UNIVERSITAS SCIENTIARUM SZEGEDIENSIS
UNIVERSITY OF SZEGED
Department of Software Engineering

Mobil alkalmazásfejlesztés - Bevezető

Dr. Bilicki Vilmos
Szoftverfejlesztés Tanszék

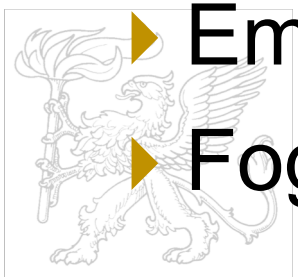
A fóliákhoz felhasznált anyagok:
Google: Android Developer Fundamentals (Version 2), https://developer.android.com/courses/fundamentals-training/overview-v2

Összefoglaló

- Célok
- Követelmények
- Osztályzat
- Háttér
- Bevezető

Bemutatkozás

- Dr. Bilicki Vilmos
- Dugonics tér 150-es szoba
- Telefon: 6781-es mellék (mobil: +36203133523)
- Web: http://www.inf.u-szeged.hu/~bilickiv
- Email: bilickiv@inf.u-szeged.hu
- Fogadóóra: Hétfő 9-10



Követelmények, osztályzat

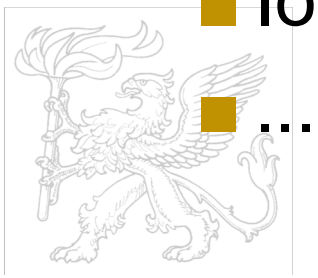
- Vizsga a vizsgaidőszakan a tételsorból két kérdésre kell írásban kifejtős választ adni.

Célok

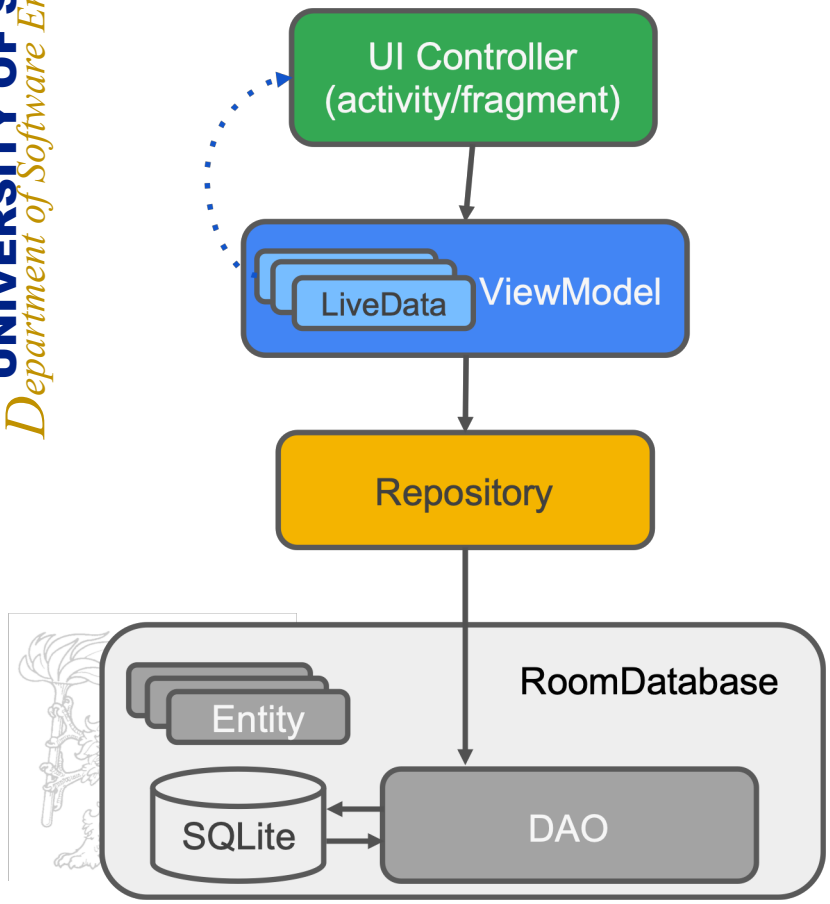
- Natív Java alapú Android fejlesztés alapvető ismeretek
  - https://developer.android.com/courses/fundamentals-training/toc-v2
  - Nem lesz érintve:
    - Fragment
    - Widget
    - Szenzor adatok
    - Teljesítmény
    - Lokalizáció
    - Egyéni grafika, ...
- IOS fejlesztés összevetés

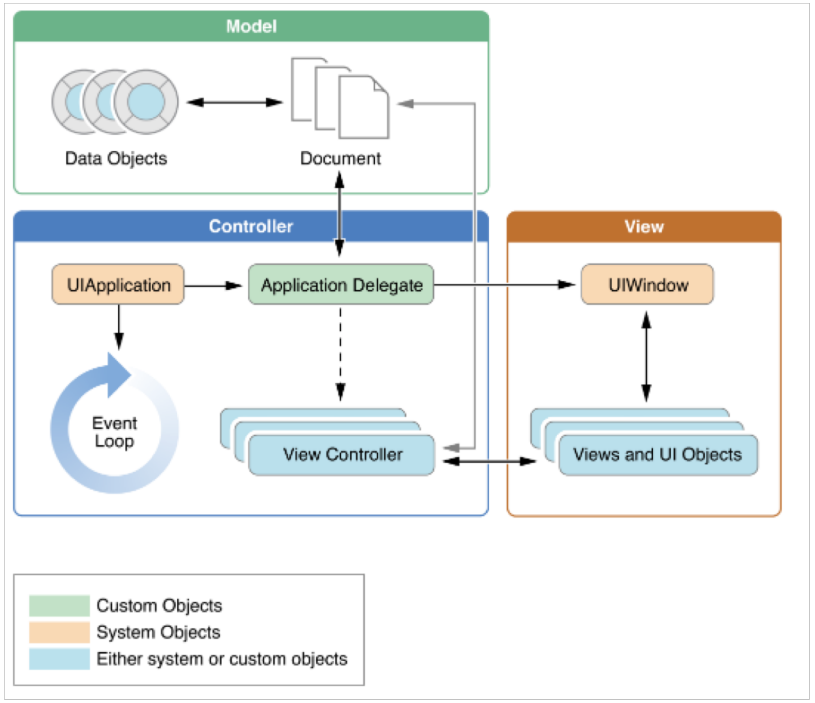
Mobil alkalmazásfejlesztés

- Natív
  - IOS – Swift
  - Andorid – Java, Kotlin
- Keresztplatformos
  - React Native
  - Ionic
  - ...



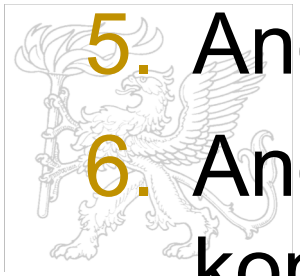
Natív Android vs. IOS





A félév áttekintése I.

