



# Snívajú používatelia o mobilných androidoch?

platforma pre mobilné aplikácie

Róbert Novotný  
robert.novotny@upjs.sk  
17. 2. 2014

# Prečo mobilné aplikácie?

dotykový displej  
+ telefón  
+ fotoaparát  
+ GPS

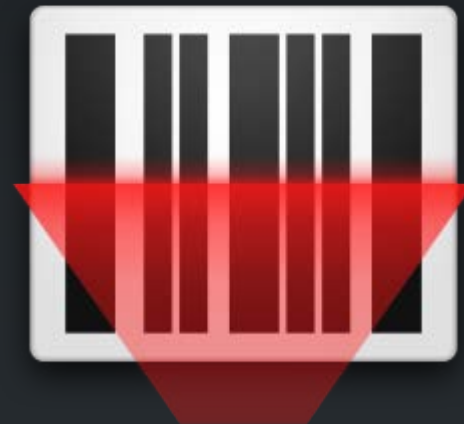
+ lacný a rýchly mobilný net  
+ veľký výkon



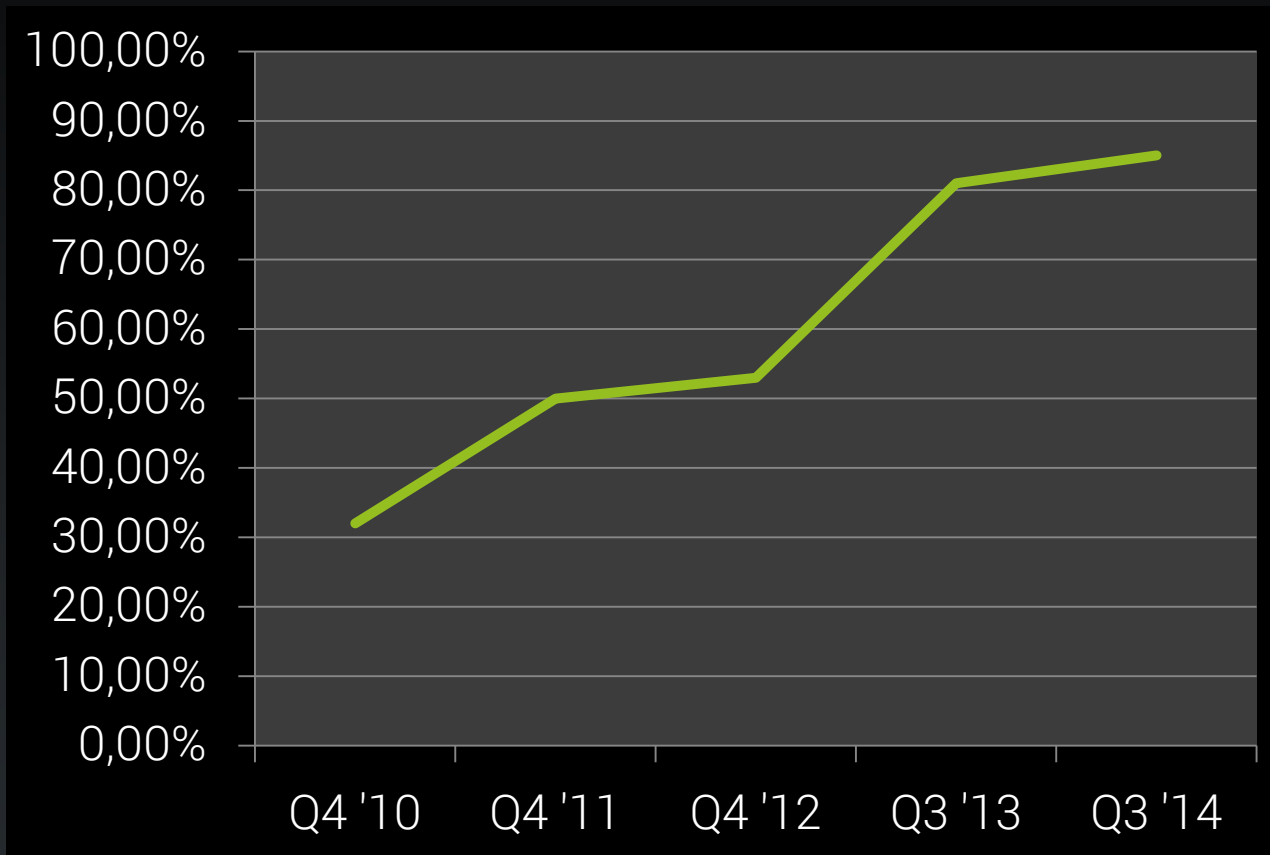


1,4 miliárd používateľov smartfonov

# Explózia nápadov



# Marketingový podiel [dáta IDC]



Apple: 12%, Windows Phone: 3%,  
Blackberry: 0,5%

# Stručná história Androidu

- november 2007: Open Handset Alliance
  - operátori + výrobcovia HW + SW spoločnosti + marketéri
- september 2008: Android 1.0:
  - HTC Dream
- november 2014: Android 5.0



19000 odlišných zariadení

# Prečo je Android úspešný?

- **otvorenosť**
  - otvorená platforma: Linux + Java
  - open-source
- **množstvo zariadení**
  - vyše výrobcov
  - pestrá paleta hardvérových možností
- **ľahký vývoj**
  - súvis s otvorenou platformou
  - rozumné nástroje
- **trh aplikácií**
  - Android Market (október 2012: 700k aplikácií)





# Bežné zariadenie: Samsung Galaxy S3 [240€]

- CPU: 1.4 GHz Quad-Core Cortex-A9  
– architektúra ARM (RISC, 32-bit)
- RAM: 1 GB
- Úložisko: 16 GB + microSD
- Displej: 4.8", 720 x 1280 px
- fotoaparát 8 Mpx, GPS, Bluetooth, WiFi, MicroUSB, FM rádio, NFC...



# Čo treba na vývoj?



bezplatné!

- multiplatformné
  - Windows, Linux, MacOS
- bezplatné
  - publikovanie na Google Play: jednorazových 25€
  - nie je nutné pri vzdelávaní

- syntax na úrovni **JDK 1.5 [2005]**
  - KitKat+: **JDK 7**
- dostupné takmer všetky známe triedy
- v mobile sa spúšťa na **virtuálnom stroji**
- v testoch sa spúšťa v **emulátore**

- bežná Java aplikácia: na Java VM
- Android aplikácia: na [Dalvik](#) / na ARTe
- virtuálny stroj optimalizovaný
  - slabšie CPU
  - menej RAM
  - obmedzená [batéria](#)
  - úplne iná architektúra než Java VM

