
UNIT 5

MOBILE APPLICATIONS

PMDM - 2DAM

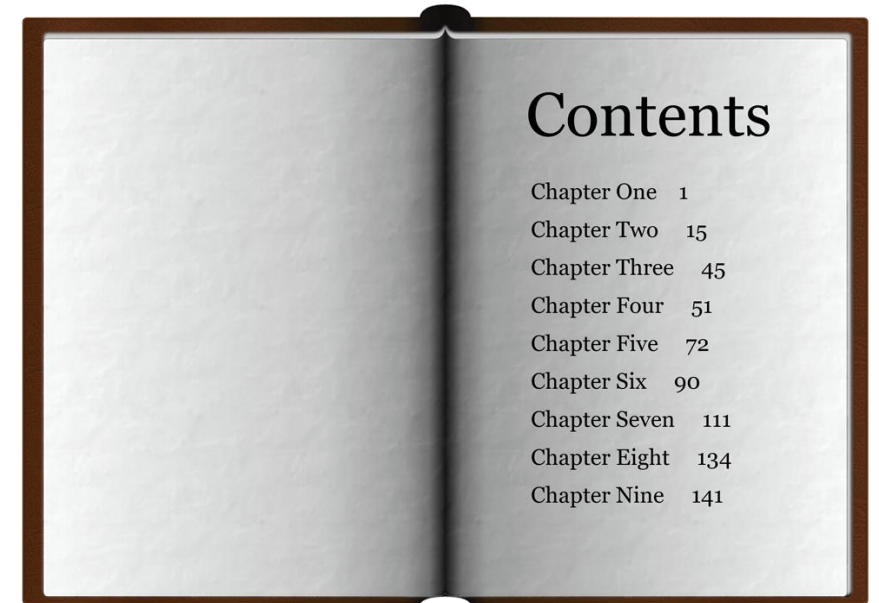
Àngel Olmos (a.olmosginer@edu.gva.es)

Jose Pascual Rocher (jp.rochercamps@edu.gva.es)



Content

- INTRODUCTION
 - MOBILE COMMUNICATION TECHNOLOGIES
 - ANDROID
 - iOS
- TYPES OF MOBILE APPLICATIONS
 - NATIVE APPLICATIONS
 - RESPONSIVE WEB APPLICATIONS
 - HYBRID APPLICATIONS
 - PROGRESSIVE WEB APPLICATIONS
 - COMPILED APPLICATIONS



INTRODUCTION

- We will take a quick look at the primary technologies at our disposal for portable app development
- Find yourself wanting to create a mobile/portable app:
 1. What's the first thing that crosses your mind? Where to start?
 2. Which programming language should be used?
 3. In which development environment?
 4. And for which devices?



INTRODUCTION

Mobile Communication Technologies

The mobile **communication network** has evolved over time with different **generations** bringing changes to the previous one and so did the mobile devices



INTRODUCTION

Mobile Communication Technologies

- Generation 0: First communications between mobile devices using **radio waves**
- 1G technology: the first automated mobile communication network was launched in **1979 in Japan**. Security limitations since the voice **calls were replicated on the radio towers** → interception
- 2G technology: 1990s. The first to provide **digital voice and data** (SMS). From the 2000s on it provided “high-speed” Internet



INTRODUCTION

Mobile Communication Technologies

- 3G technology: increase in data rates, greater voice and data capacity, as well as lowering the cost of transmissions
- 4G technology: provide **high speed** at low prices for both voice and data, as well as increasing the security of communications
- 5G technology: Began its commercial launch in 2019, is **10 times faster than 4G** (up to 1Gbps)
- 6G technology: Reduce the latency and **increase the Tx speed**. Expected commercialization in 2030 (1st real use cases could arrive as from 2026)



INTRODUCTION

Android

- Android is an OS developed by Google and based on the **Linux Kernel**
- Made specifically for touch-screen portable devices: mobile phones, tablets, smart watches, “TVs” or even some cars
- Inaugural beta on November 5, 2007 → initial commercial iteration **Android 1.0** saw the light on September 23, **2008**
- The development followed on an **annual basis since 2011**
- Sweets naming changed to version number from Android 10 on



