高等院校计算机应用系列教材

网页设计与网站建设实例教程(微课版)

赵伟　韩颖　主编

清華大學出版社
北　京

内 容 简 介

本书系统介绍了网页设计与网站建设的基础知识。本书共分为12章，主要内容包括：HTML基础与网页结构、HTML5入门、文字和段落排版设计、HTML5表单、HTML5超链接技术解析、HTML5多媒体技术、CSS3概述、CSS3选择器、CSS3文本属性、高级CSS操控、网页布局、构建企业网站等。书中通过大量实例对各种关键技术进行了深入浅出的分析，并在最后一章综合运用全书所学知识，介绍企业网站建设的基本流程和风格设计。

本书内容丰富、结构合理、思路清晰，语言简练流畅，示例翔实。本书既适合作为高等院校相关专业的教材，也适合从事网页设计与网站建设的人员参考学习。

本书的电子课件、教案和习题答案可以通过扫描前言中的“配套资源”二维码获取。读者还可以扫描书中的视频二维码，直接观看教学视频。

图书在版编目(CIP)数据

网页设计与网站建设实例教程 : 微课版 / 赵伟, 韩颖主编. -- 北京 : 清华大学出版社, 2026. 5. -- (高等院校计算机应用系列教材). -- ISBN 978-7-302-71184-1

Ⅰ. TP393.092

中国国家版本馆CIP数据核字第2026K9N971号

责任编辑：胡辰浩
封面设计：高娟妮
版式设计：妙思品位
责任校对：成凤进
责任印制：丛怀宇

出版发行：清华大学出版社
网　　址：https://www.tup.com.cn，https://www.wqxuetang.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-83470000　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn

印 装 者：三河市铭诚印务有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**16.75　　**字　　数：**408 千字
版　　次：2026 年 6 月第 1 版　　**印　　次：**2026 年 6 月第 1 次印刷
定　　价：79.80 元

产品编号：112083-01

前言

随着跨平台与离线Web应用需求的日益增长，HTML5和CSS3已成为新一代Web开发的主流技术，全面取代传统的HTML4。本书紧扣Web前端技术的发展脉络，系统阐述HTML5与CSS3的核心知识与实战应用。本书以实例驱动教学，重点讲解如何运用HTML5与CSS3实现响应式网页布局与Web应用开发，内容覆盖主流浏览器，包括Chrome、Safari、Firefox等。通过丰富的案例与项目实践，帮助读者掌握前端开发技术的精髓，提升综合设计与建设能力。

本书以读者的学习兴趣和实际需求为出发点，合理安排知识结构，遵循由浅入深、循序渐进的原则，通过图文并茂的方式讲解应用HTML5+CSS3设计网页的基本知识和常用技巧。本书共分12章，主要内容如下。

第1章介绍HTML5的发展历程、文档结构、编写方法等基础知识。

第2章介绍HTML5入门的基础知识。

第3章介绍如何在网页中插入文字和进行段落排版。

第4章介绍如何在网页中插入表单。

第5章介绍HTML5的超链接技术。

第6章介绍HTML5的多媒体技术。

第7章介绍CSS3的基础知识与基本用法。

第8章介绍CSS3的选择器。

第9章介绍应用CSS3设计网页文本样式的方法。

第10章介绍使用CSS3控制图像大小、边框样式以及阴影等特殊效果的方法。

第11章通过具体实例重点介绍弹性盒布局的页面排版模式。

第12章通过具体案例介绍如何构建企业网站。

本书内容翔实、结构合理、思路清晰，语言简练流畅。在每一章中，结合所讲述的关键技术和难点，穿插了大量极富实用价值的示例。每章末尾都安排了有针对性的练习题，这些题目有助于读者巩固基本概念，并培养实际动手能力。

本书面向希望学习HTML5和CSS3的Web开发人员，既适合作为高等院校相关专业的教材，也适合从事网页设计和网站开发的人员学习参考。

由于作者水平有限，书中难免存在不足之处，恳请各位专家和广大读者批评指正。在编写本书的过程中，我们参考了多篇相关文献，在此向这些文献的作者表示衷心的感谢。我们的电话是010-62796045，电子邮箱是992116@qq.com。

本书配套的电子课件、教案和习题答案文件可以扫描下方的二维码获取。读者还可以扫描书中的视频二维码直接观看教学视频。

配套资源

作　者

2025年12月

目录

第 1 章

HTML：网页的起点

HTML5作为一种新网页标准，自2010年问世以来，迅速获得产业界与开发者社区的广泛认可。在微软公司MIX10技术大会上，IE9预览版的发布首次使业界能够系统了解Web技术的演进路线：第一阶段(Web1.0)以静态网页技术HTML和CSS为核心；第二阶段(Web2.0)以Ajax技术体系(包括JavaScript、DOM和异步交互)为特征；第三阶段则宣告HTML5与CSS3将引领新时代。经过W3C组织长达八年的标准化工作，2014年10月，该标准正式定稿，标志着互联网正式迈入新发展阶段。

本章学习目标：

- 了解Web、网页和网站之间的关系
- 了解HTML5的特点和新特性
- 掌握静态网页的工作原理
- 了解HTML5能够解决的问题

1.1 网页的基本概念

教学视频

随着Internet技术的普及，Web应用变得非常广泛，网页已经被越来越多的人所熟悉。因此，Web开发已成为一个热门领域。学习Web开发，首先需要了解什么是网页，以及网页是如何构建并呈现在用户面前的。

1.1.1 Web

Internet(中文正式译名为“互联网”，也称“国际互联网”)是指通过标准化通信相互连接而成的全球性计算机网络。它由数以万计的网络和上亿台计算机设备互联构成，是目前世界上规模最大、覆盖最广的信息资源平台，为全球用户提供信息检索与资源共享服务。

Internet通过革命性的超文本架构，彻底重构了信息组织的范式。这种非线性的信息关

联机制使得全球网络中的任意数据节点都能通过智能链接实现自由跳转，从而形成多维度的知识网络。作为互联网的核心服务层，Web(World Wide Web，万维网)本质上是一个基于HTTP协议的超媒体生态系统。Web架构通过三大支柱技术构建信息交互体系。

1) 超文本交互系统

以超文本标记语言(Hyper Text Markup Language，HTML)和富文本格式(Rich Text Format，RTF)等标记语言构建非线性信息结构，通过嵌入式链接实现跨文档智能跳转，形成全局知识网络。日常网页中的可点击元素均属此类超文本范式。

2) 超媒体融合引擎

Internet在超文本基础上整合多媒体资源(音频、图形和动画)，支持多模态内容触发机制。用户的交互突破文本限制，将信息的链接扩展到整个Internet。Web作为一种超文本信息系统，使得文本不再是固定的线性结构，而是能从一个文本跳转到另一个文本，同时激活声音、显示图形，甚至播放动画。用户可以通过单点操作体验多媒体的复合效果。

3) HTTP通信协议

作为Web核心传输规范，HTTP建立了客户端与服务器之间的标准化对话机制。其无状态请求响应模型支撑着全球互联网90%以上的数据交换，并通过持续迭代(如HTTP/2的多路复用、HTTP/3的QUIC协议)提升传输效率。

这三个核心要素相互依存、协同工作，形成了稳定的技术架构。该技术三角共同实现以下目标。

(1) 分布式资源的语义化链接(超文本)。

(2) 多维度信息的沉浸式呈现(超媒体)。

(3) 跨平台通信的可靠保障(HTTP协议)。

该技术三角奠定了现代Web从文档互联到智能服务的进化基础，使原本离散的互联网节点转化为具有语义关联的智能信息网络。用户可以通过可视化界面实现知识图谱导航、多模态内容检索、跨平台服务调用等高级功能。当前演进中的Web3.0体系，更在此基础上引入了区块链存证、去中心化身份等新型网络能力。

1.1.2 网页与网站

什么是网页？什么是网站？两者之间有什么联系与区别？

1. 网页

构建万维网(WWW)的基本单位是网页。当我们通过手机或电脑浏览新闻或搜索某个关键词时，所呈现的内容就是网页。网页中包含所谓的“超链接”，而文字与图片是构成网页的两种最基本元素。我们可以简单地理解为：文字是网页的内容，而图片则是网页的外观。除此之外，网页的元素还包括动画、音乐和程序等。

