HZ BOOKS
华章科技

本书由业界多位移动团队技术负责人联袂推荐，为打造高质量App提供了有价值的实践指导。
书中总结了80多个Crash的分析与处理，是迄今为止最完整的Android异常分析资料。
剖析了国内上百款知名App的前沿技术实现，是最权威的竞品技术分析白皮书。

移动开发

包建强◎著

App研发录

架构设计、Crash分析和竞品技术分析

机械工业出版社
China Machine Press

移动开发

App研发录

架构设计、Crash分析和竞品技术分析

包建强◎著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

App 研发录：架构设计、Crash 分析和竞品技术分析 / 包建强著 . —北京：机械工业出版社，2015.9
（移动开发）

ISBN 978-7-111-51638-5

I. A… II. 包… III. 移动电话机 – 应用程序 – 程序设计 IV. TN929.53

中国版本图书馆 CIP 数据核字（2015）第 228298 号

本书是作者多年 App 开发的经验总结，重点介绍 Android 应用开发中常见的实用技巧和疑难问题解决方法，为打造高质量 App 提供了有价值的实践指导，可帮助读者迅速提升应用开发能力和解决疑难问题的能力。本书涉及的主题有：Android 项目的重构、网络底层框架设计、经典场景设计、命名规范和编程规范、Crash 的捕获与分析、持续集成、代码混淆、App 竞品技术分析、移动项目管理和团队建设等。本书内容丰富，文风幽默，不仅给出疑难问题的解决方案，而且结合示例代码深入剖析这些问题的实质和编程技巧，旨在帮助移动开发人员和管理人员提高编程效率，改进代码质量，打造高质量的 App。

App 研发录
架构设计、Crash 分析和竞品技术分析

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：吴　怡　　责任校对：殷　虹
印　　刷：北京诚信伟业印刷有限公司　　版　　次：2015 年 10 月第 1 版第 1 次印刷
开　　本：186mm × 240mm　1/16　　印　　张：20.5
书　　号：ISBN 978-7-111-51638-5　　定　　价：59.00 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

华章科技
HZBOOKS | Science & Technology

Preface 序 一

互联网时代什么人是核心驱动力

在我刚刚开始宣布要做奇酷手机的时候，我曾经发布公开信说我需要四类动物：程序猿、攻城狮、产品狗、设计猫。程序员被排在了第一位，而从我的个人经历来说，与程序员有着密切的关系：大学研究生时的程序员，上班时的工程师，创业后的产品经理，最近几年一直在学习和琢磨设计。

这本书的作者建强也是其中一种人，一种喜欢钻研技术的程序员。我曾经和《奇点临近》作者雷·库兹韦尔交流的时候提到，也许上帝就是一名程序员，因为程序员正在通过给基因重新编程的方式来解决人类很多疾病之类的问题。

当然，实现给基因编程解决人类疾病问题的过程是漫长的，但“程序员”的作用是重大的。而在互联网的世界里，程序员的重要性更明显。一个好的程序员能力固然重要，精神世界的升华也不能缺少，写书就是一种精神世界的升华，能说服自己，也能帮助和提高更多人。

互联网时代离不开各种移动 App，本书提到很多时下移动互联网很前沿的技术，像竞品技术分析部分就提到 ABTest、WaxPatch 等。而且据说，为了写这本书，作者分析了市场上有名的上百款 App，能够费这么多心血去研究技术实现的人，在我看来至少是一个充满好奇心的人。正是这种拥有好奇心并执着探索的人，推动了近百年来的科学发展。

移动互联网的世界更是如此，从手机产生至今，短短二三十年的时间，就已经发生了翻天覆地的变化。今天的手机已经快成为人类的器官了，未来手机是什么样子很难说，但对手机应用的要求越来越高。虽然 iOS 和安卓平台上开发 App 会有所不同，但用户在各方面体验的要求是一致的。所以在我做手机的过程中，一直要求自己要充满好奇心。

移动 App 是一个充满了未知和探索的领域，这也正是它的魅力所在，所以越来越多渴望探索的人加入到移动互联网的创业大潮中来。事实上，这些移动 App 正在改变着我们的生活，

从订餐、打车到游戏娱乐都被各种 App 所改变。

但 App 相关的技术发展、更新非常迅速，所以作为技术人员要保持对技术的敏锐嗅觉，永远抱着谦卑的心态去学习先进的技术和理念，才能时刻占据着主动。当我们认为自己对这个世界已经相当重要的时候，其实这个世界才刚刚准备原谅我们的幼稚。

