# Programmation Android
# **V. Intents / Intent Filters**

# Plan


1. Cas d'utilisation
2. Intent explicite
3. Intent implicite
4. Interception d'un Intent : `<intent-filter>`
5. Résolution des intents implicites
6. Exemples d'Intents implicites usuels
7. `Start Activity For Result`
8. Autres exemples

# android.content.Intent

API

## Définition

- Un ***Intent*** est une **demande d'action** à réaliser par un autre composant de l'OS (*activity*, *service*, *content provider*, *broadcast receiver*)
- En outre, l'***Intent* contient les données** à utiliser par l'action.

# Plan


1. Cas d'utilisation
2. Intent explicite
3. Intent implicite
4. Interception d'un Intent : `<intent-filter>`
5. Résolution des intents implicites
6. Exemples d'Intents implicites usuels
7. Start Activity For Result
8. Autres exemples

# android.content.Intent API

## Cas d'utilisation 1 : démarrage d'une activité (avec UI)

1. `startActivity(Intent)`
2. `startActivityForResult(Intent, int)`

- L'intent **décrit l'activité à lancer**
- (2) le résultat obtenu est transmis à l'activité initiatrice via un autre *intent* passé à sa méthode **`onActivityResult (int requestCode, int resultCode, Intent data)`**

# android.content.Intent API

## Cas d'utilisation 2 : démarrage d'un service (sans UI)

1. `startService(Intent)`
2. `bindService(Intent, ServiceConnection, int)`

- L'intent **décrit le service à lancer**
- (2) : pour les services de type client / serveur (dépendance entre l'application et le service)

# android.content.Intent ▸ API

## Cas d'utilisation 3 : notification d'un événement à tout l'OS (broadcast)

1. `sendBroadcast(Intent)`
2. `sendOrderedBroadcast(Intent, String)`

- L'intent **décrit l'événement**
- exemples : wifi on/off, boot terminé, prise de vue effectuée, batterie critique, etc.
- (2) : avec un ordre de préférence (géré par le système) sur les receveurs

# Plan


1. Cas d'utilisation
2. **Intent explicite**
3. Intent implicite
4. Interception d'un Intent : `<intent-filter>`
5. Résolution des intents implicites
6. Exemples d'Intents implicites usuels
7. Start Activity For Result
8. Autres exemples

# Intent explicite

## Principe

- Définit le composant à activer en spécifiant explicitement son ***nom Java complet*** **(package + classe)**
- exemples : un composant de l'application, un service identifié, etc.
- par exemple par l'intermédiaire d'un objet de type `Class` : utilisation du constructeur `Intent(Context, Class<?>)` :

# Intent explicite, exemples de construction :

```
// Executed in an Activity, so 'this' is the Context
// The fileUrl is a string URL, such as "http://www.example.com/image.png"
Intent downloadIntent = new Intent(this, DownloadService.class);
downloadIntent.setData(Uri.parse(fileUrl));
startService(downloadIntent);
```

- voir aussi Intent.setComponent(android.content.ComponentName)
- voir aussi Intent.setClass(android.content.Context, java.lang.Class<? >)
- voir aussi Intent.setClassName(java.lang.String, java.lang.String)

# Intent explicite

**Remarque**

- Si le composant cible n'est pas spécifié, l'intent est considéré de facto comme implicite
- Lors de l'appel à un service, il est fortement recommandé de toujours spécifier celui-ci explicitement afin d'être certain du service activé

