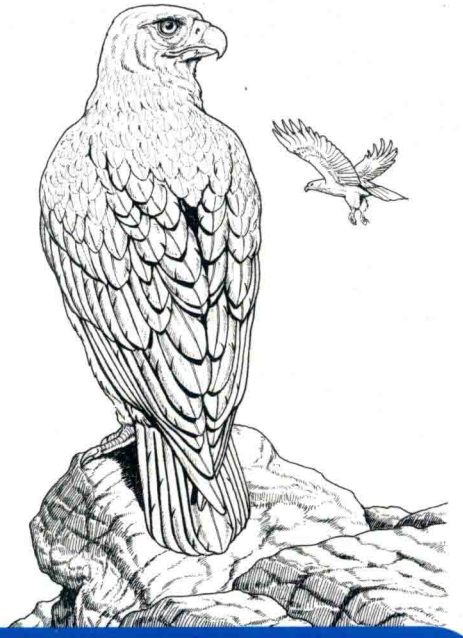
工业和信息化“十三五”
人才培养规划教材



JavaScript
前端开发|模块化教程

JavaScript Development Modular Tutorial

赵建保 ◎ 主编
刘琳 贺辉平 邱尚明 彭华 ◎ 副主编

模块化的写作手法，突出**实践动手能力**，更符合学习要求

突出**前端工程师**职业能力培养

融入 **jQuery UI** 和 **Bootstrap UI** 组件开发模式

直戳**前端组件开发**核心技术

成果导向理念、**任务驱动**教学

中国工信出版集团
人民邮电出版社
POSTS & TELECOM PRESS

工业和信息化“十三五”
人才培养规划教材

JavaScript 前端开发模块化教程

JavaScript Development Modular Tutorial

赵建保 ◎ 主编
刘琳 贺辉平 邱尚明 彭华 ◎ 副主编

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

JavaScript前端开发模块化教程 / 赵建保主编. -- 北京 : 人民邮电出版社, 2019.2
工业和信息化“十三五”人才培养规划教材
ISBN 978-7-115-49916-5

Ⅰ. ①J… Ⅱ. ①赵… Ⅲ. ①JAVA语言－程序设计－高等学校－教材 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2018)第253191号

内 容 提 要

JavaScript 是前端组件开发的核心技术，前端组件是构建 Web 应用界面的基础。本书以前端工程师岗位工作任务为起点，循序渐进，以 HTML5、CSS3、JavaScript 为技术支撑，以 Visual Studio Code 为开发环境，将网页前端开发过程的任务系统化、规范化和组件化。全书设计开发了轮播图、折叠面板、选项卡、相册、滚动监听、表单验证、银行客服电话查询、微信运动步数统计图等 23 个工作任务，包含了 JavaScript 核心技术、DOM、Canvas、JSON、Ajax 和性能优化等主题，覆盖了 Web 项目开发中的常用组件的 HTML 结构、CSS 样式和 JavaScript 交互行为技术。同时注重梳理相关联的知识，提炼设计模式，剖析组件的工作机制，以支持学习者更好地理解和运用相关技术适应前端组件项目开发的需求。

本书主要面向高等院校的数字媒体技术、软件技术、计算机应用技术、教育技术等专业的学生，可作为“JavaScript 程序设计”“HTML5 Web 开发”“Web 前端开发”等课程的教材和参考书，也可作为培训机构 Web 技术培训教材。

◆ 主　　编　赵建保
副 主 编　刘　琳　贺辉平　邱尚明　彭　华
责任编辑　范博涛
责任印制　马振武

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京市艺辉印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：22.25　　2019 年 2 月第 1 版
字数：550 千字　　2019 年 2 月北京第 1 次印刷

定价：59.80 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广登字 20170147 号

前言 FOREWORD

在作者十余年的数字媒体应用技术专业建设和 JavaScript 课程教学实践中，每逢到网上书店或者实体书店选定教材，面对琳琅满目的 JavaScript 图书时我便油然而生感激和期待。感激有这么多技术大伽、一线教师、从业人员深耕技术，并不辞劳苦地分享各自的研究、开发实践和教学心得。然而已经出版的 JavaScript 图书多以技术为主导，又或者以某个项目为主导，多侧重于介绍孤立的知识和技术，没能融入前端开发工作过程中，由此激发了我编著此书的动力和决心。

