

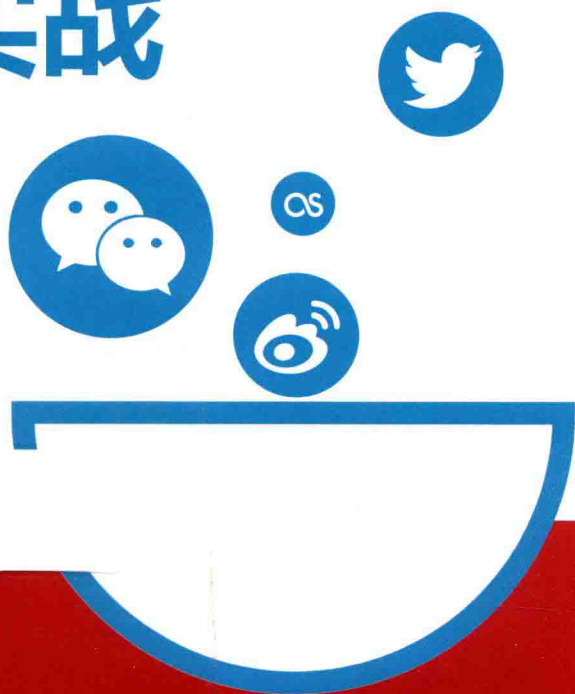
详解React Native应用从创建、开发到发布的全过程，展示各组件和API的用法

实战为王，通过典型项目案例，让读者快速掌握React Native应用开发

# React Native

## 移动开发实战

袁林◎编著



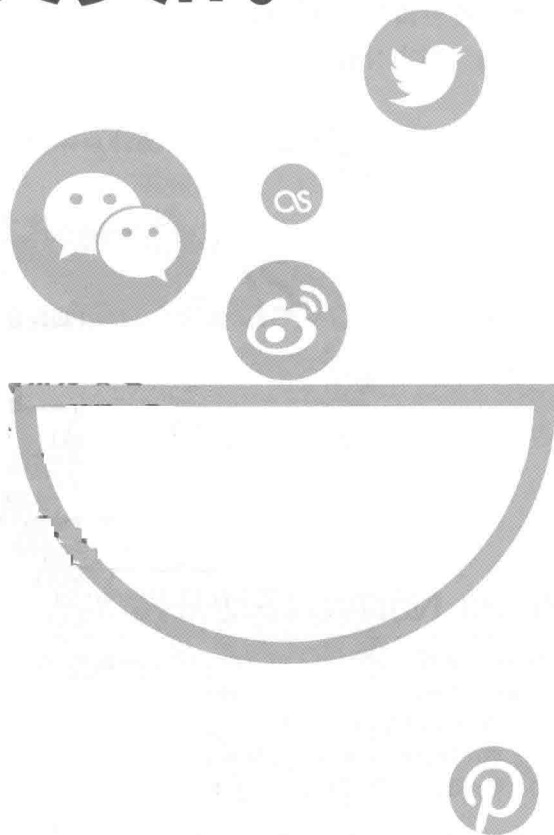
- 书中所有内容都配合详细的实例和源代码进行讲解
- 全面涵盖React Native组件、API、布局、第三方组件及原生接口开发等内容
- 详解React Native的开发工具、命令行工具及各种调试工具的使用
- 详细讲解一个电商App项目案例的开发过程，提高读者的实战开发水平
- 涉及软件开发流程、应用架构设计、代码重构，以及原生平台与跨平台开发等



# React Native

## 移动开发实战

袁林◎编著



机械工业出版社  
China Machine Press

## 图书在版编目 (CIP) 数据

React Native 移动开发实战/袁林编著. —北京: 机械工业出版社, 2017.6

ISBN 978-7-111-57179-7

I. ①R… II. ①袁… III. ①移动终端—应用程序—程序设计 IV. ①TN929.53

中国版本图书馆 CIP 数据核字 (2017) 第 146546 号

## React Native 移动开发实战

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 欧振旭

责任校对: 姚志娟

印 刷: 中国电影出版社印刷厂

版 次: 2017 年 7 月第 1 版第 1 次印刷

开 本: 185mm × 260mm 1/16

印 张: 20

书 号: ISBN 978-7-111-57179-7

定 价: 69.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88379426 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