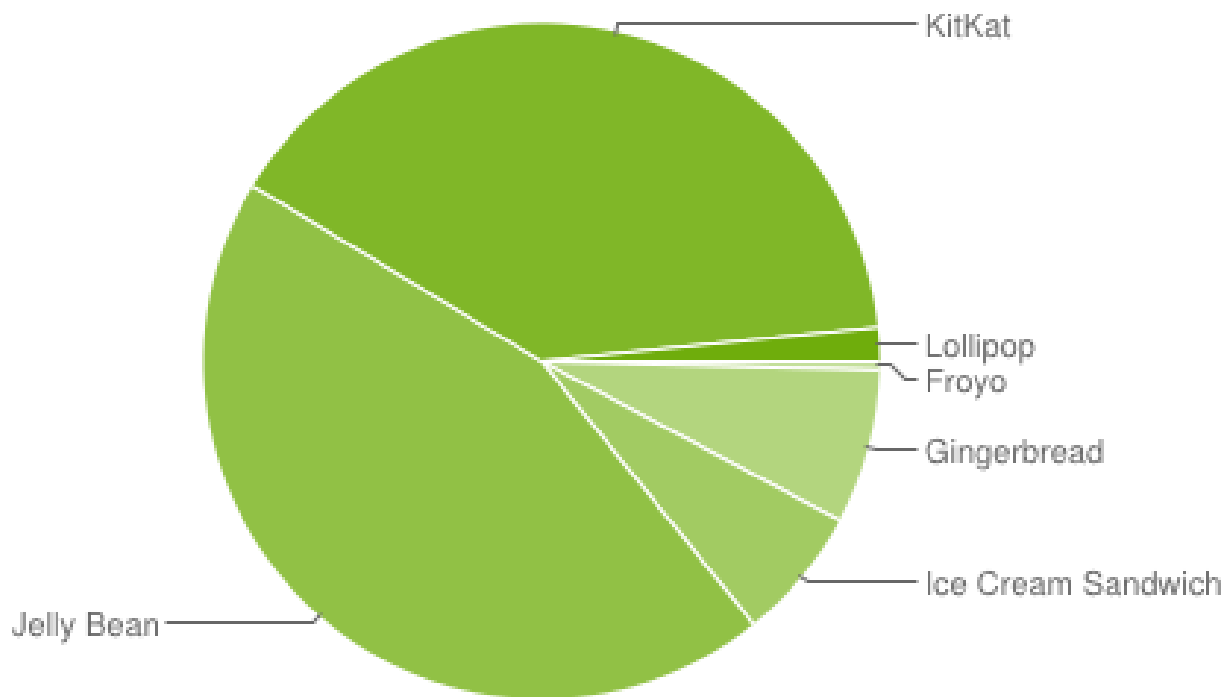
# Evolúcia platformy

- s každou verziou Androidu bohatšia platforma
  - 2.0: multi-touch
  - 2.2: Flash, WiFi hotspot/tethering
  - 2.3: Clipboard
  - ...
  - 4.4: krokover, infračervené ovládanie...
  - 5.0: nový dizajn, optimalizácia...
- vývojár sa rozhodne
  - ktorú minimálnu verziu bude podporovať
  - voči ktorej verzii platformy bude kompilovať

- nezávislé **platformy** v rámci SDK
  - podobne ako existujú verzie Java Development Kitov
  - obvykle vyvíjame pre jednu konkrétnu platformu
- **garantujú** dostupný hardvér
  - Android 2.3:
    - nutné: 2 MPx fotoaparát
    - predpokladaný: autofokus
    - voliteľné: pevné ohnisko, blesk

