INTRODUCTION

ANDROID ECLAIR

Android is a mobile operating that controls a mobile device or information appliance—similar in principle to an operating system such as Windows, Mac OS, or Linux that controls a desktop computer or laptop. Android is a mobile operating system initially developed by Android Inc. Android was purchased by Google in 2005. Android is based upon a modified version of the Linux kernel.

Android operating system initially developed by Android Inc. Android was bought by Google in 2005. Android is based upon a modified version of the Linux kernel. Google and other members of the Open Handset Alliance collaborated to develop and release Android to the world. The Android Open Source Project (AOSP) is tasked with the maintenance and further development of Android. Unit sales for Android OS smartphones ranked first among all smartphone OS handsets sold in the U.S. in the second and third quarters of 2010, with a third quarter market share of 43.6%.

Android has a large community of developers writing application programs ("*apps*") that extend the functionality of the devices. There are currently over 100,000 apps available for Android. Android Market is the online app store run by Google, though apps can be downloaded from third party sites (except on AT&T, which disallows this). Developers write in the Java language, controlling the device via Google-developed Java libraries.

The unveiling of the Android distribution on 5 November 2007 was announced with the founding of the Open Handset Alliance, a consortium of 79 hardware, software, and telecom companies devoted to advancing open standards for mobile devices. Google released most of the Android code under the Apache License, a free software and open source license.

The Android operating system software stack consists of Java applications running on a Java based object oriented application framework on top of Java core libraries running on a Dalvik virtual machine featuring JIT compilation. Libraries written in C include the surface manager, OpenCore media framework, SQLite relational database management system, OpenGL ES 2.0 3D graphics API, WebKit layout engine, SGL graphics engine, SSL, and Bionic libc.

ANDROID 2.0/2.1 ECLAIR

The Android operating system consists of 12 million lines of code including 3 million lines of XML, 2.8 million lines of C, 2.1 million lines of Java, and 1.75 million lines of C++. Android would run on Open Handset Alliance devices. It includes an operating system, middleware and key applications such as email client, calendar, maps, browser, and contacts. It also takes the basics one step further by merging contacts with maps. For example, if you have a contacts' address stored on you mobile device you can bring up that location on a map with one click of a button. No need to open a browser, navigate to Google maps or MapQuest, type in the address and wait for it to load.

HISTORY

1In July 2005, Google acquired Android Inc., a small startup company based in Palo Alto, California, USA. Android's co-founders who went to work at Google included Andy Rubin (co-founder of Danger), Rich Miner (co-founder of Wildfire Communications, Inc.), Nick Sears (once VP at T-Mobile), and Chris White (headed design and interface development at WebTV). At the time, little was known about the functions of Android, Inc. other than that they made software for mobile phones. This began rumors that Google was planning to enter mobile phone market.

On November 5th, 2007 Google announced that they would NOT be releasing a gPhone but instead, had formed the Open Handset Alliance and Android, the world's first truly open and complete mobile platform. Thirty-four companies joined together to form the OHA and Android history began. One week later, Google announced $10 Million Dollars in prizes would be awarded in the Android Developer's Challenge. The first Android SDK was released,

On 9 December 2008, it was announced that 14 new members would be joining the Android Project, including Packet Video, ARM Holdings, Atheros Communications, Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, and Vodafone Group Plc.

Android 1.0 was released on 23rd September 2008.The first mobile to run Android 1.0 was HTC Dream (or) HTC G1.This mobile was officially marketed by Google. Android 1.1 was release on 9th February 2009 for Android Dev Phones (ADP).This update included some new features like 'Search by Voice' and 'Paid Applications' in Android Market

Android Cup Cake was released on On 30 April 2009 included the following updates Ability to record and watch videos through camcorder mode Uploading videos to YouTube and pictures to Picasa directly from the phone A new soft-keyboard with text-prediction Bluetooth A2DP and AVRCP support

Android Donut was released in 30 April 2009 including the following updates An improved Android Market experience An integrated camera, camcorder, and gallery interface Gallery now enables users to select multiple photos for deletion Updated technology support for CDMA/EVDO, 802.1x, VPNs, and a text-to-speech engine Support for WVGA screen resolutions

Android Eclair was released in26 October 2009 including the following updates Optimized hardware speed Support for more screen sizes and resolutions Revamped UI New Browser UI and HTML5 support Better contrast ratio for backgrounds Improved Google Maps 3.1.2 Built in flash support for Camera MotionEvent class enhanced to track multi-touch events Improved virtual keyboard Bluetooth 2.1Live Wallpapers

ANDROID 2.0/2.1 ECLAIR

Android Froyo was released in 20 May 2010 including the following updates General Android OS speed, memory, and performance optimizations Increased Microsoft Exchange support (security policies, auto-discovery, GAL look-up, calendar synchronization, remote wipe) Wi-Fi hotspot functionality