1. Bevezető
2. Android: UI – View, Activity, Intent, Layout
3. Android: UI – Activity életciklus, Input, Gombok
4. Android: UI – Navigáció, Menük
5. Android: UI – Stílusok, RecycerView
6. Andorid: Aszinkron feladatok, kommunikáció



A félév áttekintése II.

7. Android: Szolgáltatások, értesítések
8. Android: Adattárolás, átvitel
9. Android: SQL lite
10. Android: ROOM, architektúra
11. IOS: MVC Xcode, SWIFT memóriakezelés, NIL, Layout
12. IOS: Perzisztencia

Bevezető

- Az Android egy ökorendszer
- Android platform architektúra
- Az Android alkalmazásfejelsztés kihívásai
- Alkalmazás alapok

Mi az Android?

- Linux kernel alapú mobil operációs rendszer
- Érintőképernyő alapú felhasználó intefrészek
- A telefonok több 80%-ban alkalmazott
- Olyan eszközökben is mint: okosóra, okos TV, autó
- Több mint 2M Android alkalmazás a Google Playstore-ban
- Testreszabható az aszközgyártók, forgalmazók által
- Nyílt forrású

Android felhasználói interakciók

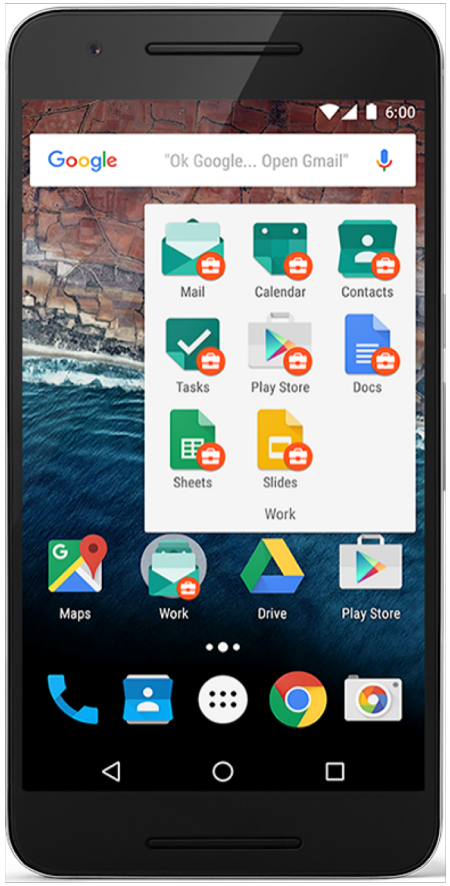
- Érintéses mozdulatsorok: húzás, csippentés, érintés
- Virtuális billentyűzet
- Bluetooth és USB vezérlők támogatása

Android és szenzorok

- A szenzorok érzékelhetik a felhasználói interakciókat és reagálhatnak ezekre
  - Az eszköz megjelentíését, forgatja mikor szükséges
  - Séta közben frissíti a térképet
  - Kormányt forgatja a virtuális autónak, vagy játéknak
  - Ha túl gyorsan mozog akkor letiltja a játék interakciókat

Android indító képernyő

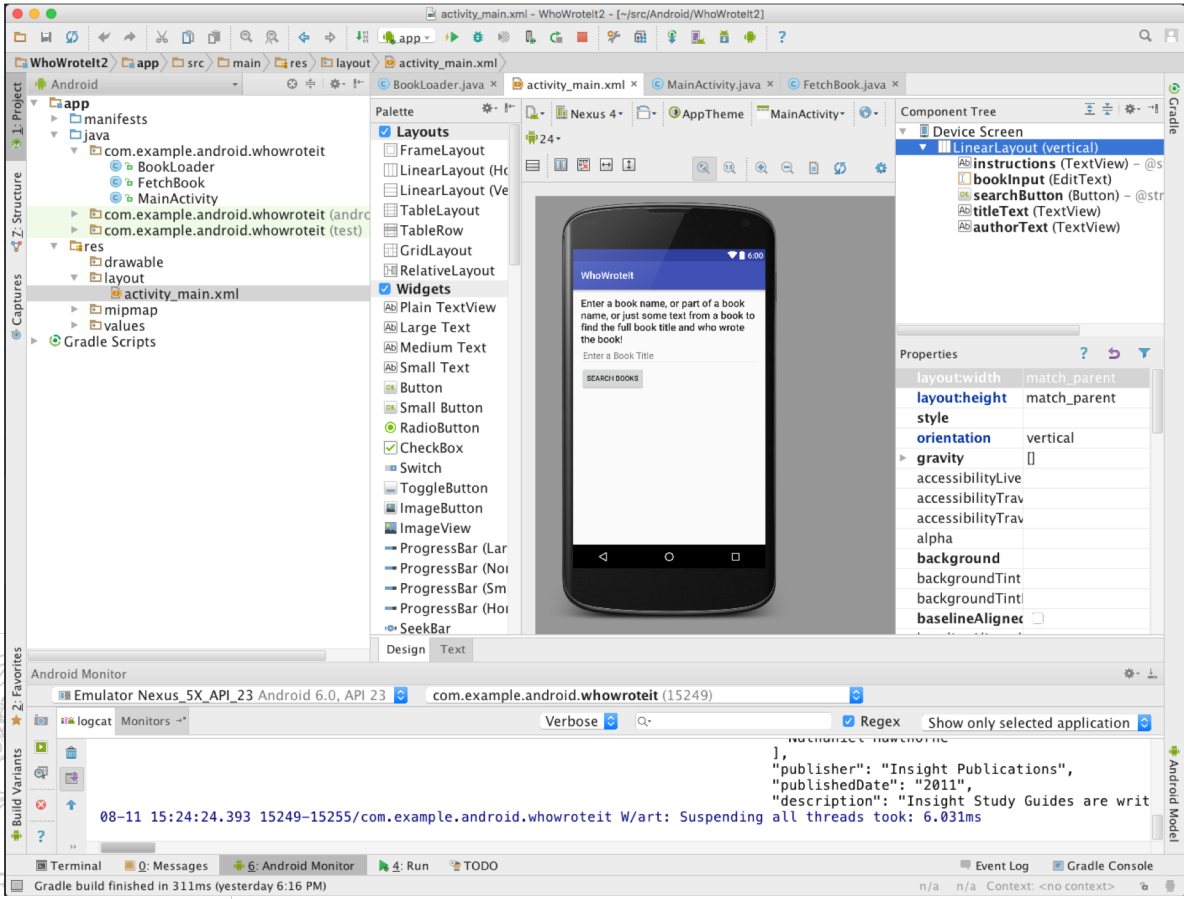
- Alkalmazásindító ikonok
- Önfrissítő modulok
- Több oldal is lehet
- Mappába rendezhető alkalmazások
- „OK Google”



Android Szoftverfejlesztő Eszköztár

- Fejlesztő eszközök (debugger, szerkesztők, naplózók)
- Könyvtárak (térképek, viselhető)
- Virtuális eszközök (emulátorok)
- Dokumentáció (developers.android.com)
- Minta kódok

