



AS Projekt

(anatómia projektu)



Peter Borovanský
KAI, I-18

MS-Teams: [2sf3ph4](#), [List](#), [github](#)

borovan 'at' ii.fmph.uniba.sk



Čo dostaneme zadarmo

(pokračujeme v minulej prednáške)

```
package com.fmph.kai.prednaska2020
```

```
import android.os.Bundle
```

```
import androidx.appcompat.app.AppCompatActivity
```

```
class MainActivity : AppCompatActivity() { // entry point pre App/Activity
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
```

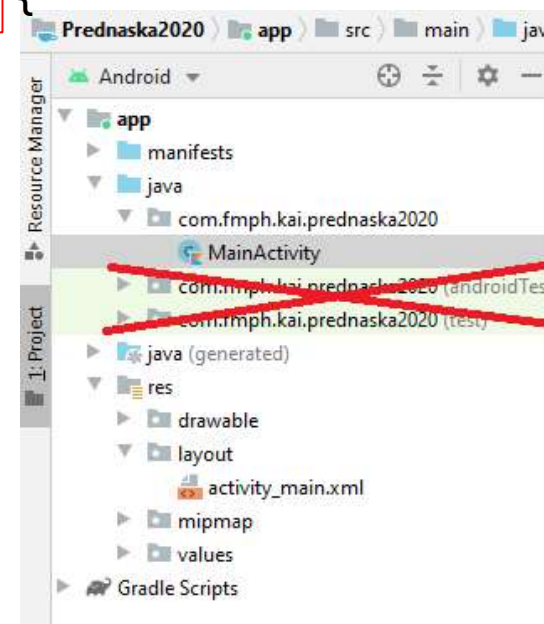
```
        super.onCreate(savedInstanceState)
```

```
        setContentView(R.layout.activity_main)
```

```
        // sem sme minule písali náš prvý kotlin kód
```

```
    }  
}
```

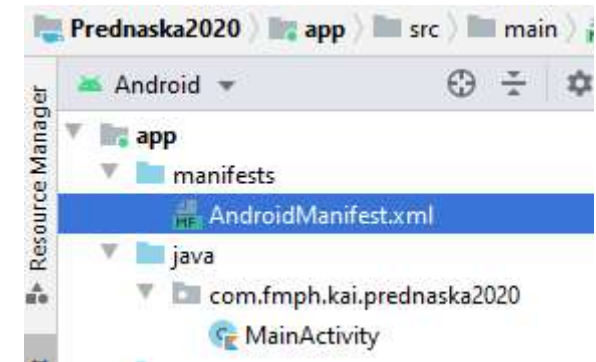
- MainActivity je inštancia triedy AppCompatActivity
- metóda onCreate() sa volá *niekde* v procese jej zobrazovania
- setContentView zobrazí layout podľa xml popisu v
R.layout.activity_main
- argument savedInstanceState:Bundle? zatiaľ neriešte
- package androidTest a test môžete vymazať, pre prehľadnosť



Project: [MyFirstApp2.zip](#)

AndroidManifest.xml

(automaticky vygenerovaný súbor aplikácie)



```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.fmph.kai.prednaska2020">
```

```
<application
```

```
    android:allowBackup="true"
```

```
    android:icon="@mipmap/ic_launcher"
```

```
    android:label="@string/app_name"
```

```
    android:roundIcon="@mipmap/ic_launcher_round"
```

```
    android:supportRtl="true"
```

```
    android:theme="@style/AppTheme">
```

```
    <activity android:name=".MainActivity">
```

```
        <intent-filter>
```

```
            <action android:name="android.intent.action.MAIN" />
```

```
            <category android:name="android.intent.category.LAUNCHER" />
```

```
        </intent-filter>
```

```
    </activity>
```

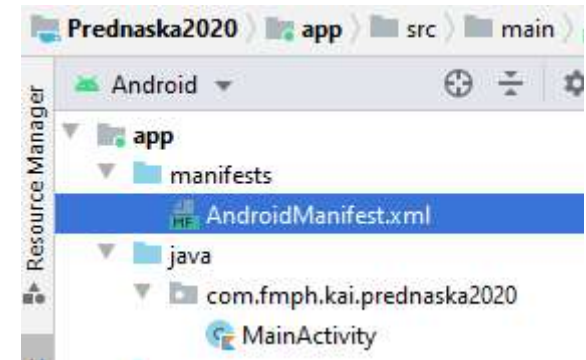
```
</application>
```

```
</manifest>
```

referencia na ikonu apky
referencia meno apky



AndroidManifest.xml



Hlavné tagy:

- `<application>` je jediný a popisuje ikony, logo, meno, štýl aplikácie
- `<activity>` môže ich byť viac a popisujú package definujúci aktivitu (Screen v MITI), intent aktivity, filtre pre aktivitu, ...
- `<service>` popisujú aplikácie bežiacie na pozadí, tzv. servisy
- `<provider>` popisuje Content Provider, napr. lokálnu databázu LiteSQL
- `<receiver>` popisuje Broadcast Receiver prijímajúci nejaké intenty

AS-manifest ochudobnel, mnohé veci sa presunuli do build.gradle:

- `<uses-configuration>` a `<uses-feature>`
popisujú HW predpoklady na spustenie apky, display, klávesnicu, senzory
- `<uses-supportScreens>` popisuje rozlíšenie HVGA, QVGA, WVGA, WQVGA
- `<uses-sdk>` popisuje min./max. SDK a cieľovú verziu SDK
<http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels>
- `<uses-permissions>` popisuje práva, ktoré apka musí mať schválené
- `<uses-library>` popisuje externé knižnice, napr. Google Maps, ...
[viac na: http://developer.android.com/guide/topics/manifest/manifest-intro.html](http://developer.android.com/guide/topics/manifest/manifest-intro.html)



build.gradle

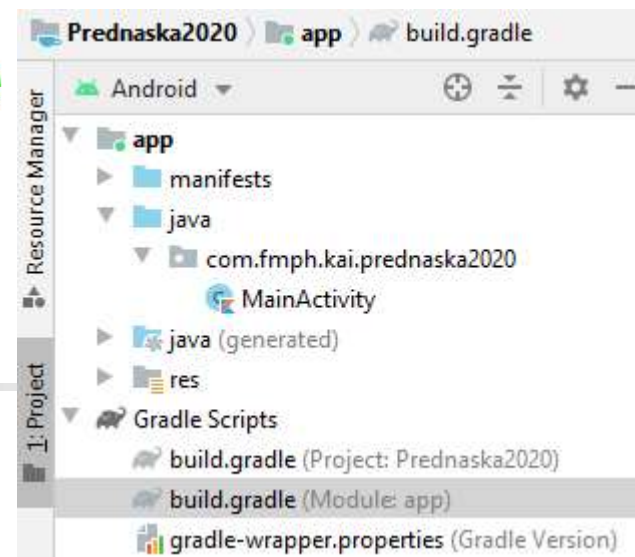
(konfiguračný súbor pre gradle)

Gradle je build tool, podobne ako make, maven

```
apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
apply plugin: 'kotlin-android-extensions'
```

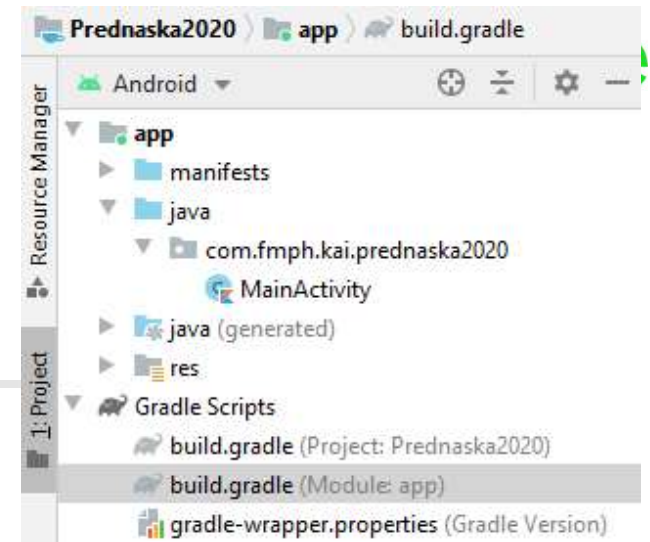
```
android {
    compileSdkVersion 29
    defaultConfig {
        applicationId "com.fmph.kai.prednaska2020"
        minSdkVersion 19
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    ...
}
```

