IUT INFORMATIQUE LYON 1 – BOURG EN BRESSE

Picarete

Ptut 2014 - 2015

Ronot Olivia | Jacquemin Simon

05/02/2015

Dossier de réalisation de Picarete. Un jeu mathématique développé avec l'idée et le soutiens de Mr Eric Duchène, de l'association Math à Modeler.

CONTENU

INFORMATIONS UTILES

GROUPE

Olivia Ronot et **Simon Jacquemin**

JEU REFERENCE

Pic Arête de *Math à Modeler*

CLIENT

Mr. **Eric Duchène**, membre de l'association mathématique **Math à Modeler**. Association qui promouvoir de façon ludique l'apprentissage des mathématiques dans les jeux.

Association Math à Modeler

CONTRAINTES

REGLES DE BASE

Le **Pic Arête** est un jeu à deux joueurs, dont le principe est somme toute très simple: Le but du jeu est de capturer les carrés blancs, pour cela, à chaque tour de jeu le joueur choisit une arête grise, qu'il transforme ainsi en arête de couleur, si par cette action un carré a ses quatre cotés colorés, alors ce dernier est capturé. On marque un point à chaque fois que l'on capture un carré.

REFERENCES

http://mathsamodeler.ujf-grenoble.fr/LAVALISE/PicArete/index.html

PLATEFORME

Uniquement Android.

Compatibilité : De 4.0 à 5.0

FRAMEWORK

Aucun Framework n'as été choisi pour ce projet. Pour la programmation, nous sommes partis sur **Android Studio** pour ces diverses fonctionnalités avancées dans le développement Android.

ANDROID STUDIO

Version originale de développement : 1.0.2

PROBLEMES ET SOLUTIONS

Nous n'avons pas eu de gros problème sur ce projet. Malgré tout, si nous devions identifier quelques points problématiques, nous pouvons parler des difficultés matérielles, et de la partie de liaisons des modules.

PREPARATION DU MATERIEL

Nous nous sommes rendu compte, heureusement assez tôt, que nous n'avions pas forcément les mêmes versions des logiciels et que rien n'était configuré (Ni Android Studio, ni le répertoire Git). Nous donc dut faire des mises à jours afin d'arriver à travailler sur des outils stables.

Toutes ces manipulations ont étés faites durant la phase de pré-production. Cela nous a permis d'éviter de devoir attendre avant de commencer la réalisation. Mais cela aurait pu nous faire perdre un temps précieux (Au moins 1,5 jours de développement sur machine)

