Mobile Application Development

Produced by

David Drohan (ddrohan@wit.ie)

Department of Computing & Mathematics
Waterford Institute of Technology
http://www.wit.ie

Android Anatomy

The Anatomy of an Android App

Agenda

- ❑ In a Nutshell…
- ❑ Platform Architecture
- ❑ Application Components
- ❑ The Manifest File
- ❑ Application Resources
- ❑ The Android Application Life Cycle

Agenda

- ❑ In a Nutshell…
- ❑ Platform Architecture
- ❑ Application Components
- ❑ The Manifest File
- ❑ Application Resources
- ❑ The Android Application Life Cycle

Android Anatomy

In a Nutshell

In a Nutshell

- ❑Android applications are created by bringing together one or more components known as Activities.

- ❑An activity is a single, standalone module of application functionality that usually correlates directly to a single user interface screen and its corresponding functionality.

- ❑An activity typically represents a single user interface screen within an app.

- ❑One option is to construct the activity using a single user interface layout and one corresponding activity class file.

In a Nutshell

- ❑Another approach is to break the activity into different sections.
- ❑Each of these sections is referred to as a fragment, each of which consists of part of the user interface layout and a matching class file (declared as a subclass of the Android Fragment class).
- ❑In this scenario, an activity simply becomes a container into which one or more fragments are embedded.
- ❑(more on this later)

In a Nutshell

- ❑ Intents are the mechanism by which one activity is able to launch another and implement the flow through the activities that make up an application. Intents consist of a description of the operation to be performed and, optionally, the data on which it is to be performed.

- ❑ Another type of Intent, the Broadcast Intent, is a system wide intent that is sent out to all applications that have registered an “interested” Broadcast Receiver. The Android system, for example, will typically send out Broadcast Intents to indicate changes in device status such as the completion of system start up, connection of an external power source to the device or the screen being turned on or off.

- ❑ Broadcast Receivers are the mechanism by which applications are able to respond to Broadcast Intents

In a Nutshell

- ❑Android Services are processes that run in the background and do not have a user interface. They can be started and subsequently managed from Activities, Broadcast Receivers or other Services

- ❑An example service would be a stock market tracking application that notifies the user when a share hits a specified price.

- ❑Content Providers implement a mechanism for the sharing of data between applications. Any application can provide other applications with access to its underlying data through the implementation of a Content Provider including the ability to add, remove and query the data (subject to permissions)

In a Nutshell

- The glue that pulls together the various elements that comprise an application is the Application Manifest file. It is within this XML based file that the application outlines the activities, services, broadcast receivers, data providers and permissions that make up the complete application

- An Android application package will also typically contain a collection of resource files. These files contain resources such as the **strings, images, fonts and colours** that appear in the user interface together with the XML representation of the user interface layouts

