

# 1

## 第 1 章

# 响应式网页开发基础知识

### 1.1 响应式网页简介

2010年5月,著名网页设计师 Ethan Marcotte 首次提出了响应式的设计概念,随后响应式迅速席卷前端和设计领域,成为如今网页设计的大趋势。

在响应式网页概念没有被提出时,人们针对不同的浏览设备分别设计相应的网站进行管理,当然那时手机等移动设备还不如现在流行,网页浏览、购物需求主要还是集中在 PC 端。但是,很快大家就发现一个难题:即使是计算机这种设备,屏幕尺寸的型号种类也很多,难道企业要针对各种不同尺寸的屏幕独立设计一个网站吗?当然不是。在传统互联网行业,当大家面对上百种屏幕尺寸时,选取所谓的标准尺寸屏幕,如 17 英寸(1 英寸=2.54 厘米)、19 英寸、22 英寸大小的屏幕,企业根据标准屏幕大小设计网站。然而,这只能确保部分用户可以得到相对较好的用户体验,另一部分用户就只能手动进行调整,如某团购网站在尺寸为 27 英寸、分辨率为 1 280×1 024 的 PC 屏幕上的显示如图 1-1 所示,在尺寸为 14 英寸、分辨率为 1 366×768 的笔记本电脑屏幕上的显示如图 1-2 所示,对比图 1-1 和图 1-2 可以看出,该网站在图 1-2 所在屏幕的水平方向显示不全,只能通过拉动浏览器横向的滚动条查看更多详情,十分不便,用户纷纷反映体验很差。

响应式网页是一种网络页面设计布局,其理念是页面的设计与开发可根据用户行为及设备环境,如系统平台、屏幕尺寸、屏幕定向等进行相应的响应和调整。无论用户正在使用 PC、笔记本电脑、iPad 或其他移动设备,网页都能够自动切换分辨率、图片及相关脚本功能等,以适应不同设备,即网页能自动适应用户的设备环境,如图 1-3 所示。

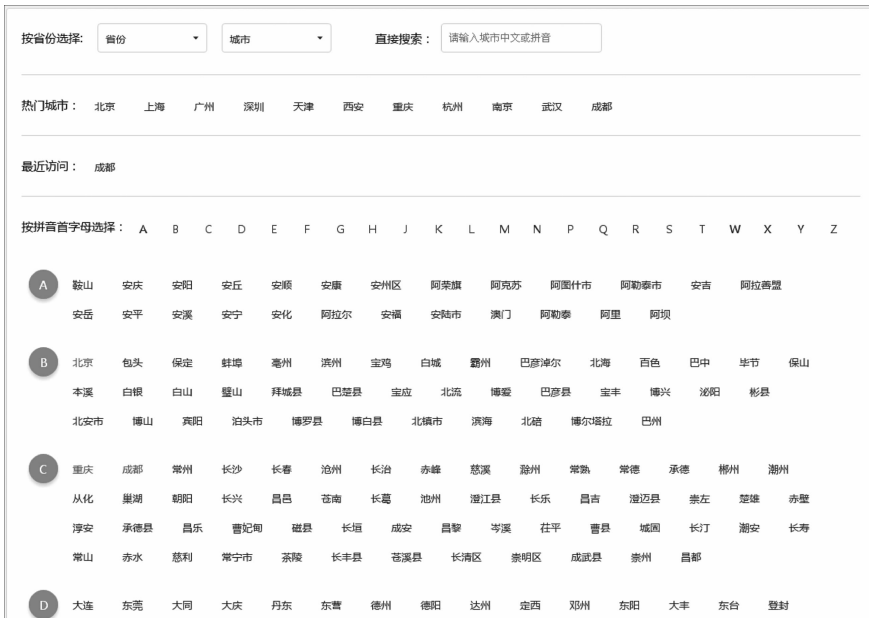


图 1-1 某团购网站在 PC 屏幕上的显示效果



图 1-2 某团购网站在笔记本电脑屏幕上的显示效果



图 1-3 响应式网页理念

## 1.2 移动互联网的发展

在传统互联网阶段,因为响应式网页技术的适用范围较小,所以它并没有得到重视,但随着移动互联网的发展,Web 开发要求同时满足 PC 客户端和移动客户端的要求,相关的 Web 设计者才开始重视响应式网页设计。

响应式网页是随着移动互联网的发展而发展起来的,可以说,移动互联网是响应式网页开发技术发展的基础。

移动互联网是移动通信和互联网的结合体,用户可以通过手机、平板电脑等移动数据终端接入 Internet,实现信息交互。我国移动互联网的发展可分为如下 4 个阶段。

### 1. 萌芽阶段(2000—2007 年)

由于移动 2G 网速和手机智能化程度的限制,该时期我国移动互联网的发展处于一个简单的 WAP 应用期。WAP 应用即把 Internet 的 HTML 信息转换成 WML 描述的信息,并显示在手机上。

由于 WAP 只要求手机和 WAP 代理服务器的支持,而不要求现有的移动通信网络协议做任何改动,因而被广泛地应用于 GSM、CDMA、TDMA 等多种网络中。在移动互联网萌芽阶段,利用手机自带的支持 WAP 协议的浏览器访问企业 WAP 门户网站是当时移动互联网发展的主要形式。

2000 年 11 月 10 日,中国移动推出“移动梦网计划”,打造开放、合作、共赢的产业价值链,催生了一大批基于梦网的 SP 服务商,用户通过短信、彩信、手机上网等模式享受移动互联网服务。

### 2. 成长培育阶段(2008—2011 年)

2009 年 1 月 7 日,中华人民共和国工业和信息化部(以下简称工信部)宣布,批准中国移动、中国电信、中国联通三大电信运营商分别增加 TD-SCDMA、CDMA2000、WCDMA 技术制式的第三代移动通信业务经营许可,我国 3G 网络大规模建设正式铺开,中国移动互联网全面进入 3G 时代。

随着 3G 移动网络的部署和智能手机的出现,移动网速大幅提升初步突破了手机上网带宽瓶颈,安装简单应用程序的移动智能终端让移动上网功能得到大大增强,中国移动互联网掀开新的发展篇章。

在此期间,各大互联网公司都在摸索如何占领移动互联网的至高点,百度、腾讯、奇虎 360 等一些大型互联网公司通过推出手机浏览器的方式来抢占移动互联网市场,新浪、优酷土豆等其他一些互联网公司则是通过与手机制造商合作的方式来争取移动互联网的入口,其在智能手机出厂时,就把企业服务应用(如微博、视频播放器等)预安装在手机中。

### 3. 高速成长阶段(2012—2013 年)

进入 2012 年之后,由于移动上网需求大增,具有触摸屏功能的智能手机的大量普及和使用、安卓智能操作系统的大规模商业化应用、手机应用程序商店的出现等极大地丰富了手

机上网功能,移动互联网应用呈现爆发式增长趋势。

腾讯公司于 2011 年 1 月 21 日推出即时通信微信服务,截至 2013 年 10 月底,腾讯微信的用户数量已经超过 6 亿,每日活跃用户 1 亿。同时,小米推出了米聊,阿里推出了来往,网易推出了易信等即时通信业务,纷纷抢占移动互联网即时通信市场,实现了移动端社交应用的快速拓展,让电信运营商措手不及。

除了利用即时通信抢占移动互联网入口之外,各大互联网公司都在推进自身业务向移动互联网转型。阿里加大了手机淘宝和手机支付宝业务的推广力度,2013 年“双十一”购物节,手机淘宝的支付宝成交额同比增长 560%,单日成交笔数占整体成交数的 21%,同比增长 420%,截至 2013 年底,手机支付宝用户数量超过 1 亿。在此期间,微信支付也快速发展,支付宝和微信展开了移动支付争夺大战。同时,百度加快将搜索等业务向移动端迁移,推出了手机搜索、手机地图等各类手机应用,新浪、网易等传统门户网站加快了在移动端的布局,纷纷加大对手机端新闻 App 应用的推广。

另外,滴滴出行、今日头条等一大批基于移动互联网应用服务创新和商业模式创新的应用大量涌现,改变着人们的生活和出行方式。

#### 4. 全面发展阶段(2014 年至今)

2013 年 12 月 4 日,工信部正式向中国移动、中国电信和中国联通三大运营商发放了 TD-LTE 4G 牌照,中国 4G 网络正式大规模铺开。2015 年 2 月 27 日,工信部又向中国电信和中国联通发放“LTE/第四代数字蜂窝移动通信业务(FDD-LTE)”经营许可。4G 网络建设让中国移动互联网走上了快速发展的轨道。

4G 网速的提高和上网的便捷性,促使移动互联网应用开始全面发展,催生了许多公司利用移动互联网开展业务。在这个阶段,基本上所有的互联网公司都会设立专门的移动终端部门,负责公司产品在移动终端的战略布局和发展,这也加剧了移动互联网的用户争夺战。在此期间,阿里、腾讯等互联网公司围绕移动支付、打车应用、移动电子商务展开了激烈的争夺,京东、苏宁等推广钱包服务,采取了支付补贴的方式来吸引客户。

2019 年 6 月 6 日,工信部正式向中国电信、中国移动、中国联通、中国广电发放 5G 商用牌照。随着 5G 的发展,网速的极大提高将带来更加丰富多彩的移动互联网应用。

## 1.3 移动网页开发

由于手机等移动设备更方便携带,能让用户随时随地上网查看,如今手机早已超越计算机成为主要用于浏览网页的终端,CNNIC(中国互联网络信息中心)发布的《第 43 次中国互联网络发展状况统计报告》显示,截至 2018 年 12 月,我国手机网民规模达 8.17 亿,网民通过手机接入互联网的比例高达 98.6%。

在移动应用开发中,最初是以原生 App 的开发为主流的,但是,随着 HTML5 技术的不断发展,越来越多的应用开始转向 Web 浏览器端,出现了多种移动 Web 开发方式。当前,针对移动端的开发方式有 3 种,其应用举例如图 1-4 所示。

(1) Web App(移动网页):在移动 Web 浏览器中运行的 Web 应用,如图 1-4(a)所示的手机搜狐网站,其网址为 <http://m.sohu.com>。

(2) Native App(原生应用):用 Android 和 iOS 的原生语言开发的移动应用,如图 1-4(b)所示的医院网上挂号应用是基于 Android 开发的 App。

(3) Hybrid App(混合应用):将移动网页封装在原生外壳中,以 App 的形式与用户交互。图 1-4(c)所示是在微信的腾讯新闻公众号中打开的新闻页面,该页面的源即为网页。



(a) 手机搜狐网站

(b) 医院网上挂号应用

(c) 腾讯新闻公众号

图 1-4 移动应用示例

表 1-1 列出了上述 3 种开发方式的特点和区别。

表 1-1 移动端开发方式比较

比较项	开发方式		
	Web App	Native App	Hybrid App
开发成本	低	高	中
维护更新	简单	复杂	简单
商店认可	不认可	认可	认可
安装	不需要	需要	需要
跨平台	优	差	中

从表 1-1 可以看出,Web App 这种开发方式具有开发成本低、维护更新简单、无须安装、跨平台等优点,在实际中得到越来越多的使用。这为响应式网页的开发提供了广阔的发展空间。

## 1.4 响应式网页设计概述

随着移动产品的日益丰富,出现了各种屏幕尺寸的手机、平板电脑等移动设备,如果针对每一种尺寸的设备都独立开发一个网站,成本会非常高。于是,响应式网页设计应运而生。值得一提的是,响应式网页设计不仅适用于各种移动设备,还可以直接用于 PC 端,响应式网页设计成为移动互联网开发的热点。

### 1.4.1 设计理念

响应式网页设计是一种针对任意设备都可以对网页内容进行完美布局的显示方式,在设计中要体现以下两点。

#### 1. 一套设计,到处使用

“一套设计,到处使用”是响应式网页设计的核心,即只做一次开发,就可以在各种屏幕上进行友好显示,该设计思想可以在成本、设计、性能之间达到很好的平衡。

#### 2. 移动优先

传统的网站大多数是先开发 PC 端,再根据 PC 端的网页及功能设计开发移动端。从用户体验来看,通常网站会在移动浏览器上缩放,这样虽然可以完整地呈现给人们想要浏览的内容,但鉴于移动设备屏幕大小的限制,过多的内容会使页面看起来杂乱不堪,用户也很难找到自己关注的内容。

随着移动互联网的发展,使用移动端上网的用户群早已经赶超 PC 端。针对移动端屏幕小、计算资源低等特点,如果先开发移动端,可以迫使开发人员在有限的环境中设计产品功能。这样做,一方面可以使产品的功能更加核心和简洁,另一方面有助于设计出性能更高的程序。因此,响应式网页设计并不是将整个网页缩放给用户,而是经过精心筛选组织,友好地将页面内容呈现给用户。

### 1.4.2 设计原则

在响应式网页设计中,需遵循以下两项原则。

#### 1. 兼容主流设备屏幕

响应式网页即让网页在各种设备屏幕上自适应显示,因此,在设计网页时,须考虑设备屏幕对它的支持显示,具体说来,须了解与屏幕相关的指标术语,再针对相关参数进行设计。

(1) 屏幕尺寸。屏幕尺寸是衡量设备大小的单位,它通过设备屏幕对角线的长度来表示,单位是英寸。图 1-5 所示的华为荣耀 7 的屏幕尺寸是 5.2 英寸。

(2) 屏幕分辨率。屏幕分辨率是指屏幕竖向的高度和横向的宽度上具有的像素块数,如图 1-6 所示。分辨率常用长(px)×宽(px)来表示,px 是像素单位。图 1-5 所示的华为荣耀 7 的分辨率是 1 920×1 080,即表示在 5.2 英寸的屏幕中,竖向的高度有 1 920 个像素块,横向的宽度有 1 080 个像素块。

