

# Introduzione ad Android Studio

Android Studio e struttura di un progetto Android

**Laboratorio 0**

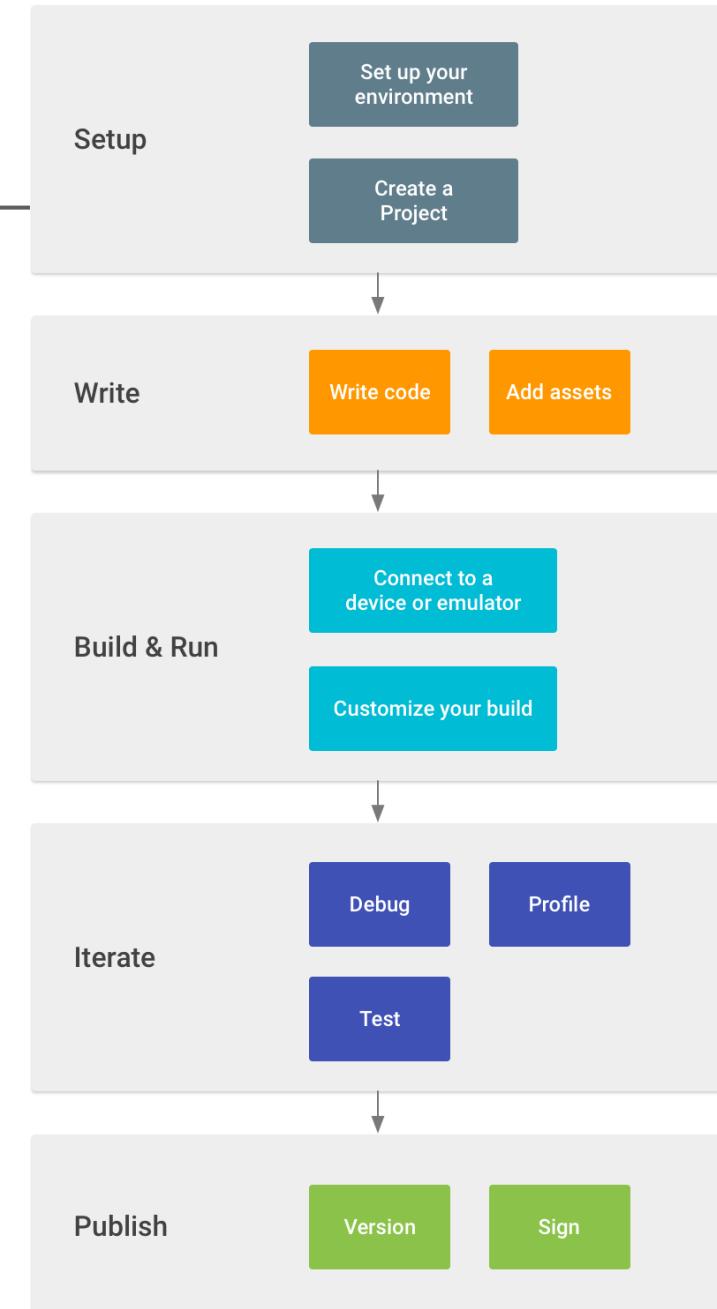
# Laboratorio

---

- Laboratorio 2.2: Parte Android
- Laboratorio 3.1: Parte iOS
- Per qualsiasi domanda potete scrivere a:
  - [catia.prandi2@unibo.it](mailto:catia.prandi2@unibo.it)
  - [gianni.tumedei2@unibo.it](mailto:gianni.tumedei2@unibo.it)

# Developer workflow

- **Setup**
  - Setup dell'ambiente di sviluppo e creazione di un progetto
- **Scrittura dell'app**
  - Android Studio include vari strumenti per lavorare in maniera più efficiente, scrivere codice di qualità, progettare un'interfaccia utente e creare risorse per diversi tipi di dispositivi
- **Build, run**
  - Durante questa fase, si crea il pacchetto APK debuggabile, che si può installare ed eseguire sull'emulatore o su un dispositivo Android dedicato
- **Debug, profile, test**
  - È una fase iterativa in cui si continua a scrivere l'app, concentrandosi però sul testing, sulla risoluzione dei bug e sull'ottimizzazione delle prestazioni.
- **Pubblicazione**



# Build your first app

---

## 1. Creare un progetto Android 😊

- Installare Android Studio
  - In laboratorio useremo la versione **Android Studio Hedgehog | 2023.1.1**
- E... basta! Android Studio si occuperà di installare tutto ciò che serve, inclusa l'SDK e un emulatore per un Andoid Virtual Device (in lab useremo un AVD con **Android 14**)



# SDK Platform release

## Android 14 (API level 34)

For details about the platform changes, see the [Android 14 documentation](#).

## Android 13 (API level 33)

For details about the platform changes, see the [Android 13 documentation](#).

## Android 12 (API levels 31, 32)

### 12L feature drop (API level 32)

For details about the platform changes, see the [12L documentation](#).

- Revision 1 (March 2022)

Released to the stable channel (no longer in preview).

### Android 12 (API level 31)

For details about the platform changes, see the [Android 12 documentation](#).

- Revision 1 (August 2021)

Released to the stable channel (no longer in preview) when [Android 12 reached the Platform Preview](#).

### Android 12 ATD system images

This Automated Test Device (ATD) image is an Android system image that is optimized for hardware tests. Early data indicates that tests that use this image should experience reduction in memory usage.

# Qualche info su Android Studio

android  
studio



- Android studio è l'IDE (Integrated Development Environment) ufficiale per sviluppare app Android
- È **multi-piattaforma** (ciò permette di sviluppare senza vincoli di sistema operativo)
  - Disponibile per sistemi operativi Windows, macOS e Linux
- È basato sull'IDE IntelliJ IDEA di JetBrains, che è lo standard di fatto per lo sviluppo con Kotlin, ed arricchito con funzionalità specifiche per la creazione di app Android
- Annunciato nel 2013, prima release nel luglio 2014, ha sostituito Eclipse come IDE ufficiale
- L'ultima major version è **Android Studio Iguana | 2023.2.1**

# Android Studio

---

- Mette a disposizione
  - Un build system basato su Gradle
  - Un emulatore veloce e ricco di funzionalità
  - Un ambiente unificato in cui è possibile sviluppare per tutti i dispositivi Android
  - Un sistema in grado di applicare modifiche al codice e alle risorse dell'app in esecuzione senza doverla riavviare
  - Funzionalità di templating per semplificare l'importazione di codice di esempio
  - Integrazione con Git e GitHub per il controllo di versione
  - Strumenti e framework per il testing
  - Strumenti automatici per monitorare prestazioni, grado di usabilità, compatibilità delle versioni e altri problemi
  - Supporto a C ++ e NDK (*Native Development Kit*)
  - Supporto integrato per Google Cloud Platform
  - ...

# Version name

---

- Ad ogni major version di Android Studio viene assegnato il nome di un animale, in ordine alfabetico dalla A alla Z:
  - Arctic Fox
  - Bumblebee
  - Chipmunk
  - Dolphin
  - ...
  - Hedgehog

# Nota sulla versione

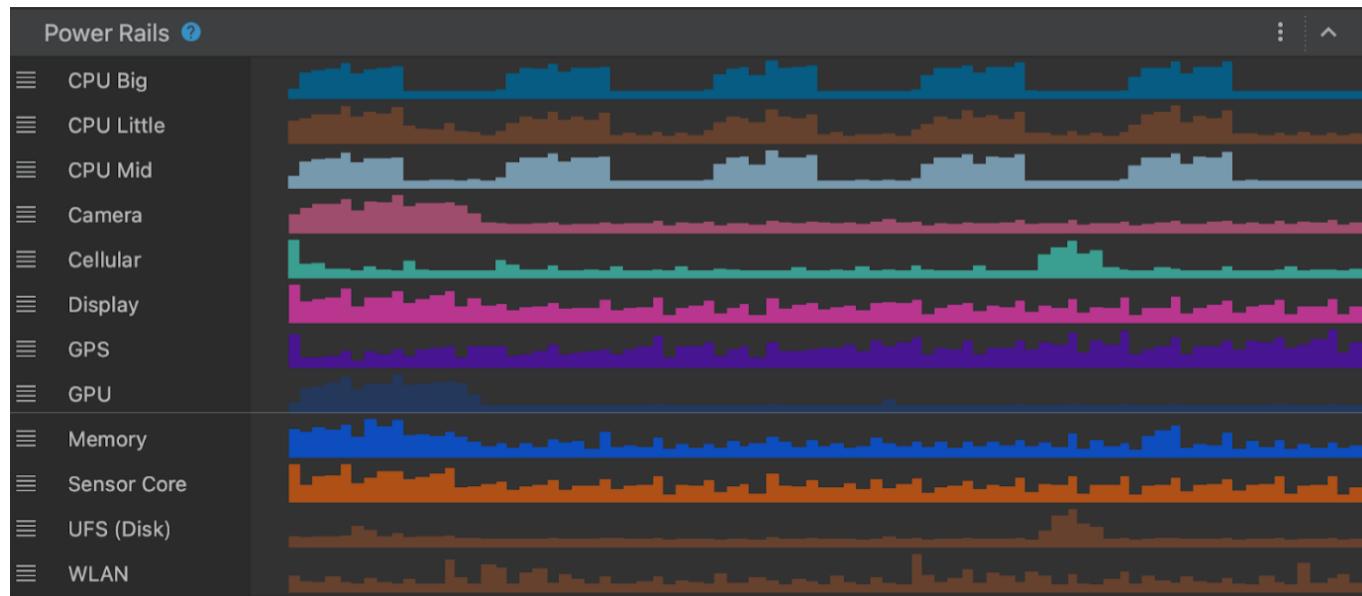
---

- La versione che utilizzeremo in laboratorio è **Hedgehog 2023.1.1 Patch 1**, rilasciata a Gennaio 2024
- Tutte le novità sono descritte qui:  
<https://developer.android.com/studio/releases>
- È appena uscita (29/02) la stable di Iguana | 2023.2.1
- Esiste già la canary di Jellyfish | 2023.3.1

Android Studio  
Viene migliorato ed  
arricchito continuamente!

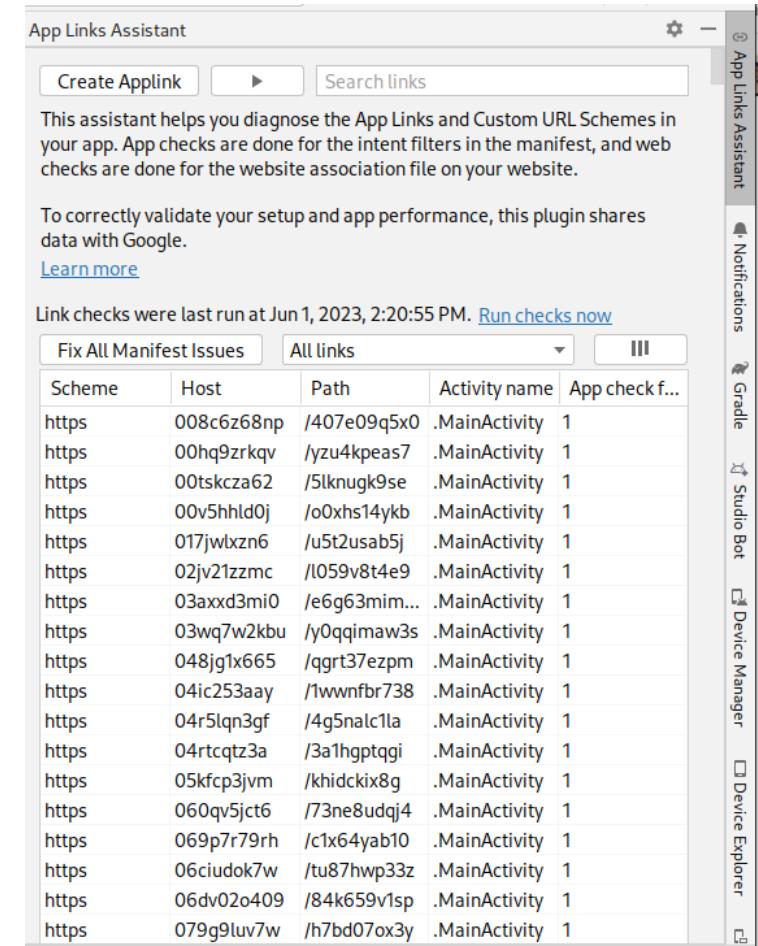
# Ultime feature – Power profiler

- Grazie al Power Profiler, è possibile correlare il consumo energetico di un dispositivo collegato alle azioni eseguite all'interno di un'app.



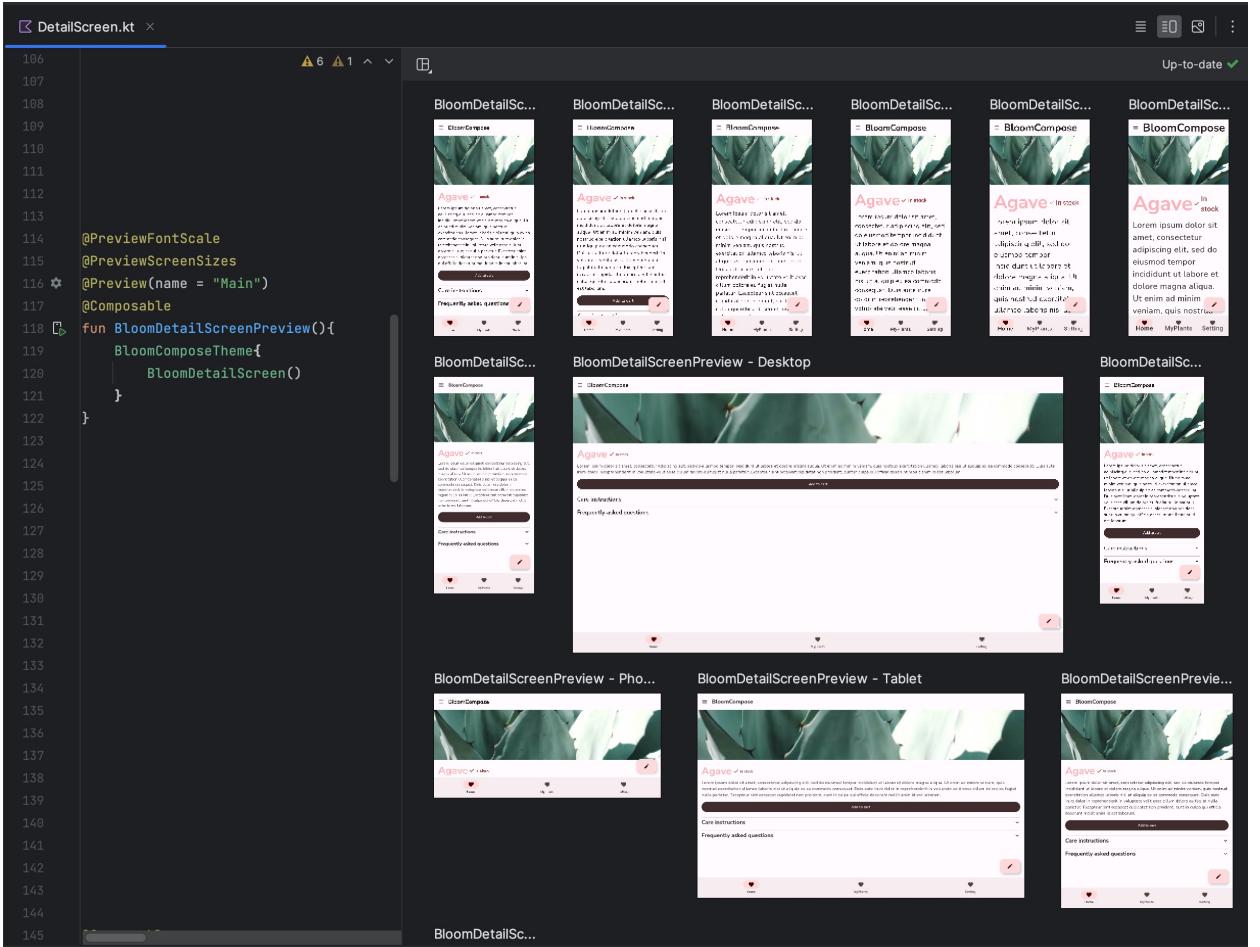
# Ultime feature – App Links Assistant

- L'App Link Assistant fornisce un'overview dei deep link creati all'interno di un'app, controllando la correttezza di ciascuno e fornendo la possibilità di risolvere eventuali errori di configurazione.



