Hitparáda aktivít

Aktivity, View
Intent, Layout

Peter Borovanský
KAI, I-18

borovan 'at' ii.fmph.uniba.sk

Hitparáda

Feedback / Hra15 / Apple Watch Tips

(Hall of Fame)

Príklad jednoduchej 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
  - **`findViewById<Button>(R.id.`*`quitBtn`*`)`**
  - **`prevBtn.setOnClickListener({ })`**
  - property `android:onClick="nextOnClickListener"`

Project:Pikas2.zip

Nestihli sme:

- aktivitu a jej život(ný cyklus)

Logovanie
(rekapitulácia)

Tri najbežnejšie spôsoby:

- Log – loguje do okna Logcat, filtrujte podľa `TAGu` metódy `Log.d(TAG,`
- Toast – potrebuje `Context` (zjednodušene aktivita, v ktorej sa toastuje)
- 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])
})
```

Project:Pikas2.zip

Pikas

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)

Project:Pikas2.zip

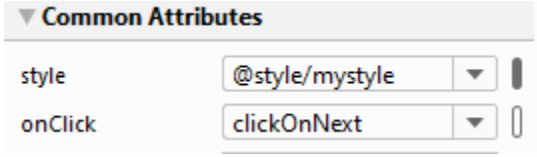
Pikas

(stav sa mieša s views a BL)

const final

State

```
val TAG = "PIKAS"
var i = 0
var imgs = arrayOf<Drawable?>()
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({
        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])
}
```



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

Project:Pikas2.zip

Pikas

(asynchrónnosť – count down)

pomocou **android.os.CountDownTimer**

```
object:CountDownTimer(20000, 1000) {
                        // how long, period
    override fun onTick(millisUntilFinished: Long) {
        Log.d(TAG, "onTICK")
        runOnUiThread {
            time.setText("Cas: ${millisUntilFinished/1000}") }
    }

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

tik

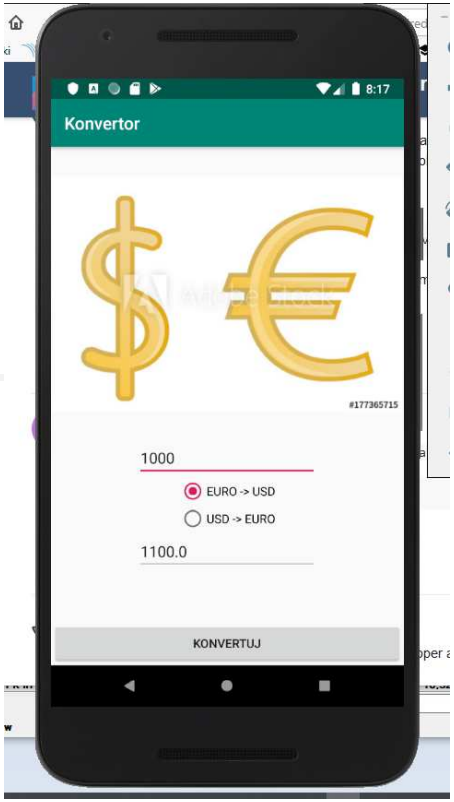
game over

ukončenie appky

Project:Pikas2.zip

Konvertor EURO USD

(logika)



Jednoduchá aplikácia na konverziu kurzov USD EURO

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

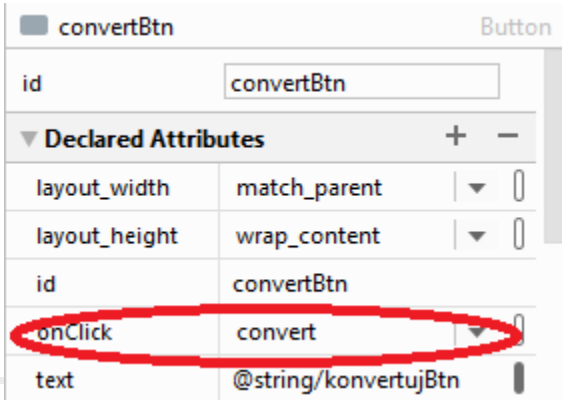
```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    convertBtn.setOnClickListener({
        Toast.makeText(this, "convert", Toast.LENGTH_SHORT).show()
        if (inputText.text.isNotEmpty()) {
            val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

Konvertuj

a

Konvertor EURO USD

(setOnClickListener)




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

// old fashion
   convertBtn.setOnClickListener { v -> convert(v) }


    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")
        }
    }
```

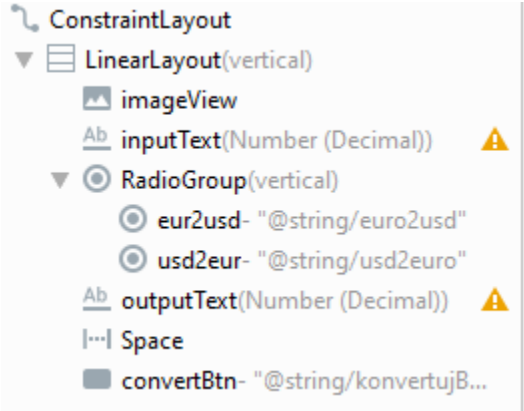
Project:Konvertor.zip

a

Konvertor EURO USD

(layout)

```
<LinearLayout
    <ImageView …/>
    <EditText …/>
    <RadioGroup
        <RadioButton …/>
        <RadioButton …/>
    </RadioGroup>
    <EditText …/>
    <Space …/>
    <Button …/>
</LinearLayout>
```



Project:Konvertor.zip

Životný cyklus apky

(prvý – zjednodušený nástrel)

AppLifeCycle

global: 0
local: 0
shared: 0

- Alt-Insert=Generate Override Implemented Methods:
  - `protected void onDestroy()`
  - `protected void onPause()`
  - `protected void onRestart()`
  - `protected void onRestoreInstanceState(Bundle savedInstanceState)`
  - `protected void onResume()`
  - `protected void onSaveInstanceState(Bundle outState)`
  - `protected void onStart()`
  - `protected void onStop()`

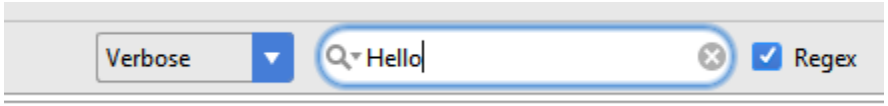
```
@Override

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Log.d("Hello", "onCreate"); // LOGUJTE, LOGUJTE, LOGUJTE
}
```

tag vhodný na filtrovanie

Project:AppLifeCycle.zip

LogCat

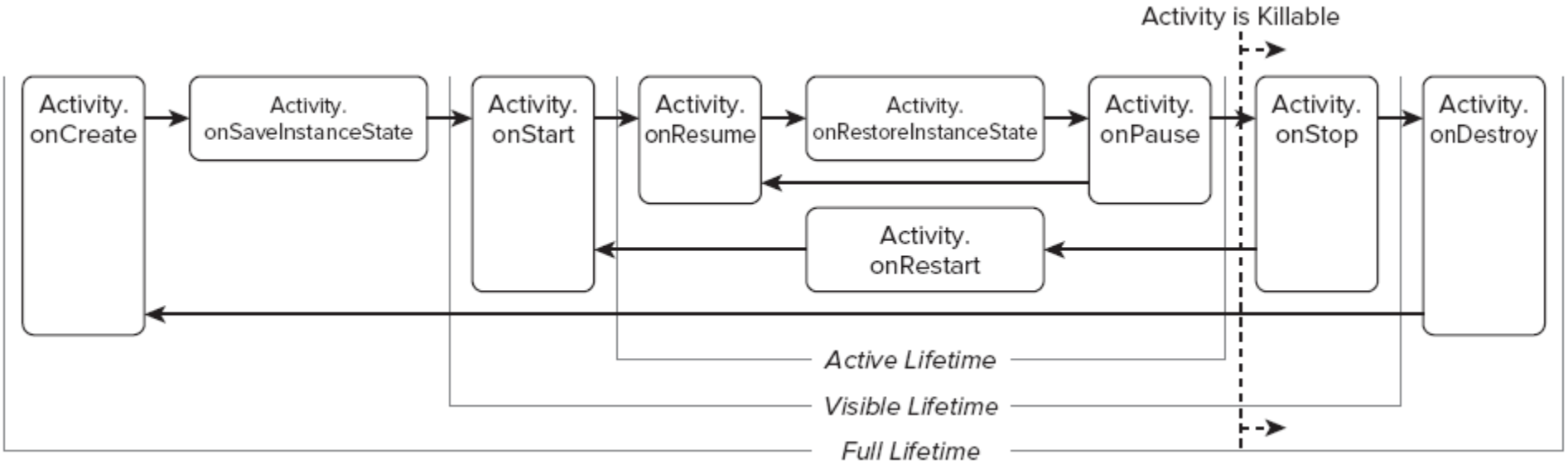
(Filtrovanie logov)



