5. LISTVIEW

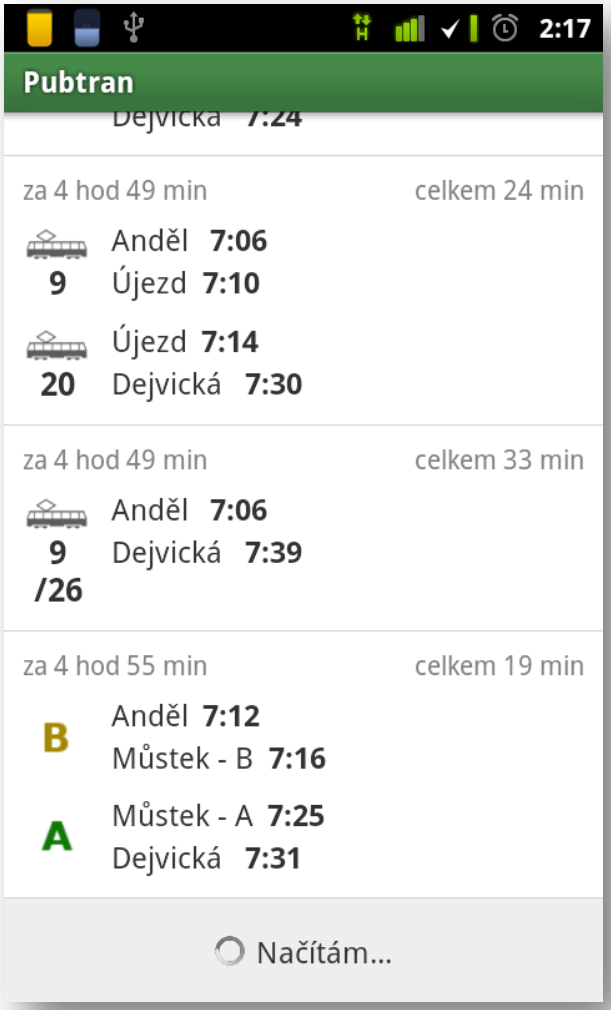
BI-AND

Obsah

- ListView
- ListActivity
- Adapter
  - BaseAdapter
  - ArrayAdapter
  - CursorAdapter
  - SimpleCursorAdapter
- Zásady tvorby uživatelského rozhraní

ListView

- Potomek `AdapterView`
- `View` pro zobrazení vertikálního posunovatelného seznamu
  - Položkami `ListView` mohou být i komplikované `View`
- Možnost přidání hlavičky a zápatí
  - `addHeaderView(View v)`
  - `addFooterView(View v)`
- Pro práci s ním se používá `ListActivity` (ale není nutností)



ListActivity

- Activita obsahující právě jeden ListView
- Layout musí obsahovat jeden `ListView` s ID `android:id="@android:id/list"`
- Obsahuje `ListAdapter`

Adapter

- Most mezi `AdapterView` a zobrazovanými daty
- Zodpovědný za vytváření View pro každou položku datové kolekce
- Využívá se při vkládání položek do `ListView`, `Spinner`, `ViewPager`, `AutoCompleteTextView` aj.

BaseAdapter

- Abstraktní třída
- Základní stavební kámen pro tvorbu vlastních adaptéru

ArrayAdapter<T>

- Generická třída
- BaseAdapter umožňující vložení pole libovolných prvků
- Pokud není přetížena metoda `getView()`, tak se naplňuje pouze jediný TextView

CursorAdapter

- BaseAdapter používající pro získání dat kurzoru (Cursor)
- Cursor musí obsahovat sloupec pojmenovaný _id

SimpleCursorAdapter

Rozdíl mezi SimpleCursorAdapter a CursorAdapter?

- CursorAdapter mapující sloupce z Cursoru na `TextView` nebo `ImageView` definované v XML
- Na jeden View je možné přiřadit více hodnot z Cursoru

ListActvity za použití ArrayAdapteru

```
public class MyList extends ListActivity {
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        String[] names = new String[] { "Linux", "Windows 7", "Eclipse", "Suse",
                "Ubuntu", "Solaris", "Android", "iPhone"};
        this.setListAdapter(new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1, names));
    }
    @Override
    protected void onListItemClick(ListView l, View v, int position, long id) {
        super.onListItemClick(l, v, position, id);
        Object o = this.getListAdapter().getItem(position);
        String keyword = o.toString();
        Toast.makeText(this, "You selected: " + keyword, Toast.LENGTH_LONG)
                .show();
    }
}
```

