Phone OS

COMP755

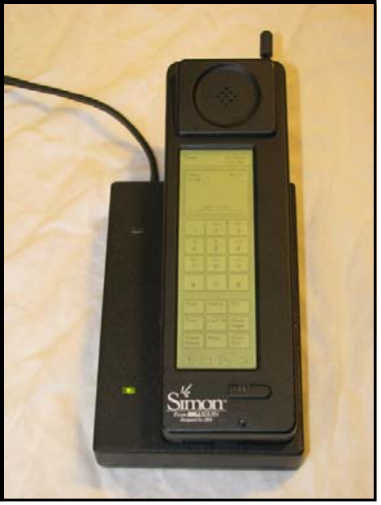
Some slides contain material taken from the Google Android website

OS for Telephones

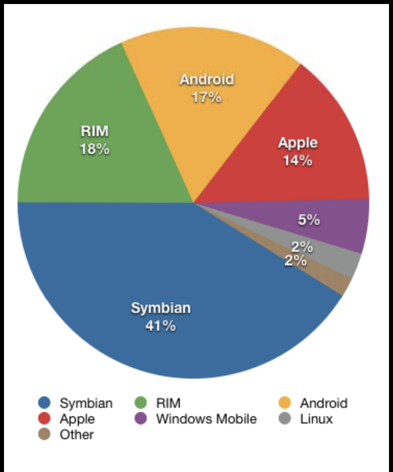
- Smartphones have an operating system that is just as complex as a desktop or laptop
- Telephones are often said to have limited resources, although they are not that limited
- Over 45.5 million people in the United States owned smartphones in 2010 [Wikipedia]
- Smartphone sales only make up 20% of total handset sales [Wikipedia]

First Smartphone

- The IBM Simon Personal Communicator was the first smartphone in 1992
- Originally priced at $899 $1358 in today's dollars
- Had a calendar, address book, calculator, note pad, e-mail and games



There are several phone OS



Windows Phone 7

- Microsoft's latest entry in the phone OS field was released this week

Separation

- Some phone OS separate the "computer" functionality from the "telephone" functionality
- Programs on these systems cannot make phone calls
- Other phone OS allow programs access the full hardware of the device

Android OS

- Android is an operating system for telephones
- Open Source system from Google
- Android is the fastest growing phone OS

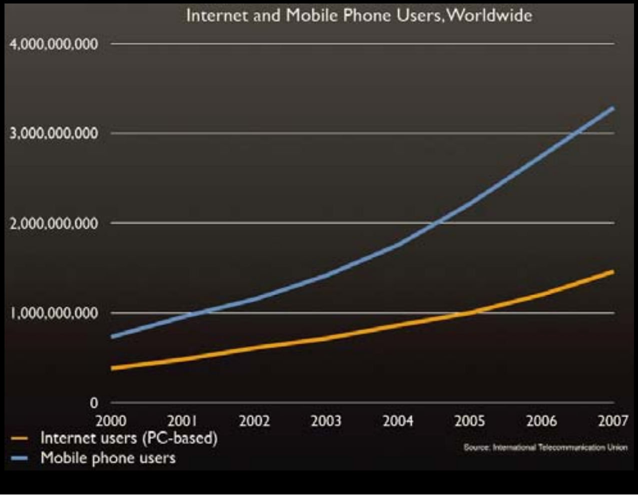
Google™

- Google's mission is "to organize the world's information and make it universally accessible and useful"
- The unofficial company slogan is "Don't be evil"

What does Google sell?

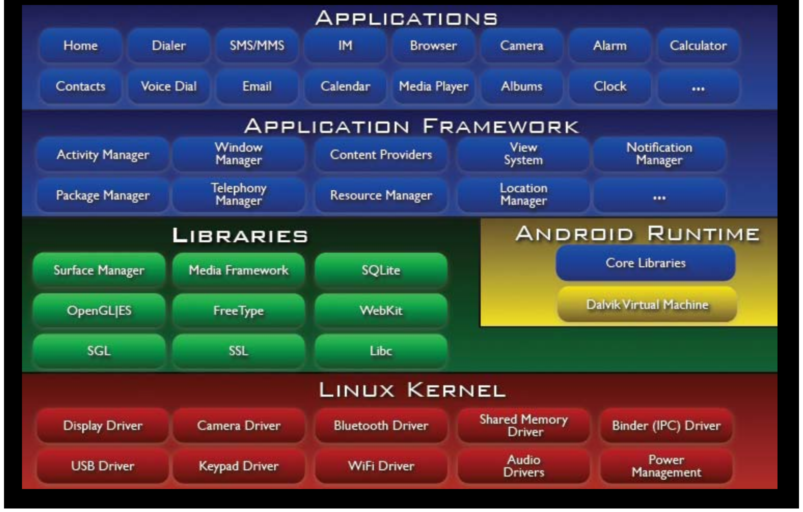
1. Email Service
2. Telephones
3. Advertising
4. All of the above
5. None of the above

Why Google is in the Phone Market



Source: International Telecommunication Union

Overview of Android



APPLICATIONS

Home, Dialer, SMS/MMS, IM, Browser, Camera, Alarm, Calculator, Contacts, Voice Dial, Email, Calendar, Media Player, Albums, Clock, ...