LIAISONS DES DIFFERENTS MODULES

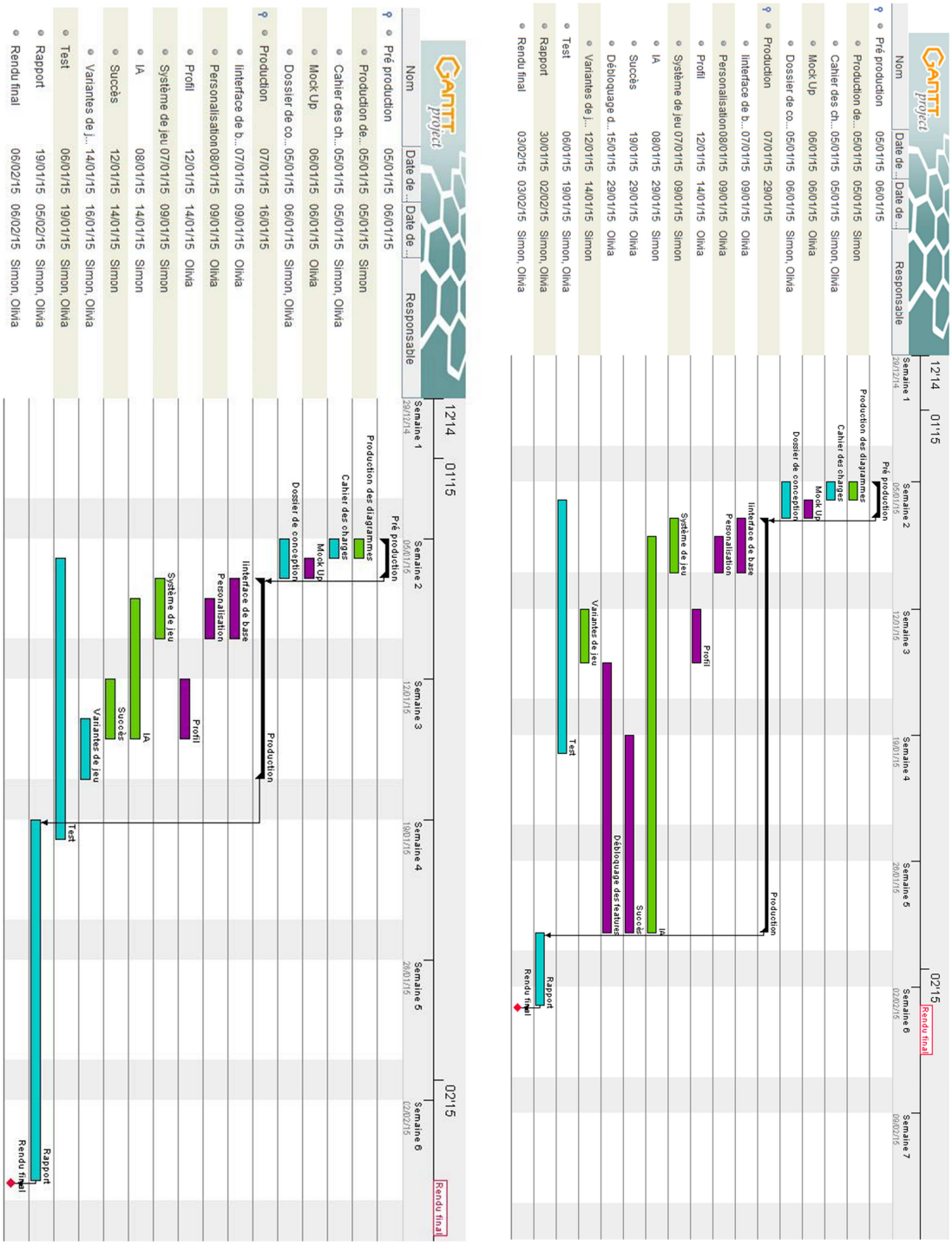
Lors de la liaison de chaque module fait par les différents membres de l'équipe, nous nous sommes confronté au problème des formats de données qui n'étaient pas forcément les mêmes, ou des fonctions de liaisons qui ne tombaient pas en face.

Nous avons dut prendre du temps pour nous remettre d'accord. Le temps nous était précieux pour arriver au bout de tous les objectifs et cela nous a légèrement déstabilisés dans notre organisation.

Nous nous sommes donc repris très vite et des solutions ont été trouvées parce que nous avons travaillé en collaboration, avec une bonne écoute et compréhension des problèmes de chacun. Cela aurait pu être évité avec une meilleure gestion des points dit « sensibles » lors de la phase de pré-production.

PLANNING DE CONCEPTION

Voici à **gauche le planning prévisionnel** et à **droite le planning corrigé** à la fin du projet afin de voir les écarts entre les prévisions et la réalité du développement.



PROFIL UTILISATEUR

Nous sommes partit sur un système de profile utilisateur afin de fidéliser l'utilisateur. Il reçoit des récompenses au fur et à mesure du jeu (Mode de jeu, IA, éléments de jeu, …).

Cela permet aussi de contrôler son expérience de jeu en jouant sur les paramètres d'expérience entre chaque niveau afin de débloquer des nouvelles fonctionnalités.

MODES DE JEU

La seule contrainte de la part du client est qu'il faut qu'il y ait le mode original du Picarete. C'est à dire le mode solo avec une IA.

Nous sommes donc partit sur une application de type jeu ou chacun joue à son tour. L'interface, minimaliste, change de couleur en fonction du joueur qui est en train de jouer actuellement pour avoir un feedback clair sur le jeu.

