Layered Architectures:
Android's Layered Architecture

Douglas C. Schmidt

d.schmidt@vanderbilt.edu

www.dre.vanderbilt.edu/~schmidt

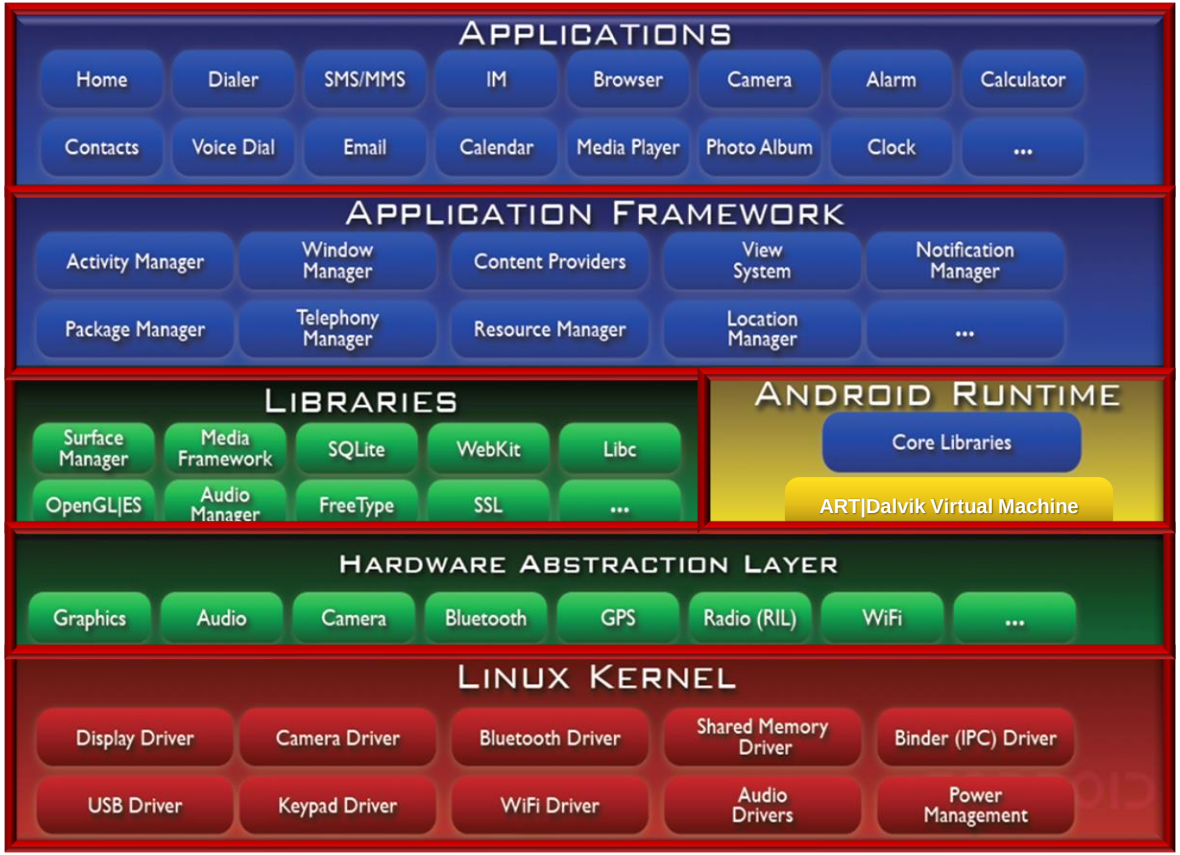
Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