Android Gingerbread was released in 6 December 2010 including the following updates Updated user interface design Support for extra-large screen sizes and resolutions (WXGA and higher) Native support for SIP VoIP telephony Support for WebM/VP8 video playback, and AAC audio encoding

Android 2.4 is named as Icecream and its next version is named as Honeycomb and will be available by 2011

FEATURES

Handset layouts

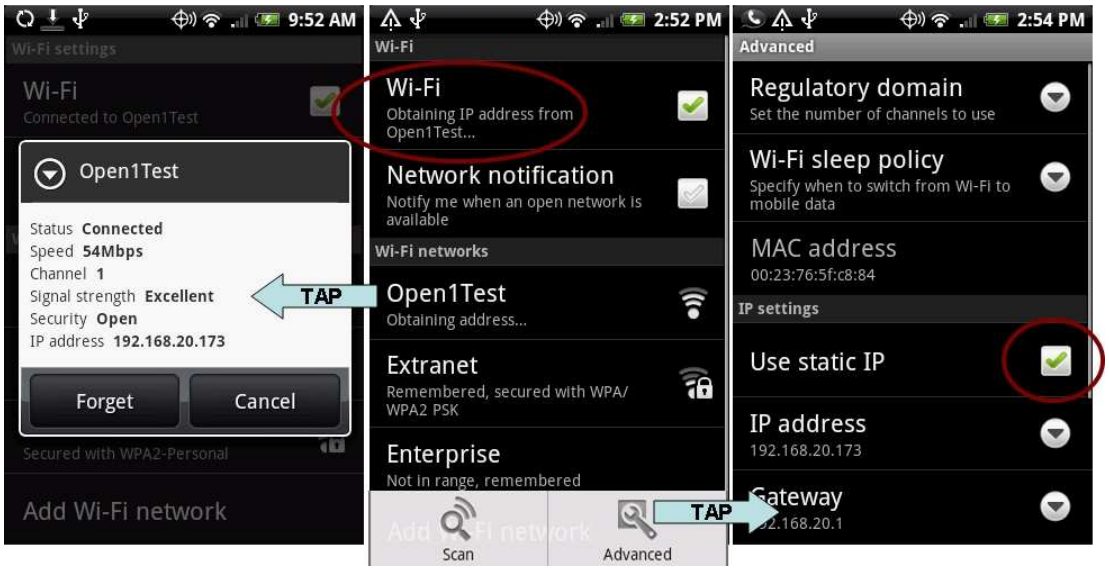
The platform is adaptable to larger, VGA, 2D graphics library, 3D graphics library based on OpenGL ES 2.0 specifications, and traditional smartphone layouts.

Storage

SQLite, a lightweight relational database, is used for data storage purposes

Connectivity

Android supports connectivity technologies including GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, and WiMAX.



Web browser

The web browser available in Android is based on the open-source WebKit layout engine, coupled with Chrome's V8 JavaScript engine. The browser scores a 93/100 on the Acid3 Test.

Development environment

Includes a device emulator, tools for debugging, memory and performance profiling. The integrated development environment (IDE) is Eclipse (currently 3.4 or 3.5) using the Android Development Tools (ADT) Plug-in.

- **Java support**

While Android applications are written in Java, there's no Java Virtual Machine in the platform and Java byte code is not executed. Java classes get recompiled into Dalvik executable and run on Dalvik virtual machine. Dalvik is a specialized virtual machine designed specifically for Android and optimized for battery-powered mobile devices with limited memory and CPU. J2ME support can be provided via third-party-application such as the J2ME MIDP Runner.

- **Media support**

Android supports the following audio/video/still media formats: WebM, H.263, H.264 (in 3GP or MP4 container), MPEG-4 SP, AMR, AMR-WB (in 3GP container), AAC, HE-AAC (in MP4 or 3GP container), MP3, MIDI, Ogg Vorbis, WAV, JPEG, PNG, GIF, BMP.

- **Streaming media support**

RTP/RTSP streaming (3GPP PSS, ISMA), HTML progressive download (HTML5 <video> tag). Adobe Flash Streaming (RTMP) is supported through Adobe Flash Player plugin. Microsoft Smooth Streaming is planned to be supported through the awaited port of Silverlight plugin to Android. Adobe Flash HTTP Dynamic Streaming is planned to be supported through an upgrade of the Flash plugin.

- **Additional hardware support**

Android can use video/still cameras, touch screens, GPS, accelerometers, gyroscopes, magnetometers, proximity and pressure sensors, thermometers, accelerated 2D bit blits (with hardware orientation, scaling, pixel format conversion) and accelerated 3D graphics.