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'androidx.appcompat:appcompat:1.0.2'
    ...
}
```





Gradle



- je plugin-based project-build/management system v AS založený na jazyku Groovy
- už existuje Kotlin Gradle Plugin pre Gradle 6+

```
build.gradle.kts
dependencies {
    implementation("fileTree(dir: 'libs', include: ['*.jar'])")
    implementation("org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version")
    implementation("androidx.appcompat:appcompat:1.0.2")
    ...
}
```

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'androidx.appcompat:appcompat:1.0.2'
    ...
}
```


MergedManifest

(spája AndroidManifest a build.gradle)

```
<manifest
  android:versionCode="1"
  android:versionName="1.0"
  package="com.fmph.kai.prednaska2020"
  xmlns:android="http://schemas.android.com/apk/res/android" >
  <uses-sdk
    android:minSdkVersion="19"
    android:targetSdkVersion="29" />
  <application
    android:allowBackup="true"
    android:appComponentFactory="androidx.core.app.CoreComponentFactory"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportRtl="true"
    android:theme="@style/AppTheme" >
    <activity
      android:name="com.fmph.kai.prednaska2020.MainActivity" >
      <intent-filter>
        <action
          android:name="android.intent.action.MAIN" />
        <category
          android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
</manifest>
```

Manifest Sources

- ☐ [core:1.3.1](#) manifest
- ☒ [app_main](#) manifest (this file)
- ☐ [build.gradle](#) injection

Other Manifest Files

(Included in merge, but did not contribute a change)

[appcompat:1.2.0](#) manifest, [viewpager:1.0.0](#) manifest, [interpolator:1.0.0](#) manifest, [savedstate:1.0.0](#) manifest, [vectordrawable:1.1.0](#) manifest, [lifecycle-viewmodel:1.0.0](#) manifest, [appcompat-resources:1.2.0](#) manifest, [lifecycle-viewmodel-ktx:1.0.0](#) manifest, [vectordrawable-animated:1.1.0](#) manifest, [fragment:1.0.0](#) manifest, [constraintlayout:2.0.1](#) manifest, [constraintlayout-solver:1.0.0](#) manifest, [versionedparcelable:1.1.0](#) manifest, [core-ktx:1.0.0](#) manifest, [loader:1.0.0](#) manifest, [activity:1.0.0](#) manifest

Merging Log

Added from the [app_main](#) manifest (this file)

referencia meno apky

```
<resources>
  <string name="app_name">MyFirstApp</string>
</resources>
```

Resources/Values

- drawables - obrázky v rôznych rozlíšeníach (ldpi, mdpi, hdpi, xhdpi, xxhdpi)
- layouts – rozloženia komponentov na aktivitách (bude dnes, na budúce)
- menus – pre aktivity (bude neskôr)
- values – pomenované konštanty (strings.xml, colors.xml, styles.xml ...)

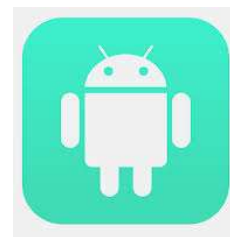
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <color name="colorPrimary">#3F51B5</color>
  <color name="colorPrimaryDark">#303F9F</color>
  <color name="colorAccent">#FF4081</color>
```

