8. INTENT FILTERS A BRODACAST RECEIVER

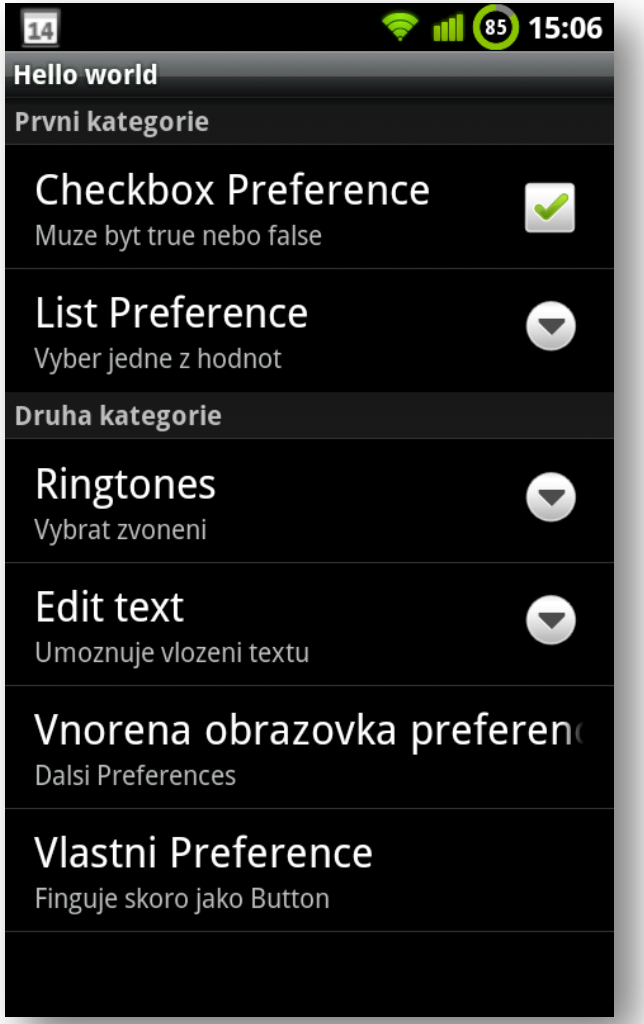
BI-AND

8. Přednáška

- PreferenceActivity
- Intenty (pokračování)
- Intent-Filter
- Broadcast Receiver

PreferenceActivity

- Framework umožňující definovat obrazovku s nastavením aplikace pomocí XML uložené v `res/xml/`
- Propojení XML komponent přímo na položky v SharedPreferences
- Konzistentní vzhled napříč aplikacemi
- Obtížněji se styluje



PreferenceActivity

- Nastavení je automaticky uloženo do defaultních `SharedPreferences` společných pro všechny komponenty aplikace

- Komponenta se skládá ze 2 částí:
  - XML layoutu
  - Třída dědící od PreferenceActivity (musí být definována v AndroidManifestu jako každá jiná activita)

PreferenceActivity – XML

- Každý preference layout je definovaný jako hierarchie začínající elementem `PreferenceScreen`

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <PreferenceScreen
3.       xmlns:android="http://schemas.android.com/apk/res/android">
4.       ...
5. </PreferenceScreen>
```

- Je možné do sebe vnořit více obrazovek `PreferenceScreen`, každá pak bude reprezentována jako položka, po jejímž výběru se zobrazí nová obrazovka s nastavením

