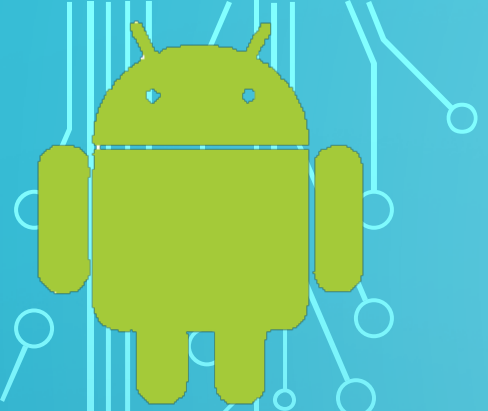


SZOFTVERFEJLESZTÉS ANDROID PLATFORMRA

DR. ISZÁLY GYÖRGY BARNA

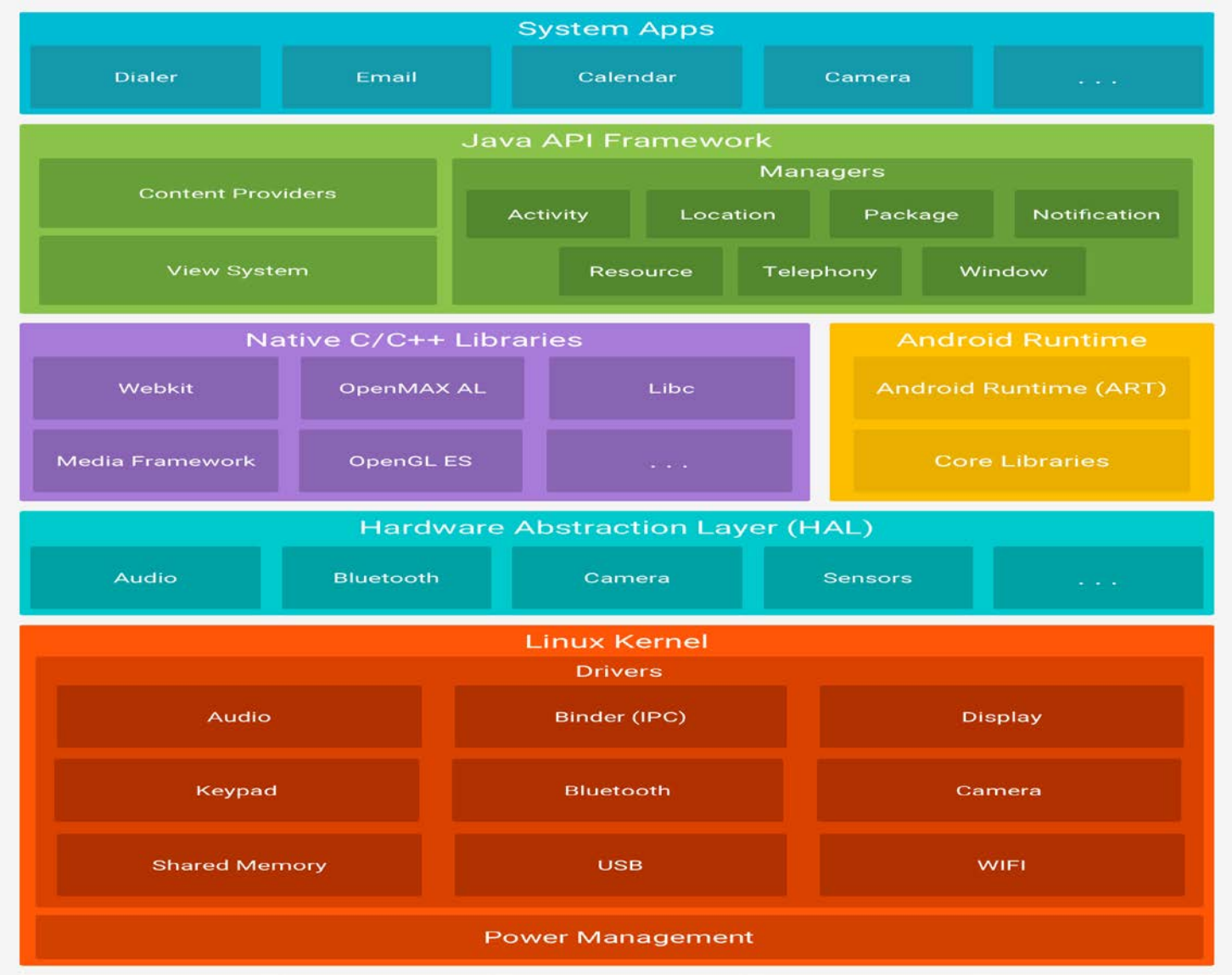

Learn to create
Android Applications!
ANDROID

ANDROID VERZIÓK JELENLEG

Android Platform/API Version Distribution

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.0 Ice Cream Sandwich	15	
4.1 Jelly Bean	16	99.8%
4.2 Jelly Bean	17	99.2%
4.3 Jelly Bean	18	98.4%
4.4 KitKat	19	98.1%
5.0 Lollipop	21	94.1%
5.1 Lollipop	22	92.3%
6.0 Marshmallow	23	84.9%
7.0 Nougat	24	73.7%
7.1 Nougat	25	66.2%
8.0 Oreo	26	60.8%
8.1 Oreo	27	53.5%
9.0 Pie	28	39.5%
10. Android 10	29	8.2%