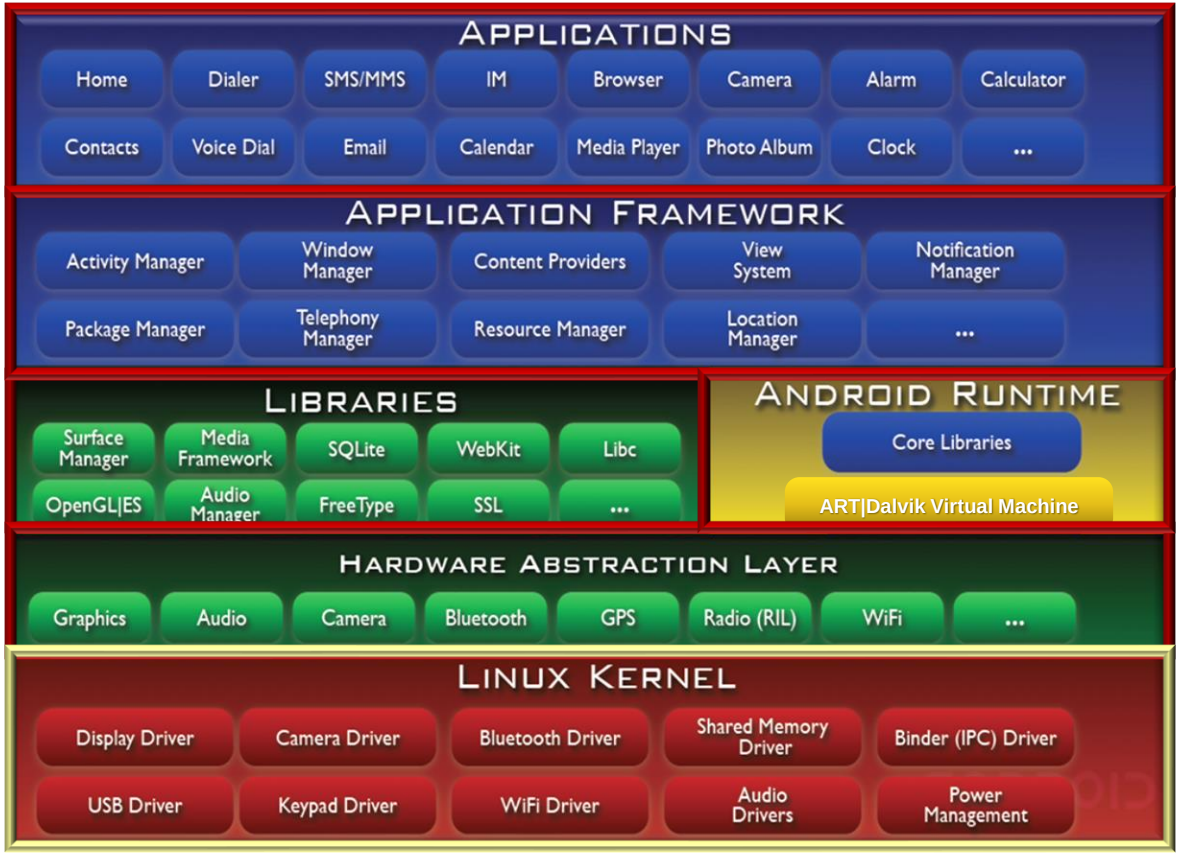
- Know what layered architectures are
- Understand the *Layers* architectural pattern
- Recognize the layers in Android's software stack



An Overview of Android's Layered Architecture

An Overview of Android's Layered Architecture

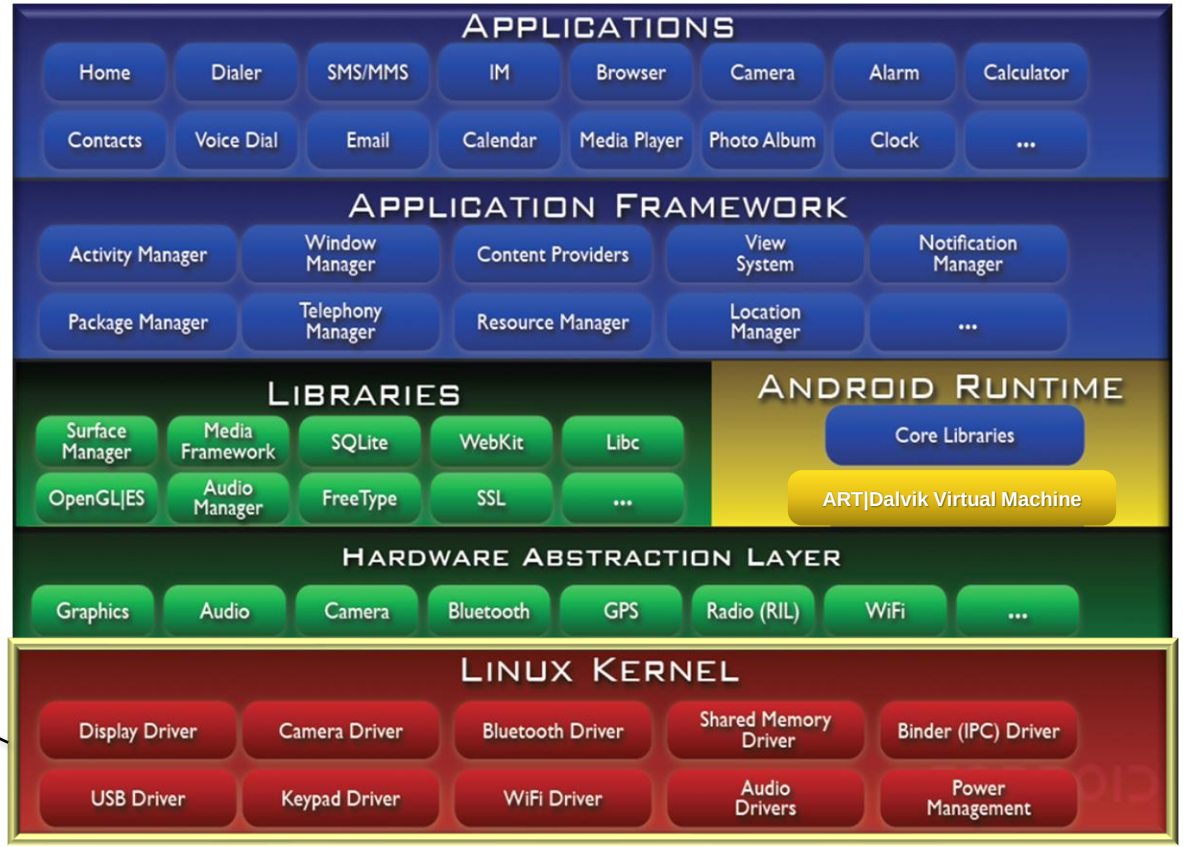
- Android's architecture is structured into multiple layers of abstraction