```
<resources>
  <!-- Base application theme. -->
  <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
  </style>
</resources>
```


Bud' kreatívny

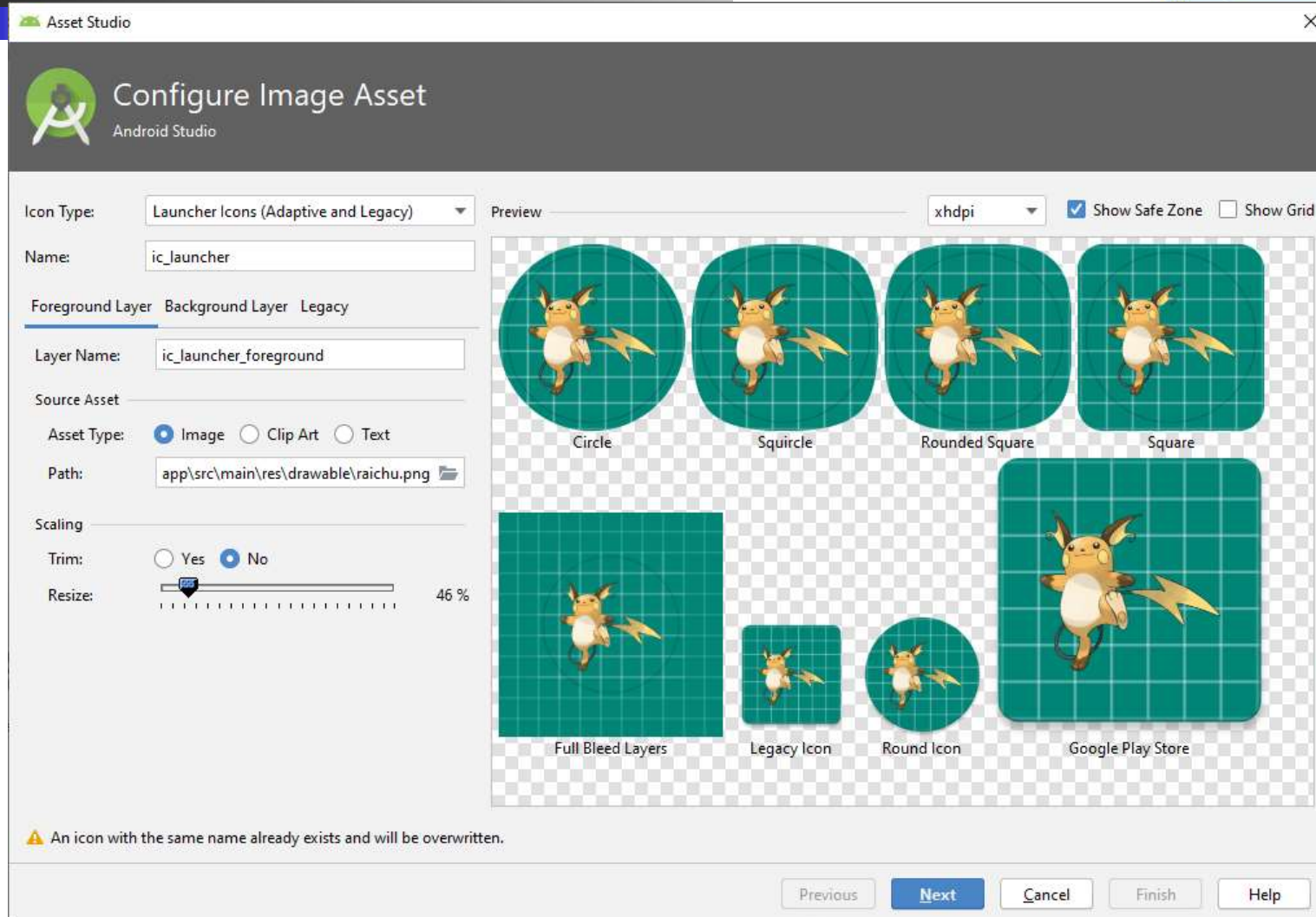
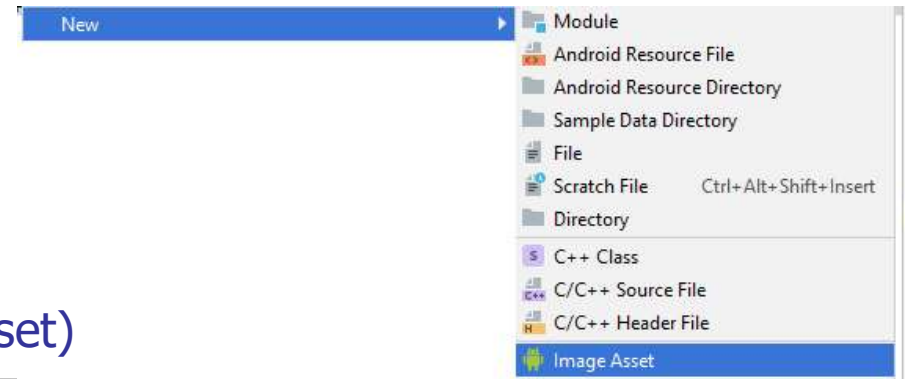
(aspoň pri ic_launcher ikone)

Je hrozné pri opravovaní mať v tablete/mobile viacero študentských riešení s generickými/neosobnými ikonami. Preto ak sa dá, tak sa zosobnite v posielanom riešení už v ikone vašej aplikácie.



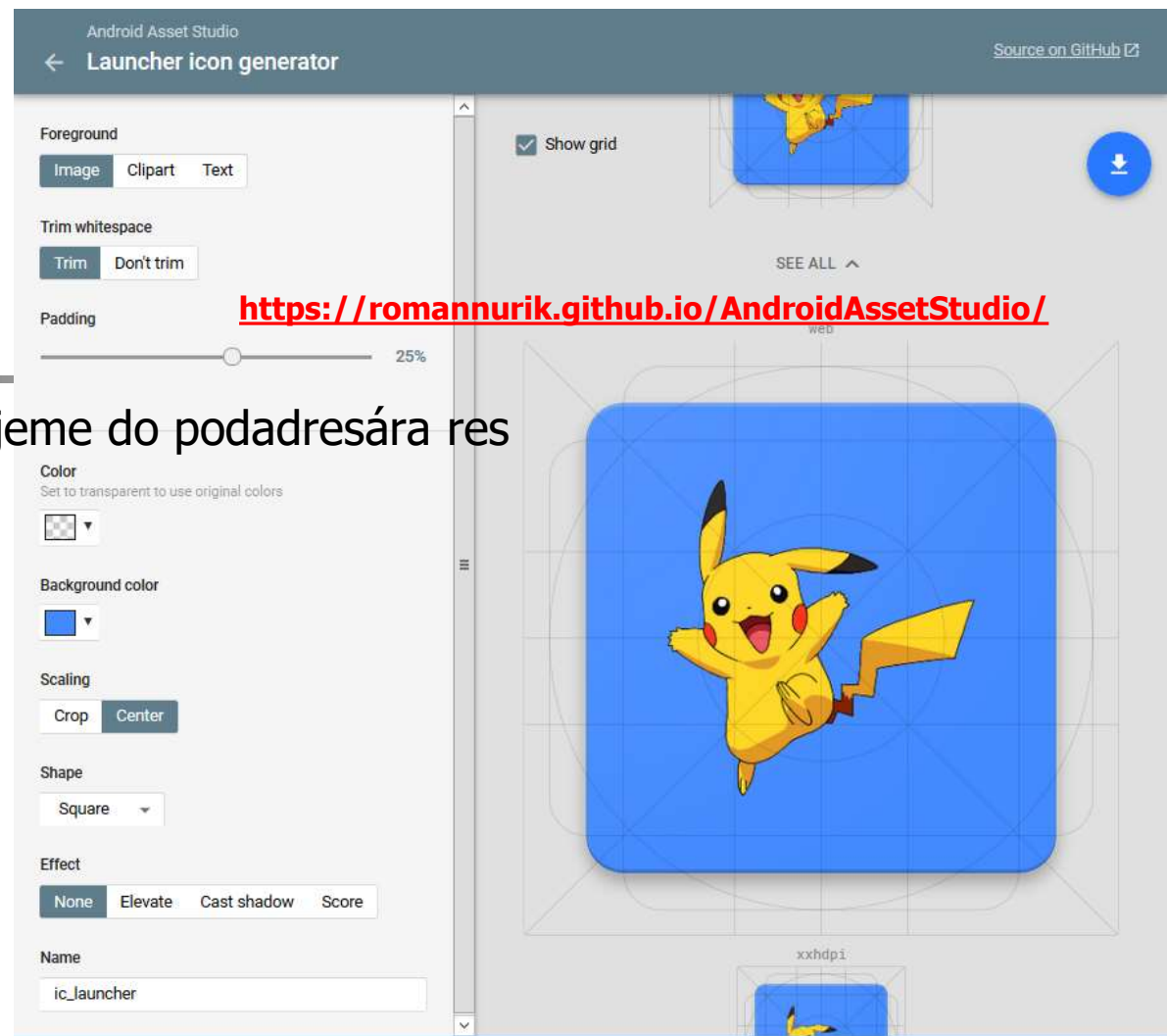
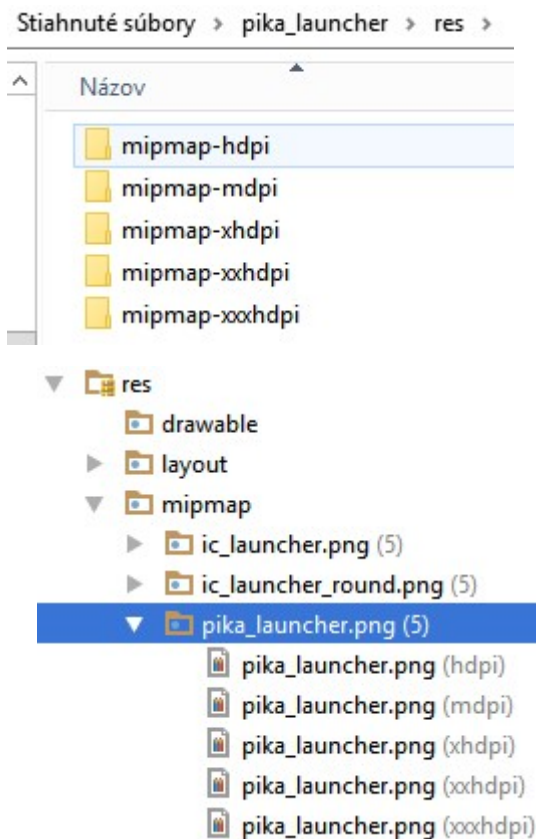
Bud' kreatívny

(a použi Asset Studio - New/ImageAsset)



Android Asset Studio Icon generator

výsledok priamo nakopírujeme do podadresára res
Ikony/obrázky sa
sa objavajú v projekte



<https://romannurik.github.io/AndroidAssetStudio/>

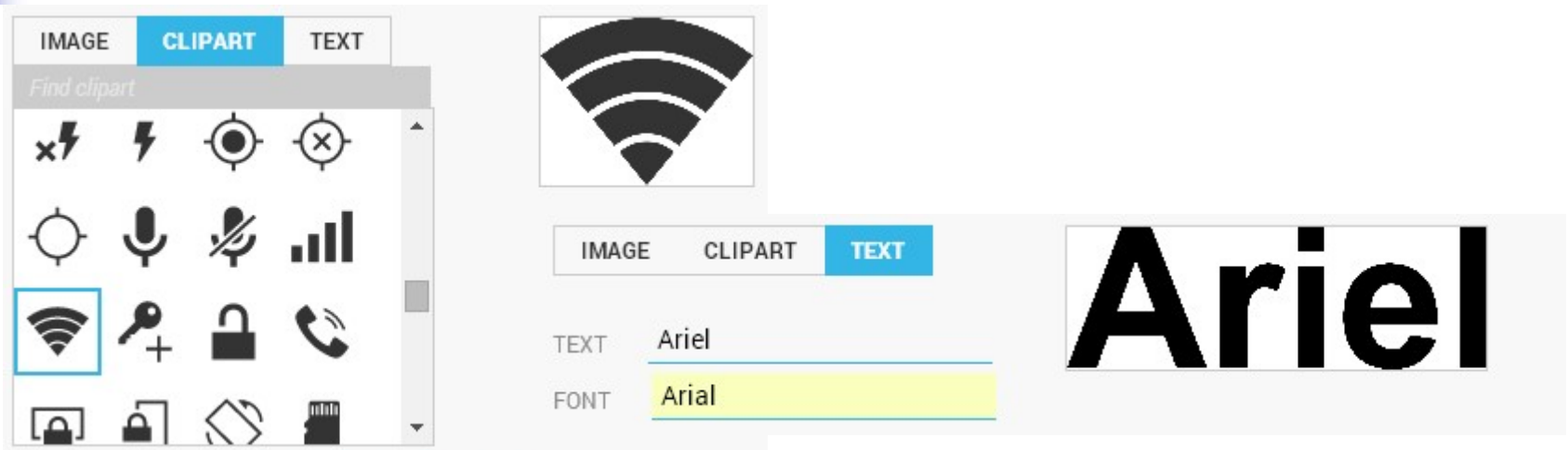
```
<application
    android:allowBackup="true"
    android:icon="@mipmap/pika_"
    android:label="@mipmap/pika_launcher"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportRtl="true"
```

Project:Pikas.zip, Pikas2.zip

Android Asset Studio

(jedna z alternatív)

<https://romannurik.github.io/AndroidAssetStudio/>

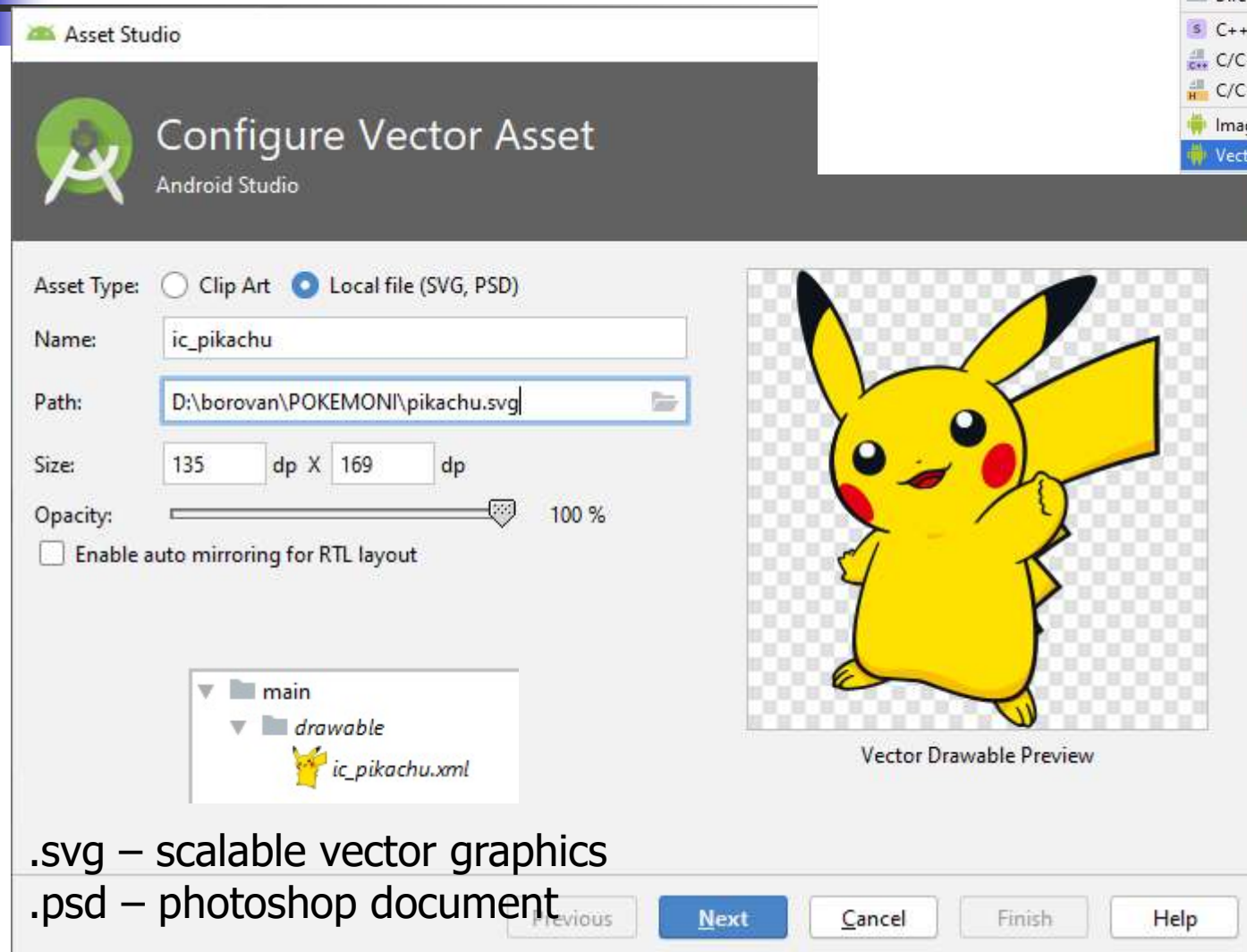
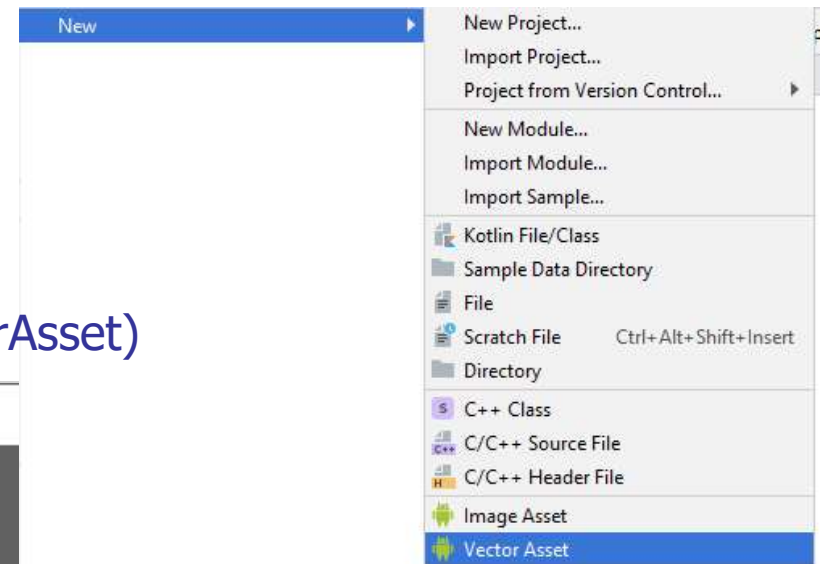