AZ ANDROID PLATFORM FELÉPÍTÉSE



AZ ANDROID PLATFORM FELÉPÍTÉSE

- Linux kernel – a memória és a hardver eszközök kezelése, az alkalmazások felügyelete, a feladatok ütemezése, valamint ez felel a biztonságért rendszer és alkalmazás szinten is.
- Hardware Abstraction Layer (HAL) – alapvető interfészek a hardware elemek és a felsőbb szinten található Java API framework-ök között. Ezek könyvtári modulok (pl. kamera, bluetooth), melyek betöltődnek, ha az eszközöket akarjuk elérni.
- Programkönyvtárak – C/C++ nyelven íródtak
- Android futtatókörnyezetet – minden alkalmazás (Android 5.0 felett) a saját Android Runtime (ART) környezetben fut. DEX fájlok alacsony memóriahasználatú futtatását teszik lehetővé. Fontosabb tulajdonságai:
  - Java 8 futtató könyvátrak
  - Ahead-of-time (AOT) and just-in-time (JIT) fordító
  - Android 9 fölött konvertálja az app csomagokat Dalvik Executable formattá (DEX) ami sokkal kompaktabb kód
  - Dalvik virtuális gépet (DVM). Minden Android alkalmazás egy különálló DVM-ban fut (nem tudja tönkretenni a rendszert), melyet a párhuzamos futási képességre és a minimális memória felhasználásra optimalizáltak.
  - Optimalizált Garbage collector

ANDROID PLATFORM FELÉPÍTÉSE

- Native C/C++ könyvtárak – az ART, HAL és egyéb natív kódban írt elemek (pl. OpenGL ES) futtatásához
- Java API Framework – az erőforrásokhoz való hozzáférést biztosítja, és kiszolgálja a legfelső rétegben található alkalmazásokat. Részei: View System, Resource Manager, Notification Manager, Activity Manager, Contetn Providers. A fejlesztőknek teljes hozzáférése van ezekhez.
- Rendszer alkalmazások – a felhasználók számára elérhető alkalmazásokat tartalmaz

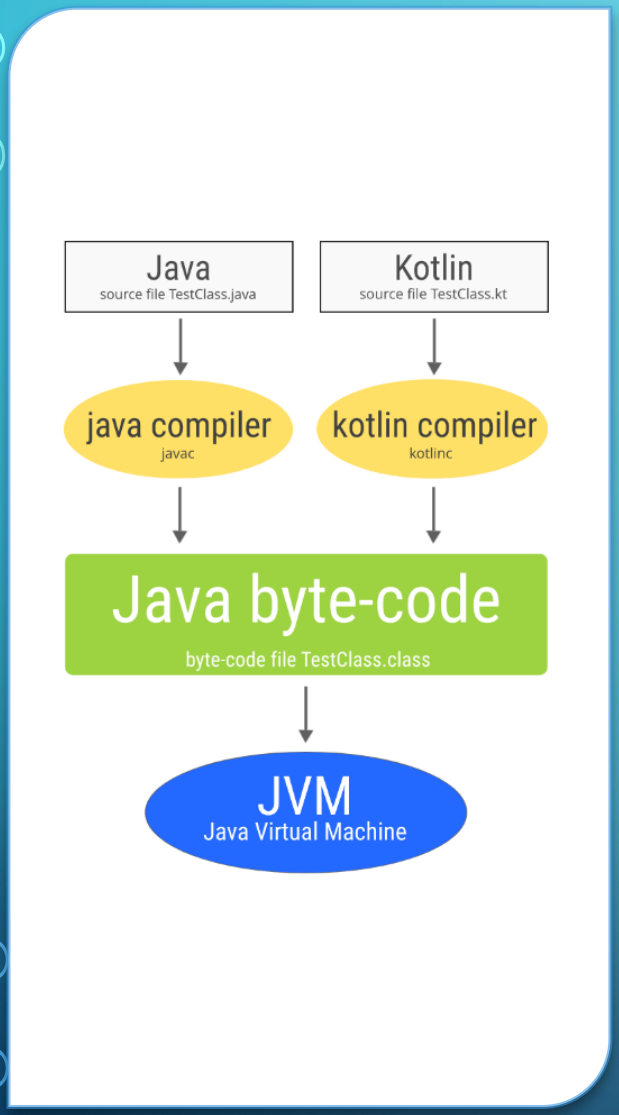
FEJLESZTÉS

- Java nyelven
  - Software Development Kit (Android SDK) fejlesztői csomag
- Kotlin nyelven
  - Software Development Kit (Android SDK) fejlesztői csomag
- Native kód (Android NDK)
  - C++ programozási nyelven
  - Native Development Kit fejlesztői csomag
- Android ADK (Accessory Development Kit)
  - Támogatás Android kiegészítő eszközök gyártásához (dokkoló, egészségügyi eszközök, időjárás kiegészítő eszközök stb.)
  - Android Open Accessory protocol (USB és Bluetooth)