网页是包含HTML标签的纯文本文件，可以存放在世界某个角落的某台计算机中，作为万维网中的一“页”。它采用超文本标记语言格式(HTML)，文件扩展名为.html或.htm。

互联网内容的呈现形式基于内容响应机制分为两大范式：静态网页与动态网页，其核心差异在于页面内容是否随用户请求而动态变化。

静态网页作为基础网页形态，其本质为预编译的HTML文档(文件扩展名通常为.htm或.html)。这种网页的内容在服务器端已完全固化，发布后不再变动。静态页面可以集成文本、图像、音频、Flash动画及客户端脚本(如JavaScript)等多媒体元素，但所有资源均预置完成，用户访问时只需直接下载渲染。静态网页的优势在于低服务器负载与高搜索引擎兼容性，因此非常适合信息更新频率低的展示型场景(如企业官网)。然而，内容修改需要手动重编文件，维护成本较高。需要特别澄清的是，“静态”仅指内容生成的逻辑，而非视觉表现——此类页面仍可通过GIF动画、Flash特效或滚动字幕等方式实现动态视觉效果。

动态网页是现代Web应用的核心载体，通过服务器端技术实现内容实时生成与用户交互。其运作机制主要依赖数据库驱动的内容管理系统。当用户发起请求时(如提交表单或触发操作)，服务器动态组合数据与模板生成个性化页面。这种技术架构使得网站能够支持用户认证(注册和登录)、数据提交、个性化推荐等多种功能。主流实现方案包括：基于Java生态的JSP、微软技术栈的ASP/ASP.NET，以及开源的PHP框架。与静态网页的本质区别在于，动态网页的内容生成发生在服务器响应阶段，而非预先固化存储。

从用户体验层面观察，静态网页与动态网页在基础内容呈现(如图文展示)上具有相似性，但在技术实现与运维方面存在显著差异。

2. 网站

网站(Web Site)是指在Internet上根据一定规则，使用HTML等工具制作的、用于展示特定内容的相关网页的集合。简单地说，网站是一种沟通工具，人们可以通过网页浏览器来访问网站，获取所需资讯或享受网络服务。网页是网站架构的基本单元，也是互联网内容呈现的基本形式。简而言之，网站是网页的有机集合体。若仅配置域名与服务器空间而未部署网页文件，则该网站将呈现“空壳”状态，无法提供有效访问服务。网站的大小在概念上是相对的，大型网站(如新浪、网易等门户网站)页面数量众多，可能分布于多台服务器上；而小型网站(如一些个人网站)可能只有几个页面，仅在某台Web服务器上占据很小的空间。

网站核心三要素包括：域名、网站空间与程序。

(1) 域名(网络定位标识)作为网站的访问入口，通过DNS解析转换为服务器IP地址，实现用户精准定位(例如，在www.baidu.com中“baidu”是主域名主体，“.com”是顶级域名后缀)。命名规则要求仅允许使用字母、数字及连字符(-)，单级长度不得超过63字符。层级结构呈现左低右高(如在blog.example.com中，“blog”为三级域名)。域名具有唯一性与资源稀缺性，优质域名的价值可能升值至数万甚至数百万元。

(2) 网站空间(数字内容载体)是指托管网站文件的虚拟存储环境(如服务器磁盘空间)，用于承载网页、图片、数据库等资源。存储容量决定了可存放的文件规模，性能要求则需保障稳定性、高速访问及安全性(如抗DDoS能力)。服务类型方面，虚拟主机(共享资源)适合小型网站，而云主机或独立服务器则可以满足高流量需求。

(3) 程序(功能实现引擎)是由编程语言(如PHP或Python)编写的源代码集合，包含前端页面(HTML、CSS、JavaScript)、后端逻辑及数据库。其核心能力在于驱动用户交互(如登录和支付)与数据处理，并支持通过CMS(如WordPress)实现非技术人员的内容管理。

这三者构成了网站运行的最小完备集合：域名指向访问入口，网站空间存储数据实

体，程序实现功能逻辑，缺一不可。

对于初学者而言，网站可以理解为由网页文件和资源文件组成的文件夹系统。设计网站的过程就是创建各个网页，并按照逻辑关系将它们分类存储在主文件夹及其子文件夹中。这些子文件夹的类别和数量可以根据实际项目需求灵活设置，没有固定的标准。

网站的入口页面通常称为“首页”或“主页”，其质量直接影响用户的第一印象和访问体验。按照行业惯例，首页文件通常命名为index.html、index.htm或default.htm等标准名称，这些文件必须放置在网站根目录下，才能被服务器正确识别为默认访问页面。网站中的其他页面数量可以根据需要随时增减，以保持灵活的可扩展性。

1.2 网页核心三要素

当前流行的Web标准设计方式是采用HTML(或XHTML)、CSS和JavaScript，将网页的内容、表现和行为分离。HTML、CSS和JavaScript都是跨平台的，与操作系统无关，只依赖于浏览器。目前，所有的浏览器都支持这三种技术。

1.2.1 HTML5概述

HTML的全称是超文本标记语言(Hyper Text Markup Language)，是Internet上用于编写网页的主要语言。它提供了一种简洁而强大的文件定义方式，可以设计出丰富多彩的超媒体文件。

HTML文件采用纯文本格式。所谓“超文本”，主要是指其强大的超链接功能。通过超链接，图片、声音、视频以及其他网页或网站可以相互连接，从而构成内容丰富的Web页面。

HTML是最早的超文本标记语言，它的发展经历了多个版本。在发展过程中，尤其是从HTML4.0开始，淘汰了很多标签和属性。2004年成立的Web超文本应用技术工作组(Web Hypertext Application Technology Working Group，WHATWG)创立了HTML5规范，同时开始专门针对Web应用开发新的功能。2006年，W3C介入HTML5的开发，并于2008年发布了HTML5的工作草案。2009年，W3C停止对XHTML2的更新。2010年，HTML5开始用于解决实际问题。此时，各大浏览器厂商开始对旗下产品进行升级，以支持HTML5的新功能，从而推动了HTML5规范的持续完善。2014年10月29日，HTML5规范最终制定完成并公开发布。

1. HTML5的目标

HTML5的目标是创建更简洁的Web程序，并编写更易读的HTML代码。例如，为了使Web应用程序的开发变得更容易，HTML5提供了多种API，同时引入了新的属性和元素，以使HTML更加简洁。总体来说，HTML5为下一代Web平台提供了许多全新的功能。

HTML5提供了以下革命性的新功能。

(1) 在HTML5之前，有很多功能必须使用JavaScript等脚本语言才能实现，例如在登录页面中经常使用的让文本框获得光标焦点的功能。而在HTML5中，这一功能只要使用元素

的标签属性即可实现。这使得整个页面变得非常清晰、直观且容易理解。因此，Web设计者可以非常放心大胆地使用HTML5中这些新增的属性标签。HTML5中提供了大量可替代脚本的属性标签，使得开发出来的界面语言更加简洁易懂。

(2) HTML5使页面结构变得清楚明了。以前常用的div标签也不再使用了，而是使用HTML5提供的更加语义化的结构标签。这样编写出来的界面结构显得更加清晰，各部分内容也一目了然。

虽然HTML5宣称的立场是“非革命性的发展”，但是它所带来的功能令人期待，且使用起来非常方便，因此深受Web设计者和开发者的欢迎。

2. HTML的特点

HTML文档制作简单且功能强大，支持导入多种数据格式的文件，这也是万维网(WWW)盛行的原因之一。HTML作为网页开发的基础语言，具有以下核心特点。

(1) 开发简易性：仅需使用文本编辑器即可编写HTML文档，无须复杂的开发环境，学习门槛极低。

(2) 高度可扩展：通过子类元素机制支持功能扩展，持续演进的标准可满足各类增强需求，兼容各类新标签和属性的引入。

(3) 跨平台兼容：完全独立于操作系统，仅需浏览器即可呈现内容，这一特性极大推动了万维网的普及。

(4) 全网通用性：作为互联网的标准标记语言，HTML支持构建包含多媒体元素的复杂页面，确保内容在各种终端设备(如PC和移动设备)及浏览器上均可访问。

上述特性共同构成了HTML作为Web基石的关键因素，使其能够持续适应互联网发展的需求。

3. HTML5新特性

1) 兼容性设计

HTML5在设计之初就特别重视对历史文档的兼容性。由于互联网上已存在大量使用旧版HTML标准的网页内容，HTML5采用渐进式增强策略，以确保对历史HTML文档的完全兼容。当浏览器不支持新特性时，系统会自动启用备选方案，确保新标准能够平滑过渡而非彻底颠覆现有技术体系。当浏览器不支持HTML5新特性时，系统会自动触发降级处理机制，在不影响页面基本功能的前提下实现优雅降级。这种向后兼容的特性既保护了既有网络资源，也为开发者提供了更平稳的技术升级路径。

