PMR Compte-Rendu du TEA1

Présentation du protocole HTTP, entêtes HTTP, requêtes GET & POST, chaînes de requêtes, API REST & bonnes pratiques, API todo-api, utilitaire postman, requêtes PUT & DELETE, Consommation d'une API Rest depuis Android dans votre application TODO

Vladimir MARCHAND ; Olivier ANOUFA

Table des matières

I - Introduction :

Nous avions pour objectif d'implémenter des requêtes à une API dans l'application que nous avions construit la semaine dernière. Il fallait donc remplacer la sauvegarde des listes et des items dans les préférences par des requêtes permettant de récupérer les listes dans la base de données de l'API.

II - Analyse :

Nous allons analyser successivement les différentes activités et les expliquer.

A – Requests :

Nous avons ajouté un nouveau fichier kotlin Requests.kt pour y créer toutes les fonctions de requêtes que nous utiliserons par la suite. Les requêtes sont écrites en utilisant la librairie Volley. Les requêtes implémentées sont :

- makeLoginRequest(url: String): JsonObjectRequest
- addListRequest(list_name: String): JsonObjectRequest
- userListsRequest(): JsonObjectRequest
- getItemsListRequest(listId: Int): JsonObjectRequest
- addListItemRequest(item_name: String, idList: Int?): JsonObjectRequest
- checkItemRequest(idItem: String, idList: Int?): JsonObjectRequest
- unCheckItemRequest(idItem: String, idList: Int?): JsonObjectRequest

Elles ont respectivement pour fonction d'envoyer une requête d'authentification, de créer une liste pour l'utilisateur connecté, de récupérer les listes de l'utilisateur connecté, d'ajouter un item dans une liste de l'utilisateur connecté et finalement de cocher ou décocher un item d'une liste de l'utilisateur connecté.

Les requêtes avec la librairie Volley sont simple d'utilisation, il suffit de créer la bonne adresse URL et d'envoyer le bon token « hash » en header. Pour ce faire, ce token est récupéré dans les préférences lors de l'authentification, puis dans les autres requêtes la fonction getHeaders() est surchargée afin d'entrer le token « hash » dans le header. On gère ensuite la réponse de la requête en cas de succès ou d'échec.

B - MainActivity :

Nous avons effectué plusieurs modifications dans l'activité principale :

- Implémentation d'une fonction permettant de connecter automatiquement l'utilisateur avec les derniers identifiants utilisés, sauf si l'utilisateur arrive sur la page de connexion depuis le bouton de déconnexion présent dans l'activité SettingsActivity.
- Implémentation d'une fonction de vérification de la connexion internet au lancement de l'activité. Si l'utilisateur n'a pas accès à Internet, ce dernier n'a pas accès au bouton de connexion et ne peut donc pas se connecter.
- Ajout d'un champ de texte pour renseigner le mot de passe.
- Lorsque l'utilisateur clique sur le bouton pour se connecter, le programme lance la nouvelle fonction authenticateUser qui envoie une requête POST à l'url contenant le nom de compte et le mot de passe. Cette requête récupère le token d'authentification et l'enregistre dans les

préférences de l'application. Enfin, si l'authentification est réussie, l'utilisateur est envoyé dans l'activité ChoixListActivity.

C - ChoixListActivity :

Les modifications de cette activité sont les suivantes :

- A l'arrivée dans l'activité, on initialise le RecyclerView puis on envoie une requête à l'API pour récupérer les listes de l'utilisateur connecté. On récupère la réponse mise en forme grâce à un callback afin de récupérer les identifiants et noms de toutes les listes. On envoie ensuite la liste contenant toutes les listes dans le RecyclerView créé dans le TEA précédent.
- Quand l'utilisateur clique sur le bouton pour ajouter une liste en ayant renseigné un nom de liste, on envoie une requête à l'API pour créer une liste pour l'utilisateur connecté avec le nom correspondant. L'affichage des listes se faisant au lancement de l'application il faut remettre à jour le RecyclerView pour voir la liste ajoutée.
- Quand l'utilisateur clique sur une liste, il est envoyé dans l'activité ShowListActivity

D – ShowListActivity :

- Au lancement, on récupère l'id de la liste cliquée dans l'intent. On fait ensuite une requête à l'API afin de récupérer les items de la liste dans la base de données de l'API et on les affiche de la même manière que dans l'activité précédente.
- Lorsque l'utilisateur clique sur le bouton pour ajouter un nouvel item, on envoie une requête à l'API pour ajouter un item avec le nom demandé dans la base de données. On recharge ensuite la page pour afficher l'item.
- L'utilisateur peut cocher les items de la même manière que dans le TEA précédent. Cependant, bien que nous ayons réussi à implémenter la requête permettant de cocher ou non les items dans la base de données, nous n'avons pas réussi à afficher visuellement que l'item était coché dans la base de données.

E – SettingsActivity :

- Nous avons ajouté un bouton permettant de se déconnecter et de revenir à la mainActivity.
- L'URL de base est stockée dans les préférences et peut être modifiée. L'url de base par défaut reste celle de l'API.

III – Conclusion :

La principale difficulté que nous avons eu à surmonter est la formulation et le fonctionnement des requêtes. Après avoir essayé avec les trois librairies proposées, Volley nous a semblé être celle au fonctionnement le plus pratique. Par la suite, même avec des requêtes fonctionnelles, la récupération et le traitement de la réponse de l'API n'étaient pas aisées. Nous n'avons pas réussi à implémenter la possibilité pour l'utilisateur de cocher les items en ligne et avoir la réponse visuelle sur l'application, cependant ce TEA nous aura permis d'apprendre et de comprendre le fonctionnement des requêtes API.