P82 Mobile Application Development - Android

Services

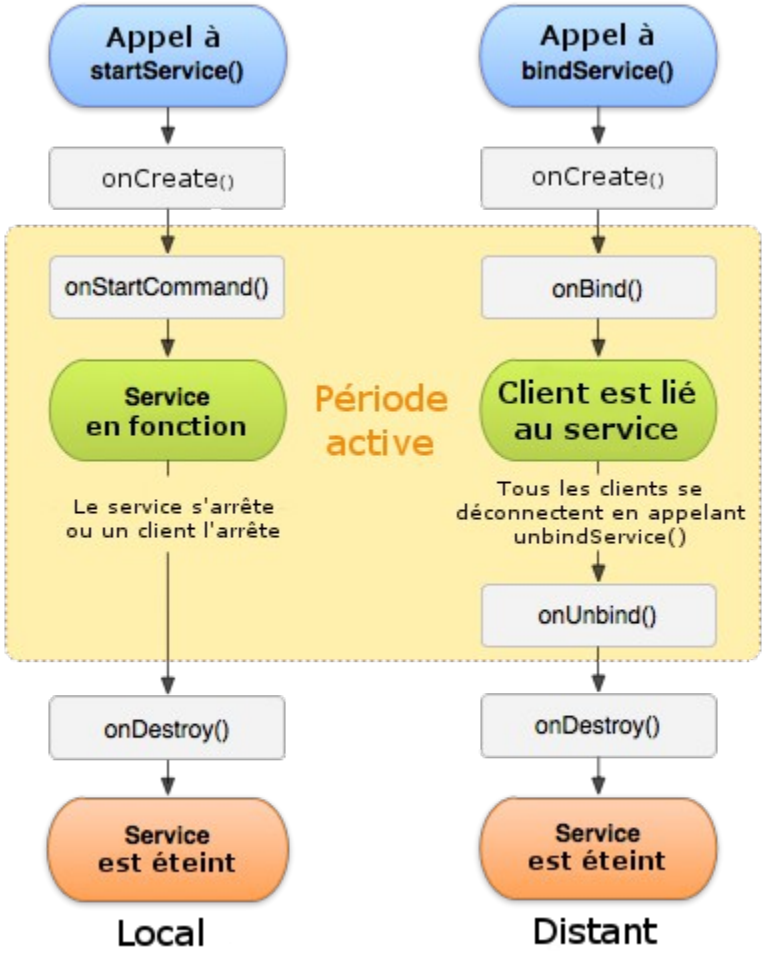
Services

- Activities without graphical interfaces and without interaction with the user.
- Can run long operations in the background
- An application can launch a service in the background and at the same time the user can launch a graphical activity
- Example: a network transaction (download), play music ....

Service Types

- Two froms exist :
  - the *local* services or started - activity that launches the service and the service itself belong to the same application
  - Remote service or bound service launched by other application component and still exists an interface that communicates with. This is called IPC or interprocess communication.

Service Life Cycle



- There are callBack methods as for activities.
- To create a service, we inherit the class Service or IntentService
- a service launches by default in the same process as that of the component that called it.

CallBack Methods

- onCreate () called when creating a service
- onStartCommand (intent intent, int flags, int startId) named after onCreate () and whenever service is called
  - Intent: intent who called the service
  - startId number of times the service has been called
  - Flag: nature of intent initiating service
  - There are three states
    - START_NOT_STICKY if system kills service, not restarted
    - START_STICKY if system kills service restarted, but intent null
    - START_REDELIVER_INTENT if system kills service, restarted, and identical intent
  - At the end of this method the service starts

CallBack Methods

- onDestroy () called when the service was destroyed
- To stop a service:
  - stopSelf()
  - Or stopService(Intent int)

Remote Service

- boolean bindService(Intent service, ServiceConnection conn, int flags)
  - conn will be an interface to monitor the remote service
- Connection to the remote service
  - bundService() to bind to the service
  - unbindService() to untie the service

Declaration in the Manifest

```
<service
    android:name=".MyService"
    ...       >
</service>
```

- android:name *   identifier
- android:permission   if the service needs special permission, if not configure takes the permissions of the app
- android:process : defines the process in which this service is supposed to run
- android:exported="true" allows the use of this service by other application

Manifest continued

```
<service
    android:name=".MyService"
    android:permission="fr.free">
    <intent-filter>
      <action
android:name="com.isi.course.myService"
/>
    </intent-filter>
</service>
```

Exercises

- Play a sound with: :

```
MediaPlayer player = MediaPlayer.create(ctx, R.raw.jack);
```

- Send messages to a service

Links

- http://stackoverflow.com/questions/4300291
- http://www.java2s.com/Code/Android/Media/