- .png, .jpg, .bmp, ...
- cliparty
- texty



Pre .svg a .psd

(a použi Vector Asset Studio - New/VectorAsset)



.svg – scalable vector graphics

.psd – photoshop document

Resources/Drawables/Mipmap

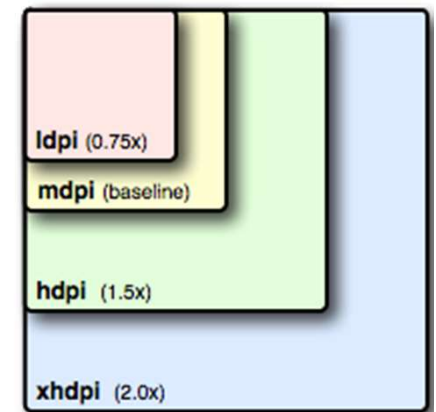
(ikona - viacero rozlíšení)

http://developer.android.com/guide/practices/screens_support.html



pomer l/m/h/xh/x²h/x³h-dpi 3:4:6:8:12:16 - geom.postupnosť s koef. Sqrt(2)

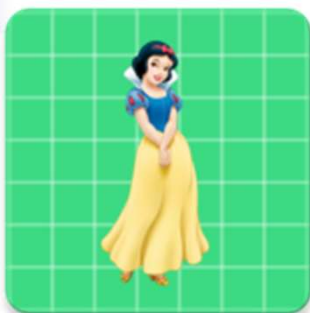
- 36x36 for low-density (LDPI = ~ 120 dpi)
- 48x48 for medium-density (MDPI = ~ 160 dpi)
- 72x72 for high-density (HDPI = ~ 240 dpi)
- 96x96 for extra high-density (XHDPI = ~ 320 dpi)
- 144x144 for extra² high-density (XXHDPI = ~ 480 dpi)
- 192x192 for extra³ high-density (XXXHDPI = ~ 640 dpi)



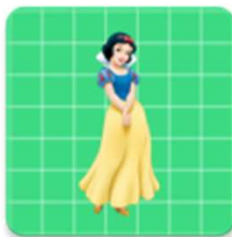
$\sqrt{2}$

Snehulienka

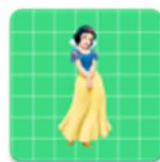
(v geometrickom rade s quocientom $\sqrt{2}$)



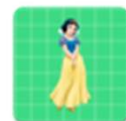
192x192 for extra³ high-density
(XXXHDPI = ~ 640 dpi)



144x144 for extra² high-density
(XXHDPI = ~ 480 dpi)



96x96 for extra high-density
(XHDPI = ~ 320 dpi)



72x72 for high-density
(HDPI = ~ 240 dpi)



48x48 for medium-density
(MDPI = ~ 160 dpi)

$\sqrt{2}$