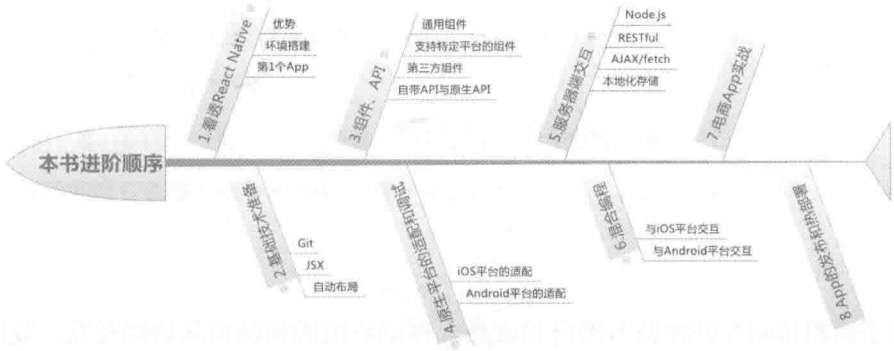
读者信箱: hzit@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光/邹晓东





## 本书特色

### 1. 每一步都有详细的源码和实例参考

为了便于读者理解本书内容，提高学习效率，本书的所有内容都有详细的源码和实例参考。对于这些源码和实例，作者均亲自编写和验证，杜绝复制、粘贴代码以敷衍读者的不负责任行为。本书源码可以在 <https://coding.net/u/learnreactnative/p/learnreactnative-sourcecode/git> 里下载。

### 2. 内容涵盖React Native开发的各个方面

本书涵盖 React Native 组件、API、布局、第三方组件以及原生接口等 React Native 应用开发的各个方面，尽量保证不出现知识“死角”。凡是涉及的一些技术（如原生、瀑布流、耦合性和 JSON），也给出了概念或原理解释。

### 3. 结合工具助力更高效的React Native开发

在本书“实战”讲解的过程中，详细介绍了 React Native 开发工具 Nuclide 的使用、React Native 命令行工具的用法及各种调试工具（包括布局、断点及实时加载等）的使用，不仅教读者如何开发，还教读者如何更高效地开发。

### 4. 项目案例典型，实战性强，有较高的应用价值

本书以开发一个电商类应用为例，涵盖了 React Native 应用开发中会用到的所有重点知识，设计和源码做到拿来可用，方便应用开发者随时查阅和参考。

### 5. 收获的不仅仅是React Native平台和编码

对于一些学习能力较强的读者，完全可以在 React Native 开发文档的帮助下快速学习和掌握 React Native。而本书希望读者在掌握平台和编码之外，还能够了解实际应用开发过程中涉及的软件开发流程、应用架构设计、代码重构技巧，以及原生平台与其他跨平台开发的相关知识，让读者融会贯通地理解应用开发技术。

## 本书内容及知识体系

### 第1篇 React Native入门和基础（第1~2章）

本篇介绍了跨平台开发的主流方案和 React Native 基础知识，主要包括开发环境搭建、

React Native 命令行工具和 React Native 布局调试。

### 第2篇 React Native应用开发实战（第3~7章）

本篇介绍了 React Native 实际应用开发中常用的技术，主要包括基本组件、使用第三方组件、搭建基于 Node.js 的服务器为应用绑定真实数据、fetch API、AsyncStorage/SQLite/Realm 数据库存储、更多 React Native 组件和 API 的用法、原生平台接口开发等。

### 第3篇 React Native混合编程（第8~10章）

本篇主要总结和回顾了前 7 章所开发的电商类应用的技术和架构，主要包括应用的文件结构、Flexbox 的整体布局、应用的逻辑结构、应用的通信过程及进一步改进的地方和思路，其中就包括了 redux 开发框架。

### 第4篇 App的发布和更新（第11~12章）

本篇主要介绍了 React Native 应用打包和发布的全过程，配以详细的截图说明，并且对 React Native 应用发布后的热更新实现和方案 CodePush 做了详细的示例说明。

## 适合阅读本书的读者

- React Native 学习人员；
- iOS 平台应用开发工程师；
- Android 平台应用开发工程师；
- Web 前端开发工程师；
- Node.js 服务端开发工程师；
- 计算机相关专业的学生；
- 专业培训机构的学员；
- 软件开发项目经理。