2) 需求驱动开发

HTML5的新元素设计遵循“实践驱动”的开发理念。所有新增元素均基于对海量网页的统计分析(例如，Google对数百万页面的研究)，针对开发者的实际需求进行标准化提炼。例如，在发现开发者普遍使用 等重复结构后，HTML5将其标准化为语义化标签(如<header>)。这种从实际应用中提炼标准的方法，确保了新功能能够切实解决开发中的痛点，而非脱离实际的理论构想。

3) 用户优先原则

HTML5规范采用“用户至上”的设计哲学，其优先级体系明确划分为：用户需求>开

发者便利>浏览器兼容性>标准组织意见>理论完整性。这种务实导向使得HTML5在语法解析上表现出极高的容错性——即使是不规范的代码写法(如未闭合标签或大小写混用)也能被现代浏览器正常解析。这种设计妥协虽然牺牲了部分理论严谨性，但显著提升了实际开发中的灵活性和用户体验。

例如，以下几种属性写法均能被正确解析。

(1) 带引号的规范形式：

```
<input disabled="disabled">
class="header active"
src="image.png"
style="color: red;"
```

(2) 省略属性的简写形式：

```
<input disabled>
```

(3) 省略引号的简写形式：

```
class=header active
```

(4) 大小写混用形式：

```
SRC="image.PNG"
```

(5) 单引号形式：

```
style='color: red;'
```

这种设计体现了HTML5“开发者友好”的理念——通过兼容实际开发中的习惯性写法(即使不够严谨)来提升编码效率。然而，需要强调的是，虽然这种灵活性降低了入门门槛，初学者仍应优先掌握标准写法，以培养良好的代码规范意识。在生产环境中，建议始终采用W3C推荐的标准写法(如使用小写属性名并加双引号)，以确保兼容性和可维护性。

4) 安全增强机制

HTML5规范创新性地采用了基于同源(Same-Origin Policy)策略的安全机制，这一设计具有三大核心优势：首先，该模型通过标准化的同源策略实现了跨API的统一安全管控；其次，其简洁的架构设计大幅降低了开发者的使用门槛；最重要的是，它彻底摒弃了传统跨域通信中那些看似巧妙实则存在安全隐患的临时解决方案，为浏览器环境提供了原生、可靠的安全对话机制。这种“安全优先”的设计理念，使得开发者无须再冒险使用不规范的“hack手法”，就能实现安全的跨域数据交互。

5) 结构与表现分离

HTML5在代码结构与表现分离方面实现了重大突破。作为新一代标准，它系统性地将内容(HTML)与样式(CSS)解耦，彻底移除了旧版HTML中过时的表现性标签和属性，推动了内容与样式的清晰分工。这种设计哲学不仅使代码更符合语义化要求，还显著提升了网页的可维护性和可访问性，标志着Web开发向专业化、模块化方向的重要演进。

6) 极简主义哲学

HTML5通过简化文档声明(DOCTYPE)、字符集定义，以及用原生API替代脚本代码等方式，降低了代码复杂度。

7) 普适性访问

访问原则主要包含三大核心理念：首先，在可访问性方面，通过深度整合WAI(Web可访问性倡议)和ARIA(可访问富互联网应用)标准，将屏幕阅读器等无障碍技术原生融入HTML元素；其次，强调媒体中立性，确保所有功能都能跨设备、跨平台稳定运行；最后，注重语言包容性，新增了如支持东亚文字的Ruby注释等特性，全面覆盖不同语种的排版需求。这些设计共同构建了一个更具包容性的Web环境。

8) 原生能力替代

HTML5通过原生支持彻底改变了传统Web依赖插件的开发模式。相较于插件方案，HTML5克服了四大缺陷：其一，安装失败风险(如缺失运行时环境)；其二，插件可被主动禁用(典型的例子如Flash被浏览器屏蔽)；其三，安全漏洞频发；其四，插件与DOM元素集成困难(涉及边界处理和图层叠加问题)。HTML5提供了更优雅的解决方案。以绘图功能为例，过去在HTML4.0中绘制简单对角线都需要复杂技巧，而HTML5的canvas元素使其变得轻而易举。

更值得称道的是HTML5 API的协同性。例如，用canvas等原生元素取代插件方案，解决插件存在的安全风险、兼容性问题及集成困难等缺陷，同时实现了更高效的API组合应用(如video与canvas的交互)。video元素的画面帧能够无缝渲染到canvas上，通过点击事件即可触发视频跳转播放。这种模块化设计极大地提升了开发效率和用户体验。

1.2.2 CSS

CSS(Cascading Style Sheets，层叠样式表)作为网页设计的核心样式语言，主要服务于HTML和XML文档的视觉呈现。它通过两种方式发挥作用：在静态层面上修饰网页外观，在动态层面上与JavaScript等脚本配合，实现交互式样式变化。其技术特性突出表现在3个方面：精确到像素的布局控制能力、全面的字体样式支持体系，以及灵活的网页对象样式编辑功能。

这种样式定义语言既可内嵌于HTML文档，也可以作为独立的.css文件存在，通过声明字体、颜色、定位等属性来规范网页内容的显示形式。“层叠”特性是其核心机制：当多个样式规则作用于同一元素时，系统会根据选择器特异性构建优先级层次(如ID选择器>类选择器>标签选择器)，具体规则会自动覆盖通用规则，最终形成级联渲染效果。这种设计既确保了样式控制的精确性，又保持了代码的可维护性。

前文已介绍HTML的一些优点，包括简单易学、易于推广、易于扩展，以及平台无关性，使开发者无须考虑浏览器的兼容性问题。然而，HTML将内容与样式混编的设计模式导致代码臃肿和维护困难。CSS技术的出现有效解决了这些问题，主要体现在以下5个关键维度。

(1) CSS 2.0以上版本实现了内容与表现的彻底分离，将视觉设计代码独立存储为样式文件，使HTML专注于内容结构化，这种分离架构显著提升了网页对搜索引擎的友好度。

(2) CSS布局相比传统表格布局能够缩减约50%的文件体积，从而大幅提升页面加载速度。

(3) 通过集中化管理样式文件，开发者仅需修改CSS即可全局更新网站的视觉风格，极

大提升了维护效率。

(4) CSS的级联特性通过样式继承机制实现代码复用，浏览器按照选择器优先级和定义顺序智能应用样式规则，既确保了设计的一致性，又降低了维护成本。

(5) 标准化的代码结构使搜索引擎能够更精准地抓取和分析网页核心内容，从而优化SEO效果。这些特性共同构成了CSS在现代Web开发中的不可替代性。

1.2.3 JavaScript脚本语言

JavaScript作为Web技术的三大基石(HTML、CSS、JavaScript)之一，是一种面向网络的脚本语言，已被广泛用于Web应用开发。它通常用于为网页添加各式各样的动态功能，为用户提供更流畅美观的浏览体验。

1995年，Netscape公司的Brendan Eich在10天内设计出了这门语言，最初命名为Mocha，后更名为LiveScript，最终因与Sun公司的合作更名为JavaScript。该语言最初集成在Netscape Navigator浏览器中。虽然名称相似，但JavaScript与Java在语法和设计理念上存在本质区别，其核心特性更接近Scheme和Self语言。JavaScript是一种解释型语言，通过内置的JavaScript引擎执行，主要运行在客户端浏览器环境，专门用于增强HTML网页的交互性和动态效果。