# Plan

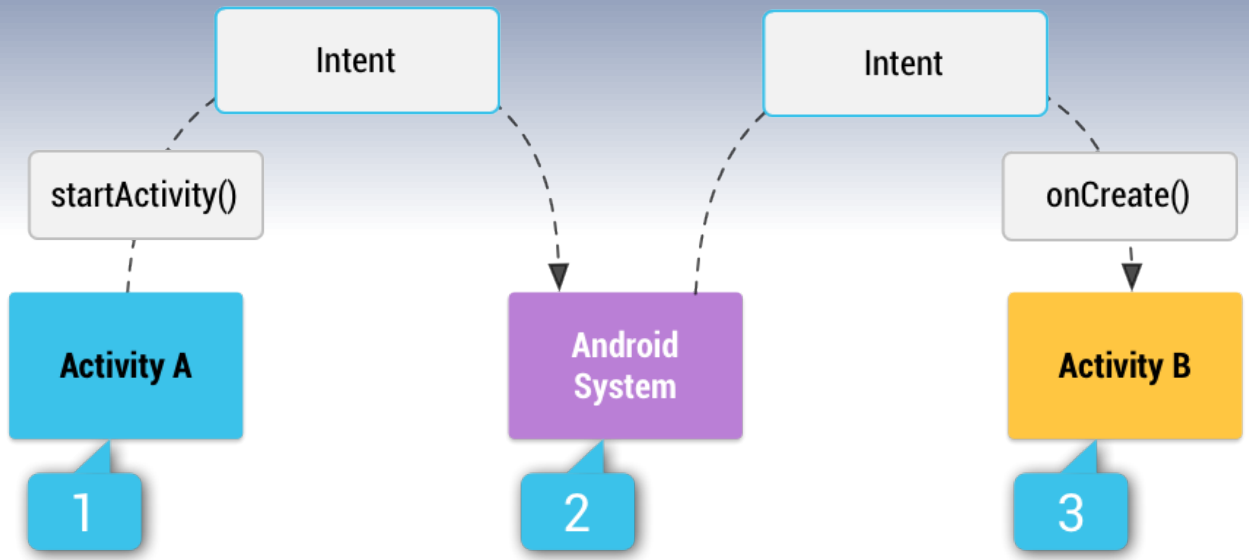

1. Cas d'utilisation
2. Intent explicite
3. **Intent implicite**
4. Interception d'un Intent : `<intent-filter>`
5. Résolution des intents implicites
6. Exemples d'Intents implicites usuels
7. Start Activity For Result
8. Autres exemples

# Intent implicite

## Principe

- Ne spécifie pas le composant visé
- Demande à l'OS l'exécution d'une action standardisée, par un autre composant
- Pour capter ce genre de sollicitations, un composant doit s'enregistrer dans l'OS comme capable de gérer ce type d'action (manifest : intent-filter)
- Si plusieurs composants sont enregistrés pour une même action, un choix est présenté à l'utilisateur
- Exemples :
  - montrer une localisation sur une carte
  - partager $\rightarrow$ *envoyé avec* (réseau social, texto, mailer...)
  - visualiser une URL
  - etc.

# Intent implicite : cycle de vie




1. activité A : `startActivity` avec un Intent implicite
2. l'OS parcourt les applications pour trouver les **intent-filter** correspondant à l'action demandée
3. l'OS démarre l'activité trouvée, B

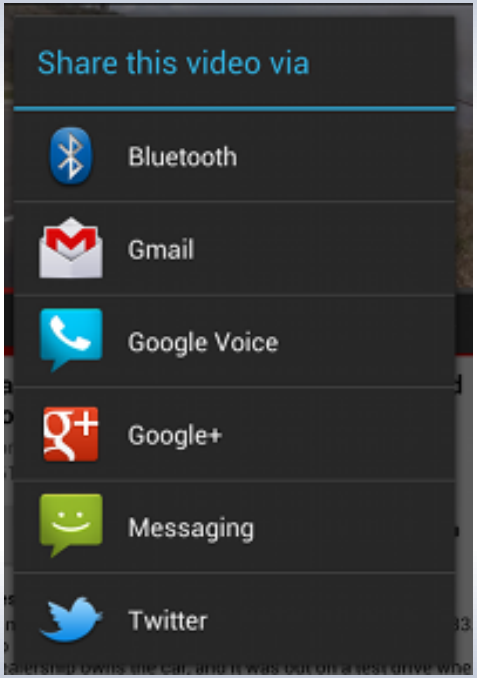
# Intent implicite, exemple de construction :

```
// Create the text message with a string
Intent sendIntent = new Intent();
sendIntent.setAction(Intent.ACTION_SEND);
sendIntent.putExtra(Intent.EXTRA_TEXT, textMessage);
sendIntent.setType(HTTP.PLAIN_TEXT_TYPE); // "text/plain" MIME type

// Verify that the intent will resolve to an activity
if (sendIntent.resolveActivity(getPackageManager()) != null) {
    startActivity(sendIntent);
}
```

