TÉCNICO LISBOA

Introduction to Android

Mobile and Ubiquitous Computing

MEIC/METI 2024/25

Nuno Santos

1. ANDROID OVERVIEW

Android = More than just an OS

Android is a full stack: OS, middleware, apps, and dev tools

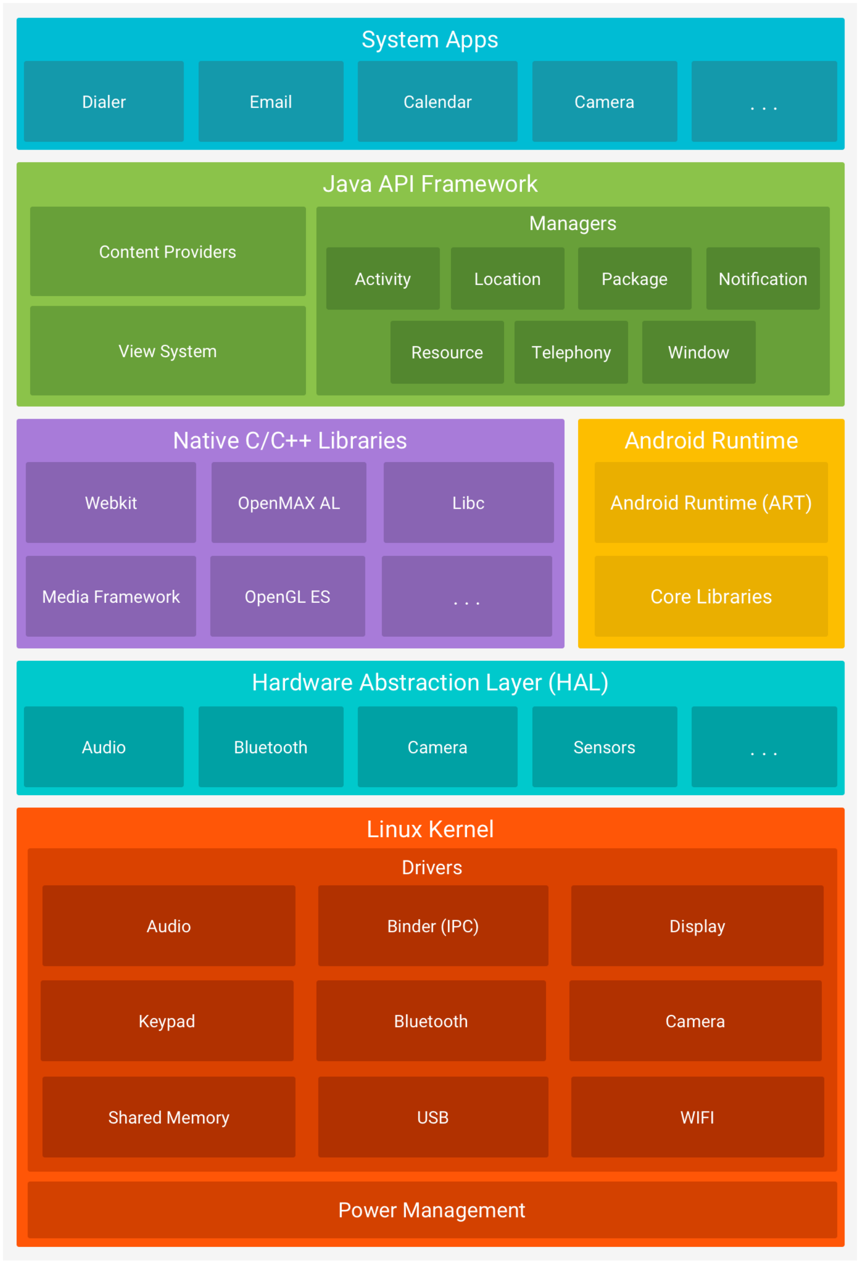
- Built on the Linux kernel, but with its own runtime and component model
- Open-source + Google-backed + supported by the Open Handset Alliance

https://developer.android.com/about

The rise of Android

- 2003: Android Inc. founded
- 2005: Acquired by Google
- 2008: Android 1.0 released
- 2017: Kotlin announced as preferred language
- 2020: Jetpack Compose revolutionizes UI
- 2023: Android 14 (Upside Down Cake)
- 2025: Android 15 (Vanilla Ice Cream)

