔者下载后的名字是“bootstrap-4.4.1-dist.zip”。解压后的目录结构如图 2.4 所示。Bootstrap 重要的文件存放在 css 目录和 js 目录之中，下面我们会进行详细的介绍。

图 2.4　压缩包解压后的目录结构

注　意
Bootstrap 4 不再支持 IE 8、IE 9 和 iOS 6，使用这些浏览器的读者可以下载 Bootstrap 3。

2.1.2　Bootstrap 开发包目录结构

从图 2.4 中可以看到，Bootstrap 开发包中包含了 css、js 两个目录，分别代表编译好的样式文件和脚本文件。下面看一下这两个目录中具体都有什么文件，如图 2.5 所示。

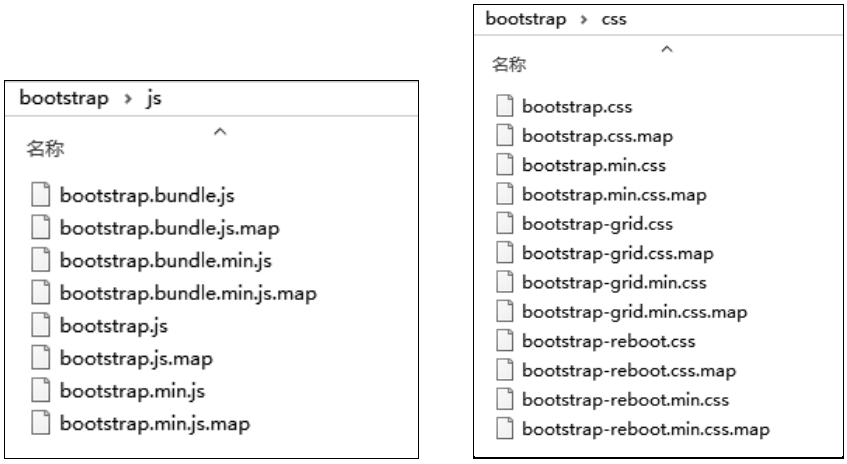

图 2.5　Bootstrap 开发包目录结构

图 2.5 中对设计人员有用的是 css 目录中的样式文件，以及 js 目录中的脚本文件。其中，文件名中不含“min”关键字的是未压缩的文件，而包含“min”关键字的是压缩好的文件（体积小，下载速度快）。在实际项目开发中，为了提高文件下载速度，都会选用压缩好的文件。

2.1.3　在网站中使用 Bootstrap 框架

在网站中使用 Bootstrap 框架的方法很简单，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，必须先引入 jQuery。

提　示
这里我们可以去 http://jquery.com/download/下载最新的 jQuery 文件，或使用当前项目中已有的 jQuery 文件。

【代码 2-1】引入 Bootstrap（详见源代码 ch02 目录中 ch02.loadBootstrap.html 文件）

```
<html>
<head>
  <link href="../bootstrap/css/bootstrap.css" rel="stylesheet">
</head>
<body>
Load Bootstrap 4...
  ……
<script src="../js/jQuery.js"></script>     <!--jQuery 应该放在前面优先加载-->
```

```
09  <script src="../bootstrap/js/bootstrap.js"></script>
10  </body>
11  </html>
```

提　示
JavaScript 文件放在文档尾部有助于提高页面加载速度。

【代码解析】

引入 Bootstrap 还可以使用第三方的 CDN 服务，Bootstrap 3 及以下版本建议使用 Bootstrap 中文网提供的 CDN，网址是 http://open.bootcss.com/；如果是做国外的项目，首选是 Google 的 CDN 服务。

本例效果如图 2.6 所示。

图 2.6　在网站中引入 Bootstrap 框架

2.2 调用 Bootstrap 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，比如宽度、高度、行高、对齐方式、边框等很多地方需要考虑，而且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，边思考如何给 id 或者 class 命名，最后可能还需要上级或者同事进行审核。

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。

【代码 2-2】应用 Bootstrap 表格样式（详见源代码 ch02 目录中 ch02.loadTableCSS.html 文件）

```
01  <!DOCTYPE html>
02  <html lang="en">
03  <head>
04      <meta charset="UTF-8">
```

```
    <link href="../bootstrap/css/bootstrap.css" rel="stylesheet">
    <title>Bootstrap - Load Table CSS</title>
</head>
<body>
<table class="table">   <!-- 只需要添加 class="table"即可 -->
    <tr>
        <th>No</th>
        <th>Name</th>
        <th>Age</th>
        <th>Gendle</th>
    </tr>
    <tr>
        <td>001</td>
        <td>Tom</td>
        <td>18</td>
        <td>male</td>
    </tr>
    <tr>
        <td>002</td>
        <td>Mary</td>
        <td>20</td>
        <td>female</td>
    </tr>
    <tr>
        <td>003</td>
        <td>Jack</td>
        <td>22</td>
        <td>male</td>
    </tr>
</table>
<script src="../js/jquery.js"></script>     <!--jQuery 应该放在前面优先加载-->
<script src="../bootstrap/js/bootstrap.js"></script>
</body>
</html>
<head>
```

