Développement Android avec Kotlin

Cours - 05 - Broadcast Receiver

Jordan Hiertz

Contact

hiertzjordan@gmail.com

jordan.hiertz@al-enterprise.com

Broadcast Receiver

L'un des quatre composants principaux d'Android.

Activités

Services

Content Providers

Retour rapide sur les Intents

1. Intent Explicite

- Utilisé pour démarrer une activité ou un service spécifique.
- La cible est désigné spécifiquement (par son nom de classe).

2. Intent Implicite

- Utilisé pour effectuer une action sans indiquer de cible.
- Le système choisit le composant approprié.

3. Broadcast Intent

- Utilisé pour partager des évènements ou des états.
- Envoie les messages vers les composants qui se déclarent intéressés.

Broadcast Intent - envoyés par le système

Le système génère des Broadcast Intents pour notifier toutes les applications intéressées par un événement particulier, par exemple :

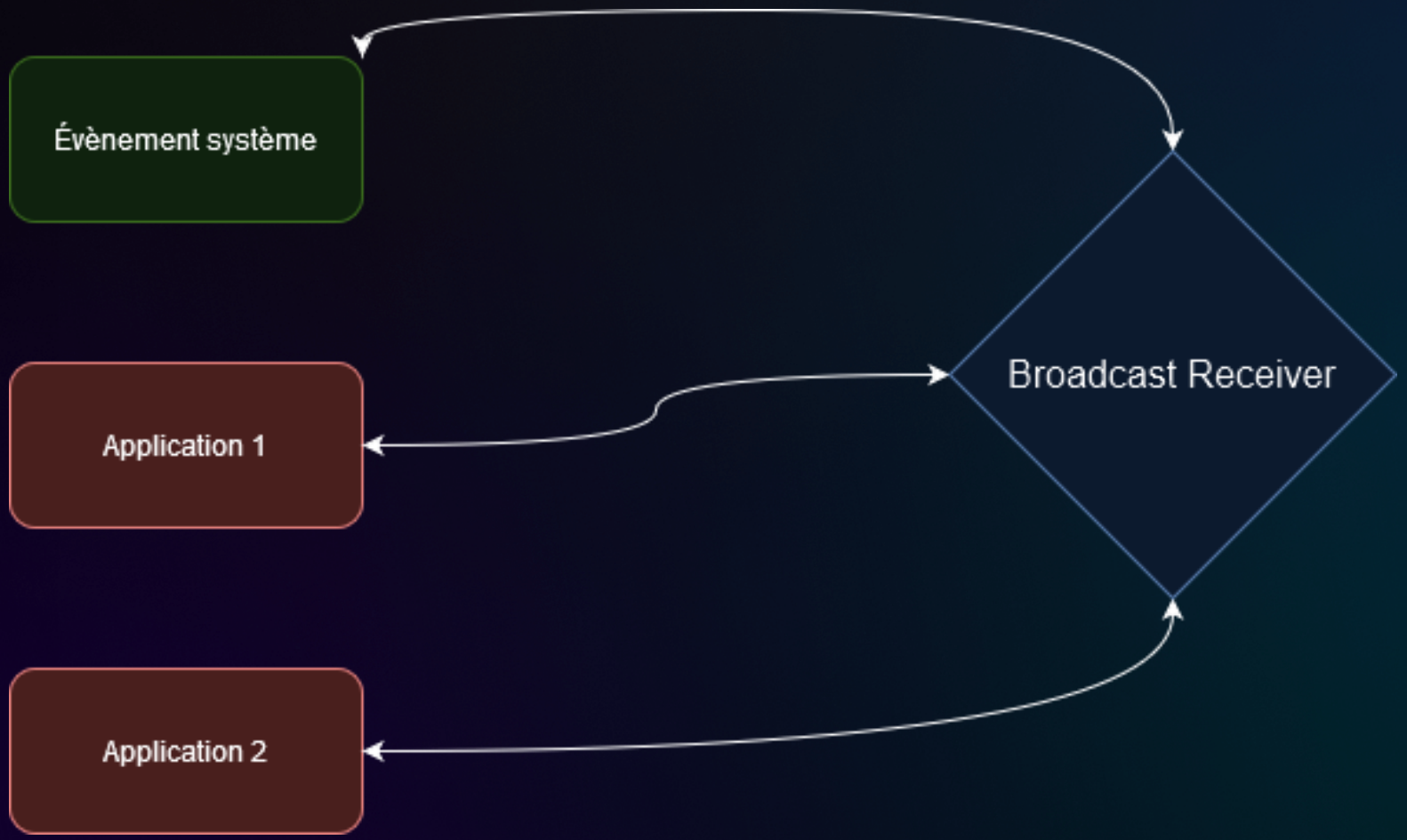
- Activation ou désactivation du **GPS** ou du **Wi-Fi**
- Réception d'un **nouveau SMS**
- Changement de l'état de la **batterie**

