4. UŽIVATELSKÉ ROZHRANÍ

BI-AND

Evropský sociální fond
Praha & EU: Investujeme do vaší budoucnosti

Uživatelské rozhraní

- Layouty
- ScrollView
- Taby
- ViewFlipper a ViewPager
- Dialogy
- Menus

FrameLayout

- Nejjednodušší a nejzákladnější layout
- Vlastnosti
  - Nelze nijak pozicovat
  - Všechny View umístěny do levého horního rohu
  - View naskládány „na sebe“
- Používané pro speciální/méně časté případy
  - Tvorba menu pomocí animací
  - „Obalovací“ layout

LinearLayout

- Všechny prvky jsou zarovnávány vertikálně či horizontálně

```
android:orientation=["vertical" | "horizontal"]
```

- Možnost vnořování do sebe navzájem
  - Nepřehlednost kódu
  - Náročnější na výkon

- Prvky mohou mít definované `layout_weight`
  - Určuje, jakým způsobem si prvky rozdělí (zbývající) místo

RelativeLayout

- Prvky jsou pozicovány relativně proti sobě, např.
  - Prvek B je napravo od prvku A
  - Prvek B je zarovnán nalevo ve svém rodiči
- Výhody:
  - Přehlednost kódu
  - Méně výpočetně náročné
  - Bezpečnější pozicování (zodpovědnější chování UI)
- Nevýhody:
  - Každý relativně pozicovaný prvek musí mít `android:id`
  - Návrh se obtížněji upravuje

Příklady možností pozicování

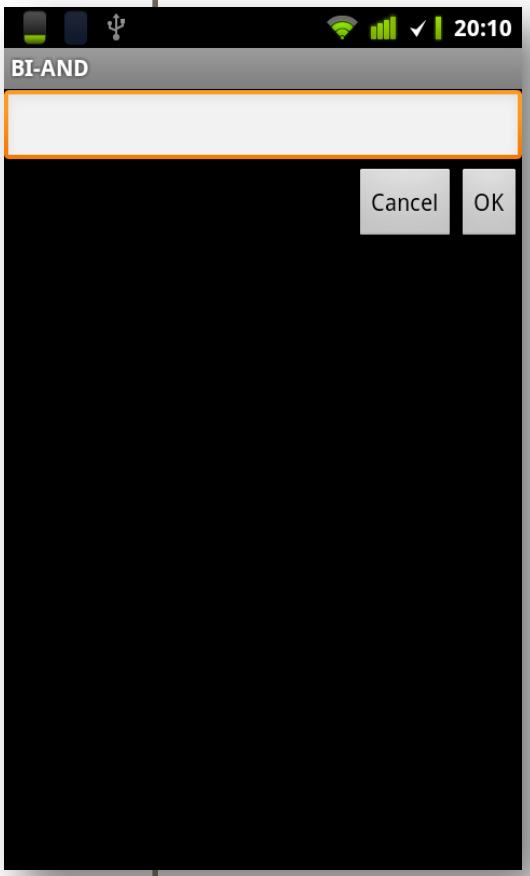
- `layout_above` – zadává se ID
- `layout_alignLeft` – zadává se ID

Pozor na kruhové závislosti

- `layout_alignParentLeft` – true/false
- `layout_centerHorizontal` – true/false
- `layout_toLeftOf` - zadává se ID
- Prvky defaultně pozicovány do levého horního rohu