```
imageView.setImageDrawable(  
    ContextCompat.getDrawable(  
        applicationContext,  
        R.drawable.snehulienka) )
```



Resources/Values

- string

```
<string name="app_name">YourFirstHello</string>
```

- color

```
resources.getString(R.string.app_name)
```

```
<color name="transparent_green">#7700FF00</color>
```

- dimensions

```
resources.getColor(R.color.transparent_green)
```

```
<dimen name="absolutLarge">144dp</dimen>
```

- style

```
resources.getDimension(R.dimen.absolutLarge)
```

```
<style name="myStyle">
```

```
    <item name="android:textSize">12sp</item>
```

```
    <item name="android:textColor">#FF00FF</item>
```

```
</style>
```

px = Pixels

in = Inches

mm = Millimeters

pt = Points, 1/72 of an inch

sp = Scale - Independent Pixels – používame pre veľkosť fontu

dp = Density - Independent Pixels – používame pre všetko ostatné



Resources/Values

- array-string/integer

```
<string-array name="poker">  
    <item>full-hand</item>  
    <item>postupka</item>  
    <item>royal</item>  
</string-array>
```

```
<integer-array name="coins">  
    <item>1</item>  
    <item>2</item>  
    <item>5</item>  
    <item>10</item>  
    <item>20</item>  
</integer-array>
```

```
resources.getStringArray(R.array.otazky) :Array<String>
```

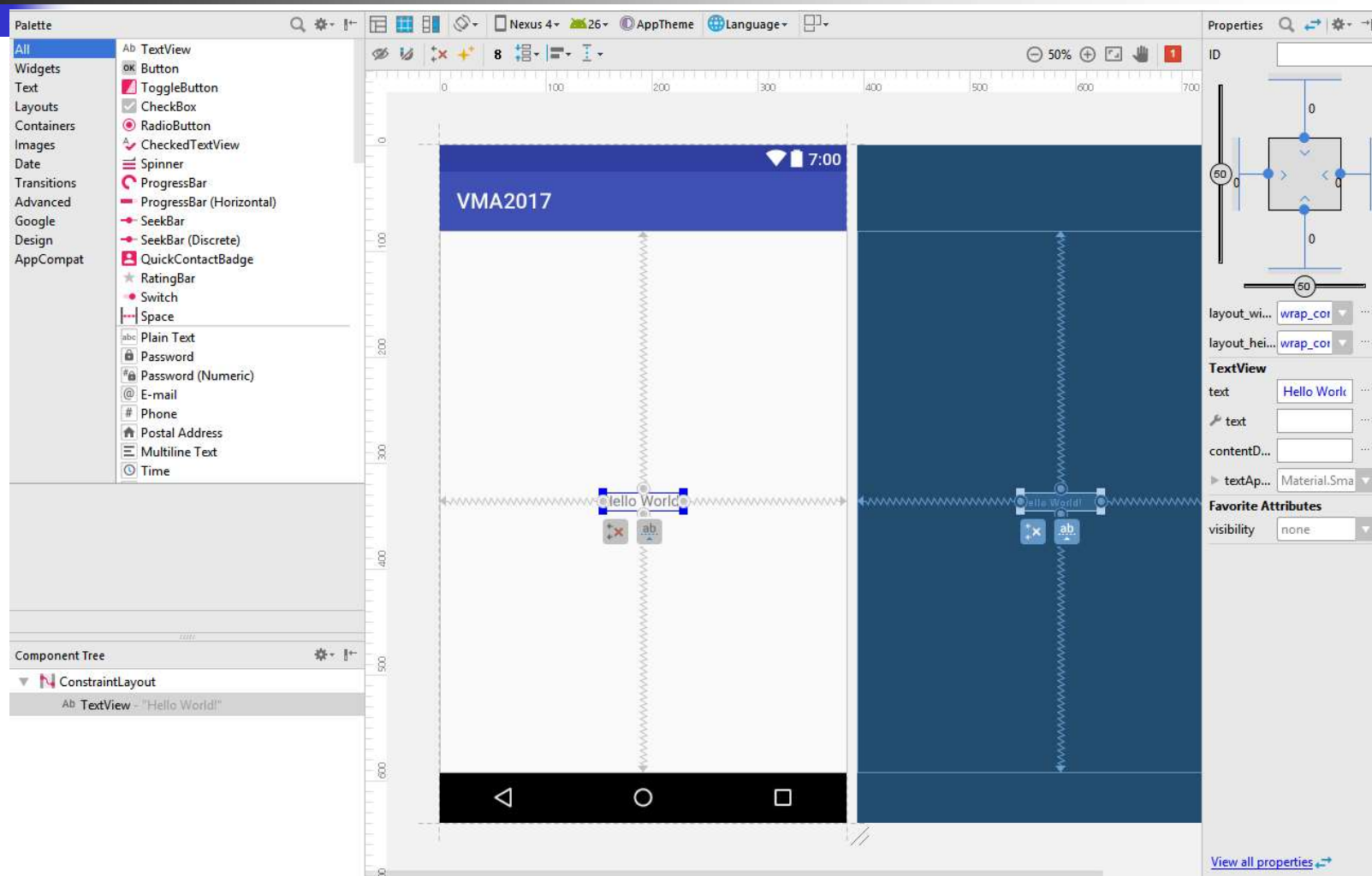
- plurals (quantity strings)