Le deux joueurs ont la même interface, mais elle change de couleur

SOLO

La particularité du solo est la présence d'une IA qui remplace le deuxième joueur. Plusieurs comportements sont disponibles à la discrétion du joueur :

- **IA simple et basique situationnelle**: Fonctionnement naïf. Cherche uniquement à compléter les cases à 3 arrêtes de prises. Sinon, elle prend une arrête qui ne permet pas au joueur de faire un carré au tour prochain.
- **IA prévisionnelle situationnelle**: Le même fonctionnement que l'IA simple. Mais qui essaye de prévoir les coups qui peuvent lui rapporter plus de carré d'un coup ou qui vont lui permettre de faire un carré dans X coups.
- **IA agressive situationnelle** : Le même fonctionnement que l'IA simple, mais qui, au lieu de prendre une arrête au hasard, cherche plutôt à bloquer l'autre joueur et à le forcer à faire des choix non-rationnel.
- **IA intelligente** : Cette IA fait des simulations de jeu afin de constituer des arbres qui lui permettront de décider intelligemment quel coup est le mieux à jouer. Grâce à Mr Eric Duchène, nous sommes partit sur un algorithme de type Monte Carlo Tree Search. Ce qui permet à l'IA de s'adapter aux différents modes de jeu.

MULTI

Uniquement sur le même écran pour des raisons de simplicité pour le joueur. En effet, un lobby pour un jeu aussi rapide est impensable car les joueurs ne prendraient pas le temps d'attendre que quelqu'un se connecte.

VARIATIONS

MALUS ARRETES

Sur certaines arrêtes se trouve un malus de points. C'est le premier joueur qui utilise l'arrête qui perd un certain nombre de point en fonction du nombre de carré fermé en même temps.

MALUS CARRE

Sur certain carré se trouve un malus de points. C'est le premier joueur qui fermer le carré qui perd un certain nombre de point.

ARRETE MAGIQUE

Sur certaines arrêtes se trouve un bonus de points. C'est le premier joueur qui arrive à utiliser l'arrête qui gagne le bonus de point en fonction du nombre de carré fermé en même temps.

CARRE MAGIQUE

Sur certain carré se trouve un bonus de points. C'est le premier joueur qui arrive à fermer le carré qui gagne le bonus de point.

ETENDUE

Chaque joueur gagne des points à la fin en fonction de la taille de chaque zone, c'est à dire le nombre de carré de la même couleur adjacent.

BERZERK

Les coups du joueur actif peuvent s'enchainer tant qu'il ferme arrive à fermer des carrés durant son coup actuel.

DIRECTION ARTISTIQUE

Pas de contrainte particulière de la part du client.

IDEES

FLAT DESIGN

Nous sommes plutôt partis sur des aplats de couleur, avec des couleurs simples et presque fade. L'utilisation d'Adobe Kolor a été requise pour créer la palette de couleur.

REFERENCES

En ce qui concerne les références, nous sommes plutôt partis sur des jeux à design très simple. Des petits jeux ayant eu beaucoup de succès malgré un design au couleur fade et sans formes complexes, uniquement des formes géométriques simples.

2048

Pour ses couleurs simples propres et non-vives. L'utilisation de formes simple rends honneur au Game Play simpliste mais efficace lui aussi (Version web / Version mobile).

DON'T TAP THE WHITE TILE

Pour son design et sa pâte graphique très marqué malgré l'utilisation de seulement 2 couleurs (Version web / Version mobile).

LISTE DES ECRANS

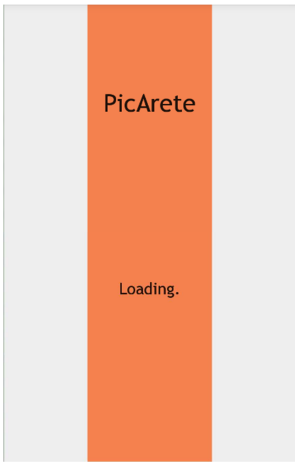
Voici la liste des écrans réalisés dans notre application.