互联网发展到今天，程序员功不可没。也许程序员真的就是上帝，但他们在创造出一个个绚丽多彩的世界之前，注定要沉浸在枯燥的代码之中。我相信，每个程序都有自己的一个小小世界，在程序世界里，一切都按照他们的设计规则运行。那么你说，这和上帝创造世界有什么不同？互联网的世界里谁才是核心驱动力？

当然，我也希望这本书能培养出更多的 App 领域高级人才，来共同繁荣移动互联网的世界。

周鸿祎

奇虎 360 董事长

Preface 序 二

十年写一本书

1998 年，Peter Norvig 曾经写过一篇很有名的文章，题为“ Teach Yourself Programming in Ten Years”（十年学会编程）。这文章怎么个有名法呢，自发表以来它的访问次数逐年增加，到 2012 年总数已经接近 300 万次。

文章的意思其实很简单，编程与下棋、作曲、绘画等专业技能一样，不花上十年以上有素的训练（deliberative practice）、忘我的（fearless）投入，是很难真正精通的。Malcolm Gladwell 后来的畅销书《异类》用 1 万小时这个概念总结了类似的观点。

那么，写书呢？

说到写书，我的另一位朋友在微博上说过一段话，让我一直耿耿于怀：“有的时候被鼓励、怂恿写书，就算书能卖 5000 册，40 块一本，8% 的版税，收益是 16000RMB，这还没缴税。我还不如跟读者化缘，把内容均匀地贡献给读者，一个礼拜就能募集到这个数额。为什么要去写书？浪费纸张，污染环境。为了名气？为了评职称？”这位朋友是 2001 年我出版的国内第一本 Python 书的译者，当时这本封面上是一只老鼠的书只有 400 页，现在英文原版最新版已经 1600 页了——时间真是最强大的重构工具。其实他的话说得挺实在的。这年头写专业图书，经济上直接的回报，的确很低。

可要是真的没人写书了，这个世界会好吗？

我曾经在出版社工作十几年，经手的书数以千计，有的书问世之初就门可罗雀，有的书一时洛阳纸贵但终归沉寂，只有少数一些，能够多年不断重印，一版再版。这后一种，往往是真正的好书，是某个领域知识系统整理的精华，其作用不可替代，Google 索引的成千上万的网页也不能——聚沙其实是不能成塔的。Google 图书计划的受挫，其实是人类文明发展的重大延迟，对此我深以为憾。

书（我说的是科技专业书）可以粗略分为两种，一种是入门教程性质的，一种是经验之谈

或者感悟心得。无论哪一种，都需要作者多年教学或者实践的积淀。很难想象，没有这种积淀，能够很好地引导读者入门，或者教授其他人需要花费十年才能掌握的专业技能。

所以，好书不易得。它不仅来之不易（有能力写的人本来就少，这些人还不一定有动力写），而且还经常面临烂书太多，可能劣币驱逐良币的厄运。最后的结果是，很多领域都缺乏真正的好书，导致整个圈子的水平偏低。因为，书这种成体系的东西，往往是最有效的交流与传承手段，是互联网（微博、微信、博客、视频等等）上碎片化的信息不能取代的。

几天前，我的 Gmail 邮箱里收到一封邮件：

> “还记得 2008 年我就想写一本书，但是感觉技术能力不够，就只好去翻译了一本。一晃 2015 年了，积累了 11 年的技术功底，写起书来游刃有余。这本书我整整写了 1 年，其中第 6 章和第 9 章是自认为写得最好的章节。请刘江老师为我这本书写一篇序言，介绍一下无线 App 的技术前瞻和趋势，以及看过样章后的一些感想吧。谢谢。”
>
> 包建强

包建强？我记得的。一个长得挺帅的大男孩儿，2004 年复旦大学毕业，2008 年被评为微软的 MVP。在技术上有追求，而且热心。多年前在博客园非常活跃，张罗着要将里面的精华文章结集出版成书。很爱翻译，曾找我推荐过国外的博客网站，想要把一个同学 WPF/Silverlight 系列文章翻译成英文。2009 年我在图灵出版了他翻译的 .NET IL 汇编语言方面的书，到现在也是这一主题唯一的一本。好多年没联系，没想到他已经从 .NET 各种技术（WPF、Silverlight、CLR 等），转向移动客户端开发了。更让人动容的是，他一直不忘初心，用了十年，终于完成了自己的书。