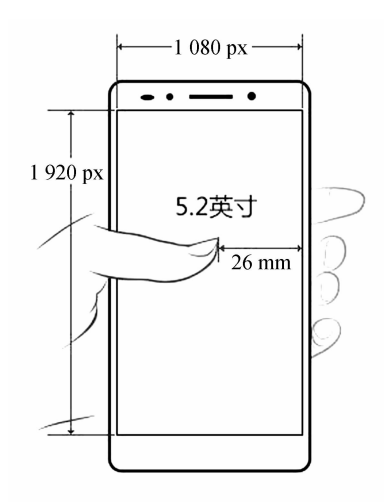


图 1-5 屏幕尺寸和分辨率示例

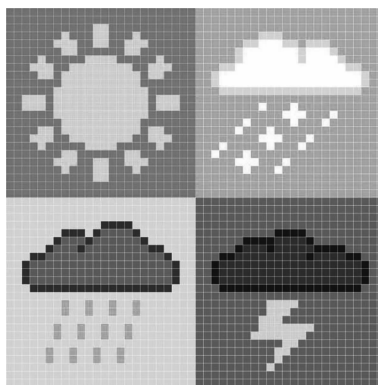


图 1-6 像素块

(3)屏幕尺寸比。屏幕尺寸比就是屏幕在高度上的像素与宽度上的像素的比例,所以也称分辨率比,如华为荣耀 7 的屏幕尺寸比=1 920/1 080=16 : 9。

(4)屏幕像素密度。屏幕像素密度是每英寸屏幕所拥有的像素数,英文简称 PPI,它反映了设备显示的精细程度。该指数与屏幕尺寸和分辨率的关系可通过如下公式表达。

$$\text{PPI} = \frac{\sqrt{\text{横向像素数}^2 + \text{纵向像素数}^2}}{\text{屏幕尺寸}}$$

从公式中可以看出,屏幕尺寸固定,分辨率越高,PPI 越大;分辨率固定,屏幕尺寸越大,PPI 越小。

表 1-2 和表 1-3 分别列出了当前主流液晶显示器和手机(平板)屏幕的尺寸和分辨率。

表 1-2 主流液晶显示器屏幕尺寸和分辨率

尺 寸	分辨率/px
18.5 英寸	1 366×768
19 英寸	1 440×900
20 英寸	1 600×900
21.5 英寸	1 920×1 080
22 英寸	1 680×1 050
23 英寸	1 920×1 080
23.6 英寸	1 920×1 080
24 英寸	1 920×1 080
24 英寸	1 920×1 200
27 英寸	1 920×1 080
27 英寸	2 560×1 440
30 英寸	2 560×1 600

表 1-3 主流手机(平板)屏幕参数和代表机型

尺寸	分辨率/px	代表机型
4 英寸	1 136×640	iPhone 5&5s
4.3 英寸	800×480	三星 I9100
4.5 英寸	1 280×720	华为 D1
5 英寸	1 920×1 080	三星 Galaxy 4、SONY L36h、小米 3&4、华为 P7& 荣耀 6
5.5 英寸	1 920×1 080	iPhone 6 Plus
6 英寸	2 160×1 080	魅族 16th
6.39 英寸	3 120×1 440	华为 Mate 20 Pro
6.39 英寸	2 340×1 080	小米 9
7.9 英寸	2 048×1 536	iPad Mini 2、iPad Mini 5
8 英寸	1 920×1 200	小米平板 4
8.4 英寸	2 560×1 600	华为 M5
9.7 英寸	2 048×1 536	iPad 3&4&Air
10.5 英寸	2 560×1 600	三星 Galaxy Tab S 10.5 系列

根据表 1-2 和表 1-3 列出的各屏幕参数,可将屏幕分为超小屏幕、小屏幕、中等屏幕、大屏幕四大类,具体如表 1-4 所示。在设计响应式网页时,需考虑适配表 1-4 所示的多种屏幕。

表 1-4 屏幕分类

屏幕类型	宽度
超小屏幕	<768 px
小屏幕	≥768 px
中等屏幕	≥992 px
大屏幕	≥1 200 px

## 2. 兼容主流浏览器

响应式网页开发的基础是 HTML5,由于 HTML5 较 HTML4 是一个全新的标准,增加了很多新的特性,对多媒体的支持也更全面。但是,各大浏览器对它的支持并不完全一致,因此,浏览器对 HTML5 的支持是判断网页兼容性的重要指标。

当前浏览器种类繁多,支持 HTML5 的主流浏览器有 Chrome、Firefox、Safari、Opera、IE 等,各浏览器对 HTML5 的具体支持程度请参见下节中浏览器对 HTML5 的支持简介。在响应式网页设计中,须考虑浏览器的支持和兼容显示问题。

### 1.4.3 设计相关技术

响应式网页设计的核心技术包括媒体查询、弹性视觉媒体和流式布局等。



### 1. 媒体查询

媒体查询是基于 CSS3 的一种特性,可以根据视口宽度、设备方向等差异来改变页面的显示方式。响应式网页布局时,通过优先建立移动客户体验,利用媒体查询调整大屏幕,按照移动端优先设计原则保证 Web 为不同类型的客户提供最佳的客户体验。

### 2. 弹性视觉媒体

弹性视觉媒体是通过手指滑动进行屏幕调整,使页面根据用户的需要进行智能化调整,不需要通过拖动滚动条等浏览网页,从而页面具有可变性。该技术可通过弹性盒模型简单地实现,开发人员只需要声明布局应该具有的行为,而不需要给出具体的布局需求。

### 3. 流式布局

流式布局可以根据浏览器的宽度和屏幕的大小自动调整显示效果,让子元素按比例限制在其父元素范围内,相应于布局的其他部分按比例缩放,以适应其设计的大小。

本书通过 HTML5、jQuery Mobile、Bootstrap 讲述响应式网页设计的核心技术。其中,HTML5 篇主要包括 HTML5+CSS3 的基本网页设计、HTML5 针对移动开发提供的离线存储和地理定位等内容;jQuery Mobile 篇针对响应式网页开发中移动优先的理念,讲述 JavaScript 框架 jQuery Mobile,主要包括 jQuery Mobile 的布局、样式、小部件等;Bootstrap 篇是为提高开发效率引入的响应式网站构建框架,主要包括 Bootstrap 布局、样式、组件等。

在本书的篇章布局中,HTML5 篇是基础,jQuery Mobile 和 Bootstrap 篇是延伸和拓展,在实际的应用开发中,常常需要多种技术混合使用。全书着重讲述网页的响应式布局,但不局限于此,通过适时地引入网页交互的内容,让内容更贴近现实应用。

注意,除了 jQuery Mobile 和 Bootstrap 外,还有其他响应式网页设计框架,但是在此介绍的两个框架是响应式网页开发的基础和当前流行的框架。

## 1.5 开发测试工具

### 1.5.1 开发工具

#### 1. Dreamweaver 简介

Dreamweaver 是集网页制作和网站管理于一身的所见即所得网页代码编辑器,最初由美国 Macromedia 公司开发。作为全球最流行、最优秀的网页编辑器,Dreamweaver 可以轻而易举地制作出跨操作系统平台、跨浏览器的充满动感的网页,是开发 Web 应用程序的理想工具。

Dreamweaver 在发展过程中经过了表 1-5 所示的版本变迁。

表 1-5 Dreamweaver 历史版本一览表

Macromedia 时代	Dreamweaver 1.0
	Dreamweaver 2.0
	Dreamweaver 2.01
	Dreamweaver 3
	Dreamweaver 4
	Dreamweaver 5
	Dreamweaver 6.0
	Dreamweaver MX
	Dreamweaver MX 2004
	Dreamweaver 8.0
Adobe 时代	Dreamweaver CS3
	Dreamweaver CS4
	Dreamweaver CS5
	Dreamweaver CS5.5
	Dreamweaver CS6
	Dreamweaver CreativeCloud(CC)
	Dreamweaver CC 2015
	Dreamweaver CC 2017
	Dreamweaver CC 2018

表 1-5 中所列的 Dreamweaver,按照自上而下的顺序,各版功能不断完善强大,最新推出的 Dreamweaver CC 新增了更加友善、直观的视觉化 CSS 编辑工具,创新的 jQuery UI Widget 和 Edge Web Fonts,CSS3 转换,完善的 HTML5、jQuery 和 jQuery Mobile 支持等。

但是,由于 Dreamweaver CC 对操作系统的兼容性和自身稳定性不如 Dreamweaver CS,所以,本书选用 CS 的最高版本 Dreamweaver CS6 作为开发环境,其安装按照提示完成即可。

## 2. Dreamweaver CS6 的使用

通过 Dreamweaver CS6 创建响应式网页时,选择页面类型为 HTML,文档类型为 HTML 5 即可,如图 1-7 所示。

为实现“所见即所得”,Dreamweaver CS6 提供了实时视图模拟网页在浏览器中打开的效果和通过多屏预览模拟网页在不同屏幕上的显示效果,如图 1-8 所示。但是,由于只有在实际的浏览器中显示的效果才是准确的,所以,在此未选用该功能,运行调试都在真实的浏览器中进行。

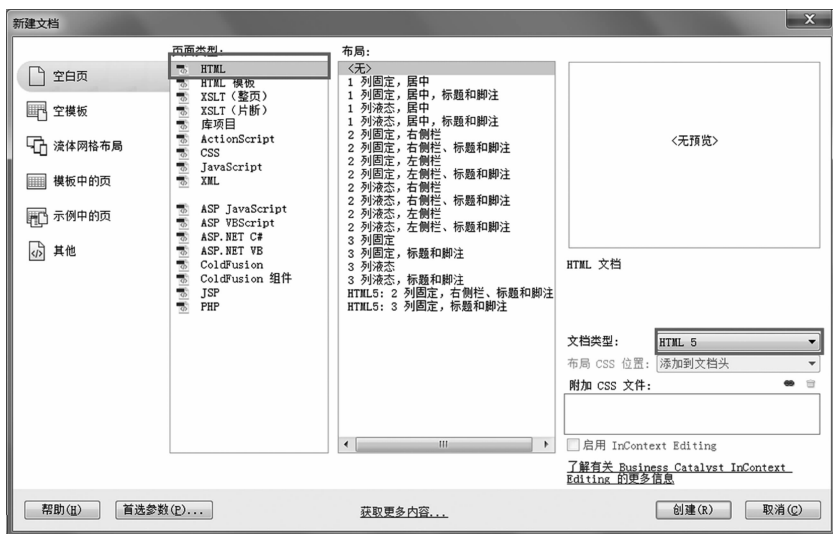


图 1-7 Dreamweaver CS6 新建文档对话框

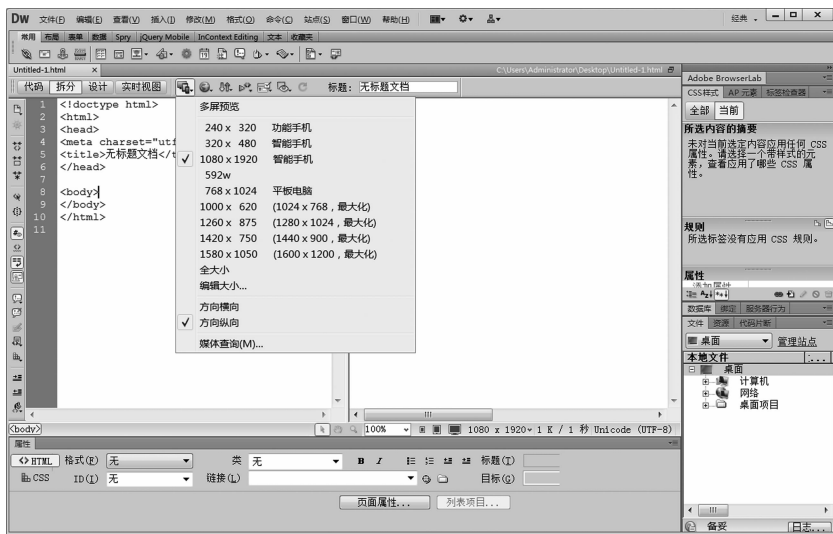


图 1-8 Dreamweaver CS6 模拟预览功能

## 1.5.2 测试工具

### 1. 浏览器对 HTML5 的支持简介

目前,市场上浏览器种类繁多,PC 端常见的浏览器有 IE、Firefox、Chrome、Opera 等,移动端常用的浏览器有 Android Browser、Mobile Safari、Chrome 等。

2014 年,W3C 正式宣布 HTML5 的标准规范已经制定完成,但是 HTML5 的标准化进程或许要到 2020 年才能最终完成,所以,各浏览器在对 HTML5 的支持上各有差异。

表 1-6~表 1-10 从不同的角度说明了 IE、Chrome、Firefox、Safari、Opera 五大主流浏览器的部分版本对 HTML5 和 CSS3 的支持情况。浏览器的各版本是向后兼容的,且越新推

出的版本对 HTML5 和 CSS3 的支持越好。

表 1-6 五大主流浏览器对 HTML5 Web 应用程序的支持情况

支持项	浏览器及其版本									
	IE 10					Chrome 3.6	Firefox 4	Safari 5	Opera 11.1	
Local Storage	×	×	√	√	√	√	√	√	√	√
Session Storage	×	×	√	√	√	√	√	√	√	√
Post Message	×	×	√	√	√	√	√	√	√	√
Offline Applications	×	×	×	×	×	√	√	√	√	√
Workers	×	×	×	×	×	√	√	√	√	√
Query Selector	×	×	√	√	√	√	√	√	√	√
WebSQL Database	×	×	×	×	×	√	×	×	√	√
IndexDB Database	×	×	×	×	×	×	×	×	×	×
Drag and Drop	√	√	√	√	√	√	√	√	√	×
Hash Change	×	×	√	√	√	√	×	√	√	√
History Management	×	×	×	×	×	√	×	√	√	×
WebSockets	×	×	×	×	×	√	×	×	√	×
Geolocation	×	×	×	√	√	√	√	√	√	√
Touch	×	×	×	×	×	×	×	×	×	×

表 1-7 五大主流浏览器对 HTML5 图形和内嵌内容的支持情况

支持项	浏览器及其版本									
	IE 10					Chrome 3.6	Firefox 4	Safari 5	Opera 11.1	
Canvas	×	×	×	√	√	√	√	√	√	√
Canvas Text	×	×	×	√	√	√	√	√	√	√
SVG	×	×	×	√	√	√	√	√	√	√
SVG Clipping Paths	×	×	×	√	√	√	√	√	√	√
SVG Inline	×	×	×	√	√	√	×	√	×	×
SMIL	×	×	×	√	√	√	×	√	√	√
WebGL	×	×	×	√	√	√	×	√	√	√
Audio	×	×	×	√	√	√	√	√	√	√
Video	×	×	×	√	√	√	√	√	√	√

表 1-8 五大主流浏览器对 HTML5 表单 input 类型的支持情况

支持项	浏览器及其版本									
	IE 10					Chrome 3.6	Firefox 4	Safari 5	Opera 11.1	
input;search	×	×	×	×	×	√	×	√	√	√
input;phone	×	×	×	×	×	√	×	√	√	√
input:url	×	×	×	×	×	√	×	√	√	√
input;email	×	×	×	×	×	√	×	√	√	√
input;datetime	×	×	×	×	×	×	×	×	×	√
input;date	×	×	×	×	×	×	×	×	×	√
input;month	×	×	×	×	×	×	×	×	×	√
input;week	×	×	×	×	×	×	×	×	×	√
input;time	×	×	×	×	×	×	×	×	×	√
input;localtime	×	×	×	×	×	×	×	×	×	√
input:number	×	×	×	×	×	√	×	×	×	√
input;range	×	×	×	×	×	√	×	×	√	√
input;color	×	×	×	×	×	×	×	×	×	√

