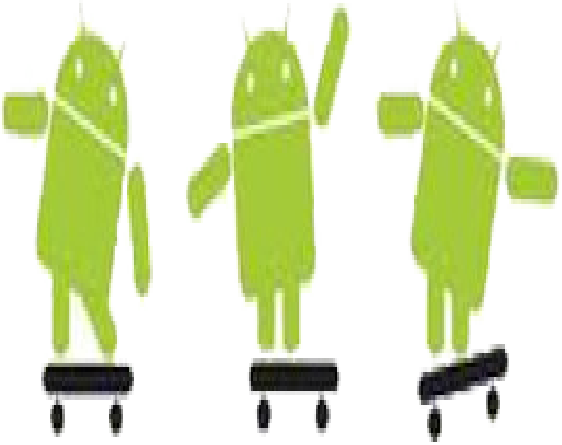
Android App Development

Professionally designing, developing & customizing diverse android based Solutions.



Application Fundamentals

- Android apps are written in Java
- The Android OS is a multi-user Linux system
- Android app lives in its own security sandbox
- System assigns each app a unique Linux user ID
- Each process has its own virtual machine
- Every application runs in its own Linux process

WHY DEVELOP FOR MOBILE?

In market terms, the emergence of modern mobile smart-phones — multifunction devices including a phone but featuring a full-featured web browser, cameras, media players, Wi-Fi, and location-based services — has fundamentally changed the way people interact with their mobile devices and access the Internet.

Mobile-phone ownership easily surpasses computer ownership in many countries, with more than 3 billion mobile phone users worldwide. 2009 marked the year that more people accessed the Internet for the first time from a mobile phone rather than a PC. Many people believe that within the next 5 years more people will access the Internet by mobile phone rather than using personal computers.

The increasing popularity of modern smart-phones, combined with the increasing availability of high speed mobile data and Wi-Fi hotspots, has created a huge opportunity for advanced mobile applications.

The ubiquity of mobile phones, and our attachment to them, makes them a fundamentally different platform for development from PCs. With a microphone, camera, touchscreen, location detection,and environmental sensors, a phone can effectively become an extra-sensory perception device.

Smart-phone applications have changed the way people use their phones. This gives you, the application developer, a unique opportunity to create dynamic, compelling new applications that become a vital part of people's lives.

Factors Driving Android’s Adoption

Developers have always been a critical element within the Android ecosystem, with Google and the OHA betting that the way to deliver better mobile software to consumers is to make it easier for developers to write it.

As a development platform, Android is powerful and intuitive, enabling developers who have never programmed for mobile devices to create innovative applications quickly and easily. It’s easy to see how compelling Android applications have created demand for the devices necessary to run them, particularly when developers write applications for Android because they can’t write them for other platforms.

As Android expands into more form-factors, with increasingly powerful hardware, advanced sensors, and new developer APIs, the opportunities for innovation will continue to grow.

Open access to the nuts and bolts of the underlying system is what’s always driven software development and platform adoption. The Internet’s inherent openness has seen it become the platform for a multimillion-dollar industry within 10 years of its inception. Before that, it was open systems such as Linux and the powerful APIs provided as part of the Windows operating system that enabled the explosion in personal computers and the movement of computer programming from the arcane to the mainstream.

This openness and power ensure that anyone with the inclination can bring a vision to life at minimal cost.

Environment Setup

Four basic steps to get an App Running

- Step 1: Install JDK
- Step 2: Configure the SDK
- Step 3: Create your first Android App
- Step 4: Build and Run your App

ANDROID DEVELOPMENT TOOLS

The Android SDK includes several tools and utilities to help you create, test, and debug your projects. A detailed examination of each developer tool is outside the scope of this book, but it's worth briefly reviewing what's available. For additional details, check out the Android documentation at http://developer.android.com/guide/developing/tools/index.html.

As mentioned earlier, the ADT plug-in conveniently incorporates many of these tools into the Eclipse IDE, where you can access them from the DDMS perspective, including the following:

- **The Android Virtual Device and SDK Managers —** Used to create and manage AVDs and to download SDK packages, respectively. The AVD hosts an Emulator running a particular build of Android, letting you specify the supported SDK version, screen resolution, amount of SD card storage available, and available hardware capabilities (such as touchscreens and GPS).

- **The Android Emulator —** An implementation of the Android VM designed to run within an AVD on your development computer. Use the Emulator to test and debug your Android applications.
- **Dalvik Debug Monitoring Service (DDMS)** — Use the DDMS to monitor and control the Emulators on which you're debugging your applications.
- **Android Debug Bridge (ADB) —** A client-server application that provides a link to virtual and physical devices. It lets you copy files, install compiled application packages (.apk), and run shell commands.
- **Logcat —** A utility used to view and filter the output of the Android logging system
- **Android Asset Packaging Tool (AAPT)** — Constructs the distributable Android package files (.apk).

The following additional tools are also available:

- **SQLite3 —** A database tool that you can use to access the SQLite database files created and used by Android.

- **Traceview and dmtracedump —** Graphical analysis tools for viewing the trace logs from your Android application.

- **Hprof-conv —** A tool that converts HPROF profiling output files into a standard format to view in your preferred profiling tool.

- **MkSDCard —** Creates an SD card disk image that can be used by the Emulator to simulate an external storage card.