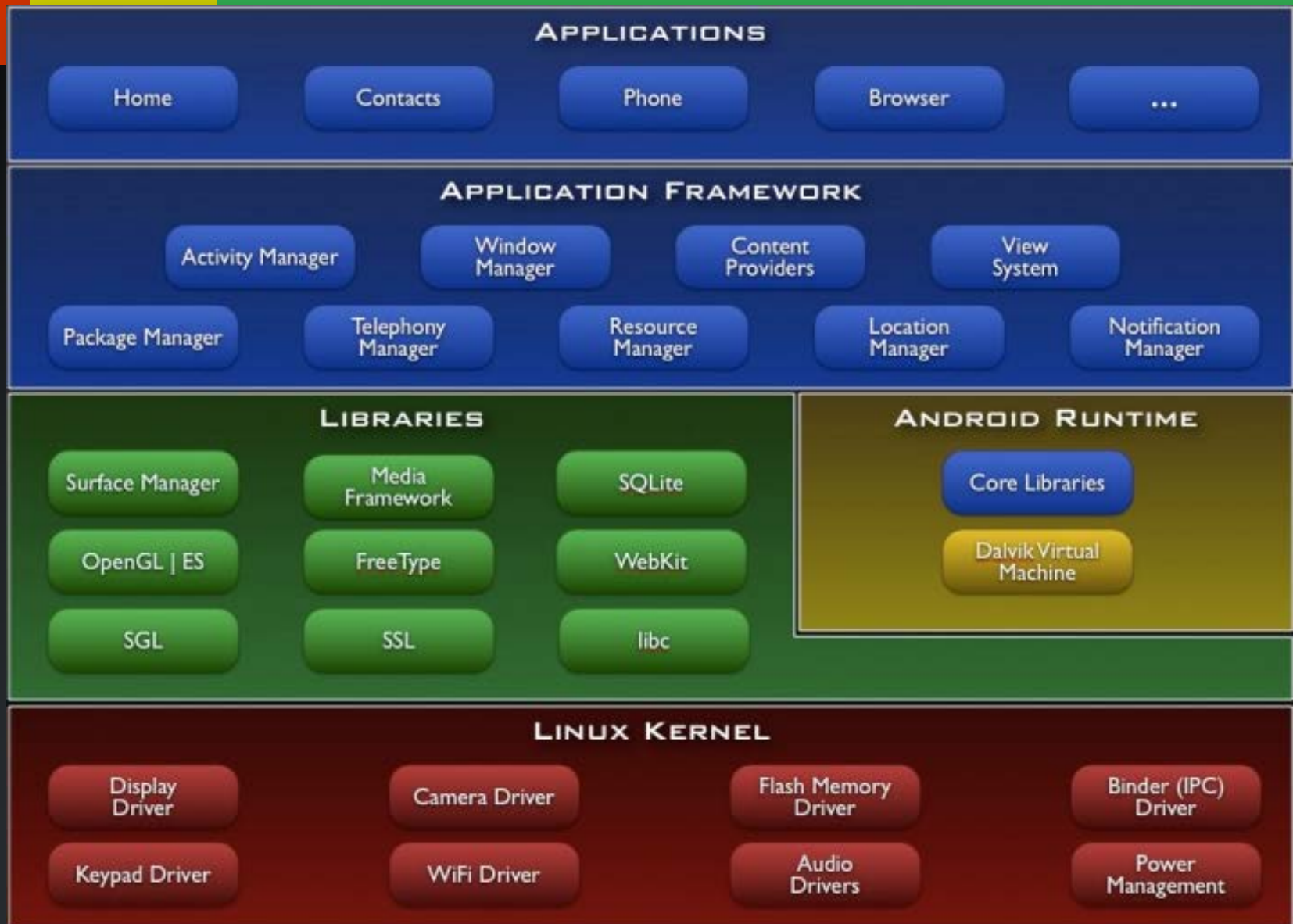
# Kompromis: novinky vs kompatibilita

- vývojár sa rozhodne
  - ktorú minimálnu verziu bude podporovať
  - voči ktorej verzii platformy bude kompilovať