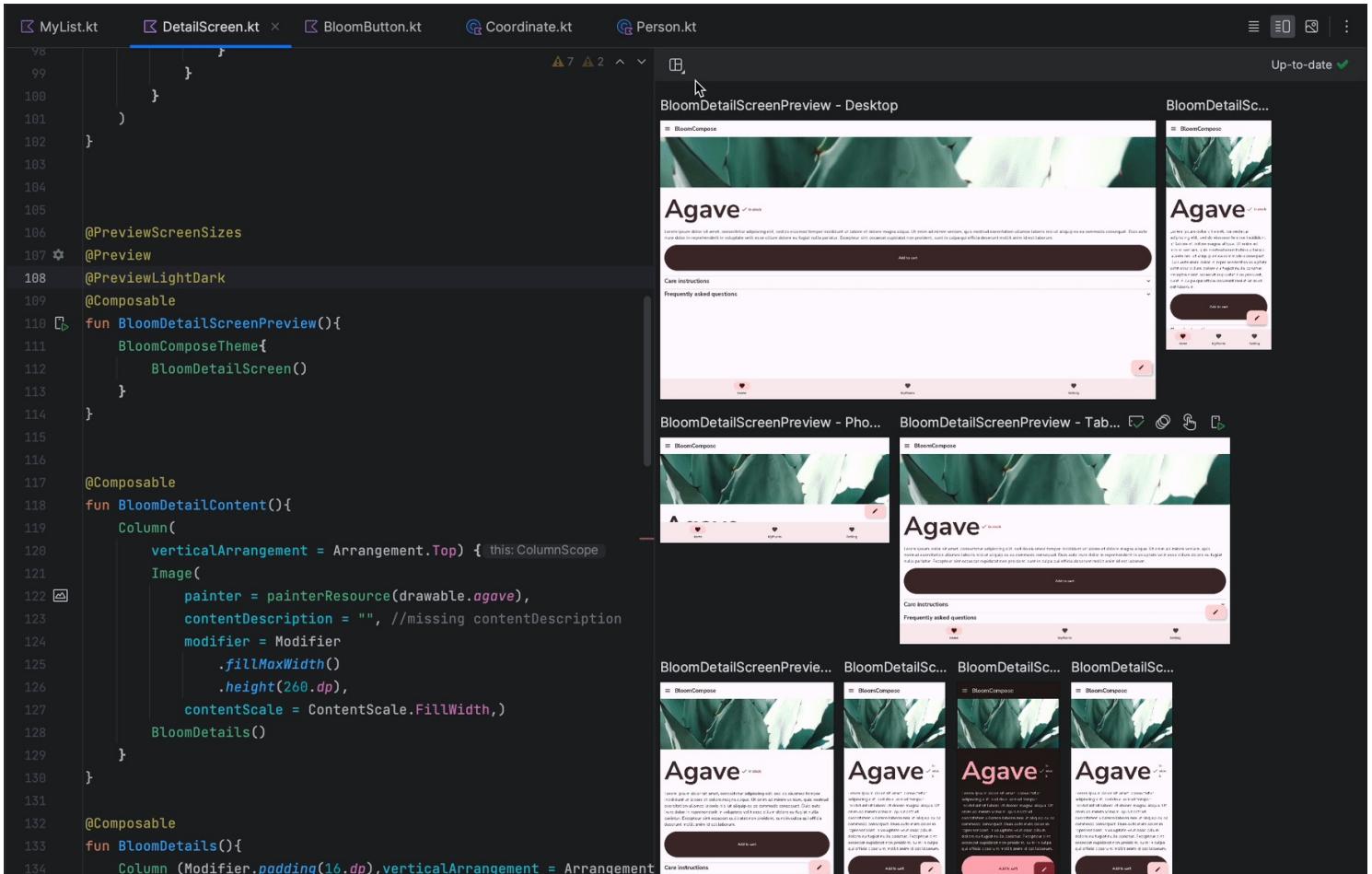
# Ultimate feature – Compose Multipreview templates

- La Multipreview API fornisce vari decoratori con cui impostare preview multiple per scenari comuni (varie risoluzioni, tema chiaro e scuro, ...).



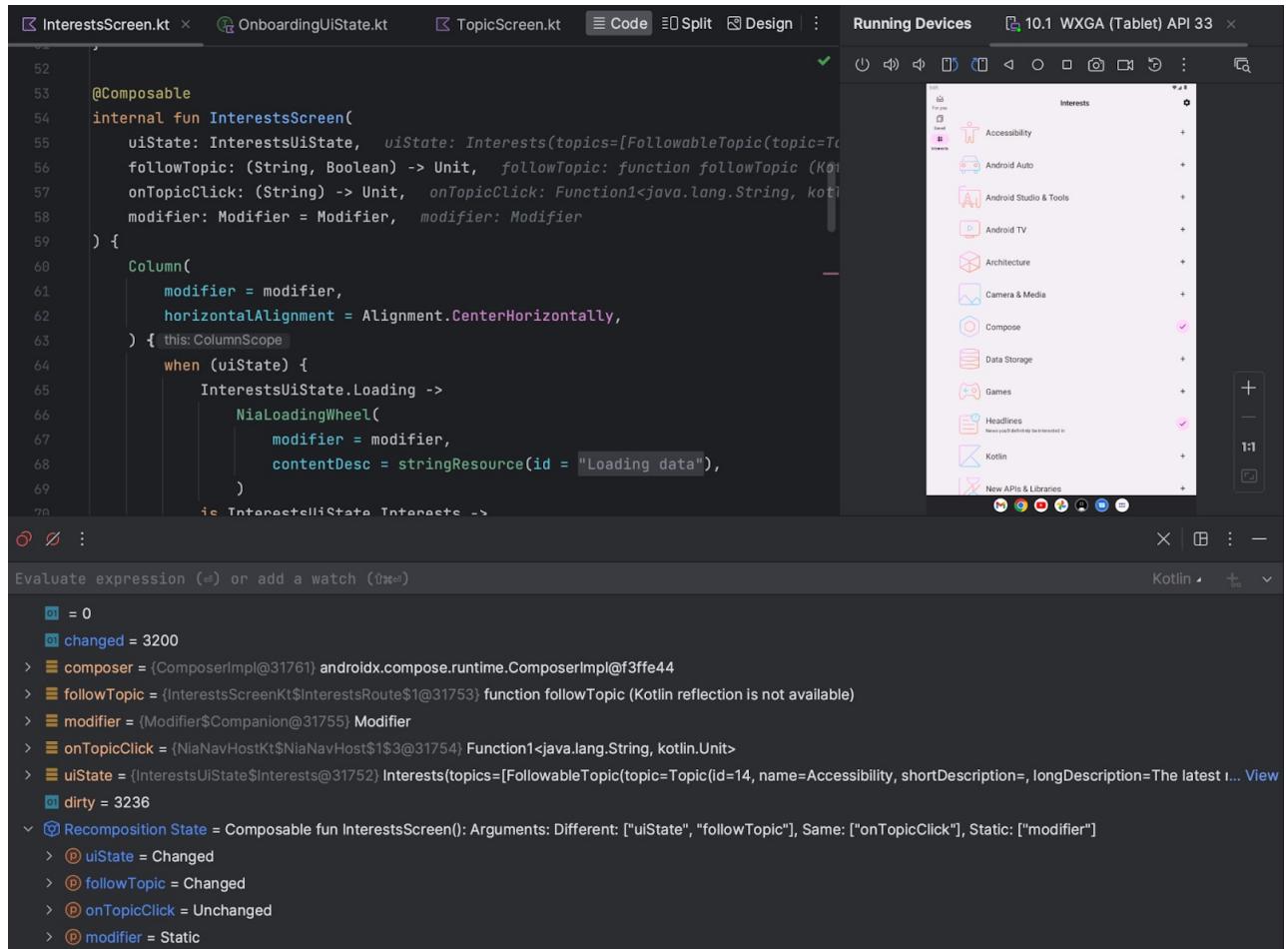
# Ultime feature – Compose Preview Gallery Mode

- La Gallery Mode fornisce un’interfaccia a tab per le multipreview, in modo da risparmiare risorse ed avere una visualizzazione semplificata delle preview multiple



# Ultime feature – State information in debugger

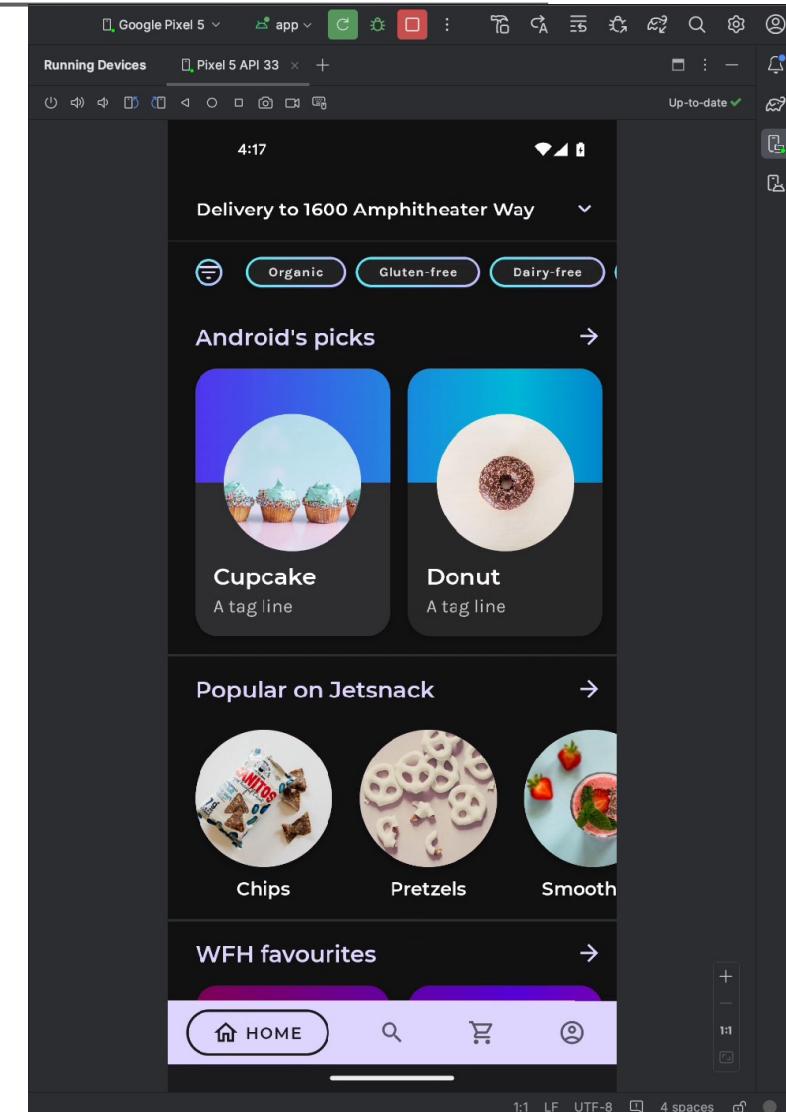
- Quando si imposta un breakpoint, il debugger è ora in grado di segnalare le variabili il cui valore è cambiato di recente, in modo da semplificare la ricerca di bug.



The screenshot shows the Android Studio interface during debugging. The top navigation bar includes tabs for 'InterestsScreen.kt', 'OnboardingUIState.kt', 'TopicScreen.kt', and 'Code'. Below the tabs, the code for the `InterestsScreen` is visible. A breakpoint is set on line 59. The bottom part of the screen displays the 'Evaluate expression' tool window, which lists variables and their current values. The variable `uiState` is highlighted in blue, indicating it has been modified since the last recomposition. Other listed variables include `changed`, `dirty`, `composer`, `followTopic`, `modifier`, `onTopicClick`, and `uiState`. To the right of the code editor, the 'Running Devices' panel shows a connected tablet running API 33. The bottom right corner of the interface shows the Kotlin logo.