Android Stúdió



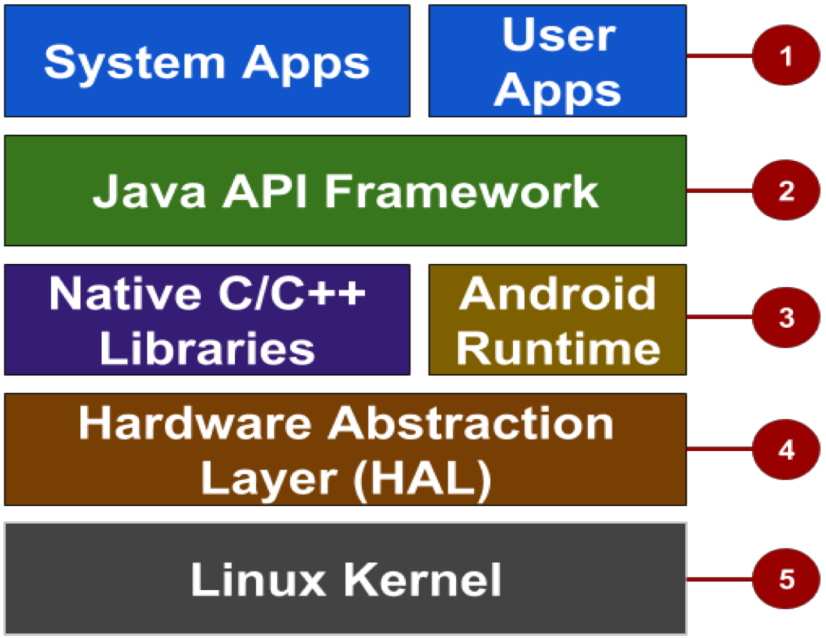
- Hivatalos Android IDE
- Fejlesztés, hibakeresés, futtatás, csomagolás
- Naplózó és teljesítmény monitorozó eszközök
- Virtuális eszközök
- Projekt nézetek
- Vizuális képernyő tervező

Google Play áruház

- Alkalmazások publikálása
  - Hivatalos Android bolt
  - A Google által működtetett digitális elosztó szolgáltatás

Android szoftver verem

1. Rendszer és felhasználói alkalmazások
2. Java-ban elérhető Android oprendszer API
3. Natív interfészek, alkalmazások futtatása
4. Eszköz HW képességek
5. Linux kernel



Rendszer és felhasználói alkalmazások

- A rendszer alkalmazásoknak nincs speciális státusza
- A rendszer alkalmazások biztosítják a szoftver fejlesztők részére a kiinduló szolgáltatás halmazt

Példa:

Az általunk fejlesztett alkalmazás egy rendszer alkalmazás segítségével tud SMS-t küldeni

Java API keretrendszer

- Az Android operációs rendszer teljes képessége elérhető JAVA API-n keresztül
  - Nézet osztályhierarchia: UI képernyők létrehozására
  - Értesítés menedzselő
  - Aktivitás menedzselő életciklus és nagiváció kezelésre

Android futtatási környezet

- Minden alkalmazás külön processzusban fut
- Ezen belül és ezt burkolja be az android futtatási környezet

C/C++ könyvtárak

- Az alap C/C++ könyvtárak natív hozzáférést biztosítanak az Android alap osztályokhoz, szolgáltatásokhoz

HAL – Hardver Absztrakció Réteg

- Szabványon interfészek az eszközök mint könyvtárak elérésére

- Pl: kamera, bluetooth modul

Linux kernel

- Szálkezelés alacsonyszintű memória menedzsment
- Biztonsági képességek
- Meghajtók

Akalmazásfejlesztés

- Egy-vagy több interkatív képernyő
- Java és XML használata
- Az Android SDK segítségével
- Android és Androdi Alkalmazás Keretrendszer könyvtárakat használ
- Az Android Virtuális Gép futtatja (ART)

Android fejlesztői kihívások

- Különböző méretű, felbontású képernyő
- Teljesítmény: az alkalmazás gördülékenyen és válaszképes legyen
- Biztonság: a forráskód és a felhasználói adatok biztonsága
- Kompatibilitás: a régebbi platformokon is jól fusson
- Reklám: értsük meg a piacot

Alkalmazás építőelemek

- Erőforrások: képernyő elrendezések, képek, karakterláncok, színek mint XML és média fájlok
- Komponensek: aktivitások, szolgáltatások és kisegítő osztályok mint Java kód
- Leíró fájl: futtatási környezet számára közzétett adat az alkalmazásról
- Build konfiguráció: APK verziók a Gradle konfig fájlokban

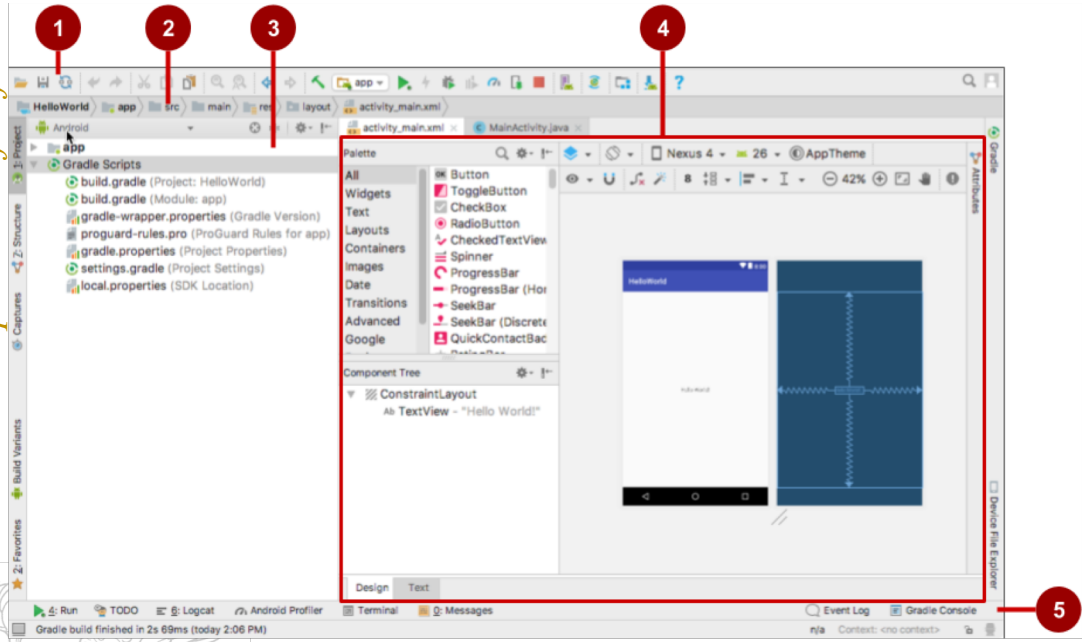
Első Android alkalmazásunk

- Android Studio
- „Hello World” alkalmazás
- Alap alkalmazás fejlesztés folyamat
- Alkalmazások futtatása virtuális s fizikai eszközökön