How Android works



A layered model, from hardware to apps

- Linux kernel (drivers, memory, power mgmt)
- Hardware Abstraction Layer (HAL)
- Android Runtime (ART): optimized VM for running apps
- Framework: APIs for UI, sensors, media, etc.
- Your app: Kotlin/Java, using Jetpack libraries

https://developer.android.com/guide/platform

Noteworthy features

- Modern application framework
  - Kotlin-first with Jetpack Compose for UI
  - Apps run on the optimized Android Runtime (ART)

- Efficient runtime environment
  - Built for low memory and energy usage
  - App hibernation and background control enhance performance

- Security-enhanced Linux base
  - Hardened with SELinux, sandboxing, and scoped storage

- Powerful development ecosystem
  - Android Studio: Emulator, Layout Inspector, Profilers
  - Jetpack libraries: Room, ViewModel, WorkManager, Compose

Security and permissions

- App isolation
  - Each app runs with a unique Linux user ID
  - Sandboxed: private files, isolated execution

- Permission model
  - Runtime permission requests, not install-time
  - Users control access to sensitive features

- Data protection
  - Scoped Storage: apps access only their own files
  - Extra permissions needed for shared data or background access

Kotlin: A modern language for Android

- Kotlin and Java
  - Interoperable: Kotlin and Java work together
  - Same runtime (JVM), same tools
- Why Kotlin?
  - Official Android language since 2017
  - Faster, safer, and more concise than Java
- Key Features
  - Null safety: fewer crashes
  - Concise syntax: less code, more clarity
  - Coroutines: easy asynchronous programming
  - Lambda expressions: simple way to handle callbacks and listeners

2. ANATOMY OF ANDROID APPS

Android components vs. Linux processes

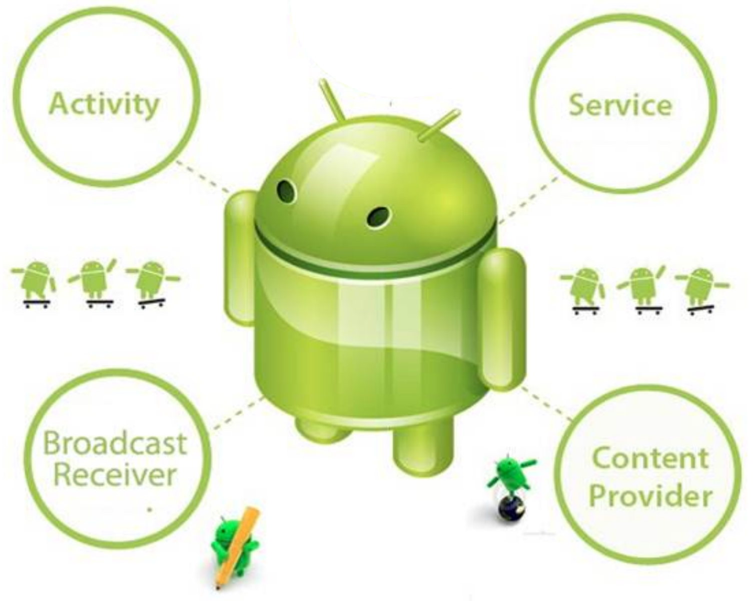
- Apps split into event-driven components, not monolithic executables
- There are four different types of application components
  - Each type has different purpose and a distinct lifecycle
  - System decides which components to run, pause, or kill