我的初心是，以学习者胜任前端工程师岗位工作任务为目标，循序渐进，掌握 HTML5、CSS3、JavaScript 等技术的综合应用，熟悉 JavaScript 开发环境，适应网页前端开发工作的规范化、模块化和组件化岗位要求。我的思路是，以任务导入激发学习者的学习积极性，以工作过程必需的任务成果和目标维持学习者的学习动力和压力，以核心知识帮助学习者构建自己新的认知结构，任务实施过程划分为编写 HTML、编写 CSS、编写 JavaScript 和浏览器测试四大步骤，以强化训练提升学习者的知识和能力的迁移能力。期待这样的设计更有利于学习者提升 JavaScript 前端开发能力，更有利于学习者掌握 JavaScript 基本知识、基本技能、基本思想和基本开发经验，更有利于学习者提高分析问题、解决问题、发现问题和提出问题的能力，以及演绎思维和归纳思维的能力。

全书由赵建保负责整体设计、编写和统稿，刘琳、邱尚明、贺辉平、彭华参与教材设计与编写，参与编写的人员还有沈佳乐、曹盼盼、赖成发、胡铭畅、罗妩媚、石燕平、严梅燕、丁伟杰、蔡嘉威、沈泳銮、陈响钦、邓金霞、朱进旺、吕晓纯、陆洁莹、李丽彤、邓洁美、谢涌彬、苏新栋。本书得到了广东省高职教育品牌专业项目的支持，企业一线前端开发人员黄亮、邱景生、杨起捷等参与了项目任务设计并给予了全程指导，还提出了很多宝贵建议，在此表示衷心的感谢。

由于 IT 行业发展迅猛，作者项目实践和知识视野有限，经验不够丰富，书中不足之处在所难免，恳请读者提出宝贵的意见和建议。联系方式（E-mail）：mpcer@163.com。

编者

2018 年 10 月

目 录 CONTENTS

此为试读，需要完整PDF请访问：www.ertongbook.com

此为试读，需要完整PDF请访问：www.ertongbook.com

Chapter 1

任务 1
搭建 JavaScript 开发环境

JavaScript

1.1 任务导入

在 Web 标准中，HTML 定义了页面结构和内容，CSS 定义了页面布局和外观，如颜色、字体、边框、边距和版式布局等，而 JavaScript 定义了页面交互行为，比如元素交互、表单验证、网页游戏等。如今，缺乏动态和交互性的网站已经没有市场，网站必须以新颖的方式与用户互动，页面交互性和动态性已经是商业 Web 应用开发中前端开发的必备能力。随着浏览器 JavaScript 引擎性能的提升，未来在 Web 图像、音频及视频处理、虚拟现实、游戏开发等方面前途无量。要想成为 Web 开发工程师，掌握 JavaScript 必不可少。

正式开始学习 JavaScript 前端开发之前，需要先配置满足 Web 前端开发和测试的学习环境，主要是 JavaScript 编辑器和 Web 浏览器，其中 JavaScript 编辑器用于编写 HTML、CSS 和 JavaScript 前端代码，Web 浏览器用于 Web 应用的开发测试。通过本任务，您将学会如何配置 JavaScript 前端编码和测试环境，并体验鼠标 mouseover、mouseout 事件时表格行背景变色效果的开发，图 1-1 所示为表格行悬停提示效果，从此开启您的前端征程。

序号	学号	姓名	手机电话	QQ号码
1	20155531 01	前端工程师 1	13566990001	4498465741
2	20155531 02	前端工程师 2	13566990002	4498465742
3	20155531 03	前端工程师 3	13566990003	4498465743
4	20155531 04	前端工程师 4	13566990004	4498465744
5	20155531 05	前端工程师 5	13566990005	4498465745