```
<plurals name="man">  
    <item quantity="one">man</item>  
    <item quantity="many">men</item>  
    <item quantity="zero">paradis</item>  
</plurals>
```

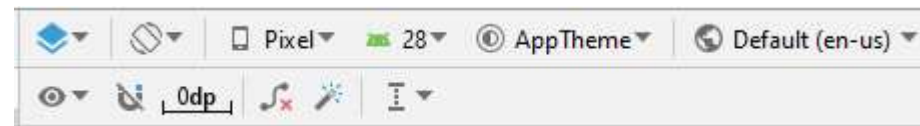
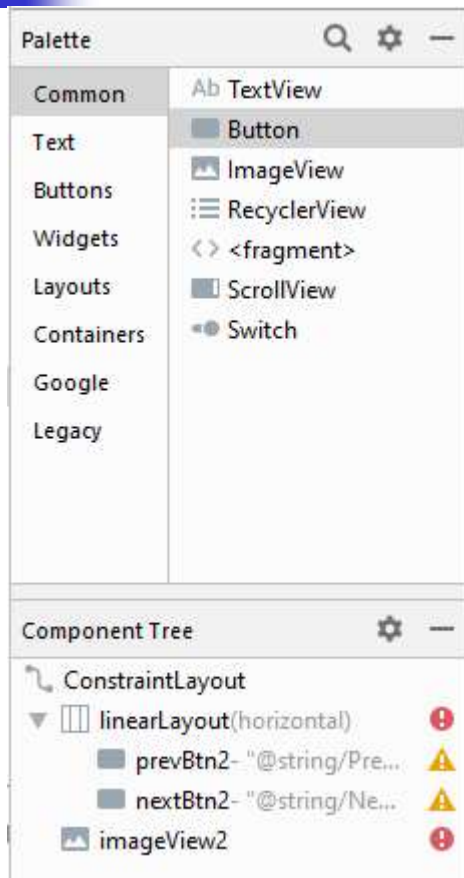
Resources/Layout

(Design View)

Konvencia:
`XYZActivity[.kt/]`
má layout
`activity_xyz.xml`



Layout Manager



Design/Blueprint/Design+Blueprint

Layout: Landscape/Portrait/...

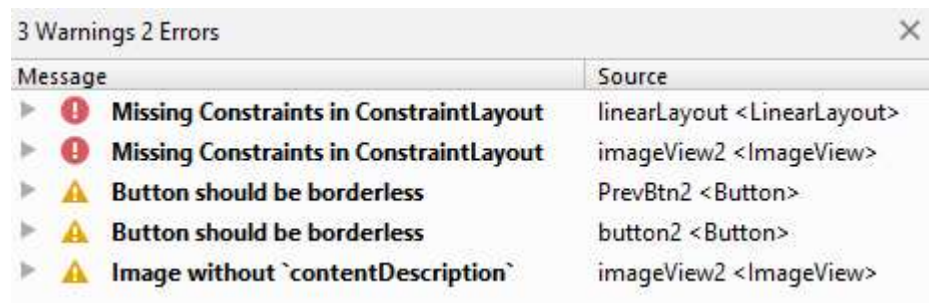
Pixel: AVD/Pixel2/Pixel#

API Level: 26/27/28/...

AppTheme :

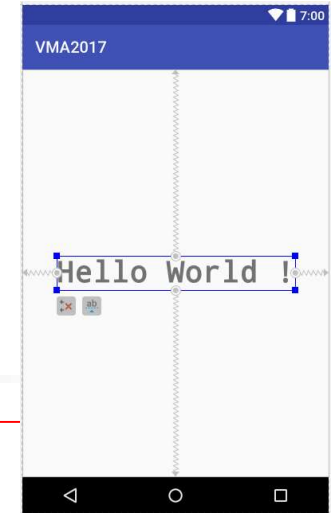
Default (en-us) : lokalizácie do rôznych jazykov

: warnings, errors



Resources/Layout

(Text View)



```
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="pokus.example.com.vma2017.MainActivity">
```

*wrap_content
fill_parent=
match_parent*

```
<TextView
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:fontFamily="monospace"
```

```
    android:text="Hello World!"
```

```
    android:textSize="36sp"
```

```
    android:textStyle="bold"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
```

```
    app:layout_constraintLeft_toLeftOf="parent"
```

```
    app:layout_constraintRight_toRightOf="parent"
```

```
    app:layout_constraintTop_toTopOf="parent" />
```

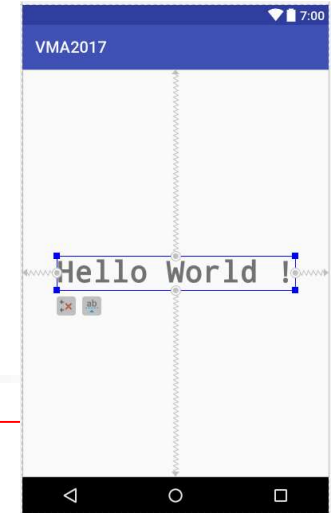
```
</android.support.constraint.ConstraintLayout>
```

Bad style

Hardcoded string "Hello World 1", should use
`@string` resource

Resources/Layout

(Text View)



```
<android.support.constraint.ConstraintLayout
```

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

```
    xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
    xmlns:tools="http://schemas.android.com/tools"
```

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

```
    android:layout_height="match_parent"
```

```
    tools:context="pokus.example.com.vma2017.MainActivity">
```

*wrap_content
fill_parent=
match_parent*

```
    <TextView
```

```
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
```

```
        android:fontFamily="monospace"
```

```
        android:text="@string/IntroString"
```

```
        android:textSize="@dimen/reallyBigFont"
```

```
        android:textStyle="bold"
```

```
        app:layout_constraintBottom_toBottomOf="parent"
```

```
        app:layout_constraintLeft_toLeftOf="parent"
```

```
        app:layout_constraintRight_toRightOf="parent"
```

```
        app:layout_constraintTop_toTopOf="parent" />
```

```
    <resources>
```

```
        <string name="app_name">VMA2017</string>
```

```
        <string name="IntroString">Hello World</string>
```

```
    </resources>
```

```
    <resources>
```

```
        <dimen name="reallyBigFont">30dp</dimen>
```

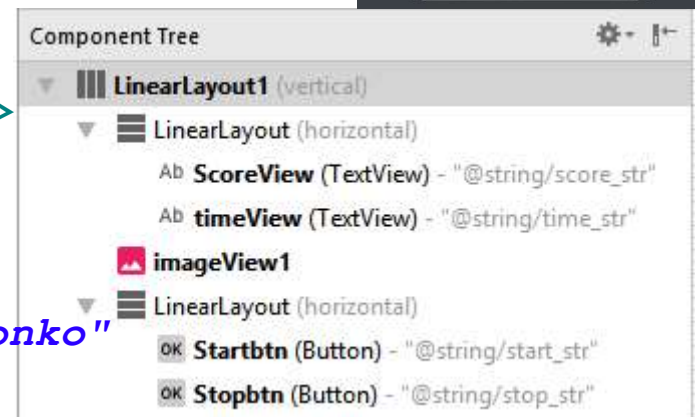
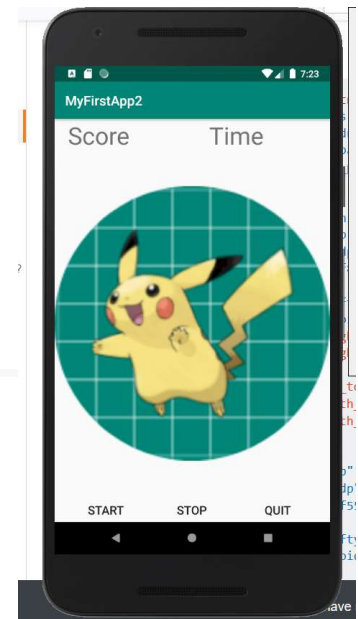
```
    </resources>
```

```
</android.support.constraint.ConstraintLayout>
```

Ako by to malo vyzerat'

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

Žiadne warnings



zjednodušené pre
účely slajdu



Logovanie

Tri najbežnejšie spôsoby:

- Log
- Toast
- Snackbar – to chce pridať závislosť do build.gradle

```
dependencies {  
    implementation 'com.android.support.design:28.0.0'  
    import com.google.android.material.snackbar.Snackbar  
  
    prevBtn2.setOnClickListener({  
        Toast.makeText(this, "prev...", Toast.LENGTH_SHORT).show()  
  
        Log.d(TAG, "prev...")  
  
