



Institut Universitaire de Technologie

Département Informatique

Site de Bourg-en-Bresse



android

TP 4 : Navigation entre activités : les Intentions

Semestre 4

lionel.buathier@univ-lyon1.fr

Les intentions (*Intent*)

- ▶ Un Intent est un message système permettant la communication entre des composants Android (Activité, Service et Broadcast receiver).
- ▶ Il est émis par le terminal pour prévenir les différentes applications du déclenchement d'évènements.
- ▶ On peut créer ses propres *Intent*, afin de lancer d'autres activités (par exemple).



Les intentions (Intent) suite...

- ▶ Il existe deux types *d'Intent* :
 - **Explicite** : appeler directement une activité connue pour exécuter une action particulière.
 - **Implicite** : demander au système quelle activité (ou service) peut exécuter une action spécifique (par exemple, ouvrir un fichier PDF).

Les Filtres d'intentions (Intent-filter)

- ▶ Les filtres d'intentions servent à indiquer à une activité, service ou broadcast receiver quelles intentions (*Intent*) ils peuvent implicitement traiter.
 - ⇒ tout composant souhaitant être prévenu par des intentions doit déclarer des filtres d'intentions.
- ▶ On déclare les filtres d'intentions dans le manifeste (AndroidManifest.xml) avec la balise **intent-filter** (cf. ex).
- ▶ On peut aussi créer des filtres d'intentions dynamiquement avec à la classe **IntentFilter**.



Les Filtres d'intentions (Intent-filter) suite...

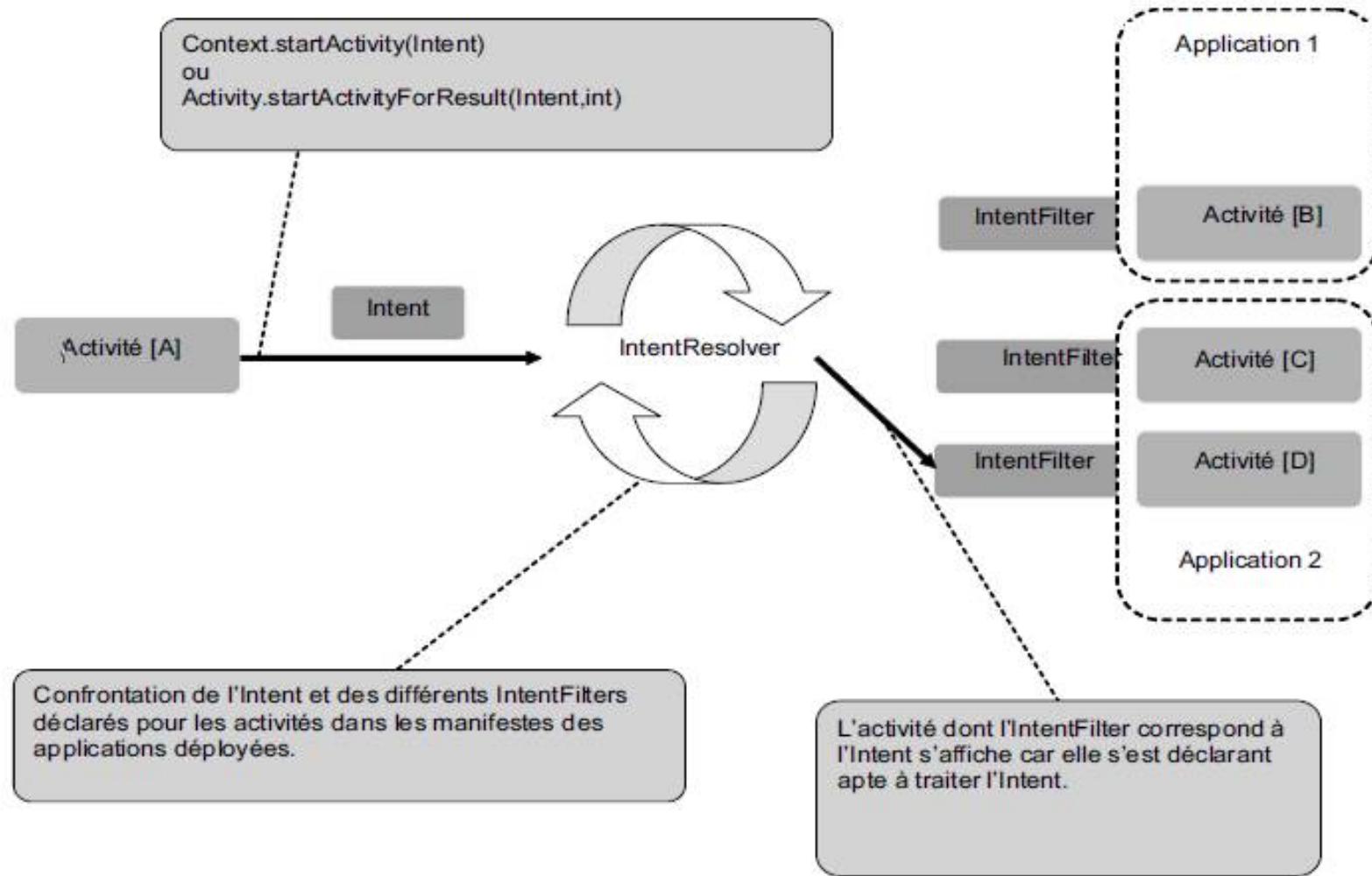
- ▶ Exemple : lors de la création d'un projet Android, l'activité principale du projet contient des filtres d'intentions.

```
<activity android:name=".HelloAndroidActivity"  
         android:label="@string/app_name" >  
<intent-filter>  
    <action android:name="android.intent.action.MAIN" />  
    <category android:name="android.intent.category.LAUNCHER" />  
</intent-filter>  
</activity>
```

