

Prvé aplikácie

ViewBinding, Nullables, Layouts, životný cyklus



borovan 'at' ii.fmph.uniba.sk





do Listu

Odovzdávajte .zip, ktorý v koreni obsahuje

- README s popisom ovládania, resp. vami implementované detaily nespomenuté v zadaní,
- projekt AS s vymazaným build adresárom (app/build) výrazne schudne, a potom nie je problém s limitom 30MB pre List
- ak sa inak nedá, link na alternatívne úložisko, ale tiež bez app/build
- vytiahnutým .apk súborom z app/build/intermediares/apk/debug do koreňa .zipu, aby ho opravujúci nemuseli hľadať v útrobách projektu

Za nedodržanie tohoto formátu sa strhávajú body, ktoré sú úmerné času cvičiaceho, aby:

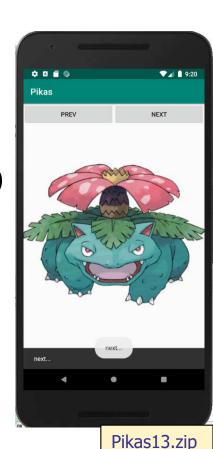
- zo zdrojáku pochopil ovládanie vašej apky (ak ste zabudli README)
- kompilovaním vášho projektu, ak ste neposkytli .apk
- resp. uviedli minSDK=33, leby vy máte zariadenie s Android 13

Code review robíme pre vás, tak nám pomôžte. Ďakujeme za pochopenie.

Príklad jednoduchej aplikácie

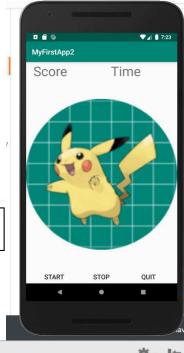
Ilustruje:

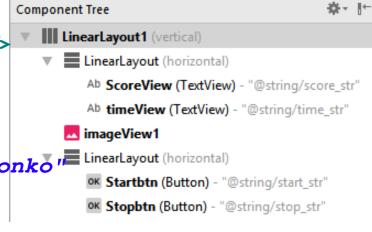
- príklad návrhu (vyklikania) jednoduchého GUI (single activity app)
- logovanie udalostí ako efektívny prostriedok ladenia pomocou
 - Log.i(...)
 - Toast.make(...)
 - Snackbar.make(...)
- používanie Image/Vector Asset (drawable/mipmap)
- používanie resource editora (pri definovaní strings.xml)
- používanie layout editora pri tvorbe rozhrania (ešte bude)
- eventhandler (.setOnClickListener) previazané cez
 - findViewById<Button>(R.id.quitBtn)
 - prevBtn.setOnClickListener({ })
 - property android:onClick="nextOnClickListener"
- aktivitu a jej životný cyklus



Ako by to malo vyzerať

```
<LinearLayout
                                          Žiadne warnings
    <TextView
         android:id="@+id/ScoreView"
         android:text="@string/score_str"/>
    <TextView
         android:id="@+id/timeView"
         android:text="@string/time_str" />
</LinearLayout>
<ImageView</pre>
    android:id="@+id/imageView1"
    android:contentDescription="@string/dronko" LinearLayout (horizontal)
    android:src="@drawable/ic_launcher" />
<LinearLayout</pre>
    <Button
        android:id="@+id/Startbtn"
        android:text="@string/start_str" />
    <Button
        android:id="@+id/Stopbtn"
        android:text="@string/stop_str" />
</LinearLavout>
```





zjednodušené pre účely slajdu

Väzba komponentov v kóde

- val btn = findViewById<Button>(R.id.button)
- val iv = findViewById<ImageView>(R.id.imageView1)

```
plugin kotlin-android-extensions
Kotlin Android Extensions is deprecated-means that using Kotlin Synthetics for view binding is no longer supported. If your app uses Kotlin synthetics for view binding, use this guide to migrate to view binding.
```

```
id 'com.android.application'
id 'kotlin-android'

id 'kotlin-android-extensions'
```

- import syntetic pomocou Alt-Enter
- import kotlinx.android.synthetic.main.activity_main.*

Unresolved reference: startBtn

```
Old school, java style

val s = findViewById<Button>(R.id.startBtn)

val iv = findViewById<ImageView>(R.id.imageView)

Deprecated 2017-2020

startBtn.setText("Start")
```

@Parcelize od 2020

Create local variable 'startBtn' Alt+Shift+Enter More actions... Alt+Enter

8. The Basics of the Android Studio Code Editor

Väzba komponentov v kóde

viewBindings

```
build.gradle.kts
android {
    buildFeatures {
        viewBinding = true
    }
```

```
Konvencia: XyzActivity[.kt]
```

- má layout activity_xyz.xml
- binding: ActivityXyzBinding

18. An Overview of Android View Binding

View Binding

- findViewById() as Button, findViewById<Button>() klasické, "javish" riešenie
- syntetics ketlin andreid extensions plugin deprecated ed 2020
- ďalší spôsob prepojenia komponentov (View) z .xml layoutu s kódom
- pozor: nepliesť si to s Data Binding, to príde s JetPack library, to je zložitejšie

```
android {
    buildFeatures {
        viewBinding = true
    }
    compileSdkVersion 30
    defaultConfig {
        applicationId "com.example.pikas"
        minSdkVersion 23
        targetSdkVersion 30
```

za

```
val binding = ActivityMainBinding.inflate(layoutInflater)
setContentView(binding.root)
```

- 3) miesto referencie nejakého View, napr. imageView2, použijete binding.imageView2
- 4) ak mimo metódy onCreateView potrebujete premennú binding, urobte ju lateinit var

```
lateinit var binding : ActivityMainBinding
```

- 5) ak sa vaša aktivita nevolá MainA..., tak nahraďte zelené za jej meno
- 6) objavte, čo je apply, resp. iné scoping functions

View Binding

príklad apply



Fyzické zariadenie

Chapter 7

7. Testing Android Studio Apps on a Physical Android Device

Android Debug Bridge (ADB)

C:\Users\borovan>adb -s emulator-5554 emu kill OK: killing emulator, bye bye OK C:\Users\borovan>adb devices
List of devices attached
emulator-5554 device

C:\Users\borovan>adb devices
List of devices attached
XVV7N17331000103 device

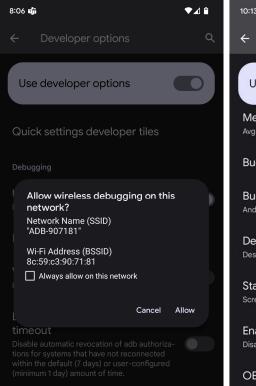
USB Debugging on Android device, stay awake

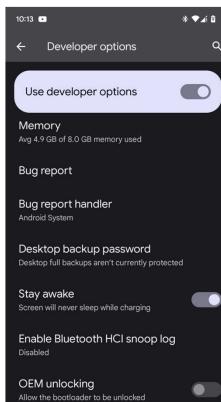


Wireless debugging...