本例效果如图 2.7 所示。

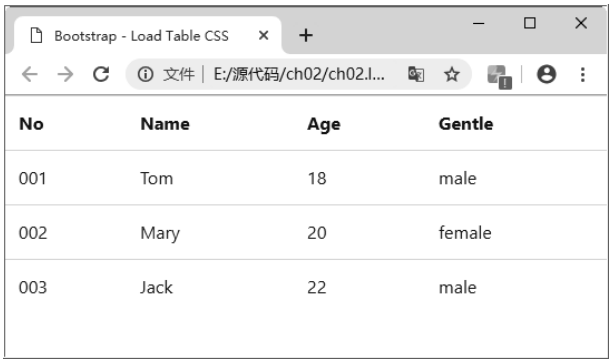

图 2.7　应用 Bootstrap 表格样式

Bootstrap 框架功能非常强大，可提供多种表格样式。下面我们添加一种类名为“table-striped”的类似斑马纹表格样式，并同时添加类名为“table-bordered”的样式来为表格加上边框。

【代码 2-3】（详见源代码 ch02 目录中 ch02.loadTableStripedCSS.html 文件）

```
<table class="table table-striped table-bordered">
    <tr>
        <th>No</th>
        <th>Name</th>
        <th>Age</th>
        <th>Gendle</th>
    </tr>
    <tr>
        <td>001</td>
        <td>Tom</td>
        <td>18</td>
        <td>male</td>
    </tr>
    <tr>
        <td>002</td>
        <td>Mary</td>
        <td>20</td>
        <td>female</td>
    </tr>
    <tr>
        <td>003</td>
        <td>Jack</td>
        <td>22</td>
        <td>male</td>
```

```
    </tr>
</table>
```

本例代码效果如图 2.8 所示。

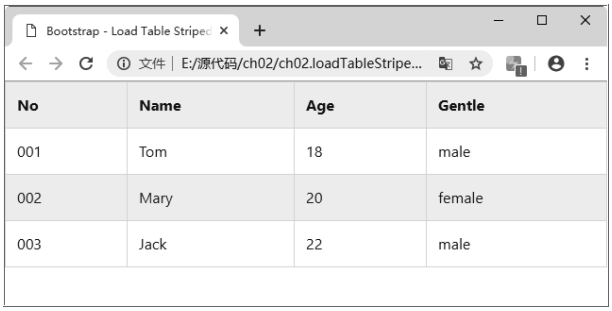

图 2.8　带斑马纹和边框的表格

2.3 调用 Bootstrap 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，就可以轻松地构建出一些通用组件。以导航条为例，导航条是在应用或网站中作为导航页头的响应式基础组件，在移动设备上可以折叠（并且可开可关），且在视口（viewport）宽度增加时逐渐变为水平展开模式。

【代码 2-4】（详见源代码 ch02 目录中 ch02.loadNavbar.html 文件）

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link href="../bootstrap/css/bootstrap.css" rel="stylesheet">
    <title>Bootstrap - Load Table Striped CSS</title>
</head>
<body>
 <ul class="nav">
    <li class="nav-item"><a class="nav-link active" href="#">Home</a>
    </li>
     <li class="nav-item"><a class="nav-link" href="#">News</a></li>
    <li class="nav-item"><a class="nav-link" href="#">BBS</a></li>
    <li class="nav-item"><a class="nav-link" href="#">About</a></li>
   </ul>
```

```
15  <script src="../js/jquery.js"></script>
    <!--jQuery 应该放在前面优先加载-->
16  <script src="../bootstrap/js/bootstrap.js"></script>
17  </body>
18  </html>
19  <head>
```

只要符合 ul .nav>li .nav-item>a .nav-link 这样的文档结构，就可以构建出一个顶部导航条，本例效果如图 2.9 所示。

图 2.9　导航条效果

2.4 调用 Bootstrap JS 特效

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。

【代码 2-5】（详见源代码 ch02 目录中 ch02.loadTabs.html 文件）

（1）编写 HTML 文档：

```
01  <ul class="nav nav-tabs " id="myTab">
02     <li class="nav-item" ><a class="nav-link active" href="#home"
    data-toggle="tab">Home</a></li>
03     <li class="nav-item" ><a class="nav-link" href="#news"
    data-toggle="tab">News</a></li>
04     <li class="nav-item" ><a class="nav-link" href="#blog"
    data-toggle="tab">Blog</a></li>
05     <li class="nav-item" ><a class="nav-link" href="#about"
    data-toggle="tab">About</a></li>
06  </ul>
07  <!-- href 属性的值要和后面 tab-pane 中的 id 值对应 -->
08  <div class="tab-content">
```

```
09      <div class="tab-pane active" id="home">Home Page</div>
    <!--tab 标签对应的内容-->
10      <div class="tab-pane" id="news">News Page</div>
11      <div class="tab-pane" id="blog">Blog Page</div>
12      <div class="tab-pane" id="about">About Page</div>
13  </div>
```

（2）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（上述代码第 2 行），这种方式的好处是无须编写任何 JavaScript 代码就可以实现功能。另一种是类似普通 jQuery 插件的调用方式，例如：

