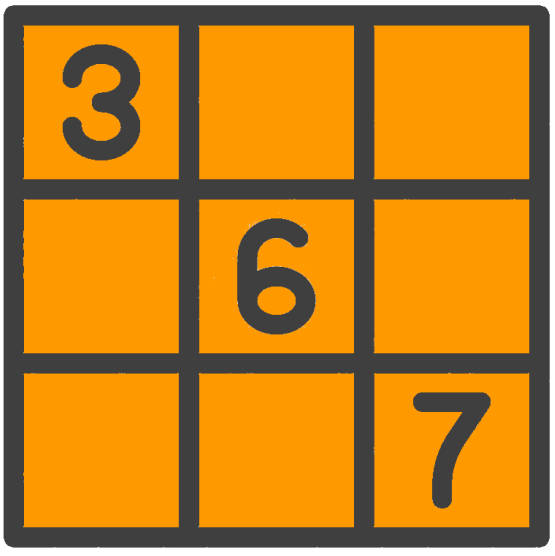
Leblanc Cyril
Bonnard Claire
LP SMIN

Compte-rendu application Android

Sudoku

Sommaire



I. Principe de l'application et fonctionnement

Dans le cadre du cours de programmation Android nous devions développer une application mobile. L'idée de cette application nous est venue dans le but de découvrir les possibilités et fonctionnalités de l'appareil. Ainsi, nous avons choisi de développer un sudoku pour le côté ludique et l'algorithmique qu'il y a derrière.

Nous voulions donc créer un sudoku fonctionnel avec trois niveaux de difficulté tout en répondant aux contraintes du barème donné par le professeur.

Notre projet contient donc différents fichiers, dont l'utilité et les fonctionnalités principales sont détaillées ci-dessous, cela vient en complément de notre code pour plus de compréhension.

MainActivity

Le fichier MainActivity contient l'activité de la page d'accueil, il gère la sélection de niveau. Le fragment PlayerInfoFragment s'affiche également dans cette activité.

PageJeu

Ce fichier contient l'activité de la page de jeu. Il gère l'affichage de la grille et des boutons de sélection des numéros grâce aux fichiers adapter qui lui renvoient directement les bons boutons à afficher. Le fragment PlayerInfoFragment est également présent sur cette activité.

Singleton

Le singleton regroupe les informations à transférer d'un fichier à l'autre, le niveau de difficulté choisi, la modélisation de la grille en début de partie par rapport à la difficulté, le TimeStamp du début de partie et les informations du joueur transférées au fragment.

AdapterNumbers

AdapterNumbers est un recyclerView, il contient les boutons numéro à sélectionner par le joueur pour savoir quel nombre s'inscrira sur la grille quand il appuiera sur une case. La forme recyclerView, nous permet de ne pas dupliquer dix fois quasiment la même fonctionnalité. Le bouton "X" permet d'effacer le numéro de la case choisie.

SelectedNumber

Il s'agit d'une interface qui nous permet de récupérer le nombre choisi parmi les numéros et de transférer cette information à la grille pour afficher le numéro choisi dessus.

AdapterGrille

Permet de générer la grille, c'est également un recyclerView, dans la même stratégie de ne pas dupliquer du code. Ce code réplique une section de jeu neuf fois pour obtenir une grille complète, au lieu de créer les 81 boutons nécessaires individuellement.

Case

La classe Case gère les cases de la grille, c'est la plus petite entité de celle-ci. Elle contient une valeur, un String de 1 à 9 et un booléen 'modifiable' qui définit si la valeur est modifiable ou non par le joueur.

Section

La classe Section gère les sections de la grille, c'est un carré de 9 Cases où chaque numéro doit être présent une seule fois. Elle contient une méthode finishSection qui vérifie si cette section est entièrement remplie et juste. La méthode isRemplie, présente également dans la classe Grille, nous informe si la section (ou la grille) est remplie, sans condition sur le fait qu'elle soit correcte. Elle nous est utile pour afficher un message au joueur que malgré que sa grille soit remplie ce n'est pas la bonne réponse.

Grille

La classe Grille gère l'entièreté de la grille de jeu, elle contient 9 Sections. Ses méthodes withoutDuplicate vérifient que chaque ligne ou colonne ne possède pas de numéros doublons. Ainsi, grâce à la méthode isTermine qui vérifie si chaque section est finie et qui appelle nos deux withoutDuplicate on peut donc savoir si la grille est complète et juste. Pas besoin de vérifier que les lignes et les colonnes sont entièrement remplies car la méthode finishSection le fait déjà.

PlayerInfoFragment

Il s'agit, comme son nom l'indique, d'un fragment s'affichant sur la page d'accueil et sur la page de jeu donnant des informations sur le joueur qu'il peut modifier en appuyant dessus, cela le redirigera vers la page SettingsMenu.

SettingsMenu

Il s'agit de la troisième activité de notre application. Depuis cette page, le joueur peut modifier son avatar en choisissant une image de sa galerie et son pseudo. Ces données seront ensuite affichées sur le fragment PlayerInfoFragment. En appuyant sur 'sauvegarder' la classe User est modifiée et l'application retourne à la page précédente.

User

La classe User a pour fonction de stocker les informations du joueur ; l'avatar et le pseudo.

II. Utilisation des fonctionnalités de l'appareil

Cette application utilise une fonctionnalité de l'appareil, la galerie photo. En effet, nous l'utilisons pour permettre au joueur de modifier son avatar.

III. Les difficultés rencontrées

Nous n'avons pas réussi à finir le niveau difficile de notre jeu, parce qu'il est vraiment difficile. Sinon, nous avons rencontré des difficultés à la gestion des adaptateurs et à l'affichage de la grille, sur toute la partie algorithmique avec la gestion des duplications et pour faire la transition entre les coordonnées x/y et les coordonnées section/case.

Une dernière difficulté rencontrée a été sur la récupération de l'image de la galerie. En effet, nous voulions y accéder directement à l'image en temps que fichier pour la stocker dans le répertoire de l'application, mais cela s'est révélé trop compliqué. Nous avons donc opté pour une récupération de l'image en Bitmap.

IV. Ce que nous avons appris

Au cours de ce projet nous avons eu l'occasion d'apprendre de nouvelles compétences, comme la gestion des recyclerView pour répliquer un même objet dans une page, ici nous en avons eu besoin pour répéter des boutons, styliser des boutons et leur donner l'apparence que nous voulons et non celle par défaut.

Grâce à ce projet nous avons eu la possibilité de manipuler des singletons, pour stocker différentes données dans la RAM et ainsi y accéder depuis n'importe quel fichier de notre projet.

Nous avons également appris l'utilisation et les possibilités offertes par les fragments pour réutiliser des widgets à plusieurs endroits sans avoir besoin de les dupliquer et enfin la gestion des interfaces pour transmettre des données à travers différentes classes.

V. Conclusion

L’application est fonctionnelle même si elle est perfectible. En effet, on pourrait ajouter certaines fonctionnalités comme les possibilités d’enregistrer l’avatar et le pseudo en rouvrant l’application, de pouvoir mettre pause en jeu, d’enregistrer les meilleurs scores, de proposer plus de niveaux…

Nous aurions aimé rajouter un appel à une API, cependant nous n’en avons pas trouvé de gratuite.

Nous avons beaucoup appris et nous sommes maintenant capables de développer des applications sur Android.