Logovanie

Tri najbežnejšie spôsoby ako (logovať, debugovať):

- trieda Log, metóda Log.i loguje do okna Logcat, filtrujte podľa TAGU metódy
 - definuite si TAG ako konštantu
- trieda Toast, metóda Toast.make potrebuje context (zjednodušene aktivita, v ktorej sa toastuje)
 - nezabudnite na volanie .show()
- trieda Snackbar, metóda Toast.make pridať import

```
import com.google.android.material.snackbar.Snackbar
prevBtn2.setOnClickListener {
   Toast.makeText(this@MainActivity, "prev...", Toast.LENGTH SHORT)
        .show()
    Log.i(TAG, "prev...")
    Snackbar.make(view, "prev...",
       Snackbar.LENGTH_SHORT) .setAction("Action", null) .show()
    alebo
                               .setAction(R.string.action,
               View.OnClickListener { nextOnClickListener(it)
               }).show()
```

Pikas13.zip

Logovanie

val TAG = "PIKAS"
Log.i(TAG, "prev...")

Pikas13.zip

```
Y- package:mine tag:PIKAS
HUAWEI EVA-L19 (XVV7N17331000103) Android 7, API 24
1 Q-
                                                 ₽ Cc W .*
                                                                  ↑ ↓ □ †<sub>1</sub> ¬<sub>1</sub> ≅<sub>1</sub> □ Y
Ш
   2023-09-26 10:43:40.786 16997-16997 PIKAS
                                                                       com.example.pikas13
                                                                                                                     prev...
                                                                       com.example.pikas13
   2023-09-26 10:43:43.241 16997-16997 PIKAS
                                                                                                                    prev ...
                                                                       com.example.pikas13
   2023-09-26 10:45:01.558 18234-18234 PIKAS
                                                                                                                    onTICK
                                                                       com.example.pikas13
   2023-09-26 10:45:02.559 18234-18234 PIKAS
                                                                                                                    onTICK
5
                                                                       com.example.pikas13
   2023-09-26 10:45:02.963 18234-18234 PIKAS
                                                                                                                    next...
                                                                       com.example.pikas13
   2023-09-26 10:45:03.174 18234-18234 PIKAS
                                                                                                                    next...
Ш
                                                                       com.example.pikas13
   2023-09-26 10:45:03.380 18234-18234 PIKAS
                                                                                                                    next...
                                      Y- package:mine tag:CYKLUS
HUAWEI EVA-L19 (XVV7N17331000103) Android 7, API 24
   2023-09-26 10:49:22.941 20719-20719 CYKLUS
                                                                      com.example.applifecycle13
                                                                                                                   onCreate
   2023-09-26 10:49:22.985 20719-20719 CYKLUS
                                                                      com.example.applifecycle13
                                                                                                                   onStart0
   2023-09-26 10:49:23.012 20719-20719 CYKLUS
                                                                      com.example.applifecycle13
                                                                                                                   onResume0
   2023-09-26 10:49:38.481 20719-20719 CYKLUS
                                                                      com.example.applifecycle13
                                                                                                                   onPause
   2023-09-26 10:49:38.713 20719-20719 CYKLUS
                                                                      com.example.applifecycle13
                                                                                                                   onStop1
   2023-09-26 10:49:38.746 20719-20719 CYKLUS
                                                                      com.example.applifecycle13
                                                                                                                   onDestroy1
                                                                                                             AppLifeCycle13.zip
```

Pikas

```
override fun onCreate(savedInstanceState: Bundle?)
   super.onCreate(savedInstanceState)
   setContentView(R.layout.activity_main)
   var i = 0
   var imqs = arrayOf(
     ContextCompat.getDrawable(applicationContext,
                                R. drawable. butterfree),
     imageView2.setImageDrawable(imgs[i])
     prevBtn2.setOnClickListener {
        Toast.makeText(this@MainActivity,
                       "prev...", Toast. LENGTH_SHORT) . show()
        if (--i < 0) i += imgs.size
        imageView2.setImageDrawable(imgs[i])
     nextBtn2.setOnClickListener {
        Toast.makeText(this@MainActivity,
                       "next...", Toast. LENGTH_LONG) . show()
        i = (++i) %imqs.size
        imageView2.setImageDrawable(imgs[i])
                                                              Pikas13.zip
```

Pikas

Konvertor EURO USD

(logika)

Jednoduchá aplikácia na konverziu kurzov USD EURO

- s modifikovateľným TextView pre zadanie sumy (reálneho čísla)
- RadioButton pre výber smeru konverzie
- s nemodifikovateľným poľom pre výsledok
- Button Konvertuj pre vykonanie akcie, výpočet

```
override fun onCreate(savedInstanceState: Bundle?)
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)
convertBtn.setOnClickListener({
    Toast.makeText(this, "convert", Toast.LENGTH_SHORT).show();
if (binding.inputText.text.isNotEmpty()) {
    val input = binding.inputText.text
    var output = input
    val exchangeRate = 1.07f
    if (eur2usd.isChecked) output = exchangeRate * output
    if (usd2eur.isChecked) output = output / exchangeRate
    binding_outputText.setText("$output") // Konvertor13.zip
```

KONVERTUJ

Konvertor EURO USD

convertBtn

layout_width

layout height

id

onClick

▼ Declared Attributes

convertBtn

match_parent

wrap_content

@string/konvertujBtn

convertBtn

convert

(setOnClickListener)

```
// very old fashion
  val cBtn = findViewById<Button>(R.id.convertBtn)
  cBtn.setOnClickListener( { v -> convert(v) } )
  cBtn.setOnClickListener { convert(it) }

// old fashion
  convertBtn.setOnClickListener { v -> convert(v) }
  convertBtn.setOnClickListener { convert(it) }
fun convert(v: View) {
```

```
Toast.makeText(this, "convert", Toast.LENGTH_SHORT).show()
                binding.apply {
                  if (inputText.text.isNotEmpty()) {
                    val input = inputText.text.toString().toFloat();
                    var output = input
                     val exchangeRate = 1.07f
                      if (eur2usd.isChecked) output = exchangeRate * output
                      if (usd2eur. isChecked) output = output / exchangeRate
extension
                    outputText.setText("${output.format(2)}")
                                                                   }} }
metóda
            fun Float.format(digits: Int) =
Float
                                                                   Konvertor13.zip
                java.lang.String.format("%.${digits}f", this)
```

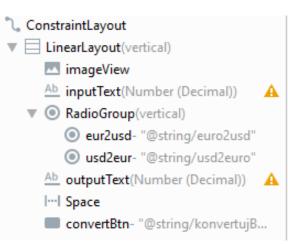
a

4

Konvertor EURO USD

(layout)







