

JavaScript

物联网硬件编程

面向Web开发人员的硬件开发

Circuitry and wires
are stashed inside,
with a vent for our
temperature sensor.

[美]丽萨·丹吉·加德纳(Lyza Danger Gardner) 著
戴礼晋 谭少辉 许琛 译

Ridiculous Feet:
because we can!

JavaScript

物联网硬件编程

[美] 丽萨·丹吉·加德纳(Lyza Danger Gardner) 著
戴礼晋 谭少辉 许琛 译

清华大学出版社
北京

Lyza Danger Gardner

JavaScript on Things: Hacking hardware for Web developers

EISBN: 978-1-61729-496-9

Original English language edition published by Manning Publications, USA (c) 2018 by Manning Publications. Simplified Chinese-language edition copyright (c) 2019 by Tsinghua University Press Limited. All rights reserved.

北京市版权局著作权合同登记号 图字: 01-2018-3784

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13701121933

图书在版编目(CIP)数据

JavaScript物联网硬件编程 / (美)丽萨·丹吉·加德纳(Lyza Danger Gardner) 著; 戴礼晋, 谭少辉, 许琛 译. —北京: 清华大学出版社, 2019

书名原文: JavaScript on Things: Hacking hardware for Web developers

ISBN 978-7-302-53109-8

I. ①J… II. ①丽… ②戴… ③谭… ④许… III. ①JAVA 语言—程序设计 IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2019)第 101124 号

责任编辑: 王军

装帧设计: 孔祥峰

责任校对: 牛艳敏

责任印制:

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮 编: 100084

社 总 机: 010-62770175 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 刷 者:

装 订 者:

经 销: 全国新华书店

开 本: 170mm×240mm 印 张: 28.25 字 数: 569 千字

版 次: 2019 年 8 月第 1 版 印 次: 2019 年 8 月第 1 次印刷

定 价: 98.00 元

产品编号:

译 者 序

中国的互联网发展已经走过了三个阶段(信息、商品和服务)，正在迈向第四个阶段(万物互联)。物联网(Internet of Things)是通过传统的电信网络、互联网等信息载体，使用中心计算机对机器、设备和人员进行集中管理、控制，对家庭设备、汽车进行遥控；同时透过搜集这些零散数据，最后汇集成大数据，让所有能行使独立功能的普通物体实现互联互通的网络。

如果你对电子开发的印象还停留在 10 年前，你可能会想到五花八门的单片机选型、烦琐的电路板焊接和连线。事实上，近几年来各种集成电路产品的流行，已大大降低了硬件开发的门槛。其中以 Arduino 和树莓派(Raspberry Pi)为主要代表。Arduino 是一款便捷灵活、方便上手的开源电子原型平台，Arduino 的硬件原理图、电路图、IDE 软件及核心库文件都是开源的，在开源协议范围内可以任意修改原始设计及相应代码。越来越多的软件开发者使用 Arduino 进入硬件、物联网等开发领域。在大学里，自动化专业、计算机软件专业，甚至艺术专业，也纷纷开设了 Arduino 相关课程。

而与 Arduino 不同，Raspberry Pi 是一款超小体积的计算机，利用 ARM 作为处理器，还具有板载内存、HDMI 接口、USB 接口、音频接口和网络接口等硬件。Raspberry Pi 具有板载网络连接功能，完成设置后可以通过计算机远程访问实现对 Raspberry Pi 的控制以及程序设计。

如今，Arduino 和 Raspberry Pi 已在国内拥有大批拥趸，网络上已有相当多的实践分享文章。好消息是 Web 开发者利用已掌握的 JavaScript 技能就能轻松地踏入硬件开发的大门。开源 Node.js 框架 Johnny-Five 的 API 提供了许多直观的元件类，你可以使用它们快速构建小工具和发明原型。JavaScript 是世界上最流行的编程语言之一，是网络的事实语言。Stack OverFlow 的联合创始人 Jeff Atwood 曾提出著名的“Atwood 定律”：任何能够用 JavaScript 实现的应用，最终都必将用 JavaScript 实现。

本书假定你对电子电路基本不了解或者知之甚少，会引导你重新回顾电子学的基础知识——包括欧姆定律、电感原理和嵌入式系统等。阅读本书，你将了解各种电路的硬件组件、电子元件以及如何编写代码，并且轻松地上手各种实验。之后你将了解各种提供硬件抽象的 JavaScript IoT 应用开发平台，例如，支持 Node.js 的 Tessel 2 开发板、Espruino Pico 等受限平台以及 Raspberry Pi 这样的单板机。当 JavaScript 和 node.js 进入电子世界时，真正的乐趣就开始了。

本书全部内容由戢礼晋、谭少辉、许琛合作翻译完成，为了完美地翻译此书，做到“信、达、雅”，我们在翻译过程中查阅了大量的中英文资料。当然，由于译者水平和精力有限，翻译中的错误和不当之处在所难免，我们非常希望得到读者的积极反馈，以利于本书后续的更正和改进。

感谢本书的作者，于字里行间都能感受到作者的职业精神和专业素养；感谢清华大学出版社给了我们翻译此书的机会。感谢编辑老师在翻译过程中给予的帮助，她们对于译文中的各种错误和问题给予的反馈和指正使我们受益终生。

我们希望本书能对那些想了解、学习和动手实践硬件开发的人们有所帮助，希望能够吸引更多的开发者加入 JavaScript 硬件开发，同时为社区和开源项目做出贡献。

译者

序 言

在 2013 年夏末的一天，我有幸站在了英格兰布莱切利公园的一个大帐篷里的舞台上。就是在这个地方，第二次世界大战时英国的密码破解者(包括著名的阿兰·图灵)破译了恩尼格码密码。因为在这里发生的一些奇妙的事情，当天可以说是我生命中最美好的日子之一。

首先，我在程序开发竞赛中获奖(这就是我在舞台上的原因)。位于布莱切利公园的国家计算机博物馆(National Museum of Computing)正在寻求技术帮助，以创建基于网络的互动时间线展览。我熬夜扩展了一个开源的 JavaScript 库并构建了一个原型，也就是我的参赛作品。让我非常高兴的是，这个作品在展会上被评为最佳作品。其次，我还获得了奖品，而且这个奖品非同一般。

我收到的奖品是一个入门级 Arduino Uno 套件——包含一块 Arduino 开发板、一组电子元件和一本教学书。这个套件极大地改变了我的生活。我后来发现，将我新学到的电子技能与我每天做的事情结合——编写开源的、基于标准的网站和应用程序，给我带来了我经历过的最迷人的炼金术之一：JavaScript 硬件编程。也就是说，我可以使用我熟悉的 JavaScript 来增强电子和物联网的开发能力。

然而，这一切是后来才发生的。最开始，我学会了通过工具包所附带的教学书中的示例来构建简单的电子电路，后来如饥似渴地在网络上搜索以了解更多信息。我学会了通过为 Arduino Uno 的微控制器编程对这些电路应用逻辑控制，用 Arduino 的类似 C/C++ 的语言编写简单的草图(程序)，并优化了开发板非常有限的程序空间和内存。

之后在 2013 年底，我发现了 Johnny-Five。这款开源 Node.js 框架当时诞生不久，但已经非常强大。我不必再编写底层的、受约束的 Arduino 代码，相反，可以编写更高级的 JavaScript 程序来控制我的 Uno。

JavaScript 和微控制器的结合只是一个小把戏，仅仅是为了让 JavaScript 覆盖整个已知领域。起初，我作为一个 Node.js 拥趸，对此也持怀疑态度：也许这是毫无意义的，也许它永远不会落地生根。

别担心，它并不是毫无意义，而且还会发展壮大。

将 JavaScript 与硬件编程结合简化了我的开发体验，使小项目的原型设计变得更快捷。作为 Web 开发人员，我可以使用更熟悉的开发工作流程，不必关心底层内存和资源优化。Johnny-Five 将行为封装到高级元件类中非常直观：生成的代码比许多

Arduino 库更清晰、更易用。它让我可以通过 npm 使用 Node.js 几乎深不可测的生态系统。我可以像使用任何其他 Node.js 脚本一样，简单地导入模块，非常便利。

我想说清楚的是：Arduino 或者更“传统的”基于 C 语言的微控制器编程本身没有任何问题。例如，我们有很好的理由关心内存管理，尤其是在编写固件或者制作生产设备时。而 Arduino 则是一个奇迹：它的全部存在理由是让新手可以使用嵌入式电子产品。从头开始使用 Arduino 和 Arduino 编程语言是非常合理且可以驾驭的方法。

但 JavaScript 可以帮助 Web 开发者更快地掌握电子开发技术。首先，Arduino(和其他平台)的介绍材料通常假定你不掌握任何编程知识，也就是你在文中会看到对什么是数组以及循环如何工作的解释。当你只是学习如何让它们正常工作时，微控制器的挑剔限制和细节可能会分散你的注意力。IDE 会很笨重。某些情况下，你最终可能会花费大量时间来配置内容，而没有太多时间花在真正要做的事情上。JavaScript 可以抽象出大部分内容，让你专注于需要学习的关键新事物上。

从这个概念出发，这本书也就诞生了：JavaScript 可以作为电子开发的敲门砖，让更多人学习如何以最小的认知成本来制作很酷的事物。JavaScript 是世界上最流行的编程语言之一，是网络的事实语言；物联网和制造文化在创造性和商业性方面都很吸引人，为什么不把两者融合在一起呢？

到最后，这些事情会很有趣。能够实现自己的发明梦想是一件好事。拥有开发低电压电子电路的基本能力，了解嵌入式系统在现实世界中的运作方式，能够极大地提升信心。也许你会像我一样真心热爱它。

也许你会为开源项目做出贡献，也许你会构建一款精巧的小工具，也许你会将自己的所学分享给他人。

也许你只是自娱自乐。这本身就已足够。

作 者 简 介

Lyza Danger Gardner 喜欢研究如何做事情，同时她也喜欢教导别人如何完成新项目。Lyza 和朋友共同创办了一家位于美国俄勒冈州波特兰市的网络咨询公司——Cloud Four。她拥有二十多年的 Web 开发经验，倡导优雅的标准、教育，以追求最好的未来网络。可以在 www.lyza.com 或 Twitter 上的@lyzadanger 在线找到她。目前她隐居在佛蒙特州的森林里，阅读了大量书籍。

致 谢

首先感谢英国国家计算机博物馆和 Over the Air 会议组成员，因为我对硬件开发的热爱从这里起步，我最初的 Arduino 入门套件也源于此。Dan Appelquist、Margaret Gold 和 Matthew Cashmore，感谢你们举办这样一次精彩的会议并邀请我参加！

