Chapter 1

Before You Start

In late 2011 as I got more into Android development, I tried to look for a book that I hoped would take my development to the next level. I'd already completed a couple of apps and wanted to know what everyone else was doing that I might have missed. Sure, there was a wealth of Android documentation from Google, but the Android docs had some odd recommendations; they suggest using jUnit3 for my unit testing, which felt like going backwards. I already knew there were existing jUnit4 testing frameworks for Android, such as Roboelectric, so maybe there were other cool things out there that I'd missed that I simply didn't know about that could really help me write better code.

This book is an attempt to pull together the research on the best practices that developers have created for the Android platform in the hope that you can find all the information you need in one place.

Once you've written an app or are part of an Android team of developers, it quickly becomes clear that Android development, just like any other language or environment, can get messy and inefficient if you don't think about how you're going to get organized. This book will help you take those steps to become a well oiled, productive team.

You may want to consider reading this book if you want to do one or more of the following:

- Get better at Android Development by looking at best practices sample code.
- Write apps that are easier to extend and maintain.
- Write more secure apps.
- Learn how to write not only the client side of the app but also its often ignored server side.

Introduction to Android

Android is a Linux-based open source operating system for smartphones. The company started back in October 2003 and was acquired by Google in August 2005. The HTC Dream, released in October 2008, was the first phone to run Android.

From a developer's perspective typically Android apps are written in Java. Google provides an Android SDK which provides the necessary libraries and applications to convert the Java code into a format that can run on Android phones. Most people use Eclipse or the command line to create Android apps. The Android Studio has recently emerged as an alternative to Eclipse and is likely to become the IDE of choice over the next year or two.

Android is the premier operating system for mobile devices, with over 75% of the world's devices and 52% of the US market running on it.

In my own personal experience there was a time when Android development was the redheaded stepchild. All development was first done on iOS and then developed in Android once the app became successful. This has changed now that Android phones have such a large market share.

Who Should Read This Book?

This book is designed to be approachable by developers who have any level of familiarity with Android. However, your degree of experience will dictate which parts you find most useful. If you're entirely new to Android development, or have only tinkered here and there, this book should help you develop great habits and practices for your future Android work. This is especially true if you find yourself doing more and more work with Android. The approaches and tools for testing, performance profiling, and so forth are great for instilling productive habits and avoiding some classic pitfalls and anti-patterns of good development. If you end up never saying "I'll write the tests later," then this book has served you well.

For the intermediate or advanced Android developer, this book will walk you through details of the current state of the art in Android tool chains; you'll see how best to refactor and improve existing code and applications, and it will push you to embrace some of the advanced topics you might have put off until now. If you've never thought of NDK-based development, you'll learn how to do it right the first time. If you've never had the wherewithal to do multiplatform, multihandset testing and modeling, you'll take the plunge and see what you've been missing all this time.

What You Need Before You Begin

To get the most out of this book, having a few of the housekeeping items sorted out up front will remove distractions later, and it will let you get straight to implementing the tools and techniques you'll learn in each chapter.

An Actual Android Application

To get the best return from this book it will help if you have already written one or two Android apps. They don't even need to have made it all the way to Google Play; but ideally it helps if you've gone through the process and have real-world users who have kicked the tires on your Android app, and you've made revisions based on their feedback or reviews.

A Working Development Environment

You need to have the Android SDK installed with the IDE of your choice: either Eclipse with the ADT toolset; Android Developer Studio; or for the more adventurous, one of the exotic third-party development environments like Intel's Beacon Mountain. You'll need an actual device to follow along with some of our examples, but the emulator will do for most of the code in the book.

All the Bells and Whistles

In addition to the stock Android Developer Studio, Eclipse with ADT, or other IDE, you should also ensure that you have the optional libraries available for the Android SDK. These include the SDK Build-tools, the Google APIs associated with your SDK release level, Android Support Library, and Web Driver and USB driver if available for your operating system.

As each chapter unfolds, you will also be introduced to specific additional tools for unit testing, handset diversity testing, performance profiling and so on. We'll discuss those tools one by one in the relevant chapters.

Source Code for the Sample Application

The Android app we're using in each of the chapters is a simple to-do list and task reminder application. You should download the code from `www.apress.com/9781430258575/` so you can follow along. We'll be using the to do list app to show best practices for Android walking you through design patterns, performance issues, security problems and more in each chapter.

What's in This Book

Here's a chapter-by-chapter summary of what you can expect over the course of this book:

Chapter 2: We begin in Chapter 2 with Patterns. You may already have some familiarity with Android's user interface (UI) patterns, which help create a consistent user experience (UX) across multiple devices. You'll also learn about how you can use other libraries such as ActionBarSherlock and NineOldAndroids to help your users on older devices get a more up-to-date Android experience.

Chapter 3: Following on from UI and UX patterns, Chapter 3 looks at implementing the MVC and MVVM developer design patterns as an alternative to the standard Android design before we dive deeply into Android Annotations and how that can help you create clean understandable Android code.

Chapter 4: Chapter 4 takes a close look at the basic Agile elements of test-driven Development (TDD), behavior-driven design (BDD), and continuous integration (CI) that you can use during development. We look at the unit testing available in the Android SDK and the benefits of looking further afield at tools such as Roboelectric, Calabash, and Jenkins and how you can use them to create a more efficient Agile development environment.

Chapter 5: Android allows you to incorporate C++ code directly using the Android NDK, but there can be a significant performance hit because of the context switch between Java and C++. There are still times, however, when it makes more sense to use new or existing C++ code in Android without porting it to Java. Chapter 5 looks at the reasons when C++ is the right answer and the best way to approach using it for Android.

Chapter 6: Chapter 6 is an up-to-date look at several industry-standard Top 10 security lists that have emerged to give you a much better idea on the do's and don'ts of Android security. The chapter ends with a new list that combines the best elements of Google and OWASP's top 10 lists.

Chapter 7: Device testing can be the bane of Android development. Whether you want to create your own testing platform or using one of the many online services Chapter 8 looks at practical approaches to tame device fragmentation.

Chapter 8: For most Android applications in the business world, the Android part of the application acts as a client to a back-end server. Information is usually but not always sent as JSON via a REST API. Chapter 8 explores in depth how to talk to both REST and SOAP APIs. You'll learn how to create a REST API and why the Richardson Maturity model is important for the longevity of your API. You'll also create your own web services using Google App Engine.