Android

Animációk, Stílusok, Fragmentek és menü kezelés

Ekler Péter
peter.ekler@aut.bme.hu

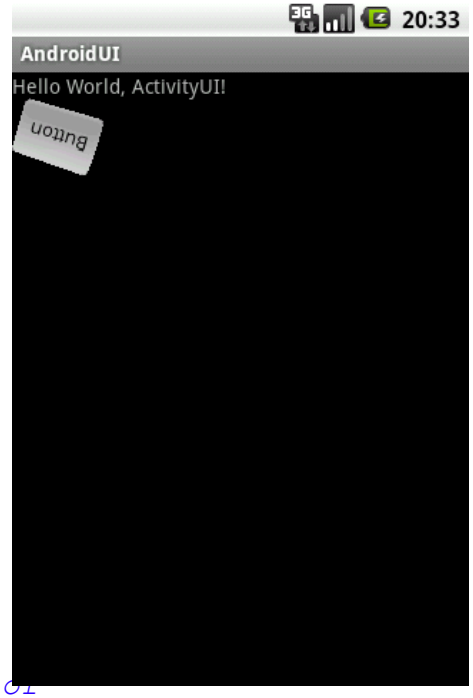
ANIMÁCIÓK

Animációk

- Animációk támogatása
  - XML erőforrás (*res/anim*)
  - Programkód
- Layout animáció
  - *Scale*
  - *Rotate*
  - *Translate*
  - *Alpha*
- Három fő típus:
  - Tween animáció
  - Frame animáció
  - Property animator

Tween animáció erőforrás

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:shareInterpolator="false">
    <scale
        android:interpolator=
          "@android:anim/accelerate_interpolator"
        android:fromXScale="0.0"
        android:toXScale="1.0"
        android:fromYScale="0.0"
        android:toYScale="1.0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="1000" />

    <alpha android:fromAlpha="0.0"
        android:toAlpha="1.0"
        android:duration="5000"/>

    <rotate
        android:interpolator="@android:anim/accelerate_interpolator"
        android:fromDegrees="0.0„
        android:toDegrees="360"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="5000" />
</set>
```



Tween animáció lejátszása

```
@Override
public void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.main);


  Button btn = (Button)findViewById(R.id.button1);
  btn.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
       Toast.makeText(ActivityUI.this,
         getResources().getString(R.string.toast_info),
           Toast.LENGTH_LONG).show();
    }
  });


  Animation showAnim = AnimationUtils.loadAnimation(this,R.anim.btnanim);
  btn.startAnimation(showAnim);
}
```

STÍLUSOK, TÉMÁK

Stílusok készítése

- Stílus file: res/values/styles.xml

```
<style name="ExampleStyle">
    <item name="android:textSize">22sp</item>
    <item name="android:textColor">#0000EE</item>
</style>
```

- Stílus használata:

```
<TextView
    android:id="@+id/tvHello"
    android:text="@string/hello_world"
    style="@style/ExampleStyle" />
```

Saját témák használata

- Stílusokhoz hasonlóan definiálhatók:

```
<style name="CustomTheme" parent="android:Theme">
  <item name="android:windowTitleSize">50dip</item>
  <item name="android:textColor">#000000</item>
  <item name=
    "android:windowBackground">@color/white_color</item>
</style>
```

- A témák öröklődhetnek egymásból
- Manifestben állítható be:

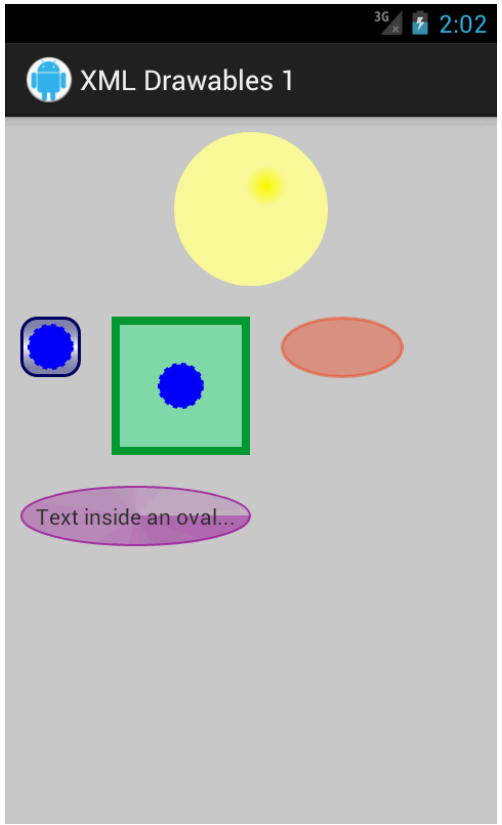
```
<activity android:label="" android:name=".MainActivity"
    android:screenOrientation="portrait"
    android:theme="@style/CustomTheme"/>
```

GRAFIKAI ERŐFORRÁSOK XML-BEN

Grafikai erőforrások 1/2

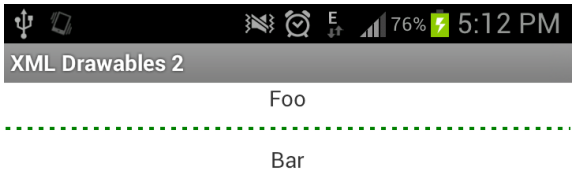
- Grafikai elemek, hátterek is leírhatók XML-ben