Bluetooth

Support for A2DP and AVRCP were added in version 1.5; sending files (OPP) and accessing the phone book (PBAP) were added in version 2.0; and voice dialing and sending contacts between phones were added in version 2.2.

Multitasking

Multitasking of applications is available.

TimeScape

Sony Ericsson's social media mashup app TimeScape is a real treat. It's a feature of the user interface on its new Android phones

The TimeScape app flows emails, text messages, notes, Twitter and Facebook notifications into one very visual home page, that's very intuitive to operate through the touchscreen.

Timescape Home Timescape tile Timescape tile open

ARCHITECTURE

LINUX KERNEL

Android is built on top of a solid and proven foundation: the Linux kernel. Created by Linus Torvalds in 1991, Linux can be found today in everything from wristwatches to supercomputers. Linux provides the hardware abstraction layer for Android, allowing Android to be ported to a wide variety of platforms in the future.

Internally, Android uses Linux for its memory management, process management, networking, and other operating system services. The Android phone user will never see Linux, and your programs will not make Linux calls directly. As a developer, though, you'll need to be aware it's there.

NATIVE LIBRARIES

The next layer above the kernel contains the Android native libraries. These shared libraries are all written in C or C++, compiled for the particular hardware architecture used by the phone, and preinstalled by the phone vendor.

Some of the most important native libraries include the following:

Surface Manager:
Android uses a compositing window manager similar to Vista or Compiz, but it's much simpler. Instead of drawing directly to the screen buffer, your drawing commands go into off-screen bitmaps that are then combined with other bitmaps to form the display the user sees.

2D and 3D graphics:
Two- and three-dimensional elements can be combined in a single user interface with Android. The library willuse 3D hardware if the device has it or a fast software renderer if it doesn't. See Chapter 4, Exploring 2D Graphics, on page 75 and Chapter 10, 3D Graphics in OpenGL , on page 200.

Media codecs:
Android can play video and record and play back audio in a variety of formats including AAC, AVC (H.264), H.263, MP3, and MPEG-4.

SQL database:
Android includes the lightweight SQLite database You can use this for persistent storage in your application.
Browser engine:
For the fast display of HTML content, Android uses the WebKit library.4 This is the same engine used in the Google Chrome browser, Apple's Safari browser, the Apple iPhone, and Nokia's S60 platform.

ANDROID RUNTIME
Also sitting on top of the kernel is the Android runtime, including the Dalvik virtual machine and the core Java libraries. Dalvik is a virtual machine (VM) designed and written by Dan Bornstein at Google. Your code gets compiled into machine- independent instructions called byte codes, which are then executed by the Dalvik VM on the mobile device. Although the byte code formats are a little different, Dalvik is essentially a Java virtual machine optimized for low memory requirements. It allows multiple VM instances to run at once and takes advantage of the underlying operating system (Linux) for security and process isolation.

The Dalvik VM is Google's implementation of Java, optimized for mobile devices. All the code you write for Android will be written in Java and run within the VM. Dalvik differs from traditional Java in two important ways:

The Dalvik VM runs .dex les, which are converted at compile time from standard .class and .jar les. .dex les are more compact and efficient than class les, an important consideration for the limited memory and battery-powered devices that Android targets.

The core Java librari both the Java Standard Edition (Java SE) libraries and the Java Mobile Edition (Java ME) libraries. There is a substantial amount of overlap, however. In Appendix A, on page 280, you'll and comparison of Android and standard Java libraries.

Application Framework

Sitting above the native libraries and runtime, you'll and the Application Framework layer. This layer provides the high-level building blocks you will use to create your applications. The framework comes preinstalled with Android, but you can also extend it with your own components as needed.

The most important parts of the framework are as follows:

Activity Manager :

This controls the life cycle of applications and maintains a common "backstack" for user navigation.

Content providers:

These objects encapsulate data that needs to be shared between applications, such as contacts.

Resource manager :

Resources are anything that goes with your program that is not code

Location manager :

An Android phone always knows where it is.

Notification manager :

Events such as arriving messages, appointments, proximity alerts, alien invasions, and more can be presented in an unobtrusive fashion to the user.

APPLICATIONS

The highest layer in the Android architecture diagram is the Applications and Widgets layer. Think of this as the tip of the Android iceberg. End users will see only these programs, blissfully unaware of all the action going on below the waterline. As an Android developer, however, you know better. Applications are programs that can take over the whole screen and interact with the user. On the other hand, widgets (which are sometimes called gadgets), operate only in a small rectangle of the Home screen application.

