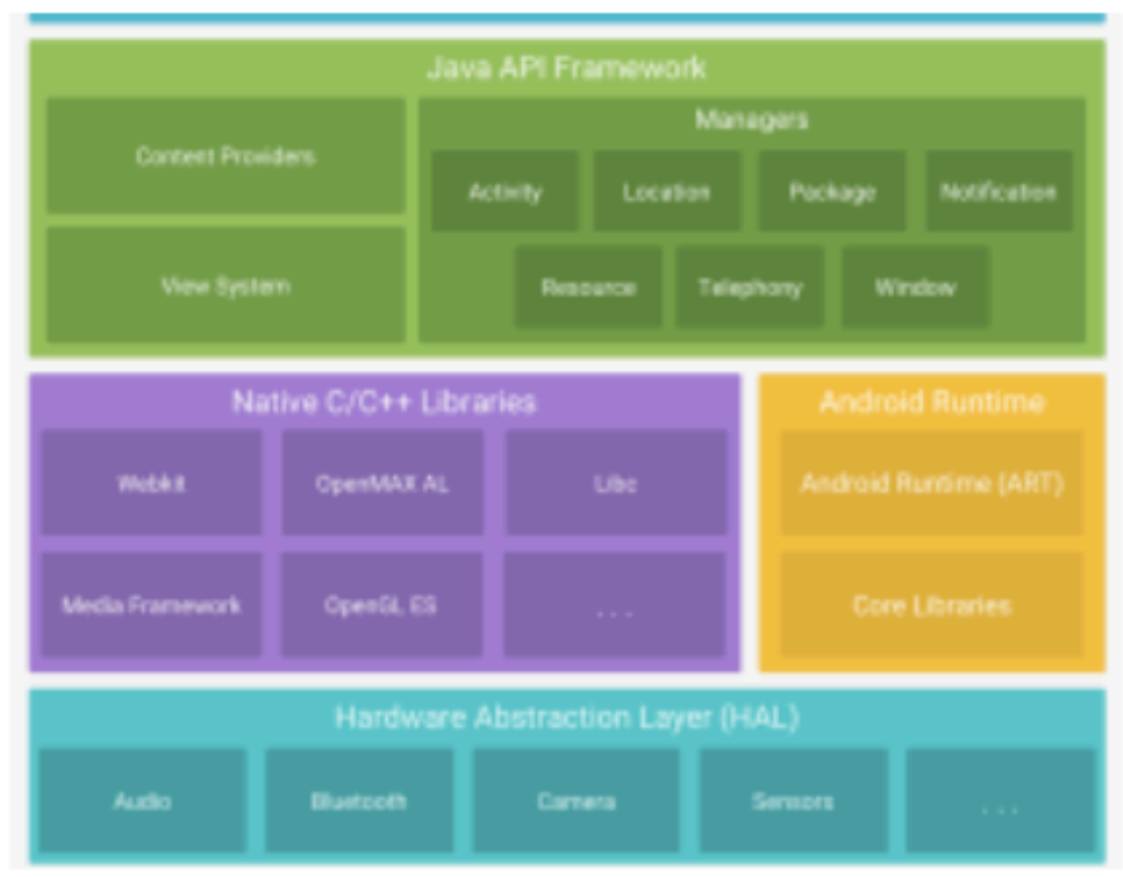
Android Platform, Components & Activities

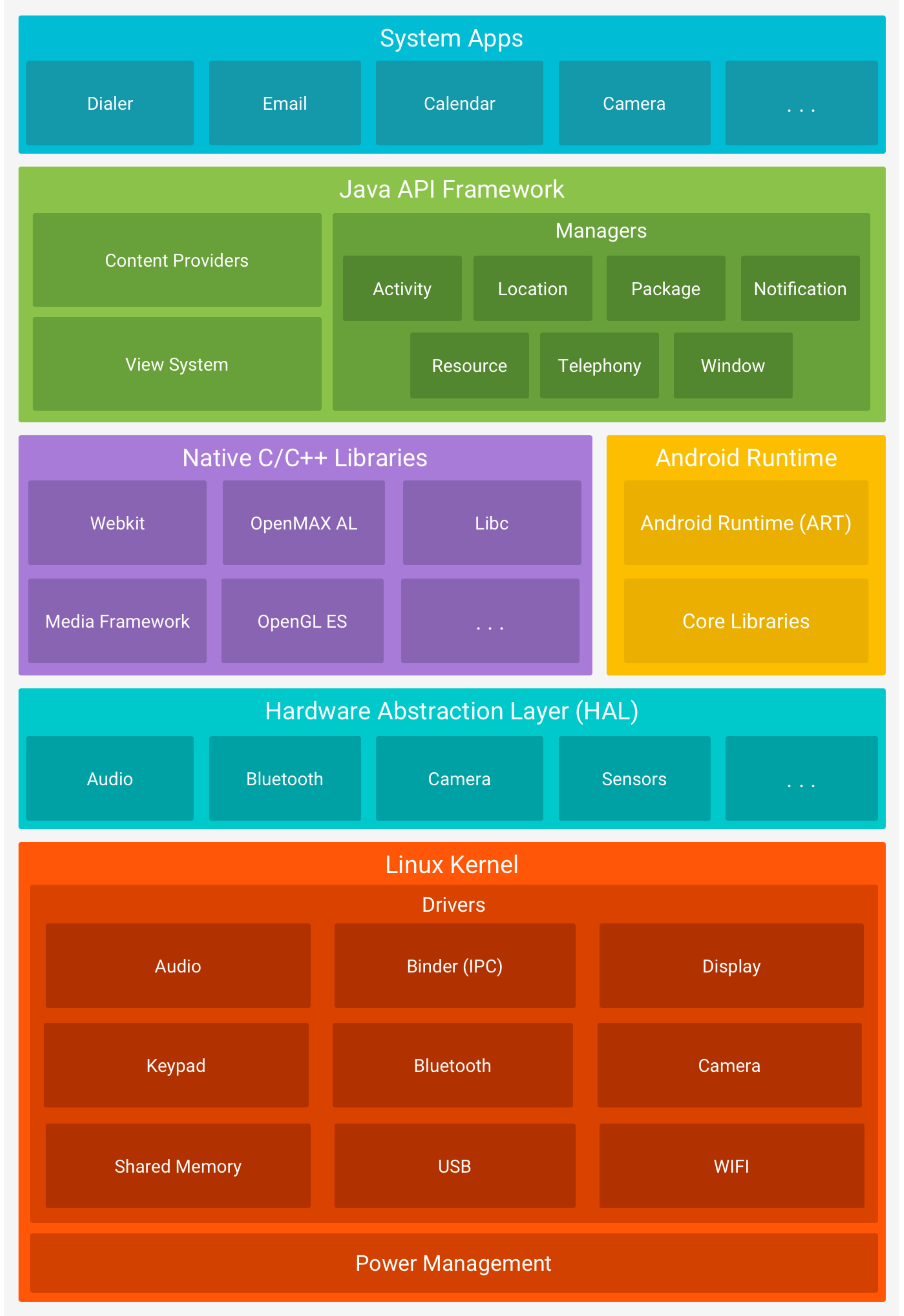
Android Platform, Components & Activities