```
1. ...
2. <PreferenceScreen
3.       android:key="SecondPrefScreen"
4.       android:title="Vnorena obrazovka preferences"
5.       android:summary="Dalsi Preferences">
6.       ...
7. </PreferenceScreen>
8. ...
```

PreferenceActivity – XML elementy

- V `PreferenceScreen` je dále možné přidávat šest různých elementů:

- `CheckBoxPreference` – jednoduchý Checkbox, který vrací true/false

- `ListPreference` – po výběru zobrazí RadioGroup, kde uživatel může zvolit jednu položku
  - V *android:entries* se definují položky z `res/values/arrays`, z kterých uživatel vybírá
  - V *android:entryValues* se k nim pak definují hodnoty, které se uloží; opět z `res/values/arrays`

PreferenceActivity – XML elementy

- `EditTextPreference` – po výběru zobrazí EditText
  - Vrací String
- `RingtonePreference` – po výběru zobrazí radioGroup s dostupnými vyzváněcími tóny
  - Vrací String s URI zvoleného tónu
- `Preference` – obecná položka, chová se v podstatě jako Button
- `PreferenceCategory` – kategorie pomocí které od sebe můžeme položky oddělit

PreferenceActivity – XML atributy

- U každé Preference můžeme definovat tyto základní parametry
  - `android:title` – titulek položky (zobrazuje se v 1. řádku)
  - `android:summary` – podrobnější popisek položky (2. řádek)
  - `android:defaultValue` – defaultní hodnota (použije se, pokud zatím nebyla žádná hodnota uložena)
  - `android:key` – klíč, pod kterém se uloží hodnota do `SharedPreferences`

PreferenceActivity – XML atributy

- Další možné parametry u každé položky
  - `android:persistent` – určuje, jestli se tato položka bude ukládat do SharedPreferences (false = nebude)
  - `android:enabled` – určuje, jestli je tato položka aktivní
  - `android:dependency` – pokud se vyplní určená položka nebo se změní její stav (enabled/disabled), daná preference půjde nebo nepůjde použít

PreferenceActivity - XML

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <PreferenceScreen
3.     xmlns:android="http://schemas.android.com/apk/res/android">
4.     <PreferenceCategory android:title="Prvni kategorie">
5.         <CheckBoxPreference
6.             android:title="title"
7.             android:defaultValue="false"
8.             android:summary="summary"
9.             android:key="checkboxPref" />
10.        <ListPreference
11.            android:title="List Preference"
12.            android:summary="Vyber jedne z hodnot"
13.            android:key="listPref"
14.            android:defaultValue="2"
15.            android:entries="@array/listArray"
16.            android:entryValues="@array/listValues"
17.            android:dependency="checkboxPref" />
18.    </PreferenceCategory>
19.    <PreferenceCategory android:title="Druha kategorie">
20.        <RingtonePreference
21.            android:name="Ringtone Preference"
22.            android:summary="Vybrat zvoneni"
23.            android:title="Ringtones"
24.            android:key="ringtonePref" />
25.    </PreferenceCategory>
26. </PreferenceScreen>
```

PreferenceActivity - Java

- Ve třídě dědící od `PreferenceActivity` přiřadíme námi vytvořené XML zavoláním `addPreferencesFromResource(id)`

- Konkrétní Preference získáme pomocí `findPreference(key);`

```
1.  public class Settings extends PreferenceActivity {
2.     @Override
3.     protected void onCreate(Bundle savedInstanceState) {
4.           super.onCreate(savedInstanceState);
5.           addPreferencesFromResource(R.xml.preferences);
6.
7.           ListPreference listPreference = (ListPreference) findPreference("listPref");
8.           //Nastavime hodnoty u listPreference
9.           listPreference.setEntries(new String[] { "polozka1", "polozka2" });
10.          listPreference.setEntryValues(new String[] { "1", "2" });
11.    }
12. }
```

PreferenceActivity - Java

- PreferenceActivity je provázána s default SharedPreferences

- Nastavení vlastního PrefFile jako default SharedPreferences

```
1. PreferenceManager prefMgr = getPreferenceManager();
2. prefMgr.setSharedPreferencesName("my_preferences");
3. prefMgr.setSharedPreferencesMode(MODE_WORLD_READABLE);
4.
5. addPreferencesFromResource(R.xml.prefs);
```