Screen/UI controller
E.g., show list of emails

Background logic
E.g., play music in background

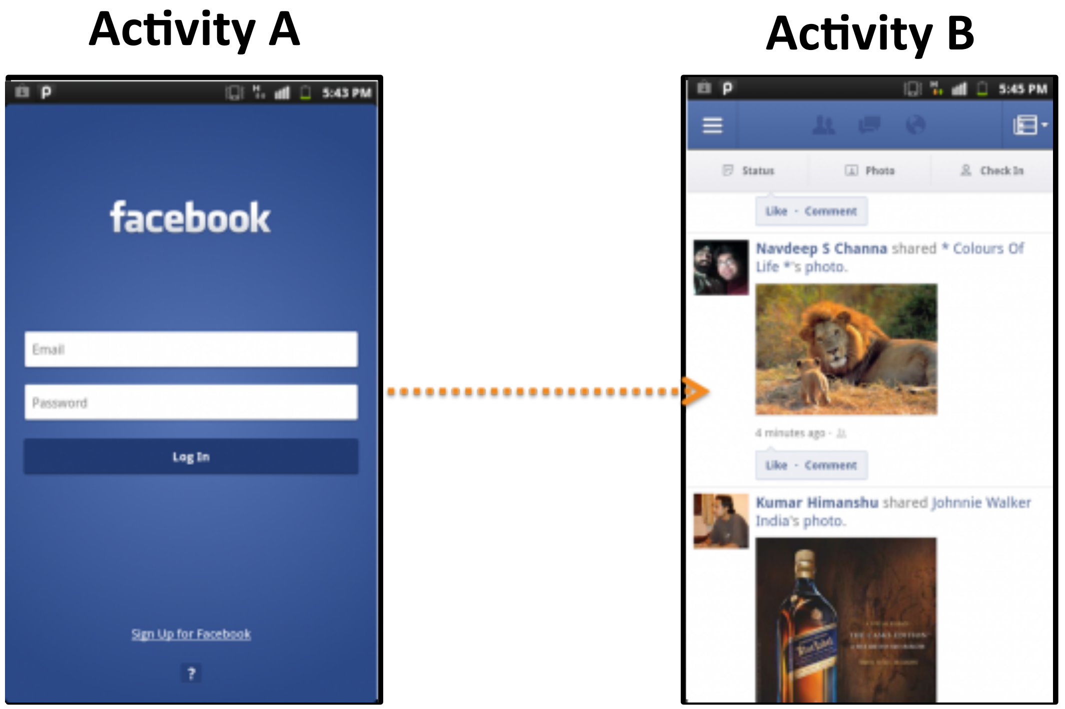
Event listener
E.g., take action if battery low

