Télécom SudParis
Année scolaire 2017/2018

Projet Developpement Informatique
PRO 3600

Mini-Games Clash

Membres : **Lavergne Eric**
Katsoulakis Christos
Wieleba Krystian
Jaouhari Youssef
Cabaniols Damien

Enseignant responsable : **Pierre Sutra**

Version du 12 février 2018.

Sommaire

1 Cahier des charges

Concept global

Le projet consiste au développement d'un jeu mobile sous Android opposant deux joueurs en ligne sur plusieurs mini-jeux rapides.

Le joueur s'identifie. Il trouve ensuite un adversaire en ligne grâce à l'un des deux modes proposés : défier un ami / défier un joueur au hasard. Ainsi le joueur pourra construire une liste d'amis en ajoutant l'identifiant d'un joueur.

Ils s'affrontent ensuite dans une série de mini-jeux différents dans un ordre aléatoire. La partie entre deux joueurs se termine lorsque l'un des deux joueurs atteint un certain score. Ces mini-jeux auront des graphismes et un gameplay simples et efficaces ; cela participera à une ambiance de jeu fun et décontractée et nous permettra de nous concentrer sur l'aspect multijoueur.

Mode Training

Un mode "Training" pourrait également être disponible ; le joueur y jouera seul au jeu de son choix (avec une optique de record ou contre l'ordinateur en fonction de la logique du jeu). Cela nous permettrait de commencer à implanter les graphismes et les fonctions de base de chaque jeu dans un premier temps et d'incorporer les difficultés du mode "Versus" dans un second temps.

Mini-jeux proposés

Les mini-jeux seront donc proposés en 2D parmis les suivants :

- Presser un bouton lorsqu'un compteur atteint zéro ou qu'un indicateur apparaît à l'écran
- Eclater sur l'écran les bulles de la couleur de son adversaire (qui se déplacent à la manière de bulles de savon donc) et pas les siennes
- Un morpion
- Une course d'escargots où il faut balayer l'écran pour faire avancer son escargot
- Un hockey
- Attraper un maximum de fruits qui tombent des cieux avec un personnage en bas de l'écran.
- Un mini-golf 2D en une manche.

Des extensions pourront être réalisées avec des mini-jeux supplémentaires, plus complexes, comme un combat de plateforme ou une course 2D.

Spécificités techniques

Cette application sera développée grâce au langage Java et au logiciel Android Studio. L'une des principales difficultés de sa programmation résidera dans la gestion du multijoueur en ligne, avec une transmission des données qui doit se faire quasiment en temps réel pour certains mini-jeux.

Aperçu de l'esprit du jeu

L'image ci-dessus (non contractuelle) tente de résumer l'esprit du jeu : d'un gameplay simple et amusant, les mini-jeux doivent s'enchaîner rapidement - un peu à la manière dont se déroulent les parties dans certaines applications de quiz à deux joueurs.

2 Développement

2.1 Analyse du problème et spécification fonctionnelle

Analyse du problème

Nous nous sommes posés plusieurs questions de manière à analyser et fixer plus précisément les objectifs et le cadre du sujet.

- Est-ce que les jeux sont proposés dans un ordre aléatoire ou sont choisis tour à tour par un des joueurs parmis une liste de trois ou quatre jeux (la liste des jeux changeant à chaque tour) ?

La seconde option peut apporter une ‘’tension’’ dans le sens où un joueur pourra sélectionner délibérément un jeu qui mettra en délicatesse son adversaire. Toutefois, ce mode de sélection ne présente un intérêt que si le nombre de mini-jeux proposé est bien supérieur au nombre de manches disputées, ce qui ne sera sans doute pas le cas. De plus la première option est moins compliquée à coder et peut apporter tout autant en tension dans le jeu. C’est pourquoi nous avons choisi une sélection des jeux au hasard.

- Combien de mini-jeux gagnants ?

Une série à un unique mini-jeu gagnant ne constitue pas à proprement parler une série et n’offre pas la possibilité de revanche immédiate. Par ailleurs, trois jeux gagnants peuvent entraîner la nécessité d’effectuer cinq manches (score 3-2) soit cinq jeux au minimum. Nous choisirons donc l’option intermédiaire de deux mini-jeux gagnants.