Rick Waldron 一天内的开发成果顶得上我一个月的成果(Rick，你真的是 JavaScript 领域的传奇人物)。Rick 加入了 ECMA 工作组 TC-39，该工作组负责 JavaScript 语言。他还发明了 Johnny-Five，这是机器人和物联网领域中领先的开源 Node.js 框架，本书的大部分内容都围绕它展开。对 Rick 的感谢之言，我还能占用几页甚至好几章的篇幅。

写书需要花费大量的时间。非常感谢 Bocoup 的主管和同事为我提供了所需要的时间和支持，并为我的写作源源不断地注入热情。我还要感谢 Cloud Four 的合作伙伴和同事的耐心支持。

在这个通常会忽视编辑过程和反馈价值的世界中，伟大的编辑是真正的馈赠。编辑 Susanna Kline 在长时间的校对中提供了有用且富有洞察力的支持。Brad Luyster，你的技术评审反馈非常好，真的非常感激你。其他几位评委也在不同阶段为本书稿提供了非常宝贵的反馈：Alessandro Campeis、Amit Lamba、Andrew Meredith、Bruno Sonnino、Earl Bingham 和 Kevin Liao。我还要感谢 Manning 的出版商 Marjan Bace，以及其他在幕后工作的编辑和制作团队。

还要感谢 Francis Gulotta 的技术投入，Manning 出版社的 Kyle Jackson 总能给予我技术支持。我的伙伴 Chau Doan 无私地与我分享他的固件和嵌入式电子经验。

Johnny-Five 和相关的 JavaScript 物联网硬件编程社区已经非常优秀。感谢来自 SparkFun 的 Derek Runberg，感谢 Donovan Buck、David Resseguie、Brian Genisio 和其他 Johnny-Five 贡献者。

与那些在项目中帮助你的人同等重要的是在你努力坚持的时候帮你保持头脑清醒的人们：谢谢你们——我的家人和朋友。

在最后这个重要的位置，我要感谢我的伴侣(一个全能的和神奇的人)Bryan Fox：没有你的陪伴和带来的欢乐，这本书不会出版。

前 言

“我对硬件、电子和物联网(IoT)很感兴趣，但是不知该从何处开始学习”，我已从 Web 开发者那里听到很多这样的说法。确实，构建机器人和精巧的装置很有乐趣。知道如何从传感器读取数据，以及使用这些数据做些有意思的事情，就可以构建你自己的自动化的、通过网络控制的设备(例如，用来及时投喂宠物，监测降雨量，显示最新的橄榄球赛比分，能做的事情的确太多了)。但是，如果你很少动手让 LED 闪烁，或者很少编写优化的固件程序并烧录到嵌入式微控制器中，那么这个新的领域就会令人畏惧。

好消息是，你可以使用已掌握的 JavaScript 技能和通用编程范式来开始学习之旅，并且可以武装你的大脑，让你在这个新世界中减少一点纷扰。JavaScript 以你熟悉的方式为你提供试金石，以简化学习电子、硬件和物联网的过程。

本书为那些熟悉基础 JavaScript 但是对基本电路甚至都没有多少经验的人讲述了电路和嵌入式系统的基础知识。重点放在软件开发者不熟悉的主题：设计和构建电路的关键基础知识，硬件元件(传感器、电动机、电阻等)，以及硬件和软件之间的接口。

在本书中，你将亲自体验各种开发板、硬件元件和软件平台。对于本书前 2/3 的实验(小项目)，我们将使用开源 Node.js 框架 Johnny-Five 和 Arduino Uno 开发板。Johnny-Five 的 API 提供了许多直观的元件类，可以使用它们快速构建小工具和发明原型。Uno 是世界上最受业余爱好者欢迎的开发板，它具有稳定、简单、可靠等特性，并且拥有庞大的用户和教育者社区。本书的后 1/3 探索了更广泛的平台，包括支持 Node.js 的 Tessel 2 和非常受欢迎的 Raspberry Pi。

在本书的最后，你应该拥有一个基础工具包——包括精神上的和物理上的——用于规划、设计、实现和扩展你自己的 JavaScript 控制的电子作品。

路线图

本书共包含 12 章内容。

- 第1章定义了什么是嵌入式系统，列举了其物理元件的构成，也解释了JavaScript和硬件协同工作的方式。
- 第2章介绍了Arduino Uno 开发板，阐述了可以让你快速上手操作的让LED闪烁的基本方法。在开始使用JavaScript 和 Johnny-Five之前，我们将简要介绍如何使用Arduino IDE 控制 Uno。
- 第3章聚焦于电子学的基础知识，这些电子学基础是你将要构建的所有电路的基础。你将深入探索欧姆定律并构建一些不同类型的简单电路。
- 第4~6章介绍了嵌入式装置的关键电子设备和概念。探索输入(传感器)、输出(制动器)和物理运动(电动机和伺服器)。使用Johnny-Five 框架，你将有机会使用Arduino Uno 开发板构建一系列不同的实验。
- 第7章研究串行通信，用于交换更复杂的数据。你将尝试使用Johnny-Five 和 Arduino Uno 等多个串行组件，包括指南针(磁力计)、加速度计和GPS。
- 第8章和第9章介绍了支持Node.js 的Tessel 2 开发板。在第8章中，将学习制作没有电线束缚的项目。在第9章中，将探索从构思到开始着手制作项目的过程。
- 第10章和第11章深入研究了其他支持I/O 的嵌入式硬件和JavaScript。第10章介绍了Espruino Pico 等受限平台上的JavaScript 和类似JavaScript 的环境。第11章探讨了更多通用的单板机(SBC)，如Raspberry Pi。
- 第12章介绍云服务和从浏览器端控制硬件，并展望未来。你将学习如何使用云服务resion.io 进行管理，将Johnny-Five 应用部署到BeagleBone Black，你将使用Puck.js 设备和Web Bluetooth API 构建一个浏览器内的无线门铃。

本书读者对象

本书适合那些有一定的JavaScript 经验，但对于电子电路和微控制器编程知之甚少或根本不了解的读者。

本书中的代码示例并不复杂。我的理念是，具有可读性和可理解性的代码更实用。当然，你不需要深入了解ECMA-262 规范(这是定义JavaScript 语言的文档)中的每个单词，但如果你还不适应箭头函数或者还不熟悉Promise，那么需要温习一下JavaScript 知识，或者参阅 *Secrets of the JavaScript Ninja, Second Edition* 一书，该书由John Resig、Bear Bibeault 和 Josip Maras 合作编写(Manning, 2016; www.manning.com/books/secrets-of-the-javascript-ninja-second-edition)。代码复杂性和现代语言功能的使用在本书的后面也逐步增加。

虽然实验的分步骤说明提供了构建项目所需要的所有命令，但你应具备安装、管理和使用 Node.js 和 npm 包管理器的基本能力。你还应该熟悉在终端环境中执行命令。

代码约定和下载

本书包含大量示例，其中包括应用程序和实验所需要的各种资源：JavaScript、HTML、CSS 和 JSON 等。列表或文本中的源代码采用等宽字体，以将其与普通文本分开。

Johnny-Five 是开源的，根据 MIT 软件授权发布。本书使用了许多其他开源软件项目，包括十几个第三方 npm 模块。所探索的大多数硬件平台也是开源的，一个例外是 Raspberry Pi 3，见第 11 章。要完成第 5 章中的“气象球”示例，你需要来自 Dark Sky 的免费的 API 密钥(<https://darksky.net/dev/register>)。

很多源代码清单都带有注释，突出显示了重要的概念。

本书中所有示例的源代码和资源都可以在 <https://github.com/lyzadanger/javascript-on-things> 上找到，也可扫封底的二维码获得。本书中的大多数示例包括正文中所需的所有代码和标记(不包括第三方模块的源代码)。你可以在本书的最后找到一些较长例子的完整资料来源，以及代码库中的二进制资源(例如，第 12 章中网页控制的门铃中使用的 MP3)。

发布时包含源代码的 zip 文件也可以在出版商的网站上找到：www.manning.com/books/javascript-on-things。

书籍论坛

购买本书可以免费访问由 Manning Publication 运营的私人网站论坛，你可以在其中对该书发布评论，提出技术问题，并从作者和其他用户那里获得帮助。论坛网址为 <https://forums.manning.com/forums/javascript-on-things>，你还可以访问 <https://forums.manning.com/forums/about>，了解有关 Manning 论坛和行为规则的更多信息。

Manning 承诺为读者提供一个场所，让读者之间以及读者与作者之间进行有意义的沟通。这并不是对作者任何具体参与的承诺，作者对论坛的贡献仍然是自愿的(而且是无偿的)。为了引起作者的兴趣，我们建议你尝试向作者提出一些具有挑战性的问题。只要本书出版，论坛和之前讨论的合集就可以从出版商的网站上获取。

目 录

第 I 部分 针对 JavaScript 开发者的硬件介绍

第 1 章 将 JavaScript 与硬件结合	3
1.1 硬件项目剖析	4
1.1.1 输入与输出	4
1.1.2 处理过程	5
1.1.3 电源、电路和系统	6
1.1.4 逻辑和固件	8
1.1.5 外壳和封装	9
1.1.6 嵌入式系统	10
1.2 JavaScript 和硬件如何协同工作	10
1.2.1 宿主机-客户端方法	10
1.2.2 嵌入式 JavaScript	13
1.2.3 其他硬件-JavaScript 组合	15
1.3 JavaScript 非常适合硬件项目吗	18
1.4 整合硬件工具包	19
1.4.1 开发板	19
1.4.2 输入和输出元件	20
1.4.3 其他电子元件	21
1.4.4 电源、电线和附件	21
1.4.5 工具	22
1.5 本章小结	24
第 2 章 用 Arduino 开启硬件之旅	25
2.1 了解 Arduino Uno	27
2.2 使用 Arduino 的工作流程	32
2.2.1 Arduino Uno 的数字引脚	32

2.2.2 草图和 Arduino IDE	33
2.2.3 将 LED 连接到数字引脚	35
2.2.4 对 LED 进行编程使其闪烁	36
2.3 使用 JavaScript 控制 Arduino	39
2.3.1 将 Arduino 配置为客户端	39
2.3.2 安装 Node.js 框架 Johnny-Five	41
2.3.3 用 Johnny-Five 让 Hello World 的 LED 闪烁	42
2.3.4 Firmata、Johnny-Five 和宿主机-客户端方法	42
2.3.5 使用 Johnny-Five 组织脚本	44
2.4 本章小结	46
第 3 章 如何构建电路	47
3.1 电压、电流和电阻	48
3.1.1 欧姆定律	52
3.1.2 问题和危险	54
3.2 构建电路	55
3.2.1 使用面包板制作原型电路	55
3.2.2 在面包板上连接简单的 LED 电路	56
3.2.3 用按钮扩展串联电路	63
3.2.4 串联 LED	65
3.2.5 并联电路和分流器	69
3.2.6 用电池为项目供电	75