那么，这是一本什么样的书呢？

我将书的几个样章转给身边从事 Android 开发的一位美团同事，他看了以后有点小激动，给了这样的评价：

> “拿到这本书的目录和样章时，感到非常惊喜，因为内容全是一线工程师正在使用或者学习的一些热门技术和大家的关注点。比如网络请求的处理、用户登录的缓存信息、图片缓存、流量优化、本地网页处理、异常捕抓和分析、打包等这些平时使用最多的技术。
>
> 我本人从事 Android 开发两年，特别想找一本能提高技术、经验之谈的书，可惜很难找到。本书不光站在技术的层面上去谈论 Android，还通过市场上比较火的一些 App 和当今 Android 在国内发展的方向等各种角度，来分析怎么样去做好一个 App。
>
> 我个人感觉这本书不仅能让你从技术上有收获而且在其他层面上让你对 Android 有更深层次的了解。我已经迫不及待这本书能够尽快上市，一睹为快了。”

的确，目前国内外市面上数百种 Android 开发类图书，基本上可以分为两类：

- 一类是从系统内核和源代码入手，作者往往是 Linux 系统背景，从事底层系统定制等方面工作。书的内容重在分析 Android 各个模块的运行机制，虽然深入理解系统肯定对应用开发者有好处，但很多时候并不是那么实用。
- 一类是标准教程，作者往往是培训机构的老师，或者不那么资深但善于总结的年轻工程师，基本内容是 Android 官方文档的变形，围绕 API 的用法就事论事地讲开去。虽然其中比较好的，写法、教学思路和例子上也各有千秋，但你看完以后真上战场，就会发现远远不够。

本书与这两类书都完全不同，纯从实战出发，在官方文档之上，阐述实际开发中应该掌握的那些来之不易的经验，其中多是过来人踩过坑、吃过亏，才能总结出来的东西。不少章节类似于 Effective 系列名著的风格，有很高的价值。

书最后的部分讨论了团队和项目管理，既有比较宏观的建议，比如流程、趋势，更多的还是实用性非常强的经验，比如百宝箱、必备文档，等等。很多章节，不限于 Android，对其他平台的移动开发者也有很大的借鉴意义。

看得出来，这里很多内容都是包建强自己平时不断记录、积累的成果，其中少量在他的博客上能看到雏形。如果说多年前，包建强在组织和翻译图书时还有些青涩的话，本书中所显现出来的，则完全是一派大将风度，用他自己的话来说，“游刃有余”了。

我曾经不止一次和潜在的作者说过：“不写一本书，人生不完整。”我说这话是认真的。人生百年，如果最后没有什么可以总结、留之后人的东西，那可不是什么值得夸耀的事情。

而说到总结，互联网各种碎片化的媒体形式当然有各种方便，但到头来逃脱不了烟花易逝的命运（想想网上有多少好的文字，链接早已失效，现在只能到 archive.org 上寻找，甚至那里也不见踪影）。还是书这种物理形式最坚实，最像那么回事儿，也最是个东西。去国家图书馆看过宋版书的人肯定会有体会。

当然，真正能立住的，是那些真正的好书，那些花费十年写出来的东西。希望有更多的同学像包建强这样，十年写一本书。希望有更多好书不断涌现出来。

我更希望，包建强不止步于书的出版，而是能将书的内容互联网化，让读者和同行也加入进来，不断生长、丰富，不断改版重印，变成一种活的东西。写一本好书，不应该限于十年。

刘江

美团技术学院院长

CSDN 和《程序员》杂志前总编

序　三 Preface

这是一本很有特点的书，没有系统的知识介绍，也没有对细分领域钻牛角尖般的头头是道。第一次看完老包的样章时我很惊讶，他不仅一个人完成了全书内容的撰写，而且其中大部分章节都非常接地气并具有时代性。

当前移动开发技术处在一个野蛮增长的时代，在移动开发从业人员逐年递增的情况下，很多公司的移动开发团队都有几十人甚至上百人。当 App 越做越大，承载了越来越多的功能时，不断地累加代码也造成了很多问题。在解决这些问题的同时，很多人从单纯的业务开发转向深入研究技术细节，沉淀了很多经验，并诞生了不少有意思的开源项目。

在 2013 年我首次遇到 Android 65536 方法数限制的时候，网络上唯一能查询到的资料就是 Facebook 上的一篇博客，其中简单介绍了博主遇到的问题及解决的大致方法。当时在没有任何参考资料的情况下只能自己开发解决方案，并且由于需要分拆 dex 引入了不少其他的问题。今天看到本书中总结的这些经验和问题，发现本书能够给我很好的启示，原本那些踩过的坑和交过的学费其实都是可以避免的。虽然书中介绍每个问题时篇幅看上去并不大，但是提炼得很精简，如果你对其中的某段不是很理解，很可能它正是在你真正遇到问题时会联想到的内容和恰到好处的解决方案。