Key features of the platform, the principle components + the activity model

Platform Fundamentals

Android is an open source, Linux-based software stack created for a wide array of devices and form factors.

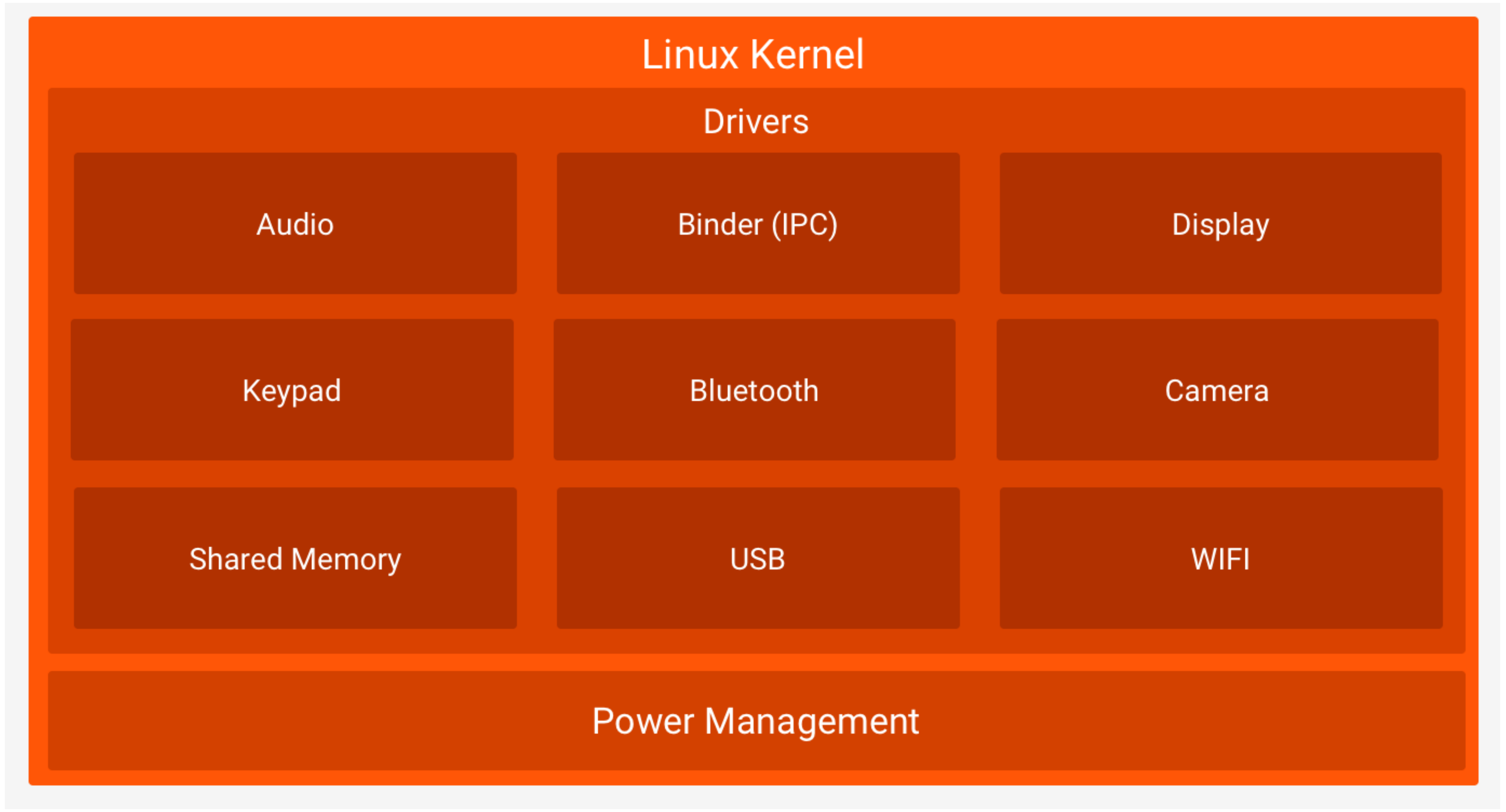


The Linux Kernel

The foundation of the Android platform

Allows Android to take advantage of key security features.

