Snívajú používatelia o mobilných androidoch?

platforma pre mobilné aplikácie

Róbert Novotný
robert.novotny@upjs.sk

22. 3. 2012

ANDROID

Prečo mobilné aplikácie?

- mobily sú **všade**
- operátori ponúkajú **lacné mobilné pripojenie** k internetu
- priestor pre zariadenia, ktoré možno mať po ruke vždy a všade
- plus výhody špeciálnych **dodatkov**
  - fotoaparát
  - GPS
  - FM rádio

Explózia nápadov

Aktuálny stav na svetovom trhu

ANDROID

Stručná história Androidu

- september 2007
  - prvé špekulácie o Google mobile
- november 2007
  - ohlásené konzorcium Open Handset Alliance
  - vízia: vyvinúť otvorené štandardy pre mobily
  - vzápäť: Android: platforma pre mobilné zariadenia
- september 2008:
  - HTC Dream a.k.a. T-Mobile G1
  - Android 1.0
- marec 2010:
  - prvé dva dotované mobily ponúkané slovenským operátorom
- marec 2011: cca 100 zariadení k dispozícii
- marec 2012: cca 150 mobilov + 50 tabletov + čítačky, netbooky...

Prečo je Android úspešný?

- otvorenosť
  - otvorená platforma: Linux + Java
  - open-source
- množstvo zariadení
  - 11 výrobcov
  - pestrá paleta hardvérových možností
- ľahký vývoj
  - súvis s otvorenou platformou
  - rozumné nástroje
- trh aplikácií
  - Android Market (júl 2010: 100 k aplikácií)

Hardvérová platforma: typické zariadenie

- HTC Desire (Bravo)
- CPU: 1 GHz Qualcomm QSD 8250
  - architektúra ARM (RISC, 32-bit)
  - schopný prehrávať lHD video
- RAM: 576 MB
- Úložisko: 512 MB flash + microSDHC karty
- Displej: 3,7", 480 x 800
- fotoaparát 5 Mpx, GPS, Bluetooth, WiFi, MicroUSB, FM rádio

Hardvérová platforma: typické zariadenie 2012

- Samsung Galaxy S2
- CPU: 1,5 GHz Qualcomm Snapdragon S3
  - architektúra ARM (RISC, 32-bit)
- RAM: 1 GB
- Úložisko: 16 GB flash + microSDHC karty
- Displej: 4,3", 480 x 800
- fotoaparát 8 Mpx, GPS, Bluetooth, WiFi, MicroUSB, USB Host, NFC, DLNA, HDMI, FM rádio

Vývoj aplikácií

- Android SDK
  - voľne dostupná sada nástrojov pre vývoj aplikácií
  - hlavne emulátor!
- API Levels
  - nezávislé platformy v rámci SDK
  - podobne ako existujú verzie Java Development Kitov
  - obvykle vyvíjame pre jednu konkrétnu platformu
- Eclipse ADT:
  - voľne dostupný plug-in do Eclipse pre vývoj aplikácií

Vývoj aplikácií: API Levels

- s každou verziou Androidu bohatšia platforma
- API Level definuje minimálne hardvérové požiadavky pre zariadenie
- ekvivalentné verzii Androidu
  - Android 2.1 = API Level 9
- aplikácia môže ďalej definovať špeciálne požiadavky
  - Android 2.3:
    - nutné: 2 MPx fotoaparát
    - predpokladaný: autofokus
    - voliteľné: pevné ohnisko, blesk