图1-1 表格行悬停提示效果

1.2 成果目标

本任务旨在学会搭建前端开发环境，熟悉前端页面开发的基本操作流程，熟悉 Visual Studio Coder 的使用，培养前端开发的意识和兴趣。

知识目标	技能目标	素质目标
1. 了解 JavaScript 发展历程 2. 了解 Web 页面渲染过程 3. 了解 Visual Studio Code 4. 理解 EMMET 语法 5. 理清 DOM 对象选取模式 6. 理解 addEventListener 参数	1. 安装和配置 Visual Studio Code 2. 使用 Chrome 开发者工具 3. 安装与配置 http-server 4. 前端页面开发流程 5. 合理确定 JavaScript 在 HTML 文档中的位置 6. 善用 addEventListener 7. 善用 style 对象操作样式	1. 能梳理 JavaScript 演进 2. 感受前端开发过程 3. 培养前端开发的思维习惯

1.3 核心知识

1.3.1 JavaScript 演进

理解 JavaScript 的最好方法之一，就是了解 JavaScript 的历史。

JavaScript 为互联网而生，紧随着浏览器的出现而问世。回顾它的历史，就要从浏览器的历史说起。1990 年 12 月 25 日，英国计算机科学家伯纳·李爵士（Tim Berners-Lee）发明了万维网（World Wide Web）。1993 年，美国国家超级计算机应用中心（NCSA）开始开发一个独立的浏览器 Mosaic。1994 年 10 月，Mosaic 通信公司（Mosaic Communications）成立，不久后改名为 Netscape。1994 年 12 月，Navigator 发布了 1.0 版，市场份额一举超过 90%。Netscape 同微软公司在浏览器市场的竞争异常激烈，争相给浏览器添加新功能，获取竞争优势。1995 年 2 月，Netscape 公司发布 Netscape Navigator 2 浏览器，该公司的布兰登·艾奇（Brendan Eich）为 Navigator 2 浏览器开发了 LiveScript 脚本语言，主要目的是处理表单数据验证，避免由服务器端验证导致的延时问题。LiveScript 语法借鉴 Java，函数借鉴 Scheme，原型继承借鉴 Self，正则表达式特性则借鉴 Perl。当时正逢 Sun 公司的股票飞涨，为搭上 Java 成功的顺风车，Netscape 将 LiveScript 改名为 JavaScript。由于 JavaScipt 1.0 获得了巨大成功，Netscape 公司随即在 Netscape Navigator 3 中又发布了 JavaScript 1.1 版本。

在 Netscape Navigator 3 发布后不久，微软不甘示弱，紧跟着 Netscape 在 Internet Explorer 3 中加入 JavaScript 脚本语言，为了避免与 Netscape 的 JavaScript 产生纠纷，微软特意将其命名为 JScript，这种语言和 JavaScript 很像，浏览器大战就此爆发。自微软在操作系统中内置 Internet Explorer 3，Netscape 就面临着即将丧失浏览器脚本语言主导权的局面。1996 年 11 月，网景公司决定将 JavaScript 提交给国际标准化组织 ECMA，希望 JavaScript 能够成为国际标准，以此抵抗微软。1997 年，欧洲计算机制造商协会（ECMA）以 JavaScript 1.1 为蓝本制订了 ECMA-262 新脚本语言的标准，并命名为 ECMAScript，这个版本就是 ECMAScript 1.0 版，它规定了浏览器脚本语言的标准。之所以不叫 JavaScript，一方面是由于商标的关系，Java 是 Sun 公司的商标，根据一份授权协议，只有 Netscape 公司可以合法地使用 JavaScript 这个名字，且 JavaScript 已经被 Netscape 公司注册为商标；另一方面也是想体现这门语言的制定者是 ECMA，不是 Netscape，这样有利于保证这门语言的开放性和中立性。因此，ECMAScript 和 JavaScript 的关系是，前者是后者的规格，后者是前者的一种实现。1998 年 6 月，ECMAScript 2.0 版发布。1999 年 12 月，ECMAScript 3.0 版发布，成为 JavaScript 的通行标准，得到了广泛支持。2007 年 10 月，ECMAScript 4.0 版草案发布，对 3.0 版做了大幅升级，由于 4.0 版的目标过于激进，各方对于是否通过这个标准发生了严重分歧。以 Yahoo、Microsoft、Google 为首的大公司，反对 JavaScript 的大幅升级，主张小幅改动；以 JavaScript 创造者 Brendan Eich 为首的 Mozilla 公司，则坚持当前的草案。2008 年 7 月，由于对于下一个版本应该包括哪些功能，各方分歧太大，争论过于激进，ECMA 开会决定，中止 ECMAScript 4.0 的开发，将其中涉及现有功能改善的一小部分发布为 ECMAScript 3.1，而将其他激进的设想扩大范围，放入以后的版本，由于会议的气氛，该版本的项目代号起名为 Harmony（和谐）。2009 年 12 月，ECMAScript 5.0 版正式发布。Harmony 项目则一分为二，一些较为可行的设想定名为 JavaScript.next 继续开发，后来

