Mobile OS:Andriod and iOS Introduction and Architecture

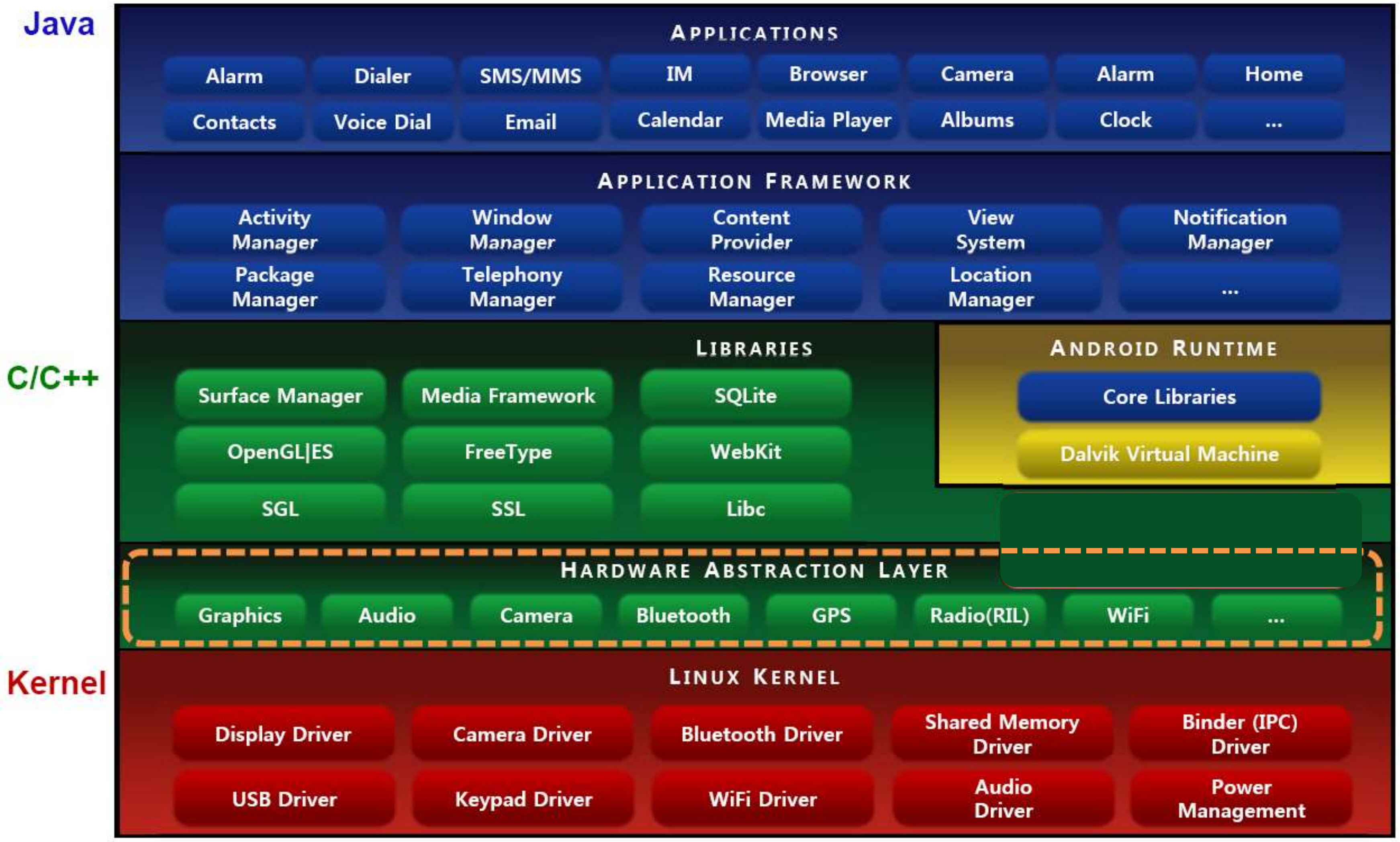
S.Lakshmi Priya
AP/CSE

Android

Android Introduction

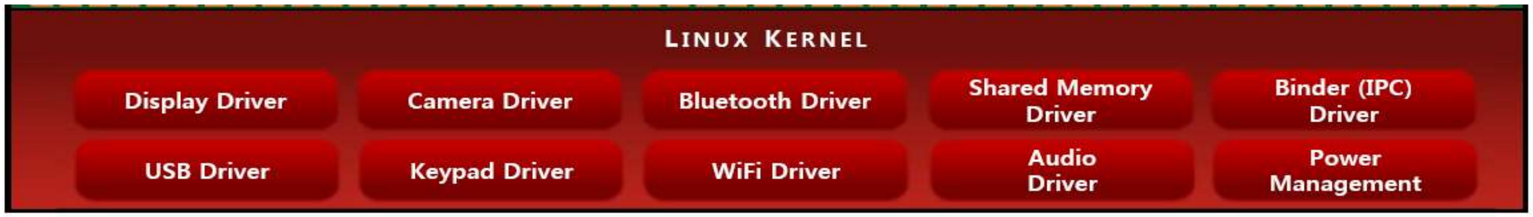
- Android is a Linux based operating system it is designed primarily for touch screen mobile devices such as smart phones and tablet computers.
- A robot with a human appearance
- Android Inc. was founded by:
  - Andy Rubin
  - Rich Miner
  - Chris White
  - Nick Sears
- They thought of Developing an advanced OS for camera but later switched to Mobile OS
- Google acquired Android Inc. in August 2005.

Android Architecture



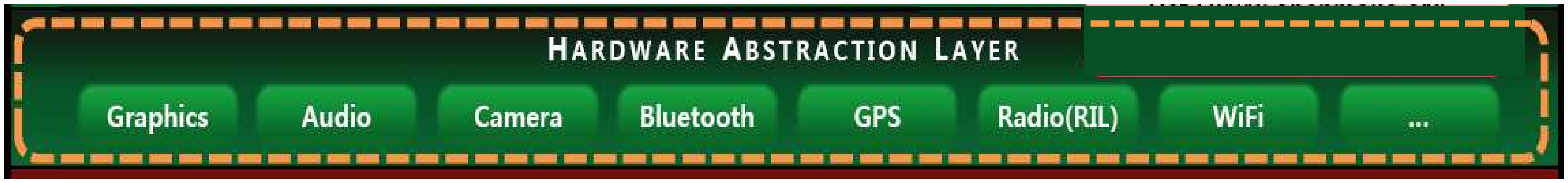
Linux Kernel

- The foundation of the Android platform is the Linux kernel.
  - For example, the Android Runtime (ART) relies on the Linux kernel for underlying functionalities such as threading and low-level memory management.
- Using a Linux kernel
  - allows Android to take advantage of key security features and allows device manufacturers to develop hardware drivers for a well-known kernel.
  - provides a level of abstraction between the device hardware and the upper layers of the Android software stack