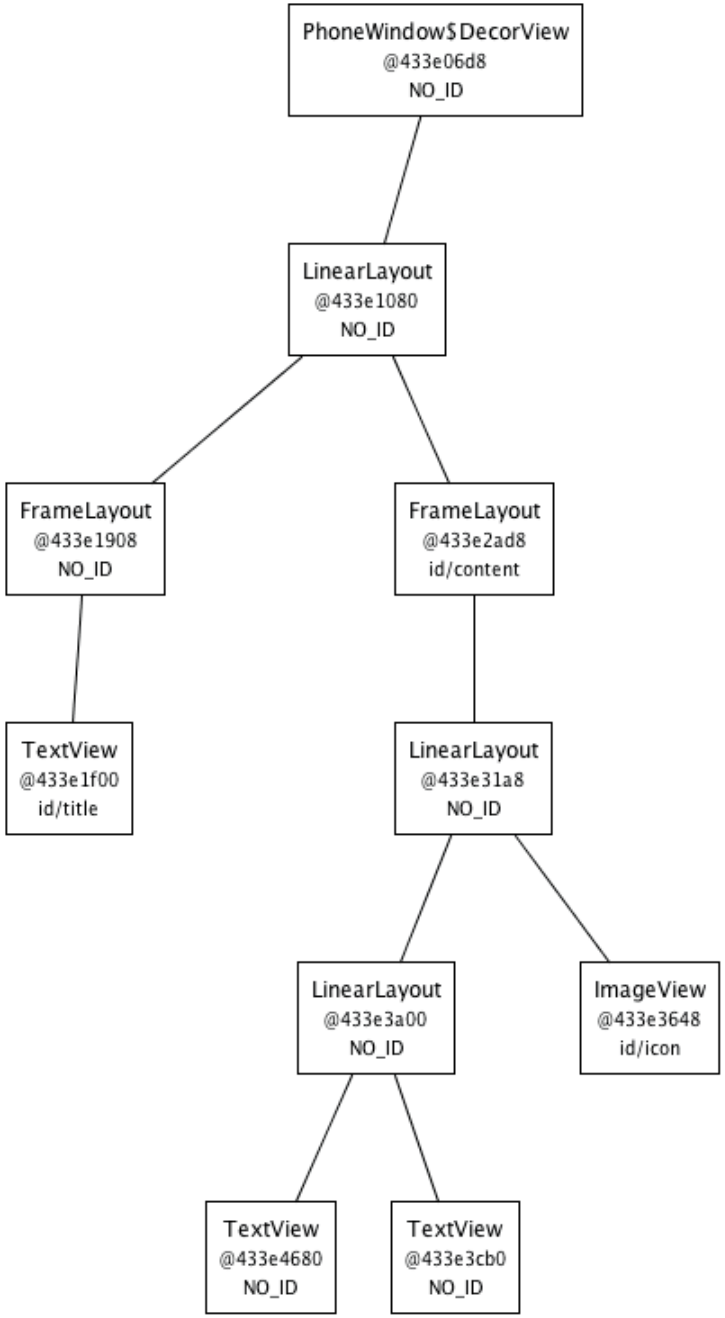
RelativeLayout - příklad

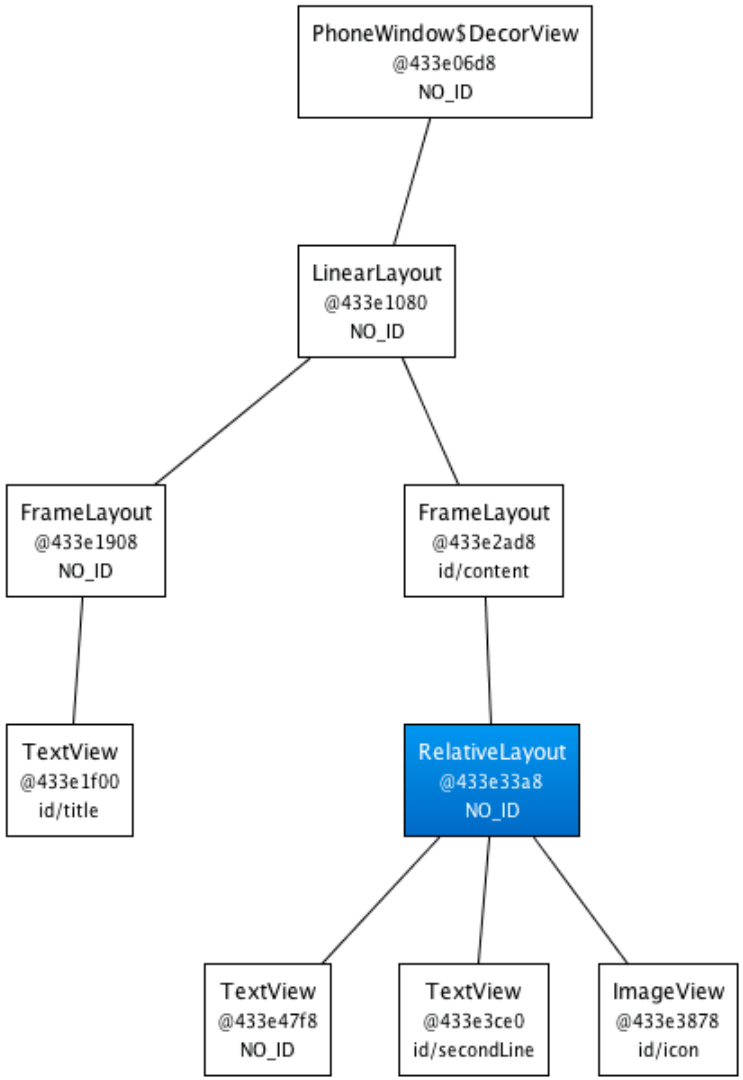
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <EditText android:id="@+id/entry"
                android:layout_width="match_parent"
                android:layout_height="wrap_content" />
        <Button   android:id="@+id/ok"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_below="@id/entry"
                android:layout_alignParentRight="true"
                android:text="OK" />
        <Button   android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_toLeftOf="@id/ok"
                android:layout_alignTop="@id/ok"
                android:text="Cancel" />
</RelativeLayout>
```



LinearLayout vs. RelativeLayout

- LinearLayout
  - Náročná operace `layout_weight`
  - Obvykle nutné použít více vnořených layoutů
    - Horší orientace v návrhu
- RelativeLayout
  - Trochu složitější návrh a úprava layoutu
- Rozdíl znatelný především u ListView a vykreslování více položek


PhoneWindow$DecorView
@433e06d8
NO_ID
LinearLayout
@433e1080
NO_ID
FrameLayout
@433e1908
NO_ID
FrameLayout
@433e2ad8
id/content
TextView
@433e1f00
id/title
LinearLayout
@433e31a8
NO_ID
LinearLayout
@433e3a00
NO_ID
ImageView
@433e3648
id/icon
TextView
@433e4680
NO_ID
TextView
@433e3cb0
NO_ID


PhoneWindow$DecorView
@433e06d8
NO_ID
LinearLayout
@433e1080
NO_ID
FrameLayout
@433e1908
NO_ID
FrameLayout
@433e2ad8
id/content
TextView
@433e1f00
id/title
RelativeLayout
@433e33a8
NO_ID
TextView
@433e47f8
NO_ID
TextView
@433e3ce0
id/secondLine
ImageView
@433e3878
id/icon

TableLayout - příklad

```
<TableLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:stretchColumns="1">
    <TableRow>
        <TextView android:padding="3dip" android:text="Open"/>
        <TextView android:text="Ctrl-O"
                  android:gravity="right"
                  android:padding="3dip" />
    </TableRow>

