

<<Node.js高级编程>>

图书基本信息

书名：<<Node.js高级编程>>

13位ISBN编号：9787302344414

10位ISBN编号：7302344418

出版时间：2013-12-1

出版时间：清华大学出版社

作者：Pedro Teixeira

译者：胡训强,张欣景

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<Node.js高级编程>>

内容概要

Node.js是一种主流框架，它允许你使用JavaScript快速构建具有高度可伸缩性的网络程序。可是，它有自己的学习曲线，这本较为深入的指南性图书首先介绍了Node.js平台的安装，然后重点关注：创建和加载模块；使用缓冲区对二进制数据进行处理、编码和编码；用事件发射器模式简化事件绑定等。本书还涵盖了从使用定时器制定函数执行计划到创建和控制外部进程等内容，可以让你为运行、构建和测试自定义模块做好准备。

主要内容

介绍了查询和读写文件

研究了流、文件系统、网络和自动化单元测试

详细描述了如何读写数据流

从零开始构建TCP和HTTP服务器与客户端

介绍使用数据报、测试和调试模块以及控制回调流程

展示了如何用Connect、Express和Socket.IO开发实时Web应用程序

引导你连接到MySQL、CouchDB和MongoDB数据库

主要内容

介绍了实现位置跟踪和接近警报的多种方法

揭秘SensorManager API中物理传感器背后的物理原理，从而让你了解正确的应用方式

展示了算法代码来解释带噪声的传感器数据并检测其变化

演示了如何测量设备属性，例如方向和移动，以及类似相对高度这样的环境属性

探究如何使用Android开放附件(Android Open Accessory, AOA)来访问外部传感器

介绍近场通信(NFC)技术及其API

提供图像和信号处理代码来检测摄像头和麦克风所捕获到的内容

给出使用Android语音识别和文本转语音API来创建可靠、用户友好、带语音功能的应用所需的所有组件

作者简介

Pedro Teixeira是一位高产的开源项目程序员，同时也是众多Node.js模块的构建者。他是Node公司的创始人之一，同时也是Nodejitsu公司的高级程序员，Nodejitsu公司是以Node.js平台作为服务的领头羊。

自从在14年前获得软件工程学位后，他从事过的职业包括咨询师和程序员，并且他还是世界知名的Node.js社区活跃成员。此外，他还是广受欢迎的Node Tuts视频的制作人。

<<Node.js高级编程>>

书籍目录

第 部分 概述和安装

第1章 安装Node 3

- 1.1 在Windows上安装Node 4
- 1.2 在MAC OS X上安装Node 6
- 1.3 使用源代码安装Node 7
 - 1.3.1 选择Node的版本 7
 - 1.3.2 下载Node源代码 7
 - 1.3.3 编译Node 8
 - 1.3.4 安装Node 8
 - 1.3.5 运行Node 8
- 1.4 安装和应用Node包管理器 9
- 1.5 本章小结 14

第2章 Node简介 15

- 2.1 事件驱动编程风格介绍 16
- 2.2 Node和JavaScript如何简化异步应用程序的编写 17
 - 2.2.1 什么是闭包 17
 - 2.2.2 闭包如何辅助异步编程 18
- 2.3 本章小结 19

第 部分 Node核心API基础

第3章 加载模块 23

- 3.1 理解Node如何加载模块 24
- 3.2 导出模块 24
- 3.3 加载模块 25
 - 3.3.1 加载核心模块 25
 - 3.3.2 加载文件模块 26
 - 3.3.3 加载文件夹模块 26
 - 3.3.4 从node_modules文件夹加载 26
 - 3.3.5 缓存模块 27
- 3.4 本章小结 28

第4章 应用缓冲区处理、编码和解码二进制数据 29

- 4.1 创建缓冲区 30
- 4.2 在缓冲区中获取和设置数据 30
- 4.3 切分缓冲区 31
- 4.4 复制缓冲区 32
- 4.5 缓冲区解码 32
- 4.6 本章小结 33

第5章 使用事件发射器模式简化事件绑定 35

- 5.1 理解标准回调模式 35
- 5.2 理解事件发射器模式 36
- 5.3 理解事件类型 37
- 5.4 应用事件发生器API 38
 - 5.4.1 使用.addListener()或.on()绑定回调函数 38
 - 5.4.2 绑定多个事件监听器 39
 - 5.4.3 使用.removeListener()从事件发射器中删除一个事件监听器 40
 - 5.4.4 使用.once()使回调函数最多执行一次 40

