

Android

További hálózati eszközök, multimédia tartalom
előállítása és kezelése

Dr. Ekler Péter
peter.ekler@aut.bme.hu



Department of
Automation and
Applied Informatics

- November 6. 12:15
- Terembeosztás a tárgy oldalon hamarosan

Adjon meg f-hez primitív függvényt, amelyre. $F(0)=0$ $f(x) = 3\sqrt[3]{x} \cdot x^2$, $f(x) = 3e^{2x+2}$
 $f(x) = \frac{5}{(2x+1)^6}$, $f(x) = 3(x-1)^4$, $f(x) = 2x^5 + 3x^4 + 2x^2 - x + 2$

Számítsa ki az alábbi határozatlan integrálokat!

$$\int (1+x)^{10}, \quad \int \frac{(x+2)^3}{x}, \quad \int \frac{x^3-1}{x-1}, \quad \int \frac{x^4-1}{x+1}, \quad \int \sqrt{x^2+2x+1}, \quad \int \left(\frac{x^3}{\sqrt{x}} + \frac{\sqrt{x}}{x^3} \right), \quad \int \frac{(2x+1)^2}{2x^2}$$

$$\int 2\sin 3x, \quad \int 3\cos 2x, \quad \int \sin 2x \cdot \cos 2x, \quad \int (\cos^2 2x - \sin^2 2x), \quad \int 2\sin^2 x, \quad \int \cos^2 2x,$$

$$\int \sin 3x \cdot \sin 2x, \quad \int \cos 2x \cdot \cos 3x, \quad \int \sin 2x \cdot \cos 3x, \quad \int e^{2x-1}, \quad \int e^{2-3x}, \quad \int 2^{2x-1}, \quad \int 3^{2-2x},$$

$$\int \frac{1}{2x+1}, \quad \int \sqrt{x^3\sqrt{x}}, \quad \int \frac{1}{x \ln x}, \quad \int \frac{x-3}{x^2-6x+27}, \quad \int \frac{e^{3x}}{e^{3x}+5}, \quad \int \operatorname{ctg} 2x$$

$$\int x \sin x^2, \quad \int \frac{e^x}{\sqrt{1-e^{2x}}}, \quad \int \frac{e^{\operatorname{tg} x}}{\cos^2 x}, \quad \int \frac{\ln x}{\sqrt{x}}, \quad \int x^4 \cdot \ln x, \quad \int (2x-1) \sin 2x,$$

$$\int (3x+2) \cos 3x, \quad \int (x^2+2x)e^{-x}, \quad \int (2x^2+x+1) \sin 4x, \quad \int x^5 e^{x^2}, \quad \int (2-x^2) \cdot \cos 2x$$

$$\int \frac{2}{(x-1)(x+2)}, \quad \int \frac{3x}{(x+1)(x-3)}, \quad \int \frac{2x+1}{(x-1)(x-3)}, \quad \int \frac{2}{x(x+2)}, \quad \int \frac{3x-2}{x^2+x-6},$$

$$\int \frac{x(2x+2)}{(x^2+x)(x^2-9)}, \quad \int \frac{x^3-4x}{(x-2)(x^2+6x+8)}, \quad \int \frac{3x+4}{(2x+3)(2-x)}$$

Tartalom

- További hálózati eszközök
- Kamera kezelés
 - > Külső kamera alkalmazás használata
 - > Saját kamera felület létrehozása és használata
- Arcfelismerés
- Android multimedia framework bemutatása
- Hangfelvétel és lejátszás
- MP3 lejátszás
- Video kezelése
- Drag & Drop

WebView használata

WebView

- Web tartalom megjelenítése *Activity*-ből
- WebKit/Chromium alapú render motor
- Fő funkciók:
 - > Előre/hátra navigáció
 - > History
 - > Zoom
 - > Szöveges keresés a tartalomban
 - > Stb.
- *JavaScript* integráció
- *android.permission.INTERNET* szükséges a használatához

WebView megjelenítése 1/3

- Engedély beállítása a Manifest-be:

```
> <uses-permission android:name="android.permission.INTERNET"/>
```

- XML erőforrás definiálása:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<WebView xmlns:android=
```

```
    "http://schemas.android.com/apk/res/android"
```

```
    android:id="@+id/webview"
```

```
    android:layout_width="match_parent"
```

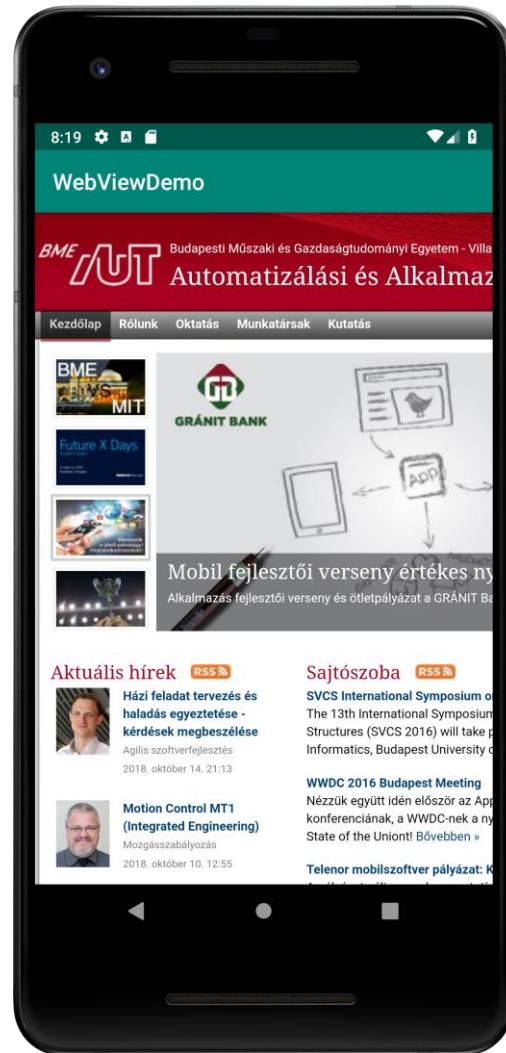
```
    android:layout_height="match_parent" />
```

WebView megjelenítése 2/3

- Vezérlés Activity-ből:

```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    setContentView(R.layout.activity_main)  
  
    webView.settings.javaScriptEnabled = true  
    webView.settings.builtInZoomControls = true  
  
    webView.loadUrl("https://www.aut.bme.hu")  
}
```

WebView megjelenítése 3/3



WebSettings

- *WebView* beállításainak kezelése
- Egy *WebView* példányosításakor az egyes tulajdonságok alapértelmezett értékeket vesznek fel
- Beállítások kezelése:
 - > `mWebView.getSettings()`
 - > getter és setter metódusok
- Példa beállítások:
 - > `textSize`
 - > `textZoom`
 - > `zoomControl`
 - > `userAgent`

Kitérő: billentyűzet kezelés

- Csak előtérben lévő Activity-n kezelhető a billentyűzet esemény
- Hangerő, vissza, egyéb gombok

