LO52 - Rapport de projet

Mériadeg Guichard

7 janvier 2021

Sommaire

1 Introduction

Le but de ce projet était de développer une application pouvant servirà un entraineur désirant perfectionner les performances de ses équipes. Cette application devait fournir à l'entraneur les moyens d'évaluer à la fois les performances de chacune de ses équipes sur une course d'obstacles constituée de deux sprints, de deux sauts, et d'un pitstop, mais également les statistiques globales de l'ensemble de ses équipes, afin de déterminer par exemple les forces et faiblesses de chacun (qui est meilleur aux sprints, auxsauts...). Par soucis de confort, il nous était demandé de développer l'application sur un support de tablette. Cependant, étant doté de ressources limitées (à savoir un ordinateur dont la puissance n'était pas cécessaire pour émuler une tablette sous AndroidStudio) j'ai dû développer mon projet sous mon propre téléphone portable. J'ai donc du m'adapter aux contraintes imposées par ce dernier, et j'ai du utiliser le version 27 du SDK d'android (Oreao 8.1) afin de tester l'application au fur et à mesure de son développement.

Plutôt que d'appréhender le projet dans sa globalité, j'ai choisi de l'approcher « composant par composant » : d'abord, je me suis demandé comment pouvait fonctionner cette application, puis, quelle type d'interface pourrait être adapté aux besoins de l'utilisateurs, comment mettre en place la persistence des données...Ce sont ces différentes réflexions que je vais aborder dans ce rapport. Concernant le fonctionnement de l'application, les choses ont été tout de suite beaucoup plus claires suite au visionnage de la démonstration d'un projet de l'année dernière, j'ai donc plus ou moins directement commencé dès lors à m'intéresser à l'interface graphique.

2 L'interface graphique

2.1 Le menu déroulant

Au démarrage de mon application, une fois la définition de mes attentes posées, j'ai directement commencé par la réalisation d'un menu de navigation en partant d'un postulat simple : créer le menu, référençant les différentes activités du projet consistait d'une part en un début, un point de départ, mais également en une sorte de journal de bord. A chaque fois que je travaillais sur l'application, je savais où j'en étais, ce que je devais faire à l'instant et ce qui me restait à faire. Mon menu se présente donc sous la forme d'un menu latéral, comme le montre la figure suivante :

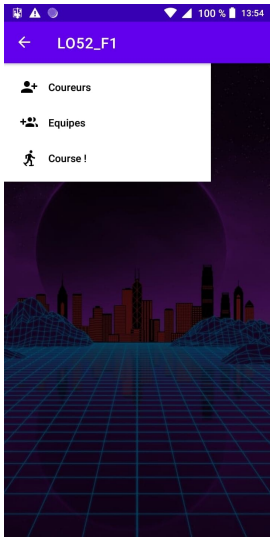

FIGURE 1 – Menu principal

Ce dernier permet d'accéder aux différentes activités de l'application. Pour réaliser les interactions avec le menu, je me suis simplement aidé de ce que j'avais vu au cours du TP1.

2.2 La gestion des coureurs

Au commencement de l'application, il y a les coureurs : il faut les générer pour créer les équipes par la suite, il faut également être en mesure de les modifier ou carrément de les supprimer. Ce sont des fonctions vitales à la bonne utilisation du projet, j'ai donc tenté de mettre en place une interface simple, ergonomique, qui permettrait à l'utilisateur de'accéder à l'ensemble des fonctions. Le résultat est sur la figure suivante :

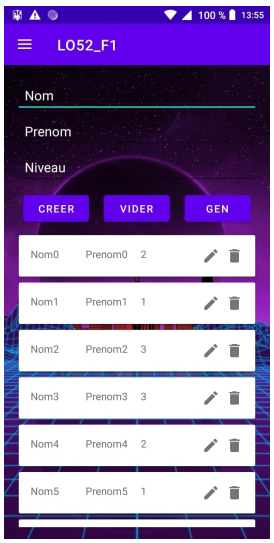

FIGURE 2 – Interface de gestion des coureurs

Ici, l'interface se décompose en deux parties distinctes : une première partie où l'utilisateur pourra créer et supprimer les profils de ses utilisateurs. Les champs de texte *Nom*, *Prenom* et *Niveau* sont les informations qui sont nécessaires à la création d'un coureur. Une fois ces champs remplis correctement, ilsuffit d'appuyer sur le bouton *AJOUTER* et le coureur se verra ajouté à la liste présente dans le seconde partie de l'interface. Avant de parler plus en détail de cette dernière, il me faut expliciter les boutons *VIDER* et *GEN (générer)*. Le premier sert à supprimer l'ensemble des coureurs en une seule fois tandis que l'autre permet de générer 30 coureurs de niveau aléatoire (pour des besoins de démonstration)

La seconde partie de l'interface se compose d'une liste déroulante affichant pour chaque ligne, le nom du coureur, son niveau, ainsi que deux icones. Le premier permet de modifier les informations du coureur sélectionné tandis que le deuxième permet de le supprimer manuellement. Sans rentrer trop loin dans les détails techniques, j'ai utilisé l'outil *listitem* d'android pour mettre en place cette partie de l'interface.