表 1-9 五大主流浏览器对 HTML5 表单属性的支持情况

支持项	浏览器及其版本									
	IE 10					Chrome 3.6	Firefox 4	Safari 5	Opera 11.1	
autocomplete	×	×	×	×	×	×	×	√	×	√
autofocus	×	×	×	×	×	√	×	√	√	√
list	×	×	×	×	×	×	×	√	√	√
placeholder	×	×	×	×	×	√	×	√	√	√
min	×	×	×	×	×	√	×	×	√	√
max	×	×	×	×	×	√	×	×	√	√
multiple	×	×	×	×	×	√	×	√	√	√
pattern	×	×	×	×	×	√	×	√	√	√
required	×	×	×	×	×	√	×	√	√	√
step	×	×	×	×	×	√	×	×	√	√

表 1-10 五大主流浏览器对 CSS3 新增属性的支持情况

支持项	浏览器及其版本									
	IE 10					Chrome 3. 6	Firefox 4	Safari 5	Opera 11. 1	
background-size	×	×	×	√	√	√	×	√	√	√
border-image	×	×	×	×	×	√	√	√	√	√
border-radius	×	×	×	√	√	√	√	√	√	√
box-shadow	×	×	×	√	√	√	√	√	√	√
text-shadow	×	×	×	×	×	√	√	√	√	√
box-flex	×	×	×	×	√	√	√	√	√	×
CSS Animations	×	×	×	×	×	√	×	×	√	×
CSS Columns	×	×	×	×	√	√	√	√	√	√
CSS Transforms	×	×	×	√	√	√	√	√	√	√
CSS Transforms 3D	×	×	×	×	×	×	×	×	×	×
CSS Transitions	×	×	×	×	×	√	×	√	√	√
CSS FontFace	√	√	√	√	√	√	√	√	√	√

观察表 1-6~表 1-10 可知,作为全球使用用户最多的 IE 浏览器的早期版本并未提供对 HTML5 的支持,IE 9 才开始支持标准,Chrome、Firefox、Safari、Opera 支持 HTML5 很多年,且支持度较好,但各有差异,如 Opera 对 HTML5 表单的支持优于其他浏览器,Chrome 对 HTML5 音频和视频的支持较好。总体说来,当前对 HTML5 和 CSS3 支持最好的是 Chrome,Safari 次之,Firefox 和 Opera 旗鼓相当。

## 2. Chrome 浏览器

Chrome 是一款由 Google 公司开发的免费网页浏览器,包括 WebKit 和 Mozilla 引擎,最初用于桌面计算机上,其 beta 测试版本在 2008 年 9 月 2 日发布,之后不断更新,其以简洁、快速、安全、健壮等特点得到广泛使用。

2012 年,Chrome 浏览器移动版发布,为 iOS、Android 等系统提供使用,在保持浏览器原有特点的情况下,实现了多终端使用浏览器,具有共享收藏历史信息等功能,是手机浏览器的一次巨大突破。

在此,选用 Chrome 浏览器的 V 67. 0. 3396. 62 作为书中示例的运行环境,并且选用 Chrome 浏览器自带的调试工具 Chrome DevTools 来做响应式测试。Chrome 浏览器可通过 Internet 免费下载,在此需根据你的系统选择下载 32 位或 64 位的安装文件。当然,若是 64 位系统,32 位和 64 位的安装文件均可。下面以 Ghost 中文网(网址为 <http://www.ghostchina.com/>)响应式网站为例,说明其使用方法。

启动 Chrome 后,在地址栏输入 <http://www.ghostchina.com/> 并回车,将看到 Ghost 中文网在 PC 屏幕显示的效果,如图 1-9 所示。其中,PC 显示器的尺寸为 27 英寸,分辨率为 1 920×1 080。



图 1-9 网站示例在 PC 屏幕显示效果

在 Chrome 浏览器中,按快捷键 F12 或执行“工具”→“更多工具”→“开发者工具”命令启用 Chrome DevTools,按照表 1-4 中的屏幕适配分类,分别选取超小屏幕和小屏幕代表 Galaxy S5 和 iPad,Ghost 中文网的显示效果如图 1-10 和图 1-11 所示。



图 1-10 网站示例在超小屏幕显示效果



图 1-11 网站示例在小屏幕显示效果

本书所有示例将分别适配表 1-4 中的超小屏幕、小屏幕和中等屏幕进行浏览显示,以测试其自适应性。

# 2

## 第 2 章

# HTML5 概述

### 2.1 HTML 的定义及版本介绍

#### 1. 定义

HTML(hypertext markup language,超文本标记语言)是一种用来制作超文本文档的简单标记语言。简单来讲,HTML 是构建一套标记符号和语法规则,将所要显示的文字、图像、声音等要素按照一定的标准要求排放,形成一定的标题、段落、列表等单元。

#### 2. HTML 版本

HTML 1.0——在 1993 年 6 月作为国际互联网工程任务组(IETF)工作草案发布(并非标准)。

HTML 2.0——1995 年 11 月作为 RFC 1866 发布,在 RFC 2854 于 2000 年 6 月发布之后被宣布过时。

HTML 3.2——1996 年 1 月 14 日,W3C 推荐标准。

HTML 4.0——1997 年 12 月 18 日,W3C 推荐标准。

HTML 4.01——微小改进,1999 年 12 月 24 日,W3C 推荐标准。

HTML5 的第一份正式草案已于 2008 年 1 月 22 日公布,目前仍在继续完善。

#### 3. HTML5 新增元素和删除元素

HTML5 是当前 HTML 规范的最新版本,它不仅有良好的用户体验,而且取消了一些过时标签和新增了一些标签,还将代码化繁为简,严格遵循“简单至上”的原则。

##### 1) HTML5 新增的元素

与 HTML4 相比,HTML5 新增的元素如表 2-1 所示。



表 2-1 HTML5 新增的元素

标 签	含 义
<section>	标签定义文档中的节(section、区段)。例如,章节、页眉、页脚或文档中的其他部分
<article>	标签规定独立的自包含内容
<aside>	标签表示 article 标签内容之外的、与 article 标签内容相关的辅助信息
<header>	标签定义文档的页眉(介绍信息)
<hgroup>	标签用于对网页或区段(section)的标题进行组合
<footer>	标签表示整个页面或页面中一个内容区块的脚注,一般来说,它包含创作者的姓名、创作日期及创作者的联系信息
<figure>	标签规定独立的流内容(图像、图表、代码等)
<figcaption>	标签定义 figure 元素的标题(caption)
<canvas>	标签定义图形,如图表和其他图像。该标签基于 JavaScript 的绘图 API
<video>	标签定义视频,如电影片段或其他视频流
<audio>	标签定义声音,如音乐或其他音频流
<embed>	标签定义嵌入的内容,如插件
email	必须输入 E-mail
url	必须输入 url 地址
number	必须输入数值
range	必须输入一定范围内的数值
Date Pickers(日期选择器)	拥有多个可供选取日期和时间的新输入类型
search	用于搜索域,域显示为常规的文本域
color	颜色选择器
<mark>	标签定义带有记号的文本。在需要突出显示文本时使用
<progress>	标签标示任务的进度(进程)
<ruby>	标签定义
<canvas>	标签定义图形,如图表和其他图像
<details>	标签用于描述文档或文档某个部分的细节
<keygen>	标签规定用于表单的密钥对生成器字段
<output>	标签定义不同类型的输出,如脚本的输出

## 2) HTML5 删除的元素

相较于 HTML4, HTML5 删除的元素如表 2-2 所示。

表 2-2 HTML5 删除的元素

标 签	含 义
能被 CSS 代替的标签	basefont、big、center、font、s、strike、tt、u。这些标签纯粹是为画面展示服务的, HTML5 中提倡把画面展示性功能放在 CSS 中统一编辑
不再使用 frame 框架	HTML5 不再支持, 只支持 iframe 框架
只有部分浏览器支持的标签	部分浏览器支持 applet、bgsound、blink、marquee 等标签
其他被废除的标签	废除 rb, 使用 ruby 替代; 废除 acronym, 使用 abbr 替代; 废除 dir, 使用 ul 替代; 废除 isindex, 使用 form 与 input 相结合的方式替代; 废除 listing, 使用 pre 替代; 废除 xmp, 使用 code 替代; 废除 nextid, 使用 guids; 废除 plaintex, 使用“text/plian”(无格式正文) MIME 类型替代

## 2.2 HTML5 发展历史

HTML 是一种标记性语言, 自 1999 年的 HTML4 版本后, 一直未推出新的标准。为了推动 Web 前端的发展, 一些致力于发展 Web App 的公司另行成立了 WHATWG 组织, 直到 2007 年, W3C 从 WHATWG 接手相关工作才开始发展 HTML5。由此可见, HTML5 的发展史, 有用户的需求在推动, 有技术开发者的需求在推动, 更有巨大的商业利益在推动。

HTML5 自诞生以来, 大致经历了两个发展阶段, 分别是 Web 增强阶段和移动互联网阶段。Web 增强阶段主要从丰富 Web 体验上扩展 HTML, 具体表现在以下几个方面。

(1) HTML5 新增了离线存储、更丰富的表单(如 input type="date")、js 线程、标准扩展 embed, 以及很多 CSS3 新语法。

(2) HTML5 新增了 audio、video 流媒体支持。

(3) HTML5 新增了 canvas、header 和 section 等标签。

随着移动互联网的发展, HTML5 的跨平台优势进一步显现出来。HTML5 是唯一一个可以在 Windows、macOS、iOS、Android 等主流平台使用的跨平台语言, 此即 HTML5 的移动互联网发展阶段。

## 2.3 HTML5 文档基本格式

HTML5 文档包括以下几个部分。

(1) `<!doctype html>` 用于向浏览器说明当前文档使用的 HTML 版本为 HTML5。

(2) `<html></html>` 分别表示文档的开始和结束,用于告知浏览器其自身是一个 HTML 文档。

(3) `<head></head>` 为头部标签,用于定义 HTML 文档的头部信息,紧跟在 `<html>` 标签之后,包括 `<title>`、`<meta>`、`<link>`、`<style>` 等标签内容。

(4) `<body></body>` 为主体标签,用于定义在浏览器中显示的网页内容,如图片、视频、音频、文本等。

下面以一个简单的 HTML5 百科(图 2-1)介绍为例,介绍 HTML 文档的基本格式。

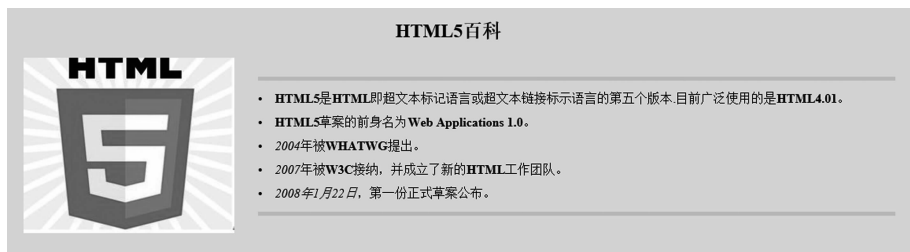


图 2-1 HTML5 百科介绍

网页显示的 HTML5 代码如下。

```
<!doctype html><!--文档类型声明,不区分大小写,告诉浏览器当前的文档类型-->
<html lang="zh-cn"><!--表示 html 文档开始,并告诉浏览器字符集为中文-->
<head><!--包含文档元数据开始-->
<meta content="width=device-width, initial-scale=1.0, minimum-scale=1.0, maximum-scale=
1.0,user-scalable=no" name="viewport" /><!--让网页在手机端正常显示-->
<title>HTML5 百科</title><!--文档标题-->
<style><!--文档的 CSS 样式-->
img {float: left;margin-left: 10%;}
ul {width: 60%;float: left;padding-left: 30px;list-style: inside;}
ul li {line-height: 30px;}
</style>
</head><!--包含文档元数据结束-->
<body><!--表示 html 内容部分开始,也就是可见部分-->
<h2 align="center">HTML5 百科</h2>

```

```
<ul>
  <hr size="3" color="#CCCCCC" >
  <li><strong>HTML5</strong>是<strong>HTML</strong>即超文本标记语言或超文本
  链接标示语言的第五个版本。目前广泛使用的是<strong>HTML4.01</strong>。</li>
  <li><strong>HTML5</strong>草案的前身名为<strong>Web Applications 1.0</strong>。
</li>
  <li><em>2004</em>年被<strong>WHATWG</strong>提出。</li>
  <li><em>2007</em>年被<strong>W3C</strong>接纳,并成立了新的<strong>HTML
</strong>工作团队。</li>
  <li><em>2008年1月22日</em>,第一份正式草案公布。</li>
  <hr size="3" color="#CCCCCC" >
</ul>
</body><!--表示 html 内容部分结束-->
</html><!--表示 html 文档结束-->
```

## 3

## 第 3 章

## HTML5 新增页面元素和属性

## 3.1 旅游网页界面设计

## 3.1.1 任务目标

- 掌握 HTML5 结构元素的使用方法。
- 掌握分组元素的使用方法。
- 掌握 HTML5 移动端布局的方法。
- 掌握单位 em 和 rem 的使用方法。

## 3.1.2 任务描述

利用 HTML5 结构元素制作图 3-1 所示的旅游网页。



图 3-1 旅游网页效果

在 Chrome 浏览器中预览,得到图 3-1 所示的显示效果。同时,分别用 Chrome 模拟器 Galaxy S5 和 iPad,得到自适应的网页显示效果,如图 3-2 和图 3-3 所示。



图 3-2 网页在 Galaxy S5 模拟器中显示效果



图 3-3 网页在 iPad 模拟器中显示效果

### 3.1.3 任务知识

#### 1. 结构元素

(1) header: 从语义化上视为文档的页眉, header 用法如下。

- ①一种具有引导和导航作用的结构元素。
- ②通常为放置在整个页面或页面内的一个内容区块的标题。
- ③一个网页内并没有限制 header 标签的个数。

```
<header>
<h1>我是大头</h1>
</header>
<article>
<header>
<h1>我是脖子</h1>
</header>
<p>我是身体</p>
</article>
```

(2) nav: 从语义化上视为导航, nav 用法如下。

- ①通常作为页面导航的链接组。
- ②侧边栏导航。

```
<nav>
<ul>
```

```

<li><a href="#">菜单 1</a></li>
<li><a href="#">菜单 2</a></li>
<li><a href="#">菜单 3</a></li>
</ul>
</nav>

```

(3) article: 从语义化上看为文档、页面, article 用法如下。

- ①通常是一篇文章、一个页面或一个独立完整的内容模块。
- ②一般会带一个标题, 并放在 header 标签中。
- ③article 元素可以互相嵌套。

```

<article>
<header>
<h1>我的标签</h1>
</header>
<p>我的段落</p>
<article>
<div>我的内容</div>
</article>
</article>