```
<?xml version="1.0" encoding="utf-8"?>
<shape
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:shape="oval">

  <gradient
    android:type="radial"
    android:gradientRadius="20"
    android:centerX=".6"
    android:centerY=".35"
    android:startColor="#FFFF00"
    android:endColor="#FFFF99" />

  <size
    android:width="100dp"
    android:height="100dp"/>
</shape>
```



Grafikai erőforrások 2/2

- Zöld szaggatott vonal:




```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android=
  "http://schemas.android.com/apk/res/android"
    android:shape="line">
    <stroke
        android:width="2dp"
        android:color="#008000"
        android:dashWidth="3dp"
        android:dashGap="4dp"/>

    <size android:height="20dp" />
</shape>
```

Állapotok leírása grafikai erőforrásokban

- Például gombfelirat színe:
  - res/color/button_text.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true"
        android:color="#ffff0000"/> <!-- pressed -->
    <item android:state_focused="true"
        android:color="#ff0000ff"/> <!-- focused -->
    <item android:color="#ff000000"/> <!-- default -->
</selector>
```

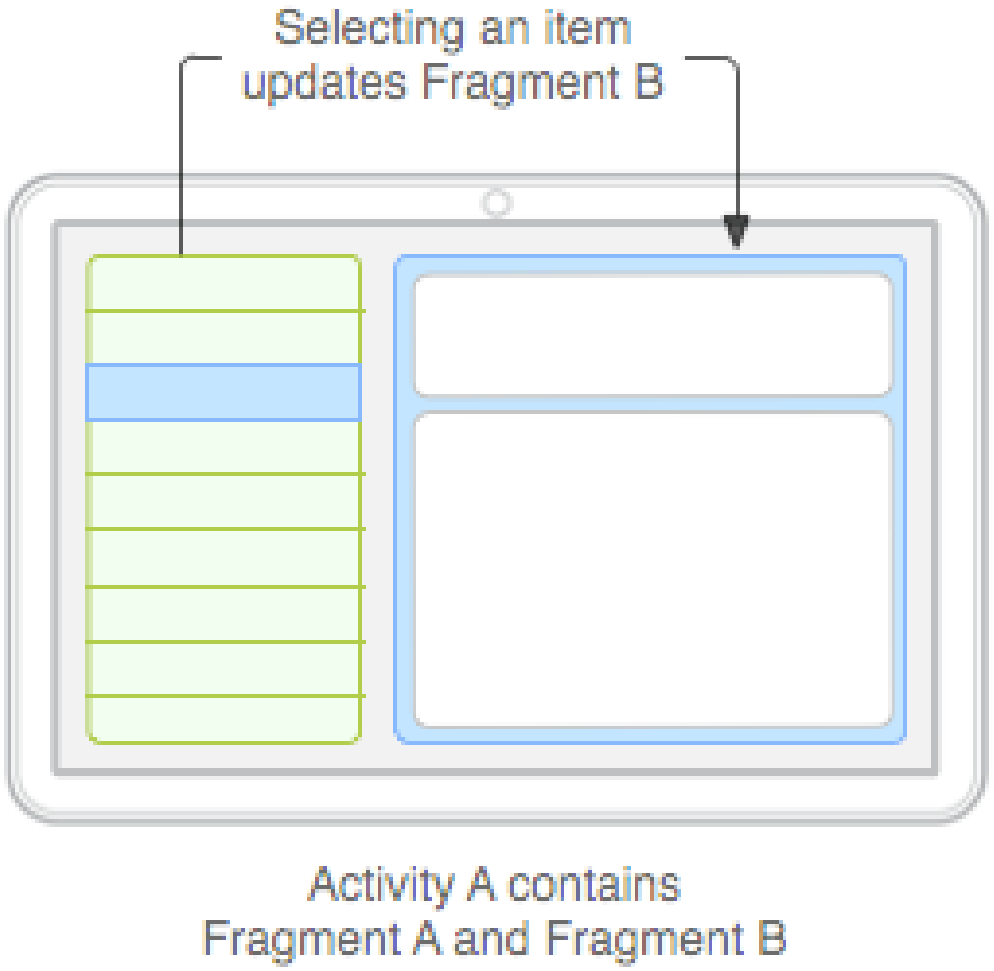
- Használat:

```
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/button_text"
    android:textColor="@color/button_text" />
```

Fragmentek

- Mik azok a Fragmentek?
  - Elsősorban: A képernyő egy nagyobb részéért felelős objektumok
  - Továbbá: A háttérben munkát végző objektumok is lehetnek
- Miért kellenek nekünk?
  - Nagy képernyőméret = több funkció egy képernyőn = bonyolultabb Activity-k
  - Fragment-ekkel modulárisabb, rugalmasabb architektúra építhető

Fragmentek


Selecting an item
updates Fragment B
Activity A contains
Fragment A and Fragment B