<<Node.js高级编程>>

- 5.4.5 使用.removeAllListeners()从事件发射器删除所有事件监听器 41
- 5.5 创建事件发射器 41
 - 5.5.1 从Node事件发射器继承 41
 - 5.5.2 发射事件 42
- 5.6 本章小结 42
- 第6章 使用定时器制定函数执行计划 45
 - 6.1 使用setTimeout推迟函数执行 46
 - 6.2 使用clearTimeout取消函数执行 46
 - 6.3 制定和取消函数的重复执行计划 47
 - 6.4 使用process.nextTick将函数执行推迟到下一轮事件循环 47
 - 6.5 阻塞事件循环 48
 - 6.6 退出事件循环 49
 - 6.7 使用setTimeout代替setInterval强制函数串行执行 49
 - 6.8 本章小结 50
- 第 部分 文件、进程、流和网络
- 第7章 查询和读写文件 53
 - 7.1 处理文件路径 54
 - 7.1.1 规范化路径 54
 - 7.1.2 连接路径 54
 - 7.1.3 解析路径 55
 - 7.1.4 查找两个绝对路径之间的相对路径 55
 - 7.1.5 提取路径的组成部分 55
 - 7.1.6 确定路径是否存在 56
 - 7.2 fs模块简介 57
 - 7.3 打开文件 58
 - 7.4 读取文件 58
 - 7.4.1 写入文件 59
 - 7.4.2 关闭文件 60
 - 7.5 本章小结 62
- 第8章 创建和控制外部进程 63
 - 8.1 执行外部命令 63
 - 8.2 生成子进程 68
 - 8.2.1 创建子进程 68
 - 8.2.2 监听子进程的输出数据 69
 - 8.2.3 向子进程发送数据 69
 - 8.2.4 当子进程退出时获得通知 71
 - 8.3 向进程发送信号并终止进程 72
 - 8.4 本章小结 73
- 第9章 读写数据流 75
 - 9.1 使用可读流 76
 - 9.1.1 等待数据 76
 - 9.1.2 暂停与恢复流 76
 - 9.1.3 了解流何时终止 77
 - 9.2 使用可写流 77
 - 9.2.1 将数据写入流 77
 - 9.2.2 等待流被清空 78
 - 9.3 考虑几个流的例子 78

<<Node.js高级编程>>

- 9.3.1 创建文件系统流 78
- 9.3.2 理解网络流 80
- 9.4 避免慢客户端问题以及挽救服务器 80
 - 9.4.1 理解慢客户端问题 80
 - 9.4.2 避免慢客户端问题 81
 - 9.4.3 应用stream.pipe()避免慢客户端问题与使用pipe()集成可读流和可写流 81
- 9.5 本章小结 82
- 第10章 构建TCP服务器 83
 - 10.1 创建TCP服务器 83
 - 10.1.1 应用套接字对象 85
 - 10.1.2 理解空闲套接字 86
 - 10.1.3 设置保持运行 87
 - 10.1.4 应用延时或非延时 87
 - 10.1.5 监听连接 88
 - 10.1.6 关闭服务器 88
 - 10.1.7 处理错误 88
 - 10.2 构建一个简单的TCP聊天服务器 89
 - 10.2.1 接受连接 89
 - 10.2.2 从连接中读取数据 89
 - 10.2.3 聚合所有客户端 90
 - 10.2.4 广播数据 91
 - 10.2.5 删除被关闭的连接 91
 - 10.2.6 使用TCP聊天服务器 92
 - 10.3 本章小结 93
- 第11章 构建HTTP服务器 95
 - 11.1 理解http.ServerRequest对象 96
 - 11.2 理解http.ServerResponse对象 98
 - 11.2.1 写入响应头 98
 - 11.2.2 修改或设置响应头 98
 - 11.2.3 删除响应头 99
 - 11.2.4 写入一块响应主体 99
 - 11.3 以流的形式传送HTTP分块响应 99
 - 11.3.1 传送文件 99
 - 11.3.2 传送其他进程的输出 100
 - 11.4 关闭服务器 100
 - 11.5 示例1：构建提交静态文件的服务器 101
 - 11.6 示例2：使用HTTP分块响应和定时器 102
 - 11.7 本章小结 102
- 第12章 构建TCP客户端 103
 - 12.1 连接服务器 104
 - 12.2 发送和接收数据 104
 - 12.3 终止连接 105
 - 12.4 处理错误 106
 - 12.5 创建命令行TCP客户端的示例 106
 - 12.5.1 连接服务器 107
 - 12.5.2 向服务器发送命令行 107
 - 12.5.3 打印服务器消息 107