PreferenceActivity - Java

- Pokud potřebujeme sledovat změny v nastavení, implementujeme OnSharedPreferenceChangeListener

```
1.  public class Settings extends PreferenceActivity
2.  implements OnSharedPreferenceChangeListener {
3.      @Override
4.      protected void onCreate(Bundle savedInstanceState) {
5.          super.onCreate(savedInstanceState);
6.          addPreferencesFromResource(R.xml.preferences);
7.
8.          // ziskame defaultni SharedPreferences, do kterych zapisuje PreferenceActivity
9.          SharedPreferences sp = PreferenceManager.getDefaultSharedPreferences(this);
10.         // zaregistrujeme OnSharedPreferenceChangeListener
11.         sp.registerOnSharedPreferenceChangeListener(this);
12.
13.     }
14.
15.     // Zavola se pri jakekoli zmene danych SharedPreferences
16.     @Override
17.     public void onSharedPreferenceChanged(SharedPreferences sp, String key) {
18.         // Provedeni potrebnych ukonu, pokud uzivatel zmeni nejake nastaveni
19.     }
20.}
```

Intent

- Zpráva s informacemi pro příjemce
  - Jakým způsobem se má naložit s daty ve zprávě
- Zpráva s informacemi pro systém
  - Kategorie komponenty a instrukce, jakým způsobem má být spuštěna
- Pro Intent lze nastavit šest hlavních parametrů:
  - Component Name
  - Action
  - Data
  - Category
  - Extras
  - Flags

Intent Filters

- Určují, pro jaký obsah se může daná komponenta použít
- V úvahu se berou tyto tři parametry Intentu:
  - Action
  - Data (URI i MIME type)
  - Category
- Definují se v AndroidManifest.xml v tagu dané komponenty

Každá komponenta může mít definováno více Intent Filters

```
1. <activity
2.    android:name=".ShareActivity"
3.    android:label="MyShare Activity">
4.    <intent-filter>
5.          <action
6.                android:name="android.intent.action.SEND" />
7.          <category
8.                android:name="android.intent.category.DEFAULT" />
9.          <data
10.               android:mimeType="text/plain" />
11.   </intent-filter>
12.</activity>
```

Intent – Component Name

- Název komponenty, která se má postarat o přijetý Intent
- Skládá se z
  - Celého názvu cílové komponenty – např. cz.cvut.fit.ContactActivity
  - Názvu package – např. cz.cvut.fit
- Název cílové komponenty a package nemusí být vždy nutně stejný
- Určení pomocí metod `setComponent()`, `setClass()` nebo `setClassName()`

Intent - Action

- Název akce, jaká má být provedena
- Podobné jako u protokolu HTTP a operací GET, POST, PUT, DELETE atd.
- V Android API je množství akcí daleko větší
- Je možné rovněž vytvářet vlastní akce

Intent Filters – Action

- Definuje akci, pro kterou se daná komponenta má spustit
- Při definování vlastní akce se doporučuje zahrnout celé jméno balíčku např. „`cz.cvut.example.SHOW_COLOR`“

Intent filtr v AndroidManifest.xml

```
1. <intent-filter>
2.         <action
3.              android:name="android.intent.action.SEND" />
4.         ...
5. </intent-filter>
```

Nastavení akce

```
1. intent.setAction(Intent.ACTION_SEND);
2. //nebo "android.intent.action.SEND"
```

Intent – nejpoužívanější akce

- `Intent.ACTION_CALL`
  - Zahájí telefonní hovor
  - Vyžaduje oprávnění `android.permission.CALL_PHONE`
- `Intent.ACTION_DIAL`
  - Otevře telefonní aplikaci s předvyplněným číslem
- `Intent.ACTION_EDIT`
  - Slouží k editaci dat
- `Intent.ACTION_MAIN`
  - Spustí hlavní activitu aplikace
- `Intent.ACTION_VIEW`
  - Zobrazí data uživateli (zobrazení obrázku, přehrání hudby…)
- `Intent.ACTION_SEND`
  - Odešle data
- `Intent.ACTION_DELETE`
  - Smaže daná data

Intent – Data

- Definuje, pro jaký typ dat se má komponenta použít
- Tag `<data>` může specifikovat URI a/nebo MIME typ
  - URI je rozděleno do 4 částí, z nichž každá má svůj atribut v tagu <data> - `scheme://host:port/path`

Intent filtr v AndroidManifest.xml

```
1. <intent-filter>
2.     ...
3.     <data
4.         android:mimeType="text/plain"
5.         android:scheme="file" />
6.     ...
7. </intent-filter>
```

Nastavení typu a URI

```
1. intent.setType("text/plain");

2. intent.setData(Uri.parse("file:///sdcard/folder/file.txt"));
```