INTRODUCTION

Android

NAME	VERSION	LAUNCHING DATE	MAIN IMPROVEMENTS
Android 1.0	1.0 - 1.1	september 2008	First stable version
Android Cupcake	1.5	april 2009	Refined design Virtual keyboard Widgets for apps Copy and paste in browser Animated transitions Automatic screen rotation
Android Donut	1.6	september 2009	Quick search Revamped Android Market Adapted to more screen formats Voice synthesiser Camera and gallery improvements CDMA and VPN support

INTRODUCTION

Android

Android Eclair	2.0 - 2.1	october 2009	Routes in Maps Support for multiple accounts Live Wallpapers Flash and zoom support Improvements to pre-installed apps like Maps, browser or calendar
Android Froyo	2.2 - 2.2.3	may 2010	Voice commands Wi-Fi hotspots Improved browser performance Flash support C2DM push notifications Move apps to SD
Android Gingerbread	2.3 - 2.3.7	december 2010	API for games NFC First easter egg Icon design changes Support for WXGA resolution and higher Select before copy Support for multiple cameras Gyroscope and barometer support Video calling in Hangouts

INTRODUCTION

Android

Android Honeycomb	3.0 - 3.2.6	february 2011	Adapted for tablets System Bar Quick settings Browser tabs Hardware acceleration USB OTG support
Android Ice Cream Sandwich	4.0 - 4.0.4	october 2011	Holo interface Navigation bar Folders Roboto Typography Screenshots Face unlock Dismiss notifications one by one
Android Jelly Bean	4.1 - 4.3.1	july 2012	Google Now Smoother movement Quick settings Better accessibility Widgets on the lock screen Native emoji support

INTRODUCTION

Android

Android KitKat	4.4 - 4.4.4	october 2013	Design changes Immersive mode ART Revamped Clock, Phone and Downloads app
Android Lollipop	5.0 - 5.1.1	november 2014	Material Design New lock screen (without widgets) Performance improvements Recent improvements Settings finder
Android Marshmallow	6.0 - 6.0.1	october 2015	Runtime permissions Doze mode USB-C and 4K support Fingerprint reader support Experimental multi-window Direct Share Now on Tap

INTRODUCTION

Android

Android Nougat	7.0 - 7.1.2	august 2016	Doze improvements JIT compiler improvements Daydream VR Multi-window mode PIP on Android TV Vulkan 3D Quick app settings Launcher shortcuts
Android Oreo	8.0 - 8.1	august 2017	Project Treble Mobile PIP mode Adaptive icons Notification changes Autocomplete API Performance optimisations
Android Pie	9.0	august 2018	Privacy enhancements Brightness and smart battery App actions App slices Digital well-being Gesture navigation


INTRODUCTION

Android

Android 10 (Quince Tart)	10.0	september 2019	Dark mode Real-time subtitles Intelligent responses New gesture navigation Foldable optimisations Privacy improvements Google Play system updates
Android 11 (Red Velvet Cake)	11.0	september 2020	Changes to notifications Chat bubbles Native screen recorder Domotics in the shutdown menu One-time permission Android Auto wireless for all
Android 12 (Snow Cone)	12.0	october 2021	Material You Privacy Enhancements Approximate location permission Microphone, camera and location usage flags Domotics disappears from shutdown menu Performance improvements Scrolling screenshots