APPLICATION FRAMEWORK

Activity Manager, Window Manager, Content Providers, View System, Notification Manager, Package Manager, Telephony Manager, Resource Manager, Location Manager, ...

LIBRARIES

Surface Manager, Media Framework, SQLite, OpenGL|ES, FreeType, WebKit, SGL, SSL, Libc

ANDROID RUNTIME

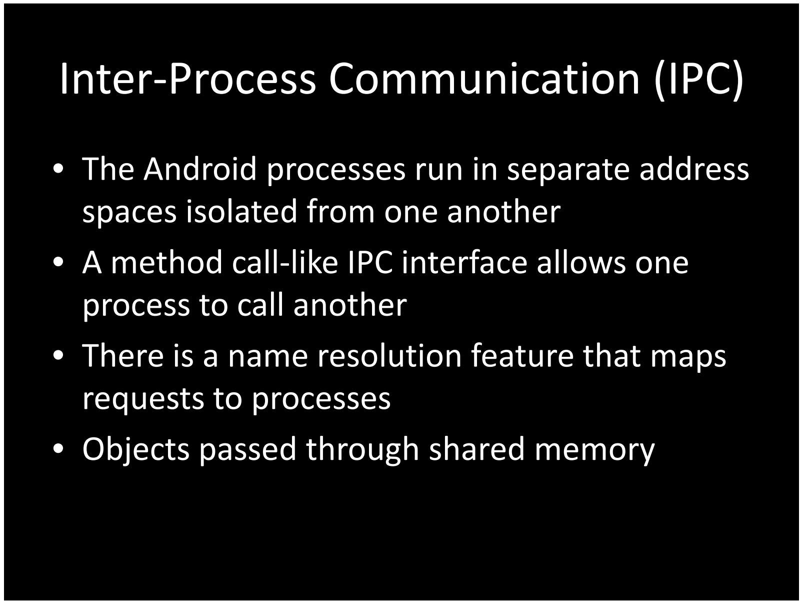
Core Libraries, Dalvik Virtual Machine

LINUX KERNEL

Display Driver, Camera Driver, Bluetooth Driver, Shared Memory Driver, Binder (IPC) Driver, USB Driver, Keypad Driver, WiFi Driver, Audio Drivers, Power Management

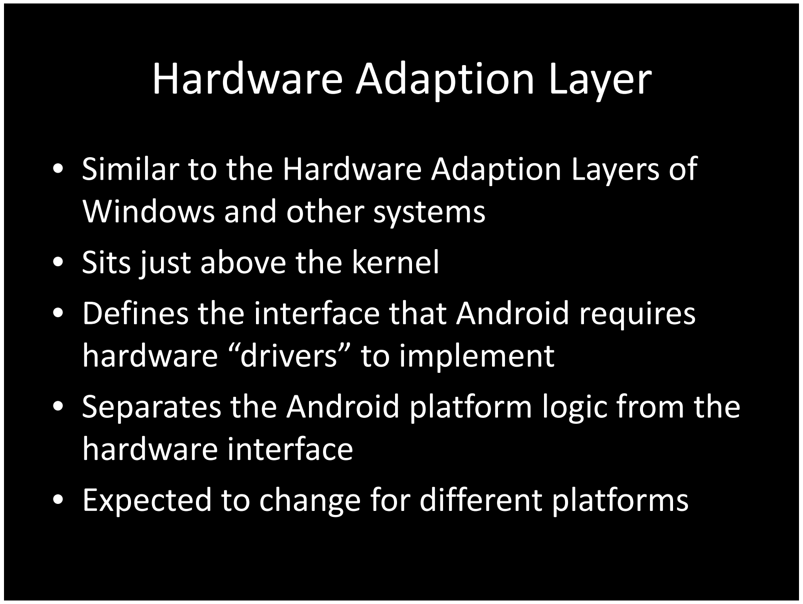
Kernel

- Android is built on the Linux kernel, but *Android is not Linux*
- Many standard Linux features are not included
- Patch of “kernel enhancements” to support Android
- Linux provides memory and process management, security and device drivers
- Linux is open source



