Pràctica introductòria #1 - Primers passos amb Android Studio

Projecte Integrat de Software (2024/25)
Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Introducció

Android Studio (AS) és el programari que fareu anar per a desenvolupar el vostre projecte Android. AS és un entorn de desenvolupament integrat basat (IDE) en IntelliJ, que a més incorpora característiques addicionals que us permeten desenvolupar la part gràfica de les aplicacions Android. Seguint aquest primer document, aprendreu a iniciar-lo, entendre com s'estructura el projecte i quins fitxers conté, saber quines eines ofereix i com es fan anar, etc.

Requeriments

- Tenir Android Studio instal·lat, el qual podeu baixar de `https://developer.android.com/studio`. Aquest document assumeix que teniu instal·lada la versió `Koala | 2024.1.1 Patch 1`.
- **Opcional**. GitHub Copilot (amb Android Studio plug-in).

Objectius

1. **Crear un nou projecte** d'Android amb Android Studio (Sec. 1).
2. Conèixer l'**estructura i els fitxers dels projectes** Android (Sec. 2).
3. **Compilar i executar el projecte** (Sec. 3).

El temps de seguiment d'aquesta guia és d'aproximadament 1h.

1 Creació d'un nou projecte d'Android

Sempre que inicieu l'IDE Android Studio, us apareixerà la finestra de benvinguda ("Welcome to Android Studio") que es mostra a la Fig. 1. Des d'aquesta finestra podreu tornar a obrir un projecte existent o crear-ne un de nou. Per a crear un nou projecte, fareu clic al botó "New Project". Nota: pot ser que hi hagi canvis cosmètics en les últimes versions o segons el sistema operatiu, però no hauria d'impedir-vos seguir aquest document.

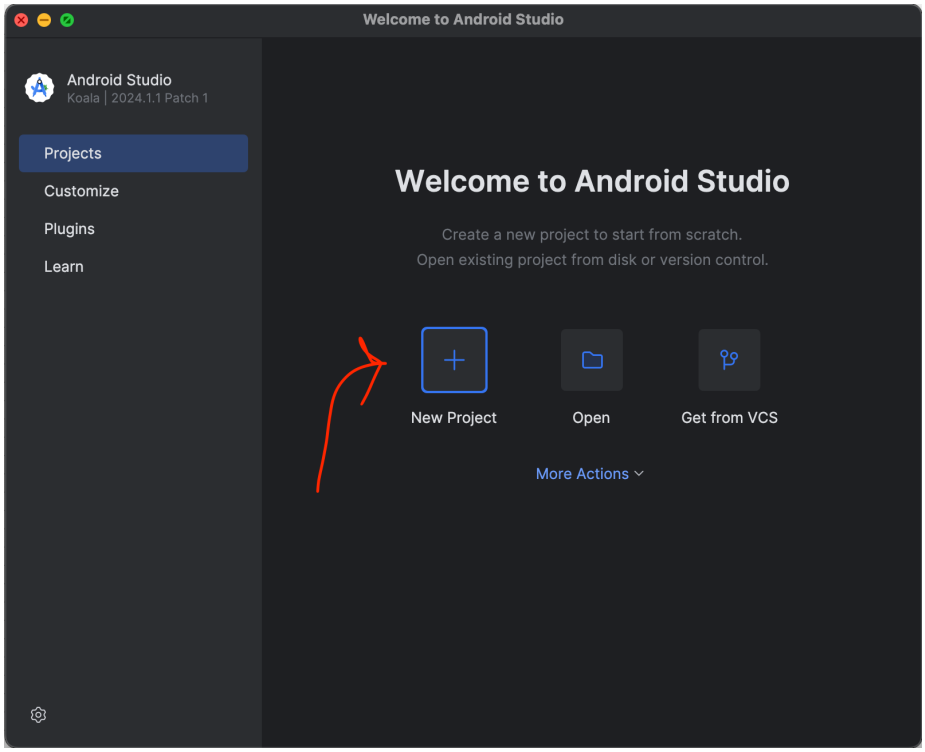

Figure 1: Finestra de benvinguda.

En una nova finestra ("New Project") se us mostraran un seguit de plantilles (o *templates*) que venen preestablerts a AS. Veure Fig. 2. Com que estem començant, seleccionem la més senzilla possible, "Empty Views Activity", i premem el botó "Next".