3.3 本章小结 76 第 II 部分 项目基础：使用 Johnny-Five 输入和输出 第 4 章 传感器和输入 81 4.1 使用模拟传感器 84 <ul style="list-style-type: none"> 4.1.1 模数转换 84 4.1.2 光敏电阻的使用 85 4.1.3 分压器 89 4.1.4 布线和使用光敏电阻 92 4.1.5 使用模拟温度传感器 97 4.2 数字输入 101 4.3 本章小结 106 第 5 章 输出：让事情发生 107 5.1 点亮 LED 108 <ul style="list-style-type: none"> 5.1.1 使用脉冲宽度调制(PWM) 使 LED 变暗 109 5.1.2 使用 PWM 让 LED 做动画 113 5.1.3 将输入与 LED 输出结合 117 5.1.4 全彩 RGB LED 122 5.1.5 构建你自己的 “气象球” 122 5.2 使用并行 LCD 显示器 126 <ul style="list-style-type: none"> 5.2.1 用 LCD 制作功能齐全的 定时器 126 5.2.2 添加可视的 LED “铃声” 138 5.3 用压电器制造噪声 141 5.4 本章小结 145 第 6 章 输出：让物体运动 147 6.1 让电动机运转 148 <ul style="list-style-type: none"> 6.1.1 电动机的工作方式 149 	6.1.2 使用按压按钮开关控制 电动机 151 6.1.3 用 Johnny-Five 控制 电动机 156 6.2 制作伺服器 159 6.3 制作你的第一个机器人 165 <ul style="list-style-type: none"> 6.3.1 机器人和电动机 167 6.3.2 制作机器人的基础底盘 169 6.3.3 控制机器人的电动机 170 6.4 本章小结 181 第 III 部分 更复杂的项目 第 7 章 串行通信 185 7.1 并行和串行通信数字数据 187 7.2 串行通信的基础知识 188 7.3 异步串行通信 189 <ul style="list-style-type: none"> 7.3.1 UART 191 7.3.2 使用 GPS 扩展板试用软件 串口 192 7.3.3 学习焊接 194 7.3.4 构建 GPS 电路 198 7.4 同步串行通信 200 <ul style="list-style-type: none"> 7.4.1 串行外围设备接口(SPI) 201 7.4.2 I²C 202 7.4.3 使用 I²C 磁力计制作数字 罗盘 204 7.5 整合在一起：摇动-改变多传 感器部件 206 <ul style="list-style-type: none"> 7.5.1 步骤 1：将罗盘与 LCD 输出 相结合 207 7.5.2 步骤 2：向设备中添加多 传感器 210 7.5.3 步骤 3：更新显示屏，显示 温度和压力 211
---	--

7.5.4 步骤 4：使用加速度计添加 摇动-交换显示功能 213	9.3.3 集成手势传感器和远程 开关 299
7.6 本章小结 217	9.3.4 将整个项目整合在一起 303
第 8 章 无线项目 219	9.4 本章小结 305
8.1 为什么你还要使用数据线 221	
8.1.1 数据交换、I/O 层和 I/O 插件 221	
8.1.2 USB 充当电源 222	
8.1.3 无线项目通信的选项 223	
8.2 使用 Tessel 2 实现无线项目 225	
8.3 设置 Tessel 226	
8.3.1 配置 Tessel 226	
8.3.2 在 Tessel 上运行 “Hello World” LED 闪烁代码 229	
8.3.3 通过 Tessel 闪烁外部的 LED 231	
8.3.4 探索 Tessel 的引脚和 功能 235	
8.4 基于 Tessel 的无线项目 236	
8.5 用电池为项目供电 252	
8.6 本章小结 259	
第 9 章 自己制作硬件 261	
9.1 消费电子产品开发 263	
9.2 用 Johnny-Five 插件控制远程 开关 270	
9.2.1 开关项目的原型设计 270	
9.2.2 编写 RemoteSwitch 插件 274	
9.3 编写复杂硬件的软件 280	
9.3.1 项目：Johnny-Five 支持 APDS-9660 手势传感器 281	
9.3.2 实现构造函数和初始化 方法 293	
	第 IV 部分 在其他环境中的硬件上 使用 JavaScript
	第 10 章 JavaScript 和受限制的 硬件 309
	10.1 Espruino Pico 平台 311
	10.1.1 设置 Pico 312
	10.1.2 Hello World 版的 LED 闪烁 313
	10.2 了解新平台 315
	10.2.1 了解平台的核心功能 316
	10.2.2 查找引脚图 318
	10.2.3 了解配置和工作流程 319
	10.2.4 查找示例和教程 319
	10.2.5 使用 API 参考文档 319
	10.3 试验 Pico 320
	10.3.1 Pico 和 BMP180 多 传感器 320
	10.3.2 Pico 和 Nokia 5110 液晶 显示器 323
	10.3.3 使用 Pico 构建高效的天气 小工具 329
	10.4 试验 Kinoma Element 平台 332
	10.4.1 Element 的核心功能 333
	10.4.2 引脚和硬件图 333
	10.4.3 配置、管理和工作 流程 334
	10.4.4 示例和教程 335
	10.4.5 API 参考 336

10.4.6 案例研究项目：实时更新罗盘读数 336 10.5 本章小结 344 第 11 章 使用 Node.js 和微型计算机进行硬件开发 347 11.1 使用微型计算机 349 11.1.1 Raspberry Pi 平台 350 11.1.2 配置方式 1：传统方式 354 11.1.3 配置方式 2：无头配置 355 11.2 了解 Raspberry Pi 3 359 11.2.1 核心特性 359 11.2.2 GPIO 特性和引脚 361 11.2.3 配置和工作流程 362 11.2.4 示例和教程 365 11.2.5 API 文档 372 11.3 为不同的平台编写 Johnny-Five 应用程序 372 11.3.1 改造迷你气象站使其适配 Pi 3 373 11.3.2 改造迷你气象站使其适配 Arduino Uno 379 11.4 使用 Raspberry Pi 作为宿主机 380 11.5 案例研究：BeagleBone Black 381 11.5.1 BeagleBone Black 381 11.5.2 气象站程序针对 BeagleBone 进行适配 387 11.6 本章小结 388	第 12 章 在云端、在浏览器中以及更多可能性 391 12.1 IoT 与云 392 12.2 使用 resin.io 进行容器化部署 394 12.2.1 创建 resin.io 应用程序 396 12.2.2 配置 BeagleBone Black 397 12.2.3 适配天气应用程序软件 399 12.3 硬件和 Web 浏览器 404 12.3.1 Web 蓝牙 API 405 12.3.2 通用传感器 API 405 12.3.3 Physical Web 405 12.4 使用 Puck.js 探索 Bluetooth LE 406 12.4.1 核心特性 407 12.4.2 GPIO 特性和引脚分布 408 12.4.3 配置和工作流程 409 12.4.4 示例、教程和 API 文档 411 12.4.5 从网页控制 LED 411 12.4.6 Physical Web 和 Puck.js 417 12.4.7 基于 Web 的蓝牙门铃 419 12.5 开拓 JavaScript 硬件编程的边界 431 12.6 本章小结 431
--	---

第 I 部分

针对 *JavaScript* 开发者的 硬件介绍

本书的这一部分介绍嵌入式系统和电子电路的基础知识。第 1 章介绍什么是嵌入式系统以及如何分析其组成元件。我们将花点时间来研究如何用 JavaScript 来“控制”硬件，并尝试用不同的方法让 JavaScript 和电子设备协同工作。

第 2 章介绍 Arduino Uno R3 开发板，第 7 章中所有的实验都会用到它。这一章会介绍开发板主要组成部分的功能，以及开发板与其他软件和硬件元件如何进行交互。还会尝试使用 Uno 进行一些基本的 LED 实验，其中会用到 Arduino IDE 和 Johnny-Five Node.js 框架。

第 3 章将介绍电子电路的关键基础知识，并深入研究欧姆定律以及电压、电流和电阻之间的关系。在面包板上构建包含多个 LED 的串联和并联电路。

学完这一部分，你将掌握基本的嵌入式系统基础和核心电路概念，此后就可以开始使用不同类型的输入和输出构建 JavaScript 控制的小型项目了。

第 1 章

将 JavaScript 与硬件结合

本章内容包含：

- 业余项目和“物联网”中涉及的元件和硬件
- 嵌入式系统中的常用元件
- 结合使用 JavaScript 与嵌入式系统的不同方法
- 开始构建项目所需要的工具和用品

作为一名精通 JavaScript 的 Web 开发者，你每天都会写一些逻辑代码。现在你的软件开发技能有了新的用武之地，即在真实世界中通过编写程序来控制硬件。本章将介绍不同类型的项目和设备所涉及的硬件，以及 JavaScript 和硬件如何协同工作。

我们周围有一些神奇物件，它们将物理世界与逻辑、连接和虚拟领域相结合(见图 1.1)。钥匙串能无线广播它的位置，这样就可以通过智能手机上的应用程序找到它。花盆在它需要浇水时能发出鸣叫声，甚至能给你发送一条督促短信。地球上几十亿种这样的东西在闪烁、哔哔作响、鸣叫、自动调暗光亮、按定制泡好茶，以及履行其专门职责。

构建这些东西很有趣。在制作这些实物小工具时体现的创造力和发明创造自制项目展现出的草根魅力，都深深吸引了 Web 开发者。我们非常适合进行原型设计、尝试新技术和自我创新。

但是一开始会让人心生畏惧。当我们看到这些电线和元件，听到一些术语，进入硬件玩家社区浏览时，其中涉及的各种技能让人感到陌生和害怕。作为 JavaScript 开发者，在初步尝试进入物理硬件世界时，会面临一些障碍，包括理解上的复杂性、丰富而零散的信息、混合在一起的硬件和软件概念。



图 1.1 我们世界中的神奇物件

拥有 JavaScript 技能是一种优势，有助于学习设计和构建组成物联网(Internet of Things, IoT)所需要的各类部件，并且能够给硬件玩家带去灵感。软件开发技能也可以用来越过一些障碍，将精力迅速集中在需要学习的新技能上。