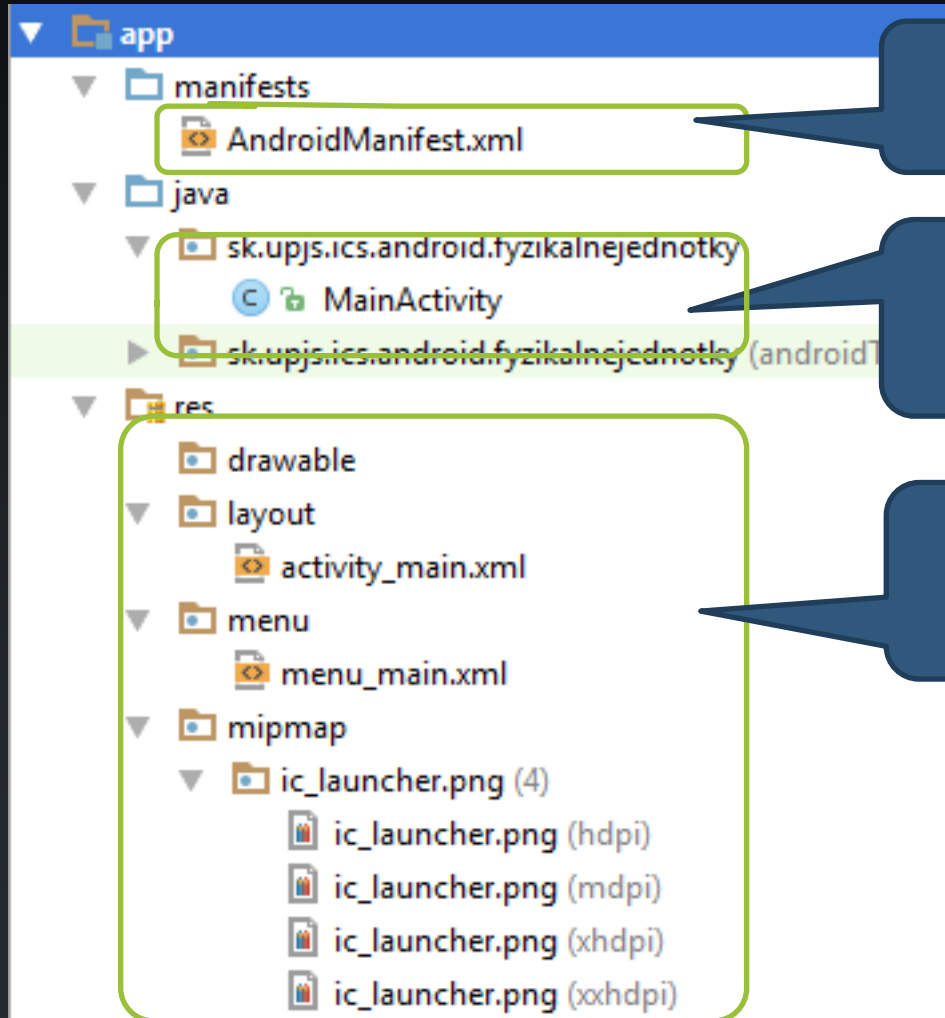
# Pohľad do útrob Androida



# Vytvárame nový projekt

1. stiahneme Android Studio
2. nainštalujeme
3. vytvoríme nový projekt

# Čo vypadne z generátora



manifest

Java zdrojáky

resources  
(ikony, obrázky, textové reťazce)

# 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



# 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">
    ...
```

referencia  
položky v  