An Overview of Android's Layered Architecture

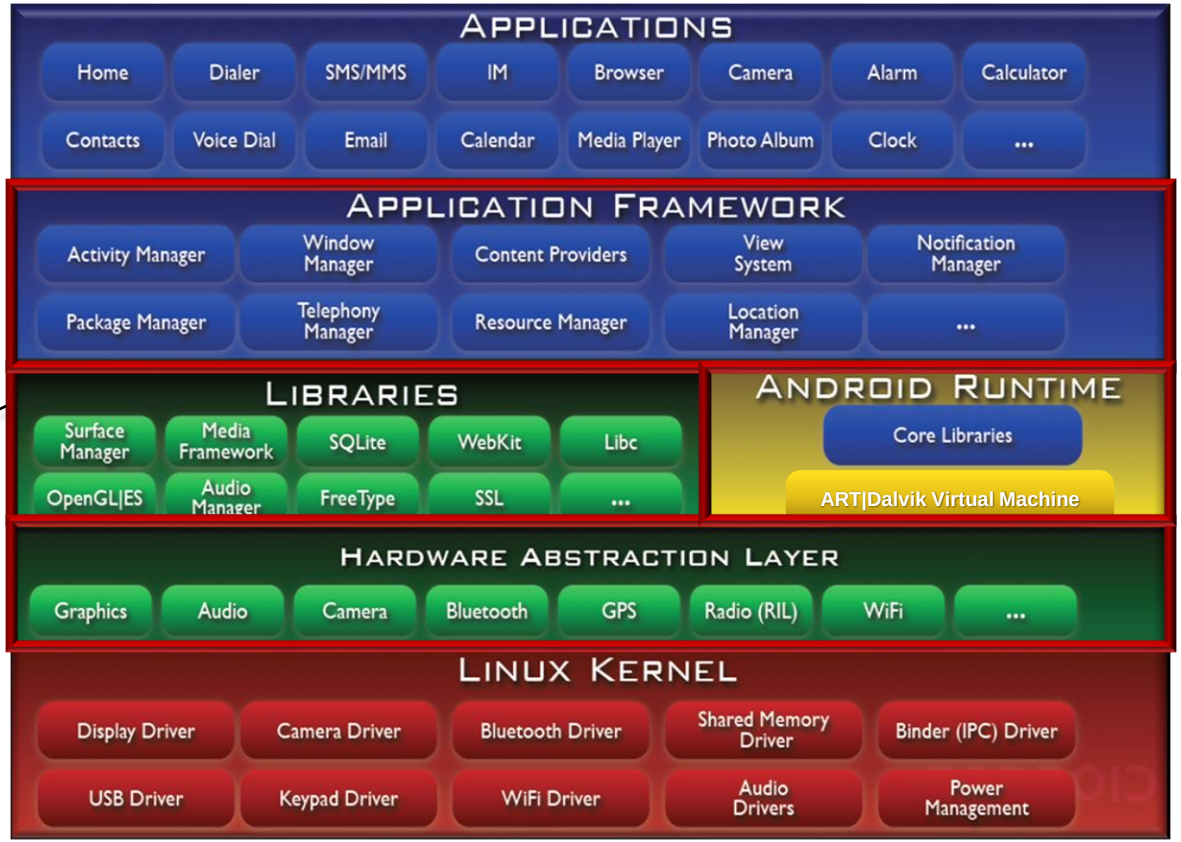
- Android's architecture is structured into multiple layers of abstraction