Attention : si aucune application n'est disponible et que la vérification n'est pas effectuée alors l'application plante

# Action.SEND, plusieurs applications :



# Forcer la sélection avec `createChooser`

```
Intent intent = new Intent(Intent.ACTION_SEND);
...

// Always use string resources for UI text.
// This says something like "Share this photo with"
String title = getResources().getString(R.string.chooser_title);
// Create intent to show chooser
Intent chooser = Intent.createChooser(intent, title);

// Verify the intent will resolve to at least one activity
if (sendIntent.resolveActivity(getPackageManager()) != null) {
    startActivity(sendIntent);
}
```

# Plan

1 Cas d'utilisation

2 Intent explicite

3 Intent implicite

4 **Interception d'un Intent : <intent-filter>**

5 Résolution des intents implicites

6 Exemples d'Intents implicites usuels

7 Start Activity For Result

8 Autres exemples

# Réception d'un Intent implicite

## Déclaration d'un intent filter

- pouvoir recevoir un intent implicite, il faut déclarer un **<intent-filter>** dans le **manifest**
- <intent-filter> définit le **type d'intent accepté** en spécifiant :
  - **type de l'action** : balise **<action>**
  - **catégorie** de l'action : balise **<category>**
  - **type de donnée** accepté (optionnel) : balise **<data>** + divers attributs (scheme, host, port, path, etc.)

Note : Un intent explicite est toujours délivré, quelque soit le filtre associé au composant

Note : **<category>** est obligatoire pour pouvoir recevoir un intent implicite (CATEGORY_DEFAULT au pire)

# Exemple de <intent-filter>

Une activité capable de traiter les `Action.SEND` (partage) dont le contenu est de type texte :

```
<activity android:name="ShareActivity">
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
</activity>
```

Note : Il est possible d'avoir plusieurs balises de chaque type pour élargir le champ des intents capturés
Note : Il est possible de définir plusieurs filtres différents pour une même activité

# Autres exemples

Pour un launcher classique :

```
<activity android:name="MainActivity">
    <!-- This activity is the main entry, should appear in app launcher -->
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

# Autres exemples

Deux filtres pour une même activité :

```
<activity android:name="ShareActivity">
    <!-- This activity handles "SEND" actions with text data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
    <!-- This activity also handles "SEND" and "SEND_MULTIPLE" with media data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>
```

# Plan

1 Cas d'utilisation
2 Intent explicite
3 Intent implicite
4 Interception d'un Intent : `<intent-filter>`
5 **Résolution des intents implicites**
6 Exemples d'Intents implicites usuels
7 Start Activity For Result
8 Autres exemples

# Résolution des intents : recherche d'une application

**Android identifie l'activité suivant les filtres déclarés. Avec un filtrage sur :**

- L'action
- La description des données contenues dans l'intent (URI et type)
- La catégorie de l'intent

# Filtrage de l'action

Un filtre peut déclarer de zéro à plusieurs actions :

```
<intent-filter>
    <action android:name="android.intent.action.EDIT" />
    <action android:name="android.intent.action.VIEW" />
    ...
</intent-filter>
```

## réussite du test lié à l'action

- L'action de l'intent doit correspondre à l'une des actions spécifiées dans le filtre
- Si le filtre ne définit aucune action, tous les intents ratent le test

# Filtrage de la catégorie

Un filtre peut déclarer de zéro à plusieurs catégories :

```
<intent-filter>
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.BROWSABLE" />
    ...
</intent-filter>
```

## réussite du test lié à la catégorie

- Toutes les catégories définies dans l'intent doivent correspondre à l'une des catégories spécifiées dans le filtre. L'inverse n'est pas nécessaire.
- Un intent sans catégorie passe toujours ce test, car il est automatiquement associé à la catégorie `CATEGORY_DEFAULT`
- Ainsi, un filtre doit au moins avoir la catégorie `CATEGORY_DEFAULT` pour pouvoir recevoir des intents

# Filtrage du type des données

Un filtre peut déclarer de zéro à plusieurs types de données :