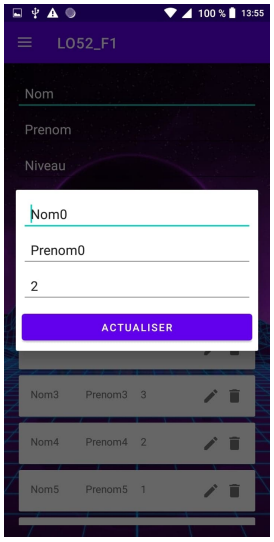

FIGURE 3 – Interface de gestion des coureurs

2.3 La gestion des équipes

L'interface de cette activité est un peu plus simple que celle des coureurs : elle dispose de deux boutons, *FORMER EQUIPES*et *VIDER*, ainsi que de la liste déroulante des équipes. Le bouton *FORMER* a pour but de générer des équipes de 3 coureurs de niveau égal (cf La création des équipes) tandis quele bouton *VIDER*, tout comme pour l'interface de gestion des coureurs, permet de supprimer l'ensemble des équipes.

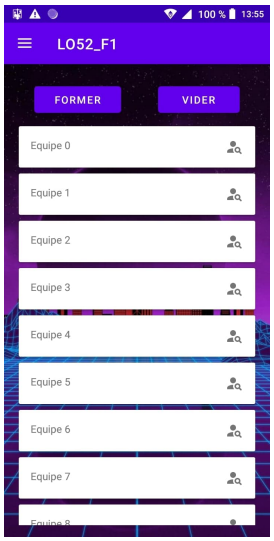

FIGURE 4 – Interface de gestion des équipes

Enfin, pour chaque équipe, il est possible de consulter la liste des membres, qui inclue le nom, le prénom, et le niveau de chaque coureur de l'équipe.

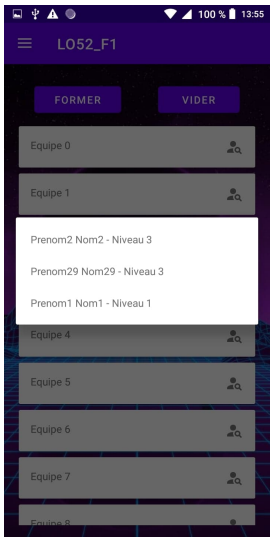

FIGURE 5 – Détails d'une équipe

2.4 La course

L'interface de la partie de l'application qui gère les courses est, dans la hiérarchisation de ses composants, à peu près similaire à celle de la gestion des équipes :

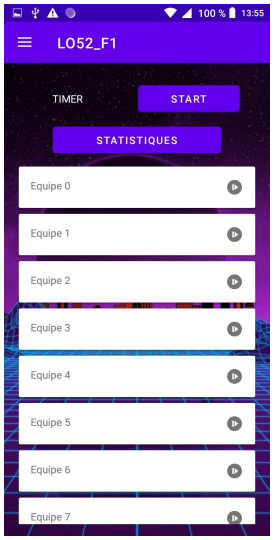

FIGURE 6 – Interface de gestion des courses

Un bouton pour lancer / stopper un simple timer, un zone de texte pour afficher le temps en cours, un bouton qui permet d'afficher des statistiques globales, et enfin, la liste des équipes. La partie autours du timer est relativement simple : le clic sur le bouton *START* se contente de laner un timer initialisé à zéro, tandis que le bouton *STOP* le réinitialise avant de le mettre en pause.

La partie suivante est centrée autours des différentes étapes d'une course. Une fois le timer lancé, chaque appui sur la touche *play* de l'équipe incrémentera la course d'une étape, tel que spécifié dans le sujet, chaque coureur verra ses temps enregistrés, et, une fois que chaque coureur aura fini, le bouton *play* se désactivera pour permettre d'afficher les statistiques individuelles de l'équipe.

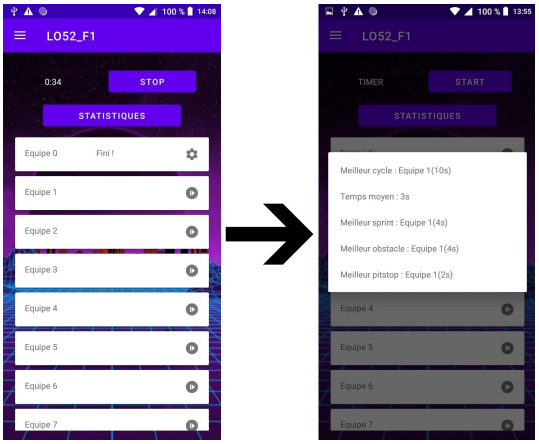

FIGURE 7 – Fin de course et affichage des statistiques de l'équipe

Les statistiques globales, elles, affichent les statistiques sur l'ensemble des équipes. En plus de cela, chaque temps est enregistré au sein d'une base de données qui fera l'objet de la partie suivante de ce rapport, de ce fait, lafermeture de l'application ne provoquera pas la perte des résultats.