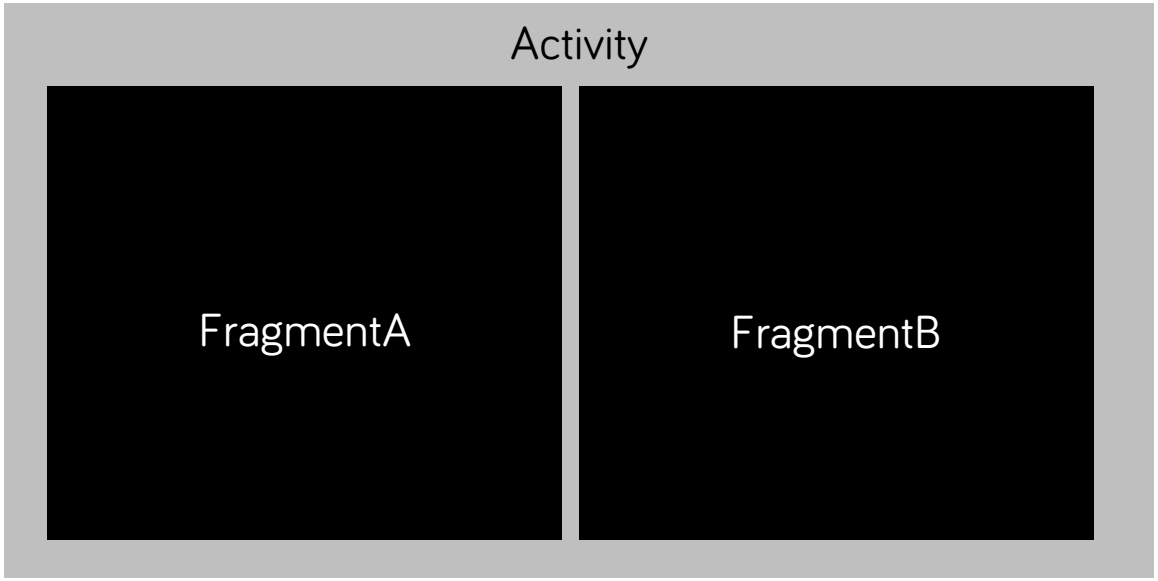
Fragmentek

- Miben másabb mint az Activity?
  - Kisebb granualitás, nem mindig teljes képernyő egy fragment
  - Az életciklusa nem mindig egyezik, pl. le lehet csatolni egy fragmentet úgy, hogy az activity előtérben marad.

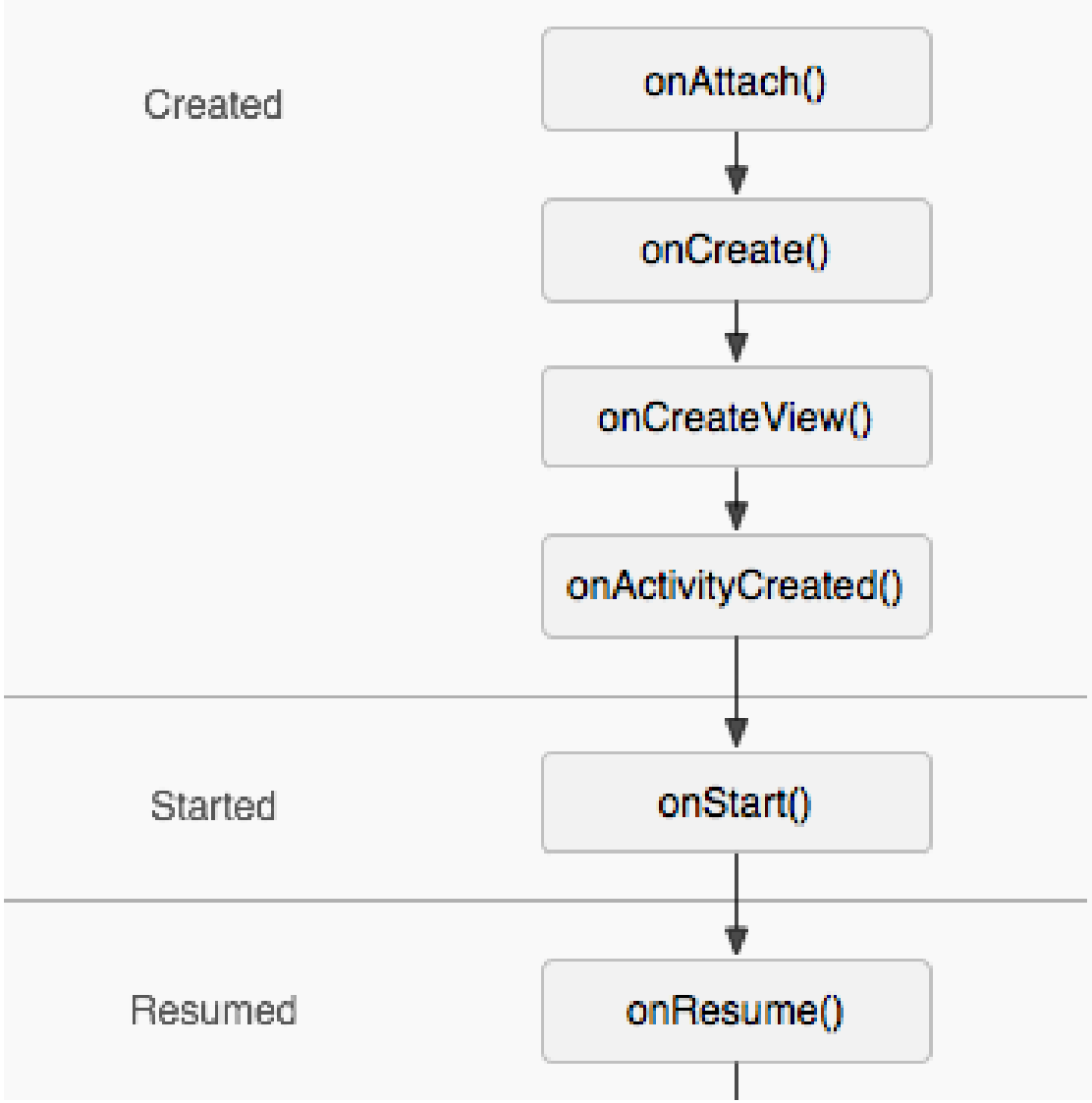
- Miben másabb mint egy Custom View
  - Összetett életciklus, mely az activity-t is figyelembe veszi
  - Előny, de hátrány is lehet!

