

Android

Az Android platform bemutatása

Dr. Ekler Péter

peter.ekler@aut.bme.hu



Department of
Automation and
Applied Informatics

Oktatók

- EA: Ekler Péter
 - > peter.ekler@aut.bme.hu
- GY1 (Kedd 14:15): Kövesdán Gábor
 - > kovesdan.gabor@aut.bme.hu
- GY2 (Csütörtök 12:15): Pásztor Dániel
 - > pasztor.daniel@aut.bme.hu
- GY3 (Csütörtök 12:15): Pomázi Kriszitán
 - > pomazi.krisztian@aut.bme.hu
- GY4 (Csütörtök 16:15): **Gazdi László**
 - > gazdi.laszlo@aut.bme.hu

Tudnivalók

- Tárgy honlap
 - > <https://edu.vik.bme.hu/course/view.php?id=8581>
 - > <https://portal.vik.bme.hu/kepzes/targyak/VIAUAV21/>
- Neptun/Teams üzenet
- GitHub
 - > <https://github.com/peekler/Android-Kotlin-BME>
 - > <https://github.com/peekler/Android-BME> (Java - régi)

Tárgykövetelmények

- Heti 1 előadás és 1 labor
- *opcionális* Házi feladat, **de ajánlott**
- ZH: November 22-én, az előadás idejében
- Vizsgák: vizsgaidőszakban
- Elővizsga pótZH-val együtt pótlási héten
- Labor részvétel - 70%

Előadás

- Az Android platform részletes bemutatása
- Példákkal fűszerezett tananyag
- Előadás és tankönyv
- Plusz pont szerzési lehetőségek!
- mindenféle mobilos információk:
 - > Versenyek
 - > Mobil Klub
 - > „Expect the unexpected” ☺
- Neptun-ban szereplő e-mail cím aktualizálása!

Könyv

- Android-alapú szoftverfejlesztés (2012)



Kotlin

- Nyelv ismertető:
 - > <https://kotlinlang.org/>
- Kotlin Koan-ok gyakorolni:
 - > <https://kotlinlang.org/docs/tutorials/koans.html>
- Java-ról Kotlinra tutorial:
 - > <https://github.com/Zhuinden/guide-to-kotlin>

Labor

- 4 különböző labor
 - > Teams csatornák
- A laborok legalább 70%-án részt kell venni
- Az előző előadáshoz kapcsolódó feladat megoldása a laborvezető támogatásával
- Labor feltöltés tárgy oldalra (File->Export to zip)
 - > <https://edu.vik.bme.hu/course/view.php?id=7653>
- Labor végén érdemjegy (jóindulatú osztályzás ☺)
- Megajánlott jegybe beleszámít

Házi feladat

- A házi feladattal megajánlott jegyet lehet szerezni
- Nem kötelező, DE ajánlott!
- Jelentkezni a portálon:
 - > Rövid specifikáció
- Beadási határidő:
 - > Utolsó előtti hét vége
- Beadás:
 - > Tárgy oldalára kell feltölteni
- Tesztelés:
 - > Emulátor
 - > Eszközöket is tudunk biztosítani
- Legalább 5 előadás téma használata
- Megfelelő komplexitás: 15-30 óra
- Ha nem sikerül megajánlott jegyet szerezni, akkor is szerezhető maximum 20 pont a vizsgára (100 pontos vizsgát feltételezve)

Minta Házi Feladatok

- Játékok
- Szintax highlight-os Notepad két nyelv támogatásával
- Arcfelismerős contact szerkesztő/telefonhívás/stb.
- Sports tracker (idő/távolság/sebesség mérés, GPS, grafikon), háttérben futó szolgáltatás
- Szenzor alapú megoldás/játék
- Kiterjesztett valóság, kamera kezelés
- Adatkezelő, több lista, komplex adatbázis (pl.: túrák, fejlesztési feladatok nyomkövetése, kategorizálással, idő naplózással stb.)

- Az előadás idejében
 - > November 22, kedd 12-14 (előadás idejében)
- Elméleti és gyakorlati feladatok
- Tipikusan az előadáson elhangzott kérdésekkel és feladatokból válogatunk
- 40%-ba beleszámít a vizsgás jegybe
- 40%-ba beleszámít a megajánlott jegybe

Vizsga

- Vizsgaidőszakban (elővizsga lehetőség szerint lesz)
- Elméleti és gyakorlati feladatok
- Tipikusan az előadáson elhangzott kérdésekből és feladatokból válogatunk
- Érdemjegy: 40% ZH + 60% Vizsga
- Inkább a megajánlott jegy legyen a cél ☺

Megajánlott jegy

- Laborvezetővel egyeztetett specifikáció
- Legjobb 6 labor eredménye
- Megajánlott jegy: 40% ZH + 20% labor
(legjobb 6) + 40% HF

Labor - GIT

- Labor anyagok GIT-en
 - > <https://github.com/VIAUAV21/Labor>
- GitHub account létrehozása szükséges
- Regisztráció szükséges az eléréshez
 - > Gazdi László készít a weboldalon egy űrlapot



Mi nem igaz az Android tárgyra?

- A. Megajánlott jegy szerezhető Házi Feladattal!
- B. A laborok során lehetőség van az előadáson elhangzottak gyakorlására!
- C. A házi feladatnak legalább 5 technológiát érintenie kell!
- D. Kötelező írásbeli vizsgát írni!

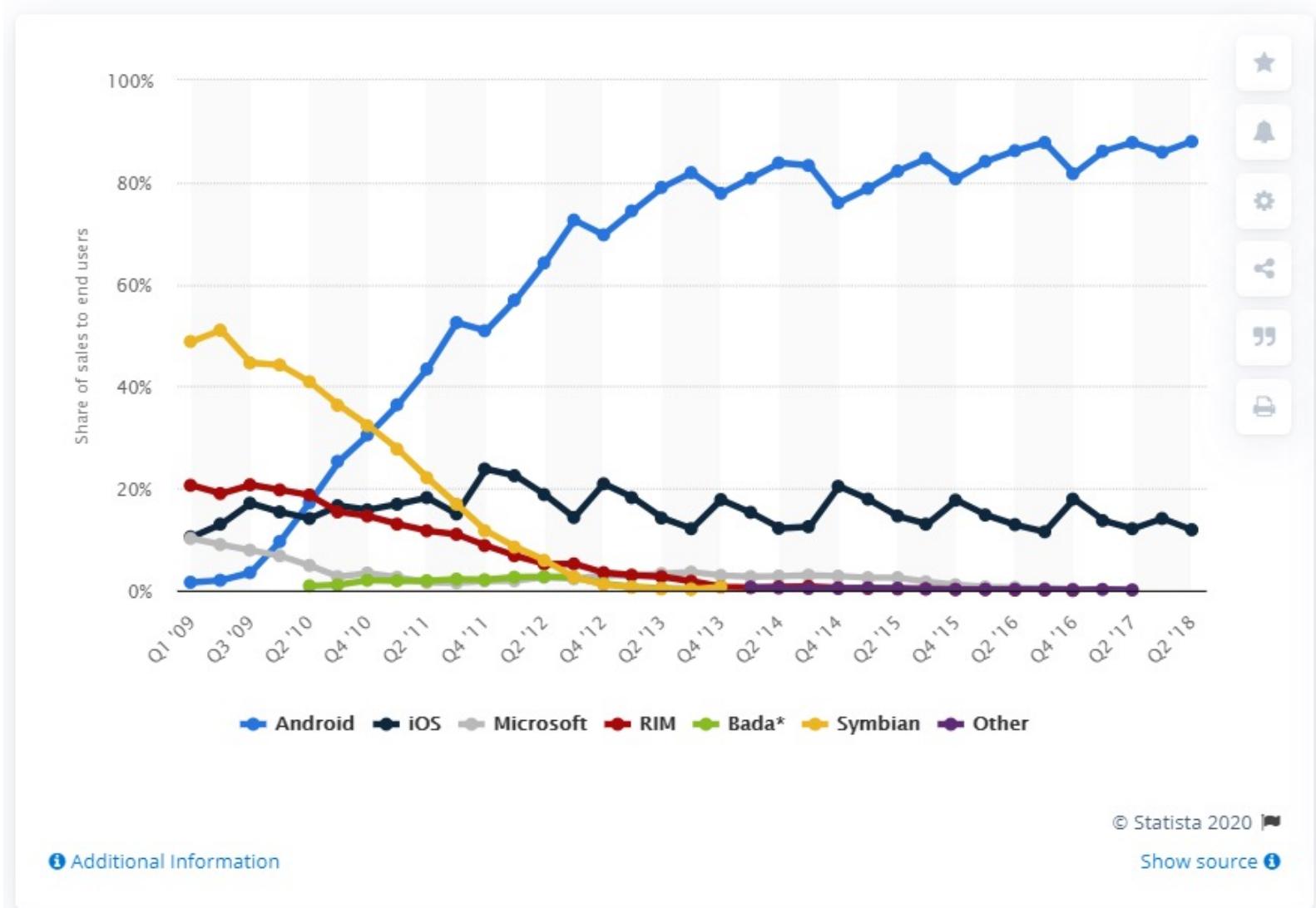
Tartalom

- Bevezetés
- Mi is az Android?
- Milyen készülékeket támogat az Android?
- Az Andriod SDK fejlett képességei
- Miért fejlesszünk mobil alkalmazásokat és miért Androidra?
- Az Android fejlesztőkörnyezet és SDK bemutatása
 - > Telepítés
 - > Használat
 - > Eszközök
- Első Android alkalmazás
 - > View Binding
 - > Data Binding
- Összefoglalás