演变成 ECMAScript 6，2015 年 6 月，ECMAScript 6 发布了正式版本，并更名为 ECMAScript 2015。如图 1-2 所示为 JavaScript 演进路线。

图1-2 JavaScript演进路线

随着 JavaScript 再次成为关注的焦点，很多杰出的编程人员致力于改善 JavaScript 解释器，极大地改善了其运行阶段的性能。随着 Gmail、GoogleMaps 这一类客户端应用和 Ajax 的相继出现，大大促进了 JavaScript 开发模式的变革，也把 JavaScript 催生成为一种成熟的、某些方面独一无二的、拥有强大原型体系的面向对象语言。

1.3.2 JavaScript 介绍

程序设计语言分为解释型和编译型两大类。Java 和 C++等语言需要一个编译器（Compiler）。编码器是一种程序，能够把用 Java 等高级语言编写出来的源代码翻译为直接在计算机上执行的文件，代码错误在编译阶段就能被发现。解释型程序设计语言不需要编译器，运行仅需要解释器。JavaScript 语言则由 Web 浏览器负责完成有关的解释和执行工作。JavaScript 是 Web 页面中的一种脚本编程语言，也是一种通用的、跨平台的、基于对象和事件驱动并具有安全性能的解释型脚本语言，不但可用于编写客户端的脚本程序，由 Web 浏览器解释执行，还可以编写在服务器端执行的脚本程序，在服务器端处理用户提交的信息并动态地向客户端浏览器返回处理结果。浏览器中的 JavaScript 解释器将直接读入源代码并执行，代码错误只能等到解释器执行到有关代码时才能被发现。如果浏览器中没有解释器，JavaScript 代码就无法执行。

JavaScript 脚本语言的主要特点如下。

（1）解释性。不同于一些编译性程序语言（如 C、C++），JavaScript 源代码不需要经过编译，而是直接嵌入在 HTML 页面中，使得前端页面支持用户交互并响应相应事件，在浏览器中运行时被解释。

（2）基于对象。许多功能运用脚本环境对象的方法与脚本的相互作用来完成。

（3）事件驱动。JavaScript 可以直接对用户页面的操作行为（鼠标、键盘、手势等）做出响应，无须经过 Web 服务器处理。

（4）跨平台。JavaScript 依赖于浏览器本身，与操作环境无关，只要计算机能运行浏览器并支持 JavaScript 的浏览器，就可以正确执行。

（5）安全性。JavaScript 是一种安全性语言，不允许访问本地硬盘，不能将数据存入服务器上，不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互，以防止数据丢失和篡改。

JavaScript 由一项主要适用于浏览器客户端的计算机技术，从一个简单的输入验证器逐渐发展成为一种多功能的强大的程序设计语言，甚至连服务端也能用它来编写，能够处理复杂的计算