The Android Linux kernel controls hardware & manages system resources



An Overview of Android's Layered Architecture

- Android's architecture is structured into multiple layers of abstraction

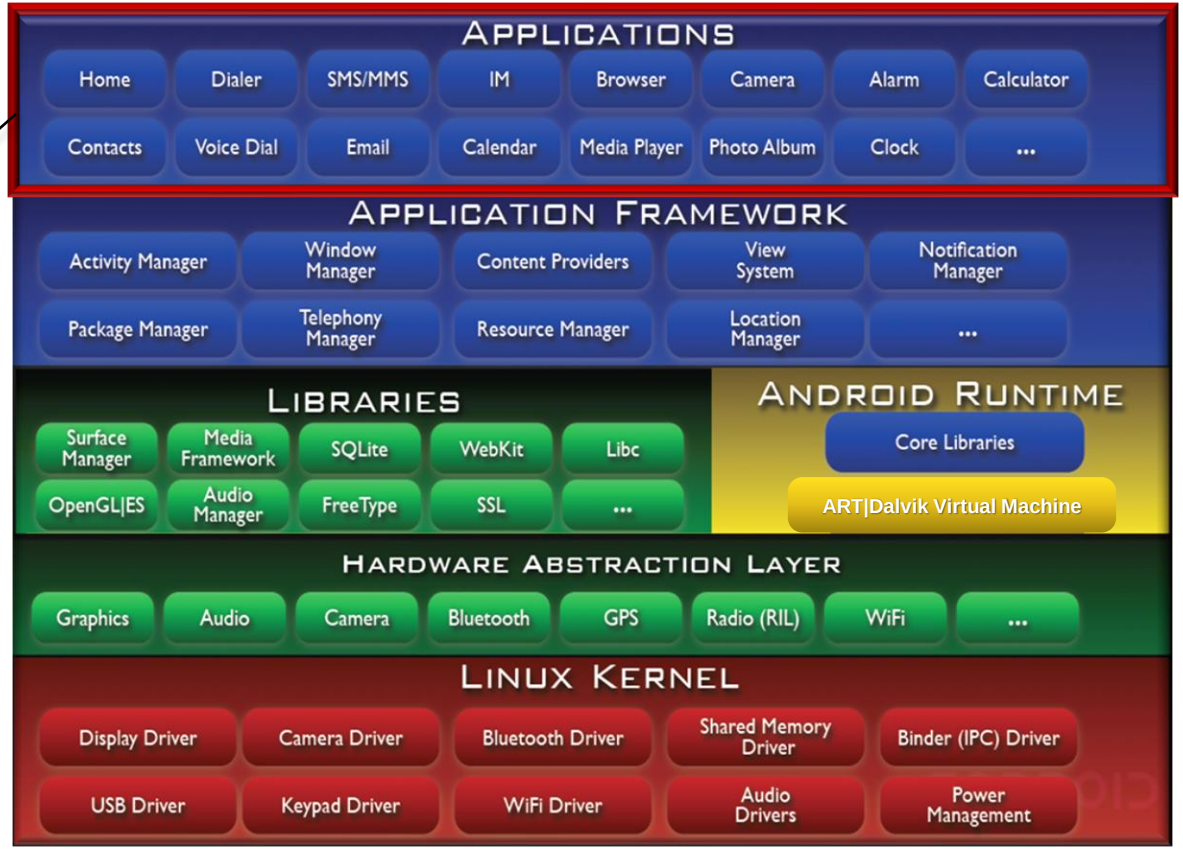


Several middleware layers provide higher-level reusable services to apps

An Overview of Android's Layered Architecture

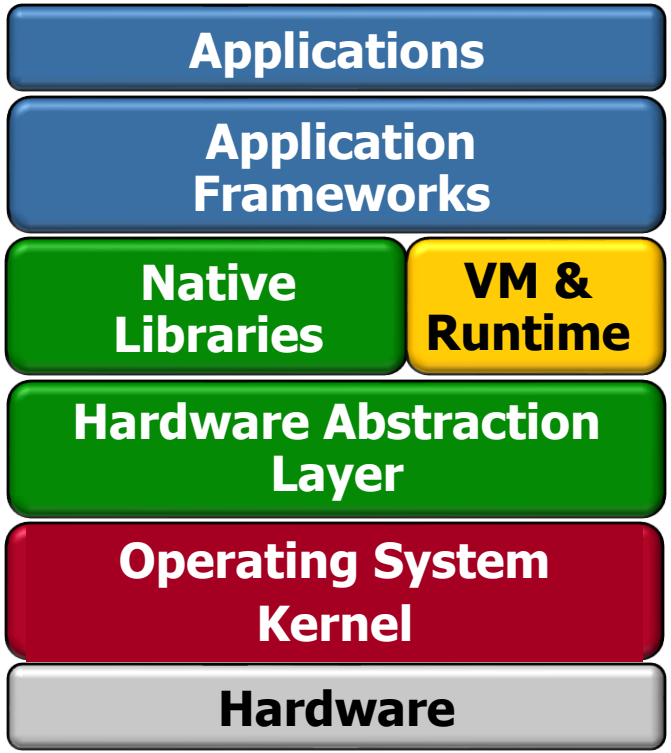
- Android's architecture is structured into multiple layers of abstraction

