



ENCS3390 (operating systems)

Instructor: Dr. Abdel salam sayyad

Date: 10/22/2023

Student name: Abdel rahman shahen

Student ID: 1211753

Homework #1

Q1: Compare and contrast the Android and iOS operating systems from the following viewpoints:

- I) customizability
- II) performance
- III) business model (how does iOS make money for Apple? How does Android make money for Google?).

Ans:



	Customizability	Performance	Business model
	Android offers significantly more customizability options compared to iOS. Android users can deeply personalize their devices, change the look and feel of the interface, and access third-party app stores.	Android performance varies based on device specs. High-end models offer fast multitasking, while budget phones might lag with demanding apps.	Google profits by offering Android for free to device manufacturers, while generating income from pre-installed Google apps and services, as well as commissions from app sales on the Google Play Store.
	iOS emphasizes simplicity and uniformity, curbing extensive customization options for users. Apple's design focus on consistency ensures a seamless and intuitive experience across all devices but limits personalization capabilities.	iOS provides consistent performance across Apple devices, ensuring smooth multitasking and app responsiveness regardless of the device's cost.	Apple earns revenue from iOS through device sales, App Store commissions, and services like Apple Music and iCloud storage.

Table 1.1 (ios Vs android)

Q2: AOT vs. JIT

- I) Why does Android use Ahead-of-Time (AOT) compilation rather than Just-in-Time (JIT) compilation?
- II) What are the advantages and disadvantages of each method?

Ans I:

1. AOT compilation allows the Android system to compile the app's bytecode into native machine code. This pre-compiled native code executes faster than bytecode interpreted at runtime, leading to improved performance.
2. Since compiling code can be CPU-intensive, pre-compiling apps conserves battery life by minimizing the processor's workload when the app is launched.
3. AOT compilation significantly reduces app startup times. By having the code pre-compiled.

Ans II:

Aspect	(AOT) Compilation	(JIT) Compilation
Advantages		
Faster Startup Times	Significant reduction in app startup times.	Initial delays can occur due to real-time compilation.
Performance	Optimized, fast execution of pre-compiled native code.	Dynamic optimization based on runtime usage patterns.
Battery Efficiency	Reduces processor workload, conserving battery life.	Real-time compilation can consume more CPU power, potentially draining battery.
Security	Reduces the surface area for potential runtime attacks.	there isn't a specific security-related
Disadvantages		
Increased App Size	Pre-compiled native code results in larger app sizes.	Smaller app size due to bytecode, but with less optimization.
Limited Dynamic Optimization	Lack of dynamic adaptability in optimizing code paths.	Offers dynamic adaptation to different hardware, leading to variable performance.
Flexibility	Less dynamic, optimized for specific hardware and conditions.	More adaptable to diverse devices and runtime conditions.

Table 2.1 (AOT Vs JIT)

Q3: Native vs. cross-Platform

- What is the difference between native mobile code and cross-platform mobile code?
- What are the advantages and disadvantages of each?
- How can a programming framework like FLUTTER produce mobile apps that work on both Android and iOS?

Ans a:

Native mobile code refers to applications that are developed specifically for a particular mobile operating system. native apps are typically written in Java, Kotlin, Swift, and Objective-C. Native apps have direct access to the device's hardware and operating system features, allowing for high performance. However, developing native apps for different platforms requires writing separate codebases, which can be time-consuming.

Cross-platform mobile code, on the other hand, refers to applications that are developed using technologies that allow the same codebase to run on multiple platforms. These frameworks use web technologies like HTML, CSS, and JavaScript, or a single language like Dart (Flutter) or C#.Cross-platform development can significantly reduce development time and effort. However, there may be limitations in accessing specific platform features, and the performance might not be as optimized as native apps.

Ans b:

Aspect	Native Development	Cross-Platform Development
Advantages	Optimized performance, native feel.	Faster development, shared codebase.
Disadvantages	Longer development time, higher costs.	Slightly less optimized, limited features.

Table 3.1 (Native Vs cross-platform)

Ans c:

Flutter achieves cross-platform compatibility by compiling code written in Dart, its programming language, into native machine code for both Android and iOS platforms. Using a single codebase. Additionally, Flutter uses platform channels to access platform-specific APIs and features, enabling developers to leverage device capabilities.

Q5: Describe google Chrome as an example of multi-process, multi-threaded application.

Ans:

Each open tab and extension runs as a separate process, enhancing stability and security. Within these processes, multiple threads handle different tasks concurrently, ensuring faster performance and responsiveness. Chrome also offloads rendering tasks to a dedicated GPU process, leveraging hardware acceleration. This intricate architecture allows for efficient resource management, prevents crashes from affecting the entire browser, and provides a secure browsing experience.