Fragment és Activity

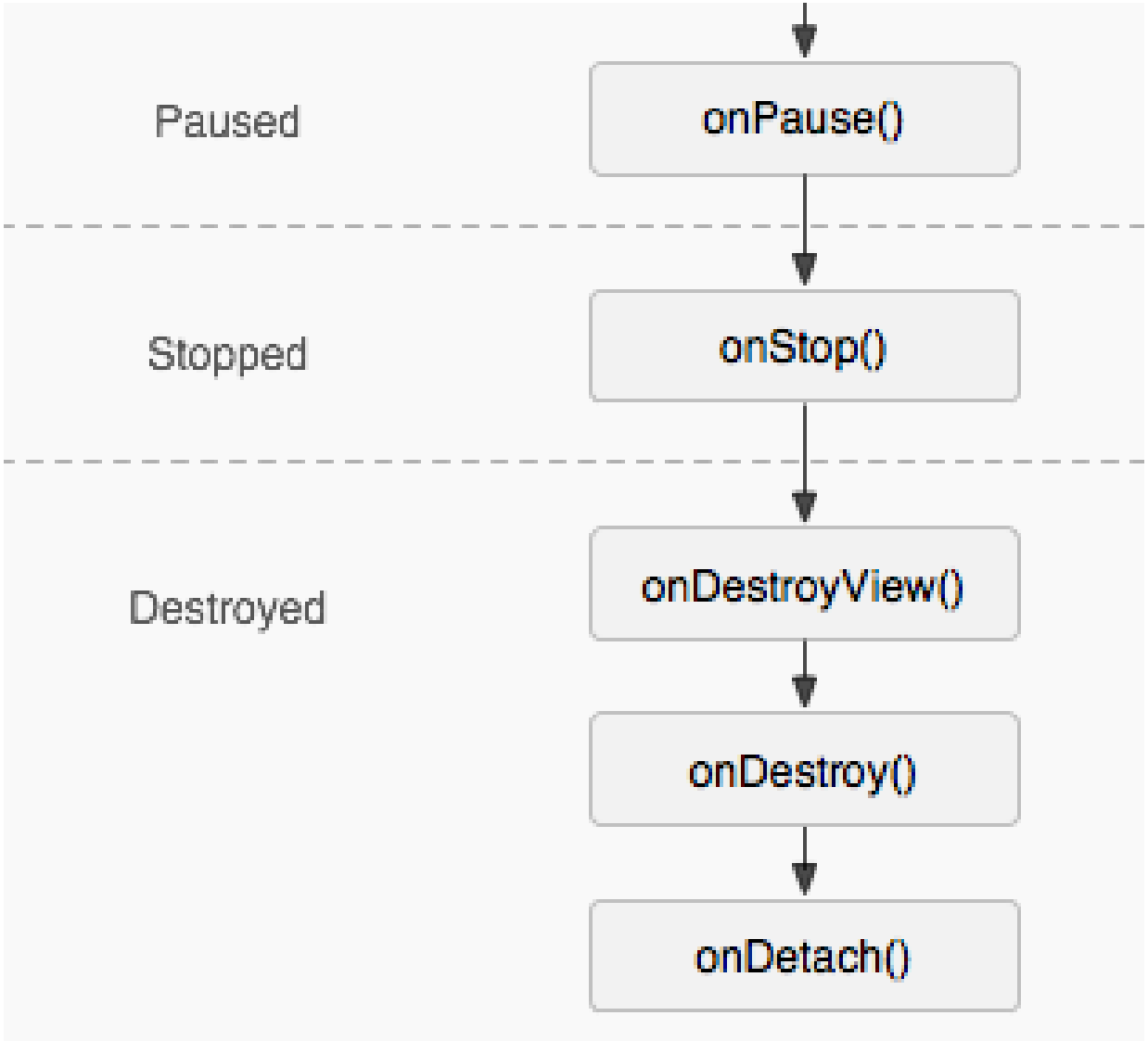
- Egy Fragment mindig egy Activity-hez csatoltan jelenik meg
- Az Activity életciklusa ráhatással van a Fragmentére