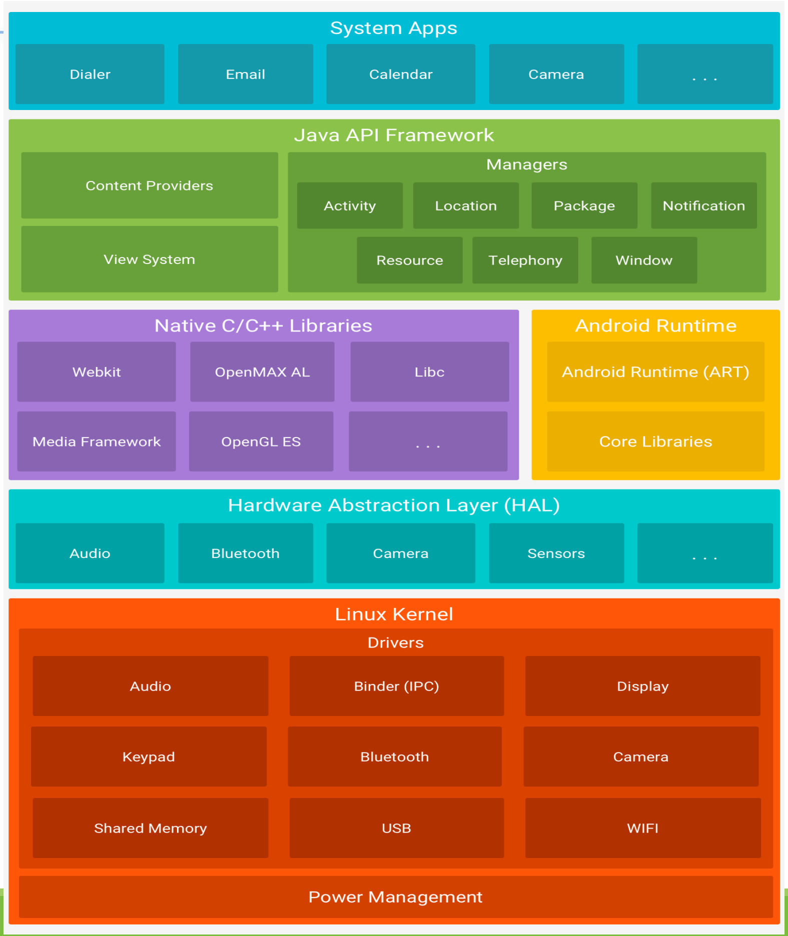
- When an application is compiled, a class named R is created that contains references to the application resources. The application manifest file and these resources combine to create what is known as the Application Context.