Bevezetés

- Okostelefonok térhódítása
- Táblagépek terjedése
- Háttértár növekedése
- Hálózat sebességének növekedése
- Adatforgalom árának csökkenése
- Alkalmazásboltok megjelenése

Piaci rész - Android

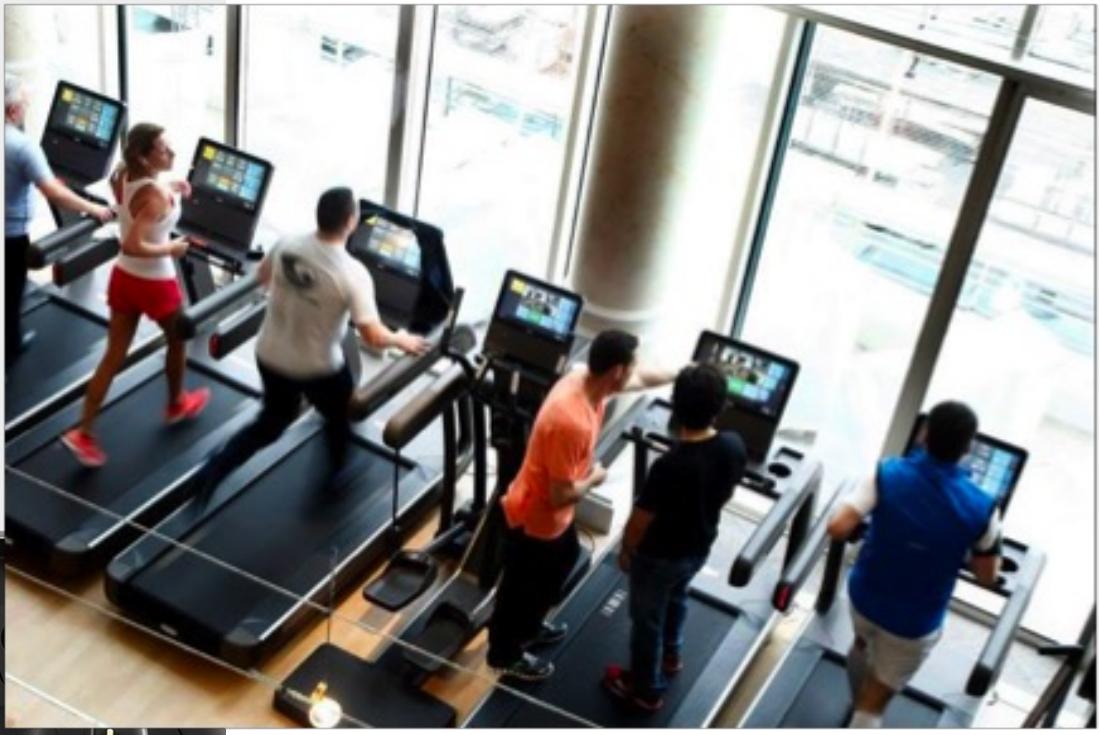
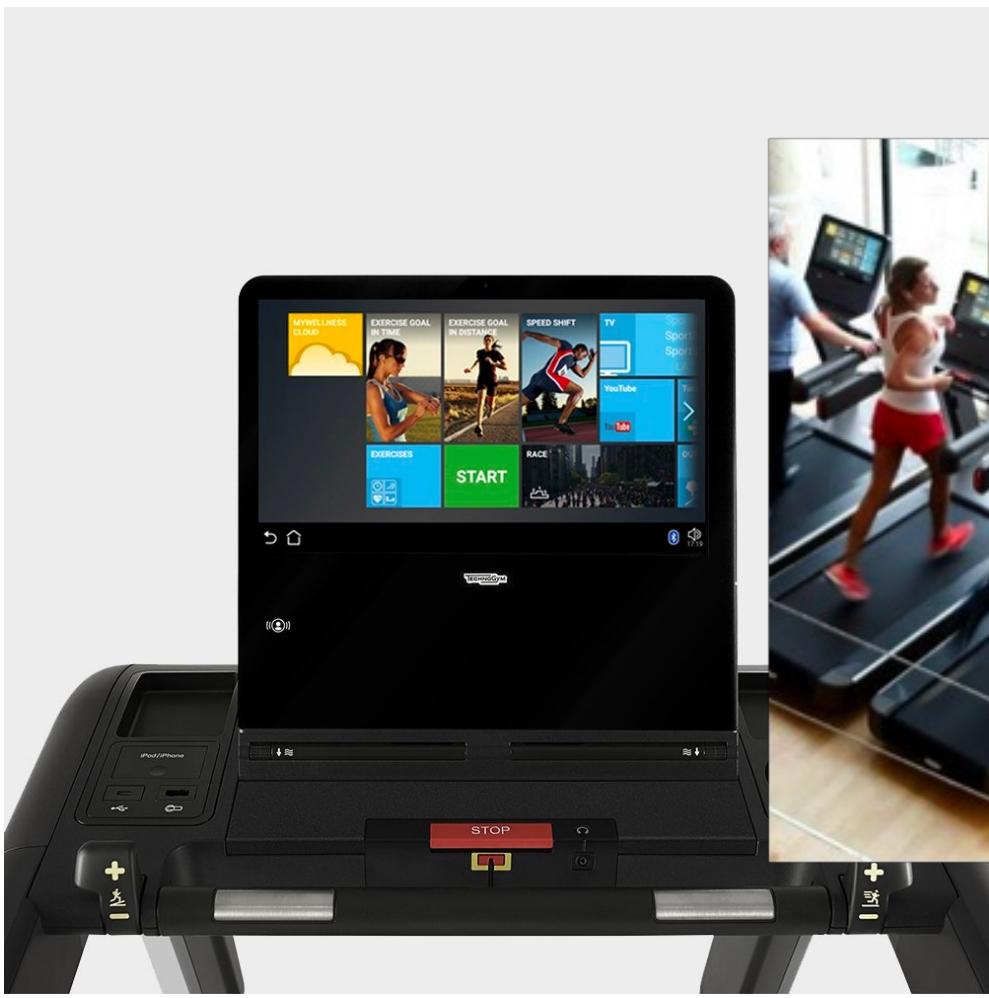


Android eszközök 1/2

- Mobiltelefon és a Tablet gyártók
- Gépjárművek fedélzeti számítógépét és navigációját szállító cégek
- Android Wear
- Ipari automatizálás irányából is
- minden olyan helyen kényelmes az Android
 - > Alapvetően kicsi a kijelző (Google TV megoldás)
 - > Más jellegű erőforrások
 - > Az adatbevitel nem tipikusan egérrel és/vagy billentyűzettel történik
 - > Android@Home



Android eszközök 2/2



Android verziók

- Fontos a verziók nyomon követése
- Egyes verziók között komoly API-beli különbségek lehetnek
- Törekednek a visszafele kompatibilitásra, de lehetnek éles szakadékok (pl. 3.0)
- Fejlesztés előtt alaposan gondoljuk át a támogatott minimum verziót
- Verzió kódnev: valamilyen édesség ☺

Android verziószámok



- Android 1.0 – 2008. October
- Android 1.1 – 2009. February
- Android 1.5 (Cupcake) – 2009. April
- Android 1.6 (Donut) – 2009. September
- Android 2.0 and 2.1 (Eclair) – 2009. October
- Android 2.2 (Froyo) – 2010. May
- Android 2.3 (Gingerbread) – 2010. December
- Android 3.0-3.2 (Honeycomb) – 2011 January-July
- Android 4.0 (Ice Cream Sandwich) – 2011. October
- Android 4.1 (Jelly Bean) – 2012. July
- Android 4.2 (Jelly Bean) – 2012. November
- Android 4.3 (Jelly Bean)
- Android 4.4 (KitKat)
- Android 5.0, 5.1 (Lollipop)
- Android 6.0 (Marshmallow)
- Android 7.0, 7.1 (Nougat)
- Android 8.0, 8.1 (Oreo)
- Android 9.0 (Pie)
- Android 10 (Q)
- Android 11
- Android 12

