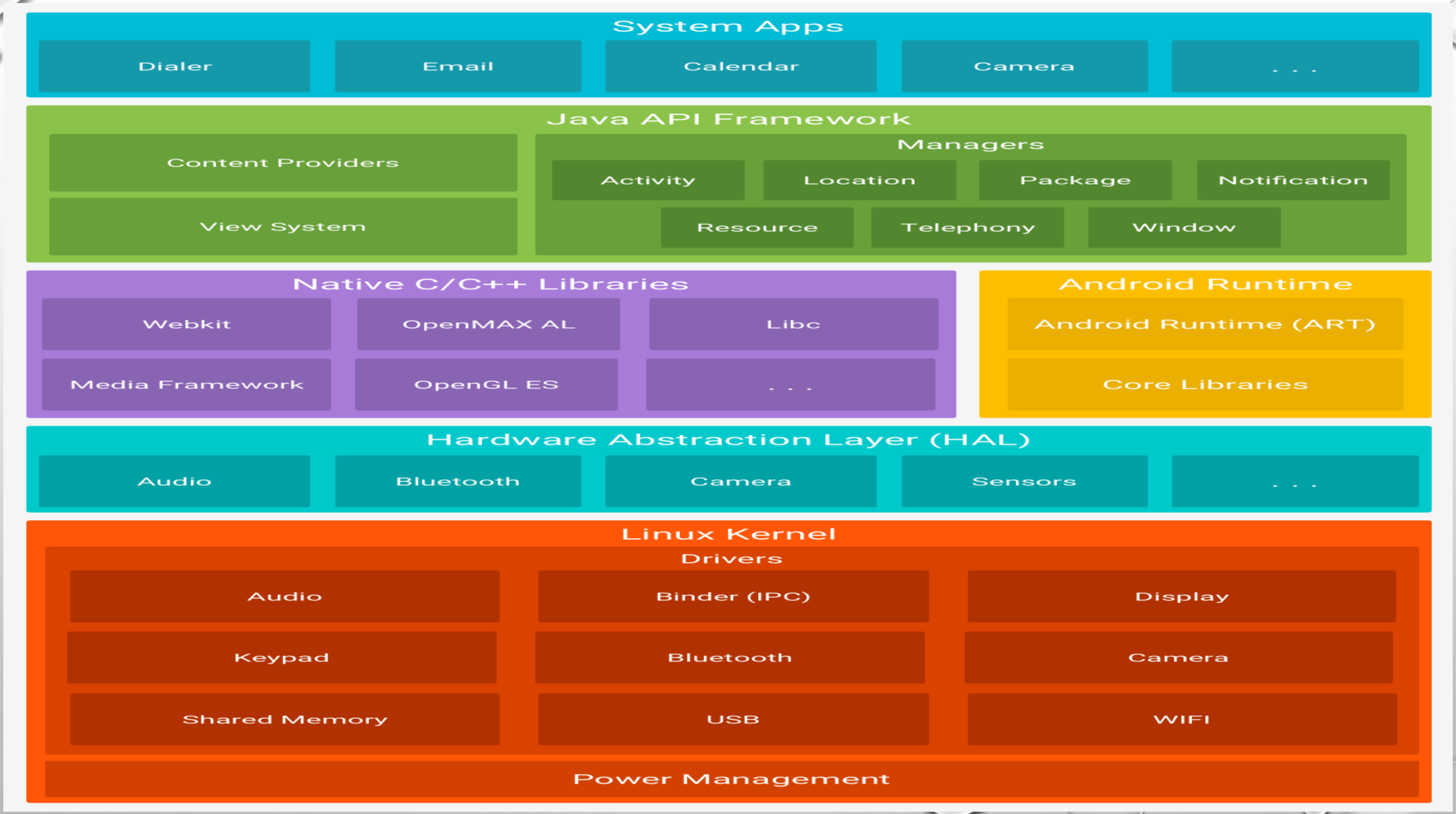
ANDROID DEVELOPMENT

PLATFORM ARCHITECTURE



System Apps

Dialer | Email | Calendar | Camera | . . .

Java API Framework

Content Providers

View System

Managers

Activity | Location | Package | Notification

Resource | Telephony | Window

Native C/C++ Libraries

Webkit | OpenMAX AL | Libc

Media Framework | OpenGL ES | . . .