Seguidament, tal i com es mostra a la Fig. 3, caldrà especificar els detalls per al nostre nou projecte. Aquests són:

- **Name**: el nom del projecte. En el nostre exemple, li hem dit "LaMevaPrimeraApp".
- **Package name**: el nom del paquet que contindrà el vostre codi. Fixeu-vos-hi que el sufix és, automàticament, el nom del vostre projecte ("lamevap-

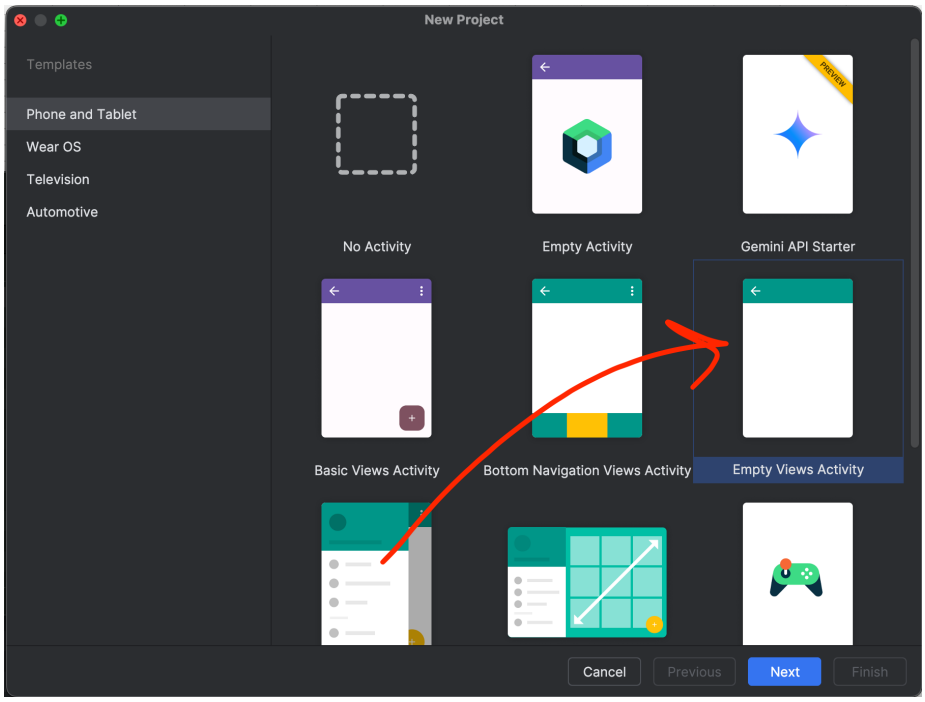

Figure 2: Selecció de plantilla predefinida.

rimerapp"). En canvi, el prefix l'haureu d'especificar manualment. *Seguirem la convenció del prefix "edu.ub.pis2425" seguit del nom del projecte.*

- **Save location**: és el directori del vostre ordinador on es guardarà el projecte.
- **Language**: és el llenguatge amb el que desenvoluparem. Triem "Java".
- **Minimum SDK**: la versió del SDK d'Android que farà servir el projecte. Per augmentar la compatibilitat entre dispositius, per a facilitar la col.laboració i facilitar la tasca del professorat, triarem la versió 'API 26 ("Oreo"; Android 8.0)'.
- **Build configuration language**: llenguatge de programació utilitzat per a definir el fitxer de configuració que farà servir l'eina de compilació Gradle. Com que triem el llenguatge Java per a programar la nostra app, *escollirem la opció "Groovy DSL (build.gradle)"*.

Un cop completats tots els detalls, cliquem "Finish" per què s'obri l'IDE amb el nou projecte.

