Marek Chrenko & Tomáš Vondráček

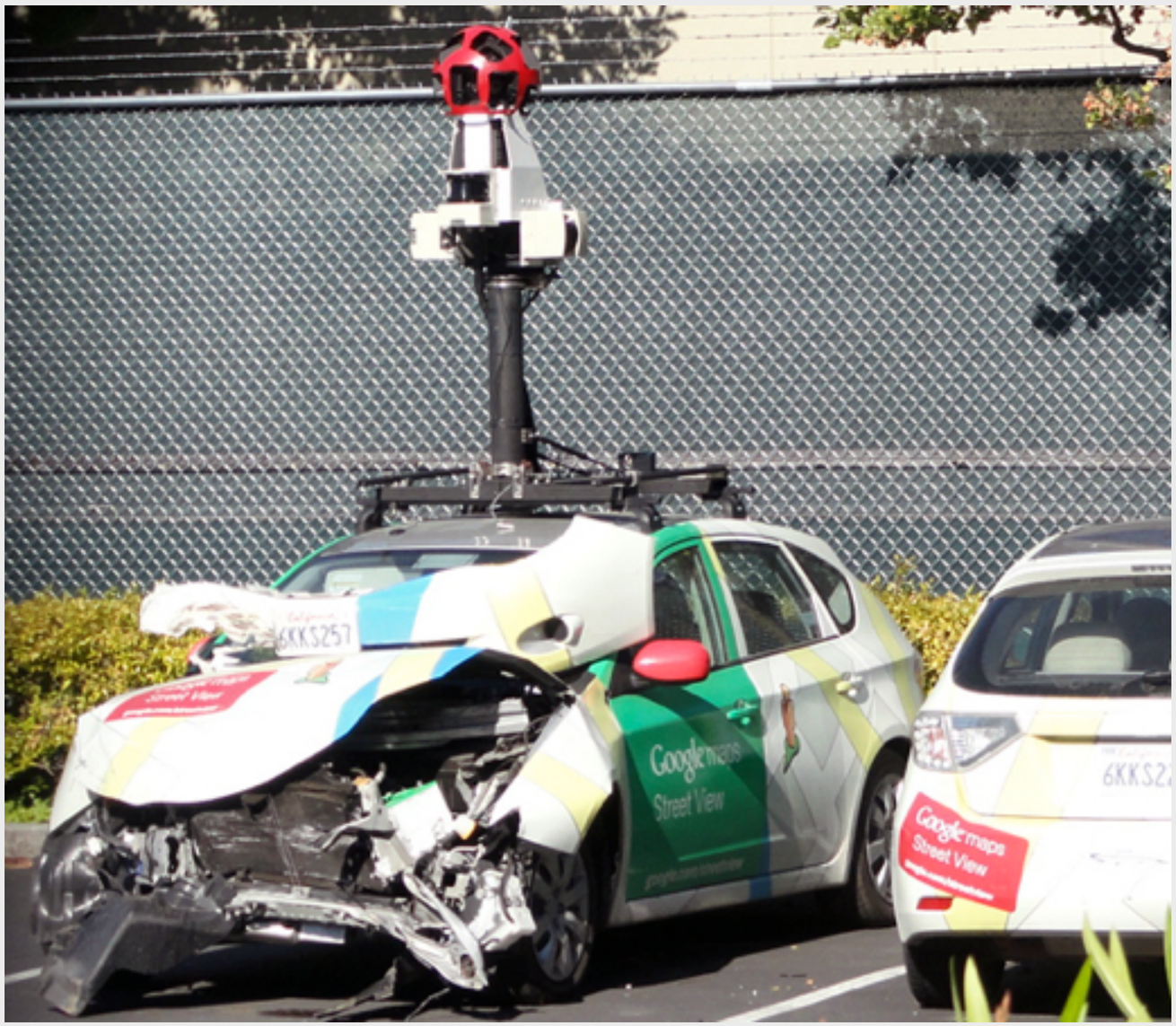
Představení

- Marek Chrenko & Tomáš Vondráček
- Android developers ve společnosti Inmite.

Kam smeruje mobilný svet?

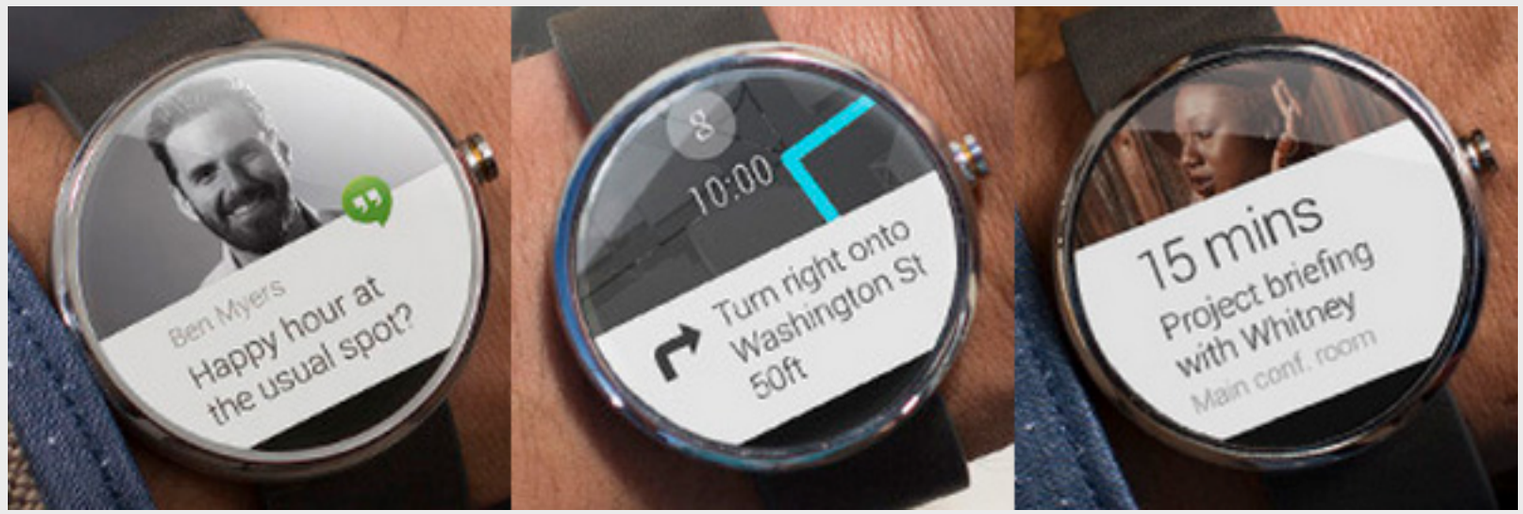
Internet of Things

- inteligentné domy
- autá bez vodiča
- RFID náramky



Kontextové aplikácie

- dopĺňajú realitu
- silné využitie senzorov
- špecifické UI (napr. hlasová navigácia)
- jednoducho dostupné



Wearables

- smartfón? such oldschool
- hodinky, náramky, oculus rift
- airbond od Inmite
- Google Glass

Google Glass

Google Glass

- snažia sa určiť smer vo wearables
- aktuálne developer preview
- OS Android 4.x
- spolupracujú s Android a iOS zariadeniami

Technické okienko

- CPU 1.2 GHz, dual-core
- RAM 1 GB
- 16 GB Flash
- displej 640x360
- zvuk skrz kosti pri uchu
- 5MPix fotoaparát s HD videom
- wifi, bluetooth, microUSB

Používanie

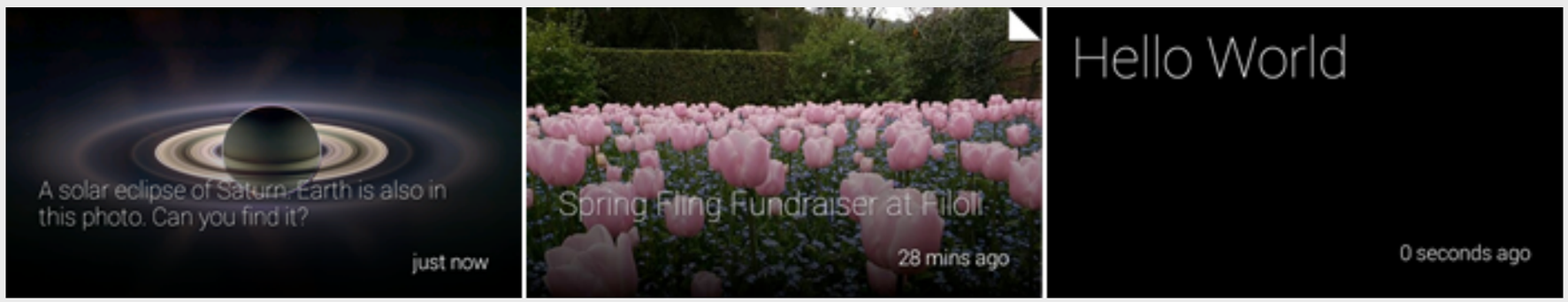
- demo hlasového vstupu a pohybu v systéme