- Est-ce que le joueur possède un identifiant stable et la possibilité de construire sa liste d'amis ou bien il peut changer d'identifiant à chaque connexion ?

Créer une liste d’amis pour l’utilisateur implique de nombreux détails techniques : il faudrait stocker l’ensemble des joueurs dans une base de donnée - dont on ne connait pas le possible emplacement. Peut-être toutefois qu’il existe un service (comme Google Play) qui permette d’utiliser de telles fonctionnalités en facilitant les choses. L’autre option est sans doute moins complexe, mais elle a des désavantages certains : l’utilisateur doit contacter ses amis et leur demander leurs identifiant pour les défier. Après réflexion, elle nécessite aussi la présence d’une base de donnée, par exemple ne serait-ce que pour éviter d’enregistrer deux utilisateurs avec le même identifiant et pour les enregistrer temporairement. Pour ce qui est du mode adversaire aléatoire, le choix entre les deux options a peu de conséquences sur la réalisation.

- Est-ce que les joueurs pourront mettre une partie en pause / la quitter ?

La possibilité de faire pause dans une partie à deux joueurs, surtout où la plupart des mini-jeux sont à temps réel, est un cadeau empoisonné pour celui qui subit la pause. A prohiber donc d'autant plus que les parties sont conçues pour être courtes et jouées d'une traite. De la même manière, il vaut mieux ne pas laisser au joueur la possibilité de quitter la partie autrement qu'en écrasant l'application (de manière à ne pas faciliter la vie des mauvais joueurs). Ainsi, si un joueur est déconnecté la partie devrait se terminer automatiquement en donnant la victoire au joueur restant.

- Est-ce que les scores des parties précédentes sont stockés et affichés au joueur ?

Une fenêtre de statistiques avec le nombre de parties jouées par le joueur et son ratio victoire/défaite est envisageable. Cependant ceci n'est possible que si chaque joueur a un identifiant stable, car dans le cas contraire il semble difficile de stocker les données des joueurs.

- Y aura-t-il de la musique et des bruitages ?

Un jeu muet, c'est un peu triste. D'autant plus que Java fournit des bibliothèques qui devraient nous permettre d'incorporer des sons sans trop de difficultés. Ainsi, on pourrait implémenter une musique de fond différente selon le jeu, et en plus des bruitages lors des actions effectuées, que ce soit lors d'un jeu ou en parcourant le menu.

- Un onglet Réglages dans le menu est-il nécessaire ?

Une interface de réglages pourrait comporter la possibilité de couper le son de l'application, de changer la luminosité ? (pour que l'interface ne soit pas vide sinon pas nécessaire); autoriser les notifications comme une demande en amis ? (beaucoup d'applis ont cette option); pour les âmes sensible qui ne veulent pas se faire insulter à la fin d'une partie, masquer le chat ?
Un onglet "Compte" avec l'identifiant du joueur et l'option de le changer.
Un onglet "App" avec les informations de notre jeux telle que le cadre dans lequel nous l'avons développer, la version actuelle et le nom de ses créateurs. Nous pourrions penser cependant à faire cet onglet à part, hors des réglages.
Malgré tout, un onglet "Réglages" est vraiment optionnel et donc dans le cadre de ce projet, à priorité faible.

- A quel moment est-ce que le joueur s'identifie ?

Le joueur peut s'identifier au lancement de l'application ; une première interface d'identification est d'abord affichée dans le cas où l'application ne retient pas l'identifiant du joueur. Une alternative pourrait être de demander la connexion seulement après que l'utilisateur ait sélectionné le choix du mode multijoueurs, puisqu'elle n'est pas nécessaire pour le mode Training.

- Un chat entre les joueurs est-il nécessaire ?