Broadcast Intent - envoyés par des applications

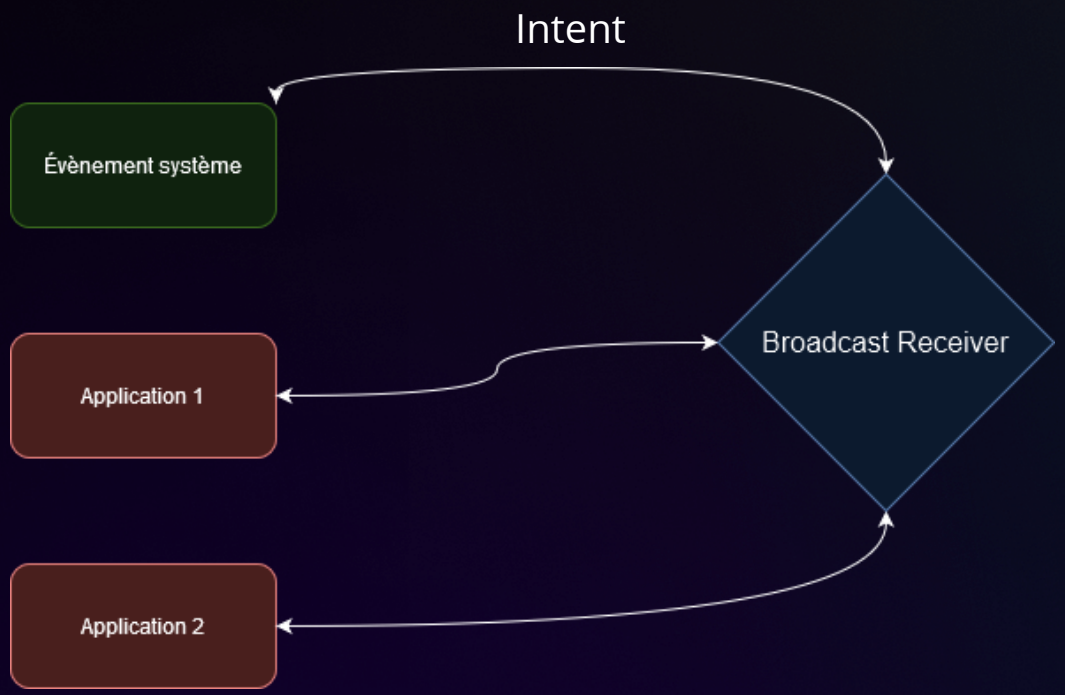
Les applications peuvent émettre leurs propres Broadcast Intents pour communiquer avec d'autres applications ou leurs propres composants. Par exemple :

- Avertir qu'un **téléchargement en arrière-plan** est terminé
- Indiquer que la **synchronisation** des données est complète
- Informer que l'utilisateur s'est déconnecté ou a changé de session

Broadcast Receiver



Broadcast Receiver



Permet de **réagir** aux événements système ou d'application diffusés globalement (Broadcasts).

Il reçoit les **Broadcast Intents** pour lesquels il s'est enregistré.

Broadcast Receiver

Pour être reconnu par le système, un **Broadcast Receiver** doit s'enregistrer.

L'enregistrement peut être :

- **Statique**, dans le fichier `AndroidManifest.xml`
- **Dynamique**, directement dans le code

Dans les deux cas, il faut spécifier les **intent-filters** pour filtrer et définir les événements auxquels on s'abonne.