作为完整的编程语言，JavaScript具备标准语言的基本要素：包含多种基本数据类型和特殊类型的数据系统、完整的运算符体系、变量存储机制以及表达式处理能力。其核心价值在于能够为静态网页注入动态功能，包括表单验证、内容更新、动画效果等，从而显著提升用户体验。现代Web应用中，JavaScript已成为实现复杂前端交互不可或缺的技术基础，从简单的DOM操作到单页应用(SPA)开发，均依赖于其强大功能。

1.3 编写第一个HTML5页面

1.3.1 使用HTML5编写简单的Web页面

HTML5作为一种纯文本标记语言，其开发环境具有高度灵活性。任何文本编辑工具均可用于编写HTML代码，从最简单的系统自带记事本到专业级的代码编辑器(如Sublime Text、WebStorm等)都能胜任。开发者只需创建以.html或.htm为扩展名的文件(例如index.html)，输入HTML代码并保存，即可通过浏览器直接查看渲染效果。这种低门槛的开发方式使得HTML5成为最容易入门的Web技术之一。

【例1-1】使用Windows系统自带的记事本工具新建一个文本文件，并将其保存为CH01-001.html(注意，扩展名为.html而不是.txt)。

代码如下：

```
<!DOCTYPE html>
<html>
```

```
    <head>
      <meta charset="UTF-8">
      <title>我的第一个HTML5页面</title>
    </head>
    <body>
      <h1>欢迎来到HTML5世界</h1>
      <p>这是一个简单的HTML5页面示例</p>
    </body>
</html>
```

该页面的运行效果如图1-1所示。

通过短短几行代码就完成一个页面的开发，可见HTML5语法的简洁性。接下来，将逐句分析HTML5文档的组成部分。

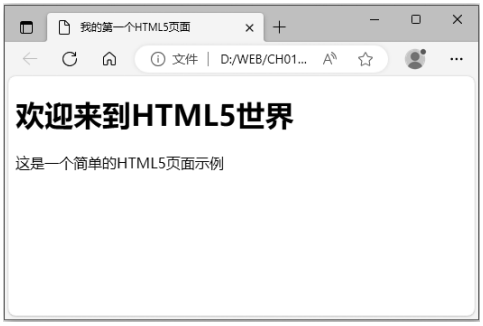

图1-1　例1-1结果展示

第一行代码如下：

```
<!DOCTYPE html>
```

这短短几个字符甚至不包括版本号，就能告诉浏览器需要一个doctype来触发标准模式，简明扼要。接下来的代码是：

```
<meta charset="UTF-8">
```

这行代码说明了文档的字符编码，确保浏览器正确解析文档。在<head>标签中使用<meta>标签定义文档的字符编码，常用的字符编码包括中文简体(GB2312)、中文繁体(Big5)和通用字符编码(UTF-8)。

在HTML5中，标签名称、结束符的大小写以及属性是否加引号都不受限制。以下代码是等效的：

```
<meta charset="utf-8">
<META charset="utf-8">
<meta charset=UTF-8>
```

根据HTML5规范，<html>、<head>和<body>等结构性标签可以省略不写，开发者可以直接编写需要显示的内容，现代浏览器能够正确解析并渲染页面。以下代码去掉了<html>、<head>和<body>标签，但效果与图1-1相同：

```
<!DOCTYPE html>
  <meta charset="UTF-8">
  <title>我的第一个HTML5页面</title>
```

```
<h1>欢迎来到HTML5世界</h1>
<p>这是一个简单的HTML5页面示例</p>
```

然而，从代码规范和工程化的角度考虑，建议开发者始终显式声明这些基础结构标签。这样做不仅能够提升代码的可读性(便于团队协作和维护)，还能确保文档结构的完整性，避免潜在的兼容性问题。通过遵循这一最佳实践，开发者可以在保持代码简洁的同时，兼顾长期的可维护性需求。

1.3.2 使用HTML5的结构化元素

上一节介绍了一个HTML5页面的创建过程。下面将通过一个较为完整的页面来介绍HTML5的页面特征。

通过研究Web页面，我们发现，使用一些带有语义性的标记可以加快浏览器解释页面中元素的速度，例如早期的<samp>、<var>元素。HTML5继承了这些元素，并根据用户使用最为频繁的类名和ID不断开发新的标签，以更好地体现开发者的真实意图。接下来，将通过实例说明HTML5是如何使用这些全新的特征来结构化页面元素的。

【例1-2】创建一个名为CH01-002.html的页面，内容包括文本、超链接、图片和版权信息等。

代码如下：

```
<div id="html">
 <div id="head">
  <meta charset="UTF-8">
  <div id="title">语义化布局示例</div>
 </div>
 <div id="body">
  <div id="header">
   <h1>网站主标题</h1>
   <div id="nav">
    <ul>
     <li><a href="#">首页</a></li>
     <li><a href="#">产品中心</a></li>
    </ul>
   </div>
  </div>

  <div id="main">
   <div id="article">
    <div id="article-header">
     <h2>技术文章标题</h2>
     <div id="time">发布于2025年7月24日</div>
    </div>
    <div id="section">
```

```
        <h3>章节标题</h3>
        <p>正文段落内容...</p>
        <div id="figure">
          <img src="test-image.jfif" alt="技术示意图" width=80>
          <div id="figcaption">技术原理图示</div>
        </div>
      </div>
    </div>

    <div id="aside">
      <h3>相关推荐</h3>
      <ul>
        <li><a href="#">关联技术文章</a></li>
      </ul>
    </div>
  </div>

  <div id="footer">
    <p>© 2025 版权所有</p>
    <div id="address">联系方式: contact@example.com</div>
  </div>
 </div>
</div>
```

运行以上代码，效果如图1-2所示。尽管上述代码没有任何错误，并且在HTML5环境中能够良好运行，但该页面结构的很多部分对于浏览器来说都是未知的。这是因为浏览器主要通过id来定位元素。因此，如果开发者使用不同的id，可能会导致元素在页面中的位置不明确，从而影响页面解析的速度。

图1-2　例1-2结果展示

幸运的是，HTML5中新增的元素可以快速定位某个标签，明确地表示其在页面中的位置。将上述代码修改为HTML5支持的格式，创建一个名为CH01-003.html的页面，代码如下：

```
<!DOCTYPE html>
<html lang="zh-CN">
<head>
  <meta charset="UTF-8">
  <title>语义化布局示例</title>
</head>
<body>
  <header>
```

```
    <h1>网站主标题</h1>
    <nav>
     <ul>
      <li><a href="#">首页</a></li>
      <li><a href="#">产品中心</a></li>
     </ul>
    </nav>
   </header>

   <main>
    <article>
     <header>
      <h2>技术文章标题</h2>
      <time datetime="2025-07-24">发布于2025年7月24日</time>
     </header>
     <section>
      <h3>章节标题</h3>
      <p>正文段落内容...</p>
      <figure>
       <img src="test-image.jfif" alt="技术示意图" width=80 >
       <figcaption>技术原理图示</figcaption>
      </figure>
     </section>
    </article>
    <aside>
     <h3>相关推荐</h3>
     <ul>
      <li><a href="#">关联技术文章</a></li>
     </ul>
    </aside>
   </main>
   <footer>
    <p>© 2025 版权所有</p>
    <address>联系方式: contact@example.com</address>
   </footer>
  </body>
  </html>
```

运行以上代码后，显示效果将与图1-2相同。

从这两段代码来看，使用HTML5新增元素创建的页面代码更加简单和高效。

可以看出，使用 等标签元素没有任何实际意义，因为浏览器不能根据标签的id属性来推断标签的真正含义。而且id值是可以变化的，不利于元素的定位。

而HTML5引入的语义化元素则直接指明了区块功能：<header>明确标识页头，<nav>用于构建页面导航区域，<article>用于构建页面内容区块，<footer>用于标识页脚或文档尾

部。这些元素可以重复使用，且语义清晰，显著提升了开发效率和代码可读性。

此外，部分HTML5语义化元素本身即可构成独立的区块结构，示例如下：

```
<header>
        <article>
                <h1>内容1</h1>
        </article>
</header>
<header>
        <article>
                <h2>内容2</h2>
        </article>
</header>
```

在HTML5中，<article>元素可以创建一个新的节点，并且每个节点都可以拥有自己的单独元素，如<h1>和<h2>。这种结构不仅使内容区域能够各自分段、便于维护，而且简化了代码，使局部修改变得更加方便。