```
<intent-filter>
    <data android:mimeType="video/mpeg" android:scheme="http" ... />
    <data android:mimeType="audio/mpeg" android:scheme="http" ... />
    ...
</intent-filter>
```

# Détails d'une balise `<data>`

## `<data>` contient

- `android:mimeType` → type de données ▸ MIME type
- `android:scheme` → structure acceptée pour l'URI, pour chacune de ses parties :
  `<scheme>://<host>:<port>/<path>`

**`content://com.example.project:200/folder/subfolder/etc`**

- `scheme` = content
- `host` = com.example.project
- `port` = 200
- `path` = /folder/subfolder/etc

# Filtrage de l'URI

**La description de chaque partie de l'URI est optionnelle mais**

- si `scheme` n'est pas spécifié, `host` est ignoré
- si `host` n'est pas spécifié, `port` est ignoré
- si `scheme` et `host` sont tous les deux absents, `path` est ignoré

Note : il est possible d'utiliser * pour définir le type de `path` accepté

# Filtrage global de `<data>`

## Règles d'acceptation : URI + type MIME

- Si l'intent ne définit ni l'un ni l'autre, le filtre ne le doit pas non plus
- Un intent ne contenant que le MIME est accepté si le filtre ne définit pas d'URI et correspond au MIME
- Si l'intent contient les deux, le filtre le doit aussi
- le type MIME de l'intent peut être inféré de l'URI. Si le filtre contient uniquement le MIME, il supporte les URI dont le `scheme` est de type `content:` et `file:` (ressource locale)

```
<intent-filter>
    <data android:mimeType="image/*" />
    ...
</intent-filter>
```

URI locale inférée

# Autre exemple de filtre courant

Le composant est capable de gérer les URI qui correspondent à des vidéos disponibles depuis une adresse internet :

```
<intent-filter>
    <data android:scheme="http" android:type="video/*" />
    ...
</intent-filter>
```

# Plan


1. Cas d'utilisation
2. Intent explicite
3. Intent implicite
4. Interception d'un Intent : `<intent-filter>`
5. Résolution des intents implicites
6. **Exemples d'Intents implicites usuels**
7. startActivityForResult
8. Autres exemples

## android.provider.AlarmClock.ACTION_SET_ALARM

### Créer une alarme ▸ AlarmClock

```
public void createAlarm(String message, int hour, int minutes) {
    Intent intent = new Intent(AlarmClock.ACTION_SET_ALARM)
            .putExtra(AlarmClock.EXTRA_MESSAGE, message)
            .putExtra(AlarmClock.EXTRA_HOUR, hour)
            .putExtra(AlarmClock.EXTRA_MINUTES, minutes);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

### Nécessite d'ajouter dans le manifest :

```
<uses-permission android:name="com.android.alarm.permission.SET_ALARM" />
```

# ACTION_SET_ALARM :

## `intent_filter` associé

```
<activity ...>
    <intent-filter>
        <action android:name="android.intent.action.SET_ALARM" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

## android.provider.AlarmClock.ACTION_SET_TIMER

Créer un compte à rebours ▸ AlarmClock

```
public void startTimer(String message, int seconds) {
    Intent intent = new Intent(AlarmClock.ACTION_SET_TIMER)
            .putExtra(AlarmClock.EXTRA_MESSAGE, message)
            .putExtra(AlarmClock.EXTRA_LENGTH, seconds)
            .putExtra(AlarmClock.EXTRA_SKIP_UI, true);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

Nécessite d'ajouter dans le manifest :

```
<uses-permission android:name="com.android.alarm.permission.SET_ALARM" />
```

# ACTION_SET_TIMER :

## `intent_filter` associé

```
<activity ...>
    <intent-filter>
        <action android:name="android.intent.action.SET_TIMER" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

## `Intent.ACTION_INSERT`

### Ajouter un événement au calendrier

▸ Intent

```
public void addEvent(String title, String location, Calendar begin, Calendar end) {
    Intent intent = new Intent(Intent.ACTION_INSERT)
            .setData(Events.CONTENT_URI)
            .putExtra(Events.TITLE, title)
            .putExtra(Events.EVENT_LOCATION, location)
            .putExtra(CalendarContract.EXTRA_EVENT_BEGIN_TIME, begin)
            .putExtra(CalendarContract.EXTRA_EVENT_END_TIME, end);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# ACTION_INSERT : `intent_filter` associé

```
<activity ...>
    <intent-filter>
        <action android:name="android.intent.action.INSERT" />
        <data android:mimeType="vnd.android.cursor.dir/event" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Plan


1. Cas d'utilisation
2. Intent explicite
3. Intent implicite
4. Interception d'un Intent : `<intent-filter>`
5. Résolution des intents implicites
6. Exemples d'Intents implicites usuels
7. **Start Activity For Result**
8. Autres exemples

## MediaStore.ACTION_IMAGE_CAPTURE

MediaStore