Android Runtime

Android Runtime (ART)

Core Libraries

Hardware Abstraction Layer (HAL)

Audio | Bluetooth | Camera | Sensors | . . .

Linux Kernel

Drivers

Audio | Binder (IPC) | Display

Keypad | Bluetooth | Camera

Shared Memory | USB | WIFI

Power Management

CONT …

- 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 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.
- **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.

CONT …

- **ANDROID RUNTIME**
  - 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 OPTIMIZED FOR MINIMAL MEMORY FOOTPRINT.
  - BUILD TOOLCHAINS, SUCH AS JACK, COMPILE JAVA SOURCES INTO DEX BYTECODE, WHICH CAN RUN ON THE ANDROID PLATFORM.
  - SOME OF THE MAJOR FEATURES OF ART INCLUDE THE FOLLOWING:
    - AHEAD-OF-TIME (AOT) AND JUST-IN-TIME (JIT) COMPILATION
    - OPTIMIZED GARBAGE COLLECTION (GC)
    - BETTER DEBUGGING SUPPORT, INCLUDING A DEDICATED SAMPLING PROFILER, DETAILED DIAGNOSTIC EXCEPTIONS AND CRASH REPORTING, AND THE ABILITY TO SET WATCHPOINTS TO MONITOR SPECIFIC FIELDS
  - 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.

CONT …

- **JAVA API FRAMEWORK**
  - THE ENTIRE FEATURE-SET OF THE ANDROID OS IS AVAILABLE TO YOU THROUGH APIS WRITTEN IN THE JAVA LANGUAGE. THESE APIS FORM THE BUILDING BLOCKS YOU NEED 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 AND PROVIDES A COMMON NAVIGATION BACK STACK
    - CONTENT PROVIDERS THAT ENABLE APPS TO ACCESS DATA FROM OTHER APPS, SUCH AS THE CONTACTS APP, OR TO SHARE THEIR OWN DATA

CONT …

- 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.

AHEAD-OF-TIME (AOT) COMPILATION

- AT INSTALL TIME, ART COMPILES APPS USING THE ON-DEVICE **DEX2OAT** TOOL.
- THIS UTILITY ACCEPTS DEX FILES AS INPUT AND GENERATES A COMPILED APP EXECUTABLE FOR THE TARGET DEVICE.
- THE UTILITY SHOULD BE ABLE TO COMPILE ALL VALID DEX FILES WITHOUT DIFFICULTY.
- HOWEVER, SOME POST-PROCESSING TOOLS PRODUCE INVALID FILES THAT MAY BE TOLERATED BY DALVIK BUT CANNOT BE COMPILED BY ART.

JUST-IN-TIME (JIT) COMPILER

- JIT COMPILATION WORKS IN THIS MANNER:
  - THE USER RUNS THE APP, WHICH THEN TRIGGERS ART TO LOAD THE .DEX FILE.
  - IF THE .OAT FILE (THE AOT BINARY FOR THE .DEX FILE) IS AVAILABLE, ART USES THEM DIRECTLY. NOTE THAT .OAT FILES ARE GENERATED REGULARLY. HOWEVER, THAT DOES NOT IMPLY THEY CONTAIN COMPILED CODE (AOT BINARY).
  - IF NO .OAT FILE IS AVAILABLE, ART RUNS THROUGH EITHER JIT OR AN INTERPRETER TO EXECUTE THE .DEX FILE. ART WILL ALWAYS USE THE .OAT FILES IF AVAILABLE. OTHERWISE, IT WILL USE THE APK AND EXTRACT IT IN MEMORY TO GET TO THE .DEX INCURRING A BIG MEMORY OVERHEAD (EQUAL TO THE SIZE OF THE DEX FILES).
  - JIT IS ENABLED FOR ANY APPLICATION THAT IS NOT COMPILED ACCORDING TO THE "SPEED" COMPILATION FILTER (WHICH SAYS, COMPILE AS MUCH AS YOU CAN FROM THE APP).
  - THE JIT PROFILE DATA IS DUMPED TO A FILE IN A SYSTEM DIRECTORY. ONLY THE APPLICATION HAS ACCESS TO THE DIRECTORY.
  - THE AOT COMPILATION (DEX2OAT) DAEMON PARSES THAT FILE TO DRIVE ITS COMPILATION.