Shared data manager
E.g., manage user's contact information



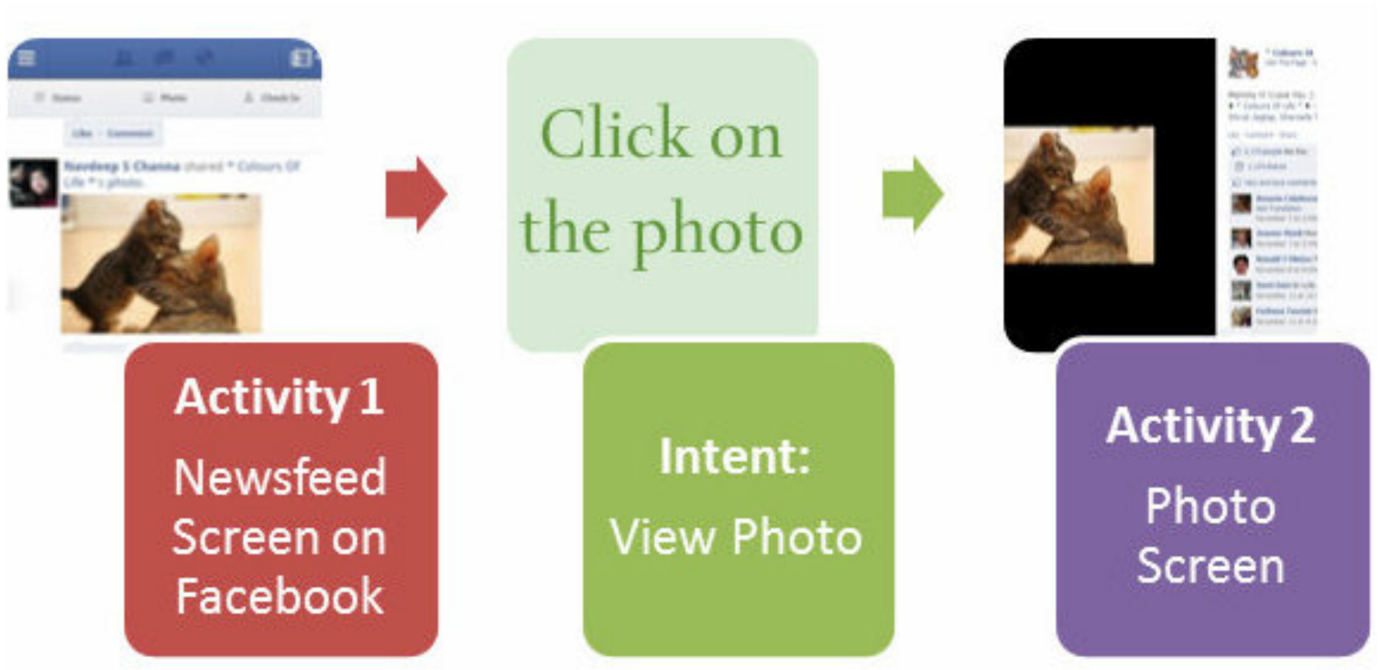
The king of components: Activities

- **Activities**: are like the pages in a website
  - Provide an interface for users to interact with the app and take an action