为了感受一下我们将要进行的旅程，这里首先介绍将要学习的内容，解释这里提到的部件(thing)或硬件(hardware)的具体含义。

1.1 硬件项目剖析

我们可以构建一个小工具，温度升高时会自动打开风扇。这种微型的、独立的气象控制设备会持续地监测周边环境的温度。当气温变得太热时，风扇开启；当气温适宜和变冷时，风扇关闭。

虽然这种发明的想象力不足以为我们赢得什么有名气的奖项，但是它的基本组成和你将要学会构建的更有创意的项目是一样的。

1.1.1 输入与输出

温控设备需要做的最重要的也是唯一的一件事，就是过热时打开风扇，周围温度降至较冷时，再把风扇关闭。这个电动风扇就是一个输出设备。

为了持续获取周围环境的温度信息，让设备可以决定何时打开或关闭风扇，我们需要从输入设备获取数据，在这个例子中，输入设备就是温度传感器(见图 1.2)。

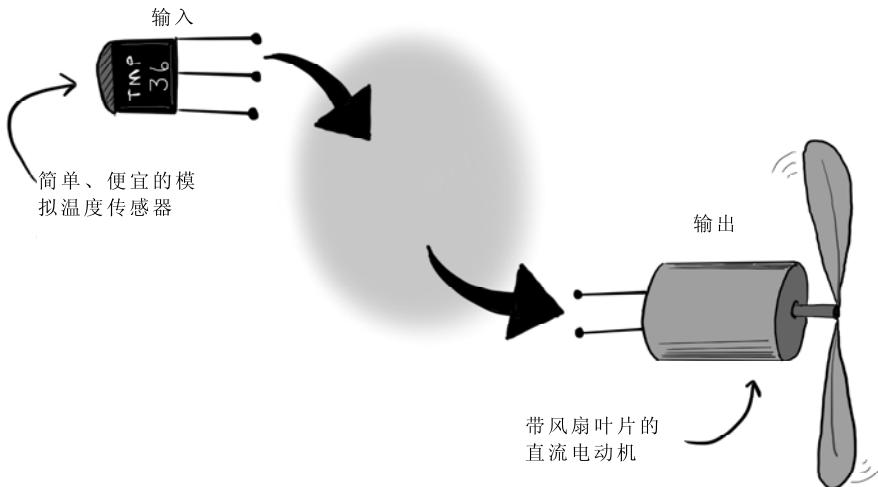


图 1.2 自动风扇系统需要从温度传感器中获取输入信息并管理电动风扇的输出

输入为系统提供输入数据，传感器就是一种提供物理环境数据的输入设备。在项目中可以使用各种传感器，如光、热、噪声、振动、蒸汽、湿度、气味、移动、火焰等传感器。有些传感器提供一些简单的数据，比如风扇的温度传感器只提供一个代表温度的数值；而其他的传感器，比如 GPS 或加速度计，可以产生更精细的数据。

项目的输出代表这个项目提供给使用者的实际功能。闪烁的灯光、刺耳的哔哔声、LCD 屏幕上的状态读数、向侧面移动的机械臂——所有这些都是输出。在这个项目中，风扇是唯一的输出。

并非所有输入输出都必然体现在物理世界中。购物者尝试在线购买某产品(虚拟的输入)时遇到报错，可能会使技术支持人员桌面上设备的红灯闪烁(物理输出)。相反，土壤湿度的变化(物理感应输入)会让花盆发送一条指令信息(虚拟输出)。

1.1.2 处理过程

自动风扇也需要大脑，能注意到温度传感器的读数，并且在温度过高时打开风扇。这里所需要的大脑实际上是一台小型计算机，包含一个处理器、一些存储器以及处理输入和控制输出的能力。当处理器、存储器和输入输出(I/O)功能都被集成在一个小型的物理封装中时，把这样的集成芯片称为微控制器(见图 1.3)。

微控制器(MCU)不像笔记本电脑中的通用处理器那么强大。大多数(不是全部)都无法运行一个完整的操作系统，但是它们价格便宜、功能可靠、体积小，功耗极低——这就是它们在类似电动风扇等硬件项目和产品中无处不在的原因。

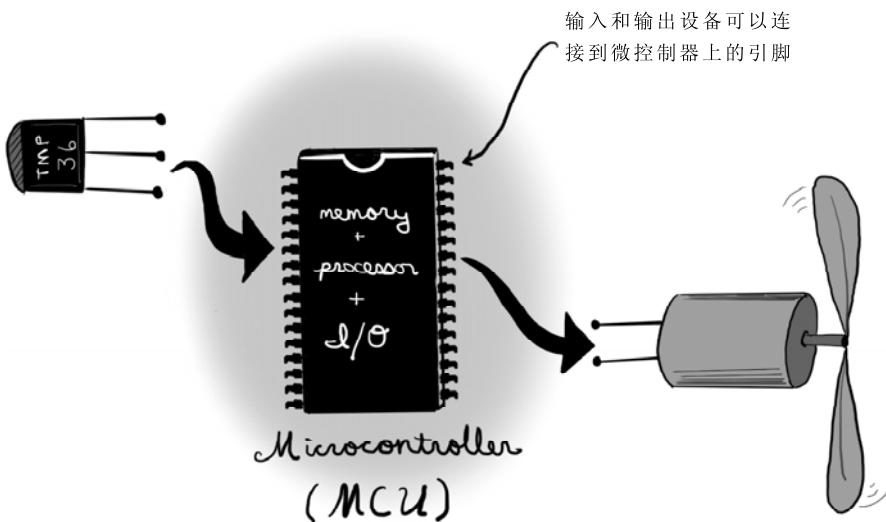


图 1.3 自动风扇需要大脑。通常使用微控制器，它将处理器、存储器和输入输出功能集成在一个芯片中

1.1.3 电源、电路和系统

现在我们有了输入输出和大脑——是时候把它们组合在一个系统里了。我们需要使用一个或多个电子电路来连接元件并且接上电源。构建一个系统涉及电路设计和在物理空间中操作元件(见图 1.4)。

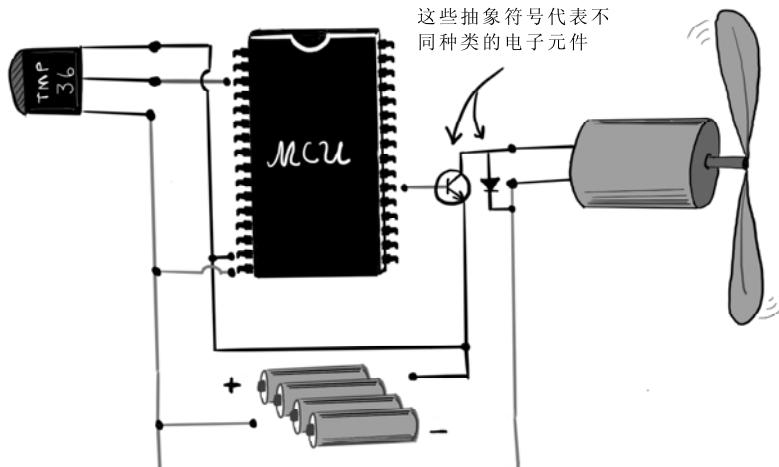


图 1.4 粗略的原理图显示了电扇的输入输出和微控制器是如何与电源和电路连接成一个系统的。如果是初次看这些符号，也不要紧张，后面还会学习这些电路

将导线直接连接到微控制器的微小引脚需要焊接和稳定的手法。更不用说到最后很多零部件都杂乱地放在周围。为了方便硬件开发者的使用，微控制器通常安装在物理开发板上(见图 1.5)。此外，开发板还可以更轻松地将 I/O 设备连接到微控制器。

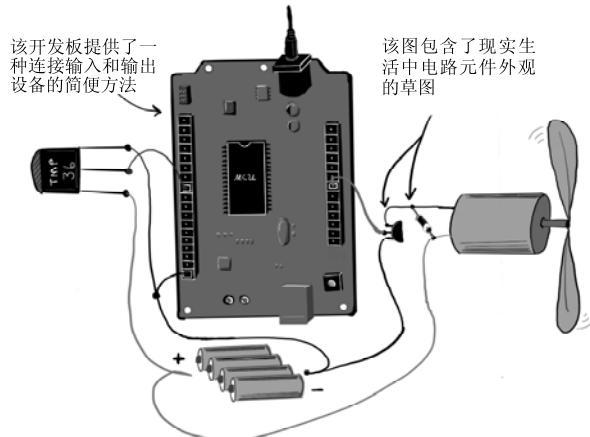


图 1.5 基于微控制器的开发板使连接输入和输出设备更加方便

开发板的作用很强大，但我们还是要处理很多零散的电线和元件。为了让元件摆放整齐，硬件开发者可以使用一种称为面包板(breadboard)的原型工具(见图 1.6)，在面包板的物理空间中布置电路。

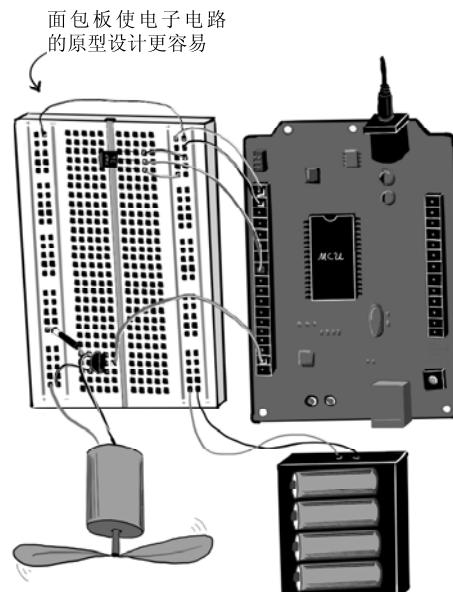


图 1.6 面包板提供电子连接的网格，在该网格上可以进行电子电路的原型设计

1.1.4 逻辑和固件

我们的硬件设计正在推进，但是你可能想知道微控制器如何知道它该做什么。这里就涉及逻辑，如代码清单 1.1 所展示：监听传感器，做出决策、发送指令来打开或者关闭风扇。

代码清单 1.1 温度触发风扇逻辑的伪代码

```
initialize temperatureSensor
initialize outputFan
initialize fanThreshold to 30 (celsius temperature)

loop main
    read temperatureSensor value into currentTemp
    if currentTemp is greater than fanThreshold
        if outputFan is off
            turn outputFan on
        else if currentTemp is less than or equal to fanThreshold
            if outputFan is on
                turn outputFan off
```

C 语言(或者类似 C 语言的衍生语言)是为微控制器编程所使用的主要语言。为微控制器编写 C 语言程序往往与平台有关，并且非常底层化。对指定内存地址的引用和按位操作很常见。