L'IDE que hauríeu de veure és el de la Fig. 4 o alguna cosa molt similar. A l'esquerra hi teniu l'estructura de directoris i, seguidament, movent-nos cap

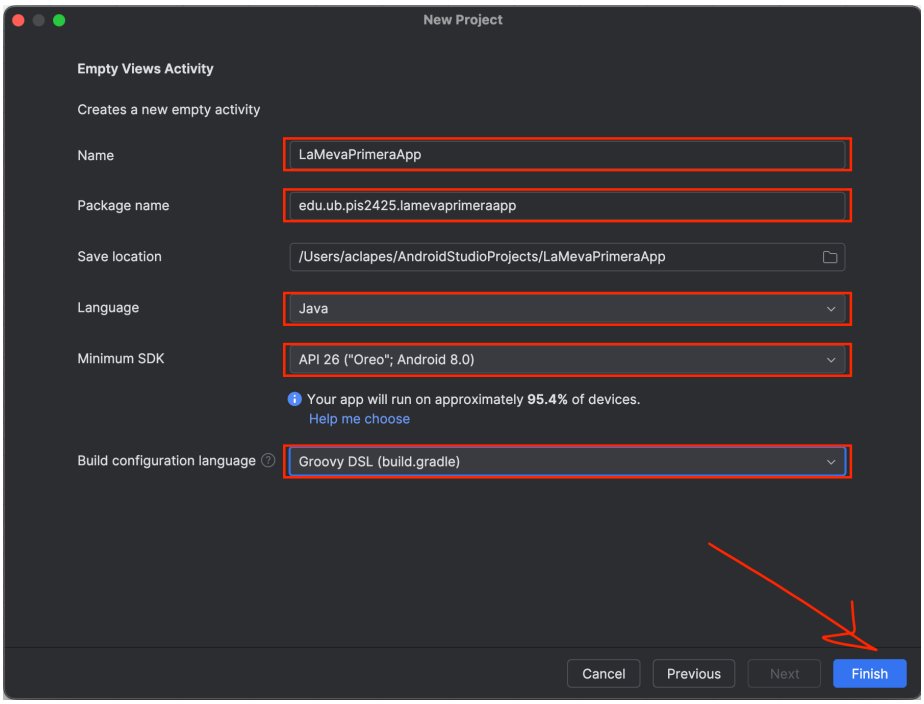

Figure 3: Detalls del nostre nou projecte. En vermell marcats els camps que heu de recordar de revisar (i, si cal, modificar).

a la dreta l'editor de codi. A la imatge, seguint cap a la dreta, també veieu una subfinestra anomenada "Assistant", la qual s'ha obert automàticament. Aquesta, igual que d'altres subfinestres, s'obren clicant a les icones que hi ha per tot el lateral. Intenteu localitzar l'icona que tanca la subfinestra "Assistant".

2 Estructura i fitxers dels projectes

A la subfinestra del "Projecte", hi veureu un directori `app` que podeu desplegar (amb el botó ∨) per veure 3 directoris principals: `manifests`, `java` i `res`, que són on s'hi guarden – respectivament – el manifest de l'aplicació, el codi (incloent-hi tests) i els recursos de l'aplicació (imatges, xml, etc).

Directori `java`. Hi tindreu el paquet principal creat `edu.ub.pis2425.lamevaprimeraapp`, que és on hi tindreu la implementació de la vostra aplicació amb els vostres propis paquets. Veurem més endavant quins són els paquets que haurà de tenir el vostre projecte. Els dos altres paquets pels tests, els quals ignorarem per ara, són pel codi dels tests i dels tests d'integració.

