Communication entre applications : la classe Intent

Kamal Mehaoued

Université de Béjaia

Troisième année licence informatique

2017/2018

Table des matières

Introduction

Les Intents forment un mécanisme puissant et complet permettent aux activités et aux applications de communiquer entre elles.
Un objet intent issu de la classe Intent véhicule toutes les informations nécessaires à la réalisation d'une action.

1. Informations destinées au composant récepteur de l'intent(action à effectuer et les données associées).
2. Informations destinées au système pour le traitement de l'intent(catégorie du composant cible du message).

Introduction

Le démarrage d'un composant d'une application(Activité, service ...) est réalisé au moyen d'un objet intent.
Le choix d'un composant cible externe sera ainsi décidé au moment de l'exécution, ce qui rend dynamique l'utilisation des applications.
Les intents sont envoyés au système de deux façon différentes : mode explicite et mode implicite.
Un système de filtres est défini au niveau de chaque application afin de gérer et de filtrer les intents qui sont pertinents pour elle.
De cette façon, une application même en état d'inactivité peut être à l'écoute des intents circulant dans le système.

Fonctionnement

Les objets Intents ont trois utilisations principales :

1. Démarrer une activité au sein de la même application
2. Solliciter d'autres applications
3. Envoyer des informations

Fonctionnement

Le démarrage d'une activité au sein d'une même application est utilisé pour naviguer entre les écrans d'une même application.
C'est le cas du mode explicite.
Lorsqu'un besoin ne peut être satisfait à l'intérieur d'une application, d'autres applications peuvent êtres sollicitées.
Le système va se charger de trouver le composant et l'application les plus appropriés pour répondre au besoin.
Cela s'appelle résolution d'intentions.

Fonctionnement

Les Intents sont aussi utilisés pour démarrer des services
mettre la lecture d'une musique en pause en cas d'appel entrant.
Il est également possible de diffuser une information à l'ensemble des applications ouvertes du système
les informer d'un événement (défaillance d'une batterie par exemple)
Chaque application met en place un filtre permanent pour ne conserver que les Intents qu'elle juge nécessaire.

Une application est un ensemble d'écrans(activités) qui défilent l'un après l'autre
Le principal rôle d'un Intent est le démarrage de ces activités(une à la fois)
Il existe deux façons de démarrer une activté :

1. Vouloir connaître le résultat de l'activité démarrée
2. Sans s'intéresser au résultat de l'activité démarrée

Afin de démarrer une activité sans retour, On utilise la méthode startActivity
qui prend comme paramètre un objet de la classe Intent
qui précise le type de la classe de l'activité à éxecuter

```
Intent intent = new Intent(this, ActiviteAdemarrer.class);
startActivity(intent);
```

Pour pouvoir démarrer une activité, il faut qu'elle soit déclarée dans le fichier de configuration (AndroidManifest.xml)

A défaut de quoi une exception ActivityNotFoundException est générée lors de l'appel de startActivity

Exemple de déclaration d'une activité dans le manifest.

```
<application ...>
        <activity android :name=".MonActivite" >
<application>
```

Initialiser un Intent

L'initialisation d'un Intent peut se faire par les méthodes suivantes :

Initialiser un Intent

- La méthode setAction permet de spécifier une action à un Intent
- La méthode setData permet de spécifier la donnée de l'Intent
- La méthode setType permet de spécifier le type de la donnée
- La méthode setDataAndType() permet de spécifier et la donnée et le type
- La méthode addCategory() permet de spécifier une catégorie à un Intent

Une activité enfant proposant à l'utilisateur un formulaire de réponse Oui/Non.
Comment récupérer la réponse de l'utilisateur dans l'activité enfant depuis l'activité appelante.
Pour pouvoir récupérer un retour il faut utiliser la méthode startActivityForResult qui est prévue à cet effet.
Losque l'activté appelée termine sa tache, elle en avertira l'activité appelante.

Démarrer une activité avec retour

La méthode startActivity ne fournit aucun mécanisme de retour. Prenons l'exemple suivant :

Figure – Un formulaire Réponse Oui Non

```
private static final int CODE_MON_ACTIVITE = 1;
.....
Intent intent = new Intent(this, ActiviteCible.class);
startActivityForResult(intent, CODE_MON_ACTIVITE);
```

La constante CODE_MON_ACTIVITE représente l'identifiant de la requête(request code)

Ce code sera utilisé par la suite pour identifier l'activité renvoyant la valeur de retour.

Pour renvoyer la valeur de retour à l'activité principale, appelez la méthode setResult de la classe Activity
en indiquant comme paramètre le code de retour.
Plusieurs valeurs par défaut sont définies dans android.
RESULT_OK, RESULT_CANCELLED qui seront utilisés dans cet exemple.

renvoyrer une valeur de retour

```
public void onClick(View v) {
        switch(v.getId()){
                case R.id.button1 :
                        setResult(RESULT_OK);
                        break;
                case R.id.button2 :
                        setResult(RESULT_CANCELED);
                        break;
        }
}
```

Récupérer la valeur de retour

```
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {
        switch(requestCode){
                case CODE_MON_ACTIVITE :
                        switch(resultCode) {
                                case RESULT_OK :
                                        instruction1 ; return ;
                                case RESULT_CANCELED :
                                        instruction2 ; return ;
                                default :
                                        Faire quelque chose ; return ;
                        }
                        default :
                                Faire quelque chose ; return ;
        }
}
```

Solliciter d'autres applications

Faire appel à un composant d'une autre application
La solution est d'utiliser toujours un objet de la classe Intent.
C'est le système qui décide alors de l'application à utiliser pour répondre à l'intention décrite dans l'objet Intent.
Pour décider du composant le plus approprié, le système se base sur les informations spécifiées dans l'objet Intent

