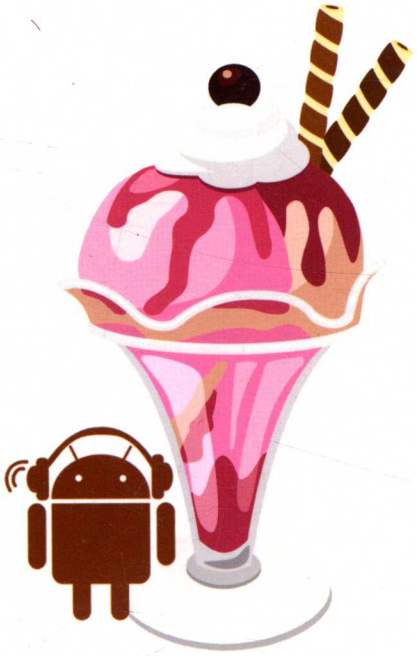
HZ BOOKS
华章IT

腾讯音乐Android开发总监，从事移动开发10余年，曾主导并参与过多款用户规模上亿的Android应用的开发工作

以流畅、稳定、省电、省流量4个方向为目标，从UI、内存、存储、稳定性、省电、安装包大小6个维度深入剖析性能优化的技术和方案

移动开发



罗彧成◎著

Best Practices for Android Performance Tuning

Android应用性能优化最佳实践

机械工业出版社
China Machine Press

移动开发

Best Practices for Android Performance Tuning

Android应用性能优化最佳实践

罗彧成◎著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Android 应用性能优化最佳实践 / 罗彧成著 . —北京：机械工业出版社，2017.1
（移动开发）

ISBN 978-7-111-55616-9

I. A… II. 罗… III. 移动终端 – 应用程序 – 程序设计 IV. TN929.53

中国版本图书馆 CIP 数据核字（2016）第 315986 号

Android 应用性能优化最佳实践

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：李 艺 责任校对：董纪丽
印 刷：中国电影出版社印刷厂 版 次：2017 年 2 月第 1 版第 1 次印刷
开 本：186mm × 240mm 1/16 印 张：13.75
书 号：ISBN 978-7-111-55616-9 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

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

华章IT
HZBOOKS | Information Technology

Preface 前　言

为什么写这本书

一个好的应用，除了要有吸引人的功能和交互之外，在性能上也应该有高的要求，即使应用非常具有特色，或者功能和业务具有唯一性，在产品前期可能吸引了部分用户，但用户体验不好的话，也会给产品带来很差的口碑，如果有在体验上更好的竞品，用户也会很快转移。那么一个好的应用应该如何定义呢？主要有三方面：

- 业务 / 功能
- 符合逻辑的交互
- 优秀的性能

众所周知，Android 系统作为以移动设备为主的一款操作系统，硬件配置有一定的限制，虽然配置现在越来越高级，但仍然无法和 PC 相比，在 CPU 和内存上的使用不合理或者耗费资源多时，就会碰到内存不足导致的稳定性问题、CPU 消耗太多导致的卡顿问题等。例如，我们发布一款产品后会收到很多的反馈，这些反馈来自很多渠道，有用户反馈，有应用发布平台的反馈通道等。

面对这些问题时，大家想到的都是联系用户，然后看日志，特别是有关性能类问题的反馈，原因也非常难找，日志大多用处不大，为什么呢？因为性能问题大部分是非必现的问题，定位时很难复现，而又没有关键的日志，当然就无法找到原因了。这些问题非常影响用户的体验和功能的使用，所以解决这些问题是非常重要的。当前市场上讲解性能优化的书太少，即使有些书讲到，很多也是一笔带过，没有深入分析和寻找解决方案，所以有必要用一本书来从多个维度讲解在性能上我们面临了什么问题，如何解决这些问题，并在实际的项目中来优化我们的应用，以提高用户体验。

试读结束：需要全本请在线购买：www.ertongbook.com

本书面向的读者

本书适合所有 Android 应用开发从业人员及在校学生，特别是有一定 Android 应用开发经验的开发人员，高级开发人员也可以通过本书了解更多的性能调优知识。