```

(4) aside: 从语义化上看为在旁边、侧边, aside 用法如下。

①在 article 标签内使用时, 作为主要内容的附属信息部分, 如有关的参考资料、名词解释等。

②在 article 标签外使用时, 作为页面或站点全局的附属信息部分, 如侧边栏、博客的友情链接部分、广告区域等。

```

<!--在 article 标签内使用时-->
<article>
<h1>马云是谁</h1>
<p>马云,男,1964年9月10日出生于浙江省杭州市,中国著名企业家,阿里巴巴集团、淘宝网、支付宝创始人……</p>
<aside>
<h1>参考资料</h1>
<p>百度网、维基百科……</p>
</aside>
</article>

```

(5) section: 从语义化上看为部分, section 用法如下。

- ①用于页面内容的独立分块, 往往是文章的一段。
- ②通常由内容和标题组成, 没有标题的内容不推荐使用 section。

**注意:** 一个容器需要被定义样式或为脚本定义行为时, 推荐用 div 而非 section, 不要将 section 用作设置样式的容器。

```

<section>
  <h1>水果</h1>
  <article>
    <h2>苹果</h2>
    <div>苹果是啥? </div>
  </article>
  <article>
    <h2>橘子</h2>
    <div>橘子是啥? </div>
  </article>
</section>
<!--article 可以看成一种特殊种类的 section 元素,它比 section 更强调独立性-->
<article>
  <h1>中国人物</h1>
  <p>三国、两晋</p>
  <section>
    <h2>三国</h3>
    <p>猛将猛将猛将猛将</p>
  </section>
  <section>
    <h2>两晋</h3>
    <p>猛将猛将猛将猛将</p>
  </section>
</article>

```

(6) footer:从语义化上看为文档的脚注,footer用法如下。

- ①一个内容块区的脚注。
- ②通常内容为联系信息、相关阅读、版权信息等。

```

<footer>
  <ul>
    <li>关于我们</li>
    <li>版权信息</li>
    <li>注意事项</li>
  </ul>
</footer>

```

## 2. 分组元素

(1) figure。定义媒介内容的分组及它们的标题,figure用法如下。

- ①标签规定独立的流内容(图像、图表、照片、代码等)。
- ②元素的内容应该与主内容相关,但若被删除,则不应对文档流产生影响。

```

<figure>
  <p>我的大学</p>

```



```
<p>拍摄者: ** </p>

</figure>
```

(2) figcaption。figcaption 用法如下。

- ① 标签定义 figure 元素的标题(caption)。
- ② figcaption 元素应该被置于 figure 元素的第一个或最后一个子元素的位置。

```
<figure>
<figcaption>我的大学</figcaption>

</figure>
```

(3) hgroup: 定义有关文档中的 section 的信息, hgroup 用法如下。

标签用于对网页或区段(section)的标题进行组合。

```
<hgroup>
<h2>旅行的意义……</h2>
<h3>发现美</h3>
<h3>感受美</h3>
</hgroup>
```

(4) 相关 CSS3 样式。

- ① box-sizing: content-box | border-box | inherit。

表 3-1 列出了 box-sizing 的取值及含义。

表 3-1 box-sizing 的取值及含义

取 值	解 释
content-box	这是由 CSS2.1 规定的宽度和高度行为;宽度和高度分别应用到元素的内容框;在宽度和高度之外绘制元素的内边距和边框
border-box	为元素设定的宽度和高度决定了元素的边框盒。也就是说,为元素指定的任何内边距和边框都将在已设定的宽度和高度内进行绘制。通过从已设定的宽度和高度分别减去边框和内边距才能得到内容的宽度和高度
inherit	继承父元素 box-sizing 属性的值

使用时应注意浏览器的兼容性,下面是各个浏览器能识别的样式。

```
-webkit-box-sizing:100px;//forios-safari,android
-moz-box-sizing:100px;//forff
box-sizing:100px;//forother
<style>
.content-box{
box-sizing:content-box;
-moz-box-sizing:content-box;
width:100px;
```

```

height:100px;
padding:20px;
border:5pxsolid#E6A43F;
}
.border-box{
box-sizing:border-box;
-moz-box-sizing:border-box;
width:100px;
height:100px;
padding:20px;
border:5pxsolid#3DA3EF;
}
</style>
Body 内容为:
<div class="content-box">在宽度和高度之外绘制元素的内边距和边框</div>
<div class="border-box">在已设定内进行绘制</div>

```

效果图如图 3-4 所示。

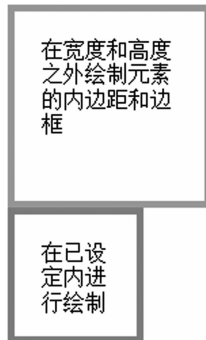


图 3-4 box-sizing 效果图

②box-shadow:h-shadow v-shadow blur spread color inset。

box-shadow 的含义是添加一个或多个阴影,其取值及含义如表 3-2 所示。

表 3-2 box-shadow 取值及含义

取 值	解 释
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
inset	可选。将外部阴影(outset)改为内部阴影

```
<style>
div{
  box-shadow:0 0 10px rgba(0,0,0,0.4);/* 盒子阴影:水平偏移,垂直偏移,模糊值,阴影颜色(其中
rgb 指颜色,a 指透明度) */}
</style>
```

(5) 移动端布局。

① meta 元素。

```
<meta name = " viewport " content = " width = device-width, initial-scale = 1. 0,
minimum-scale=1. 0,maximum-scale=1. 0,user-scalable=no">
```

解释:

- a. 窗口设定 name="viewport"。
- b. 页面大小与屏幕等宽 width=device-width。
- c. 初始缩放比例 initial-scale=1.0。
- d. 允许缩放最小比例 minimum-scale=1.0。
- e. 允许缩放最大比例 maximum-scale=1.0。
- f. 用户是否可以缩放 user-scalable=no。

② 不使用绝对宽度。所谓不使用绝对宽度,就是说 CSS 代码不能指定像素宽度,如 width:××px,只能使用最大宽度(max-width)和最大高度(max-height)。

③ @media。@media 规则用于同一个 CSS 文件,根据不同的屏幕分辨率,选择不同的 CSS 规则,后面章节会进行详细介绍。

④ 单位 em 和 rem。em 是相对长度单位,是相对于当前对象内文本的字体尺寸。若当前对行内文本的字体尺寸未进行设置,则为相对于浏览器的默认字体尺寸。

任意浏览器的默认字体高都是 16 px。所有未经调整的浏览器都符合 1 em=16 px。那么 12 px=0.75 em,10 px=0.625 em。为了简化 font-size 的换算,需要在 CSS 中的 body 选择器中声明 font-size=62.5%,这就使 em 值变为 16 px×62.5%=10 px,这样 12 px=1.2 em,10 px=1 em,也就是说,只需要将原来的 px 数值除以 10,然后以 em 作为单位就可以了。

所以在写 CSS 代码时,需要注意以下 3 点。

- a. body 选择器中声明 font-size=62.5%。
- b. 将原来的 px 数值除以 10,然后以 em 作为单位。
- c. 重新计算那些被放大的字体的 em 数值。避免字体大小的重复声明。

rem 是指相对于根元素的字体大小的单位。这个单位与 em 的区别在于,使用 rem 为元素设定字体大小时,仍然是相对大小,但相对的只是 HTML 根元素。这个单位可谓集相对大小和绝对大小的优点于一身,通过它既可以做到只修改根元素就成比例地调整所有字体大小,又可以避免字体大小逐层复合的连锁反应。

**注意:** ① IE 9/IE 10 在用于伪元素时或使用字体简写声明时不支持 rem。

② Safari 5.0 和 5.1 虽然支持 rem,但是在使用媒体查询时不支持 rem。

### 3.1.4 任务实施

#### 1. 任务分析

根据效果图分析布局,如图 3-5 所示。

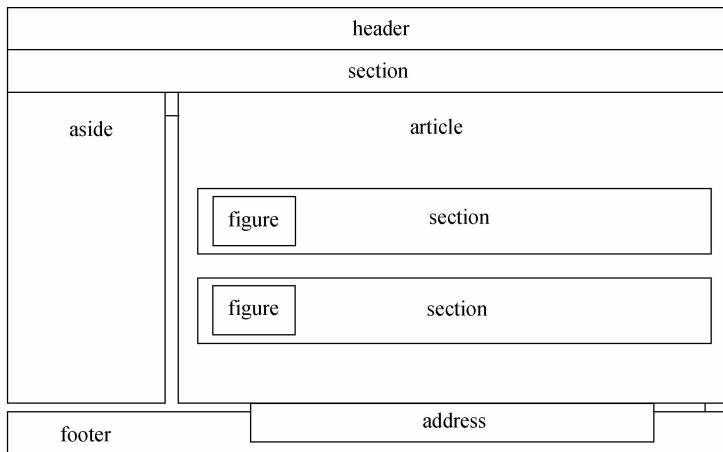


图 3-5 旅游网页布局

#### 2. 任务代码

任务描述中需完成的网页是一个 HTML5 旅游基本页面,详细代码见本书配套资源中的 Task3-1.html,该网页所涉及的 CSS 文件见 Task3-1-1.css 和 Task3-1-2.css。

### 3.1.5 任务实训

利用前面所学的结构元素制作图 3-6 所示的新闻网页。



图 3-6 新闻网页效果

新闻网页的布局如图 3-7 所示,详细代码见 Task3-2. html。

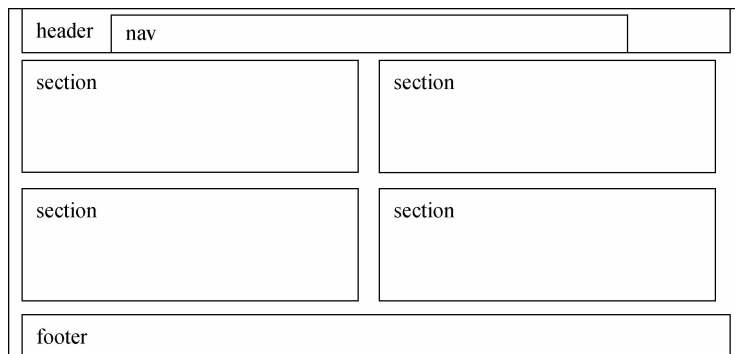


图 3-7 新闻网页布局

## 3.2 学习情况调查表界面设计

### 3.2.1 任务目标

- 掌握页面交互元素的使用方法。
- 掌握文本层次元素的使用方法。
- 掌握全局属性的使用方法。

### 3.2.2 任务描述

利用页面交互元素和文本层次元素制作图 3-8 所示的学习情况调查表网页。

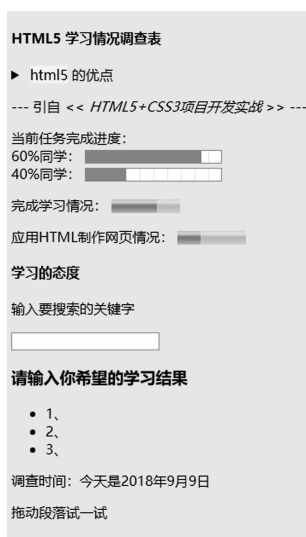


图 3-8 学习情况调查表网页的效果

### 3.2.3 任务知识

#### 1. 页面交互元素

(1) details 标签: 规定了用户可见的或隐藏的补充细节。

① details 标签用来供用户开启关闭的交互式控件。任何形式的内容都能被放在 details 标签里边。

② details 元素的内容对用户是不可见的, 除非设置了 open 属性。

**提示:** 与 <summary> 标签配合使用可以为 details 定义标题。标题是可见的, 用户单击标题时, 会显示出 details。

```
<details>
<summary><mark>html5</mark>的优点</summary>
<p>1. 取消过时标签, 新增一些标签</p>
<p>2. 解决浏览器兼容问题</p>
<p>3. 代码化繁为简</p>
</details>
```

(2) progress 标签: 定义运行中的任务进度。

```
<p>
当前任务完成进度:
<progress value="85" max="100">85% </progress>
</p>
```

(3) meter 标签: 定义度量衡, 仅用于已知最大和最小值的度量。例如, 磁盘使用情况、查询结果的相关性等。注意: <meter> 不能作为一个进度条来使用, 进度条标签为 <progress>。

**注意:** IE 浏览器不支持 meter 标签。

optimum 的值小于 low 值时显示绿色, 大于 high 值时显示红色, 在 low 值和 high 值之间时是黄色。

```
<p>
完成学习情况:<meter value="80" max="120" min="0">80/120</meter>
</p>
<p>
应用 HTML 制作网页情况:<meter value="40" max="120" min="0" low="50" high="90" optimum="
"40">40/120</meter>
</p>
```

#### 2. 文本层次元素

(1) time: 用于定义日期或时间。该元素能够以机器可读的方式对日期和时间进行编码。例如, 用户代理能够把生日提醒或排定的事件添加到用户日程表中, 搜索引擎也能够生成更智能的搜索结果。

该元素可以代表 24 小时中的某一时刻, 在表示时刻时, 允许有时间差。在设置时间或日期时, 只需将该元素的属性“datetime”设为相应的时间或日期即可。

```

<p id="p1">
  <time datetime="2018-9-9">今天是 2018 年 9 月 9 日</time>
</p>
<p id="p2">
  <time datetime="2018-9-9T21:00">现在时间是 2018 年 9 月 9 日晚上 9 点</time>
</p>
<p id="p3">
  <time datetime="2020-12-31">新款 PSP2 掌上游戏机将于 2020 年底上市</time>
</p>
<p id="p4">
  <time datetime="2018-5-1" pubdate="true">本消息发布于 2018 年 9 月 1 日</time>
</p>

```

<p>元素 ID 号为“p1”中的<time>元素表示的是日期。页面在解析时,获取的是属性“datetime”中的值,而标记之间的内容只是用于显示在页面中。

<p>元素 ID 号为“p2”中的<time>元素表示的是日期和时间,它们之间使用字母“T”进行分隔。如果在整个日期与时间的后面加上一个字母 Z,则表示获取的是 UTC(世界统一时间)格式。

<p>元素 ID 号为“p3”中的<time>元素表示的是将来时间。

<p>元素 ID 号为“p4”中的<time>元素表示的是发布日期。

为了在文档中将这两个日期进行区分,在最后一个<time>元素中增加了“pubdate”属性,表示此日期为发布日期。

说明:<time>元素中的可选属性“pubdate”表示时间是否为发布日期,它是一个布尔值,该属性不仅可以用于<time>元素,还可以用于<article>元素。

(2)mark:主要功能是在文本中高亮显示某个或某几个字符,旨在引起用户的特别注意。其使用方法与<em>和<strong>有相似之处,但相比而言,HTML5 中新增的<mark>元素在突出显示时更加随意与灵活。

```

<h5>优秀开发人员的<mark>素质</mark></h5>
<p class="p3_5">
  一个优秀的 Web 页面开发人员,必须具有
  <mark>过硬</mark>的技术与
  <mark>务实</mark>的专业精神
</p>