Android Anatomy

Platform Architecture

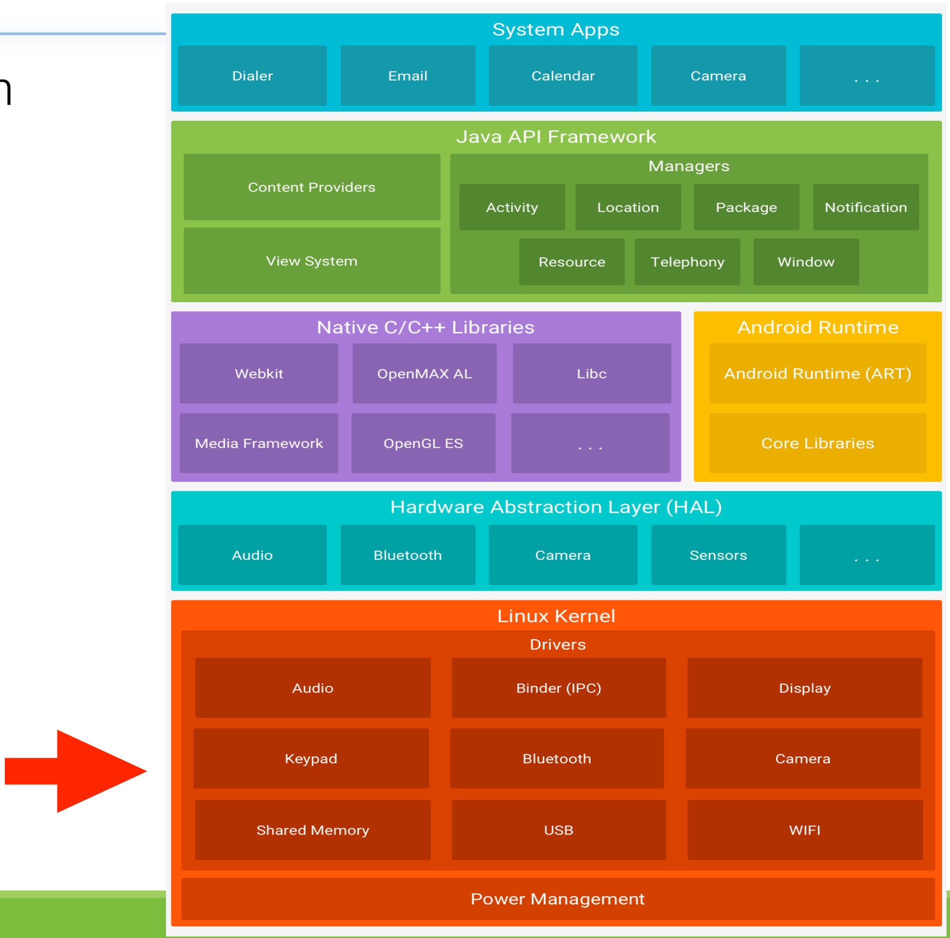
Platform Architecture

- Android is an open source, Linux-based software stack created for a wide array of devices and form factors.
- Its major components are:
  - The Linux Kernel
  - Hardware Abstraction Layer
  - Android Runtime
  - Native C/C++ Libraries
  - Java API Framework
  - System Apps



Platform Architecture

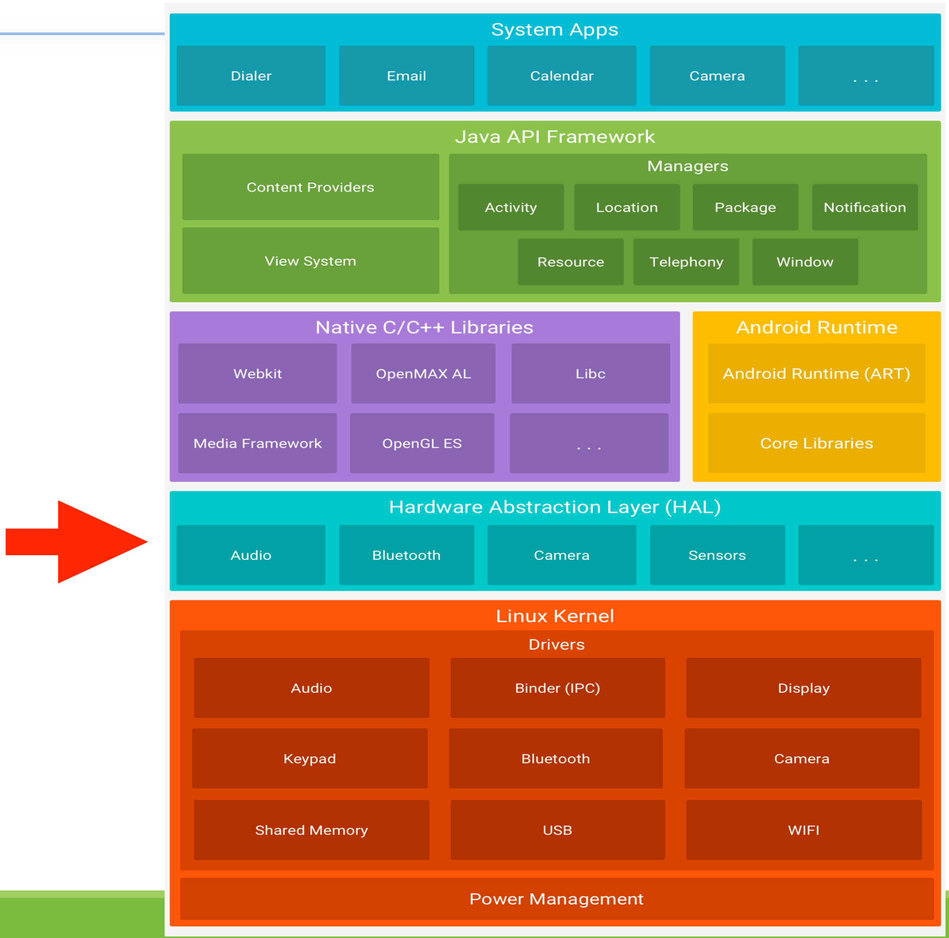
- ❑ The foundation of the Android platform is the **Linux kernel**. For example, the Android Runtime (ART) relies on the Linux kernel for underlying functionalities such as threading and low-level memory management.
- ❑ Using a Linux kernel allows Android to take advantage of key security features and allows device manufacturers to develop hardware drivers for a well-known kernel.