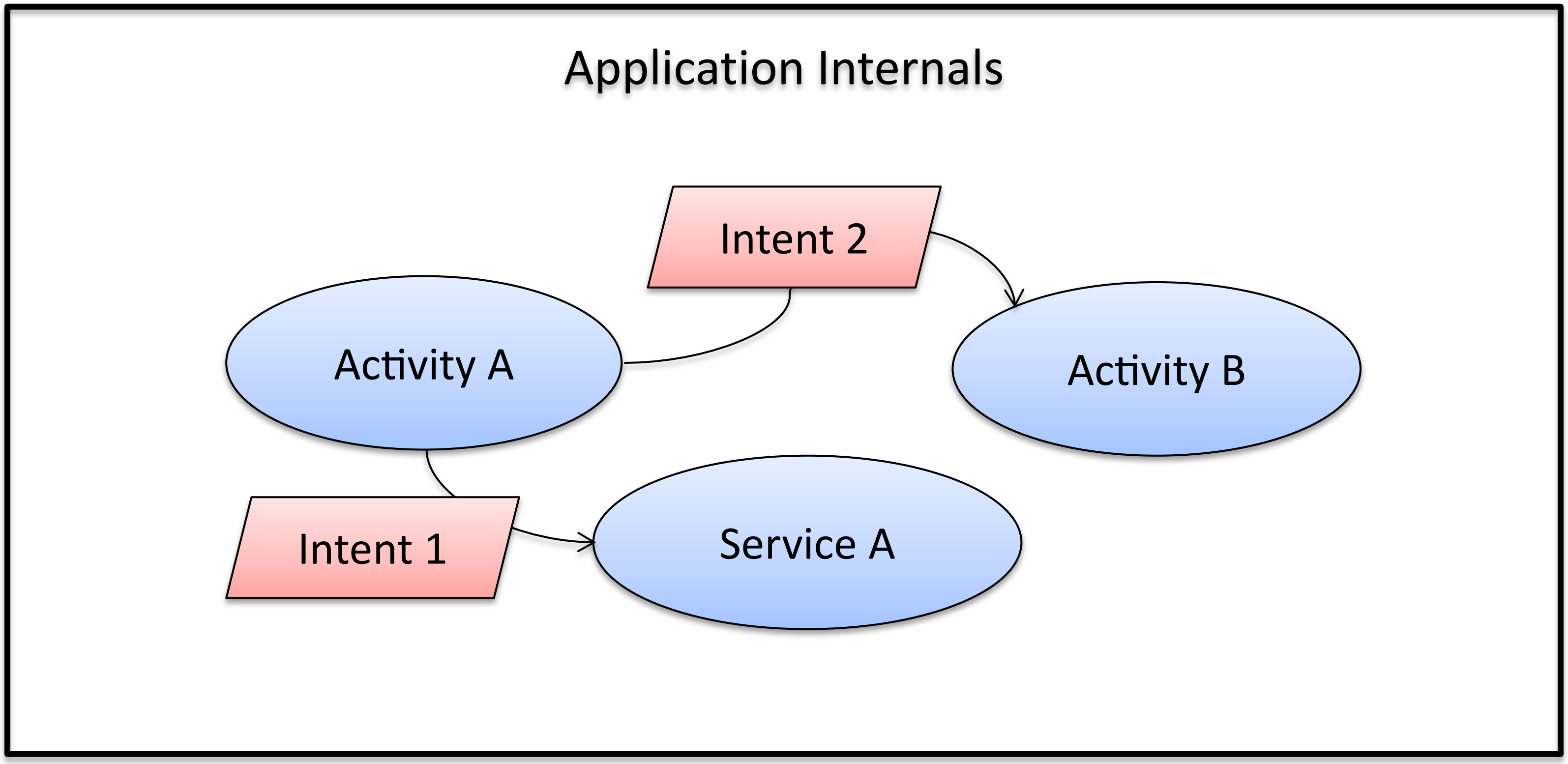


Intents: Connecting components

- Messages that enable communication across components



Android application



Components
Modules

Intents
Messages

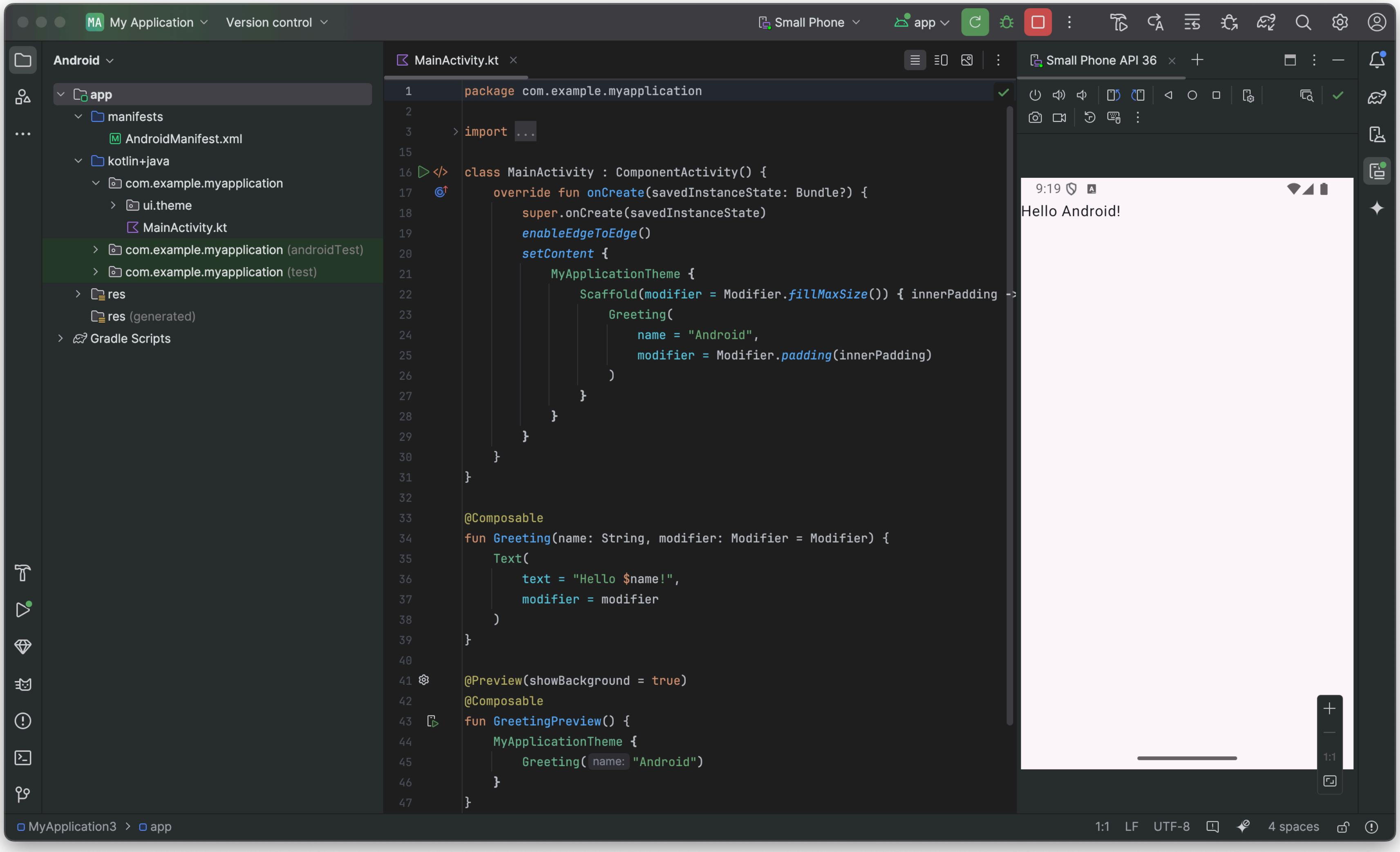
4. APPLICATION DEVELOPMENT

Android development tools

- Android SDK (Software Development Kit)
  - Core tools to create, compile, and package Android apps
  - Includes device emulator and AVD (Android Virtual Device) manager
  - ADB (Android Debug Bridge): connect, debug, and control devices

- Android Studio
  - Full-featured IDE based on IntelliJ IDEA
  - Smart code editing, real-time UI preview, and Compose support
  - Integrated emulator, profilers, layout inspector, and AVD creation tools

Android Studio



```kotlin
package com.example.myapplication

import ...

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding -
                    Greeting(
                        name = "Android",
                        modifier = Modifier.padding(innerPadding)
                    )
                }
            }
        }
    }
}

@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
    Text(
        text = "Hello $name!",
        modifier = modifier
    )
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    MyApplicationTheme {
        Greeting( name: "Android")
    }
}
```