- ▶ Ces deux filtres d'intentions signifient respectivement que l'activité :
 - Est la principale de l'application.
 - Appartient à la catégorie *Launcher*, c'est-à-dire qu'elle sera disponible en raccourci depuis le menu principal de lancement d'application d'Android.



Processus de résolution des intentions



Communication entre activités

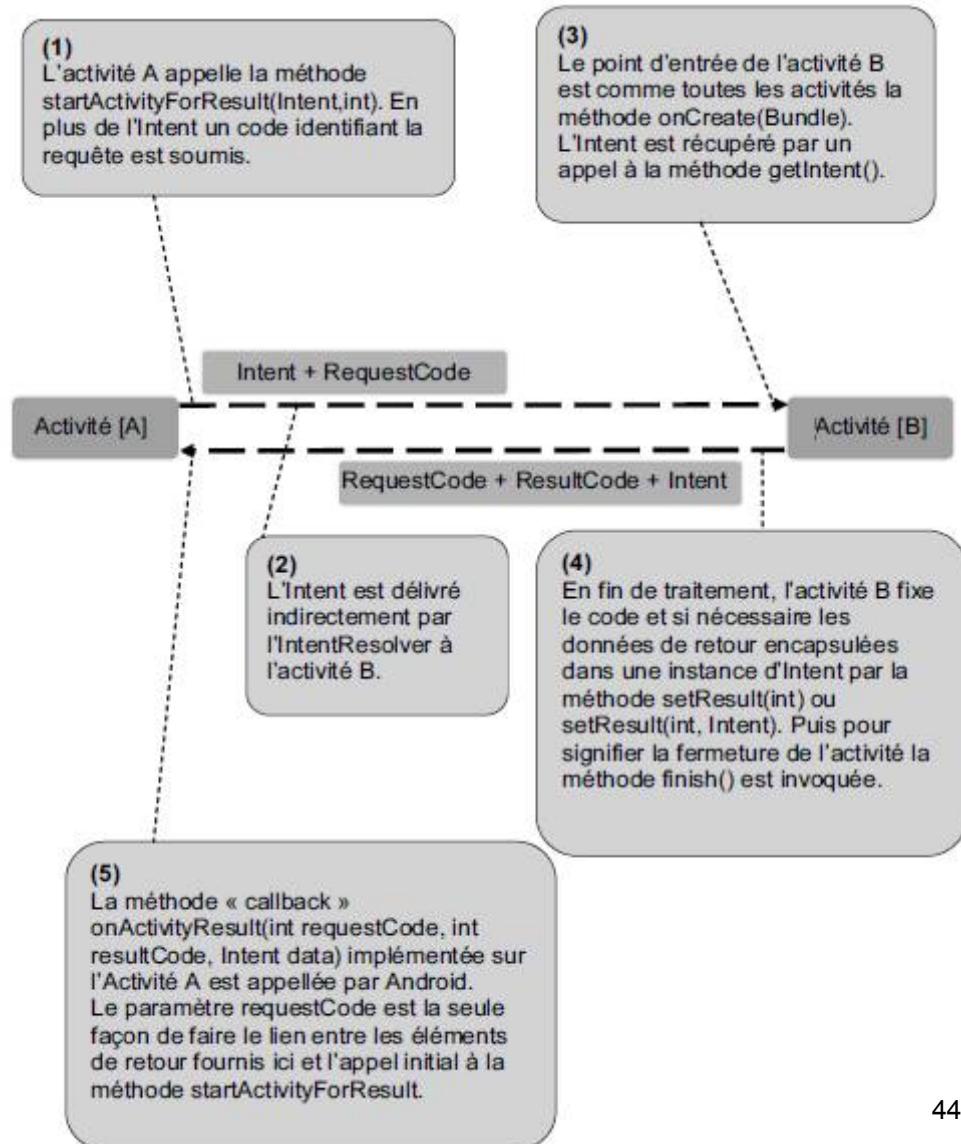
- ▶ Pour lancer une activité avec transmission d'un message, sans message de retour après retour de cette activité :
 - Activity A :
 - ⇒ Intent myIntent = new Intent(MainActivity.this, SecondaryActivity.class)
 - ⇒ myIntent.putExtra(" MessageKey ", " le message ");
 - ⇒ startActivityForResult(myIntent);
 - Activity B :
 - ⇒ Récupération du message :

```
String maChaine = getIntent().getStringExtra("MessageKey");
```
 - Pour revenir de l'activité B (en détruisant l'activity B):
 - ⇒ finish() ; // le bouton back appelle finish()