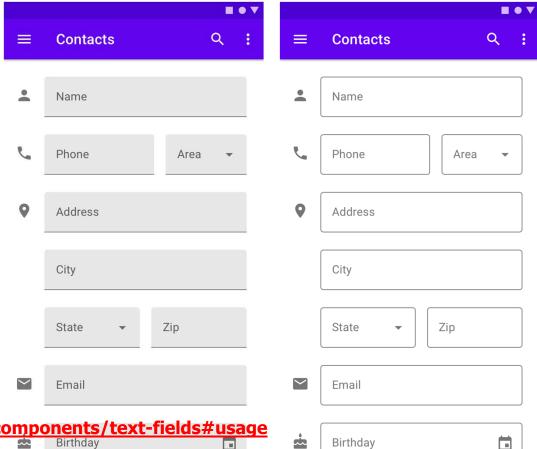
Text Fields

prvý dotyk s Material Design

Material Design je Google knižnica GUI komponentov unifikovaná pre Android, iOS, Flutter, web, ... dependencies {

implementation 'com.google.android.material:material:1.9.0'

zahŕňa Button, Text fields, SnackBars, Sliders, a mnoho ďalších vizuálnych komponentov Views



TextInput[Layout/EditText]

```
<com.google.android.material.textfield.TextInputLayout</pre>
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:startIconDrawable="@drawable/ic_launcher_foreground"
    app:startIconContentDescription="@string/iconDescription"
    app:startIconCheckable="true"
                                                           TextViewDemo
    app:endIconMode="clear_text"
    app:counterEnabled="true"
                                                           borovan@ii.fmph.uniba.sk
    app:counterMaxLength="15"
                                                                USEF
    app:errorEnabled="true">
                                                                pedro
    <com.google.android.material.textfield.TextInputEditText</pre>
                                                                              5/15
           android:id="@+id/userTV"
                                                           password
                                                                              0
           android:layout_width="match_parent"
           android:layout_height="wrap_content"
           android:hint="@string/userHint"
           android:maxLength="15"
           android:inputType="textPersonName" />
</com.google.android.material.textfield.TextInpu</pre>
                                                                    vbnm 🖾
https://material.io/components/text-fields#usage
                                                           7123
                                                               TextViewDemo13.zip
```

TextWatcher

```
override fun beforeTextChanged(s: CharSequence, ...) { }
   override fun afterTextChanged(s: Editable?) { }
   override fun onTextChanged(s: CharSequence?, ...) {
       button.isEnabled =
                 emailTV.text?.isNotEmpty()?:false &&
                 userTV.text?.isNotEmpty()?:false &&
                passwordTV.text?.isNotEmpty()?:false
       button.isEnabled =
           if (emailTV.text != null && userTV.text != null &&
               passwordTV.text != null)
             emailTV.text(!!.)isNotEmpty() &&
             userTV.text!!.isNotEmpty() &&
             passwordTV.text!!.isNotEmpty()
           else
               false
emailTV.addTextChangedListener(textWatcher)
userTV.addTextChangedListener(textWatcher)
passwordTV.addTextChangedListener(textWatcher)
                                                TextViewDemo13.zip
```

12. Kotlin Data Types, Variables, and Nullability

Kotlin – pokračovanie

Cheat sheets

- https://www.programming-idioms.org/cheatsheet/Kotlin
- https://github.com/vmandro/Prednasky/tree/master/Kotlin

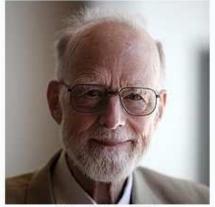
The billion-dollar mistake

I call it my billion-dollar mistake. It was the invention of the **null** reference in 1965...This has led to innumerable errors, vulnerabilities, and system crashes, which have probably caused a billion dollars of pain and damage in the last forty years.

Kotlin Null Safety

Sir Tony Hoare

FRS FREng



Tony Hoare in 2011

Born Charles Antony Richard Hoare

11 January 1934 (age 85) Colombo, British Ceylon

Residence Cambridge
Other names C. A. R. Hoare

Alma mater University of Oxford (BA)

Moscow State University

Known for Quicksort

Quickselect Hoare logic Null reference

Communicating Sequential

Processes

Structured programming

Awards Turing Award (1980)

Harry H. Goode Memorial

Award (1981)

Faraday Medal (1985)

Computer Pioneer Award

(1990)

Kyoto Prize (2000)

IEEE John von Neumann

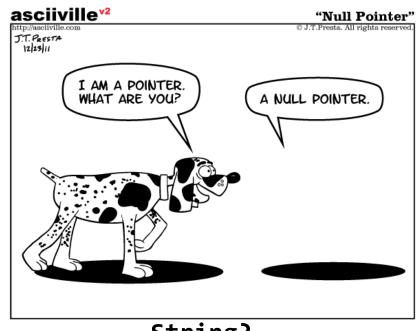
Medal (2011)

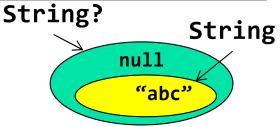
Nullables

To, čo je

- Optional v Jave, resp.
- Option v Scale, resp. kdekade iné inde

Napr. String? je typ pre reťazec alebo null Ale String je typ len pre SKUTOČNÝ REŤAZEC, not-null





Preto a:String? nemôžete priradiť do b:String, lebo čo, ak by a == null

Ak ste skalo-pevne presvedčený, že hodnota a:String? != null, môžete opatrne použiť BANG-BANG (!!) operátor a oklamať type-checker val b:String = a!!

Ak ale neviete, či a:String? =?= null, tak použijete tzv. **Elvis operátor** val c:String = a**?:**"default, ak je prázdny reťazec"





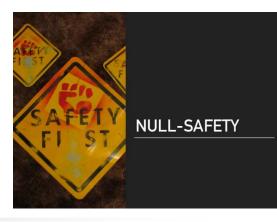


- Elvis operátor
 obj?:default = if (obj == null) default else obj
- Safe call operátor (Elvis na Žižku)
 obj?.m() = if (obj == null) null else obj.m()
- Not-null assertion (bang-bang !!)
 obj!! = if (obj != null) obj else N.P.E. null pointer Ex.
- Safe cast
 obj as? T = if (obj typeof T) obj else null
 obj as T = if (!obj typeof T) cast exception
- let
 obj?.let {...it...} = if (obj != null) {...it <- obj...}</pre>



J

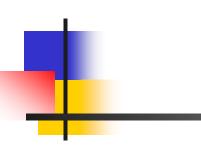
(ešte raz, podrobnejšie)



```
V Jave je typ String skutočný reťazec alebo null
V Kotline String je LEN skutočný reťazec a null nepatrí do typu String
Existuje String? čo je String alebo null, vo všobecnosti: T? = T ∪ null
T? Podobne vo Swingu, Java Optional[T] =, Scala Option[T]
fun foo(str : String?) {
  println(str)
   if (str != null) println(str.toUpperCase())
  println(str?.toUpperCase()) // safe call operátor
                       // x?.m == if (x != null) x.m else null
}
fun stringLen(s: String?): Int = s?.length?:0 // Elvis operátor
if (if (s == null) then null else s.length) == null then 0 else s.length
fun nonEmptystringLen(s: String?): Int {
   val sNotNull: String = s!! // určite nebude null,
             // ak bude tak exception kotlin.KotlinNullPointerException
   return sNotNull.length
```