Výsledek ListActivity s ArrayAdapterem

ListActivity s vlastním layoutem položky

layout/rowlayout.xml

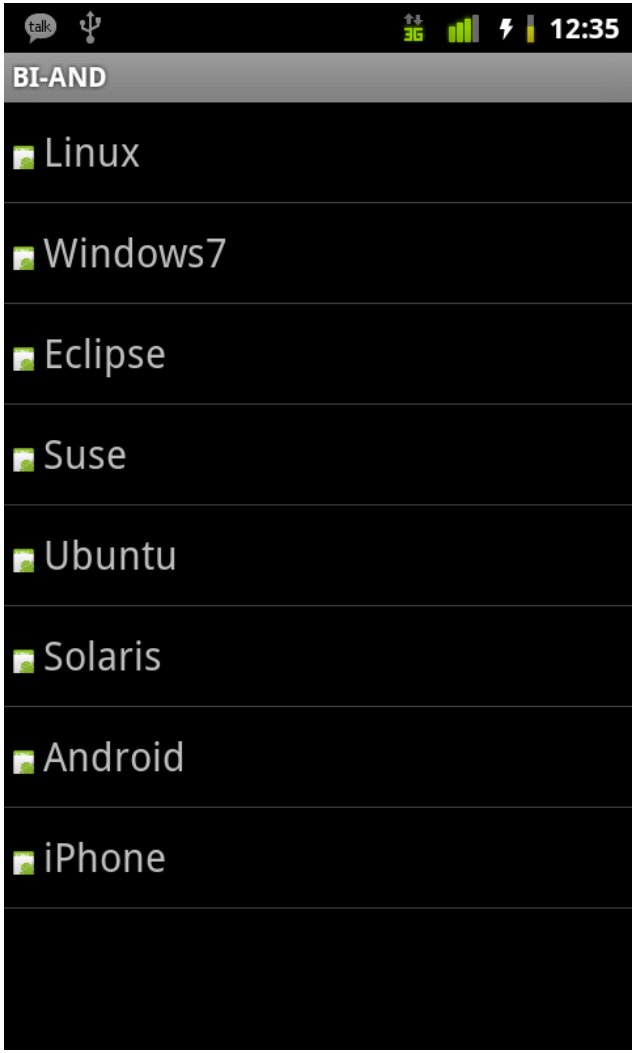
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
     android:layout_width="wrap_content" android:layout_height="wrap_content"
      android:gravity="center_vertical">

     <ImageView android:id="@+id/icon" android:layout_height="wrap_content"
          android:src="@drawable/icon" android:layout_width="22dp"
          android:layout_marginTop="4dp" android:layout_marginRight="4dp"
          android:layout_marginLeft="4dp">
     </ImageView>
     <TextView android:text="@+id/TextView01" android:layout_width="wrap_content"
          android:layout_height="wrap_content" android:id="@+id/label"
          android:textSize="30dp"></TextView>
</LinearLayout>
```

rowlayout.xml nastavíme nově vytvořenému adaptéru:

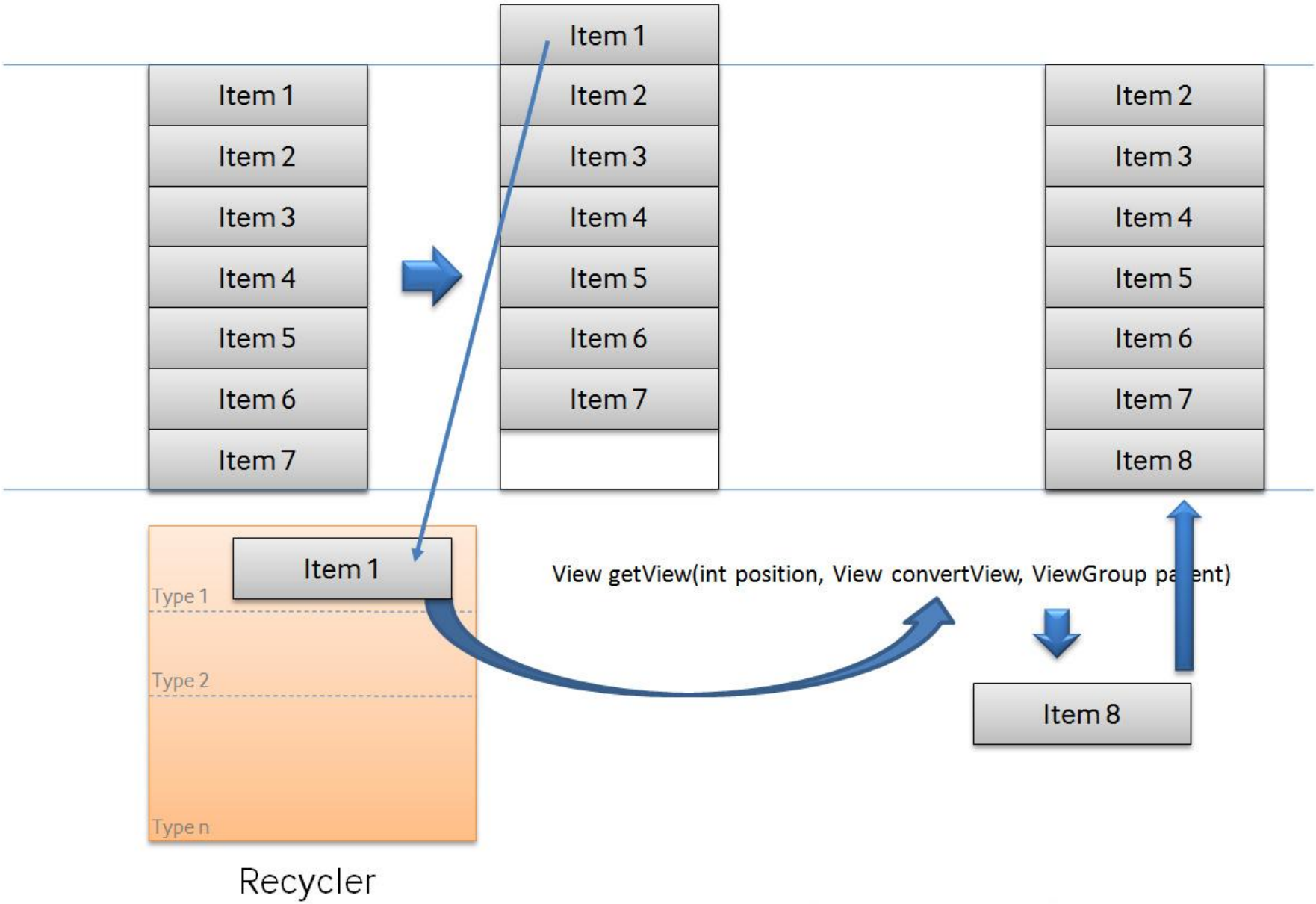
```
this.setListAdapter(new ArrayAdapter<String>(this, R.layout.rowlayout,
                    R.id.label, names));
```

Výsledek ListActivity s vlastním layoutem položky



ConvertView

- ListView obsahuje vždy pouze tolik položek, kolik je potřeba na displeji
- Při scrollování jsou jednotlivé View recyklovány
  - Prvky, které nejsou vidět jsou přesunuty do tzv. Recycleru
- Do metody `getView()` adapteru se vrací v podobě `convertView`
- ConvertView se naplní daty odpovídající položky


Item 1
Item 2
Item 3
Item 4
Item 5
Item 6
Item 7
Item 1
Item 2
Item 3
Item 4
Item 5
Item 6
Item 7
Item 2
Item 3
Item 4
Item 5
Item 6
Item 7
Item 8
Item 1
Type 1
Type 2
Type n
View getView(int position, View convertView, ViewGroup parent)
Item 8
Recycler

Holder Pattern

- Operace `findViewById()` je velmi náročná
  - Volat ji pouze v nezbytných případech
- „Zapamatovat“ si reference View jednotlivých řádků
- ViewHolder
  - Drží reference na View layoutu řádku
  - Přidat referenci ViewHolderu k řádku pomocí metody `setTag()`

Vlastní ArrayAdapter

