Android alkalmazásfejlesztés

Felhasználói felület megismerése, különböző felbontások támogatása

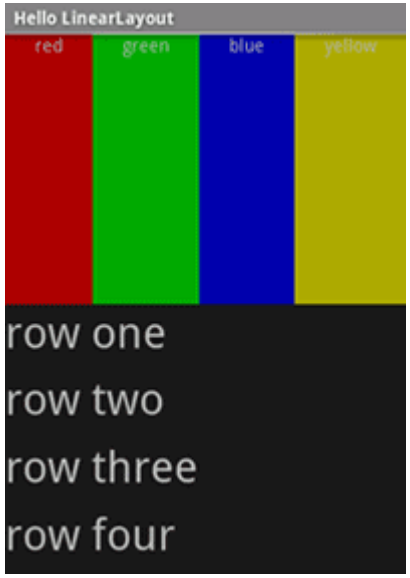
OE-NIK

2012. szeptember 18.

Sicz-Mesziár János

sicz-mesziar.janos@nik.uni-obuda.hu

Layout-ok megismerése



LinearLayout

- UI Elemek egymás után.
- Orientáció iránya beállítható.
- Súly értékekkel befolyásolhatjuk az eloszlás nagyságát.

RelativeLayout

- Többi elemhez képest adhatjuk meg pozíciót.
- Hatékonyan lehet vele dolgozni.

TabLayout

- Gyökér: TabHost
- Fül: TabWidget + FrameLayout
- Csak TabActivity-vel használható.

Grafikus editorban bug-os

FrameLayout

- Minden gyermek a bal felső sarokhoz igazodik

~~**TableLayout**~~
~~**AbsoluteLayout**~~

GridLayout

- Tetszőleges, mozaik szerű elrendezés
- Android 4.0-tól
- != GridView!
- Bővebben itt!

UI elemek

- Számtalan Widget és View:
  http://developer.android.com/resources/tutorials/views/index.html
- View leszármazottai.
- Sok-sok tulajdonsággal rendelkeznek. *(nem részletezzük)*
- Szélesség és magasság megadása mindig kötelező.
  `layout_width, layout_height`
- Az UI elemeket egyedi azonosítóval érjük el:
  `android:id="@+id/gomb"`
- Egy UI elem elérése Java kódban:

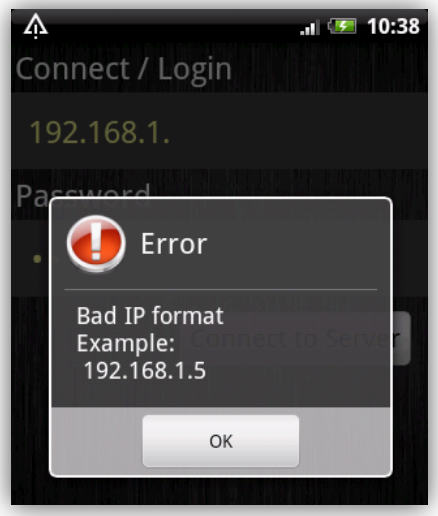
```
Button b = (Button)findViewById(R.id.gomb);
```

Ha R.id.gomb nem látható → Build!

- **Definiálhatóak egyedi felületek is!**
- Szöveges feliratok, értékek, színek, … lehetnek.
  - Beégetettek (android:text="Gombocska")
  - Külső XML-ben tároltak (android:text="@string/gomb_title")