- 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



zdroj: Reto Meier: PA2AD

Project:AppLifeCycle.zip

Persistencia
(prvý dotyk)

AppLifeCycle

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`**

Project:AppLifeCycle.zip

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, globalCounter)
  ...
}
```

Project:AppLifeCycle.zip

SharedPreferences

SharedPreferences má metódy 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()
    }
}
```

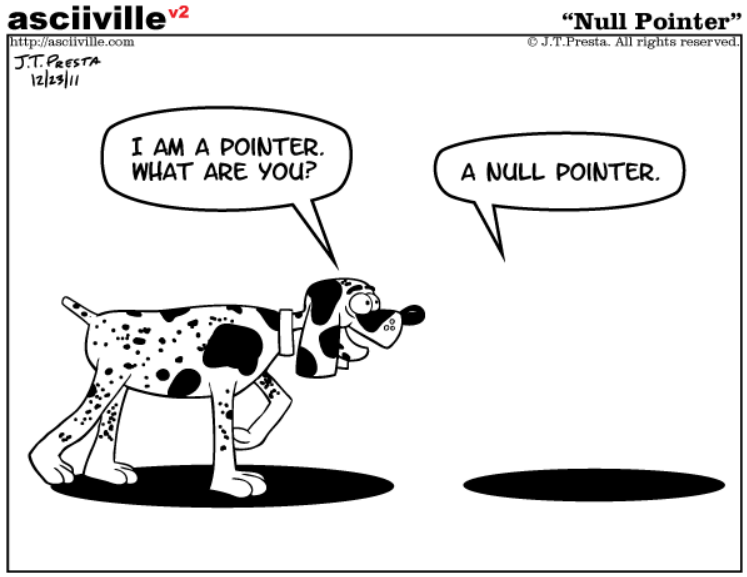
Project:AppLifeCycle.zip

Kotlin

Cheat sheets

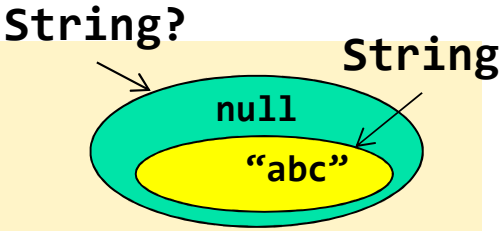
- https://www.programming-idioms.org/cheatsheet/Kotlin
- https://github.com/vmandro/Prednasky/tree/master/Kotlin

Nullables



To, čo je

- Optional v Jave, resp.
- Option v Scale, resp. 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 Elvis operátor

```
val c : String = a ?: "default, ak je prázdny reťazec"
```

Nullables

(ďalšie operátory na konverziu medzi type a type?)

- Elvis operátor

  `obj ?: default = if (obj == null) default else obj`

- Safe call (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.`

- Safe cast

  `obj as? T = if (obj typeof T) obj else null`

- let

  `obj?.let {…it…} = if (obj != null) {…it <- obj…}`

Nullables

(ešte raz, podrobnejšie)

V Jave je String skutočný retazec 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
}
```

Pikas 2018

(pikas – automaticky vygenerovaný 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])
}
```

01Intro/Pikas2018.zip

Kotlin

>=AS 3.0.1

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

Podobné ako java

Toast

bez väzby findViewByID

ohľahčená λ syntax

Project:01Intro/PikasKotlin.zip

Pikas 2018

(pikas – automaticky vygenerovaný Code/Convert Java->Kotlin)

```
val timer = Timer()
        timer.scheduleAtFixedRate(object : TimerTask() {
            override fun run() {
                cas++
                Log.d("PIKA", "onTICK")
                runOnUiThread { tv.text = "Cas:$cas" }
            }
        }, 1000, 1000)
    }
}
```

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

01Intro/Pikas2018.zip

Pikas 2018

(pikas – automaticky vygenerovaný Code/Convert Java->Kotlin)

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        Log.d("PIKA", "onCreate")
        // sem pisem moj start
        var prev: Button        var next: Button            var quit: Button
        var iv: ImageView       var i = 0                   var images: Array<Drawable
        var tv: TextView        var cas = 0

        prev = findViewById<View>(R.id.PrevBtn) as Button
        next = findViewById<View>(R.id.NextBtn) as Button
        quit = findViewById<View>(R.id.QuitBtn) as Button
        iv = findViewById<View>(R.id.imageView) as ImageView
        tv = findViewById<View>(R.id.textview) as TextView
        images = arrayOf<Drawable>(
            ContextCompat.getDrawable(applicationContext, R.drawable.butterfree)!! ,
            ContextCompat.getDrawable(applicationContext, R.drawable.golbat)!! ,
            ContextCompat.getDrawable(applicationContext, R.drawable.kakuna)!! ,
            ContextCompat.getDrawable(applicationContext, R.drawable.pikachu)!! ,
            ContextCompat.getDrawable(applicationContext, R.drawable.raichu)!! ,
            ContextCompat.getDrawable(applicationContext, R.drawable.venusaur)!! ,
            ContextCompat.getDrawable(applicationContext, R.drawable.venomoth)!! )
```

bang-bang operátor

01Intro/Pikas2018.zip

Konverzie Java-Kotlin

- Java -> Kotlin (ste videli minule)

Code/Convert Java File to Kotlin File (neuzná sa za plus bod v DÚ)

- Kotlin -> JVM Byte code

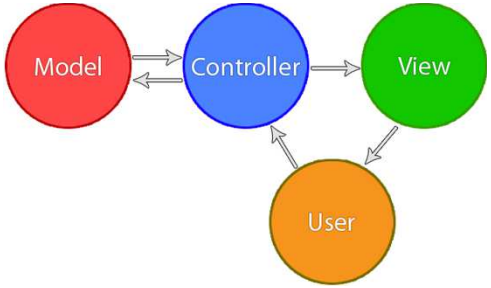
Tools/Kotlin/Show Byte Code

- 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



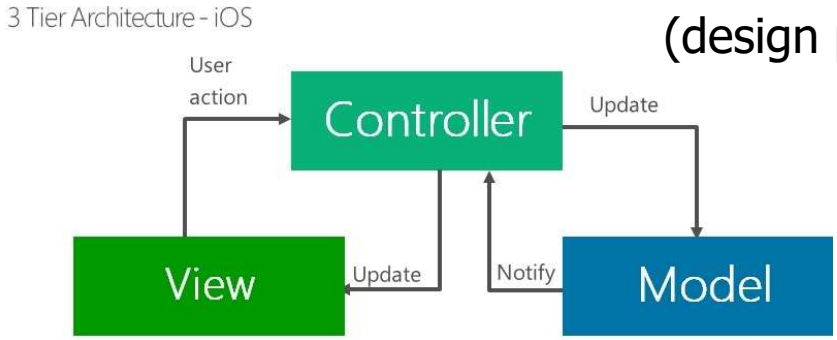
Architektonický *mess*

(MVC)

- vzniká, ak vizuálne komponenty (Views) sú zviazané s dátovými objektami a opačne

```
prev.setOnClickListener(new OnClickListener() {
  @Override
  public void onClick(View v) {
    i++;
    i %= imgs.length;
    iv.setImageDrawable(imgs[i]);
  }
});
```

- preto sa pri návrhu GUI používajú návrhové vzory, Model-View-Controller (design patterns)

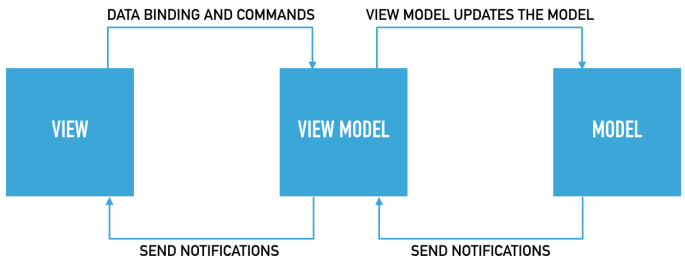


- motto: the architecture of most Android-apps is a mess.

http://doridori.github.io/Android-Architecture-MV%3F/#sthash.SiE5eude.IQq3XhmU.dpbs

MVC a MVVM patterny

- MVC (Model-View-Controler)
- MVVM (Model-View-ViewModel)

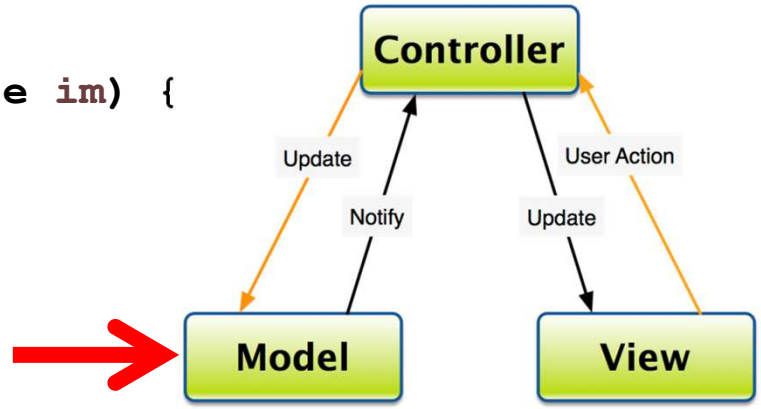


- Model = dáta, stav, Blogic
- View = vizuálne komponenty GUI
- ViewModel
- DataBinding library Google, 2015 Android JetPack 2018
- LiveData

Model View Controller (MVC)

(model – len data, netuší nič o ich prezentácii)