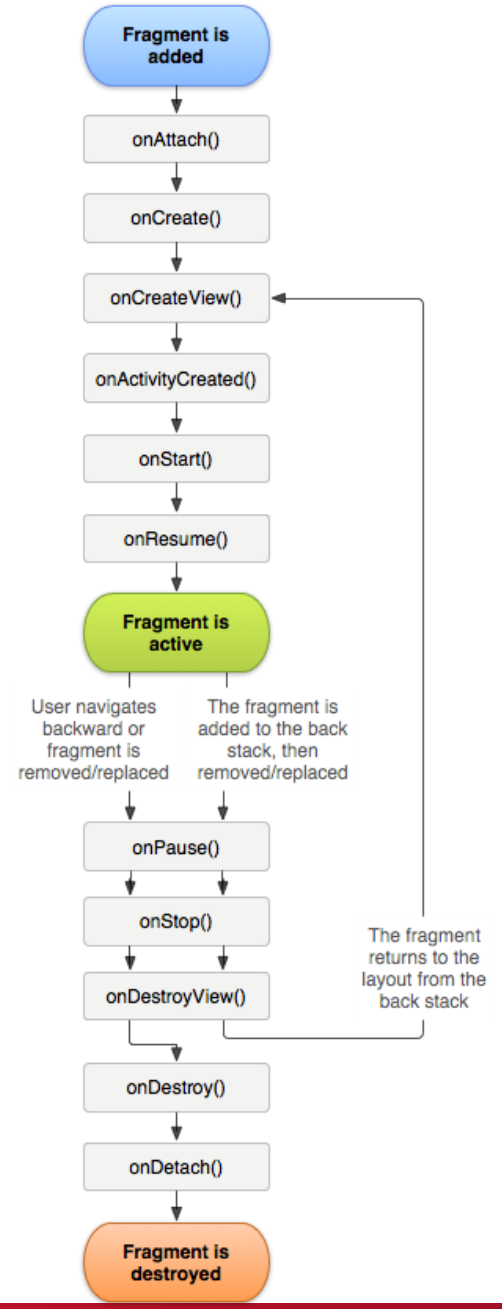


Fragment életciklus I.



Fragment életciklus II.




Fragment is added
onAttach()
onCreate()
onCreateView()
onActivityCreated()
onStart()
onResume()
Fragment is active
User navigates backward or fragment is removed/replaced
The fragment is added to the back stack, then removed/replaced
onPause()
onStop()
onDestroyView()
The fragment returns to the layout from the back stack
onDestroy()
onDetach()
Fragment is destroyed

Support Library

- Fragment API az Android API része Android 3.0 óta
- Support Library – Fragmentek Android 3.0 előtt
- Ma már 3.0 alá nem is targetáluk, miért használjuk?
  - Android verzionként eltér a beépített Fragment api (~~android.app.Fragment~~)
  - A support környtárral minden Android verzión ugyan azt kapjuk (android.support.v4.app.Fragment)

Nem alkalmazás komponens

- Mi hozzuk létre, nem a rendszer
- Nem kell feltüntetni a manifestben
- Nem intentekkel kommunikálunk
  - Mezei függvényhívás az objektumon
  - pl. Activity hívja a Fragment objetumon
  - getActivity()/activity property – szülő activity

UI Fragment készítése...

- A megjelenítendő View-hierarchiát az .onCreateView() metódusban kell visszaadni

```
class FragmentProfile : Fragment() {

    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?,
savedInstanceState: Bundle?): View? {
        val rootView = inflater.inflate(R.layout.fragment_profile, container, false)
        return rootView
    }

}
```

… és csatolása

- Statikusan
  - Az Activity-hez tartozó layout-ban beégetjük a Fragment-et, nem módosítható később
  - <fragment …/> tag
- Dinamikusan
  - Az Activity futás közben tölti be a megfelelő Fragment-eket, adott ViewGroup-okba
  - Fragment-Tranzakciókkal módosítható

Statikus csatolás példa

```
<fragment class="hu.bme.aut.fragment.MenuListFragment"
    android:tag="MenuListFragment"
    android:layout_width="0dip"
    android:layout_height="fill_parent"
    android:layout_weight="1"/>
```

A FragmentManager

- A FragmentManager-el menedzselhetők a Fragment-ek
  - Activity: supportFragmentManager property
  - FragmentTransaction indítása
  - Aktív Fragment-ek közt keres
    - Tag alapján
    - ID alapján
  - **Fragment-stack-et menedzseli**

A FragmentManager

- Az activitynek a FragmentActivityből kell származnia – Neki van FragmentManagerje
- Activity.supportFragmentManager== fragment.fragmentManager
- Kezeli a fragmenteket, backstacek, állapotmentést … stb.
- Fragmenten belül fragmentek, lehet: Fragment.childFragmentManager

FragmentTransaction osztály I.

- Ezen keresztül módosíthatók az aktív Fragment-ek
- A FragmentManager .beginTransaction() metódusával indítható
- Fontosabb műveletek:
  - .add(...) / .remove(...) / .replace(...)
    - Fragment példányok le- és felcsatolása az adott Activity-re
  - .commit()
    - Tranzakció végrehajtása

FragmentTransaction osztály II.

- .show(...) / .hide(...)
  - Fragment példány elrejtése / újra megjelenítése
- .setTransition(...) / .setCustomAnimations(...)
  - A tranzakció végrehajtásakor lejátszandó animáció beállítása
- .addToBackStack(...)
  - Rákerüljön-e a FragmentTransaction backstack-re a tranzakció?
- .commit()
  - Tranzakció végrehajtása

FragmentTransaction példa I.