1.3.3　使用CSS美化HTML5文档

在支持HTML5新增元素的浏览器中，样式化各个新增元素变得十分简单。可以对任意一个元素应用CSS，既可以直接设置样式，也可以通过引入CSS文件来实现。

【例1-3】在例1-2的基础上，通过添加若干CSS样式对页面进行美化。

在CH01-003.html的代码中，添加以下代码：

```
 <style type="text/css">
/* 1. 基础页面样式 */
body {
  font-family: 'Microsoft YaHei', sans-serif;
  line-height: 1.6;
  max-width: 1000px;
  margin: 0 auto;
  padding: 20px;
  background-color: #f9f9f9;
}
/* 2. 头部渐变背景 */
header {
  background: linear-gradient(135deg, #4285f4, #34a853);
  color: white;
  padding: 20px;
  border-radius: 8px;
  margin-bottom: 20px;
}
/* 3. 内容卡片阴影效果 */
article, aside {
```

```
  box-shadow: 0 4px 8px rgba(0,0,0,0.1);
  padding: 20px;
  border-radius: 5px;
  background: white;
}
/* 4. 页脚样式 */
footer {
  margin-top: 30px;
  padding: 15px;
  text-align: center;
  background: #f1f1f1;
  border-radius: 5px;
}
</style>
```

运行以上代码，效果如图1-3所示。

图1-3　使用CSS美化HTML5页面

由于某些浏览器(如IE8或更早版本)并不支持HTML5中新增的元素，因此其CSS样式仅适用于IE所支持的元素。为了能够为新增的HTML5元素应用样式，可以在页面的头部标签<head>中加入以下JavaScript代码：

```
    <script type="text/javascript">
        document.createElement('article');
        document.createElement('header');
    </script>
```

考虑到不同浏览器的兼容性，可以对上述JavaScript代码进行优化，使用条件语句确保这段代码仅在浏览器不支持HTML5的情况下执行。

1.4　本章小结

总体而言，HTML5的出现并非偶然，而是业界专家与工程师们针对过去互联网技术面临的复杂问题所做出的回应。基于许多开发者在实际项目实践中经常遇到的问题、习惯性操作和解决方案，并结合当前技术发展的需求与设计原则，HTML5作为一种标准被制定出来。本章系统介绍了网页开发的基础知识体系。首先明确了Web作为基于HTTP协议的信息空间，由网页(单个HTML文档)和网站(相互链接的网页集合)构成。现代网页开发的核心三要素包括HTML5(用于内容结构与语义标记)、CSS(负责控制表现层样式)、JavaScript(实现交互逻辑)。接着，我们介绍了如何搭建编写和运行HTML5的环境，以及如何编写HTML5文档。希望通过本章的学习，读者能够对HTML5有一个全面的认识。

1.5　练习题

1.5.1　单项选择题

(1) HTML5标准的制定组织是(　　)。

A. IEEE　　B. W3C　　C. Mozilla基金会　　D. ECMA

(2) HTML5文档类型声明正确的是(　　)。

A. <!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.01//EN”>

B. <!DOCTYPE html>

C. <!DOCTYPE HTML5>

D. <html5>

(3) (　　)标签用于定义HTML5文档的字符编码。

A. <charset>　　B. <encoding>　　C. <meta>　　D. <head>

1.5.2　填空题

(1) 根据网页内容是否因请求不同而发生变化，可以将网页分为_____和_____。

(2) 由于HTML文件是标准的_____文件，因此可以使用任意文本编辑器来打开和编辑HTML文件，例如Windows自带的_____程序。

(3) 网页的核心三要素包括_____、_____和_____。

1.5.3　简答题

(1) 简单介绍一下网页与网站的关系。

(2) 简单描述HTML5的目标及特性。

(3) 简述静态网页的工作原理。

(4) 编写一个HTML5文档。

第 2 章

HTML5 入门

HTML5通过引入多项技术革新，对传统HTML文档进行了全面升级。本章将总体介绍HTML5与以往版本的不同之处，以及HTML5和HTML4之间的区别。新版本显著优化了文档结构，使其更具逻辑性和可读性，不仅降低了学习门槛，还提升了浏览体验和开发效率。本章将系统讲解HTML5的文档结构，深入解析新增和废除的元素与属性，以及表格的创建和使用，并着重剖析结构性元素的应用实践。

本章学习目标：

- 掌握HTML5的结构和语法
- 掌握HTML5的元素
- 熟悉HTML5中新增和废除的元素
- 掌握HTML5中的常用全局属性及废除属性
- 掌握创建表格的常用标签

2.1 HTML5基础

教学视频

HTML5是超文本标记语言HTML的第5次修订版，标志着近年来Web标准的巨大飞跃。与之前的版本相比，HTML5不仅用于表示Web内容，它还为用户提供了一个无缝的网络环境，使人们无论使用计算机、平板电脑还是智能手机，都能够方便地浏览基于HTML5的各类网站。作为一种网页制作语言，HTML拥有自己的结构和语法规则，HTML5在HTML4的基础上进行了大量的改进和扩展。

2.1.1 HTML5结构

在上一章中，我们已经制作了一个简单的HTML页面。在这个HTML文档中，出现了很多用尖括号括起来的字符，这些带尖括号的字符就是HTML的“标签”。

通常，一个HTML文档中包含许多标签，而且大多数标签是成对出现的。尖括号中没有斜线(/)的标签为起始标签，而尖括号中第一个字符为斜线(/)的标签为结束标签，例如</html>。HTML文件中所有用于定义文档结构和内容的标记被称为标签，标签的格式如下：

```
<标签>HTML语言元素</标签>
```

实际上，整个HTML文档都包含在起始标签<html>和结束标签</html>之间。大多数HTML 元素都可以包含其他 HTML 元素，即HTML元素可以嵌套。包含另一个元素的元素称作“父元素”，而被包含的元素则称为父元素的“子元素”。因此，<title>元素是<head>元素的子元素，而<head>元素是<title>元素的父元素，以此类推。

标签分为成对标签和非成对标签。成对标签由两个部分组成，第一个是开始标签，第二个是结束标签，例如，<table>为成对标签，而
、<hr>等则属于非成对标签。标签不区分大小写，书写形式非常灵活。可使用标签的属性来进一步限定标签，一个标签可以有多个属性，属性的顺序没有限制，各属性项之间用空格分隔。例如：

```
<img src="images/girl.gif" width="50" height="50">
```

通常，HTML文档都包含在开始标签<html>和结束标签</html>之间(除了第一行的DOCTYPE声明)。在<html>元素内部，主要存在以下两个部分。

(1) <head>元素：经常被称为页面的头部，包含页面的相关信息(此处不是页面的主体内容)。例如，它可能包含一个<title>元素和一段页面描述或指示信息，用于告知浏览器从哪里可以找到用于解释文档外观的CSS规则。

(2) <body>元素：通常被称为页面的主体，包含实际希望在浏览器主窗口中显示的内容。

<html>、<head>以及<body>元素构成了一个HTML文档的框架，它们是所有网页构建的基础。一个完整的HTML文档的基本结构如下：

```
<html> <!-- 语法开始 -->
<head>
    <!-- 头部信息，如<title>标签定义网页的标题 -->
</head>
<body>
    <!-- 主体信息，包含网页显示的内容 -->
</body>
</html> <!-- 语法结束 -->
```

可以看到，每个标签都是成对出现的，第一个标签(如<html>)表示标识的开始位置，而第二个标签(如</html>)表示标识的结束位置。<html>标签中包含<head>和<body>标签，而<head>和<body>标签则是并列排列的。

如果将上述代码保存为一个文本文件，并命名为test.html，就可以在浏览器中打开并查看。需要注意的是，由于这个简单的HTML文档尚未包含任何可显示的信息，因此在浏览器中打开时看不到任何内容。

2.1.2 HTML5语法

HTML语言的规范相对简单易懂。从逻辑上分析，这些标签包含的内容可以表示一类对象，也称为网页元素。从形式上来看，这些网页元素通过标签进行分隔，从而表达特定的语义。很多时候，人们将网页标签和网页元素混为一谈；实际上，网页文档就是由元素和标签组成的容器。