## 本书作者

本书由袁林主笔编写。其他参与编写的人员有高旭、贺庆端、黎林、李伟浩、刘成智、刘利、刘源、谭建利、吴贵文、吴娟、夏秀英、肖太平、郑星。

## 致谢

感谢本书的编辑，让我有机会和本书结缘。感谢我的伙伴们：邵长磊、刘冬冬、袁方、袁满、翟绍虎、洪敏、郭晨光及张砚，与我一起探讨新技术并和 React Native 结缘。感谢我的妻子韩丽、女儿可可及我的父母，写作占用了我很多陪伴家人的时间和精力，正是有了家人的支持，才得以坚持下去。

最后还要感谢读者，本书的价值因你们而存在。

编者

# 目录

前言

## 第 1 篇 React Native 入门和基础

第 1 章 为什么要学习 React Native .....	2
1.1 看透 React Native .....	2
1.1.1 React Native 与 React.js .....	2
1.1.2 React Native 的跨平台 .....	3
1.1.3 解剖 React Native 应用的结构 .....	4
1.2 React Native 的特点 .....	5
1.2.1 其一：Learn Once, Write Anywhere .....	5
1.2.2 其二：简单易学的开发语言 .....	6
1.2.3 其三：接近原生应用的性能和体验 .....	7
1.2.4 其四：完善的生态系统 .....	7
1.3 搭建 React Native 开发环境 .....	9
1.3.1 安装原生开发工具——Android .....	9
1.3.2 安装原生开发工具——iOS .....	11
1.3.3 安装 Node.js .....	12
1.3.4 安装 React Native .....	13
1.3.5 安装其他辅助工具 .....	14
1.4 第一个 React Native 应用 .....	16
1.4.1 初始化项目 .....	16
1.4.2 运行项目 .....	17
1.4.3 调试项目 .....	18
1.5 小试牛刀——更改 React Native 项目源码 .....	18
1.6 小结 .....	20
第 2 章 全局解析 React Native 开发的基础技术 .....	21
2.1 开发具备的基础知识说明 .....	21
2.2 Git 版本控制工具 .....	22
2.2.1 安装 Git .....	22
2.2.2 Git 常用命令 .....	22
2.3 React Native 的 JSX 解决方案 .....	24
2.4 React Native 的 Flexbox 布局 .....	25
2.4.1 flexDirection 设置组件的排列 .....	26

2.4.2	flexWrap 设置是否换行	28
2.4.3	justifyContent 设置横向排列位置	30
2.4.4	alignItems 设置纵向排列位置	31
2.4.5	alignSelf 设置特定组件的排列	33
2.4.6	flex 设置组件所占空间	34
2.5	如何调试 React Native 项目	35
2.6	实战——设计一个电商 App	37
2.6.1	电商 App 的模块划分	37
2.6.2	设计首页布局	41
2.6.3	实现搜索栏	44
2.6.4	设计轮播广告	46
2.6.5	展示商品列表	51
2.6.6	实现交互功能和状态栏	52
2.7	小结	56

## 第 2 篇 React Native 应用开发实战

第 3 章	React Native 的组件 (1)	58
3.1	创建新的电商 App	58
3.1.1	移植旧电商项目	58
3.1.2	重构现有的代码	60
3.2	完善搜索框功能——TextInput 组件	64
3.2.1	搜索提示框	64
3.2.2	调试搜索结果	66
3.2.3	优化搜索框样式	67
3.3	完善轮播广告——Image 组件	68
3.3.1	使用网络图片	68
3.3.2	使用本地图片	69
3.3.3	添加指示器组件	71
3.4	完善商品列表——ListView 组件	73
3.4.1	对图片资源进行重构	74
3.4.2	重新定义商品模型	75
3.4.3	商品布局的优化	76
3.5	拖曳刷新列表——RefreshControl 组件	80
3.6	添加页面跳转功能——Navigator 组件	83
3.7	二级页面的跳转——TouchableOpacity 组件	86
3.8	实现页面间的数据传递	89
3.9	小结	90
第 4 章	React Native 的组件 (2)	91
4.1	只支持特定平台的组件	91
4.1.1	实现多页面分页 TabBarIOS/ViewPagerAndroid	91
4.1.2	加载指示器——ActivityIndicator	96