Android Studio

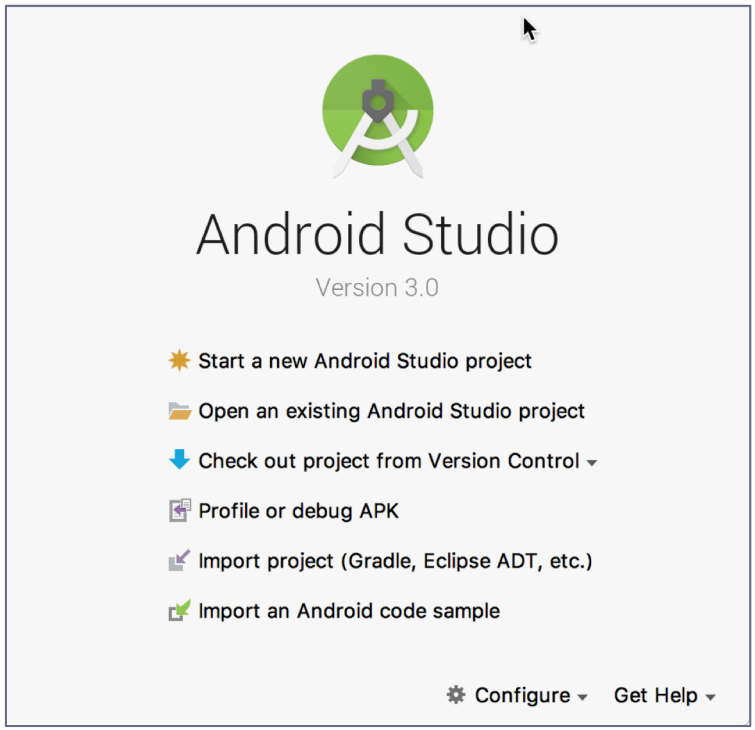
- Android IDE: Android integrált fejlesztői környezet
- Projekt és Android sablonok
- Képernyő tervező
- Teszt eszközök
- Gradla alapú build folyamat
- Log konzol és hibakereső
- Emulátorok