    <TableRow>
        <TextView android:text="Save as...„"
                  android:padding="3dip" />
        <TextView android:text="Ctrl-Shift-S"
            android:gravity="right" android:padding="3dip" />
    </TableRow>
</TableLayout>
```

Nepoužívat k pozicování!

Nelze nastavit layout width u řádků

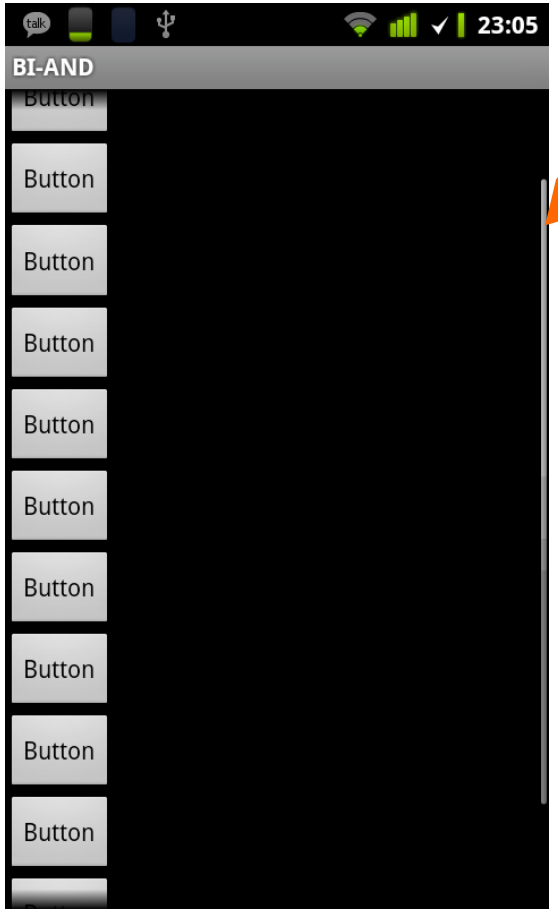
GridLayout

Rozdíl mezi GridLayout a GridView

- Prvky rozděleny do mřížky
  - Lze použít prvek „prázdné místo“
  - Lze určovat „šířku“ buňky
- Určuje se počet sloupců a řádků
- Není možné určovat váhy
- Dostupné od API Level 14
  - Součástí Compatibility Library v7

ScrollView

- Umožňuje scrollování, pokud velikost vnořených prvků přesahuje fyzickou velikost displeje
- Pouze jeden vnořený element



Neobalovat ListView do ScrollView!

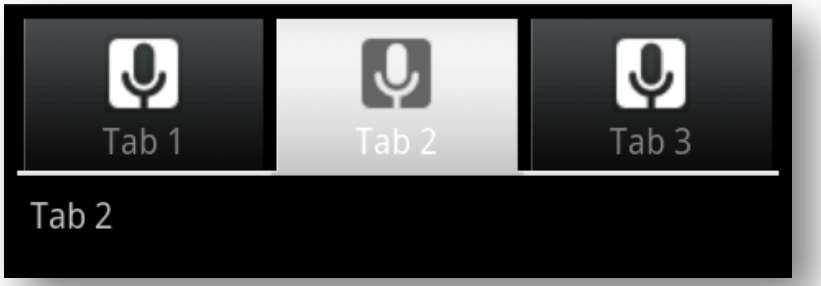
```
<ScrollView
      xmlns:android=
            "http://schemas.android.com/apk/res/android"
      android:layout_height="wrap_content"
      android:layout_width="fill_parent">
      <LinearLayout android:layout_height="fill_parent"
            android:layout_width="fill_parent"
            android:orientation="vertical">
            <Button    android:text="Button"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content" />
            ...
            <Button    android:text="Button"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content" />
      </LinearLayout>
</ScrollView>
```

Taby

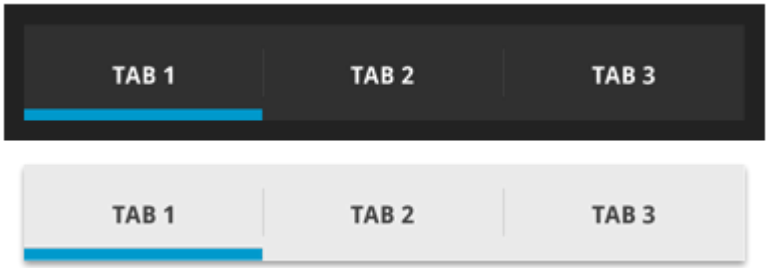
- Umožňují vložení více Activit nebo Views do jedné obrazovky a přepínání mezi nimi pomocí tabů

- Existují 2 možné implementace:
  - Přepínání activit
    - V každém tabu běží normální nezávislá Activita
    - Neaktivní activity se chovají úplně stejně jako běžné activity na pozadí (při nedostatku paměti je systém může ukončit)
  - Přepínání Views
    - Přepnutím tabu ze pouze změní View v aktuální aktivitě
  - Tyto 2 metody je možné kombinovat

TabActivity je deprecated!



Taby



- TabActivity je již deprecated
- Taby se nyní přidávají většinou do ActionBaru
- Nahrazení TabActivity přímo TabHost s TabWidget za použití TabManageru
- Pro přepínání se používají fragmenty (nutnost přidat Compatibility Library pro starší zařízení)
- Více v 12. přednášce

ViewFlipper

- Widget pro rychlé přepínání mezi jednotlivými obrazovkami
- Vhodné např. pro tvorbu průvodců

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent">
      <ViewFlipper android:id="@+id/ViewFlipper01"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">
            <include
                  layout="@layout/screen1" />
            <include
                  layout="@layout/screen2" />
            <include
                  layout="@layout/screen3" />
      </ViewFlipper>
```

Jednotlivé obrazovky průvodce umístíme do zvláštních souborů ve složce `layout`

ViewFlipper - pokračování

```
<RelativeLayout
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:layout_alignParentBottom="true"
      android:background="@drawable/bottom_bar"
      android:gravity="center_vertical">
      <Button
            android:layout_width="100dp"
            android:layout_height="wrap_content"
            android:id="@+id/Button02"
            android:layout_alignParentRight="true"
            android:text="Next" />
      <Button
            android:layout_height="wrap_content"
            android:layout_width="100dp"
            android:id="@+id/Button01"
            android:text="Previous"
            android:layout_alignParentLeft="true" />
</RelativeLayout>
</RelativeLayout>
```

Vytvoříme wizard-like dolní lištu pomocí RelativeLayout

Pozadí nebudeme referencovat na to, co je v systému, a raději si ho zkopírujeme do své aplikace např. ze složky `sdk/platforms/android-10`