4.1.3	地图——MapView	97
4.1.4	渲染——Picker	98
4.1.5	选择范围——Slider	99
4.1.6	开关组件——Switch	100
4.1.7	打开网页——WebView	101
4.2	第三方组件	102
4.2.1	react-native-swiper 的使用	103
4.2.2	NativeBase 的使用	104
4.2.3	NativeBase 如何解决跨平台问题	111
4.3	小结	113
<b>第 5 章</b>	<b>原生平台的适配和调试</b>	<b>114</b>
5.1	iOS 平台的适配	114
5.1.1	Images.xcassets 适配	115
5.1.2	自动布局 Auto Layout	115
5.1.3	Size Class 适配	116
5.2	iOS 开发的调试技巧	117
5.3	Android 平台的适配	118
5.3.1	适配原理	118
5.3.2	常用的适配属性	119
5.4	Android 平台的调试技巧	122
5.5	小结	124
<b>第 6 章</b>	<b>React Native 的服务器端处理</b>	<b>125</b>
6.1	学习 Node.js	125
6.1.1	什么是 Node.js	125
6.1.2	为什么选择 Node.js	126
6.1.3	安装和使用 npm	128
6.1.4	Node.js 的开发流程	129
6.2	服务端接口的设计: RESTful	132
6.3	实现电商 App 的服务器端接口	133
6.3.1	Express 框架	133
6.3.2	查询商品接口	138
6.3.3	新建商品接口	142
6.3.4	更新商品接口	143
6.3.5	删除商品接口	144
6.4	网络前后端交互的原理 fetch	145
6.5	App 从服务器获取数据	146
6.5.1	获取商品信息	148
6.5.2	更新商品信息	151
6.5.3	新建商品	157
6.5.4	删除商品	158
6.6	App 数据的本地化存储	160
6.6.1	AsyncStorage 异步键值存储	160
6.6.2	SQLite 数据库	164
6.6.3	Realm 数据库	166

6.7 小结	168
<b>第 7 章 常用 React Native API</b>	<b>169</b>
7.1 屏幕设置相关 API	169
7.1.1 获取屏幕宽高——Dimensions API	170
7.1.2 获取屏幕分辨率——PixelRatio API	173
7.2 动画 API	174
7.2.1 RequestAnimationFrame API 帧动画	175
7.2.2 LayoutAnimation API 布局动画	177
7.2.3 Animated API 高级动画	179
7.3 组件、React Native API、原生平台 API	184
7.3.1 组件和 API	184
7.3.2 API 和原生平台 API	184
7.4 实现自己的 Platform API	185
7.4.1 支持 iOS 平台	186
7.4.2 支持 Android 平台	188
7.5 为应用添加更丰富的 API	189
7.5.1 提示框和编辑框——AlertIOS	190
7.5.2 前后台状态变化——AppState	193
7.5.3 Android 物理“返回键”——BackAndroid	195
7.5.4 日期和时间选择器——DatePickerAndroid/TimePickerAndroid	196
7.5.5 基于位置的 Geolocation	200
7.5.6 键盘事件——Keyboard	202
7.5.7 设备联网状态——NetInfo	204
7.5.8 权限设置——PermissionsAndroid	205
7.5.9 悬浮提示框——ToastAndroid	207
7.6 小结	208

## 第 3 篇 React Native 混合编程

<b>第 8 章 React Native 与原生平台混合编程 (1)</b>	<b>210</b>
8.1 创建并移植项目	210
8.2 访问设备	211
8.2.1 访问 iOS 设备	213
8.2.2 访问 Android 设备	214
8.3 访问相册	217
8.3.1 读取 iOS 相册中的图片	219
8.3.2 读取 Android 相册中的图片	224
8.4 React Native 与原生平台的通信原理	228
8.5 React Native 平台调用原生页面	229
8.5.1 React Native 平台调用原生 iOS 页面	231
8.5.2 React Native 平台调用原生 Android 页面	234
8.6 原生平台调用 React Native 组件	238

