# Révision Android

**Slide 12 : 5 composants forment le socle d'une app:**

- Activity
- Service
- Content Provider
- Broadcast Receiver
- Intent et intent-filter

Ils ne sont pas forcement tous présents

**Activity:**

- Fonctionnel
  - Un ecran affiché sur le mobile
  - Une app est composé d'une/+sieurs activity
- Technique
  - Une c lasse java qui etend la classe activity

**Definition type Manifest / activity :**

```
<application android:icon="@drawable/ic_launcher" android:label="@string/app_name">
    <activity android:name=".MainActivity" android:label="@string/app_name" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity android:name=".OtherActivity" />
</application>
```

**Le context**

Il représente les informations globales de l'app accessibles à ses composants (activity)

- getREssources / getServices
- Demarrer activity (startActivity) Service (stratService)
- Accès système de fichiers / BDD

**Cycle de Vie Activity**


- onCreate
- OnRestart
- OnStart
- OnResume
- OnPause
- OnStop
- onDestroy

**Pour ne pas perdre ses données si l'application est détruite :**

- OnSaveInstanceState()
- OnRestoreInstanceState()

Puis dans le onCreate --> onCreate(Bundle savedInstanceState)


**Passer d'un activité A à l'activité B**

1. ActivityA onPause()
2. ActivityB onCreate() puis onStart() puis onResume()
3. ActivityA onStop()

Si il y a transfert de données entre A et B : dans Activity A OnPause()

## Service:

**Un service :**

- s'execute en tache de fond
- Permet de faire des traitement long
- n'a pas d'ihm

**Fonctionnel : 2 Types de statuts :**

- **Started**
  Démaré depuis une activity mais est indépendant et ne s'arretera qu'a la fin de son traitement
- **Bound**
  Lié a des composants, si il n'y a plus de compos il est détruit.

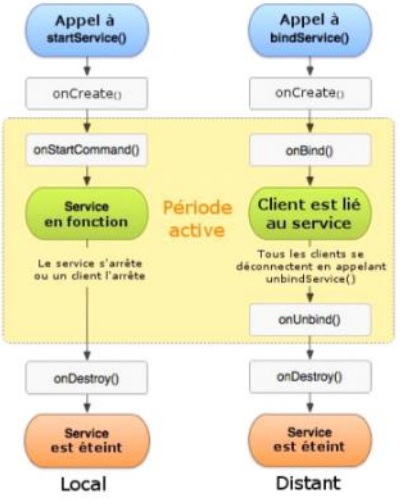
**Technique:**

- Etend la classe service
- Démarré en mode started --> startService()
- Démarré en mode bound --> bindservice()

**Manifest :**

```
...
<application android:icon="@drawable/ic_launcher" android:label="@string/app_name">
    ...
    <service android:name=".MyService" />
</application>
```

**Cycle de vie service :**




**Thread:**

Le code d'un service s'exécute dans le thread de l'application, il est donc nécessaire de lui alloué le sien pour qu'il puisse librement traiter sa tache.

**IntentService** -->service qui hérite de Service et qui simplifie le threading.
2 choses a définir :

- Définir un constructeur sans paramètre
- onHandleIntent (methode appelé a chaque lancement)

**Implémentation:**

Lancement

```
// Démarrage explicite d'un Service ou d'un IntentService
Intent intent = new Intent(this, MyService.class);
startService(intent);
```

```
// Connexion à un Service
Intent intent = new Intent(this, MyService.class);
bindService(intent, mServiceConnection, Context.BIND_AUTO_CREATE);
```

Arrêt

```
// Arrêt explicite d'un Service
Intent intent = new Intent(this, MyService.class);
stopService(intent);
```

```
// Déconnexion du Service
unbindService(mServiceConnection);
```

**Activity**

### Lancement

```
// Démarrage explicite d'une Activity
Intent intent = new Intent(this, OtherActivity.class);
startActivity(intent);
```

```
// Démarrage d'une Activity répondant à une Action
Intent intent = new Intent(Intent.ACTION_SEND);
intent.putExtra(Intent.EXTRA_EMAIL, "my_email@email.com");
startActivity(intent);
```

```
// Démarrage d'une Activity avec un résultat récupéré dans l'intent de la méthode onActivityResult
Intent intent = new Intent(Intent.ACTION_PICK, Contacts.CONTENT_URI);
startActivityForResult(intent, PICK_CONTACT_REQUEST);
```

### Arrêt

```
// Arrêt de l'Activity
finish();
```