```
override fun onKeyDown(keyCode: Int, event: KeyEvent): Boolean {  
    if (keyCode == KeyEvent.KEYCODE_BACK) {  
        // Billentyű esemény kezelése...  
    }  
    return super.onKeyDown(keyCode, event)  
}
```

WebView navigáció

- A *WebView* komponensben előre/hátra navigálhatunk:

```
override fun onBackPressed() {  
    if (webView.canGoBack()) {  
        webView.goBack()  
    } else {  
        super.onBackPressed()  
    }  
}
```

WebViewClient

- Weboldalak megnyitásakor fontos események léphetnek fel, melyeket Android oldalról kezelhetünk
- Például ha a *WebView*-ban beállított weboldal átirányít valahova, akkor alapértelmezetten a beépített böngésző hívódik meg
- De definiálhatunk egy *WebViewClient*-et a különféle események kezelésére
 - > `onFormResubmission(...)`
 - > `onLoadResource(...)`
 - > `onPageFinished(...)`
 - > `onReceivedError(...)`
 - > `onReceivedHttpAuthRequest (...)`
 - > `onReceivedLoginRequest (...)`
 - > `onReceivedSslError(...)`
 - > `shouldOverrideKeyEvent(...)`
 - > `shouldOverrideUrlLoading(...)`
 - > Stb.

URL átirányítás felüldefiniálása

```
webView.webViewClient = object : WebViewClient() {  
    override fun shouldOverrideUrlLoading(view: WebView,  
        request: WebResourceRequest): Boolean {  
        loadSite(request.url.toString())  
        return true  
    }  
  
    override fun onPageFinished(view: WebView, url: String) {  
        super.onPageFinished(view, url)  
        progressBarWebLoad.progress = 100  
        progressBarWebLoad.visibility = View.GONE  
    }  
}
```

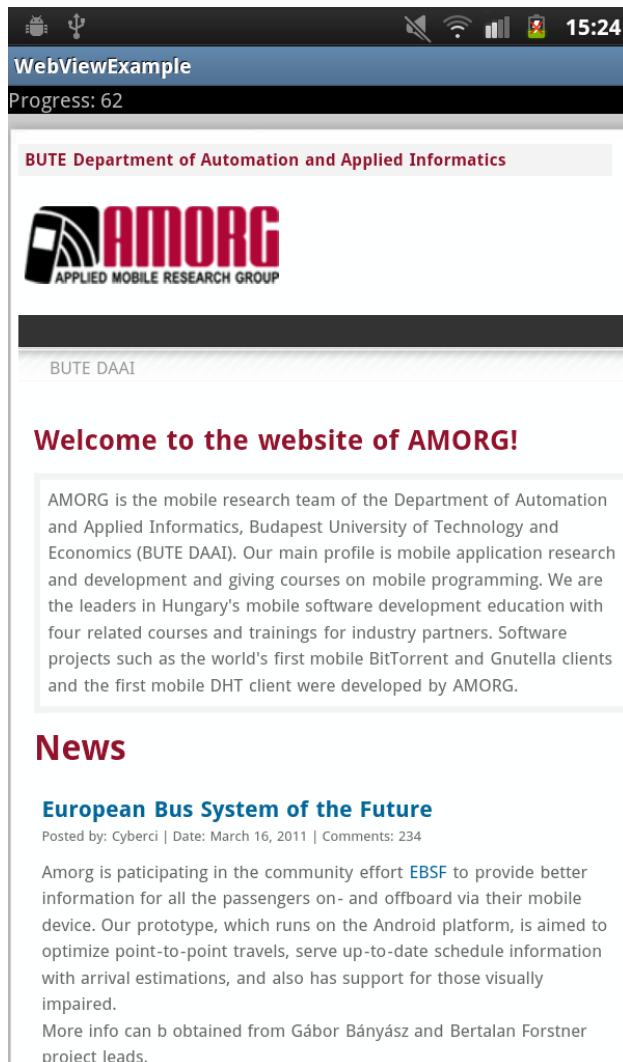
WebChromeClient

- További böngésző események kezelése
- JavaScript események kezelése ☺
- Callback függvények felüldefiniálása
- Példa callback-ok:
 - > `onJSTimeout()`
 - > `onJSAlert()`
 - > `onProgressChanged()`
 - > `onJSConfirm()`
 - > `onCreateWindow()`
 - > Stb.

Weboldal letöltés állapota 1/2

```
webView.webChromeClient = object : WebChromeClient() {  
    override fun onProgressChanged(view: WebView,  
        newProgress: Int) {  
        progressBarWebLoad.progress = newProgress  
    }  
}
```

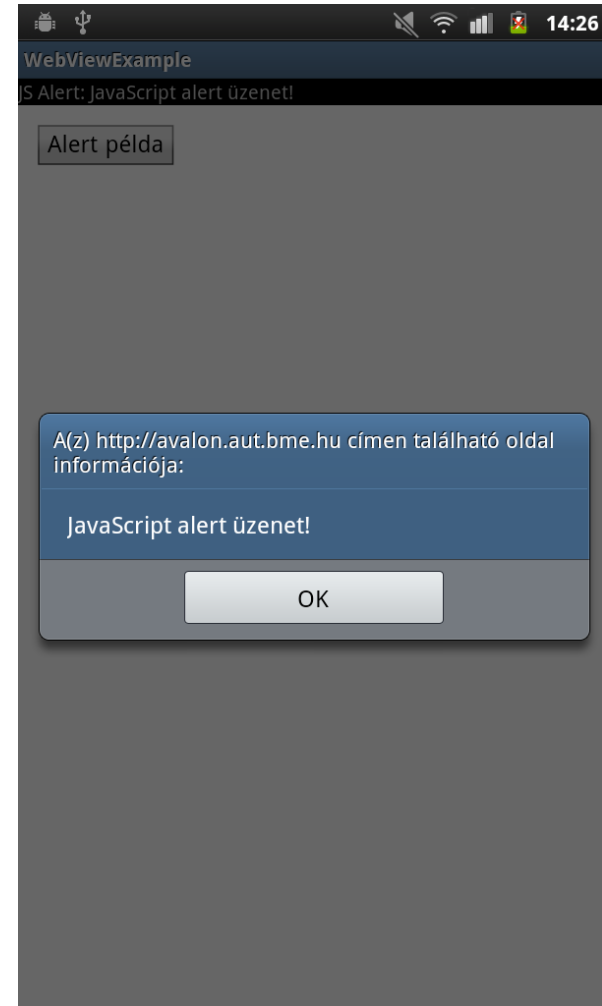
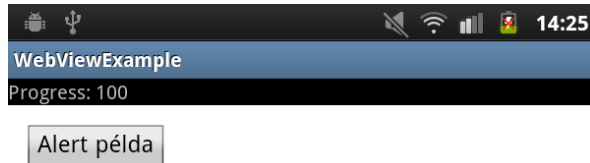
Weboldal letöltés állapota 2/2



JavaScript Alert példa 1/2