Platform Architecture

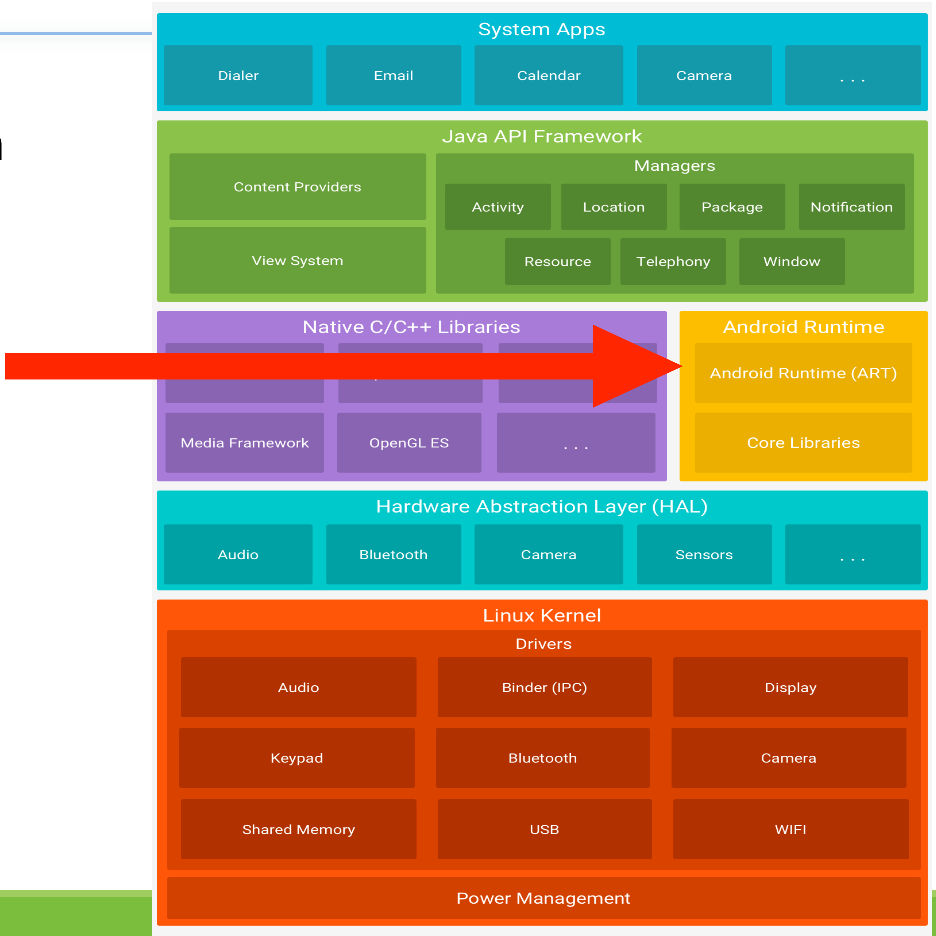
- ❑The **Hardware Abstraction Layer** (HAL) provides standard interfaces that expose device hardware capabilities to the higher-level Java API framework.

- ❑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.