```
public class Model extends Observable {
int indx = 0;            // actual picture on the screen
ArrayList<Drawable> list = new ArrayList<Drawable>(); // all pics

public void addDrawableImage(Drawable im) {
   list.add(im);
}
public Drawable getDrawable() {
   return list.get(indx);
}
public void nextValue() {
   indx++;
   indx %= list.size();
   setChanged();
   notifyObservers();
}
```



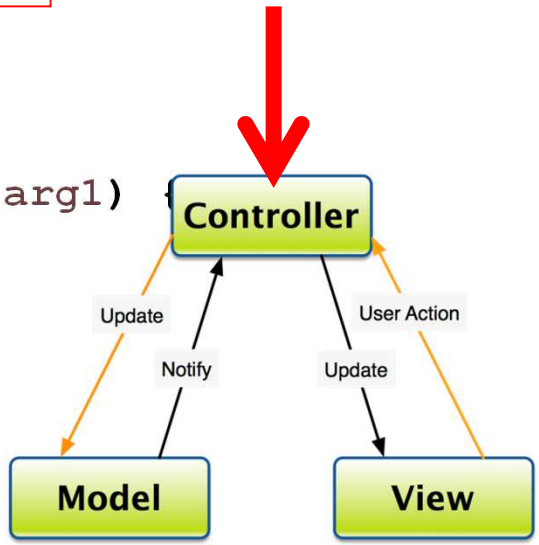

```
public void prevValue() {
   indx--;
   if (indx < 0)
      indx = list.size()-1;
   setChanged();
   notifyObservers();
}
```

01Intro/PikatchuMVC.zip

Model View Controller (MVC)

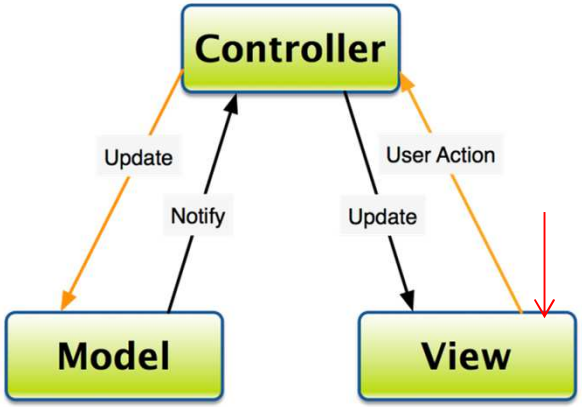
(controller – komunikuje medzi modelom a view)

```
public class Controller extends … implements Observer {

mModel = new Model();
mModel.addObserver(this);
mModel.addDrawableImage(getResources().getDrawable(R.drawable.pok0));
mModel.addDrawableImage(getResources().getDrawable(R.drawable.pok1));

mView = new myView(this);

@Override
public void update(Observable arg0, Object arg1) {
   mView.update(mModel.getDrawable());
}
```



01Intro/PikatchuMVC.zip

Model View Controller (MVC) (view)




```
public class myView {
    final Controller controller;
    ImageView iv;
    Button prev, next;

public myView(Controller c) {
    this.controller = c;
    iv = (ImageView)mainActivity.findViewById(R.id.imageView1);
    Button prev = (Button)mainActivity.findViewById(R.id.prevBtn);
    prev.setOnClickListener(new OnClickListener() {
    @Override
        public void onClick(android.view.View v) {
          controller.mModel.prevValue(); }
    });
public void update(android.graphics.drawable.Drawable im) {
    iv.setImageDrawable(im);
}
```

01Intro/PikatchuMVC.zip

Hitparáda DU1

(Hall of Fame)

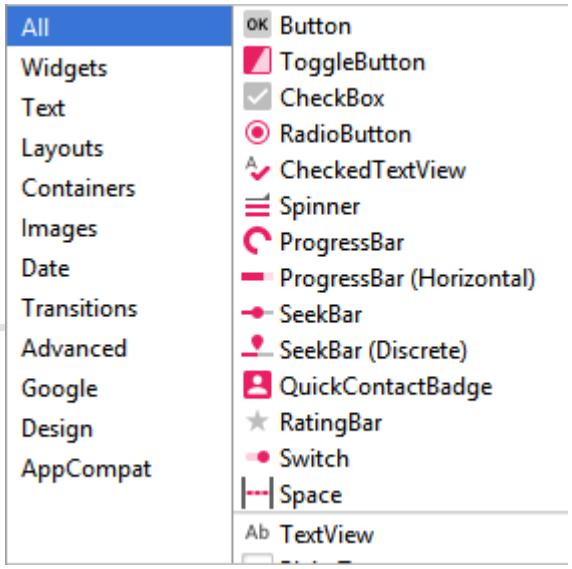
najviac sa mi páčili appky (poradie je náhodne):

Labilout – TamaraS. – vyhýbanie sa prekážkam s akcelerometrom

Miškove skúškové – LukášS. – Flappy birds s indexom ☹

Circle – FilipP. - chytanie správnej farby, používa akcelerometer

GUI komponenty



Layout

- LinearLayout (Verical/Horizontal)
- RelativeLayout, ConstraintLayout

View, ViewGroup

- všetky viditeľné komponenty (widgets)

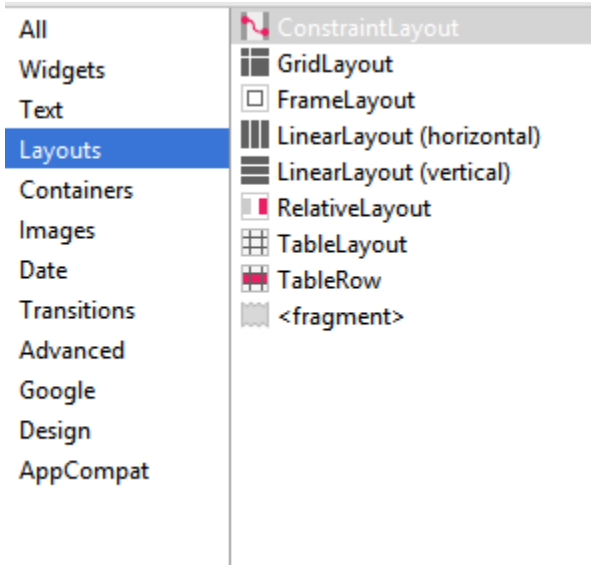
Activity - analógia Screenu (MIT), resp. Form/Frame

najznámejšie podtriedy

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

Fragment (>= API level 11)

- reusable UI components



Layouts

- 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)
- GridLayout (od API Level 14)

```
<FrameLayout
        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
          android:src="@drawable/ic_launcher" />

</FrameLayout>
```

Kód na slajde je zjednodušený, originál najdete v Layouts1.zip

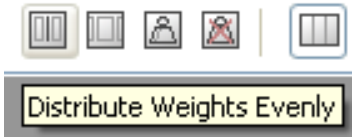
LinearLayout


```
<LinearLayout
    android:orientation="vertical"
    <LinearLayout
       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 najdete v Layouts1.zip

LinearLayout


```
<LinearLayout   ... Pokračovanie
        <LinearLayout
        android:orientation="horizontal"
        <Button
                android:id="@+id/logBtn"
                android:layout_weight="50"
                android:text="@string/Login"/>
        <Space
                android:layout_weight="50" />
        <Button
                android:id="@+id/forgetPass"
                android:layout_weight="50"
                android:text="@string/forget" />
        </LinearLayout>
</LinearLayout>
```



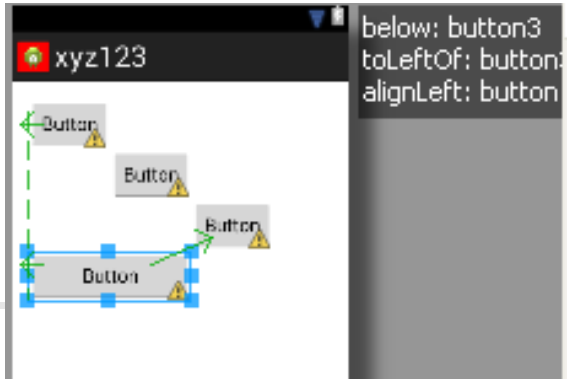
Kód na slajde je zjednodušený, originál najdete v Layouts1.zip