1. Loading : Chargement des assets graphiques et des données trop longues à charger en jeu
2. Home : Menu principal de l'application. C'est là que le joueur peut naviguer entre les différents écrans et fonctionnalités
3. Solo : Ecran de jeu contenant l'interface pour un seul joueur
4. Multi : Ecran de jeu contenant l'interface pour deux joueurs
5. Profil : Recueil des informations et statistiques du joueur. Nous avons décidé de fusionner cet écran et l'écran de Customization afin d'avoir un flow de navigation au sein de l'application plus simple et de réunir les informations au même endroit

MOCK UP & ECRANS

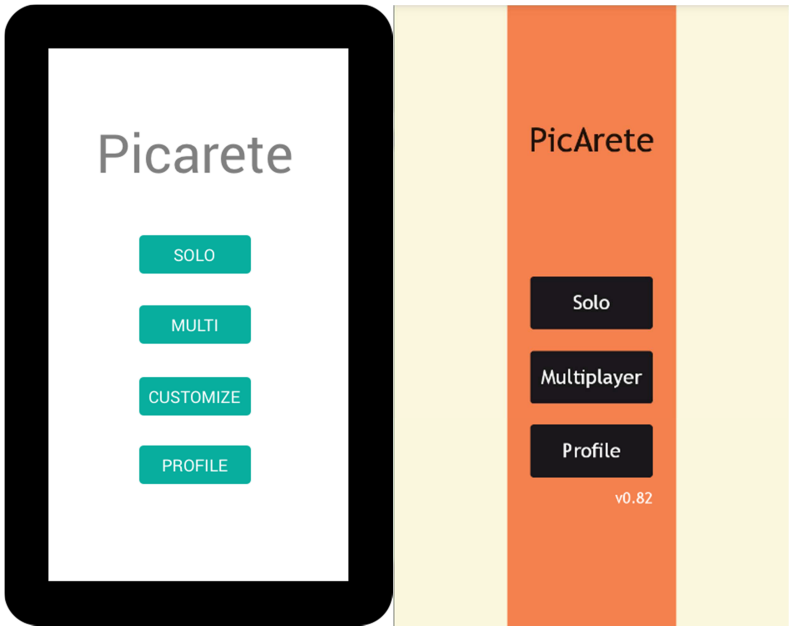
LOADING

Ecran de chargement très simple, mais avec une animation sur les points après le « *Loading* » afin de toujours signifier au client que l'application n'as pas crashé et qu'un chargement est en train d'être effectué.



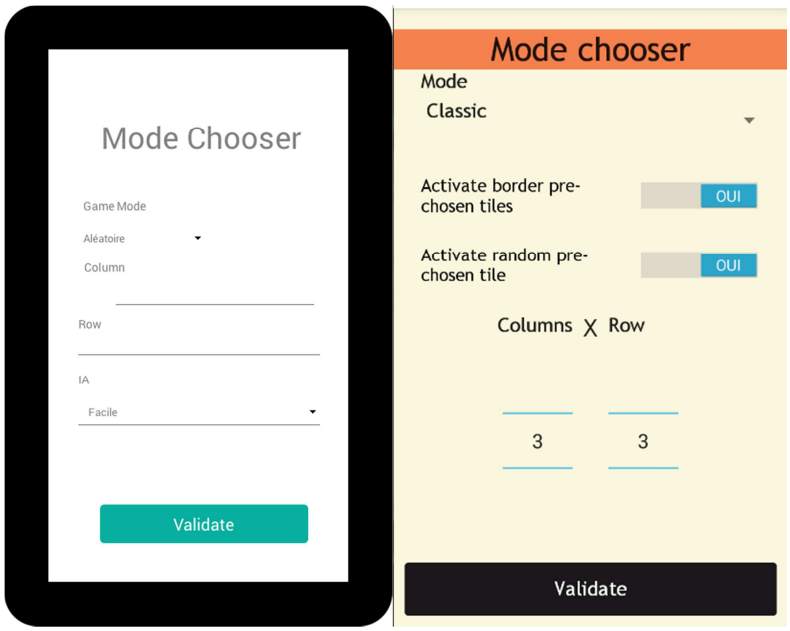
HOME

Sur cet écran se situe les différents boutons pour accéder aux écrans de l'application. L'intention est de faire un menu très sobre. Avec seulement le nom du jeu, les boutons et éventuellement un numéro de version.