1. Action
2. Donnée
3. Catégorie

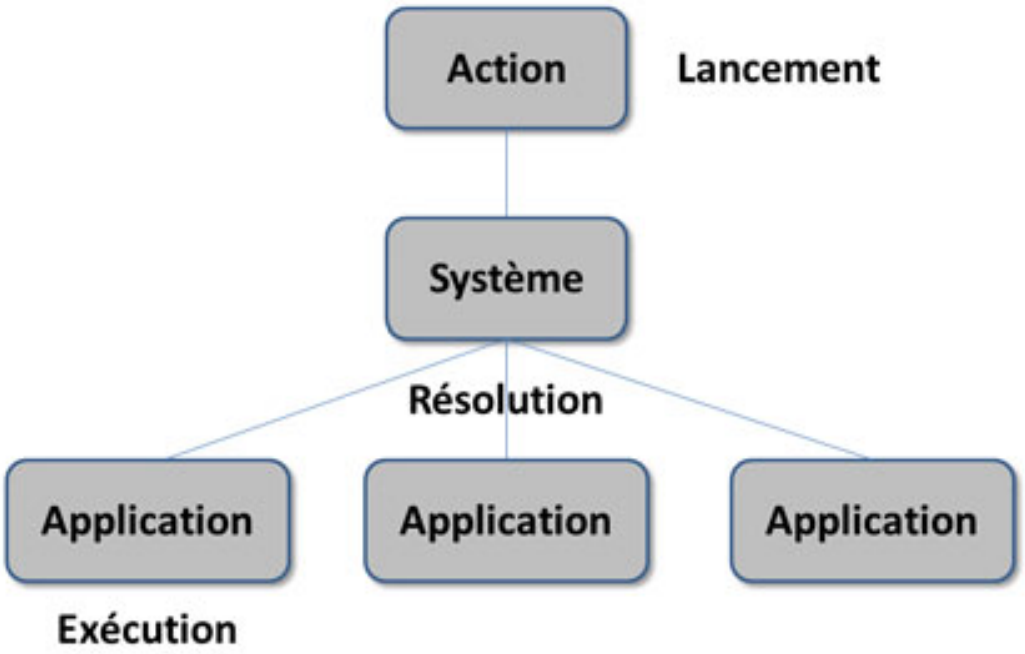

Figure – Mode de résolutions des Intents

C'est un système dutilisation d'intent implicite
Le système recourt à des filtres comme point d'entrée pour choisir les composants les plus appropriés.
Exemple : si nous souhaitons composer un numéro de téléphone, nous utilisons le type d'action ACTION_DIAL
Uri uri = Uri.parse(« tel :061000000 ») ;
Intent intent = new Intent(Intent.ACTION_DIAL,uri) ;
startActivity(intent) ;
Les URIs (Uniform Ressource Identifier) est un identifaint unique permettant d'identifer une ressource avec une syntaxe à respecter.

Les actions natives

ACTION_ANSWER : Prendre en charge un appel entrant.
ACTION_CALL : Appeler un numéro de téléphone. Cette action lance une activité affichant l'interface pour composer un numéro puis appelle le numéro contenu dans l'URI spécifiée
ACTION_DELETE : Démarrer une activité permettant de supprimer une donnée identifiée par l'URI spécifiée en paramètre
ACTION_DIAL : Afficher l'interface de composition des numéros. Celle-ci peut être pré-remplie par les données contenues dans l'URI spécifiée en paramètre.
ACTION_EDIT : Éditer une donnée.
ACTION_SEND : Envoyer des données texte ou binaire par courriel ou SMS. Les paramètres dépendront du type d'envoi.
ACTION_SENDTO : Lancer une activité capable d'envoyer un message au contact défini par l'URI spécifiée en paramètre.
ACTION_VIEW : Démarrer une action permettant de visualiser l'élément identifié par l'URI spécifiée en paramètre.

Récupérer l'uri dun intent

Uri data = getIntent().getData() ; if(data != null) Ici le code à exécuter

Démarrer un service

```
Intent intent = new Intent(this,ServiceADemarrer.class);
startService(intent);
```

Diffuser et recevoir des intents

Un intent peut être diffusé à l'ensemble des applications pour les informer sur l'environnement
appel entrant
wifi disponible
Dans ce cas ces intents sont appelés les broadcast intents
Les récèpteurs sont appelés les broadcast reveivers
Android utilise souvent ces intents pour communiquer des informations systèmes.

Recevoir et traiter des intents diffusés

```
public final class MyBroadcastReceiver extends BroadcastReceiver {
        public void onReceive(Context context, Intent intent) {
                ... // Insérer le code de traitement de l'Intent ici.
        }
}
```

Les messages informatifs natifs

ACTION_BOOT_COMPLETED : Est diffusée lorsque le système a fini de démarrer.
ACTION_SHUTDOWN : Indique que le système est en train de s'éteindre. Seul le système peut envoyer ce type de message
ACTION_CAMERA_BUTTON : Survient lorsque le bouton de l'appareil photo est appuyé.
ACTION_DATE_CHANGED : Survient lorsque la date a été changée.
ACTION_TIME_CHANGED : Survient lorsque l'heure a été changée.
ACTION_SCREEN_ON / OFF : Permet d'être informé lorsque que l'écran s'éteint et s'allume.
ACTION_POWER_CONNECTED / DISCONNECTED : Diffusée lorsque l'alimentation électrique a été branchée / débranchée.
ACTION_MEDIA_MOUNTED / UNMOUNTED : Survient lorsque qu'un support de stockage, une carte SD par exemple, a été correctement monté ou démonté