本书第 6 章常见的异常分析，就是完全基于实践积累完成的。就阅读这章本身来说，可能学到的知识点非常分散，但是包含了很多不为人知的冷门或者非常细节的知识。如果你对其有深刻的共鸣，多数都是因为自己曾有过被坑的经历。在我自己的异常分析过程中，会遇到一些非常难理解的异常，俗称“妖怪问题”。这类异常的表象很难和原因联系到一起，光读取栈信息不足以理解异常的机理，这时候就需要有更完善的异常收集系统，能够把应用的当前状态进行回溯，这对分析问题是很有帮助的。

本书第 9 章我认为是最接地气也是最有特色的章节，从分析国内热门的 App 开始，帮助读者了解最前沿的大公司的移动开发的技术方向。有很多技术点是小的 App 开发团队并不会

花精力关注的，比如资源文件如何组织，如何应对线上故障等，但是如果在应用规模急剧增长后再去解决相应的问题就会花费不小的代价，不如从一开始就遵循这些已经在其他成熟团队中积淀的经验和法则。对于应用开发来说，很多高深的技术和复杂的框架也许并不会对最终的结果带来很大的帮助，学习一些业界真实的方案，并对其进行扩展可能是更加稳妥的方式。

从 Android 和 iOS 诞生至今，技术虽然一直在进步，但它们分别是由 Google 和 Apple 主导的。开源社区虽然有很多热门的项目，但是不同于服务端的 Apache 扶持的大型开源项目，客户端受限于体积、硬件及部署方式的限制，一直没有形成大而全的框架，反而出色的开源项目都聚焦在一个点上。回想 Joe Hewitt 当年在 Facebook 开源的 Three20 项目引领了当时的 iOS 应用架构，到目前已经被大多数的应用抛弃，只能说这是一个大浪淘沙的时代，移动技术在飞速发展，技术被淘汰的速度非常之快。优秀的开发人员需要具备的不光是对平台的了解和写代码的能力，更重要的是对技术的整合和对发展趋势的理解。本书就像是对 2015 年整个移动技术的一份快照，非常富有这个时代的特征。整本书并不是从枯燥的文档提炼而来，而是真切地从一个互联网从业者的切身经历和与他人的交流中得来。对于一个需要时刻紧跟移动浪潮的 App 开发人员来说，本书是值得一读的好书。

屠毅敏

大众点评首席架构师

前　言 Preface

皇皇三十载，书剑两无成

在你面前娓娓而谈的我，曾经是一位技术宅男。我写了 6 年的技术博客，500 多篇技术文章。十年编程生涯，我学习了 .NET 的所有技术，但是从微软出来，踏上互联网这条路，却发现自己还是小学生水平，当时恰逢三十而立之年，感慨自己多年来一事无成，于是又开始了新一轮的学习。选择移动互联网这个方向，是因为这个领域所有人都是从零开始，大家都是摸索着做，初期没有高低上下之分。

在此期间，我做过 Window Phone 的 App，学会了 Android 和 iOS，慢慢由二把刀水平升级到如今的著书立说，本来我想写的是 iOS 框架设计，因为当时这方面的经验积累会更多一些，2013 年的时候我在博客上写了一系列这方面的文章，可惜没有写完。如今这本书是以 Android 为主，但是框架设计的思想是和 iOS 一致的。

作为程序员，不写本书流传于世，貌似对不起这个职业。2008 年的时候我就想写，可那时候积累不够，所知所会多是从书本上看到的，所以没敢动笔，而是选择翻译了一本书《 MSIL 权威指南》。翻译途中发现，我只能老老实实地按照原文翻译，而不能有所发挥。我渴望能有一个地方，天马行空地将自己的风格淋漓尽致地表现出来，在写这本书之前，只有我的技术博客。