GridLayout


```
<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" />
```

Kód na slajde je zjednodušený, originál najdete v Layouts1.zip

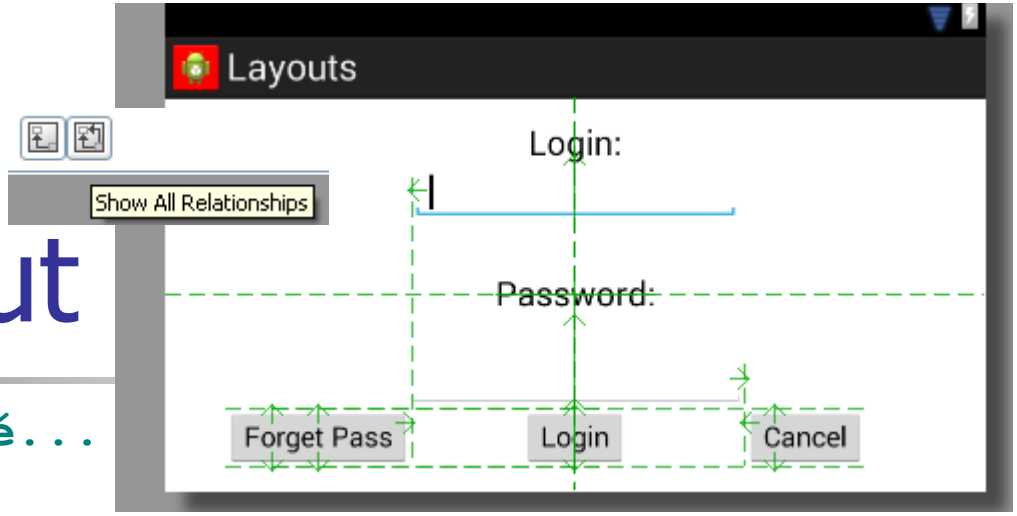
RelativeLayout




```
<RelativeLayout
    <Button
        android:id="@+id/button1"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"/>
    <Button
        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 najdete v Layouts1.zip

RelativeLayout



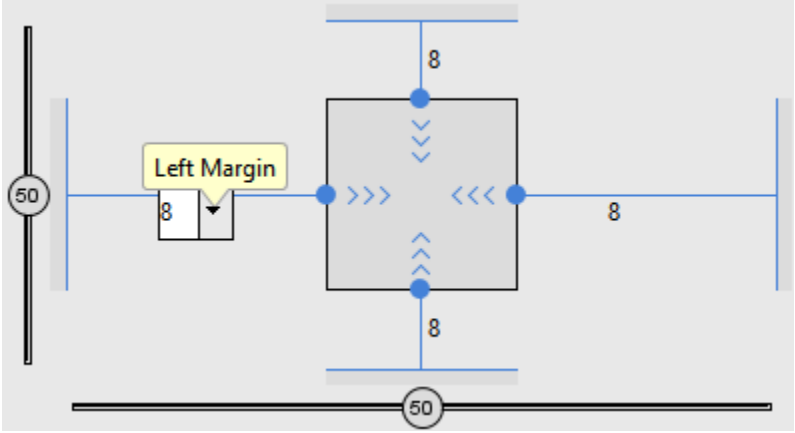

```
<RelativeLayout> ... skrátené...
 <EditText
            android:id="@+id/passwdtv"
            android:layout_below="@+id/pass"
            android:layout_centerHorizontal="true"/>
 <Button
            android:id="@+id/loginBtn"
            android:layout_below="@+id/passwdtv"
            android:text="@string/Login" />
 <Button
            android:id="@+id/forgetBtn"
            android:layout_alignBottom="@+id/loginBtn"
            android:layout_alignTop="@+id/loginBtn"
            android:layout_toLeftOf="@+id/passwdtv"
            android:text="@string/forget" />
```

Layouts1.zip

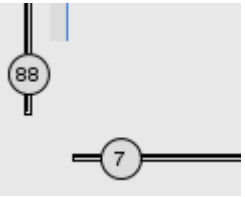
Constraint Layout

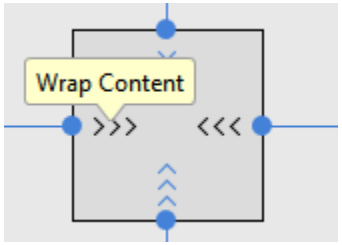
Umožňuje nastaviť

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



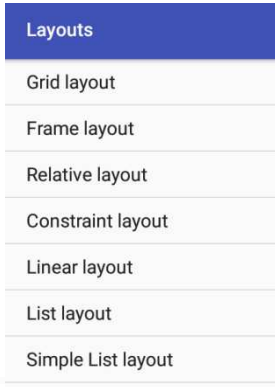
https://developer.android.com/reference/android/support/constraint/ConstraintLayout.html
https://www.youtube.com/watch?v=z53Ed0ddxgM





Kód na slajde je zjednodušený, originál najdete v Layouts1.zip

Intent - filter



Pohľad do AndroidManifest: intent-filter hovorí, na aký intent aktivita reaguje

```
<activity
    android:name=".Main2Activity"
    android:label="Layouts">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

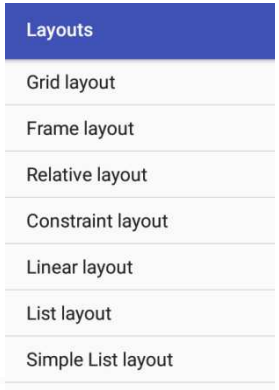
Spustí sa ako prvá

```
<activity
    android:name=".FrameLayoutActivity"
    android:label="@string/title_activity_frame_layout">
    <intent-filter>
        <action android:name="com.example.xyz123.FrameLayoutActivity"/>
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

nespustí sa

Kód na slajde je zjednodušený, originál najdete v Layouts1.zip

Intent - startActivity



ListView – najjednoduchšie použitie (ďalšie zložitejšie príklady budú v neskôr)

```
ListView listview = (ListView) findViewById(R.id.listViewID);
listview.setAdapter(new ArrayAdapter<String>(
            this, android.R.layout.simple_list_item_1,
            new String[]{
                    "Grid layout", "Frame layout", "Relative layout",
                    "Constraint layout", "Linear layout", "List layout",
                    "Simple List layout" }
    ));
    listview.setOnItemClickListener((adapterView, view, index, l) -> {//λ
     Intent in = null;
     switch (index) {  // škaredé, ale zrozumiteľné
       case 0: in = new Intent("com.example.xyz123.GridLayoutActivity"); break;
       case 1: in = new Intent("com.example.xyz123.FrameLayoutActivity"); break;
       ...
       case 6: in = new Intent("com.example.xyz123.MainActivity"); break;
     }
     if (in != null) {
        startActivity(in);
     }
```

Kód na slajde je zjednodušený, originál najdete v Layouts1.zip

ActiList project

V ďalšom uvidíme sériu rôznych nezávislých aktivít, ktoré ilustrujú:

- main_activity
  - button, TextView, onClickListener, onClick
- intro_activity
  - logo, intent, Thread-timer, MediaPlayer
- email_activity
  - listView, intent.putExtra, startActivityForResult, Toast
- canvas_activity
  - canvas/view – Draw, MultiTouch, onTouch, Option & Context Menu
- pisky_activity
  - piškvorky, začiatok aj koniec jednoduchej hry
- login_activity
  - ukladanie informácie pomocou SharedPreferences
- picpic_activity
  - čítanie/písanie do súboru, prehliadač obrázkov na SDkarte
- SQL_activity
  - prístup k SQLite databáze, create/drop/insert/select

Button

(findViewById, setOnClickListener)

MainActivity – 3 buttony, ktorými zarovnávame iný TextView vľavo, vpravo, stred

```
public class MainActivity extends Activity
                                    implements OnClickListener {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Button bLeft = (Button)findViewById(R.id.bLeft);
        Button bCenter = (Button)findViewById(R.id.bCenter);
        Button bRight = (Button)findViewById(R.id.bRight);

        bLeft.setOnClickListener(this);
        bCenter.setOnClickListener(this);
        bRight.setOnClickListener(this);
    }
```

3 views a jeden listener

01Intro/ActiList.zip

OnClick
(v OnClickListeneri)