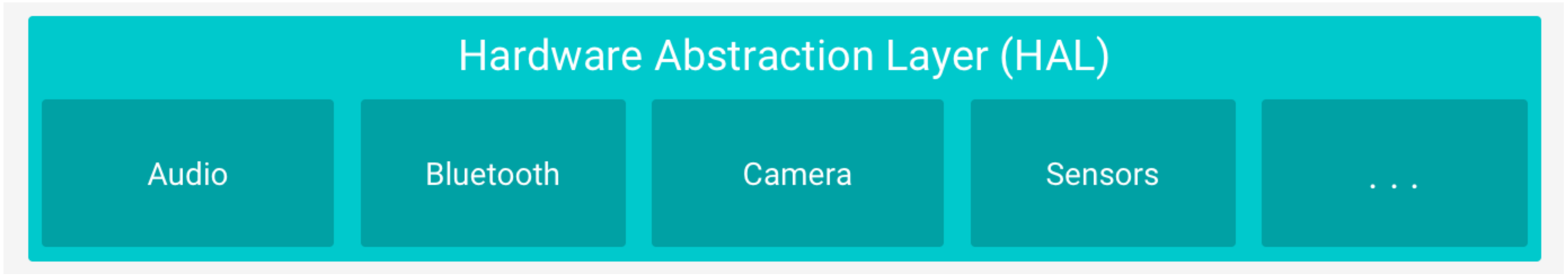
Allows device manufacturers to develop hardware drivers for a well-known kerne



Hardware Abstraction Layer (HAL)

Provides standard interfaces that expose device hardware capabilities to the higher-level Java API framework.

Consists of multiple library modules, each of which implements an interface for a specific type of hardware component.



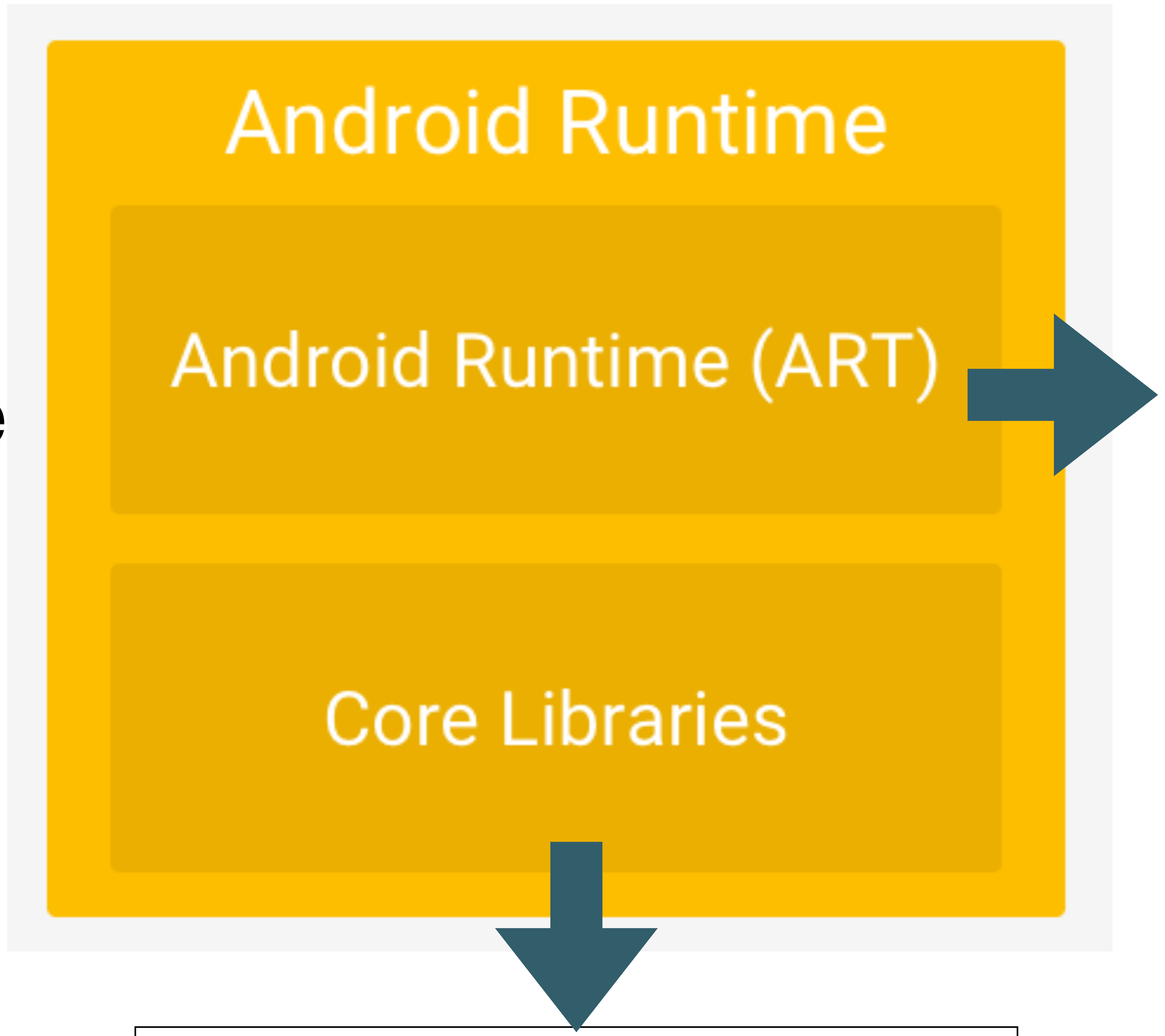
When a framework API makes a call to access device hardware, the Android system loads the library module for that hardware component.

Android Runtime

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,

Dex is a bytecode format designed specially for Android that's optimized for minimal memory footprint



Core runtime libraries that provide most of the functionality of the Java programming language