```
$('#myTab a').click(function (e) {
e.preventDefault();
$(this).tab('show');
})
```

本例最终实现的效果如图 2.10 所示，单击标签就可以切换内容。

图 2.10　标签页效果

2.5 实战：一个 Bootstrap 实现的响应式页面

Bootstrap 4 默认就引入了响应式设计，有以下两点特色：

- 拥抱各种屏幕大小（大屏幕、小屏手机、大屏手机等），列会根据屏幕大小自动重新排列。
- 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，更为复杂，但使用更为灵活，能适应更多的场景。

Bootstrap 3 采用 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。这方面 Bootstrap 4 做了调整，Bootstrap 3 和 Bootstrap 4 最大的区别在于 Bootstrap 4 使用 Flexbox（弹性盒子），而不是浮动的。Flexbox 的一大优势是没有

指定宽度的网格列将自动设置为等宽与等高列。Bootstrap 4 的网格宽度可以是 col-、col-sm-、col-md-、col-lg-和 col-xl-。

说　明
具体的栅格或布局相关，在第 3 章会详细介绍，这里简单测试即可。

通过表 2.1 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 2.1　Bootstrap的响应式布局区间

	超小屏幕设备 手机（<576px）	小屏幕设备 大手机（≥576px）	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）
最大.container 宽度	None (自动)	540px	720px	960px	1140px
class 前缀	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
列数	12				
槽宽	30px（每列左右均有 15px）				
可嵌套	Yes				
列排序	Yes				

【代码 2-6】一个 Bootstrap 实现的响应式页面（详见源代码 ch02 目录中 ch02.firstBootstap.html 文件）

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link href="../bootstrap/css/bootstrap.css" rel="stylesheet">
    <title>Bootstrap -  first page</title>
</head>
<body style="margin:20px">
<div class="container">
    <div class="row">
        <div class="col-sm-12 col-md-3 col-lg-5 col-xl-4">
    <!-- 左侧边栏-->
            <h1>News</h1>
            <h1>Blog</h1>
            <h1>About</h1>
        </div>
        <div class="col-sm-12 col-md-9 col-lg-7 col-xl-8">  <!--
    右侧边栏-->
        <p>This is a first Bootstrap page.Please tests this page in
    various screen size.</p>
```

```
        </div>
    </div>
</div>
<script src="../js/jquery.js"></script>     <!--jQuery 应该放在前面优先加载-->
<script src="../bootstrap/js/bootstrap.js"></script>
</body>
</html>
```

下面尝试使用手机的屏幕尺寸来显示该页面（可以通过浏览器插件 Responsive Web Design Test 来实现）。图 2.11 是 Portrait 样式，图 2.12 是 Landscape 样式。

图 2.11　Bootstrap 中的 Portrait 响应式页面

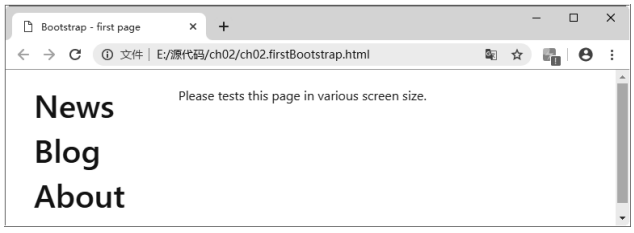

图 2.12　Bootstrap 中的 Landscape 响应式页面

2.6 本章小结

本章主要介绍了应用 Bootstrap 框架开发的入门知识，包括调用样式、组件和 JS 特效等方面的内容，并配合具体代码实例进行讲解，希望对读者有一定的帮助。

第 3 章

◂ Bootstrap脚手架 ▸

脚手架的英文原名是 Scaffolding，好多中文翻译为脚手架，也有人翻译为基础架构，实际就是网页的整体模板和构架。这一章我们介绍 Bootstrap 脚手架方面的内容，其实就是介绍网页在设计、布局方面的知识，因为这些内容偏理论概念，所以还会配合具体实例帮助读者加深理解。

本章主要内容包括：

- 了解 Bootstrap 全局样式
- 认识栅格系统和流式栅格系统
- 掌握页面的几种布局
- 学习如何进行响应式设计

3.1 Bootstrap 全局样式

本节介绍 Bootstrap 全局样式，主要包括全局样式的基本概念、使用特点方面的内容，是学习 Bootstrap 框架的起步要点。

3.1.1 什么是全局样式

所谓 Bootstrap 全局样式，其实就是通过 Bootstrap 框架为页面设置的全局 CSS 样式表。该全局样式表包括基本的 HTML 元素样式以及栅格系统，可以增强页面及其元素的 CSS 效果。另外，开发人员深入学习 Bootstrap 全局样式，有助于理解 Bootstrap 框架的底层结构，有助于 Web 开发获得更好、更快、更强的实践效果。