```
private static final int IMAGE_CAPTURE = 2001;
private static final String PICTURE = "picture";

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);   setContentView(R.layout.activity_main);
    Button b = findViewById(R.id.camera);      b.setOnClickListener(v -> capturePhoto());
}
public void capturePhoto(){
    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    //This check does not work without adding queries in manifest starting from 33
    if(intent.resolveActivity(this.getPackageManager()) != null) {
        startActivityForResult(intent,IMAGE_CAPTURE);
    }
}
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == IMAGE_CAPTURE && resultCode == RESULT_OK) {
        Bitmap bmp = (Bitmap) data.getExtras().get("data");
        Intent intent = new Intent( packageContext: this, DisplayCapture.class);
        intent.putExtra(PICTURE, bmp);
        startActivity(intent);
    } else {
        Toast.makeText( context: this,  text: "You didn't pick an image!", Toast.LENGTH_LONG).show();
    }
}
```

# Pour des versions 33+

`intent.resolveActivity(this.getPackageManager())` nécessite pour fonctionner d'ajouter une balise < *queries* > dans le manifest :

```
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
        </activity>
    </application>
    <queries>
        <intent>
            <action android:name="android.media.action.IMAGE_CAPTURE" />
        </intent>
    </queries>
</manifest>
```

# Pour des versions 33+

`startActivityForResult` *deprecated (33)* : ▸ utilisez AndroidX

Pour vérifier l'Intent, on utilise un simple `try catch` et on gère l'exception `ActivityNotFoundException`

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    Uri location = Uri.parse("geo:0,0?q=LIRMM");
    Intent mapIntent = new Intent(Intent.ACTION_VIEW, location);

    try {
        startActivity(mapIntent);
    } catch (ActivityNotFoundException e) {
        // Define what your app should do if no activity can handle the intent.
        Toast.makeText( context: this, text: "I can't do that",Toast.LENGTH_LONG).show();
    }
}
```

# Avec AndroidX : ActivityResultContracts

## registerForActivityResult

```
public class MainActivity extends AppCompatActivity {

    ActivityResultLauncher<String> mGetContent = registerForActivityResult(new ActivityResultContracts.GetContent(),
            new ActivityResultCallback<Uri>() {
                @Override
                public void onActivityResult(Uri uri) {
                    Intent intent = new Intent( packageContext: MainActivity.this, DisplayImage.class);
                    intent.putExtra( name: "uri", uri);
                    startActivity(intent);
                    Toast.makeText( context: MainActivity.this,  text: "Thank you for getting  picture!", Toast.LENGTH_LONG).show();
                }
            });

    ActivityResultLauncher<Void> mTakePicture = registerForActivityResult(new ActivityResultContracts.TakePicturePreview(),
            new ActivityResultCallback<Bitmap>() {
                @Override
                public void onActivityResult(Bitmap data) {
                    startDisplayCapture(data);
                    Toast.makeText( context: MainActivity.this,  text: "Thank you for taking picture!", Toast.LENGTH_LONG).show();
                }
            });

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);   setContentView(R.layout.activity_main);
        Button b = findViewById(R.id.button2);
        b.setOnClickListener(v -> mGetContent.launch( input: "image/*"));
        b = findViewById(R.id.button);
        b.setOnClickListener(v -> mTakePicture.launch( input: null));
```

## MediaStore.ACTION_IMAGE_CAPTURE : intent_filter associé

Ne pas oublier : utiliser cette action nécessite qu'une application ait déclaré le filtre suivant :

```
<activity ...>
    <intent-filter>
        <action android:name="android.media.action.IMAGE_CAPTURE" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Still image mode

```
public void capturePhoto() {
    Intent intent = new Intent(MediaStore.INTENT_ACTION_STILL_IMAGE_CAMERA);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivityForResult(intent);
    }
}
```

filtre associé :