和交互，拥有闭包、匿名函数等，甚至元编程特性。现在已经具备了与浏览器窗口及其内容等几乎所有方面交互的能力。在互联网发展的早期，JavaScript 就已经成为了支撑网页内容交互体验的基础技术，现在 JavaScript 毫无疑问已经成为了 Web 的核心技术，能够地理定位、播放视频和音频、绘图等，速度更快，性能更强。

JavaScript 是世界上最流利的编程语言之一。使用 JavaScript 不仅可以开发浏览器应用程序，也可以开发手机应用（APP）和服务器端程序。JavaScript 可实现以下功能。

（1）创建拥有强大而丰富功能的 Web 应用程序，多数运行于 Web 浏览器，基于 HTML5 可开发应用缓存、本地存储、本地数据库等 Web 应用。

（2）使用 Node.js 编写服务器端脚本。

（3）开发移动设备应用程序。比如基于 HTML5+的移动 APP 开发。

1.3.3　Web 页面渲染过程

Web 技术的根基是 HTML、CSS 和 JavaScript，其中 HTML 控制内容和结构，CSS 控制表现，而 JavaScript 则控制交互行为。换句话说，JavaScript 是让 HTML 和 CSS 协同运作的黏合剂。一个完整的 JavaScript 实现应该包括以下三个不同部分：核心（ECMAScript）、文档对象模型（DOM）和浏览器对象模型（BOM），如图 1-3 所示。

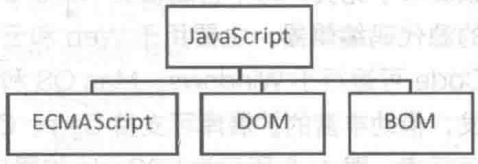


图1-3　JavaScript实现

ECMAScript 规定了 JavaScript 的语法、类型、语句、关键字、保留字、操作符和对象，它是语言的核心部分，包括变量、函数、循环等，独立于浏览器，并可以在其他环境中使用。文档对象模型（DOM）是用于操作 HTML 和 XML 文档的应用程序接口（Application Programming Interface，API），提供了一种与 HTML、XML 文档交互的方式。在浏览器中，主要用来操作 HTML 文档。HTML 和 DOM 把整个页面映射为一个多层节点结构。HTML 或 XML 页面中的每个组成部分都是某种类型的节点，这些节点又包含着不同类型的数据。DOM1 级（DOM Level1）于 1998 年 10 月成为 W3C 推荐标准，包括 DOM 核心（DOM Core）和 DOM HTML。DOM 核心规定如何映射基于 XML 的文档结构，以便简化对文档中任意部分的访问和操作。DOM HTML 模块添加了针对 HTML 的对象和方法。DOM2 包括 DOM 视图、DOM 事件、DOM 样式、DOM 遍历和范围。浏览器对象模型（BOM）实际上是一个与浏览器环境有关的对象集合，使开发人员可以控制浏览器显示的页面以外的部分，BOM 只处理浏览器窗口和框架。

Web 页面渲染过程如图 1-4 所示，图中实线表示先后关系，虚线表示渲染过程可能调用的模块。如页面下载时需要使用网络和存储，计算布局和绘图时需要使用 2D/3D 图形模块生成可视化结果。具体渲染过程是：网页内容输入到 HTML 解释器，HTML 解释器在解析后构建 DOM 树，如果遇到 JavaScript 代码则交给 JavaScript 引擎去处理，如果网页包含 CSS 则交给 CSS 解释器去解释，没有被定义 CSS 的元素渲染引擎将默认样式应用到 HTML 元素上。当 DOM 建立的时候，渲染引擎接收来自 CSS 解释器的样式信息，构建一个新的内部绘图模型。该模型由布局模块计算模型内部各元素的位置和大小信息，最后由绘图模块完成从模型到图像的绘制。

C、C++、Java 等典型的编程语言，执行代码前必须先编译，而 JavaScript 只需直接在网页中编写代码，再在浏览中加载网页，浏览器 JavaScript 引擎就能直接执行代码。运行时，JavaScript 不会修改服务器端的 HTML 文件，而是操控浏览器基于 HTML 文件构建的 DOM 树。

此为试读，需要完整PDF请访问：www.ertongbook.com