```
public void onClick(View arg0) {
    Log.d("MainActivity", "onClick");
    TextView tv = (TextView)findViewById(R.id.textView1);
    Button btn = (Button)findViewById(arg0.getId());
    tv.setText(btn.getText());
    switch (arg0.getId()) {  // od koho prišiel click event
     case R.id.bLeft:tv.setGravity(Gravity.LEFT); break;
     case R.id.bRight: tv.setGravity(Gravity.RIGHT); break;
     case R.id.bCenter: tv.setGravity(Gravity.CENTER); break;
     default: Log.d("MainActivity", "nieco zle sa udialo...");
    }
```

tu zistíme na koho sme klikli (a je to int)

01Intro/ActiList.zip

Nepoužívame

Layout Design Editor

Dynamicky vytvárame layout v runtime:

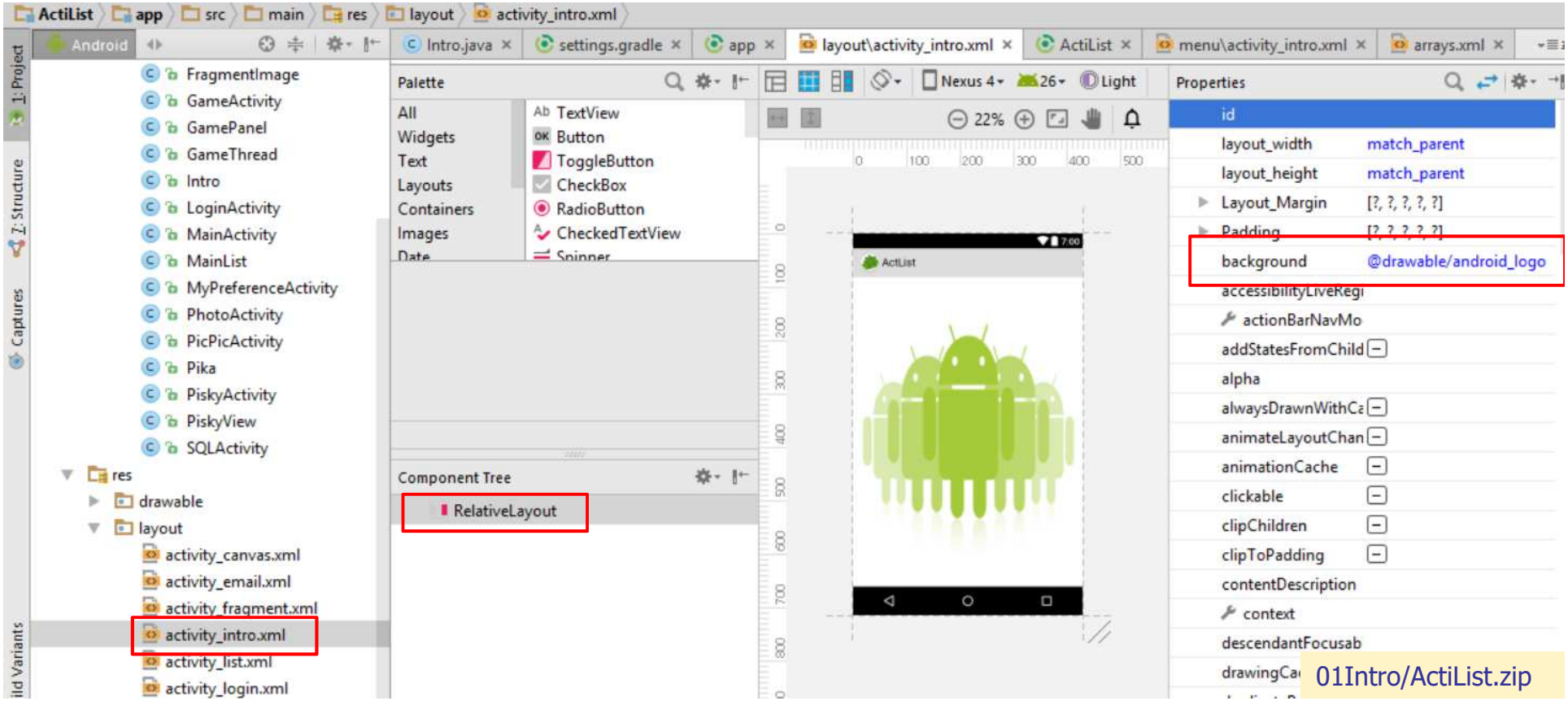
```
   LinearLayout ll = new LinearLayout(this);
   ll.setOrientation(LinearLayout.HORIZONTAL);
        Button startBtn = new Button(this);
        startBtn.setText("Start");
   ll.addView(startBtn);
        Button stopBtn = new Button(this);
        stopBtn.setText("Stop");
   ll.addView(stopBtn);
   setContentView(ll);  // zobraz vytvorený layout
   // Button startBtn = (Button)findViewById(R.id.bStart);
   startBtn.setOnClickListener(this);
public void onClick(View arg0) {
   switch (arg0) {      // od koho prišiel event
     case startBtn: /* go, go, go, ... */ break;
```

01Intro/ActiList.zip

Intro

chceme, aby sa pred spustením našej skutočnej apky:

- zobrazilo logo, intro-screen,
- zahrala melódia, ...



01Intro/ActiList.zip

IntroActivity

(Intent, timer)

IntroActivity – spustí timer (thread) odpočítavajúci čas pre úvodné logo+.mp3

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_intro); // tu sa zobrazí logo
    Thread timer = run() -> {  //λ
        try {
            sleep(4000);   // vychutnávame si logo
        } catch (Exception E) {
        } finally {        // na konci,zavolaj MainActivity
            Intent in = new   // vytvor Intent,ktorý ona chytá
                Intent("com.example.actilist.MainActivity");
            startActivity(in);        // tým sa IntoActvity pauzuje
        }                             // teda prejde cez onPause
    };
    timer.start();    // spusti timer, nezabudnúť...
```

01Intro/ActiList.zip

Intro/AndroidManifest.xml

```
<activity
   android:name=".MainActivity"
   android:label="@string/title_activity_main">
   <intent-filter>
     <action android:name="com.example.actilist.MAIN" />
   <category android:name="android.intent.category.LAUNCHER" />
   </intent-filter>
</activity>
<activity
    android:name=".Intro"
    android:label="@string/title_activity_intro" >
    <intent-filter>
    <action android:name="com.example.actilist.Intro" />
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

com.example.actilist.MainActivity

DEFAULT

…meno Intentu, na ktoré počúva aktivita

…táto sa nešpúšťa automaticky při štarte apky

…táto áno

01Intro/ActiList.zip

Intent - filter

- CATEGORY_BROWSABLE - ovláda web browser
- CATEGORY_LAUNCHER - ovláda spúšťač aplikácie
- *android.intent.action.MAIN – vstupný bod programu*

```
<intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

- CATEGORY_DEFAULT – startActivity/startActivityForResults

```
<intent-filter>
    <action android:name="com.example.actilist.CanvasActivity" />
    <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

spustenie:

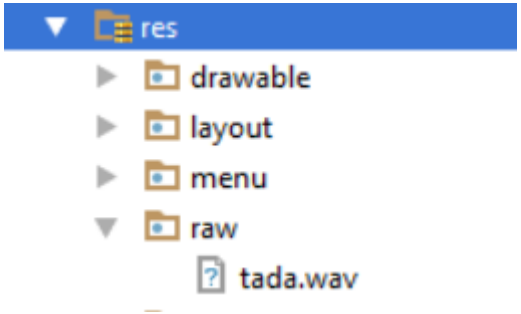
```
startActivity(new Intent("com.example.actilist.CanvasActivity"));
startActivity(new Intent(this, CanvasActivity.class))
```

ak máme:

```
package com.example.actilist;
   public class CanvasActivity extends Activity {
    ...
```

01Intro/ActiList.zip

MediaPlayer
(lokálne)



tada.mp3 [.wav] uložíme do project/res/raw
... a bude zakompilovaná do apky, a zazipovaná do zipky ☺

```
MediaPlayer mp;          // globálna premenná
// ak je muzička lokálna, v res/raw/tada.wav
mp = new MediaPlayer();
mp = MediaPlayer.create(Intro.this, R.raw.tada);
//mp.setLooping(true);
mp.start();

@Override
protected void onPause() { // IntroActivity je pauzovaná, keď
   super.onPause(); // odštartujeme com.example.actilist.MainActivity
   mp.release();    // uvoľníme MediaPlayer objekt, mp
   finish();        // akonáhle sa rozbehne MainActivy,
                    // IntroActivity  zanikne
}
```

01Intro/ActiList.zip

MediaPlayer

(cez Uri, more : http://dai.fmph.uniba.sk/courses/VMA/wave.mp3)

iná možnosť, tada.mp3 je prístupná niekde na sieti, dotiahneme ju a zahráme