Intent – Category

- (Dodatečná) informace o tom, jaká komponenta má zpracovat Intent

- Nejpoužívanější kategorie:

- `Intent.`*`CATEGORY_PREFERENCE`*
  - Cílová activita je panel s nastavením
- `Intent.`*`CATEGORY_DEFAULT`*
  - Přidávána defaultně při `startActivity()`
  - Cílová komponenta není určena

Intent – nejpoužívanější kategorie

- `Intent.`*`CATEGORY_BROWSABLE`*
  - Cílová activita je uplatněná jako prohlížeč dat předaných odkazem
- `Intent.`*`CATEGORY_HOME`*
  - Activita zobrazuje domovskou obrazovku (náhrada defaultního launcheru)
- `Intent.`*`CATEGORY_LAUNCHER`*
  - Počáteční activita, která se spustí po vybrání z launcheru

Intent Filtres – Category

- Intent projde intent filtrem, pokud filtr obsahuje všechny v intentu nastavené kategorie
- Všechny intenty předané do startActivity() obsahují minimálně tuto kategorii: `android.intent.category.DEFAULT`

Intent filtr v *AndroidManifest.xml*

Výjimka pro Explicit Intents a MAIN Action/Launcher Category

```
1. <intent-filter>
2.     ...
3.     <category android:name="android.intent.category.DEFAULT" />
4.     <category android:name="android.intent.category.BROWSABLE" />
5.     ...
6. </intent-filter>
```

Přidání kategorie

```
1. intent.addCategory(Intent.CATEGORY_BROWSABLE);
2. intent.addCategory(Intent.CATEGORY_PREFERENCE);
```

Intent – Extras a Flags

- Extras
  - Předání dalších informací pomocí klíč/hodnota
  - Přenáší se zde další potřebné údaje v rámci systémových Intent
  - Viz. 7. přednáška

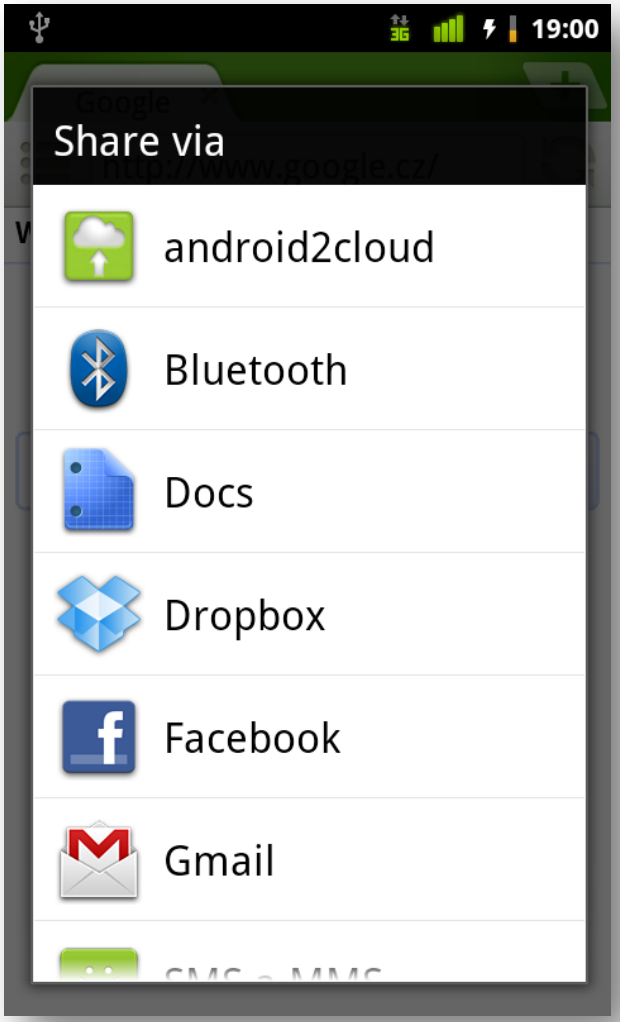
- Flags
  - Určuje, jakým způsobem se má spustit daná komponenta
    - Řazení na zásobník, animace, launchmode
  - Viz. 3. přednáška