FEJLESZTÉS

- A felhasználói felület és a program forráskódja teljesen elkülönül
  - Felhasználói felület kialakítása
    - Mint erőforrások jelennek meg a *res* alkönyvtárban
    - XML állományok
    - Forráskódból – lehetőleg kerülendő
  - Program forráskódja
    - Az *src* alkönyvtárban jelenik meg
- A kettő között a kapcsolatot az *R.java* állomány teremti majd meg
  - Ez a gen alkönyvtárban jelenik meg
  - Az sdk hozza létre
  - Lehetőleg ne nyúljunk bele!!!
- A manifest állományban és a gradle álloműnyaiban kell beállítani az alkalmazáshoz tartozó jogosultságokat

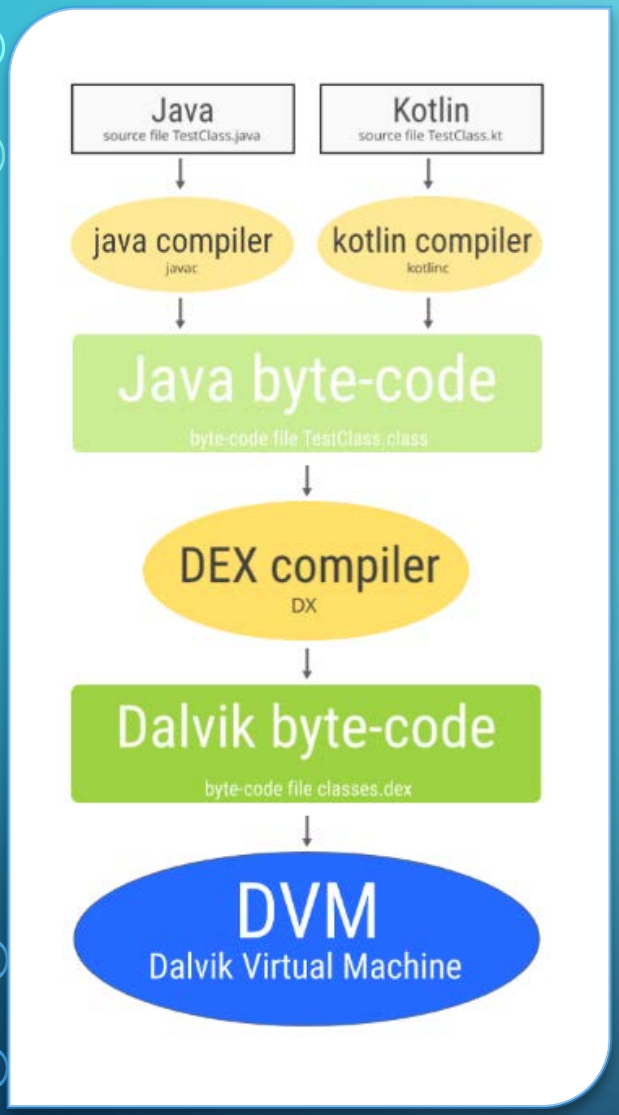
FORDÍTÁS



- A Java és a Kotlin fordítási eljárása szinte teljesen megegyezik

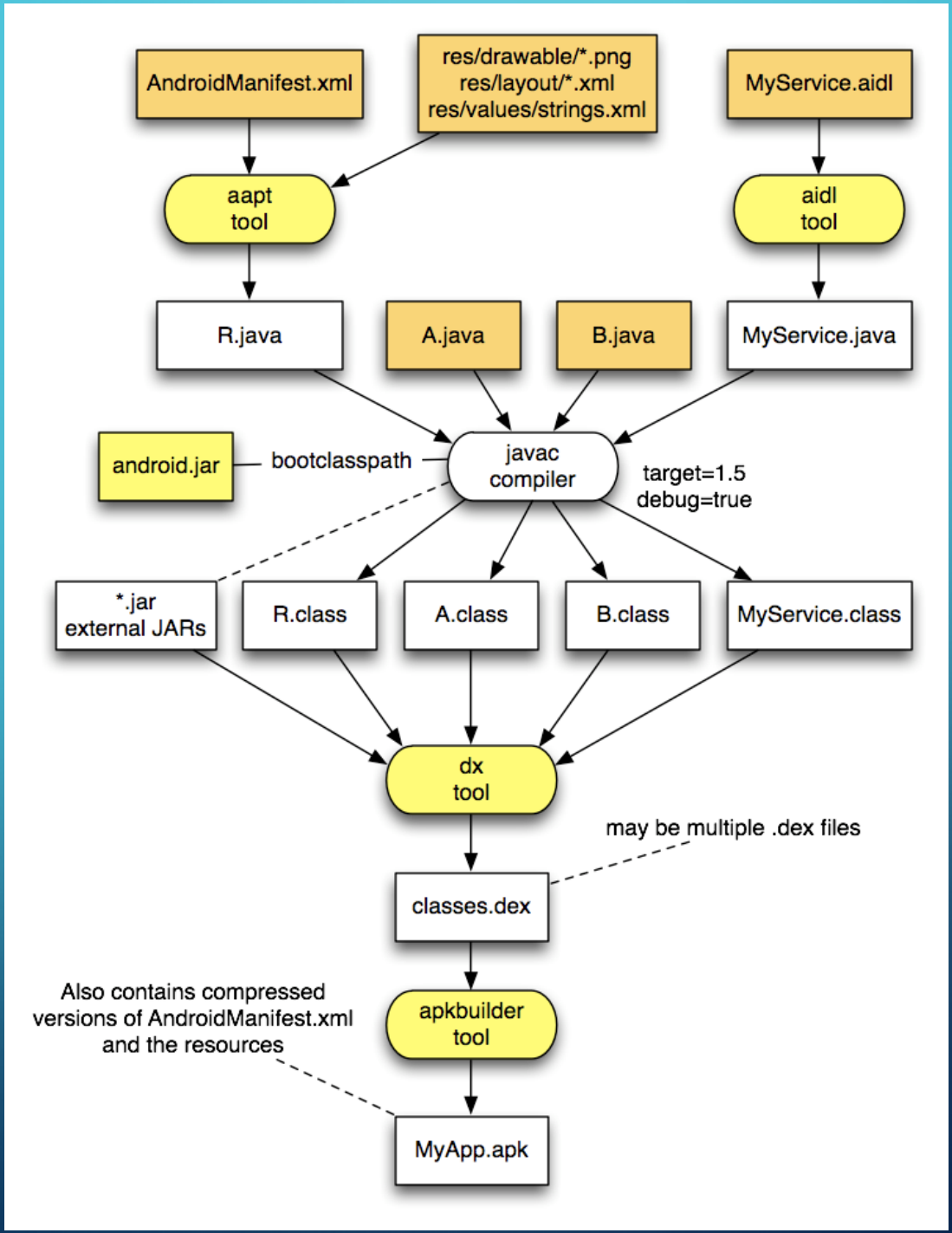
FORDÍTÁS

- Az Android nem dolgozik a JVM-el
- Az Androidban két speciális virtuális gép van elhelyezve
  - DVM (Dalvik Virtual Machine) - JIT (Just-In-Time) használ, azaz a fordítás a program végrehajtásakor történik, ezért lassabban indulnak az alkalmazások, mivel minden indításkor fordítás történik.