8.6.1	iOS 平台调用 React Native 组件 .....	238
8.6.2	Android 平台调用 React Native 组件 .....	239
8.7	小结 .....	240
<b>第 9 章</b>	<b>React Native 与原生平台混合编程 (2)</b> .....	<b>241</b>
9.1	使用相机拍摄图片 .....	241
9.1.1	使用 iOS 相机拍摄 .....	241
9.1.2	使用 Android 相机拍摄 .....	244
9.2	添加图片选择提示框 .....	247
9.2.1	iOS 平台的提示 .....	247
9.2.2	Android 平台的提示 .....	249
9.3	重构图片选择库 .....	251
9.3.1	iOS 平台的重构 .....	251
9.3.2	Android 平台的重构 .....	253
9.4	向 iOS 项目中添加 React Native 支持 .....	256
9.4.1	新建 iOS 项目 .....	256
9.4.2	新建 React Native 项目 .....	257
9.4.3	在 iOS 页面打开 React Native 组件 .....	259
9.5	向 Android 项目中添加 React Native 支持 .....	261
9.5.1	新建 Android 项目 .....	261
9.5.2	新建 React Native 项目 .....	261
9.5.3	在 Android 页面打开 React Native 组件 .....	262
9.6	小结 .....	264
<b>第 10 章</b>	<b>电商 App 的复盘</b> .....	<b>265</b>
10.1	电商 App 的文件 .....	265
10.1.1	JavaScript 文件 .....	266
10.1.2	iOS 原生代码文件 .....	266
10.1.3	Android 原生代码文件 .....	267
10.2	电商 App 的结构 .....	267
10.2.1	Flexbox 的整体布局 .....	268
10.2.2	应用的逻辑结构 .....	268
10.2.3	应用的通信过程 .....	269
10.3	优化和改进 .....	270
10.3.1	redux 是什么 .....	270
10.3.2	redux 代码示例 .....	271
10.3.3	redux 生态 .....	274
10.4	用到的组件 .....	275
10.5	小结 .....	276

## 第 4 篇 App 的发布和更新

<b>第 11 章</b>	<b>App 的发布</b> .....	<b>278</b>
11.1	App Store 苹果应用商店 .....	278

11.1.1	加入开发者计划	278
11.1.2	生成发布证书	280
11.1.3	注册 App ID	283
11.1.4	生成描述文件	283
11.1.5	打包应用	284
11.1.6	发布到 App Store	284
11.2	Android 应用商店	285
11.2.1	生成签名文件	285
11.2.2	打包应用	287
11.2.3	发布到应用商店	288
11.3	小结	289
<b>第 12 章</b>	<b>App 的热部署</b>	<b>290</b>
12.1	什么是热部署	290
12.2	解析 React Native 应用的工作原理	290
12.3	实现 React Native 的热部署	292
12.3.1	服务端实现	292
12.3.2	客户端实现	292
12.4	微软的热部署方案 CodePush	295
12.4.1	CodePush 简介	295
12.4.2	CodePush 安装和注册	295
12.4.3	集成 CodePush SDK	297
12.4.4	更改 iOS 应用	297
12.4.5	更改 Android 应用	301
12.5	小结	303
<b>附录 A</b>	<b>ES 6 语法</b>	<b>304</b>

# 第 1 篇

## React Native 入门和基础

▶▶ 第 1 章 为什么要学习 React Native

▶▶ 第 2 章 全局解析 React Native 开发的基础技术

# 第 1 章 为什么要学习 React Native

无论读者是移动平台开发者，还是 Web 前端开发者，想必对现在“大红大紫”的 React Native 都有所耳闻。那么，除了“乘着 Facebook 这棵大树好乘凉”的优势之外，React Native 到底是何方“神圣”，有什么令大家“趋之若鹜”的优点呢？下面带着这样的好奇，来随本书一探究竟吧！

本章主要内容有：

- React Native 与 React.js 的对比。
- 为什么说 React Native 是跨平台的。
- React Native 应用的结构。
- React Native 的特点。
- React Native 的环境搭建。
- 创建第一个 React Native 应用。

## 1.1 看透 React Native

React Native (<http://facebook.github.io/react-native/>) 第一次进入公众的视野是在 2015 年 1 月的 React.js Conf (<http://conf.reactjs.org/>) 上，随后，同年 5 月份，Facebook 在 F8 Conference (<https://www.fb8.com/>) 上正式宣布：React Native 项目（如图 1.1 所示）在 Github 开源。结果一天之内，就收获了 5000 多颗星，受关注程度可见一斑！