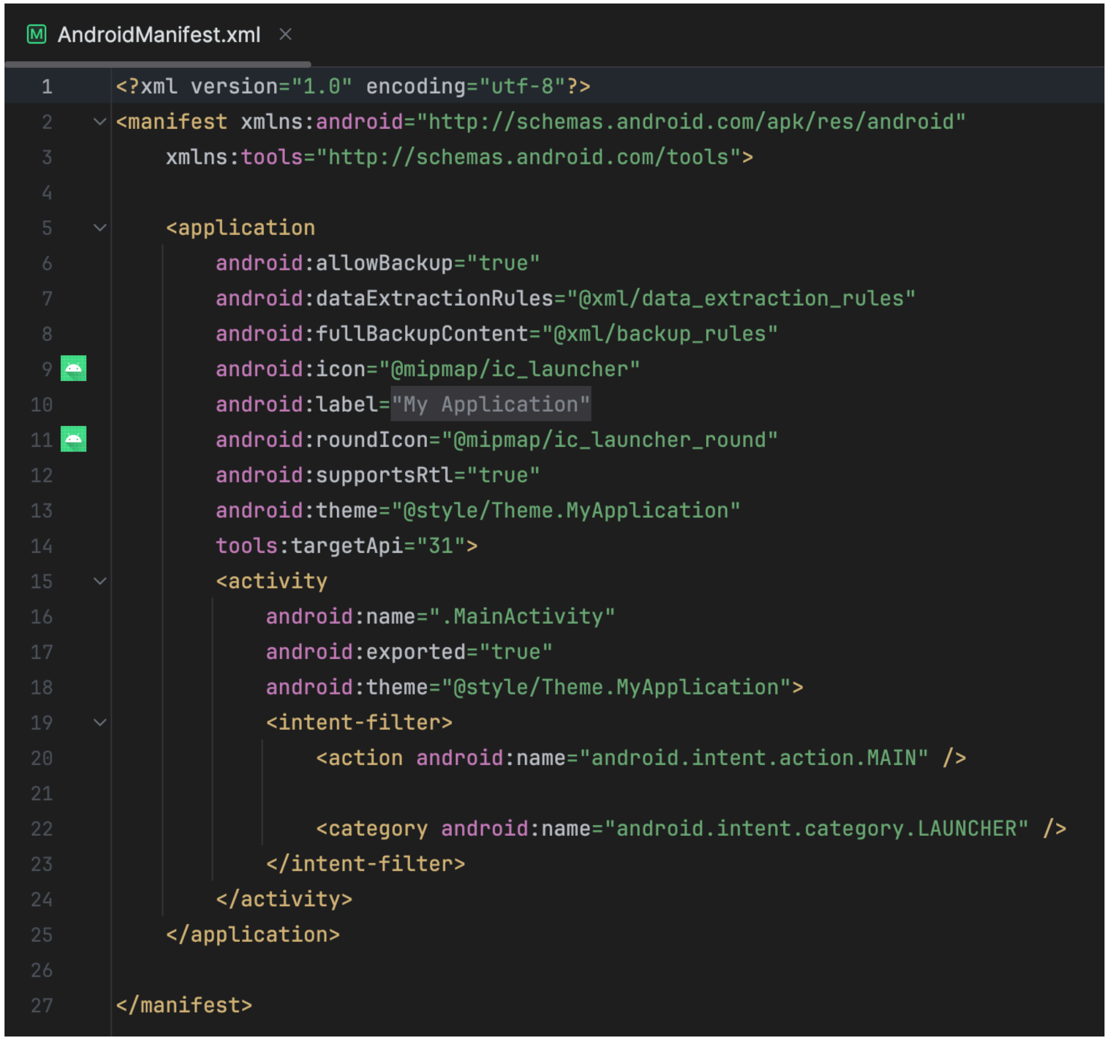
Android project structure

Your app = **Code + Resources + Manifest + Gradle scripts**

- **Project Root**
  - Main folder with app code and build files.
- app/
  - All app-specific files
- src/main/AndroidManifest.xml
  - Declares app components and permissions
- src/main/java/
  - Kotlin/Java source code
- src/main/res/
  - App resources: layouts, images, strings
- build.gradle.kts
  - Module build configuration
- settings.gradle.kts
  - Project module settings

Manifest

- Central configuration file for your app
- Things defined:
  - Components
  - Permissions
  - Intents and filters
  - App details