一对标签及其之间包含的内容称为“元素”(element)。如图2-1所示，页面中的<h1>元素就是一个例子。

图2-1 元素与标签示意图

标签通常包含左尖括号、右尖括号以及二者间的字母和数字，如<title>。而元素则是指开始标签、结束标签以及二者之间的任何内容。

(1) 起始标签包含元素的名称和可选属性。换句话说，元素的名称和属性都必须包含在起始标签中。结束标签以反斜杠开始，后接元素名称。例如：

```
<tag>元素主体</tag>
```

(2) 元素的属性包含属性名称和属性值两部分，中间通过等号连接，多个属性之间通过空格进行分隔。属性和元素名称之间也用空格进行分隔。例如：

```
<标签 属性1="v1" 属性2="v2" 属性3="v3" ... 属性n="vn">元素主体</标签>
```

(3) 少数元素的属性可能不包含属性值，仅包含属性名称。例如：

```
<标签 属性1 属性2 属性3 ... 属性n>元素主体</标签>
```

(4) 一般来说，属性值应该包含在引号内。虽然没有引号浏览器也能解析，但是初学者应养成良好的习惯。

(5) 属性是可选的，元素可以包含任意数量的属性，这主要取决于具体元素的定义。不同的元素会包含不同的属性。HTML为所有元素定义了公共属性，例如title、id、class和style等。

虽然大部分标签都是成对出现的，但是也有少数标签不是成对出现的，这些孤立的标签都被称为空标签。空标签仅包含起始标签，没有结束标签。例如：

```
<标签>
```

同样，空标签也可以包含多个属性，以标识特殊效果或功能。例如：

```
<标签 属性1="v1" 属性2="v2" 属性3="v3" ... 属性n="vn">
```

(6) 标签可以相互嵌套，形成文档结构。嵌套必须匹配，且不能交错，例如 是不合法的，而 或 是合法的。

(7) HTML文档的所有信息都必须包含在<html>标签中。所有元素的元信息都应该包含在<head>子标签中，而网页显示内容则应包含在<body>子标签内。

对于HTML文档而言，除了必须遵循基本的语法规范外，还必须保证文档结构信息的完整性。完整的文档结构如下：

```
<!DOCTYPE html >
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
    <title>文档标题</title>
</head>
<body></body>
</html>
```

HTML文档应包括以下内容(HTML文档的扩展名为.html或.htm，保存时必须正确使用扩展名，否则浏览器无法正确解析)。

1. 文档类型声明(DOCTYPE)

必须在首行定义文档的类型，过渡型文档可省略。HTML4和XHTML的DOCTYPE定义相对复杂，通常是一长串字符，示例如下：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4 /strict.dtd">
```

在编写HTML5规范时，WHATWG意识到了这一问题，将DOCTYPE简化为构成有效文档类型声明的最短字符序列，摒弃了HTML4中冗长的PUBLIC声明。使用这个声明会触发浏览器以标准模式显示页面。网页有多种显示模式，包括怪异模式、近标准模式和标准模式(其中标准模式也称为非怪异模式)。浏览器会根据DOCTYPE来识别应使用的显示模式以及验证页面的规则。

```
<!doctype html>
```

这一声明看起来非常简单，且易于记忆。它也是所有HTML5页面中的第一行代码。

2. <html>元素

<html>元素是整个HTML文档的包含元素，位于DOCTYPE声明之后。<html>标签应为文档命名设置空间，过渡型文档可省略。<html>元素可以包含以下几个属性：id、dir和lang。

3. <head>元素

<head>元素是所有其他头部元素的容器，它是紧跟在开始标签<html>之后的第一个标签。通常，<head>元素内都包含一个<title>元素，用以指定文档的标题。此外，它还可以包含以下元素的任意组合，且顺序不拘。

(1) <base>元素：用于为页面指定基础URL地址。通过这种设置，浏览器可以将相对地址与基础地址结合，生成完整的绝对地址。基础URL地址的值可以在<base>元素的href属性中进行设置。

(2) <link>元素：用于链接外部文件，例如CSS样式表。学习CSS时将详细介绍。

(3) <style>元素：用于在文档内包含CSS规则。学习CSS时将详细介绍。

(4) <script>元素：用于在文档内包含脚本。学习JavaScript时将详细介绍。

(5) <meta>元素：包含文档的相关信息，例如一段描述或作者姓名等。必须定义文档的

字符编码，通常使用<meta>标签在头部定义。常用字符编码包括中文简体(GB2312)、繁体中文(Big5)和通用字符编码(UTF-8)。

在HTML4中，可以使用<meta>元素指定文件中的字符编码，例如：

```
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
```

在HTML5中，可以使用charset属性来指定文件中的字符编码，例如：

```
<meta charset="utf-8">
```

目前，这两种方法都可以使用，但是在一个文档中只能使用一种形式，不能两种形式混合使用。

<head>开始标签可包含以下几个属性：id、dir和lang。

4. <title>元素

<title>元素用于定义文档的标题，通常作为<head>的子元素出现。标题主要通过以下方式呈现和使用。

(1) 在浏览器窗口的标题栏中显示。

(2) 在IE、Firefox、Chrome等浏览器中作为书签的默认名称。

(3) 搜索引擎使用其内容来帮助建立页面索引。

因此，标题必须能够准确描述网站的内容。检验标题好坏的标准是：访问者能否在不必查看网页实际内容的情况下，仅通过标题就能够理解他们将在页面中找到什么，以及标题是否包含搜索相关信息时常用的关键字。

需要注意的是，<title>元素应仅包含标题文本，不应包含任何其他元素。<title>元素可以包含以下属性：id、dir和lang。

5. 链接与样式表

<link>元素用于添加样式表。该元素可以使用href属性指向Web上的某个资源。需要注意的是，这里的href不是为了在单击链接时打开一个新的页面或网站，而是指向为当前页面提供样式信息的文件的位置。使用rel属性可以指明链接的文档是样式表，浏览器会根据这一信息进行相应的处理。

```
<link rel="stylesheet" href="css/main.css">
```

向页面添加脚本则更加简单。只需在页面中添加一个<script>元素，并使用src属性指向所需的JavaScript文件位置。

```
<script src="js/main.js"></script>
```

6. <body>元素

<body>元素位于<head>元素之后，它包含实际将在浏览器主窗口中显示的内容，通常被称为“主体内容”。

7. 注释

在HTML中，注释语句的格式为<!--注释内容-->。注释内容可插入到HTML代码的任何位置，并且不会显示在网页中。例如：

```
<!--单行注释 -->
```

8. 版本兼容

HTML5致力于与旧版本的语法兼容，主要体现在以下几个方面。

1) 可以省略标签的元素

可以省略标签的元素分为3种类型：不允许写结束标签的元素、可以省略结束标签的元素，以及可以省略全部标签的元素。

(1) 不允许写结束标签的元素是指不允许使用开始标签与结束标签将元素括起来的形式，只允许使用<元素/>的形式进行书写，包括area、base、br、col、command、endbed、hr、img、input、keygen、link、meta、param、source、track和wbr。

(2) 可以省略结束标签的元素是指元素可以省略结束标签，但开始标签必须保留，包括li、dt、dd、p、rt、rp、optgroup、option、colgroup、thead、tbody、tfoot、tr、td和th。

(3) 可以省略全部标签的元素是指开始标签与结束标签均可省略，浏览器会在解析时自动补全其结构，包括html、head、body、colgroup和tbody。

2) 具有布尔值的属性

对于具有布尔值的属性，如disabled和readonly等，若只写属性名，则表示属性值为true。如果想要将属性值设置为false，则可以不使用该属性。例如：

```
<!--只写属性，不写属性值，代表属性值为true-->
<input type="checkbox" checked>
<!--不写属性，代表属性为false-->
<input type="checkbox">
<!-- 属性值=属性名，代表属性为true -->
<input type="checkbox" checked="checked">
<!-- 属性值=空字符串，代表属性为true -->
<input type="checkbox" checked="">
```

3) 省略引号

当属性值不包括空字符串、<、>、=、单引号或双引号等字符时，属性值两边的引号可以省略。例如，以下写法是合法的：

