4. Principe des intentions : Les intents

- 4.1 Principe des Intents
- 4.2 Intents pour une nouvelle activité
- 4.3 Ajouter des informations
- 4.4 Caractéristiques supplémentaires
- 4.5 Broadcaster des informations
- 4.6 Recevoir et filtrer les Intents

4.1 Le principe des intents

Les *Intents* permettent de gérer l'envoi et la réception de messages afin de faire coopérer les applications ou les composantes d'une même application. Le but des *Intents* est de déléguer une action à un autre composant, une autre application ou une autre activité de l'application courante.

Un objet **Intent** peut contenir les information suivantes:

- le nom du composant ciblé (classe)
- l'action à réaliser, sous forme de chaîne de caractères
- les données: contenu MIME et URI
- des données supplémentaires sous forme de paires clef/valeur (extras)
- une catégorie pour cibler un type d'applications
- des drapeaux (flags:informations supplémentaires)

On peut envoyer des *Intents* informatifs pour faire passer des messages. Mais on peut aussi envoyer des *Intents* servant à lancer un composant d'une application (activité, service...)

4.2 Intent pour lancer une activité

Il y a plusieurs façons de créer l'objet de type *Intent* qui permettra de lancer une nouvelle activité. Si l'on passe la main à une activité interne à l'application, on peut créer l'Intent et passer la classe de l'activité ciblée par l'Intent:

```
Intent login = new Intent(this,
GiveLogin.class);
startActivity(login);
```

Le premier paramètre de construction de l'Intent est en fait le contexte de l'application. Dans certains cas, il ne faut pas mettre **this** mais faire appel à **getApplicationContext()** si l'objet manipulant l'*Intent* n'hérite pas de **Context**.

S'il s'agit de passer la main à une autre application, on donne au constructeur de l'Intent les données et l'URI cible: l'OS est chargé de trouver une application pouvant répondre à l'Intent.

```
Button b = (Button)findViewById(R.id.Button01);
b.setOnClickListener(new OnClickListener() {
public void onClick(View v) {
  Uri telnumber = Uri.parse("tel:0248484000");
  Intent call = new Intent(Intent.ACTION_DIAL, telnumber);
  startActivity(call);
}
});
```

Il ne faut pas oublier de déclarer toute nouvelle activité dans le Manifest.

Retour d'une activité

Lorsque le bouton *retour* est pressé, l'activité courante prend fin et revient à l'activité précédente. Cela permet par exemple de terminer son appel téléphonique et de revenir à l'activité ayant initié l'appel.

Au sein d'une application, une activité peut vouloir récupérer un code de retour de l'activité "enfant". On utilise pour cela la méthode **startActivityForResult** qui envoie un code de retour à l'activité enfant. Lorsque l'activité parent reprend la main, il devient possible de filtrer le code de retour dans la méthode **onActivityResult** pour savoir si l'on revient ou pas de l'activité enfant.

L'appel d'un *Intent* devient donc:

```
  Intent login = new Intent(getApplicationContext(),
GivePhoneNumber.class);
  startActivityForResult(login,48);
```

Le filtrage dans la classe parente permet de savoir qui avait appelé cette activité enfant, et quel activité enfant vient d'être fermée :

```
protected void onActivityResult(int requestCode, int
resultCode, Intent data)
{
  if (requestCode == 48)
    Toast.makeText(this, "Code de requête récupéré",
                   Toast.LENGTH_LONG).show();
}
```

Retour d'une activité

Il est aussi possible de définir un résultat d'activité, avant d'appeler la méthode **finish()**. Dans l'activité enfant, on met donc:

```
Button finish =
(Button)findViewById(R.id.finish);
  finish.setOnClickListener(new
OnClickListener() {

  @Override
  public void onClick(View v) {
    setResult(50);
    finish();
  }});
```

Et la classe parente peut filtrer ainsi:

```
protected void onActivityResult(int
requestCode, int resultCode, Intent data)
{
  if (requestCode == 48)
    Toast.makeText(this, "Code de requête
récupéré (je sais d'ou je viens)",
                Toast.LENGTH_LONG).show();
  if (resultCode == 50)
    Toast.makeText(this, "Code de retour
ok (on m'a renvoyé le bon code)",
                Toast.LENGTH_LONG).show();
}
```