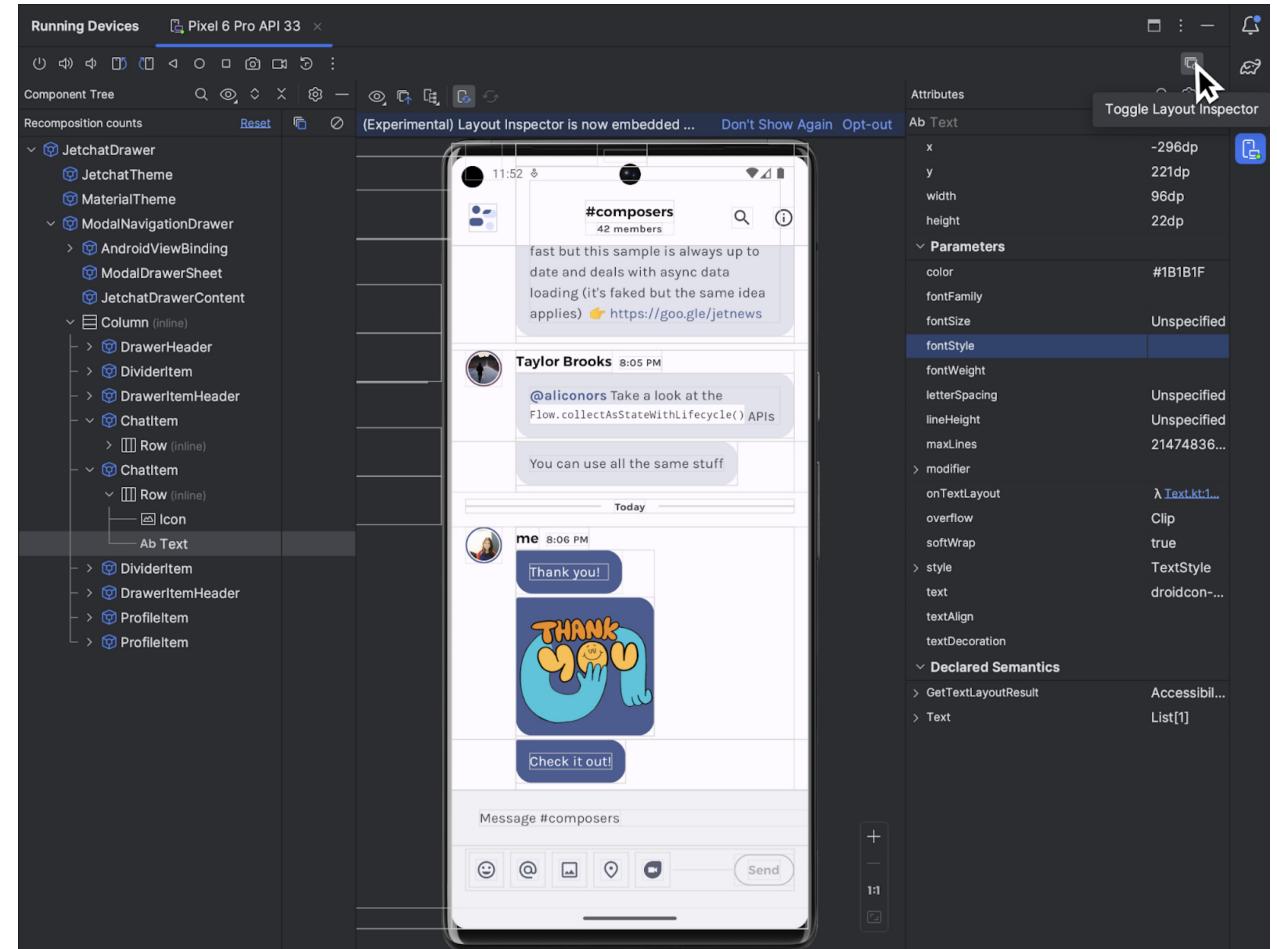
# Ultime feature – Device mirroring

- È ora possibile eseguire il mirroring di un dispositivo collegato con USB o wireless debugging, e controllarlo direttamente dall'interfaccia di Android Studio.

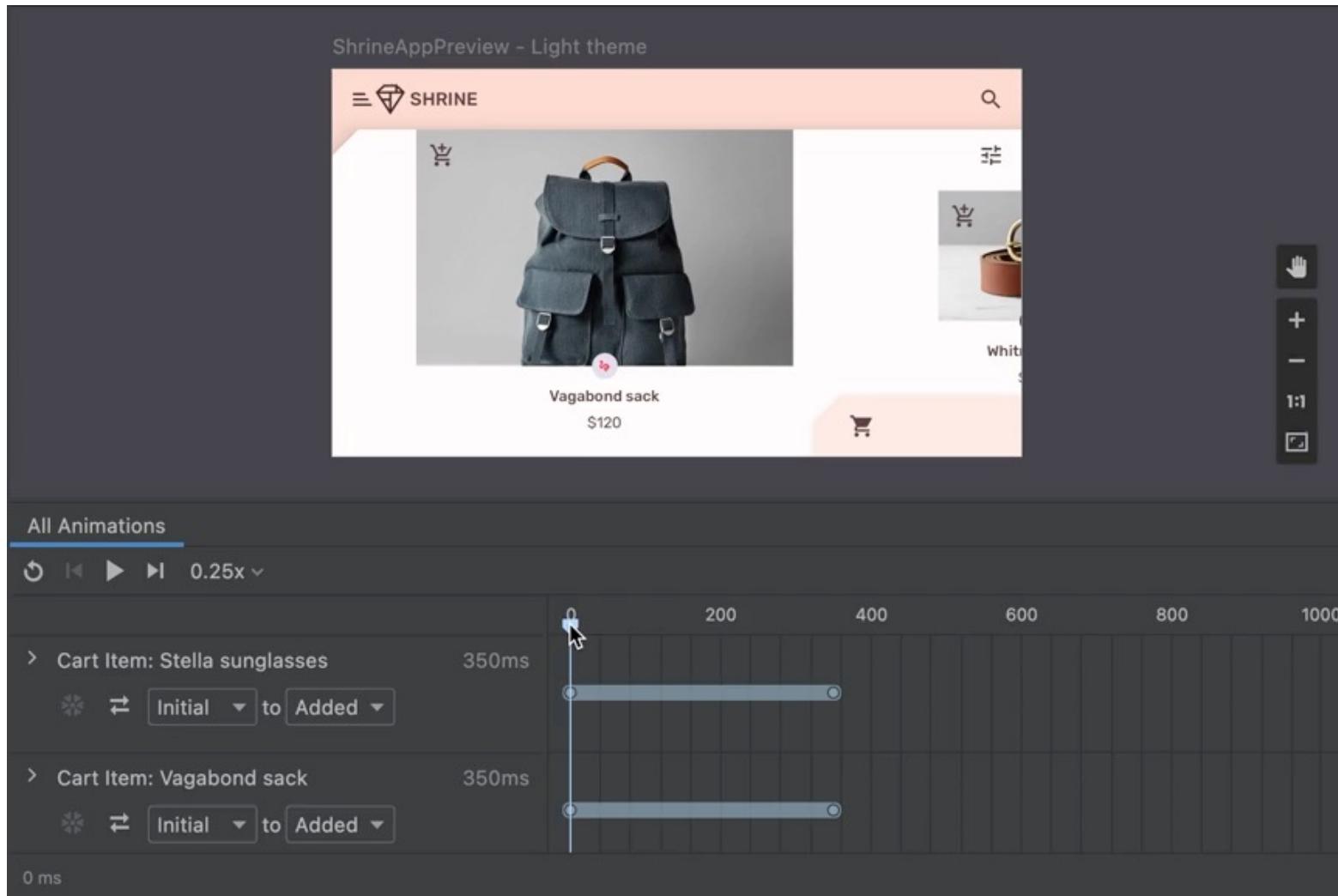


# Ultime feature – Embedded Layout Inspector

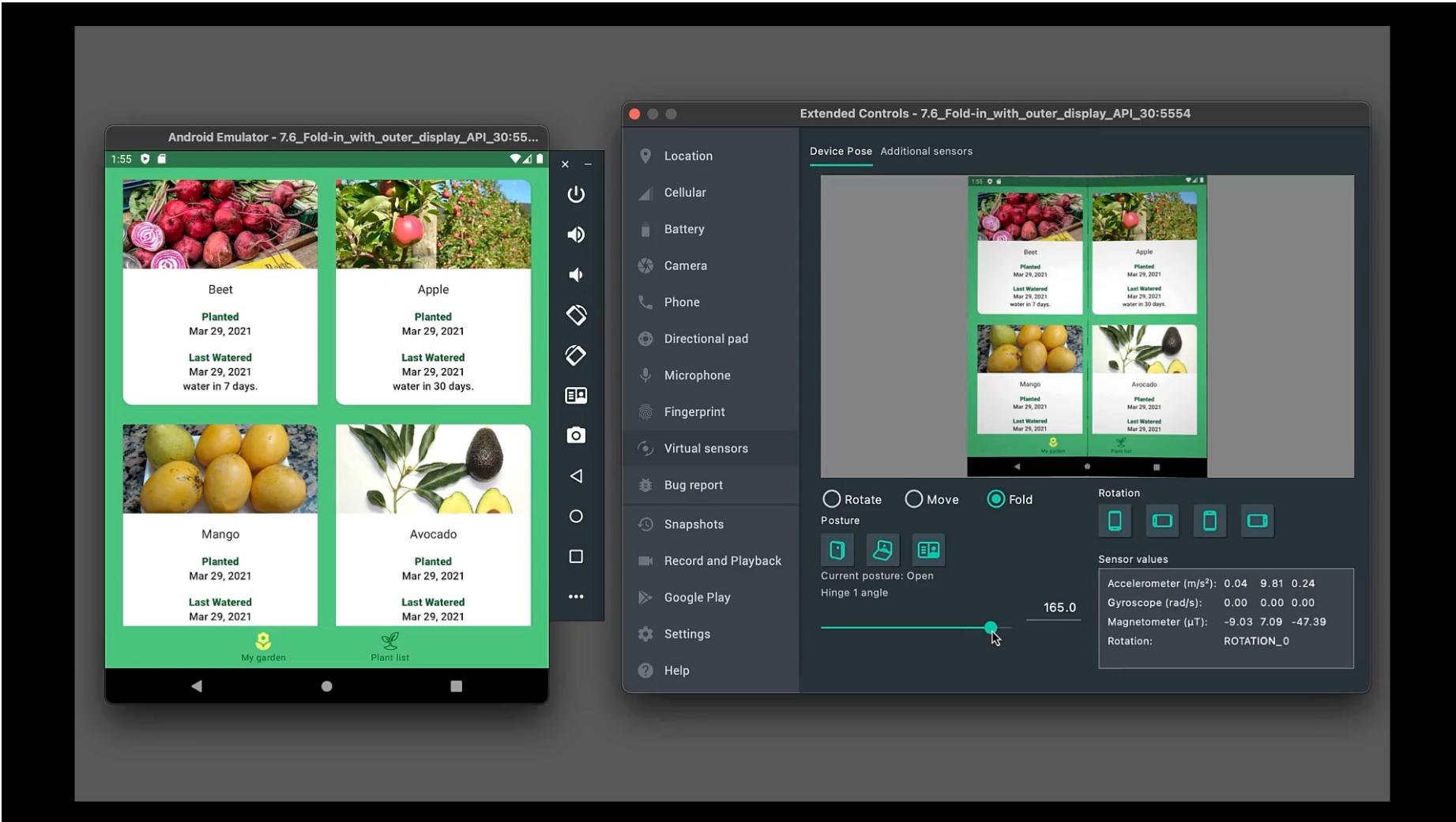
- Il layout inspector permette, in modalità sperimentale, di ispezionare il layout e i componenti di un'app direttamente dalla finestra Running Devices.



# Design with Compose



# Easily emulate any device



# Installazione

---

- Le slide e le esercitazioni di laboratorio sono fatte sulla versione **2023.1.1 Patch 1** di Android Studio
    - La stessa attualmente installata nei PC di laboratorio
  - Nel vostro portatile/PC personale, installate Android Studio (preferibilmente 2023.1.1 Patch 1) e il sistema si occuperà di scaricare tutto il necessario
    - Download: <https://developer.android.com/studio/archive> - versione 2023.1.1 Patch 1
    - Poi seguite il tutorial che trovate qui:  
<https://developer.android.com/studio/install> per tutti i sistemi operativi supportati
- MA è davvero molto semplice!**

# android studio

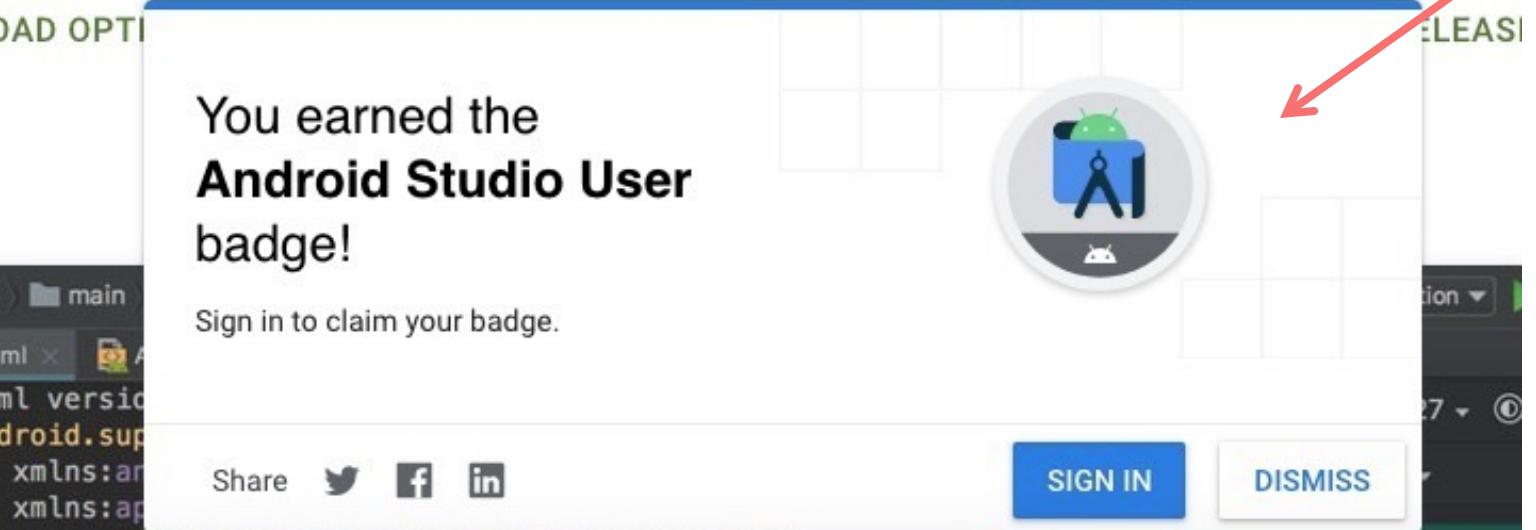


Android Studio provides the fastest tools for building apps on every type of Android de

DOWNLOAD ANDROID STUDIO

4.1.2 for Mac (877 MiB)