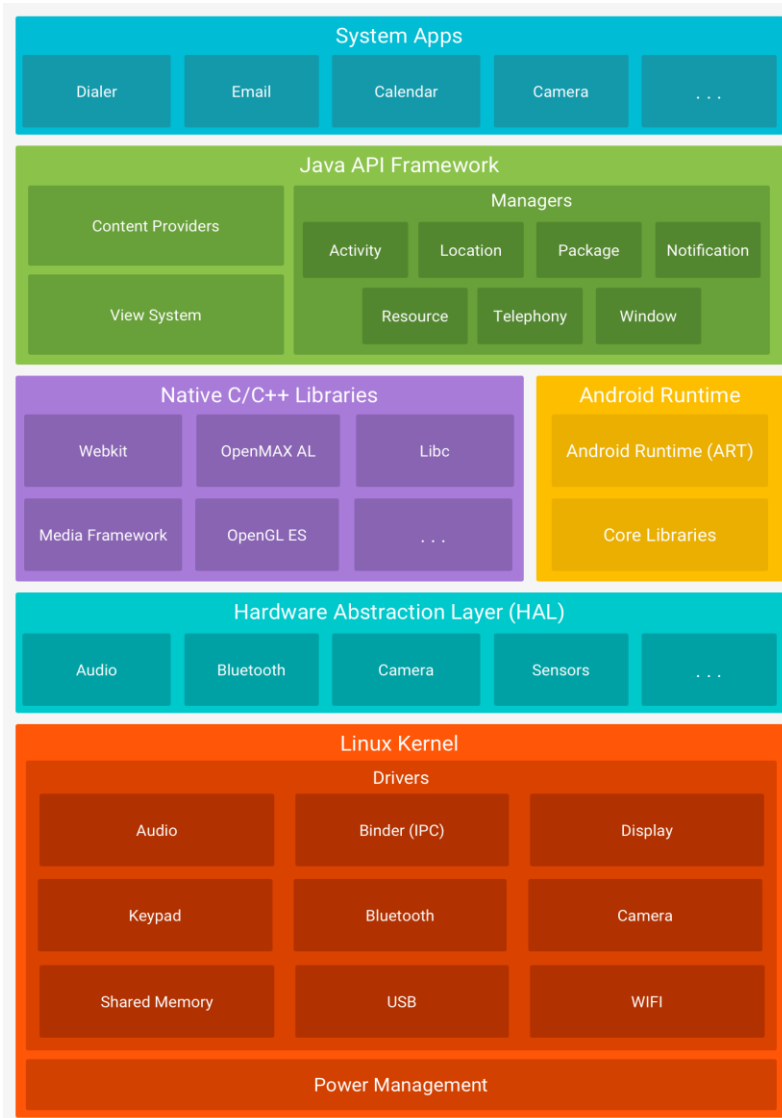
INTRODUCTION

Android

Android 13 (Tiramisu)	13.0	august 2022	More Material You customisation Permission changes New notification permissions Choose language for each app QR reader New photo selector Active apps New clipboard menu
Android 14 (Upside Down Cake)	14.0	october 2023 on Pixel devices	
Android 15 (Vanilla Ice Cream)	15.0	expected 2024	

INTRODUCTION

Android



- The Android software stack is composed of applications that run in a **Java framework**
- The **JVM** was Dalvik until version 5.0 ...
- ... to change in later versions to the **Android Runtime (ART) environment**
- Difference: Dalvik performed the compilation at runtime, while ART compiles during the installation
- Used libraries are written in C/C++: surface manager, SQLite DB, WebKit rendering engine...

INTRODUCTION

The language for Android development has traditionally been Java

Google has adopted Kotlin as the official Android programming language, which generates executable code directly in the JVM



Android

Kotlin

```
data class Person(var name: String,
var age: Int)

/* var: read and write
   val: read-only i.e. no setters
*/
```

Java

```
public class Person {
    private String name;
    private String age;
```

```
    public Person(String name, String age) {
        this.name = name;
        this.age = age;
    }
```

```
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
```