Vývoj

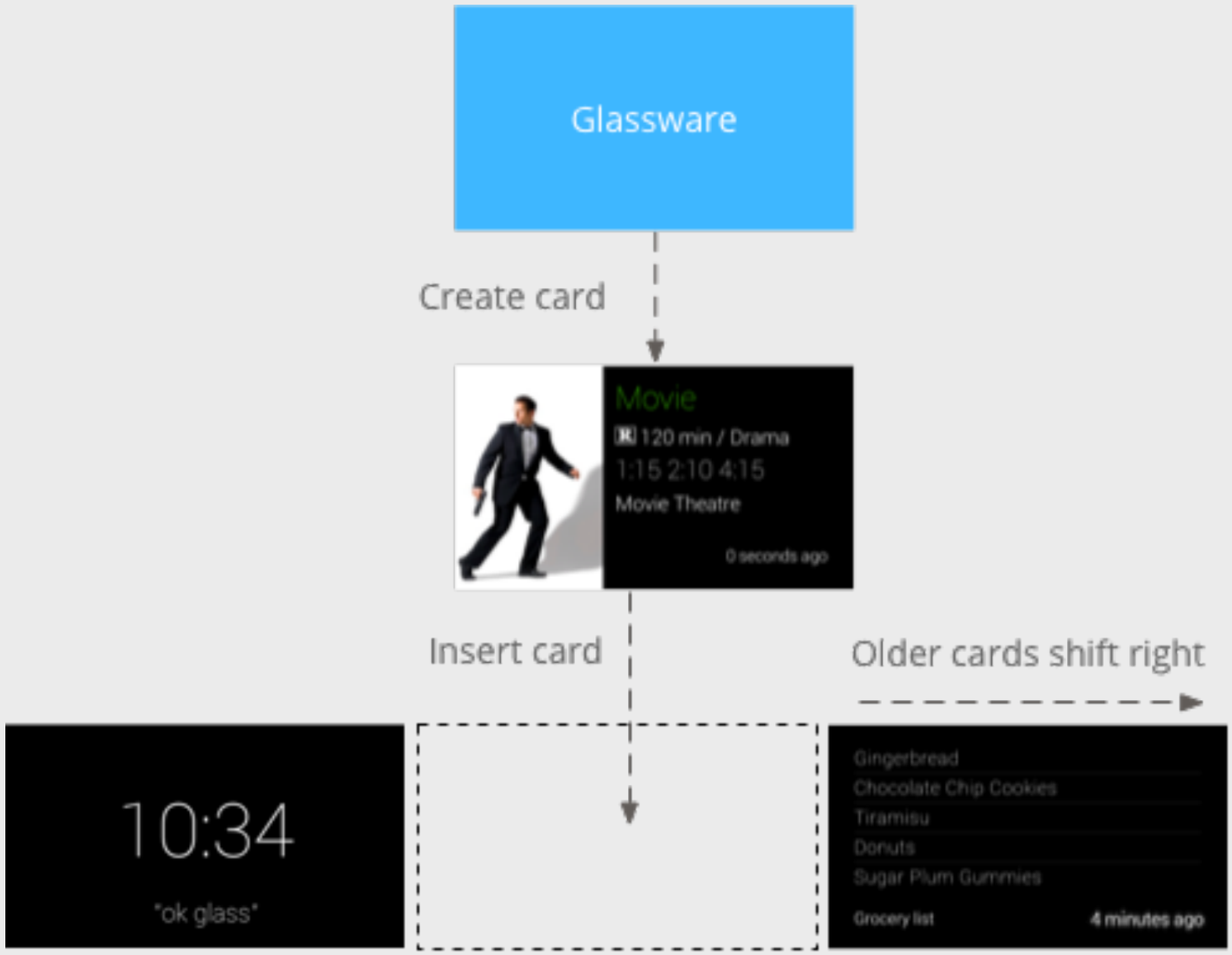
- Android SDK od API 15
- Mirror API (REST)
- natívne GDK

Mirror API

- REST prístup, Glass API backend
- komunikácia cez JSON
- karty sú hlavná UI zložka
- všetko ide do timeline
- demo



Statické karty



Statické karty

```
POST /mirror/v1/timeline HTTP/1.1
Host: www.googleapis.com
Authorization: Bearer {auth token}
Content-Type: application/json
Content-Length: 26
```

```
{ "text": "Hello world" }
```

Mirror API toho vie viac

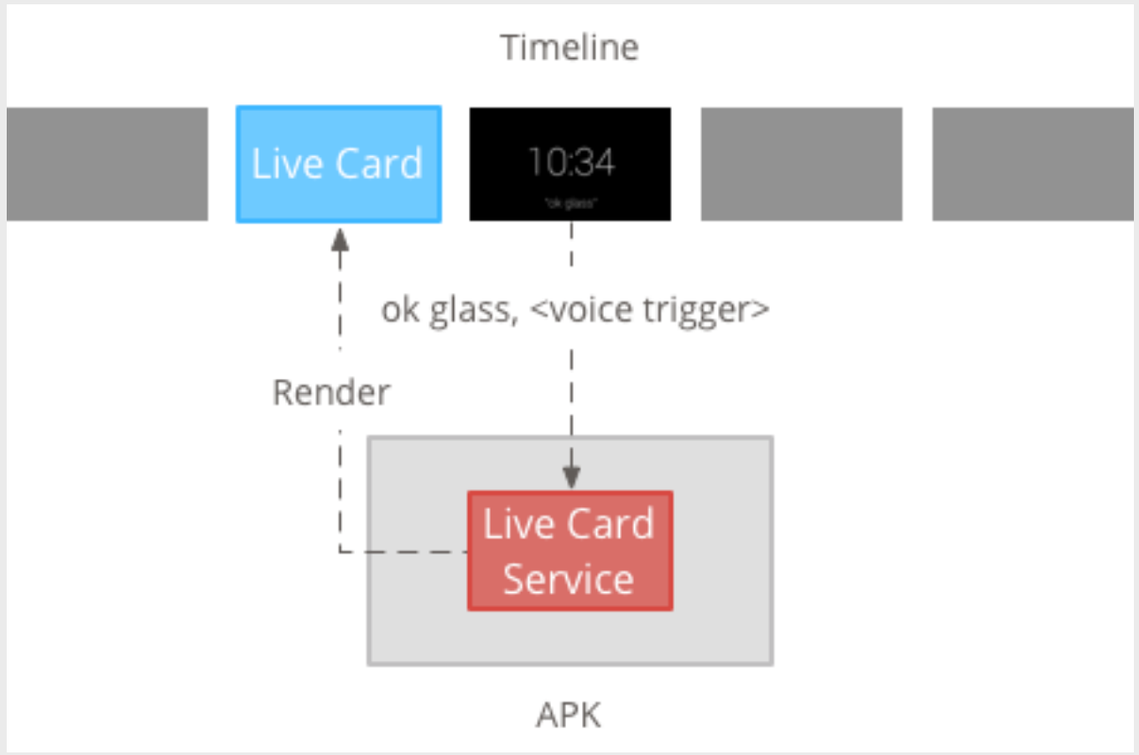
- vytvárať menu items
- zdieľať karty
- interagovať s kartami
- bundlovať karty
- je Google cloud based

Glass Development Kit

- beží priamo na Glass HW
- od API level 19
- rozširuje Android SDK o Cards, Voice (triggers), Gesture Detector
- interakcia s používateľom v reálnom čase
- funguje offline
- prístup k HW