```
<input type=text>
```

2.2　HTML5元素

教学视频

HTML5引入的新标签类型主要涵盖语义结构化、多媒体增强、表单优化及交互功能四大类。具体分类及代表性标签如表2-1所示。

表2-1　HTML5新增的标签类型

标签类型	说明
语义结构化标签	通过取代传统 标签提升文档逻辑性与 SEO(搜索引擎优化)表现，如<figure>、<main>、<aside>等
多媒体标签	原生支持音视频与图形渲染，减少插件依赖，如<canvas>、<svg>等
表单增强标签与类型	优化数据输入体验与验证机制，如<progress>、<datalist>等
内容交互功能标签	提升用户操作体验，如<time>、<mark>等

本章将举例介绍语义结构化标签和内容交互功能标签；多媒体标签将在第6章讲解，表单增强标签与类型将在第4章讲解。

2.2.1　语义结构化元素

HTML5 引入的语义结构化标签旨在通过明确的标签含义替代传统的 布局，例如 可以简化为<header>，这不仅简化了代码，还提升了文档逻辑性、SEO优化和可访问性(如屏幕阅读器支持)。这些标签定义了网页内容的层次结构，使搜索引擎和开发者能更直观地理解页面组织结构。表2-2系统总结了核心标签的功能、使用场景及注意事项。

表2-2　HTML5语义结构化标签

标签名称	功能描述	使用场景示例	说明
<header>	定义文档/区块的顶部区域	包含 LOGO、标题或主导航栏	可重复出现在不同区块顶部
<nav>	定义导航链接容器	主导航栏、侧边栏导航、页脚导航	一个页面可包含多个导航区块
<main>	定义文档主体内容	包裹页面核心内容区域	每个页面应唯一
<article>	定义独立完整的内容块	博客文章、新闻卡片、评论模块	可独立分发，也可嵌套使用
<section>	定义文档中的主题章节	内容分块(如章节、标签页)	需配合标题标签使用
<aside>	定义侧边辅助内容	相关链接、广告、引用内容	独立于主要内容
<footer>	定义文档/区块的底部	版权信息、联系方式、相关链接	可出现在页面或区块底部
<figure>	封装独立媒体内容	图片组、图表、代码示例	需配合<figcaption>使用
<figcaption>	为<figure>提供标题	图片说明、图表标注	必须嵌套在<figure>内

下面将对一些常用的元素进行介绍，并通过示例加以说明。

1. <article>元素

<article>元素表示文档、页面或应用程序中独立的、完整的内容块，这些内容可以单独被外部引用。除了内容部分，<article>元素通常有自己的标题(一般为<header>元素)，有时还有脚注(<footer>元素)。此外，<article>元素也可以嵌套使用。在嵌套使用时，内层的内容原则上需要与外层的内容紧密相关，即嵌套的内外层描述的应为独立但关联的事物。

2. <section>元素

<section>元素通常由标题和内容组成，主要用于对网站或应用程序中页面上的内容进行分块。<section>元素表示文档或应用的一部分。需要注意的是，<section>元素并不是容器元

素，因此不应使用CSS渲染。当一个容器需要直接定义样式或通过脚本控制行为时，应使用 元素。<section>元素的作用就是对页面内容进行分块，或将文章进行分段；而<article>元素则表示具有完整、独立内容的部分。因此，需要明确区分这两个元素的不同功能。

3. <header>元素

<header>元素是一种具有引导和导航作用的结构元素，通常用于放置整个页面或页面内内容区块的标题。此外，它还可以包含其他内容，如数据表格、搜索表单或相关的LOGO图片。因此，整个页面的标题都应该放在页面的开头。<header>元素定义文档或文档某一部分区域的页眉。在一个文档中，可以定义多个<header>元素。需要注意的是，<head>与<header>元素是不同的：<head>元素是HTML文档中所有头部元素的容器，而<header>元素是<body>元素的一个结构元素。虽然可以在<article>元素内使用<header>元素，但不能在<footer>、<address>或另一个<header>元素内使用<header>元素。

【例2-1】以下是根据语义化标签规范创建的完整HTML代码示例，包含文章主体和嵌套评论系统，严格遵循W3C标准并优化了时间标签。该案例实现了完整的文档结构：主文章使用<article>封装，评论区域通过<section>划分，每条评论作为独立<article>存在。样式部分进行了基础的排版优化，可直接扩展为实际项目模板(本例创建文件CH02-001.html)。

代码如下：

```
<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <meta charset="UTF-8">
    <title>语义化文章示例</title>
    <style>
    body { font-family: 'Microsoft YaHei', sans-serif; max-width: 800px; margin: 0 auto; padding: 20px; }
        article { margin-bottom: 30px; border-bottom: 1px solid #eee; padding-bottom: 20px; }
        section { background-color: #f9f9f9; padding: 15px; border-radius: 5px; }
        time { color: #666; font-size: 0.9em; }
    </style>
</head>
<body>
    <article>
        <header>
            <h1>HTML5语义化标签实践指南</h1>
            <p>发布日期：<time datetime="2025-07-28" pubdate>2025年7月28日</time></p>
        </header>

        <p>本文详细介绍如何正确使用HTML5语义化标签构建结构化文档，包括&lt;article&gt;、
        &lt;section&gt;等元素的嵌套规则和典型应用场景。</p>

        <section>
            <h2>用户评论</h2>
            <article>
```

```
            <header>
              <h3>评论者：技术达人</h3>
              <p><time datetime="2025-07-28T14:30:00+08:00">30分钟前</time></p>
            </header>
            <p>示例中对&lt;article&gt;的嵌套使用非常规范，特别是时间标签的ISO8601格式处
            理值得学习。</p>
          </article>
          <article>
            <header>
              <h3>评论者：前端新手</h3>
              <p><time datetime="2025-07-28T15:45:00+08:00">5分钟前</time></p>
            </header>
            <p>请问如何在移动端适配这种评论结构？建议补充响应式设计的实现方案。</p>
          </article>
        </section>

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

以上示例添加了读者的评论内容，整体内容比较独立且完整，因此使用了<article>元素。文章标题放在<header>元素中，而正文则位于<header>元素后面的<p>元素中，接着使用<section>元素对正文与评论部分进行区分。在<section>元素中嵌入评论内容，每条评论也是相对独立且完整的内容，因此每一条评论都使用一个<article>元素进行组织。在评论的<article>元素中，可以包含评论标题与评论内容，分别放在<header>元素和<p>元素中。运行例2-1的代码，效果如图2-2所示。

图2-2　例2-1网页效果

4. <footer>元素

<footer>元素可以作为内容块的脚注，例如在父级内容块中添加注释，或者在网页中添加版权信息等。脚注信息的形式可以包括作者介绍、相关阅读链接及版权信息等。<header>元素与<footer>元素的用法基本相同，前者位于区块的头部，后者位于区块的尾部。与<header>元素一样，一个网页也可以重复使用<footer>元素，并且还可以为<article>元素和<section>元素添加<footer>元素。

5. <nav>元素

<nav>元素用于定义页面导航的链接组，其中的导航元素链接到其他页面或当前页面的其他部分。一般情况下，只需要将主要的、基本的链接组放进<nav>元素中即可。例如，在页脚中，通常会有一组链接，包含服务条款、首页和版权声明等，此时使用<nav>元素来组

织并不适合，使用<footer>元素更为恰当。一个页面可以包含多个<nav>元素，以支持整体或不同部分的导航。一般来说，<nav>元素适用于以下场景：传统导航条、侧边栏导航条、页内导航及翻页操作。

6. <aside>元素

<aside>元素表示页面与其他内容关联性不强或者没有直接关联的内容，通常用于显示一些附属信息。<aside>元素常见的使用场景包括在侧边栏中展示目录、索引、术语表等；也可以用来展示相关的广告宣传、作者介绍、Web应用、相关链接以及当前页面的内容简介等。

【例2-2】使用<nav>创建简易导航、<aside>展示辅助内容、<footer>定义页脚，文件保存为CH02-002.html。

代码如下：

```
<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>企业门户网站</title>
</head>
<body>
    <header>
        <h1>企业门户网站</h1>
    </header>
    <section>

        <nav>
            <a href="index.html">首页</a> |
            <a href="about.html">关于我们</a> |
            <a href="news.html">新闻动态</a> |
            <a href="products.html">产品中心</a> |
            <a href="contact.html">联系我们</a>
        </nav>

</section>
    <main>
        <article>
            <h2>企业最新动态</h2>
            <p>我们近期发布了新一代智能产品系列...</p>
        </article>
    </main>

    <aside>
        <h3>快速链接</h3>
        <ul>
```

```
        <li><a href="#">人才招聘</a></li>
        <li><a href="#">投资者关系</a></li>
        <li><a href="#">企业社会责任</a></li>
    </ul>
</aside>

<footer>
    <ul>
        <li>友情链接: <a href="#">合作伙伴</a></li>
        <li>© 2025 企业名称 版权所有</li>
    </ul>
</footer>

</body>
</html>
```