strings.xml

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

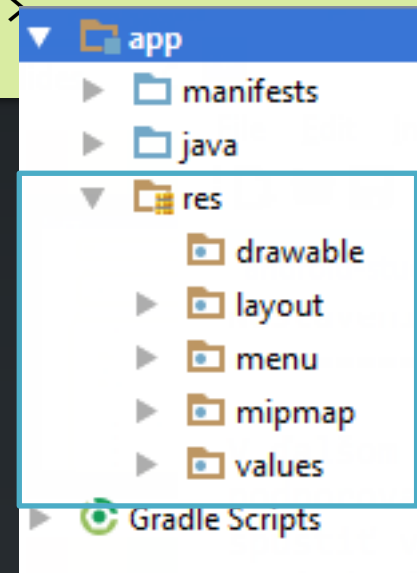
strings.xml

# 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">

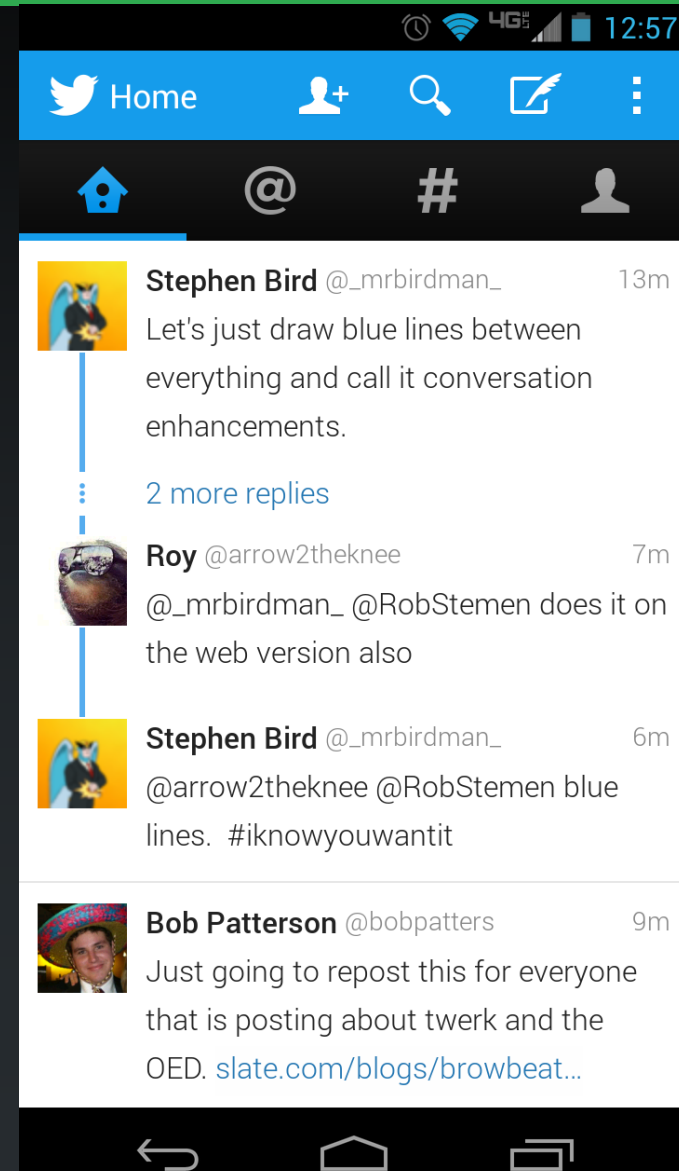
    <application android:icon="@drawable/icon"
        android:label="@string/app_name">
        <activity android:name=".MainActivity"
            android:label="@string/app_name">
        ...
```

