MOBILE OS

ANDROID ✓
IOS ✓

Basics of Mobile OS

- The software that allows mobile devices such as phones, tablets, and other smart devices such as wearable technologies to run applications and other programmes is known as a mobile operating system (OS). When a mobile device is turned on, it typically displays a screen with icons or squares that display information and offer an access to the application.
- An Operating System is a software that provides a link between the user and the system hardware. Furthermore, it manages all interactions between software and hardware.

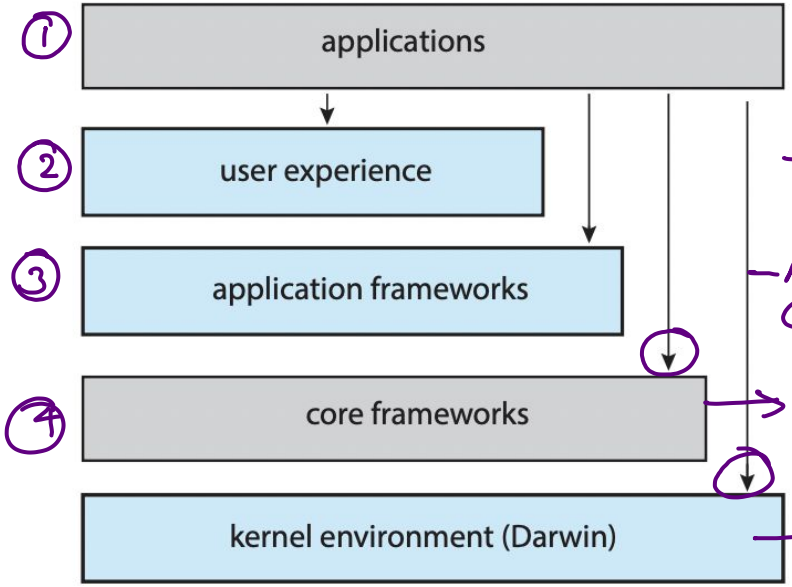
Services Provided by an Operating System

The Operating System provides certain services to the users which can be listed in the following manner:

- **Program Execution:** The Operating System is responsible for the execution of all types of programs whether it be user programs or system programs. The Operating System utilizes various resources available for the efficient running of all types of functionalities.
- **Handling Input/Output Operations:** The Operating System is responsible for handling all sorts of inputs, i.e, from the keyboard, mouse, desktop, etc. The Operating System does all interfacing most appropriately regarding all kinds of Inputs and Outputs.
  For example, there is a difference between all types of peripheral devices such as mice or keyboards, the Operating System is responsible for handling data between them.
- **Manipulation of File System:** The Operating System is responsible for making decisions regarding the storage of all types of data or files, i.e, floppy disk/hard disk/pen drive, etc. The Operating System decides how the data should be manipulated and stored.
- **Error Detection and Handling:** The Operating System is responsible for the detection of any type of error or bugs that can occur while any task. The well-secured OS sometimes also acts as a countermeasure for preventing any sort of breach of the Computer System from any external source and probably handling them.
- **Resource Allocation:** The Operating System ensures the proper use of all the resources available by deciding which resource to be used by whom for how much time. All the decisions are taken by the Operating System.

- **Networking:** Operating systems provide networking capabilities, allowing the computer system to connect to other systems and devices over a network. This can include features such as network protocols, network interfaces, and network security.
- **Scheduling:** Operating systems provide scheduling algorithms that determine the order in which tasks are executed on the system. These algorithms prioritize tasks based on their resource requirements and other factors to optimize system performance.
- **Interprocess Communication:** Operating systems provide mechanisms for applications to communicate with each other, allowing them to share data and coordinate their activities.
- **Performance Monitoring:** Operating systems provide tools for monitoring system performance, including CPU usage, memory usage, disk usage, and network activity. This can help identify performance bottlenecks and optimize system performance.
- **Backup and Recovery:** Operating systems provide backup and recovery mechanisms to protect data in the event of system failure or data loss.

IOS



→ User interface macOS ⇒ Aqua
iOS ⇒ Springboard

→ API for obj-C & swift.
Cocoa & Cocoa touch

→ graphics / media / openGL / quicktime

→ Mach / BSD

Architecture of Apple's macOS and iOS operating systems.