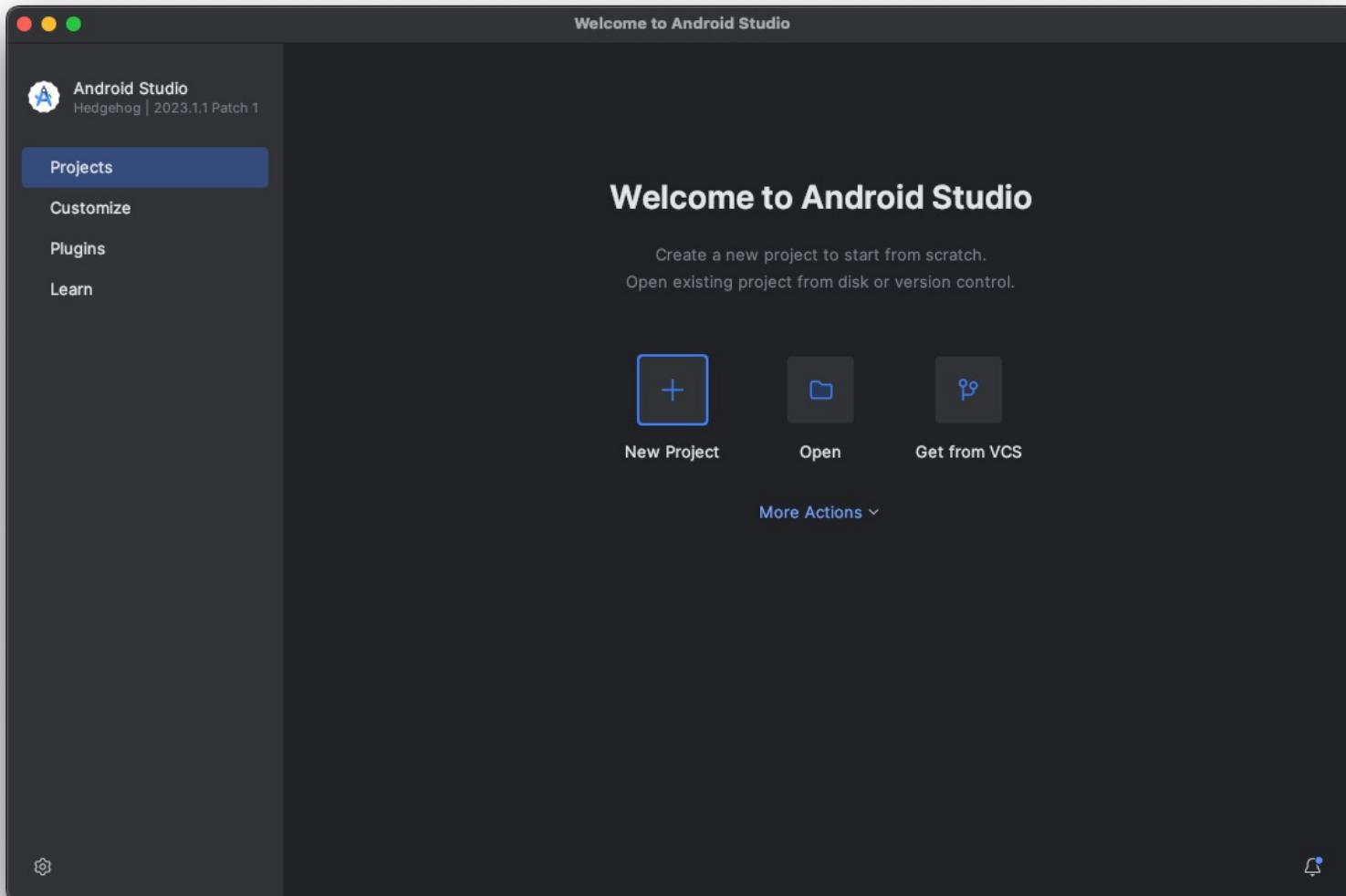
GAMIFICATION!



# Android Studio Hedgehog

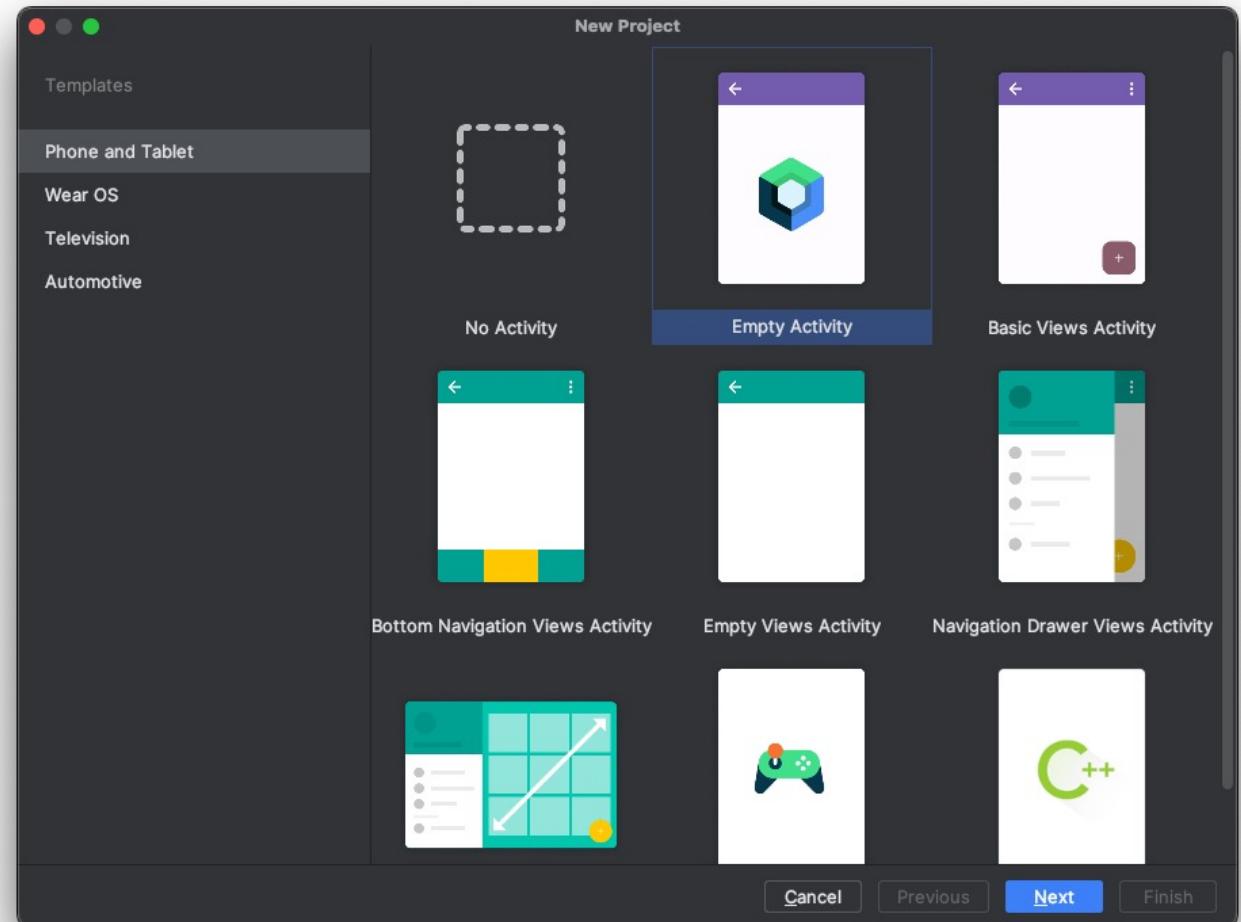


# Android Studio – Start a new Project

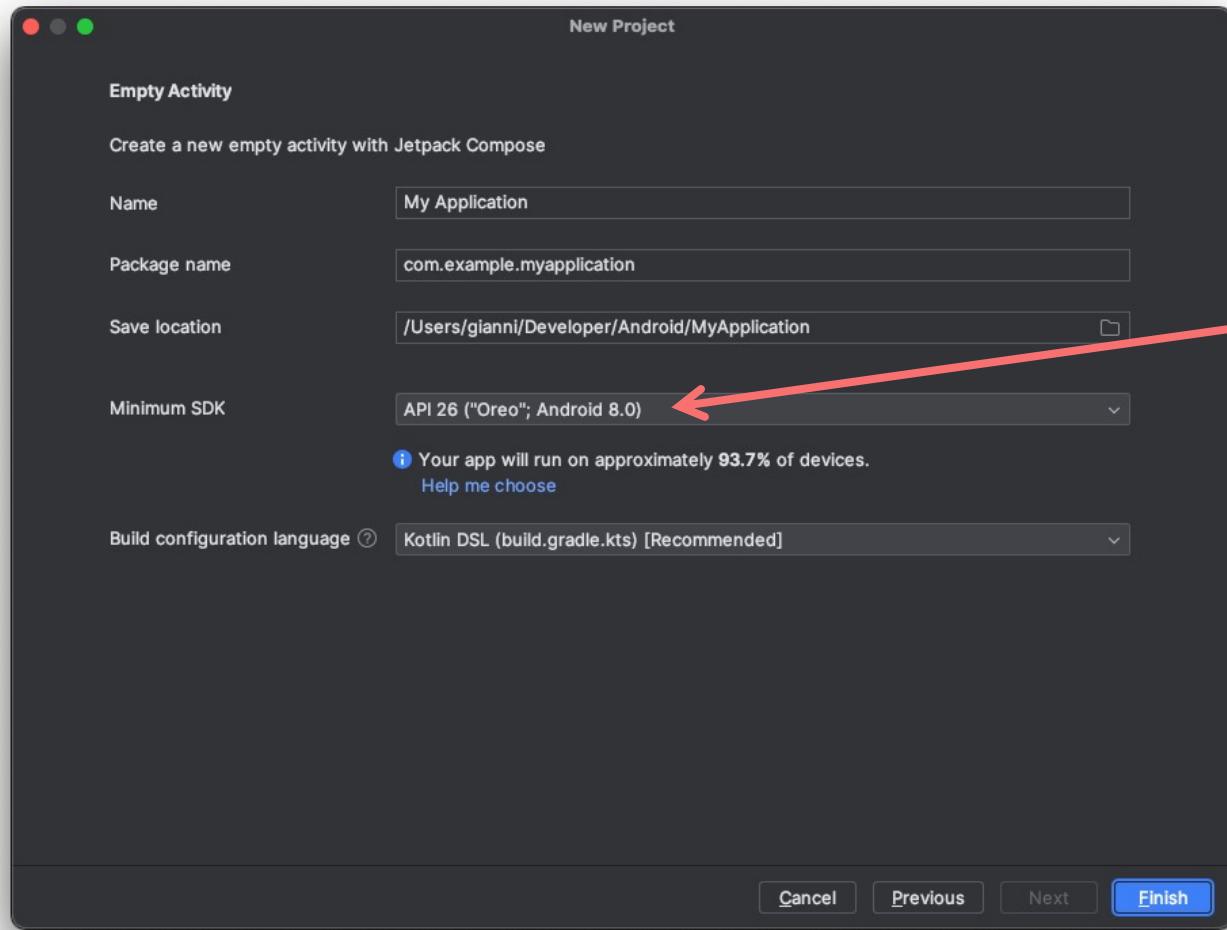


# Android Studio – Start a new Project

- A meno di casi particolari, è sempre consigliabile partire da una **Empty Activity** per avere un maggiore controllo sull'applicazione

