université de BORDEAUX

Développement Android

Tâches asynchrones et requêtes HTTP

TD3

Dans ce TD nous continuons à développer l'application de météo des stations de ski. En particulier, nous récupérons des données sur internet et mettons à jour nos composants graphiques avec ces données.

Lorsqu'une question est précédée d'un "▷ ", vous devez consigner votre réponse dans un fichier texte. Ce fichier sera à remettre à la fin de la séance.

1 Interrogation du service web

Pour récupérer les données concernant la météo d'une station de ski, nous allons envoyer une requête HTTP à une adresse dédiée. Pour tester cette adresse, ouvrez un navigateur web et rendez-vous sur :

```
http://snowlabri.appspot.com/snow?station=gourette
```

Vous pouvez constater que le serveur renvoie les données voulues. Ne vous occupez pas du format des données pour l'instant, nous verrons ça un peu plus tard.

Nous allons implémenter une requête HTTP à l'aide de l'objet `HttpURLConnection` pour récupérer ces données. Nous les afficherons en l'état dans LogCat dans un premier temps. Une difficulté vient du fait qu'Android interdit les opérations réseaux depuis le thread principal. Nous devons donc faire ces opérations depuis un autre thread.

AsyncTask

La classe `AsyncTask` (tâche asynchrone) est justement faite pour effectuer des tâches de courtes durée dans un autre thread.

→ Lisez le *Class Overview* de la classe `AsyncTask` dans la javadoc, puis parcourez le reste de la page à la recherche des informations suivantes :

▷ À quoi servent chacune de ces méthodes : `onPreExecute()`, `doInBackground()`, `onProgressUpdate()` et `onPostExecute()`.

▷ Dans quel(s) thread(s) s'exécutent-elles ?

▷ Quelle instruction dans le thread de départ déclenche l'invocation de `doInBackground` ?

→ Héritez de cette classe pour créer votre propre tâche asynchrone (très basique), qui affichera simplement "`Hello from another thread`" dans logcat.

▷ Que signifient les trois points `...` lorsqu'ils suivent un nom de classe dans la déclaration des paramètres d'une méthode ?

▷ A quoi servent chacun des trois noms de classes dans la signature $< \cdot, \cdot, \cdot >$ de votre tâche ?

HttpURLConnection

→ Implémentez la requête HTTP et exécutez là dans `doInBackground()`. Récupérez le résultat sous forme d'une unique chaîne de caractères, que vous afficherez dans LogCat.

Créez une méthode `updateWidgets(String result)` dans votre activité, que vous invoquerez une fois la réponse obtenue en lui passant la chaîne de caractère résultant de la requête. (Astuce : vous pouvez changer le constructeur de la tâche asynchrone...)

▷ Comment vous y êtes-vous pris pour invoquer cette méthode depuis la tâche asynchrone ?

2 Parser la réponse et mettre à jour l'interface graphique

La réponse fournie par le service web est au format JSON (Javascript Object Notation). La plupart des services web renvoient des données au format XML ou JSON. Cependant, JSON est souvent utilisé pour envoyer des données sur les mobiles parce qu'il est plus simple et moins verbeux que XML. Les données prennent donc moins de place ce qui permet d'économiser de la bande passante (qui peut être rare sur certains mobiles). Renseignez vous a propos du format JSON et comprenez son fonctionnement. Depuis votre méthode `updateWidgets`, parsez la réponse du serveur web à l'aide de la classe `JSONObject`, dont l'un des constructeurs permet d'utiliser une chaîne de caractère...

Assurez-vous que vous traitez le cas où la réponse contiendrait des erreurs ou des anomalies. Pour finir mettez à jour votre UI avec les données récupérées.

Certains d'entre vous ont certainement placé leur appel à `updateWidgets` directement depuis la méthode `doInBackground`. D'autres l'auront effectué dans `onPostExecute`. La première option causera une erreur.

▷ Tester cette option et notez l'exception que vous obtenez.

▷ Pourquoi faut-il utiliser `onPostExecute` à la place ?

À rendre par courriel :

Une archive zip nommée `td3-nom.zip` si vous êtes seul ou `td3-nom1-nom2.zip` si vous êtes deux, où `nom1` est le nom de famille alphabétiquement le plus petit de votre binôme, et `nom2` le suivant. L'archive doit contenir :

1. Votre fichier de réponse `.txt`
2. Votre fichier de layout `.xml`
3. Le fichier source de votre activité `.java`