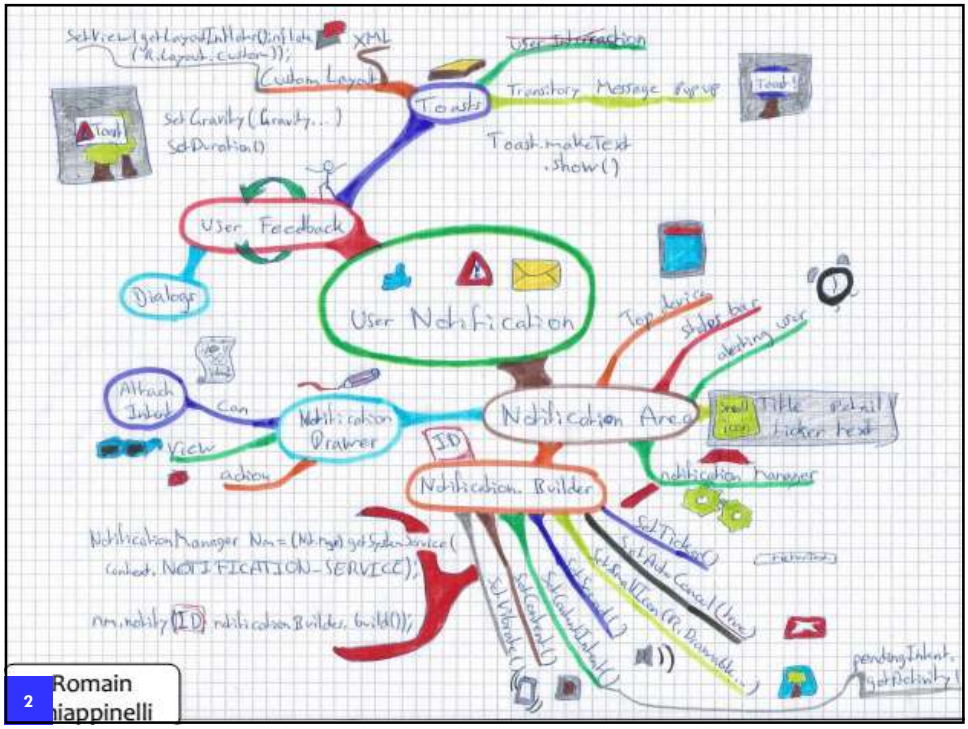
CHAPITRE 7
LES NOTIFICATIONS ET LES BROADCASTRECEIVERS

• Mr. MEGHAZI

2019-2020

Cours pour les Master II - GL



Contenu

- Les Notifications
  - Toast 
  - Les notifications de la zone de Notifications 
- Les broadcastReceivers 
  - Les broadcasts
  - Traitement des broadcasts

3

La notification de l'utilisateur (1)

- Ce sont des messages fournies à l'utilisateur en dehors de la zone normale de l'UI
  - **Exemple** : MMS, SMS, …

4

La notification de l'utilisateur (2)

Ceci inclut les messages qui ont pour objectifs:

- **Donner un FeedBack à l'user**
  - **Toasts**
  - **Dialogs** (Boites de dialogue)
- **Notification d'évènements**
  - **Les notifications de la zone de notification**
    - Evènements imprévisibles
    - Sans faire une interruption

5

Toast

- Ce sont des messages transitoire qui apparaissent (surgissent) sur la fenêtre courante.
  - **Exemple**: informer l'user qu'une opération vient de se terminer avec succès
- Apparaitre et disparaitre **automatiquement** et progressivement (le **Fade-In** et le **Fade-Out**) de la vue.
- **Pas d'interaction** ou réponse de l'user.

6

Toast – Création d'une notification

- **Instancier** un objet « Toast » en faisant l'appel de la méthode:

  Toast.makeText(context, text, duration)

- **Afficher** le « Toast » en faisant l'appel de:

  Toast.show()

7

Un simple « Toast »

8

Toast avec une vue personnalisée

- On peut créer des « Layouts » personnalisés en XML
- Les attacher en utilisant `Toast.setView()`

9

Un « Toast » personnalisé

10

Notifications de la zone notification

- **Android** fournit une **zone de notification** afin d'avertir les utilisateurs des évènements
- Il fournit aussi un « **Drawer** » (Tiroir) que l'user peut le tirer vers le bas pour voir plus d'information détaillées sur les notifications.

