

Fishing Clicker

27/01/2021
Application Mobile

Adrien Paysant
Joris Monnet

haute école
neuchâtel berne jura

arc⁺

ingénierie
www.he-arc.ch

Table des matières

Fishing Clicker	1
Remerciements	3
Introduction	4
Choix du sujet.....	4
Analyse	5
Cahier des charges	5
Planification	5
Conception	6
Pêche à la main	6
Gestion de l'argent	7
Bateaux	8
Niveaux	10
Poissons	11
Persistance	14
Page About	15
Page Spécifications	16
Usage des ressources	17
Musique	17
Discussion	18
Conclusion	18
Pour aller plus loin.....	18
Références	19
Annexes	20
Diagramme de classe	20

Remerciements

Nous remercions Madame le Professeur A.Rizzotti qui nous a encadrés et permis de mener à bien ce projet. Nous remercions de plus Etienne Hüsler, pour les remarques constructives lors de la présentation et l'encadrement également.

Introduction

Le projet de semestre en développement mobile vise à investir et approfondir les connaissances qui sont appréhendées au sein du cours de développement mobile à la Haute Ecole Arc de Neuchâtel. Ce projet devant être réalisé entièrement dans le langage Kotlin, et utiliser au choix deux technologies parmi les suivantes : mesure de performance, persistance, utilisation moyenne de communication et utilisation de deux capteurs. Le choix est porté sur la mise en place de persistance ainsi que l'utilisation de deux capteurs, qui seront l'accéléromètre et le microphone.

Ce document vise à présenter les différentes étapes de la réalisation du projet Fishing Clicker. Dans ce but, les phases de choix de sujet, d'analyse, de conception et de discussion seront abordées successivement.

Choix du sujet

Afin de choisir le sujet, plusieurs facteurs ont été pris en compte. Tout d'abord, un jeu nous semblait tout indiqué. En effet, ce type d'application permet d'utiliser des capteurs et de sauvegarder des données de jeu. Cela répond donc au besoin de mise en place de persistance. En outre, Android Studio étant un nouvel IDE (*Integrated Development Environment*) et Kotlin un nouveau langage dans notre éventail de compétences en voies de développement.

Notre démarche de recherche de jeu s'est orientée vers un jeu à mécanisme itératif. C'est-à-dire une application dont la base est simple et permet de directement avoir un résultat. Ensuite on ajoute chaque fonctionnalité de façon unitaire augmentant les possibilités de notre application.

Les membres de l'équipes de développement étant tous des joueurs de clicker, un type de jeu qui a pour but de cliquer frénétiquement pour emmagasiner une ressource. Cette dernière servant elle-même par la suite à l'automatisation de sa propre collecte, permettant alors une expérience de jeu passive.

Les exemples les plus connus de ce type de jeu sont Cookie Clicker, Heroes Clicker, IDLE Mine Tycoon, et un jour peut être Fishing Clicker.

En somme, nous avons décidé de créer un clicker dont la progression est sauvegardée et d'utiliser l'accéléromètre et le microphone. Etant donné l'état de fait qu'un clicker est un jeu répétitif dont la mécanique est presque toujours la même, son corps propre tient dans la richesse de son monde. Dans notre situation, il s'agit de la pêche. Ainsi un tapotement de l'écran permet de pêcher un poisson, et des bateaux sont disponibles à l'achat pour pêcher en autonomie. De plus, des poissons au pouvoir spéciaux apparaissent aléatoirement et donnent lieu, quand ils sont attrapés, à un événement pouvant utiliser les capteurs, tout cela étant détaillé dans la partie conception sur ces poissons.

Analyse

Cahier des charges

Une fois le cadre de notre projet déterminé, il s'agit alors de définir le cahier des charges. Cela permet d'être en mesure de spécifier et de conceptualiser les besoins liés au jeu. Le but de la manœuvre étant finalement de dessiner les premiers traits du projet. Nous avons ainsi partagé nos objectifs en trois catégories : principaux, secondaires et facultatifs. Voici donc la liste de nos objectifs :

- *Objectifs principaux :*
 - i. Créer une page où le joueur clic ;
 - ii. Créer différents objets à acheter afin de développer la partie idle du jeu ;
 - iii. Créer une page de menu pour améliorer et acheter ces objets ;
 - iv. Sauvegarde et lecture des données de progression ;
 - v. Création d'événements utilisant les capteurs.
- *Objectifs secondaires :*
 - i. Changement de niveau ;
 - ii. Animer le jeu ;
 - iii. Changement de design de page de jeu ;
 - iv. Implémentation de différents poissons ;
 - v. Mise en place d'objectifs journaliers.
- *Objectif Facultatif :*
 - i. Mise en place de musiques.