ViewFlipper - pokračování

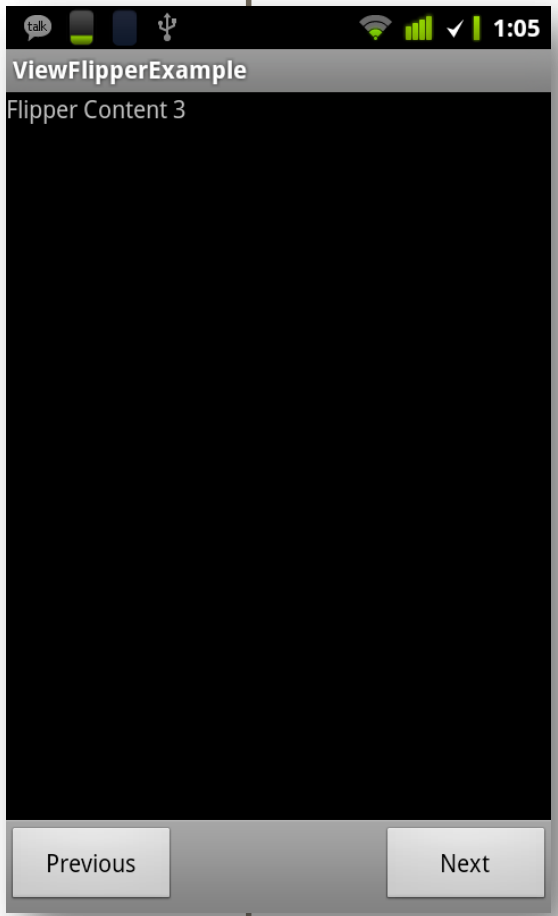
- Následujícím způsobem vytvoříme jednotlivé obrazovky průvodce a pojmenujeme `screen[1-3].xml`
- Pozn.: `<merge>` není nutné použít a obsah obrazovek průvodce můžeme rovnou psát do `<ViewFlipper>`

```
<?xml version="1.0" encoding="UTF-8"?>
<merge
    xmlns:android="http://schemas.android.com/apk/res/android">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Flipper Content 1" />
</merge>
```

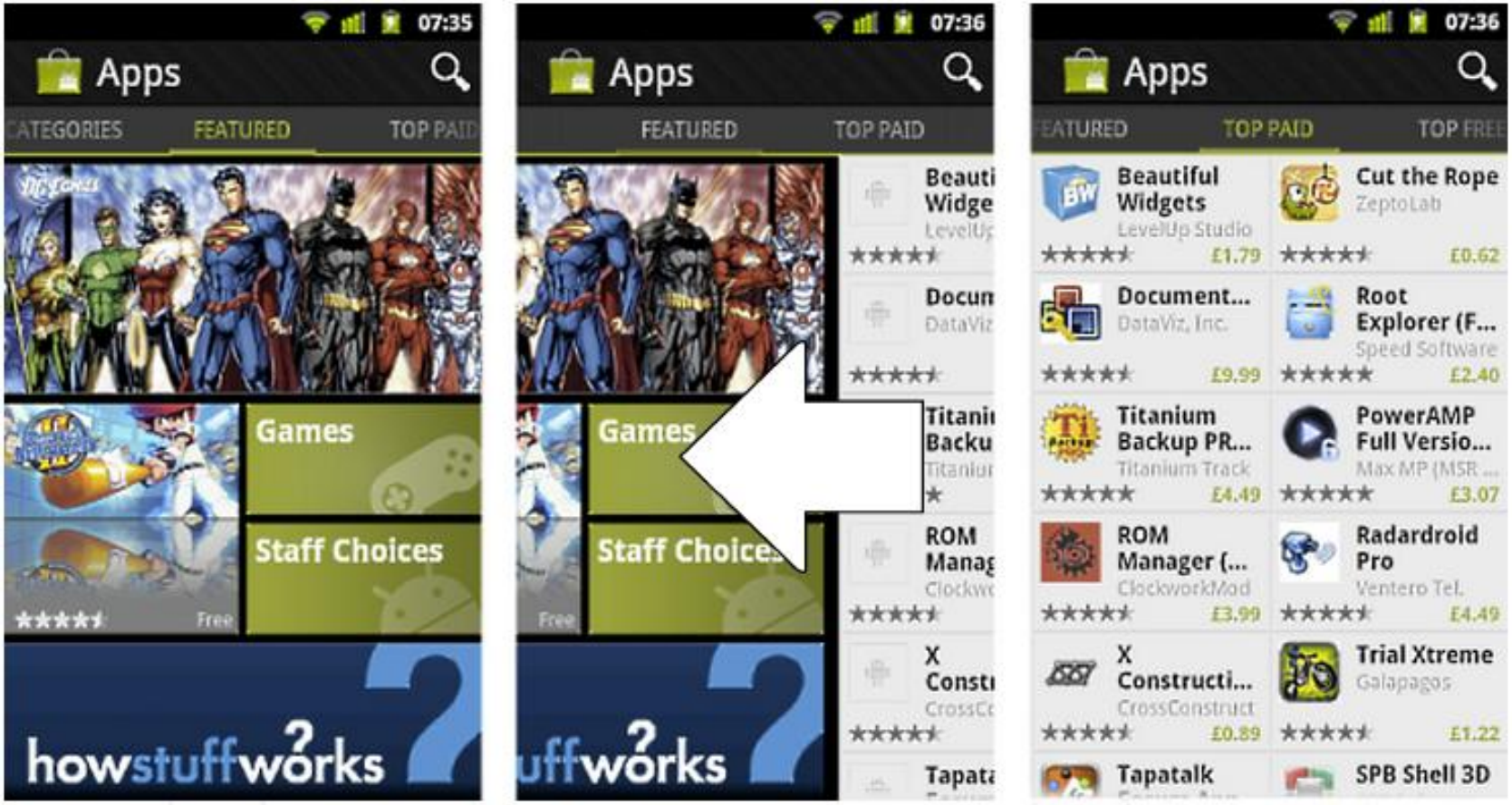
Merge je „náhradou“ za root XML element

ViewFlipper – pokračování

```
public class ViewFlipperExample extends Activity implements OnClickListener {
	private Button next;
	private Button previous;
	private ViewFlipper vf;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
		vf = (ViewFlipper) findViewById(R.id.ViewFlipper01);
		previous = (Button) findViewById(R.id.Button01);
		next = (Button) findViewById(R.id.Button02);
		next.setOnClickListener(this);
		previous.setOnClickListener(this);
	}
	@Override
	public void onClick(View v) {
		if (v == next) {
			vf.showNext();
		}
		if (v == previous) {
			vf.showPrevious();
		}
	}}
```