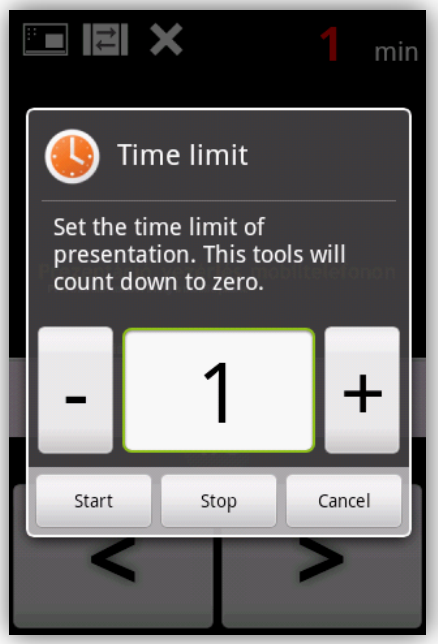
AlertDialog és ProgressDialog

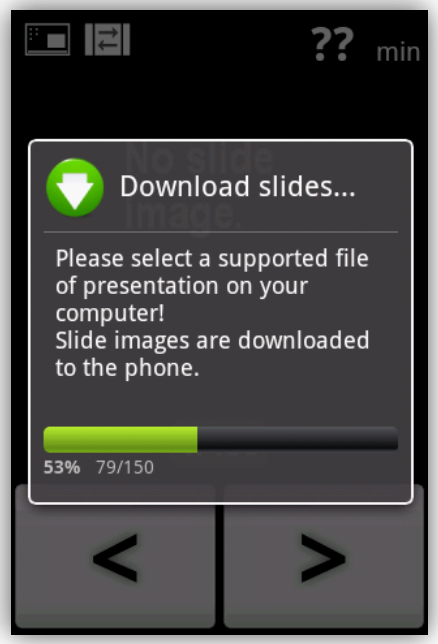
- Egy párbeszédablak, ami az aktuális Activity előtt jelenik meg.
- Dialog osztály leszármazottja.
- Testreszabhatjuk (Ikon, szöveg, gombok)
- XML fájllal egyéni felületet adhatunk hozzá



- Beépített gombok:

1. Pozitív *(PositiveButton)*
2. Negatív *(NegativeButton)*
3. Semleges *(NeutralButton)*





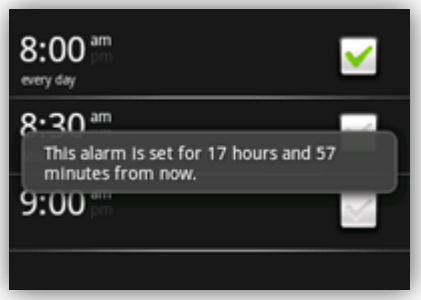
AlertDialog példa:
http://developer.android.com/guide/topics/ui/dialogs.html#AlertDialog

ProgressDialog példa:
http://developer.android.com/guide/topics/ui/dialogs.html#ProgressDialog

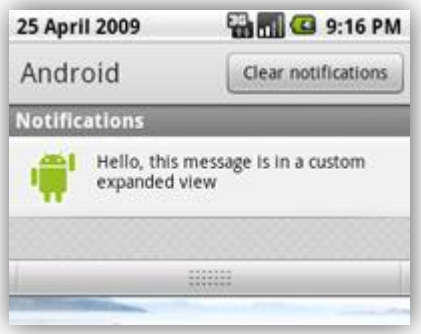
Toast, notify

- Toast

```
Toast toast = Toast.makeText(getApplicationContext(),
„Buborék”, Toast.LENGTH_SHORT);
toast.show();
```



- Status Bar notifications
  Lényegében egy értesítési terület
  Jelly Bean: pár újítás – pl.: expand



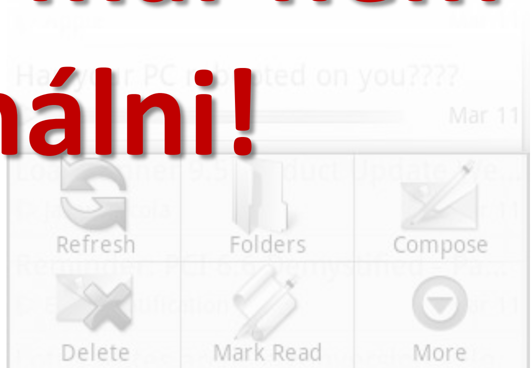
- Két típus jellemző
  - Notifications: Egyszeri értesítés, „Clear” gomb hatására törölhető
  - Ongoing: folyamatban lévőekről értesítés Pl.: zenelejátszó. Nem törölhető.
- Testreszabható
  Ikon, leírás, elrendezés, értesítés módjai, esemény, …

Bővebben:
http://developer.android.com/guide/topics/ui/notifiers/notifications.html

Menü (Options menu)

- Képernyő alján megjelenő opciók. Több, mint 6 elem esetén a 6. elem „More" lesz és mögötte található a maradék menüpont.