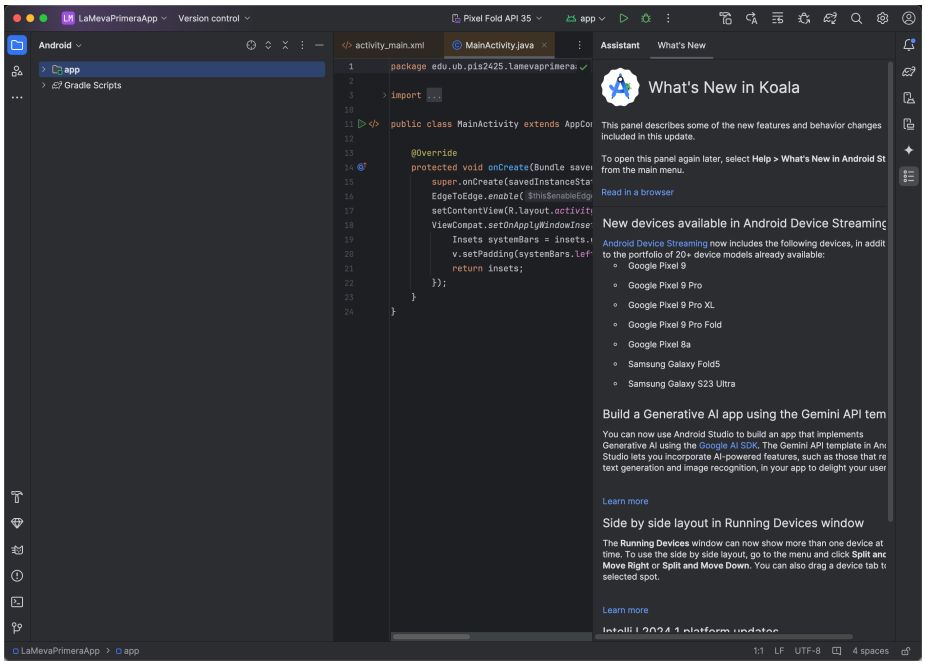

Figure 4: Pantalla de l'IDE just després de crear un nou projecte.

Directori `manifests`. Conté un sol fitxer xml, anomenat `AndroidManifest.xml`. En aquest fitxer, s'hi defineixen des de temes d'estil fins a quin és el punt d'entrada del codi. Pel que fa a això últim, exploreu el fitxer i fixeu-vos en com es defineix que el punt d'entrada serà aquesta classe `MainActivity.java`. Més endavant veurem què són les activitats, per ara podeu pensar-les com a "pantalles" de la vostra aplicació (equivaldria a un JDialog).

Directori `res`. Són els recursos de l'aplicació, inclosos els que tenen a veure amb la interfície gràfica. Si desplegueu el directori (∨), veureu que conté ja una sèrie de subdirectoris per defecte: `drawable`, `layout`, `mipmap`, `values` i `xml`. És important destacar el subdirectori de `layout`, que és on hi tindrem almenys els *fitxers de layout*; almenys un per a cada classe de tipus Activitat del directori java.

El projecte conté el mínim codi necessari per a córrer com una aplicació: una Activitat ("MainActivity") que conté el codi Java (`java/edu.ub.pis2425.lamevaprimeraapp/MainActivity.java`) i el seu *layout*[1] associat (`res/layouts/activity_main.xml`). A continuació, veurem com compilar i executar el projecte.

[1] Un *layout* és la disposició dels elements gràfics d'una interfície, en aquest cas, d'una pantalla (activitat).

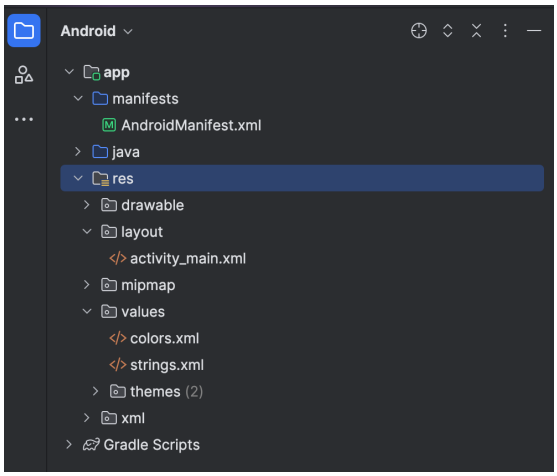

Figure 5: Directori de recursos i els seus continguts.

3 Compilar i executar el projecte

Per a compilar, podeu anar al menú superior, buscar el menú "Build" i fer "Make Project" – tal com s'il·lustra a la Fig. 6. **A Windows, la barra superior està amagada i heu de clicar un botó amb tres ratlles horitzontals per mostrar-lo**. A la part inferior de l'IDE us hauria d'aparèixer un missatge que diu "Gradle Build Running" i una barra de progrés.

Un cop compilat el projecte, proveu d'executar-lo. Per a això aneu una altra vegada al menú superior i feu "Run" →"Run 'app'" o anar a buscar el botó *play* de la part superior de la finestra (veure Fig. 7). Noteu que fer "Run" també implica, implícitament, compilar en cas que sigui necessari degut a canvis respecte la darrera compilació.