```

(3)cite:可以创建一个引用标记,用于文档中参考文献的引用说明,如书名或文章名称。一旦在文档中使用了该标记,被标记的文档内容将以斜体的样式展示在页面中,以区别于段落中的其他字符。

```

<p>
  --引自<<<<cite>HTML5+CSS3 项目开发实战</cite>>>>--
</p>

```

### 3. 全局属性

(1) draggable: 规定元素是否可拖动。

**提示:** 链接和图像默认是可拖动的。

```
<p draggable="true">这是一个可拖动的段落。</p>
```

(2) hidden: 规定该元素是无关的。被隐藏的元素不会显示。

```
<ul hidden="true">
<!--这里用来创建列表,hidden="true"表示不可见;若为 false 表示可见-->
<li>列表 1</li>
<!--列标签-->
<li>列表 2</li>
<li>列表 3</li>
</ul>
```

(3) spellcheck: 规定是否必须对元素进行拼写或语法检查。

```
<input type="text" spellcheck="true|false"><!--spellcheck 开启注释-->
```

(4) contenteditable: 规定是否允许用户编辑内容。

```
<h2>可编辑列表</h2>
<ul contenteditable="true">
<!--这里用来创建列表,contenteditable="true"表示可编辑;若为 false 表示不可编辑-->
<li>列表 1</li><!--列标签-->
<li>列表 2</li>
<li>列表 3</li>
</ul>
```

#### 3.2.4 任务实施

任务描述中需完成的网页是一个学习情况调查表网页,其源代码详见 Task3-3.html。单击页面中的“html5 的优点”,显示内容如图 3-9 所示。

contenteditable="true"的效果如图 3-10 所示。



图 3-9 details 和 summary 使用效果

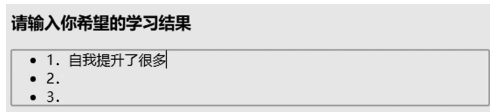


图 3-10 contenteditable 使用效果

#### 3.2.5 任务实训

综合练习前面所学的标记,制作图 3-11 所示的网页。



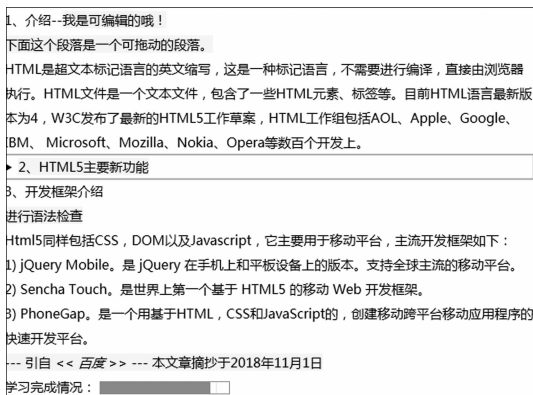


图 3-11 百度介绍网页效果图

其源代码见 Task3-4. html。

## 3.3 用户注册页面设计

### 3.3.1 任务目标

掌握表单元素和属性的使用。

### 3.3.2 任务描述

利用表单元素制作一个自适应的用户注册页面，如图 3-12 所示。

用户注册	
昵称:	<input type="text"/> *
邮箱:	<input type="text" value="123456@qq.com"/> *
密码:	<input type="password"/> *
性别:	<input checked="" type="radio"/> 男 <input type="radio"/> 女
年龄:	<input type="text"/> *
爱好:	<input type="checkbox"/> 足球 <input type="checkbox"/> 篮球 <input type="checkbox"/> 游泳 <input type="checkbox"/> 唱歌 <input type="checkbox"/> 跑步
颜色:	<input type="color"/>
自我介绍:	<input type="text" value="Please enter your message"/> *
<input type="button" value="立即注册"/>	

图 3-12 用户注册页面效果

当将鼠标指针放在文本框(文本框获取焦点)时,会出现图 3-13 所示的效果。若输入正确,则效果如图 3-14 所示。

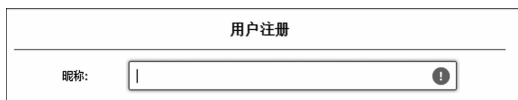


图 3-13 文本框获取焦点效果

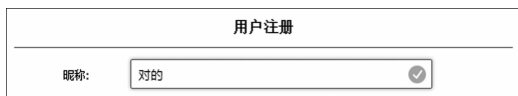


图 3-14 正确输入效果

在 Chrome 浏览器中预览,得到图 3-12 所示的显示效果。同时,分别用 Chrome 模拟器 Galaxy S5 和 iPad,得到了自适应的网页显示效果,如图 3-15 和图 3-16 所示。



图 3-15 网页在 Galaxy S5 模拟器中显示效果



图 3-16 网页在 iPad 模拟器中显示效果

### 3.3.3 任务知识

#### 1. 表单属性

(1)autocomplete;autocomplete 属性规定 form 或 input 域应该拥有自动完成功能。autocomplete 适用于< form> 标签,以及以下类型的< input> 标签: text、search、url、telephone、email、password、datepicker、range 及 color。

当用户在自动完成域中开始输入时,浏览器应该在该域中显示填写的选项。

```
<form action="" method="get" autocomplete="on">
Firstname:<input type="text" name="fname"/><br/>
Lastname:<input type="text" name="lname"/><br/>
```

```
E-mail:<input type="email" name="email" autocomplete="off"/><br/>
<input type="submit"/>
</form>
```

(2) novalidate: 指定在提交表单时取消对表单进行有效的检查。为表单设置该属性时, 可以关闭整个表单的验证, 这样可以使 form 内的所有表单控件不被验证。novalidate 适用于 <form> 标签, 以及以下类型的 <input> 标签: text、search、url、telephone、email、password、datepicker、range 及 color。

```
<form action="1.html" method="get" novalidate="true">
E-mail:<input type="email" name="user_email"/>
<input type="submit"/>
</form>
```

(3) input 元素的 type 属性。

① 单行文本输入框 <input type="text"/>。单行文本输入框常用来输入简短的信息, 如用户名、账号、证件号码等, 常用的属性有 name、value 和 maxlength。

② 密码输入框 <input type="password"/>。密码输入框用来输入密码, 其内容将以圆点的形式显示。

③ 单选按钮 <input type="radio"/>。单选按钮用于单项选择, 在定义单选按钮时, 必须为同一组中的选项指定相同的 name 值, 这样单选才会生效。

④ 复选框 <input type="checkbox"/>。复选框常用于多项选择, 如选择兴趣、爱好等, 可对其应用 checked 属性, 指定默认选中项。

⑤ 普通按钮 <input type="button"/>。普通按钮常常配合 JavaScript 脚本语言使用, 初学者了解即可。

⑥ 提交按钮 <input type="submit"/>。提交按钮是表单中的核心控件, 用户完成信息的输入后, 一般都需要单击提交按钮才能完成表单数据的提交。可以对其应用 value 属性, 改变提交按钮上的默认文本。

⑦ 重置按钮 <input type="reset"/>。当用户输入的信息有误时, 可单击重置按钮取消已输入的所有表单信息。可以对其应用 value 属性, 改变重置按钮上的默认文本。

⑧ 图像形式的提交按钮 <input type="image"/>。图像形式的提交按钮用图像代替了默认的按钮, 外观上更加美观。需要注意的是, 必须为其定义 src 属性指定图像的 url 地址。

⑨ 隐藏域 <input type="hidden"/>。隐藏域对用户是不可见的, 通常用于后台程序, 初学者了解即可。

⑩ 文件域 <input type="file"/>。当定义文件域时, 页面中将出现一个文本框和一个浏览按钮, 用户可以通过填写文件路径或直接选择文件的方式将文件提交给后台服务器。

⑪ email 类型 <input type="email"/>。email 类型的 input 元素是一种专门用于输入 E-mail 地址的文本输入框, 用来验证 email 输入框的内容是否符合 E-mail 邮件地址格式; 如果不符合, 就提示相应的错误信息。

⑫ url 类型 <input type="url"/>。url 类型的 input 元素是一种用于输入 URL 地址

的文本框。如果所输入的内容是 URL 地址格式的文本,就会提交数据到服务器;如果输入的值不符合 URL 地址格式,就不允许提交并有提示信息。

⑬tel 类型<input type="telephone"/>。telephone 类型用于提供输入电话号码的文本框,由于电话号码的格式千差万别,很难实现一个通用的格式。因此,telephone 类型通常会和 pattern 属性配合使用。

⑭search 类型<input type="search"/>。search 类型是一种专门用于输入搜索关键词的文本框,它能自动记录一些字符,如站点搜索或 Google 搜索等。在用户输入内容后,其右侧会附带一个删除图标,单击这个图标可以快速清除内容。

⑮color 类型<input type="color"/>。color 类型用于提供设置颜色的文本框,实现一个 RGB 颜色输入。其基本形式是 #RRGGBB,默认值为 #000000,通过 value 属性值可以更改默认颜色。单击 color 类型文本框,可以快速打开拾色器面板,方便用户选取一种颜色。

⑯number 类型<input type="number"/>。number 类型的 input 元素用于提供输入数值的文本框。在提交表单时,会自动检查该输入框中的内容是否为数字。如果输入的内容不是数字或数字不在限定范围内,就会出现错误提示。number 类型的输入框可以对输入的数字进行限制,规定允许的最大值和最小值、合法的数字间隔或默认值等。具体属性说明如下。

- a. value:指定输入框的默认值。
- b. max:指定输入框可以接受的最大输入值。
- c. min:指定输入框可以接受的最小输入值。
- d. step:输入域合法的间隔,若不设置,则默认值是 1。

⑰range 类型<input type="range"/>。range 类型的 input 元素用于提供一定范围内数值的输入范围,在网页中显示为滑动条。它的常用属性与 number 类型一样,通过 min 属性和 max 属性,可以设置最小值与最大值,通过 step 属性指定每次滑动的步幅。

⑱datepicker 类型<input type="date,month,week..."/>。datepicker 类型是指日期和时间类型,HTML5 中提供了多个可供选取日期和时间的输入类型,用于验证输入的日期和时间,具体如表 3-3 所示。

表 3-3 日期和时间类型

日期和时间类型	说 明
date	选取日、月、年
month	选取月、年
week	选取周和年
time	选取时间(小时和分钟)
datetime	选取时间、日、月、年(UTC 时间)
datetime-local	选取时间、日、月、年(本地时间)

## 2. input 元素的其他属性

(1) autofocus 属性。在 HTML5 中,autofocus 属性用于指定页面加载后是否自动获取焦点,将标记的属性值指定为 true 时,表示页面加载完毕后会自动获取该焦点。

(2) form 属性。在 HTML5 之前,如果用户要提交一个表单,必须把相关的控件元素都放在表单内部,即<form>和</form>标签之间。在提交表单时,会将页面中不是表单子

元素的控件直接忽略掉。

(3)list 属性。通过 datalist 元素实现数据列表的下拉效果。而 list 属性用于指定输入框所绑定的 datalist 元素,其值是某个 datalist 元素的 id。

(4)multiple 属性。multiple 属性指定输入框可以选择多个值,该属性适用于 email 和 file 类型的 input 元素。multiple 属性用于 email 类型的 input 元素时,表示可以向文本框中输入多个 E-mail 地址,多个地址之间通过逗号隔开;multiple 属性用于 file 类型的 input 元素时,表示可以选择多个文件。

(5)min、max 和 step 属性。HTML5 中的 min、max 和 step 属性用于为包含数字或日期的 input 输入类型规定限值,也就是给这些类型的输入框加一个数值的约束,适用于 number 和 range 标签。具体属性说明如下。

- a. max:规定输入框所允许的最大输入值。
- b. min:规定输入框所允许的最小输入值。
- c. step:为输入框规定合法的数字间隔,若不设置,则默认值是 1。

(6)pattern 属性。pattern 属性用于验证 input 类型输入框中用户输入的内容是否与所定义的正则表达式相匹配。pattern 属性适用于的类型是: text、search、url、telephone、email 和 password 的<input/>标记。正则表达式详述请参见其他教材。

(7)placeholder 属性。placeholder 属性用于为 input 类型的输入框提供相关提示信息,以描述输入框期待用户输入何种内容。在输入框为空时显式出现,而当输入框获得焦点时则会消失。

(8)required 属性。HTML5 中的输入类型不会自动判断用户是否在输入框中输入了内容,如果开发者要求输入框中的内容是必须填写的,那么需要为 input 元素指定 required 属性。required 属性用于规定输入框填写的内容不能为空,否则不允许用户提交表单。

(9)datalist 元素。datalist 元素用于定义输入框的选项列表,列表通过 datalist 内的 option 元素进行创建。如果用户不希望从列表中选择某项,也可以自行输入其他内容。datalist 元素通常与 input 元素配合使用来定义 input 的取值。在使用<datalist>标签时,需要通过 id 属性为其指定一个唯一的标识,然后为 input 元素指定 list 属性,将该属性值设置为 option 元素对应的 id 属性值即可。

```
<datalist id="browsers">
  <option value="InternetExplorer">
  <option value="Firefox">
  <option value="Chrome">
  <option value="Opera">
  <option value="Safari">
</datalist>
<input list="browsers">
```

(10)keygen 元素。keygen 元素是用于表单的密钥生成器,能够使用户验证更为安全、可靠。当提交表单时会生成两个键:一个是私钥,它存储在客户端;一个是公钥。如果新的浏览器能够对 keygen 元素的支持度再增强一些,则有望使其成为一种有用的安全标准。keygen 元素的常用属性及说明如表 3-4 所示。

表 3-4 keygen 元素常用属性及说明

属 性	说 明
autofocus	使 keygen 字段在页面加载时获得焦点
challenge	若使用,则将 keygen 的值设置为在提交时询问
disabled	禁用 keytag 字段
form	定义该 keygen 字段所属的一个或多个表单
keytype	定义 keytypersa 生成 RSA 密钥
name	定义 keygen 元素的唯一名称 name 属性,用于在提交表单时搜集字段的值

```
<form action="1.html" method="get">
  用户名:<input type="text" name="usr_name">
  加密:<keygen name="security">
  <input type="submit">
</form>
```

(11)output 元素。output 元素用于不同类型的输出,可以在浏览器中显示计算结果或脚本输出。其常用属性及说明如表 3-5 所示。

表 3-5 output 元素常用属性及说明

属 性	说 明
for	定义输出域相关的一个或多个元素
form	定义输入字段所属的一个或多个表单
name	定义对象的唯一名称

### 3.3.4 任务实施

任务描述中需完成的网页是一个用户注册页面,其源代码详见 Task3-5.html。

### 3.3.5 任务实训

利用表单元素,设计图 3-17 所示的表单。

图 3-17 表单制作

其完整源代码详见 Task3-6.html。

## 3.4 视频播放网页设计

### 3.4.1 任务目标

- 掌握 HTML5 的 video 元素的属性方法事件。
- 掌握 HTML5 的 audio 元素的属性方法事件。

### 3.4.2 任务描述

利用多媒体技术制作图 3-18 所示的视频播放网页,且能在 Chrome 模拟器 Galaxy S5 和 iPad 中自适应显示。



图 3-18 视频播放网页效果

### 3.4.3 任务知识

在 HTML5 出现之前并没有将视频和音频嵌入页面的标准方式,多媒体内容在大多数情况下都是通过第三方插件或集成在 Web 浏览器的应用程序中来置于页面中的。通过这样的方式实现的视频和音频功能,不仅需要借助第三方插件且实现代码复杂冗长,运用 HTML5 新增的 video 标签和 audio 标签可以避免这样的问题。