3.1.2 基于 HTML5 文档类型

Bootstrap 框架使用到的某些 HTML 元素和 CSS 属性需要将页面设置为 HTML5 文档类型。在具体 Web 项目中，每个 HTML 页面都要参照下面的格式进行设置。

```
<!DOCTYPE html>
<html lang="zh-CN">
```

```
......
</html>
```

注　意
为了 Bootstrap 框架获得更好的应用效果，我们可以将上面的页面格式理解为是强制执行的。

3.1.3　屏幕、排版与链接

Bootstrap 框架为屏幕、排版和链接设置了基本的全局样式，具体定义在 Bootstrap 源码包 scss 目录中的_variables.scss 文件中，读者可以打开该源文件进行参阅。下面列举一些_variables.scss 文件中的源码，分析一下 Bootstrap 是如何定义全局样式的。

注　意
要想查看这些文件，必须从官网下载 Bootstrap 框架的源文件。

【代码 3-1】（详见 Bootstrap 源码包中_variables.scss 文件）

```
// Body
//
// Settings for the `<body>` element.

$body-bg:                 $white !default;
$body-color:               $gray-900 !default;
```

【代码 3-1】设置了页面 Body 的全局样式，具体如下：

- 背景为变量$white。
- 颜色为变量$gray-900。

其中，!default 表示默认值。

【代码 3-2】（详见 Bootstrap 源码包中_variables.scss 文件）

```
// Links
//
// Style anchor elements.

$link-color:                          theme-color("primary") !default;
$link-decoration:                      none !default;
$link-hover-color:                     darken($link-color, 15%) !
```

【代码 3-2】设置了超链接的全局样式，具体如下：

- 颜色为 theme-color("primary")。
- 当超链接处于 ":hover" 状态时颜色为 darken($link-color, 15%)。

3.2 栅格系统

栅格系统是 Bootstrap 框架的一大特色，通过使用栅格系统使得页面布局更简单、更合理、更美观、更易于维护。

3.2.1 默认栅格系统

1. 栅格系统特性

Bootstrap 框架默认的栅格系统最多为 12 列，形成一个 960px 宽的容器，默认就是响应式布局特性。栅格系统会根据可视窗口的宽度从 540px 到 1140px 进行自适应的动态调整。在可视窗口小于 540px 宽的情况下，列将不再固定并且会在垂直方向进行自动堆叠。以上这几条描述的就是 Bootstrap 栅格系统的特性。

图 3.1 就是 Bootstrap 栅格系统（12 列栅格）的一个示例。

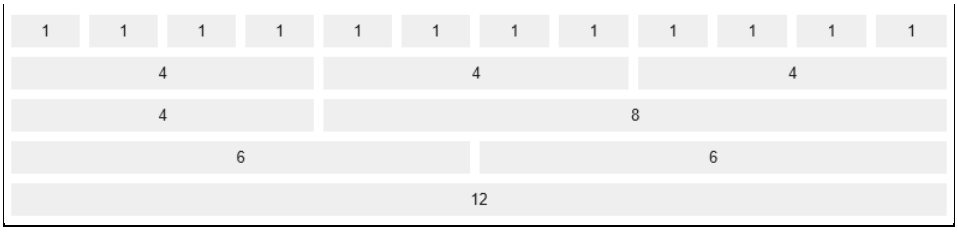

图 3.1　Bootstrap 栅格系统

2. 带有基本栅格的 HTML 代码

假设应用简单的列式布局，设计时创建一个类名为.row 的容器，并在容器中加入合适数量的列（类名为.col）即可。Bootstrap 默认是 12 列的栅格，所以.col 列所跨越的栅格数之和应该是 12（或者等于其父容器的栅格数）。.col 的取值前面简单介绍过，比如.col-sm-4 就是将手机屏幕分为 3 列，12 除以 4 是 3，而 sm 代表大手机（540px~720px 屏幕之间）。.col-xl-3 就是将大型桌面分为 4 列。

【代码 3-3】一个基本的栅格布局设计（详见源代码 ch03 目录中 ch03.gridBase.html 文件）

```
<div class="row">
  <div class="col">首页</div>
```

```
    <div class="col">文档</div>
    <div class="col">帮助</div>
  </div>
```

上面的代码展示了等宽 3 列的栅格布局，页面效果如图 3.2 所示，这是 Bootstrap 自动布局的效果。

图 3.2　基本栅格系统

3. 偏移列

Bootstrap 栅格系统支持偏移列，可以使用 offset-*-*类将列向右移动，相当于将列的左边距增加了指定单位的宽度。offset-*-*类中第一个星号（*）可以是 sm、md、lg、xl，表示屏幕设备类型，第二个星号（*）可以是 1~11 的数字，比如 offset-*-4 就是往右移动 4 格。

【代码 3-4】偏移列的栅格布局设计（详见源代码 ch03 目录中 ch03.gridOffset.html 文件）

```
<section id="global">
       <div class="page-header">
          <h3>栅格系统 --- 嵌套列</h3>
       </div>
</section>
<div class="row">
  <div class="col-sm-4  offset-sm-1">首页</div>
  <div class="col-sm-4">文档</div>
  <div class="col-sm-4">帮助</div>
</div>
```