Aplikačné prvky GDK

- live cards
- immersions
- UI Widgety a preddefinované štýly
- prístup k senzorom



Demo

- trochu bankovej teórie
- a viac hernej zábavy
- viac info - David Vávra / destil.cz

Android UI

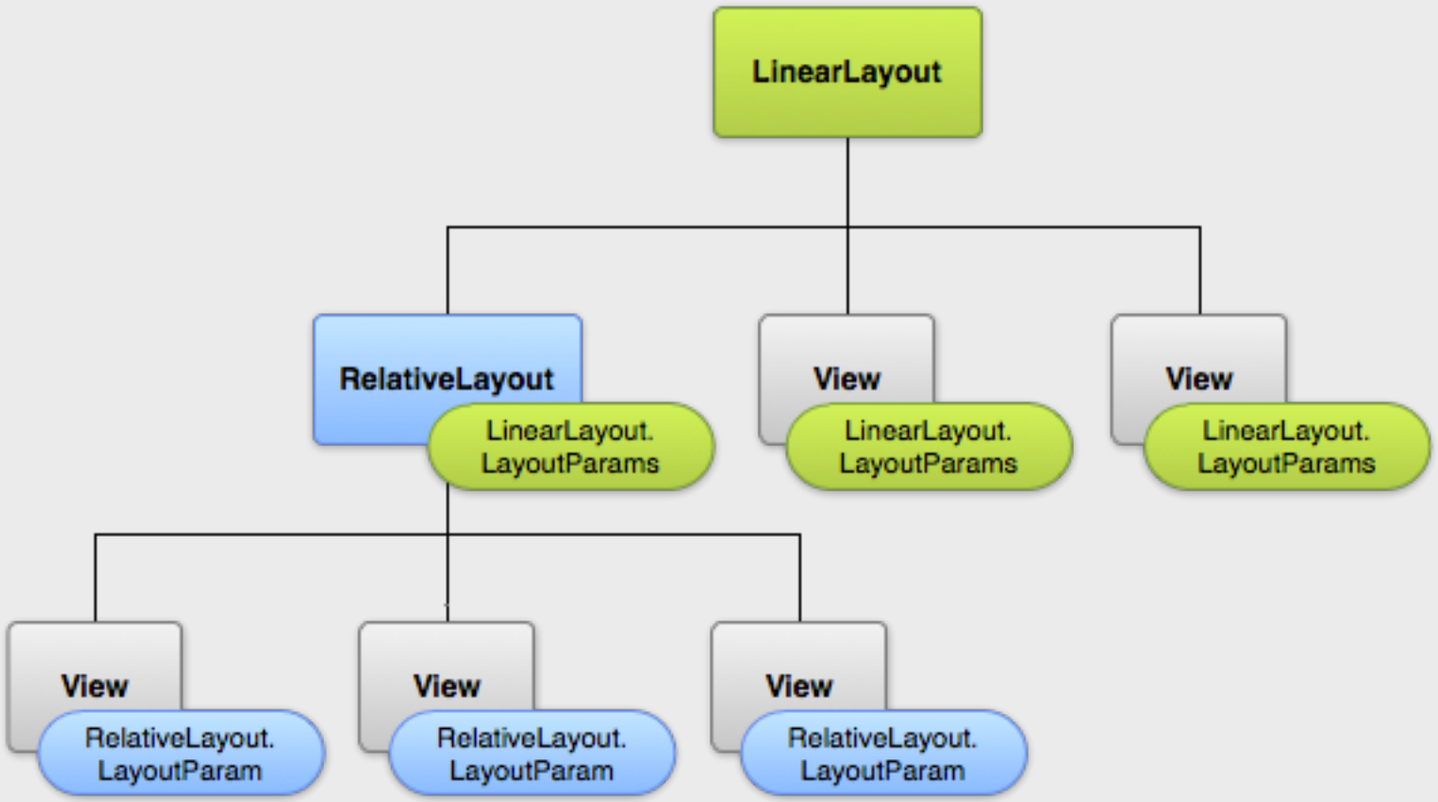
25.4.2014

Marek Chrenko & Tomáš Vondráček

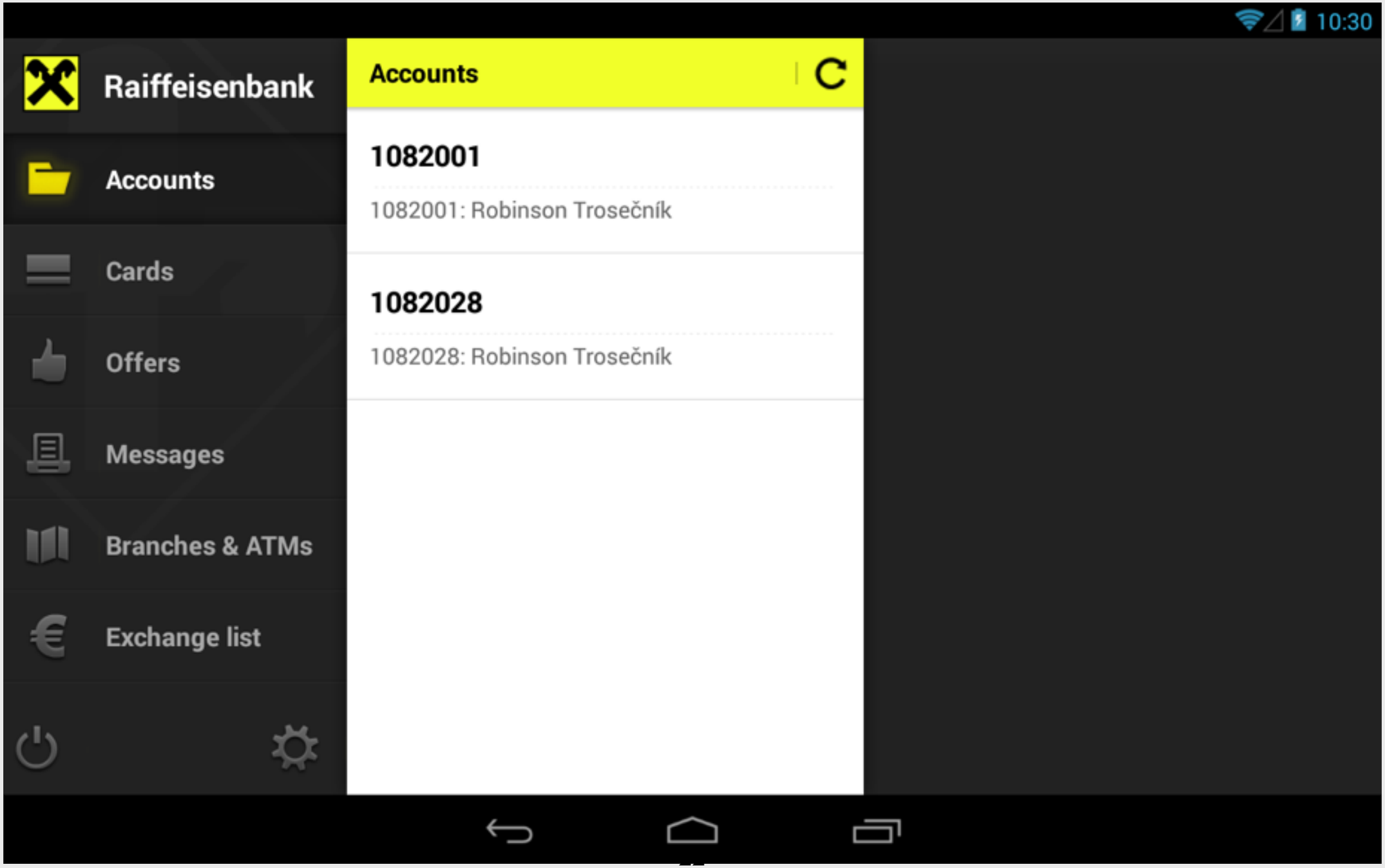
Agenda

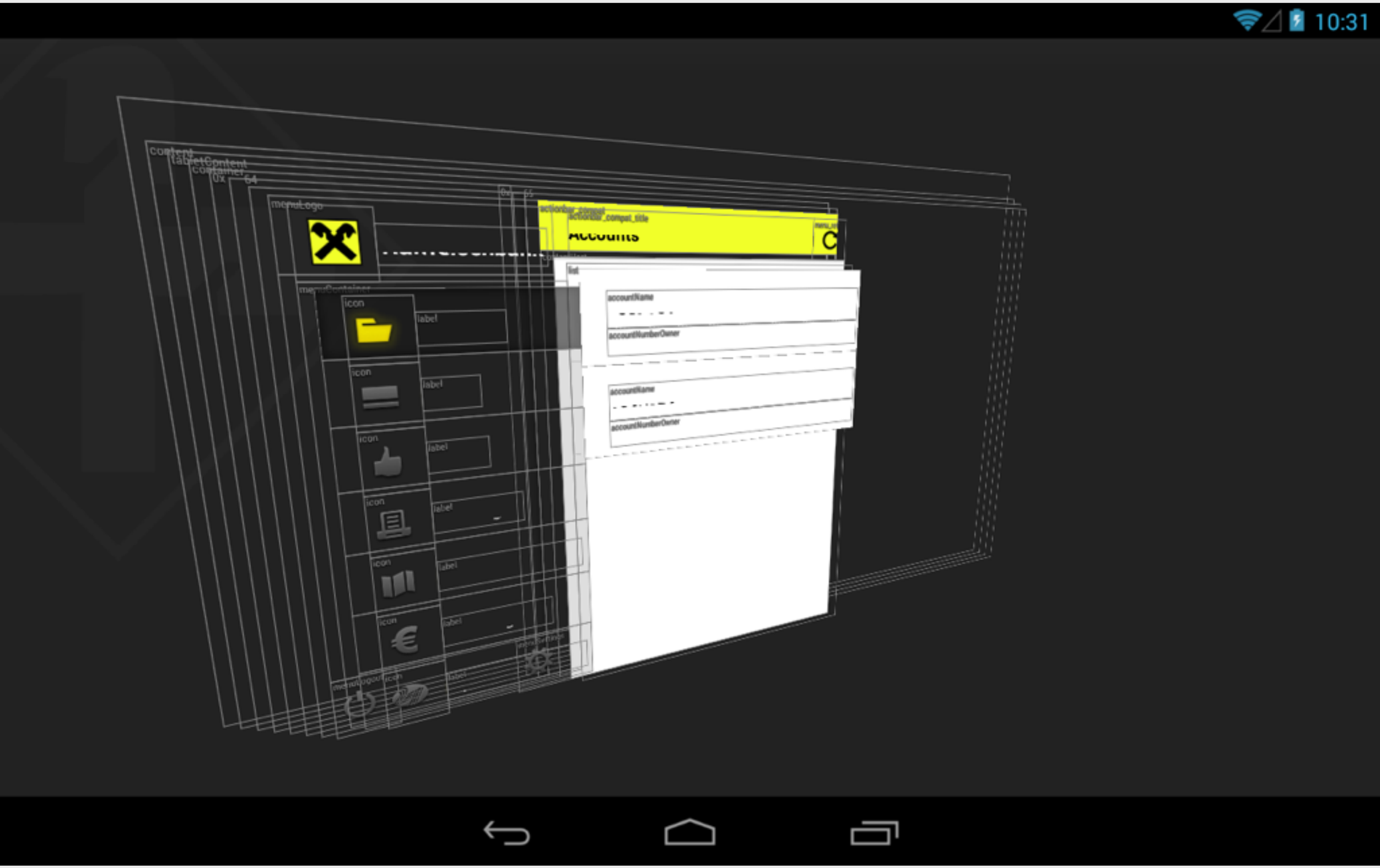
- View hierarchy
- Layouts
- ListView
- Canvas
- Custom View
- Animace
- Styly

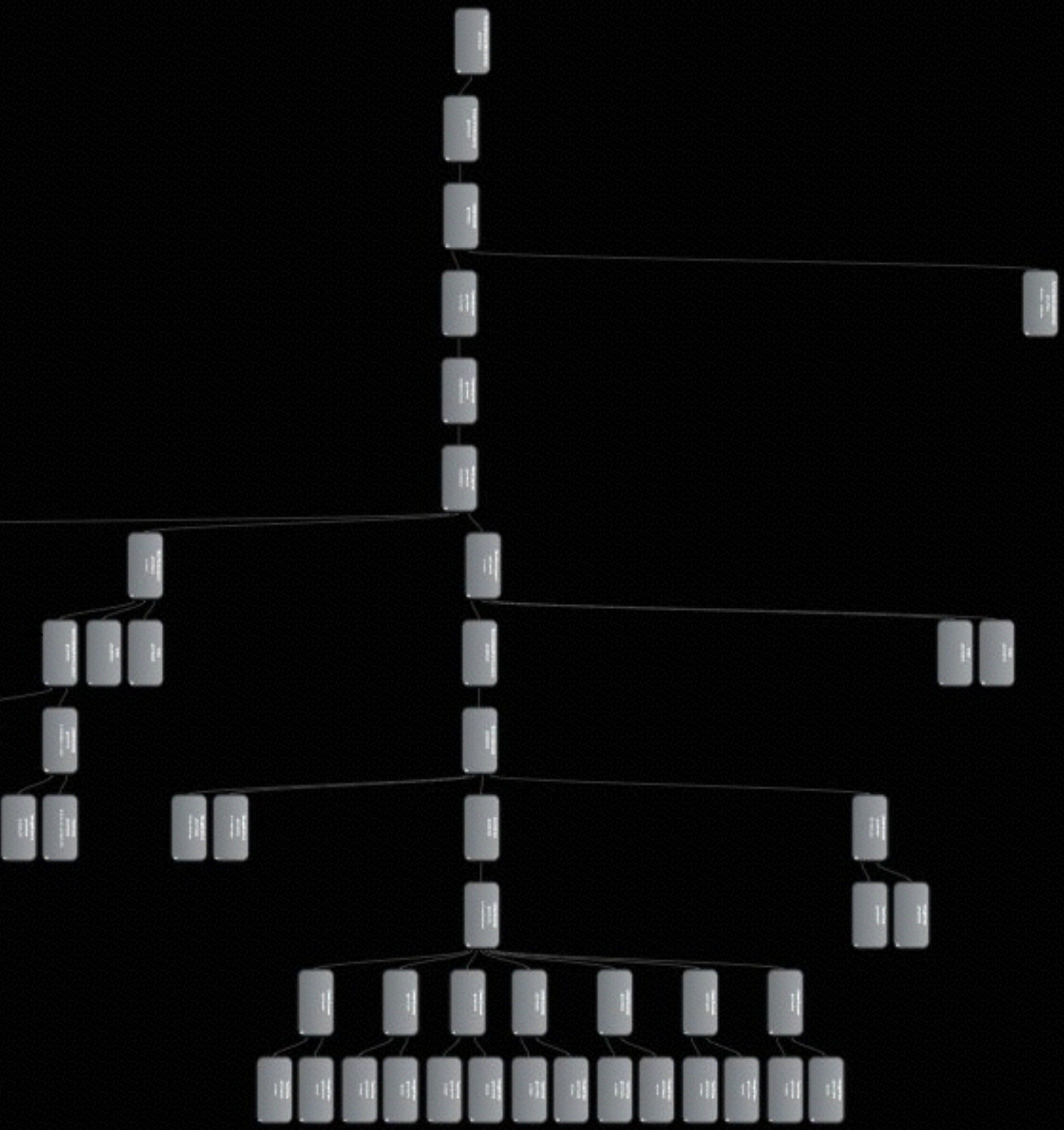
View hierarchy



View hierarchy




10:31
content
tabletContent
container
0/x
64
menuLogo
actionbar_compat_title
Accounts
list
accountName
accountNumberOwner
accountName
accountNumberOwner
icon
label
icon
label
icon
label
icon
label
icon
label
menuLogout
icon
label



View

- základní stavební blok
- View vs ViewGroup
  - každé View má layoutParams pro svého parenta

Layouts (ViewGroup)

- Definují strukturu UI
- Mohou se vnořovat, ale čím hlubší hierarchie, tím pomalejší měření a layout

Views

- v xml

```
<TextView android:id="@+id/title"
    android:layout_width=“match_parent”
    android:layout_height="wrap_content"
    android:text="@string/text" />
```