1. Menüelemek definiálása XML-ben (res/menu/*.xml)
   - <menu>
   - <item
     android:id
     android:icon
     android:title >



2. onCreateOptionsMenu() metódus felülírása. (Csak első alkalommal hívódik meg)
3. onOptionsItemSelected() metódus felülírása.

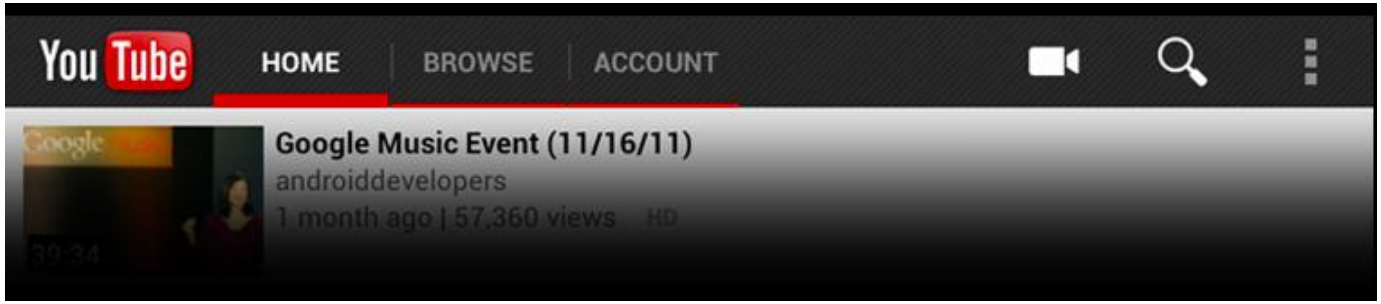
Szoftver ergonómiai szempontok miatt már nem ajánlott használni!

Android 3.0-tól ActionBar!

http://developer.android.com/guide/topics/ui/menus.html#options-menu

ActionBar

- Android 3.0 (API level 11) óta
- Többletes funkciók
  - Kitüntetett hely az alkalmazás nevének és ikonjának – brand
  - Navigáció: beépített Tab kontroll a Fragment váltásokhoz
  - Fontos funkciók (Pl.: keresés, megosztás, létrehozás) megszokott helyen



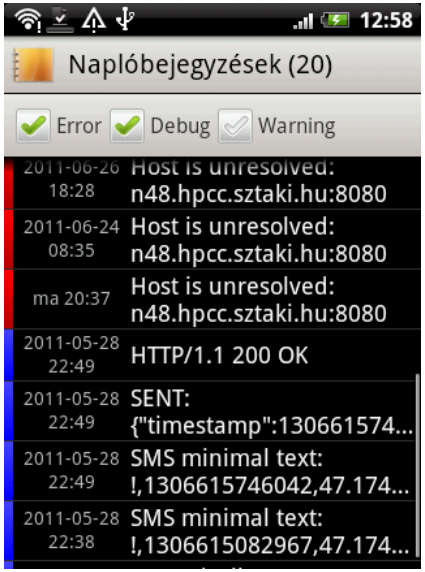
További részletek: http://developer.android.com/guide/topics/ui/actionbar.html

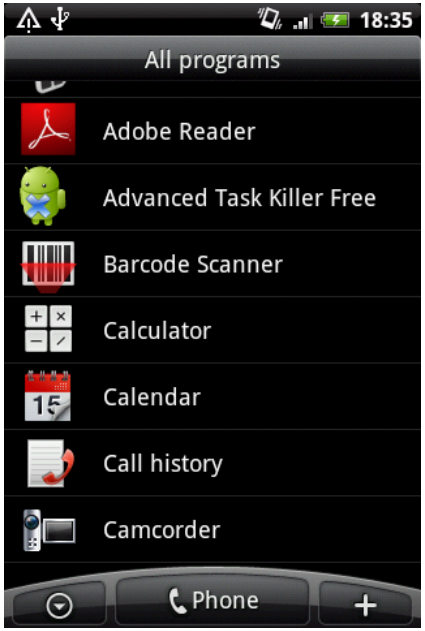
Néhány XML leíró bemutatása

- Layer list
  Több kép együttes rétegkezelése.