上面的代码展示了.offsetXX 的栅格布局，页面效果如图 3.3 所示。首列偏移后，最后一列折叠到了第 2 行。

图 3.3　偏移列栅格系统

4. 不等宽列

Bootstrap 栅格系统前面都是设计的等宽列，如果有 12 列，前面是 3 格，中间是 6 格，是不是也可以呢？答案是可以的。

【代码 3-5】不等宽的栅格布局设计（详见源代码 ch03 目录中 ch03.gridNesting.html 文件）

```
<div class="row show-grid">
    <div class="span12">
        level 1
        <div class="row show-grid">
            <div class="span3">
                level 2
            </div>
            <div class="span6 offset3">
                level 2
            </div>
        </div>
    </div>
</div>
```

上面的代码展示了不等宽列的栅格布局，页面效果如图 3.4 所示，其中第一列宽是 3 格，第二列宽是 6 格，第三列宽是 3 格。

图 3.4　不等宽栅格系统

3.2.2 流式栅格系统

流式栅格系统的特点是对每一列的宽度使用百分比，这是它与固定栅格系统的主要区别。流式栅格系统与固定栅格系统一样拥有响应式布局的能力，这就保证其能对不同的分辨率和设备做出适当的调整。

1. 基本的流式栅格的 HTML 代码

流式栅格将固定栅格的.row 类替换为.row-fluid 类，就能让任何一行“流动”起来，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。

【代码 3-6】一个基本的流式栅格布局设计（详见源代码 ch03 目录中 ch03.fluidBase.html 文件）

```
<div class="bs-docs-grid">
    <div class="row-fluid show-grid">
        <div class="span1">1</div>
    </div>
    <div class="row-fluid show-grid">
        <div class="span2">2</div>
    </div>
    <div class="row-fluid show-grid">
        <div class="span4">4</div>
    </div>
</div>
```

上面的代码展示了.span1、.span2 和.span4 的流式栅格布局，页面效果如图 3.5 所示。

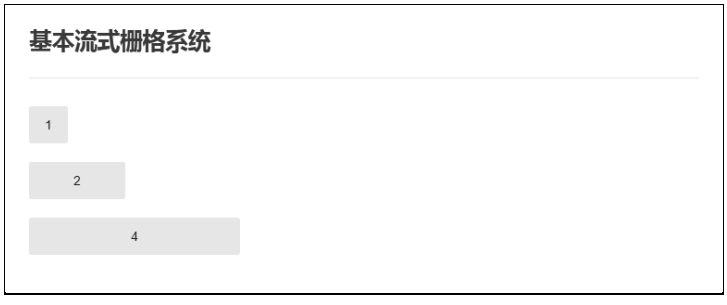

图 3.5　基本流式栅格系统

2. 流式栅格的偏移

Bootstrap 流式栅格系统同样支持偏移列，使用.offsetXX 类即可，相当于将列的左边距增加了指定百分比的宽度。

【代码 3-7】流式栅格布局偏移设计（详见源代码 ch03 目录中 ch03.fluidOffset.html 文件）

```
<div class="bs-docs-grid">
    <div class="row-fluid show-grid">
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
        <div class="span1">1</div>
    </div>
    <div class="row-fluid show-grid">
        <div class="span2">span2</div>
        <div class="span3 offset3">3 offset 3</div>
    </div><!-- /row -->
    <div class="row-fluid show-grid">
        <div class="span3 offset3">3 offset 3</div>
        <div class="span2 offset2">2 offset 2</div>
    </div><!-- /row -->
    <div class="row-fluid show-grid">
        <div class="span6 offset6">6 offset 6</div>
    </div><!-- /row -->
</div>
```

上面的代码展示了.offsetXX 的流式栅格布局，页面效果如图 3.6 所示。

图 3.6　流式栅格系统偏移

3. 流式栅格的嵌套

Bootstrap 流式栅格系统同样支持嵌套列，在默认栅格系统里将已有的.spanXX 列内添加一个新的.row-fluid 并加入.spanXX 列，就可实现嵌套。需要注意的是，被嵌套列中的每列列数总和要等于父级列。

【代码 3-8】流式栅格布局嵌套设计（详见源代码 ch03 目录中 ch03.fluidNesting.html 文件）

```
<div class="row-fluid show-grid">
    <div class="span12">
        level 1
        <div class="row-fluid show-grid">
            <div class="span3">
                level 2
            </div>
            <div class="span6 offset3">
                level 2
                <div class="row-fluid">
                    <div class="span6">Fluid 6</div>
                    <div class="span6">Fluid 6</div>
                </div>
            </div>
        </div>
    </div>
</div>
```