<<Node.js高级编程>>

- 12.5.4 在连接终止时重新连接 108
- 12.5.5 关闭连接 110
- 12.5.6 前述内容综合 111
- 12.6 本章小结 112
- 第13章 创建HTTP请求 113
- 13.1 创建GET请求 113
- 13.2 使用其他HTTP动词 114
- 13.2.1 查看响应对象 115
- 13.2.2 获取响应主体 116
- 13.2.3 以流的方式传送响应主体 116
- 13.3 使用HTTP.Agent维护套接字池 116
- 13.4 应用第三方请求模块简化HTTP请求 118
- 13.4.1 安装和应用request模块 118
- 13.4.2 创建测试服务器 119
- 13.4.3 跟随重定向 121
- 13.4.4 设置请求选项 122
- 13.4.5 对请求体进行编码 125
- 13.4.6 流式传送 126
- 13.4.7 使用Cookie Jar 127
- 13.5 本章小结 127
- 第14章 使用用户数据报 129
- 14.1 理解用户数据报 129
- 14.2 理解用户数据报的使用 130
- 14.3 构建数据报服务器 130
- 14.3.1 监听消息 130
- 14.3.2 测试服务器 131
- 14.3.3 查看附加的消息信息 132
- 14.4 创建简单的数据报回送服务器 132
- 14.4.1 等待消息 132
- 14.4.2 向发送端发回消息 132
- 14.4.3 前述内容综合 133
- 14.5 构建数据报客户端 133
- 14.5.1 创建客户端 134
- 14.5.2 发送消息 134
- 14.5.3 关闭套接字 134
- 14.6 创建一个简单的数据报命令行客户端 134
- 14.6.1 读取命令行 135
- 14.6.2 向服务器发送数据 135
- 14.6.3 从服务器接收数据 135
- 14.6.4 前述内容综合 136
- 14.7 理解和使用数据报多播 136
- 14.7.1 接收多播消息 137
- 14.7.2 发送多播消息 137
- 14.7.3 理解数据报最大容量 138
- 14.8 本章小结 138
- 第15章 用TLS/SSL保证服务器的安全性 139
- 15.1 理解私钥和公钥 139

<<Node.js高级编程>>

- 15.1.1 产生私钥 140
- 15.1.2 产生公钥 140
- 15.2 构建TLS服务器 140
 - 15.2.1 初始化服务器 141
 - 15.2.2 监听连接 141
 - 15.2.3 从客户端读取数据 142
 - 15.2.4 向客户端发送数据 142
 - 15.2.5 终止连接 142
- 15.3 构建TLS客户端 142
 - 15.3.1 初始化客户端 143
 - 15.3.2 连接服务器 143
 - 15.3.3 验证服务器证书 143
 - 15.3.4 向服务器发送数据 144
 - 15.3.5 从服务器读取数据 144
 - 15.3.6 终止连接 144
- 15.4 创建几个示例 144
 - 15.4.1 创建TLS聊天服务器 145
 - 15.4.2 创建TLS命令行聊天客户端 146
 - 15.4.3 验证客户端证书 147
- 15.5 本章小结 148
- 第16章 用HTTPS保证HTTP服务器的安全性 149
 - 16.1 构建安全的HTTP服务器 149
 - 16.1.1 设置服务器选项 150
 - 16.1.2 监听连接 150
 - 16.1.3 验证HTTPS客户端证书 151
 - 16.2 创建HTTPS客户端 152
 - 16.2.1 初始化客户端 152
 - 16.2.2 创建请求 152
 - 16.2.3 验证HTTPS服务器证书 153
 - 16.3 本章小结 154
- 第 部分 构建与调试模块及应用程序
- 第17章 测试模块及应用程序 157
 - 17.1 应用测试运行工具 157
 - 17.1.1 编写测试 158
 - 17.1.2 运行测试 159
 - 17.2 使用断言测试模块 159
 - 17.2.1 使用断言模块 159
 - 17.2.2 使用Node-Tap中的内置断言函数 161
 - 17.3 测试异步模块 163
 - 17.4 本章小结 166
- 第18章 调试模块及应用程序 167
 - 18.1 使用console.log 167
 - 18.2 使用Node内置调试器 169
 - 18.3 使用Node检查器 173
 - 18.4 本章小结 175
- 第19章 控制回调流程 177
 - 19.1 理解飞去来器效应 177