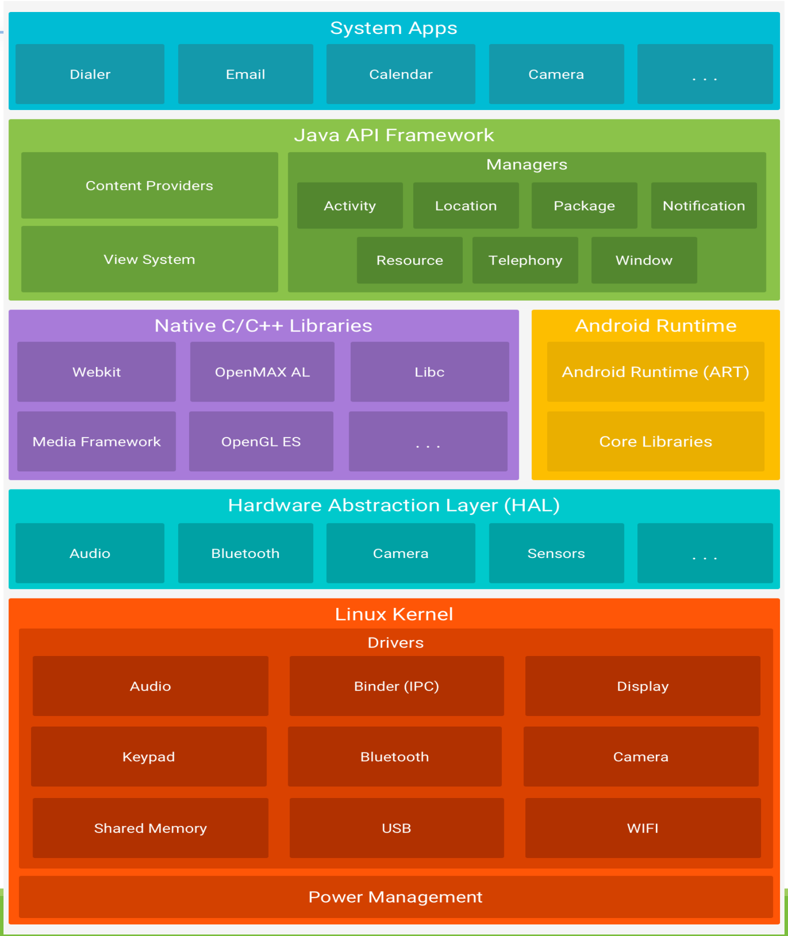
Platform Architecture

- ❑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).
- ❑ART is written to run multiple virtual machines on low-memory devices by executing DEX files, a bytecode format designed specially for Android that's optimised for minimal memory footprint.
- ❑Build toolchains, such as Jack, compile Java sources into DEX bytecode, which can run on the Android platform.