```
public class MyArrayAdapter extends ArrayAdapter<String> {
	private final Activity context;
	private final String[] names;

	public MyArrayAdapter(Activity context, String[] names) {
		super(context, R.layout.rowlayout, names);
		this.context = context;
		this.names = names;
	}

	private class ViewHolder {
		public ImageView imageView;
		public TextView textView;
	}
…
```

Drží jednotlivé `View` vlastního layoutu položky a při použití `View.getTag()` není nutné volat časově náročnou operaci `findViewById(int)`

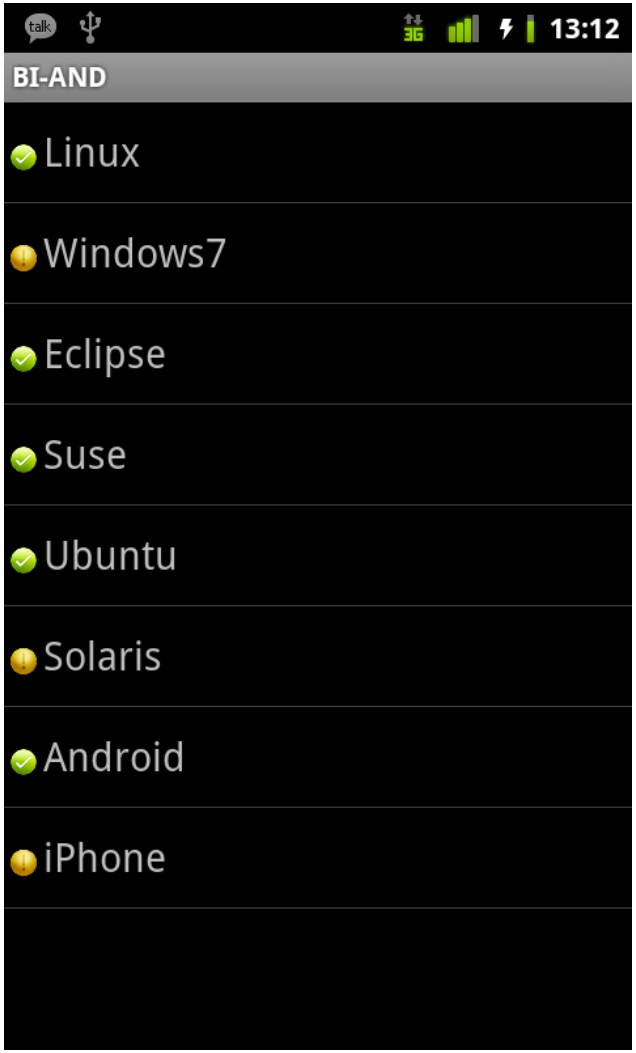
Vlastní ArrayAdapter - pokračování