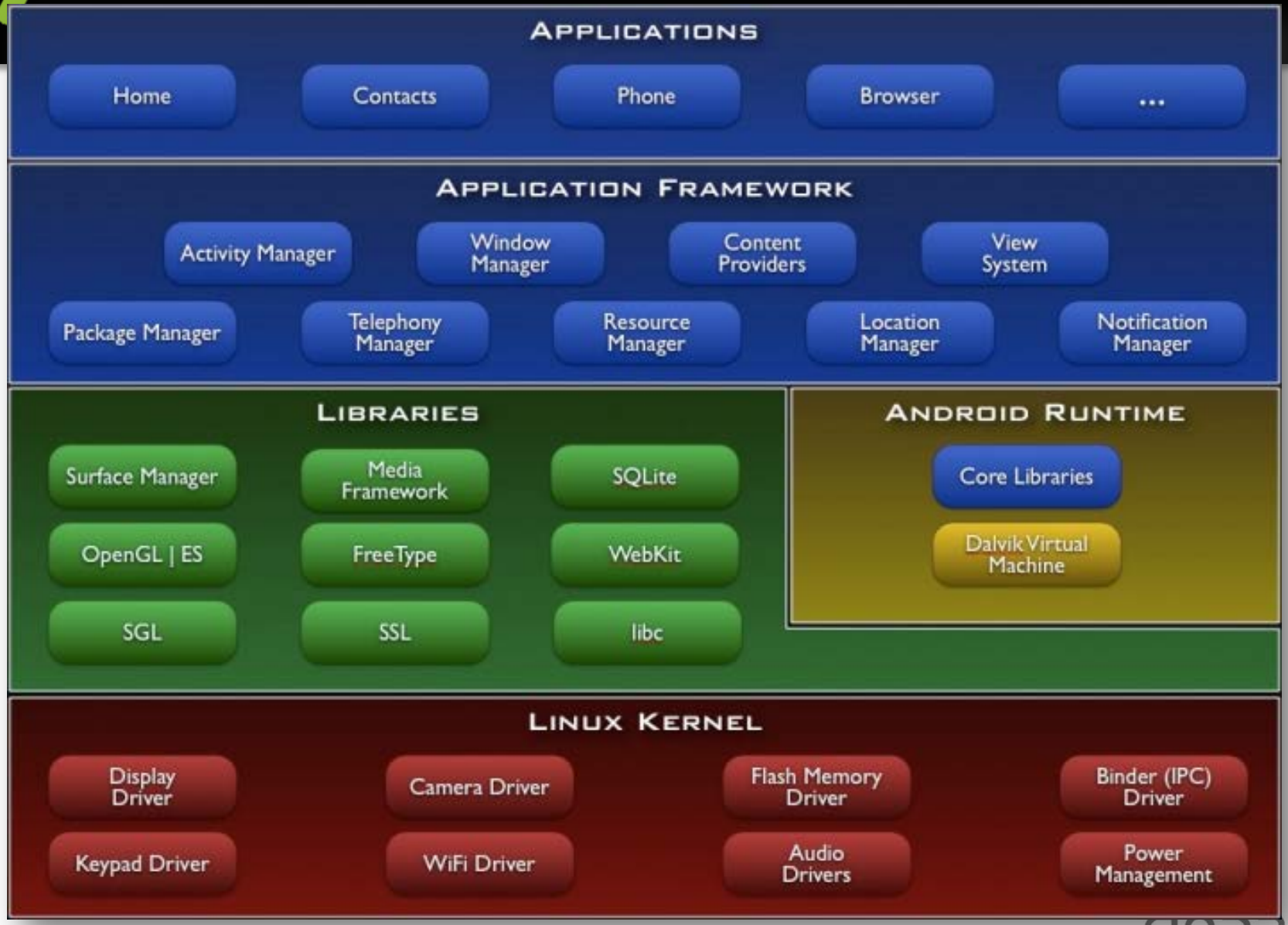
Java!

- primárny jazyk vývoja: **Java**!
- ustálený objektovo-orientovaný jazyk
- veľký marketingový podiel
- tony knižníc
- množstvo nástrojov
- prijateľná licencia
- koncept virtuálneho stroja, ktorý je dostatočne výkonný

Sú aj iné jazyky?

- v prípade výpočtovo náročných vecí sa možno ponoriť o úroveň nižšie
- C/C++
- Android Native Development Kit

Pohľad do útrob Androida



android

Dalvik: virtuálny stroj pre Javu

- deja-vu z Java ME
- Cieľ: *virtuálny stroj určený pre slabšie CPU, s málom pamäte, bez swapu, napájané batériou*
- vlastná sada inštrukcií
  - iná než má Java
- registrová architektúra
  - Java VM: zásobníková
  - o tretinu menej inštrukcií
  - o tretinu väčšie inštrukcie