<<Node.js高级编程>>

- 19.2 通过声明函数避免飞去来器效应 179
- 19.3 使用ASYNC流程控制库 183
 - 19.3.1 串行执行 184
 - 19.3.2 并行执行 185
 - 19.3.3 连续传递 186
 - 19.3.4 排队 187
 - 19.3.5 迭代 189
 - 19.3.6 映射 190
 - 19.3.7 规约 191
 - 19.3.8 过滤 192
 - 19.3.9 检测 193
- 19.4 本章小结 194
- 第 部分 构建Web应用程序
- 第20章 构建和使用HTTP中间件 197
 - 20.1 理解Connect HTTP中间件框架 198
 - 20.2 构建自定义HTTP中间件 198
 - 20.2.1 创建异步中间件 199
 - 20.2.2 在中间件内部注册回调函数 201
 - 20.2.3 在中间件内处理错误 202
 - 20.3 使用捆绑在Connect中的HTTP中间件 206
 - 20.3.1 记录请求 206
 - 20.3.2 处理错误 208
 - 20.3.3 提交静态文件 209
 - 20.3.4 解析查询字符串 210
 - 20.3.5 解析请求主体 211
 - 20.3.6 解析Cookies 212
 - 20.3.7 使用会话 213
 - 20.3.8 其他可用的中间件 216
 - 20.4 本章小结 216
- 第21章 用Express.js创建Web应用程序 217
 - 21.1 初始化Express.js应用程序 218
 - 21.2 在应用程序中设置中间件 220
 - 21.3 路由请求 222
 - 21.3.1 处理路由 222
 - 21.3.2 使用会话 229
 - 21.3.3 使用路由中间件 234
 - 21.4 本章小结 238
- 第22章 使用Socket.IO创建通用的实时Web应用程序 241
 - 22.1 理解WebSockets如何工作 242
 - 22.2 使用Socket.IO创建WebSocket应用程序 243
 - 22.2.1 在服务器上安装和运行Socket.IO 243
 - 22.2.2 使用Socket.IO创建实时网络聊天应用程序 245
 - 22.2.3 扩展聊天应用程序 250
 - 22.2.4 检测连接断开 253
 - 22.2.5 将用户分隔到聊天室中 255
 - 22.2.6 使用名称空间 259
 - 22.2.7 使用Redis分布运行服务器端应用程序 260

<<Node.js高级编程>>

- 22.3 本章小结 263
- 第 部分 连接数据库
- 第23章 使用node-mysql连接MySQL数据库 267
 - 23.1 应用库与MySQL数据库进行连接和通信 267
 - 23.2 向数据库添加数据时请记住安全性 269
 - 23.3 高效读取数据 272
 - 23.4 本章小结 276
- 第24章 使用Nano连接CouchDB数据库 277
 - 24.1 安装Nano 278
 - 24.2 连接和创建数据库 281
 - 24.3 存储文档 285
 - 24.4 创建和使用CouchDB视图 286
 - 24.5 将文件附加到CouchDB文档上 299
 - 24.6 本章小结 312
- 第25章 使用Mongoose连接MongoDB 数据库 313
 - 25.1 安装Mongoose 315
 - 25.2 理解Mongoose如何使用模型封装对数据库的访问 315
 - 25.3 连接MongoDB数据库 316
 - 25.4 定义模式 316
 - 25.5 定义模型 316
 - 25.5.1 使用验证器 326
 - 25.5.2 使用修改器 332
 - 25.5.3 使用取值器 333
 - 25.5.4 使用虚拟属性 334
 - 25.5.5 使用默认值 340
 - 25.5.6 定义索引 341
 - 25.5.7 使用DB Refs引用其他文档 343
 - 25.5.8 定义实例方法 349
 - 25.5.9 定义静态方法 350
 - 25.6 本章小结 351

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>