```
<activity ...>
    <intent-filter>
        <action android:name="android.media.action.STILL_IMAGE_CAMERA" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Sélection d'un contact :

```
static final int REQUEST_SELECT_CONTACT = 1;

public void selectContact() {
    Intent intent = new Intent(Intent.ACTION_PICK);
    intent.setType(ContactsContract.Contacts.CONTENT_TYPE);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivityForResult(intent, REQUEST_SELECT_CONTACT);
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == REQUEST_SELECT_CONTACT && resultCode == RESULT_OK) {
        Uri contactUri = data.getData();
        // Do something with the selected contact at contactUri
        ...
    }
}
```

# Uniquement le numéro d'un contact :

```
static final int REQUEST_SELECT_PHONE_NUMBER = 1;

public void selectContact() {
    // Start an activity for the user to pick a phone number from contacts
    Intent intent = new Intent(Intent.ACTION_PICK);
    intent.setType(CommonDataKinds.Phone.CONTENT_TYPE);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivityForResult(intent, REQUEST_SELECT_PHONE_NUMBER);
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == REQUEST_SELECT_PHONE_NUMBER && resultCode == RESULT_OK) {
        // Get the URI and query the content provider for the phone number
        Uri contactUri = data.getData();
        String[] projection = new String[]{CommonDataKinds.Phone.NUMBER};
        Cursor cursor = getContentResolver().query(contactUri, projection,
                null, null, null);
        // If the cursor returned is valid, get the phone number
        if (cursor != null && cursor.moveToFirst()) {
            int numberIndex = cursor.getColumnIndex(CommonDataKinds.Phone.NUMBER);
            String number = cursor.getString(numberIndex);
            // Do something with the phone number
            ...
        }
    }
}
```

# Récupérer/ouvrir/créer des fichiers

**`android.content.Intent`**

- ACTION_GET_CONTENT : récupère le fichier
- ACTION_OPEN_DOCUMENT : ouverture du fichier géré par une autre application, URI de type `content:`
- ACTION_CREATE_DOCUMENT : création d'une nouvelle URI de type `content:`

# Récupérer une photo (URI + données)

```
static final int REQUEST_IMAGE_GET = 1;

public void selectImage() {
    Intent intent = new Intent(Intent.ACTION_GET_CONTENT);
    intent.setType("image/*");
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivityForResult(intent, REQUEST_IMAGE_GET);
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == REQUEST_IMAGE_GET && resultCode == RESULT_OK) {
        Bitmap thumbnail = data.getParcelable("data");
        Uri fullPhotoUri = data.getData();
        // Do work with photo saved at fullPhotoUri
        ...
    }
}
```

# filtre associé

```
<activity ...>
    <intent-filter>
        <action android:name="android.intent.action.GET_CONTENT" />
        <data android:type="image/*" />
        <category android:name="android.intent.category.DEFAULT" />
        <!-- The OPENABLE category declares that the returned file is accessible
             from a content provider that supports OpenableColumns
             and ContentResolver.openFileDescriptor() -->
        <category android:name="android.intent.category.OPENABLE" />
    </intent-filter>
</activity>
```

# Ouvrir une photo (URI + données)

```
static final int REQUEST_IMAGE_OPEN = 1;

public void selectImage() {
    Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
    intent.setType("image/*");
    intent.addCategory(Intent.CATEGORY_OPENABLE);
    // Only the system receives the ACTION_OPEN_DOCUMENT, so no need to test.
    startActivityForResult(intent, REQUEST_IMAGE_OPEN);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == REQUEST_IMAGE_OPEN && resultCode == RESULT_OK) {
        Uri fullPhotoUri = data.getData();
        // Do work with full size photo saved at fullPhotoUri
        ...
    }
}
```

# Plan

1 Cas d'utilisation
2 Intent explicite
3 Intent implicite
4 Interception d'un Intent : `<intent-filter>`
5 Résolution des intents implicites
6 Exemples d'Intents implicites usuels
7 Start Activity For Result
8 Autres exemples

# VIEW / EDIT

Lancer l'affichage un contact :

```
public void viewContact(Uri contactUri) {
    Intent intent = new Intent(Intent.ACTION_VIEW, contactUri);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

Lancer l'édition un contact :