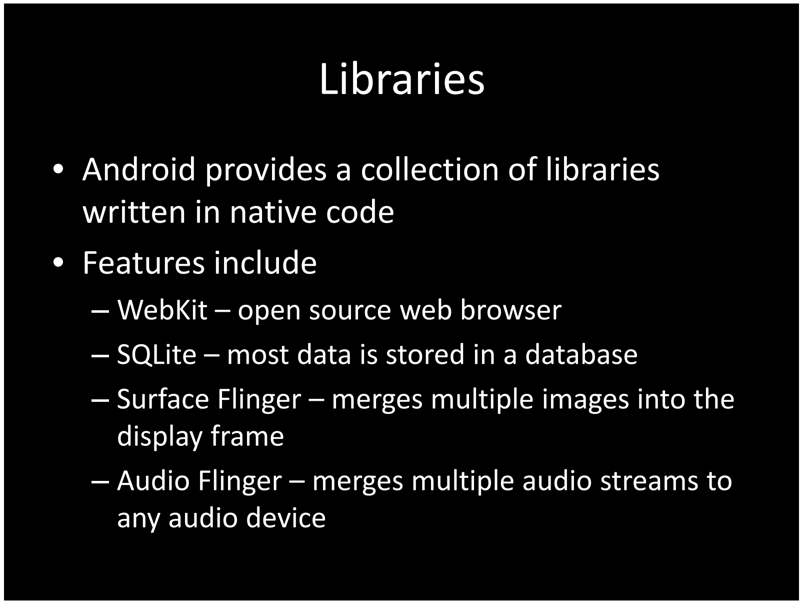
Inter-Process Communication (IPC)

- The Android processes run in separate address spaces isolated from one another
- A method call-like IPC interface allows one process to call another
- There is a name resolution feature that maps requests to processes
- Objects passed through shared memory



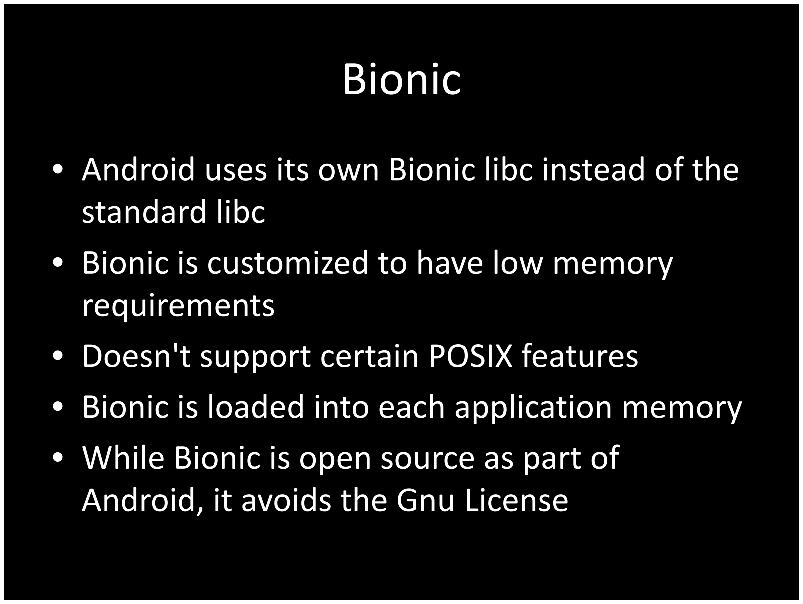
Hardware Adaption Layer

- Similar to the Hardware Adaption Layers of Windows and other systems
- Sits just above the kernel
- Defines the interface that Android requires hardware "drivers" to implement
- Separates the Android platform logic from the hardware interface
- Expected to change for different platforms