本书特色

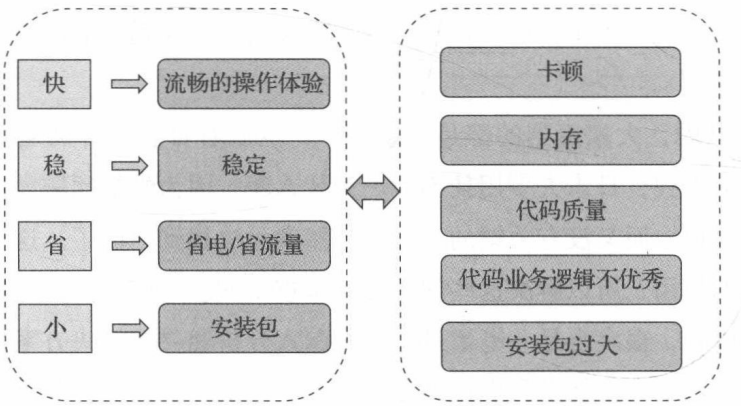
本书为进阶类图书，对于一些基础技术和基础理论知识不会做过多的阐述，特别是入门类的知识点，大家可以从其他书籍获取相关的知识。书中以性能优化为核心，深入剖析性能优化具体涉及的技术背景与优化方案，同时提供典型案例，帮助读者更深入地掌握 Android 应用开发技术，理解 Android 的运行机制和原理，掌握 Android 性能优化的思想，让开发者快速成长，打造高质量的 Android 应用。

本书的主要内容

可以把用户能体验到的性能问题主要总结为 4 个类别：

- 流畅
- 稳定
- 省电
- 省流量

性能问题的主要原因是什么，原因有相同的，也有不同的，但归根结底，不外乎内存使用、代码效率、合适的策略逻辑、代码质量这一类问题。本书讲解内容的目标和方向如下图所示。



从上图可以看到，打造一个高质量的应用应该以 4 个方向为目标：快、稳、省、小。

- 快：使用时避免出现卡顿，响应速度快，减少用户的等待时间，满足用户预期。

- 稳：降低 crash 率和 ANR 率，不要在用户使用过程中崩溃和无响应。
- 省：节省流量和耗电，减小用户使用成本，避免使用时导致手机发烫。
- 小：安装包小可以降低用户的安装成本。

这 4 类问题需要从根源上解决，也就是要解决图中第二个框里的问题：卡顿、内存使用不合理、代码质量差、代码逻辑不优秀、安装包过大。这些问题也是在开发过程中碰到最多的问题，在实现业务需求的同时，也需要考虑到这些点，多花时间去思考，避免功能完成后再来做优化和修复 Bug，这个时候带来的成本会增加。如果是维护之前的代码，就需要使用一系列工具来发现问题点。

性能优化不是更新一两个版本可以解决的，是持续性的需求，结合到实际中，在一个新产品 / 项目开始时，由于人力和上线时间的限制，可以把优先级放低，但有些点是在写代码时就要考虑的，这就体现出程序员的技术功底。

本书强调性能调优的核心思想和方向如下：

发现问题→分析问题原因及背景→寻找最优解决方案→解决问题。

本书一共 7 章，在简单介绍了 Android Studio 的使用指南后，分别从绘制（UI）、内存、存储、稳定性、耗电以及安装包 6 个方面进行优化，从系统上深入分析绘制和内存的原理，一步步深入了解导致性能问题的本质原因，同时讲述了多种性能优化工具的使用，通过分析典型案例，得到有效的优化方案，从而实现更高质量的应用。书中所讲述的内容均基于 Android 6.0 系统。

勘误和资源下载

由于写作时间实在有限，在书稿交付时仍有些许不安，为此先为此书可能存在的错误或者描述不清楚的地方致以真诚的歉意，如果你发现此书存在瑕疵或者有任何建议，请发邮件到 5482586@qq.com，我会尽快回复，非常期待大家的反馈。

本书代码的下载地址：https://github.com/lyc7898/AndroidTech。

致谢

由于时间的问题，本书写作时间非常长，非常感谢杨福川编辑对我的鼓励和宽容，并且分享了非常有用的碎片化写作方法，使我一直坚持把本书写完。同时感谢李艺编辑的校对和勘误，才完成了这本图文并茂、格式清晰的技术书籍。