```
try { // problém:apka musí deklarovať, že chce prístup na internet
   mp = new MediaPlayer();
   Uri uri = // ak je muzička na webe, jej dotiahnutie može niečo trvať
         Uri.parse("http://dai.fmph.uniba.sk/courses/VMA/wave.mp3");
   mp.setAudioStreamType(AudioManager.STREAM_MUSIC);
   mp.setOnPreparedListener(this); // info, keď sme ready zahrať
   mp.setDataSource(getApplicationContext(), uri);
   mp.prepare();  // tu sa spustí doťahovanie súboru
 } catch (Exception e) {
   Log.d("Intro", "Uri error: " + e.getMessage());
 }
public class Intro ... implements MediaPlayer.OnPreparedListener
public void onPrepared(MediaPlayer arg0) {
   mp.start();    // keď sme ready, tak môžeme zahrať
}  AndroidManifest.xml
<uses-permission android:name="android.permission.INTERNET" />
```

01Intro/ActiList.zip

MediaPlayer

(na SD-karte, resp. v internej pamäti)

Môžeme sa skúšať triafať do správnej cesty muziky, obrázku, či súboru:

```
mp.setDataSource("/mnt/sdcard/Music/tada.wav");
mp.setDataSource("/mnt/sdcard/Music/wave.mp3");
mp.setDataSource("/storage/sdcard0/Music/wave.mp3");
mp.setDataSource("/Removable/SD/Music/wave.mp3");
```

alebo použiť symbolickú cestu:

```
String filePath =    Environment.getExternalStorageDirectory()+
                     "/Music/wave.mp3";
Log.d("Intro",filePath); // vždy si zalogujte cestu,
                         // aby ste vedeli, kde súbor hľadá
mp.setDataSource(filePath); // hneď viete, prečo to nehrá...
mp.prepare();
```

```
10-23 08:08:11.907: D/Intro(27414): /storage/sdcard0/Music/wave.mp3
```

01Intro/ActiList.zip

ListActivity vs. ListView

(setListAdapter – pre jednoduchý typ ListView)

- Všetky aktivity zoradíme do zoznamu, z ktorého sa vybratím daná aktivita spustí

```
... ako naplniť zoznam/list ...
public class MainList extends ListActivity {
// konštanta v programe, ktorou naplníme ListView
String[] myActivities = {"Intro“,"MainActivity","EmailActivity”,…};
@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
// alternatíva: zoznam aktivít je v
// res/values/strings
myActivities = getResources().getStringArray(R.array.listofallact);
setListAdapter(
   new ArrayAdapter<String>(MainList.this,        // kde je ListView
        android.R.layout.simple_list_item_1,      // typ ListView
        myActivities));                     // hodnoty do ListView
}
```

```
<string-array name="listofallact">
    <item>Intro</item>
    <item>MainActivity</item>
    <item>EmailActivity</item>
</string-array>
```

01Intro/ActiList.zip

ListActivity
(onListItemClick, reflexivita)

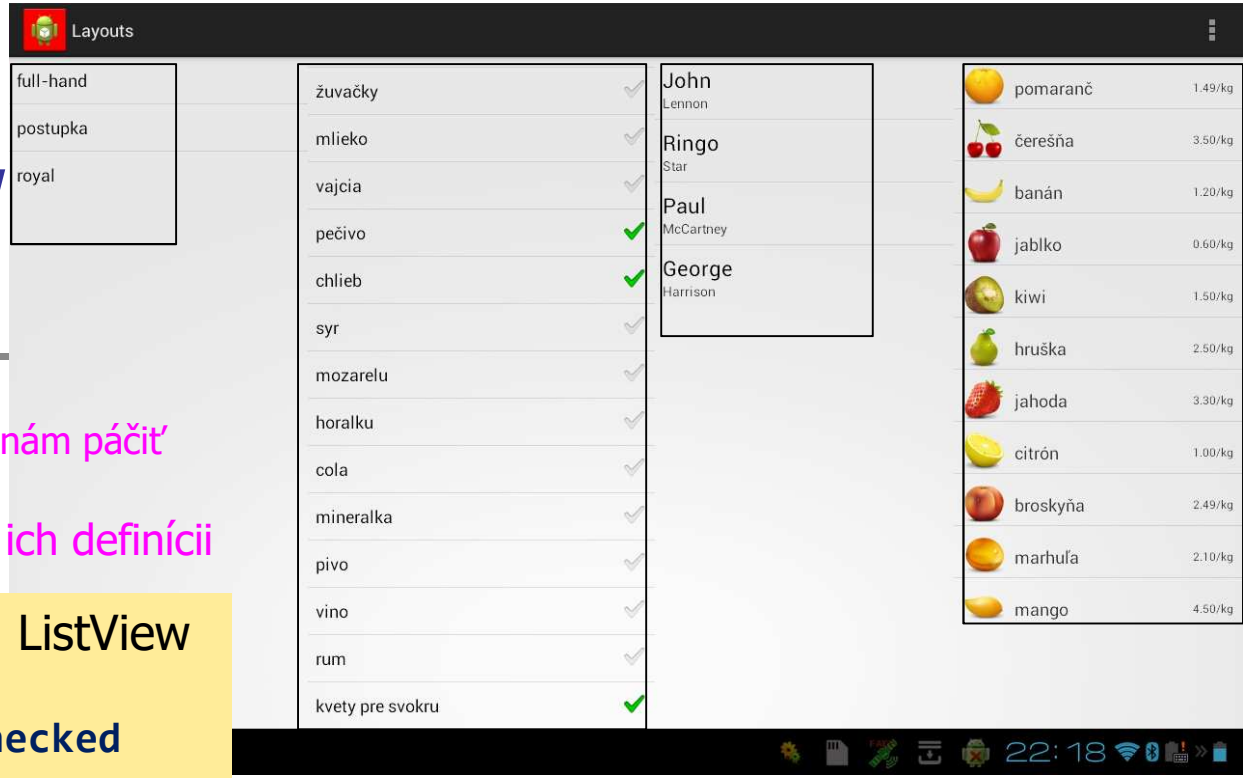
Ako z mena aktivity vyrobiť intent, ktorý aktivitu spustí, resp. meno balíka (package), v ktorom je aktivita definovaná

```
protected void onListItemClick(ListView l, View v,
                                    int position, long id) {
   String className = myActivities[position]; // napr. "Intro"
   try {
        Class mainClass =        // wow !! Reflexivita v praxi ;-)
                Class.forName("com.example.actilist."+ className);
        Intent in = new Intent(MainList.this, mainClass);
        startActivity(in);
   } catch (Exception E) {
        E.printStackTrace();
   }
}
```

01Intro/ActiList.zip

ListView
(variabilita)

- preddefinovaný štýl
  - môžu/nemusia sa nám páčiť
- user defined
  - narobíme sa pri ich definícii

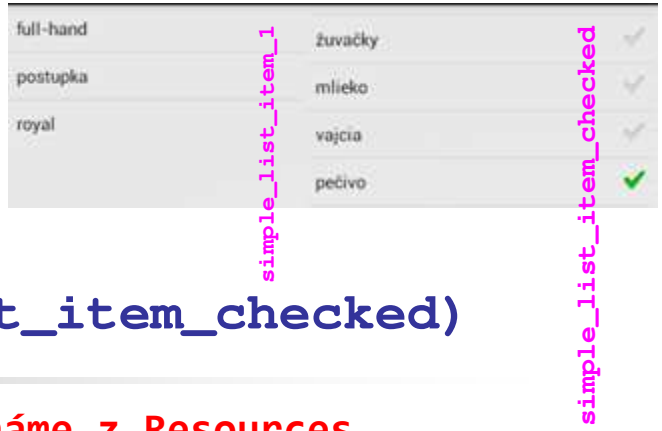


Štyri rôzne inštancie ListView
`simple_list_item_1`,
`simple_list_item_checked`
`simple_list_item_2`
a vlastný row layout

Sústredili sme sa na layout, a neriešili sme odchytávanie udalostí v ListView

```
10-19 16:11:00.179: D/onItemClick(17189): item click: 0:full-hand
10-19 16:11:01.569: D/onItemClick(17189): item click: 1:postupka
10-19 16:11:02.729: D/onItemClick(17189): item click: 2:royal
10-19 16:11:04.659: D/onItemClick(17189): checked: true:maslo
10-19 16:11:06.839: D/onItemClick(17189): checked: true:mlieko
10-19 16:11:07.769: D/onItemClick(17189): checked: true:kvety pre svokru
10-19 16:11:10.409: D/onItemClick(17189): item click: 1
```

01Intro/ActiList.zip



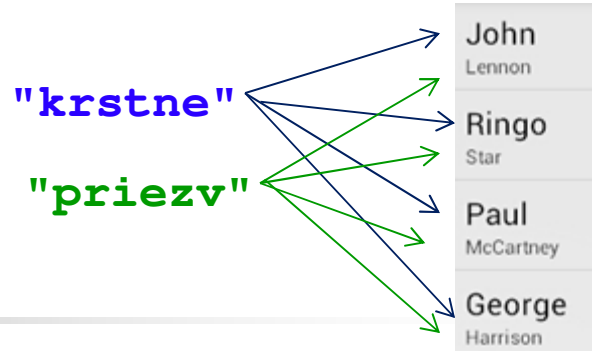
ListView 1

(simple_list_item_1, simple_list_item_checked)