终于给了自己一个交代，东隅已逝，桑榆非晚。

文章本天成，妙手偶得之

这是一本前后风格迥异的书，以至于完稿后，不知道该给本书起一个什么样的书名。只希望各位读者看过之后能得到一些启示，我就心满意足了。

下面介绍一下本书的章节概要。本书分为三个部分共计 12 章。

第 1 章讲重构。这是后续 3 章的基础。先别急着看其他章节，先看一下这一章介绍的内

容，你的项目是否都做到了。

第 2 章讲网络底层封装。各个公司都对 App 的网络通信进行了封装，但都稍显臃肿。我介绍的这套网络框架比较灵巧，而且摆脱了 AsyncTask 的束缚，可以在底层或上层快速扩展新的功能。这样讲多少有些自卖自夸，好不好还是要听读者的反馈，建议在新的 App 上使用。

第 3 章讲 App 中一些经典的场景设计，比如说城市列表的增量更新、缓存的设计、App 与 HTML5 的交互、全局变量的使用。对于这些场景，各位读者是否有似曾相识的感觉，是否能从我的解决方案中产生共鸣？

第 4 章介绍 Android 的命名规范和编码规范。网上的各种规范多如牛毛，但我们不能直接拿来就使用，要有批判地继承吸收，要总结出适合自己团队的规范。所以，即使是我这章内容，也请各位读者有选择地采纳。我写这一章的目的，就是要强调“无规矩不成方圆”，代码亦如是。

第 5 章和第 6 章组成了 Android 崩溃分析三部曲。写这本书用了一年，其中有半年多时间花在这两章上。一方面，要不断优化自己的算法，训练机器对崩溃进行分类；另一方面，则是对八十多种线上崩溃追根溯源，找到其真正的原因。

第 7 章讲 Android 中的代码混淆。本不该有这一章，只是在工作中发现网上关于 ProGuard 的介绍大都只言片语。官方倒是有一份白皮书，但是针对 Android 的介绍却不是很多，于是便写了这章，系统而全面地介绍了在 Android 中使用 ProGuard 的理论和实践。

第 8 章讲持续集成（CI）。十年传统软件的经验，使我在这方面得心应手。这一章所要解决的是，如何把传统软件的思想迁移到 App 上。

第 9 章讲 App 竞品分析，是研究了市场上几十款著名 App 并参阅了大量技术文章后写出的。之前积累了十年的软件研发经验，这时极大地帮助了我。

第 10 章讲项目管理，是为 App 量身打造的敏捷过程，是我在团队中一直坚持使用的开发模式。App 一般 2 周发一次版本，迭代周期非常快，适合用敏捷开发模式。

第 11 章讲日常工作中的问题解决办法。那是在一段刀尖上舔血的日子中总结出的办法，那时每天都在战战兢兢中度过，有问题要在最短时间内查找到原因并尽可能修复；那也是个人能力提升最快的一段时光，每一次成功解决问题都伴随着个人的成长。

第 12 章讲 App 团队建设。我是一个孔雀型性格的老板，所以我的团队中多是外向型的人，或者说，把各种闷骚型技术宅男改造成明骚；我是从技术社区走出来的，所以我会推崇技术分享，关心每个人的成长；我有 8 年软件公司的工作经验，所以我擅长写文档、画流程图，以确保一切尽在掌握之中。有这样一位奇葩老板，对面的你，还不快到我的碗里来，我的邮箱是 16230091@qq.com，我的团队，期待你的加入。

心如猛虎，细嗅蔷薇

话说，我也是无意间踏上编程这条道路的。如果不是在大三实在学不明白实变函数这门课的话，我现在也许是一个数学家，或者和我的那些同学一样做操盘手或是二级市场。

我真正的爱好是看书，最初是资治通鉴、二十四史，后来发现在饭桌上说这些会被师弟师妹们当做怪物，于是按照中文系同学的建议翻看张爱玲、王小波的小说，读梁实秋的随笔。在复旦的四年时光，熏出了一身的“臭毛病”，比如说看着夜空中的月亮会莫名其妙地流眼泪，会喜欢喝奶茶并且挑剔珍珠的口感。

不要以为程序员只会写代码。程序员做烘焙绝对是逆天的，因为这用到软件学中的设计模式，我也曾研发出失败的甜品，做饼干时把黄油错用成了淡奶油，然后把烤得硬邦邦的饼干第二天拿给同事们吃。

我涉及的领域还有很多，比如煮咖啡、唱K、看老电影，都是在编程技术到了一定瓶颈后学会的，每一类都有很深的学问。不要一门心思地看代码，生活能教会我们很多，然后反过来让我们对编程有更深刻的认识。

心若有桃园，何处不是水云间。

会当凌绝顶，一览众山小

如果后续还有第二卷，我希望是讲数据驱动产品。就在本书写作期间，我的思想发生了一次升华，那是在2015年初的一个雪夜，我完成了从纠结于写代码的方法到放眼于数据驱动产品的转变。这也是这本书前面代码很多，越到后面代码越少的原因。