感谢我的妻子李萍女士对我的理解和支持，在我几乎将所有的时间投入工作中时一直给予最大的宽容和鼓励，使我每天即使再忙再累时仍然可以回到温馨的家。同时感谢我的父母

和岳父母，感谢他们对我无私的帮助，他们都是伟大的父母。

特别感谢我的爷爷罗志华老先生，在我的学习生涯中给予的无私帮助，在工作和生活上的谆谆教诲。还要感谢刘景瑜老师，在求学阶段的鼓励和教诲，告诉我有很多需要去做的事情。

最后感谢我现在工作的公司，在这里我得到了最快的成长，学习到非常多的东西，感谢公司领导及所有同事，在这里工作，能感受到大家每天都在成长。

Contents 目　录

试读结束：需要全本请在线购买： www.ertongbook.com

第 1 章 Chapter 1

Android Studio 使用指南

假设我们要选择一个 IDE 来开发应用，目前主流的 IDE 有 Eclipse、Android Studio、Idea，在 2015 年前，这三个 IDE 各有优缺点，但现在，Android Studio 是首选，因为随着 Google 对 Android Studio 的大力完善和支持，优势已经越来越明显，但目前仍有不少开发人员在使用 Eclipse。为什么要首选 Android Studio，它有什么优势，具体要如何使用，本章将逐一揭示。

本书的例子都是使用 Android Studio 开发的。

1.1 Android Studio 的优势

为什么本书要推荐使用 Android Studio 呢，Android Studio 的核心是一个智能代码编辑器，可进行高级代码完成、重构和调试。这款功能强大的代码编辑器可以帮助你成为更高产的 Android 应用开发人员。虽然 Android 发布初期有很多 bug，功能也不完善，但随着版本的更新，已经在各方面领先其他 IDE，下面列出了 Android Studio 的几点优势。

- 稳定速度快：使用 Eclipse 的开发人员都会碰到突然假死、卡顿、内存占用高等一系列影响开发效率的老问题，Android Studio 在这块性能上得到了明显的提升，并且 Android Studio 使用了单项目管理模式，在启动速度上明显比 Eclipse 快。
- 功能强大的 UI 编辑器：集合了 Eclipse+ADT 的优点，并且能更实时地展示界面布局效果。

- 完善的插件管理：Android Studio 支持了多种插件，可直接在插件管理中下载所需的插件。
- 完善地支持多种代码管理工具：不需要任何操作，直接支持 SVN、Git 等主流的代码管理工具。
- 整合了 Gradle 构建工具：Gradle 继承了 Ant 的灵活性和 Maven 的生命周期管理，不使用 XML 作为配置文件格式，采用了 DSL 格式，使得脚本更加灵活简洁。
- 智能：智能保存，智能补齐，在实际的编辑代码中熟练使用后，可极大提高代码编写效率。
- 内置终端：不需要自己打开一个终端来使用 ADB 等工具。
- 谷歌官方支持：是 Google 官方专门为 Android 应用开发打造的利器，也是目前 Google 官方唯一推荐，并且不再支持其他 IDE。

Android Studio 的更多优势会在开发工作的细节中体现出来，可以参考一些 Android Studio 的使用书籍和文档，以便了解它的强大之处。

1.2 Android Studio 使用入门

1.2.1 Android Studio 安装

这里我们以在 Windows 系统上安装 Android Studio 为例，具体的安装步骤如下：

1）安装 JDK，且为 JDK 1.6 及以上版本。

2）下载 Android Studio 安装包：developer.android.com/sdk/installing/studio.html。

3）单击安装包开始安装，首先进入选择组件界面，如图 1-1 所示。

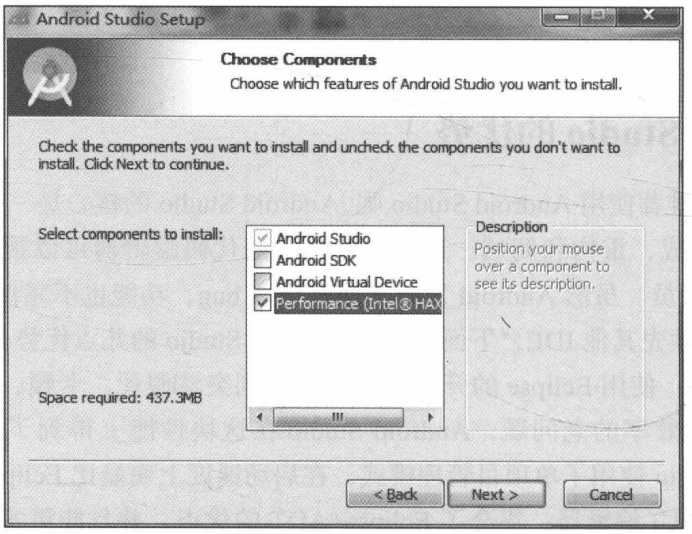


图 1-1 选择安装组件

一般已经安装 Eclipse 或其他 Android 开发环境的，只需要安装默认的选项（Android Studio）即可。

4）单击 Next，如果已经下载过 SDK，并且在前面的组件没有选择安装 SDK，会弹出一个设置本地 SDK 的界面，选择本地 SDK 目录，如图 1-2 所示。