```
@Override
public View getView(int position, View convertView, ViewGroup parent) {
     ViewHolder holder;
     if (convertView == null) {
          LayoutInflater inflater = context.getLayoutInflater();
          convertView = inflater.inflate(R.layout.rowlayout, null, true);
          holder = new ViewHolder();
          holder.textView = (TextView) convertView.findViewById(R.id.label);
          holder.imageView = (ImageView) convertView.findViewById(R.id.icon);
          convertView.setTag(holder);
     } else {
          holder = (ViewHolder) convertView.getTag();
     }
     holder.textView.setText(getItem(position));
     String s = getItem(position);
     if (s.startsWith("Windows 7") || s.startsWith("iPhone")
               || s.startsWith("Solaris")) {
          holder.imageView.setImageResource(R.drawable.no);
     } else {
          holder.imageView.setImageResource(R.drawable.ok);
     }
     return convertView;
  }
}
```

Recyklace existujícího `View`

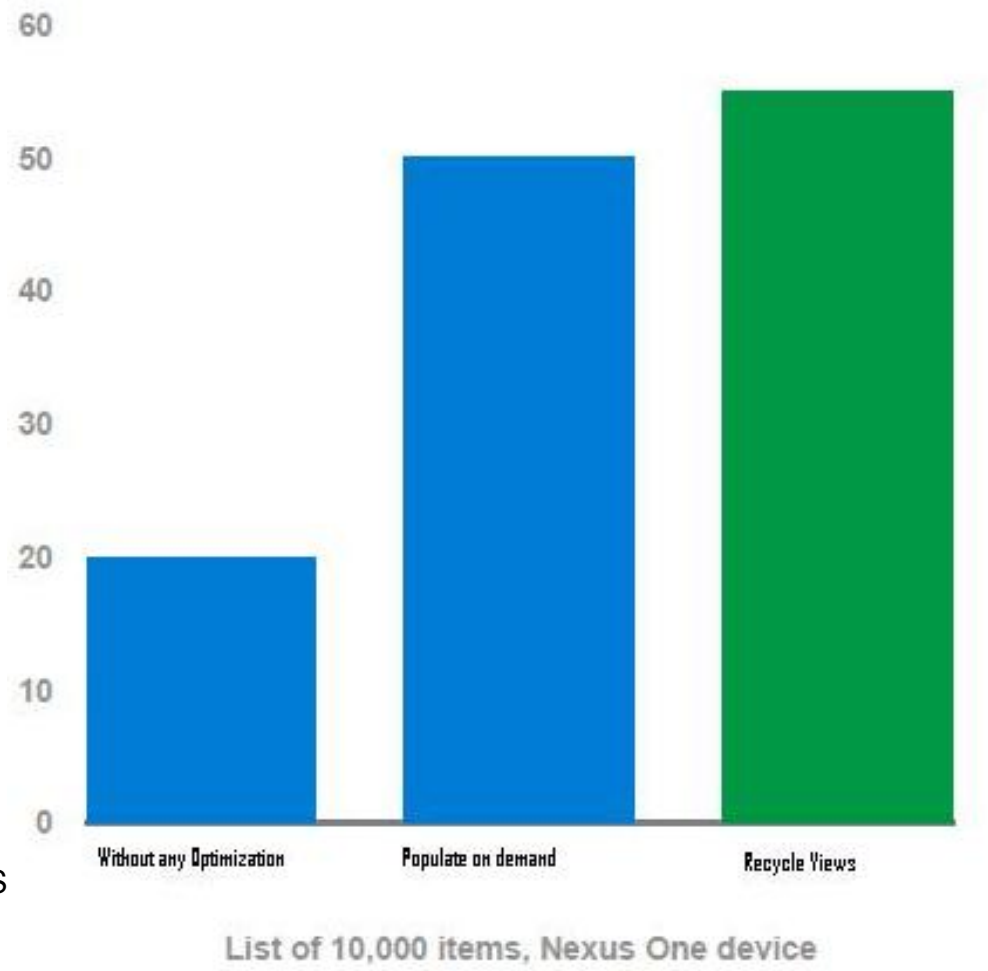
`getView()` vrací jednu položku Adaptéru

Výsledek vlastního ArrayAdapteru



Výsledky optimalizačních technik

- Řádek ListView obsahuje pouze LinearLayout s TextView a ImageView



Číselná hodnota na ose Y je v FPS
Zdroj: Google I/O 2010

Zobrazení různých vlastních layoutů položek

```
@Override
public View getView(int position, View convertView, ViewGroup parent) {
    ViewHolder holder;
    if (convertView == null) {
       if (getItemViewType(position) == 0) {
           convertView = inflator.inflate(R.layout.row_even, null);
       } else {
           convertView = inflator.inflate(R.layout.row_odd, null);
       }
       ViewHolder viewHolder = new ViewHolder();
       viewHolder.text =(TextView)
                        convertView.findViewById(R.id.TextView01);
       viewHolder.image = (ImageView)
                        convertView.findViewById(R.id.ImageView01);
       convertView.setTag(viewHolder);
    } else holder = (ViewHolder) convertView.getTag();
    holder.text.setText(getItem(position));
    return convertView;
}
```

Pozor při použití vlastního pozadí prvků

SimpleCursorAdapter

- Mapuje sloupce z Cursoru do jednotlivých View

Příklad

- Získání `Cursoru` kontaktů:

```
private Cursor getContacts() {
  Uri uri = ContactsContract.Contacts.CONTENT_URI;
  String[] projection = new String[] {
          ContactsContract.Contacts._ID,
          ContactsContract.Contacts.DISPLAY_NAME
  };
  String selection = ContactsContract.Contacts.IN_VISIBLE_GROUP
                     + " = '" + ("1") + "'";
  String[] selectionArgs = null;
  String sortOrder = ContactsContract.Contacts.DISPLAY_NAME
                     + " COLLATE LOCALIZED ASC";

  return managedQuery(uri, projection, selection, selectionArgs,
                  sortOrder);
}
```

Příklad SimpleCursorAdapteru - pokračování

```
@Override
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Cursor mCursor = getContacts();
        startManagingCursor(mCursor);
        ListAdapter adapter = new SimpleCursorAdapter(this,
                android.R.layout.two_line_list_item,
                mCursor,
                new String[] {
                        ContactsContract.Contacts._ID,
                        ContactsContract.Contacts.DISPLAY_NAME
                },
                new int[] {
                        android.R.id.text1, android.R.id.text2
                });

        setListAdapter(adapter);
}
```

Context

Pole potřebných sloupců

ID View odpovídajících daným sloupcům v layoutu položky

Aktualizace položek v Adaptéru

- `notifyDataSetChanged()`
  - Notifikace observeru, že došlo ke změně dat a zobrazená View se mají aktualizovat
  - Nastavený `List<T> objects` v konstruktoru Adaptéru nesmí být nahrazen jiným `List<T> objects`

- `cursor.requery()`
  - Lze volat nad Cursorem vloženým do `CursorAdapteru`

ChoiceMode v ListView

- `none`
- `singleChoice`
  - Pouze jedna položka stavu „chosen state“
- `multipleChoice`
  - Více položek stavu „chosen state“
  - Vypsání např. pomocí `getCheckedItemPosition()`
    - Vrací `SparseBooleanArray(`získání hodnoty pomocí `valueAt())`
- `multipleChoiceModal`
  - Více položek stavu „chosen state“ podle vlastního výběru
  - Implementací `AbsListView.MultiChoiceModeListener`
  - Za použití `setMultiChoiceModeListener(AbsListView.MultiChoiceModeListener)`

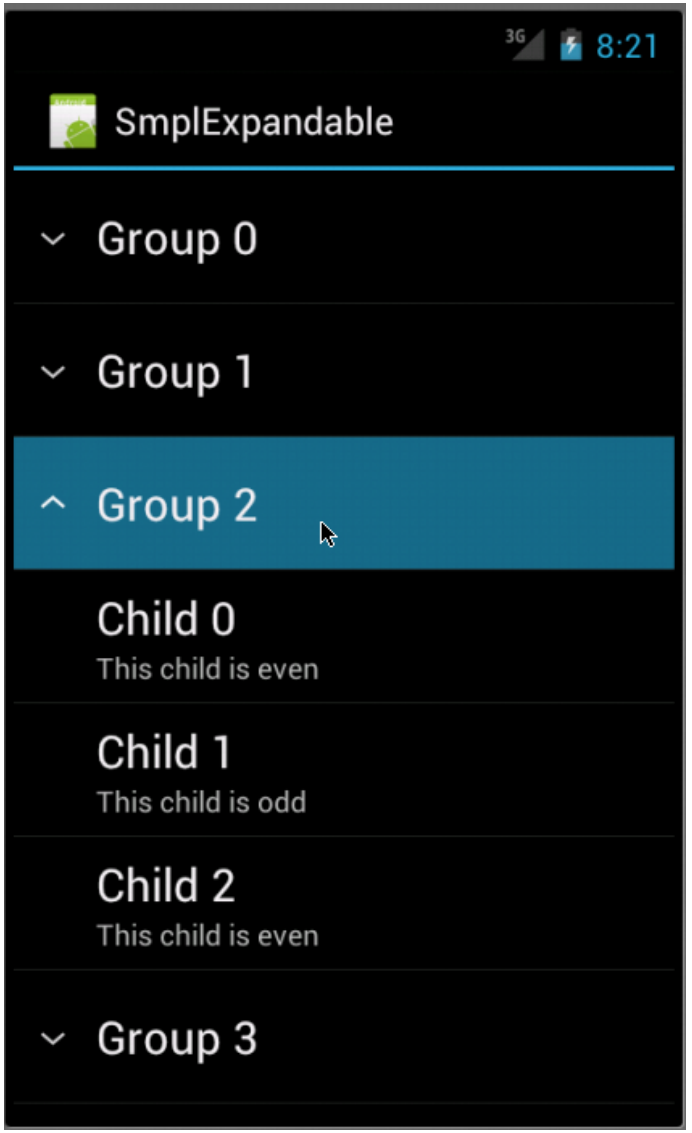
Filtrace ListView

- Adaptéru je možné doimplementovat `Filterable`
  - V metodě `getFilter()` je poté nutné vrátit vlastní Filter
  - `Filter` musí implementovat metody `performFiltering()` a `publishResults()`

