

Chapitre 3

Intents (intentions)

- Android restreint une application à des actions bien définies. Il protège ainsi le système enfermant l'application dans son propre environnement.
- Comment faire donc si une application veut communiquer avec une autre application?
- C'est le but de l'objet « intent ».
- Cet objet permet donc à une application de communiquer entre ses différents composants (communiquer avec un service par exemple).
- Il permet aussi la communication entre les applications.
- Par ailleurs, les 4 composants fondamentaux d'Android sont activés à l'aide de ces « intents ».

- Les « intents » sont appelés dans les situations suivantes :

Lancer une activité

startActivity (intent)

Envoyer un « intent » à un composant « BroadcastReceiver »

sendBroadcast (intent)

Communiquer en arrière-plan avec un service

startService(intent)

bindService(intent)

C02 : SimpleBroadcastReceiverApp

```
private Intent broadcastIntent;  
public void onClick(View v) {  
    sendBroadcast(broadcastIntent);  
}
```

C02 : MonService

```
// cette méthode sera appelée après avoir cliqué sur le bouton "start service"  
public void démarreService(View view) {  
    // on appelle la méthode pour démarrer un service en lui passant comme  
    // argument une intention définie par le contexte actuel, et comme point  
    // d'entrée la classe définie dans le fichier MonService.java  
    startService(new Intent(getApplicationContext(), MonService.class));  
}
```

- Comment démarrer une activité à l'aide de « intent » :

```
Intent Uneactivite = new Intent (action, data);  
startActivity (Uneactivite);
```

- Les arguments d'un « intent » sont :

- Action : l'action que le destinataire doit réaliser. On peut utiliser une des actions fournies par le système, « ACTION_DIAL » pour afficher un composeur de numéros téléphoniques, « ACTION_VIEW » pour afficher une page web ou un contact, etc. Il est possible aussi d'en définir de nouvelles actions propres à l'activité.
- Data : les données sur lesquelles le traitement va avoir lieu, exprimées en « URI ». Il faudra donc utiliser « tel:1234 » pour téléphoner, « <http://www.iro.umontreal.ca> » pour afficher une page dans un navigateur, etc.

- Un « intent » peut-être « implicite » ou « explicite » :

- « implicite » : il ne définit pas une cible précise. Nous voulons visualiser une page web. Le système cherche les navigateurs disponibles sur l'appareil qui peuvent répondre à une telle demande.

```
Intent alpha = new Intent (Intent.ACTION_VIEW,  
Uri.parse("http://www.iro.umontreal.ca"));
```

- « explicite » : on cible précisément le composant que le système Android doit appeler, en utilisant pour cela un identificateur de la classe Java. Si la classe représente un « intent » d'activité, le système Android va se charger de la démarrer.

```
Intent beta = new Intent (this, HelloWorld.class);
```

Intent Filters

- Les filtres sont utilisés pour permettre au système de connaître les « intents » qu'il doit utiliser pour gérer les activités, les services et « Broadcast Receivers ». Ces éléments peuvent avoir un ou plusieurs filtres « intents ».
- Les filtres sont généralement définis dans le fichier « AndroidManifest.xml ».

C02 : AirplaneMode

```
<receiver android:name="AirplaneMode">  
  <intent-filter>  
    <action android:name="android.intent.action.AIRPLANE_MODE"/>  
  </intent-filter>  
</receiver>
```

- Pour les « BroadcastReceiver », il est possible de les définir dans le programme Java.

C02 : SimpleBroadcastReceiverApp

```
private static final String MYBROADCAST_INTENTFILTER =  
    "9android.net.mybroadcastReceiver";  
  
private BroadcastReceiver myBroadcastReceiver = new BroadcastReceiver() {...}  
  
registerReceiver(myBroadcastReceiver, new IntentFilter(MYBROADCAST_INTENTFILTER));
```

- Si un composant ne définit pas un filtre d'intention, ce composant ne peut être appelé que par un « intent » explicite.

Transfert de données

- Comme nous l'avons mentionné, un « intent » contient une action et des données. Le composant qui reçoit l'intention peut récupérer ces deux informations à l'aide des méthodes « `getAction()` » et « `getData()` » de l'objet « intent ».
- L'objet « intent » peut être récupéré par la méthode « `getIntent()` ».
- Le composant qui crée l'intention peut lui ajouter des données en redéfinissant la méthode « `putExtra()` ».

C02 : SimpleBroadcastReceiverApp

```
public void onReceive(Context context, Intent intent) {  
    // TODO Auto-generated method stub  
    showToast(intent, getExtras().get("9Android.net").toString());  
}  
  
broadcastIntent.putExtra("9Android.net",  
    "This is BroadcastReceiver tutorial of 9Android.net!");
```

Les exemples associés à ce chapitre sont comme suit:

- « C03: Intents » : utilisation des intentions
- « C03 : EntryPoint » : une série d'activité avec passage de valeurs
- « C03 : DeuxActivIntent » : deux activités et passage de valeurs entre elles