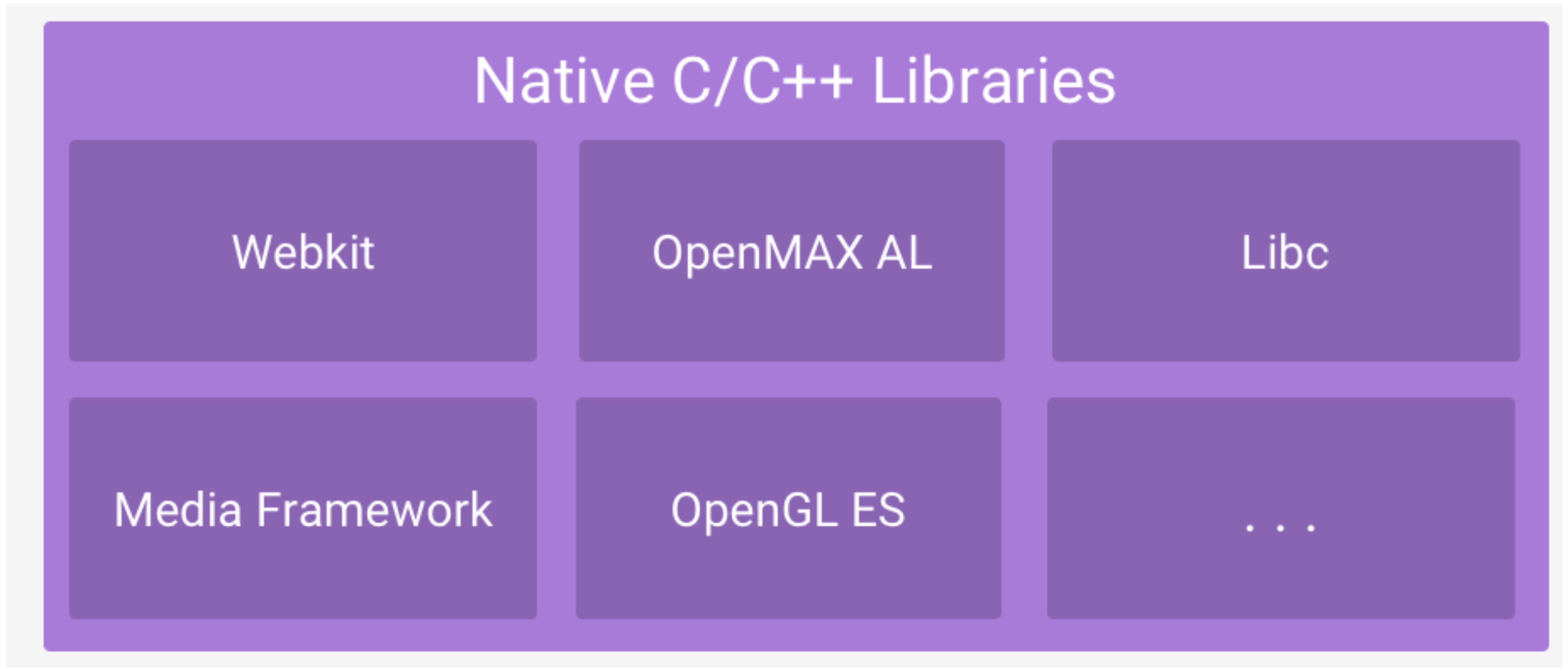
ART Features

- Ahead-of-time (AOT) and just-in-time (JIT) compilation
- Optimized garbage collection (GC)
- Conversion of an app package's Dalvik Executable format (DEX) files to more compact machine code (API 28)
- Debugging support, including a dedicated sampling profiler, detailed diagnostic exceptions and crash reporting

Native C/C++ Libraries

Many core components and services, (ART & HAL), are built from native code that require native libraries written in C and C++.

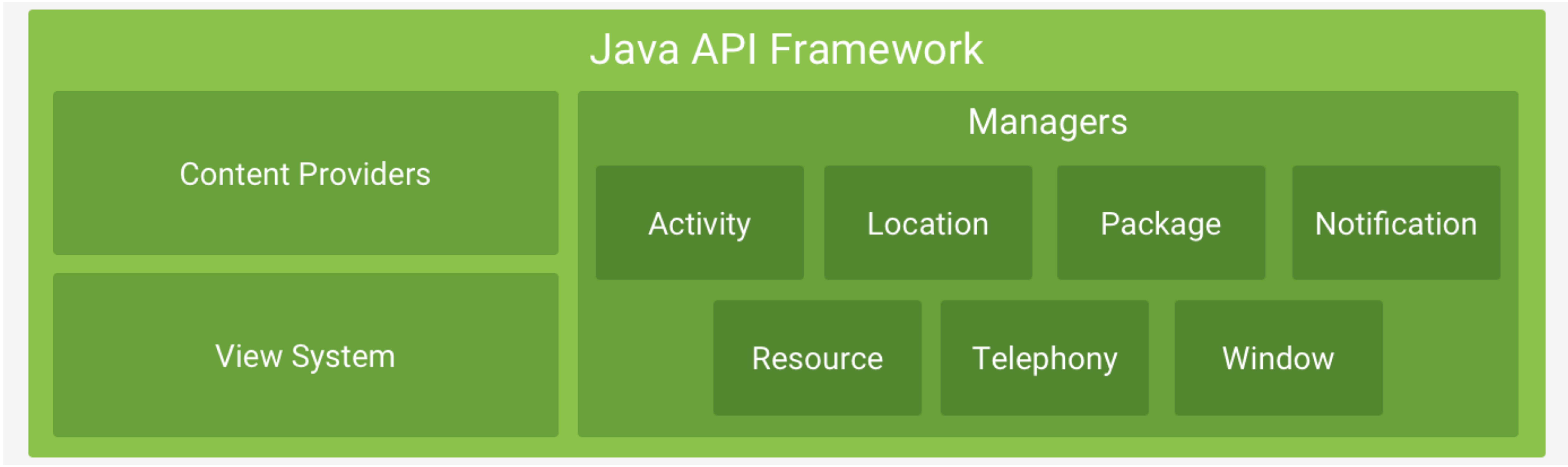
The Android platform provides Java framework APIs to expose the functionality of some of these native libraries to apps.



E.G:

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.