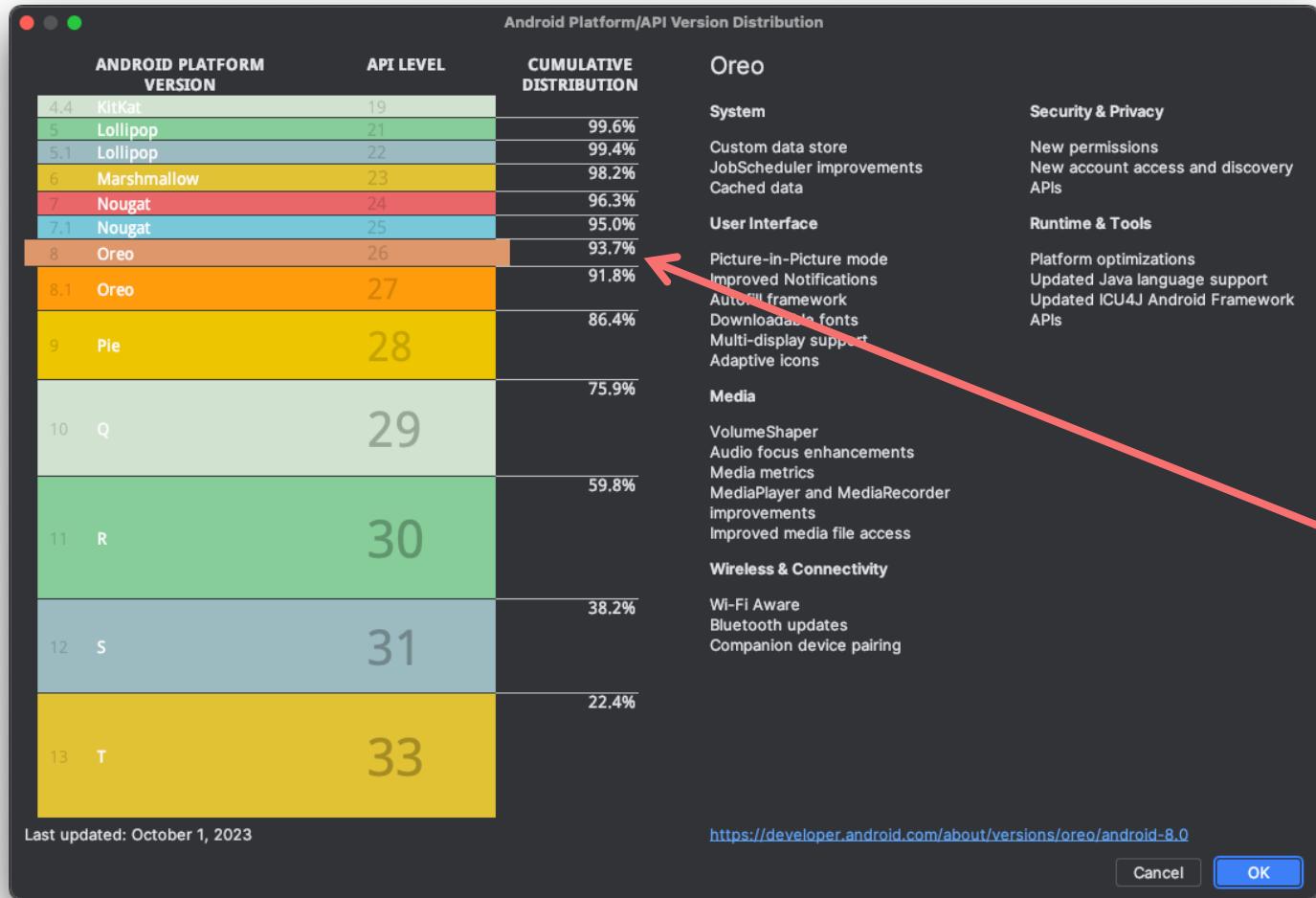


# Android Studio – Start a new Project



Utilizzeremo Android 8  
come SDK minima

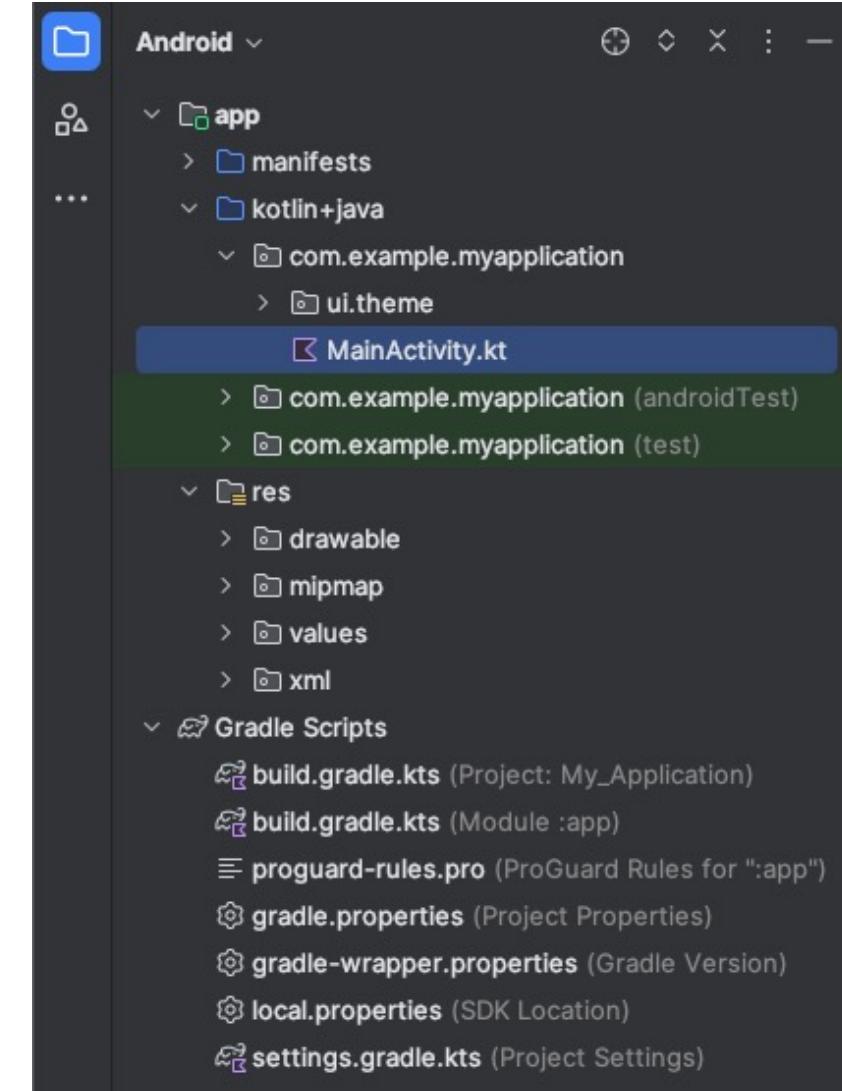
# Android Studio – Start a new Project



Percenutale di device compatibili in base alla min SDK scelta

# Progetto Android Studio

- Un progetto Android Studio definisce il workspace di un'app.
- Contiene tutto il codice sorgente e quello di tesò, l'elenco delle dipendenze, la configurazione della build, e gli asset utilizzati dall'app.
- Con la creazione di un progetto, Android Studio crea la struttura di file e cartelle necessaria (visibile sulla finestra di sinistra dell'IDE)



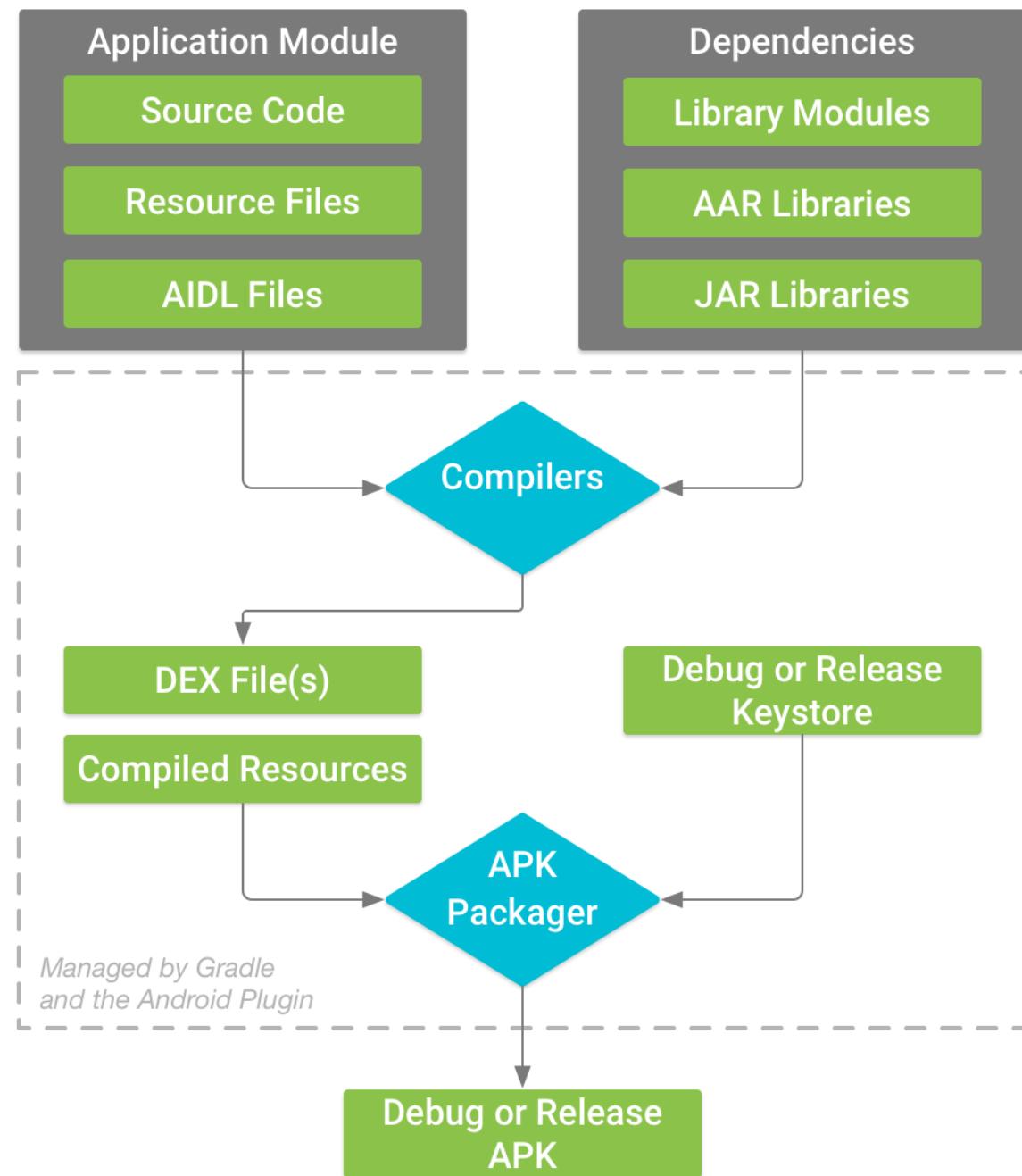
# Due parole su Gradle



- Android Studio utilizza **Gradle** come build system e gestore delle dipendenze
  - Con ulteriori funzionalità specifiche per Android fornite dal plug-in Android per Gradle
- Questo build tool, benché controllabile da terminale, è integrato direttamente nell'interfaccia di Android Studio
- Ma che cos'è? In due parole:
  - è un sistema open source per l'*automazione dello sviluppo*, ispirato a Apache Ant e Apache Maven, che permette di definire la configurazione del progetto tramite domain-specific language (DSL) dichiarativo basato su **Kotlin**, al posto dei template XML usati da Apache Maven

# Processo di build di un'app

1. Il **compilatore** converte il codice sorgente in vari file **DEX** (Dalvik EXecutable), che includono il **bytecode** da eseguire sui dispositivi Android, e in **risorse** compilate.
2. L'**APK Packager** combina i file DEX e le risorse compilate in un singolo file **APK**. Prima che l'applicazione possa essere distribuita e installata su un dispositivo Android, tuttavia, l'APK deve essere firmato.
3. L'APK Packager **firma** l'APK utilizzando il **keystore** di debug o di release:
  - Se l'app è una versione di debug, cioè un'applicazione da utilizzare solo per i test, il packager la firma applicazione con il keystore di debug. Android Studio configura automaticamente i nuovi progetti con un keystore di debug.
  - Se l'app è una versione di release che va pubblicata negli store, il packager firma l'applicazione con il keystore di release.
4. Prima di generare l'APK finale, il packager ottimizza l'applicazione in modo che **utilizzi meno memoria** quando viene eseguita su un dispositivo.
5. Al termine del processo di build, si dispone di un APK di debug o release della propria applicazione.



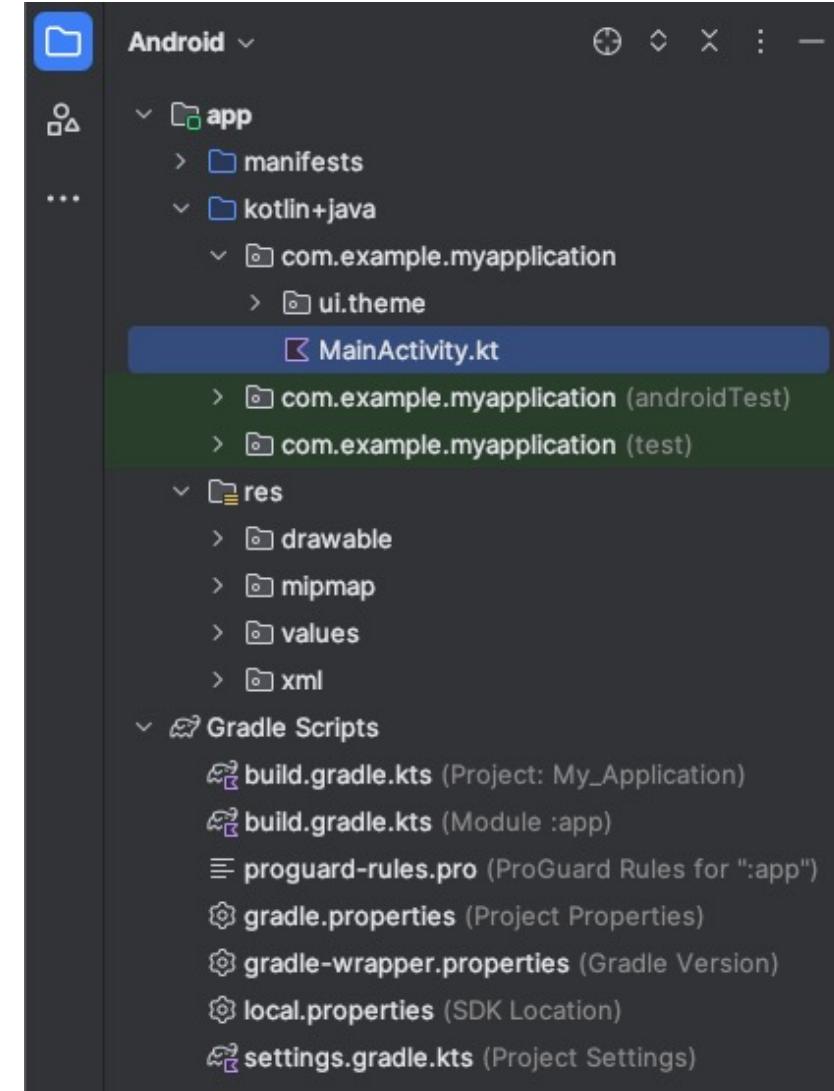
# Struttura di un progetto Gradle



- Un progetto Gradle può essere organizzato in vari moduli e viene gestito principalmente attraverso due tipologie di file:
  - **settings.gradle.kts**: identifica una directory come root di un progetto Gradle e permette di configurare aspetti comuni all'intero progetto, come il nome e l'elenco delle repository da cui scaricare le dipendenze.
  - **build.gradle.kts**: identifica una directory come root di un modulo e ne configura aspetti come: l'elenco delle dipendenze, le opzioni in fase di compilazione e, nel caso di progetti Android, l'SDK minima e target. È inoltre possibile creare un file **build.gradle.kts** nella root del progetto per raggruppare configurazioni comuni a tutti i moduli

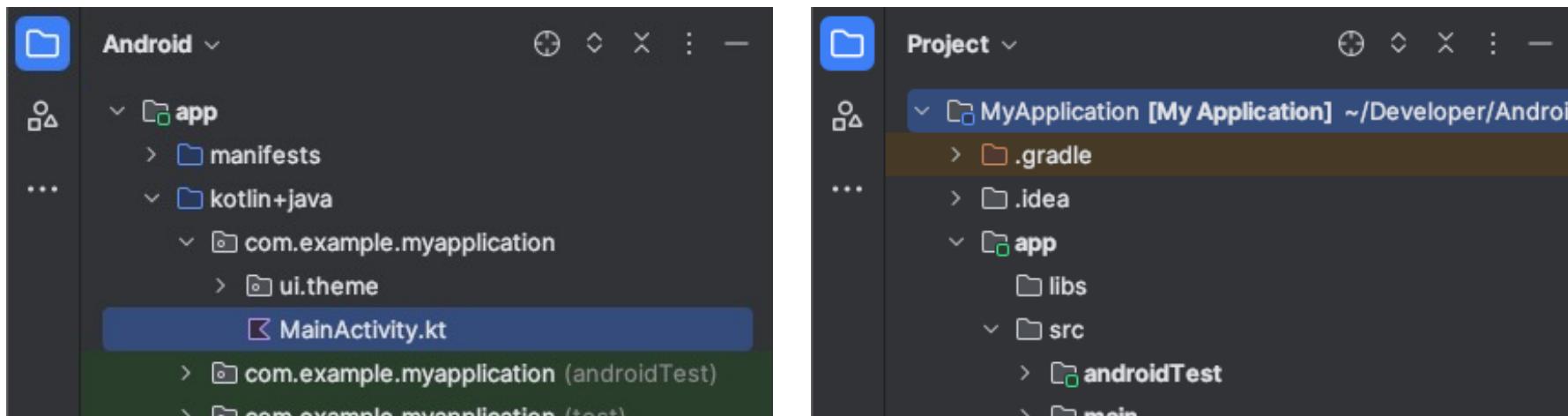
# Struttura di un progetto Android

- Cartella **app**
  - Modulo di default
- Cartella **manifests**
  - Include in file **AndroidManifest.xml**
- Cartella **java / kotlin+java**
  - Include tutto il codice Kotlin e/o Java
- Cartella **res**
  - Include varie risorse come immagini, colori, valori, perlopiù in formato XML
- Cartelle **java / kotlin+java o res generated**
  - Contengono file autogenerati che NON vanno modificati dallo sviluppatore