The application layer provides nicely packaged functionality to end-users



An Overview of Android's Layered Architecture

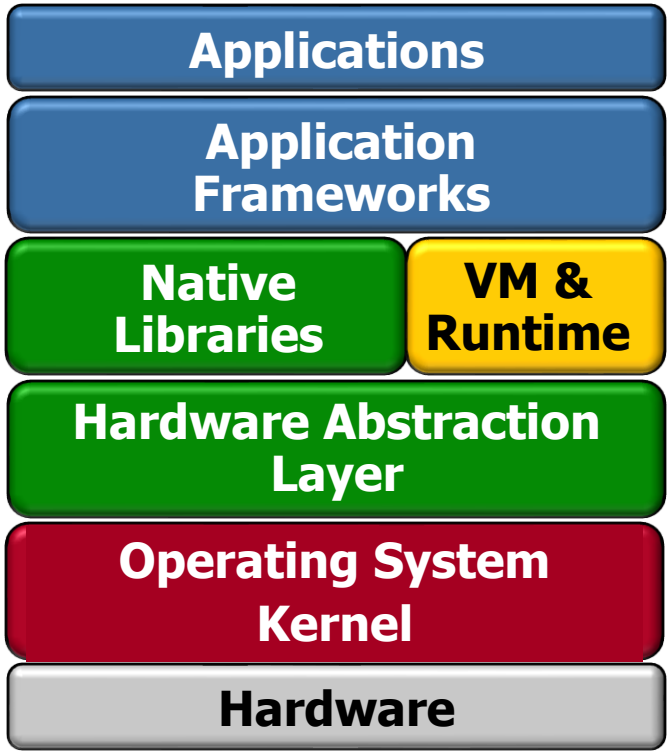
- Layering is applied in complex systems like Android for several reasons