(1)video 元素的常用属性、方法和事件。在 HTML5 中,video 元素用于定义播放视频文件的标准,它支持 3 种视频格式,分别为 OGG、WebM 和 MPEG4,其基本语法格式如下。

```
<video src="视频文件路径" controls="controls"></video>
```

在上面的语法格式中,src 属性用于设置视频文件的路径,controls 属性用于为视频提供播放控件,这两个属性是 video 元素的基本属性。可以对它增加一些属性来优化视频的播放效果。video 元素的常用属性如表 3-6 所示。

表 3-6 video 元素的常用属性

属 性	值	描 述
autoplay	autoplay	当页面载入完成后自动播放视频
loop	loop	视频结束时重新开始播放
preload	preload	若出现该属性,则视频在页面加载时进行加载并预备播放。若使用“autoplay”,则忽略该属性
poster	url	当视频缓冲不足时,该属性值链接一个图像,并将该图像按照一定的比例显示出来
width		视频宽度
height		视频高度
muted		设置或返回视频是否静音

video 元素的常用方法如表 3-7 所示。

表 3-7 video 元素的常用方法

方 法	描 述
canPlayType(type)	能否播放某个资源文件,返回三个字符串之一:empty、maybe 或 probably
load()	重新加载资源
play()	播放
pause()	暂停

video 元素的常用事件如表 3-8 所示。

表 3-8 video 元素的常用事件

事 件	描 述
loadstart	当浏览器开始查找视频时
durationchange	当视频的时长已更改时
loadedmetadata	当浏览器已加载视频的元数据时
loadeddata	当浏览器已加载视频的当前帧时
progress	当浏览器正在下载视频时
canplay	当浏览器可以播放视频时
canplaythrough	当浏览器可在不因缓冲而停顿的情况下进行播放时

(2)audio 元素的常用属性、方法和事件。在 HTML5 中,audio 元素用于定义播放音频文件的标准,它支持 3 种音频格式,分别为 OGG、MP3 和 WAV,其基本格式如下。

```
<audio src="音频文件路径" controls="controls"></audio>
```

在上面的基本格式中,src 属性用于设置音频文件的路径,controls 属性用于为音频提供播放控件,这和 video 元素的属性非常相似。同样,<audio>和</audio>之间也可以插入



文字,用于不支持 audio 元素的浏览器显示。audio 元素的属性、方法、事件和 video 一样,在此不再赘述。

(3)source 元素。在 HTML5 中,运用 source 元素可以为 video 元素或 audio 元素提供多个备用文件。运用 source 元素添加音频的基本格式如下。

```
<audio controls="controls">
<source src="音频文件路径" type="媒体文件类型">
<source src="音频文件路径" type="媒体文件类型">
.....
</audio>
```

source 元素一般设置两个属性,即 src 用于指定媒体文件的路径,type 用于指定媒体文件的类型。

### 3.4.4 任务实施

任务描述中需完成的网页是一个视频播放网页,在 Dreamweaver CS6 中,创建 Task3-7.html,其用到的 CSS 文件为 css/Task3-3.css,js 文件为 js/Task3-1.js。

### 3.4.5 任务实训

制作一个音乐播放器,如图 3-19 所示。

音乐播放器实现的功能是可以播放、暂停,以及显示音乐的当前播放时间及总时间。例如,当单击“播放”按钮时,效果如图 3-20 所示。

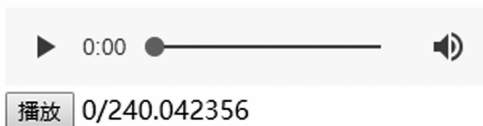


图 3-19 音乐播放器

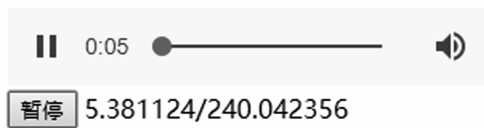


图 3-20 播放时效果

音乐播放器实现的完整源代码详见 Task3-8.html。

## 3.5 个人信息管理网页设计

### 3.5.1 任务目标

- 利用媒体查询实现响应式设计。
- 利用弹性盒布局实现响应式设计。

### 3.5.2 任务描述

利用媒体查询和弹性盒布局实现个人信息管理网页响应式设计,大屏幕下的网页如

图 3-21 所示。

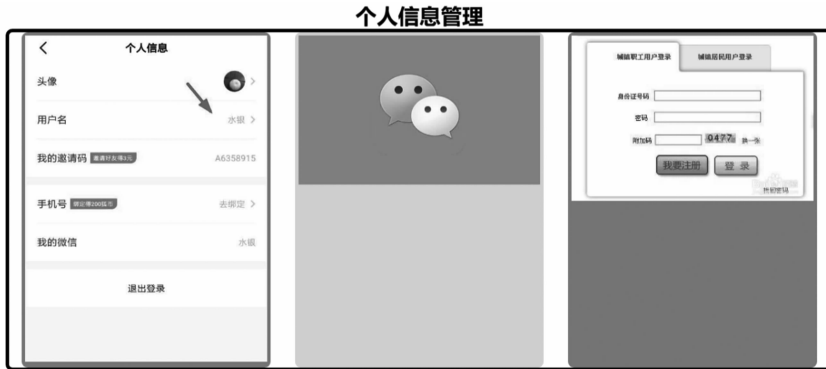


图 3-21 个人信息管理网页效果

在 Chrome 浏览器中,分别用模拟器 Galaxy S5 和 iPad 得到自适应的网页显示效果,如图 3-22 和图 3-23 所示。



图 3-22 网页在 Galaxy S5 模拟器中显示效果



图 3-23 网页在 iPad 模拟器中显示效果

### 3.5.3 任务知识

#### 1. 媒体查询

##### 1) 媒体查询的概念

使用媒体查询,可以针对不同的媒体类型定义不同的样式。媒体查询可以针对不同的屏幕尺寸设置不同的样式,如果需要设计响应式的页面,那么媒体查询是非常有用的。在重置浏览器大小的过程中,页面也会根据浏览器的宽度和高度重新进行渲染。

##### 2) 媒体查询的作用

媒体查询可以在不改变页面内容的情况下为特定的一些输出设备定制显示效果,是响应式布局实现的主要方式。

##### 3) 媒体查询的语法

###### (1) 外联 CSS 语法。

```
<link rel="stylesheet" media="mediatype and|not|only (mediafeature)" href="mystylesheet.css">
```

① 媒体类型:mediatype。

② 关键字:and|not|only。

③ 媒体特征:mediafeature。

实例:

```
<link rel="stylesheet" media="screen and (min-width:1024px)" href="red.css"/>
```

###### (2) 内嵌样式的语法。

```
@media mediatype and|not|only (mediafeature){...}
```

① 媒体类型:mediatype。

② 关键字:and|not|only。

③ 媒体特征:mediafeature。

例如,当页面宽度小于等于 960 px 时,执行下面的 CSS 代码。

```
@media screen and (max-width:960px){
body{background:red;}
```

当页面宽度大于 960 px 小于 1 200 px 时,执行下面的 CSS 代码。

```
@media screen and (min-width:960px) and (max-width:1200px){
body{background:yellow;}
```

#### 2. 弹性盒布局

弹性盒布局是 CSS3 规范中提出的一种新的布局方式。它的目的是提供一种更加高效的方式来对容器中的条目进行布局、对齐和空间分配;这种布局模式已被主流浏览器所支持,可以在 Web 应用开发中使用。这里需要说明的是,flex 是 display 的一个属性值;设置了“display:flex”属性的元素称为 Flex 容器,它里面的所有子元素统称为容器成员,称为 Flex 项目。后面我们就使用 Flex 容器和 Flex 项目来进行介绍。Flex 容器有两根坐标轴:主轴(mainaxis)和交叉轴(crossaxis),如图 3-24 所示。

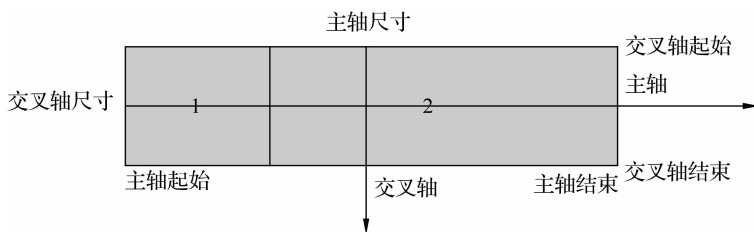


图 3-24 弹性盒布局中的轴

### 1) 容器的属性

上面提到过,采用 Flex 布局元素,称为 Flex 容器,简称容器,那么容器属性就是设置在已经设置了 flex 属性的这个容器上的属性。

**注意:** 需要设置“display: flex;”属性,可以选择性设置 flex-direction; flex-wrap; flex-flow; justify-content; align-items; align-content 属性(以上这些属性都是设置在容器上面的)。其含义如表 3-9 所示。

表 3-9 容器属性 1

属 性	解 释
flex-direction	给容器设置主轴的方向
flex-wrap	设置容器是单行还是多行
flex-flow	是 flex-direction 和 flex-wrap 的复合简写形式
justify-content	定义容器中的项目在主轴方向上的对齐方式
align-items	定义弹性盒子元素在交叉轴上的对齐方式
align-content	定义多根轴线的对齐方式,若项目只有一根轴线,则不起作用

下面具体介绍容器属性相关的设置。

(1) display: flex 元素的弹性容器属性。以前学过的 display 相关属性值有 block、inline-block、inline 和 none。那现在再加一个弹性盒子属性值,即 flex 属性值。英文中的 flex 是弯曲的意思,所以在前端布局中,可以理解为布局相对比较随意。

display: flex 默认的排列方向为主轴为 x 轴,交叉轴默认为 y 轴,并且其方向是从右往左和从上往下的。

```
<div class="main">
  <div class="box1">
    <ul class="myflex">
      <li>A</li><li>B</li><li>C</li><li>D</li>
    </ul>
  </div>
</div>
```

设置如下 CSS 后,效果如图 3-25 所示。

```

<style type="text/css">
* {
margin:0;
padding:0;
list-style:none;
color:#FFF;
}
.main{
width:800px;
margin:100pxauto;
box-shadow:0 0 15px 0 deeppink;
}
.main.box1.myflex{
display:flex; /* 给容器一个 display:flex 属性 */
}
.main.box1ulli{
width:150px;
height:40px;
font:bold 20px/40px "";
text-align:center;
background:#F00;
margin-left:70px;
box-shadow:0 0 10px 0 blue;
}
.main.box1ulli:nth-child(1){
margin-left:0px;
}
</style>

```



图 3-25 display:flex 效果

(2) flex-direction 给容器设置主轴的方向。以前学过的 transform-origin、background-origin、background-clip 等,可以修改参考点的属性,那么 flex-direction 属性也是如此,其就是对默认的主轴或交叉轴的方向进行修改,主要有表 3-10 所示的几个属性值。

表 3-10 容器属性 2

属 性	解 释
row	主轴为水平方向,并且从左到右排列(默认)
row-reverse	主轴为水平方向,但是与默认相反,从右往左排列
column	主轴切换为垂直方向,起点在上沿,从上往下进行显示
column-reverse	主轴在垂直方向上,起点在下沿,从下往上进行显示

在如下 body 区域中设置样式表内容后,其在 iPad 中的效果如图 3-26 所示。

```
<div class="main">
<div class="box1">
<h3>row</h3>
<ul class="myflex">
<li>1</li>
<li>2</li>
<li>3</li>
<li>4</li>
</ul>
</div>
<div class="box2">
<h3>row-reverse</h3>
<ul class="myflex">
<li>1</li>
.....
</ul>
</div>
<div class="box3">
<h3>column</h3>
<ul class="myflex">
<li>1</li>
.....
</ul>
</div>
<div class="box4">
<h3>column-reverse</h3>
<ul class="myflex">
<li>1</li>
.....
</ul>
</div>
</div>

<style type="text/css">
* {
margin:0;
padding:0;
list-style:none;
color:#FFF;
}
.main{
```

```
width:800px;
margin:100px auto;
box-shadow:0 0 15px 0 deeppink;
}
.maindiv{
width:800px;
height:200px;
box-shadow:0 0 15px 0 green yellow;
}
h3{
text-align:center;
color:#000;
}
.myflex{
display:flex;
}/* 给容器一个 display:flex 属性 */
.mainulli{
width:150px;
height:40px;font:bold 20px/40px "";
text-align:center;
background:red;
box-shadow:0 0 10px 0 blue;
}
.main.box1ul{
flex-direction:row;
}/* 主轴为水平方向,并从左到右排列(默认),容器属性需要填在容器元素上 */
.main.box2ul{
flex-direction:row-reverse;
}/* 主轴为水平方向,并从右往左排列 */
.main.box3ul{
flex-direction:column;
}/* 主轴为垂直方向,并从上往下排列 */
.main.box4ul{
flex-direction:column-reverse;
}/* 主轴为垂直方向,并从下往上进行显示 */
</style>
```

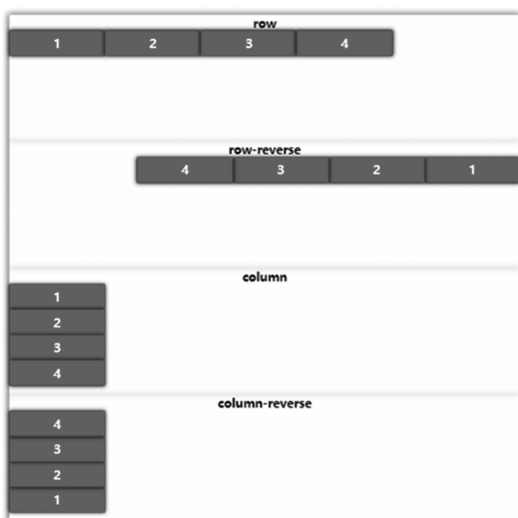


图 3-26 flex-direction 效果

(3) flex-wrap 给 flex 容器设置是单行还是多行显示。wrap 属性最早是在学习文字单行显示还是多行显示时出现的,里面有个属性值为 wrap 和 nowrap。这里同样也有这样的属性,不过多了一个 wrap-reverse 属性,如表 3-11 所示。