```
String[] pList =          // ak dáta/zoznam ťaháme z Resources
        getResources().getStringArray(R.array.poker);
ListView lv1 = (ListView) findViewById(R.id.listView1);
// pre lv1 vytvoríme ArrayAdapter, poskytneme dáta pre ListView
// ListView Adapter prepojí data v zozname s ich view
lv1.setAdapter(new ArrayAdapter<String>(this,
        android.R.layout.simple_list_item_1,
        pList));// to isté aj pre simple_list_item_checked
lv1.setOnItemClickListener(this);        // priviaž listener k lv1

-------- onItemClick interface vyžaduje onItemClick metódu
public void onItemClick(AdapterView<?> la, View v, int indx...
final String item = (String) la.getItemAtPosition(indx);
   Log.d("onItemClick", "item click: " + indx + ":" + item);
   CheckedTextView tv = (CheckedTextView) v;
   tv.toggle();  // zmení checked na unchecked, a vice-versa
   Log.d("onItemClick","checked:"+tv.isChecked()+":"+ item1)
```

01Intro/ActiList.zip

ListView 2
(simple_list_item_2)



Naplniť iný, napr. dvojriadkový ListView je náročnejšie

```
ArrayList<Map<String, String>> pairs = new ArrayList<Map<String, String>>();
HashMap<String, String> item1 = new HashMap<String, String>();
    item1.put("krstne", "John"); // 1.riadok 1.položky
    item1.put("priezv", "Lennon");        // 2.riadok 1.položky
    pairs.add(item1);                     // 1.položka zoznamu pairs
   .....                          // a ďalší „bítlsáci“ ...
ListView lv3 = (ListView) findViewById(R.id.listView3);
String[] from = {"krstne", "priezv" };// symbolické mená riadkov
int[] to = { android.R.id.text1, android.R.id.text2 };
//čo je text1,2 http://developer.android.com/reference/android/R.id.html

lv3.setAdapter(
        new SimpleAdapter(this, pairs, // context a zoznam riadkov
        android.R.layout.simple_list_item_2, from, to)); // šablóna
lv3.setOnItemClickListener(this);
```

Kód na slajde je zjednodušený, originál najdete v Layouts1.zip

Rôzne preddefinované ListView

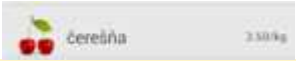
(prehľad)

zdroj: https://developer.xamarin.com/guides/android/user_interface/working_with_listviews_and_adapters/part_3_-_customizing_a_listview's_appearance/

Vlastný ListView

Myšlienka: musíme definovať náš vlastný ArrayAdapter

```
FruitArrayAdapter fruitArrayAdapter = new FruitArrayAdapter(
                ActivityList3.this,
                R.layout.listview_row_layout);
lv4.setAdapter(fruitArrayAdapter);  // ktorý podhodíme ListView
lv4.setOnItemClickListener(this);

fruitArrayAdapter.add(          // ktorý naplníme vlastnými objektami
   new Fruit(R.drawable.apple,"jablko","0.60/kg"));

public class Fruit {
   private int fruitImg;
   private String fruitName;
   private String price;
...                         // musí poskytovať getView:index->View
}
```

1 –> 

Kód na slajde je zjednodušený, originál najdete v Layouts.zip

ArrayAdapter

```
public class FruitArrayAdapter extends ArrayAdapter<Fruit> {
  private List<Fruit> fruitList = new ArrayList<Fruit>();
  static class FruitViewHolder {
       ImageView fruitImg;
       TextView fruitName, price;
  }
  public void add(Fruit object) {
        fruitList.add(object);
  }
  public View getView(int position, ...) {
      View row = ... R.layout.listview_row_layout ...
      row.setTag(viewHolder);
      Fruit fruit = getItem(position);
      viewHolder.fruitImg.setImageResource(fruit.getFruitImg());
      viewHolder.fruitName.setText(fruit.getFruitName());
      viewHolder.price.setText(fruit.getPrice());
      return row;
  }
```

1 ->

Kód na slajde je zjednodušený, originál najdete v Layouts1.zip

EmailActivity

(data do intentu, startActivityForResult s callbackom)

```
Toast.makeText(EmailActivity.this, "posielam mail cez mail klienta",
                Toast.LENGTH_LONG)
                .show(); // na toto nezabudnite ...
                         // ako zistím meno správneho intentu ???
   Intent in = new Intent(android.content.Intent.ACTION_SEND);
   in.setType("text/plain");              // a ešte aj argumenty
                http://developer.android.com/reference/android/content/Intent.html
   in.putExtra(android.content.Intent.EXTRA_SUBJECT, // subjekt mailu
                subjectString);
   in.putExtra(android.content.Intent.EXTRA_EMAIL,
                new String[] { emailString }); // pole adresátov
   in.putExtra(android.content.Intent.EXTRA_TEXT, // text mailu
                bodyString);
   // startActivity(in);  // nie!keďže chcem sa dozvedieť výsledok ☹
   startActivityForResult(in, 777);// očakávame výsledok od aktivity
   // 777 je symbolické meno, pod ktorým dostaneme výsledok
```

```
final int ACTION_EMAIL_SEND = 777
```

01Intro/ActiList.zip

EmailActivity

(onActivityResult = callback)