- Linux version 2.6.x for core system services like memory management, process management, security model, networking and lot of core OS infrastructure



Hardware Abstraction Layer (HAL)

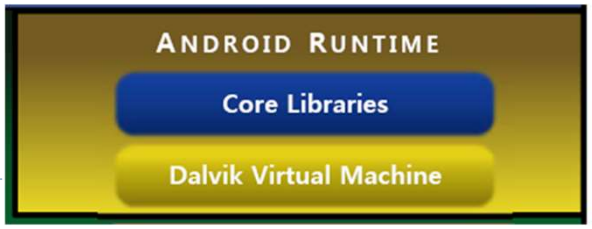
- The hardware abstraction layer (HAL) provides standard interfaces that expose device hardware capabilities to the higher-level Java API framework.
- The HAL consists of multiple library modules, each of which implements an interface for a specific type of hardware component, such as the camera or bluetooth module.
- When a framework API makes a call to access device hardware, the Android system loads the library module for that hardware component.



Android Runtime

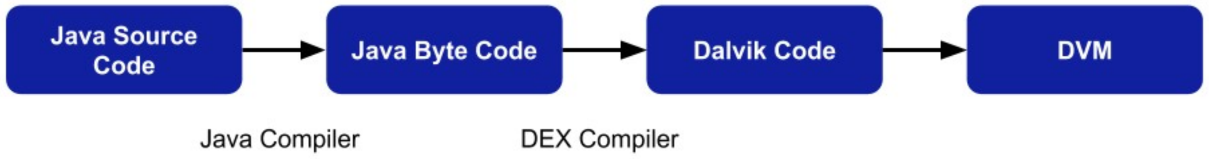
- Android runtime meet the needs of Android OS running in an embedded environment
  - i.e., where is limited battery, limited Memory and limited CPU.
- Android Runtime consists of Dalvik Virtual machine and Core Java libraries. (Before KitKat)
- Prior to Android version 5.0 (API level 21), Dalvik was the Android runtime. If your app runs well on ART, then it should work on Dalvik as well, but the_reverse may not be true.
- For devices running Android version 5.0 (API level 21) or higher, each app runs in its own process and with its own instance of the Android Runtime (ART).