- **Dx —** Converts Java .class bytecode into Android .dex bytecode.

- **Hierarchy Viewer —** Provides both a visual representation of a layout's View hierarchy to debug and optimize your UI, and a magnified display to get your layouts pixel-perfect.

- **Lint —** A tool that analyzes your application and its resources to suggest improvements and optimizations.

- **Draw9patch:** A handy utility to simplify the creation of Nine Patch graphics using a WYSIWYG editor.

- **Monkey and Monkey Runner:** Monkey runs within the VM, generating pseudo-random user and system events. Monkey Runner provides an API for writing programs to control the VM from outside your application.

- **ProGuard —** A tool to shrink and obfuscate your code by replacing class, variable, and method names with semantically meaningless alternatives. This is useful to make your code more difficult to reverse engineer.

ANDROID SDK FEATURES

The following list highlights some of the most noteworthy Android features:

- GSM, EDGE, 3G, 4G, and LTE networks for telephony or data transfer, enabling you to make or receive calls or SMS messages, or to send and retrieve data across mobile networks
- Comprehensive APIs for location-based services such as GPS and network-based location detection
- Full support for applications that integrate map controls as part of their user interfaces
- Wi-Fi hardware access and peer-to-peer connections
- Full multimedia hardware control, including playback and recording with the camera and microphone
- Media libraries for playing and recording a variety of audio/video or still-image formats

- APIs for using sensor hardware, including accelerometers, compasses, and barometers

- Libraries for using Bluetooth and NFC hardware for peer-to-peer data transfer

- IPC message passing

- Shared data stores and APIs for contacts, social networking, calendar, and multi-media

- Background Services, applications, and processes

- The ability to integrate application search results into the system searches

- An integrated open-source HTML5 WebKit-based browser

- Mobile-optimized, hardware-accelerated graphics, including a path-based 2D graphics library and support for 3D graphics using OpenGL ES 2.0

- An application framework that encourages the reuse of application components and the replacement of native applications

APPLICATION MARKETING, PROMOTION, AND DISTRIBUTION STRATEGIES

The first step in effectively marketing and promoting your application is ensuring that you provide the full set of high quality assets for your Google Play listing.

Several promotional opportunities are available within Google Play (described in the next section). However, with more than 450,000 other applications available, it's important that you consider alternative avenues for marketing and promotion rather than simply launching your application.

While your marketing and promotion strategies will vary widely depending on your goals and budget, the following list details some of the most effective techniques to consider:

- **Offline cross promotion** — If you have a significant offline presence (such as a stores or branches), or a large media presence (such as within newspapers, magazines, or on TV), cross promoting your application through those channels can be a particularly effective way to increase awareness and help to ensure users trust the download. Traditional advertising techniques such as TV and newspaper advertisements can be extremely effective in raising awareness of your application.

- **Online cross promotion** — If you have a significant web presence, promoting your application through direct links to Google Play can be an effective way to drive downloads. If your application provides a better user experience than your mobile website, you can detect browser visitors from Android devices and direct them to Google Play to download your native app.

- **Third-party promotion** — Distributing a promotional video on YouTube and leveraging social networks, blogs, press releases, and online review sites can help provide positive word of mouth.

- **Online advertising** — Online advertising using in-app ad networks (such as AdMob) or traditional search-based advertising (such as Google AdWords) can drive significant impressions and downloads for your application.

Application Launch Strategies

Ratings and reviews can have a significant impact on your application's ranking in category lists and within Google Play search results. As a result, it can be difficult to recover from a poor launch. The following list describes some of the strategies you can use to ensure a successful launch:

- **Iterate on features not quality** — A poorly implemented but feature-rich application will receive worse reviews than a well-polished application that doesn't do everything. If you are using an agile approach of releasing early and often, ensure each release is of the same high quality, adding new features as part of each release. Similarly, each release should be more polished and stable than the last.

- **Create high quality Google Play assets** — The first impression your application makes is through its appearance in Google Play. Maximize the likelihood of that impression resulting in an installation by creating assets that represent the quality of your application.

- **Be honest and descriptive** — Disappointed users who find your application is not as it was described are likely to uninstall it, rate it poorly, and leave negative comments.

Promotion Within Google Play

In addition to the effect of reviews, downloads, and installs will have on your Google Play listing, there are several automated and curated lists that are used by the Google Play editorial team to highlight high quality applications.

Additionally, a small number of applications are chosen as “featured” applications, receiving priority placement in Google Play. Featured applications typically receive a significant boost in download numbers, making featuring highly prized goal.

While the criteria used to determine which applications become featured is not publicly available, there are certain criteria that have generally become associated with featured applications, including:

- **High quality and innovative** — The featured applications in Google Play act as a showcase for the platform. As a result, the easiest way to be featured is by creating a high quality application that is useful and innovative.

- **A high degree of fit-and-finish** — The listings for featured applications include all the requisite promotional assets, while the applications themselves have few bugs and a high quality user interface.

- **Broad device and platform support** — Featured applications typically support a broad range of device types and platform versions, including both handsets and tablets.

- **Use of newly released features** — Applications that leverage hardware features and software APIs from new Android platform releases offer an opportunity for the Google Play team to highlight those new features to reviewers and early adopters.

- **Consistency with the platform user experience** — Featured applications provide a compelling user experience that is consistent with the UI and interaction models offered by the Android platform.