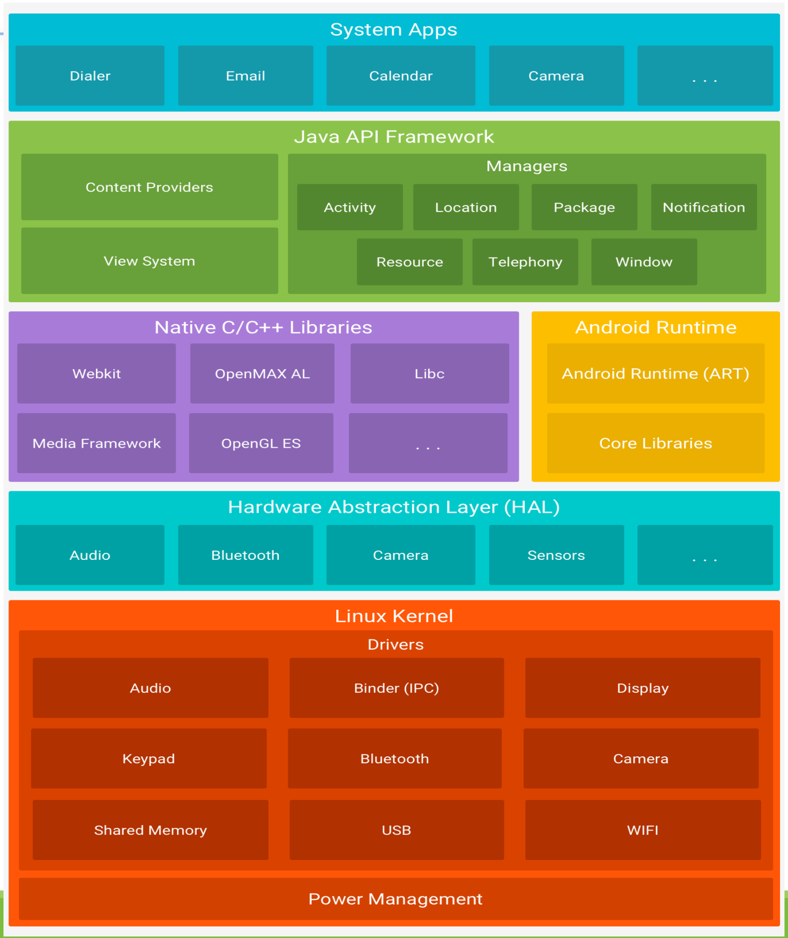
Platform Architecture

- ❑The entire feature-set of the Android OS is available to you through APIs written in the Java language.
- ❑Kotlin uses the Java API
- ❑These APIs form the building blocks you need to create Android apps by simplifying the reuse of core, modular system components and services
- ❑The rich and extensible **View System** allows you to build an app's UI, including lists, grids, text boxes, buttons, and even an embeddable web browser



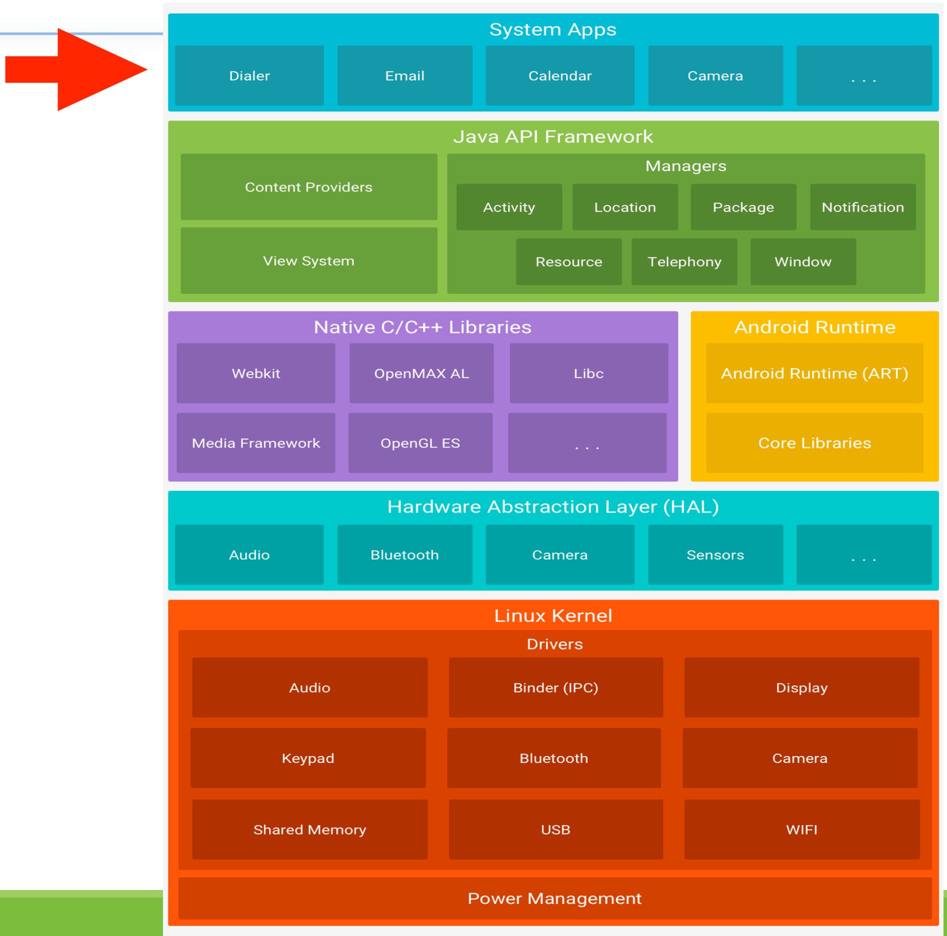
Platform Architecture

- ❑The **Resource Manager** provides access to non-code resources such as localised strings, graphics, and layout files
- ❑The **Notification Manager** enables all apps to display custom alerts in the status bar
- ❑The **Activity Manager** manages the lifecycle of apps and provides a common navigation back stack
- ❑**Content Providers** enable apps to access data from other apps, such as the Contacts app, or to share their own data