- CORE LIBRARIES:
  - Java Programming Language contains all the collection classes, utilities, IO. All these utilities which you come across and expected to use.



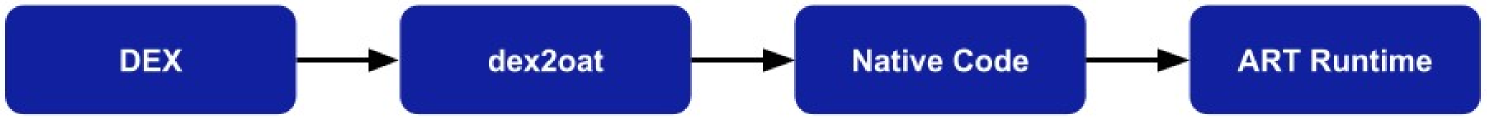
ART vs Dalvik Virtual Machine

- DALVIK VIRTUAL MACHINE
  - application code must be transformed from standard Java class files to the Dalvik executable (.dex) format, which has a 50% smaller memory footprint than standard Java bytecode
  - Just In Time compiler
  - Apps uses very less physical space on your device.
  - JIT results in more battery utilization.



ART vs Dalvik Virtual Machine

- ART or Android Runtime is an Android runtime that uses Ahead Of Time(AOT) compiler.
- By using AOT, what is does is it converts or compiles the whole High-level language code into Machine level code and at the time of installation of the app
- No lag when using the app
- More space occupied
- More battery utilization (poor battery performance)
- Better garbage collection



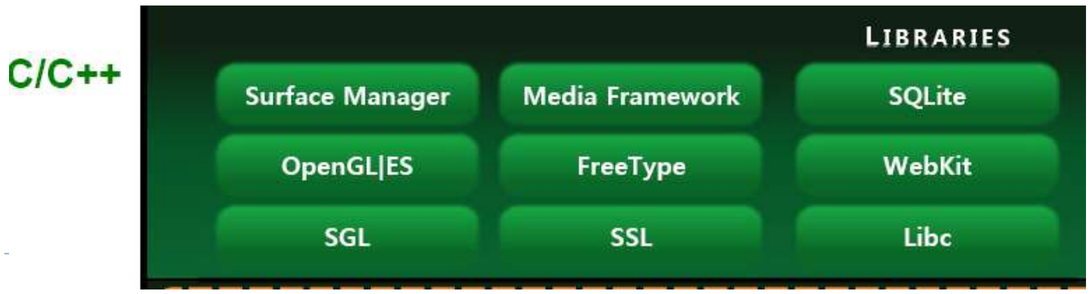
Android Runtime

- Applications are essentially sandboxed
- Enforced level of abstraction makes applications platform neutral in that they are never tied to any specific hardware.

- Major features of ART include:
  - Optimized garbage collection
  - AOT or JIT time compilation
  - Better debugging support

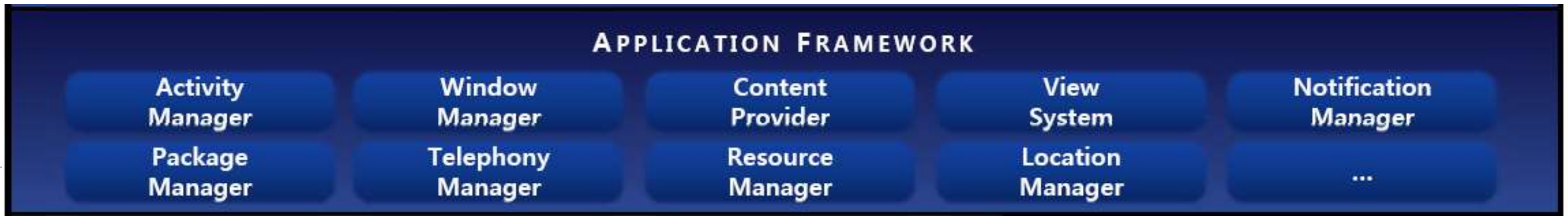
Native C/C++ Libraries

- Many core Android system components and services, such as ART and HAL, are built from native code that require native libraries written in C and C++.
- The Android platform provides Java framework APIs to expose the functionality of some of these native libraries to apps
  - For example, you can access OpenGL ES through the Android framework's Java OpenGL API to add support for drawing and manipulating 2D and 3D graphics in your app.
  - Media Framework : Core part of the android multimedia- MPEG4,MP3,…
  - FreeType: To render the fonts.
  - WebKit:Open source browser engine. Helps to work well on small screen.
  - SQLite: Embedded Database