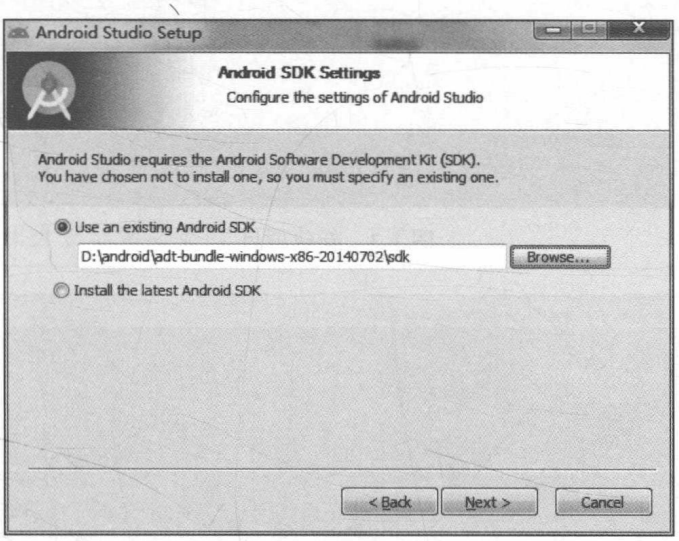


图 1-2　选择 SDK 目录

5）一直单击 Next，直到安装完成，单击 Finish，首次启动 Android Studio 会有一个配置的过程，需要等待一下。

因为本书主要是讲性能优化，所以这里只是简单地介绍下基本的安装。需要了解更多安装细节，可以参考 Android 开发官网的详细介绍文档：

http://developer.android.com/intl/zh-cn/sdk/installing/index.html?pkg=studio。

注意　如果首次启动出现错误导致启动失败，一般来说是因为联网下载一些配置文件失败，可以使用记事本打开 studio 的安装目录下 /bin 中的 idea.properties 文件，在最后一行添加 disable.android.first.run=true。

1.2.2　创建一个 Android Studio 工程

又到熟悉的 HelloWold 环节了，安装完成后，如果没有新建过项目，会出现一个欢迎对话框，选择新建项目。如果已经有项目，则可以通过 File → New → New Project 来新建一个项目，填写相关名称，包括应用名（Application name）、公司名（Company Domain）、包名（Package name）、项目本地路径（Project Location），如图 1-3 所示。