- v kódu

```
TextView txt = new TextView(context);
txt.setText(R.string.text)
txt.setLayoutParams(new LayoutParams(
LayoutParams.MATCH_PARENT,
LayoutParams.WRAP_CONTENT));
```

Views

- view vytvořené v xml se "nahustí" do hierarchie

```
LayoutInflater inflater = LayoutInflater.from(ctx)
View v = inflater.inflate(R.layout.fragment_commits,
container, false);
```

Views

- na jednotlivá view se přistupuje přes ID

```
TextView title = view.findViewById(R.id.title);
```

View hierarchy

- perform traversal (asynchronní)
- measure
- layout
- draw

Layouts

- Základní layouty z frameworku:
  - FrameLayout, LinearLayout, RelativeLayout, TableLayout, GridLayout
- Vlastní layout
- AdapterView
  - ListView, GridView

FrameLayout

- Nejjednodušší
- Nejefektivnější

- layout_margin
- layout_gravity

FrameLayout

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ListView
        android:id="@android:id/list"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <TextView android:id="@android:id/empty"
        android:text="@string/loading"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"/>

</FrameLayout>
```

LinearLayout

- Horizontální nebo vertikální
- layout_weight - potomci mohou mít váhu
- layout_gravity
- layout_margin

LinearLayout

LinearLayout

```
<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <TextView
        android:id="@+id/commit_author"
        android:layout_width="0dp"
        android:layout_weight="1"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceSmall"
        tools:text="Commit author" />
    <TextView
        android:id="@+id/commit_email"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceSmall"
        tools:text="Commit email" />
</LinearLayout>
```

RelativeLayout

- Jednotlivá View jsou pozicována vůči sobě nebo vůči předkovi
- Umožňuje vytvořit plochou hierarchii

RelativeLayout

RelativeLayout

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView android:id=“@+id/commit_img"
        android:layout_width=“match_parent"
        android:layout_height=“wrap_content" />

    <TextView android:id="@+id/commit_title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/commit_img"/>
    <TextView android:id="@+id/commit_author"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toRightOf=“@+id/commit_email"
        android:layout_alignParentRight=“true”
        android:layout_below=“@+id/commit_img />
</RelativeLayout>
```

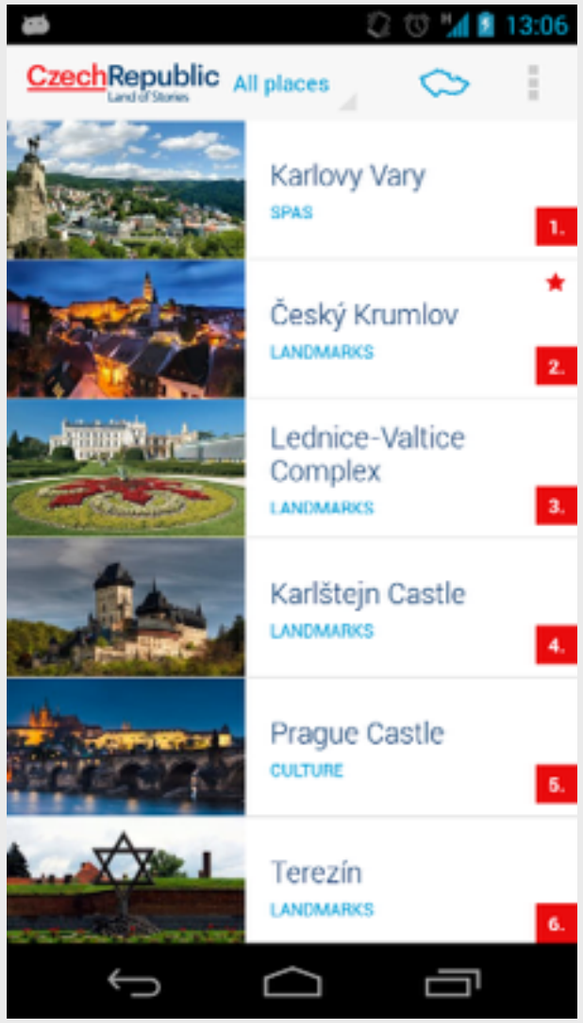
ListView & GridView

- vytvářejí kolekci View na základě dat z adaptéru
- recyklace View
  - vždy je vytvořeno pouze tolik view kolik je zobrazeno (+ 1 či 2)

ListView & GridView

ListView


13:15
bene+
SEZNAM
MAPA
Odměna 2 % při platbě kreditní kartou
11.3 km
Odměna až 3 % při platbě kartou
1.16 km
Sleva 50 Kč na balení Magne B6 Forte tablety. Doplněk stravy
44.0 m
22 Dní
25 % sleva na FERPLAST akvarijní set Capri 60
703 m
21 Dní
Řadit podle
Popularity


13:06
CzechRepublic
Land of Stories
All places
Karlovy Vary
SPAS
1.
Český Krumlov
LANDMARKS
2.
Lednice-Valtice Complex
LANDMARKS
3.
Karlštejn Castle
LANDMARKS
4.
Prague Castle
CULTURE
5.
Terezín
LANDMARKS
6.


1:30
air bank
Běžné účty
288 680,36 CZK
Běžný účet CZK
1022995074/3030
288 680,36 CZK
289 655,36 CZK
Spořicí účty
309 824,52 CZK
Spořicí účet CZK
1022995031/3030
309 824,52 CZK
309 824,52 CZK
Půjčky
Ihned k dispozici
30 000 CZK
Rozpracovaná žádost
30 000 CZK
Aktuálně čerpáno
29 672 CZK

ListView

- nasledovné postupy platia aj pre GridView
- zobrazujú veľké počty záznamov
- náročné na CPU/pamať - pri akciách ako scroll a fling
- použitie adaptérov

ListView

- view sú ťažké objekty (1-2 kb)
- recyklujú sa
- naplnenie view obsahom sa deje v `getView(position, convertView, parent)`

getView so sekaním

```
public View getView(int position, View convertView, ViewGroup parent)
{
  View item = mInflater.inflate(R.layout.list_item_icon_text, null);
  ((TextView) item.findViewById(R.id.text)).setText(DATA[position]);
  ((ImageView) item.findViewById(R.id.icon)).setImageBitmap(
    (position & 1) == 1 ? mIcon1 : mIcon2);
  return item;
}
```

getView klasicky

```
public View getView(int position, View convertView, ViewGroup parent)
{
  if (convertView == null) {
     convertView = mInflater.inflate(R.layout.item, parent, false);
   }

  ((TextView)convertView.findViewById(R.id.text))
  .setText(DATA[position]);

  ((ImageView) convertView.findViewById(R.id.icon))
  .setImageBitmap((position & 1) == 1 ? mIcon1 : mIcon2);

  return convertView;
}
```

getView plynulo

```
public View getView(int position, View convertView, ViewGroup parent)
{
 ViewHolder holder;
 if (convertView == null) {
     convertView = mInflater.
             inflate(R.layout.list_item_icon_text, parent, false);
   holder = new ViewHolder();
   holder.text = (TextView) convertView.findViewById(R.id.text);
   holder.icon = (ImageView) convertView.findViewById(R.id.icon);
   convertView.setTag(holder); }
 else {
   holder = (ViewHolder) convertView.getTag();
 }

 holder.text.setText(DATA[position]);
 holder.icon.setImageBitmap((position & 1) == 1 ? mIcon1 : mIcon2);
 return convertView;
}
```

Adaptér

- nerobiť si lokálnu view cache!
- nesnažiť sa vybabrať s ListView
- nemeniť štruktúru convertView - to sa vracia typu, ktorý určí komponenta a je vždy správny
- notifyDataSetChanged()
- notifyDataSetInvalidated()

Typy View

- vstavané typy (jednoduché texty)
- ak chceme viac typov view, preťažíme `getViewTypeCount()`, `getItemViewType(position)`

Typy View

```
@Override
public int getViewTypeCount() {
   return super.getViewTypeCount() + 1; // activate view type
}
```

```
@Override
public int getItemViewType(int position) {
   if (getItem(position) instanceof CardTO) {
      return VIEW_TYPE_ACTIVATE;
   }
   return super.getItemViewType(position);
}
```

Properties

- enabled / disabled položky
- single / multiple choice mode
- hasStableIds() = = true ako optimalizácia

Header & Footer view

- skrolujú s položkami listview
- Quick Return pattern

Problémy

- čierny list počas skrolovania - nastaviť cacheColorHint na transparentnú farbu
- nevkladať listview do scrollview a naopak
- potrebujeme listview? nestačí layout generovať?