- `ArrayAdapter`
  - Má již implementovaný vlastní `ArrayFilter`, který filtruje položky `List<T> objects` podle metody `toString()` objektu `T`
  - `ArrayFilter` je dobrou inspirací pro vlastní filtrování

Expandable ListView

- Dvou úrovňový seznam
  - Group
  - Children
- K naplnění se používá ExpandableListAdapter
- Zobrazuje stav u každé skupiny – expanded/collapsed
- Trochu obtížnější naplnění daty

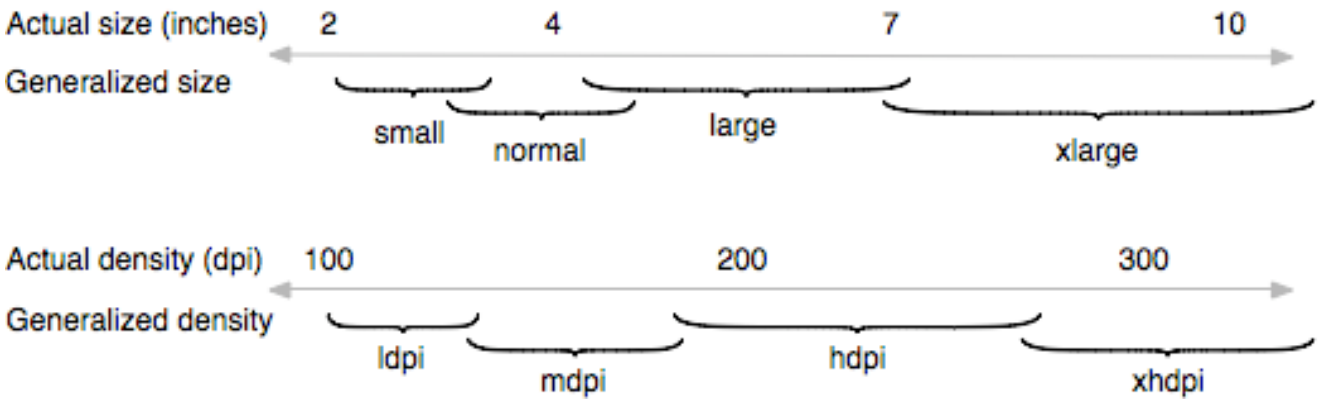


Parametry displeje

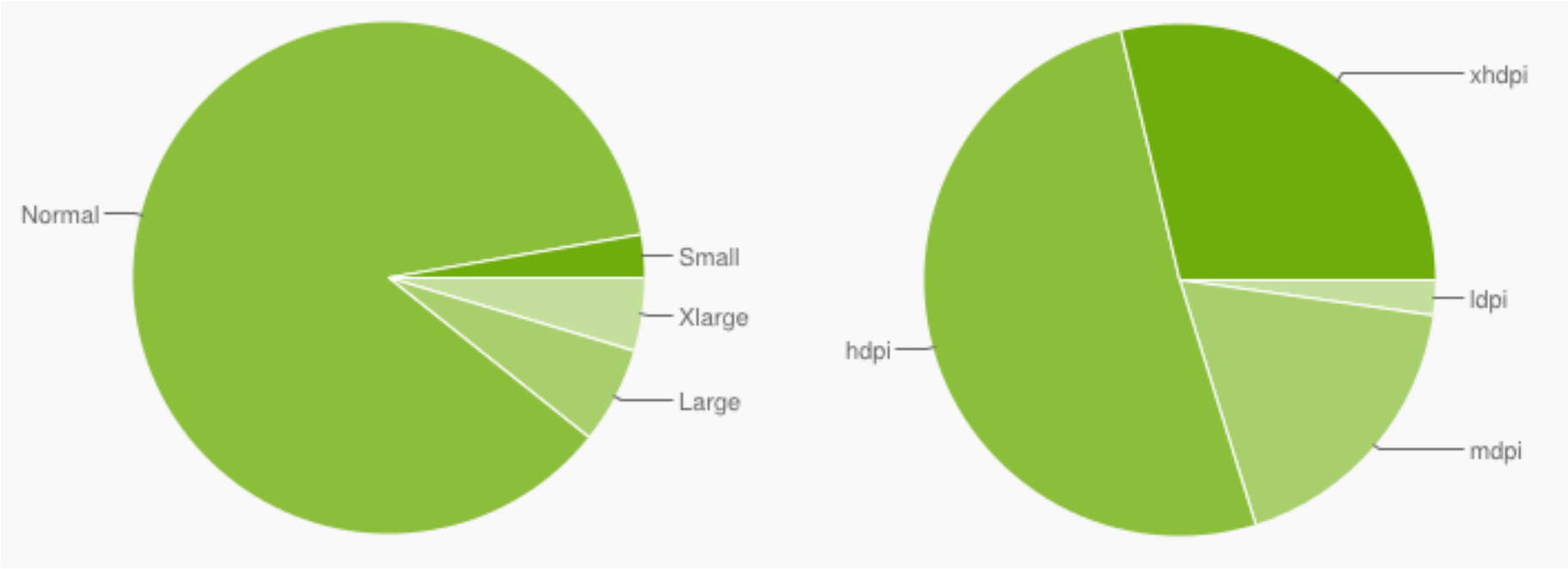
- Velikost displeje (Screen size)
  - Úhlopříčka displeje v palcích (inches – např. 3.7“)
- Hustota displeje (Screen density)
  - Počet pixelů na danou plochu displeje (dpi - dots per inch)
- Orientace
  - Na šířku (landscape) nebo na výšku (portrait)
  - Žádná orientace není defaultní

Parametry displeje

- Rozlišení (Resolution)
  - Počet fyzických pixelů na displeji (px - pixels)
  - Nepoužívá se při návrhu pro podporu více různých displejů
- Density independent pixel
  - Virtuální jednotka používaná pro pozicování a rozměry prvků nezávislé na dpi (dip, dp nebo dps)
  - sp, sip - obdoba dip pro velikost písma (lze změnit v systému)



Hustota a velikost displejů



Aktuální: http://developer.android.com/resources/dashboard/screens.html

Podpora různých displejů

- Explicitně stanovit v manifestu, jaké displeje jsou podporovány

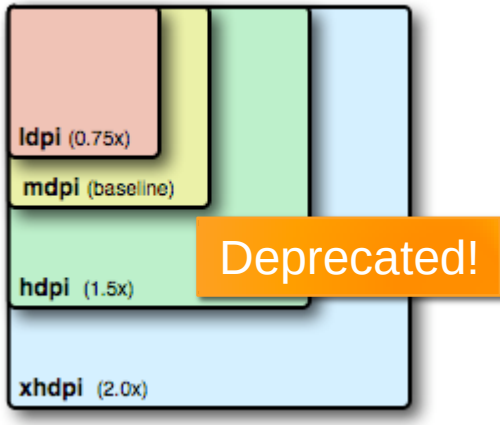
Zacílení pouze na některá zařízení?

- Poskytnout různé layouty pro různé displeje
- Poskytnout různé (bitmap) drawables pro různé displeje
- Používat Nine-patch bitmaps – 7. přednáška

Podpora různých displejů

- Používat `wrap_content` a `match_parent`
- Používat RelativeLayout
- Používat tzv. configuration qualifiers ve složce res
  - `<resources_name>-<qualifier>`
  - Qualifiers – hustota, orientace, rozměry displeje
  - Více viz. 2. přednáška

Podpora různých displejů



- Čtyři skupiny rozlišení jsou již deprecated
- Od verze 3.2 se používají parametry:
  - Smallest width
  - Available screen width
  - Availabe screen height
- Potřeba zahrnout i starší zařízení

- Přepočet density independent pixel na pixely

$$px = dp \times (dpi / 160)$$

- Např. na 240 dpi displeji, 1 dp odpovídá 1,5 fyzickému pixelu

Přepočty

- Dva různě kvalitní displeje, kde však při použití density independend pixelů nedojde ke změně ve vzhledu layoutů apod.