```
webView.getSettings().setJavaScriptEnabled(true)
webView.webChromeClient = object : WebChromeClient() {
    override fun onJsAlert(
        view: WebView, url: String,
        message: String, result: JsResult
    ): Boolean {
        tvStatus.setText("JS Alert: $message")
        // return true; esetén nem fut le a default
        // implementáció (pop-up ablak)
        return super.onJsAlert(view, url, message, result)
    }
}
```

JavaScript Alert példa 2/2



JavaScript kommunikáció WebViewn keresztül

- Fejlett JavaScript kezelő lib:
 - > <https://github.com/tcoulter/jockeyjs>
- Szenzor értékek böngészőn keresztül
 - > <http://atleast.aut.bme.hu/mobilesensor/>
 - > Nyissuk meg egy mobil böngészőben és tekintsük át mely szenzor adatok érhetők el

TCP/IP és UDP kommunikáció

TCP/IP Socket

- Szabványos *Socket* implementáció
- Jól ismert *java.net.Socket* osztály a kapcsolatok megnyitására
- *java.net.ServerSocket* osztály a bejövő kapcsolatok fogadására
 - Localhoston az alkalmazások egymás közti kommunikációja is megoldható
- *InputStream* és *OutputStream* támogatás az adatok olvasására és írására

Socket példa

```
val socket = Socket("192.168.2.112", 8787)
val inputStream = socket.getInputStream()
val isr = InputStreamReader(
    inputStream,
    "UTF-8"
)
val resultBuffer = StringBuilder()
var inChar: Int
while ((inChar = isr.read()) != -1) {
    resultBuffer.append(inChar.toChar())
}
val result = resultBuffer.toString()
// result kezelése
// ...
inputStream.close()
socket.close()
```

UDP communication

- User Datagram Protocol (UDP) for supporting Datagram messages
- Short, quick message handling
- UDP does not ensures that the packet will arrive
- Used typically when packet loss is not as important
- Example usages:
 - > Real time multimedia transfer
 - > Computer games
 - > Etc.

UDP on Androidon

- Use the standard Java classes
- *java.net.DatagramSocket*. socket

- Use it in case of broadcast:

```
DatagramSocket s = DatagramSocket()  
s.setBroadcast(true)
```

- *java.net.DatagramPacket*. UDP package object

UDP üzenetek küldése

```
val msg = "UDP Test"
val socket = DatagramSocket()
// In case of broadcast
socket.setBroadcast(true)
//InetAddress localAddr =
//  InetAddress.getByName("192.168.0.110");
val message = msg.toByteArray()
val p = DatagramPacket(
    message,
    msg.length, localAddr, 10100
)
socket.send(p)
```

UDP üzenet fogadása

```
val message = ByteArray(1500)
val packet = DatagramPacket(message, message.size)
val socket = DatagramSocket(10100)
socket.receive(packet)
val msg = String(
    message, 0, packet.getLength()
)
Log.d("MYTAG", "received: $msg")
socket.close()
```

Népszerű osztálykönyvtárak

- KryoNet

- > <https://github.com/EsotericSoftware/kryonet>

- Near

- > <https://github.com/adroitandroid/Near>

KryoNet példa

- Adat osztály regisztráció (szerializációhoz)

```
class TextMessage {  
    var text: String? = null  
}
```

```
class ImageMessage {  
    var image: ByteArray? = null  
}
```

```
Log.set(Log.LEVEL_DEBUG) // sokat segít ☺
```

```
server = Server(131072, 131072) // write és object buffer méretek  
val kryo = server.kryo  
kryo.register(TextMessage::class.java)  
kryo.register(ImageMessage::class.java)  
kryo.register(ByteArray::class.java)
```

KryoNet – szerver inicializálás

```
server.addListener(object : Listener() {  
    override fun connected(connection: Connection?) {  
        super.connected(connection)  
    }  
  
    override fun received(connection: Connection, obj: Any) {  
        ...  
    }  
  
    override fun disconnected(connection: Connection?) {  
        super.disconnected(connection)  
        ...  
    }  
})  
  
server.start()  
server.bind(54555, 54777) // tcp és udp portok
```

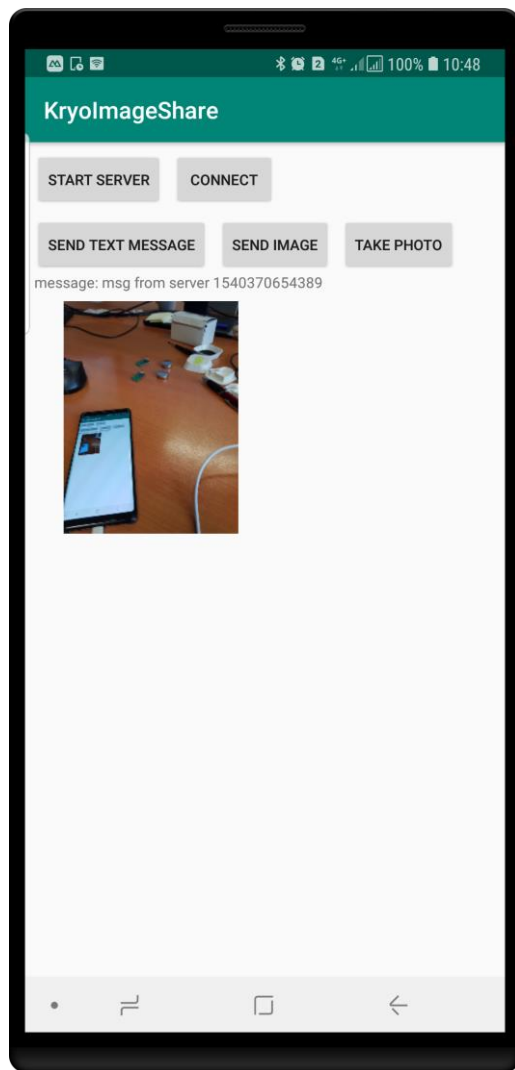
KryoNet – kliens felderítés és kapcsolódás

```
var client = Client()
client.start()
val kryo = client.kryo
kryo.register(TextMessage::class.java)
kryo.register(ImageMessage::class.java)
kryo.register(ByteArray::class.java)
val address = client.discoverHost(54777, 20000)

client.addListener(object : Listener() {
    override fun connected(connection: Connection?) {
        super.connected(connection)
    }

    override fun received(connection: Connection, obj: Any) {
        handleNetworkMessage(obj)
    }