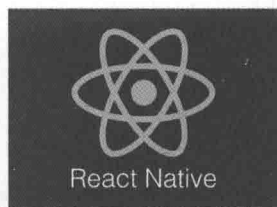


图 1.1 React Native Logo

**小知识：**React.js Conf 是指 Facebook 的 React 开发者大会，F8 Conference 是指 Facebook 的开发者大会，Github 是全球最大的软件项目托管平台，也被戏称为“人类的代码仓库”。

### 1.1.1 React Native 与 React.js

想必读者在还没弄清 React Native 之前，又发现了一个“新朋友” React.js（下文简称 React），那到底什么是 React 呢？它和 React Native 又是什么关系呢？

先来看看 Facebook 官方 (<https://code.facebook.com/projects/176988925806765/react/>) 对 React 的定义：

React is a JavaScript library for building user interfaces

从上述官方的定义可以知道：React 是一个用于前端 UI 开发的 JavaScript 库。和其他类似的前端框架相比，例如，老牌的 Backbone (<http://backbonejs.org/>)、Google 推出的 Angular (<https://angularjs.org/>) 和以轻量级著称的 Vue.js (<http://cn.vuejs.org/>)，React 最大的不同是提出了 Virtual DOM (即虚拟 DOM) 的设计，可以大大提升页面渲染的效率。

**小知识：**移动平台开发很好理解，即移动平台上（例如 Apple 的 iOS 平台，Google 的 Android 平台）的软件开发，开发语言和技术主要有 Objective C、Swift 及 Java 等。而前端开发是相对于后端（又称服务器端）开发而言的，前端主要负责开发通过浏览器和用户交互的部分，开发语言和技术主要有 HTML、CSS 及 JavaScript 等。

但是，Facebook 不仅仅满足于 React 对前端开发技术的革新，又将 React 的设计移植到原生开发中，从而诞生了 React + Native 结合的产物，即 React Native。

虽然，React Native 刚开始只支持 iOS App 开发，但是从 2015 年 9 月起，React Native 也支持 Android App 开发，而且随着微软、三星等“IT 大佬”的加入，React Native 还将支持更多的移动平台，例如，Samsung 的 Tizen 平台 (<https://www.tizen.org/>)、Microsoft 的 Window Phone (<http://microsoft.github.io/code-push/articles/ReactNativeWindows.html>)。

所以，简单来说：

- React Native 和 React 使用了相同的开发语言 JavaScript 和相同的设计理念 React。
- React Native 和 React 运行的环境和平台却是完全不同的，React Native 是基于移动平台（如 iOS、Android 等），而 React 是基于浏览器。

**提示：**国内网络环境下访问 React Native 官网 (<http://facebook.github.io/react-native/>) 可能较慢，读者可以访问国内的中文资源网站，例如 React Native 中文网 (<http://reactnative.cn/>)，或者自行搜索加快 React Native 官网访问速度的办法。

## 1.1.2 React Native 的跨平台

简单了解了 React Native 的由来之后，读者或许会有这样的疑问，开发移动平台 App 使用原生开发平台和语言就好了，为什么要出现使用 React Native 来开发移动平台 App 的技术呢？换句话说，React Native 到底可以解决什么问题呢？

在进一步讨论之前，笔者觉得有必要明确一下什么是原生应用和跨平台应用。

### 1. 原生应用

所谓的原生应用是指：使用原生开发语言、工具和平台开发的应用。原生应用开发的优势在于拥有较高的平台成熟度，包括平台的稳定性、运行时的性能及完善的生态。

**小知识：**所谓的“生态”应该算是比较抽象的概念，开发平台的生态圈包含了很多方面，从硬件上芯片和各种电子元器件的生产、供应，到软件上所使用的语言、开发工具及第三方开源库的数量质量，以及人的方面，如开发者的数量、水平等因素。

但是，原生应用开发也不是没有任何缺点，那就是开发成本较高，导致开发效率相对较低。例如，当一个产品需要支持多种类型的移动终端时，就需要熟悉多个原生平台开发的工程师。