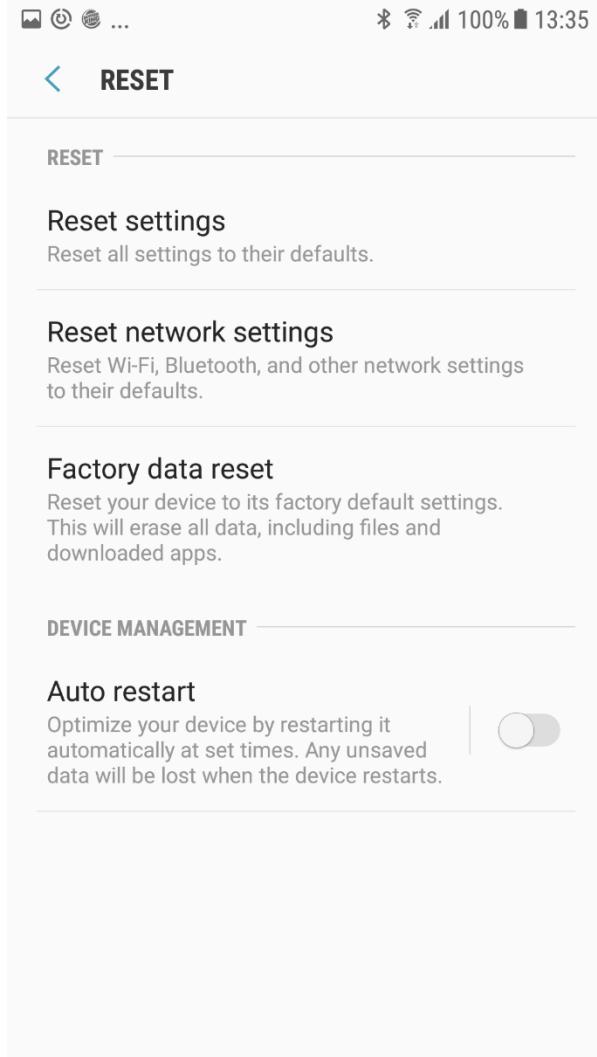
Android verziók

A B C D E F G H I J
K L M N O P 10 11
12 13 14 ...

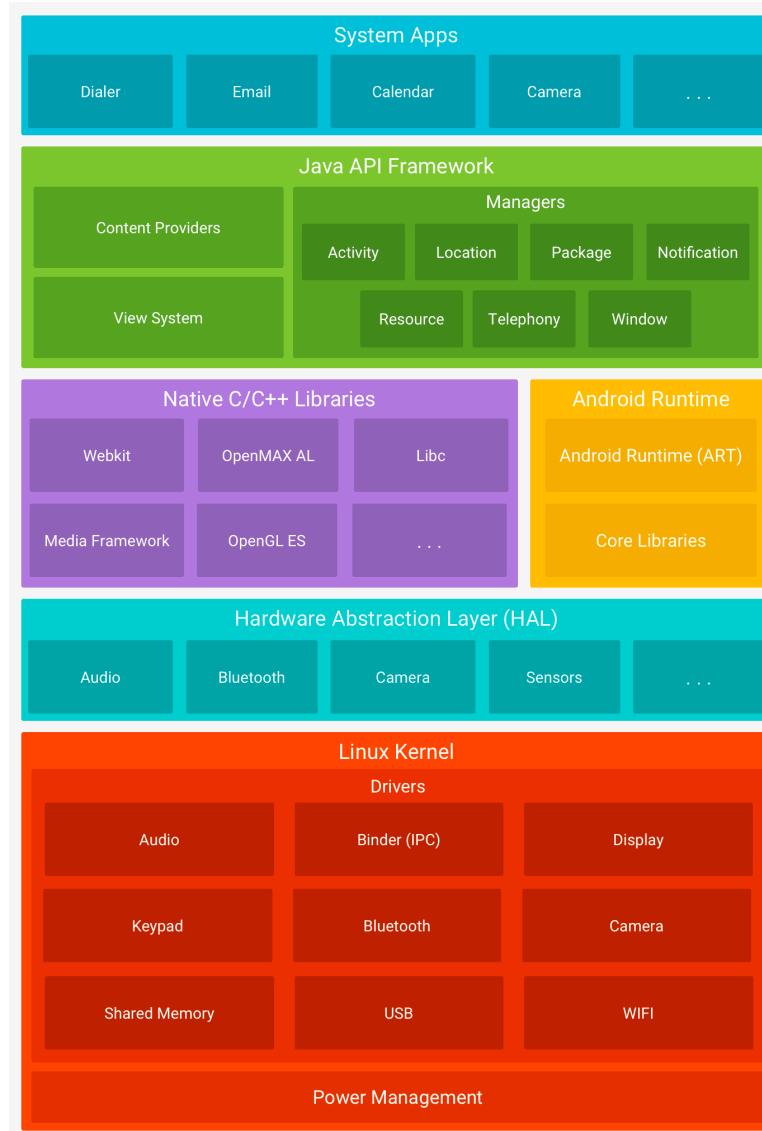
Még van ~∞ évünk ☺



Érdekesség ☺



Az Android platform szerkezete



Szoftverfejlesztési eszközök Android platformra

- **Android SDK (Software Development Kit):**
 - > Fejlesztő eszközök
 - > Emulátor kezelő (AVD Manager)
 - > Frissítési lehetőség
 - > Java, Kotlin
- **Android NDK (Native Development Kit):**
 - > Lehetővé teszi natív kód futtatását
 - > C++
 - > Eclipse plugin
- **Android ADK (Accessory Development Kit):**
 - > Támogatás Android kiegészítő eszközök gyártásához (dokkoló, egészségügyi eszközök, időjárás kiegészítő eszközök stb.)
 - > Android Open Accessory protocol (USB és Bluetooth)