An Overview of Android's Layered Architecture

- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse

An intentional strategy for increasing productivity & improving software quality

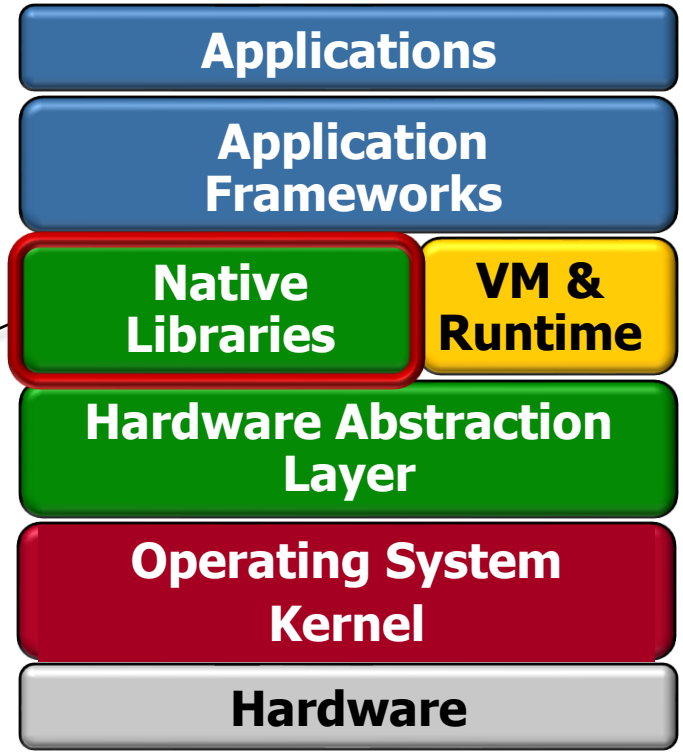


See en.wikipedia.org/wiki/Code_reuse#Systematic_software_reuse

An Overview of Android's Layered Architecture

- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse

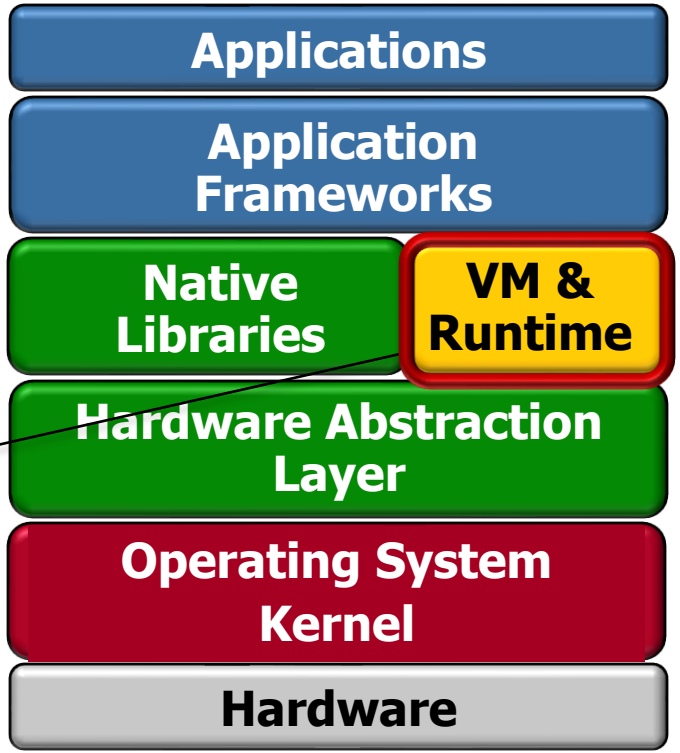
libC provides a common API for accessing OS kernel capabilities



An Overview of Android's Layered Architecture

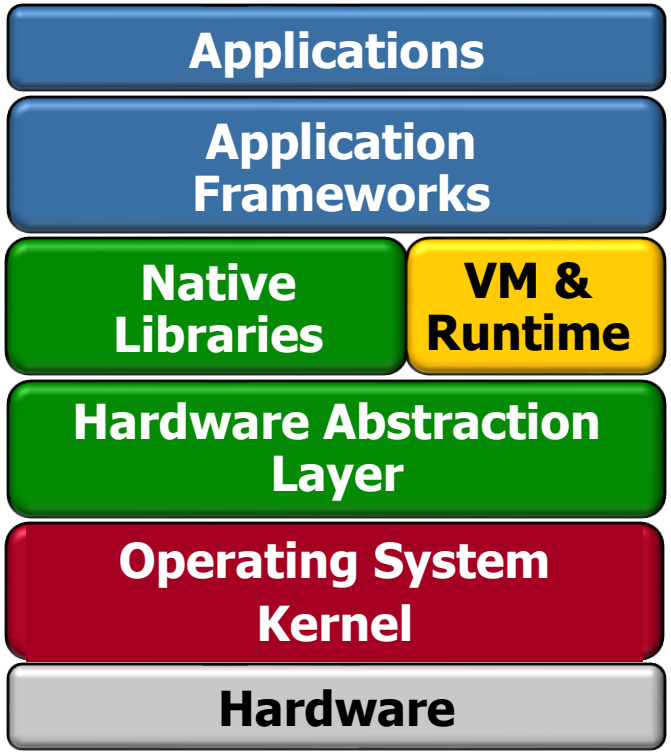
- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse

Enable apps to run concurrently over various types of multi-core hardware



An Overview of Android's Layered Architecture

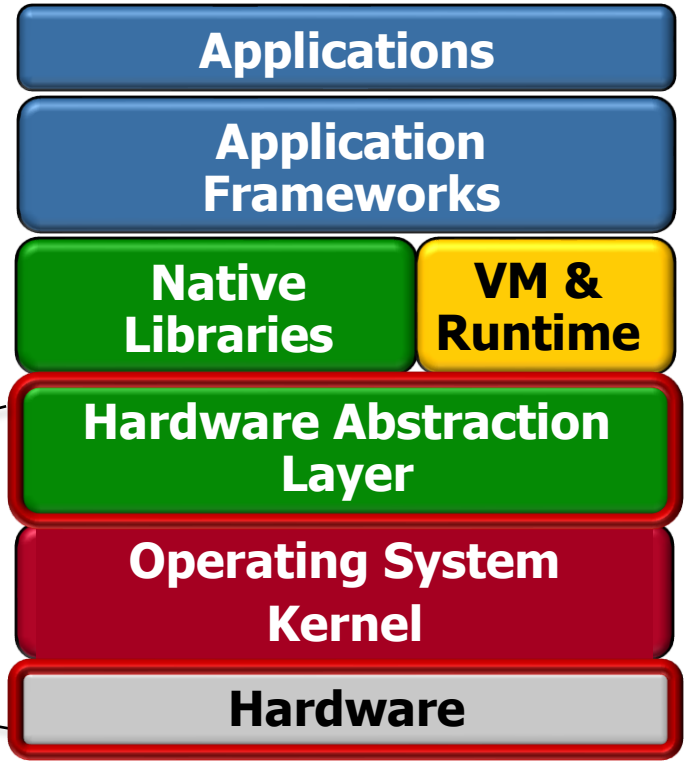
- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse
  - Enable "plug & play" replacement of certain layer implementations