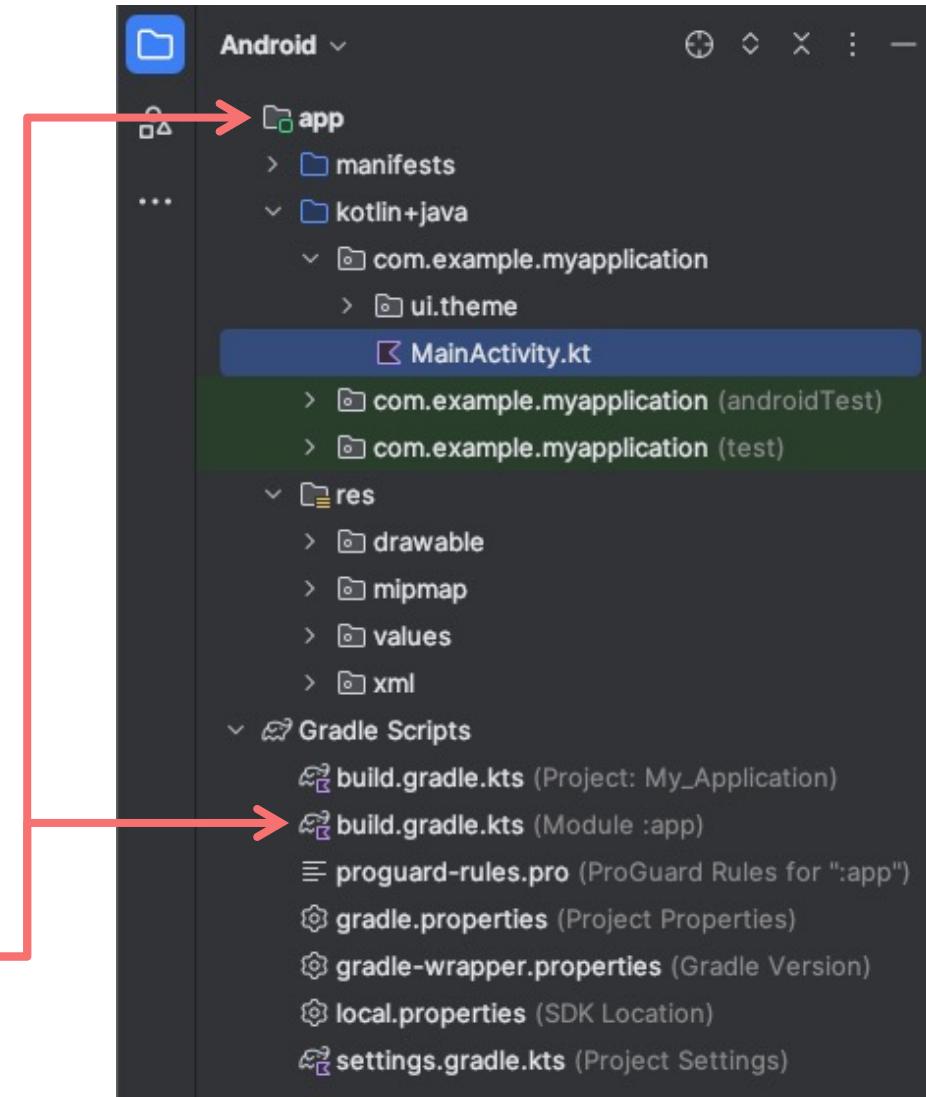
# View di Android Studio

- Attenzione! La struttura del progetto mostrata di default nella sidebar di Android Studio non è quella effettiva salvata nel file system, ma una versione semplificata chiamata **Android view**.
- È possibile visualizzare la struttura di file e cartelle tramite le view **Project** o **Project Files**



# Approfondimento: Moduli

- Sono un costrutto di Gradle che permette di suddividere un progetto in vari blocchi (moduli) autonomi, in modo da semplificarne lo sviluppo e la manutenzione.
- Ogni modulo di un'app Android è configurabile tramite un file **build.gradle.kts**, e può contenere codice sorgente, risorse, file manifest e altro.
- Quando inizializza un nuovo progetto, Android studio crea automaticamente un modulo predefinito di nome **app**.

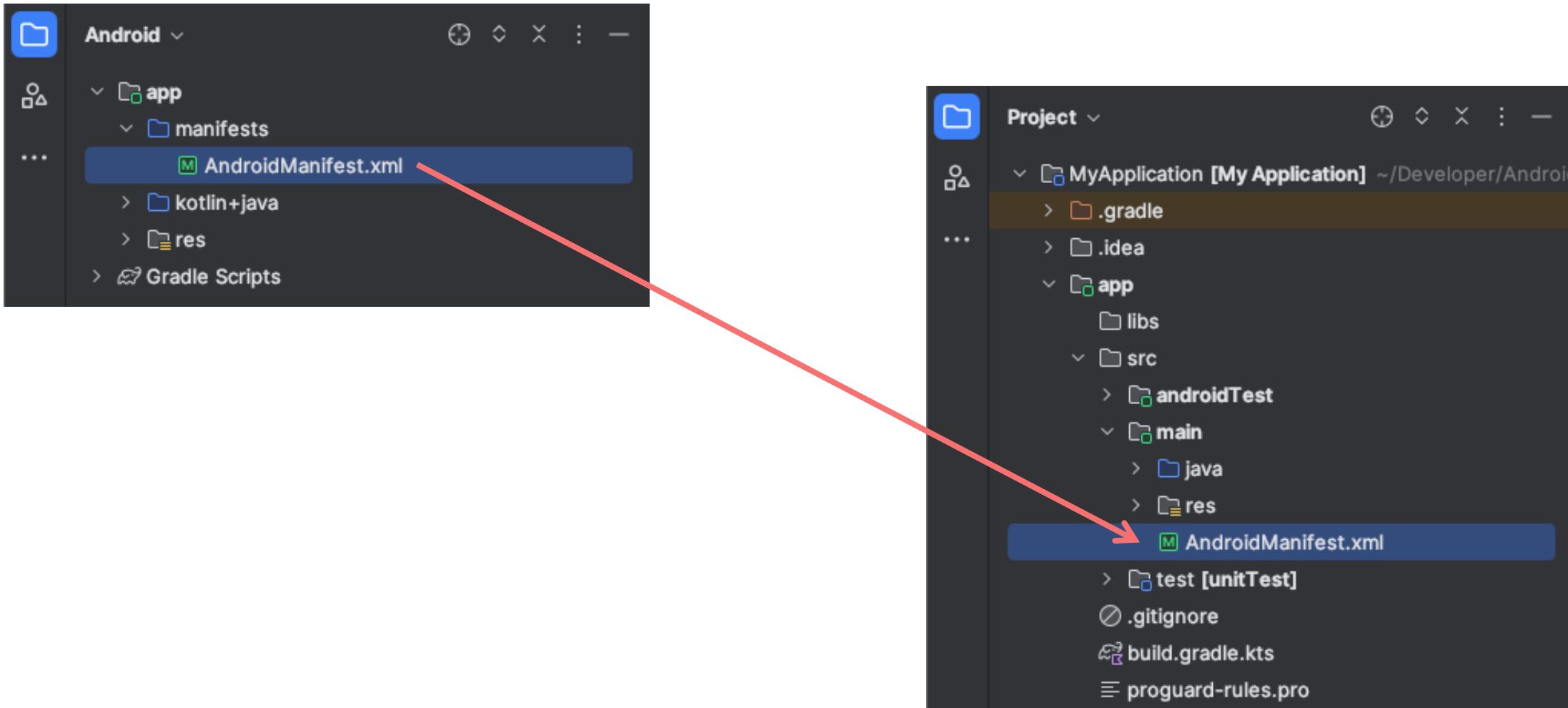


# Esempi di suddivisione in moduli

---

- Alcune tipologie di moduli in cui può essere suddiviso un progetto:
  - **App**, ad esempio: Phone & Tablet Module, Wear OS Module, Android TV Module, Glass Module.
  - **Library**: raggruppano codice riutilizzabile, che può essere usato come dipendenza in altri moduli dell'app o importato in altri progetti.
  - **Google App Engine e Google Cloud**: codice di Google Cloud che fa da backend per le app del progetto.

# AndroidManifest.xml - location



# Manifest

Componente Activity

Intent filter

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:tools="http://schemas.android.com/tools">
4
5     <application
6         android:allowBackup="true"
7         android:dataExtractionRules="@xml/data_extraction_rules"
8         android:fullBackupContent="@xml/backup_rules"
9         android:icon="@mipmap/ic_launcher"
10        android:label="My Application"
11        android:roundIcon="@mipmap/ic_launcher_round"
12        android:supportsRtl="true"
13        android:theme="@style/Theme.MyApplication"
14        tools:targetApi="31">
15
16         <activity
17             android:name=".MainActivity"
18             android:exported="true"
19             android:label="My Application"
20             android:theme="@style/Theme.MyApplication">
21             <intent-filter>
22                 <action android:name="android.intent.action.MAIN" />
23
24                 <category android:name="android.intent.category.LAUNCHER" />
25             </intent-filter>
26         </activity>
27     </application>
28 </manifest>
```

# Manifest

Gli attributi `allowBackup`, `dataExtractionRules` e `fullBackupContent` permettono di impostare le regole per il backup automatico dei dati dell'utente dell'applicazione.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:tools="http://schemas.android.com/tools">
4
5     <application
6         android:allowBackup="true"
7         android:dataExtractionRules="@xml/data_extraction_rules"
8         android:fullBackupContent="@xml/backup_rules"
9         android:icon="@mipmap/ic_launcher"
10        android:label="My Application"
11        android:roundIcon="@mipmap/ic_launcher_round"
12        android:supportsRtl="true"
13        android:theme="@style/Theme.MyApplication"
14        tools:targetApi="31">
15
16         <activity
17             android:name=".MainActivity"
18             android:exported="true"
19             android:label="My Application"
20             android:theme="@style/Theme.MyApplication">
21             <intent-filter>
22                 <action android:name="android.intent.action.MAIN" />
23
24                 <category android:name="android.intent.category.LAUNCHER" />
25             </intent-filter>
26         </activity>
27     </application>
28 </manifest>
```

# Manifest

Gli attributi icon e label permettono di associare una piccola immagine e una label testuale al componente. Se settato nell'app (come in questo esempio), diventano il default per tutti i componenti activity.

Si possono associare icon e label anche a <intent-filter> per definire l'icona e il testo che verrà presentato all'utente. Di default questi valori sono ereditati dall'elemento padre (che sia <activity> se esplicitati, o <application>)

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools">
4
5      <application
6          android:allowBackup="true"
7          android:dataExtractionRules="@xml/data_extraction_rules"
8          android:fullBackupContent="@xml/backup_rules"
9          android:icon="@mipmap/ic_launcher"
10         android:label="My Application"
11         android:roundIcon="@mipmap/ic_launcher_round"
12         android:supportsRtl="true"
13         android:theme="@style/Theme.MyApplication"
14         tools:targetApi="31">
15             <activity
16                 android:name=".MainActivity"
17                 android:exported="true"
18                 android:label="My Application"
19                 android:theme="@style/Theme.MyApplication">
20                 <intent-filter>
21                     <action android:name="android.intent.action.MAIN" />
22
23                     <category android:name="android.intent.category.LAUNCHER" />
24                 </intent-filter>
25             </activity>
26         </application>
27
28     </manifest>
```

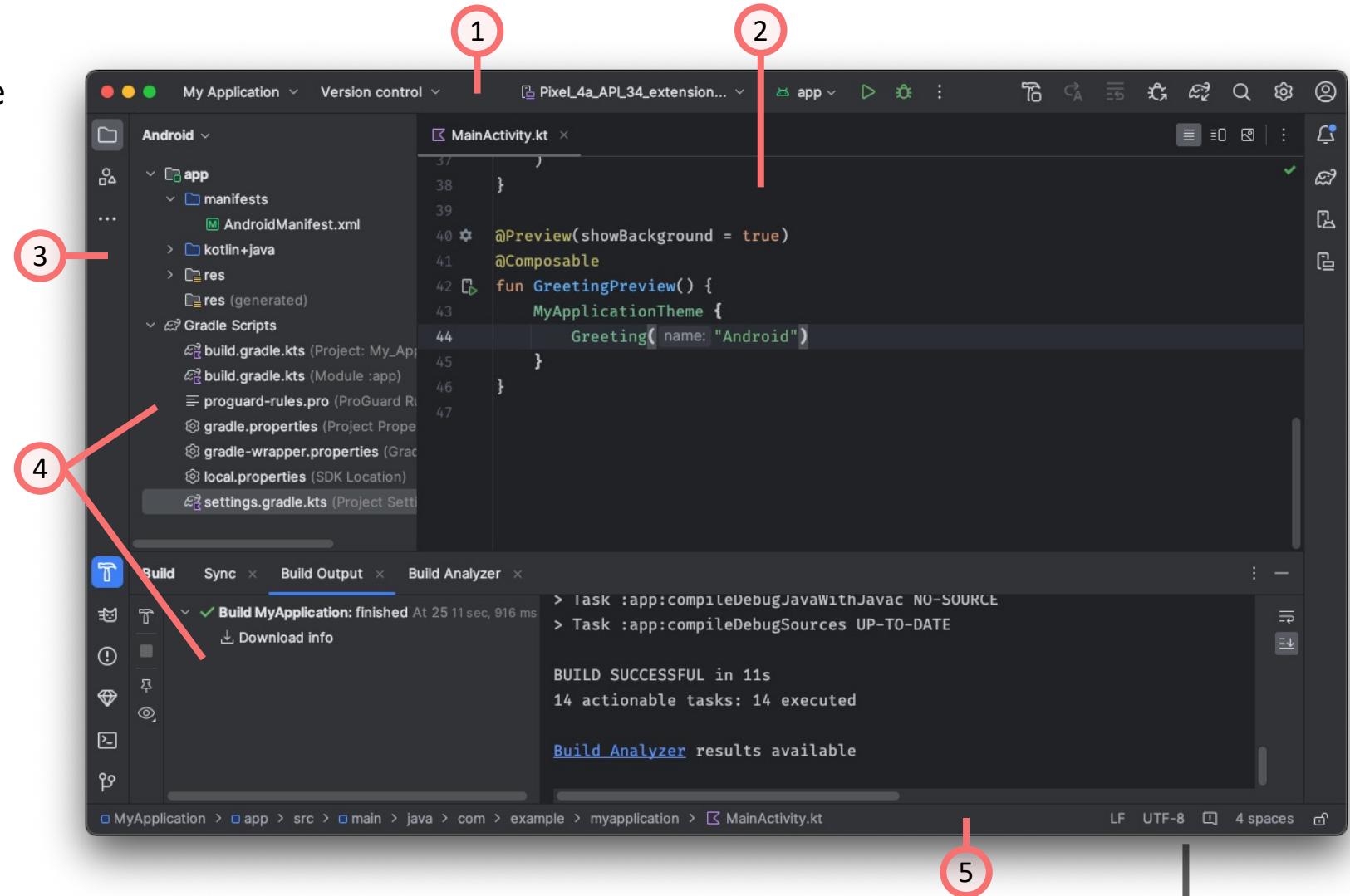
# Manifest

Gli attributi `supportsRtl`, `theme` e `targetApi` permettono rispettivamente di specificare il supporto per i layout right-to-left, settare un tema specifico all'applicazione e definire l'SDK minima per far girare l'applicazione.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:tools="http://schemas.android.com/tools">
4
5     <application
6         android:allowBackup="true"
7         android:dataExtractionRules="@xml/data_extraction_rules"
8         android:fullBackupContent="@xml/backup_rules"
9         android:icon="@mipmap/ic_launcher"
10        android:label="My Application"
11        android:roundIcon="@mipmap/ic_launcher_round"
12        android:supportsRtl="true"
13        android:theme="@style/Theme.MyApplication"
14        tools:targetApi="31">
15
16         <activity
17             android:name=".MainActivity"
18             android:exported="true"
19             android:label="My Application"
20             android:theme="@style/Theme.MyApplication">
21             <intent-filter>
22                 <action android:name="android.intent.action.MAIN" />
23
24                 <category android:name="android.intent.category.LAUNCHER" />
25             </intent-filter>
26         </activity>
27     </application>
28 </manifest>
```