表 3-11 flex-wrap 取值

属 性	解 释
wrap	flex 容器为多行,该情况下 flex 子项在溢出的部分会被放置到新行中,换行
nowrap	不换行,该情况下 flex 子项可能会溢出容器(默认情况)
wrap-reverse	反转 wrap 排列

在如下 body 区域中设置样式表内容后,其在 iPad 中的效果如图 3-27 所示。

```
<div class="main">
<div class="box1">
<h3>nowrap</h3>
<ul class="myflex">
<li>1</li>
<li>2</li>
<li>3</li>
<li>4</li>
<li>5</li>
<li>6</li>
<li>7</li>
</ul>
</div>
<div class="box2">
```



```

<h3>wrap</h3>
<ul class="myflex">
<li>l</li>
.....
</ul>
</div>
<div class="box3">
<h3>wrap-reverse</h3>
<ul class="myflex">
<li>l</li>
.....
</ul>
</div>
</div>

```

```

<style type="text/css">
* {
margin:0;
padding:0;
list-style:none;
color:#FFF;
}
.main{
width:800px;
margin:100px;
auto;
box-shadow:0 0 15px 0 deeppink;
}
.maindiv{
width:800px;
height:200px;
box-shadow:0 0 15px 0 green yellow;
}
h3{
text-align:center;
color:#000;
}
.myflex{
display:flex;
}/* 给容器一个 display:flex 属性 */
.main.box1ul{
flex-direction:row;

```

```

}/* 主轴为水平方向,并从左到右排列(默认),容器属性需要填在容器元素上 */
.mainulli{
width:150px;
height:40px;
font:bold 20px/40px "";
text-align:center;
background:red;
box-shadow:0 0 10px 0 blue;
}
.main.box1ul{
flex-wrap:nowrap;
}/* 默认值,不换行,会压缩,不会超出父级 */
.main.box2ul{
flex-wrap:wrap;
}/* 会换行 */
.main.box3ul{
flex-wrap:wrap-reverse;
}/* 只是单纯地上下反转,不是序号改变 */
</style>

```

flex-flow 是 flex-direction 和 flex-wrap 的复合属性形式,默认值是 rownowrap,即 x 为主轴方向,不换行,也就是两个单独属性的默认值。

若将上述 HTML 代码中各部分显示的标题修改后,样式表内容改为如下 CSS 代码,则显示效果如图 3-28 所示。

```

<style type="text/css">
* {
margin:0;
padding:0;
list-style:none;
color: #FFF;
}
.main{
width:90%;
height:auto;
margin:0 auto;
box-shadow:0 0 15px 0 deeppink;
}
.maindiv{
width:100%;
height:25%;
box-shadow:0 0 15px 0 greenyellow;
}
h3{

```

```

text-align:center;
color:#000;
}
.myflex{
display:flex;
}/* 给容器一个 display:flex 属性 */
.mainulli{
width:150px;
height:40px;
font:bolder 20px/40px "";
text-align:center;
background:red;
box-shadow:0 0 10px 0 blue;
}
.main.box1ul{
flex-flow:rowwrap;
}/* 主轴为水平方向,并从左到右排列(默认),容器属性需要填在容器元素上 */
.main.box2ul{
flex-flow:row-reverse nowrap;
}
.main.box3ul{
flex-flow:columnwrap;
}
.main.box4ul{
flex-flow:column-reverse wrap-reverse;
}
</style>

```

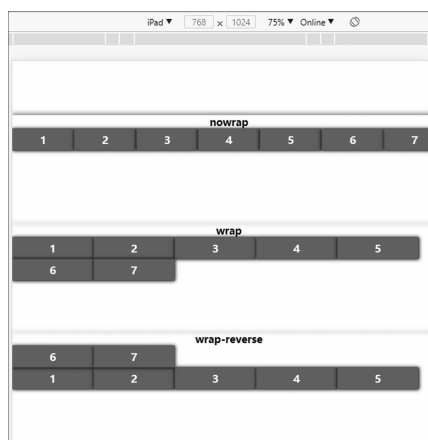


图 3-27 flex-wrap 效果

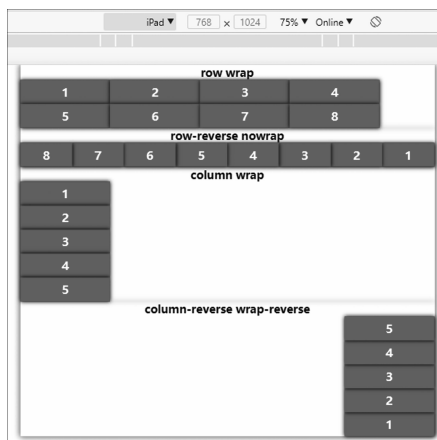


图 3-28 wrap-reverse 效果

比较图 3-27 和图 3-28 可知,虽然设置了 wrap,但是在列方向上依旧是不会改变的,但是在设置 wrap-reverse 之后,却是可以进行反转的,所以 wrap 其实主要是针对行的属性,在

列上时没有任何效果。

(4) justify-content(给容器)定义了项目在主轴上的对齐方式。justify-content 属性的主要作用就是定义项目在主轴方向上的对齐方式,可以是左对齐、右对齐、居中对齐、两端对齐、间隔相等的对齐方式,如表 3-12 所示。

表 3-12 justify-content 属性

属性值	属性含义
flex-start	沿着主轴开始的方向进行对齐
flex-end	从主轴结束的方向开始对齐
center	居中对齐
space-between	两端对齐
space-around	每个项目两侧间隔是相等的

在上述 HTML 代码的基础上,修改样式表内容为如下所示后,其在 iPad 上的水平显示效果如图 3-29 所示,垂直显示效果如图 3-30 所示。

```

* {
margin:0;
padding:0;
list-style:none;
color: #FFF;
}
.main{
width:90%;
height:auto;
margin:0 auto;
box-shadow:0 0 15px 0 deeppink;
}
.maindiv{
width:100%;
height:25%;
box-shadow:0 0 15px 0 green yellow;
}
h3{
text-align:center;
color: #000;
}
.myflex{
display:flex;
}/* 给容器一个 display:flex 属性 */
.mainulli{
width:150px;
height:40px;
font:bolder 20px/40px "";

```

```
text-align:center;
background:red;
box-shadow:0 0 10px 0 blue;
}
.main.box1ul{
flex-direction:row;
justify-content:flex-start;
}/* 沿着主轴的方向,从 start 到 end */
.main.box2ul{
flex-direction:row;
justify-content:flex-end;
}/* 沿着主轴的方向,从 end 到 start */
.main.box3ul{
flex-direction:row-reverse;
justify-content:center;
}/* 沿着主轴的方向,从中心往两端 */
.main.box4ul{
flex-direction:row;
justify-content:space-around;
}/* 沿着主轴的方向,间隔相等 */
.main.box5ul{
flex-direction:row;
justify-content:space-between;
}/* 沿着主轴的方向,从两端往中心 */
.main.box6ul{
flex-direction:column;
justify-content:flex-start;
}/* 沿着主轴的方向,从 start 到 end */
.main.box7ul{
flex-direction:column;
justify-content:flex-end;
}/* 沿着主轴的方向,从 end 到 start */
.main.box8ul{
flex-direction:column-reverse;
justify-content:center;
}/* 沿着主轴的方向,从中心往两端 */
.main.box9ul{
flex-direction:column;
justify-content:space-around;
}/* 沿着主轴的方向,间隔相等 */
.main.box10ul{
flex-direction:column;
justify-content:space-between;
}/* 沿着主轴的方向,从两端往中心 */
```

主轴在垂直方向上时,justify 是没有任何效果的,所以 justify-content 对齐属性只能在水平方向上使用,不能在垂直方向上使用。

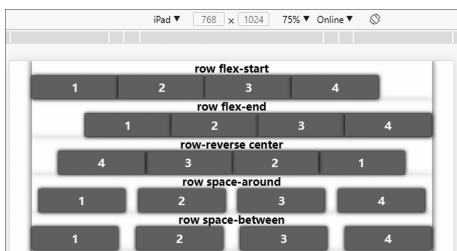


图 3-29 justify-content 水平方向效果

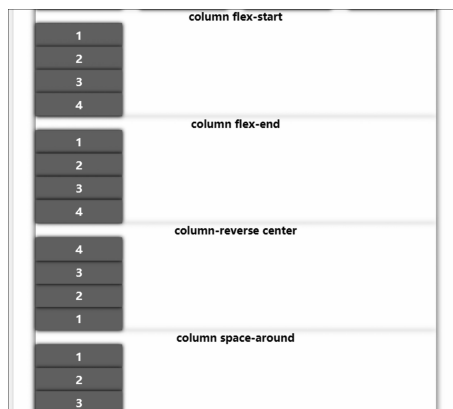


图 3-30 justify-content 垂直方向效果

(5)align-items 弹性盒子元素在交叉轴上的对齐方式。与主轴垂直的轴称为交叉轴。所以如果定义了主轴,交叉轴也就自然出来了。align-items 定义项目在弹性盒子交叉轴上的对齐方式,那么在交叉轴上的对齐属性使用的就是 align-items 这个属性,如表 3-13 所示。

表 3-13 align-items 属性

属性值	解释
stretch	默认值,即交叉轴的上沿
flex-end	与交叉轴的终点
center	与交叉轴的中点对齐
baseline	项目的第一行文字的基线
flex-start	交叉轴的起点对齐

若样式表内容修改为如下所示,则在 iPad 水平方向上的显示效果如图 3-31 所示。

```
<style type="text/css">
* {
margin:0;
padding:0;
list-style:none;
color: #FFF;
}
.main{
width:800px;
margin:100px auto;
box-shadow:0 0 15px 0 deeppink;
}
.maindiv{
width:800px;
```

```
height:200px;
box-shadow:0 0 15px 0 green yellow;
}
h3{
text-align:center;
color:#000;
}
.myflex{
display:flex;
}/* 给容器一个 display:flex 属性 */
.main.box1ul{
flex-direction:row;
}/* 主轴为水平方向,并从左到右排列(默认),容器属性需要填在容器元素上 */
.mainulli{
width:150px;
height:40px;
font:bold 20px/40px "";
text-align:center;
background:red;
box-shadow:0 0 10px 0 blue;
}
.mainulli:nth-child(2){
height:130px;
}
.main.box1ul{
align-items:stretch;
}/* 默认的对齐方式 */
.main.box2ul{
align-items:flex-start;
}
.main.box3ul{
align-items:flex-end;
}
.main.box4ul{
align-items:baseline;
}
.main.box4.myflexli:nth-child(1){
font-size:16px;
}
.main.box4.myflexli:nth-child(2){
font-size:20px;
}
.main.box4.myflexli:nth-child(3){
```

```
font-size:34px;
}
.main.box5.myflex{
  -webkit-align-items:center;
}
</style>
</div>
```

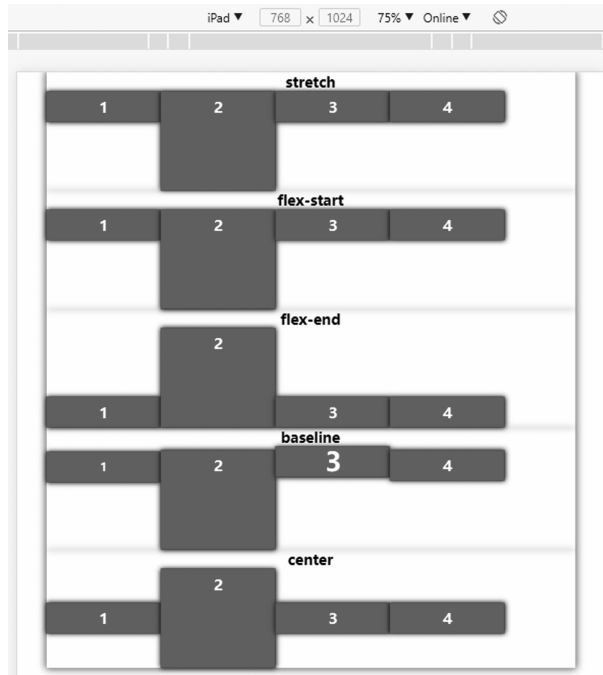


图 3-31 justify-content 水平方向效果

(6) align-content 定义了多根轴线的对齐方式,若项目只有一根轴线,则不起作用。align-content 属性如表 3-14 所示。

表 3-14 align-content 属性

属性值	属性含义
stretch	默认值,即交叉轴的上沿
flex-end	与交叉轴的终点对齐
center	与交叉轴的中点对齐
space-between	与交叉轴两端对齐,之间平均分配
space-around	交叉轴两侧间隔相等
flex-start	交叉轴的起点对齐

样式表内容如下。



```
<style type="text/css">
* {
margin:0;
padding:0;
list-style:none;
color:#FFF;
}
.main{
width:800px;
margin:100px auto;
box-shadow:0 0 15px 0 deeppink;
}
.maindiv{
width:800px;
height:200px;
box-shadow:0 0 15 px 0 green yellow;
}
h3{
text-align:center;
color:#000;
}
.myflex{
display:flex;
}/* 给容器一个 display:flex 属性 */
.main.box1ul{
flex-direction:row;
}/* 主轴为水平方向,并从左到右排列(默认),容器属性需要填在容器元素上 */
.main.box1ul{
flex-wrap:wrap;
height:200px;
align-content:stretch;
}/* 默认值,主轴线占满整个交叉轴(每行元素下沿都有一个主轴,且每行主轴的高度相等) */
.main.box2ul{
flex-wrap:wrap;
height:200px;
align-content:flex-start;
}/* 与交叉轴上沿紧密对齐 */
.main.box3ul{
flex-wrap:wrap;
height:200px;
align-content:flex-end;
}/* 与交叉轴下沿紧密对齐 */
.main.box4ul{
flex-wrap:wrap;
```

```

height:200px;
align-content:center;
}/* 与交叉轴中部紧密对齐 */
.main.box5ul{
flex-wrap:wrap;
height:200px;
align-content:space-around;
}/* 与交叉轴两端对齐,中间主轴宽度平均分配 */
.main.box6ul{
flex-wrap:wrap;
height:200px;
align-content:space-between;
}/* 与交叉轴两端对齐,中间主轴宽度平均分配 */</style>