Intent - Explicitní

- Spouštějí přesně definovanou třídu (resp. komponentu)
- Komponenta může být:
  - Activity
  - Service
  - Broadcast Receiver
- Většinou nejsou používány vývojáři jiných aplikací
  - Neznají přesné jméno komponenty
  - Název komponenty se může změnit

Intent - Implicitní

- Nemají přímo definovaný cíl
- Pokud existuje více potenciálních Activit pro daný Intent, může si uživatel zvolit, jakou chce spustit
- Jednoduché rozšíření funkčnosti programu
- Většinou se používají ke spouštění komponent jiných aplikací

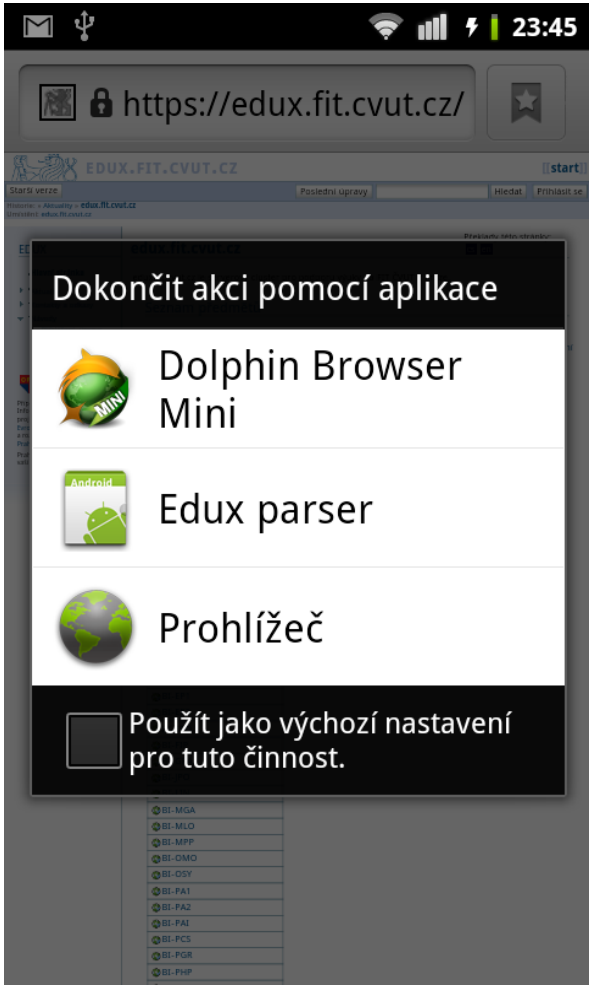


Intent - Implicitní

- Název komponenty zůstává nevyplněn
- Systém porovná obsah Intentu s Intent Filtres komponent a vybere ty, které mohou daný Intent přijmout
- Lze dopředu zjistit jaké komponenty mohou Intent přijmout pomocí
  - PackageManager – metody `queryIntentActivities()`, `queryIntentServices()`, `queryIntentReceivers()`