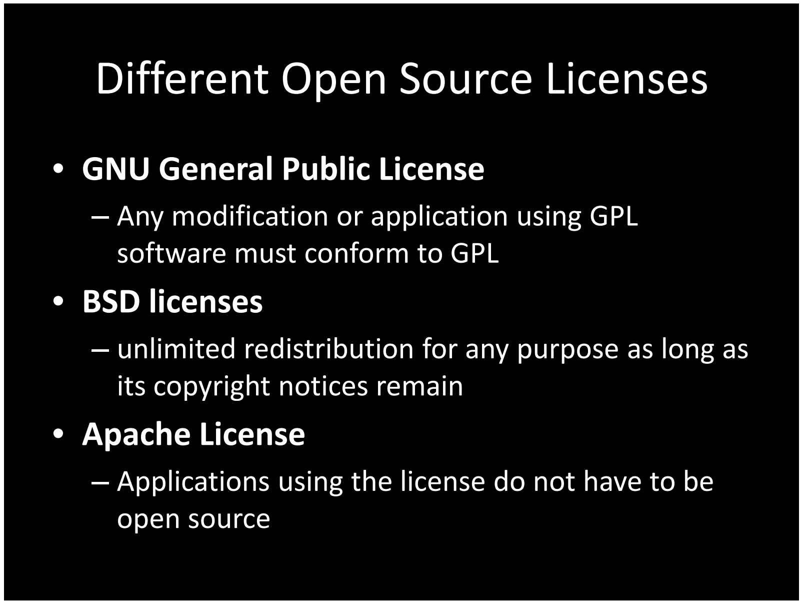
Libraries

- Android provides a collection of libraries written in native code
- Features include
  - WebKit – open source web browser
  - SQLite – most data is stored in a database
  - Surface Flinger – merges multiple images into the display frame
  - Audio Flinger – merges multiple audio streams to any audio device



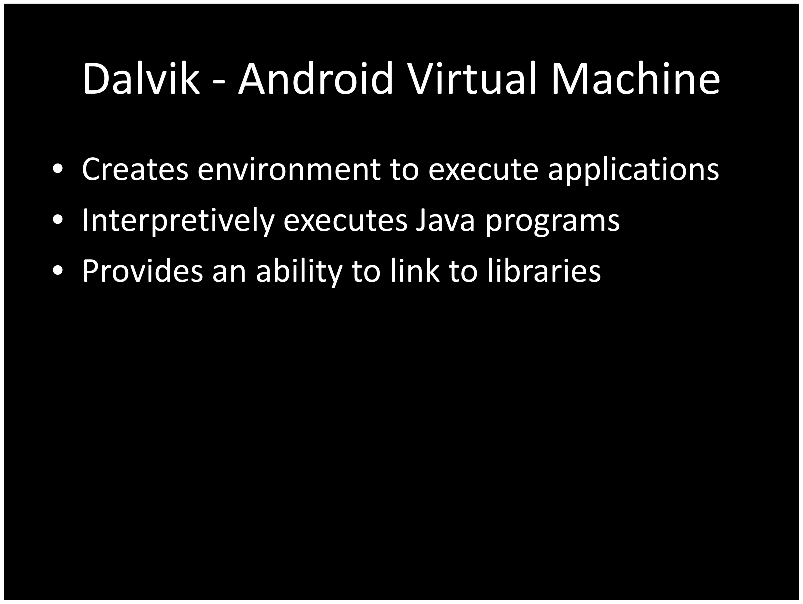
Bionic

- Android uses its own Bionic libc instead of the standard libc
- Bionic is customized to have low memory requirements
- Doesn't support certain POSIX features
- Bionic is loaded into each application memory
- While Bionic is open source as part of Android, it avoids the Gnu License



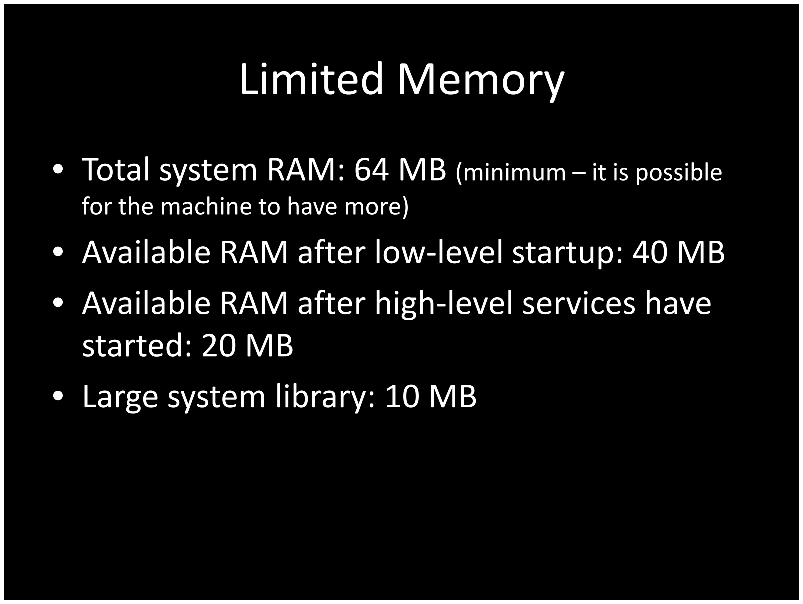
Different Open Source Licenses

- **GNU General Public License**
  - Any modification or application using GPL software must conform to GPL
- **BSD licenses**
  - unlimited redistribution for any purpose as long as its copyright notices remain
- **Apache License**
  - Applications using the license do not have to be open source