Quan, finalment, proveu d'executar el projecte poden passar dues coses: (a) que no tingueu un emulador configurat i us aparegui un missatge flotant a la part inferior dreta; o (b), que tingueu configurat l'emulador d'un dispositiu molt nou, però alhora pesat (p. ex. Pixel Fold). Es recomana canviar-lo per un de més antic, però lleuger. A continuació, expliquem com triar el dispositiu per emular.

3.1 Configurar l'emulació d'un dispositiu

Aneu al menú superior →"Tools" →"Device manager". Us apareixerà una subfinestra a la part dreta de la pantalla (veure Fig. 8) amb els dispositius i el tipus de dispositiu ("Virtual" o "Remote"). En aquesta finestra heu de crear un dispositiu virtual fent clic al botó que s'indica a la mateixa figura. A partir d'aquí seguireu els següents passos:

1. **Select Hardware**. Teniu una llista de dispositius per triar. Per a maxim-

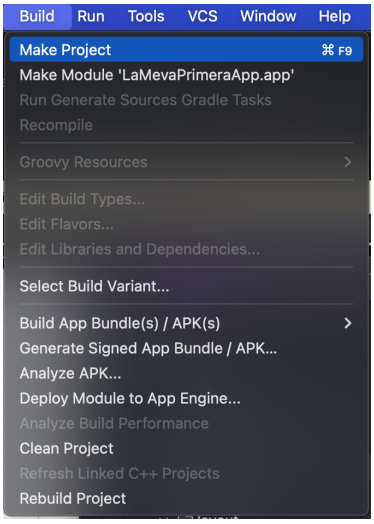

Figure 6: Compilar projecte.

Figure 7: Executar el projecte.

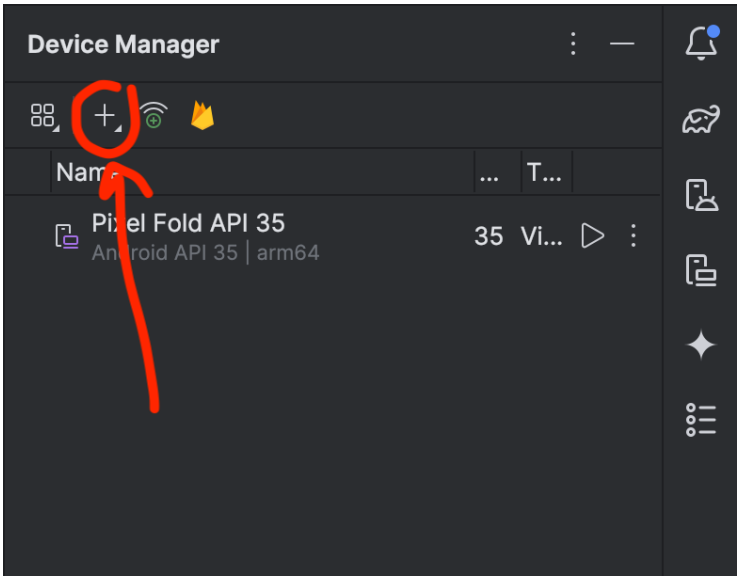

Figure 8: Finestra del gestor de dispositius (o Device Manager).

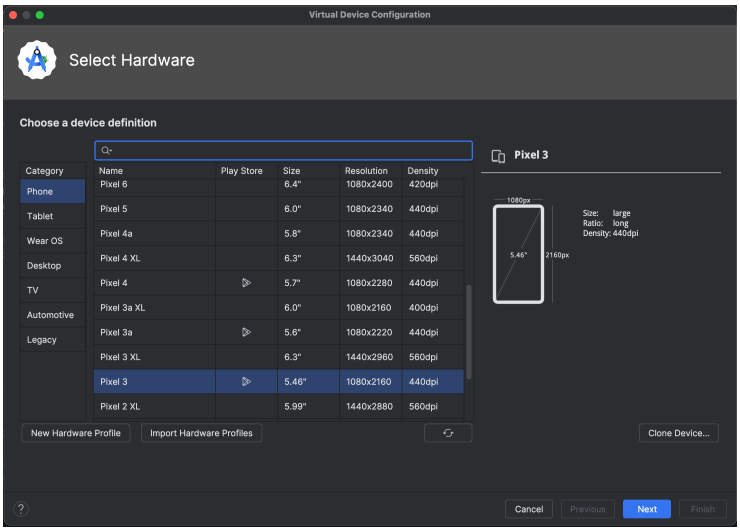

Figure 9: Tria del dispositiu (hardware) que emularà l'emulador.

itzar la compatibilitat i evitar problemes als ordinadors de laboratori, la nostra recomanació és basar-se sempre en el dispositiu "Pixel 3" – tal com mostra la Fig. 9. Altres versions superiors poden resultar feixugues i no guanyar característiques prou rellevants. Un cop tirada la versió, feu "Next".