Java API Framework

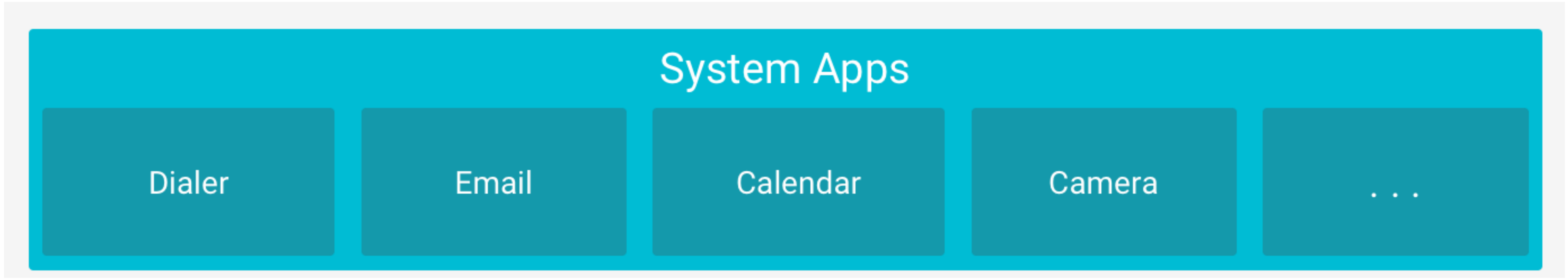


The entire feature-set of the Android OS is available through APIs written in the Java.

These APIs form the building blocks for Android apps, simplifying the reuse of core, modular system components and services

- **View System:** to build an app's UI, including lists, grids, text boxes, buttons, and even an embeddable web browser
- **Resource Manager:** providing access to non-code resources such as localized strings, graphics, and layout files
- **Notification Manager** : enables all apps to display custom alerts in the status bar
- **Activity Manager:** manages the lifecycle of apps and provides a common navigation back stack
- **Content Providers:** enable apps to access data from other apps (eg)Contacts app) or to share their own data

System Apps



A set of core apps for email, SMS messaging, calendars, internet browsing, contacts, etc..

Apps included with the platform have no special status among the apps the user chooses to install.

These are app for users + they provide capabilities that developers can access.

→

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).

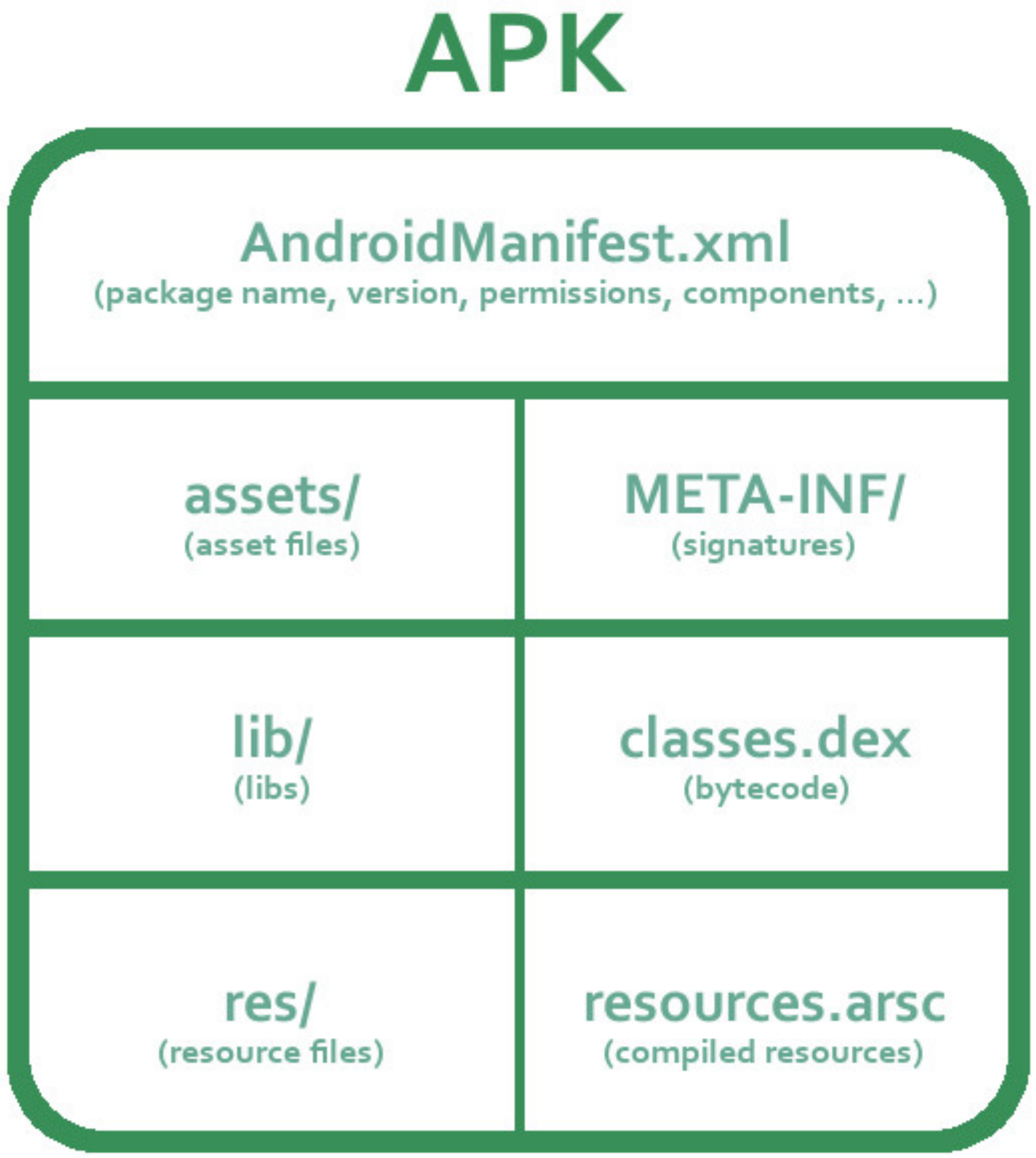
Application Component Fundamentals

Android Application

Android apps can be written using Kotlin, Java, and C++ languages.

The Android SDK tools compile code along with any data and resource files into an APK, an Android package, which is an archive file with an .apk suffix.

One APK file contains all the contents of an Android app and is the file that Android-powered devices use to install the app.