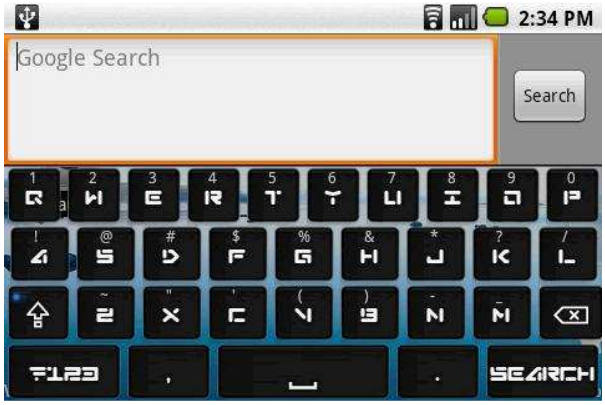
When someone buys an Android phone, it will come prepackaged with a number of standard system applications, including the following:

- Phone dialer
- Contacts
- Email
- Web browser
- Android Market

SERVICES

SMART VIRTUAL KEYBOARD

The addition of a virtual keyboard means that Android 1.5 devices can support both physical and virtual keyboards, packing the best of both worlds. It's up to you to use whichever input method fits you in a specific scenario.
Android's virtual keyboard is provided in portrait or landscape orientation and works in any application, including Gmail, the browser, SMS and even in third-party programs.
It comes with an auto-correct feature, suggestions and user dictionary for custom words. You can also set it up so that it gives you tactile feedback by vibrating the screen. Unlike rival mobile platforms, Android 1.5 supports user installation of third-party virtual keyboards.



DATA STORAGE

Android provides several options for you to save persistent application data. The solution you choose depends on your specific needs, such as whether the data should be private to your application or accessible to other applications (and the user) and how much space your data requires.

Your data storage options are the following:

<u>Shared Preferences</u>

Store private primitive data in key-value pairs.

<u>Internal Storage</u>

Store private data on the device memory.

<u>External Storage</u>

Store public data on the shared external storage.

SQLite Databases

Store structured data in a private database.

Network Connection

Store data on the web with your own network server.

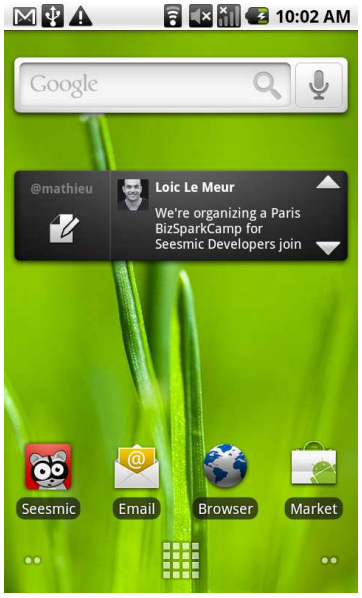
Android provides a way for you to expose even your private data to other applications — with a content provider. A content provider is an optional component that exposes read/write access to your application data, subject to whatever restrictions you want to impose. For more information about using content providers, see the Content Providers documentation.

◄HOME SCREEN

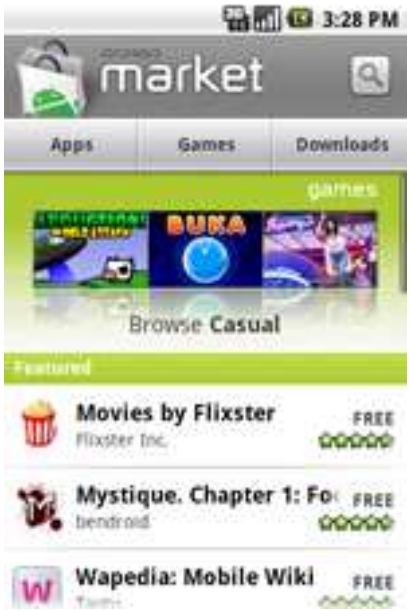
A new tips widget will grace your homescreen. This widget will assist new users on how to configure the home screen, adding shortcuts, adding widgets and how to use multiple home screens.

◄WIDGET SUPPORT

The Android 2.0 update will bring widget support to the Android platform's home screen.When widgets are dropped onto the home screen, they are given a reserved space to display custom content provided by your app. Users can also interact with your app through the widget, for example pausing or switching music tracks. If you have a background service, you can push widget updates on your own schedule, or the AppWidget framework provides an automatic update mechanism.



ANDROID MARKET



New Android Market UI
For devices with Android Market, the latest version improves the overall user experience and makes it easier for users to discover great apps and games from developers.

At the homescreen, users can choose among Apps, Games, and Downloads.

Inside a category, users can explore titles that are Top paid, Top free, and Just in.

For each title, users can now see screenshots submitted by developers in addition to reviews from other users.