Android Studio interfész

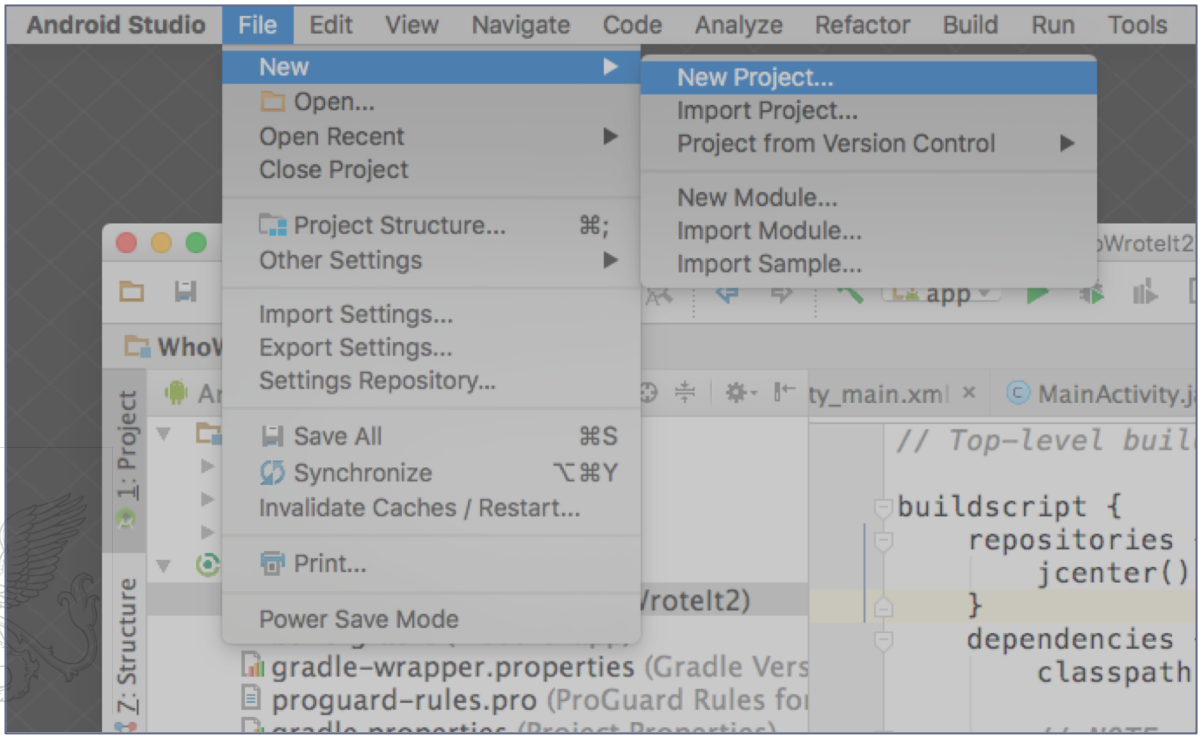


1. Eszköztár
2. Navigácós mező
3. Projekt áttekintés
4. Szerkesztő
5. Fülek és egyébb lapok

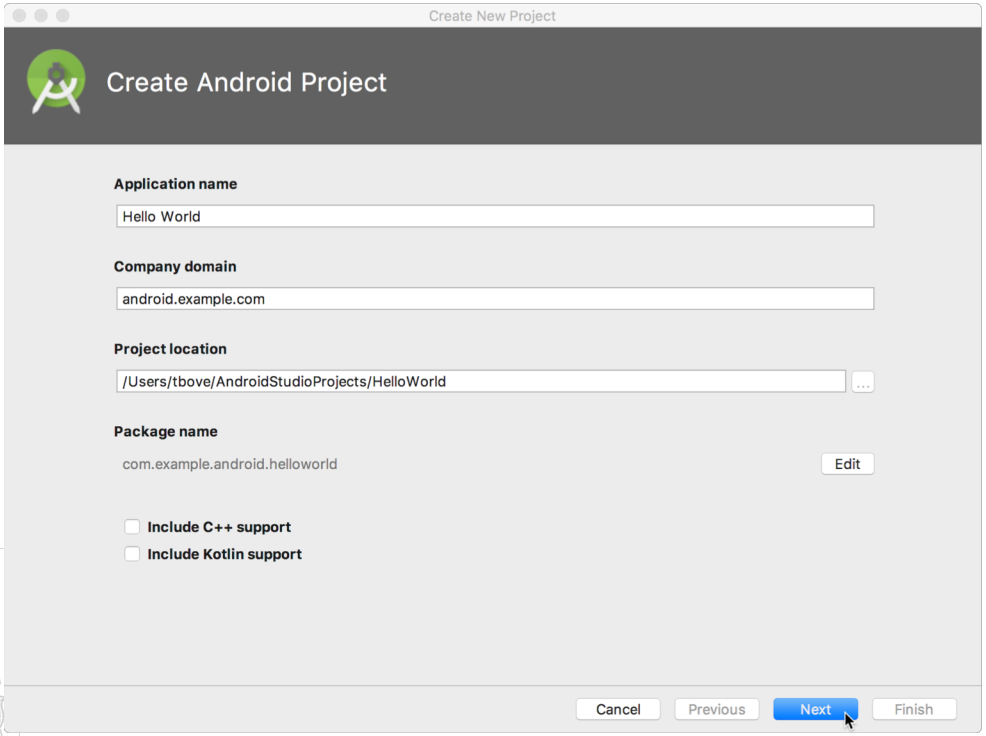
Android alkalmazás készítése



Projekt létrehozása

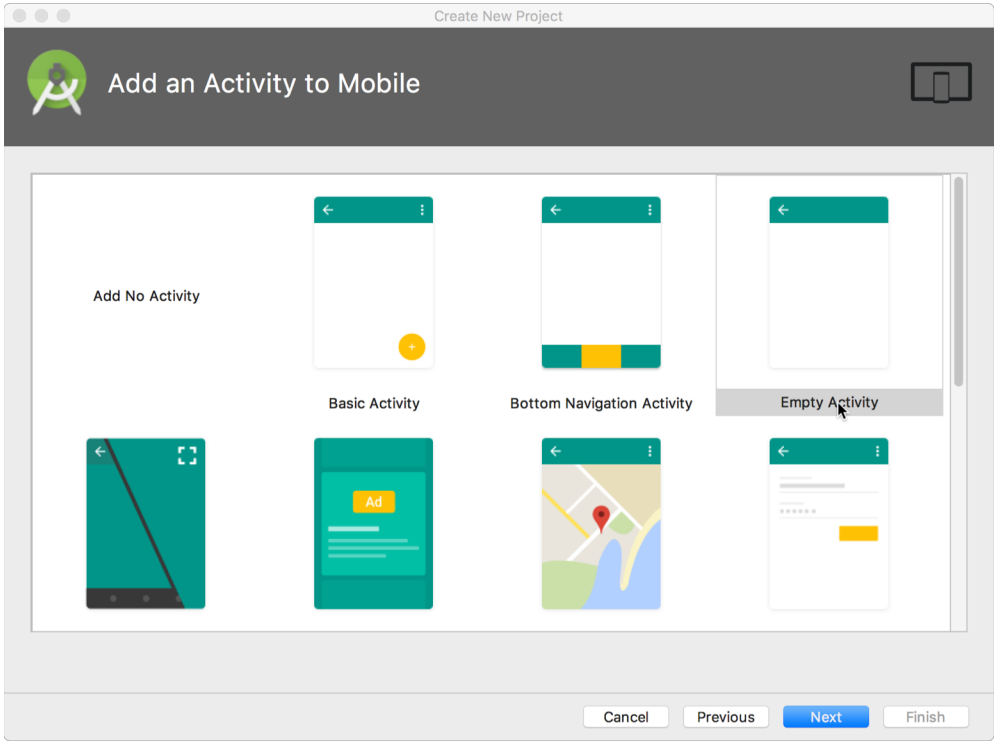


Alkalmazás elnevezése



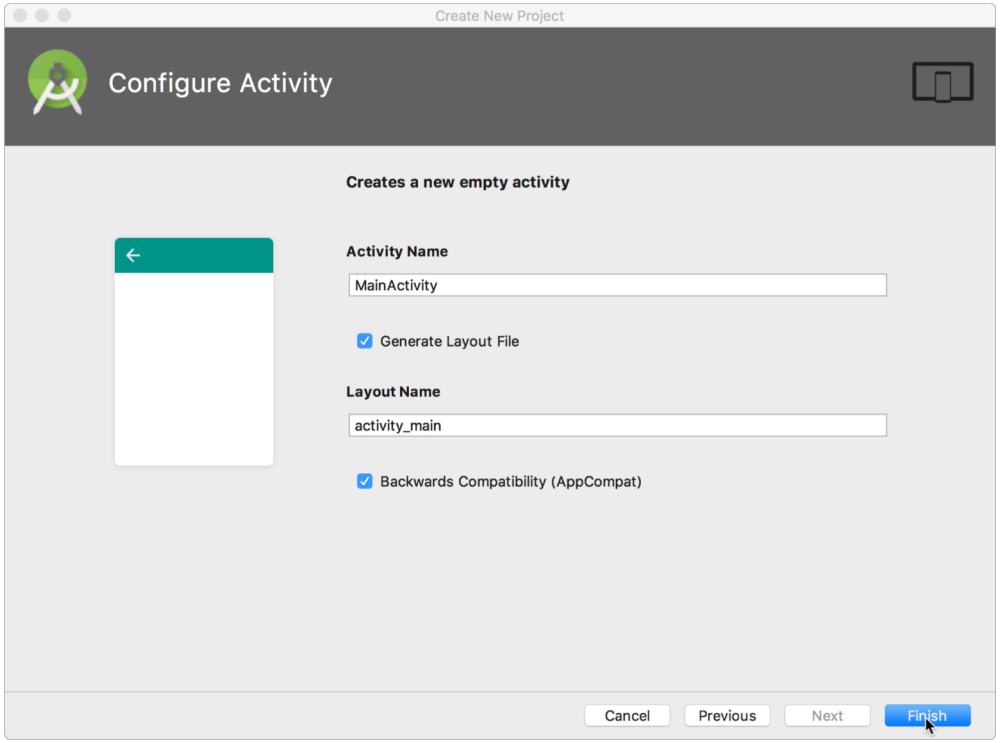
Sablon kiválasztása

- Válasszuk ki a sablonok közül a specifikus vagy álltalános aktivitásokat.