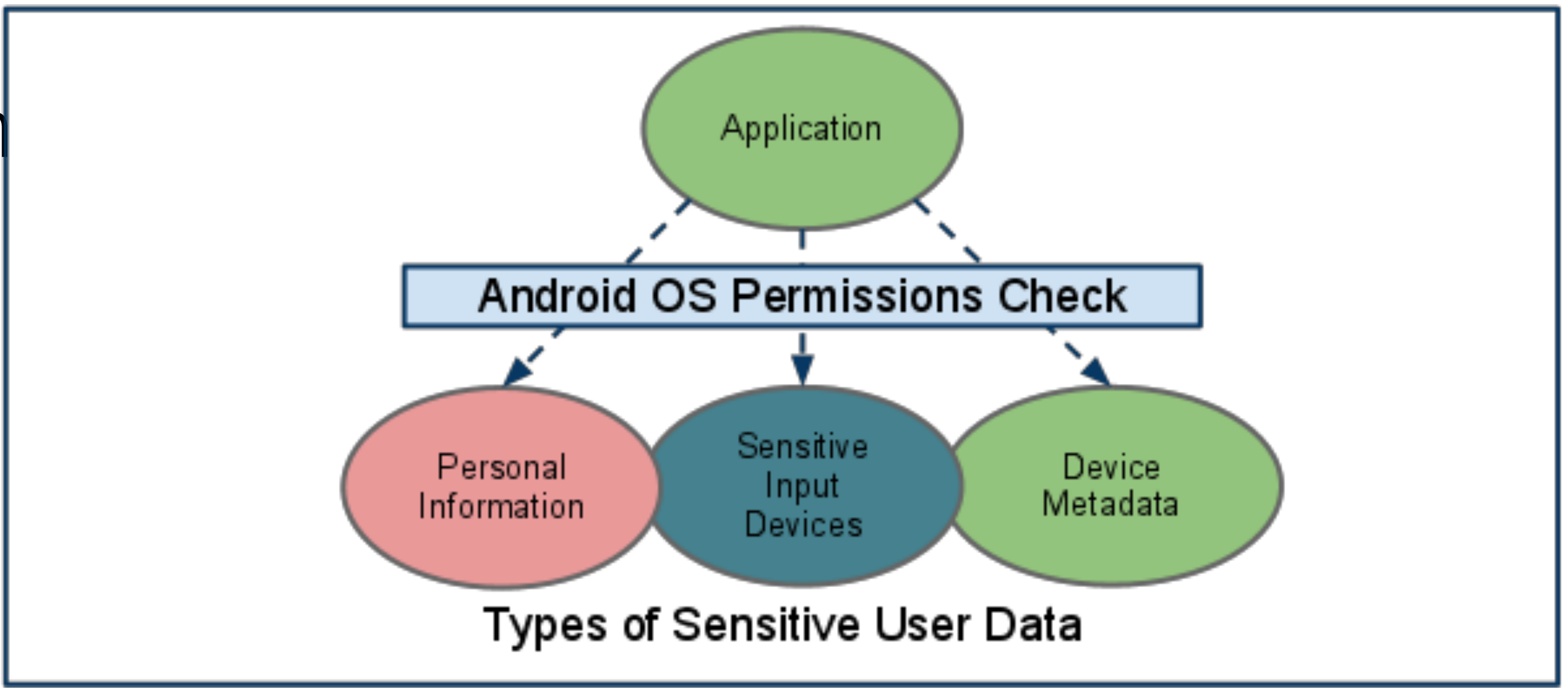


Security Model

Each Android app lives in its own security sandbox, protected a range of security features. Specifically, each app:

- Is assigned a unique Linux user ID. The system sets permissions for all the files in an app so that only the user ID assigned to that app can access them.
- Is a process with its own virtual machine (VM), so an app's code runs in isolation from other apps.

The Android system starts the process when any of the app's components need to be executed, and then shuts down the process when it's no longer needed or when the system must recover memory for other apps.



Principle of Least Privilege.

Each app, by default, has access only to the components that it requires to do its work and no more.

This creates a secure environment in which an app cannot access parts of the system for which it is not given permission.

However, there are ways for an app to share data with other apps and for an app to access system services:

- An app can request permission to access device data such as the user's contacts, SMS messages, the mountable storage (SD card), camera, and Bluetooth.
- The user has to explicitly grant these permissions.