```

在 iPad 上的显示效果如图 3-32 所示。

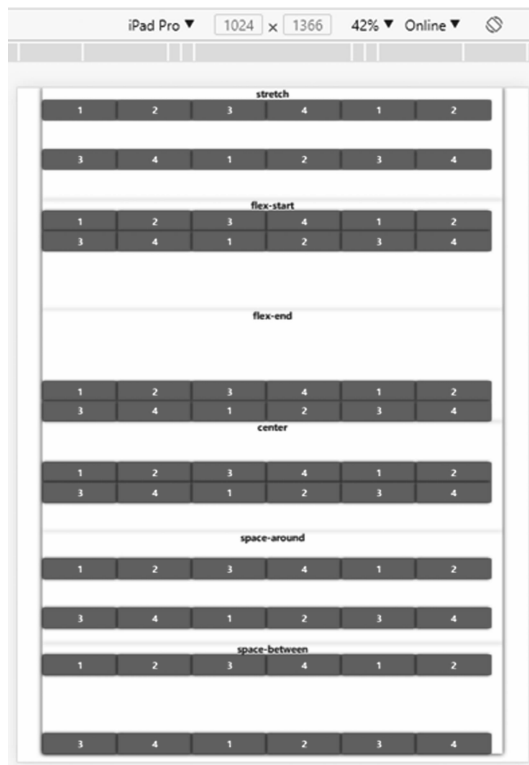


图 3-32 align-content 水平方向效果

## 2) 弹性子元素属性

### (1) 子元素排序 order。

```
order:<integer>
```

order 属性的默认值为 0, 可以为负值。指定 order 属性后, 浏览器将根据 order 属性的值对子元素进行分组, order 值相同的元素属于同一组, 值小的组排在前面, 组内则根据元素

在 HTML 代码中出现的先后顺序排列。

在如下 body 区域代码中

```
<h3>order</h3>
<ul class="myflex">
<li>1</li>
<li>2</li>
<li>3</li>
</ul>
```

添加如下 CSS 代码后,显示效果如图 3-33 所示。

```
* {
margin:0;
padding:0;
list-style:none;
color: #FFF;
}
.myflex{
display:flex;
}/* 给容器一个 display:flex 属性 */
ul li{
width:150px;
height:40px;
font:bolder 20px/40px "";
text-align:center;
background: #6F6;
border:1px solid #000;
}
ul li:nth-child(1){
order:2;/* 给项目排序 */
-webkit-order:2;
}
ul li:nth-child(2){
order:1;
-webkit-order:1;
}
ul li:nth-child(3){
order:-1;
-webkit-order:-1;
}
```



图 3-33 order 排序效果

可以看出,1、2、3 的出现顺序与它们在 HTML 中的定义顺序无关,而是按照 order 属性值的大小(数值小的排在前面)排列的。

(2)flex-grow 属性。flex-grow 属性用来定义子元素的拉伸因子,当容器有多余的空间时,浏览器把所有子元素的 flex-grow 属性值相加,再根据各自在总值中所占的份额分配容器的多余空间。语法为

```
flex-grow:<number>
```

flex-grow 属性的默认值为 0,支持整数或小数,不允许负值。如果某个子元素没有定义 flex-grow 属性,或将 flex-grow 属性显式设置为 0,或定义了固定尺寸,则不再为它分配父元素的多余空间。

body 内容为

```
<h3>flex-grow</h3>
<div class="father">
  <div class="box1">1</div>
  <div class="box2">2</div>
  <div class="box3">3</div>
</div>
```

CSS 代码为

```
<style>
* {
  margin:0;
  padding:0;
}
.father{
  width:600px;
  height:50px;
  display:-webkit-flex;
  display:flex;
  border:2pxsolid#000;
}
.father div{
  height:50px;
  width:100px;
  background:#0F9;
  border:1pxsolid#F00;
  text-align:center;
  font-weight:bold;
  font-size:20px;
}
.box1{
  -webkit-flex-grow:2;
```

```

    flex-grow:2;
  }
  .box2{
    -webkit-flex-grow:1;
    flex-grow:1;
  }
</style>

```

在上述代码中,父元素的宽度是 600 px,每个子元素的宽度为 100 px,于是,父元素的剩余宽度就变成  $600 - 3 \times 100 = 300$  px。box1 和 box2 定义了 flex-grow 属性,box3 未定义。故 box1 和 box2 把剩余空间分成  $2 + 1 = 3$  份,box1 占  $2/3$ ,box2 占  $1/3$ 。于是,box1、box2 按  $3 : 1$  来分配这 300 px,box3 不分配。box1 分得  $300 \times (2/3) = 200$  px,box2 分得  $300 \times (1/3) = 100$  px。最终的结果是,box1 和 box2 在预分配空间的基础上又分得了剩余空间,而 box3 仅有原本的 100 px。运行结果如图 3-34 所示。



图 3-34 flex-grow 拉伸效果

可以看出,box1 的实际宽度为  $100 + 200 = 300$  px,box2 的实际宽度为  $100 + 100 = 200$  px,box3 的实际宽度仍然是 100 px。

(3)flex-shrink 属性。flex-shrink 属性定义子元素的收缩因子,当容器空间不足时,浏览器把所有子元素的 flex-shrink 属性值相加,再根据各自在总值中所占的份额进行收缩。语法为

```
flex-shrink:<number>
```

flex-shrink 属性的默认值为 1,不允许负值,0 表示不收缩。例如,父元素 father 下有两个子元素 box1 和 box2。HTML 代码如下。

```

<h3>flex-shrink</h3>
<div class="father">
  <div class="box1">1</div>
  <div class="box2">2</div>
</div>

```

CSS 代码为

```

* {
  margin:0;
  padding:0;
}
.father{
  width:600px;
  display:-webkit-flex;

```

```

display: flex;
}
.fatherdiv{ height:50px;
background: #0F9;
border: 1pxsolid#F00;
text-align:center;
font-weight:bold;
font-size:20px;
}
.box1{
color:red;
width:400px;
flex-shrink:3;
-webkit-flex-shrink:3;
}
.box2{
width:400px;
}

```

在上述代码中,父元素的宽度是 600 px,每个子元素的宽度是 400 px,于是,父元素的宽度不足。为了能够容纳所有子元素,子元素需要收缩,待收缩的宽度为  $2 \times 400 - 600 = 200$  px。box1 定义了 flex-shrink 属性,box2 未定义,box2 的 flex-shrink 属性使用默认值 1。box1、box2 把待收缩空间分成  $3+1=4$  份,box1 占  $3/4$ ,box2 占  $1/4$ 。于是,box1 和 box2 按  $3:1$  来分配这 200 px,box1 要收缩  $200 \times (3/4) = 150$  px,box2 要收缩  $200 \times (1/4) = 50$  px。最终的结果是,box1 的实际宽度为  $400 - 150 = 250$  px,box2 的实际宽度为  $400 - 50 = 350$  px。运行结果如图 3-35 所示。



图 3-35 flex-shrink 收缩效果

(4) flex-basis 属性。flex-basis 属性用来定义为子元素预留的主轴尺寸。语法为

```
flex-basis: | auto
```

默认值为 auto。取值为 auto 时,实际使用的值是主轴尺寸属性的值,即 width 或 height 属性的值。如果主轴尺寸属性的值也是 auto,则要根据其内容和 flex-grow 属性来分配空间。取值为长度值时,首先为该元素预留空间,预留后剩余的空间再分配给所有子元素。这样一来,该元素除了预留空间外,还会分得容器的剩余空间。在父元素的空间有剩余时,该元素分配的空间 = flex-basis + flex-grow;在父元素空间不足时,该元素分配的空间 = flex-basis - flex-shrink。例如,父元素 father 下有两个子元素 box1 和 box2。将 flex-shrink 中的 HTML 代码样式修改为

```

* {
  margin:0;
  padding:0;
}
.father{
  border:5px solid #000;
  width:400px;
  display:-webkit-flex;
  display:flex;
  flex-flow:row;
}
.fatherdiv{
  border:1px solid #000;
  height:50px;
}
.box1{
  -webkit-flex-grow:1;
  flex-grow:1;
  flex-basis:200px;
  -webkit-flex-basis:200px;
  background:#7f3ffd;
}
.box2{
  -webkit-flex-grow:1;
  flex-grow:1;
  background:#ef39d5;
}

```

在上述代码中,父元素的宽度是 400 px。box1 定义了 flex-basis 属性,预留了 200 px 的空间。而 box2 未定义 flex-basis 属性,并未预留空间。这样一来,父元素剩余的空间就变成了  $400 - 200 = 200$  px。由于 box1、box2 的 flex-grow 属性的值都是 1,故由 box1、box2 均分父元素的剩余空间,各分得 100 px。最终结果是,box1 的实际宽度为  $200 + 100 = 300$  px, box2 的实际宽度为 100 px。运行结果如图 3-36 所示。



图 3-36 flex-basis 预留尺寸效果

(5) flex 属性。flex 属性是 flex-grow、flex-shrink、flex-basis 这 3 个属性的复合属性,用来定义子元素如何分配父元素的空白空间。语法格式为

```
flex:none|[<"flex-grow"><"flex-shrink">? ||<"flex-basis">]
```

取值为关键字 none 时,相当于 flex-grow 的值为 0、flex-shrink 的值为 0、flex-basis 的值为 auto。也就是说,flex:none 就相当于 flex:00auto。除此之外,flex 属性还可以使用以下常用值:flex:initial,这是 flex 属性的初始值,它表示 flex-grow、flex-shrink、flex-basis 这 3

个属性均使用默认值,就相当于 `flex:0 1 auto`; `flex:auto` 相当于 `flex:1 1 auto`,表示子元素的实际尺寸为 `width` 或 `height` 属性的值,并且子元素可以根据弹性容器的空间自由伸缩;`flex:<positive-number>` 相当于 `flex:<positive-number> 10%`,表示为子元素预留的主轴尺寸为 0,并根据设置的比率分配弹性容器的剩余空间。如果一个伸缩容器里的所有项目都使用此模式,则它们按各自指定的伸缩比率分配弹性容器的剩余空间。

HTML 代码如下。

```
<div class="flex-container">
  <div class="flex-item item1">flexitem1</div>
  <div class="flex-item item2">flexitem2</div>
  <div class="flex-item item3">flexitem3</div>
</div>
```

设置 CSS 代码为

```
.flex-container{
  display:-webkit-flex;
  display:flex;
  width:400px;
  height:250px;
  background-color:lightgrey;
}
.flex-item{
  background-color:cornflowerblue;
  margin:10px;
}
.item1{
  -webkit-flex:2;
  flex:2;
}
.item2{
  -webkit-flex:1;
  flex:1;
}
.item3{
  -webkit-flex:1;
  flex:1;
}
```

效果如图 3-37 所示。

(6) `align-self` 属性。`align-self` 属性用来定义当容器在侧轴方向上有额外的空白空间时,子元素如何分配多余的空间,或当容器的空间不足时,如何分配溢出的空间。

```
align-self:auto|flex-start|flex-end|center|baseline|stretch
```

`align-self` 属性用来覆写容器上的 `align-items` 属性,除了 `align-items` 属性的所有可选值外,`align-self` 属性还可以设置为 `auto`,表示其计算值是弹性容器的 `align-items` 属性值;如果



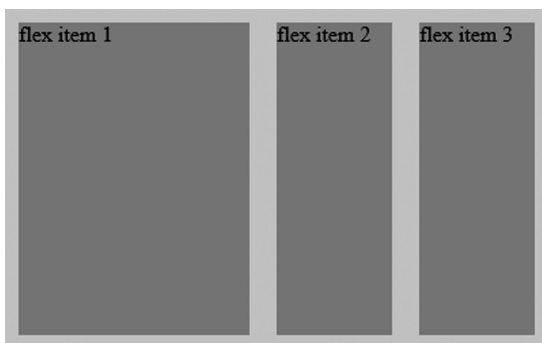


图 3-37 flex 分配子元素空间效果

弹性容器未定义 align-items 属性,则为 stretch。各个值解析如下。

①auto。如果 align-self 的值为 auto,则其计算值为元素的父元素的 align-items 值,如果其没有父元素,则计算值为 stretch。

②flex-start。弹性盒子元素的侧轴(纵轴)起始位置的边界紧靠住该行的侧轴起始边界。

③flex-end。弹性盒子元素的侧轴(纵轴)起始位置的边界紧靠住该行的侧轴结束边界。

④center。弹性盒子元素在该行的侧轴(纵轴)上居中放置(若该行的尺寸小于弹性盒子元素的尺寸,则会向两个方向溢出相同的长度)。

⑤baseline。若弹性盒子元素的行内轴与侧轴为同一条,则该值与 flex-start 等效。其他情况下,该值将参与基线对齐。

⑥stretch。若指定侧轴大小的属性值为 auto,则其值会使项目的边距盒的尺寸尽可能接近所在行的尺寸,但同时会遵照“min/max-width/height”属性的限制。

HTML 代码如下。

```
<div class="flex-container">
  <div class="flex-itemitem1">flex-start</div>
  <div class="flex-itemitem2">flex-end</div>
  <div class="flex-itemitem3">center</div>
  <div class="flex-itemitem4">baseline</div>
  <div class="flex-itemitem5">stretch</div>
</div>
```

设置 CSS 代码为

```
.flex-container{
display: -webkit-flex;
display: flex;
width: 400px;
height: 250px;
background-color: lightgrey;
}
.flex-item{
background-color: cornflowerblue;
```

```
width:60px;
min-height:100px;
margin:10px;
}
.item1{
-webkit-align-self:flex-start;
align-self:flex-start;
}
.item2{
-webkit-align-self:flex-end;
align-self:flex-end;
}
.item3{
-webkit-align-self:center;
align-self:center;
}
.item4{
-webkit-align-self:baseline;
align-self:baseline;
}
.item5{
-webkit-align-self:stretch;
align-self:stretch;
}
```

效果如图 3-38 所示。

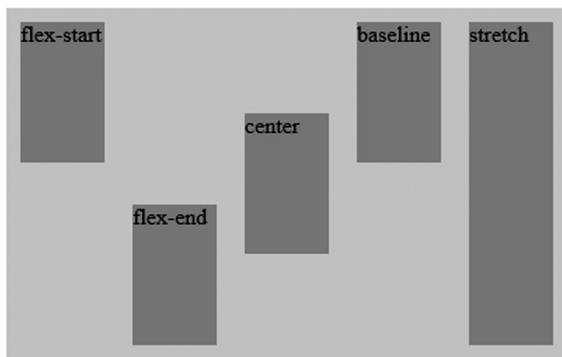


图 3-38 align-self 分配元素空间效果

### 3.5.4 任务实施

任务实施的源代码详见 Task3-9.html。其中：

(1)在父容器导航 element 中有左、中、右 3 个部分，给 element 设置“display: flex;”，使之成为弹性盒模型，并设置主轴方向为 row，同时将容器分为 10 份，左、中、右各占 1/3。

(2)当窗口小于等于 640 px 时,添加媒体查询,将左、中、右的“order:0”3 个部分将按自然顺序排列。

### 3.5.5 任务实训

利用媒体查询,实现 3 种布局效果,当屏幕小于 300 px 时,4 个方块显示效果如图 3-39 所示;当屏幕为 300~500 px 时,方块显示效果如图 3-40 所示;当屏幕为 500~800 px 时,显示效果如图 3-41 所示。



图 3-39 屏幕小于 300 px 效果



图 3-40 屏幕为 300~500 px 效果



图 3-41 屏幕为 500~800 px 效果

其实现的完整代码参见 Task3-10.html。