ViewPager


07:35
Apps
CATEGORIES
FEATURED
TOP PAID
Games
Staff Choices
howstuffworks
07:36
Apps
FEATURED
TOP PAID
Games
Staff Choices
07:36
Apps
FEATURED
TOP PAID
TOP FREE
Beautiful Widgets
LevelUp Studio
£1.79
Cut the Rope
ZeptoLab
£0.62
Document...
DataViz, Inc.
£9.99
Root Explorer (F...
Speed Software
£2.40
Titanium Backup PR...
Titanium Track
£4.49
PowerAMP Full Versio...
£3.07
ROM Manager (...
ClockworkMod
£3.99
Radardroid Pro
Ventero Tel.
£4.49
X Constructi...
CrossConstruct
£0.89
Trial Xtreme
Galapagos
£1.22
Tapatalk
SPB Shell 3D

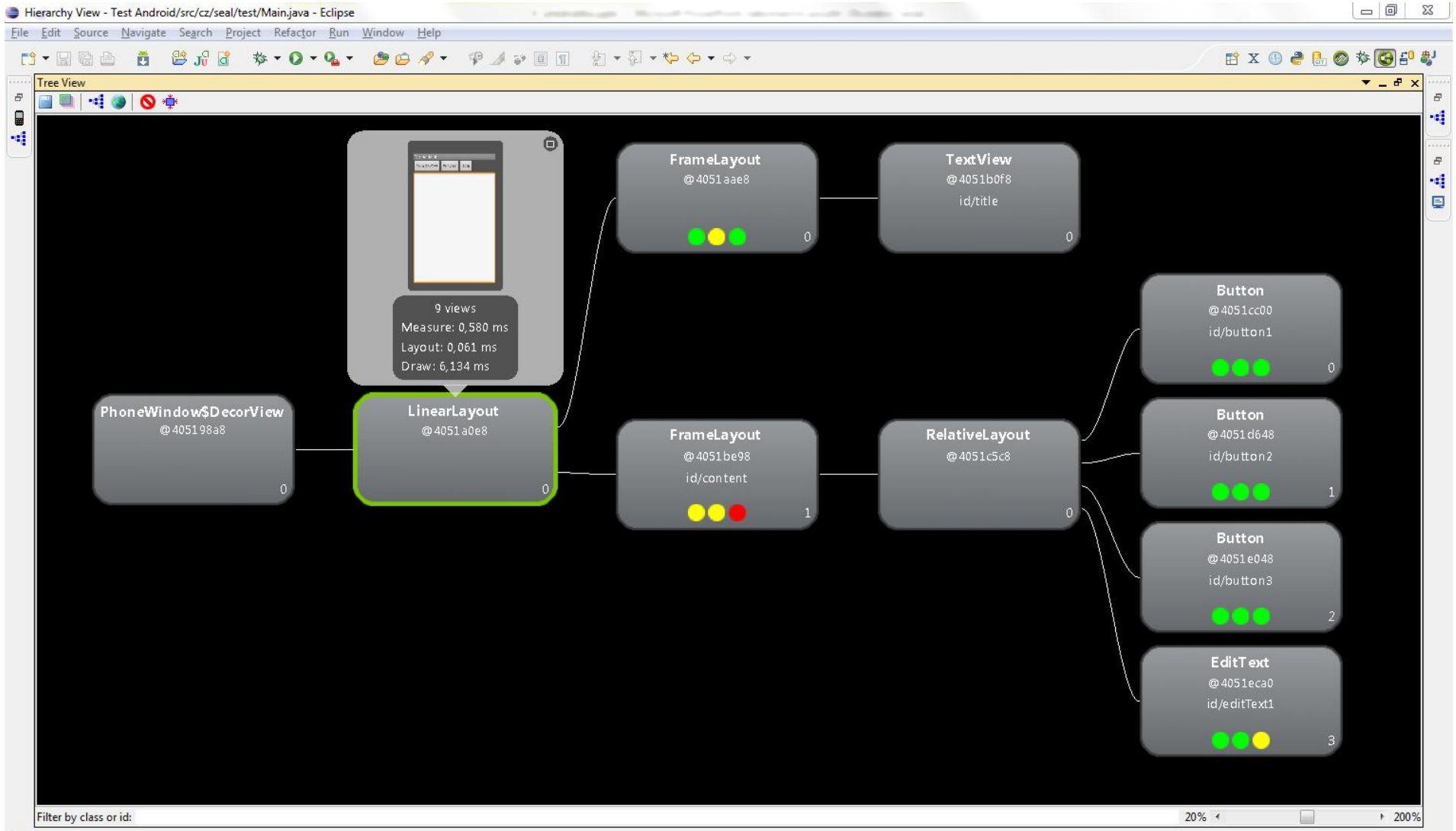
ViewPager

- Jednoduchý XML element
- Samotná logika se řeší v kódu pomocí PagerAdapter
- PagerAdapter se stará o:
  - Vytvoření a přidání nového View
  - Odstranění View
  - Lze přepsat i další metody přechodu
- Nutné přidat Compatibility Library pro starší zařízení

Hierarchy Viewer

- Nástroj z SDK pro zobrazení TreeView
- Vhodné pro optimalizaci layoutů a jejich porovnání
- Časy vykreslení (od verze 2.3)
- Zobrazení okrajů prvků layoutu a jejich property
- Možnost uložit strom nebo celý náhled
- Lze prozkoumat i cizí layouty (pouze na emulátoru)

Hierarchy Viewer



Dialogy

- Předání (důležité) informace uživateli
- Získání informace od uživatele
- Vytvářeny v překryté metodě `onCreateDialog()`
- Přístup a nastavení z hlavního vlákna

```
protected Dialog onCreateDialog(int id) {
    Dialog dialog;
    switch(id) {
    case DIALOG_PAUSED_ID:
        break;
    case DIALOG_GAMEOVER_ID:
        break;
    default:
        dialog = null;
    }
    return dialog;
}
```