  - ART (Android Runtime) - AOT (Ahead-Of-Time) használ, azaz a fordítás a telepítéskor történik, így nem használ sok processzor erőforrást indításkor
- Ezek kompatibilisen tudnak futni a Dex byte kód alatt



FORDÍTÁS

- A Java byte kódok és a DVm között tehát van még egy DEX fordító, ami a Java bájt kódot Dalvik bájtkóddá alakítja
- A bájtkódból, valamint a nem lefordított (pl. képek) és a lefordított erőforrásokból a fordító létrehoz egy aláírás nélküli apk állományt.
- Ez még nem telepíthető sehová!!!
- Szükséges, hogy a fejlesztő digitális aláírással lássa el a programot.
  - Keytool utility segítségével megoldható
  - Fejlesztéskor automatikusan egy debug kulcs kerül elkészítésre
- Végeredmény egy telepíthető alkalmazás


AndroidManifest.xml
res/drawable/*.png
res/layout/*.xml
res/values/strings.xml
MyService.aidl
aapt
tool
aidl
tool
R.java
A.java
B.java
MyService.java
android.jar
bootclasspath
javac
compiler
target=1.5
debug=true
*.jar
external JARs
R.class
A.class
B.class
MyService.class
dx
tool
may be multiple .dex files
classes.dex
Also contains compressed
versions of AndroidManifest.xml
and the resources
apkbuilder
tool
MyApp.apk