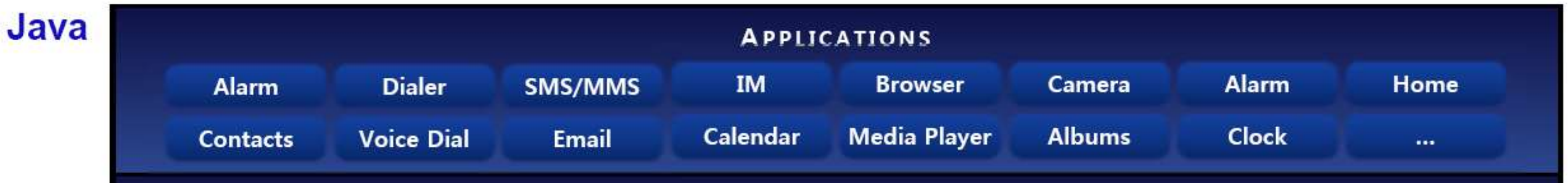
Java API Framework

- The entire feature-set of the Android OS is available to you through APIs written in the Java language.
  - building blocks to create Android apps by simplifying the reuse of core, modular system components and services, which include the following:
  - A rich and extensible View System you can use to build an app's UI, including lists, grids, text boxes, buttons, and even an embeddable web browser
  - A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files
  - A Notification Manager that enables all apps to display custom alerts in the status bar
  - An Activity Manager that manages the lifecycle of apps
  - Content Providers that enable apps to access data from other apps, such as the Contacts app, or to share their own data



System Apps

- Android comes with a set of core apps for email, SMS messaging, calendars, internet browsing, contacts, and more.
- Apps included with the platform have no special status among the apps the user chooses to install
- The system apps function both as apps for users and to provide key capabilities that developers can access from their own app.
  - For example, if your app would like to deliver an SMS message, you don't need to build that functionality yourself—you can instead invoke whichever SMS app is already installed to deliver a message to the recipient you specify.



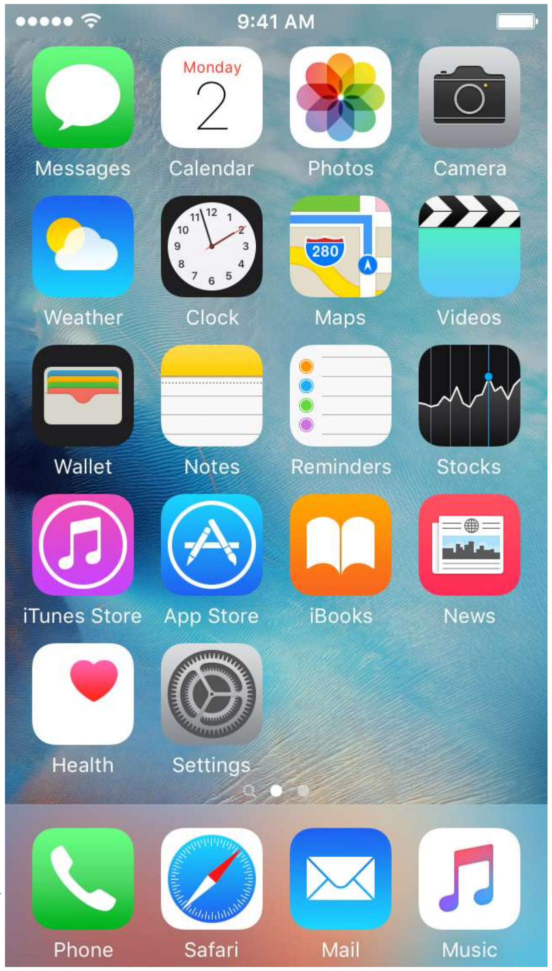
Features

- Beautiful UI
  - Android OS basic screen provides a beautiful and intuitive user interface.
- Connectivity
  - GSM/EDGE, Bluetooth, Wi-Fi, NFC.
- Storage SQLite
  - A lightweight relational database, is used for data storage purposes.
- Media support
  - MPEG-4, MP3, MIDI, WAV, JPEG, PNG, GIF, and BMP
- Messaging
  - SMS and MMS
- Also, Multitouch, Multitasking, Resizable widgets,