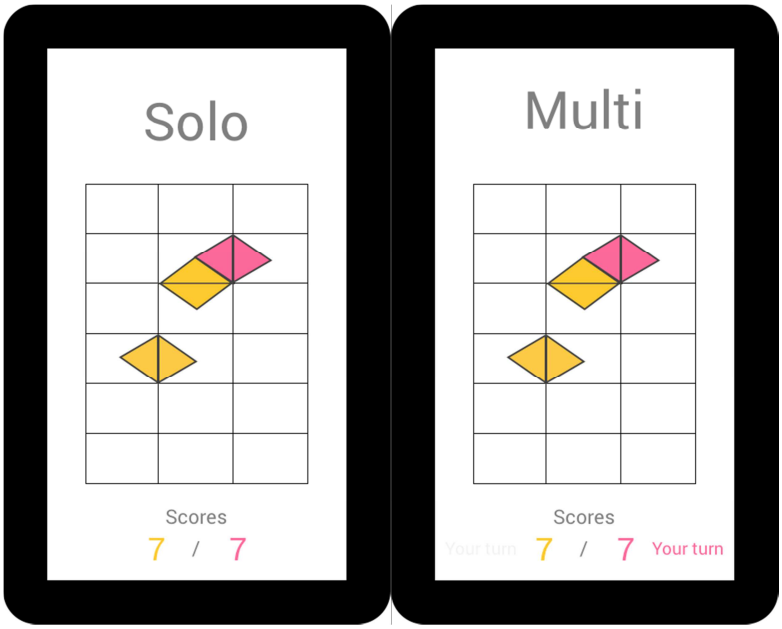
EN JEU

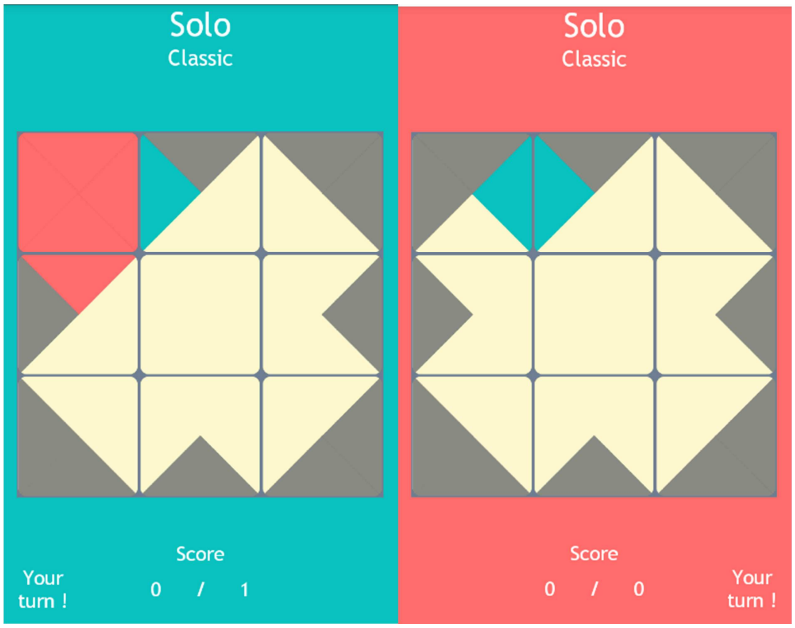
Le premier écran de jeu est le menu de sélection. Le plus clair possible, et donc potentiellement avec des icônes pour représenter chaque élément, le joueur peut choisir divers éléments qui vont conditionner sa partie (Taille du terrain, mode de jeu et en solo, la difficulté de l'intelligence artificielle)