AZ .APK ÁLLOMÁNY

- Ez egy tömörített állomány, amely a lefordított forráskódot és metainformációkat tartalmazza. Ezek a következőek:
  - META-INF alkönyvtár:
    - CERT.RSA: alkalmazástanúsítvány,
    - MANIFEST.MF: metainformációk kulcs-érték párokba rendezve,
    - CERT.SF: erőforrások listája az SHA-1 hash értékükkel
  - res alkönyvtár: az alkalmazásnál felhasznált erőforrásokat tartalmazza,
  - AndroidManifest.xml: a z alkalmazásra vonatkozó nevet, verziószámot valamint a jogosultságokat tartalmazza
  - classes.dex: itt találhatóak a lefordított osztályok
  - resources.arsc: erőforrásadatok találhatóak benne
- Az .apk állomány elindítása telepíteni fogja az alkalmazásunkat a céleszközön a *PackageManagerService szolgáltatás* segítségével

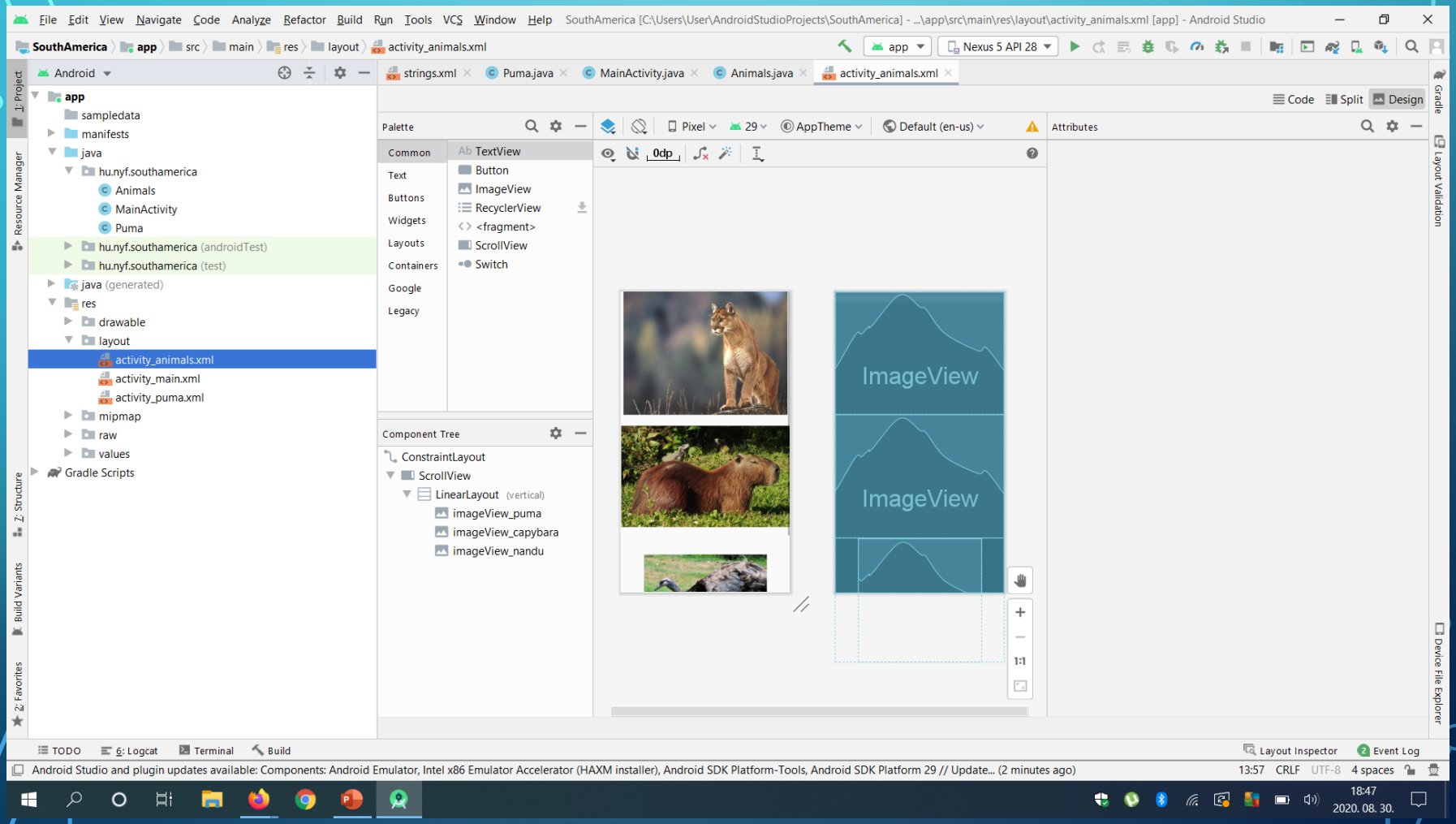
BIZTONSÁG?