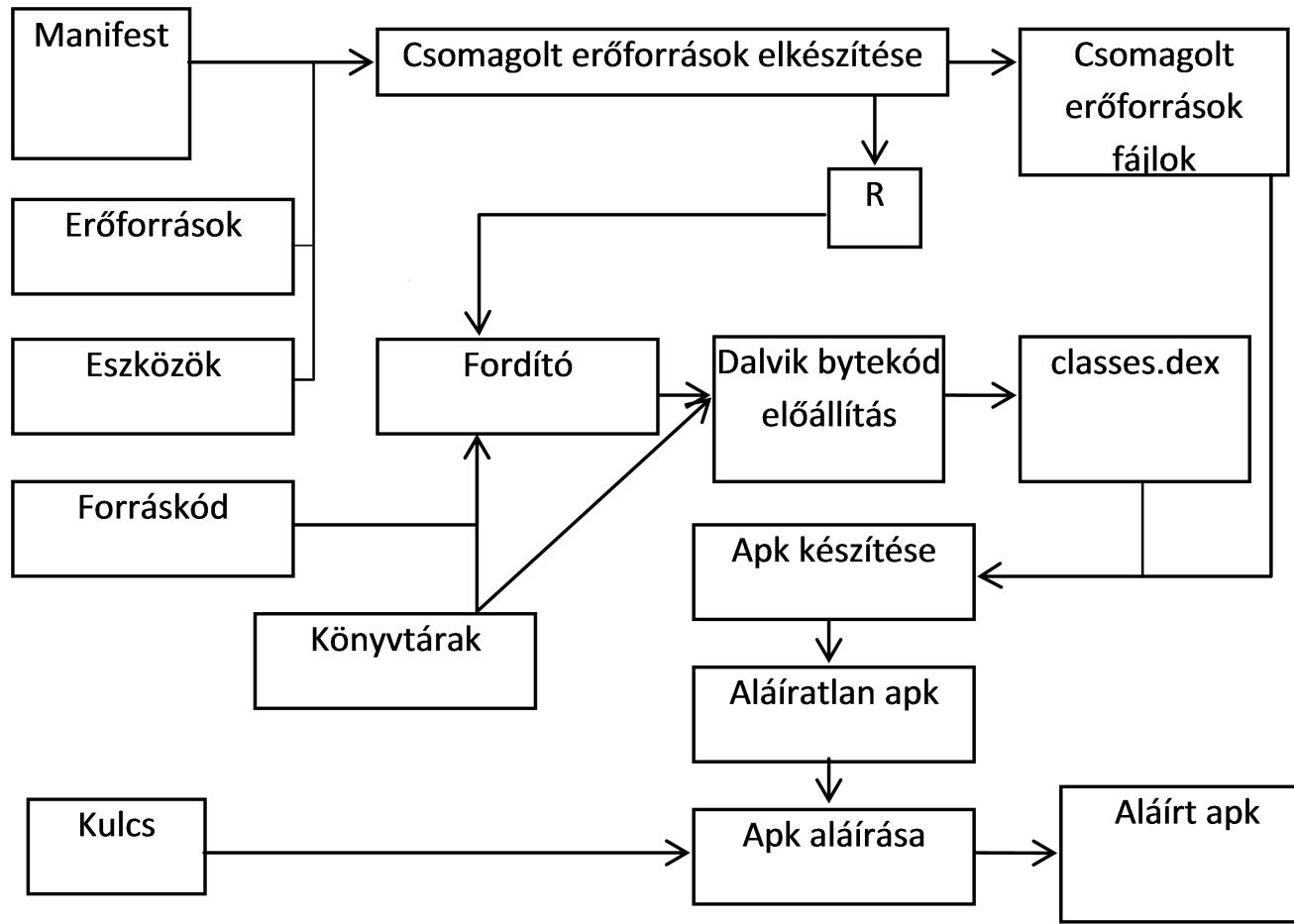
Holnaptól Kotlin
kell tanítanunk

Bazzeg

SDK komponensek

- SDK minden Android verzióra
- Dokumentáció
- Példakódok
- USB Driverek (ADB)
- Third party kiegészítők
 - > Google APIs (Térkép)
 - > Galaxy Tab API
 - > Stb.
- Konzolos felhasználás is támogatott, pl projekt létrehozás:
 - > android create project --target android-16 --name MyFirstApp --path D:\tmp\MyFirstApp --activity MainActivity --package com.example.myfirstapp

A fordítás menete (forrás->.apk)



Az Android .apk állomány

- Leginkább a Java világban megszokott .jar-hoz hasonlítható, de vannak jelentős eltérések
- Tömörített állomány, mely tipikusan a következő tartalommal rendelkezik:
 - > META-INF könyvtár
 - CERT.RSA: alkalmazás tanúsítvány
 - MANIFEST.MF: meta információk kulcs érték párokban
 - CERT.SF: erőforrások listája és SHA-1 hash értékük, pl:

```
Signature-Version: 1.0
Created-By: 1.0 (Android)
SHA1-Digest-Manifest: wxqnEAI0UA5nO5QJ8CGMwj kGGWE=
...
Name: res/layout/exchange_component_back_bottom.xml
SHA1-Digest: eACjMjESj7Zkf0cBFTZ0nqWrt7w=
...
Name: res/drawable-hdpi/icon.png
SHA1-Digest: DGEqylP8W0n0iV/ZzBx3MW0WGCA=
```
 - > Res könyvtár: erőforrásokat tartalmazza
 - > AndroidManifest.xml: név, verzió, jogosultság, könyvtárak
 - > classes.dex: lefordított osztályok a Dalvik számára érhető formátumban
 - > resources.arsc

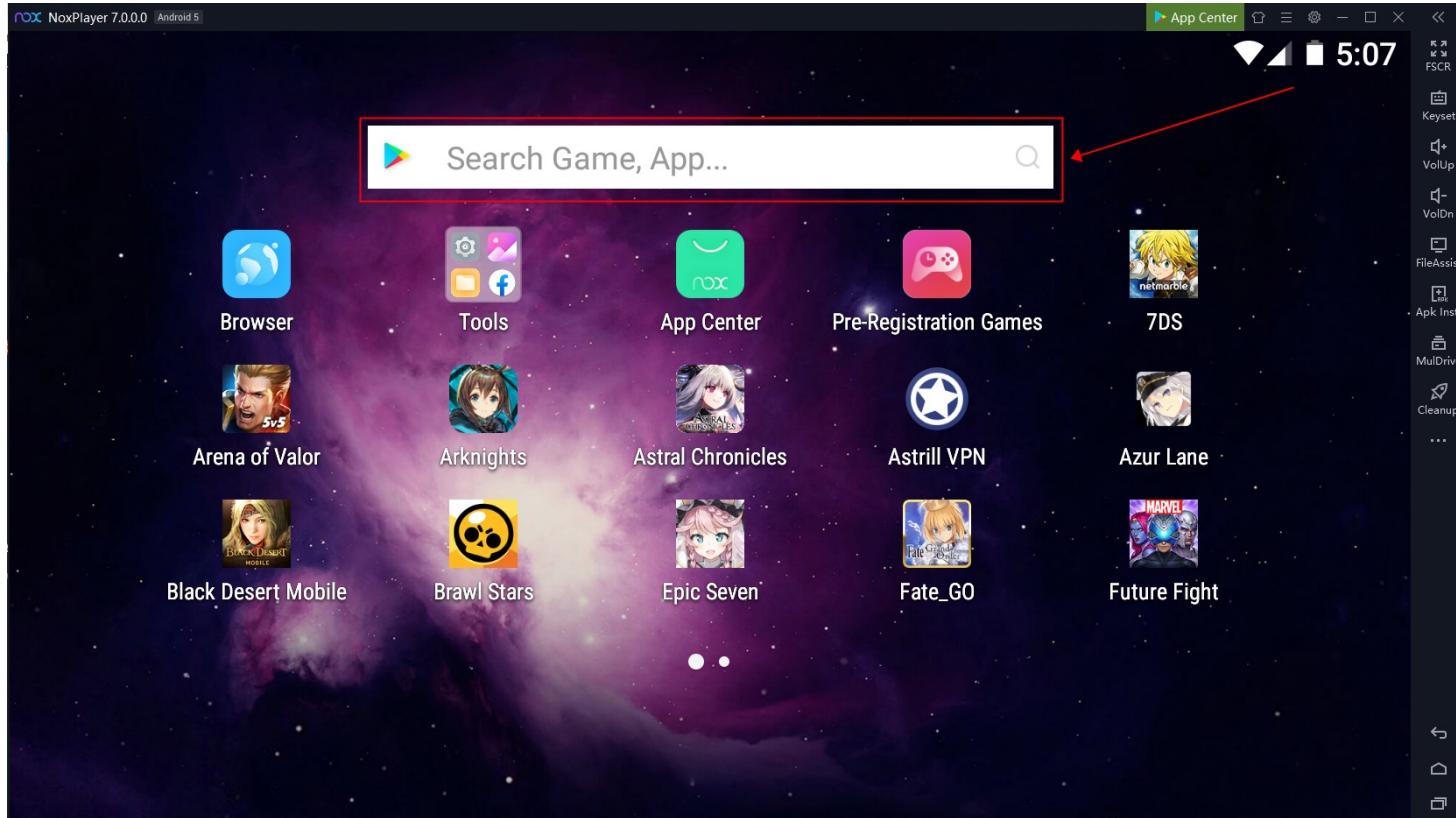
Emulátor

- Teljes operációs rendszer emulálása (lassú)
 - > Beépített alkalmazások elérhetők
 - > Ctrl+F11 (screen orientáció állítás)
- Alternatíva: Genymotion emulátor (<https://www.genymotion.com/>)