这个示例包含以下几个部分。

(1) header区域：展示网站主标题。

(2) navigation区域：使用<nav>创建主导航菜单，包含5个主要栏目链接。

(3) main内容区：使用<article>展示企业最新动态。

(4) aside侧边栏：提供快速链接(如人才招聘、投资者关系等)。

(5) footer页脚：包含友情链接和版权信息。

运行例2-2的代码，效果如图2-3所示。

图2-3　例2-2网页效果

2.2.2　内容交互元素

根据HTML5规范，内容交互标签主要用于增强用户与网页内容的动态交互体验。表2-3所示为主要标签的详细分类。

表2-3　HTML5内容交互标签

标签名称	功能描述	核心属性	交互特性
<details>+<summary>	创建可折叠内容区域，默认隐藏详细信息	open(设置默认展开)	单击标题切换内容显隐，支持键盘操作
<progress>	可视化任务进度条(如文件上传)	value, max	动态展示进度状态
<meter>	显示标量测量值(如磁盘用量、评分)	min, max, value	根据值域自动着色(低、中、高)
<datalist>	为输入框提供下拉选项列表(需配合<input>)	id(与input的list绑定)	输入时自动匹配提示项

(续表)

标签名称	功能描述	核心属性	交互特性
<output>	实时显示计算或表单处理结果	for(关联输入元素)	动态更新计算结果
<time>	定义机器可读的时间	发布日期、事件时间戳	支持datetime属性标准化时间
<mark>	突出显示文本	关键词高亮、搜索结果标记	默认黄色背景

【例2-3】以下是一个基于HTML5原生内容交互标签的综合案例，展示如何结合多个标签实现动态用户界面。该案例模拟了一个“订单跟踪面板”，包含折叠详情、时间标注、进度反馈、输入建议等交互功能。所有交互均无须使用JavaScript即可正常工作(创建文件CH02-003.html)。

代码如下：

```
<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <meta charset="UTF-8">
    <title>订单跟踪面板</title>
</head>
<body>
    <h2>我的订单 #12345</h2>
    <!-- 使用 details/summary 折叠订单详情 -->
    <details>
        <summary>查看订单详情</summary>
        <p>创建时间：<time datetime="2025-07-29T14:30:00">2025年7月29日</time></p>
        <p>状态：<mark>已发货</mark></p>
        <p>预计送达：<time datetime="2025-08-02">2025年8月2日</time></p>
    </details>
    <!-- 使用 progress 显示发货进度 -->
    <p>发货进度： <progress value="75" max="100"></progress> 75%</p>
    <!-- 使用 datalist 提供反馈建议 -->
    <label for="feedback">反馈问题：</label>
    <input type="text" id="feedback" list="suggestions">
    <datalist id="suggestions">
        <option value="包裹未收到">
        <option value="商品损坏">
        <option value="送货延迟">
        <option value="其他问题">
    </datalist>
</body>
</html>
```

运行以上代码，效果如图2-4所示。

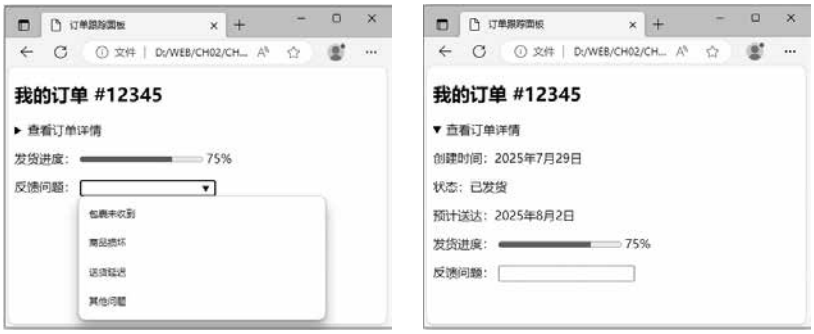

图2-4　例2-3网页效果

例2-3包含以下几个交互功能。

(1) 折叠详情区域：单击“查看订单详情”标题时，面板会自动展开或收起订单信息(如时间和状态)。默认情况下，隐藏详情区域，以减少页面杂乱，提升用户体验。

(2) 动态进度反馈：<progress>标签以进度条形式直观显示发货状态值(如75%)。进度条自动更新外观，无须手动调整样式。

(3) 智能输入建议：在反馈输入框中输入时，<datalist>提供预定义选项(如“包裹未收到”)以实现自动补全，增强表单填写效率，减少用户输入错误。

(4) 语义化标记：<time> 标签确保时间数据可被搜索引擎和辅助设备解析。而<mark>高亮显示“已发货”状态，引起用户注意。

【例2-3】的案例优势如下。

(1) 纯原生实现：所有交互都依赖HTML5内置功能，无须使用JavaScript。

(2) 无障碍友好：标签自带键盘导航支持(如Tab键切换焦点)。

(3) SEO优化：语义化标签(如<time>)提升了页面的可读性。

2.2.3　HTML5废除的元素

HTML5中废除了HTML4中的一些元素，主要原因如下。

(1) 样式与结构分离：表现型元素由CSS接管。

(2) 语义化增强：模糊标签由更精准的语义标签替代。

(3) 安全与性能：某些框架类元素易被恶意利用。

(4) 标准化需求：淘汰非通用或兼容性差的元素。

完整的替代方案可参考HTML5规范文档。需要注意的是，部分旧浏览器仍支持废除的元素，但新项目应严格遵循标准。HTML5主要废除了以下元素，具体分为4类。

1. 能用CSS替代的表现性元素

HTML4中的一些表现文本效果的元素，如<basefont>、<big>、 、<font>、<s>、<strike>、<tt>和<u>已被废除。这些样式控制元素可以通过CSS实现相同效果，如<s>/<strike>可以用<del>替代，<tt>则可以用CSS的font-family: monospace替代。

2. 框架类元素

<frame>、<frameset>和<noframes>因可用性差且存在安全风险被废除，仅保留<iframe>作为替代方案。

3. 浏览器支持度低的元素

对于只有部分浏览器(如早期IE)支持且未被广泛采纳的元素，HTML5已将其废除，并提供了相应的替代方案。

(1) <applet>(Java小程序)由<object>或<embed>替代。
(2) <bgsound>(背景音乐)由<audio>替代。
(3) <blink>(闪烁文本)无直接替代，可用CSS动画模拟。
(4) <marquee>(滚动文本)可用JavaScript实现。

4. 语义模糊或冗余元素

部分元素因语义不清晰或功能重叠，在HTML5中被更明确的元素所替代。

(1) <acronym>(首字母缩写)由<abbr>替代。
(2) <dir>(目录列表)由<ul>替代。
(3) <isindex>(搜索输入)由<form>+<input>替代。
(4) <listing>和<xmp>(代码块)由<pre>或<code>替代。
(5) <nextid>和<plaintext>技术已过时，无直接替代。

2.3　HTML5属性

HTML5除了新增和废除一些元素外，还新增了一些元素属性并废除了HTML4中的一些元素属性。

2.3.1　HTML属性

属性为HTML元素提供了更多附加信息。HTML属性通常以名称/值对的形式出现在开始标签中。例如<style>标签的type属性指定其内容为CSS：

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

有些属性则只包含一个名称，如required或checked属性。这些属性被称作“布尔属性”。在一个标签中如果只出现一个布尔属性的名称而没有值，则表示该布尔属性的值为true。因此，以下两种写法完全等价：

```
<input type="text" required >
<input type="text" required="true">
```

1. 核心全局属性

在HTML中，几乎所有元素都支持以下4个核心全局属性(它们是构建网页内容的基础)：id、title、class和style。

1) id属性

id属性用于唯一标识页面中的某个元素，可用于CSS样式或JavaScript代码的精准操作，也可作为页面内跳转的锚点目标。