Le chat possède plusieurs intérêts comme la possibilité d'envoyer des piques à son adversaire et ainsi d'augmenter la rivalité, ou au contraire de féliciter son opposant. De cette manière la tension et l'enjeu d'une manche en sortent renforcés. De plus cela pourrait permettre de programmer une revanche. Toutefois, dans le jeu que nous concevons, où les parties sont rapides et jouées d'une traite, on peut se demander ou le situer : seulement un échange à la fin qui se coupe ensuite ? Ou entre les parties avec n'importe qui ? Avec la possibilité de rejouer contre son adversaire en fin de partie, un chat rajouterai une interaction de plus qui pourrait mener à une revanche. Alors que un chat "permanent" permettrait de discuter avec ses amis. Cependant, cela représente trop de travail supplémentaire et n'est pas d'une utilité fondamentale ; nous nous concentrerons donc sur les autres fonctionnalités.

- Comment mettre deux joueurs en contact, avec le mode aléatoire et avec le choix par identifiant ?

En considérant que le jeu ne se déroule pas au tour par tour mais en temps réel, la problématique de réunir deux joueurs au même moment se pose. Dans le cas du choix par identifiant, l'application peut afficher une fenêtre ou un onglet intermédiaire que le joueur défié peut accepter ou rejeter. Il faudra aussi veiller à transmettre l'échec de la requête au joueur défiant dans le cas d'un rejet ou d'une connexion handicapante. Il faudrait également autoriser le joueur à annuler la recherche d'adversaire afin de ne pas être coincé si aucun adversaire n'est trouvé. Dans le cas du mode aléatoire, il faut que deux joueurs se connectent pendant un faible intervalle de temps qui coïncide. L'application se charge de mettre en relation les deux joueurs, et la partie démarre automatiquement sur leur écran après le temps de recherche, avec le nom de l'adversaire affiché sur la première interface intermédiaire. Une fois encore, il existe sans doute des bibliothèques ou des modules pour faciliter ces opérations.

- Un mode Training est-il nécessaire ?

La principale visée du jeu est le mode multijoueur mais le mode Training conserve un intérêt car il permettra aux joueurs de mieux appréhender certains jeux. Surtout, cela nous permettra dans un premier temps d'affronter uniquement la conception du jeu puis dans un second temps de faire face aux difficultés du mode multijoueur.

Spécification fonctionnelle

Les fonctionnalités choisies sont donc les suivantes :

- *S'identifier*
- *Jouer seul pour s'entraîner sur les jeux proposés (Mode Training)*

- *Défier un adversaire aléatoirement*
- *Défier un adversaire à l'aide de son identifiant*
- *Jouer contre l'adversaire défié*
- *Disposer de musique et de bruitage pendant la partie*
- *Accéder à l'aide*

Tests de validation :

Beaucoup de cas d'erreurs seront bien évidemment spécifiques aux jeux.
Cependant, on peut déjà penser les tests suivants comme pertinents :

- Si un joueur lance l'application
  Résultat : une interface d'identification est ouverte ou l'utilisateur est automatiquement connecté
- Si un joueur entre un nom d'utilisateur déjà pris
  Résultat : message d'erreur "Nom d'utilisateur déjà prit, veuillez entrer un nom différent"
- Si un utilisateur entre un nom vide comme identifiant
  Résultat : message d'erreur "Veuillez entrer un nom d'utilisateur non vide".
- Si la recherche d'adversaire n'aboutit à rien au bout d'un certain temps
  Résultat : annulation de la recherche et message d'erreur "Aucun adversaire trouvé".
- SI le joueur adverse se déconnecte au cours de la partie
  Résultat : affichage du message d'erreur "Adversaire déconnecté" suivi d'un retour au menu
- Si une partie dure trop longtemps au cours d'un jeu (blocage du jeu)
  Résultat : égalité entre les joueurs et passage à l'interface intermédiaire à la fin d'un compteur

2.2 Conception préliminaire

Découpage en modules

Main : ce premier module correspond à la sempiternelle fonction principale avec initialisation, lancement de l'application et affichage du menu.