2. **System Image**. Ara haureu de triar la versió d'Android que s'instal·larà en el dispositiu virtual. La recomanació és instal·lar-hi "Pie" (Android 9.0), que equival a la versió d'API 28. Si no us apareix a la pestanya de "Recommended", busqueu-la a la pestanya "Other images". D'entre les versions de Pie (API 28), veureu diverses "ABI". Trieu-la segons el sistema operatiu: amb Windows, la versió amb ABI "x86_64" i, amb OSX amb MacOS (silicon), la versió amb ABI "arm64-v8b". Per fer la instal.lació de la imatge del sistema, heu de clicar el botó al costat del nom de la versió per a que s'obri una finestra emergent "SDK Component Installer". Pot trigar una estona. Quan hagi acabat, tornareu a la finestra "System Image", seleccionareu la versió i fareu "Next".

3. **Android Virtual Device (AVD)**. No cal tocar res. Clicar "Finish".

El nou dispositiu virtual s'hauria d'haver establert com a preferit. Podeu comprovar-ho a la part superior de la finestra, a l'esquerra d'on havíeu trobat el botó *play*, mirant si apareix el nom del dispositiu ("Pixel 3 API 28").

Executeu amb *play* per veure l'aplicació corrent en el dispositiu virtual i, finalment, una pantalla missatge "Hello World!", igual que il·lustrem a la Fig. 11. Enhorabona, acabeu de compilar i executar la vostra primera aplicació Android!

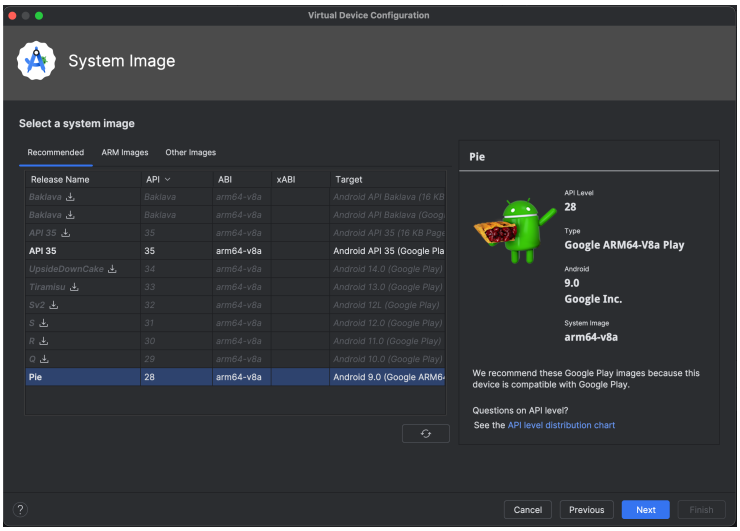

Figure 10: Tria de la versió del sistema (software) que emularà el dispositiu virtual. Per poder triar-la, haureu d'haver instal·lat baixat la imatge del sistema amb el botó ⬇ corresponent.

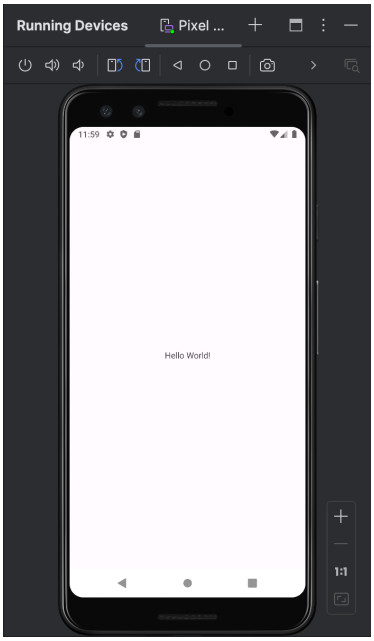

Figure 11: Aplicació bàsica de "Hello world!" corrent en dispositiu virtual amb Android.