CustomView

- možnosť odlíšenia od davu
- zapúzdrenie kódu
- optimalizácia
- prístup k protected metódam
- demo: ClipboardEditText

Život View

- pridanie / odobranie z window
- meranie, pozicovanie a vykreslenie
- perzistencia stavu

pridanie / odobranie

- po volani ViewGroup.addView
- zavolá sa onAttachedToWindow()
- idealne miesto na nastavenie listenerov, zresetovanie stavu
- pri odoberaní - onDetachFromWindow()
- o svojich zmenách view informuje okolie cez rozhrania a callbacky (nemá onCreate,..)

odmeraj, rozlož, nakresli

- všetky view sa dokážu odmerať (pevné šírky, príp. LayoutParams)
- requestLayout() sa stará o prepočet layoutu rekurzívne, preto treba držať plochý view strom
- onMeasure() rozhoduje o veľkosti view a jeho potomkov - musí volať setMeasuredDimension() pred návratom
- následne sa prevedie layout - paddingy,…

a uchovaj stav

- pri rotácií obrazovky dochádza k zahodeniu views a znovuvytvoreniu UI
- onSaveInstanceState(), onRestoreInstanceState()
- vytvoriť potomka triedy BaseSavedState

Canvas

- přímé kreslení pomocí primitiv
- můžeme přes něj vykreslit obsah View nebo si můžeme kreslit do bitmapy
- umožňuje základní transformace
  - translace, rotace, scale

Canvas - pojmy

- Canvas - malíř, pamatuje si draw calls
- Bitmap - pixely, na které se aplikují draw calls
- Paint - štětec, určuje barvu a styl

Canvas primitiva

- drawText(),
- drawBitmap(),
- drawLine(),
- drawPath()
- drawRect(),
- …

Canvas - stav

- uložení/obnovení stavu pro aplikování transformací

Canvas - stav

```
canvas.save();
canvas.translate(textX, 0);

canvas.drawText(text, x, y, mPaint);
canvas.restore();
```

Canvas - tip

```
Bitmap bmp =
Bitmap.createBitmap(view.getWidth(),
view.getHeight(),
Bitmap.Config.ARGB_8888);

Canvas canvas = new Canvas(bmp);
view.draw(canvas);
```

Canvas - tip

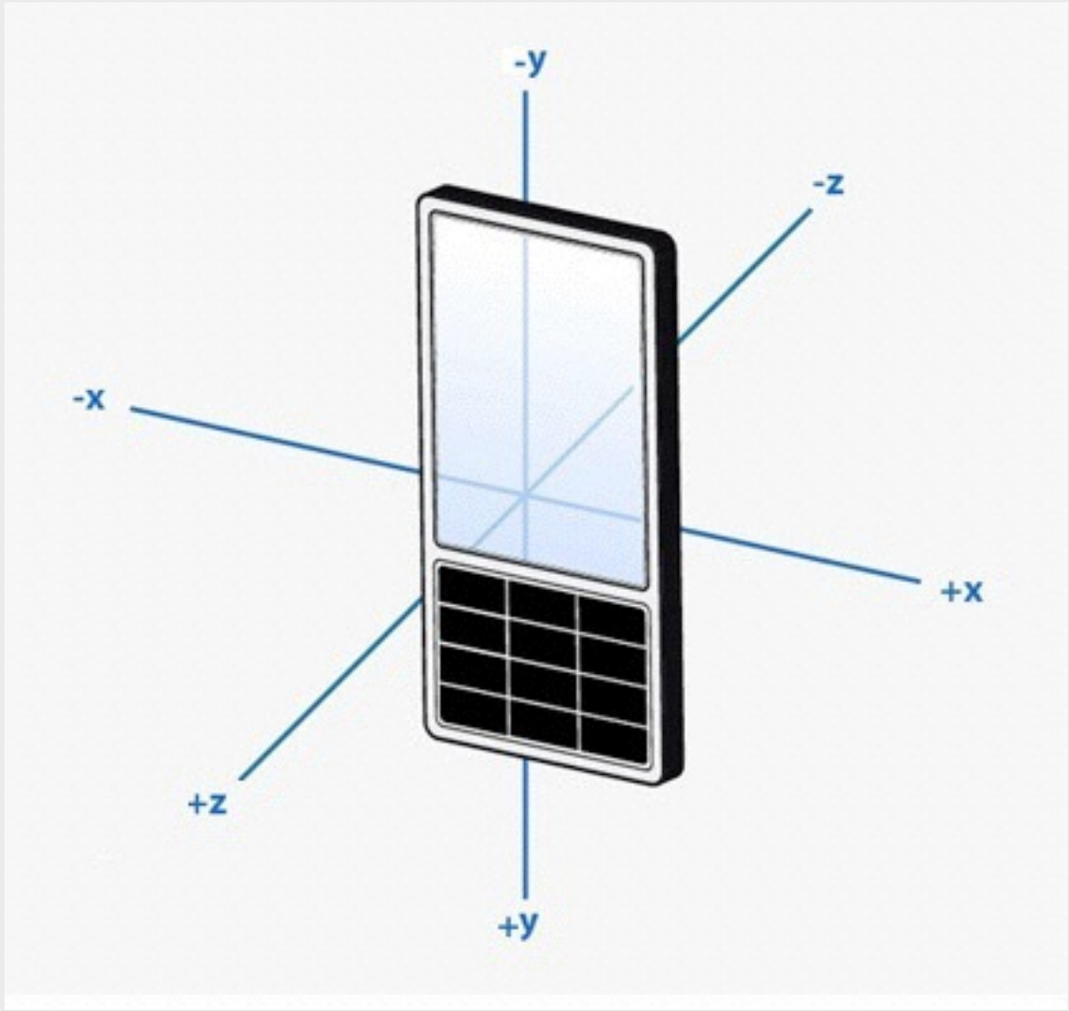
```
Bitmap bmp =
Bitmap.createBitmap(view.getWidth()/4,
view.getHeight()/4, Bitmap.Config.ARGB_8888);

Canvas canvas = new Canvas(bmp);
canvas.setScale(0.25, 0.25)
view.draw(canvas);
```

Animace

- “Anima” = duše

Souřadný systém



Property animation

- Animace čehokoli na jakémkoli objektu.
- Včetně vlastních objektů a vlastních property.

Property animation:

- Animator: tiká
- Interpolator: $y = f(t)$
  - y začíná na 0, končí na 1
- Evaluator: $x_0 + y * (x_1 - x_0)$

Property animation

- ObjectAnimator
- Typicky volá nad view:
  - setRotationX, setRotationY
  - setTranslationX, setTranslationY
  - setScaleX, setScaleY
  - setAlpha

Property animation

```
ObjectAnimator animator =
    ObjectAnimator.ofFloat(view,
    View.ROTATION_X, 0f, 180f);
animator.setDuration(1000L);
animator.start();
```

Property animation

```
view.animate()
        .rotationX(180f)
        .setDuration(1000L);
```

Property animation

```
view.animate()
.rotationX(180f)
.translationY(100f)
.alpha(0.5f)
.scaleY(0f)
.setDuration(1000L);
```

Property animation

```
view.animate()
.rotationX(180f)
.setDuration(1000L)
.withLayer();
```

Tipy a triky

- LayoutTransition
- LayoutAnimationController
- ViewPager Transformer
- Window Animation

Layout Transition

- v xml: android:animateLayoutChanges=”true”
- final LayoutTransition transition = new LayoutTransition();
  view.setLayoutTransition(transition);

Layout Animation Controller

- Animace itemů ListView/GridView.
- Využívá api level 1 animace.

Layout Animation controller

```
Animation anim =
AnimationUtils.loadAnimation(getActivity(),
R.anim.slide_bottom_top);

LayoutAnimationController c =
   new LayoutAnimationController(anim);

getListView().setLayoutAnimation(c);
```

ViewPager Transformer

- Transformace každé stránky podle současné pozice.
- ViewPager.setPageTransformer(...)