JEU

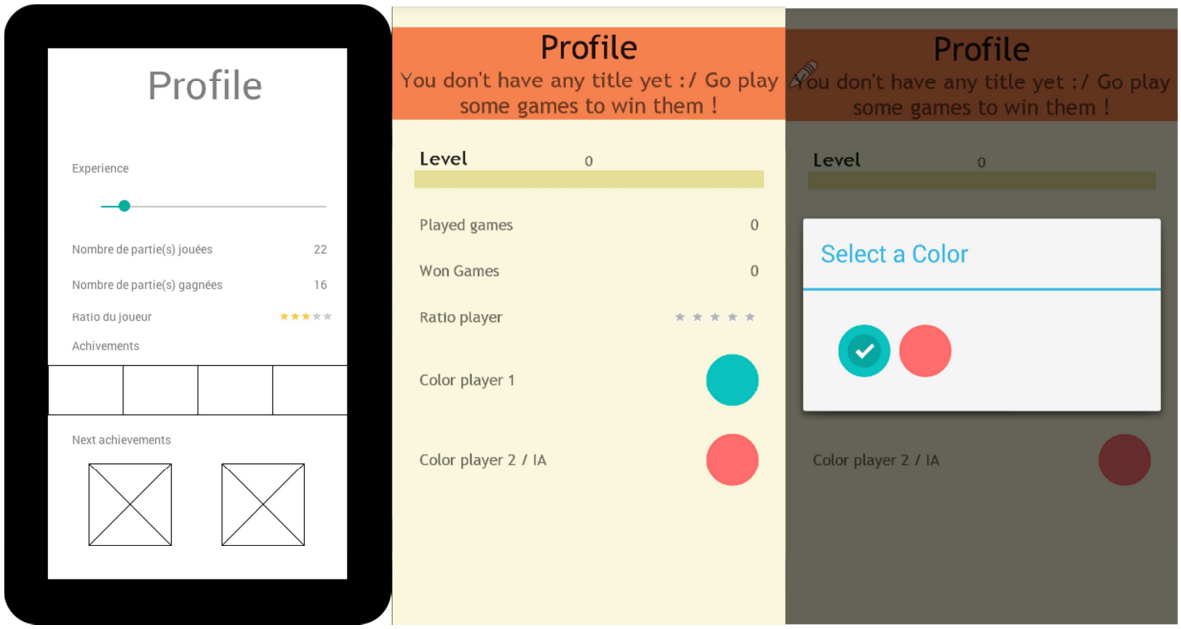
L'écran de jeu est très simple. Seuls quelques éléments d'interfaces sont présents pour montrer au joueur les informations utiles (Score, …)



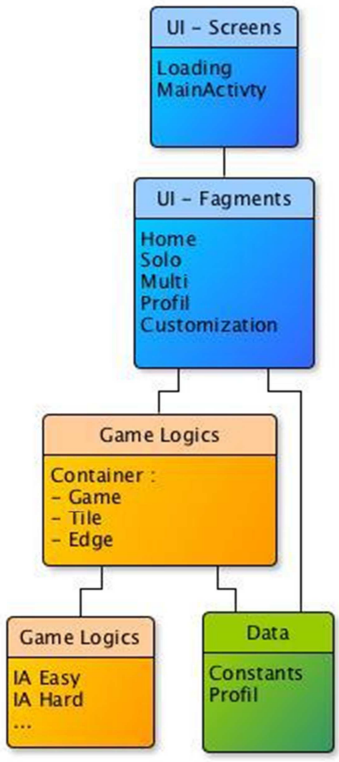


PROFILE

Le profil sert à montrer les différentes statistiques du joueur. Les informations sont simples et concises. Nous avons rajouté des pastilles de couleurs afin que chaque joueur puisse sélectionner la couleur de son jeu. Une touche de personnalisation qui permet au chacun d'avoir son interface et de s'y attacher.



ARCHITECTURE DE L'APPLICATION



L'application est constituée d'une couche UI (en bleu) qui va gérer tout ce qui est affichage des informations, du menu, et de la gestion des inputs. La couche Data (en vert) est accessible quasiment depuis toute l'application. Et enfin la partie Game Logics (en jaune) qui sert à faire fonctionner les mécaniques de jeu. Avec les différentes classes qui vont contenir les éléments de jeu.