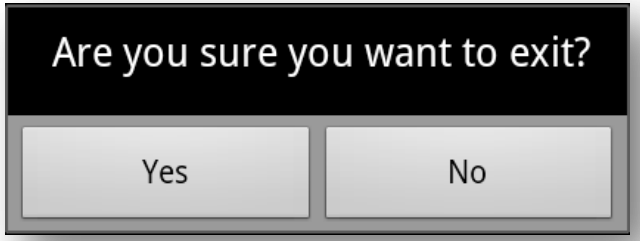
```
showDialog(DIALOG_PAUSED_ID);
```

Operace s dialogy

- `dismissDialog()`
  - Zavře dialog
- `cancel()`
  - Stejné jako dismissDialog() a navíc zavolá `DialogInterface.OnCancelListener`
- `removeDialog(int)`
  - Pouze při použití `onCreateDialog(int)`
  - Zruší referenci na dialog a zavolá `dismissDialog()`, pokud se zobrazuje

- Nastavení listenerů přes `DialogInterface`
  - `OnDismissListener`
  - `OnCancelListener`
    - Zavolán při zrušení dialogu uživatelem nebo při stisku Back

AlertDialog

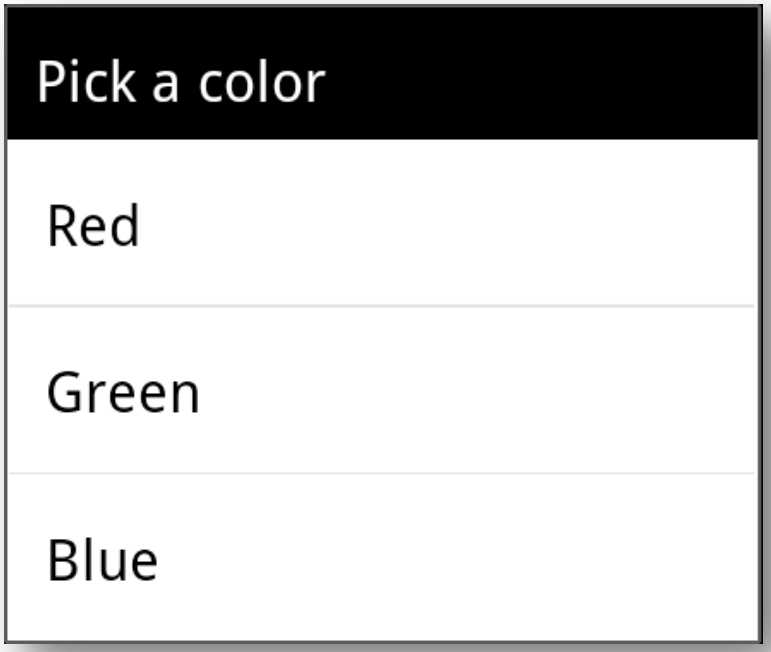


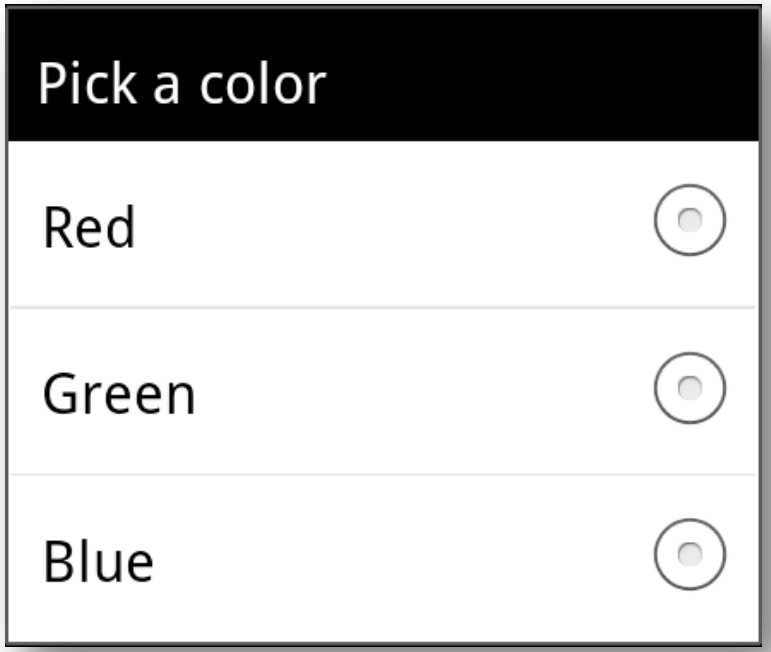
- Velice jednoduchá implementace
- Lze nastavit titulek, obsah zprávy, maximálně tři tlačítka, položky výběru (CheckBox, RadioButton)

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);
builder.setMessage("Are you sure you want to exit?")
    .setCancelable(false)
    .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            MyActivity.this.finish();
        }
    })
    .setNegativeButton("No", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            dialog.cancel();
        }
    });
AlertDialog alert = builder.create();
```

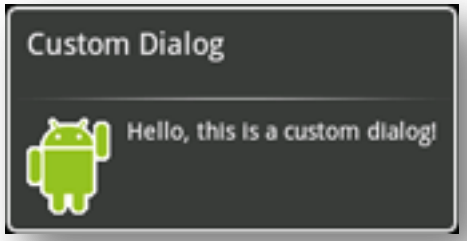
Další možnosti AlertDialogu

- Se seznamem
- S RadioButtony





Vlastní vzhled dialogu



1. Vytvoříme si xml s vlastním vzhledem:

```
<LinearLayout
      xmlns:android="http://schemas.android.com/apk/res/android"
      android:id="@+id/layout_root"
      android:orientation="horizontal"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:padding="10dp" >
    <ImageView android:id="@+id/image"
          android:layout_width="wrap_content"
          android:layout_height="fill_parent"
          android:layout_marginRight="10dp„ />
    <TextView android:id="@+id/text"
          android:layout_width="wrap_content"
          android:layout_height="fill_parent"
          android:textColor="#FFF"/>
</LinearLayout>
```

Vlastní vzhled dialogu - pokračování

```
Context mContext = getApplicationContext();
Dialog dialog = new Dialog(mContext);

dialog.setContentView(R.layout.custom_dialog);
dialog.setTitle("Custom Dialog");