- State list: egy állapotlista
  Állapotokhoz kapcsolt képek meghatározása.
  Pl.: Focus, Press
- Level list
  Különböző szintekhez képek kapcsolása.
  Pl.: Fényerő (%), Wi-Fi jelerősségek (dB)
- Shape drawable: alakzatok leírása XML-ben
  Pl.: alak meghatározása, sarkok lekerekítése, színátmenetek, margók, méret, …

ListView, Gallery, Spinner, GridView, …





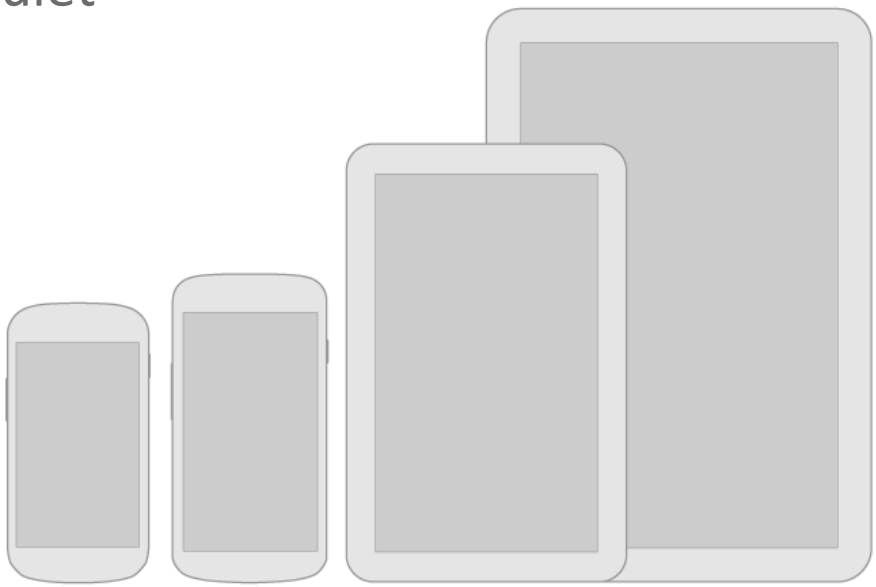
- Az adatokat egy adapteren keresztül biztosítjuk az UI számára.
  - Előre implementált adapter (Pl.: ArrayAdapter)
  - Mi implementáljuk (BaseAdapter leszármazott)
- A módszer előnyei:
  - Az adatok tárolási módja nem meghatározott
    Pl.: SD kártya, SQLite adatbázis, internet, …
    Vagy: lista, tömb, hashmap, …
  - UI szétválasztva az adattól
  - Optimális erőforrás felhasználás
  - Nagy adatmennyiség kezelése
    (akár >10E listaelem kezelése)

ListView a legnépszerűbb UI elem!

Példát később nézünk.