Alternatív emulátor

- <https://www.bignox.com/>



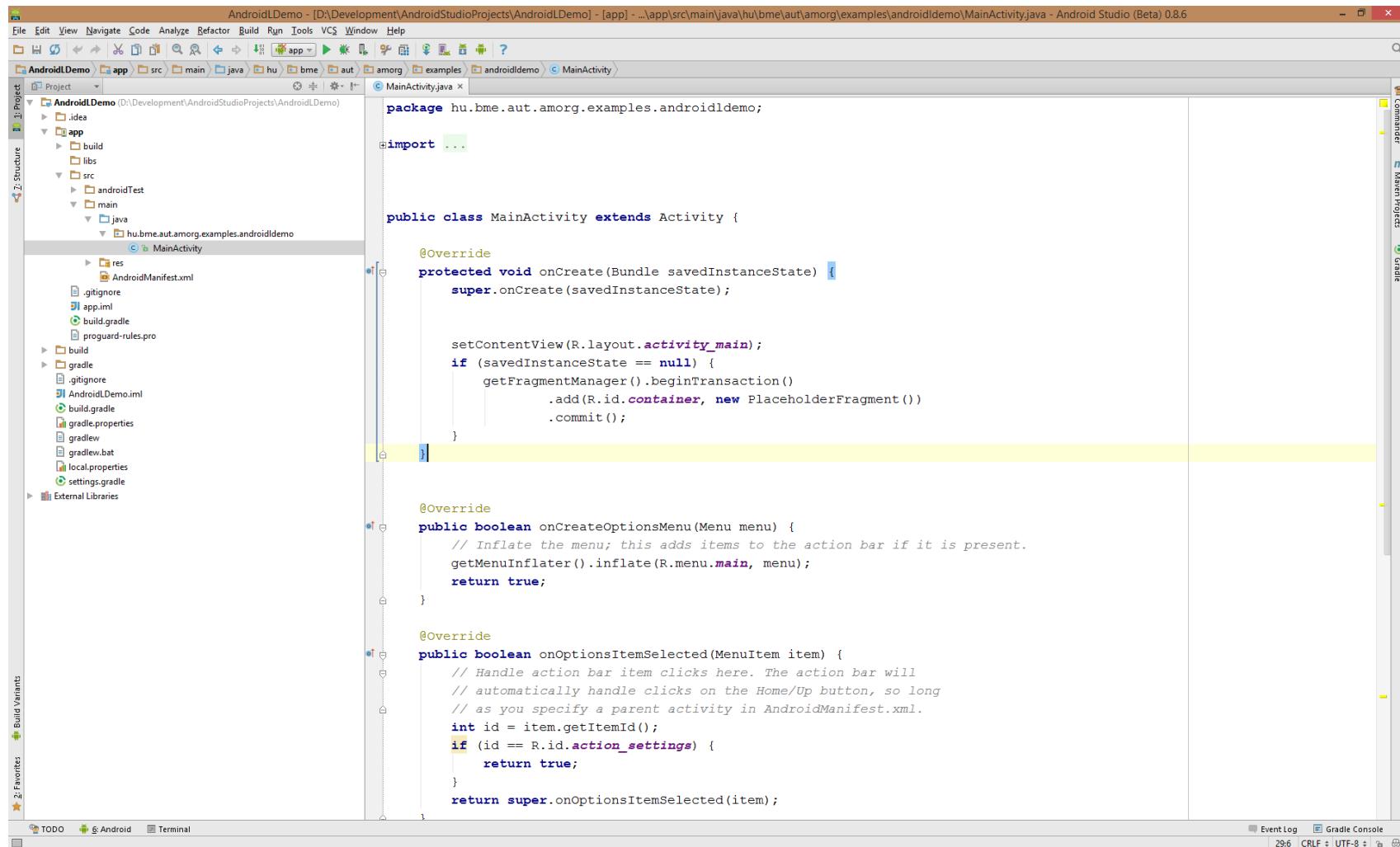
Emulátor elérése konzolról

- Csatlakoztatott emulátorok/eszközök listázása
- ADB: Android Debug Bridge
 - > adb devices
- Shell elérése
 - > adb shell
- Csatlakozás telneten keresztül:
 - > Indítsunk telnet klienst
 - > o localhost 5554
- SMS küldése:
 - > sms send <küldő száma> <üzenet>
- Hanghívás
 - > gsm call <hívó száma>

Debugolás folyamata

- On-device debug teljes mértékben támogatott
 - > Megfelelő USB driver szükséges!
 - > Készüléken engedélyezni kell az USB debugolást
- minden alkalmazás önálló process-ként fut
- minden ilyen process saját virtuális gépet (VM) futtat
- minden VM egy egyedi portot nyit meg, melyre a debugger rögzíthető (8600, 8601, stb.)
- létezik egy úgynevezett „base port” is (8700), mely minden VM portot figyel és erre csatlakozva az összes VM-et debugolhatjuk

Hello Android Studio



The screenshot shows the Android Studio interface with the title bar "AndroidLDemo - [D:\Development\AndroidStudioProjects\AndroidLDemo] - (app) - ...app\src\main\java\hu\bme\aut\amorg\examples\androidldemo>MainActivity.java - Android Studio (Beta) 0.8.6". The main window displays the Java code for MainActivity.java:

```
package hu.bme.aut.amorg.examples.androidldemo;

import ...

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);
        if (savedInstanceState == null) {
            getFragmentManager().beginTransaction()
                .add(R.id.container, new PlaceholderFragment())
                .commit();
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

The left sidebar shows the project structure with files like .gitignore, build.gradle, and gradlew. The bottom navigation bar includes tabs for TODO, Android, Terminal, Event Log, and Gradle Console.

További részletek a laborokon😊

Az első Android alkalmazás

Az első Android alkalmazás

Ősosztály

```
public class HelloAndroid extends Activity {
```

Ősosztály
implementáció
meghívása

 * /
 * Called when the activity is first created. */

 * Ide

```
    public void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
```

```
        TextView tv = new TextView(this);
```

```
        tv.setText("Hello Android!");
```

```
        setContentView(tv);
```

TextView
megjelenítése



Android HelloWorld XML alapú UI-al 1/2

Hello Android XML (*res/layout/activity_main.xml*):

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android=  
    "http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical" >  
    <TextView  
        android:id="@+id/tvHello"  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:text="@string/hello" />  
</LinearLayout>
```

Egyedi ID

Android HelloWorld XML alapú UI-al 2/2

```
package hu.bute.daai.amorg.examples;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class HelloWorldActivity extends Activity {

    @Override

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        TextView myTextView = (TextView) findViewById(R.id.tvHello);
        myTextView.append("\n--MODIFIED--");
    }
}
```

XML alapú layout

UI komponens kikeresése ID
alapján

Egyszerű esemény kezelés

```
public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    final TextView myTextView =  
        (TextView) findViewById(R.id.tvHello);  
    myTextView.append("#");  
    myTextView.setOnClickListener(new OnClickListener() {  
        public void onClick(View v) {  
            myTextView.append("\n--CLICKED--");  
        }  
    });  
}
```

Mivel anonim
osztályból férünk
hozzá

Egyszerű érintés
esemény kezelés

Az első Android alkalmazás Kotlin-ban ☺

Egyszerű esemény kezelés

Kotlin extensions miatt
használható

Lambda hívás

```
class MainActivity : AppCompatActivity() {  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
        myTextView.append("#")  
  
        myTextView.setOnClickListener{  
            myTextView.append("\n--CLICKED--")  
        }  
    }  
}
```

Függvény mint paraméter

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        btnTime.setOnClickListener(::click)  
    }  
  
    private fun click(view: View) {  
        Toast.makeText(this,  
            Date(System.currentTimeMillis()).toString(),  
            Toast.LENGTH_LONG).show()  
    }  
}
```

Eseménykezelő megadása layout-ban

View Binding

<https://developer.android.com/topic/libraries/view-binding>

View Binding

- *findViewById* (és Kotlin synthetic) kiváltása
- Amint aktiváltuk akkor a modulba található összes layouthoz generálódik egy binding class
 - > Binding class referenciát tartalmaz minden View-hoz aminek van ID-je és a root

```
buildFeatures {  
    viewBinding true  
}
```

- Layout file *ignore*-álható

```
<LinearLayout  
    ...  
    tools:viewBindingIgnore="true" >  
    ...  
</LinearLayout>
```

View Binding példa

- Tegyük fel, hogy adott egy *activity_main.xml*
- Ebből generálja a rendszer az *ActivityMainBinding* osztályt, ami tartalmazza a View-kat, melyeknek van *id*-ja

```
class MainActivity : AppCompatActivity() {  
    private lateinit var binding: ActivityMainBinding  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        binding = ActivityMainBinding.inflate(LayoutInflater)  
        val view = binding.root  
        setContentView(view)  
  
        binding.tvHello.text="DEMO"  
    }  
}
```

View Binding - Fragment

```
private var _binding: ResultProfileBinding? = null
// This property is only valid between onCreateView and
// onDestroyView.
private val binding get() = _binding!!

override fun onCreateView(
    inflater: LayoutInflater,
    container: ViewGroup?,
    savedInstanceState: Bundle?
): View? {
    _binding = ResultProfileBinding.inflate(inflater, container, false)
    val view = binding.root
    return view
}

override fun onDestroyView() {
    super.onDestroyView()
    _binding = null // FONTOS!!!
}
```

View Binding vs. Kotlin Extensions with synthetic views

- <https://stackoverflow.com/questions/58351239/viewbinding-vs-kotlin-android-extensions-with-synthetic-views>

	View binding	ButterKnife	Kotlin synthetics
Always null-safe	✓	!	✗
Only reference ids from current layout	✓	✗	✗
Supports Kotlin & Java	✓	✓	✗
Amount of code needed	Low	Some duplication	Low

Data Binding

<https://developer.android.com/topic/libraries/data-binding>

Adatkötés a gyakorlatban

- A cél nem elsősorban a „boilerplate” kód elkerülése, mint például `findViewById(...)/setText(...)`
- Legfőbb előnyök:
 - > UI frissítés egyszerűen ha egy érték a kódban több helyen is változhat
 - > Az adatkötés egy valós kapcsolat az adat és a UI között

Android Data Binding

- Cél a felület és a modell szoros összekapcsolása
- Szálkezelés nem probléma
- Rövidíti a kódot
- Gradle:

```
buildFeatures {  
    dataBinding true  
}
```

- További részletek:
 - > <http://developer.android.com/tools/data-binding/guide.html>

Android Data Binding példa 1/3

- Layout:

```
<layout xmlns:android=
    "http://schemas.android.com/apk/res/android">
<data>
    <variable name="user"
    type="com.example.MainActivity.User"/>
</data>
<LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@{user.firstName}"/>
    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@{user.lastName}"/>
</LinearLayout>
</layout>
```

Android Data Binding példa 2/3

- POJO:

```
data class User(var firstName: String,  
               var lastName: String)
```

Android Data Binding példa 3/3

- Bind-olás Activity-ben:

```
class MainActivity : AppCompatActivity() {

    data class User(var firstName: String, var lastName: String)

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        val binding: ActivityMainBinding = DataBindingUtil.setContentView(this,
            R.layout.activity_main)
        binding.user = User("Test", "User")
    }
}
```

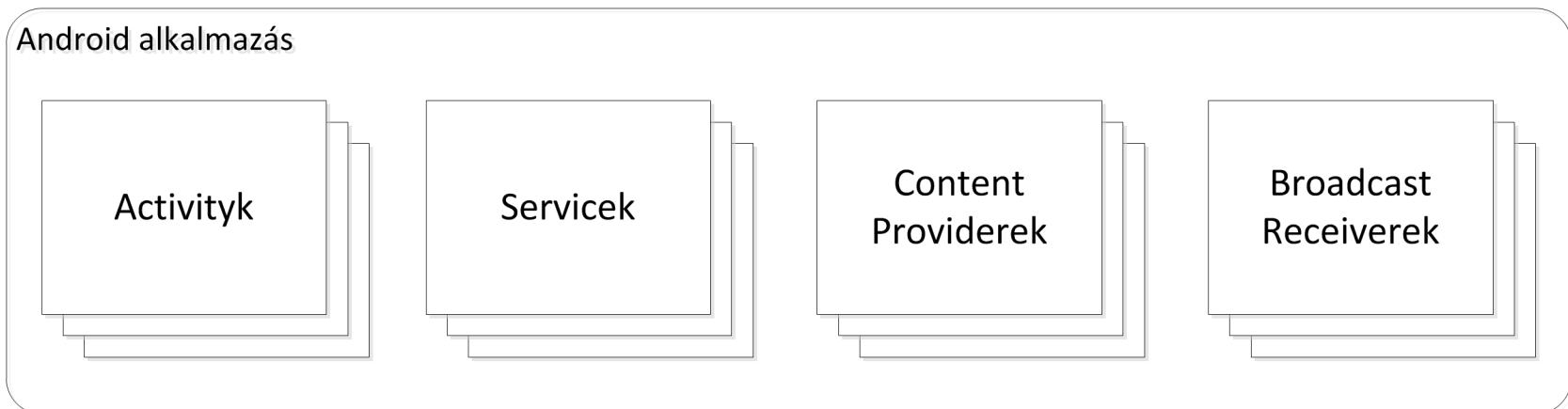
Generált osztály

View Binding vs. Data Binding

- Mindkettő Binding osztályt generál
- View Binding előnyök
 - > Gyorsabb fordítás: nem használ annotáció feldoglozást
 - > Könnyű használat: nincsenek speciális XML layout tag-ek/módosítások
- View Binding hátrányok:
 - > Nem támogat layout változókat, vagy kifejezéseket
 - > Nem támogat két irányú adatkötést
- Javaslat:
 - > „it is best in some cases to use both view binding and data binding in a project. You can use data binding in layouts that require advanced features and use view binding in layouts that do not”

Android alkalmazás felépítése

- Egy Android alkalmazás egy vagy több alkalmazás komponensből épül fel:
 - > Activity-k
 - > Service-k
 - > Content Provider-ek
 - > Broadcast Receiver-ek



Manifest állomány

- Alkalmazás leíró, definiálja az alkalmazás komponenseit
- XML állomány
- Komponens indítás előtt a rendszer a manifest állományt ellenőrzi, hogy definiálva van-e benne a kért komponens
- További feladatokat is ellát (pl. mik az alkalmazás futtatásának minimális követelményei)
- Alkalmazás telepítésekor ellenőrzi a rendszer



Manifest állomány tartalma

- Alkalmazást tartalmazó java package – egyedi azonosítóként szolgál
- Engedélyek, amelyekre az alkalmazásnak szüksége van (pl. internet elérés, névjegyzék elérés, stb.)
- Futtatáshoz szükséges minimum API szint
- Hardware és software funkciók, amit az alkalmazás használ (pl. kamera, bluetooth, stb.)
- Külső API könyvtárak (pl. Google Maps API)

Manifest példa 1/2

```
<?xml version="1.0" encoding="utf-8"?>  
  
<manifest xmlns:android=  
          "http://schemas.android.com/apk/res/android"  
          package="hu.bute.daai.amorg.examples"  
          android:versionCode="1"  
          android:versionName="1.0" >  
  
<uses-sdk android:minSdkVersion="7" />  
  
<application  
          android:icon="@drawable/ic_launcher"  
          android:label="@string/app_name" >  
    <activity ...>...</activity>  
  
</application>  
  
</manifest>
```

Egyedi package név
(azonosító)

Legkisebb támogatott
verzió

Alkalmazás ikon és
címke

Manifest példa 2/2

```
<?xml version="1.0" encoding="utf-8"?>  
<manifest .../>  
...  
<application ...>  
    <activity  
        android:name=".AndHelloWorldActivity"  
        android:label="@string/app_name">  
        <intent-filter>  
            <action android:name=  
                    "android.intent.action.MAIN"/>  
            <category android:name=  
                    "android.intent.category.LAUNCHER"/>  
        </intent-filter>  
    </activity>  
</application>  
</manifest>
```

Activity osztály és cím

Alkalmazás belépési pont jelölő

Megjelenik a futtatható alkalmazások listájában (Launcher)

Manifest attribútumok és tag-ek

- android:icon: alkalmazás ikonja
- android:name: Activity teljes neve package-el együtt
- android:label: A készülék felületén, a felhasználók által látható név
- Komponensek:
 - > <activity>: Activity
 - > <service>: Service
 - > <provider>: Content provider
 - > <receiver>: Broadcast receiver
- A manifest-ben nem szereplő Activity-k, Service-k és Content provider-ek nem láthatók a rendszer számára
- Broadcast receiver-ek viszont dinamikusan is ki/be-regisztrálhatnak (kódból – registerReceiver())

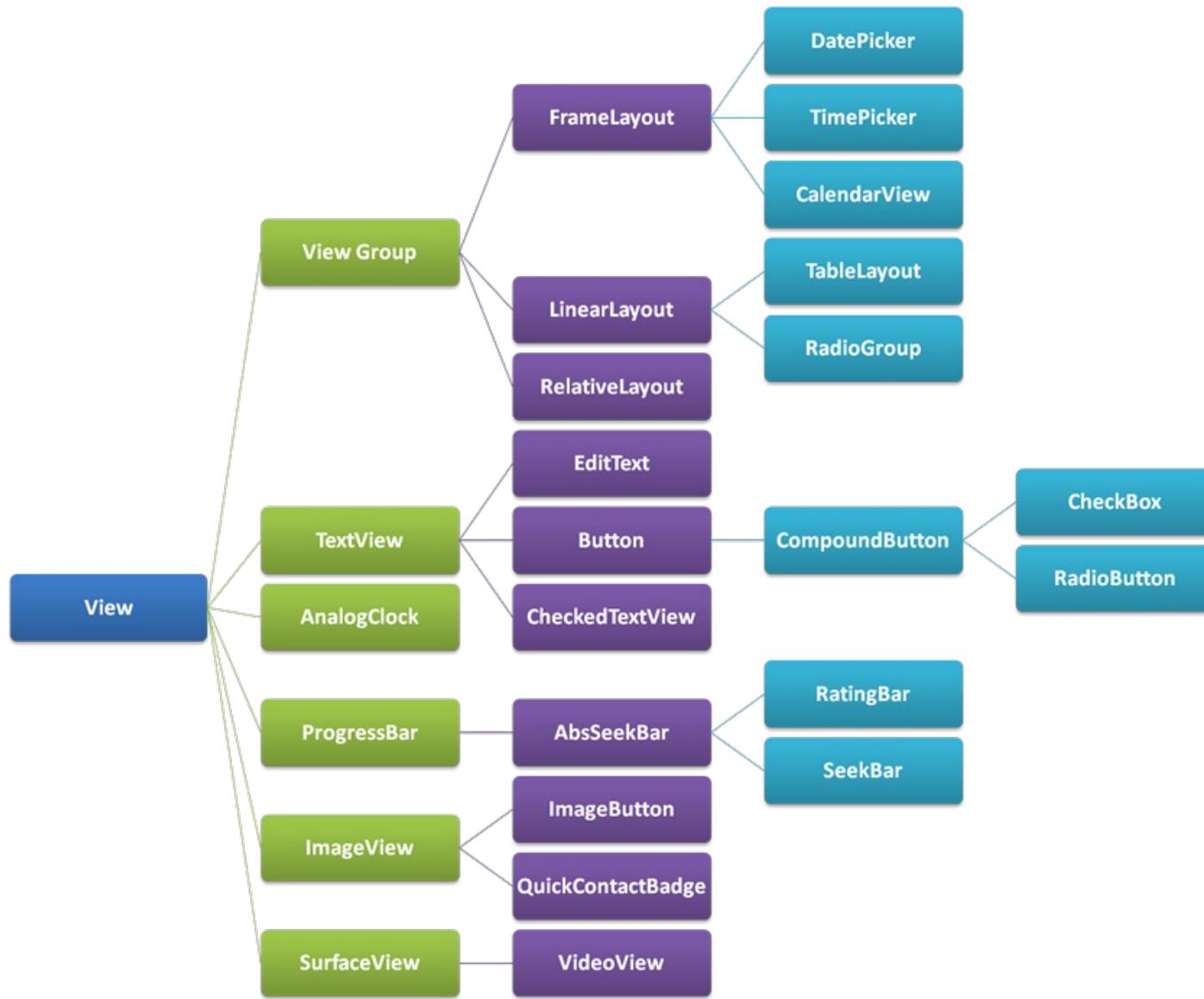
Application beállítások