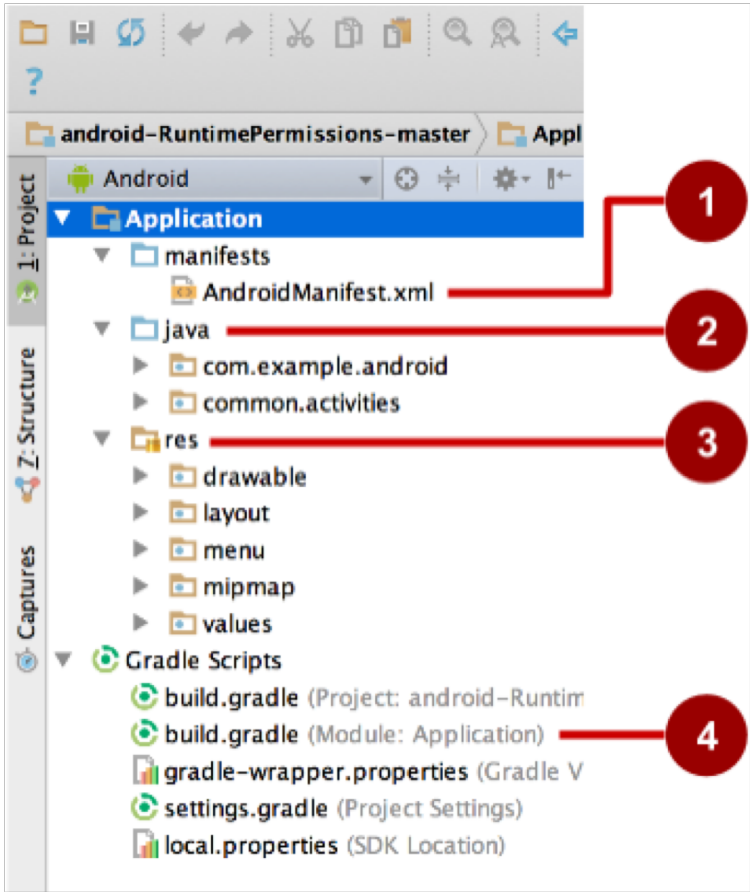
Nevezzük el az aktivitást

- Legjobb gyakorlat:
  - Nevezzük a fő aktivitást: MainActiivity-nek
  - A képernyőt: activity_amin-nek
- Használjuk az AppCompat-ot
- Generáljuk a képernyő fájlt



Projekt mappák

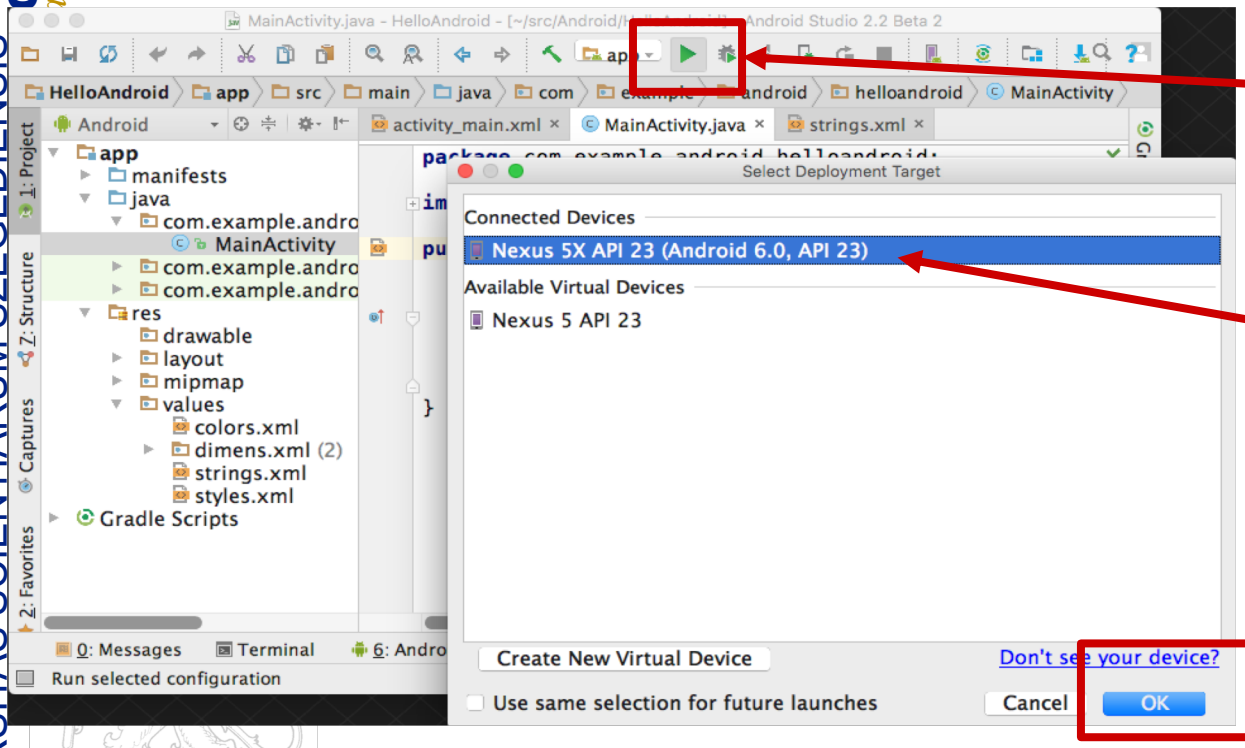
- **manifests** – Android leíró melyet a futtatási környezet olvas
- **java** – Java forrásfájlok és csomagok
- **res** – Erőforrások (XML), képernyők, ...
- build.gradle – Gradle build fájlok



Gradle build rendszer

- Az Android studio modern build alrendszere
- build.gradle
  - project
  - module
  - settings
- Az alapokhoz nem szükséges a Gradla alaposabb ismerete
- https://gradle.org/

Alkalmazás futtatása



1. Run

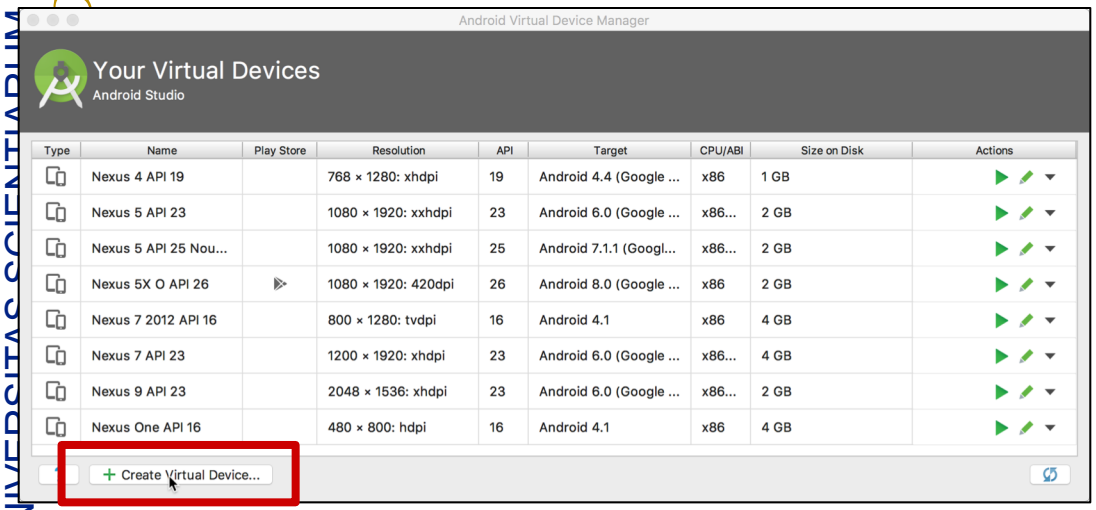
2. Válasszuk ki a virtuális vagy fizikai eszközt