- Apple's macOS operating system is designed to run primarily on desktop and laptop computer systems, whereas iOS is a mobile operating system designed for the iPhone smartphone and iPad tablet computer.

  - **User experience layer**. This layer defines the software interface that allows users to interact with the computing devices. macOS uses the ***Aqua*** user interface, which is designed for a mouse or trackpad, whereas iOS uses the ***Springboard*** user interface, which is designed for touch devices.
  - **Application frameworks layer**. This layer includes the ***Cocoa*** and ***Cocoa Touch*** frameworks, which provide an API for the Objective-C and Swift programming languages. The primary difference between Cocoa and Cocoa Touch is that the former is used for developing macOS applications, and the latter by iOS to provide support for hardware features unique to mobile devices, such as touch screens.
  - **Core frameworks**. This layer defines frameworks that support graphics and media including, Quicktime and OpenGL.

- **Kernel environment**. This environment, also known as **Darwin**, includes the Mach microkernel and the BSD UNIX kernel. We will elaborate on Darwin shortly.

- applications can be designed to take advantage of user-experience features or to bypass them and interact directly with either the application framework or the core framework. Additionally, an application can forego frameworks entirely and communicate directly with the kernel environment.
- The iOS operating system is generally much more restricted to developers than macOS and may even be closed to developers. For example, iOS restricts access to POSIX and BSD APIs on iOS, whereas they are openly available to developers on macOS.

CORE SERVICES Layer:

Some important frameworks are present in the CORE SERVICES Layer which helps the iOS operating system to cure itself and provide better functionality. It is the 2nd lowest layer in the Architecture as shown above. Below are some important frameworks present in this layer:

1. **Address Book Framework-**
   The Address Book Framework provides access to the contact details of the user.
2. **Cloud Kit Framework-**
   This framework provides a medium for moving data between your app and iCloud.
3. **Core Data Framework-**
   This is the technology that is used for managing the data model of a Model View Controller app.
4. **Core Foundation Framework-**
   This framework provides data management and service features for iOS applications.
5. **Core Location Framework-**
   This framework helps to provide the location and heading information to the application.
6. **Core Motion Framework-**
   All the motion-based data on the device is accessed with the help of the Core Motion Framework.

MEDIA Layer:

With the help of the media layer, we will enable all graphics video, and audio technology of the system. This is the second layer in the architecture. The different frameworks of MEDIA layers are:

1. **ULKit Graphics-**
   This framework provides support for designing images and animating the view content.
2. **Core Graphics Framework-**
   This framework support 2D vector and image-based rendering and it is a native drawing engine for iOS.
3. **Core Animation-**
   This framework helps in optimizing the animation experience of the apps in iOS.
4. **Media Player Framework-**
   This framework provides support for playing the playlist and enables the user to use their iTunes library.
5. **AV Kit-**
   This framework provides various easy-to-use interfaces for video presentation, recording, and playback of audio and video.
6. **Open AL-**
   This framework is an Industry Standard Technology for providing Audio.

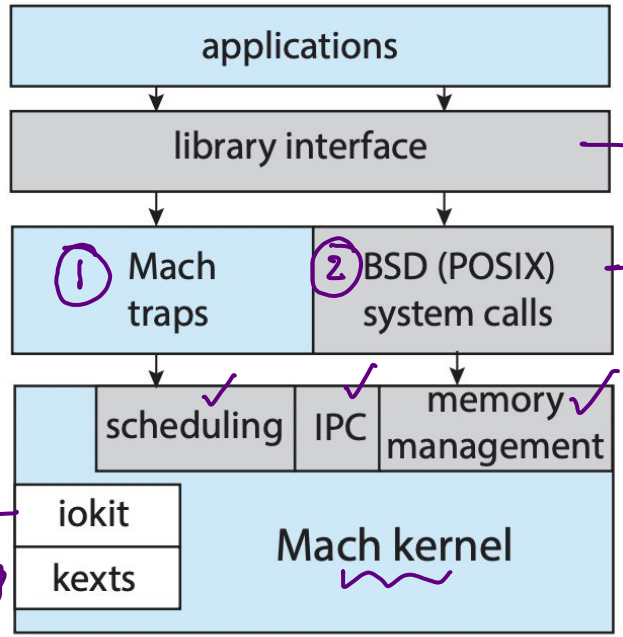
COCOA TOUCH:

COCOA Touch is also known as the application layer which acts as an interface for the user to work with the iOS Operating system. It supports touch and motion events and many more features. The COCOA TOUCH layer provides the following frameworks :

1. **EvenKit Framework-**
   This framework shows a standard system interface using view controllers for viewing and changing events.
2. **GameKit Framework-**
   This framework provides support for users to share their game-related data online using a Game Center.
3. **MapKit Framework-**
   This framework gives a scrollable map that one can include in your user interface of the app.
4. **PushKit Framework-**
   This framework provides registration support.

(2)

Darwin



C libraries + n/w + security + pgm lang.

portable OS interface

device drivers & dynamically loadable modules

Figure 2.17 The structure of Darwin.

Darwin

- Darwin is a layered system that consists primarily of the Mach microkernel and the BSD UNIX kernel.
- Most OS provides single system-call interface to the kernel—such as through the standard C library on UNIX and Linux systems —Darwin provides two system-call interfaces: Mach system calls and BSD system calls.
- The interface to these system calls is a rich set of libraries that includes not only the standard C library but also libraries that provide networking, security, and progamming language support.
- Beneath the system-call interface, Mach provides fundamental OS services, including memory management, CPU scheduling, and interprocess communication (IPC) facilities such as message passing and remote procedure calls (RPCs). Much of the functionality provided by Mach is available through kernel abstractions
- In addition to Mach and BSD, the kernel environment provides an I/O kit for development of device drivers and dynamically loadable modules