An Overview of Android's Layered Architecture

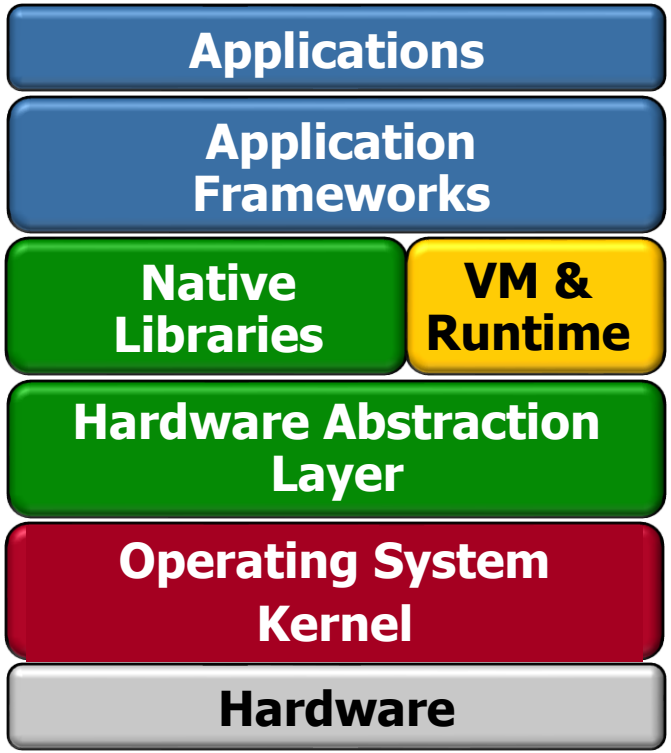
- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse
  - Enable "plug & play" replacement of certain layer implementations

Shield apps from inconsistent hardware APIs



An Overview of Android's Layered Architecture

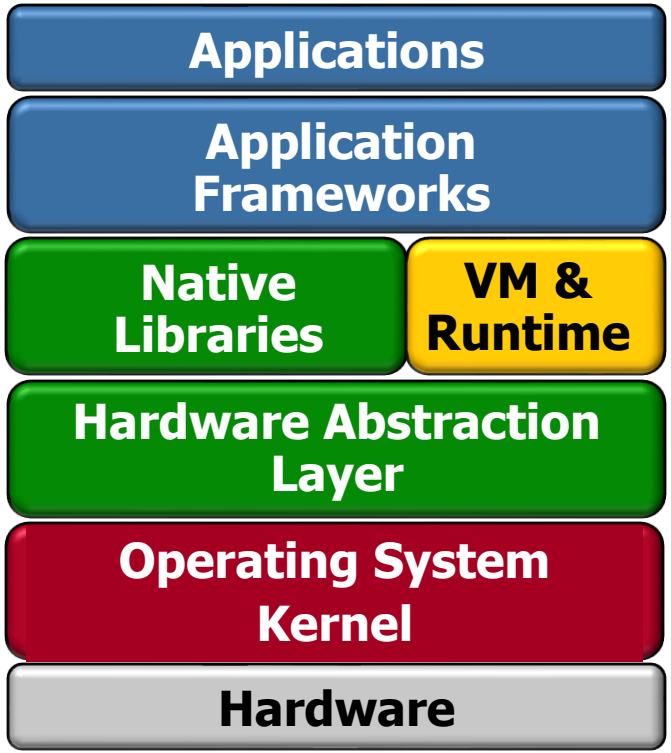
- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse
  - Enable "plug & play" replacement of certain layer implementations



Effects of updates can be confined to the layer whose implementation changes

An Overview of Android's Layered Architecture

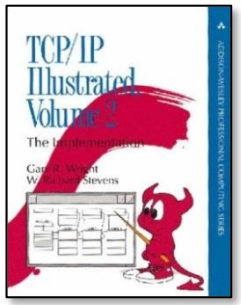
- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse
  - Enable "plug & play" replacement of certain layer implementations
  - Reduce the complexity of APIs that app developers must understand