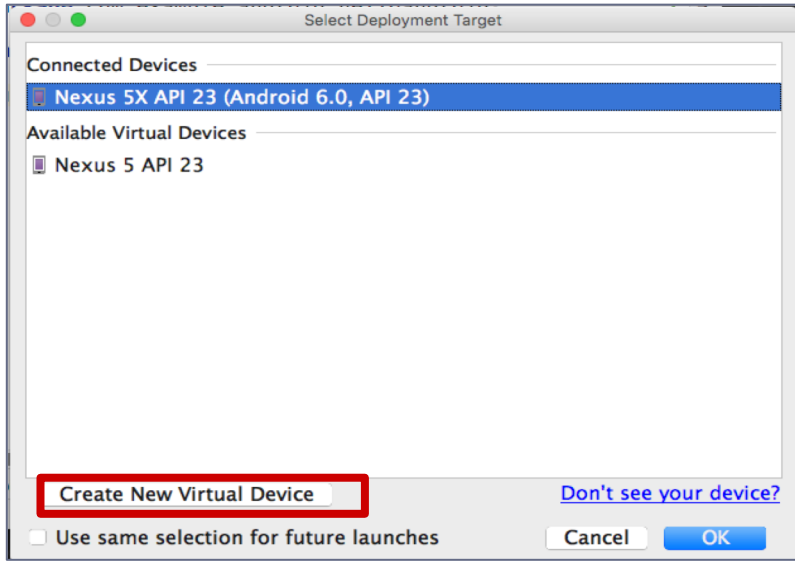
3. OK

Virtuális eszköz létrehozása

- Emulátorokkal tudjuk a különböző android verziók hatását megvizsgálni

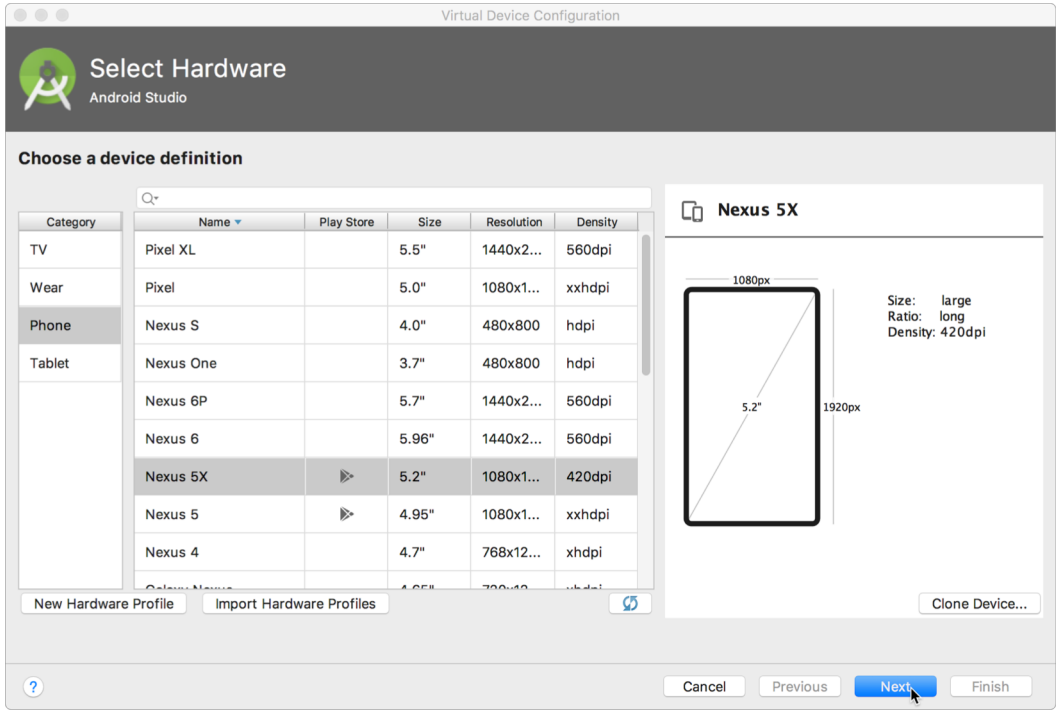
Tools > Android > AVD Manager vagy:



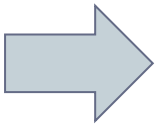


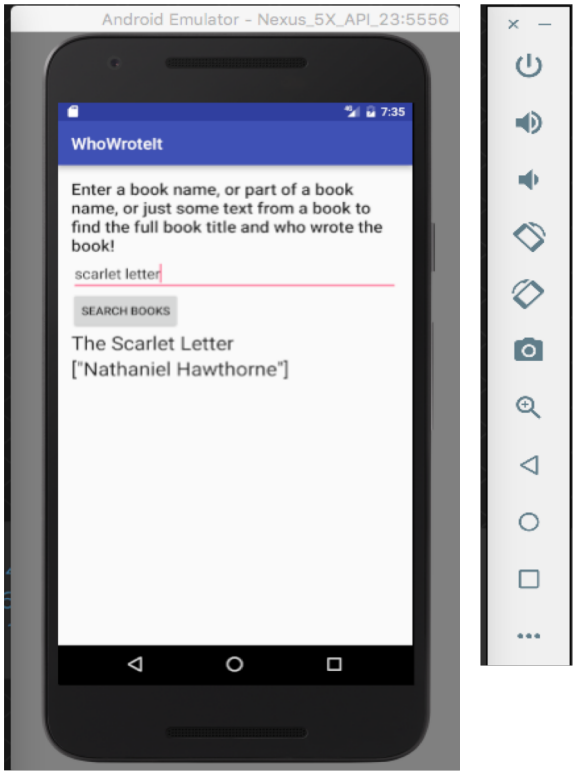
Virtuális eszköz beállítása

1. Válasszuk ki a HW-t
2. Válasszuk ki az Android verziót
3. Befejezés



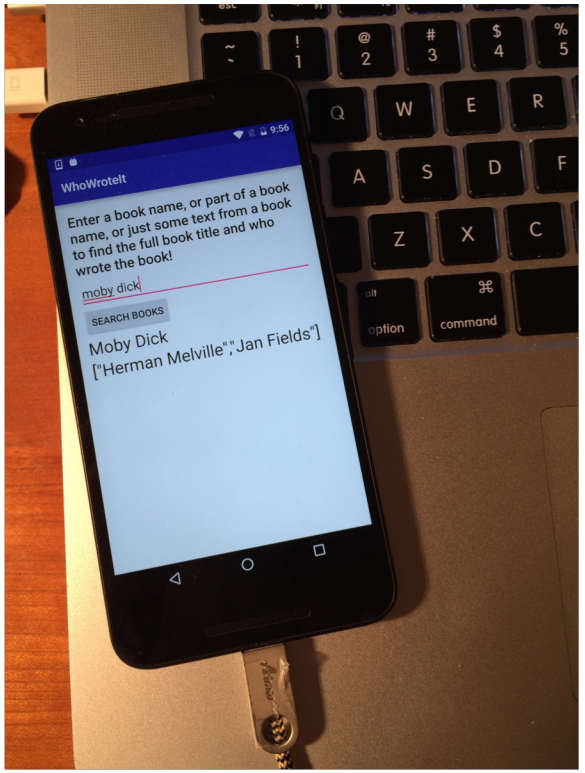
Futtassuk virtuális eszközö





Futtassuk fizikai eszközön

1. Developer Options bekapcsolása:
   a. **Settings > About phone**
   b. Nyomjuk meg a **Build number** hétszer
2. Kapcsoljuk be az USB Debugging-ot
   a. **Settings > Developer Options > USB Debugging**
3. Kössük össze a telefonunkat a számítógéppel