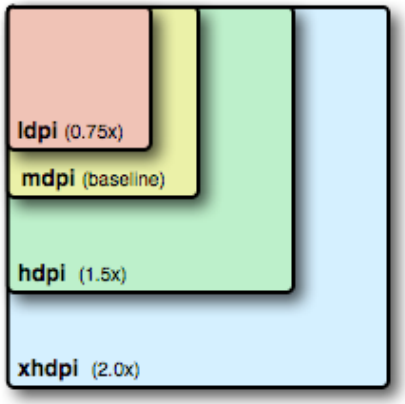
Parametry displeje	Displej s nižším rozl.	Displej s vyšším rozl.
Úhlopříčka (palce)	1,5	1,5
Hustota (dpi)	160	240
Pixely (= úhl × hust)	240	360
Hustota (faktor 160)	1.0	1.5
Density ind. pixels	240	240

Tvorba grafiky

- Spousta faktorů ovlivňuje grafickou podobu aplikace např.
  - Vlastnosti displeje (rozměry, rozlišení, technologie, orientace)
  - Verze OS (velikost nativních prvků)
  - Výbava zařízení (chybějící fotoaparát, GPS apod.)

- Postup (platí pro kreslení jednotlivých prvků)
  - Určit si podporované displeje
  - Grafiku tvořit pro nejvyšší podporované rozlišení a určit si rozměry prvku
  - Převést (viz. obrázek) na menší rozměry

Pohled ze strany grafika

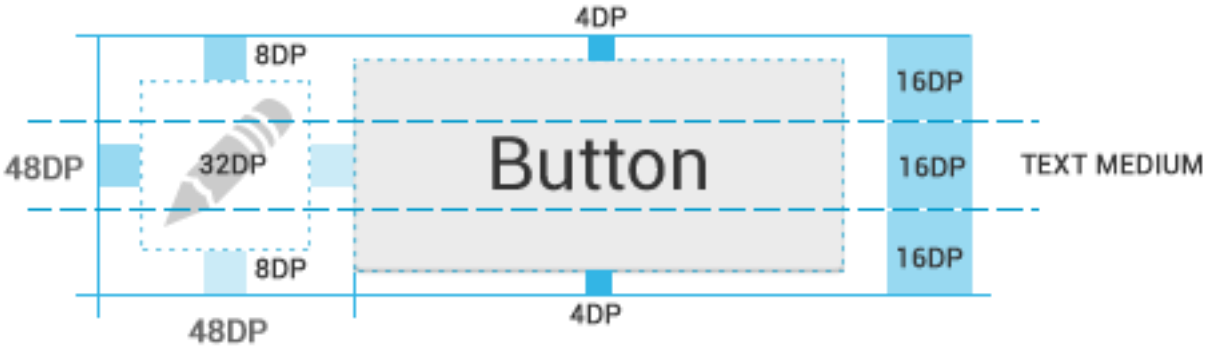


UI Guidelines

- Doporučené postupy ze stránek ADG
  http://developer.android.com/guide/practices/ui_guidelines/index.html
- UI Guidelines (zejména) pro Android 3.0+
  http://developer.android.com/design/index.html
- Následuje výběr několika guidelines

Velikost prvků

- Velikost „Stisknutelných“ prvků - 48dp
- Mezery mezi částmi UI - 8dp



- Proč 48dp?
  - Odpovídá cca 9mm – doporučení pro dotykové displeje 7–10mm

Styl psaní dialogů

Uživatel si spíše přečte krátkou hlášku než delší text

- Stručně – nastavit si limit na cca 40 znaků včetně mezer
- Jednoduše – používat známé pojmy
- Přátelsky – mluvit přímo k uživateli
- To nejdůležitější nejdříve – první tři slova alespoň náznak toho podstatného
- Popsat pouze to podstatné
- Neopakovat se – nepoužívat ten samý pojem několikrát

Návrh ikon

- Používat dostupné templaty
  - Nespoléhat se, že budou obsaženy v systému
- Používat ikony v různých rozlišeních
- Vektorová grafika dostupná pouze pomocí uživatelských knihoven
  - Neměla by se používat pro návrh ikon
- Pro zjednodušení si nechat vygenerovat ikony
  - Např. Android Asset Studio
  - http://android-ui-utils.googlecode.com/hg/asset-studio/dist/index.html

Návrh UI - Skica

- Většinou ručně kreslená
- Výhoda
  - Jednoduchost
- Nevýhoda
  - Špatně se překresluje
    - Možnost vystřihnout jednotlivé komponenty a znovu použít
  - Neodhadnutí velikosti prvků
    - Použít čtverečkovaný papír
- Lze použít šablony rozměrů displeje

Návrh UI - Wireframe

- Zaměřuje se na:
  - Funkcionalitu
  - Rozložení prvků
  - „Interakci“ jednotlivých prvků
  - Postrádá typografický styl, barvy nebo grafiku