4.3 Transmettre des informations

Les *Intent* permettent de transporter des informations à destination de l'activité cible. On appelle ces informations des *Extra*: les méthodes permettant de les manipuler sont **getExtra** et **putExtra**. Lorsqu'on prépare un *Intent* et que l'on souhaite ajouter une information de type "clef -> valeur" on procède ainsi:

```
Intent callactivity2 = new
Intent(getApplicationContext(), Activity2.class);
callactivity2.putExtra("login", "jfl");
startActivity(callactivity2);
```

Du côté de l'activité recevant l'Intent, on récupère l'information de la manière suivante:

```
Bundle extras = getIntent().getExtras();
String s = new String(extras.getString("login"));
```

4.4 Caractéristiques d'un Intent : l'action

Le premier paramètre de construction de l'*Intent* est le type de l'action véhiculé par cet Intent. Ces types d'actions peuvent être les actions natives du système ou des actions définies par le développeur.

Plusieurs actions natives existent par défaut sur Android. La plus courante est l'action **Intent.ACTION_VIEW** qui permet d'appeler une application pour visualiser un contenu dont on donne l'URI.

Dans ce cas ou l'on utilise une action native, il faut souvent ajouter des informations supplémentaires à l'aide de **putExtra**:

```
Intent emailIntent = new
Intent(android.content.Intent.ACTION_SEND);
String[] recipients = new String[]{"my@email.com", "",};
emailIntent.putExtra(android.content.Intent.EXTRA_EMAIL,
recipients);
emailIntent.putExtra(android.content.Intent.EXTRA_SUBJECT,
"Test");
emailIntent.putExtra(android.content.Intent.EXTRA_TEXT,
"Message");
emailIntent.setType("text/plain");
startActivity(Intent.createChooser(emailIntent, "Send
mail..."));
finish();
```

Pour définir une action personnelle :

```
Intent monIntent = new Intent("nom_du_message");
```

Exemples de combinaisons Action+Data

- `ACTION_VIEW` ***content://contacts/people/1*** -- Afficher les informations du contact dont l'identifiant est "1".
- `ACTION_DIAL` ***content://contacts/people/1*** -- Afficher le dialer avec le contact dont l'id est "1".
- `ACTION_VIEW` ***tel:0661876876*** -- Affiche le "Dialer" pour lancer un appel vers la personne dont le numéro de téléphone est donné dans l'URI. L'action VIEW cherche l'application la plus appropriée pour afficher la donnée de l'URI
- `ACTION_DIAL` ***tel:0661876876*** -- Affiche le "Dialer" pour lancer un appel vers le numéro de téléphone donné dans l'URI
- `ACTION_EDIT` ***content://contacts/people/1*** -- Editer les informations de la personne dont l'identifiant est "1".
- `ACTION_VIEW` ***content://contacts/people/*** -- Afficher la liste de contacts, Cette opération est typique comme première étape pour afficher les détails d'un contact, ou pour récupérer une information particulière d'un contact.Ensuite on peut afficher les information de ce contact avec un nouvel Intent du genre {`ACTION_VIEW` ***content://contacts/people/N*** }.

Caractéristiques d'un Intent : Les catégories

Les catégories d'*Intent* permettent de grouper les applications par grands types de fonctionnalités (clients emails, navigateurs, players de musique, Messageries, Appels, etc...). Par exemple, on trouve les catégories suivantes qui permettent de lancer:

- **DEFAULT**: catégorie par défaut
- **BROWSABLE**: une activité qui peut être invoquée depuis un clic sur un navigateur web, ce qui permet d'implémenter des nouveaux types de lien, e.g. foo://truc
- **APP_CALCULATOR**: une activité qui permet d'effectuer des calcules.
- **APP_CALENDAR**: une activité qui peut servir d'agenda.
- **APP_MUSIC**: une activité qui permet de parcourir et jouer de la musique
- ...

Exercice

- En utilisant votre projet précédemment réalisé, ajoutez un bouton “appel” permettant de passer un appel au profil actuel. Le numéro de téléphone peut être récupéré avec le Webservice.
-

- Créez une nouvelle activité MatiereActivity qui permet d’ajouter une nouvelle Matière/Note à un étudiant. L’activité doit afficher le nom de l’étudiant dans son titre (à transmettre via Intent).
- Une fois l’activité fermée l’activité principale doit récupérer la nouvelle Matière/Note et l’ajouter à la liste des notes (mode paysage).