MEMORY MANAGEMENT

Android's process and memory management is a little unusual. Like Java and .NET, Android uses its own run time and virtual machine to manage application memory. Unlike either of these frameworks, the Android run time also manages the process lifetimes. Android ensures application responsiveness by stopping and killing processes as necessary to free resources for higher-priority applications.

Each Android application runs in a separate process within its own Dalvik instance, relinquishing all responsibility for memory and process management to the Android run time, which stops and kills processes as necessary to manage resources.

Dalvik and the Android run time sit on top of a Linux kernel that handles low-level hardware interaction including drivers and memory management, while a set of APIs provides access to all of the under- lying services, features, and hardware.

Dalvik Virtual Machine Dalvik is a register-based virtual machine that's been optimized to ensure that a device can run multiple instances efficiently. It relies on the Linux kernel for threading and low-level memory management.

The Dalvik Virtual Machine

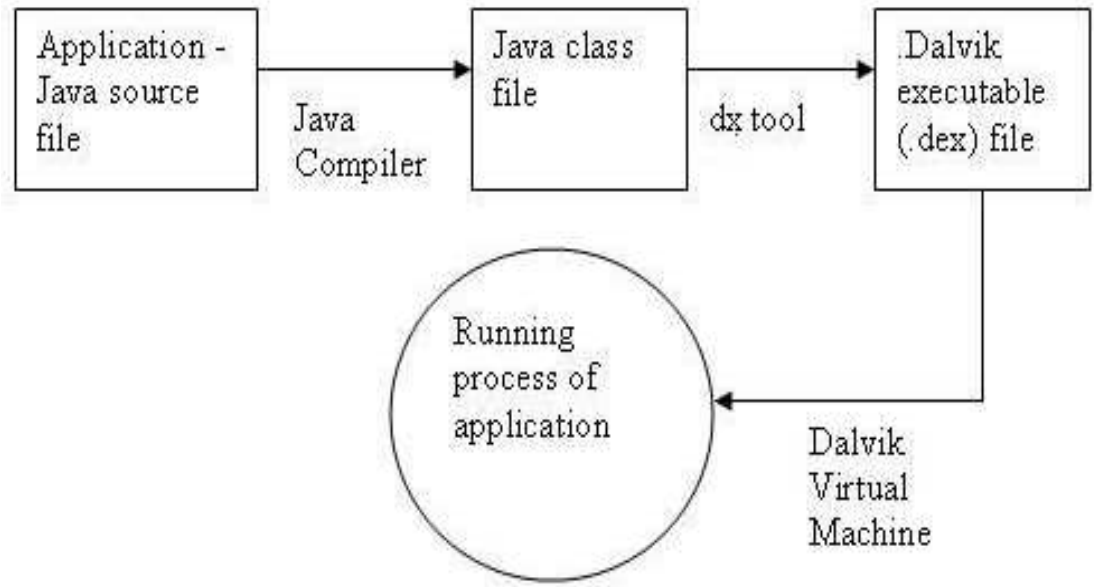
One of the key elements of Android is the Dalvik virtual machine. Rather than use a traditional Java virtual machine (VM) such as Java ME (Java Mobile Edition), Android uses its own custom VM designed to ensure that multiple instances run efficiently on a single device.

The Dalvik VM uses the device's underlying Linux kernel to handle low-level functionality including security, threading, and process and memory management.

All Android hardware and system service access is managed using Dalvik as a middle tier. By using a VM to host application execution, developers have an abstraction layer that ensures they never have to worry about a particular hardware implementation.

The Dalvik VM executes Dalvik executable files, a format optimized to ensure minimal memory foot- print. The **.dex** executables are created by transforming Java language compiled classes using the tools supplied within the SDK.

The Dalvik virtual machine is simple Java interpreter machine, completely optimized for Android platform and which is developed to run on low-end memory mobile devices. One of the prominent aspects in Dalvik its capability to run along an application compilation enhancing the runtime performance of the applications. Dalvik is not exactly, a Java machine, because Dalvik could not read Java code, but consists its own byte code called “dex” and so the executable files compacted using Dalvik holds the file type name '.dex'. Google states that the credit for Androids successful development goes to Dalvik VM, because this type of virtual machine, delivers a good performance over various stages of an application runtime environment, conserving more battery-power during long run of an application.



A small drawback with Dalvik, is its none-compatibility with Java SE libraries, Java ME class libraries and Swing Java libraries, while they cannot be and need not to-be run directly on this virtual machine. It uses its own Apache Harmony Java implementation libraries. Despite of the earlier reason, its escalated Android's value proposition due to its minimal-electrical power consumption, vast library resources, and non-fragmentary application programming interface, unlike its Java rivals.