Enregistrement statique

```
1 <!-- AndroidManifest.xml -->
2
3 <!-- android:exported="true" signifie que le composant est
4  exposé au système ou à d'autres applications -->
5 <receiver android:name=".BootReceiver" android:exported="true">
6     <intent-filter>
7         <action android:name="android.intent.action.BOOT_COMPLETED" />
8     </intent-filter>
9 </receiver>
```

Enregistrement statique

```kotlin
class BootReceiver : BroadcastReceiver() {
    override fun onReceive(context: Context, intent: Intent) {
        if (Intent.ACTION_BOOT_COMPLETED == intent.action) {
            // Code exécuté après le Boot
        }
    }

    companion object {
        private const val LOG_TAG = "BootReceiver"
    }
}
```

Enregistrement dynamique

```kotlin
val screenStateReceiver = object : BroadcastReceiver() {
    override fun onReceive(context: Context, intent: Intent) {
        if (Intent.ACTION_SCREEN_ON == intent.action) {
            // Exécuté quand l'écran est allumé
        }
    }
}

applicationContext.registerReceiver(screenStateReceiver, IntentFilter(Intent.ACTION_SCREEN_ON))


// Ne pas oublier d'enlever le receiver plus tard
applicationContext.unregisterReceiver(screenStateReceiver, IntentFilter(Intent.ACTION_SCREEN_ON))
```

Envoyer des annonces

La méthode **`sendBroadcast`** envoie des diffusions à **tous les récepteurs enregistrés**, dans un **ordre indéfini**.

Les récepteurs :

- **Ne peuvent pas interrompre** la diffusion.
- **Ne peuvent pas partager ou modifier** les données pour d'autres récepteurs.

```
1 val intent = Intent("com.example.SYNC_COMPLETE")
2 intent.putExtra("timestamp", System.currentTimeMillis())
3 sendBroadcast(intent)
```

Envoyer des annonces

La méthode **`sendOrderedBroadcast`** envoie des diffusions à **un seul récepteur à la fois**, selon une **priorité définie**.

Les récepteurs :

- Peuvent **propager un résultat** au récepteur suivant.
- Peuvent **interrompre la diffusion** avant qu'elle n'atteigne d'autres récepteurs.

```
1 val intent = Intent("com.example.CHECK_BATTERY")
2 intent.putExtra("batteryLevel", 85)
3 sendOrderedBroadcast(intent, null)
```

Envoyer des annonces - Diffusion locale

`LocalBroadcastManager.sendBroadcast(Intent)`

- Envoie des diffusions uniquement au sein de **l'application**.
- N'est pas accessible par d'autres applications, ce qui garantit la confidentialité et la sécurité.
- Plus **efficace**, car les diffusions ne traversent pas le système.

Pratique pour communiquer entre différents composants d'une même application, par exemple entre un service et une activité.

```
1 val intent = Intent("com.example.SYNC_COMPLETE")
2 LocalBroadcastManager.getInstance(context).sendBroadcast(intent)
3
4 // Enregistrement ailleurs dans l'application
5 val filter = IntentFilter("com.example.SYNC_COMPLETE")
6 LocalBroadcastManager.getInstance(context).registerReceiver(syncReceiver, filter)
```

Envoyer des annonces avec des autorisations

Lorsque vous appelez `sendBroadcast` ou `sendOrderedBroadcast` vous pouvez spécifier un **paramètre d'autorisation**.

Seuls les récepteurs ayant demandé cette autorisation (via `<uses-permission>` dans leur fichier manifeste) peuvent recevoir l'annonce.

⚠️ Si l'autorisation est **dangereuse**, elle doit être accordée à l'application avant que le récepteur ne reçoive la diffusion.

```
1 val intent = Intent("com.example.SYNC_COMPLETED")
2 intent.putExtra("syncTime", System.currentTimeMillis())
3 context.sendBroadcast(intent, android.Manifest.permission.INTERNET)
```

Conclusion

À retenir :

- Les **Broadcast Receivers** sont un moyen puissant d'écouter et de répondre à des événements dans l'environnement Android.
- **Les diffusions** peuvent être envoyées de manière ordonnée ou non ordonnée, avec la possibilité d'appliquer des **priorités** ou des **autorisations** pour contrôler qui peut recevoir l'annonce.

Ce qu'on a vu :

- **Types de Broadcast Receivers** : statiques (enregistrés dans le manifeste) et dynamiques (enregistrés dans le code).
- **Méthodes d'envoi des diffusions** : `sendBroadcast()`, `sendOrderedBroadcast()` et `sendBroadcast(Intent, String)`.