TextView text = (TextView)
dialog.findViewById(R.id.text);
text.setText("Hello, this is a custom dialog!");
ImageView image = (ImageView)
dialog.findViewById(R.id.image);
image.setImageResource(R.drawable.android);
```

LayoutInflater

- Třída slouží k vytváření instancí XML souborů do odpovídajících View objektů
- Nikdy nepoužívat přímo. Vždy volat:
  - getLayoutInflater()
  - getSystemService(String)
- Nelze zatím použít s XmlPullParser na plain XML během runtime (XML musí být předzpracovano během built-time)

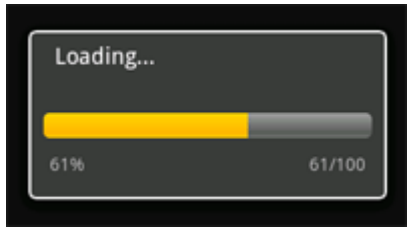
Každý View XML soubor má odpovídající Java class

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);
    .setView(getLayoutInflater().inflate(R.layout.custom_dialog,
null)).create();
```

AlertDialog.Builder – nelze použít setContentView()

Další dialogy

- ProgressDialog – 9. přednáška



- DatePickerDialog

- TimePickerDialog

Menu

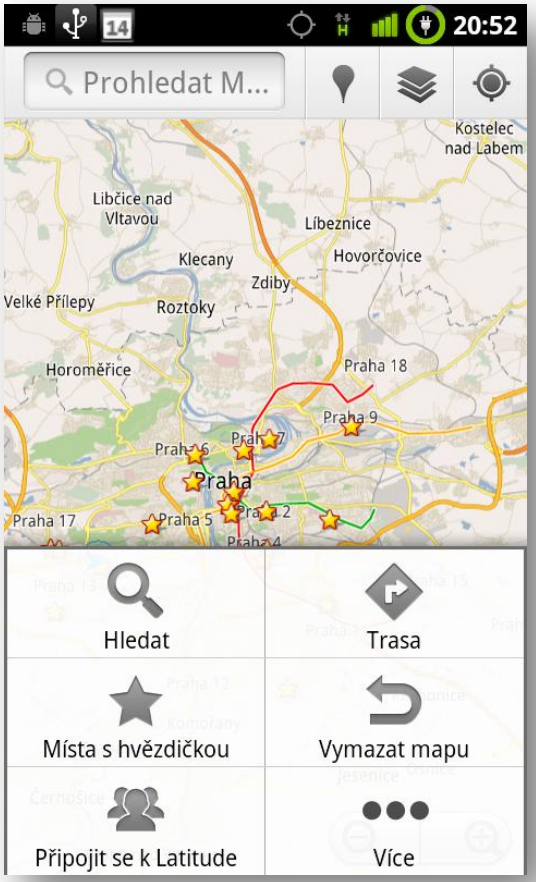
- Rychlý přistup k funkcím aplikace bez obětování cenného prostoru na obrazovce
- Každá activita může mít vlastní menu
- Vyvolání pomocí tlačítka menu
- Změna filozofie menu od Android 3.0+
- Menu podporují submenu, checkboxy, radio buttony, klávesové zkratky a ikonky
- Připevnění kontextového menu k libovolnému View

Options menu vs. Context menu

Options menu

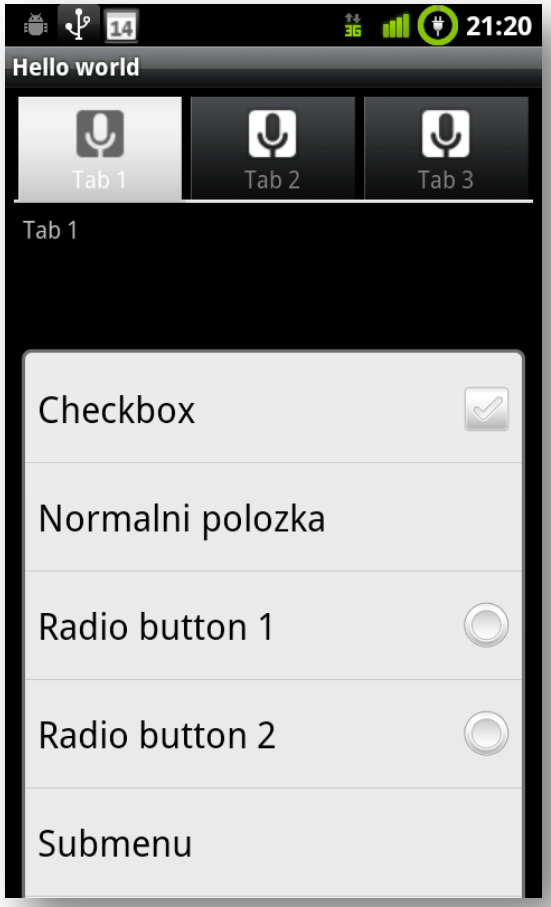
Maximální počet ikon se liší podle zařízení

- Objeví se na spodní straně obrazovky po stisknutí tlačítka menu
- Pro omezený počet položek (většinou 6) zobrazí ikonku a text
  - Zbytek se schová do *Více*
- Nezobrazuje checkboxy, radio buttony ani klávesové zkratky
- Styl vytváření ikon – viz. 5. přednáška



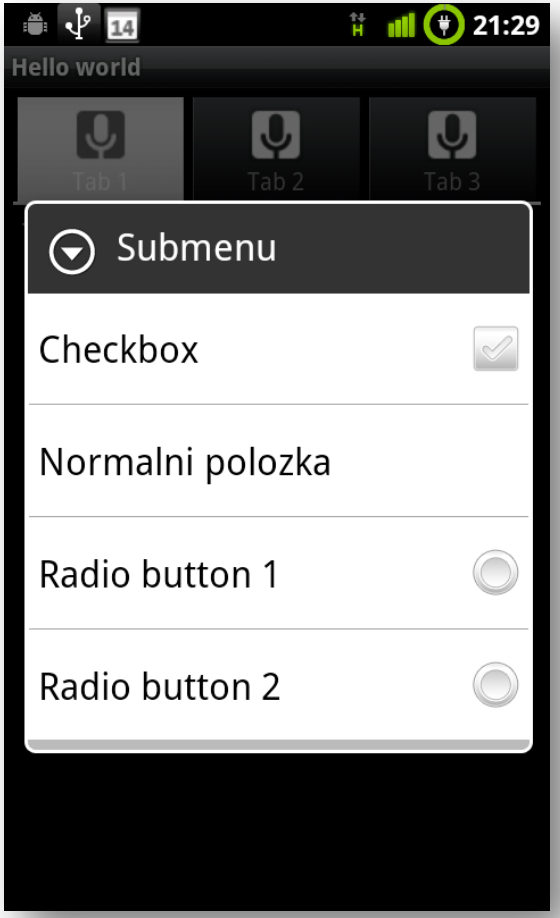
Rozšířené menu

- Volba *Více* v menu
- Scrollovatelný seznam zbylých položek,
- Zobrazují se checkboxy, radio buttony, plná délka popisku
- Nezobrazují se ikony
- Nelze vynutit zobrazení rozšířeného menu