Windows/Linux további lépések:

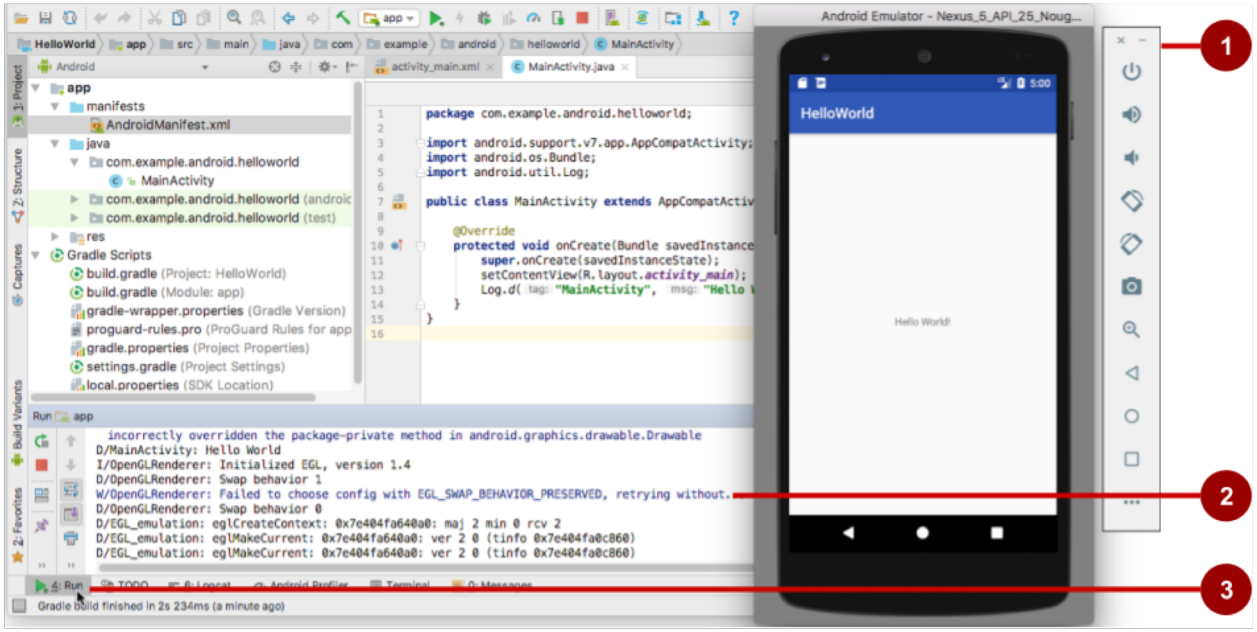
- Using Hardware Devices

Windows driver-e:

- OEM USB Drivers

Visszajelzés az alkalmazás futtatása közben

1. Alkalmazást futtató emulátor
2. Futtató képernyő
3. Run fül a futtató képernyő nyitására/zárására

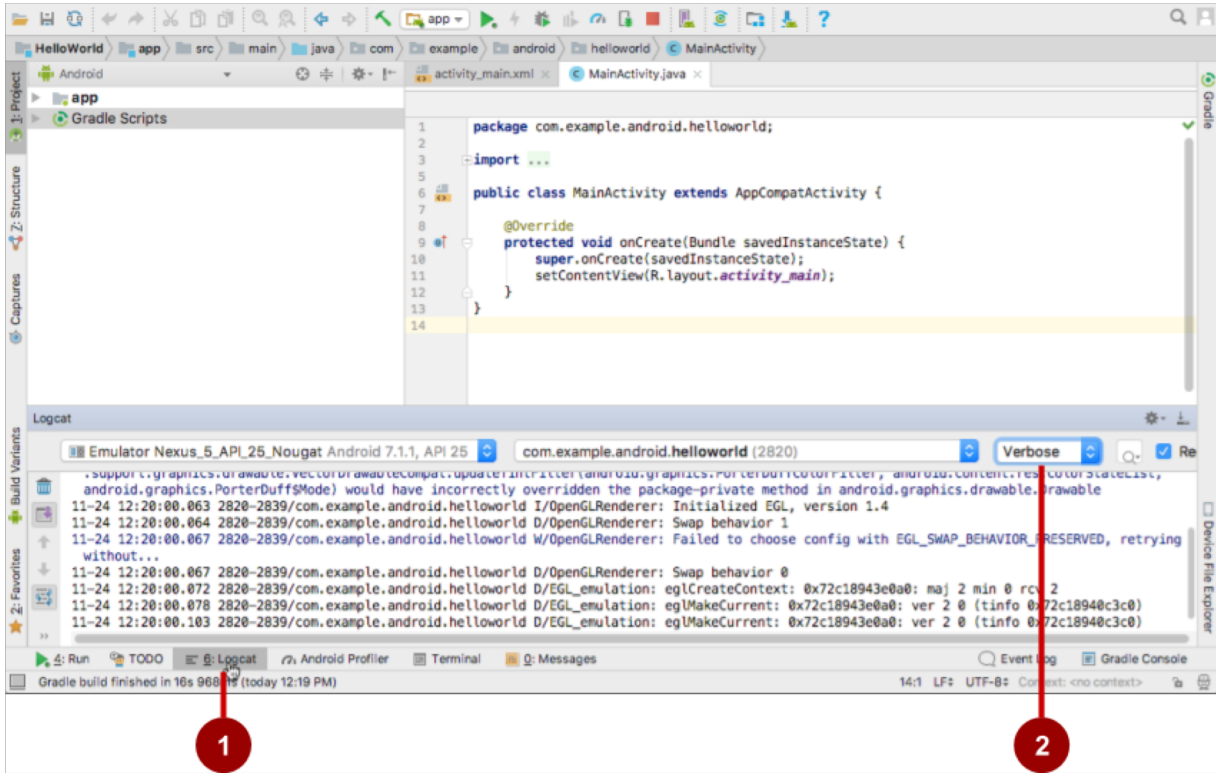


Logolás hozzáadása az alkalmazáshoz

- Az alkalmazás futása közben a **Logcat** ablak mutatja a kimenetet
- Az alkalmazásban log kifejezések segítségével tudunk ide kiíratni
- Szűrhetjük a listát – **Logcat** filetrs
- Kereshetünk címkékkel

Logcat ablak

1. Logcat ablak
2. Log szint menü



Logoló kifejezés

```
import android.util.Log;

// Use class name as tag
private static final String TAG =
    MainActivity.class.getSimpleName();

// Show message in Android Monitor, logcat
pane
// Log.<log-level>(TAG, "Message");
Log.d(TAG, “Creating the URI…”);
```

Összefoglaló

- Célok
- Követelmények
- Osztályzat
- Háttér
- Bevezető