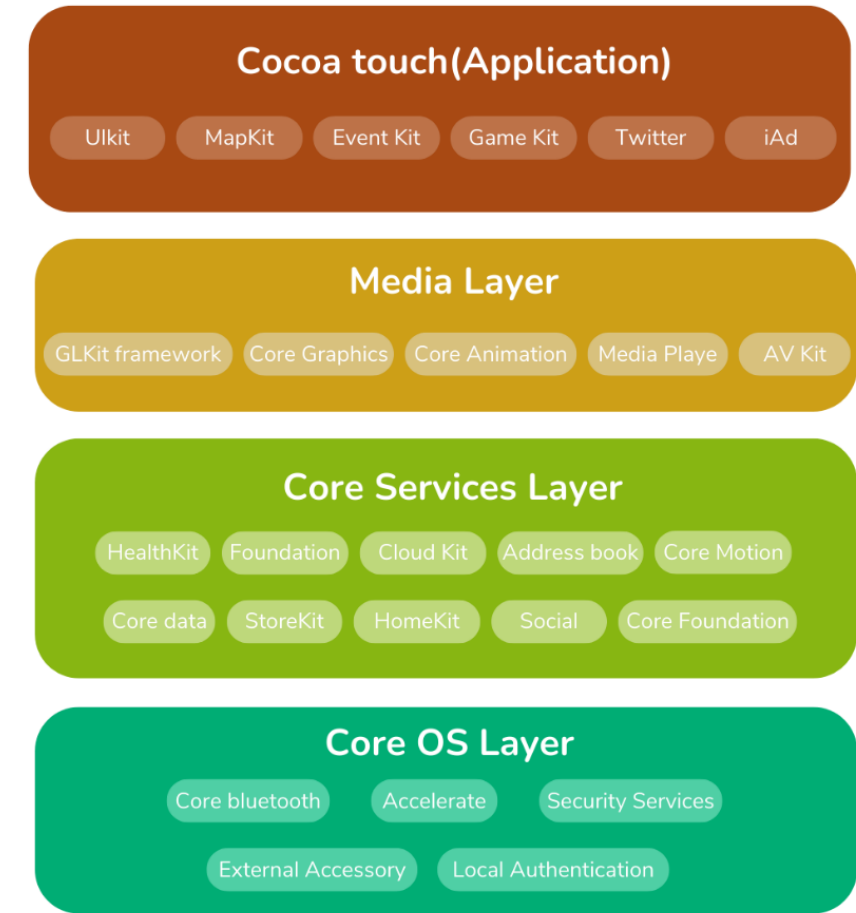
```
    public String getAge() {
        return age;
    }
    public void setAge(String age) {
        this.age = age;
    }
```

```
    @Override
    public boolean equals(Object o) {
        /*
        Code for equals function
        */
    }
    @Override
    public int hashCode() {
        /*
        Code for hashCode function
        */
    }
}
```

INTRODUCTION

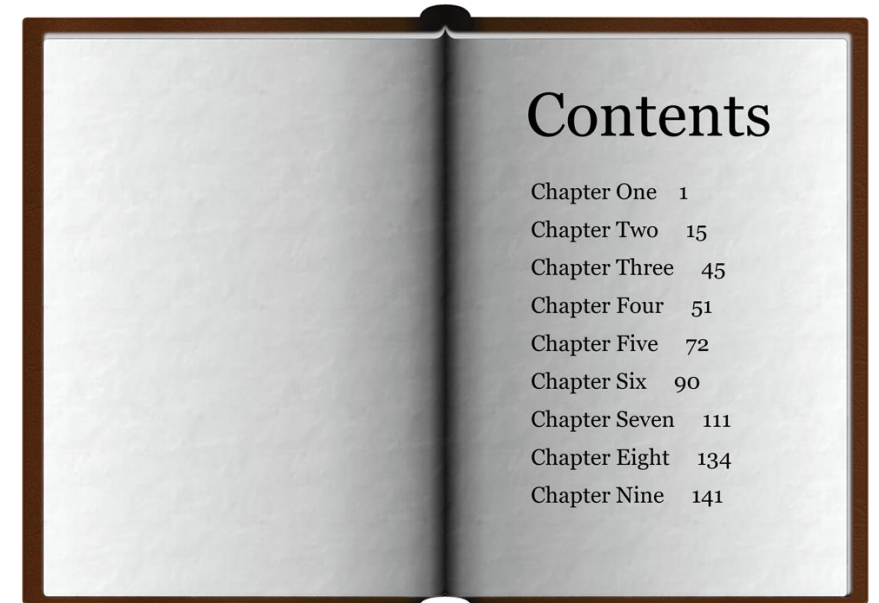
iOS

- iOS is the 2nd best-selling portable OS
- It was created for the iPhone, and later adopted in the iPod and iPad
- iOS does not allow installation on hardware from other companies
- iOS is derived from macOS, and macOS is derived from Darwin BSD, which is a **Unix-like** operating system
- Native apps development for iOS involves languages such as **Objective-C** and **Swift**