- Fragment kicserélése:

```
val fragment=DetailsFragment.newInstance()

val ft = supportFragmentManager.beginTransaction()
ft.replace(R.id.fragmentContainer, fragment, DetailsFragment.TAG)
ft.commit()
```

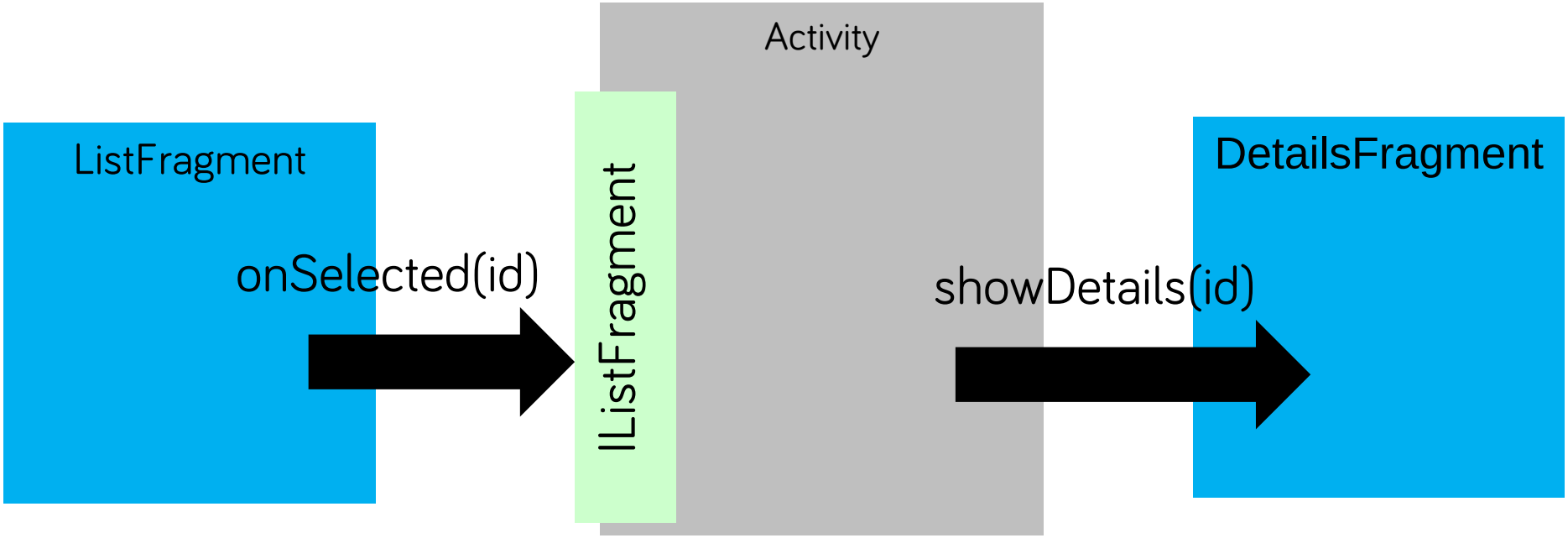
FragmentTransaction példa II.

- Fragment hozzáadása, a tranzakciót a backstack-re téve:

```
val fragment=DetailsFragment.newInstance()

val ft = supportFragmentManager.beginTransaction()
ft.add(R.id.fragmentContainer, fragment, TAG)
ft.setCustomAnimations(R.anim.slide_in_top,R.anim.slide_out_bottom)
ft.addToBackStack(null)
ft.commit()
```

Fragment kommunikáció I.

- Egy Fragment-nek egységbezártnak kell lennie
  => közvetett kommunikáció
  - > Az Activity közvetít



Fragment kommunikáció II.

- Ha mégis szükség van a közvetlen Fragment-kommunikációra:
  - > Fragment.targetFragment propertyvel állítható/elérhető
- Az így létrejövő kapcsolat túléli a forgatásokat is

Fragment paraméterek

- Szükség lehet paraméter átadás
  - pl. Melyik cikk részleteit jelenítse meg a hírolvasó, stb..

- Első ötlet
  - ~~Mi inicializáljuk – Konstruktor paraméter~~
  - Nem csak mi inicializálhatjuk! – pl. Elforgatás
- Használjuk a Factory pattern-t
  - Fragment.arguments:Bundle property
  - Mentésre kerül a Fragment példánnyal

DialogFragment I.

- Egy Fragment dialógusként is megjelenhet
  - Dialógus egyedi layout-tal
  - Az AlertDialog.Builder továbbra is használható
- Így egy dialógus is ugyanolyan életciklussal rendelkezik, mint egy Fragment
- A FragmentDialog-ok is rákerülhetnek a BackStack-re

DialogFragment II.

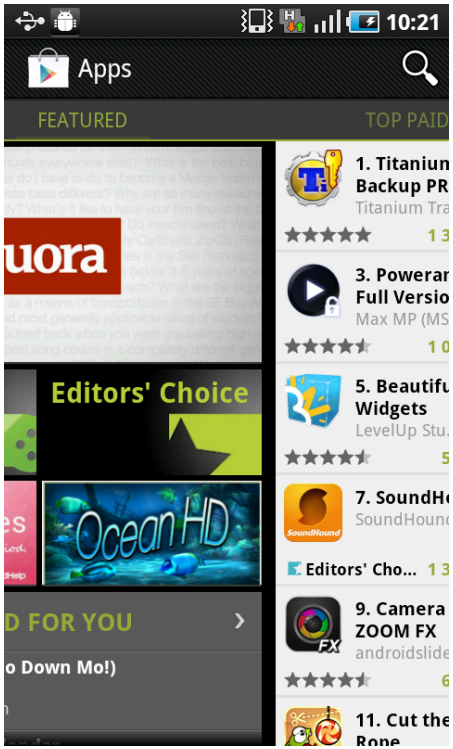
- .onCreateDialog()
  - Ez a metódus is visszatérhet a megjelenítendő Dialog-gal
- .onCreateView()
  - Ha nem használjuk az .onCreateDialog()-ot
  - Tetszőleges megjeleníthető tartalom
- Egy DialogFragment egyben Fragment is!
  - Ha kell, akár Activity-be ágyazottan is megjeleníthető