Čo sú Scope functions

Marek Žitnik



V skratke: Scope funkcie sú funkcie ktoré vám umožňujú spusitiť blok kódu v kontexte objektu. Scope funkcie vytvárajú dočasný "scope", v ktorom môžte v objektu pristupovať cez **it** a **this**

Máme 5:

- let
- run
- also
- apply
- with

Sú medzi nimi 2 typy rozdielov a to:

- čo vracajú
- v akom kontexte referujeme na objekt



Čo to znamená a ako sa to používa

Najlepšie je to vidieť na príklade: Scope functions nám umožňujú spraviť z niečoho takéhoto

val cisla = mutableListOf("1" ,"11","100","101","110","111","1111") val resturnList = cisla.map {it.length}.filter { it < 3 }.let { println(it) }</pre>

Niečo takéto

```
val cisla = mutableListOf("1" ,"11","100","101","110","111","1111")
val resturnList = cisla.map {it.length}.filter { it < 3 }
println(resturnList)</pre>
```

Niečo podobné môžeme vidieť aj u

```
| var meno: String? = null | var vek: Int? = null | val clovek2 = s().αpply{ | meno = "Jozko Mrkvicka" | vek = 42 | vek
```

Príklad pre

```
class osoba {
   var meno: String = "Jozko Mrkvicka"
   var vek: Int = 42
}

val clovek : osoba? = osoba()

val bio = clovek?.run{
   println(meno)
   print(vek)
   vek + 2
   "Je to Jozko"

println(bio)

class osoba {
   var meno: String = "Jozko Mrkvicka"
   var vek: Int = 42

val clovek : osoba? = osoba()

val bio = clovek?.run{
   println(meno)
   print(vek)
   vek + 2
   "Je to Jozko"

}

println(bio)
```



Kedy použiť akú funkciu:

- with: keď potrebuješ operovať nad non-nullovím objektom
- let: keď potrebuješ použiť iba lambda výraz na nullable objektom a vyhnúť sa NullPointerException
- run: keď potrebuješ operovať na nullable objekte, použiť lambda výraz vyhnúť sa NullPointerEception
- apply: keď potrebuješ inicializovať alebo konfigurovať objekt
- also: keď potrebuješ dodatočné objektové konfigurácie alebo operácie



Content under the Creative Commons Attribution 4.0 BY License

with

Return: lambda result Context object: this

let

Return: lambda result Context object: it

run

Return: lambda result Context object: this

apply

Return: context object Context object: this

also

Return: context object Context object: it

Pikas

(rekapitulácia)

activity entry point

```
override fun onCreate(savedInstanceState: Bundle?) {
         super.onCreate(savedInstanceState)
         binding = ActivityMainBinding.inflate(layoutInflater)
         setContentView(binding.root)
         var i = 0
         var imqs = arrayOf(
           ContextCompat.getDrawable(applicationContext,
                                      R.drawable.butterfree)
           binding.imageView2.setImageDrawable(imgs[i])
           binding.prevBtn2.setOnClickListener {
              Toast.makeText(this, "prev...", Toast.LENGTH_SHORT).show()
View(s)
              if (--i < 0) i += imqs.size
              imageView2.setImageDrawable(imgs[i])
           binding.nextBtn2.setOnClickListener{
              Toast.makeText(this, "next...", Toast.LENGTH_LONG).show()
              i = (++i)%imgs.size
              imageView2.setImageDrawable(imgs[i])
```

Pikas13.zip

Pikas

const

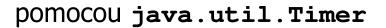
final

(stav sa mieša s views a logikou – riešenie príde)

```
val TAG = "PIKAS"
var i = 0
                                          State
var imgs = arrayOf<Drawable?>()
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = ActivityMainBinding.inflate(layoutInflater)
    setContentView(binding.root)
    imgs = arrayOf(ContextCompat.getDrawable(applicationContext,
                                        R.drawable.butterfree), ...)
    binding.imageView2.setImageDrawable(imgs[i])
    binding.prevBtn2.setOnClickListener { // it:View -> { ...
        if (--i < 0) i += imgs.size
       binding.imageView2.setImageDrawable(imgs[i])
// prepojene cez property android:onClick="nextOnClickListener"
fun nextOnClickListener(v: View) {
                                                     Common Attributes
    i = (++i) % imgs.size
                                                             @style/mystyle
                                                    style
    binding.imageView2.setImageDrawable(imgs[i])
                                                             clickOnNext
                                                    onClick
                                                                Pikas13.zip
```



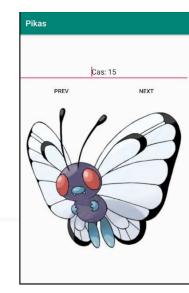
(asynchrónnost' - timer)



```
Timer("tik-tak").schedule(1000,1000) { // delay, period
    Log.d(TAG, "onTICK")
    cas++
    runOnUiThread { binding.time.setText("Cas: $cas ") }
} . run()
```

- nezabudnite na .run()
- runOnUiThread
 - má argument java.lang.Runnable, ktorý vykoná v hlavnom GUI vlákne

```
zabitie timera:
override fun onPause() {
    super.onPause()
    timer.cancel()
```



Pikas

(asynchrónnosť – count down)

pomocou android.os.CountDownTimer

```
Cas: 15

PREV NEXT
```

```
object:CountDownTimer(20000, 1000) { // 20sek, tik po 1sek
                            // how long, period
 tik
           override fun onTick(millisUntilFinished: Long) {
             Log.d(TAG, "onTICK")
             runOnUiThread {
                binding.time.setText(
                            "Cas: ${millisUntilFinished/1000}") }
game
           override fun onFinish() {
over
               Log.d(TAG, "onFinish")
                                             ukončenie appky
               exitProcess(-1)
       }.start()
```

19. Understanding Android Application and Activity Lifecycles



Životný cyklus apky

(prvý – zjednodušený nástrel)

local: 0

global: 0

Alt-Insert = Generate Override Implemented Methods:

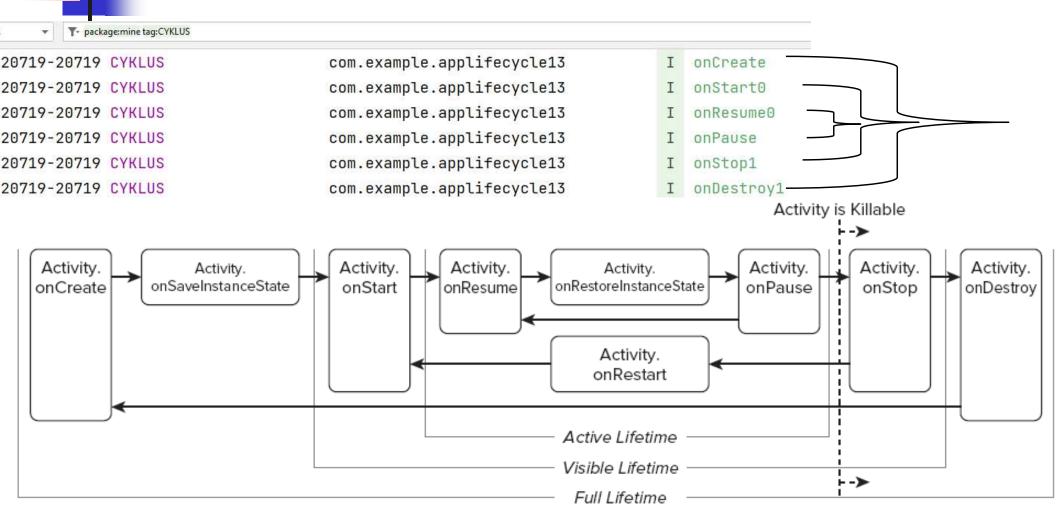
shared: 0

- override fun onDestroy()
- override fun onPause()
- override fun onRestart()
- override fun onRestoreInstanceState (Bundle savedInstanceState)
- override fun onResume()
- override fun onSaveInstanceState(Bundle outState)
- override fun onStart()
- override fun onStop()
- do každej metódy dáme kontrolný výpis, aby sme pochopili životný cyklus

```
override fun onCreate(Bundle savedInstanceState?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    Log.i("CYKLUS", "onCreate") // LOGUJTE, LOGUJTE
}
tag vhodný na filtrovanie
```

21. Android Activity State Changes by Example





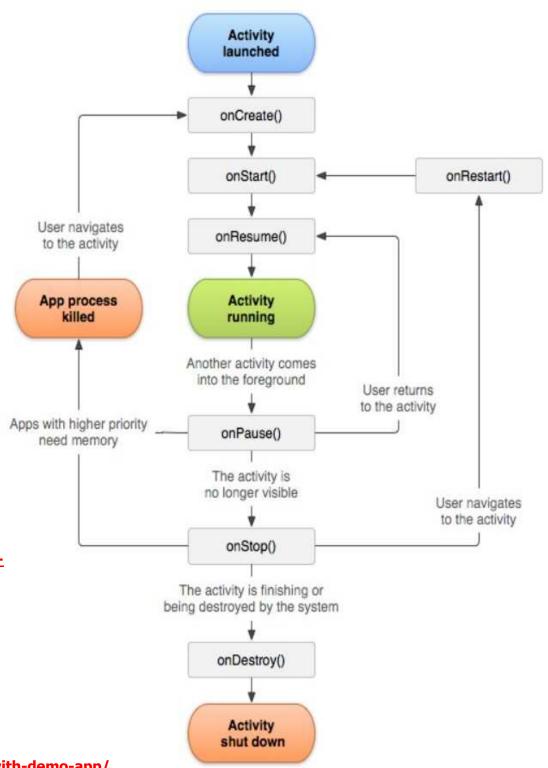
zdroj: Reto Meier: PA2AD

Chapter 20

20. Handling Android Activity State Changes



https://media.geeksforgeeks.org/wpcontent/uploads/20191125171002/Activity-Lifecycle-in-Android-Demo-App.mp4



22. Saving and Restoring the State of an Android Activity



Persistencia

(prvý dotyk)

global: 0

local: 0

shared: 0

- globalCounter je premenná, ktorá sa
 - pri onSaveInstanceState uloží do Bundle (HashMap<String, Value>)
 - pri onCreate(savedInstanceState: Bundle?) pride táto Bundle ako argument
- localCounter je bežná lokálna triedna premená v MainActivity
- sharedCounter je premenná, ktorá sa ukladá
 - pri onPause sa uloží do SharedPreferences (HashMap<String, Value>)
 - pri onResume Sa prečíta zo SharedPreferences
- všetky tri premenné sa inkrementujú pri onPause

Zistíte, že:

- aktivita, <u>ak zmení orientáciu, tak sa reštartne</u>, vytvorí sa nová inštancia a zavolá sa onCreate. Preto premenná localCounter sa vynuluje.
- ak si chcete niečo <u>uchovať aj po zmene orientácie aktivity</u>, treba to uložiť do bundle, zapíšete to tam v onSaveInstanceState a prečítate v onCreate
- ak si chcete niečo <u>uchovať aj po reštarte</u> aplikácie, treba to uložiť do SharedPreferences

22. Saving and Restoring the State of an Android Activity

Bundle?

Bundle má metódy [put/get][Int/Boolean/Char/Float/Any/...]

```
override fun onRestoreInstanceState(
           savedInstanceState: Bundle)
  super.onRestoreInstanceState(savedInstanceState)
  globalCounter = savedInstanceState.getInt("COUNTER")?:0
 ... OLD SCHOOL:
 if (savedInstanceState != null &&
    savedInstanceState.getInt("COUNTER") != null) {
    globalCounter = savedInstanceState!!.getInt("COUNTER")!!
 } else
    globalCounter = 0
override fun onSaveInstanceState(outState: Bundle,
                   outPersistentState: PersistableBundle) {
  super.onSaveInstanceState(outState, outPersistentState)
  outState.putInt("COUNTER", globalCounter)
```

• • •

22. Saving and Restoring the State of an Android Activity

SharedPreferences

```
SharedPreferences má metódy get[Int/Boolean/Char/Float/Any/...]
private lateinit var preferences: SharedPreferences
private lateinit var preferences: SharedPreferences
override fun onCreate(savedInstanceState: Bundle?) {
   super.onCreate(savedInstanceState)
   preferences = getSharedPreferences("lifecycle",
                                    Context. MODE_PRIVATE)
override fun onResume() {
   sharedCounter = preferences.getInt("kluc", 0)
                            val editor = preferences.edit()
override fun onPause() {
                             editor.putInt("kluc",
   preferences.edit {
                                     sharedCounter)
     putInt("kluc",
         sharedCounter)
                             editor.apply()
     apply()
```

Čo je Kotlin?