下一步选择开发平台，选择 Android Phone，SDK 使用 Android 6.0，如图 1-4 所示。

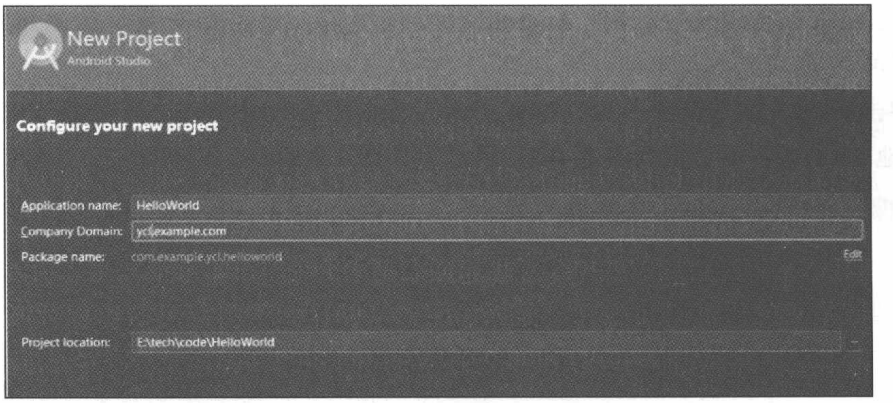


图 1-3 新建项目

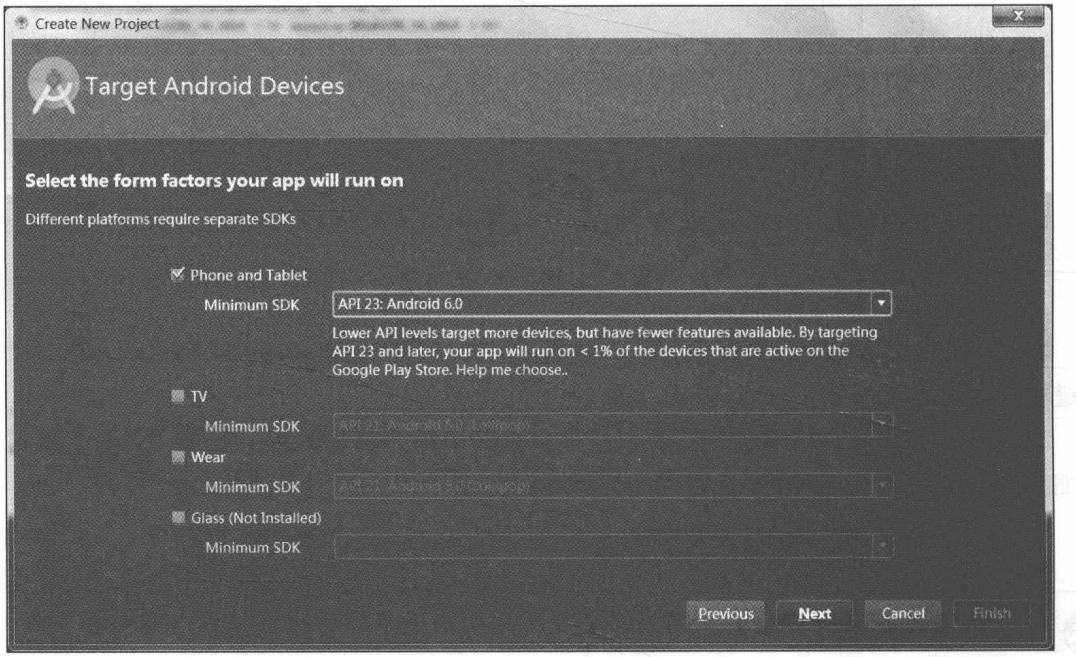


图 1-4 选择应用运行平台

后面两步都使用默认设置，最后单击 Finish 进入我们的项目。进入后可以看到，左侧的项目文件区域显示的文件树结果和 Eclipse 并不相同，而且本地目录的文件层级也不相同。因为 Android Studio 使用了 Gradle 项目构建工具，而 Eclipse 使用 Ant 构建项目。

如果不习惯这种结构，可以通过 Gradle 设置与 Eclipse 相同的目录结构。

进入后发现不能编译，这是因为还未设置编译工具。需要先配置 Gradle，由于 Android Studio 没有自带 Gradle 插件，所以会自动下载 Gradle，但需要设置代理，并且设置代理下载

仍然会比较慢，这里介绍离线配置 Gradle 的方法，步骤如下：

1）从官网下载 Gradle：http://gradle.org/gradle-download/，这里我们选择 1.3 版本。

2）进入 C:\Users\.gradle\wrapper\dists\gradle-2.2-all\1vevkra640w3rb9hkuw50q5we，最后这个文件夹是随机生成的，所以直接进入，把下载好的 gradle-1.3-all.zip 放到这个文件夹内即可。

3）重新启动 Android Studio，就可以正常编译了。

注意　Gradle 是一种依赖管理工具，基于 Groovy 语言，抛弃了基于 XML 的各种繁琐配置，取而代之的是一种基于 Groovy 的内部领域特定语言（DSL），掌握 Gradle 脚本的编译和打包是应用开发非常必要的。

1.2.3　从 Eclipse 项目迁移到 Android Studio

Eclipse 项目和 Android Studio 相比，在项目结构、构建系统以及 IDE 的风格上都有了较大的变化。从 Eclipse ADT 项目迁移到 Android Studio 有两种方法，一是直接把 Eclipse 项目导入 Android Studio 中，二是从 Eclipse 导出 Gradle 项目，然后在 Android Studio 中可以直接打开导出的项目。