Fragment-ek fej nélkül

- Ha nem adjuk hozzá a Fragmentet egy ViewGroup-hoz -> a háttérben dolgozik
- .setOnRetainInstanceState(true)
  - Az .onRetainNonConfigurationInstance() helyett használható
  - Forgatásnál csak lecsatolja a megsemmisülő Activity-ről a Fragmentet, majd rácsatolja az újra
    - A memóriában marad, nincs szükség állapotmentésre

ViewPager

- ViewGroup, ahol az elemek közt swipe-al lehet mozogni
  - Pl.: Google Play



FragmentPagerAdapter I.

- Általában Fragment-eket adják a ViewPager oldalait
- Ekkor egy FragmentPagerAdapter példány szolgáltatja az egyes oldalakat
  - Hasonló elven működik, mint a BaseAdapter
  - Fragment getItem(int position):
    - Visszaadjuk a megfelelő Fragment példányt
  - int getCount():
    - Visszaadjuk, hogy összesen hány oldalunk van

FragmentPagerAdapter II.

- String getTitle(int position):
  - Visszaadjuk az adott oldal címét

- Demo!

PagerTitleStrip

- Widget, ami mutatja a ViewPager-ben szereplő oldalak címeit
  - A Support lib. tartalmazza
  - Nem interaktív

PagerTabStrip

- Widget, ami mutatja a ViewPager-ben szereplő oldalak címeit
  - A Support lib. tartalmazza
  - Interaktív!

ViewFlipper komponens

- View közti egyszerű váltás
- Több view található rajta, de mindig csak egyet jelenít meg
- XML-ben felsorolhatók a tartalmazott view-k
- Dinamikusan vezérelhető
- Fragmenteket is kezel
- Ha sok view-t teszünk rá, lassú lehet, mivel mindegyik komponenst előre előkészíti!

Mi nem igaz a Fragment-ekre?

A. Önálló életciklussal rendelkeznek.

B. Kötelező felhasználói felületet tartalmazniuk.

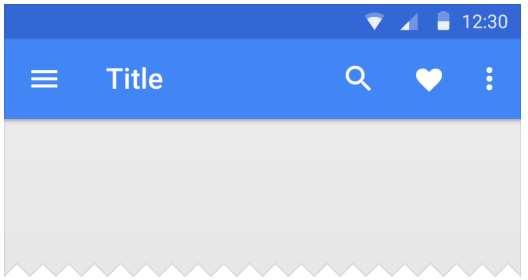
C. Dinamikusan és statikusan is csatolhatók.

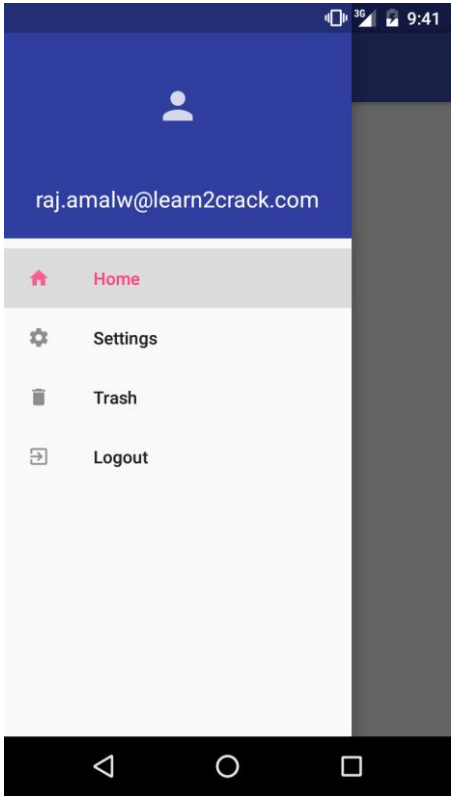
D. A tabletek felhasználói felületének kialakításakor különösen hasznosak.

Menü kezelés

Menü típusok

- „Régi” Menü
- „Elavult” ActionBar
- Toolbar
- Bottom Navigation View
- NavigationDrawer





Menü kezelés

- Forráskódból és XML erőforrásból is megadható
- Dinamikus menü elemek
  - Láthatóság állítás Java/Kotlin kódból
- Almenük támogatása