代码被编译成与体系结构相关的汇编代码。代码通过物理上传或者烧录到微控制器的程序存储器中，可以导入项目。

这种程序存储器通常是非易失存储器——ROM，它可以让微控制器在断电的情况下依然能“保存”程序(而 RAM 仅在供电时才会保存其内容)。程序的可用空间有限，通常在几万字节左右，这也意味着在微控制器上运行的程序需要很好地优化。

一旦将程序烧录到微控制器，它就像微控制器的固件一样——供电时，微控制器持续运行程序，直到它可能被编写了其他程序功能(或复位)。

对于习惯更上层逻辑的 JavaScript 开发者来说，对这种底层技术可能不是很适应。不必担心，正是 JavaScript 使得我们可以基于微控制器的硬件编写程序，而又不必使用 C 语言，或者纠结十六进制寄存器地址等细节。

芯片技术的进步和对业余爱好者友好的开发板带来的广泛可用性(见图 1.7)，使得把程序固件烧录到微控制器的过程更加容易。

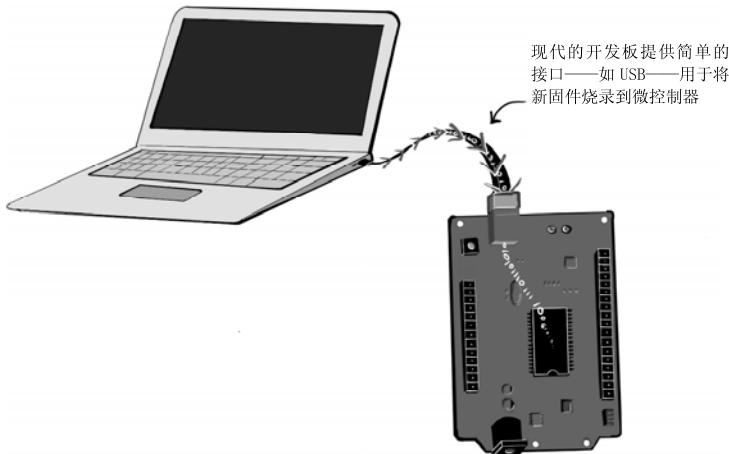


图 1.7 非易失性程序存储器(EEPROM 和闪存)和对用户友好的开发板使得用固件对微控制器编程变得更容易

EEPROM(Electrically Erasable Programmable ROM, 带电可擦可编程存储器)是一种众所周知的闪存介质，广泛用于微控制器。这种可重写存储器可以用不同的逻辑一遍又一遍地重新对微控制器编程。

开发板除了让 I/O 连接更容易外，还为开发板的微控制器编程提供了方便的接口(通常是 USB)，为硬件开发者带来便利。这使得我们不必精通具体的硬件编程设备。目前，微控制器编程通常就像插入 USB 线，再单击 IDE 中的按钮一样简单。

1.1.5 外壳和封装

至此，电扇的设计已基本完成。但是我们要将其提升到一个新的水平，将自动风 扇封装在一个漂亮的外壳内，将系统嵌入里面，将电线和电路隐藏起来(见图 1.8)。



图 1.8 这个完整的、封装好的自动风扇就是一个嵌入式系统的例子。输入和输出由带有微控制器的微型计算机处理，并由电源和电路支持。整体都被隐藏在一个漂亮的盒子中，有什么理由不这样做呢

1.1.6 嵌入式系统

虽然“嵌入式系统”这个术语听起来令人生畏，但实际上并不是很复杂。一个由处理器、存储器和 I/O 组成的微型计算机构成了大脑。如你所见，自动风扇将输入输出和微型计算机连接在一起，并为其供电，从而创建了一个独立的系统。之所以说它是嵌入式的，是因为它总是被塞进一些东西里，如一个外壳、一只泰迪熊、一台洗衣机的控制面板、一把伞。

虽然自动风扇、下雨时会点亮的雨伞、一只吱吱叫的泰迪熊看上去完全不同，但是它们拥有的共同点比你想象的要多。这些示例以及构成物联网的大多数硬件项目和设备都可以称为嵌入式系统。现在看看 JavaScript 是如何融入其中的。

1.2 JavaScript 和硬件如何协同工作

在 JavaScript 和嵌入式系统相结合时，仍然以与其他类型硬件项目相同的方式构建电子电路。仍然有输入和输出、电线和元件。但这里不使用汇编语言或 C 语言，而是使用 JavaScript 来定义项目的微控制器或处理器的功能。

使用 JavaScript 为硬件项目提供逻辑有几种不同的方法。这些方法按照 JavaScript 逻辑本身执行的载体位置分类：要么在与嵌入式系统分离的主机上，要么在嵌入式系统的微控制器上，或者其他任何地方。

1.2.1 宿主机-客户端方法

为了规避一些微处理器的限制，宿主机-客户端方法(host-client method)可以让你在更强大的宿主机上执行 JavaScript。宿主机运行代码时，它与嵌入式硬件交换指令和数据，这种嵌入式硬件就像是一个客户端(见图 1.9)。

有些微控制器的局限性会影响其运行 JavaScript 的能力。程序存储器的容量是有限的，这意味着复杂的程序要么不适合，要么必须大力优化。此外，许多廉价的微控制器都是以 8 位或者 16 位架构构建，其运行时钟速度要低于桌面计算机。在这些廉价的微控制器中，大多数都不能运行操作系统，它们摒弃了在芯片上直接运行 Node.js 或其他 JavaScript 运行时的能力。

相反，宿主机-客户端方法是在宿主机上执行 JavaScript 逻辑，如笔记本电脑，它能够运行完整的操作系统。宿主机能够运行 Node.js，并且可以利用整个 JavaScript 软件生态系统(包括 NPM 和 Web)。

这种设置的工作原理就是让客户端硬件(如微控制器)和宿主机系统(笔记本电脑)使用可以相互理解的“语言”(一种通用 API)来互相通信(见图 1.10)。

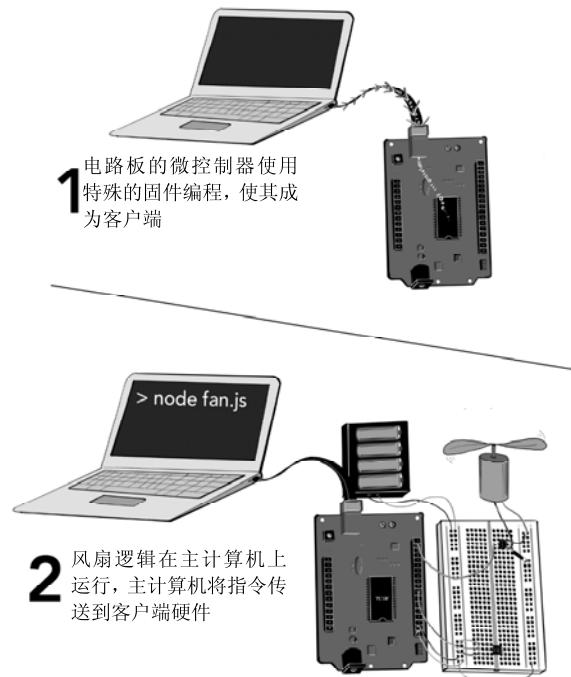


图 1.9 用 JavaScript 控制硬件的宿主机-客户端方法

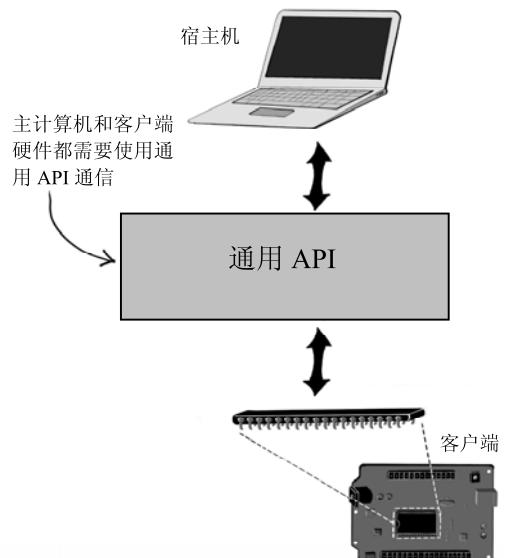


图 1.10 要使宿主机和客户端硬件用此方法通信, 两者都要使用通用 API

要使用此方法配置自动风扇系统，首先需要将特定的固件上传到微控制器的程序存储器中来准备嵌入式硬件。这种固件程序不仅用来控制风扇，还使微控制器能够与其他使用相同“语言”的源(API)相互通信。也就是说，基于微控制器的硬件转变为客户端，监听来自宿主机的指令(见图 1.11)。

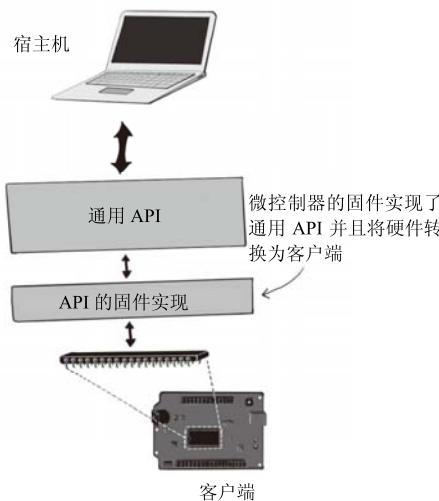


图 1.11 特定固件将微控制器转换为客户端

现在硬件已经准备好进行通信——下一步是使用宿主机为风扇编写软件。为了使硬件和软件能互相理解对方，宿主机需要用微控制器可以理解的语言来表达指令。为此，我们编写代码时可以使用实现了通用 API 的工具库或框架(见图 1.12)。

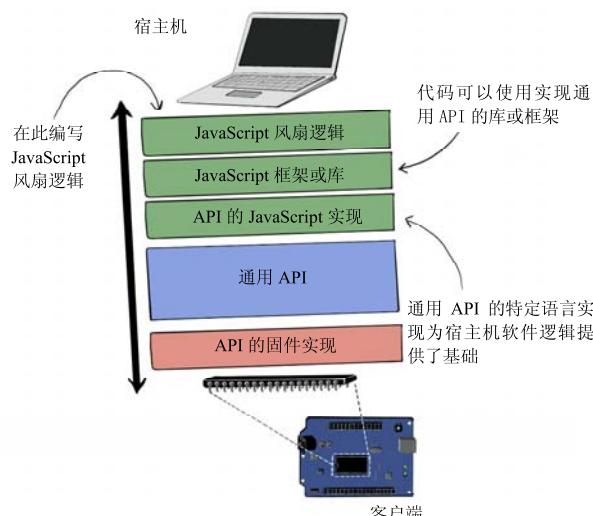


图 1.12 宿主机也需要使用通用 API 进行通信

宿主机与客户端硬件通过物理有线连接(通常是USB)或无线(Wi-Fi或蓝牙)连接。然后我们在宿主机上执行控制风扇的JavaScript程序。宿主机持续向客户端发送运行风扇的指令。客户端也可以向宿主机返回消息,如来自温度传感器的数据(见图1.13)。