Communication entre activités

- ▶ Pour lancer une activité avec message de retour ⇔
- ▶ Le *Request Code* permet de savoir de quelle Activity on revient, sans avoir à passer d'extra (renvoyé par *setResult*)



Communication entre activités

► Activity A

Lancement de l'activité B :

```
Intent myIntent =  
    new Intent(ActA.this, ActB.class);  
myIntent.putExtra(KEY_FROM_A, message);  
startActivityForResult(myIntent, RQC_B);
```

Méthode de call back :

```
... onActivityResult(int requestCode, int resultCode, Intent data) {  
    if (requestCode == RQC_B && resultCode == RESULT_OK) {  
        String result = data.getStringExtra(KEY_FROM_B);  
    }  
}
```

► Activity B

Dans onCreate ou onStart():

```
String str =  
    getIntent().getStringExtra(KEY_FROM_A);
```

en quittant l'activité par le bouton back :

```
@Override  
public void finish() {  
    Intent intentB = new Intent();  
    intentB.putExtra(KEY_FROM_B, messB);  
    setResult(RESULT_OK, intentB);  
    super.finish();
```



Ressources

- ▶ Pour démarrer :

<http://developer.android.com/training/basics/firstapp/starting-activity.html>

- ▶ Plus complet :

<http://developer.android.com/guide/components/intents-filters.html>

- ▶ La doc de référence de l'API :

<http://developer.android.com/reference/android/content/Intent.html>

- ▶ Un bon tutoriel, de Lars Vogel (LA référence) :

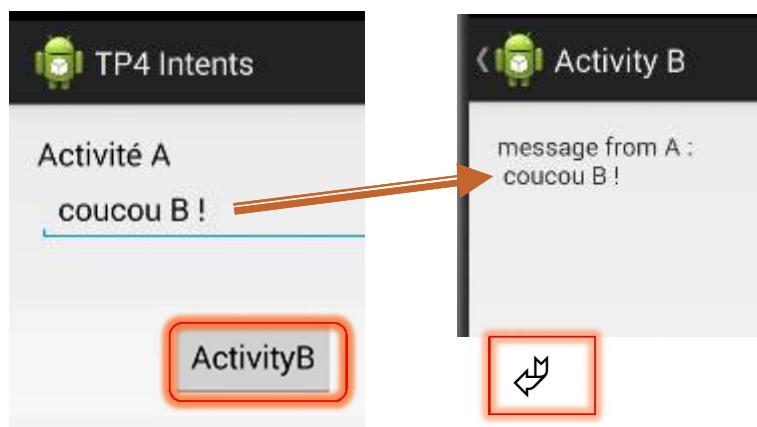
<http://www.vogella.com/articles/AndroidIntent/article.html>



Exercices

- ▶ 1. Créer un nouveau projet qui comporte 2 activités contenant chacune un bouton. Naviguer d'une activité à l'autre grâce aux boutons. Penser à laisser Android gérer le retour ou à défaut, utiliser la méthode finish() pour l'activité secondaire (et pas un intent de retour).
- ▶ 2. Ajouter un EditText dans l'activité A et un TextView dans l'activité B. Le but étant d'afficher le texte tapé dans l'activité A vers l'activité B en le transmettant avec la méthode putExtra().