```
public void editContact(Uri contactUri, String email) {
    Intent intent = new Intent(Intent.ACTION_EDIT);
    intent.setData(contactUri);
    intent.putExtra(Intents.Insert.EMAIL, email);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# Insertion d'un contact :

```
public void insertContact(String name, String email) {
    Intent intent = new Intent(Intent.ACTION_INSERT);
    intent.setType(Contacts.CONTENT_TYPE);
    intent.putExtra(Intents.Insert.NAME, name);
    intent.putExtra(Intents.Insert.EMAIL, email);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# Composer un mail

ACTION_SENDTO (for no attachment)
ACTION_SEND (for one attachment)
ACTION_SEND_MULTIPLE (for multiple attachments)

```
public void composeEmail(String[] addresses, String subject, Uri attachment) {
    Intent intent = new Intent(Intent.ACTION_SEND);
    intent.setType("*/*");
    intent.putExtra(Intent.EXTRA_EMAIL, addresses);
    intent.putExtra(Intent.EXTRA_SUBJECT, subject);
    intent.putExtra(Intent.EXTRA_STREAM, attachment);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# Composer un mail (pas de pièce jointe)

```java
public void composeEmail(String[] addresses, String subject) {
    Intent intent = new Intent(Intent.ACTION_SENDTO);
    intent.setData(Uri.parse("mailto:")); // only email apps should handle this
    intent.putExtra(Intent.EXTRA_EMAIL, addresses);
    intent.putExtra(Intent.EXTRA_SUBJECT, subject);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# filtres associés

```
<activity ...>
    <intent-filter>
        <action android:name="android.intent.action.SEND" />
        <data android:type="*/*" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
    <intent-filter>
        <action android:name="android.intent.action.SENDTO" />
        <data android:scheme="mailto" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Vélo tracking

```
public void startBikeRide() {
    Intent intent = new Intent(FitnessIntents.ACTION_TRACK)
            .setType("vnd.google.fitness.activity/biking")
            .putExtra(FitnessIntents.EXTRA_STATUS, FitnessIntents.STATUS_ACTIVE);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

filtre associé :

```
<activity ...>
    <intent-filter>
        <action android:name="vnd.google.fitness.TRACK" />
        <data android:mimeType="vnd.google.fitness.activity/biking" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Course à pied

```
public void startRun() {
    Intent intent = new Intent(FitnessIntents.ACTION_TRACK)
            .setType("vnd.google.fitness.activity/running")
            .putExtra(FitnessIntents.EXTRA_STATUS, FitnessIntents.STATUS_ACTIVE);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

filtre associé :

```
<activity ...>
    <intent-filter>
        <action android:name="vnd.google.fitness.TRACK" />
        <data android:mimeType="vnd.google.fitness.activity/running" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Rythme cardiaque

```
public void showHeartRate() {
    Intent intent = new Intent(FitnessIntents.ACTION_VIEW)
            .setType("vnd.google.fitness.data_type/com.google.heart_rate.bpm");
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

filtre associé :

```
<activity ...>
    <intent-filter>
        <action android:name="vnd.google.fitness.VIEW" />
        <data android:mimeType="vnd.google.fitness.data_type/com.google.heart_rate.bpm" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Marche à pied

```
public void showStepCount() {
    Intent intent = new Intent(FitnessIntents.ACTION_VIEW)
            .setType("vnd.google.fitness.data_type/com.google.step_count.cumulative");
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# Actions localisées

```
public void callCar() {
    Intent intent = new Intent(ReserveIntents.ACTION_RESERVE_TAXI_RESERVATION);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

filtre associé :

```
<activity ...>
    <intent-filter>
        <action android:name="com.google.android.gms.actions.RESERVE_TAXI_RESERVATION" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Actions localisées

```
public void showMap(Uri geoLocation) {
    Intent intent = new Intent(Intent.ACTION_VIEW);
    intent.setData(geoLocation);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

## Schéma des URI de type geo :

- geo :latitude,longitude → geo :47.6,-122.3
- geo :latitude,longitude ?z=zoom → geo :47.6,-122.3 ?z=11
- geo :0,0 ?q=lat,lng(label) →
  geo :0,0 ?q=34.99,-106.61(Treasure)
- geo :0,0 ?q=my+street+address →
  geo :0,0 ?q=1600+Amphitheatre+Parkway%2C+CA