- **@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**





# Manifest: AndroidManifest.xml

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

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

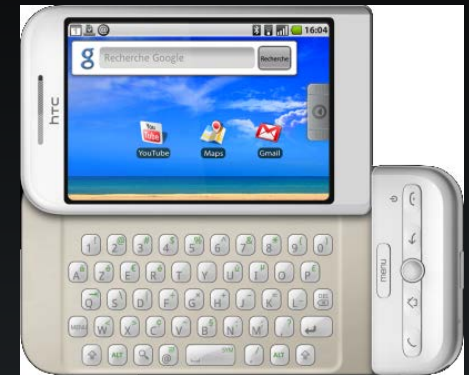
public class MainActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

spustené po prvom  
vytvorení aktivity

- aktivita reaguje na udalosti **životného cyklu**
- pokrýva 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

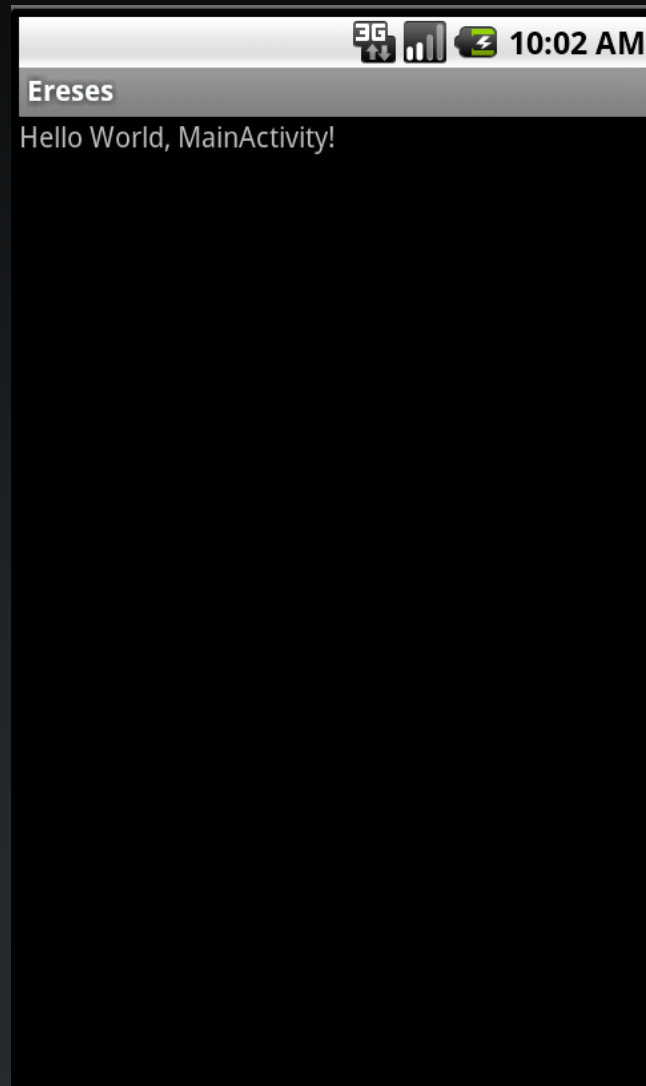
```
package sk.upjs.ics.ereses;

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 identifikátorom
- 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



# Čo s hotovou aplikáciou?

- zabalit!
  - aplikácie sa distribuujú vo formáte APK
- podpísať!
  - kvôli bezpečnosti, na testovanie možno použiť aj self-signed certifikát
- vyhodit' na **Google Play**
  - buď ako free alebo ako platenú



Otázky?