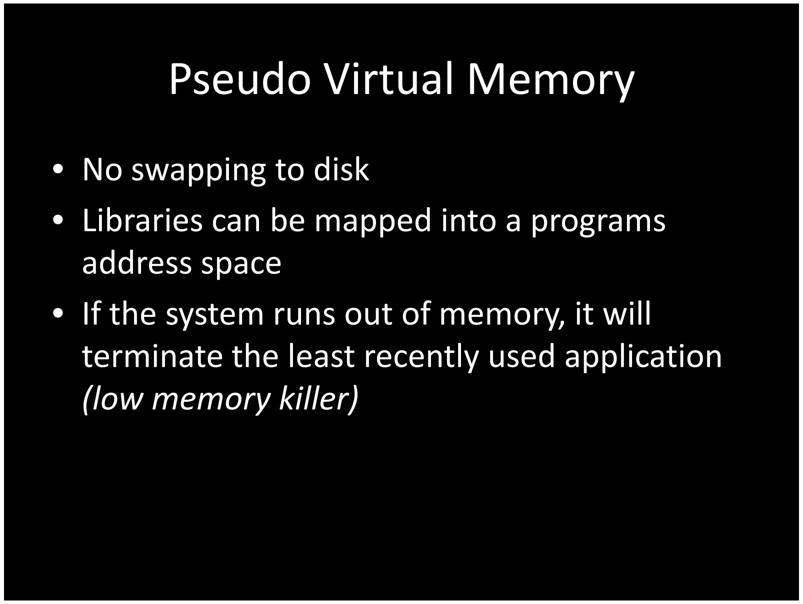
Dalvik - Android Virtual Machine

- Creates environment to execute applications
- Interpretively executes Java programs
- Provides an ability to link to libraries



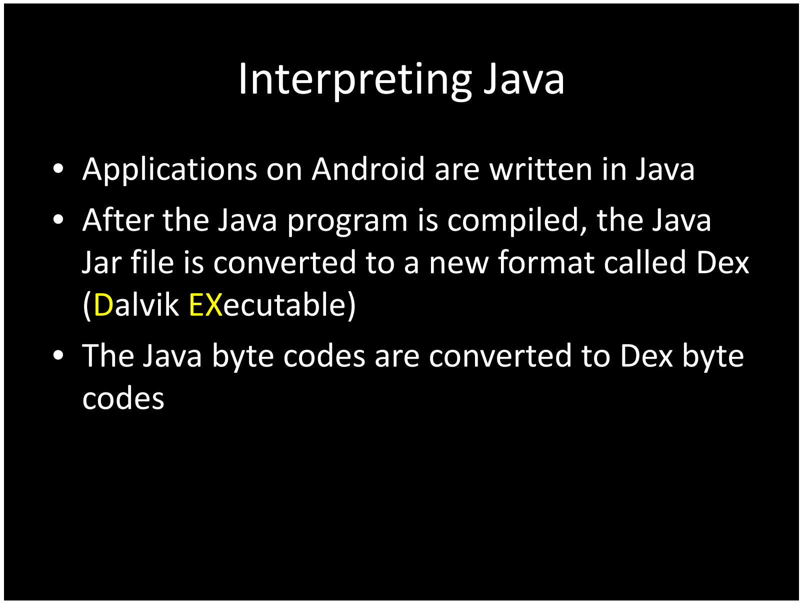
Limited Memory

- Total system RAM: 64 MB (minimum – it is possible for the machine to have more)
- Available RAM after low-level startup: 40 MB
- Available RAM after high-level services have started: 20 MB
- Large system library: 10 MB



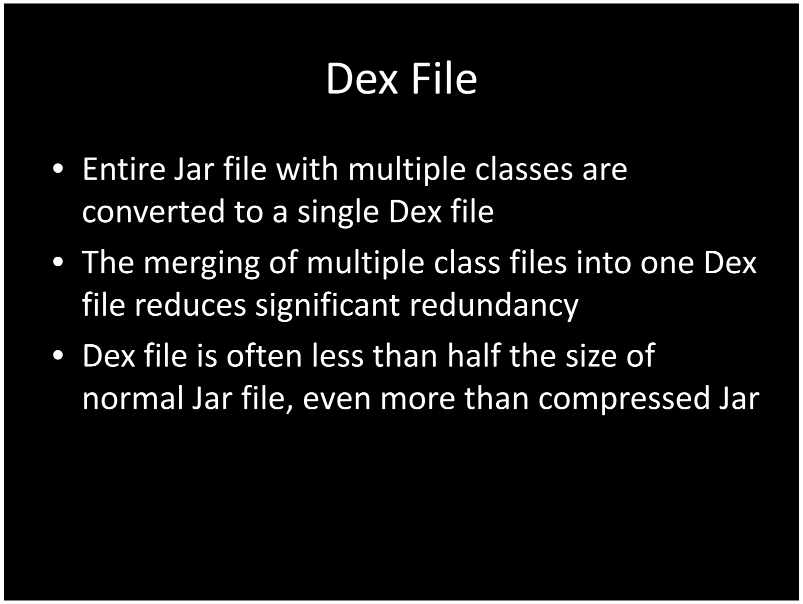
Pseudo Virtual Memory

- No swapping to disk
- Libraries can be mapped into a programs address space
- If the system runs out of memory, it will terminate the least recently used application *(low memory killer)*