Window animation

```
ActivityOptions options =
ActivityOptions.makeScaleUpAnimation(view,
0, 0, view.getWidth(), view.getHeight());

Intent intent = new Intent(getActivity(),
MyActivity.class);
startActivity(intent, options.toBundle());
```

Window animation

```
overridePendingTransition(android.R.anim.fade_in,
android.R.anim.fade_out);
```

Styly a témata

- Style
  - sada vlastností které určují vzhled a formát okna nebo konkrétního view
- Theme
  - styl aplikovaný na celou aplikaci

Inline vs Style vs Theme

- Level 0 - inline
- Level 1 - style
- Level 2 - theme

Inline vs Style vs Theme

- Level 0 - inline

```
<TextView
        android:id="@+id/row_validity_tv"
        android:layout_width="wrap_content"
        android:layout_height="?android:attr/listPreferredItemHeight"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textStyle="bold"
        android:gravity="center_vertical"
        android:layout_marginLeft="3dip"
        android:text="@string/l_invalid"
        android:textColor="@color/orange" />
<TextView
        android:id="@+id/row_valid_to_hint_tv"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/l_valid_to_long"
        android:layout_gravity="center_vertical"
        android:gravity="right"
        android:layout_weight="1" />
```

layout.xml

Inline vs Style vs Theme

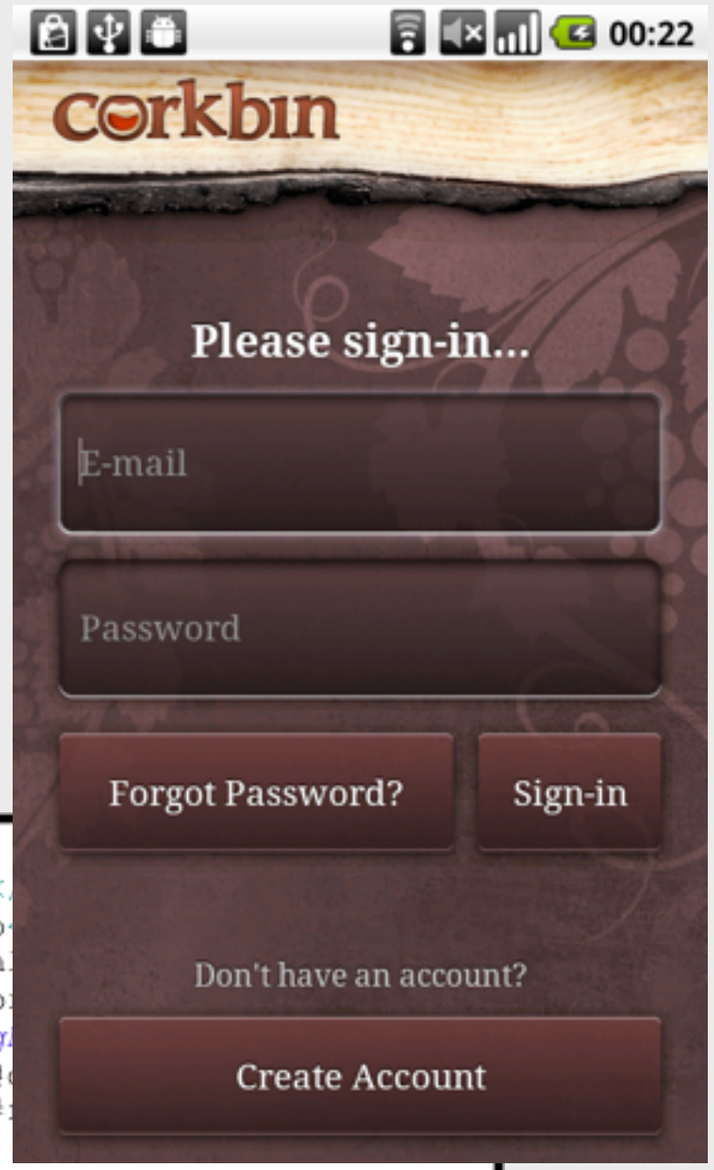
- Level 1 - style

```
<EditText
        android:id="@+id/login_email_et"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0"
        android:hint="E-mail"
        android:inputType="textEmailAddress"

android:textAppearance="@style/TextAppearance.Medium.Inverse"
/>
<EditText
        android:id="@+id/login_pass
        android:layout_width="fill_pa
        android:layout_height="wrap_c
        android:layout_weight="0"
        android:hint="Password"
        android:inputType="textPassw

android:textAppearance="@style/TextApp
        android:imeOptions="actionGo"
/>
```

layout.xml




```
<style name="TextAppearance">
    <item name="android:typeface">serif</
    <item name="android:textSize">16.0sp
    <item name="android:textStyle">norma
    <item name="android:textColor">@colo
    <item name="android:textColorHighlig
    <item name="android:textColorHint">@
    <item name="android:textColorLink">#
</style>
<style name="TextAppearance.Medium">
    <item name="android:textSize">14.0sp</item>
</style>
<style name="TextAppearance.Medium.Inverse">
    <item name="android:textColor">@color/white</item>
    <item name="android:textColorHint">@color/gray</item>
</style>
```

Styles.xml

Inline vs Style vs Theme

- Level 2 - theme


```
<Button
        android:id="@+id/login_forget_bt"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="0.5"
        android:text="@string/l_forget_password"
/>
<Button
        android:id="@+id/login_login_bt"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight=
        android:text="@string/
/>
```

layout.xml

```
<style name="Button" parent="@android:style/Widget.But
        <item name="android:background">@drawable/bu
        <item name="android:textAppearance">@style/T
        <item name="android:padding">15dip</item>
    <item name="android:textColor">@color/button_text_color</item>
</style>

<style name="MyTheme" parent="@android:style/Theme.NoTitleBar">
        <item name="android:buttonStyle">@style/Button</item>
</style>
```

themes.xml

Výhody

- Šetří čas
- Neopakuje se stále stejný kód
- Oddělení vzhledu od struktury UI
  - je vyměnitelný

Přepínání témat

- Plné využití themes jak jsme si ukázali
- Navíc:
  - Vlastní view
  - Ikonky
  - Různé styly pro text

Attribute

- V layout.xml nepřiřadíme přímo style, ale attribute
- Hodnotu attributu nastavíme v tématu
- Attribute se vyhodnotí v runtimu podle aktuálního tématu

Attribute

```
<declare-styleable name="homeIcons">
        <attr name="homeBranchesIcon" format="integer" />
        <attr name="homeATMsIcon" format="integer" />
        <attr name="homeContactsIcon" format="integer" />
        <attr name="homeNewsIcon" format="integer" />
        <attr name="homeSettingsIcon" format="integer" />
</declare-styleable>
```

attrs.xml

Attribute

```
<style name="CustomTheme" parent="android:Theme.Light.NoTitleBar">
      <item name="android:radioButtonStyle">@style/radioStyle</item>
      <item name="android:seekBarStyle">@style/seekbarStyle</item>
      <item name="android:editTextStyle">@style/editTextStyle</item>
```

```
      <item name=“homeBranchesIcon">
         @drawable/ic_affiliate_selector</item>
      <item name="homeATMsIcon">@drawable/ic_atm_selector</item>
      <item name=“homeContactsIcon">
         @drawable/ic_contact_selector</item>
      <item name="homeNewsIcon">@drawable/ic_news_selector</item>
      <item name=“homeSettingsIcon">
         @drawable/ic_settings_selector</item>
</style>
```

theme.xml

Attribute

```
<style name="CustomTheme.Green">
      <item name=“android:radioButtonStyle">
         @style/greenRadioStyle</item>
      <item name=“android:seekBarStyle">
         @style/greenSeekbarStyle</item>
      <item name=“android:editTextStyle">
         @style/greenEditTextStyle</item>

      <item name=“homeBranchesIcon">
         @drawable/green_ic_affiliate_selector</item>
      <item name="homeATMsIcon">@drawable/green_ic_atm_selector</item>
      <item name=“homeContactsIcon">
         @drawable/green_ic_contact_selector</item>
      <item name=“homeNewsIcon">
         @drawable/green_ic_news_selector</item>
      <item name=“homeSettingsIcon">
         @drawable/green_ic_settings_selector</item>
</style>
```

theme.xml

Attribute

```
<ImageButton android:id="@+id/homeAtm"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
             android:src="?homeATMsIcon" />
```

layout.xml

Přepínání témat

```
public abstract class BaseActivity extends Activity {

    private SkinManager mManager;
    private int mCurrentThemeId;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        ...

        final int themeId = this.mManager.getSelectedThemeId();
        mCurrentThemeId = themeId;

        this.setTheme(themeId);
    }

  ...

}
```