iOS

Introduction

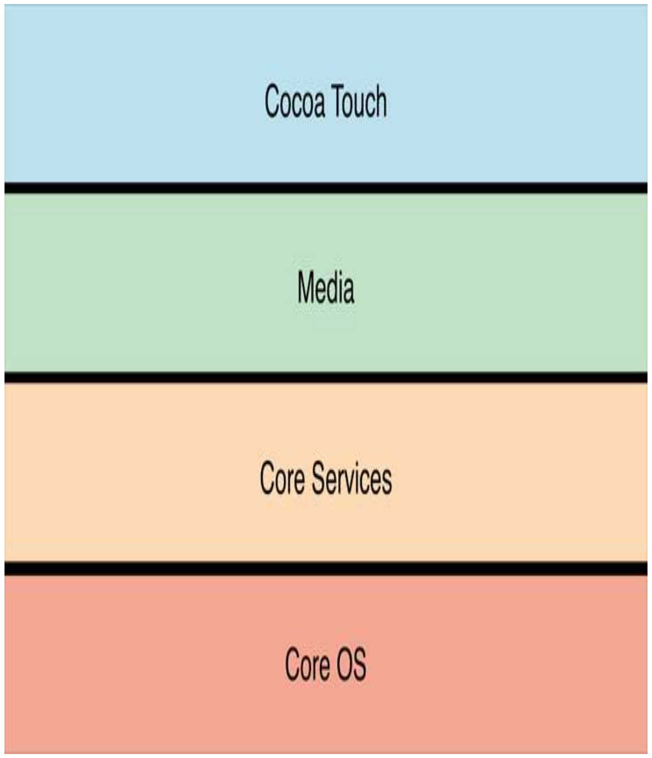
- **iOS** (formerly iPhone OS) is a mobile operating system created and developed by Apple Inc. exclusively for its hardware.
- Originally unveiled in 2007 for the iPhone, it has been extended to support other Apple devices such as the iPod Touch (September 2007), iPad (January 2010), iPad Mini (November 2012) and second-generation Apple TV (September 2010).
- Unlike Microsoft's Windows Phone and Google's Android, Apple does not license iOS for installation on non-Apple hardware.
- Based on Unix-like, BSD OS
- Written in C, C++, Objective C, Swift



iOS Architecture

- ***Core OS Layer***
  - The Core OS Layer occupies the bottom position of the iOS stack and, as such, sits directly on top of the device hardware.
  - The layer provides a variety of services including low level networking, access to external accessories and the usual fundamental operating system services such as memory management

- ***Core Services Layer***
  - Core Services layers contain the fundamental interfaces for iOS, including those used for accessing files, low-level data types, network sockets, and so on.
  - These interfaces are mostly C-based .



iOS Architecture (Contd.,)

- ***Media Layer***
  - Media Layer allows you to create the best multimedia experience available on a mobile device with the frameworks
  - More advanced technologies that use interfaces based on a mixture of C and Objective-C is used in Media Layer.
  - The Graphics libraries live here
    - Core Graphics (Quartz), OpenGL, Metal(3D API), Photos Library, Animation
  - Audio
    - Media player, OpenAL, Core Audio
  - AirPlay

iOS Architecture (Contd.,)

- ***Cocoa Touch Layer***
  - API(Application Programming Interface) called Cocoa Touch which includes gesture recognition, animation, and a different user interface library, and is for iOS apps.
  - Cocoa Touch is primarily written in Objective-C
  - Drives the UI
    - Provides the Controllers, Widgets, etc.
  - Provides access to main system functions
    - Contacts, Camera, touch input, share with other apps, push notifications, etc.

Features – Pros and Cons

- Security
  - Apple promises that all passwords will be encrypted.
  - iOS keeps track of app permissions at the root level. This means that you can adjust your privacy settings for each app individually.
  - You also have the choice to activate the Limit Ad Tracking feature, which will block marketers from planting cookies on your phone.
  - Apple has introduced a password-protect reset feature, which ensures that stolen Apple devices can't be used without the owner's Apple ID and password.
- Customizability
  - iOS 7 does not allow for a lot of customization.
  - Like Android's live wallpapers, Apple gives you the choice to use dynamic wallpapers on your home and lock screens.
  - However, there is not a whole lot of choice yet.

Thank You