Submenu

- Zobrazí se po kliknutí na položku obsahující v sobě další menu
- Jméno položky se objeví v titulku dialogu
- Nelze vložit submenu do submenu
- Stejné chování pro zobrazení položek jako u rozšířeného menu



Options menu

- Zavolání metody `onCreateOptionsMenu` při stisku menu tlačítka
- Jako parametr obdržíme Menu, do kterého přidáme položky
- Můžeme použít `MenuInflater` a naplnit menu položkami z XML(`res/menu/main_menu.xml`) a/nebo je přidat ručně

```
1.  @Override
2.  public boolean onCreateOptionsMenu(Menu menu) {
3.          //naplnění menu z res/menu/main_menu.xml
4.          MenuInflater inflater = getMenuInflater();
5.          inflater.inflate(R.menu.main_menu, menu);
6.          //ruční přidání položky
7.          int groupId = Menu.NONE;//skupina pro rozlišení radiogroup
8.          int itemId = 10;//ID položky pro onOptionsItemSelected
9.          int order = Menu.NONE;//pořadí položky v menu
10.         menu.add(groupId, itemId, order, "Položka").setIcon(R.drawable.icon);
11.         //ruční odebraní položky
12.         menu.removeItem(10);
13.         return super.onCreateOptionsMenu(menu);
14. }
```

Vytvoření pomocí XML nebo v kódu

Options menu – main_menu.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <menu xmlns:android="http://schemas.android.com/apk/res/android">
3.     <!-- Normální položka s ikonou -->
4.     <item android:id="@+id/item1"
5.           android:title="Normální položka"
6.           android:icon="@drawable/icon" />
7.     <!-- Checkbox -->
8.     <item android:checkable="true"
9.           android:id="@+id/groupItem1"
10.          android:title="Checkbox" />
11.    <!-- Skupina radio buttonu -->
12.    <group android:checkableBehavior="single">
13.          <item android:id="@+id/radio1"
14.                android:title="Radio button 1" />
15.          <item android:id="@+id/radio1"
16.                android:title="Radio button 2" />
17.    </group>
18.    <!-- Submenu -->
19.    <item android:id="@+id/submenu"
20.          android:title="Submenu">
21.          <menu>
22.                <item android:id="@+id/submenuItem11"
23.                      android:title="Položka submenu" />
24.
25.          </menu>
26.    </item>
27.</menu>
```

Options menu

- Po vybrání položky z menu se zavolá metoda `onOptionsItemSelected`, která dostane v atributu vybranou položku - `MenuItem`

```
1.  @Override
2.  public boolean onOptionsItemSelected(MenuItem item) {
3.      switch (item.getItemId()) {
4.      case R.id.item1:
5.              ...
6.              break;
7.      case R.id.radio1:
8.              ...
9.              break;
10.     case R.id.submenuItem11:
11.             ...
12.             break;
13.     }
14.     return super.onOptionsItemSelected(item);
15. }
```

ID v XML vs. ID v add()

Options menu

- Metoda `onCreateOptionsMenu` se zavolá pouze před prvním vytvořením menu
- Pokud potřebujeme menu dynamicky měnit, je třeba tak činit v `onPrepareOptionsMenu`, které se zavolá před každým vysunutím menu
- Při zavírání menu se volá `onOptionsMenuClosed`

Context menu

- Zobrazeno po dlouhém podržení daného View nebo zavoláním `View.createContextMenu(menu)`
  - Lze změnit i na libovolnou jinou akci
- Podobné jako Submenu
- Dané View se musí zaregistrovat na použití kontextového menu pomocí `registerForContextMenu(View);`
  - Lze tedy mít více kontextových menu v jedné activitě

Context menu

onCreateContextMenu() voláno pokaždé

- Pro zobrazení menu je třeba přepsat metodu `onCreateContextMenu(ContextMenu, View, ContextMenuInfo)`
  - Implementace podobná jako u `onCreateOptionsMenu(Menu);`
  - Atribut View – View, které uživatel vybral
  - Atribut `ContextMenuInfo` – podrobnosti o kontextovém menu

- Pro provedení akce po výběru položky z menu je třeba přepsat `onContextItemSelected(MenuItem);`
  - Stejné jako `onOptionsItemSelected(MenuItem);`
  - Pozor na unikátní ID u každé položky, pokud je více kontextových menu

Context menu – příklad

```
1.  @Override
2.  protected void onCreate(Bundle savedInstanceState) {
3.       ...
4.       registerForContextMenu(view);
5.       ...
6.  }
```

```
1.  @Override
2.  public void onCreateContextMenu(ContextMenu menu, View v
3.                                  , ContextMenuInfo menuInfo) {
4.       MenuInflater inflater = getMenuInflater();
5.       inflater.inflate(R.menu.context_menu, menu);
6.       super.onCreateContextMenu(menu, v, menuInfo);
7.  }
```

```
1.  @Override
2.  public boolean onContextItemSelected(MenuItem item) {
3.       switch (item.getItemId()) {
4.       case R.id.new_game:
5.                newGame();
6.                return true;
7.       default:
8.                return super.onContextItemSelected(item);
9.       }
10. }
```

Další zdroje

- http://developer.android.com/resources/articles/layout-tricks-merge.html
- http://android-developers.blogspot.com/2011/08/horizontal-view-swiping-with-viewpager.html
- http://developer.android.com/guide/topics/ui/layout/relative.html