- Az alkalmazások a Google Play nélkül is telepíthetőek
- Nincs alapos, hosszas ellenőrzés a Google Play-en
- Viszont telepítéskor jóvá kell hagyni a hozzáférési szinteket
- A telepítés felhasználói odafigyelést igényel!!!
- 2011 DroidDream – trójai alkalmazás
  - Információkat gyűjtött
  - A Google lépet közbe

FEJLESZTŐI KÖRNYEZET

- Java SE Development Kit (JDK)
- Android Software Development Kit (SDK) – célszerű szóköz mentes elérési úttal telepíteni!!!
  - SDK Manager – letölthetjük a számunkra szükséges Android verziókat
- AVD Manager – a kiválasztott verziónak megfelelő emulátort hozhatjuk létre
- Emulátor – képes a teljes Android rendszert szimulálni.
  - telnet localhost <portszám>
  - Beérkező telefonhívás szimulálása: gsm call <telefonszám>
- Android Studio – ennek telepítésével gyakorlatilag a többit is telepítjük

ANDROID STUDIO



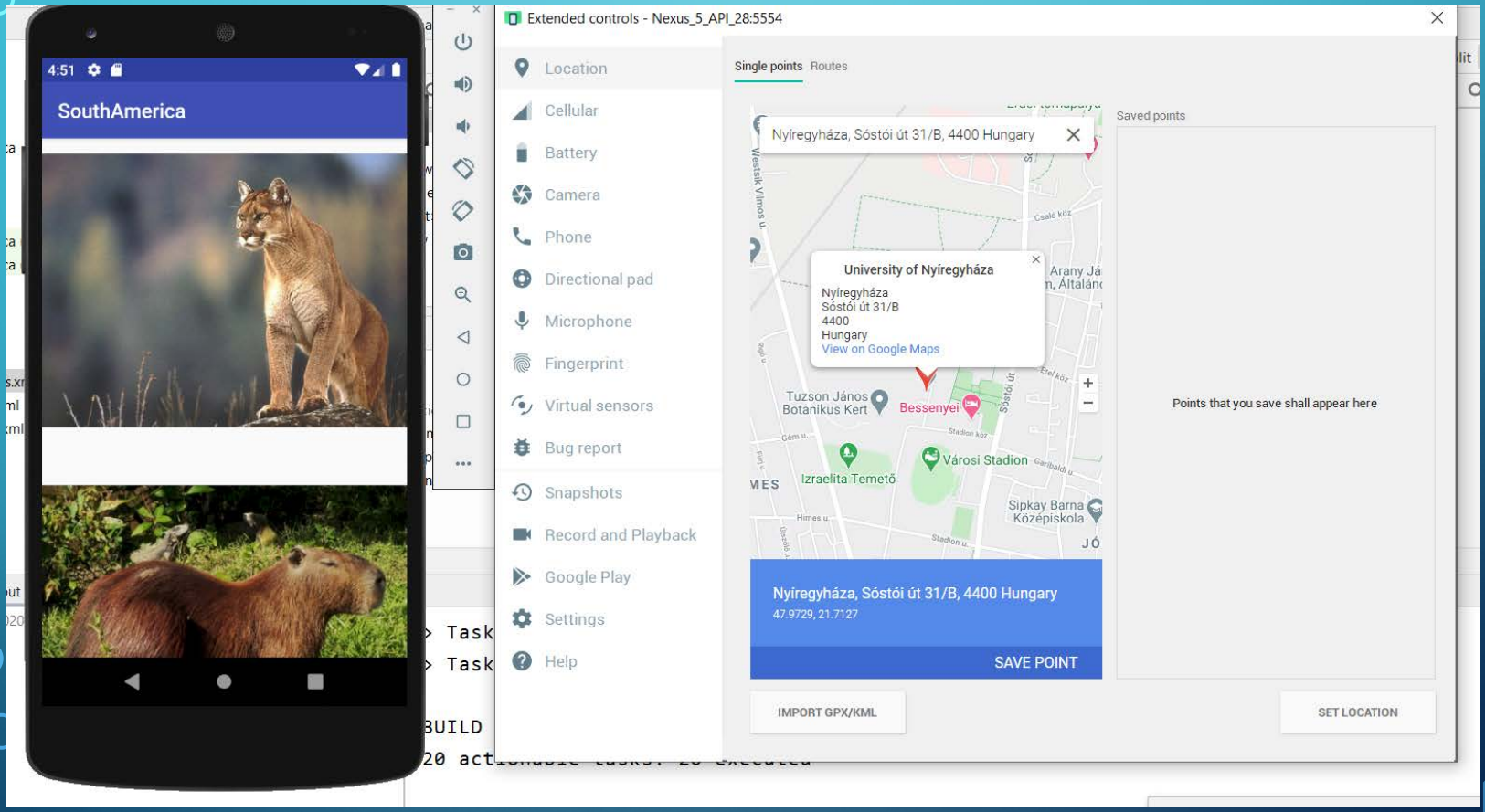
ANDROID STUDIO

- Nem árt, ha van egy erős vas az Android Studio alatt ☹
- Funkciók
  - Vizuális tervező felület
  - APK analizáló
  - Emulátor – a készülék szinte minden tulajdonságát képes emulálni
  - Inteligens kódszerkesztő
  - Rugalmas fordítási rendszer – több eszközre is egyszerűen testreszabhatóak az alkalmazások
  - Valós idejű monitorozás