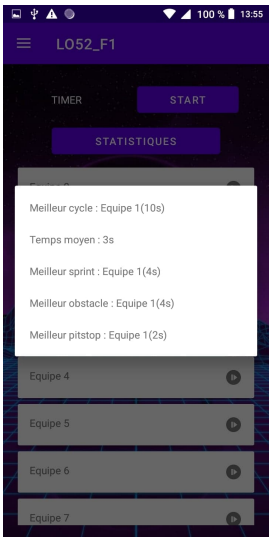

FIGURE 8 – Statistiques globables des courses

3 La persistence des données

Une fois les bases de l'application étant posées, et une fois l'interface graphique réfléchie, il fallait mettre en place une moyen de conserver les différentes données de l'application (coureurs, équipes, temps...). Après quelques recherches, mon choix s'est porté sur la librairie Room et ce pour deux raisons :

1. Premièrement, cette librairie est celle utilisée par Android, j'étais donc sûr de trouver une documentation fournie, et beaucoup de réponses à mes éventuelles questions sur divers forums
2. Secondement, car après une analyse un peu plus détaillée, il s'est avéré que Room était plutôt simple d'utilisation, et ce sans se montrer trop exigeant en terme de programmation

Une fois Room pris en main, j'ai pu directement passer à son implémentation dans mon projet, pour cela, j'ai tout d'abord crée les classes *Coureur*, *Equipe*, qui décrivaient de manière formelle les entités coureur et équipe. La figure suivante présente l'organisation de la base de données :

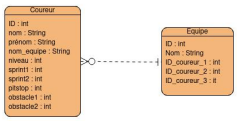

FIGURE 9 – UML de la base de données

Puis j'ai dû créer leurs classes associées, *CoureurDao* et *EquipeDao*, qui permettaient de faire le pont entre la base de données et les entités Java. Ces classes permettent d'implémenter des méthodes qui seront traduites par la suite en requêtes SQL.

4 Fonctionnement de l'application

4.1 Structure globale de l'application

Finalement, après avoir mis en place l'iterface graphique et la base de données de l'application, il me restait à implémenter l'ensemble des méthodes permettant l'utilisation telle que je l'avais planifiée. Cette première partie va expliquer un peu plus en détail le fonctionnement de l'application en elle même.
Tout part de l'activité principale *MainActivity*.

4.1.1 L'activité principale

Cette activité consiste en la première interaction de l'utilisateur avec l'application : elle présente le menu latéral à partir duquel nous pouvons naviguer dans les différentes activités. Chacune de ces activités (que ce soit la gestion des coureurs, des équipes ou des courses) fonctionne sur le même principe que les interfaces graphiques : en deux parties.

4.1.2 Le fonctionnement des activités secondaires

L'interface de chaque activité se décomposant en deux parties distinctes (utilisation de deux layouts différents : un pour la création, la génération aléatoire et le vidage global des entites, un autre pour la gestion individuelle des entités) il me fallait mettre en place deux classes par activité pour les gérer de manière complète. Ainsi, toute la partie « gestion globale des entités » était assurée par l'activité en elle même tandis que la gestion individuelle était gérée par une activité secondaire, qu'on appelle dans ce cas *Adapter*. Dans une relation ou chaque activité possédait son *Adapter*.

4.2 La création des équipes

Cette partie décrit les choix que j'ai pu faire concernant la création d'équipes respectant les contraintes de sujet. Pour rappel, il était demandé de former des équipes aussi homogènes que possible. Afin d'arriver à un résultat cohérent, j'ai pris une décision basé sur mon expérience personelle : ayant déjà fait de la compétition au sein de mon club d'échecs, je me suis souvenu que nos équipes étaient elles aussi réparties par niveau selon le découpage suivant :

1. Chaque équipe se devait de regrouper 4 membres, tous ayant à peu près le même niveau
2. Il y avait les quatre niveaux suivants : débutants, expérimentés, confirmés, excellents.

Le but ici est non pas de regrouper les équipes par niveau des coureurs, mais de mélanger les niveaux pour avoir un niveau moyen par équipe à peu près similaire. J'ai donc choisi de personnellement travailler avec trois niveaux différents (débutants, intermédiaires, expérimentés), ceci également à des fins de simplifications : j'avais peur de manquer de temps et je préférais rendre un travail le plus complet possible. Concernant l'algorithme de répartition, je suis parti du principe que pour chaque équipe possédant un coureur expérimenté, il fallait au moins un coureur débutant. En outre, lors de la création des équipes, chaque coureur est stocké dans une liste en fonction de son niveau. On se retrouve donc avec 3 listes à l'initialisation de la fonction. L'algorithme suivant découle de la réflexion que j'ai pu mener, c'est celui que j'ai décidé d'implémenter au sein de mon application, et après les phases de tests (générations aléatoires de coureurs), il s'est avéré que chaque équipe ne présentait pas de différence de niveaux supérieure à 2.