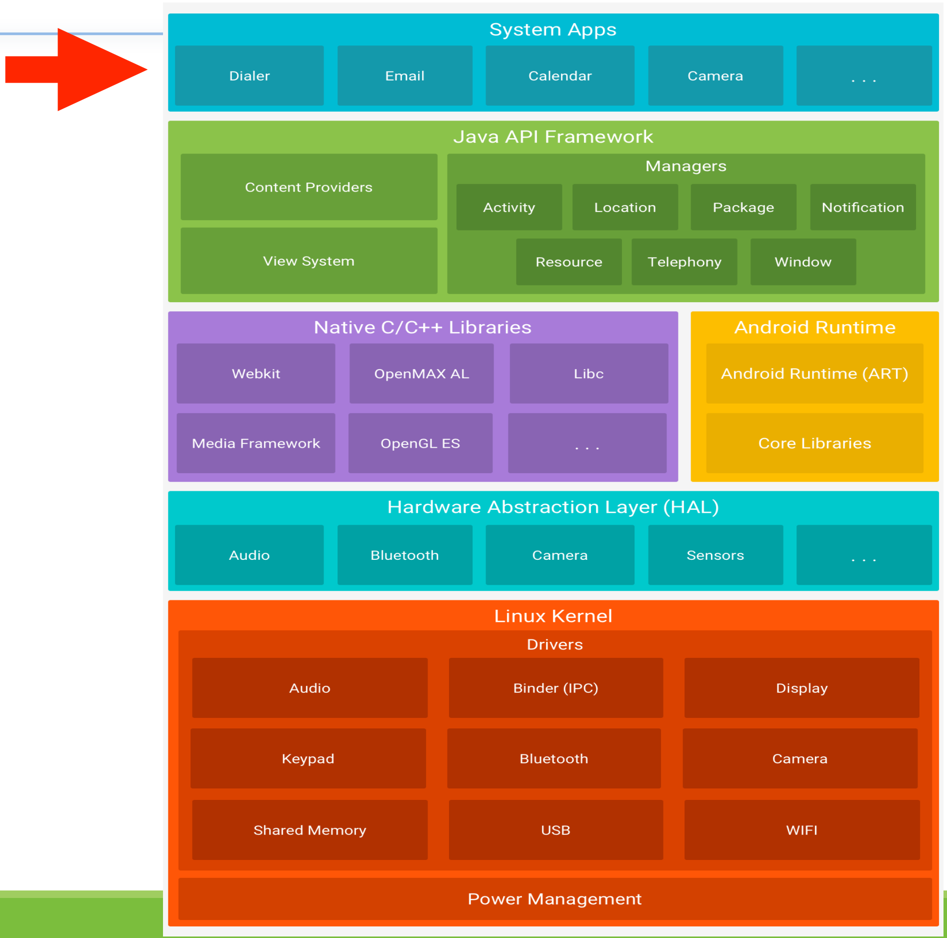
Platform Architecture

- ❑ Android comes with a set of core **System 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. So a third-party app can become the user's default web browser, SMS messenger, or even the default keyboard (some exceptions apply, such as the system's Settings app).



Platform Architecture

- ❑ 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.



Android Anatomy

Sources:

- https://developer.android.com/guide/platform/
- https://developer.android.com/guide/components/fundamentals
- https://www.techotopia.com/index.php/The_Anatomy_of_an_Android_Application

Thanks.