Felhasználó barát felület

- Fő iránymutató példák
  - http://developer.android.com/design/
- További ajánlások
  - Letisztult, minimalista felület
    - csak a lényegi információ,
    - kiemelések,
    - színek használata
  - Ikonok használata, mert:
    - nem kell olvasni,
    - több nyelven is érthető,
    - kevesebb hely, …
  - Másokra is gondolni
    - színvakok: screenshot → grayscale,
    - idősek: lassú, remegő kezek → nagy gombok, nincs longclick,
    - Bal kezesek
  - Felbontások: 240x320 ---> 320x480 ---> 480x800 ---> …



Új Activity hozzáadása

1. Új layout XML létrehozása
2. Új Java osztály hozzáadása
   - Legyen az Activity leszármazottja: extends Activity
   - onCreate() metódus implementálása
   - setContentView() segítségével új layout XML beállítása
3. AndroidManifest.xml fájlban Activity hozzáadása

- Egy új Activity indítása:

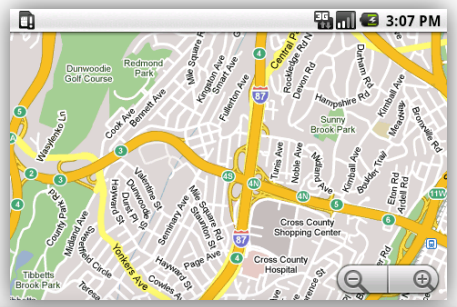
```
Intent masikActivity = new Intent(this, Masik.class);
newActivity.putExtra(„szam”, 100); // adat átadása
startActivity(newActivity);
```

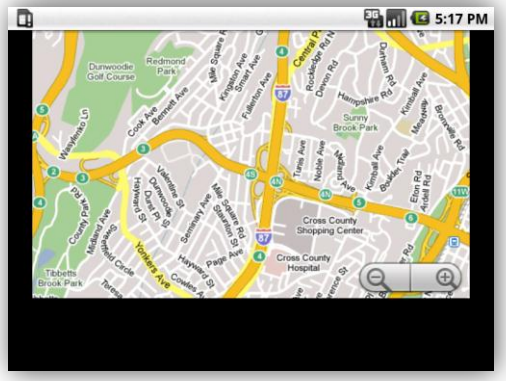
- Másik activity-ben átadott adat olvasása

```
int szam = getIntent().getExtras().getInt(„szam”, 0);
```

Különböző felbontás támogatása

- Problémák





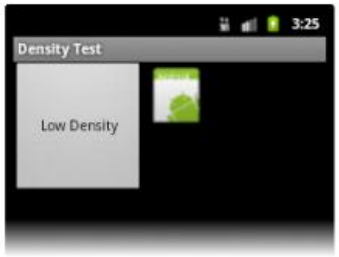
240 dpi, landscape

160 dpi (medium)

240 dpi (high)

API Level 6+ Launcher Icon

API Level 4+ Launcher Icon



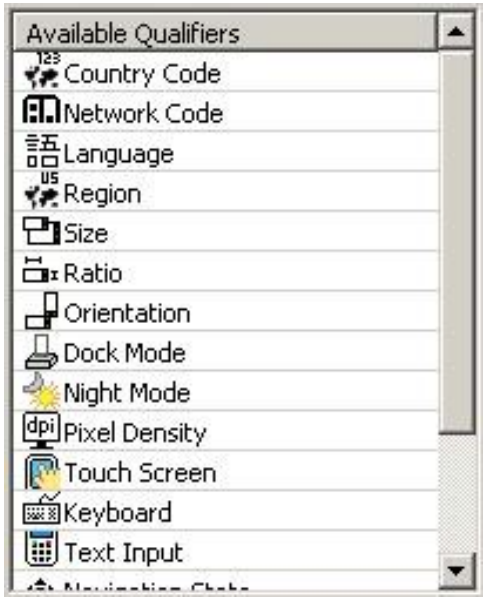
Források

- http://developer.android.com/resources/samples/MultiResolution/index.html
- http://developer.android.com/guide/practices/screens_support.html

Minősítők (qualifiers) használata

- Különböző esetekre különböző megközelítés szükséges.
- Sok IF és SWITCH helyett minősítőket definiáltak.
- Kiértékelésük automatikus, éppen jellemző könyvtárból dolgozik mindig.