Dalvik: virtuálny stroj pre Javu

.java

- zdrojový kód v jazyku Java

.class

- skompilovaný súbor („binárka")
- sada inštrukcií pre virtuálny počítač JVM

.dex

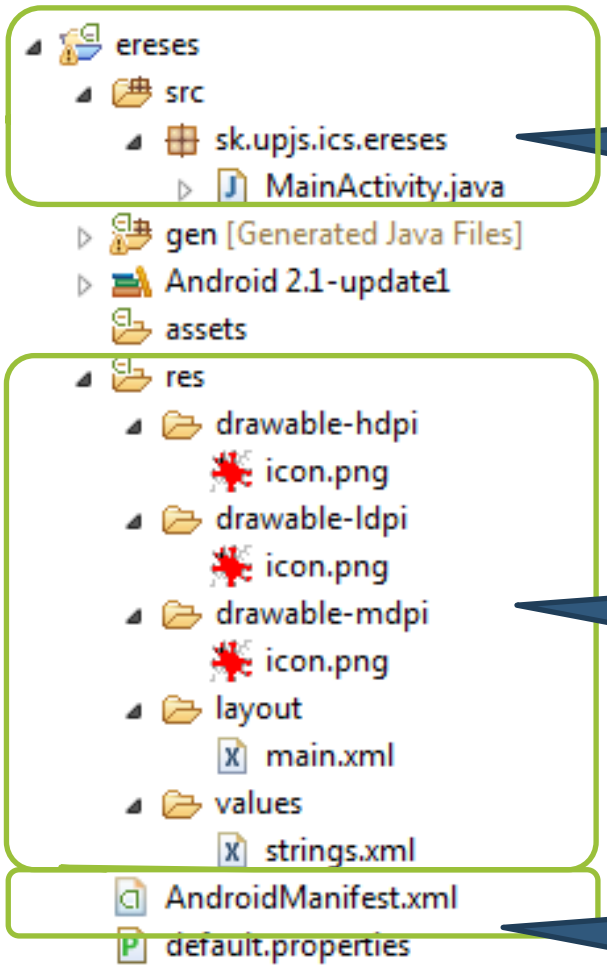
- sada inštrukcií pre Dalvik
- efektívnejší, viac tried v 1 súbore

Java či Nejava?

- licencia Javy: GPL + Classpath Exception
  - všetko je GPL = modifikácie musia ísť pod GPL
  - okrem kódu, ktorý beží pod JVM Standard Edition
- na Java ME sa nevzťahuje Classpath Exception!
  - výrobca sa musí dohodnúť s Oraclom
  - alebo vypustiť kód virtuálneho stroja pod GPL

Dalvik: implementácia na zelenej lúke,
ktorá nie je Java ME!

Čo vypadne z generátora



Java zdrojáky

resources
(ikony, obrázky, textové reťazce)

manifest

Manifest: AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="sk.upjs.ics.ereses"
      android:versionCode="1" android:versionName="1.0">

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".MainActivity"
                  android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

- hlavný konfiguračný súbor pre aplikáciu
- definuje jednotlivé komponenty aplikácie
- určuje konfiguračné nastavenia
- špecifikuje oprávnenia aplikácie

FONDAZIONE E MANIFESTO
DEL
FUTURISMO
(Pubblicato dal "FIGARO" di Parigi il 20 Febbraio 1909)
Avevamo vegliato tutta la notte — i miei amici ed io — sotto lampade di moschea dalle

Reťazce: globalizáciou k lokalizácii

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, MainActivity!</string>
    <string name="app_name">Ereses</string>
</resources>
```

- všetky reťazce sú zhromaždené do súboru **strings.xml**
- umožňuje to efektívnejšie uloženie
- podporuje to internacionalizáciu a lokalizáciu
- možno sa na ne odkázať v manifeste i v kóde

Reťazce v manifeste

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="sk.upjs.ics.ereses"
      android:versionCode="1" android:versionName="1.0">

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".MainActivity"
                  android:label="@string/app_name">

...
```

strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, MainActivity!</string>
    <string name="app_name">Ereses</string>
</resources>
```

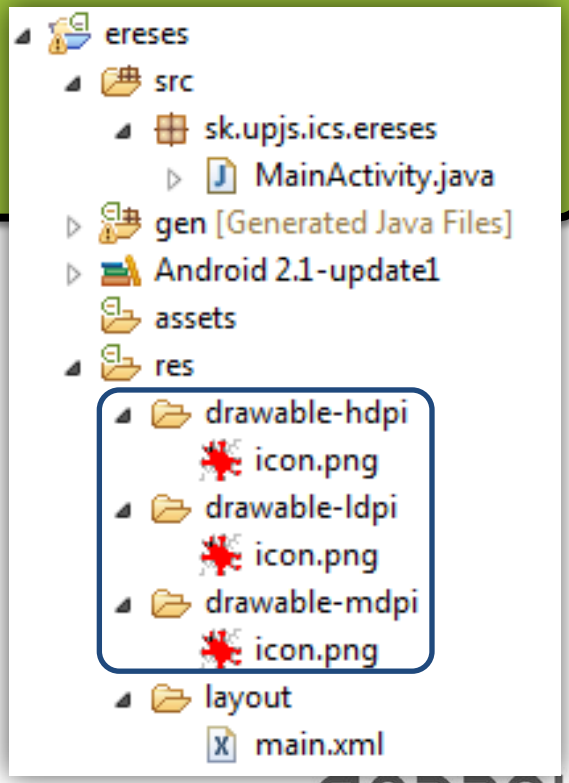
- @string/app_name: odkaz do kľúča app_name v strings.xml

Ostatné prostriedky v manifeste

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="sk.upjs.ics.ereses"
      android:versionCode="1" android:versionName="1.0">

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".MainActivity"
                  android:label="@string/app_name">

...
```



- @drawable/icon: odkaz na súbor v adresári res/drawable
- súbory sú stavané na rozličné rozlíšenia a DPI obrazovky

Komponenty aplikácie: aktivity

- reprezentuje jednu „obrazovku" s používateľským rozhraním
- v kóde: objekt, ktorého trieda dedí od **Activity**
- **súčasný stav**: v danej chvíli je aktívna len jediná aktivita
  - Android do verzie 3.0 nepodporuje viacero okien na obrazovke

Manifest: AndroidManifest.xml

```
package sk.upjs.ics.ereses;

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

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

spustené po prvom vytvorení aktivity

- aktivita reaguje na udalosti životného cyklu
- prekrýva príslušné metódy a vykonáva kód

Ako definovať layout aktivít?

- zariadenia môžu mať pestrú paletu displejov
  - rozličné rozlíšenia
  - rozličné DPI
  - rozličná orientácia
- osvedčili sa layout managery
  - rozličný spôsob, ako dynamicky ukladať komponenty na stránku
- deklarácia pomocou XML!

Manifest: layout/main.xml

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

LinearLayout
(vertikálny)

komponenty ukladá pod seba

textové poličko

- definujeme škatule obsahujúce škatule
- layout špecifikuje konkrétny spôsob ukladania komponentov

Použitie layoutu v aktivite

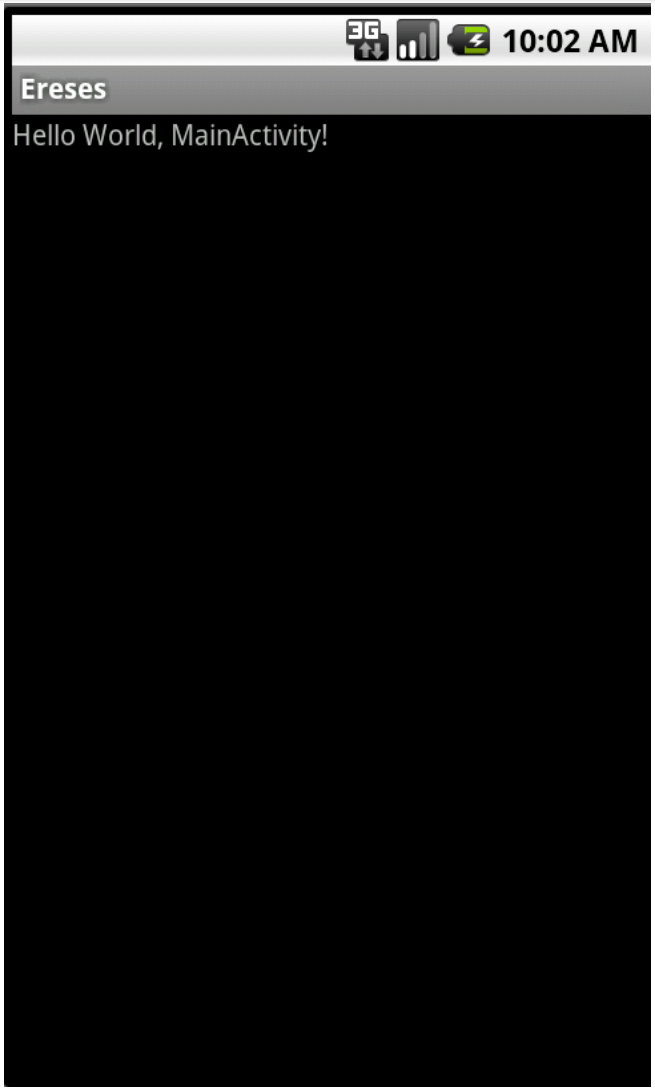
```
package sk.upjs.ics.ereses;