23. Understanding Android Views,

View Groups and Layouts

GUI komponenty

Layout

- LinearLayout (Vertical/Horizontal)
- TableLayout, ConstraintLayout
- RelativeLayout, GridLayout

View, ViewGroup

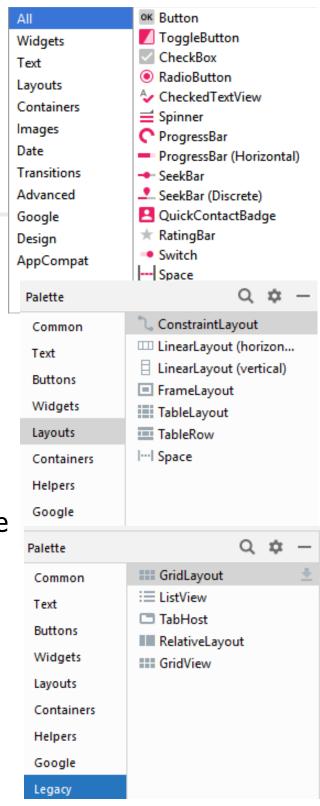
všetky viditeľné komponenty (widgets)

Activity - analógia Screenu (MIT), resp. Form/Frame najznámejšie podtriedy

- ListActivity pre ListView, zobrazenie zoznamu
- RecycleView
- MapActivity pre MapView (zobrazenie mapy)

Fragment (>= API level 11)

reusable UI components



Layouts

(match_parent, wrap_content)

- FrameLayout objeky umiestni v ľavom hornom rohu
- LinearLayout horizontálny/vertikálny | | | | | |
- RelativeLayout dovolí umiestniť objekty relatívne k pozíciám iných objektov
- ConstraintLayout (support library, API 9, od Android Studio 2.2)

android:src="@drawable/ic_launcher" />

GridLayout (od API Level 14)

<FrameLayout</pre>

```
android:id="@+id/FrameLayout1"
android:layout_width="match_parent"
android:layout_height="match_parent"
<ImageView
android:id="@+id/imageView1"
android:layout_width="match_parent" --roztiahni podľa
android:layout_height="match_parent" -- rodičovského</pre>
```



Layouts



LinearLayout

(horizontal/verical)

```
<LinearLayout</pre>
    android:orientation="vertical"
    <LinearLayout</pre>
      android: orientation="horizontal"
       <TextView
              android:id="@+id/lb1"
              android:text="@string/login"/>
      <EditText
              android:id="@+id/logintv"
              android:layout_width="match_parent" --roztiahni
              android:layout_height="wrap_content"-na výšku fontu
              android:inputType="textEmailAddress" /> -- filter
    </LinearLayout>
      podobne pre password
```

Kód na slajde je zjednodušený, originál nájdete v Layouts2.zip

Login: tu ma byt nejaky hint

Password: tu ma byt nejaky hint

Login

Forget

LinearLayout

(weight, gravity, spacer)

```
<LinearLayout ... Pokračovanie</pre>
       <LinearLayout
      android:orientation="horizontal"
       <Button
              android:id="@+id/logBtn"
             android:layout_weight="20"
              android:text="@string/Login"/>
       <Space
              android:layout_weight="20" />
       <Button
              android:id="@+id/forgetPass"
              android:layout_weight="20"
              android:text="@string/forget" />
       </LinearLayout>
```

</LinearLayout>

Kód na slajde je zjednodušený, originál nájdete v Layouts2.zip

RelativeLayout

(legacy)

<RelativeLayout

```
<Button
       android:id="@+id/button1"
       android:layout_alignParentLeft="true"
       android:layout_alignParentTop="true"/>
   <But.ton
       android:id="@+id/button2"
       android:layout_below="@+id/button1"
       android:layout_toRightOf="@+id/button1"/>
... <Button
       android:id="@+id/button4"
       android:layout_alignLeft="@+id/button1"
```

android:layout_below="@+id/button3"

android:layout_toLeftOf="@+id/button3" />

</RelativeLayout>

Kód na slajde je zjednodušený, originál nájdete v Layouts2.zip

xyz123

Button

Button

Button

Button

below: button3

toLeftOf: button: alignLeft: button



android:layout_below="@+id/pass"

android:layout_centerHorizontal="true"/>

<Button

android:id="@+id/loginBtn"

android:layout_below="@+id/passwdtv"

android:text="@string/Login" />

<Button

android:id="@+id/forgetBtn"

android:layout_alignBottom="@+id/loginBtn"

android:layout_alignTop="@+id/loginBtn"

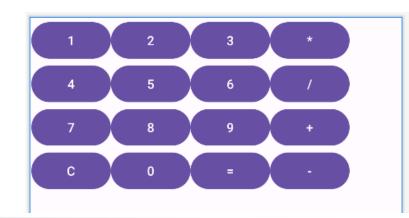
android:layout_toLeftOf="@+id/passwdtv"

android:text="@string/forget" />

originál nájdete v Layouts2.zip

GridLayout

(legacy)



< GridLayout

```
android:layout_width="wrap_content"
android:layout_height="match_parent"
android: columnCount="4"
android:rowCount="4">
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="1"
    android:id="@+id/button1"
    android:layout_row="0"
    android:layout_column="0" />
<Button ...
      android:layout_row="0"
    android:layout_column="1" />
```

Table vs. GridLayout

```
<TableLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <Button
        android:id="@+id/button7"
        android:text="Button" />

        <TableRow
        . . .</pre>
```

```
BUTTON
   BUTTON
                      BUTTON
                       123
          CLEAR
0
         2
         3
```

```
<GridLayout
    android:id="@+id/grid"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:columnCount="4"
    android:rowCount="6">
    <TextView
        android:layout_columnSpan="4"
        android:layout_gravity="right"/>
```

Dynamický vs. statický layout

Layout môžete vyklikať (niekedy náročné) alebo naprogramovať

<GridLayout</pre>

</GridLayout>

```
android:id="@+id/bigGrid"
android:columnCount="3"
android:rowCount="3">
```

Dynamický vs. statický layout

```
val smallGrid = GridLayout(this)
smallGrid.columnCount = SIZE
smallGrid.rowCount = SIZE
val smallGridParams = ViewGroup.MarginLayoutParams(
ViewGroup.LayoutParams.WRAP_CONTENT, ViewGroup.LayoutParams.WRAP_CONTENT)
smallGrid.layoutParams = smallGridParams
for (row in 0 until smallGrid.rowCount) {
    for (col in 0 until smallGrid.columnCount) {
        val button = Button(this)
        button.id = ... + 3*3*3*rowb + 3*3*row + 3*colb + col
        button.text = "."
        val buttonSize = resources.getDimension(R.dimen.buttonSize).toInt()
        val buttonParams = ViewGroup.MarginLayoutParams(buttonSize,buttonSize)
        button.layoutParams = buttonParams
        button.setOnClickListener { onClick(it) }
        smallGrid.addView(button)
                                     fun onClick(v: View) {
                                         Log.d("SUDOKU", "clicked on ${v.id}")
bigGrid.addView(smallGrid)
                                                           originál nájdete v Sudoku.zip
```