上面的代码展示了嵌套列的流式栅格布局，页面效果如图 3.7 所示。

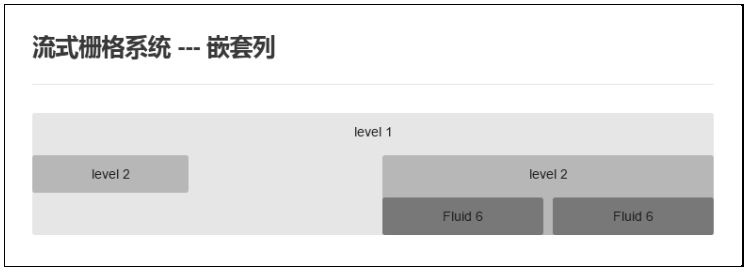

图 3.7　流式栅格系统嵌套

3.3 页面布局

Bootstrap 框架设计了固定布局、流式布局、弹性布局等页面布局方式，默认是弹性布局。下面分别针对这些布局进行介绍。

3.3.1　固定布局和流式布局

固定布局使用.container 类，流式布局使用.container-fluid 类。我们直接通过一个例子来说明。

【代码 3-9】一个 Bootstrap 布局的设计（详见源代码 ch03 目录中 ch03.fixedLayout.html 文件）

```
<div class="container bg-info">
      <h2>你好</h2>
</div>
<div class="container-fluid bg-warning">
      <h2>你好</h2>
</div>
```

在手机屏幕小于 768px 时，上述代码的效果如图 3.8 所示；在屏幕大于 768px 时，效果如图 3.9 所示。也就是说，两者都是响应式的，只有采用流式布局时，div 的宽度与屏幕才相等。

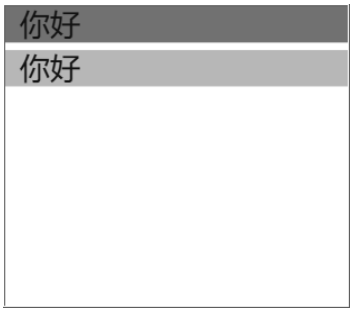

图 3.8　布局（小于 768px）

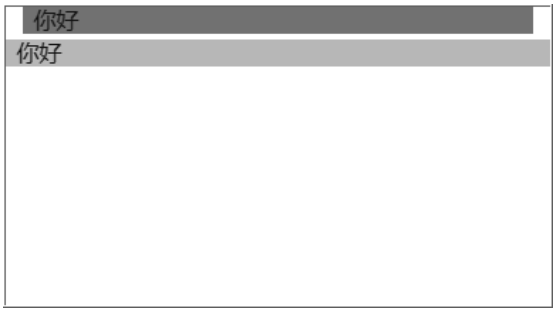

图 3.9　布局（大于 768px）

3.3.2　Bootstrap 4 弹性布局（FlexBox）

弹性布局实际是 CSS 3 的一个特性，是一种当页面需要适应不同的屏幕大小以及设备类型时，确保元素拥有恰当行为的布局方式。Bootstrap 也采用了这种布局方式，也有人称为弹性盒子。Bootstrap 4 使用一整套响应式 FlexBox 快速管理网格列、导航等组件的布局、对齐和大小。对于更复杂的实现，还可以根据需要自定义 CSS。

注　意
IE 9 及以下版本不支持弹性布局，如果需要兼容这些浏览器，需要使用 Bootstrap 3 版本。

【代码 3-10】一个 Flex 布局的设计（详见源代码 ch03 目录中 ch03.flexLayout.html 文件）

```
<div class="d-flex p-2 bg-ight">我是一个弹性盒子</div>
<div class="d-inline-flex p-2 bg-light">我是一个嵌入式的弹性盒子</div>
```

在【代码 3-10】中，.d-flex 类代表这是一个弹性盒子，.p 代表 padding，.bg-light 代表背景色。弹性盒子有两种类型：.d-*-flex 和.d-*-inline-flex，其中*可以省略，也可以是任意的屏幕大小，如 lg、md 等，例如.d-md-inline-flex。

页面效果如图 3.10 所示。

图 3.10　弹性布局

FlexBox 布局还有很多重要的类，这里我们先简单介绍一下，下一节再详细举例。

3.4 响应式设计

Bootstrap 框架设计的初衷就是为了更好地满足响应式设计原理，本节将介绍针对 Bootstrap 如何应用响应式设计的内容。

3.4.1 启用响应式特性

通过在文档中的<head>标签里添加合适的 meta 标签并引入一个额外的样式表即可启用响应式 CSS。如果已经在定制页面编译好一个 Bootstrap，那么只需添加一个 meta 标签。

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link href="assets/css/bootstrap-responsive.css" rel="stylesheet">
```

提　示
Bootstrap 在默认情况下是没有引入响应式特性的，因为不是任何情况都需要使用到。我们并不是鼓励开发者移除此功能，而是在需要使用的时候才启用它。

3.4.2 响应式 Bootstrap 特点

媒体查询允许在一些条件基础上自定义 CSS，例如 ratios、widths、display type 等，但通常都是围绕着 min-width 和 max-width 这两个属性进行。响应式 Bootstrap 特点可以概括如下：

- 修改栅格系统中列的宽度。
- 需要的时候，用堆叠元素代替浮动。
- 调整标题和文本的大小以便适合各种设备。

提　示
谨慎地使用媒体查询，先从手机屏幕开始。对于大型的项目，应该考虑使用专门的代码库，而不是构筑在媒体查询之上。

3.4.3　Bootstrap 支持的设备

Bootstrap 所支持的几个媒体查询都放在了一个文件中，可以让项目更灵活地适应不同设备和屏幕分辨率，具体如表 3.1 所示。

表 3.1　媒体设备

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	

【代码 3-11】媒体查询的代码示例