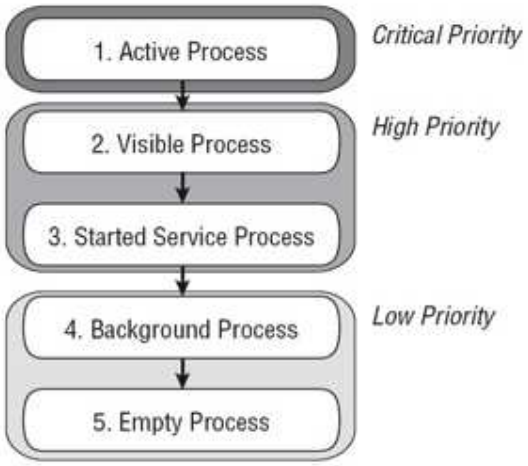
PROCESS MANAGEMENT

Understanding Application Priority and Process States

The order in which processes are killed to reclaim resources is determined by the priority of the hosted applications. An application's priority is equal to its highest-priority component.

Where two applications have the same priority, the process that has been at a lower priority longest will be killed first. Process priority is also affected by interprocess dependencies; if an application has a dependency on a Service or Content Provider supplied by a second application, the secondary application will have at least as high a priority as the application it supports.

All Android applications will remain running and in memory until the system needs its resources for other applications.



It's important to structure your application correctly to ensure that its priority is appropriate for the work it's doing. If you don't, your application could be killed while it's in the middle of something important.

The following list details each of the application states shown in Figure , explaining how the state is determined by the application components comprising it:

Active Processes Active (foreground) processes are those hosting applications with components currently interacting with the user. These are the processes Android is trying to keep responsive by reclaiming resources. There are generally very few of these processes, and they will be killed only as a last resort.

Active processes include:

- Activities in an “active” state; that is, they are in the foreground and responding to user events. You will explore Activity states in greater detail later in this chapter.
- Activities, Services, or Broadcast Receivers that are currently executing an onReceive event handler.
- Services that are executing an onStart, onCreate, or onDestroy event handler.

Visible Processes Visible, but inactive processes are those hosting “visible” Activities. As the name suggests, visible Activities are visible, but they aren’t in the foreground or responding to user events. This happens when an Activity is only partially obscured (by a non-full-screen or transparent Activity). There are generally very few visible processes, and they’ll only be killed in extreme circumstances to allow active processes to continue.

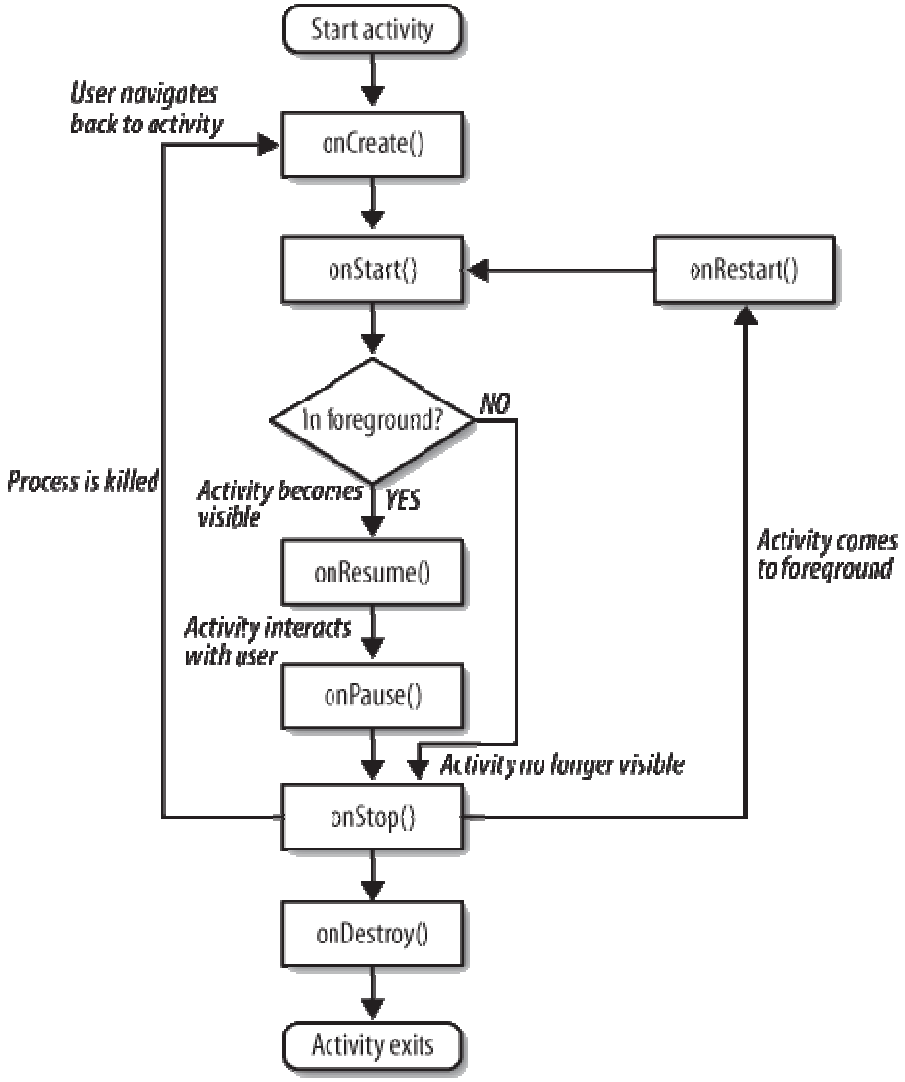
Started Service Processes Processes hosting Services that have been started. Services support ongoing processing that should continue without a visible interface. Because Services don’t interact directly with the user, they receive a slightly lower priority than visible Activities. They are still considered to be foreground processes and won’t be killed unless resources are needed for active or visible processes.