11

La zone de notification

12

Architecture d'une notification

- **Notification**
  - Titre, petite Icône, détail
- **Zone de notification**
  - Ticker Text, petite Icône
- **Notification Drawer** (Tiroir)
  - **Vue**
  - **Action**

13

Notifications Manager

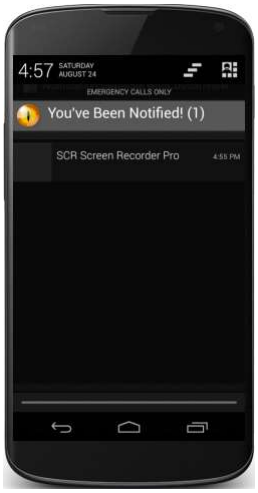
- Un service « Système » qui gère les notifications
- Envoi et annule les notifications

14

Exemple (1)

15

Exemple (2)



16

A suivre …

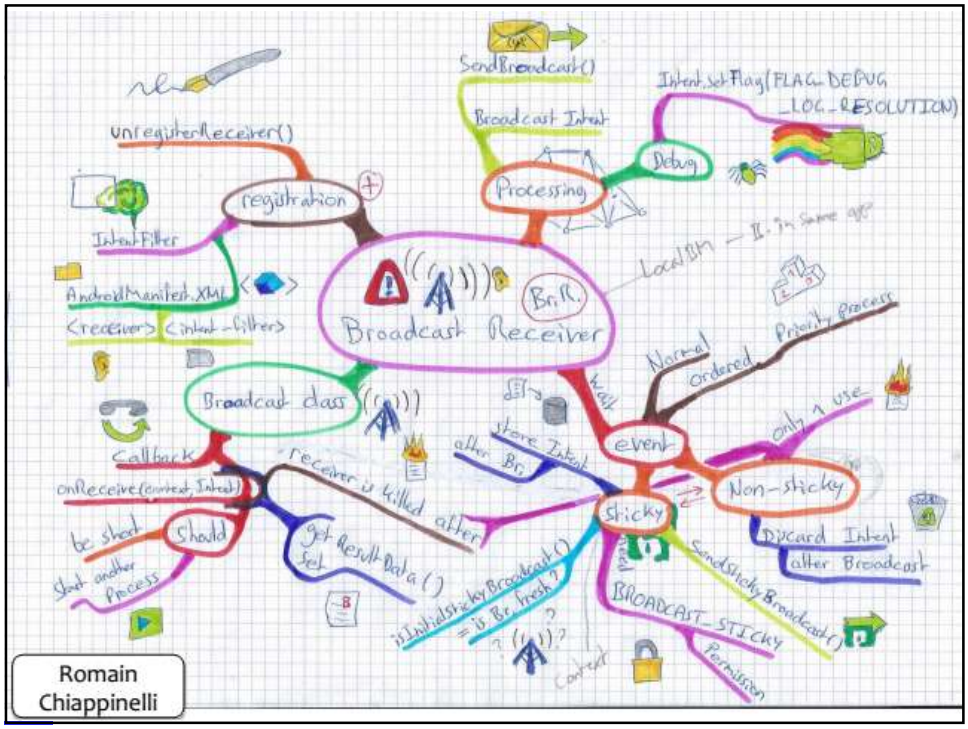
Les BroadCastReceivers

17

Contenu

- La Classe Broadcast
- L'enregistrement d'un récepteur
- Les broadCasts
- Traitement des broadCasts

18



BroadCastReceiver (1)

- Classe de base pour les composants qui reçoivent et réagissent aux évènements.
- Les « BroadCast Receivers » s'enregistrent pour recevoir les évènements dans lesquels ils sont intéressés
  - **Exemple**: MMS

20

BroadCastReceiver (2)

- Quand des évènements se produisent, ils sont représentés comme des « Intents ».
- Ces « Intents » sont ensuite diffusés au système.