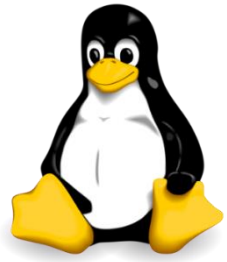


See en.wikipedia.org/wiki/Facade_pattern

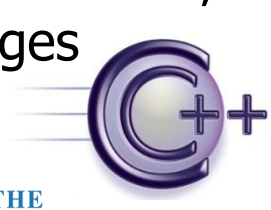
An Overview of Android's Layered Architecture

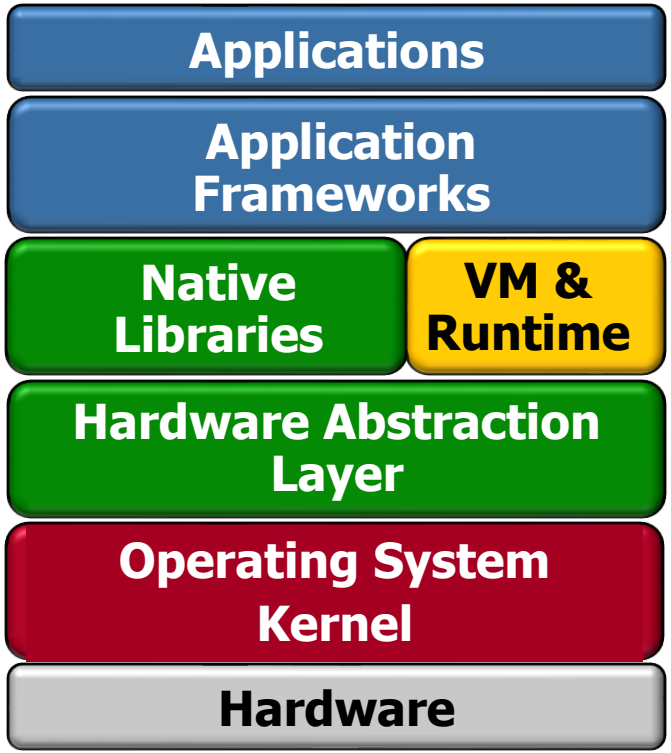
- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse
  - Enable "plug & play" replacement of certain layer implementations
  - Reduce the complexity of APIs that app developers must understand
  - Enable use of popular protocols, APIs, & programming languages





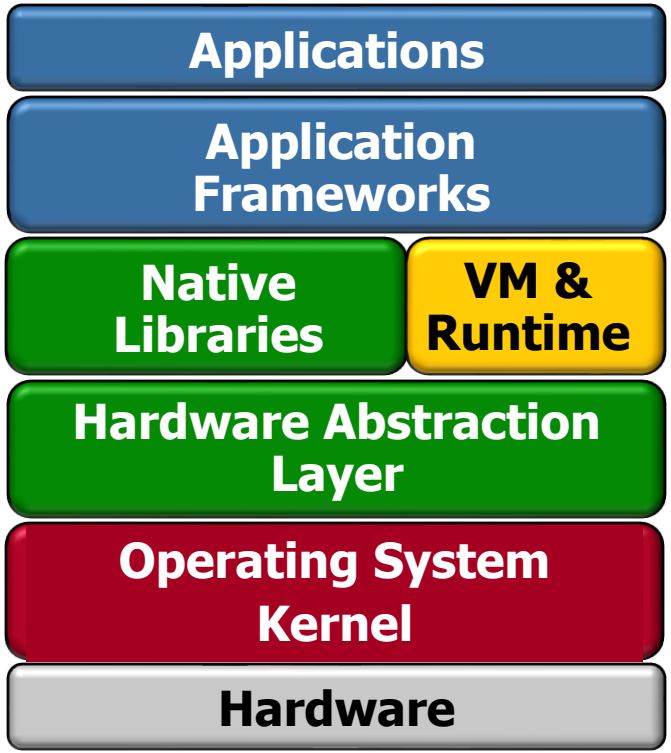
THE C PROGRAMMING LANGUAGE





An Overview of Android's Layered Architecture

- Layering is applied in complex systems like Android for several reasons, e.g.
  - Enhance systematic software reuse
  - Enable "plug & play" replacement of certain layer implementations
  - Reduce the complexity of APIs that app developers must understand
  - Enable use of popular protocols, APIs, & programming languages
    - These popular protocols & APIs are available in open-source form



See source.android.com & source.android.com/source/building-kernels.html

End of Layered Architectures: Android's Layered Architecture