3.2 Ajustament de les versions de l'SDK

Aquest pas no és estrictament necessari, però pot interessar-vos saber que en el desenvolupament Android es tenen en compte tres versions diferents de l'Android Sdk i que, molt probablement, us topeu amb elles durant el desenvolupament del projecte. Per tant, és bo conèixer-ne les diferències:

- ***minSdk***: és la versió que hem triat quan hem creat el projecte (API 28). Estableix la versió mínima que l'Android OS d'un dispositiu ha de tenir disponible per a poder instal·lar i executar l'aplicació. En cas contrari, apareix un error alhora d'instal·lar-la/executar-la. És la manera que tenen els desenvolupadors d'impedir la instal·lació en dispositius massa antics (amb versions d'Android desfasades) i haver de garantir retrocompatibilitat.

- ***targetSdk***: la versió que amb la que el desenvolupador pot assegurar que l'aplicació funciona[2], perquè és la versió on ha testejat. En el nostre cas, les proves les farem en el Pixel 3 amb API 28 i, per tant, la versió del *targetSdk* seria 28. Fixeu-vos que, de fet, sol tenir sentit que *targetSdk* = *minSdk*.

- *compileSdk*: especifica la versió de l'SDK amb la qual es compila el codi. Aquesta versió determina les característiques, optimitzacions i correccions de seguretat disponibles durant el desenvolupament. Podeu utilitzar l'última versió disponible (probablement, API 34). Si la versió de *compileSdk* fos massa recent i inclogués funcionalitats no retrocompatibles amb la versió del dispositiu virtual (API 28), caldria ajustar-la a la baixa.

En cas d'haver de modificar cap de les versions, ho podeu fer al fitxer `buil d.gradle(Module:app)` que trobareu seguint la Fig. 12.

Exercicis

Aquesta primera pràctica no té exercicis, es tracta de seguir els passos descrits en la guia.

[2]En el cas de voler publicar l'aplicació a la *Google Playstore*[3], se'ns requerirà una certa versió objectiu (en el moment d'escriure aquesta documentació, API 31).

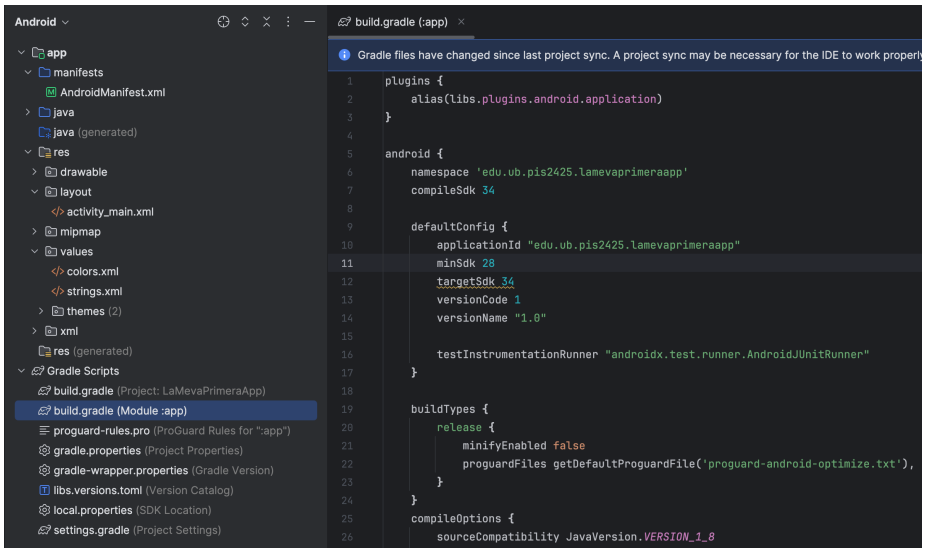

Figure 12: Les diferents versions de l'AndroidSdk s'editen en el fitxer gradle a nivell de mòdul. Podeu accedir-hi tal com s'indica a la imatge.