Android Studio 是基于 intelliJ IDEA 开发的一款 IDE，与 Eclipse 有很多不同的地方，特别是工程的目录结构，区别非常大。在 Eclipse Android 中的 Project 在 Android Studio 中是一个 Module，并且 Android Studio 是单项目管理，每个窗口只能打开一个 Project，这也是 Android Studio 打开更快的原因之一。在 Android Studio 中，有多种视图结构类型：Packages、Project Files、Scratches、Problems、Production、Tests，可以根据自己需求来使用不同的视图类型。

Eclipse 和 Android Studio 中项目结构的对应关系如表 1-1 所示。

表 1-1　Eclipse 与 Android Studio 项目结构对应的关系

Eclipse ADT	Android Studio
Workspace	Project
Project	Module
Project-specific JRE	Module JDK
Classpath variable	Path variable
Project dependency	Module dependency
Library Module	Library
AndroidManifest.xml	app/src/main/AndroidManifest.xml
assets/	app/src/main/assets
res/	app/src/main/res/
src/	app/src/main/java/
tests/src/	app/src/androidTest/java/

从表 1-1 可以看出，结构发生了非常大的变化，如果直接按对应关系来迁移项目中的文件对应到 Android Studio 的目录结果，工作量不小并且容易出错。因此，为了简化迁移过程，Android 的 Studio 提供了一个快速导入 Eclipse 项目的方法，可以将 Eclipse 和 Ant 构建脚本快速转换到 Android Studio 上。导入步骤如下：

1）在 Android Studio 菜单栏选择 File → New → Import Project。如果是首次使用，在欢迎页可直接选择 Import Project。

2）选择需要导入的 Eclipse 工程目录，AndroidManifest.xml 必须是在根目录下。

3）在导入过程中会提示是否重新配置 Eclipse 中的所有第三方库和项目依赖关系，如果勾选，就会重新生成新的第三方管理库和依赖关系，依赖关系和第三方库也可以在 build.gradle 文件上修改。如图 1-5 所示，有三个选项可以选择，前两项目是 jar 包的引用规则，建议勾选，这样就不用管理这些 jar 包了。第三项是指是否需要把 Module 名创建为 camelCase 风格（首字母小写的命名规则）。

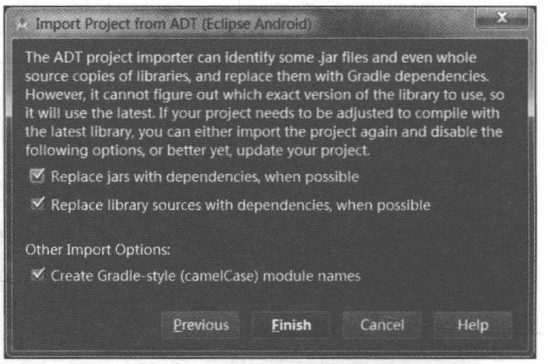


图 1-5 导出规则提示

4）单击 Finish 按钮，完成后会生成一个 import-summary.txt 文件，它是导入过程中产生的日志，这个文件比较重要，如果项目比较大，导入过程不一定顺利，可以通过这个文件来发现导入过程中出现的问题。

这样就把 Eclipse Android 项目导入 Android Studio 了，注意前面讲过，导入完成后生成了一个描述导入过程的日志文件：import-summary.txt，文件中包含以下几个内容：

- Ignored Files：描述在导入过程中忽略了哪些文件，这些文件是没有拷贝过来的，如果有需要，就要手动添加过来，一般会忽略 Eclipse ADT 相关的配置文件和一些自定义的文件 / 文件夹。
- Replaced Jars with Dependencies 和 Replaced Libraries with Dependencies：列出那些替换的 JAR 包，在 Android Studio 中，如果在仓库中有这个库或者 SDK，就会被替换。
- Moved Files：文件移动路径列表。
- Next Steps：接下来需要做些什么，如果没有问题，一般告诉你可以编译了。
- Bugs：如果不能编译，就会列出当前项目存在的问题。

上面介绍的这种导入项目的方式也是 Android 官网推荐的，理由是简单高效，有问题也容易发现，最主要是不会改变原来的文件组织架构。

更多有关从 Eclipse ADT 迁移到 Android Studio 的内容可以从 Android 开发官网上查询：http://developer.android.com/intl/zh-cn/sdk/installing/migrate.html#prerequisites。

注意 这样导入的项目还是会保留 Eclipse 的构建方式，比如在 Eclipse 上使用 Ant 构建，迁

试读结束：需要全本请在线购买： www.ertongbook.com