```
<application android:allowTaskReparenting=["true" | "false"]
    android:allowBackup=["true" | "false"]
    android:backupAgent="string"
    android:debuggable=["true" | "false"]
    android:description="string resource"
    android:enabled=["true" | "false"]
    android:hasCode=["true" | "false"]
    android:hardwareAccelerated=["true" | "false"]
    android:icon="drawable resource"
    android:killAfterRestore=["true" | "false"]
    android:largeHeap=["true" | "false"]
    android:label="string resource"
    android:logo="drawable resource"
    android:manageSpaceActivity="string"
    android:name="string"
    android:permission="string"
    android:persistent=["true" | "false"]
    android:process="string"
    android:restoreAnyVersion=["true" | "false"]
    android:requiredAccountType="string"
    android:restrictedAccountType="string"
    android:supportsRtl=["true" | "false"]
    android:taskAffinity="string"
    android:testOnly=["true" | "false"]
    android:theme="resource or theme"
    android:uiOptions=["none" | "splitActionBarWhenNarrow"]
    android:vmSafeMode=["true" | "false"] >
    ...
</application>
```

Mi igaz a Manifest állományra?

- A. Csak az Activity komponenseket kell felsorolni benne.
- B. Csak egy Service komponenst tartalmazhat.
- C. Az összes alkalmazás komponenst fel kell sorolni benne kivéve a dinamikusan regisztrálható BR komponenseket.
- D. XML és Java kód keveredhet benne.

Android UI architektúra



Egyedi nézetek

- View leszármazott saját nézet osztály
 - > `onDraw(...)` felüldefiniálása
 - > `onTouchEvent(...)` felüldefiniálása
 - > XML layout-ban használhatók
- Beépített nézetek és *LayoutGroup*-ok is felüldefiniálhatók, pl. saját nézet *RelativeLayout*-ból leszármaztatva

Összefoglalás

- Tárgykövetelmények
- Android verziók
- A platform felépítése
- Fejlesztőkörnyezet beállítása
- Android alkalmazás komponensei
- Manifest állomány, jogosultságok

Mi várható a félévben?

Szoftver minőség

Hány százalékát használják valójában az alkalmazásoknak?

5%
desktop/web

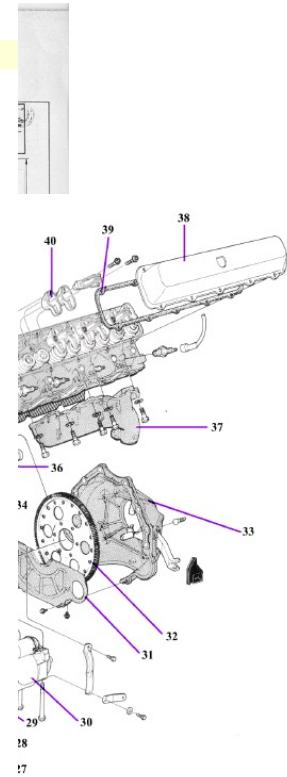


~1% mobil

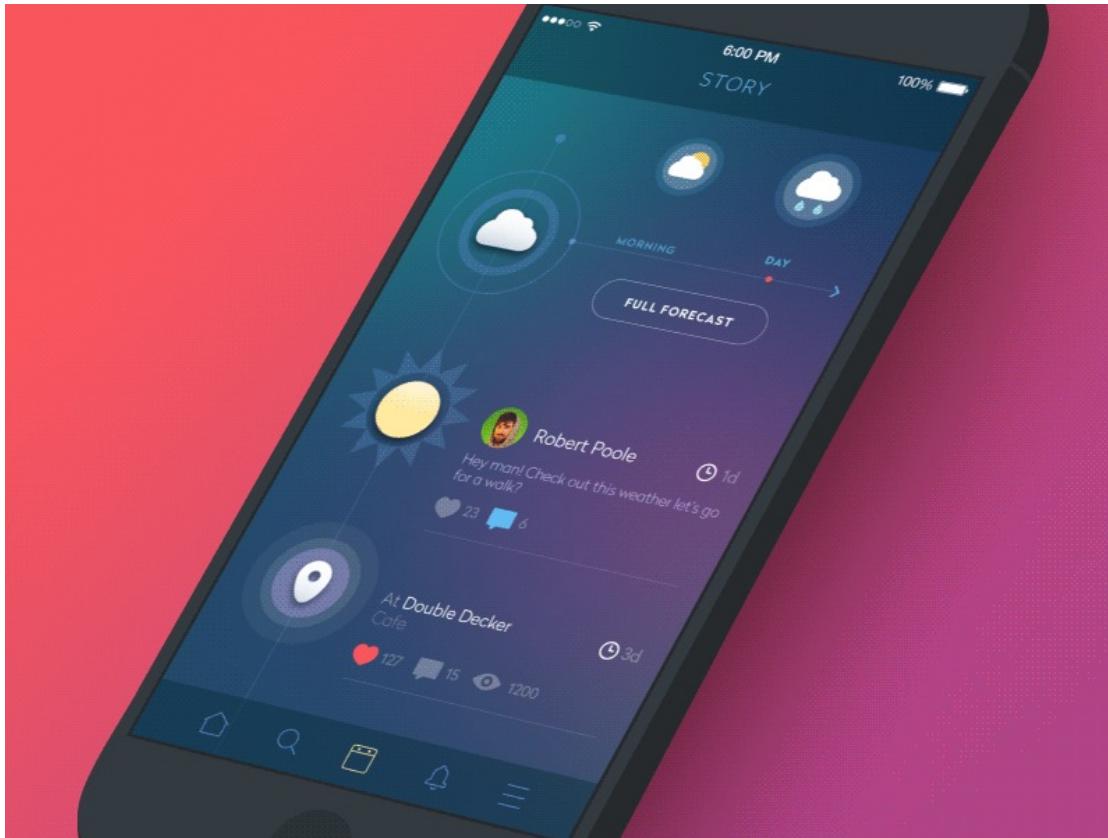
Szoftverfejlesztő feladata



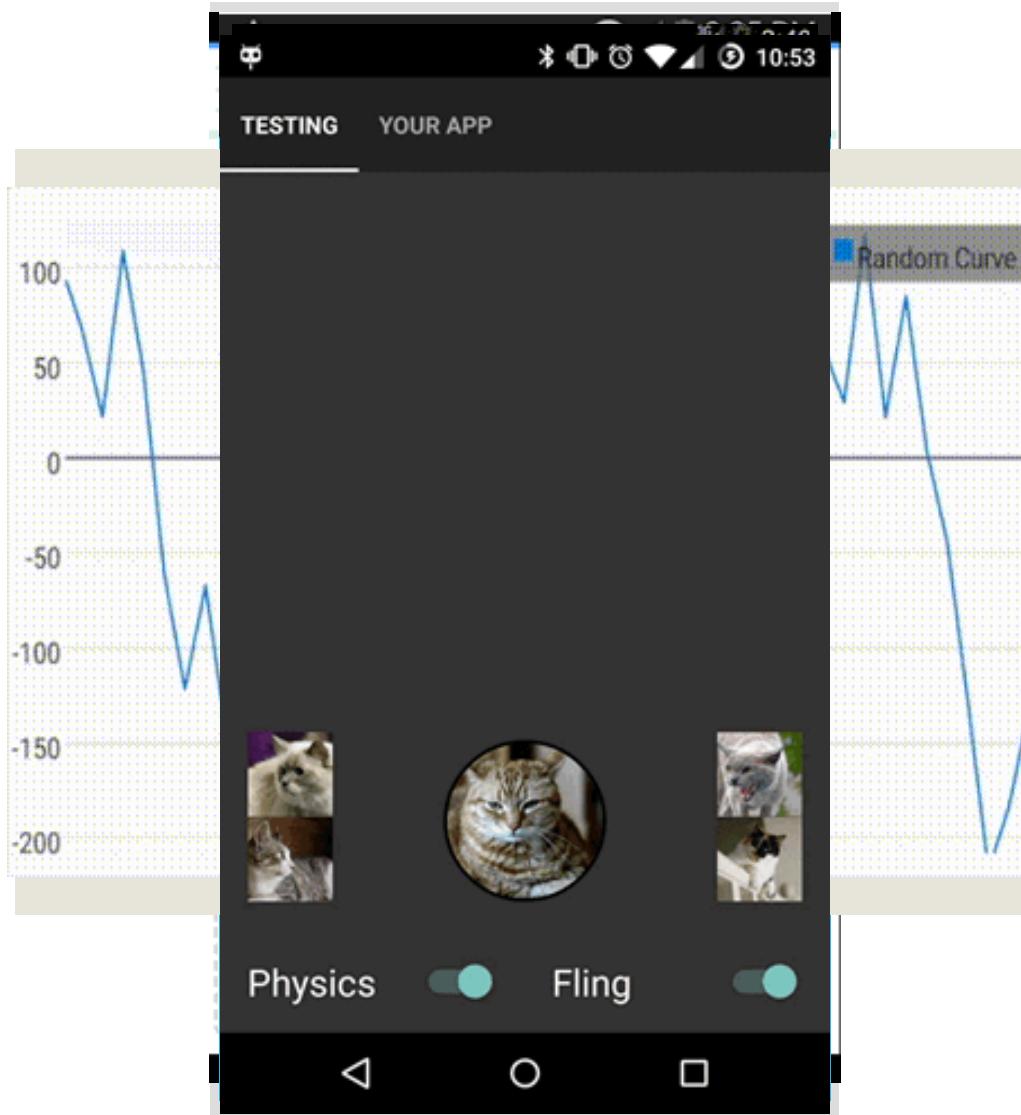
```
* upload, independent of whether the original activity is paused, stopped,  
💡 or finished.  
*/  
  