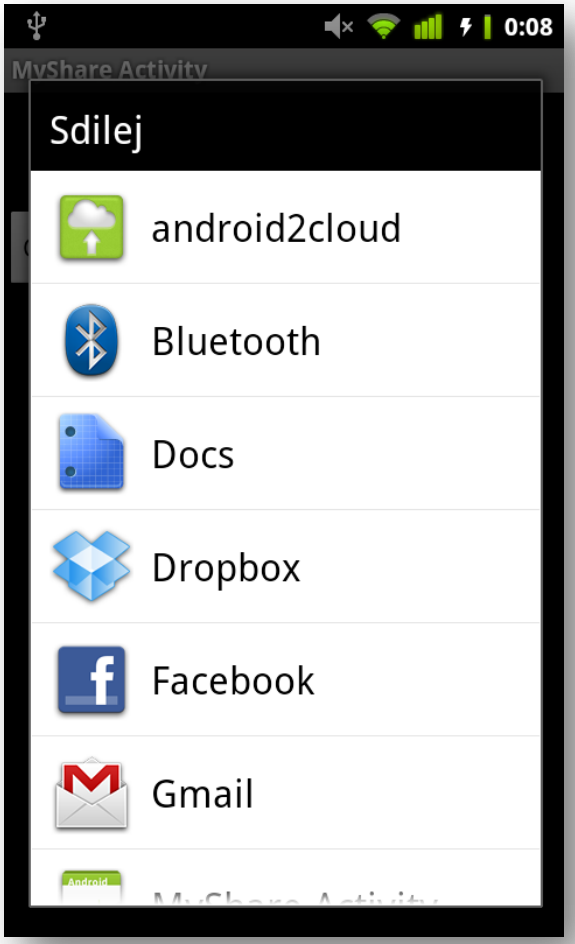
Intent Filter – edux filtr

```
<activity
     android:name=".EduxActivity"
     android:label="Edux">
     <intent-filter
          android:label="Edux parser">
          <action
               android:name=
               "android.intent.action.VIEW" />
          <category
               android:name=
              "android.intent.category.DEFAULT"/>
          <category
               android:name=
              "android.intent.category.BROWSABLE"/>
          <data
               android:scheme="https"
               android:host="edux.fit.cvut.cz" />
     </intent-filter>
</activity>
```



Intent – výběr activity

- Dialog výběru activit umožňuje standardně nastavit 1 activitu jako defaultní
- Pro vynucené zobrazení dialogu použijeme
  `Intent.createChooser(Intent,String);`
- Vhodné při sdílení




```
1. Intent intent=new Intent(Intent.ACTION_SEND);
2. intent.setType("text/plain");
3. intent.putExtra(Intent.EXTRA_TEXT, "text, ktery chceme sdilet");
4. startActivityForResult(Intent.createChooser(intent, "Sdilej"), 0);
```

Další informace o Intents

- Registr Intents a přidružených aplikací
  http://www.openintents.org

- Další pravidla pro zachytávání Intents
  http://developer.android.com/guide/topics/intents/intents-filters.html#ifs

Broadcast

- Doposud jsme Intenty používali ke spouštění nové Activity, Service apod.
- Intenty však mohou sloužit také jako posílání anonymních zpráv mezi komponentami
  - Většinou reakce na určité změny v systému
  - Naprosto oddělený mechanismus od toho používaného např. při `startActivity()`
- Po odeslání broadcastu se pošle daný Intent všem BroadcastReceiverům zaregistrovaným na danou akci v Intentu

Broadcast Receiver

- Zachytávání Broadcastu při komunikaci jednotlivých komponent v systému
- Je potřeba implementovat pouze metodu `onReceive()`
- Životní cyklus vázaný na danou komponentu. Jinak skončí po přijetí zprávy (dokončení `onReceive()`)
- Není vázaný na UI – nemůže vytvořit dialog, nastartovat background thread apod.

Broadcast Receiver - zaregistrování

- Staticky v AndroidManifest.xml

```
1. <receiver
2.       android:name=".MyReceiver">
3.       <intent-filter>
4.               <action
5.                       android:name="cz.cvut.example.TEST_BROADCAST" />
6.       </intent-filter>
7. </receiver>
```

- Nebo dynamicky v kódu (např. pokud je potřeba pouze když je activita viditelná)

```
1. IntentFilter filter = new IntentFilter(TEST_BROADCAST);
2. BroadcastReceiver myReceiver = new TestReceiver();
3. registerReceiver(myReceiver, filter);
4. ...
5. unregisterReceiver(myReceiver);
```

Provést např. v onResume()

Provést např. v onPause()

Broadcast Receiver

- Startují se automaticky obdržením broadcastu
- `onReceive` musí skončit do deseti sekund
- Pro delší operace je zde vhodné spustit službu
- BroadcastReceiver nedědí z Contextu, ten je předán metodě onReceive

```
1.  public class MyReceiver extends BroadcastReceiver {
2.
3.      @Override
4.      public void onReceive(Context context, Intent intent) {
5.              int id = intent.getIntExtra("id", 0);
6.              ...
7.      }
8.  }
```