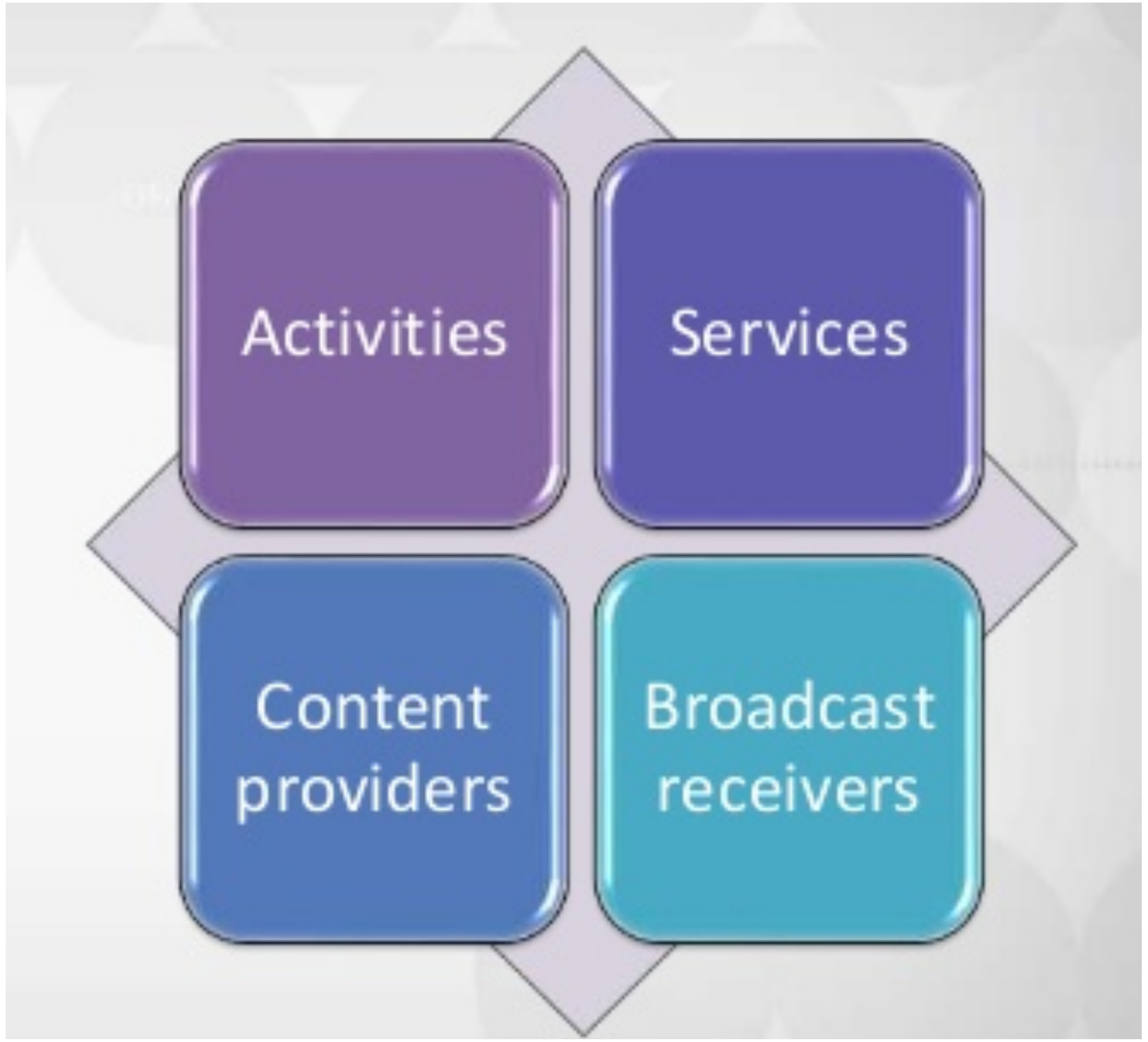
Application Components

App components are the essential building blocks of an Android app.

Each component is an entry point through which the system or a user can enter the app

Each type serves a distinct purpose and has a distinct lifecycle that defines how the component is created and destroyed

4 Types



Android Components

Application components are the essential building blocks of an Android application.

Activities

An activity represents a single screen with a user interface, in-short Activity performs actions on the screen.

Services

A service is a component that runs in the background to perform long-running operations.

Broadcast Receivers

Broadcast Receivers simply respond to broadcast messages from other applications or from the system.

Content Providers

A content provider component supplies data from one application to others on request.

Additional Components

There are additional components which will be used in the construction of above mentioned entities

http://yellowsblog.com

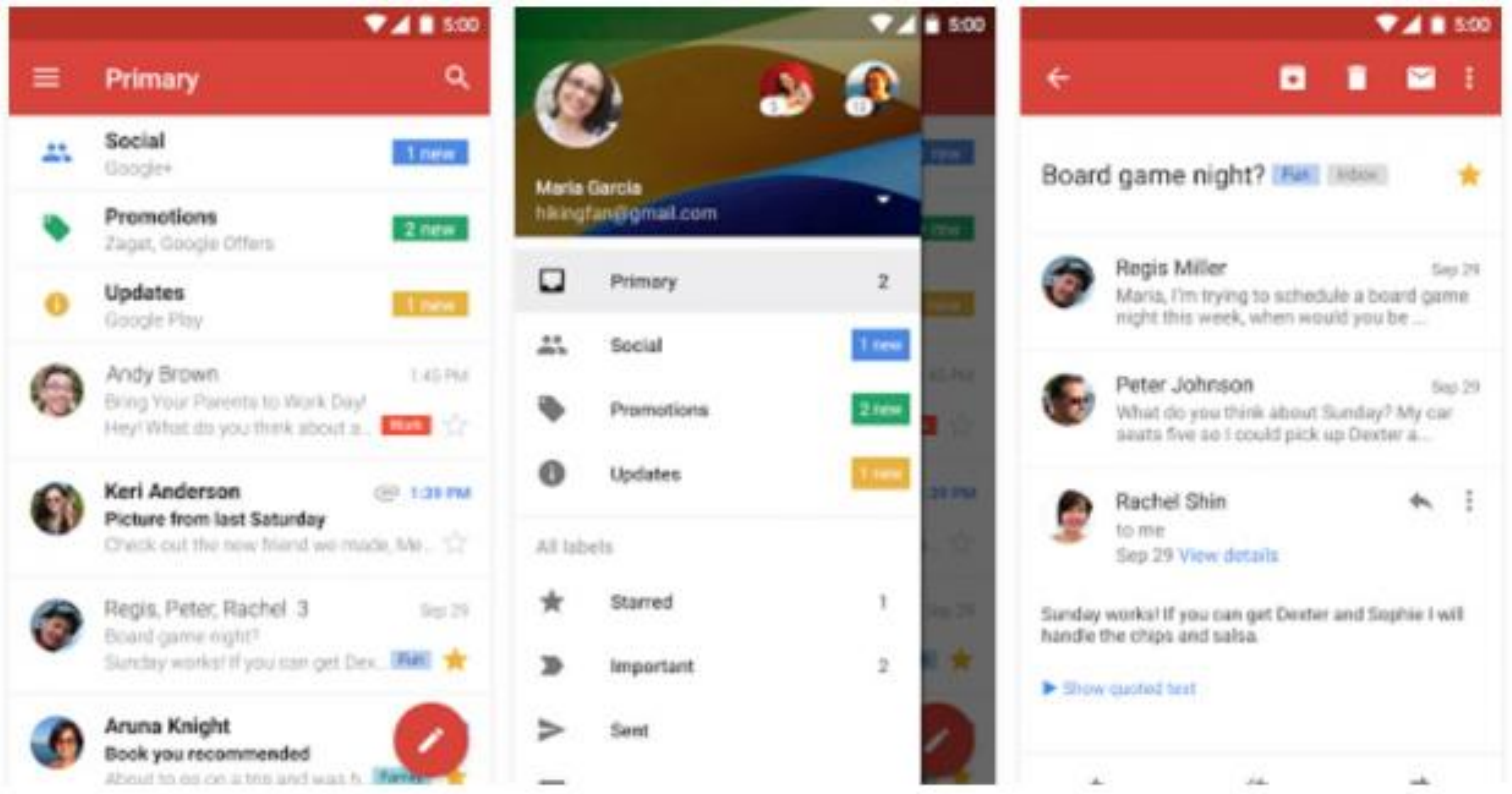
Activities I

An activity is the entry point for interacting with the user.