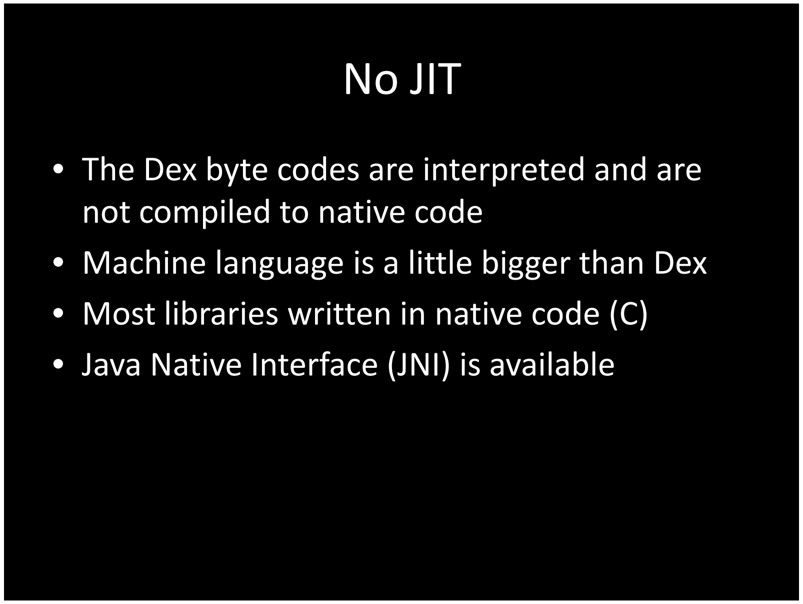
Interpreting Java

- Applications on Android are written in Java
- After the Java program is compiled, the Java Jar file is converted to a new format called Dex (Dalvik EXecutable)
- The Java byte codes are converted to Dex byte codes



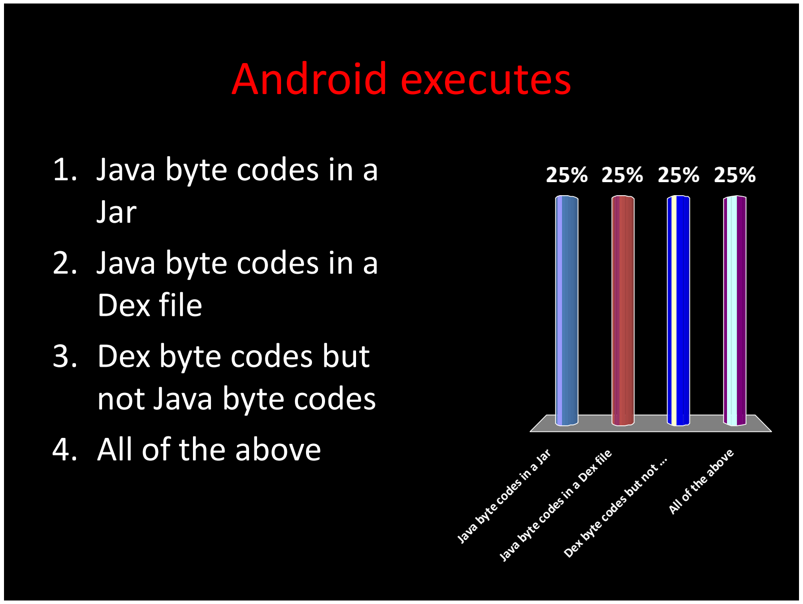
Dex File

- Entire Jar file with multiple classes are converted to a single Dex file
- The merging of multiple class files into one Dex file reduces significant redundancy
- Dex file is often less than half the size of normal Jar file, even more than compressed Jar



No JIT

- The Dex byte codes are interpreted and are not compiled to native code
- Machine language is a little bigger than Dex
- Most libraries written in native code (C)
- Java Native Interface (JNI) is available



Android executes

1. Java byte codes in a Jar
2. Java byte codes in a Dex file
3. Dex byte codes but not Java byte codes
4. All of the above