Broadcast – odeslání

Sticky Intents

- Broadcast můžeme odeslat dvěma způsoby
- Normal Broadcast (Non-Ordered)
  - Pošle se pomocí `Context.sendBroadcast`
  - Je kompletně asynchronní – všechny BroadcastReceivery obdrží tuto zprávu v nedefinovaném pořadí
  - Většinou jsou zpracovány postupně, kvůli zabránění přílišnému vytížení systému
  - Není možného ho zrušit

Broadcast – odeslání

- Ordered Broadcast
  - Posílá se pomocí `Context.sendOrderedBroadcast`
  - V jeden okamžik zpracovává zprávu maximálně jeden Broadcast Receiver
  - Pořadí se určí atributem `android:priority` v Intent filteru daného receiveru
  - Po zpracování zprávy může každý Brodcast Receiver určit, zda-li
    - Předá výsledek dalšímu Broadcast Receiveru v pořadí
    - Kompletně zruší daný broadcast a dál už se předávat nebude

Broadcast – odeslání

```
1.  ...
2.  public static final String TEST_BROADCAST=
3.      "cz.cvut.example.TEST_BROADCAST";
4.  ...
5.
6.  Intent i = new Intent(TEST_BROADCAST);
7.  i.putExtra("id", 3);
8.  sendBroadcast(i);
9.  ...
```

- Obvykle se používá jméno balíčku, aby se zajistila jedinečnost
- Tímto řetězcem se musí zaregistrovat všechny BroadcastReceivery, které mají reagovat na tuto zprávu

Broadcast – nejpoužívanější

- `Intent.`*`ACTION_BOOT_COMPLETED`*
  - Odešle se po nastartování systému
  - Vyžaduje oprávnění RECEIVE_BOOT_COMPLETED
- `Intent.`*`ACTION_MEDIA_BUTTON`*
  - Odešle se po stisknutí tlačítka pro ovládání přehrávače (např. na sluchátkách)
  - V Intentu je v extra poli EXTRA_KEY_EVENT informace o daném tlačítku
- `Intent.`*`ACTION_CAMERA_BUTTON`*
  - Odešle se po stisknutí tlačítka fotoaparátu

Broadcast – nejpoužívanější

- `Intent.`*`ACTION_MEDIA_EJECT`*
  - Pokud se uživatel rozhodne vyjmout externí úložiště (média)
  - Po obdržení této zprávy by aplikace měly ukončit IO z/do externího úložiště
- `Intent.`*`ACTION_BATTERY_LOW`*
  - Odešle se spolu s hláškou o nízkém stavu baterky
- `Intent.`*`ACTION_SCREEN_ON`* a *`ACTION_SCREEN_OFF`*
  - Odešle se při zapnutí a vypnutí obrazovky
- `Intent.`*`ACTION_MEDIA_MOUNTED`* a *`ACTION_MEDIA_UNMOUNTED`*
  - Odešle se po vložení/vyjmutí externího úložiště (média)

Broadcast - oprávnění

- Pro vyžádání oprávnění při odesílání broadcastu předáme nenullový parametr jedné z metod
  - `sendBroadcast(intent, receiverPermission);`
  - `sendOrderedBroadcast(intent, receiverPermission);`

- Daný broadcast přijmou pouze receivery, které mají toto oprávnění definované v *AndroidManifest.xml* pomocí `<uses-permission>`

Broadcast - oprávnění

- Pro vyžádání oprávnění při přijímání definujeme toto oprávnění při zaregistrování Broadcast Receiveru

- Dynamicky – `registerReceiver(receiver, filter, broadcastPermission, scheduler);`
- Staticky

Scheduler – vlákno, které má Intent přijmout

```
1. <receiver
2.       android:name=".MyReceiver"
3.       android:permission="android.permission.INTERNET">
4.       ...
5. </receiver>
```

- Broadcast Receiver přijme daný broadcast pouze z aplikací, které mají toto oprávnění definované v AndroidManifest.xml pomocí `<uses-permission>`

Další zdroje

- http://www.openintents.org
- http://developer.android.com/guide/topics/intents/intents-filters.html#ifs
- http://android-developers.blogspot.com/2012/02/share-with-intents.html