Exemple d'échange :

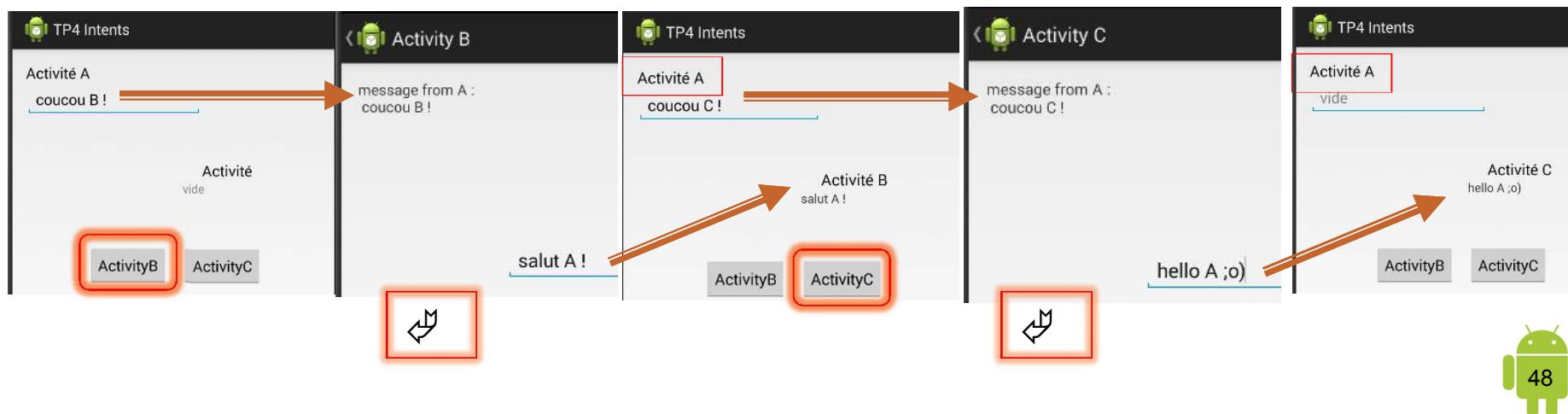


Exercices (suite)

- 3. Ajouter une 3^e activité avec un TextView et un EditText. Dans l’activité A, ajouter également deux TextView (label et message en provenance des autres activités), ainsi qu’un 2^e bouton pour envoyer le message vers l’activité C.

On doit pouvoir maintenant taper un texte dans l’activité A, l’envoyer vers l’activité B ou C, puis retourner un message de l’une de ces activités vers l’activité A et l’afficher dans la zone de texte. Pour connaître l’activité de provenance du message, insérez son nom en entête du message.

Exemple d’échange :



Exercices (suite)

- ▶ 4. Modifier le projet du TP précédent de manière à lancer une nouvelle activité permettant d'afficher la synthèse des informations saisies dans le formulaire.
- ▶ 5. Ajouter un bouton pour envoyer une confirmation sous forme de sms (ou de mail) à la personne (le numéro de téléphone de l'AVD est affiché sur celui-ci ; généralement 5554).

http://www.tutorialspoint.com/android/android_sending_sms.htm

!! Penser à déclarer les permissions et les valider dans les paramètres du téléphone (si API ≥23)

<https://x-team.com/blog/android-runtime-permissions/>

- ▶ 6. Remplacer l'avatar du formulaire par une photo présente dans la galerie.

<http://stackoverflow.com/questions/5309190/android-pick-images-from-gallery>

- ▶ 7. En plus du choix précédent, permettre de prendre une photo directement avec le mode camera du téléphone.

<http://developer.android.com/training/camera/photobasics.html>

<http://developer.android.com/guide/topics/media/camera.html#intents>

<http://blog-emildesign.rhcloud.com/?p=590>



Runtime Permissions (Android 6 ->)

- ▶ Depuis Android 6, on peut autoriser les permissions "critiques" indépendamment les unes des autres.
- ▶ Mais celles-ci sont désactivées par défaut.
- ▶ On peut le faire depuis les paramètres du téléphone, mais il est recommandé de le proposer à l'utilisateur depuis l'application, et à chaque fois qu'on a besoin d'accéder à la ressource critique (Runtime Permissions)
- ▶ Ressources :

<https://developer.android.com/training/permissions/requesting.html>

<https://x-team.com/blog/android-runtime-permissions/>

Une solution élégante avec une classe abstraite :

<http://www.truiton.com/2016/04/obtaining-runtime-permissions-android-marshmallow-6-0/>