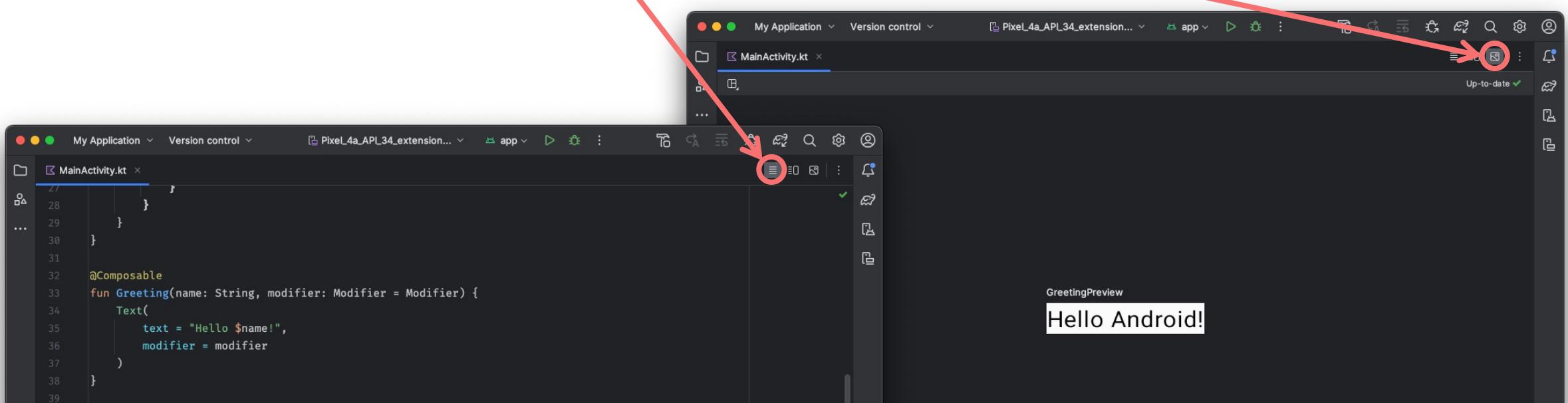
# Android Studio - Interfaccia

- 1 **Toolbar:** permette di eseguire diverse azioni, incluso l'avvio dell'app.
- 2 **Editor:** contiene un tab per ogni file aperto e permette di modificare il codice.
- 3 **Tool window bar:** contiene i pulsanti che consentono di espandere o comprimere le singole tool windows.
- 4 **Tool windows:** consentono di accedere ad attività specifiche come la gestione dei progetti, la ricerca, il controllo di versione e altro ancora.
- 5 **Navigation e status bar:** permette di navigare il progetto e ne mostra lo stato, con particolare focus sul file attualmente selezionato nell'editor.



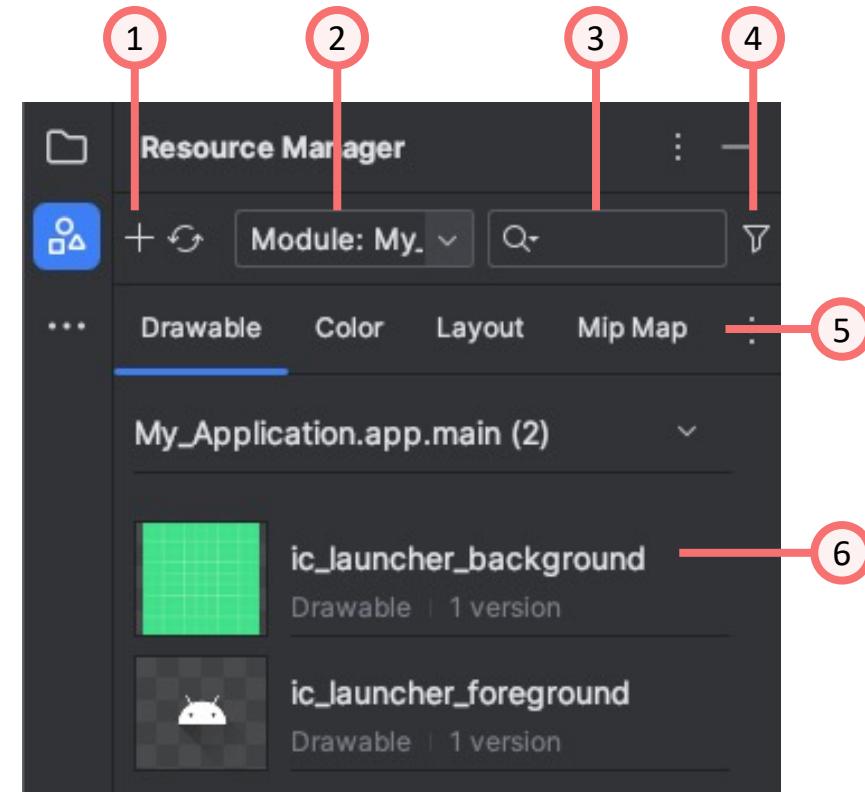
# Layout dell'editor

- Se si apre un file che contiene dei componenti, è possibile scegliere come visualizzarlo all'interno dell'editor.
- Tre modalità:      1. Code      2. Design      3. Entrambe



# Resource Manager

- Permette di
  - 1 Aggiungere una nuova risorsa al progetto. È possibile aggiungere immagini raster o vettoriali, font, file, o importare drawable.
  - 2 Filtrare le risorse in base a un modulo selezionato.
  - 3 Cercare una risorsa in tutti i moduli del progetto.
  - 4 Visualizzare le risorse in base al tipo.
  - 5 Applicare filtri avanzati, come filtrare le risorse utilizzate dai moduli locali, dalle librerie esterne e dal framework Android.
  - 6 Visualizzare l'anteprima delle risorse. Facendo clic con il tasto destro del mouse su una risorsa, si apre un menu in cui è possibile rinominarla e cercare il punto dell'app in cui è utilizzata.



# Connect to Firebase

- **Firebase** è una piattaforma di Google per la creazione di app, videogiochi e siti web, che aiuta a velocizzare lo sviluppo e far crescere la user base.
- Fornisce varie funzionalità complementari, che si possono abilitare in base alle necessità, come autenticazione, database, storage di file e analitica.
- Per semplificarne ulteriormente l'adizione, l'Assistant di Android Studio offre una finestra dedicata a Firebase, tramite cui è possibile integrarne le funzionalità all'interno di un progetto Android.

[https://developer.android.com/studio/write.firebaseio](https://developer.android.com/studio/write/firebase)

The screenshot shows the official Firebase website. At the top, there are tabs for 'Assistant' and 'Firebase', with 'Firebase' being the active tab. The page features the Firebase logo (a yellow flame icon) and a brief introduction: 'Firebase gives you the tools and infrastructure from Google to help you develop, grow and earn money from your app. Learn more'. Below this, a list of services is presented with icons and brief descriptions:

- Analytics**: Measure user activity and engagement with free, easy, and unlimited analytics. [More info](#)
- Authentication**: Sign in and manage users with ease using popular login providers like Google, Facebook, and others. You can even use a custom authentication system. [More info](#)
- Realtime Database**: Store and sync data with this cloud-hosted NoSQL database. Data is synced across all clients in realtime and remains available when your app goes offline. [More info](#)
- Cloud Firestore**: Store and sync your app data with this flexible, scalable NoSQL cloud-hosted database. [More info](#)
- Cloud Storage for Firebase**: Store and retrieve large files like images, audio, and video without writing server-side code. [More info](#)
- (...) Cloud Functions for Firebase**: Automatically run backend code in response to events triggered by Firebase features and HTTPS requests. [More info](#)
- Firebase ML**: Firebase ML is a mobile SDK that brings Google's machine learning expertise to Android and iOS apps in a powerful yet easy-to-use package. [More info](#)
- Crashlytics**: Get clear, actionable insight into app issues that erode your app quality. [More info](#)
- Performance Monitoring**: Gain insight into the performance characteristics of your app. [More info](#)
- Test Lab**: Test your apps against a wide range of physical devices hosted in Google's cloud. [More info](#)
- App Distribution**: Distribute pre-release versions of your app to your trusted testers. [More info](#)

# Eseguire l'app

---

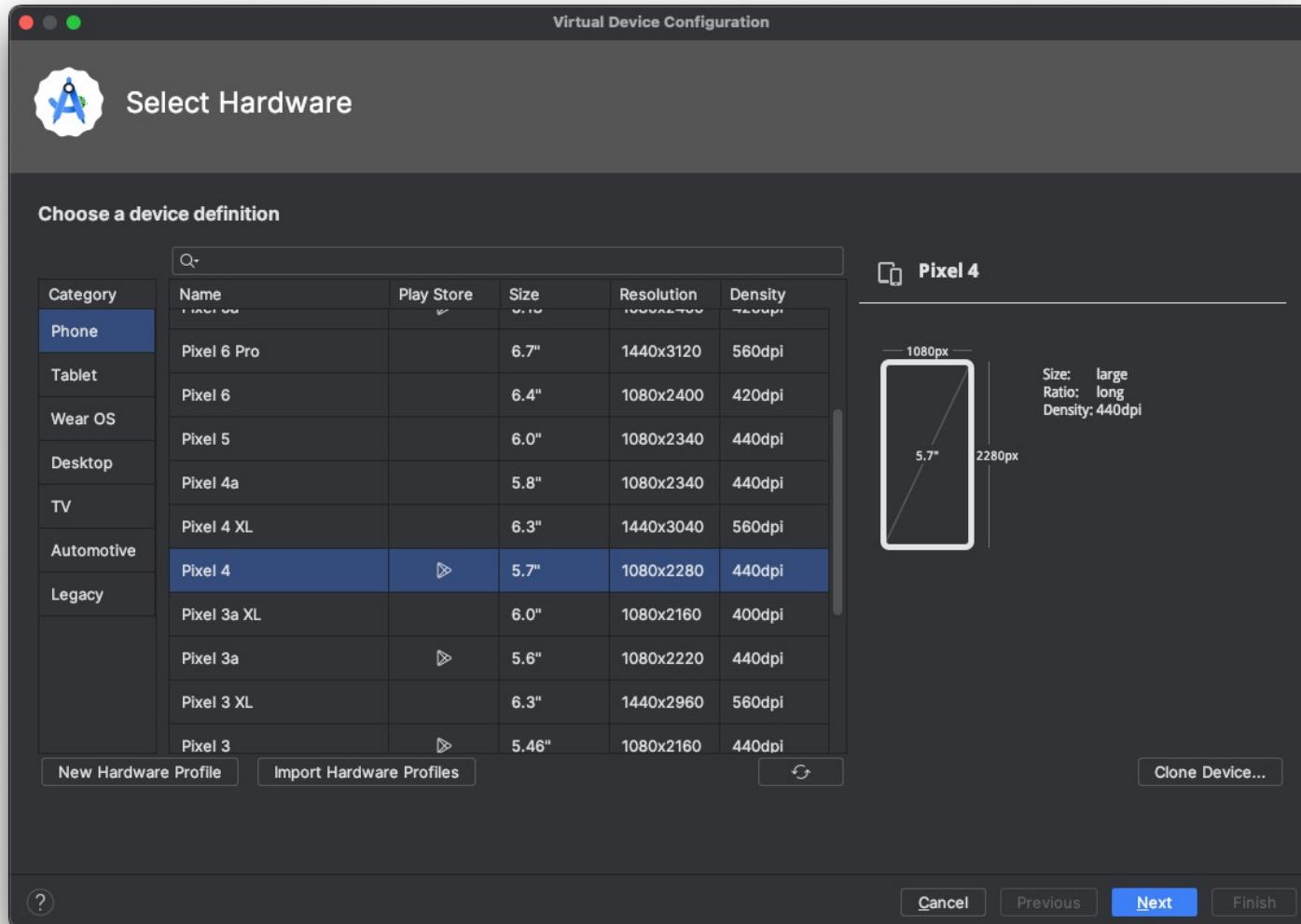
- Due possibilità
  - Su un device fisico
    - Con debugging USB o wireless abilitato
    - Connesso al PC con un cavo o accoppiato tramite Wi-Fi  
<https://developer.android.com/studio/run/device>
  - Sull'emulatore di Android studio
    - Soluzione che useremo nei laboratori
    - Richiede [la creazione di un Android Virtual Device \(AVD\)](#)
    - **Caldamente consigliato:** AVD con l'immagine di sistema di **Android 14**
    - Consigliato: AVD che utilizza **Pixel 6** come dispositivo