```
/* 大屏幕 */
@media (min-width: 1200px) { ... }
/* 平板电脑和小屏电脑之间的分辨率 */
@media (min-width: 768px) and (max-width: 979px) { ... }
/* 横向放置的手机和竖向放置的平板之间的分辨率 */
@media (max-width: 767px) { ... }
/* 横向放置的手机及分辨率更小的设备 */
@media (max-width: 480px) { ... }
```

3.4.4　响应式布局辅助类

为了更快地开发移动设备，可以使用的辅助类针对不同设备显示和隐藏内容。表 3.2 是可用的类列表，以及它们在媒体查询布局下的效果。

表 3.2　辅助类

类	手机（767px 及以下）	平板（979px～768px）	电脑
.visible-phone	显示	隐藏	隐藏
.visible-tablet	隐藏	显示	隐藏
.visible-desktop	隐藏	隐藏	显示

（续表）

类	手机（767px 及以下）	平板（979px～768px）	电脑
.hidden-phone	隐藏	显示	显示
.hidden-tablet	显示	隐藏	显示
.hidden-desktop	显示	显示	隐藏

3.4.5 响应式 FlexBox 的重要类

除了使用 d-flex 类实现 FlexBox 布局外，Bootstrap 4 还提供了很多辅助 FlexBox 布局的类。

【代码 3-12】一个带子元素的 Flex 布局（横向 row，详见源代码 ch03 目录中 ch03.flexLayout1.html 文件）

```
<div class="d-flex flex-row bg-light mb-3">
  <div class="p-2 bg-light">弹性布局 1</div>
  <div class="p-2 bg-light">弹性布局 2</div>
  <div class="p-2 bg-light">弹性布局 3</div>
</div>
<div class="d-flex flex-row-reverse bg-light">
  <div class="p-2 bg-light">弹性布局 1</div>
  <div class="p-2 bg-light">弹性布局 2</div>
  <div class="p-2 bg-light">弹性布局 3</div>
</div>
```

【代码 3-12】中，.flex-row 类代表行布局，表示这些子元素都在一行，默认从左到右排列。.flex-row-reverse 类则代表从右到左排列。.mb 类代表 margin bottom，如果是.mt，则是 margin top。

页面效果如图 3.11 所示。

图 3.11 弹性布局（一）

【代码 3-13】一个带子元素的 Flex 布局（竖向 column，详见源代码 ch03 目录中 ch03.flexLayout2.html 文件）

```
<div class="d-flex flex-column bg-light mb-3">
  <div class="p-2 bg-light">弹性布局 1</div>
  <div class="p-2 bg-light">弹性布局 2</div>
  <div class="p-2 bg-light">弹性布局 3</div>
</div>
<div class="d-flex flex-column-reverse bg-light">
  <div class="p-2 bg-light">弹性布局 1</div>
  <div class="p-2 bg-light">弹性布局 2</div>
  <div class="p-2 bg-light">弹性布局 3</div>
</div>
```

在【代码 3-13】中，.flex-column 类代表列布局，表示这些子元素都在一列，默认从上到下排列。.flex-column-reverse 类代表从下到上排列。页面效果如图 3.12 所示。

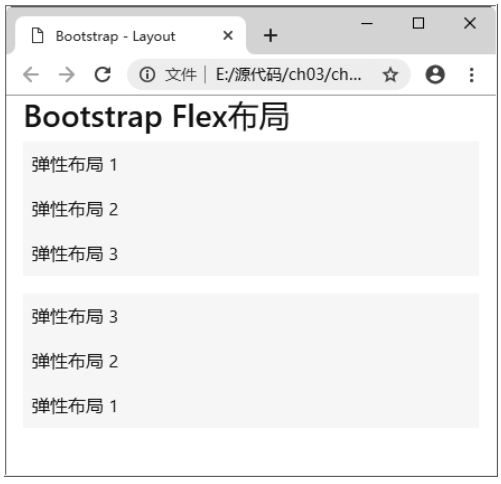

图 3.12　弹性布局（二）

.justify-content-*类用于修改弹性布局中子元素的排列方式，*允许的值有 star（默认）、end、center、between 和 around。

【代码 3-14】一个 Flex 布局的元素排列方式（详见源代码 ch03 目录中 ch03.flexLayout2.html 文件）

```
<div class="d-flex flex-row bg-light justify-content-start">
  <div class="p-2 bg-light">弹性布局 1</div>
  <div class="p-2 bg-light">弹性布局 2</div>
  <div class="p-2 bg-light">弹性布局 3</div>
</div>
<div class="d-flex flex-row bg-light justify-content-end">
  <div class="p-2 bg-light">弹性布局 1</div>
```