GYORSBILLENTYŰK

- Lásd: https://developer.android.com/studio/intro/keyboard-shortcuts

EMULÁTOR


4:51
SouthAmerica
Extended controls - Nexus_5_API_28:5554
Location
Cellular
Battery
Camera
Phone
Directional pad
Microphone
Fingerprint
Virtual sensors
Bug report
Snapshots
Record and Playback
Google Play
Settings
Help
Single points
Routes
Nyíregyháza, Sóstói út 31/B, 4400 Hungary
University of Nyíregyháza
Nyíregyháza
Sóstói út 31/B
4400
Hungary
View on Google Maps
Tuzson János Botanikus Kert
Bessenyei
Izraelita Temető
Városi Stadion
Sipkay Barna Középiskola
Nyíregyháza, Sóstói út 31/B, 4400 Hungary
47.9729, 21.7127
SAVE POINT
Saved points
Points that you save shall appear here
IMPORT GPX/KML
SET LOCATION

EMULÁTOR

- Úgy viselkedik mint egy telefon
- Az összes beépített alkalmazás elérhető rajta
- Ha Google szolgáltatásokat szeretnénk igénybe venni az emulátoron (térkép), akkor a GoogleAPIs API szintet kell neki megadnunk

ANDROID DEBUG BRIDGE – ADB

- Az Android platform-tools alkönyvtárában található
- A konzol segítségével vezérelehetjük az Android eszközeinket
- Ha lecsatlakozik az emulátorunk a fejlesztői környezettel, ezzel lehet újra csatlakoztatni hozzá
  - adb kill-server
  - adb start-server
- Csatlakoztatott emulátorok/eszközök listázása:
  - adb devices
- Shell elérése
  - adb shell
- LogCat elérése
  - adb logcat

AZ ALKALMAZÁSOK FUTTATÁSI KÖRNYEZETE

- Az alkalmazások csak olyan rendszerszolgáltatáshoz férhetnek hozzá, amelyre engedélyt kaptak
- Minden alkalmazás szeparált környezetben fut *(virtual sandbox)*
- Az alkalmazások különböző rendszerazonosítók *(Linux user ID és group ID)* alatt futnak
- Az alkalmazások saját Linux processben, folyamatban futnak
- Minden alkalmazás olyan tanúsítvánnyal van aláírva, amelynek a privát kulcsa az alkalmazás fejlesztőjénél található

A KOMPONENSEK

- Minden Android alkalmazás egy, vagy több komponensből épül fel
- Az alkalmazás ugyanabból a komponensből is tartalmazhat többet.
- A komponensek különböző belépési pontokat biztosítanak a programunk felé a rendszer számára.
- Nem minden komponens biztosít azonban a felhasználó számára belépést a programba.
- Egy új komponens indítása előtt a rendszer megnézi, fut-e már a komponenst tartalmazó alkalmazás egy processze.
  - Ha igen, alapesetben ehhez a processzhez rendeli a létrehozandó komponenst.
  - Ha nem, akkor elindítja az alkalmazást, majd példányosítja a komponenshez szükséges osztályokat.

ACTIVITY

- Egy felhasználói felülettel rendelkező képe a programnak
- Egy alkalmazás több Activityt is tartalmazhat
- Más alkalmazáshoz tartozó Activity is meghívható alkalmazásunkból
- Leggyakrabban teljes képernyő méretű, de lehet úszó ablakban vagy beágyazott ablakban, sőt Fragment-ben is megjeleníteni

SERVICE

- Nem rendelkezik önálló felülettel.
- Általában egy háttérfolyamat, vagy elhúzódó tevékenység kiszolgálására alkalmas.
- Service behívhat Activity-ket a futása közben szükség esetén.
- Alapesetben az őt kiszolgáló folyamat fő szálán fut.

CONTENT PROVIDER

- Egy adatforrás (filerendszerben, SQLite adatbázisban, weben, stb.) kezelését megvalósító tartalomszolgáltató komponens.
- A tartalomszolgáltató engedélyével bármilyen más alkalmazás hozzáférhet ezekhez az adatokhoz, akár azok lekérdezése, akár azok megváltoztatása a cél.
- Az Android tartalmaz beépített providereket, többek között képek, hanganyag, videók, vagy névjegyzék-adatok megosztása érdekében.