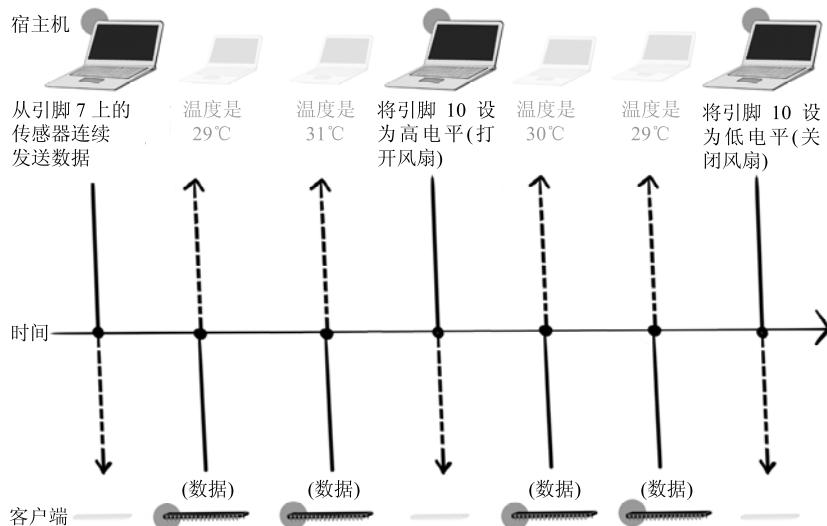


图1.13 当宿主机执行JavaScript逻辑时,客户端和宿主机之间通过使用通用API不断地交换指令和数据

不要惊慌,你不必编写底层的固件协议API软件。有很多简便的、开源的且实现了这些固件协议的固件和Node.js框架可供选择,因此你可以轻松编写宿主机端的JavaScript逻辑。

宿主机-客户端方式的好处是易于配置,并且得到众多平台的支持。更重要的是,它使你可以访问整个Node.js生态系统,同时规避了便宜的微控制器带来的性能和内存限制。缺点是客户端硬件在没有宿主机的情况下无能为力——它只在宿主机运行软件时才能正常工作。

我们最终会进行无线连接,但是要从最简单的宿主机-客户端选项——USB连接开始。这意味着,在一段时间内,你的项目是以物理方式附在计算机上的。

1.2.2 嵌入式JavaScript

使用嵌入式JavaScript时,控制项目的JavaScript逻辑可以直接在硬件的微控制器上运行。

很多微控制器本身并不能运行 JavaScript，但有些可以。随着技术的进步，廉价的微控制器也越来越先进。直接在某些嵌入式处理器上运行 JavaScript 或 JavaScript 的优化变体，已经成为可能。

每个嵌入式 JavaScript 平台都是硬件和软件组件以串联方式协同工作的组合。在硬件端，开发板本身运行代码的任务要基于更强大(但仍然便宜)的芯片。

大多数平台还为实现硬件配套了一系列软件工具。一般会提供一个类库或者框架，用于编写兼容的 JavaScript 代码和 CLI(命令行接口)，或其他准备代码并上传到微控制器的方法。

Espruino(www.espruino.com)是一个基于 JavaScript 的嵌入式平台的示例。在 Espruino 平台上，JavaScript 的风格在于将优化的核心 JavaScript 与具有硬件相关功能的 API 相结合。例如，在 Web IDE 中为 Espruino Picoboard 编写代码并通过 USB 上传到开发板中(见图 1.14)。为了使自动风扇适配 Espruino 开发板，需要使用 Espruino 的 API 编写逻辑。

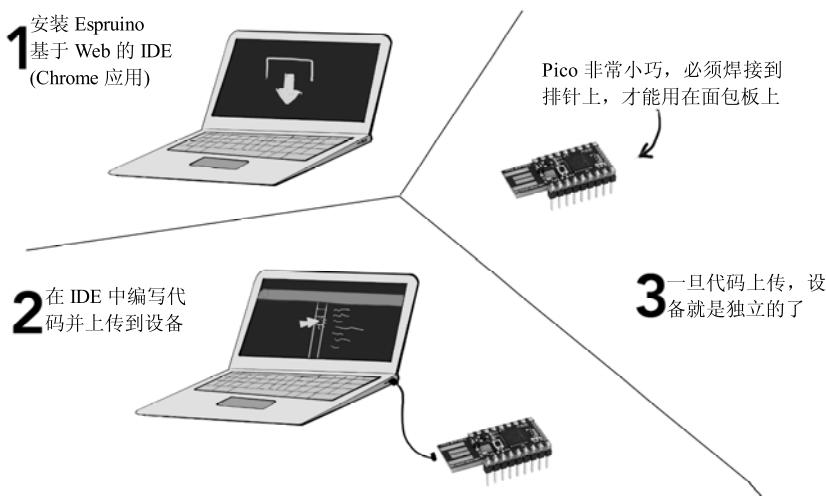


图 1.14 Espruino 平台结合了小型硬件开发板和一个 IDE 开发环境

嵌入式 JavaScript 的另一个示例是 Tessel 2 (<https://tessel.io/>)。这是一个基于 Node.js 的开发平台。由于 Tessel 2 有内置的 Wi-Fi，因此可以使用 `tessel-clinpm` 模块，通过无线方式来控制代码并将其部署到 Tessel(见图 1.15)。

可直接在嵌入式硬件中运行 JavaScript，一般来说功耗较低并且相对独立。项目是可以独立运行的独立系统。与需要固件从 JavaScript 转换为机器代码的宿主机-客户端不同，JavaScript 和硬件之间的抽象层通常更少。

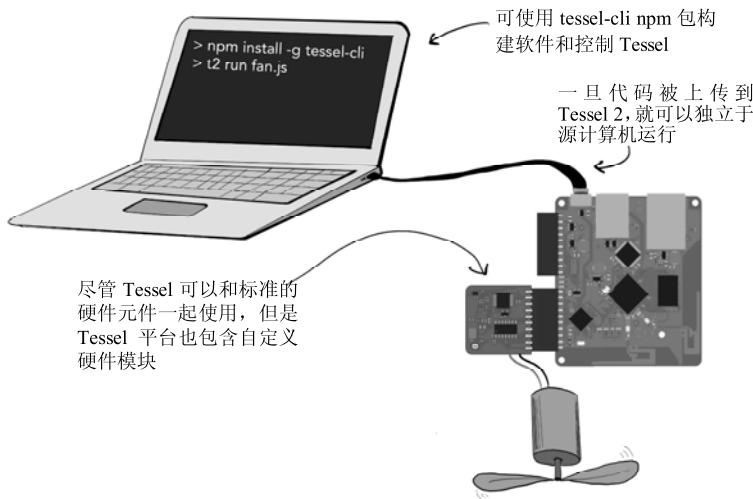


图 1.15 Tessel 2 是原生运行 Node.js 的开源平台

这听起来很不错，你可能想知道为什么我们不只使用这种方法。这种方法有一些缺点。首先，目前可选的硬件较少。此外，每个平台有自己的平台相关技术(软件、工具和方法)，这会给学习硬件基础知识带来困扰。无论是在 JavaScript 语言功能支持，还是在支持输入和输出的类型中，大多数也有一些限制。但这是一个鼓舞人心的方法，前景非常光明。

1.2.3 其他硬件-JavaScript 组合

除了宿主机-客户端方法和运行嵌入式 JavaScript 之外，还有一些方法可以将 JavaScript 与硬件项目相结合。

微型单板计算机(Single-Board Computer, SBC)将宿主机和客户端合为一个单元。云服务可以在线编写 JavaScript 代码并将其无线部署到硬件上。Web 浏览器本身不断涌现的新功能和实验性功能，为数百万 Web 开发者进入硬件世界提供了便利。

1. 在微型计算机上运行 JavaScript

Raspberry Pi 系列和 BeagleBone Blackcan 这样的单板计算机(SBC，简称单板机)可以运行完整的操作系统环境(通常是 Linux)，并且可以通过扩展运行 Node.js。单板机(SBC)不是 8 位或 16 位微控制器，而是具有更高性能的通用处理器。但是许多单板机还在同一块开发板上内置了 I/O 引脚和功能(见图 1.16)。

使用单板机控制硬件项目融合了宿主机-客户端方法和运行嵌入式 JavaScript 的各个方面。处理器必须持续运行 JavaScript 逻辑，才能使项目正常工作(就像在宿主机-客户端模式中一样)，但是是将整个软件包放在一块开发板上，更像是一个独立的嵌入式设置。

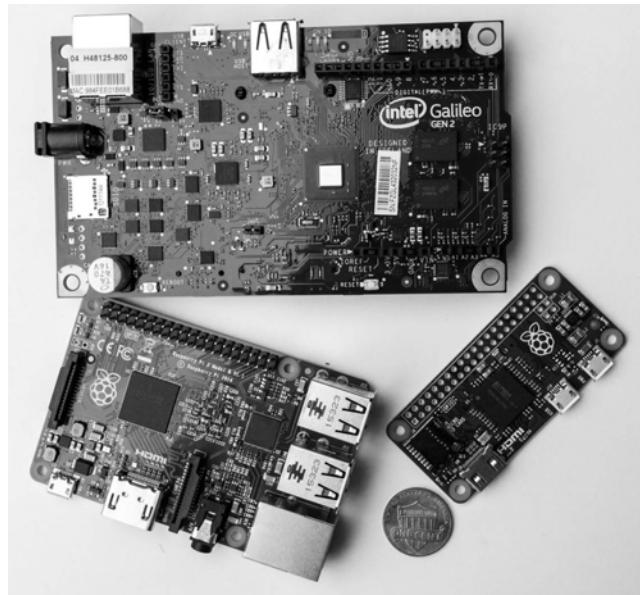


图 1.16 几个单板机(SBC): Intel Galileo, Gen 2(上)、Raspberry Pi 2 Model B(左下)和 Raspberry Pi Zero(右下)

与运行嵌入式 JavaScript 逻辑的微控制器不同，单板机上的处理器不仅能运行单一用途的程序，它还可以同时运行其他进程。

这些单板机越来越便宜。如今，有 5 美元的 Raspberry Pi Zero(通常不容易买到，总是缺货)，支持 Wi-Fi 的 Pi Zero W 则贵一点。低功耗的微控制器硬件同具有与平板电脑和智能手机相媲美的处理器的微型计算机之间不再存在很大的成本差异。