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

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

- layout aktivity sa prevezme z **layout/main.xml**

Výsledok snaženia, verzia 0.0.1



Vylepšenie: dodajme Button

```
<Button android:layout_width="fill_parent"
        android:id="@+id/button"
        android:layout_height="wrap_content"
        android:text="@string/button_title"
        android:layout_weight="0"/>
```

layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="button_title">Reload</string>
</resources>
```

strings.xml

- komponent označme idenfikátorom
- vieme sa naň odkázať v kóde

Dodajme button a aktivizujme ho

```
Button button = (Button) findViewById(R.id.listView);
button.setOnClickListener(new OnClickListener() {
  @Override
  public void onClick(View v) {
    Toast.makeText(MainActivity.this,
                   "Click!",
                   Toast.LENGTH_LONG).show();
  }
});
```

- findViewById(): nachádza komponent z layout.xml podľa ID
- každému komponentu prislúcha klasická trieda

Dodajme button a aktivizujme ho

```
Button button = (Button) findViewById(R.id.listView);
button.setOnClickListener(new OnClickListener() {
  @Override
  public void onClick(View v) {
    Toast.makeText(MainActivity.this,
                   "Click!",
                   Toast.LENGTH_LONG).show();
  }
});
```

toast – krátky oznam používateľovi

- GUI framework je udalostne orientovaný
- klasické filozofie známe zo Swingu
- tlačidlu priradíme poslucháča, ktorý bude zavolaný v prípade vyvolania udalosti

Životný cyklus aktivity alebo kde je Exit?

- všetky komponenty aplikácie bežia v 1 procese
- každá aplikácia beží pod **jedinečným používateľom** Linuxu
  - práva sa nastavia tak, aby súbory aplikácie videla len ona
  - **sandbox** = bezpečnosť
- každý proces má svoju **vlastnú inštanciu VM**
- **proces = kontajner** pre aplikácie
- aplikácia môže bežať raz v jednom, raz v druhom procese
- proces a aplikácia existujú nezávisle od seba!

Životný cyklus aktivity alebo kde je Exit?

- aktivita, ktorá nie je bežiaca, môže byť kedykoľvek odstrelená
- odstrelenie sa vykonáva vo chvíli, keď je nedostatok voľnej RAM
- Android môže tiež odstreľovať celé procesy

Životný cyklus aktivity alebo kde je Exit?

- aktivity sú radené do zásobníka
- novospustená aktivita sa objaví na vrchole zásobníka
- tlačidlom **Back** vyvolávame predošlú aktivitu zo zásobníka
- aktivity na spodku zásobníka môžu byť odstrelené

Exit je úplne zbytočný

Prečo nekupovať lacné telefóny?

- lacný telefón **nemá multitasking**
- **málo RAM** znamená, že **neostane miesto** pre aktivity v pozadí
- nutné minimum: 256 MB
- typický use-case na lacnom telefóne:
  - prehliadam web
  - niekto mi zatelefonuje
  - minie sa RAM
  - Android odstrelí prehliadač
  - po dotelefonovaní sa aktivita reštartne a stránka sa načíta znova

Čo s hotovou aplikáciou?

- zabaliť!
  - aplikácie sa distribuujú vo formáte APK
- podpísať!
  - kvôli bezpečnosti, na testovanie možno použiť aj self-signed certifikát
- vyhodiť na **Android Market**
  - buď ako free alebo ako platenú

Otázky?