数据驱动产品是未来十年的战略布局。之前，我们过多地关注于写代码的方法了，却始终搞不清用户是否愿意为我们辛辛苦苦做出来的产品买单，技术人员不知道，产品人员更不知道。产品人员需要技术人员提供工具来帮助他们进行分析，比如说ABTest，比如说精准推送平台，比如说用户画像，而我们检查自己的代码，却发现连PV和UV都不能确保准确。

这也是我接下来的研究和工作方向。

本书全部代码均可以从作者的博客上下载，地址是：www.cnblog.com/Jax/p/4656789.html。

包建强

2015年8月3日于北京

Contents 目　录

序一
序二
序三
前言

第一部分　高效 App 框架设计与重构

第 1 章　重构，夜未眠 …… 3

1.1　重新规划 Android 项目结构 …… 3
1.2　为 Activity 定义新的生命周期 …… 5
1.3　统一事件编程模型 …… 7
1.4　实体化编程 …… 9
1.4.1　在网络请求中使用实体 …… 9
1.4.2　实体生成器 …… 11
1.4.3　在页面跳转中使用实体 …… 12
1.5　Adapter 模板 …… 14
1.6　类型安全转换函数 …… 16
1.7　本章小结 …… 17

第 2 章　Android 网络底层框架设计 …… 19

2.1　网络低层封装 …… 19

2.1.1 网络请求的格式 …… 19
2.1.2 AsyncTask 的使用和缺点 …… 21
2.1.3 使用原生的 ThreadPoolExecutor + Runnable + Handler …… 24
2.1.4 网络底层的一些优化工作 …… 28
2.2 App 数据缓存设计 …… 32
2.2.1 数据缓存策略 …… 32
2.2.2 强制更新 …… 35
2.3 MockService …… 36
2.4 用户登录 …… 38
2.4.1 登录成功后的各种场景 …… 39
2.4.2 自动登录 …… 41
2.4.3 Cookie 过期的统一处理 …… 44
2.4.4 防止黑客刷库 …… 45
2.5 HTTP 头中的奥妙 …… 46
2.5.1 HTTP 请求 …… 46
2.5.2 时间校准 …… 48
2.5.3 开启 gzip 压缩 …… 51
2.6 本章小结 …… 52

第 3 章 Android 经典场景设计 …… 53

3.1 App 图片缓存设计 …… 53
3.1.1 ImageLoader 设计原理 …… 53
3.1.2 ImageLoader 的使用 …… 54
3.1.3 ImageLoader 优化 …… 55
3.1.4 图片加载利器 Fresco …… 56
3.2 对网络流量进行优化 …… 58
3.2.1 通信层面的优化 …… 58
3.2.2 图片策略优化 …… 59
3.3 城市列表的设计 …… 61
3.3.1 城市列表数据 …… 61
3.3.2 城市列表数据的增量更新机制 …… 63

3.4 App 与 HTML5 的交互 …… 64
3.4.1 App 操作 HTML5 页面的方法 …… 64
3.4.2 HTML5 页面操作 App 页面的方法 …… 65
3.4.3 App 和 HTML5 之间定义跳转协议 …… 66
3.4.4 在 App 中内置 HTML5 页面 …… 67
3.4.5 灵活切换 Native 和 HTML5 页面的策略 …… 68
3.4.6 页面分发器 …… 68
3.5 消灭全局变量 …… 70
3.5.1 问题的发现 …… 70
3.5.2 把数据作为 Intent 的参数传递 …… 71
3.5.3 把全局变量序列化到本地 …… 71
3.5.4 序列化的缺点 …… 75
3.5.5 如果 Activity 也被销毁了呢 …… 79
3.5.6 如何看待 SharedPreferences …… 80
3.5.7 User 是唯一例外的全局变量 …… 80
3.6 本章小结 …… 81

第 4 章 Android 命名规范和编码规范 …… 83

4.1 Android 命名规范 …… 83
4.2 Android 编码规范 …… 86
4.3 统一代码格式 …… 89
4.4 本章小结 …… 90

第二部分 App 开发中的高级技巧

第 5 章 Crash 异常收集与统计 …… 93

5.1 异常收集 …… 93
5.2 异常收集与统计 …… 96
5.2.1 人工统计线上 Crash 数据 …… 96
5.2.2 第一个线上 Crash 报表：Crash 分类 …… 97