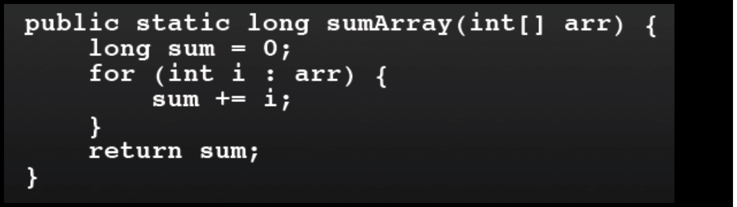
Dex Checks

- Byte codes are verified when loaded on device
- Optimizations are made to byte codes
  - Empty methods are removed
  - Static linking where possible
  - “inlining” special native methods

Register Machine Model

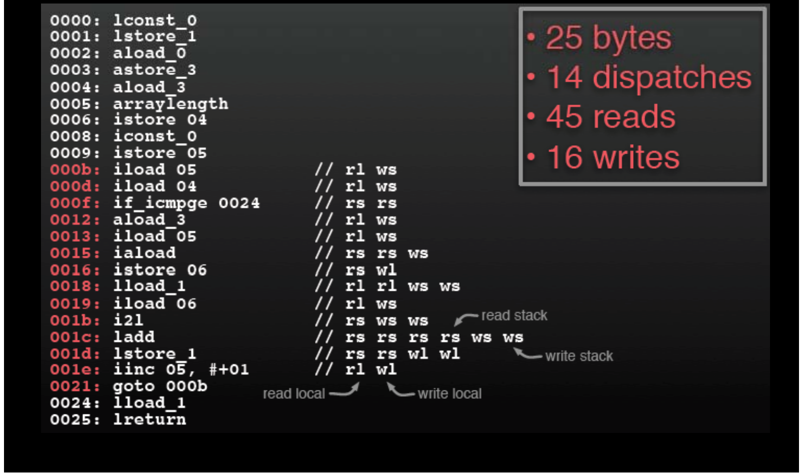
- The traditional JVM model is a stack machine
- Dex assumes the CPU has registers
- Higher semantic density per instruction
- Significantly fewer instructions are interpreted
  - about 30% fewer instructions
  - Important in an interpreter

Example Java Method



```
public static long sumArray(int[] arr) {
    long sum = 0;
    for (int i : arr) {
        sum += i;
    }
    return sum;
}
```

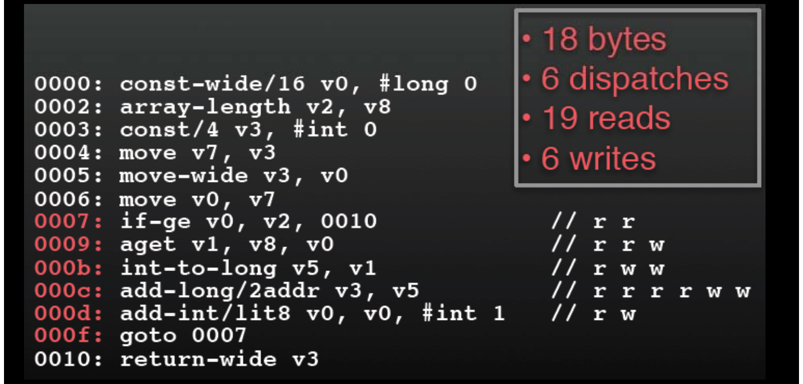
Traditional Java Byte Codes



```
0000: lconst_0
0001: lstore_1
0002: aload_0
0003: astore_3
0004: aload_3
0005: arraylength
0006: istore 04
0008: iconst_0
0009: istore 05
000b: iload 05               // rl ws
000d: iload 04               // rl ws
000f: if_icmpge 0024         // rs rs
0012: aload_3                // rl ws
0013: iload 05               // rl ws
0015: iaload                 // rs rs ws
0016: istore 06              // rs wl
0018: lload_1                // rl rl ws ws
0019: iload 06               // rl ws
001b: i2l                    // rs ws ws
001c: ladd                   // rs rs rs rs ws ws
001d: lstore_1               // rs rs wl wl
001e: iinc 05, #+01          // rl wl
0021: goto 000b
0024: lload_1
0025: lreturn
```