    override fun disconnected(connection: Connection?) {
        super.disconnected(connection)
        runOnUiThread {
            Toast.makeText(this@MainActivity, "DISCONNECTED", Toast.LENGTH_LONG).show()
        }
    }
})
client.connect(20000, address, 54555, 54777)
```



Gyakoroljunk

Készítsünk egy kép megosztó alkalmazást KryoNet-el

Hálózati adatforgalom felügyelete - TrafficStats

- Gyakran szükség lehet az adatforgalom felügyeletére, fel/letöltés kijelzésére
- Megoldás: *android.net.TrafficStats* osztály
- Statikus függvények az adatforgalom lekérdezésére, például:
 - > *static long getMobileRxBytes()*
- Adatforgalom tiltásakor törlődik az eddigi statisztika
- UDP és TCP forgalom külön lekérdezhető
- Adatforgalom lekérdezése az egyes processzekhez (UID megadásával):
 - > *static long getUidRxBytes(int uid)*

Multimédia képességek

Bevezetés

- Napjainkban a multimédia tartalmaknak meghatározó szerepük van mobil eszközökön
- Fontos a multimédia tartalom előállítása és kezelése is
- Használjuk kreatívan a készülék képességeit
- Mindig gondoljunk a lefoglalt erőforrások felszabadítására!
- A példák nagyon fontosak!

Kamerakezelés

- Széleskörű kamera támogatás
- Gazdag Android API
- Több kamera kezelése
- Különböző kamera típusok kezelése
- Kamera funkciók elérése API-ból (zoom, flash, stb.)
- Kép és video rögzítés
- Használjuk az a kamerát kreatív módon, például:
 - > Mozgásérzékelés
 - > Kiterjesztett valóság (Augmented Reality)

Kamerahasználat engedélyezése

- Meg kell adnunk a megfelelő *manifest* engedélyeket
- Kamera használata:
 - > `<uses-permission android:name="android.permission.CAMERA" />`
- Kötelező-e a kamera megléte:
 - > `<uses-feature android:name="android.hardware.camera" android:required="true/false" />`
- Ha el kívánjuk tárolni a médiát a file rendszerben:
 - > `<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />`
- Audio felvétele a video tartalomhoz:
 - > `<uses-permission android:name="android.permission.RECORD_AUDIO" />`

Beépített kamera alkalmazás használata

- Leggyorsabb módszer kép/video készítéséhez
- Egy megfelelő *Intent* összeállításával meghívhatjuk az alapértelmezett kamera alkalmazást
- Az *Intent* segítségével elindul a beépített kamera alkalmazás és készít egy képet vagy videót
- Végül a vezérlés visszakerül az alkalmazásunkhoz és az eredmény kiolvasható

Képkészítés lépései

- Megfelelő Intent összeállítása
 - > Kép készítése: `MediaStore.ACTION_IMAGE_CAPTURE`
 - > Video készítése: `MediaStore.ACTION_VIDEO_CAPTURE`
- Intent paraméterek megadása (*putExtra()*)
 - > *MediaStore.EXTRA_OUTPUT*: kép mentési helye (Uri), ha nem adjuk meg, egy alapértelmezett könyvtárba menti
- Camera Intent indítása:
 - > *startActivityForResult()*
- Visszatérés kezelése
 - > Activity-ben az *onActivityResult()* felüldefiniálása
 - > A rendszer ezt hívja meg, amikor visszatértünk a kamera alkalmazásból
 - > A visszaadott *Intent* *getData()* függvényével lekérdezhető az alapértelmezett mentés helye

Példa: Fotó készítése beépített kamera alkalmazással

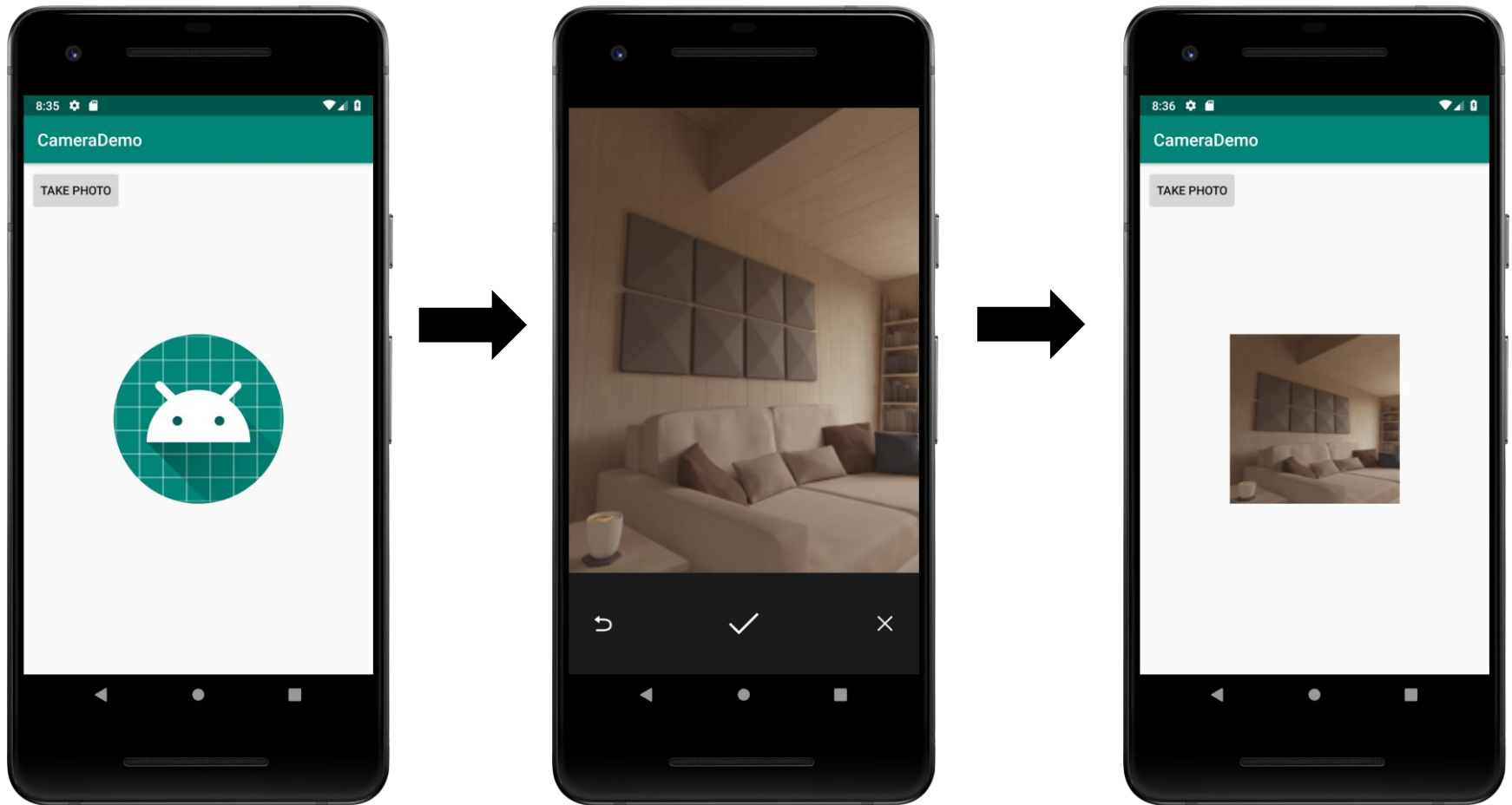
- Fotó készítése

```
btnPhoto.setOnClickListener {  
    Intent(MediaStore.ACTION_IMAGE_CAPTURE).also { takePictureIntent ->  
        takePictureIntent.resolveActivity(packageManager)?.also {  
            startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE)  
        }  
    }  
}
```

- Válasz kezelése és kép megjelenítése