```
    <div class="p-2 bg-light">弹性布局 2</div>
    <div class="p-2 bg-light">弹性布局 3</div>
  </div>
  <div class="d-flex flex-row bg-light justify-content-center">
    <div class="p-2 bg-light">弹性布局 1</div>
    <div class="p-2 bg-light">弹性布局 2</div>
    <div class="p-2 bg-light">弹性布局 3</div>
  </div>
  <div class="d-flex flex-row bg-light justify-content-between">
    <div class="p-2 bg-light">弹性布局 1</div>
    <div class="p-2 bg-light">弹性布局 2</div>
    <div class="p-2 bg-light">弹性布局 3</div>
  </div>
  <div class="d-flex flex-row bg-light justify-content-around">
    <div class="p-2 bg-light">弹性布局 1</div>
    <div class="p-2 bg-light">弹性布局 2</div>
    <div class="p-2 bg-light">弹性布局 3</div>
  </div>
```

在【代码 3-14】中，使用.justify-content-*类显示了 5 种子元素的布局方式，包括左对齐、右对齐等。页面效果如图 3.13 所示。

图 3.13　弹性布局（三）

说　明
如果要设置单行的子元素对齐可以使用.align-items-* 类来控制，包含的值有.align-items-start 、 .align-items-end 、 .align-items-center 、 .align-items-baseline 和 .align-items-stretch （默认）。

Flex 布局还支持一行放不下时的换行，有 3 个类，即.flex-nowrap、.flex-wrap 和.flex-wrap-reverse，分别是不换行、顺序换行和换行之后的在上方。

【代码 3-15】一个带子元素换行的 Flex 布局（详见源代码 ch03 目录中 ch03.flexLayout4.html 文件）

```
<div class="d-flex flex-nowrap bg-light mb-3">
  <div class="p-2 bg-light">弹性布局 1</div>
  <div class="p-2 bg-light">弹性布局 2</div>
  <div class="p-2 bg-light">弹性布局 3</div>
……
</div>
<div class="d-flex flex-wrap bg-light">
  <div class="p-2 bg-light">弹性布局 21</div>
  <div class="p-2 bg-light">弹性布局 22</div>
  <div class="p-2 bg-light">弹性布局 23</div>
……
</div>
<div class="d-flex flex-wrap bg-light">
  <div class="p-2 bg-light">弹性布局 31</div>
  <div class="p-2 bg-light">弹性布局 32</div>
  <div class="p-2 bg-light">弹性布局 33</div>
……
</div>
```

页面效果如图 3.14 所示。

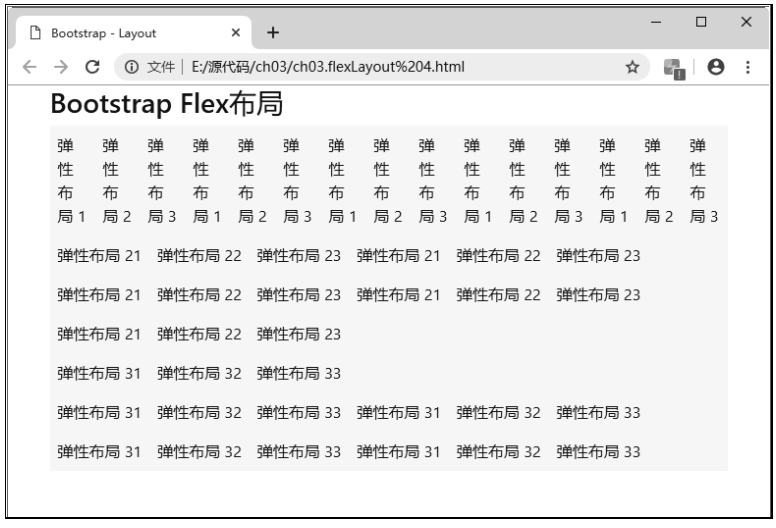

图 3.14　弹性布局（四）

3.5 从 Less 到 Sass

Sass 和 Less 都属于 CSS 预处理器。CSS 预处理器定义了一种新的语言，基本思想是用一种专门的编程语言为 CSS 增加一些编程的特性，将 CSS 作为目标生成文件，然后开发者使用这种语言进行 CSS 编码工作（用一种专门的编程语言，进行 Web 网页样式设计，再通过编译器转化为正常的 CSS 文件，以供项目使用）。

之前 Bootstrap 中的 CSS 使用的都是 Less 编译，到了 Bootstrap 4 后，全部使用 Sass 编译，所以如果读者下载了 Bootstrap 源代码，就会发现所有样式文件都在 scss 文件夹中，且扩展名都是.scss。

3.6 本章小结

本章主要介绍了应用 Bootstrap 框架开发的全局样式表、栅格系统和页面布局等方面的内容，并配合具体代码实例进行讲解，希望对读者有一定的帮助。