It represents a single screen with a user interface.

Activities work together to form a cohesive user experience

An email app might have one activity that shows a list of new emails, another activity to compose an email, and another activity for reading emails



Andy Brown 1:45 PM
Bring Your Parents to Work Day!
Hey! What do you think about a...

Keri Anderson 1:39 PM
Picture from last Saturday
Check out the new friend we made, Me...

Regis, Peter, Rachel 3 Sep 29
Board game night?
Sunday works! If you can get Dex...

Aruna Knight
Book you recommended
About to go on a trip and was h...

Board game night?

Regis Miller Sep 29
Maria, I'm trying to schedule a board game night this week, when would you be ...

Peter Johnson Sep 29
What do you think about Sunday? My car seats five so I could pick up Dexter a...

Rachel Shin
to me
Sep 29

Sunday works! If you can get Dexter and Sophie I will handle the chips and salsa

Activities II

Although the activities work together to form a cohesive user experience each one is independent of the others.

If permitted, a different app can start activity in another app.

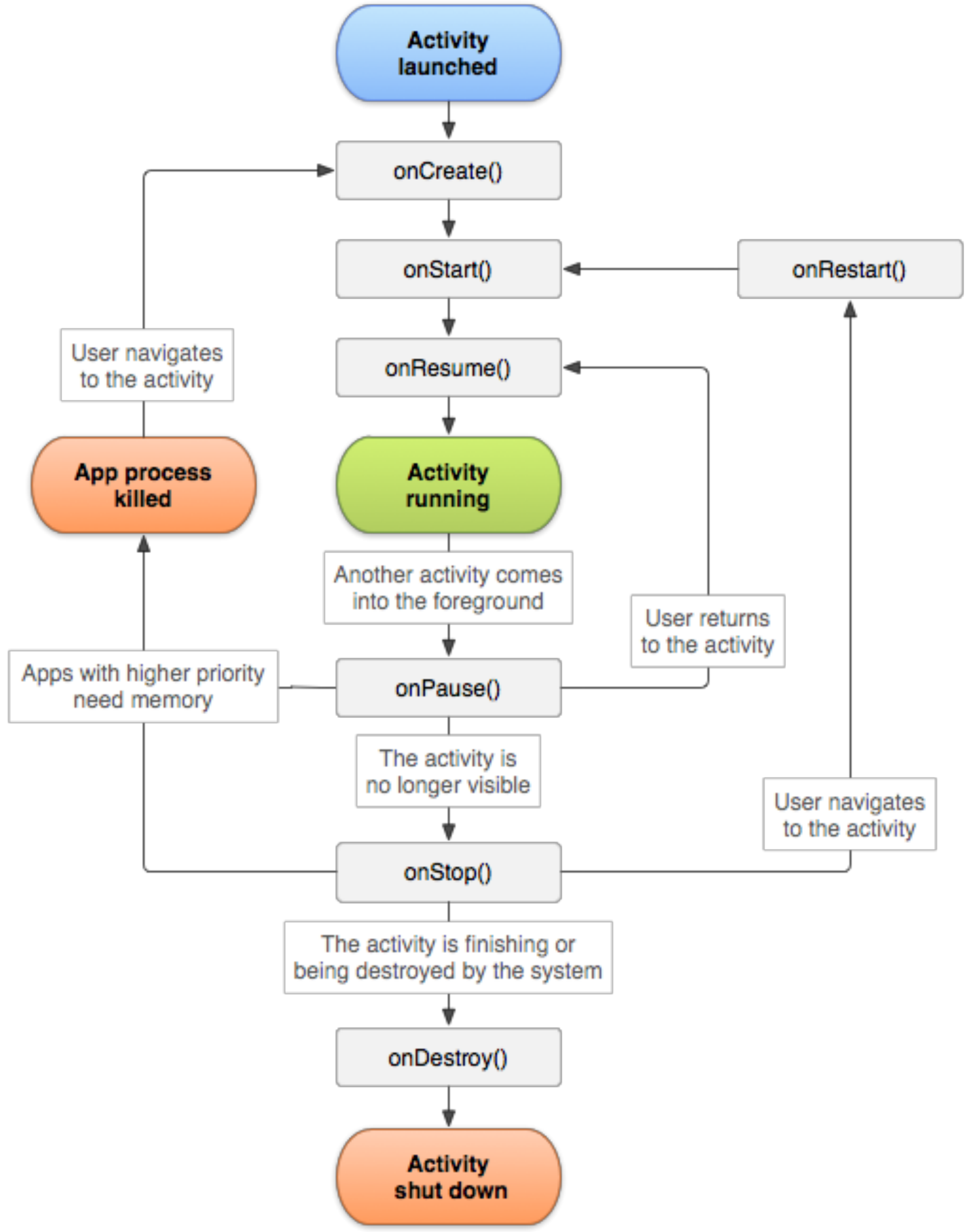
A camera app can start the activity in the email app that composes new mail to allow the user to share a picture

Activity Lifecycle

To navigate transitions between stages of the activity lifecycle, the Activity class provides a core set of six callbacks:

onCreate(),
onStart(),
onResume(),
onPause(),
onStop(),
onDestroy().

The system invokes each of these callbacks as an activity enters a new state.


Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down