```
override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {  
    if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {  
        data?.also {  
            val imageBitmap = it.extras.get("data") as Bitmap  
            ivPhoto.setImageBitmap(imageBitmap)  
        }  
    }  
}
```

Példa: Fotó készítése beépített kamera alkalmazással



Video készítés lépései

- Hasonló, mint a kép készítés
- Intent paraméterek:
 - > *MediaStore.EXTRA_OUTPUT*: felvett video állomány helye (Uri)
 - > *MediaStore.EXTRA_VIDEO_QUALITY*: 0 és 1 közötti float, 0: legrosszabb minőség és legkisebb file méret
 - > *MediaStore.EXTRA_DURATION_LIMIT*: Video hossz korlát másodpercben
 - > *MediaStore.EXTRA_SIZE_LIMIT*: Méret korlát a felvett videóra
- Visszatéréskor az Activity `onActivityResult(...)` függvényében visszaadott Intent `getData()` függvényével lekérdezhető az alapértelmezett mentés helye

Példa: Visszatérés video felvételből

```
val CAPTURE_VIDEO_REQUEST_CODE = 200

override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
    if (requestCode == CAPTURE_VIDEO_REQUEST_CODE) {
        if (resultCode == Activity.RESULT_OK) {
            // Video felvéve és elmentve a megfelelő könyvtárba
            Toast.makeText(this,
                "Video mentés helye:\n" + data?.data?, Toast.LENGTH_LONG).show()
        } else if (resultCode == Activity.RESULT_CANCELED) {
            // A felhasználó mégsem mentett videót
        } else {
            // Video felvétel nem sikerült, figyelmeztessük a felhasználót
        }
    }
}
```

Video felvétel programozottan

- *Camera open()* és *release()* mellett a *lock()* és *unlock()*-ot is kezelnünk kell (4.0-tól már nem! 😊)
- Felvételhez több művelet szükséges, mint a kép készítéshez
- Ügyeljünk arra, hogy ne hagyjunk felesleges video állományt a készüléken, mert sok helyet foglalhat
- **Mindig szabadítsuk fel a Camera eszközt!**

Video lejátszása

```
<VideoView
    android:id="@+id/videoView"
    android:layout_width="200dp"
    android:layout_height="200dp"
/>
```

```
VideoView videoView =
    (VideoView) this.findViewById(
        R.id.videoView);
MediaController mc =
    new MediaController(this);
videoView.setMediaController(mc);
videoView.setVideoURI(
    Uri.parse("https://www.sample-
    videos.com/video/mp4/480/big_buck
    _bunny_480p_1mb.mp4"));
// videoView.setVideoPath(
//     "/sdcard/movie.mp4");
videoView.requestFocus();
videoView.start();
```

Egyedi kamera nézet készítése

- Camera API V2

- > <https://developer.android.com/reference/android/hardware/camera2/package-summary>

- GoldenEye osztálykönyvtár:

- > <https://github.com/infinum/Android-GoldenEye>

```
val goldenEye = GoldenEye.Builder(activity).build()
```

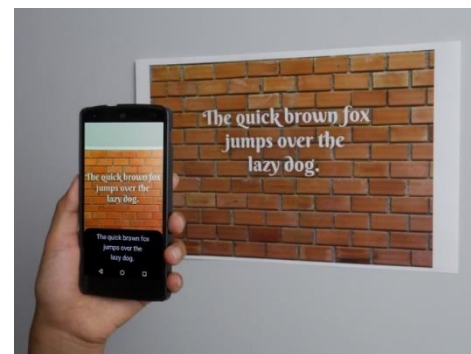
```
val backCamera = goldenEye.availableCameras.find { it.facing == Facing.BACK }
```