Dynamický vs. statický layout

```
val SIZE = 3
val bigGrid = GridLayout(this)
bigGrid.columnCount = SIZE
bigGrid.rowCount = SIZE
bigGrid.layoutParams = ViewGroup.LayoutParams(
    ViewGroup.LayoutParams.WRAP_CONTENT,
    ViewGroup.LayoutParams.WRAP_CONTENT)
for (rowb in 0 until bigGrid.rowCount) {
    for (colb in 0 until bigGrid.columnCount) {
        val smallGrid = GridLayout(this)
         ... celý kód z predošlého slajdu
        bigGrid.addView(smallGrid)
                                          <LinearLayout</pre>
                                              android:id="@+id/ll"
                                              android:orientation="horizontal"
11.addView(bigGrid)
                                              android:layout width="wrap content"
                                              android:layout height="wrap content"
                                          />
```

Puzzle 8

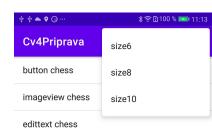
```
    1
    2
    3
    4
    8
```

SHUFFLE

SOLVE

```
fun shuffleBtn(view: View) {
   bindings.shuffleBtn.isEnabled = false
   bindings.solveBtn.isEnabled = false

   object : CountDownTimer(10000, 100) {
      override fun onTick(p0: Long) {
        swap one pair
      }
      override fun onFinish() {
        bindings.shuffleBtn.isEnabled = true
        bindings.solveBtn.isEnabled = true
    }
}.start()
}
```



 \triangleleft

0

Cvicenie4

ListView

```
preferences = getSharedPreferences("meno", Context.MODE_PRIVATE)
SIZE = preferences.getInt("SIZE", 8)
binding.list1.adapter = ArrayAdapter(
    this,
    android.R.layout.simple_list_item_1,
    arrayOf("button chess", "imageview chess", "edittext chess")
binding.list1.setOnItemClickListener {adv, v, index, l ->
  intent.putExtra("kind", index)
  intent.putExtra("SIZE", SIZE)
  startActivity(
       Intent(
          this, com.example.cvicenie4.ChessActivity::class.java))
```



Cvicenie4

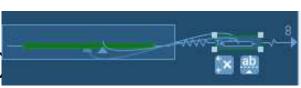
00 01 02 03 04 05 10 11 12 13 14 15 20 21 22 23 24 25 30 31 32 33 34 35 40 41 42 43 44 45 50 51 52 53 54 55

menu

```
override fun onCreateOptionsMenu(menu: Menu): Boolean {
    menuInflater.inflate(R.menu.menu, menu)
    return true
}
override fun onOptionsItemSelected(item:MenuItem):Boolean{
    when (item.itemId) {
        R.id.size6 -> SIZE = 6
        R.id.size8 -> SIZE = 8
        R.id.size10 -> SIZE = 10
        else -> super.onOptionsItemSelected(item)
    }
    return true
}
```

Constraint Layout

Je zovšeobecnením RelativeLayout umožňuje nastaviť väzby, či obmedzenia (constraints)



- relatívnu pozíciu
- spoločná baseline pre text
- okraje
- wrap/match content/fixná veľkosť
- vychýlenie (bias)



Left Margin

8

8

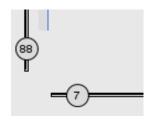
8

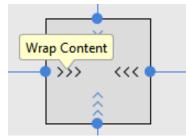
8

8

8

https://developer.android.com/reference/androidx/constraintlayout/widget/ConstraintLayout https://www.youtube.com/watch?v=z53Ed0ddxgM







Niektoré možnosti

```
B
A
```

В

right

```
<Button android:id="@+id/buttonA" ... />
                                                  Α
 <Button android:id="@+id/buttonB" ...
       app:layout_constraintLeft_toRightOf="@+id/buttonA" />
```

left

layout_constraintBaseline_toBaselineOf

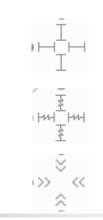
start end top Α baseline bottom

android:layout_marginLeft

B margin

layout_constraintVertical_bias

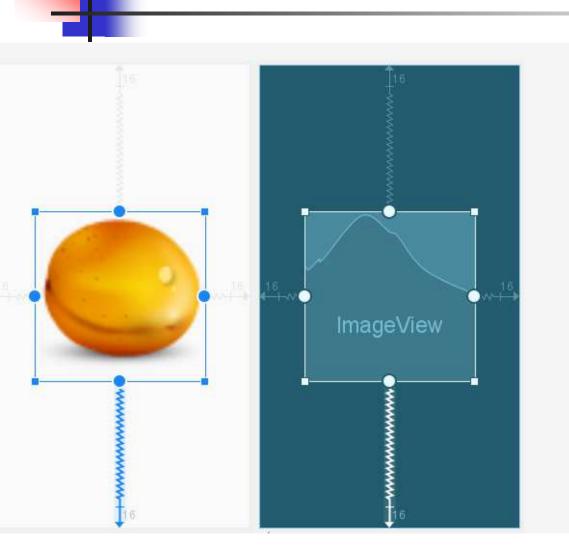
je ich d'aleko viac, ale zaujímavejšie je to v designeri



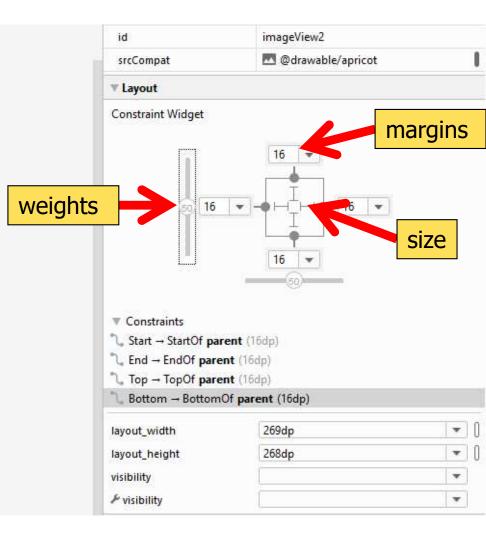
fixed size in dp ⊗

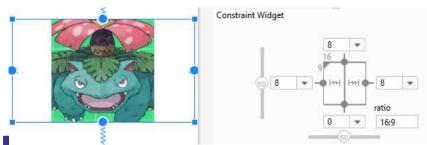
match parent

wrap content



Constraint Layout





Constraint Layout

- Convert View
- BluePrint vs. Design móde
- Show constraints zobrazí constraints v BluePrint resp. Design móde
- remove constraint (ctrl-)
- Horizontálne vertikálne constraints nemožno miešať
- Komponent musí mať aspoň jeden horizontálny a verikálny constraint
- Clear All Constraints zmaže všetky
- Okraje layout_marginStart
- Infer Constraints –
- wrap_content/match_constraint=0dp (deprecated match_parent)
- layout_constraintWidth_min, layout_constraintWidth_max
- base line
- ImageView ratio layout_constraintDimensionRatio
- rotation[X,Y,Z], scale[X,Y,Z], translation[X,Y,Z]