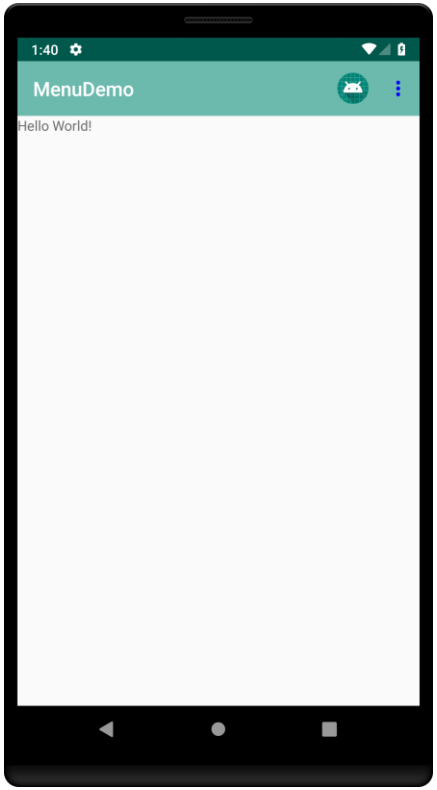
Menu erőforrás

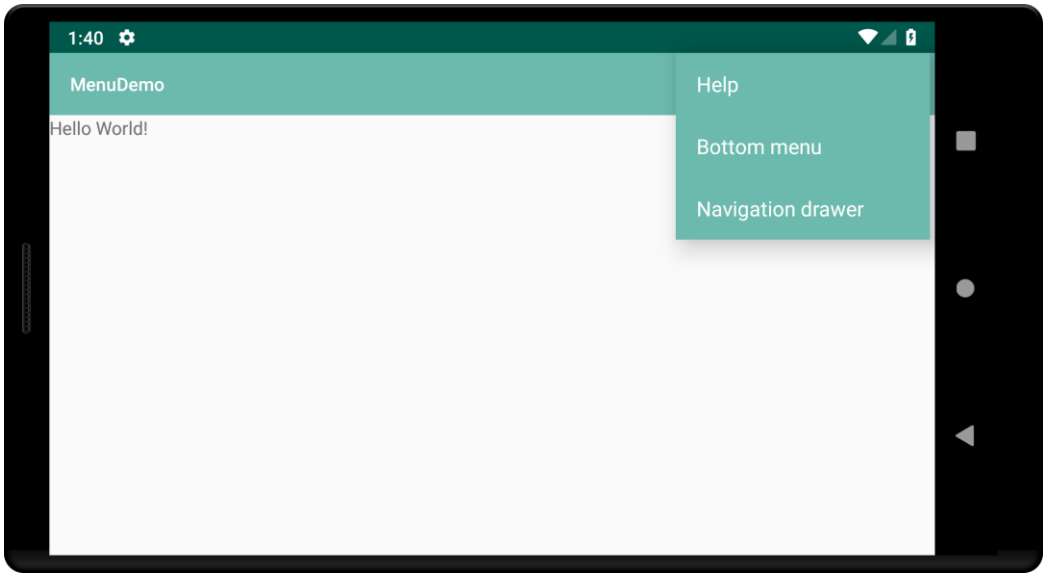
```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <item
        android:id="@+id/action_about"
        android:title="About"
        android:icon="@mipmap/ic_launcher"
        app:showAsAction="always|withText"/>

    <item
        android:id="@+id/action_help"
        android:title="Help" />

</menu>
```

Menü esemény kezelés

```
override fun onCreateOptionsMenu(menu: Menu): Boolean {
    menuInflater.inflate(R.menu.main_menu, menu)
    return super.onCreateOptionsMenu(menu)
}

override fun onOptionsItemSelected(item: MenuItem): Boolean {
    when (item.itemId) {
        R.id.action_about -> {
            Toast.makeText(this, "ABOUT", Toast.LENGTH_LONG).show()
        }
        R.id.action_help -> {
            Toast.makeText(this, "HELP", Toast.LENGTH_LONG).show()
        }
    }

    return true
}
```

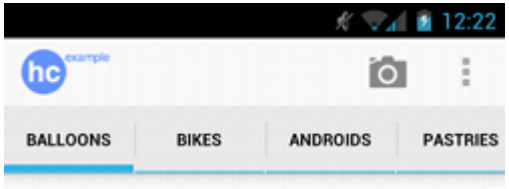
Menük és almenük





ActionBar és Menük

- Dedikált alkalmazás menü, logo és cím
- Tipikus felhasználás:
  - Menü
  - Branding (logo/background) és alkalmzás ikon
  - Konzisztens alkalmazás navigáció
  - Fő funkciók bemutatása



ActionBar specifikus XML menü paraméterek

```
<item android:id="@+id/action_time"
        android:title="@string/action_show_time"
        android:orderInCategory="5"
        android:icon="@drawable/clock_icon"
        android:showAsAction="always|withText" />
```

ActionBar -> Toolbar

- ActionBar helyett ToolBar
- Sokkal dinamikusabb viselkedés
- Menü erőforrások támogatása
- Custom elemek támogatása
- Manuális pozicionálás
- Toolbar tutorialok:
  - http://javatechig.com/android/android-lollipop-toolbar-example
  - http://www.101apps.co.za/index.php/articles/using-toolbars-in-your-apps.html

Toolbar használat

- ActionBar nélküli téma(styles.xml):
  - Theme.AppCompat.NoActionBar
- Layout erőforrás:

```
<android.support.v7.widget.Toolbar
        android:id="@+id/toolbar"
        android:minHeight="?attr/actionBarSize"
        android:background="#2196F3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
</android.support.v7.widget.Toolbar>
```

- Activity onCreate(...):

```
@Override
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Set a toolbar to replace the action bar.
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
}
```

Toolbar testreszabás

- Styles.xml:

```
<style name="ToolBarStyle" parent="Theme.AppCompat">
    <item name="android:textColorPrimary">#ff0000</item>
    <item name="android:textColorSecondary">#0000ff</item>
    <item name="actionMenuTextColor">@android:color/white</item>
</style>
```

- Toolbar layout

```
<android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#21f348"
    app:theme="@style/ToolBarStyle"
    android:minHeight="?attr/actionBarSize"/>
```

- Layout app namespace:

```
xmlns:app="http://schemas.android.com/apk/res-auto"
```

Fragment és BottomNavigation

Kérdések