# Android Studio – Emulator

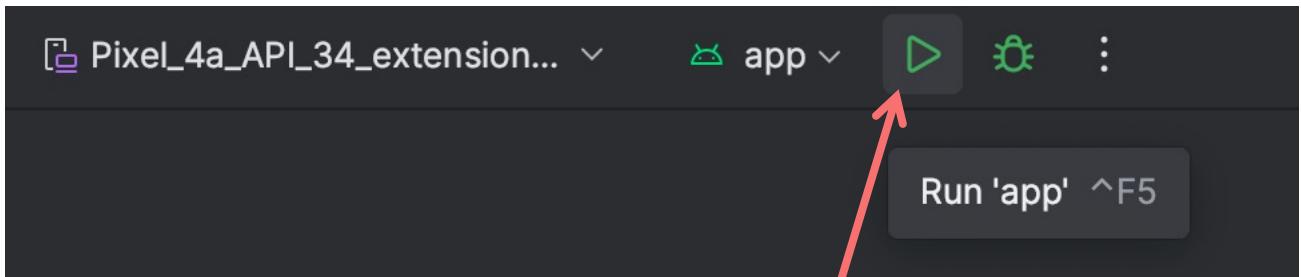
---

**Android Emulator:**  
Fast and Feature-Rich

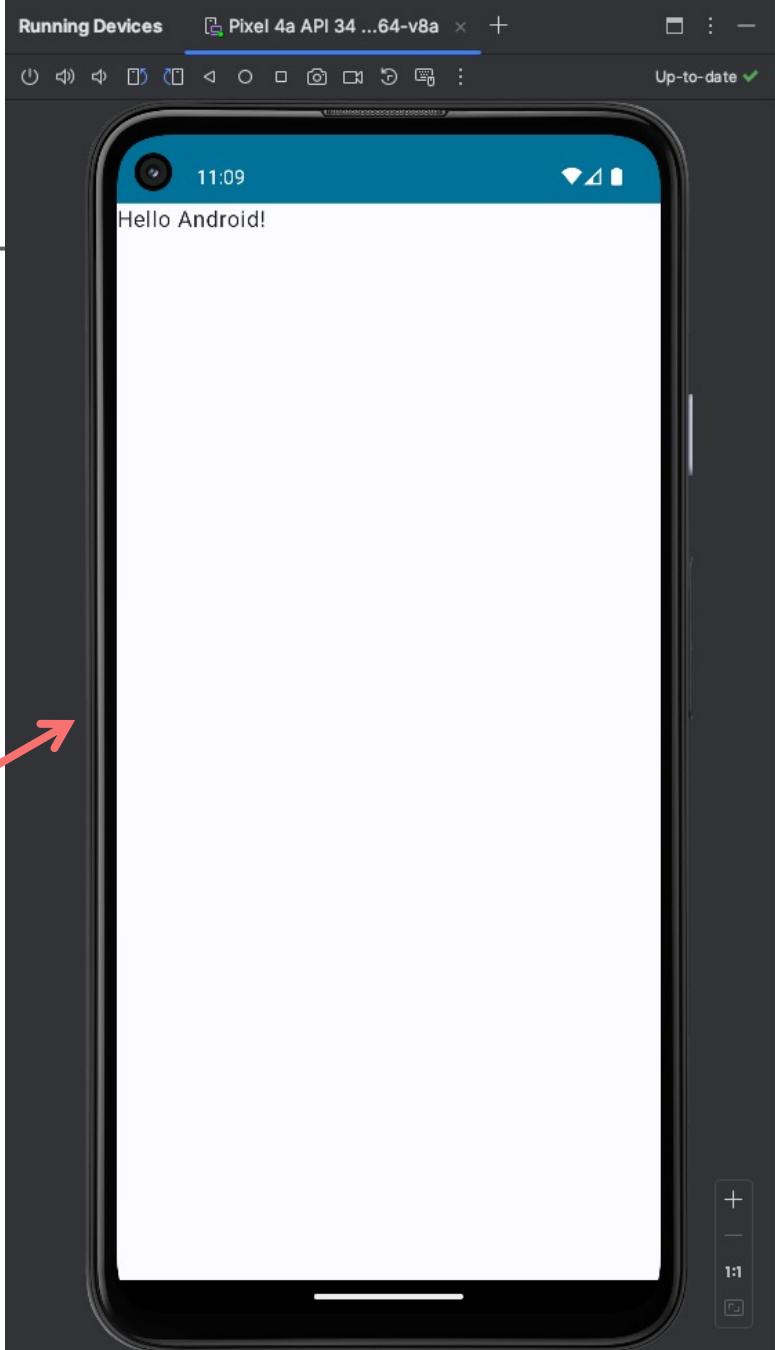
# Android Studio – Emulator



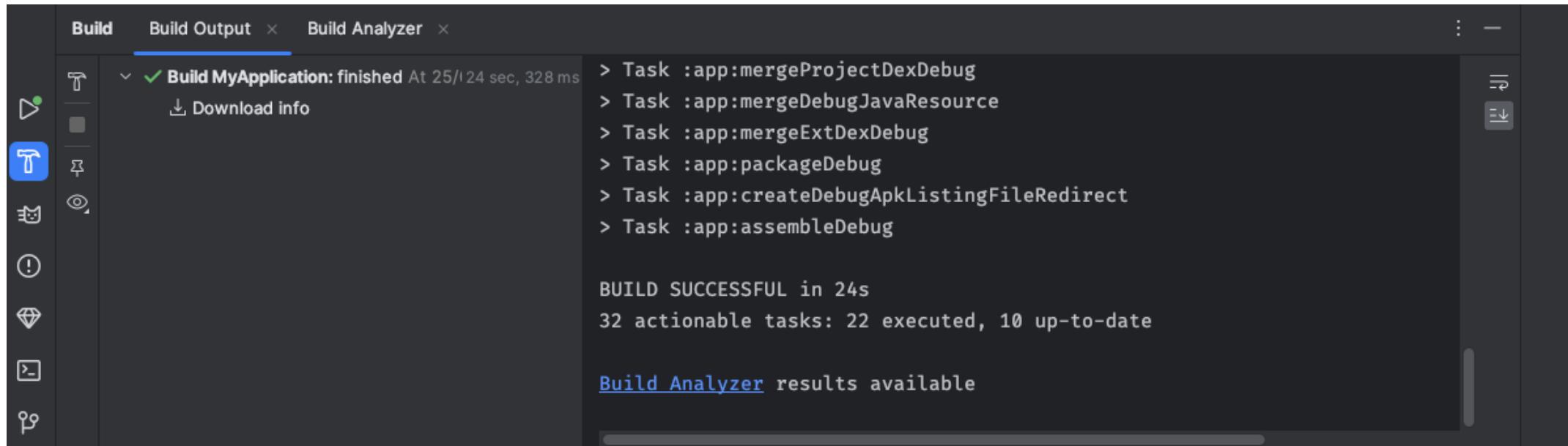
# Android Studio – Run app



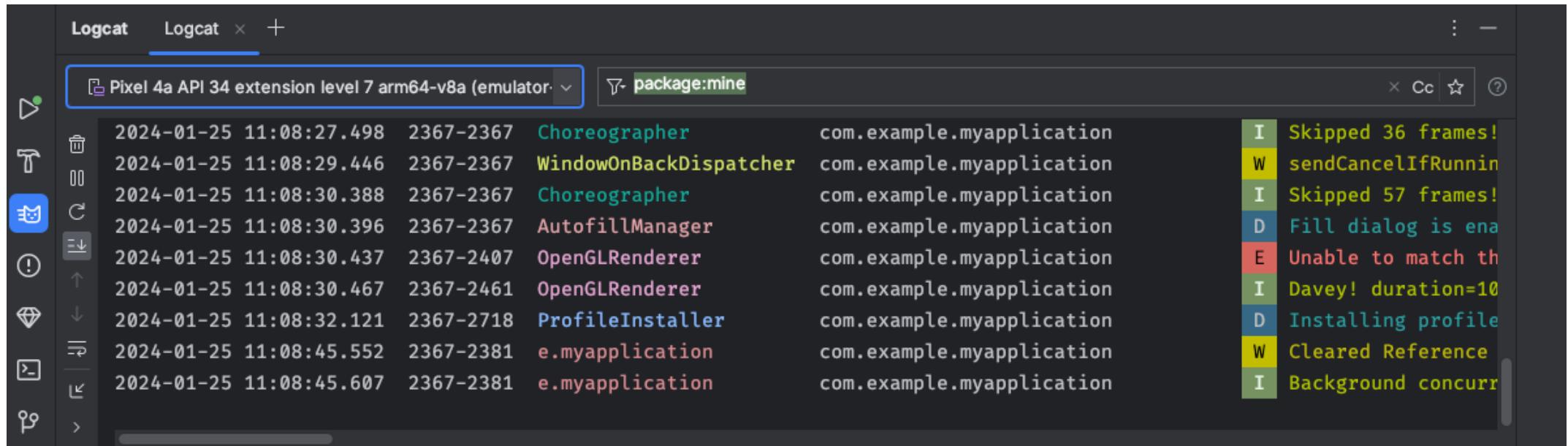
- Basta fare click su Run!
- Android Studio avvia l'emulatore configurato e vi installa l'app.
- Dopo "qualche" secondo, potrete vedere il vostro "Hello, Android!".



# Monitor the build process



# Monitor the log

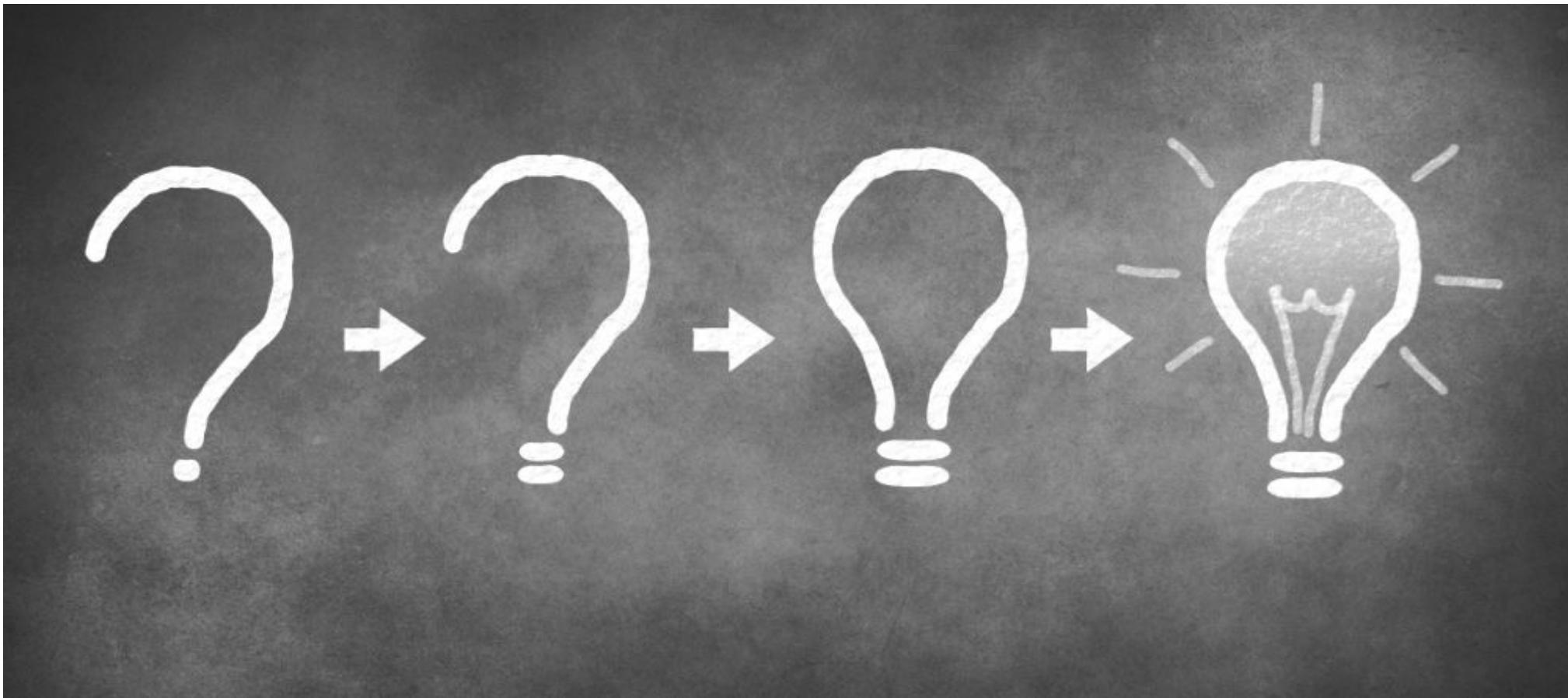


The screenshot shows the Android Logcat interface. The title bar says "Logcat" and has a "+" button. Below it is a dropdown menu showing "Pixel 4a API 34 extension level 7 arm64-v8a (emulator)". A search bar contains the text "package:mine". On the left is a toolbar with various icons: a play button, a trash can, a refresh, a cat icon, a warning, an up arrow, a diamond, a square, a left arrow, and a right arrow. The main area displays a list of log entries:

Date	Time	Thread ID	Category	File	Message	Level
2024-01-25	11:08:27.498	2367-2367	Choreographer	com.example.myapplication	I Skipped 36 frames!	I
2024-01-25	11:08:29.446	2367-2367	WindowOnBackDispatcher	com.example.myapplication	W sendCancelIfRunning	W
2024-01-25	11:08:30.388	2367-2367	Choreographer	com.example.myapplication	I Skipped 57 frames!	I
2024-01-25	11:08:30.396	2367-2367	AutofillManager	com.example.myapplication	D Fill dialog is ena	D
2024-01-25	11:08:30.437	2367-2407	OpenGLRenderer	com.example.myapplication	E Unable to match th	E
2024-01-25	11:08:30.467	2367-2461	OpenGLRenderer	com.example.myapplication	I Davey! duration=10	I
2024-01-25	11:08:32.121	2367-2718	ProfileInstaller	com.example.myapplication	D Installing profile	D
2024-01-25	11:08:45.552	2367-2381	e.myapplication	com.example.myapplication	W Cleared Reference	W
2024-01-25	11:08:45.607	2367-2381	e.myapplication	com.example.myapplication	I Background concurr	I

# Domande?

---



# Riferimenti

---

- Documentazione Android Developers  
<https://developer.android.com/>
- Download di Android Studio  
<https://developer.android.com/studio/archive/>
- Guida all'installazione  
<https://developer.android.com/studio/install/>
- Gradle  
<https://gradle.org/>
- Firebase  
<https://firebase.google.com/>