- Néhány minősítő:



- Gyakorlatban:

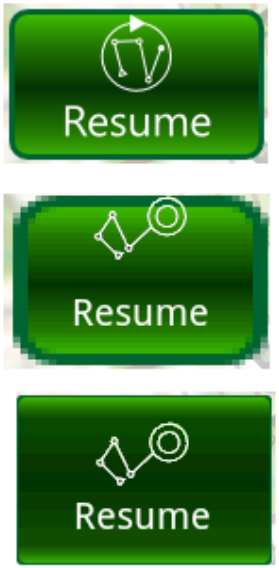
Ugyanaz a kép 3 különböző méretben

Ugyanazon layout XML különböző leírása „portrai„ és „landscape” módban

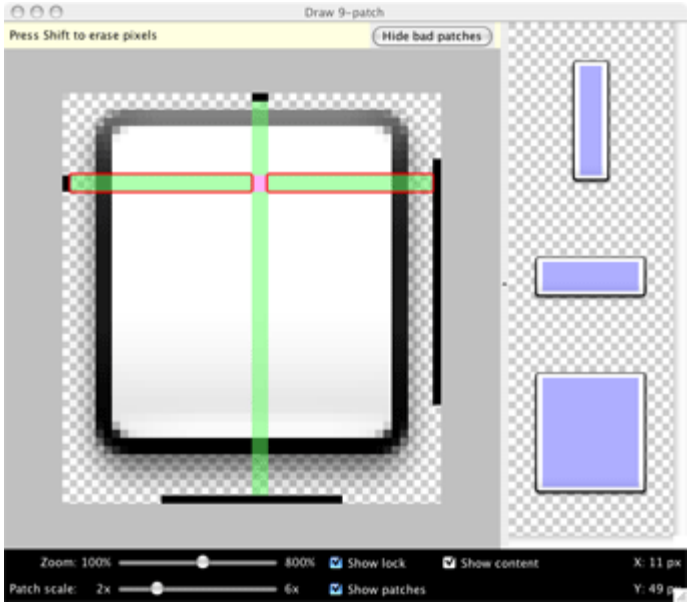
A minősítők kombinálhatóak! ☺

9-patch

- Dinamikus tartalom esetén, ha egyéni hátteret használunk akkor az eltorzulhat. Pl.: egy gomb esetén
- Speciális PNG fájllal meghatározhatjuk mely részeket nyújthatjuk meg a mintaképen. (bal oldal és fent) Szélén fekete pixelekkel jelöljük meg ezt a nyújtható területet.
- Jobb oldalt és lent a kitöltési területet jelölhetjük meg.
- Mindig ***.9.png** kiterjesztésű
- Van hozzá eszköz: [SDK path] / tools / draw9patch.bat



Not Scaled	Scaled Horizontally Only	Not Scaled
Scaled Vertically Only	Scaled Horizontally and Vertically	Scaled Vertically Only
Not Scaled	Scaled Horizontally Only	Not Scaled



Szép példa: http://radleymarx.com/blog/simple-guide-to-9-patch/

Alternatív mértékegység

- DP vagy DIP (**D**ensity-**I**ndependent **P**ixel)
  Egy virtuális pixel-egység, sűrűség-független képpont.
  160dpi felbontású készüléken = 1 képpont.
  Eltérő felbontás esetén automatikusan átváltja az alábbi módon:

```
pixels = dips * (density / 160)
```

  - Példa:
    160dpi felbontás esetén, 10dp = 10px Ezt tekintik alapnak!
    240dpi felbontás esetén, 10dp = 15px

Gondoskodik arról, hogy az ikonok közel egyméretűek legyenek különböző felbontású készülékeken.

- SP vagy SIP (**S**cale-**I**ndependent **P**ixel)
  Mint a DP, csak szöveg esetén használatos.

```
android:textSize="16sp"
```

Használati engedély kérése

Különböző tevékenységekhez, hardverek eléréséhez használati engedélyeket kell kérni

1. AndroidManifest.xml > Permissions > Add > Uses permission:
   `android.permission.VIBRATE`
2. Java fájlban service elérése, példa:

```
Vibrator vib =
(Vibrator)getSystemService(VIBRATOR_SERVICE);
vib.vibrate(500); //500ms rezgetés
```