- Např.: „Řádek se stavem uživatele bude nalevo pod jeho fotkou.“
- Wireframe Sketcher
  - http://wireframesketcher.com/

Wireframe je o funkci


Sketching - Android/Android.screen - WireframeSketcher Studio
File Edit View Object Format Window Help
Project Explorer
Android
Android.screen
NewScreen.screen
Properties
Links
Screen
Font:
<System Font>
Theme:
Clean
Default theme preferences...
Outline
Phone
Status Bar
Home Screen
Phone Landscape
Status Bar Landscape
Keyboard
Keyboard Landscape
Zoom Buttons
Message Dialog
Modal Screen
Menu Two Rows
Menu Single Row
Banner
Banner Left
Dropdown List
Options List
Popup Menu
*NewScreen.screen
Android.screen
Gmail
Calendar
Clock
Browser
Search
Notifications
Settings
Dialog Title
Message text
OK
Cancel
Add
Manage apps
Walpaper
Search
Notifications
More
Vibrate
Label
Label
Wireless & network settings
Airplane mode
Disable all wireless connections
Wi-Fi
Connected to network
Wi-Fi settings
Set up & manage wireless access points
Bluetooth
Turn on Bluetooth
Bluetooth settings
Manage connection, set device name & discoverability
Tethering & portable hotspot
Share your phone's mobile data connection via USB or as a portable Wi-Fi hotspot
Add bookmark
FInd on page
Select text
Page info
Share page
Downloads
Settings
Palette
Widgets
Containers
Input
Text
Layout
Annotations
Assets
Accordion
Arrow
Breadcrumbs
Browser
Button
Button Bar
Callout
Circle
Color Picker
Combo
Component
Cover Flow
Cross Out
Curly Brace
Group

Návrh UI - Mockup

- Zaměřuje se na:
  - Look and feel
  - Staví na wireframe a přidává barvy a grafiku
  - Může upravit rozložení prvků, ale vychází především z wireframe
- Např.: „Řádek se stavem uživatele bude dvakrát menší než fotka a bude zabírat maximálně čtvrtinu řádky.“
- Balsamiq Mockup
  - http://www.balsamiq.com/products/mockups

Mockup je o formě

Návrh UI - Prototyp

- Ukazuje „finální“ design a jeho funkcionalitu
- Přichází po ujasnění požadavků a omezení z předchozích fází
- Může obsahovat různě velkou míru implementace
  - Stále se však jedná o „artefakt“
- Stále se nemusí jednat o finální podobu designu

Další zdroje

- http://android-developers.blogspot.com/2009/01/why-is-my-list-black-android.html
- http://androidniceties.tumblr.com/
- http://developer.android.com/guide/practices/design/accessibility.html
- http://code.google.com/p/android-ui-utils
- http://pencil.evolus.vn