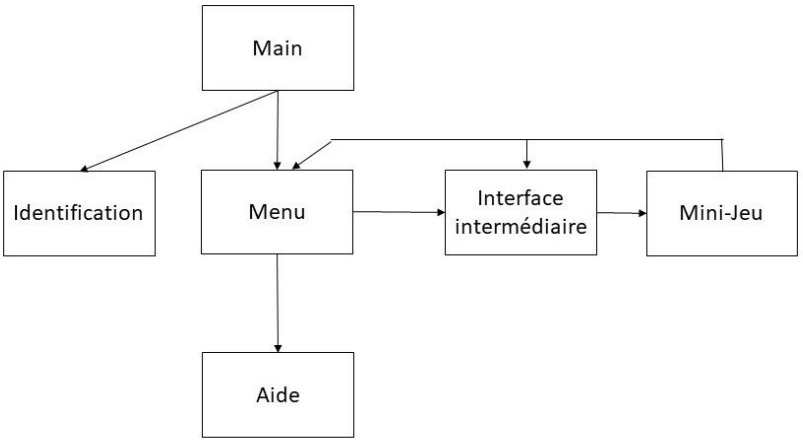
Menu : Cette première interface guide l'utilisateur vers les options qui s'offrent à lui : Jeu à deux joueurs en ligne, Jeu en mode Training, Liste d'amis, Aide, …

Aide : Il s'agit d'une interface qui expose le fonctionnement du jeu dans les grandes lignes.

Interface intermédiaire : Ce module affiche une interface avant / après chaque mini-jeux qui présente la situation (les scores respectifs des deux joueurs, l'identifiant du gagnant si la partie est terminée) et présente le mini-jeu qui va suivre (avec un message simple tel que "Eclatez toutes les boules rouges - et pas les bleues").
Avant un mini-jeu, cette interface attendra un clic du joueur pour disparaître et laisser la main au mini-jeu. Dans le cas de l'affichage de fin de partie, deux options seront affichées en bas de l'écran ; une option permettant aux joueurs de rejouer devrait être disponible - puisque les parties sont rapides, les joueurs voudront avoir leur revanche sur leurs amis sans repasser par le menu, l'autre permettra de revenir au menu.

Mini-jeu : Chaque mini-jeu représente un module à part entière. Il attend que les deux partis soient prêts (voir précédemment) pour démarrer. Il prend fin à la victoire d'un joueur et cède alors la place à l'interface intermédiaire.

Identification : Ce module affiché au lancement du jeu a pour but de permettre au joueur de créer son identifiant. Il fera également lien avec une base de donnée des joueurs inscrits.



Structures de données, stockage des information

Différentes informations devront être enregistrées de manière à ce que l'utilisateur puisse y accéder fréquemment.
Certaines pourront être stockées dans l'appareil mobile du joueur : ce serait par exemple le cas d'un possible enregistrement des scores des parties jouées, que l'utilisateur consulterait dans un onglet du menu.
C'est aussi le cas pour la liste des identifiants des amis du joueur, sous la forme d'une liste de caractères.
Une base de données sera utilisée pour lister les identifiants de tous les joueurs inscrits dans l'application. Elle regroupe donc leurs identifiants et leur adresse IP.

Tests d'intégration

Il faut faire un choix parmis plusieurs méthodes de test d'intégrations : top-down, up-down, sandwich, big-bang.
Les deux méthodes les plus employées sont les méthodes top-down et up-down.
Elles consistent toutes les deux à effectuer les tests de manière hiérarchique cependant dans un ordre différent. En effet, la méthode top-down consiste à tester les modules à partir du haut, et donc à partir du main, puis en redescendant. Alors que la méthode up-down, au contraire, consiste à partir des modules inférieurs et donc les différents mini-jeux pour ainsi remonter en conséquence.
Dans notre cas cela paraît plus cohérent de tout d'abord tester les différents jeux, et ensuite remonter vers le main. Et donc d'adopter un test d'intégration du type up-down.

3 Bibliographie

Beginning Android Games, Mario Zechner.

Algorithmique et langage de programmation, Pierre Sutra et Gaël Thomas, Cours CSC 3101 à Télécom SudParis.

Créez des applications pour Android, Frédéric Espiau, https://openclassrooms.com/courses/creez-des-applications-pour-android.

Génie logiciel et gestion de projets informatiques, J. Paul Gibson, Cours PRO 3600 à Télécom SudParis.