```
goldenEye.open(textureView, backCamera, initCallback)
```

```
goldenEye.takePicture(pictureCallback)
```

Karakter/szöveg felismerés

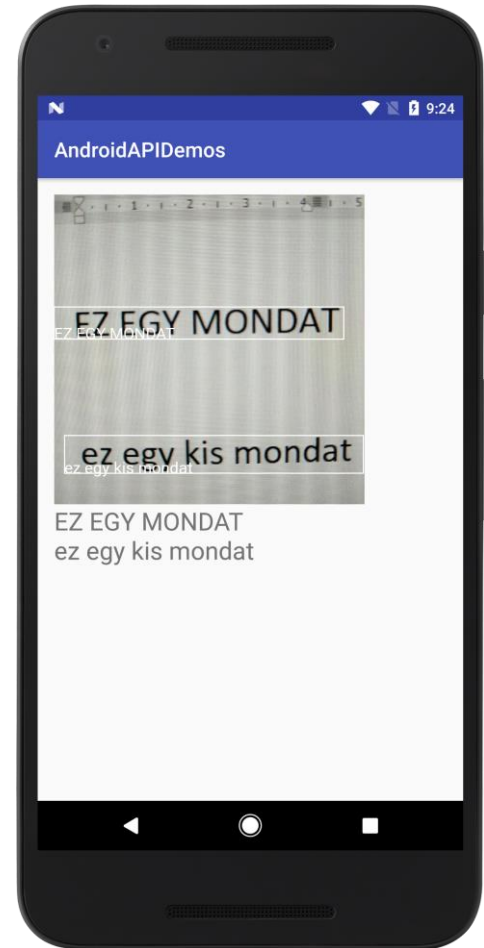
- Play Services -> Vision API



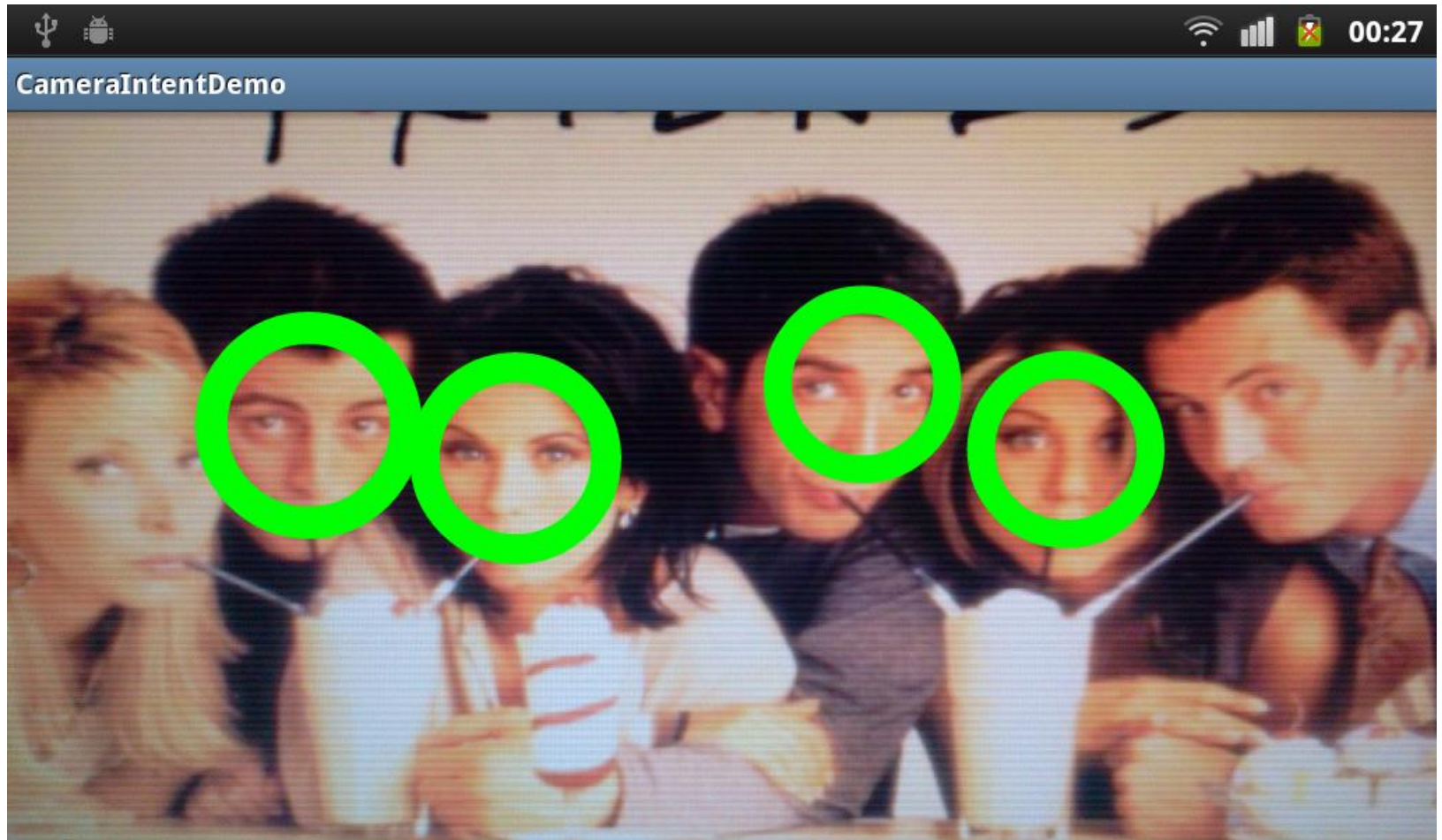
- Arcfelismerés
- QR/Bar/stb kód detektálás
- Karakter/szöveg felismerés
- Gradle:
 - > com.google.android.gms:play-services-vision:16.2.1

Szöveg felismerés

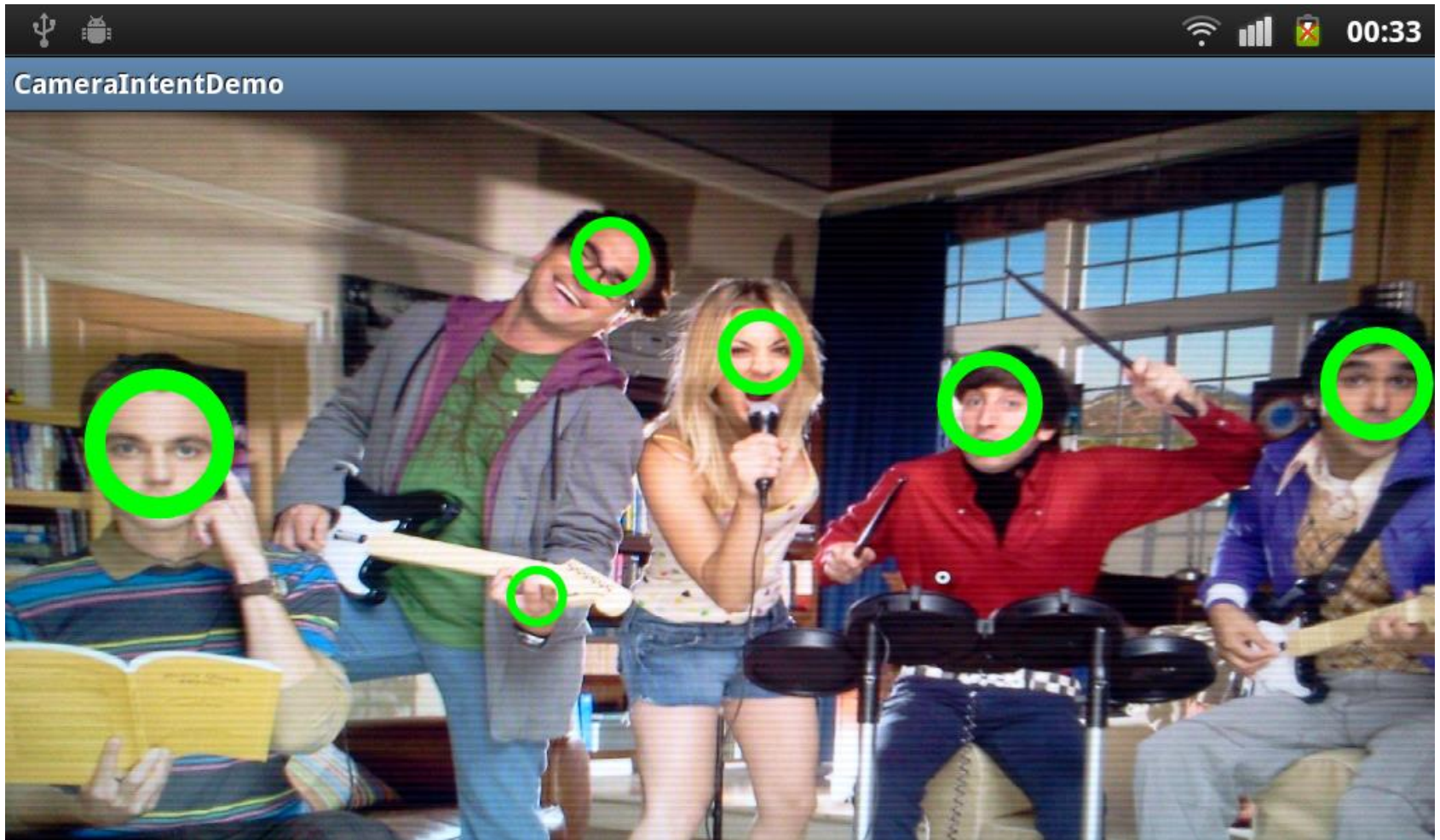
- Szöveg és mondat felismerés
- Szöveg blokkok azonosítása
- Javasolt beállítás:
 - > `android:keepScreenOn="true"`



Arcfelismerés példa 1/2



Arcfelismerés példa 2/2



Arcfelismerés

- Beépített *android.media.FaceDetector* osztály az arcfelismeréshez
- Kép alapján automatikusan detektálja az arcokat (Face osztály)
- Megadható a maximálisan detektálandó arc egy képen
- Nem real time a detektálás!
- Használat:
 - > Kamera focus automatikus beállítása
 - > Kép effekt alkalmazása az arcokon

Face osztály képességei

- *Confidence* (megbízhatóság): helyes detektálás valószínűsége
- *EyesDistance*: szemek közti távolság
- *MidPoint*: két szem közötti középpont PointF-ben
- *Pose*: A felismert arc Euler szöge a megadott tengelyhez képest (elfordulás a kiválasztott X, Y vagy Z tengelyekhez képest)

Vision API változások

- „Offline” Vision API Deprecated lesz hamarosan
- Áttérés ML Kit-re (Firebase)

„The Mobile Vision API is now a part of [ML Kit](#). We strongly encourage you to try it out, as it comes with new capabilities like on-device image labeling! Also, note that we ultimately plan to wind down the Mobile Vision API, with all new on-device ML capabilities released via ML Kit.”

ML Kit – Machine Learning

- Gépi tanulás könyvtárak felhőbe kiszervezve
- Alap API-k:
 - > Kép azonosítás (objektum, tevékenység, termék, állat, stb.)
 - > Szöveg felismerés
 - > Arc felismerés
 - > Vonalkód olvasás
 - > Épületek/ismert helyek azonosítása
 - > Smart reply (hamarosan...)
- Firebase integráció szükséges hozzá

ML Kit demo

- Eredeti forrás:
 - > <https://github.com/riggaroo/android-demo-mlkit>
- Teendők:
 - > Projekt importálása
 - > Firebase projekt létrehozása és google-services.json másolása a projekt *app* modulja alá
 - > Firbase conslse-on ML Kit bekapcsolása
- Fizetős funkciók (pl. Landmark)



QR és Bar kód olvasás külső library-val 1/2

- Zxing library
- Régóta népszerű a fejlesztők körében
- Teljes képernyős és saját nézetes scanner felület
- Valós idejű olvasás
- További információk:
 - > <https://github.com/zxing/zxing>
 - > <https://github.com/dm77/barcodescanner>

QR kód olvasó példa

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <me.dm7.barcodescanner.zxing.ZXingScannerView
        android:id="@+id/zxingView"
        android:layout_width="match_parent"
        android:layout_height="300dp"/>

    <TextView
        android:id="@+id/tvScan"
        android:textSize="24sp"
        android:autoLink="all"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"/>

</LinearLayout>
```