public class Activity extends ContextThemeWrapper  
    implements LayoutInflater.Factory2,  
    Window.Callback, KeyEvent.Callback,  
    OnCreateContextMenuListener, ComponentCallbacks2,  
    Window.OnWindowDismissedCallback {  
  
    private static final String TAG = "Activity";  
    private static final boolean DEBUG生命周期 = false;  
  
    /** Standard activity result: operation canceled. */  
    public static final int RESULT_CANCELED = 0;  
    /** Standard activity result: operation succeeded. */  
    public static final int RESULT_OK = -1;  
    /** Start of user-defined activity results. */  
    public static final int RESULT_FIRST_USER = 1;  
  
    static final String FRAGMENTS_TAG = "android:fragments";  
  
    private static final String WINDOW_HIERARCHY_TAG = "android:viewHierarchyState";  
    private static final String SAVED_DIALOG_IDS_KEY = "android:savedDialogIds";  
    private static final String SAVED_DIALOGS_TAG = "android:savedDialogs";  
    private static final String SAVED_DIALOG_KEY_PREFIX = "android:dialog_";  
    private static final String SAVED_DIALOG_ARGS_KEY_PREFIX = "android:dialog_args_";  
  
    private static class ManagedDialog {  
        Dialog mDialog;  
        Bundle mArgs;  
    }  
    private SparseArray<ManagedDialog> mManagedDialogs;  
  
    // set by the thread after the constructor and before onCreate(Bundle savedInstanceState) is called.  
    private Instrumentation mInstrumentation;  
    private IBinder mToken;  
    private int mIdent;  
    /*package*/ String mEmbeddedID;  
    private Application mApplication;  
    /*package*/ Intent mIntent;
```



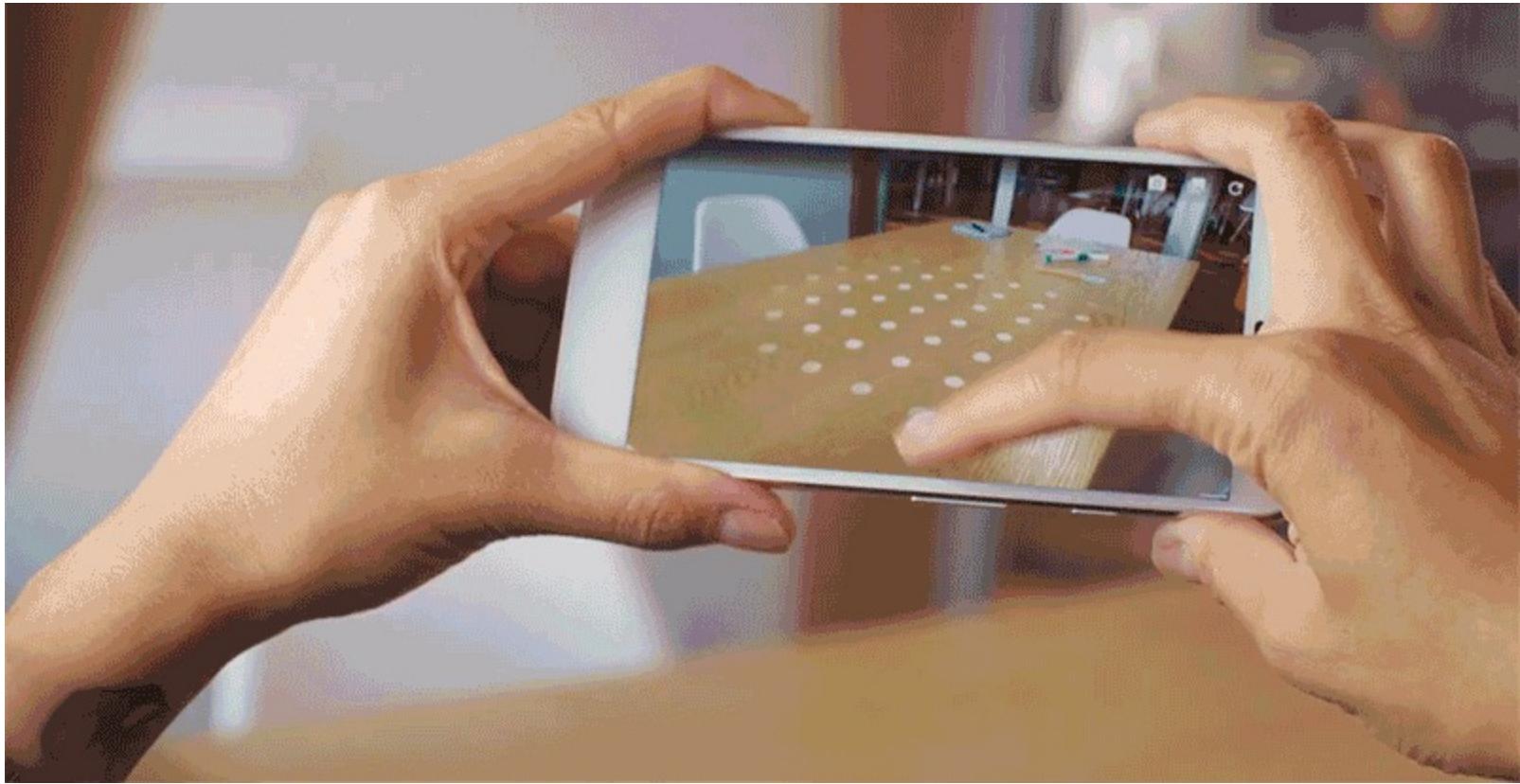
Felhasználói felület



Felületi elemek



Kiterjesztett valóság



Kotlin (Android) – Swift (iOS)



Swift is like Kotlin

Swift

```
var movieCount = 0
var songCount = 0

for item in library {
    if item is Movie {
        movieCount += 1
    } else if item is Song {
        songCount += 1
    }
}
```

Kotlin

```
var movieCount = 0
var songCount = 0

for (item in library) {
    if (item is Movie) {
        ++movieCount
    } else if (item is Song) {
        ++songCount
    }
}
```

Karrier út

- Junior Android fejlesztő
- Medior Android fejlesztő
- Senior Android fejlesztő
- Vezető fejlesztő
- Architekt



Kérdések