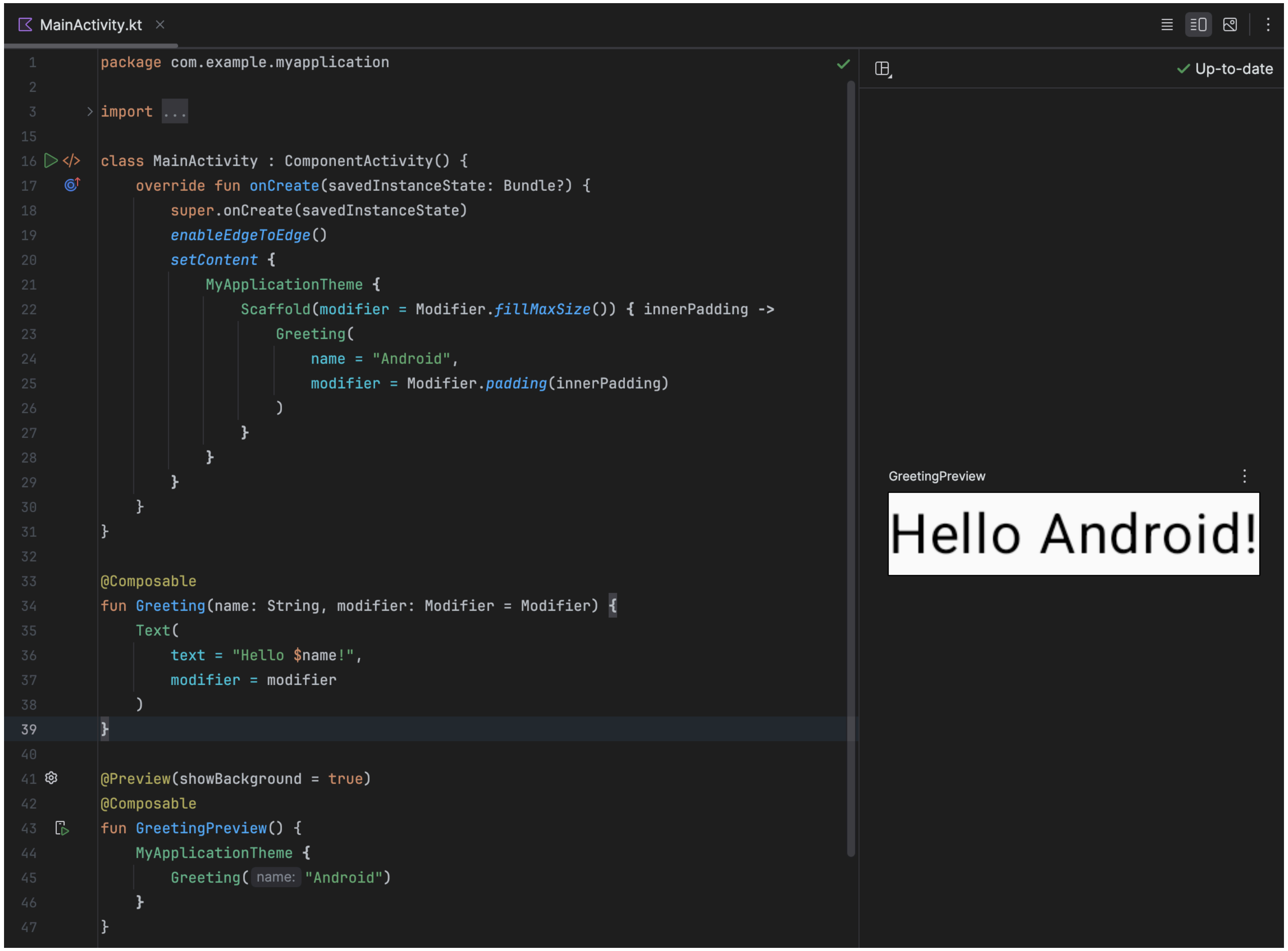
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:theme="@style/Theme.MyApplication">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Source code


MainActivity.kt

```kotlin
package com.example.myapplication

import ...

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            MyApplicationTheme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                    Greeting(
                        name = "Android",
                        modifier = Modifier.padding(innerPadding)
                    )
                }
            }
        }
    }
}

@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
    Text(
        text = "Hello $name!",
        modifier = modifier
    )
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    MyApplicationTheme {
        Greeting( name: "Android")
    }
}
```

From code to app: Android build process

- **1. Write Code and Resources**
  - Kotlin/Java source code in src/main/java/
  - Layouts, images, and strings in src/main/res/
- **2. Compile Source Code**
  - Kotlin/Java code → compiled into .class files
  - Kotlin compiler (kotlinc) or Java compiler (javac)
- **3. Convert to DEX Format**
  - .class files → .dex (Dalvik Executable) format
  - Tool: d8 compiler (part of Android build tools)
- **4. Package into APK or AAB**
  - Combine .dex files, resources, manifest, assets
  - Output: .apk (for testing) or .aab (for Play Store upload)
- **5. Deploy and Run**
  - Install APK on emulator or device using **ADB** (Android Debug Bridge)
  - App launches and runs inside Android Runtime (ART)

Useful Links

- Android Developer Guides
  - https://developer.android.com
- Jetpack Compose Documentation
  - https://developer.android.com/jetpack/compose
- Kotlin Language Reference
  - https://kotlinlang.org/docs
- Android Studio Download
  - https://developer.android.com/studio
- Material Design Guidelines
  - https://m3.material.io
- JetBrains Kotlin Courses
  - https://hyperskill.org/tracks/18