QR kód olvasó példa

```
class MainActivity : AppCompatActivity(), ZXingScannerView.ResultHandler {  
    public override fun onCreate(state: Bundle?) {  
        super.onCreate(state)  
        setContentView(R.layout.activity_main)  
    }  
    public override fun onResume() {  
        super.onResume()  
        startCamera()  
    }  
    @WithPermissions(  
        permissions = [Manifest.permission.CAMERA]  
    )  
    private fun startCamera() {  
        zxingView.setResultHandler(this)  
        zxingView.startCamera()  
    }  
    public override fun onPause() {  
        super.onPause()  
        zxingView.stopCamera()  
    }  
    override fun handleResult(rawResult: Result) {  
        tvScan.text = rawResult.text  
        zxingView.resumeCameraPreview(this)  
    }  
}
```

QR kód olvasó példa

- Modern QR kód generáló szolgáltatás:
> <https://www.qrcode-monkey.com/>



Hangok lejátszása és felvétele

Egyszerű hangok lejátszása

- Az Android platform lehetőséget biztosít arra, hogy egyszerű figyelmeztető hangokat gyorsan és egyszerűen le tudjunk játszani
- Lejátszáshoz szükség van egy *MediaPlayer* objektumra
- *RingtoneManager*: figyelmeztető hangok elérése (Alarm, Notification, RingTone)
- *ToneGenerator*: bonyolultabb hang szekvenciák előállítása

Példa: figyelmeztető hang lejátszása

```
private fun playNotificationTone() {  
    val uriNotif = RingtoneManager.getDefaultUri(  
        RingtoneManager.TYPE_NOTIFICATION  
    )  
    val r = RingtoneManager.getRingtone(  
        applicationContext, uriNotif  
    )  
    r.play()  
}
```

Android média lejátszás

- Az Android Multimédia API lehetővé teszi, hogy egyszerű módon lejátszunk média tartalmakat különböző helyről:
 - > Alkalmazás erőforrás (*res/raw* könyvtár)
 - > File
 - > Hálózaton keresztül
- A jelenlegi média API hangot csak a standard hangkimeneten v. Bluetooth Headset-en tud lejátszani, hívásba például nem tud hangot bekeverni
- Legfontosabb osztályok:
 - > *MediaPlayer*: elsődleges osztály hang és videó lejátszásához
 - > *AudioManager*: audio forrás (felvételhez) és kimenet megadása

Média lejátszás erőforrásból

- Nincs szükség *prepare()* hívásra, ebben az esetben a *create()* elvégzi ezt

```
mediaPlayer = MediaPlayer.create(this@MainActivity,  
    R.raw.mysound)  
mediaPlayer.start()
```

Média lejátszás HTTP URL-ről

```
class MainActivity : AppCompatActivity(), MediaPlayer.OnPreparedListener {  
  
    private var mediaPlayer: MediaPlayer? = null  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        btnStart.setOnClickListener {  
            mediaPlayer = MediaPlayer.create(this@MainActivity,  
                Uri.parse("http://babcomaut.aut.bme.hu/tmp/demo.mp3"))  
            mediaPlayer?.setOnPreparedListener(this@MainActivity);  
        }  
  
        btnStop.setOnClickListener {  
            mediaPlayer?.stop();  
        }  
    }  
  
    override fun onPrepared(player: MediaPlayer) {  
        mediaPlayer?.start();  
    }  
  
    override fun onStop() {  
        mediaPlayer?.stop()  
        super.onStop()  
    }  
}
```


Hangfelvétel folyamata 1/2

1. *android.media.MediaRecorder* példány létrehozása
2. Audio forrás beállítása:
MediaRecorder.setAudioSource(...), pl.:
MediaRecorder.AudioSource.MIC
3. Kimeneti formátum beállítása
4. Kimeneti file beállítása
5. Audio encoding beállítása
6. Felvétel előkészítése: *MediaRecorder.prepare()*

Hangfelvétel folyamata 2/2

- Felvétel indítása: *MediaRecorder.start()*
- Felvétel leállítása: *MediaRecorder.stop()*
- Felszabadítás: *MediaRecorder.release()* (nagyon fontos!)
- Szükséges engedélyek:
 - > `<uses-permission
android:name="android.permission.RECORD_AUDIO"/>`
 - > `<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>`

Hangfelvétel példa előkészítés

```
private val fileName = Environment.getExternalStorageDirectory()  
    .getAbsolutePath() + "/audiorecordtest.3gp"  
private var myPlayer: MediaPlayer? = null  
private var myRecorder: MediaRecorder? = null
```

Hangfelvétel indítása

```
private fun startRecording() {  
    try {  
        myRecorder = MediaRecorder()  
        myRecorder?.setAudioSource(  
            MediaRecorder.AudioSource.MIC  
        )  
        myRecorder?.setOutputFormat(  
            MediaRecorder.OutputFormat.THREE_GPP  
        )  
        val outputFile = File(fileName)  
        if (outputFile.exists())  
            outputFile.delete()  
        outputFile.createNewFile()  
        myRecorder?.setOutputFile(fileName)  
  
        myRecorder?.setAudioEncoder(  
            MediaRecorder.AudioEncoder.AMR_NB  
        )  
        myRecorder?.prepare()  
        myRecorder?.start()  
    } catch (e: IOException) {  
        Log.e(LOG_TAG, "prepare() failed")  
    }  
}
```