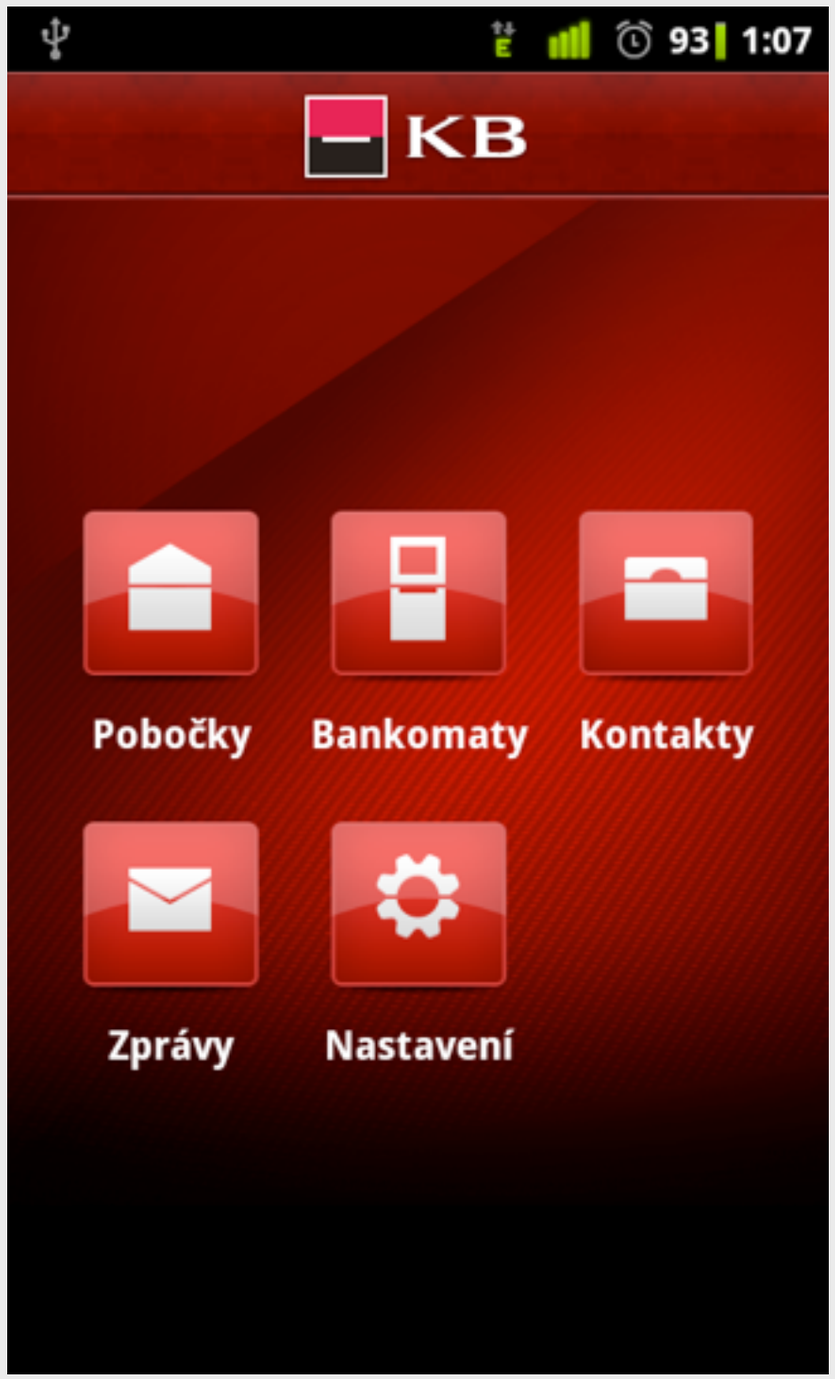
Přepínání témat

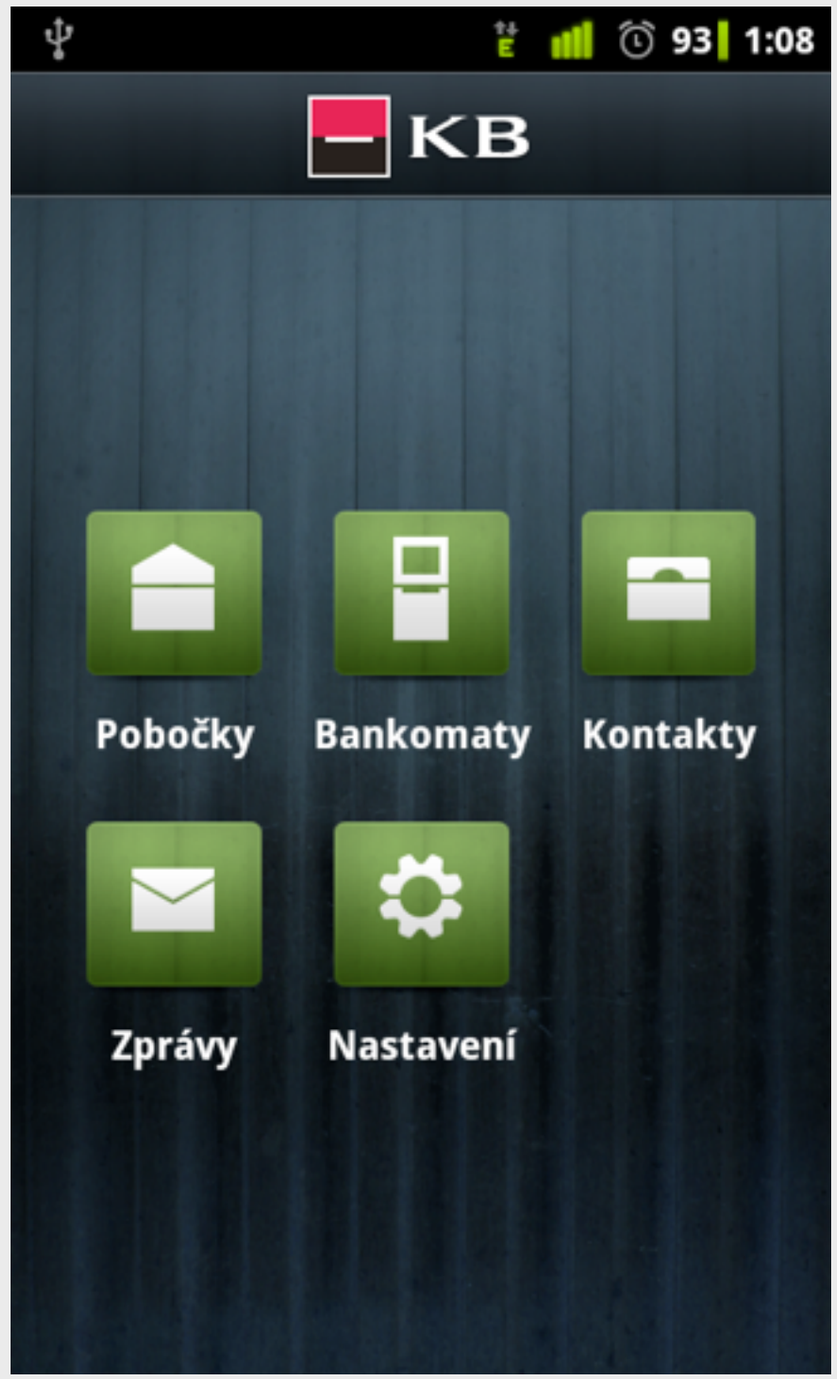
```
@Override
protected void onResume() {
    if (mCurrentThemeId != mManager.getSelectedThemeId()) {
        Log.d(Utils.TAG, "new theme selected, restarting activity");

        Intent starterIntent = this.getIntent();
        starterIntent.setAction(null);

        startActivity(starterIntent);
        finish();
        super.onResume();
        return;
    }
}
```

Přepínání témat





Díky.

Otázky?