Constraint Layout

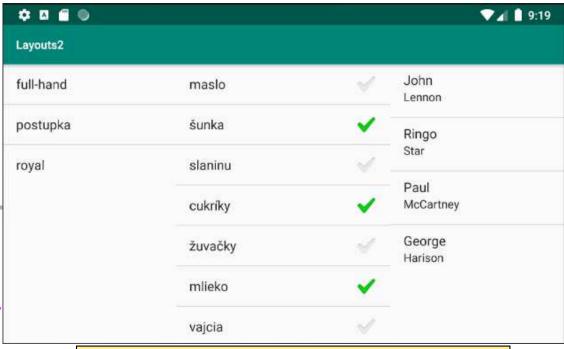
- zreťazenie chain
- zarovnanie alignment
- centrovanie
- guidelines
- Barriers
- Groups

(variabilita)

ListView a ListActivity zobrazujú zoznam položiek a môžu mať

- preddefinovaný štýl
 - môžu/nemusia sa nám páčiť
 - ale sú ready to use
- user defined
 - narobíme sa pri ich definícii

com.example.layouts2 D/ZOZNAM: item click: 0:full-hand



```
Rôzne inštancie ListView simple_list_item_1, simple_list_item_activated_1 simple_list_item_checked simple_list_item_2
```

Odchytávanie udalostí v ListView

```
com.example.layouts2 D/ZOZNAM: beatles click: 2:{krstne=Paul, priezv=McCartney} com.example.layouts2 D/ZOZNAM: beatles click: 1:{krstne=Ringo, priezv=Star} com.example.layouts2 D/ZOZNAM: beatles click: 3:{krstne=George, priezv=Harison} com.example.layouts2 D/ZOZNAM: check click: 3:cukríky com.example.layouts2 D/ZOZNAM: check click: 4:žuvačky com.example.layouts2 D/ZOZNAM: item click: 1:postupka com.example.layouts2 D/ZOZNAM: item click: 2:royal
```

com.example.layouts2 D/ZOZNAM: check click: 2:slaninu Project: Layouts2.zip



(simple_list_item_1)

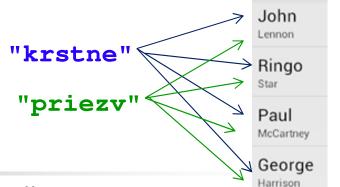
```
// poker - simple_list_item1 view
listView1.adapter = ArrayAdapter<String>(
   this,
   android.R.layout.simple_list_item_1, // jednoriadkový
      // simple list item activated 1
    resources.getStringArray(R.array.poker) // hodnoty
// listView1.choiceMode = ListView.CHOICE MODE MULTIPLE
listView1.setOnItemClickListener {
   adapterView, view, index, 1 -> // View.OnItemClickListener
      val hodnota = adapterView.getItemAtPosition(index)
      Log.d(TAG, "item click: $index:$hodnota")
```



(simple_list_item_checked)

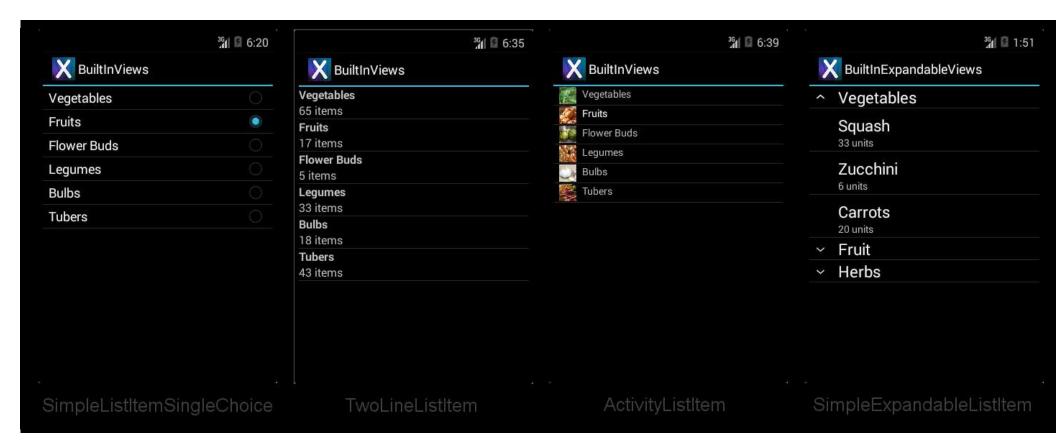
```
// nákup - checked box list view
listView2.adapter = ArrayAdapter<String>(
    this,
    android.R.layout.simple_list_item_checked, //ch-box
    resources.getStringArray(R.array.nakup)
listView2.setOnItemClickListener {
    adapterView, view, index, l ->
      val hodnota = adapterView.getItemAtPosition(index)
       (view as CheckedTextView).toggle() // prekresli
      Log.d(TAG, "check click: $index:$hodnota")
```

(simple_list_item_2)



Naplniť iný, napr. dvojriadkový ListView je náročnejšie //beatles list view val pairs = listOf(// hodnoty sú zoznam máp kľúč->hodnota mapOf("krstne" to "John", "priezv" to "Lennon"), mapOf("krstne" to "Ringo", "priezv" to "Star"), mapOf("krstne" to "Paul", "priezv" to "McCartney"), mapOf("krstne" to "George", "priezv" to "Harison") listView3.adapter = SimpleAdapter(this, // hodnoty pairs, android.R.layout.simple_list_item_2, // format ListView arrayOf(android.R.id.text1, android.R.id.text2) // riadky .toIntArray() listView3.setOnItemClickListener { adapterView, view, index, l -> val hodnota = adapterView.getItemAtPosition(index) Log.d(TAG, "beatles click: \$index:\$hodnota:"+ "\${ (hodnota as Map<String, String>) ["krstne"]

Rôzne preddefinované ListView (prehľad)



Domáca úloha 2

(jedna z 3 alternatív, na budúcu prednášku bude Menus)

Debilnicek	
zavolat svokre, ze ju obdivujem	21/11/12
vencit Harryho	21/11/12
(plvo	21/11/12
syr	21/11/12
mlieko	21/11/12

Vytvorte (malú) aplikáciu zvanú Debilníček, resp. Nákupný košík:

- umožní poznamenať si, veci, predmety, činnosti do tzv. ToDo listu,
- dovolí nastaviť deadline na splnenie činnosti pomocou dátumu/času,
- ak to bude verzia nákupný košík, tak aj počet predmetov,
- umožní ich vymazať, resp. označiť za vybavené/nakúpené, resp. vymazať všetky vybavené,
- kontroluje deadline, a upozorní správou, zvukom na prešvihnutý deadline,
- pri vypnutí aplikácie si zoznam zapamätá, pri otvorení sa zoznam obnoví