# filtre associé

```
<activity ...>
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <data android:scheme="geo" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Média

```
public void playMedia(Uri file) {
    Intent intent = new Intent(Intent.ACTION_VIEW);
    intent.setData(file);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

**Data URI Scheme**

`file:<URI>`
`content:<URI>`
`http:<URL>`

**MIME Type**

`"audio/*"`
`"application/ogg"`
`"application/x-ogg"`
`"application/itunes"`
Or any other that your app may require.

# filtre associé

```
<activity ...>
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <data android:type="audio/*" />
        <data android:type="application/ogg" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

# Initier un appel

## Nécessite

```
<uses-permission
android:name="android.permission.CALL_PHONE"/>
```

```
public void dialPhoneNumber(String phoneNumber) {
    Intent intent = new Intent(Intent.ACTION_DIAL);
    intent.setData(Uri.parse("tel:" + phoneNumber));
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# filtre pour activer la recherche dans une application

```
<activity android:name=".SearchActivity">
    <intent-filter>
        <action android:name="com.google.android.gms.actions.SEARCH_ACTION"/>
        <category android:name="android.intent.category.DEFAULT"/>
    </intent-filter>
</activity>
```

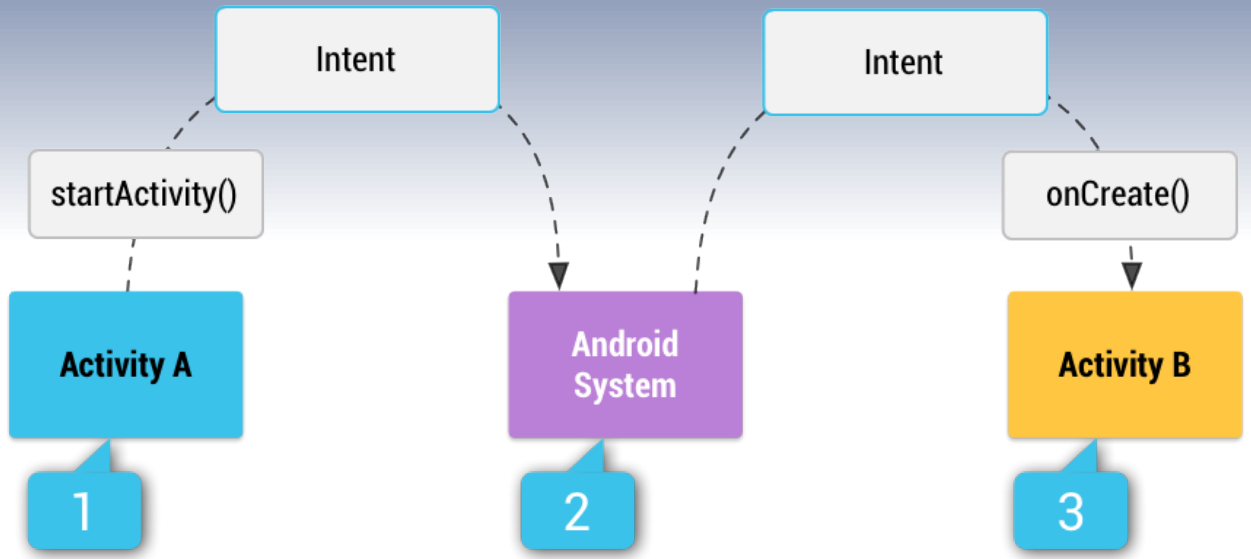
# Recherche web

```
public void searchWeb(String query) {
    Intent intent = new Intent(Intent.ACTION_SEARCH);
    intent.putExtra(SearchManager.QUERY, query);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# Autres

```
public void openWifiSettings() {
    Intent intent = new Intent(Intent.ACTION_WIFI_SETTINGS);
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

# Résumé global




Il existe déjà énormément d'intents implicites ! Utilisez la doc pour vous y retrouver : ▸ android.content.Intent

Ce cours reprend largement les tutoriaux en ligne proposés par Google :
▸ Android developers