        Snackbar.make(it, "prev...",  
            Snackbar.LENGTH_SHORT).setAction("Action", null).show()  
        ...  
        if (--i < 0) i += imgs.size  
        imageView2.setImageDrawable(imgs[i])  
    })  
})
```


Pikas

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

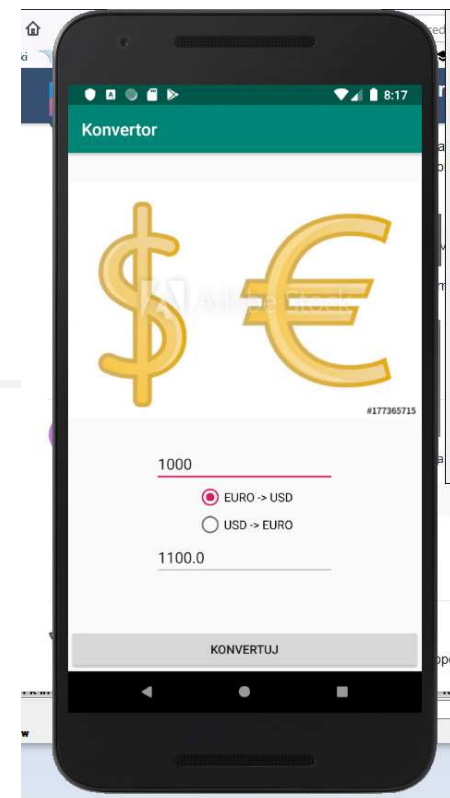


Konvertor EURO USD

(logika)

Jednoduchá aplikácia na konverziu kurzov USD EURO

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



```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    setContentView(R.layout.activity_main)  
    convertBtn.setOnClickListener({  
        Toast.makeText(this, "convert", Toast.LENGTH_SHORT).show();  
        if (inputText.text.isNotEmpty()) {  
            val input = inputText.text.toString().toFloat(); // get  
            var output = input  
            if (eur2usd.isChecked) output = 1.1F * output  
            if (usd2eur.isChecked) output = output / 1.1F  
            outputText.setText("$output") // set  
        }  
    })  
}
```

Klik na Konvertuj

Konvertor EURO USD

(setOnClickListener)

convertBtn		Button
id	convertBtn	
Declared Attributes		+ -
layout_width	match_parent	▼ 0
layout_height	wrap_content	▼ 0
id	convertBtn	
onClick	convert	▼ 0
text	@string/konvertujBtn	0

// 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()
    if (inputText.text.isNotEmpty()) {
        val input = inputText.text.toString().toFloat()
        var output = input
        if (eur2usd.isChecked) output = 1.1F * output
        if (usd2eur.isChecked) output = output / 1.1F
        outputText.setText("${output.format(2)}")
    }
}
```

metóda
Float



```
fun Float.format(digits: Int) =
    java.lang.String.format("%.${digits}f", this)
```

Project:Konvertor.zip

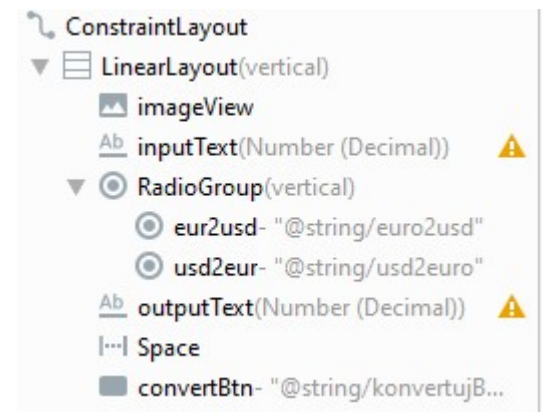
Konvertor EURO USD

(layout)

```

<LinearLayout
    <ImageView .../>
    <EditText .../>
    <RadioGroup
        <RadioButton .../>
        <RadioButton .../>
    </RadioGroup>
    <EditText .../>
    <Space .../>
    <Button .../>
</LinearLayout>

```



Príklad jednoduchéj aplikácie

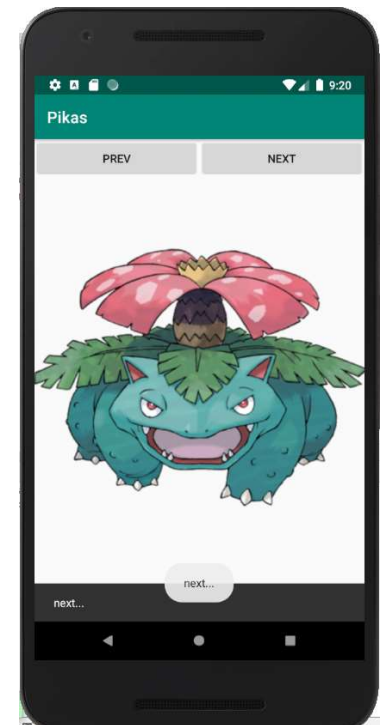
(ktorú sme si vyklikali minule)

Ilustrovali sme:

- príklad návrhu (vyklikania) jednoduchého GUI (single activity app)
- logovanie udalostí ako efektívny prostriedok ladenia pomocou
 - `Log.d(...)`
 - `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"`

Nestihli sme:

- aktivitu a jej životný cyklus



Project:Pikas2.zip



Logovanie

(rekapitulácia)

Tri najbežnejšie spôsoby:


- Log – loguje do okna Logcat, filtrujte podľa **TAGu** metódy `Log.d(TAG,`
- Toast – potrebuje **Context** (zjednodušená aktivita, v ktorej sa toastuje)
- Snackbar – to chce pridať závislosť do build.gradle a import snackbaru

```
dependencies {  
    implementation 'com.android.support.design:28.0.0'  
    import com.google.android.material.snackbar.Snackbar
```


```
prevBtn2.setOnClickListener({
```



```
    Toast.makeText(this, "prev...", Toast.LENGTH_SHORT).show()
```



```
    Log.d(TAG, "prev...")
```



```
    Snackbar.make(it, "next...",  
        Snackbar.LENGTH_SHORT).setAction("Action", null).show()  
    alebo .setAction(R.string.action,  
        View.OnClickListener { nextOnClickListener(it) }).show()
```

```
    ...
```

```
})
```

Pikas

(rekapitulácia)

activity entry point

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

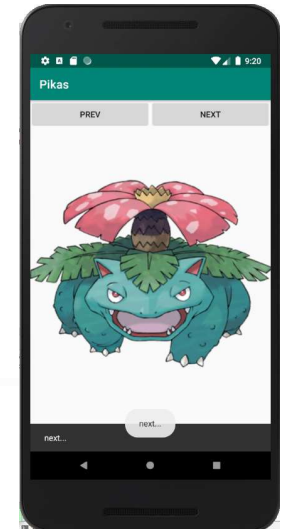


logovanie

View(s)

Pikas

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



const
final

val TAG = "PIKAS"

var i = 0

var imgs = arrayOf<Drawable?>()

State

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity_main)

imgs = arrayOf(ContextCompat.getDrawable(applicationContext,
R.drawable.butterfree), ...)

imageView2.setImageDrawable(imgs[i])

prevBtn2.setOnClickListener { // it:View -> { ... }

if (--i < 0) i += imgs.size

imageView2.setImageDrawable(imgs[i])

}

}

// prepojene cez property android:onClick="nextOnClickListener"

fun nextOnClickListener(v: View) {

i = (++i) % imgs.size

imageView2.setImageDrawable(imgs[i])

}

Common Attributes

style	@style/mystyle	
onClick	clickOnNext	

Project:Pikas2.zip

Pikas

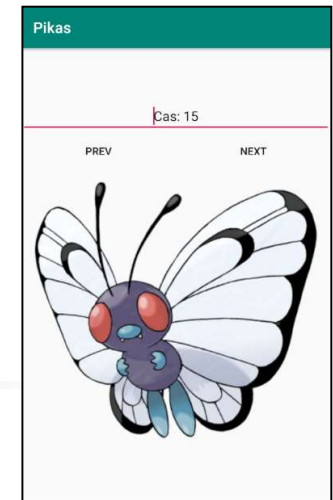
(asynchrónnosť - timer)

pomocou `java.util.Timer`

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

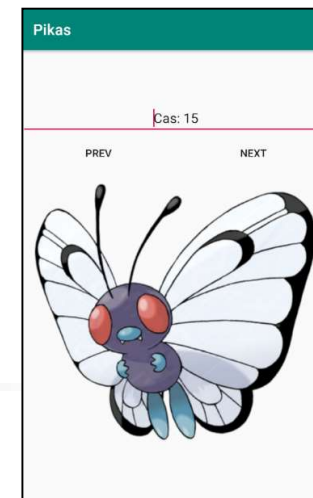
- nezabudnite na `.run()`
- `runOnUiThread`
 - má argument `java.lang.Runnable`, ktorý vykoná v hlavnom GUI vlákne

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



Pikas

(asynchrónnosť – count down)



`pomocu android.os.CountDownTimer`

```
object:CountDownTimer(20000, 1000) { // 20sek, tik po 1sek  
                        // how long, period
```

tik

```
→ override fun onTick(millisUntilFinished: Long) {  
    Log.d(TAG, "onTICK")  
    runOnUiThread {  
        time.setText("Cas: ${millisUntilFinished/1000}") }  
    }  
}
```

game
over

```
→ override fun onFinish() {  
    Log.d(TAG, "onFinish")  
    exitProcess(-1)  
}  
}.start()
```

