BroadCast Receiver sous Android

Tutos-android

Benbourahla

par Feanorin

Date de publication : 12 mai 2011

Dernière mise à jour :

Ce tutoriel va aborder les **broadCast Receiver** sous Android. Nous allons commencer par expliquer le principe des **broadcast receiver**, puis nous allons faire un petit exemple d'un broadcast receiver qui reçoit les messages du téléphone.

I - Qu'est ce qu'un BroadCast Receiver ?

Pour pouvoir recevoir des **intents**, Android vous permet de créer une classe qui implémente **BroadcastReceiver**. Ces objets sont conçus pour recevoir des intents (intentions) et appliquer des comportements spécifiques à votre code.
L'interface **BroadcastReceiver** ne possède qu'une seule méthode **onReceive()** que votre classe devra implémenter.
Un **BroadcastReceiver** ne vit que le temps de traiter votre **onReceive()**. L'instance peut être supprimée par le Garbage Collector.
Les receivers sont limités : ils ne peuvent pas ouvrir de boîte de dialogue par exemple. Le système Android envoie l'intention à tous les **BroadCast Receiver** abonnés par ordre de priorité (priorité de votre BroadCast dans le fichier **AndroidManifest.xml**).
Si un **BroadCast** souhaite interrompre la réception du **Broadcast** à ceux d'un niveau inférieure de priorité, il faut utiliser la méthode **abortBroadcast()**.

II - Mon premier BroadCast Receiver

On va créer un nouveau projet qu'on nommera "**MyBroadCastReceiver**", avec les informations suivantes :

- **Version du SDK** : 2.1 ;
- **Application Name** : My First Broadcast ;
- **Package Name** : com.tuto.android ;
- **Min SDK version** : 7 ;
- décocher la cache : **Create Activity**.

Nous allons créer une classe "**SMSReceiver**", qui hérite de "**BroadcastReceiver**". Il vous demandera de rajouter les méthodes à implémenter (**OnReceive** dans notre cas).
Votre classe doit ressembler à ça :

```
package com.tuto.android;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;

public class SMSReceiver extends BroadcastReceiver
{

 @Override
 public void onReceive(Context arg0, Intent arg1)
 {
 // TODO Auto-generated method stub

 }

}
```

Donc pour recevoir les **messages** vous devez définir la permission ainsi que le Broadcast Receiver dans votre **AndroidManifest.xml**.

- **Lecture et réception des SMS**

```
<uses-permission android:name="android.permission.RECEIVE_SMS" />
<uses-permission android:name="android.permission.READ_SMS" />
```

- **Déclaration du Broadcast Receiver**

```
<receiver class="com.tuto.android.SMSReceiver"
    android:name="com.tuto.android.SMSReceiver">
    <intent-filter android:priority="100">
        <action android:name="android.provider.Telephony.SMS_RECEIVED" />
    </intent-filter>
</receiver>
```

Donc on déclare notre receiver, on peut lui définir une priorité dans la réception d'évènement(s) (100 dans notre exemple) et dans la partie intent filtres, on définit que notre receiver est appelé quand le téléphone reçoit un SMS.
Voilà ce que devrait donner votre **AndroidManifest.xml**.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.tuto.android" android:versionCode="1" android:versionName="1.0">
    <uses-sdk android:minSdkVersion="7" />

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <receiver class="com.tuto.android.SMSReceiver"
            android:name="com.tuto.android.SMSReceiver">
            <intent-filter android:priority="100">
                <action android:name="android.provider.Telephony.SMS_RECEIVED" />
            </intent-filter>
        </receiver>
    </application>

    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    <uses-permission android:name="android.permission.READ_SMS" />
</manifest>
```

Revenons à notre classe, nous allons gérer la réception de notre évènement, donc dans la méthode **OnReceive**.

```
private final String                ACTION_RECEIVE_SMS = "android.provider.Telephony.SMS_RECEIVED";

 @Override
 public void onReceive(Context context, Intent intent)
 {
  if (intent.getAction().equals(ACTION_RECEIVE_SMS))
  {
 }
}
```

On déclare une chaîne de caractères qui représente l'action correspondant à l'évènement que l'on va recevoir et on teste si l'évènement reçu correspond bien à celui-là.
Maintenant mettons en place la lecture du dernier message reçu et son affichage sous forme de toast.

```
package com.tuto.android;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.SmsMessage;
import android.widget.Toast;

public class SMSReceiver extends BroadcastReceiver
{
  private final String   ACTION_RECEIVE_SMS  = "android.provider.Telephony.SMS_RECEIVED";

  @Override
  public void onReceive(Context context, Intent intent)
  {
   if (intent.getAction().equals(ACTION_RECEIVE_SMS))
   {
    Bundle bundle = intent.getExtras();
```

```
    if (bundle != null)
    {
     Object[] pdus = (Object[]) bundle.get("pdus");

     final SmsMessage[] messages = new SmsMessage[pdus.length];
     for (int i = 0; i < pdus.length; i++)  {  messages[i] = SmsMessage.createFromPdu((byte[])
pdus[i]);  }  if (messages.length > -1)
     {
      final String messageBody = messages[0].getMessageBody();
      final String phoneNumber = messages[0].getDisplayOriginatingAddress();

      Toast.makeText(context, "Expediteur : " + phoneNumber, Toast.LENGTH_LONG).show();
      Toast.makeText(context, "Message : " + messageBody, Toast.LENGTH_LONG).show();

     }
    }
  }
 }
}
```

- On récupère le message grâce aux extras de l'**intent**.
- Puis on récupère dans le **bundle**, l'extra correspondant à l'identifiant "**pdus**".
- On récupère et on parcourt tous les messages pour obtenir le dernier.
- Puis on récupère les informations souhaitées dans le dernier message.

III - Résultat

Maintenant il suffit de lancer le projet dans votre émulateur préféré et puis à l'aide du contrôle de l'émulateur vous pouvez simuler l'envoi d'un message. Vous devriez obtenir le résultat suivant :

IV - Conclusion

En espérant que ce tutoriel vous aura aidé à mieux comprendre comment fonctionnent les **broadcast receiver** sous Android.

V - Remerciements

Je tiens à remercier tout particulièrement **Feanorin** qui a mis ce tutoriel au format Developpez.com.
Merci également à **jacques_jean** ainsi qu'à **jpcheck** d'avoir pris le temps de le relire et de le corriger.

VI - Liens

Tutoriel origine sur Tutos-Android