- 25 bytes
- 14 dispatches
- 45 reads
- 16 writes

Dex Byte Codes

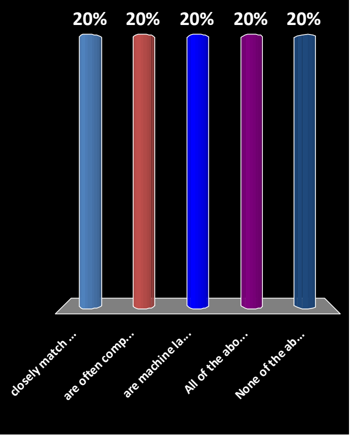


- 18 bytes
- 6 dispatches
- 19 reads
- 6 writes

```
0000: const-wide/16 v0, #long 0
0002: array-length v2, v8
0003: const/4 v3, #int 0
0004: move v7, v3
0005: move-wide v3, v0
0006: move v0, v7
0007: if-ge v0, v2, 0010           // r r
0009: aget v1, v8, v0              // r r w
000b: int-to-long v5, v1           // r w w
000c: add-long/2addr v3, v5        // r r r r w w
000d: add-int/lit8 v0, v0, #int 1  // r w
000f: goto 0007
0010: return-wide v3
```

Traditional Java byte codes

1. closely match common CPU architecture
2. are often compiled to machine language
3. are machine language for a Pentium
4. All of the above
5. None of the above



Memory Categories

- Clean or Dirty
  - Similar in concept to traditional virtual memory
  - Clean memory is mapped into the address space and has not been changed
- Shared or Private
  - shared: used by many processes
  - private: used by only one process

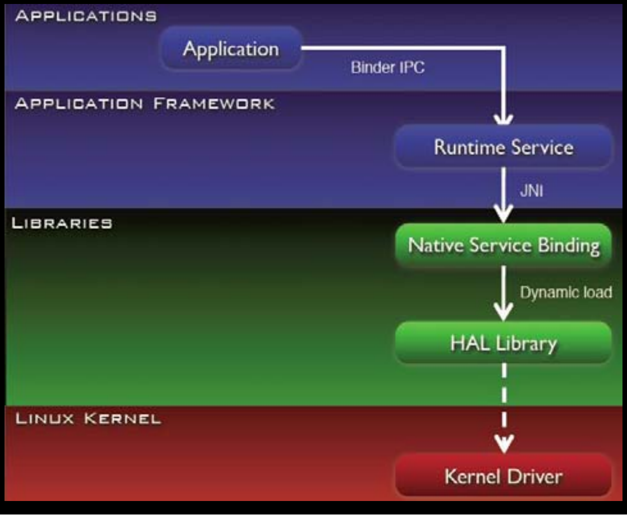
Memory Use

- Application Dex files are clean private
- Library Dex files are clean shared
- Application heap and local data are dirty private
- Dirty shared memory includes
  - library “live” Dex structures
  - shared copy-on-write heap (mostly not written)

Application Framework

- Provides APIs to a collection of services including:
  - Windows Manager
  - Location Manager (GPS)
  - Telephony Manager

Android Runtime Services

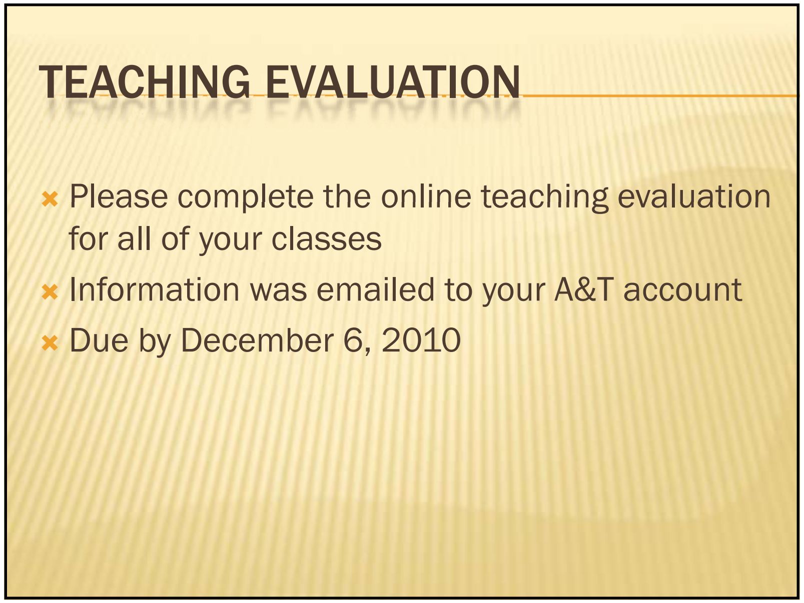


Developer Tools

- Most phone systems provide a set of developer tools for the desktop
- Android applications are written in Java using Eclipse and Apache Ant
- Iphone apps are written in Objective C
- Simulators allows you to run and debug applications on your desktop

Consistency

- Phone developers have complained that it is difficult to create an application that runs the same way on different phones
- The layout of objects on the screen has been a complaint. Screens can be of different sizes



TEACHING EVALUATION

- Please complete the online teaching evaluation for all of your classes
- Information was emailed to your A&T account
- Due by December 6, 2010