DESIGN PATTERN ET THEORIE

Lors de la phase de conception de l'application, nous avons décidé d'utiliser des design patterns et des concepts indispensables de l'objet afin de nous faciliter la tâche. Nous avons utilisé :

- **Design pattern Stratégie** : Grâce à l'utilisation du polymorphisme pour les éléments de jeux ayant une forte similitude, comme les IA ou la construction des modes de jeu.
- **Les factory** : Très en lien avec le design pattern stratégie, il nous a permis de délivrer toujours la bonne stratégie en fonction des paramètres en entré de la factory.
- **Le MVC** : Le classique design Model, Vue, Controlleur. Il permet une séparation des données afin de garder une certaine indépendance des composants. Et une séparation logique du code.
- **Observer** : Il nous a été très utile pour faire remonter des évènements dans l'architecture complexe du jeu. Chacun écoutant l'autre afin de se faire passer les différents messages de fonctionnement du jeu (Click, Nouveau joueur, Jeu terminé, …)

LIBRAIRIES

Nous avons utilisé deux librairies pour ce projet.

La première est **Gson**, elle nous a permis de convertir des données (En l'occurrence la classe de stockage des informations, la classe User) en un objet de type chaine de caractères afin de pouvoir le sauvegarder dans les préférences partagées (Shared Preferences) pour garder en mémoire les informations utiles de chaque utilisateur.

La deuxième a été **ColorPickerCollection**. C'est une librairie de sélection graphique de couleur. Elle nous a été utile dans le profil afin que les joueurs puissent choisir leurs couleurs de jeu. Les principaux intérêts ont étés de nous faire un menu de sélection propre facilement et assez clair sans avoir l'aide de designer ou de graphistes.

AMELIORATIONS

Le jeu est, pour nous, déjà conforme aux objectifs que l'on s'était fixés. Malgré tout, si nous devions trouver des améliorations que nous pourrions envisager, nous pourrions parler de l'IA, des modes de jeux ou encore de la configuration fine du système d'expérience.

IA

Les IA situationnelles, c'est-à-dire 3 IA sur 4 ne sont pas programmés pour réagir correctement aux différents modes de jeux. Cela aurait été intéressant pour que l'expérience de jeu puisse se renouveler, au fur et à mesure des niveaux, que les IA apprennent à jouer avec les nouveaux éléments de Gameplay.

MODE MULTIJOUEUR A DISTANCE

Pour le mode multijoueur nous avons uniquement prévu un affichage sur un seul écran et donc sur le même périphérique. L'idée d'une interface de connexion à un mode multijoueur à distance grâce à un serveur était séduisante. Malgré le fait que l'on n'ait pas eu le temps d'imaginer les mécanismes liées à cette fonctionnalité, nous aurions pu penser à un système de classement, de partage sur les réseaux sociaux, …

NOUVEAU MODES DE JEUX

L'architecture du projet est un des gros points forts du projet car elle nous permet de réaliser des changements et de les intégrer assez facilement pour le client. Ce qui nous a donné beaucoup d'idée, notamment au niveau des modes de jeux qui pourraient être encore plus diversifiés. En apportant toujours plus de mécanique de jeux pour chaque type de joueur.

Par un exemple un mode qui mélange bonus et malus sur les carrés ou les arrêtes. Ou bien un mode qui cache les différentes cases spéciales et qui ne les dévoilent qu'une fois révélée.

CONFIGURATION DE L'EXPERIENCE DE JEU

Grâce à une phase de pré-production importante et structuré, nous avons réussi à programmer un jeu paramétrable entièrement grâce à des fichiers de configuration. L'idée intéressante derrière cela aurait été de faire une analyse poussée des données utilisateurs afin de pouvoir améliorer l'expérience de jeu en modifiant les données configurables (Expériences, déblocage d'éléments de jeu, difficultés des IA, …)