虽然在支持 GPIO(通用 I/O)的单板机上运行 JavaScript 可以在一个封装的硬件上提供多选项，但它也有一些缺点。单板机并不像许多基于微控制器的主板那样具有低功耗——Raspberry Pi 2 Model B 型功耗为 4W。单板机确实有 GPIO 支持，但是引脚映射和使用可能令人困惑，文档过于简单或太过技术性，如果只是学习硬件使用，这可能是一个挑战。还需要准备好面对系统管理障碍，因为单板机的 Linux 发行版(特别是与 Node.js 结合使用时)需要做一些调试。

2. 云服务和浏览器

不可否认，这是硬件-JavaScript组合的最后一个包罗万象的类别。技术更新非常快，目前物联网基于云的商业服务的增长已经呈现出曲棍球棒形状，前面介绍了非常先进的技术，利用这些技术可以从浏览器直接与硬件交互。

云服务试图简化大规模管理物联网设备带来的复杂性。其中许多都是针对企业的。例如，Resin.io(见图1.17)构建、打包容器化的应用程序代码并将其部署到分配的设备，从而处理一些安全和自动化问题。



图1.17 Resin.io服务有助于简化应用程序的部署和支持Linux的单板机的管理

然后就是浏览器本身，许多最先进的硬件-JavaScript组合开始出现。有些浏览器已经允许试着使用Web蓝牙，这种API虽然目前不在标准轨道上，但可能是未来网络化的先驱。顾名思义，Web蓝牙允许你从浏览器中使用JavaScript来连接和控制Bluetooth Low Energy(BLE)硬件。

谷歌推出的另一个开放项目——物理网络(Physical Web)，提出了一个简单的想法：让小型设备能够使用Bluetooth Low Energy(BLE)广播URL。这样使用的信标可以通过该信息将URL广播到Web应用程序，将车站牌转换为实时到达跟踪器(见图1.18)。虽然概念简单，但胜在灵活。

在JavaScript和硬件的所有结合中，这种变体——网络与硬件的更深度集成——是最不稳定的。它具有吸引力，也有不可预测性。通过更多方式使用JavaScript构建物联网产品的需求可能会给这个领域带来令人炫目的进步。

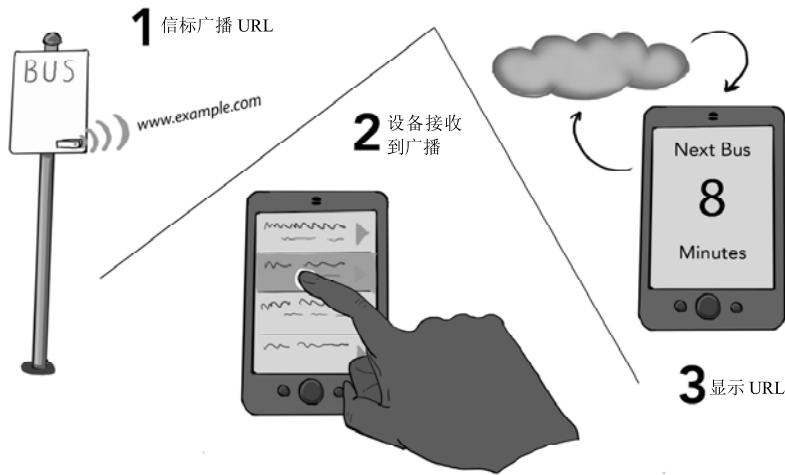


图 1.18 在这个物理网络示例应用程序中，车站牌通过一个 BLE 信标每秒广播一次 URL(1); 附近的人能够在他们的设备上扫描到可用的信标并选择对应这个车站牌的信标(2); 公交车司机的设备能获取信标广播的 URL 并显示在浏览器上(3)

1.3 JavaScript 非常适合硬件项目吗

也许我们可以使用 JavaScript 在硬件上以各种方式进行开发，但是应该这样做吗？这种方式的确有用还是仅仅是一种任性的小把戏？

几年前，使用 JavaScript 进行硬件开发的想法开始出现时，并没有大受欢迎。它被一些人视作武断的和错位的小聪明——“我们必须到处使用 JavaScript 吗？”其他人则认为除了业余项目，在受限的硬件上执行 JavaScript，其性能对于项目而言是不可接受的。相当多的守旧顽固派出现了，评论栏中满是对 C/C++ 以外的语言的激烈指责，并且反对者警告说，对新人来说，更高层次的语言会掩盖一些重要的底层硬件的细微差别。

然而，仍有很多人对此保持开放的态度。为什么 C/C++ 已经足够好了还要使用 JavaScript？这个问题可以用早期硬件编程范式的转变来回应：为什么汇编语言足够好了，还要使用 C 语言？

无论好坏，人们现在都不会有这种争论了，JavaScript 是互联网事实上的编程语言。人们都了解它，使用它，它无处不在。JavaScript 无处不在的特性赋予了它独特的潜力，它为数百万有意愿加入物联网的 Web 开发者打开了大门。

JavaScript 编程的某些方面非常适合硬件，特别是它在事件处理和异步进程方面的成熟度。JavaScript 也是原型设计的好工具，是快速迭代的福音。

看看我们最终会走到哪里更有吸引力。JavaScript列车正在驶出硬件站，很多人都会搭上这趟顺风车远行。

1.4 整合硬件工具包

你已经快速浏览了组成嵌入式系统的各部分以及结合硬件与 JavaScript 的方法。本节更具体地说明构建这类项目所需要的物理硬件、附件和工具的类型。然后将准备一个基本工具包来帮助你入门。

我们的项目将开发板与输入和输出硬件相结合。要构建电路并将系统连接在一起，需要一些辅助的电子元件，以及电线、电源和附件。给你几个基本工具，你就能做好准备。

1.4.1 开发板

开发板(也称为原型板或面板)是结合了微控制器或其他有支持功能的处理元件的物理开发平台(见图 1.19)，它们是硬件开发中的面包和黄油。开发板的成本从几美元到类似高端单板机的 100 美元以上不等。

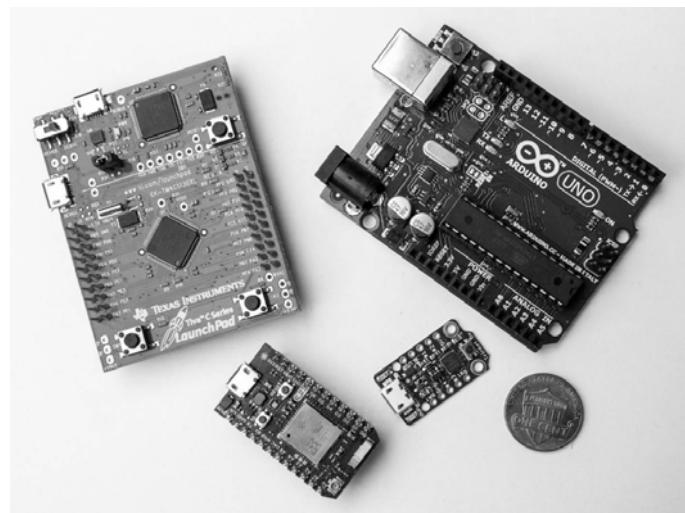


图 1.19 一些典型的基于微控制器的开发板，从左上角开始按顺时针方向依次是：来自德州仪器(TI)的 Tiva C 系列、LaunchPadArduino Uno R3、Adafruit Trinket(5V 型号)及 Particle Photon

开发板以其大脑为中心，结合了处理器、存储器和 I/O 设备。大多数微控制器是 8 位或 16 位 Arduino 的直接入门级原型开发板的核心(见图 1.20)。拥有更复杂的 32 位微控制器的开发板可以运行嵌入式 JavaScript。

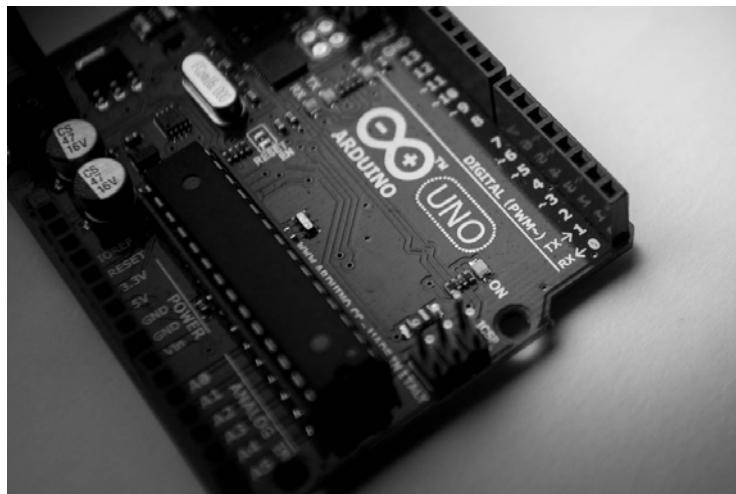


图 1.20 Arduino Uno 开发板由 8 位微控制器 AVR Atmega 328-P 驱动

并非所有的开发板都基于微控制器。更强大的单板机是由计算机主板上常用的元件驱动的。因此，这些开发板的架构更复杂，涉及一个或多个小型化的片上系统(SoC)以及 HDMI、音频或以太网等附加的互连设备。

虽然单板机可能具有板载物理 I/O 接口，如 Raspberry Pi，但它们的通用处理器可以很容易地用于支持不以硬件为中心的项目。

1.4.2 输入和输出元件

有很多传感器和小装置可以用来连接到开发板，从而增强项目的性能。这会带来各种乐趣，但是一开始你也会感到不知所措。本书有大量的技术术语，以及很多数字、数值和规格，等着你去慢慢消化和吸收。

我们将用到的大部分输入和输出元件都设计得很简单，在面包板上是即插即用的(这意味着它们是面包板友好的)，有些被组装成扩展板。通过将微控制器的微型引脚连接到更方便的连接件，开发板使 I/O 传输更容易，同样，将引脚连接到更方便的连接件，扩展板可以更轻松地使用单用途传感器或输出设备(见图 1.21)。