Background Processes Processes hosting Activities that aren’t visible and that don’t have any Services that have been started are considered background processes. There will generally be a large number of background processes that Android will kill using a last-seen-first-killed pat- tern to obtain resources for foreground processes.

Empty Processes To improve overall system performance, Android often retains applications in memory after they have reached the end of their lifetimes. Android maintains this cache to improve the start-up time of applications when they’re re-launched. These processes are rou- tinely killed as required.

Android Activity Lifecycle

Android is designed around the unique requirements of mobile applications. In particular, Android recognizes that resources (memory and battery, for example) are limited on most mobile devices, and provides mechanisms to conserve those resources. The mechanisms are evident in the Android Activity Lifecycle, which defines the states or events that an activity goes through from the time it is created until it finishes running. The lifecycle is shown diagrammatically in Figure .



Your activity monitors and reacts to these events by instantiating methods that override the Activity class methods for each event:

onCreate

Called when your activity is first created. This is the place you normally create your views, open any persistent datafiles your activity needs to use, and in general initialize your activity. When calling onCreate, the Android framework is passed a Bundle object that contains any activity state saved from when the activity ran before.

onStart

Called just before your activity becomes visible on the screen. Once onStart completes, if your activity can become the foreground activity on the screen, control will transfer to onResume. If the activity cannot become the foreground activity for some reason, control transfers to the onStop method.

onResume

Called right after onStart if your activity is the foreground activity on the screen. At this point your activity is running and interacting with the user. You are receiving keyboard and touch inputs, and the screen is displaying your user interface. *onResume is* also called if your activity loses the foreground to another activity, and that activity eventually exits, popping your activity back to the foreground. This is where your activity would start (or resume) doing things that are needed to update the user interface (receiving location updates or running an animation, for *example).*

onPause

Called when Android is just about to resume a different activity, giving that activity the foreground. At this point your activity will no longer have access to the screen, so you should stop doing things that consume battery and CPU cycles unnecessarily. If you are running an animation, no one is going to be able to see it, so you might as well suspend it until you get the screen back. Your activity needs to take advantage of this method to store any state that you will need in case your activity gains the foreground again—and it is not guaranteed that your activity will resume. If the mobile device you are running on runs out of memory, there is no virtual memory on disk to use for expansion, so your activity may have to make way for a system process that needs memory. Once you exit this method, Android may kill your activity at any time without returning control to you.

onStop

Called when your activity is no longer visible, either because another activity has taken the foreground or because your activity is being destroyed.

onDestroy

The last chance for your activity to do any processing before it is destroyed. Normally you'd get to this point because the activity is done and the framework called its finish method. But as mentioned earlier, the method might be called because Android has decided it needs the resources your activity is consuming.

It is important to take advantage of these methods to provide the best user experience possible. This is the first place in this book we've discussed how programming for mobile devices is different from programming for desktop devices, and there will be many more such places as you go through later chapters. Your users will appreciate it if you write your activities with the activity lifecycle in mind, and you will ultimately benefit.

NETWORKING

MOBILE NETWORKS CONNECTIONS

Wi-Fi

A Wi-Fi enabled device such as a personal computer, video game console, smartphone or digital audio player can connect to the Internet when within range of a wireless network connected to the Internet.'Wi-Fi' is not a technical term. However, the Alliance has generally enforced its use to describe only a narrow range of connectivity technologies including wireless local area network (WLAN) based on the IEEE 802.11 standards, device to device connectivity [such as Wi-Fi Peer to Peer AKA Wi-Fi Direct], and a range of technologies that support PAN, LAN and WAN connections.