21

BroadCastReceiver (3)

- Android route les « Intents » vers les « broadCastReceivers » qui se sont enregistrés pour les recevoir.
- Les « broadCastReceivers » reçoivent l' « Intent » par l'intermédiaire d'un appel à onReceive()

22

Cas d'utilisation typique

1. Enregistrement des « **BroadCastReceivers** »
2. Diffusion d'un « **Intent** »
3. **Android** délivre l' « **Intent** » vers les récipients enregistrés en invoquant leurs méthode onReceive()
4. Prise en charge de l'évènement dans onReceive()

23

Enregistrement pour un « Intent »

- Les « BroadCastReceivers » peuvent s'enregistrer en utilisant deux manières:
  - **Statiquement**, dans « AndroidManifest.XML »
  - **Dynamiquement**, en invoquant la méthode registerReceiver()

24

Enregistrement Statique (1)

- Mettre les tags <receiver> et <intent-filter> dans « AndroidManifest.xml »
  - **C.à.d:** quand un « Intent » qui correspond à cet « Intent-filter » est diffusé, ce « BroadCastReceiver » veut le savoir.

25

Format du < Receiver >

```
<receiver
        android:enabled=["true" | "false"]
        android:exported=["true" | "false"]
        android:icon="drawable resource"
        android:label="string resource"
        android:name="string"
        android:permission="string"
        android:process="string" >
    . . .
</receiver>
```

26

Intent Filter

- Spécifier le tag <intent-filter> à l'interieur de <receiver>

27

Enregistrement Statique (2)

- Les « Receivers » sont enregistrés avec le système quand il Boot ou quand un package d'une application est ajouté à l'éxecution.

28

Exemple (1)

29

Exemple (2)

BcastRecSinBcastStatReg

```
package course.examples.BroadcastReceiver.singleBroadcastStaticRegistration;

import android.app.Activity;

public class SimpleBroadcast extends Activity {

    private static final String CUSTOM_INTENT = "course.examples.BroadcastReceiver.show_toast";

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

        Button button = (Button) findViewById(R.id.button);
        button.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {

                sendBroadcast(new Intent(CUSTOM_INTENT),
                        android.Manifest.permission.VIBRATE);
            }
        });
    }
}
```

30

Exemple (3)

BcastRecSinBcastStatReg

```
package course.examples.BroadcastReceiver.singleBroadcastStaticRegistration;

import android.content.BroadcastReceiver;

public class Receiver extends BroadcastReceiver {

    private final String TAG = "Receiver";

    @Override
    public void onReceive(Context context, Intent intent) {

        Log.i(TAG, "INTENT RECEIVED");

        Vibrator v = (Vibrator) context
                .getSystemService(Context.VIBRATOR_SERVICE);
        v.vibrate(500);

        Toast.makeText(context, "INTENT RECEIVED by Receiver", Toast.LENGTH_LONG).show();

    }
}
```

31

Enregistrement Dynamique (1)

- Créer un filtre d'Intent.
- Créer un BroadCastReceiver
- Enregistrer le **BroadCastReceiver** en utilisant registerReceiver()
  - **LocalBroadcastManager**
  - **Context**
- Invoquer unRegisterReceiver() pour désenregistrer le **BroadCastReceiver**

32

Exemple (1)

BcastRecSinBcastDynReg

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    mBroadcastMgr = LocalBroadcastManager
            .getInstance(getApplicationContext());
    mBroadcastMgr.registerReceiver(receiver, intentFilter);

    setContentView(R.layout.main);

    Button button = (Button) findViewById(R.id.button);
    button.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
            mBroadcastMgr.sendBroadcast(new Intent(CUSTOM_INTENT));
        }
    });
}
```

33

Diffusion d'évènements (1)

- Plusieurs manières de diffusion sont prises en charge
- Normal vs Ordered
  - **Normal**: l'ordre de traitement est indéfini
  - **Ordered**: traitement séquentiel suivant un ordre de priorité

34

Diffusion d'évènements (2)

- Sticky vs. Non-Sticky
  - **Sticky** : maintient l'Intent après sa diffusion initiale
  - **Non-Sticky**: se débarrasse de l'Intent après sa diffusion initiale.
- Avec ou sans des permissions sur le « Receiver »

35

Quelques astuces de débogage

LOG EXTRA INTENT RESOLUTION INFORMATION

```
Intent.setFlag(FLAG_DEBUG_LOG_RESOLUTION)
```

LIST REGISTERED BROADCASTRECEIVERS

DYNAMICALLY REGISTERED

```
% adb shell dumpsys  activity b
```

STATICALLY REGISTERED

```
% adb shell dumpsys  package
```

36