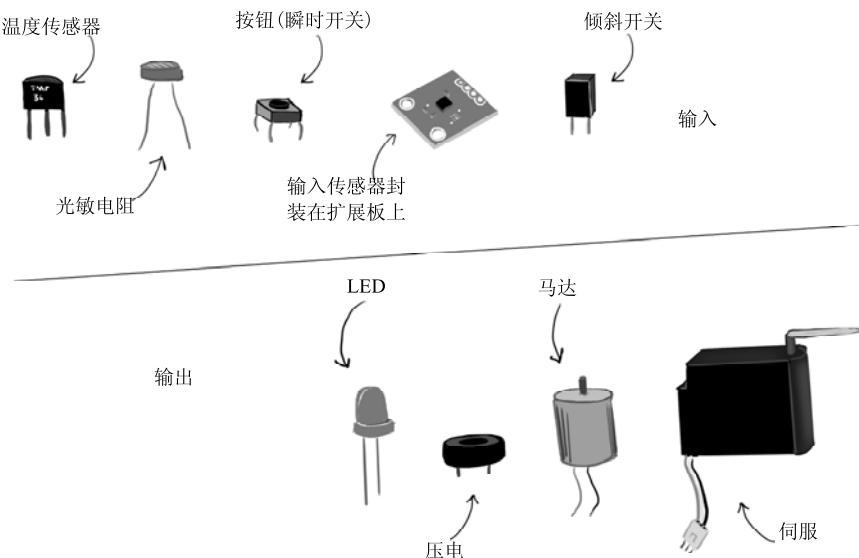


图 1.21 各种常见的输入和输出元件

1.4.3 其他电子元件

将电子电路组装在一起需要一系列的辅助电子元件。

虽然各种小元件很多，但电阻、电容、二极管和晶体管等基本元件都不贵，很容易在入门套件中买到(见图 1.22)。我们将会花些时间来介绍这些，很快它们就会成为你的老朋友。

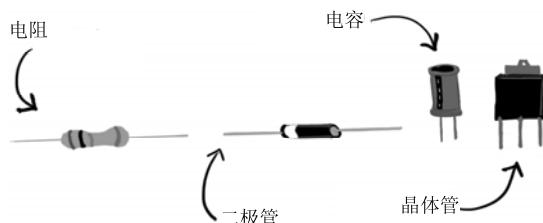


图 1.22 这样的常用元件可以帮助构建实用的电子电路

1.4.4 电源、电线和附件

你很快就会意识到，为项目供电的方法有很多。

开发板可以通过 USB 供电，也可以插入一个直流(DC)适配器(壁式)。很多情况下，其他项目元件也可以使用这些电源(见图 1.23)。

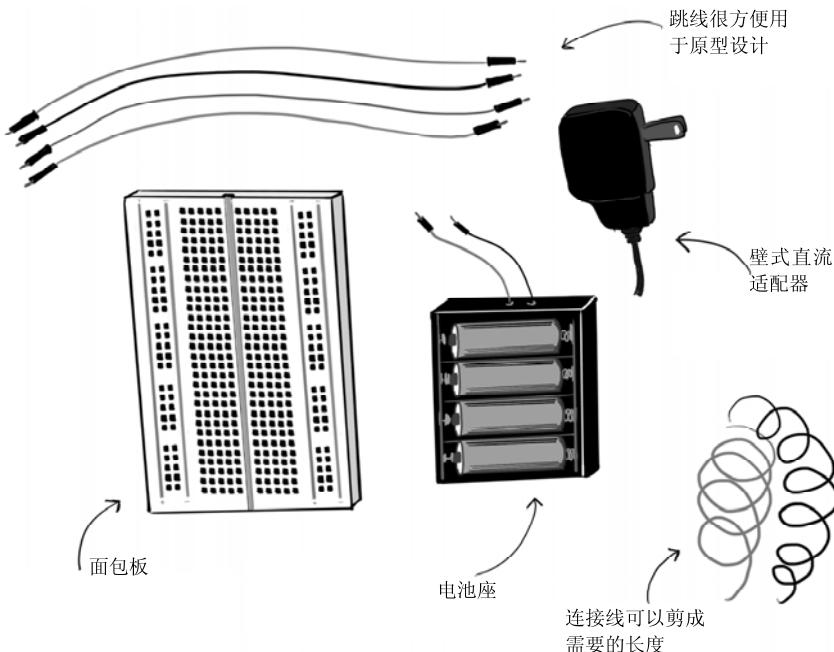


图 1.23 用于电源和电路的电线和配件

电池可以让项目不用插线，也可以在不同的电压下提供额外的功率。有很多种电池卡扣和支架可用于将电池连接到项目。

要让这些器件都连接在一起，需要导线。跳线是预切割线。跳线中有一种类型特别方便，其在每一端都有引脚，很容易插入面包板中，也有 I/O 引脚可以插入其他开发板中。连接线特别适合快速搭建原型。此外，单连线通常是卷在线轴上的，可以按照需要剪成合适的长度。

1.4.5 工具

在构建项目时，一对尖嘴钳和一两个精密螺丝刀都会很有用。还需要一对剥线钳，通常有内置的钢丝钳，这在切割或剥离连接线时会用得上(预割连接线不需要切割或剥离)。随着项目的进展，可能会用到万用表，这是一种测量电压、电流和电阻的工具。

存储电子元件

开始构建项目时储备电子元件，最终会得到很多小部件。你可以在硬件和业余爱好者商店找到分格的储物盒或抽屉柜。这些设计用来放置鱼饵的盒子和箱子，由于它

们的格子都很小，并且分隔板贴合地很紧密，因此特别适合用来收纳电子零件(见图 1.24)。

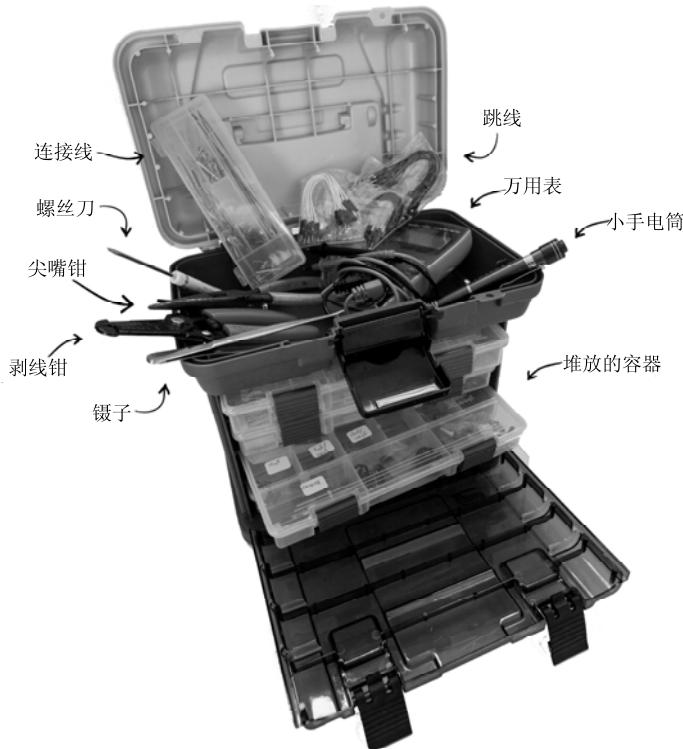


图 1.24 这种紧凑的工具箱顶部有放置工具的空间，可存放堆放的可拆卸容器中的部件

该开始我们的旅程了。使用低功耗嵌入式硬件构建项目既有趣又有创意，令人兴奋，它在商业领域中的用途也越来越多。Web 开发者已经拥有的技能，可以成为开发道路上的垫脚石。你可以使用无处不在的网络语言 JavaScript 行动起来以减少干扰。

在我们的探险中，你将了解使电子电路工作的几个基本关系。数学很糟糕？别担心，我的数学也很糟糕。在这个过程中你会遇到一些有用的角色：元件和模块、不同类型的开发板和软件。我们将尝试不同的组合，并学习如何重温这些知识，并在我们点亮 LED 时再试一次。

本书不会讲解所有内容，但在本书的最后，你将准备评估和使用未来技术。在旅行的中途，可能出发时走的路已经发生了显著变化。但依靠一些不变的事物作为指南针，如硬件基础知识、JavaScript 应用程序和网络技术，最终会到达目标。

1.5 本章小结

- 从头构建一个嵌入式电子产品不容易，但是所具备的 JavaScript 技能对你大有裨益。
- 嵌入式系统将大脑——微控制器或功耗低的处理器——与输入和输出组合在一个小型封装包中。
- 微控制器将处理器、存储器和 I/O 组合在一个芯片中。逻辑(也称为固件)定义微控制器的行为，通常写入微控制单元的程序存储器中。
- JavaScript 控制硬件的方法有几种：宿主机-客户端模式、嵌入式 JavaScript、在单板机上运行 Node.js，甚至是在浏览器中控制硬件等。
- 在宿主机-客户端设置中，Node.js 在主机运行，并且使用消息传递协议(API)与微控制器交换指令和数据。项目没有主机不能工作。
- 一些受限的微控制器通过优化，可以直接在芯片上运行 JavaScript 或 JavaScript 的子集(嵌入式 JavaScript)。
- 单板机(SBC)拥有更复杂的处理器和附加功能，如 USB 端口或音频连接。这些设备通常可以运行完整的操作系统，并且其表现和微型计算机一样。其中很多提供了使用高级语言(如 Python、C++ 和 JavaScript)来控制 I/O 和行为。
- 开发板是将微控制器(或其他处理组件)与便捷的支持功能相结合的平台。它们提供了与 I/O 引脚的便捷连接，可以快速进行项目原型设计。
- 构建项目涉及一些电子设备：包括开发板、输入和输出组件、电阻和二极管等基础电子元件、电源连接和基础工具。

第2章

用 Arduino 开启硬件之旅

本章内容包含：

- Arduino 是什么以及 Arduino Uno R3 开发板的特性
- 将元件和电源连接到 Arduino Uno
- 使用 Arduino IDE 编程并上传草图让 LED 闪烁
- 使用 Firmata 固件和 Johnny-Five Node.js 框架在宿主机-客户端设置中配置 Arduino Uno
- 使用 JavaScript 控制 Arduino Uno 并使 LED 闪烁

Arduino 是一家公司，一个项目，一种硬件，也是一个用户社区。Arduino 就是一个结合了开源硬件和软件的广泛概念，旨在让初学者轻松(而且便宜)地构建交互式设备。

像大多数开发板一样，Arduino 主板拥有微处理器、I/O 引脚、电源连接和其他标准功能。目前有十几种开发板型号，包括图 2.1 中所示的 Uno。每个 Arduino 开发板都有标准的尺寸和布局，可以使用模块化的屏蔽。制造的屏蔽符合 Arduino 的形状，并提供附加功能，如 Wi-Fi 或 GPS，这些功能并不是由开发板本身提供的(分线板是扩展开发板功能的另一种方式，但屏蔽是专门针对 Arduino 的外形而定制的)。