BROADCAST RECEIVER

- Az operációs rendszer alacsony szintű eseményeire iratkozhatunk fel vele
- Broadcast események:
  - képernyő kikapcsolódása
  - az akkumulátor alacsony töltöttségi szintje
  - bejövő GSM hívás
  - Stb.
- Saját Broadcast is írható
- Esemény bekövetkezésekor az Android megvizsgálja, mely alkalmazásoknak van olyan broadcast receiver komponense, ami az adott eseményben érintett.
- Ha ilyeneket talál, elindítja őket.
- Önmagában a komponens ugyan nem rendelkezik saját felhasználói felülettel, de természetesen elindíthat más komponenseket, vagy megjeleníthet úgynevezett notification-öket.

BROADCAST RECEIVER

- A kapott broadcastek alapvetően kétfélék lehetnek:
  - Normál broadcast
    - *Context.sendBroadcast()* metódus küldi
    - aszinkron módon működnek
    - A receiverek rendezetlenül, gyakran egyszerre kapják meg őket
  - Rendszerezett (ordered) broadcast
    - *Context.sendOrderedBroadcast()* metódus küldi
    - A receiverek egy megadott sorrendben kapják meg a broadcast-et
    - Továbbadhatják egymásnak annak eredményét
    - Fel is függeszthetik a továbbadást. A továbbadás sorrendjét a megfelelő intent filter
    - Továbbadás sorrendjét az adott intent filter*android:priority* attribútuma határozhatja meg
- A komponensek indítását általában egy Intent kézbesítésével kezdeményezhetjük.

INTENT

- Az alkalmazások komponensei közötti adatcsere eszköze
- Különböző programok komponensei között is megvalósíthat kommunikációt
- Az operációs rendszeren keresztül történik a kézbesítése
- Nemcsak az elvárt, hanem a bekövetkezett események attribútumait is tartalmazhatja
- A megcélzott komponensek fajtája metódusfüggő, a következők szerint:
  - *startActivity()*, vagy *startActivityForResult()* metódusok esetében egy Activityt tudunk indítani
  - *startService()*, vagy *bindService()* metódusok segítségével Service komponenst indíthatunk, vagy kapcsolódhatunk hozzá
  - Különféle Broadcastek kezdeményezhetők egy Intent átadásával a következő metódusok valamelyikének:
    - *sendBroadcast()*
    - *sendOrderedBroadcast()*
    - *sendStickyBroadcast()*a

INTENT

- Alapvetően kétfélék:
  - Explicit Intentek
    - pontosan meghatározza a kívánt komponenst a *setComponent()*, vagy a *setClass()* metódusok segítségével
  - Implicit Intentek
    - nem nevezzük meg a kérés kiszolgálására legalkalmasabb komponenst
    - az operációs rendszerre bízzuk a döntést
    - egy Intent filter (szűrő) jelezheti a kompnens, hogy lehetősége van bizonyos adatokat fogadni, valamilyen eseményt vagy egy kérést kiszolgálni. (Manifest állomány)

MANIFEST ÁLLOMÁNY

- Az alkalmazást leíró XML formátumú fájl
- Feladatai
  - Az alkalmazás csomag neveit/névtereit írja le
  - Tájékoztatja a rendszert az alkalmazás részeiről, leírja az alkalmazás komponenseit.
  - Rögzíti a futtatáshoz szükséges követelményeket és jogosultságokat
  - A szoftver és hardver követelményeket is megadhatja
    - Pl.: Meghatározza az alkalmazás szükségleteit (kijelző mérete stb.)
  - Meghatározza az alkalmazás futtatásához szükséges (nem az Android frameworkhöz tartozó) API könyvtárakat. Pl.: Google Maps könyvtárai.

MANIFEST ÁLLOMÁNY PÉLDA

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="hu.nyf.southamerica">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
```

Egyedi csomagnév

Az indítási ikon, név és téma jelölése.

Az alkalmazás tulajdonságai és komponensei

MANIFEST ÁLLOMÁNY PÉLDA

```
<activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER"
/>
        </intent-filter>
    </activity>
    <activity android:name=".Animals" />
    <activity android:name=".Puma"></activity>
  </application>
</manifest>
```

A megvalósított komponens osztályának a neve

Ez az Activity a belépési pontja az alkalmazásnak

A komponens látható neve

BUILD.GRADLE (MODULE: APP)

```
pply plugin: 'com.android.application'
 android {
    compileSdkVersion 26
    defaultConfig {
        applicationId "hu.nyf.southamerica"
        minSdkVersion 14
        targetSdkVersion 26
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner
"android.support.test.runner.AndroidJUnitRunner"
    }
```

Hányas Android verzióval fordítod az alkalmazást

Mi az az azonosítója

Melyik legalacsonyabb Android verzión futhat

Hányas Android verzióra készíted az alkalmazást

A verzió kódja. Mindig egyel nő

Az általunk megadott verzió szám

BUILD.GRADLE (MODULE: APP)

```
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:26.1.0'
    implementation 'com.android.support.constraint:constraint-layout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
}
```

Fordítás típusa: publikálásra szánt

Milyen függőségei vannak az alkalmazásnak, amelyet fordításkor be kell építeni