Content

- INTRODUCTION
 - MOBILE COMMUNICATION TECHNOLOGIES
 - ANDROID
 - iOS
- TYPES OF MOBILE APPLICATIONS
 - NATIVE APPLICATIONS
 - RESPONSIVE WEB APPLICATIONS
 - HYBRID APPLICATIONS
 - PROGRESSIVE WEB APPLICATIONS
 - COMPILED APPLICATIONS



TYPES OF MOBILE APPLICATIONS

NATIVE APPLICATIONS

- Developed specifically for the OS in which they will be executed
- Will make a better use of the device resources
- Allow access to all the functionalities of the platforms
- Fluid applications that offer the best user experience
- Increase in the cost of production and maintenance



TYPES OF MOBILE APPLICATIONS

RESPONSIVE WEB APPLICATIONS (WebApps)

- Based on web technology: HTML + CSS + JavaScript
- To run they only need a web browser
- Not necessary to develop in native code
- Fully cross-platform since they run on the OS web browser
- Same app can run on Android, iOS, Windows, Linux or Mac
- It will not offer as good experience to the user as native apps



TYPES OF MOBILE APPLICATIONS

HYBRID APPLICATIONS

- Use web technology to build a website (HTML + CSS + JS) and load it in a *WebView*
- A *WebView* is basically a web browser without the navigation bar → so it looks like a native app on the device
- The WebView will be in native code for each OS, but the web application is the same
- They react slower since they must communicate with the website to obtain much of the content
- Good quality network connection is a must



TYPES OF MOBILE APPLICATIONS

PROGRESSIVE WEB APPLICATIONS (PWA)

- Provide the user with many of the advantages of native applications, but with development based on web technologies
- Unlike hybrid applications, they allow operation with a poor connection to the server
- When navigating back to a previous page, user can see what was already loaded instead of getting the dreaded “currently offline” page
- Frameworks: **React** PWA Library, **Angular** PWA Framework, **Vue** PWA Framework ...



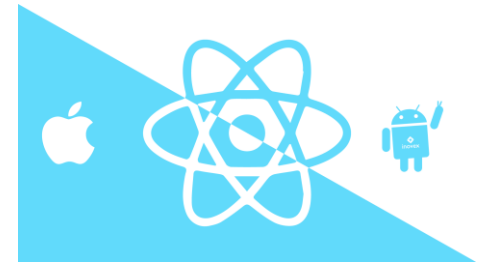
TYPES OF MOBILE APPLICATIONS

	NATIVE	WEB	HYBRID	PROGRESSIVE
DESC	Built for a specific platform	Rely on a web browser and a working Internet connection to run	Combine the functionalities of native and web apps	Web apps designed to be faster, more lightweight, and borrow native app features
PROS	Faster speed Processing efficiency Smoother UI Hardware compatibility Access to the device's functionality	Cheaper to make No device memory or storage issues Easy maintenance Accessibility	Faster development Cross-platform compatibility Cost-effective Offline capability	No installation needed Data efficiency Versatility Automatic updates
CONS	Programming is not easy Takes time Multiple codebases for the same app	Browser dependent Useless without Internet Limited functionalities	Slower speed Limited hardware access Less smooth UI	Hardware integration issues Limited hardware access Browser UI issues

TYPES OF MOBILE APPLICATIONS

COMPILED APPLICATIONS

- **Technologies to develop native applications** (not a type of mobile App) working with a **single programming language** and **compile the code to be native** on the different platforms
- **React Native:** Uses **JS** and the React library
- **Native Script:** create native applications using pure **JS**, Angular or Vue
- **Flutter:** the code is written in **Dart** and compiled to native code that runs on the device



LICENSE



Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:



Attribution — You must give [appropriate credit](#), provide a link to the license, and [indicate if changes were made](#). You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.



NonCommercial — You may not use the material for [commercial purposes](#).



ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the [same license](#) as the original.

No additional restrictions — You may not apply legal terms or [technological measures](#) that legally restrict others from doing anything the license permits.

<https://creativecommons.org/licenses/by-nc-sa/3.0/>