3G

3G or 3rd Generation, is a generation of standards for mobile phones and mobile telecommunications services fulfilling specifications by the International Telecommunication Union.[1] Application services include wide-area wireless voice telephone, mobile Internet access, video calls and mobile TV, all in a mobile environment.

GPRS

General packet radio service (GPRS) is a packet oriented mobile data service on the 2G and 3G cellular communication systems global system for mobile communications (GSM).

UMTS

Universal Mobile Telecommunications System (UMTS) is one of the third-generation (3G) mobile telecommunications technologies, which is also being developed into a 4G technology.

SECURITY

The Internet may be rife with virus alerts, phishing scams, worms and trojans, but these threats have been slow to infect our mobile phones. They can in principle (there has been a worm on jailbroken iPhones), but as of yet Android phones have not suffered an attack. There have been some scare stories about potential exploits in third-party software for Android cell phones, but this is still the subject of much debate.

Even though the threat is just potential at the moment, you might want to safeguard yourself. It is also worth considering software which would help you to wipe or recover your Android phone if it was stolen or just protect private information from prying eyes. In this article we'll take a look at what is available for people seeking security apps for Android.

Lookout Mobile Security

Here's an excellent, highly rated app that offers a complete suite of protection for your Android phone and it's completely free (no trial or catch). You'll get anti-virus and security to block viruses, malware and spyware. It will scan the apps you download and automatically update the virus definitions with the latest threats. You'll also be able to locate your lost or stolen phone on a map, activate an alarm, and remotely turn on GPS. Lookout even offers a secure online backup and restore feature for your contacts. Of all the security apps for Android that are available in the Android Market Lookout Mobile Security is probably the best.

WaveSecure Mobile Security

Another one of the best security suites for Android is WaveSecure Mobile Security from McAfee. If your phone is lost or stolen you can take advantage of the remote lock feature. You can also remotely wipe the data stored on your phone so no one else can get their hands on your private messages, photos or documents. The ability to backup your phone to a secure online location is very handy and you can restore that data to a new phone or even just access it online if you want. Finally there is the locate and track feature which will show you the location of your phone on a map. It is also capable of tracking SIM cards inserted into the phone and the phone calls made in order to help you recover your phone.
This used to be completely free but sadly now it's just the 7 day trial that's free and after that you have to purchase a license which costs $19.90 for 1 year.

aFirewall Blocker

Are you sick of receiving sales calls or perhaps you want the option to block calls and texts at specific times or from a specific person? Well this Android security app is for you. It allows you to block calls and texts from unknown or private callers, and you can specify a list of unblocked or blocked numbers. You can set it up to forward blocked calls to voicemail if you want or even auto reply with a text. This is a really handy security app for your Android phone and it costs $1.99.

Antivirus

The simply named Antivirus app from NetQin Security is a full Android security suite. This app offers protection against viruses, malware and spyware. It also has a backup and restore feature for contacts and the ability to remotely lock, wipe or locate your missing phone. It will scan new apps on your phone and alert you if a malicious app is found so you can delete it. This Android security app only uses 0.1% of your battery so it won't be a major drain. It is a completely free security app which is available from the Android Market and it is well worth having.

MERITS

Refreshed UI with actionable browser URL bar enables users to directly tap the address bar for instant searches and navigation.

An improved keyboard layout to makes it easier to hit the correct characters and improve typing speed.

License free, royalty free, and open source. That's Android. No costly licensing fees.

In addition, as a result of many mobile phones carrying Google Android, companies will come up with such innovative products like the location – aware services that will provide users with any information they might be in need of.

DE-MERITS

since Google Android is an open source platform driving all Android phones, it could significantly increase the risk of hacking these devices. If this happens, then locking mobiles as a security measure will be a mockery.

Android uses a non-standard jvm: there is no guarantee that the same software will run on multiple devices

Currently Android 2.2 allows users to install application on SD card. But if you are using lower version and have lot of Applications on your phone it will make device slowdown.

CONCLUSION

Android is open to all : industry,developers and users. Participating in many of tha successful open source projects. Aims to be as easy to build for as the web.Google Android is stepping into the next level of mobile internet

The entire system was divided into 3 sub-systems, initially. The andriod application objective is to provide a user interface (UI) which be the one on charge of having the interaction with the users,

The second sub-system, communication module part (micro-controller) considered as an input and output it main function is to receive information from the android phone

The last sub-system is the construction of a transmitter and a receiver operating at 915MHz, it purpose is to interface with the transceiver from the actual automobile.

We can only hope that the next versions of Android have overcome the actual limitations and that the future possibilities became a reality.

BIBILOGRAPHY

We refer the following web pages

http://wapedia.mobi/en/Android

http://developer.android.com/index.html

http://mobworld.wordpress.com

http://www.droidxforums.com/

http://android.nextapp.com