← ukončenie appky

Životný cyklus apky

(prvý – zjednodušený nástrel)

global: 0
local: 0
shared: 0

■ Alt-Insert = Generate Override Implemented Methods:

- `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()`

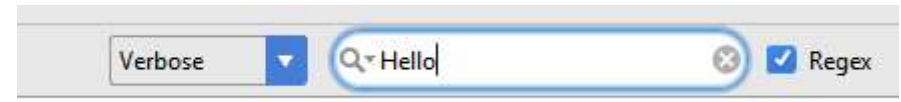
■ do každej metódy dáme kontrolný výpis, aby sme pochopili životný cyklus

```
override fun onCreate(Bundle savedInstanceState?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    Log.d("CYKLUS", "onCreate") // LOGUJTE, LOGUJTE, LOGUJTE
}
```

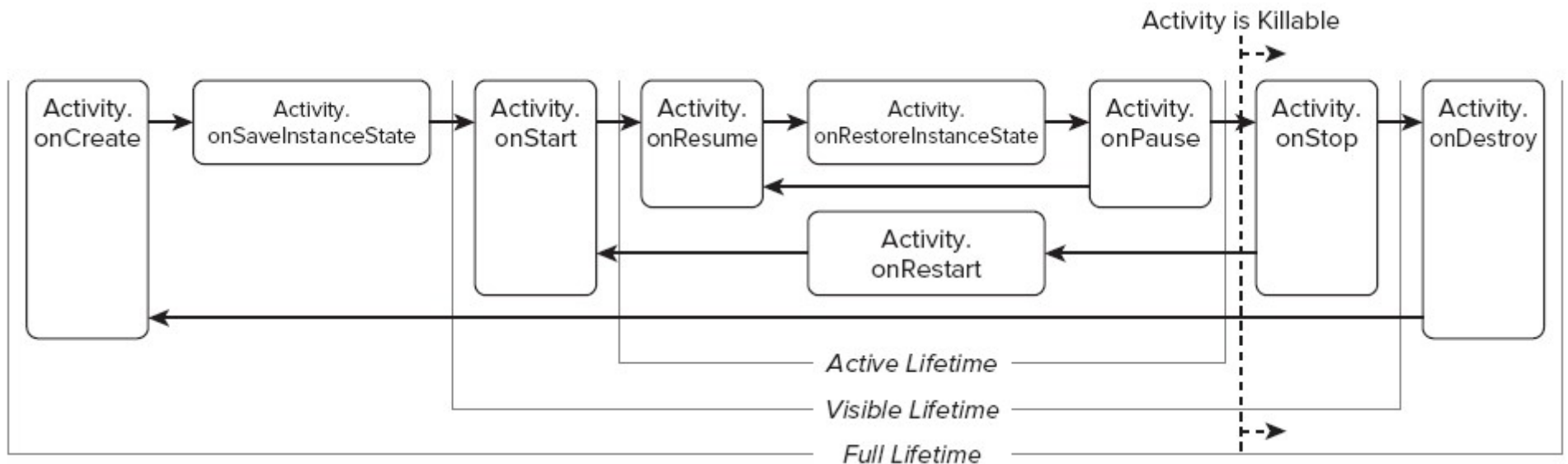
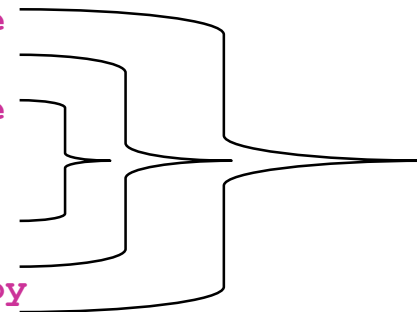
tag vhodný na filtrovanie

LogCat

(Filtrovane logov)



- 10-13 12:55:41.091: D/Hello(405): onCreate
- 10-13 12:55:41.091: D/Hello(405): onStart
- 10-13 12:55:41.100: D/Hello(405): onResume
- kill
- 10-13 12:56:45.061: D/Hello(405): onPause
- 10-13 12:56:45.681: D/Hello(405): onStop
- 10-13 12:56:45.681: D/Hello(405): onDestroy





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



Bundle?

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

```
override fun onRestoreInstanceState(  
    savedInstanceState: Bundle?) {  
    super.onRestoreInstanceState(savedInstanceState)  
    globalCounter = savedInstanceState?.getInt("COUNTER")?:0  
    ...  
}
```

```
override fun onSaveInstanceState(outState: Bundle?,  
    outPersistentState: PersistableBundle?) {  
    super.onSaveInstanceState(outState, outPersistentState)  
    outState?.putInt("COUNTER", globalCounter)  
    ...  
}
```



SharedPreferences

SharedPreferences má metody get[Int/Boolean/Char/Float/Any/...]

```
private lateinit var preferences: SharedPreferences
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    preferences = getSharedPreferences("lifecycle",
                                    Context.MODE_PRIVATE)
}
override fun onResume() {
    sharedCounter = preferences.getInt("kluc", 0)
}
override fun onPause() {
    preferences.edit {
        this.putInt("kluc", sharedCounter)
        this.commit()
    }
}
```

Pikas.java

(auto-generovaný Code/Convert Java->Kotlin)

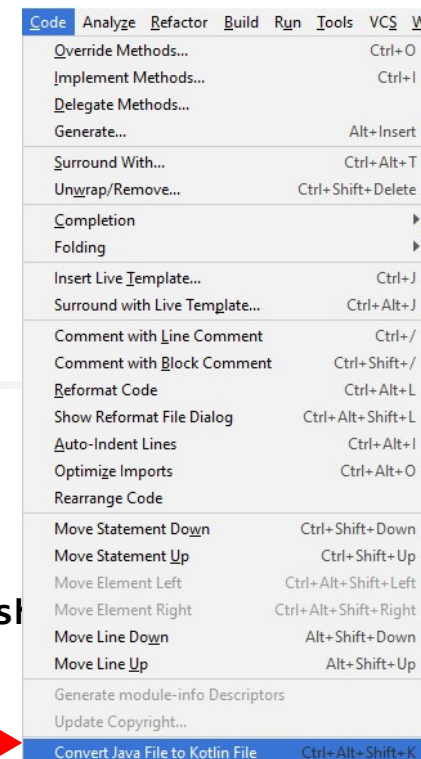
```
i = 0
iv.setImageDrawable(images[i])

quit.setOnClickListener { v ->
    Toast.makeText(this, "BYE BYE", Toast.LENGTH_LONG).show()
    this.finishAffinity()
}

prev.setOnClickListener {
    Log.d("PIKA", "onPREV")
    Toast.makeText(this@MainActivity, "PREV", Toast.LENGTH_SHORT).show()
    i--
    if (i < 0) i = images.size - 1
    iv.setImageDrawable(images[i])
}

next.setOnClickListener { v ->
    i++
    Log.d("PIKA", "onNEXT")
    Toast.makeText(this@MainActivity, "NEXT", Toast.LENGTH_SHORT).show()
    i = i % images.size
    iv.setImageDrawable(images[i])
}
```

v java
projekte
nájdete





Konverzie Java <-> Kotlin

- **Java -> Kotlin**

Code/Convert Java File to Kotlin File (neuzná sa to ako DÚ v Kotlin)

- **Kotlin -> JVM Byte code**

Tools/Kotlin/Show Byte Code

- **Decompile Byte code (to Java)**

```
protected void onCreate(@Nullable Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    this setContentView(2131296283);  
    final ObjectRef images = new ObjectRef();  
    final IntRef i = new IntRef();  
    View var10000 = this.findViewById(2131165189);  
    if (var10000 == null) {  
        throw new TypeCastException("null cannot be cast to non-null type android.widget.Button");  
    } else {
```



Čo je Kotlin ?



<https://proandroiddev.com/modern-android-development-with-kotlin-september-2017-part-1-f976483f7bd6>