```
final int ACTION_EMAIL_SEND = 777
```

```
protected void onActivityResult(int requestCode, int
                                resultCode, Intent data) {
    if (requestCode == 777)     // prišiel výsledok s menom 777
        if (resultCode == RESULT_OK) {
            Toast.makeText(EmailActivity.this,
                "mail poslany", Toast.LENGTH_SHORT).show();
            finish();
        } else if (resultCode == RESULT_CANCELED) {
            Toast.makeText(EmailActivity.this,
                "mail neposlany", Toast.LENGTH_SHORT).show();
        } else
            Toast.makeText(EmailActivity.this,
                "neviem poslat mail: " + resultCode,
                Toast.LENGTH_SHORT).show();
    else
        Toast.makeText(EmailActivity.this,
            "nejaky iny vysledok", Toast.LENGTH_SHORT).show();
}
```

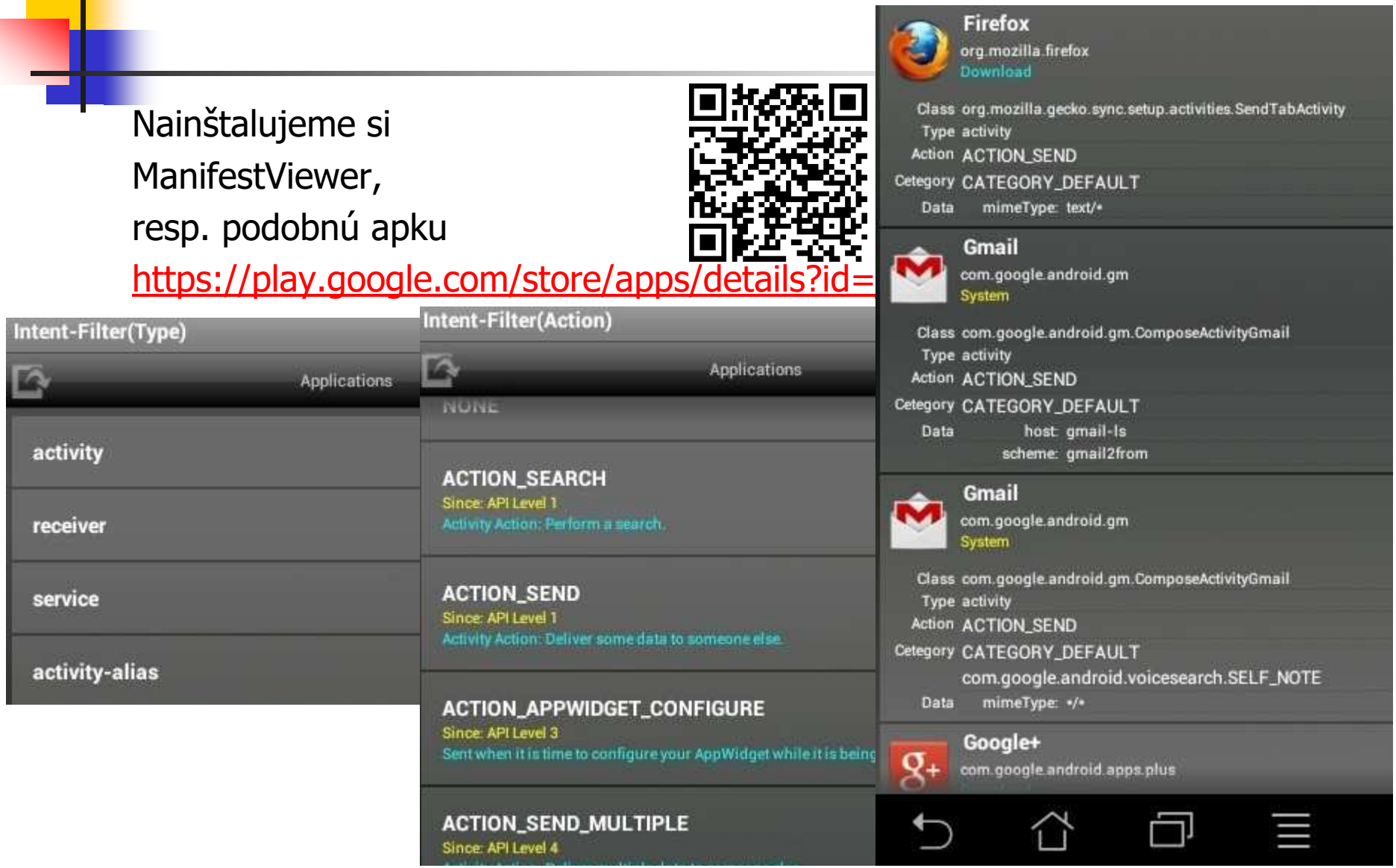
Project:01Intro/ActiList.zip

01Intro/ActiList.zip

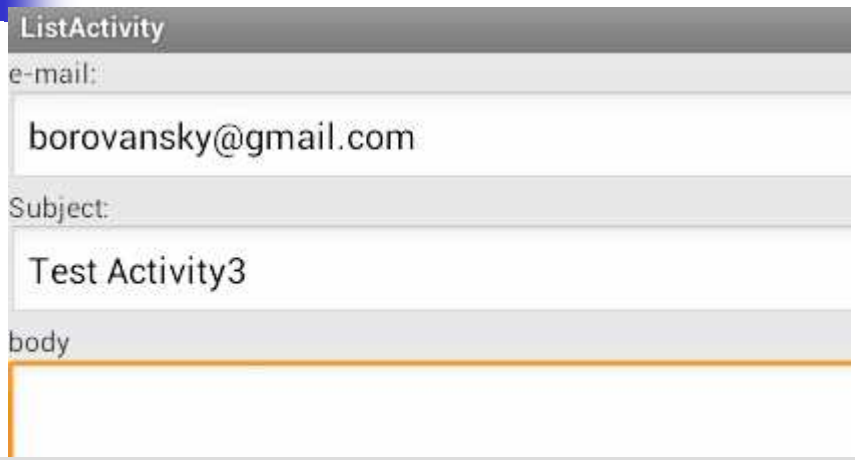
Kto všetko chytá intent ?

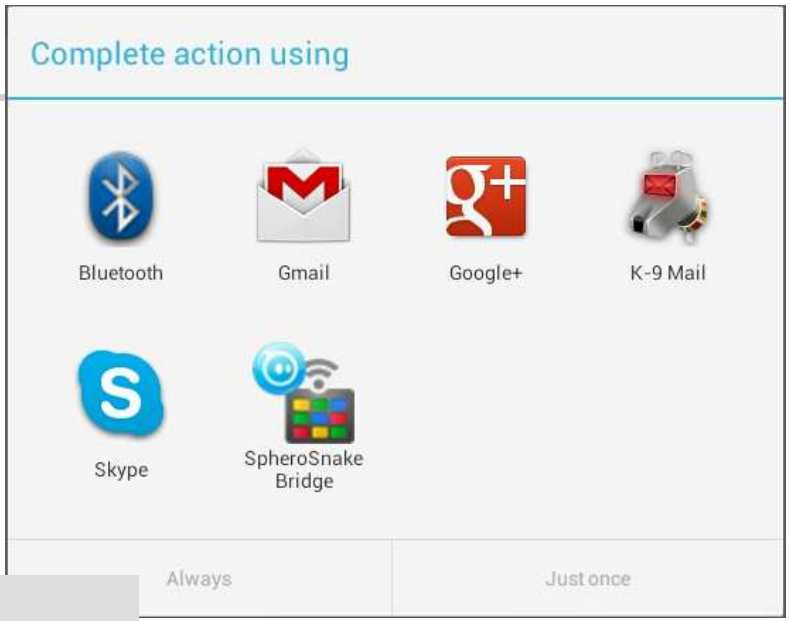
`android.content.Intent.`*`ACTION_SEND`*

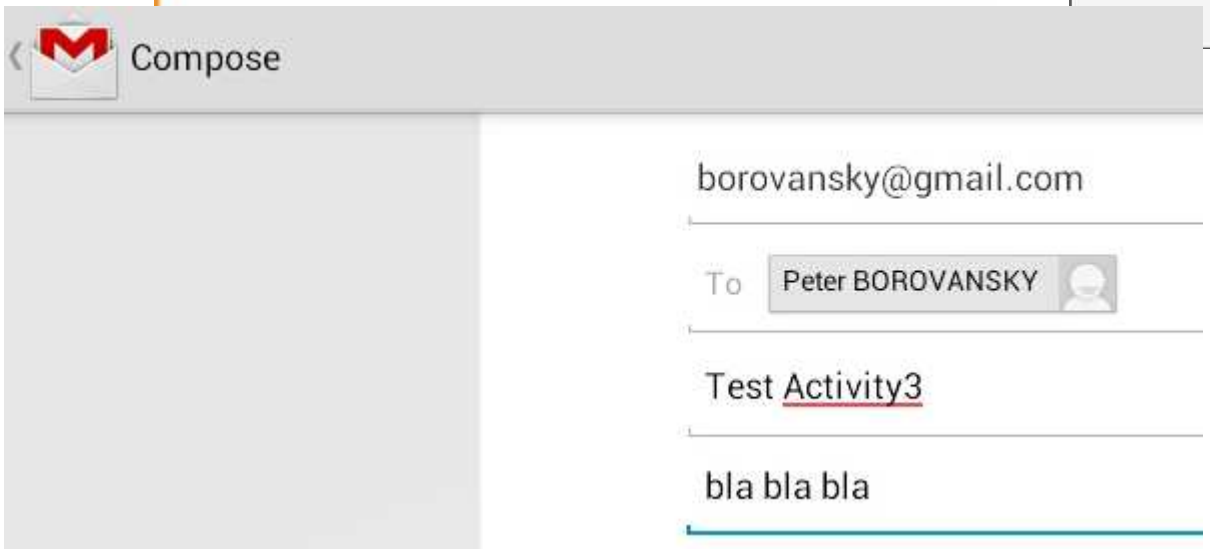
Nainštalujeme si
ManifestViewer,
resp. podobnú apku
https://play.google.com/store/apps/details?id=

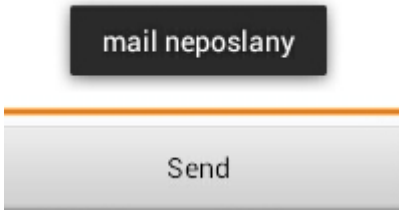


EmailActivity









01Intro/ActiList.zip

Prečo to vždy vracia

`resultCode == RESULT_CANCELED`

http://developer.android.com/reference/android/content/Intent.html#ACTION_SEND

PhotoActivity

(data z intentu)

Princíp intent-startActivityForResult spolu s onActivityResult ešte raz:

```
public class PhotoActivity extends Activity {
   final private static int REQUEST_IMAGE_CAPTURE = 888;
   protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_photo);
     Button takePhoto = (Button)findViewById(R.id.takePictureBtn);
     takePhoto.setOnClickListener(new View.OnClickListener() {
       public void onClick(View v) {
         Intent in = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
         if (in.resolveActivity(getPackageManager()) != null) {
            Toast.makeText(PhotoActivity.this,
             "smile ... taking picture", Toast.LENGTH_LONG).show();
            startActivityForResult(in, REQUEST_IMAGE_CAPTURE);
          } else {
            Toast.makeText(PhotoActivity.this, "sorry ... no picture", Toast.LENGTH_LONG).show();
          }
        }
     });
   }
```

https://developer.android.com/reference/android/provider/MediaStore.html#ACTION_IMAGE_CAPTURE

01Intro/ActiList.zip

PhotoActivity
(data z intentu)

V callback onActivityResult získavame z indentu data/odfotený obrázok:

```
protected void onActivityResult(int requestCode, int resultCode,
        Intent data) {

   if (requestCode == REQUEST_IMAGE_CAPTURE &&
        resultCode == RESULT_OK) {
      Toast.makeText(PhotoActivity.this,
        "thanks ... finito", Toast.LENGTH_LONG).show();

      ImageView photoImageView = (ImageView)
                findViewById(R.id.pictureImageView);
      Bundle extras = data.getExtras();
      Bitmap imageBitmap = (Bitmap) extras.get("data");

      photoImageView.setImageBitmap(imageBitmap);
   }
}
```

01Intro/ActiList.zip