PROGRAMMATION DE COMPOSANTS MOBILES (ANDROID)

Wieslaw Zielonka

DownloadManager

DownloadManager

DownloadManager est un service d'Android qui permet de télécharger des fichiers depuis internet.

Comme AlarmManager le DownloadManager est un service autonome, disponible pour routes les applications sur votre appareil.

Ce service n'a rien à voir avec votre programme, c'est un service de système android disponible pour toutes les applications, en particulier pour la vôtre.

Pour utiliser DownloadManager mettre

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

dans AndroidManifest.xml

Obtenir la référence vers le DownloadManager

Dans l'activité :

```
val downloadManager =
        getSystemService(Context.DOWNLOAD_SERVICE) as DownloadManager
```

Dans une classe dérivée de AndroidViewModel :

```
val downloadManager =
application.getSystemService(Context.DOWNLOAD_SERVICE) as
DownloadManager
```

`getSystemService()` est une méthode de Context.

Envoyer une demande de téléchargement

```
val String : adr = "https://....."       /* String avec l'adresse */
val uri = Uri.parse(adr)                 /* l'objet Uri */
val request = DownloadManager.Request(uri)  /* créer l'objet Request */
/* soumettre Request à downloadManager */
val idDownload : Long = downloadManager.enqueue(request)
```

la valeur retournée par `enqueue()` est un identifiant unique (unique dans tout système Android) de téléchargement.

Il faut impérativement mémoriser `idDownload`, sinon il sera impossible de vérifier si le téléchargement a terminé et impossible de récupérer le fichier téléchargé.

Dans cet exemple DownloadManager télécharge le fichier dans son répertoire privé. Il est peut-être plus commode d'indiquer (à l'aide des attributs de Request) votre propre chemin de téléchargement.

Téléchargement par DownloadManager

- DownloadManager est un processus qui ne fait pas partie de votre application.

- Où DownloadManager télécharge les fichiers ?
  Deux possibilités :

  A. Si on ne spécifie pas la destination le fichier est téléchargé dans un répertoire privé de DownladManager. Il est toujours possible de lire ce fichier mais c'est le DownloadManager qui le gère (et sans doute décide si le fichier doit être effacé).

  B. Il est aussi possible de spécifier dans DownloadManager.Request la destination de fichier téléchargé. (Il semble que le fichier est quand même aussi stocké dans le répertoire privé de DownloadManager.)

- Comment votre application est avertie que le téléchargement a terminé ?
  Réponse: via un "message" (Intent) de réponse envoyé par le DownloadManager (mais qui reçoit cet Intent ?)

- A partir de moment d'envoie de demande de téléchargement il y a une notification qui s'affiche. Elle peut être utilisé pour annuler la demande de téléchargement (mais ce n'est pas automatique.)

Téléchargement par DownloadManager

Demande de téléchargement vers un fichier spécifique :

/* adr : adresse http ou autre */

```
val uri = Uri.parse(adr)
```

val nomFichier : String = ... /* nom de fichier local */

```
val request = DownloadManager.Request(uri)
    .setNotificationVisibility(DownloadManager
                                    .Request.VISIBILITY_VISIBLE)
    .setDestinationInExternalFilesDir(
        getApplication<Application>(), /* context */
        Environment.DIRECTORY_PICTURES,
        nomFichier
    )
val idLoad: Long = downloadManager.enqueue(request)
```

On demande que le fichier téléchargé soit placé dans la mémoire externe dans un répertoire sensé d'accueillir les images. Et le nouveau fichier portera le nom nomFichier.

Téléchargement par DownloadManager

- C'est un BroadcastReceiver que nous devons installer qui reçoit un Intent à la fin de téléchargement. Quand DownloadManager termine le téléchargement il envoie un message (Intent) en mode broadcast. Le message est capté **par tous les BroadcastReceivers** installés sur l'appareil.

- Votre BroadcastReceiver doit filtrer les messages (Intents) reçu. Il y a deux filtrages à faire :

  - un filtrage par le type de message (par exemple les messages annonçant la fin de téléchargement)

  - un filtrage par idDownload : votre application n'a pas intérêt de récupérer un fichier qui a été téléchargé à la demande d'une autre application (enfin si votre application est une application espion qui veut intercepter tous les téléchargements).