Hangfelvétel leállítása

```
private fun stopRecording() {  
    myRecorder?.stop()  
    myRecorder?.release()  
}
```

Hanglejátzás indítása

```
private fun startPlaying() {  
    myPlayer = MediaPlayer()  
    try {  
        myPlayer?.setDataSource(fileName)  
        myPlayer?.prepare()  
        myPlayer?.start()  
    } catch (e: IOException) {  
        Log.e(LOG_TAG, "prepare() failed")  
    }  
}
```

Hanglejátzás befejezése

```
private fun stopPlaying() {  
    myPlayer?.release()  
}
```

Hangfelismerés

- SpeechRecognition API
- Folyamat:
 - > Hangfelismerés indítása
 - > Értesítések a felismerés állapotáról: indítás, vége, hiba, stb.
 - > Eredmény egy „String tömb”

Hangfelismerés példa - indítás

```
val sr = android.speech.SpeechRecognizer
    .createSpeechRecognizer(this)
sr.setRecognitionListener(SpeechRecognizer())
...
val intent = Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH)
intent.putExtra(
    RecognizerIntent.EXTRA_LANGUAGE_MODEL,
    RecognizerIntent.LANGUAGE_MODEL_FREE_FORM
)
//intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_PREFERENCE,
//    "hu-HU");
intent.putExtra(
    RecognizerIntent.EXTRA_CALLING_PACKAGE,
    "hu.aut.android.ttsvoicerecogkotlin"
)

intent.putExtra(RecognizerIntent.EXTRA_MAX_RESULTS, 5)
sr.startListening(intent)
```

Hangfelismerés példa - eredmények

```
override fun onResults(results: Bundle) {  
    val str = String()  
    Log.d(TAG, "onResults $results")  
    val data = results  
        .getStringArrayList(  
            android.speech.SpeechRecognizer.RESULTS_RECOGNITION  
        )  
    tvDetectedText.text = ""  
  
    for (text in data) {  
        tvDetectedText.append(text + "\n")  
    }  
}
```

TextToSpeech

- Szövegfelolvasó API
- Google Translate-n ismert megoldás
- Nyelveket külön kell beszerezni
- Egyszerű használat
- Indítás előtt meg kell várni az inicializálás befejeződését

Példa: TextToSpeech 1/3

- onCreate(...)-be:

```
tts = TextToSpeech(this, this)
```

```
btnRead.setOnClickListener { speak(etData.text.toString()) }
```

```
private fun speak(text: String) {  
    tts.speak(text, TextToSpeech.QUEUE_FLUSH, null)  
}
```

Példa: TextToSpeech 2/3

```
override fun onInit(status: Int) {  
    if (status == TextToSpeech.SUCCESS) {  
  
        //int result = tts.setLanguage(Locale.forLanguageTag("hu-HU"));  
        val result = tts.setLanguage(Locale.forLanguageTag("en-EN"))  
  
        // tts.setSpeechRate((float) 0.8);  
        // tts.setPitch(1.0f); tts.setPitch(1.1f);  
  
        if (result == TextToSpeech.LANG_MISSING_DATA || result ==  
            TextToSpeech.LANG_NOT_SUPPORTED) {  
            val installIntent = Intent()  
            installIntent.action = TextToSpeech.Engine.ACTION_INSTALL_TTS_DATA  
            startActivity(installIntent)  
        } else {  
            speak("Speech system works perfectly!")  
        }  
  
    } else {  
        val installIntent = Intent()  
        installIntent.action = TextToSpeech.Engine.ACTION_INSTALL_TTS_DATA  
        startActivity(installIntent)  
    }  
}
```

Példa: TextToSpeech 3/3

```
override fun onDestroy() {  
    super.onDestroy()  
  
    try {  
        tts.stop()  
        tts.shutdown()  
    } catch (e: Exception) {  
        e.printStackTrace()  
    }  
  
    try {  
        sr.destroy()  
    } catch (e: Exception) {  
        e.printStackTrace()  
    }  
  
}
```

Média használati javaslatok

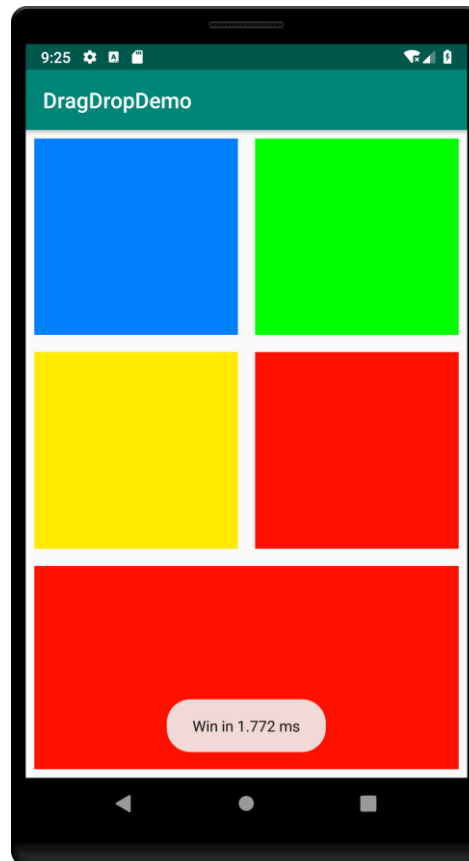
- A multimédia kezelő mobil alkalmazások száma rohamosan növekszik
- Média eszközök kreatív használata
- Mindig pontosan valósítsuk meg a média lejátszás teljes életciklusát (pl. *prepare()*)
- Beépített kamera alkalmazás használata
- Multimedia framework
- Ne feledkezzünk el az erőforrások felszabadításáról!

Drag & Drop támogatás

- Gyakorlatilag bármelyik komponens „drag”-elhető:
 - > `View.startDrag(...)`
- Felületek/komponensek feliratkozhatnak „drag” esemény érzékelésre:
 - > `setOnDragListener(...)`
- Többféle „drag” esemény:
 - > `ACTION_DRAG_STARTED`
 - > `ACTION_DRAG_ENTERED`
 - > `ACTION_DRAG_EXITED`
 - > `ACTION_DROP`

Gyakorljunk!

- Készítsünk egy színválasztó Drag&Drop játékot reakció idő mérésre!



Összefoglalás

- WebView
- TCP/IP és UDP lehetőségek
- A multimédia kezelő mobil alkalmazások száma rohamosan növekszik
- Média eszközök kreatív használata
- Mindig pontosan valósítsuk meg a média lejátszás teljes életciklusát (pl. *prepare()*)
- Beépített kamera alkalmazás használata
- Multimedia framework
- Ne feledkezzünk el az erőforrások felszabadításáról!
- Drag & Drop

Kérdések