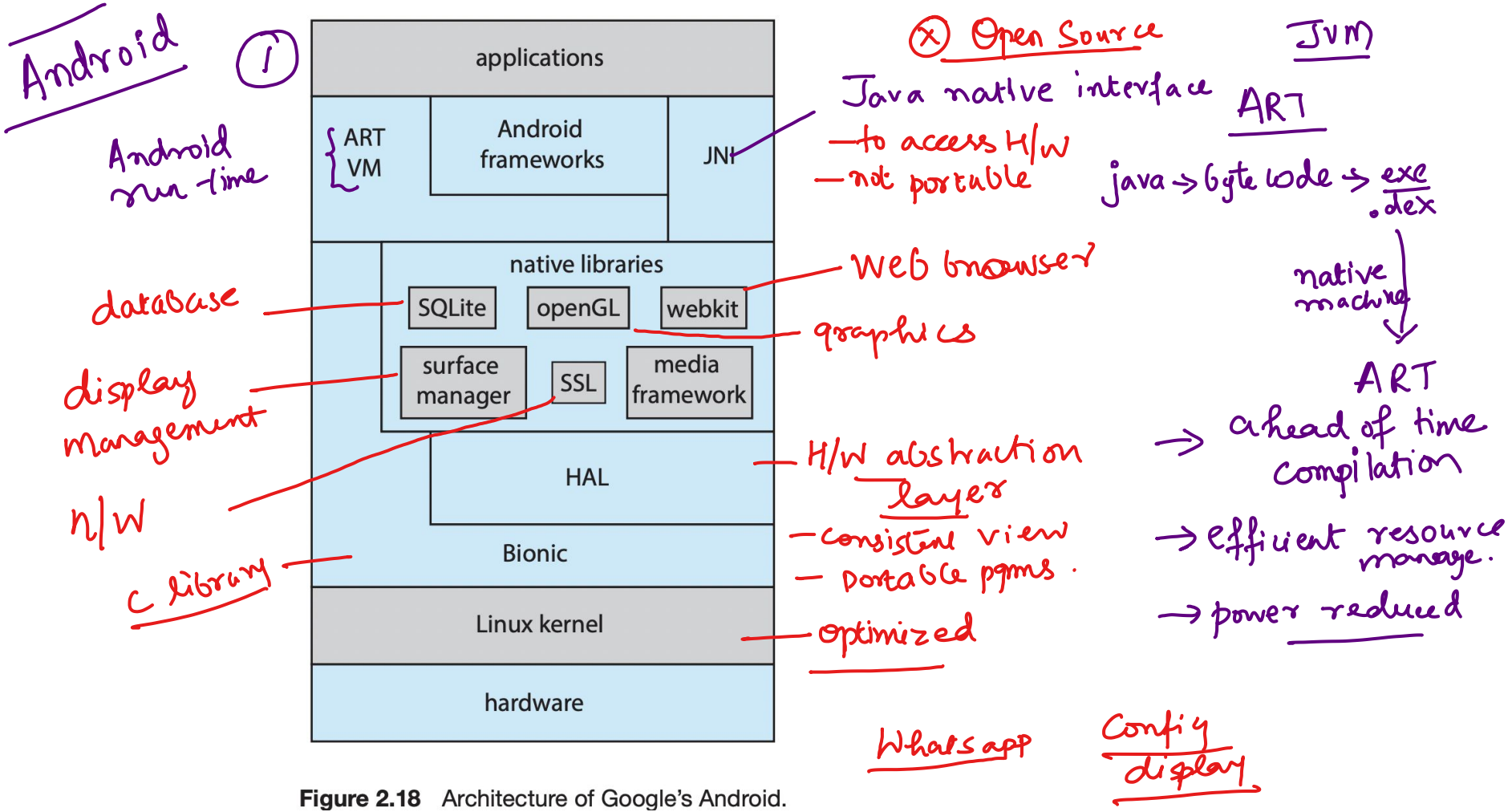


Figure 2.18 Architecture of Google's Android.

Key points

- The Android operating system was designed by the Open Handset Alliance (led primarily by Google) and was developed for Android smartphones and tablet computers.
- Android is similar to iOS in that it is a layered stack of software that provides a rich set of frameworks supporting graphics, audio, and hardware features.
- Software designers for Android devices develop applications in the Java language using Android API for Java development. Java applications are compiled into a form that can execute on the Android RunTime ART, a virtual machine designed for Android and optimized for mobile devices with limited memory and CPU processing capabilities.
- Java programs are first compiled to a Java bytecode .class file and then translated into an executable .dex file.
- ART performs ahead-of-time (AOT) compilation. .dex files are compiled into native machine code when they are installed on a device, from which they can execute on the ART. AOT compilation allows more efficient application execution as well as reduced power consumption, features that are crucial for mobile systems.

- Android developers can also write Java programs that use the Java native interface—or JNI—which allows developers to bypass the virtual machine and instead write Java programs that can access specific hardware features.
- The set of native libraries available for Android applications includes frameworks for developing web browsers (webkit), database support (SQLite), and network support, such as secure sockets (SSLs).
- By abstracting all hardware, such as the camera, GPS chip, and other sensors, the HAL provides applications with a consistent view independent of specific hardware.
- The standard C library used by Linux systems is the GNU C library (glibc). Google instead developed the Bionic standard C library for Android. Not only does Bionic have a smaller memory footprint than glibc, but it also has been designed for the slower CPUs that characterize mobile devices.
- At the bottom of Android's software stack is the Linux kernel. Google has modified the Linux kernel used in Android in a variety of areas to support the special needs of mobile systems, such as power management. It has also made changes in memory management and allocation and has added a new form of IPC known as Binder

Android Process Hierarchy

- **Foreground process**—The current process visible on the screen, representing the application the user is currently interacting with
- **Visible process**—A process that is not directly visible on the foreground but that is performing an activity that the foreground process is referring to (that is, a process performing an activity whose status is displayed on the foreground process)
- **Service process**—A process that is similar to a background process but is performing an activity that is apparent to the user (such as streaming music)
- **Background process**—A process that may be performing an activity but is not apparent to the user.
- **Empty process**—A process that holds no active components associated with any application

Extra

Platform libraries –

The Platform Libraries includes various C/C++ core libraries and Java based libraries such as Media, Graphics, Surface Manager, OpenGL etc. to provide a support for android development.

- **Media** library provides support to play and record an audio and video formats.
- **Surface manager** responsible for managing access to the display subsystem.
- **SGL** and **OpenGL** both cross-language, cross-platform application program interface (API) are used for 2D and 3D computer graphics.
- **SQLite** provides database support and **FreeType** provides font support.
- **Web-Kit** This open source web browser engine provides all the functionality to display web content and to simplify page loading.
- **SSL (Secure Sockets Layer)** is security technology to establish an encrypted link between a web server and a web browser.