## 2. 跨平台应用

为了解决产品满足多个平台的需求，就有了所谓的跨平台应用开发。根据实现跨平台方案的不同，也就有了以下几种常见的跨平台解决方案。

- 混合应用开发：在移动浏览器中嵌入 HTML 页面来开发移动应用，代表的有 Apache Cordova (<http://cordova.apache.org/>)，以及基于 Apache Cordova 衍生的 Inoic (<http://ionicframework.com/>) 等，如图 1.2 所示。
- 跨平台的语言：例如，基于 .NET 和 C# 的 Xamarin (<https://www.xamarin.com/>)，以及基于 Ruby 的 RubyMotion (<http://www.rubymotion.com/cn/>)，如图 1.3 所示。



图 1.2 Apache Cordova LOGO



图 1.3 Xamarin LOGO

- React Native: 使用的是 Web 开发语言 (JavaScript) 和环境 (Node.js)。除了本书介绍的 React Native 之外，类似的技术方案还有 NativeScript (<http://www.telerik.com/nativescript>)、Weex (<http://weex-project.io/>) 等，如图 1.4 所示。



图 1.4 Weex LOGO

**提示：**想要了解关于更多 React Native 的架构和原理，可以参考 1.1.3 节。

### 1.1.3 解剖 React Native 应用的结构

在了解完这么多关于 React Native 的故事和优势之后，让我们走近 React Native，来进一步了解 React Native 的原理和架构。

React Native 应用的整体结构如图 1.5 所示。

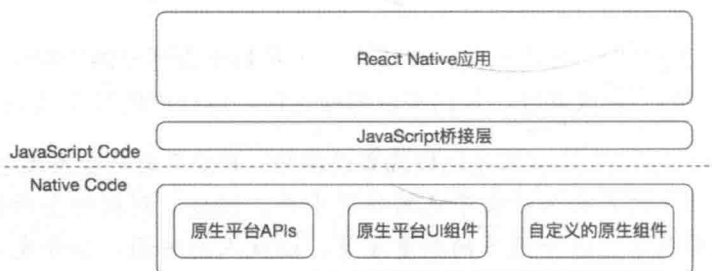


图 1.5 React Native 应用的整体结构



通过之前的介绍和图 1.5 可以看出：React Native 应用开发使用的是与 React 相同的开发语言 JavaScript 和设计思想 React，而底层仍然是基于原生平台的。这样，不同平台的适配就交由 React Native 平台去处理，而开发者只需要专注于 React Native 平台应用开发本身，体现出的优势如下。

- 应用层的开发变得简单、高效和跨平台。
- 应用稳定性、运行时的性能和原生平台接近。
- 在理解 React Native 原理之后，开发者也可以根据实际的产品需求开发自己的 React Native 组件，以复用已有原生平台的大量优秀组件。

## 1.2 React Native 的特点

那么，作为跨平台应用开发的“新贵”，React Native 相比其他跨平台技术到底有哪些优势呢？

### 1.2.1 其一：Learn Once, Write Anywhere

这句话是 React Native 官网 (<http://facebook.github.io/react-native/>) 对 React Native 的概述，简单明了地概括了 React Native 的最大特点和优点。

只需要学习 React Native 这一种开发方式（包括平台、语言和开发环境等）就可以开发多个不同平台的 App。

这句话简单来说就是 Learn Once, Write Anywhere，这也是 React Native 的宣传广告，如图 1.6 所示。



图 1.6 Learn Once, Write Anywhere 宣传广告

**小知识：**除了 React Native 提出的 Learn Once, Write Anywhere 的口号，Java 语言也提出过类似的口号 Write Once, Run Anywhere，两者看起来类似，但其实是完全不同的。React Native 就像上面介绍的，降低的是学习成本，针对不同平台可能还需要单独开发；而 Java 语言的意思是只需要开发一次，就可以成功运行在不同的平台和设备上。

目前，React Native 对 iOS、Android 平台的支持已经非常好了，在不远的将来，应该还会支持 Windows、Tizen 等更多的移动平台。

而且，React Native 的大多数组件也是可以在多个平台复用的，所以学习了 React Native 开发之后，完全可以胜任多个平台的开发任务且不需要很高的额外学习成本，大大降低了