Planification

Afin de pouvoir établir un planning, nous avons dressé un diagramme de Gantt concernant notre projet. Le voici :

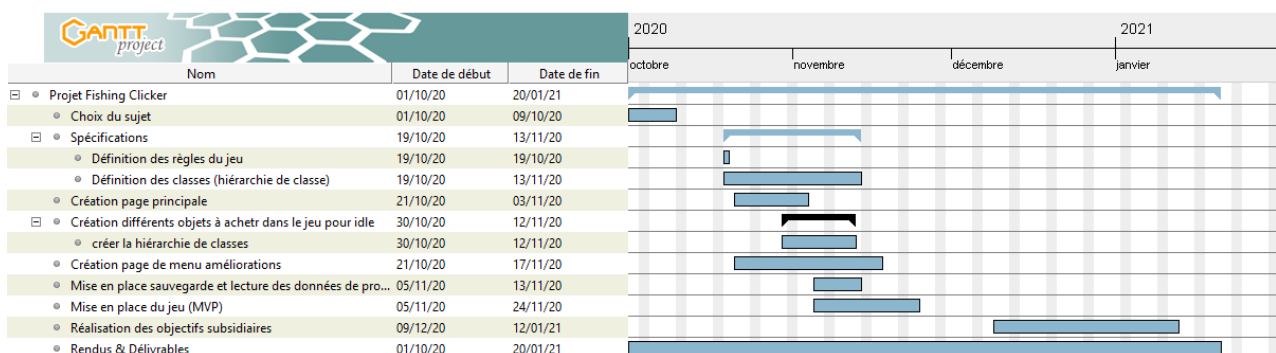


Figure 1-Diagramme de Gantt

A postériori, ce planning s'est avéré ambitieux, tout a été substantiellement décalé.

Conception

Pêche à la main

La base de notre projet est la pêche à la main. Elle se déroule dans la page principale de jeu :



Figure 2-Page principale

En haut à droite, on peut observer une somme d'argent. Cela correspond à l'argent globale de l'utilisateur. Son affichage, et sa gestion seront plus amplement détaillés ci-dessous. En haut à gauche, se trouve le bouton permettant d'accéder au menu pour acheter les bateaux. De même pour accéder à ce menu, on peut swiper vers la droite. Sur l'image ci-dessus, le fond est uni, il s'agit de la page principale de jeu en l'état de premier démarrage.

Par la suite, il accueillera une liste d'images des bateaux achetés ainsi que les poissons permettant de gérer les événements. Quel que soit l'avancement dans le jeu, un clic sur cette page, permet de pêcher un poisson est donc de gagner de l'argent. C'est sur cette page que les débuts du clicker se feront. En effet en attendant d'avoir suffisamment d'argent pour acheter son premier bateau, le joueur doit cliquer sur cet écran afin d'atteindre au plus vite le prix du premier bateau. Notons que le gain du clic change en fonction du niveau de l'utilisateur.

Gestion de l'argent

Dans les jeux de type clicker, le gain d'argent est exponentiel, plus on avance dans le jeu plus on gagne de l'argent. Mais également plus les améliorations et achats de bateaux coûtent cher. De ce fait, il faut pouvoir gérer de très grand nombre, plus grand que le type Long. De ce fait nous utilisons la classe BigInteger qui permet de gérer des grands nombres.

De surcroit, un affichage direct des grands nombres serait difficile à lire pour l'utilisateur et par ailleurs absolument désastreux sur un plan ergonomique. Nous avons revisité le système d'IDLE mine Tycoon. Ainsi on affiche un maximum de 6 chiffres puis une ou deux lettres pour donner la puissance de 1000 ayant cours.

Afin d'expliciter, les chiffres vont aller de 0 à 999 999, ensuite ce sera 1000k jusqu'à 999 999 k puis les lettres sont M, B et ensuite l'alphabet d'a à z. Afin d'être sûr d'avoir suffisamment de chiffres possibles notamment dans le cadre d'une suite à ce projet, une fois le z attends, on passe à aa, puis ab et ainsi de suite jusqu'à zz(inatteignable car trop massif).

Afin de faciliter notre utilisation, notre classe Money permet cet affichage. L'utilisateur en possède une instance (l'argent global qui est affiché). Et chaque bateau en possède deux instances, une pour le prix d'achat et une pour le prix d'amélioration.

Bateaux

Les bateaux sont les objets permettant de gagner de l'argent passivement. Nous avons fait le choix (cela dépend des clickers) de ne gagner de l'argent passivement que si le jeu est en premier plan afin de privilégier les joueurs très actifs sur l'application.

Ces bateaux sont accessibles par le menu qui vient du côté gauche de l'application :

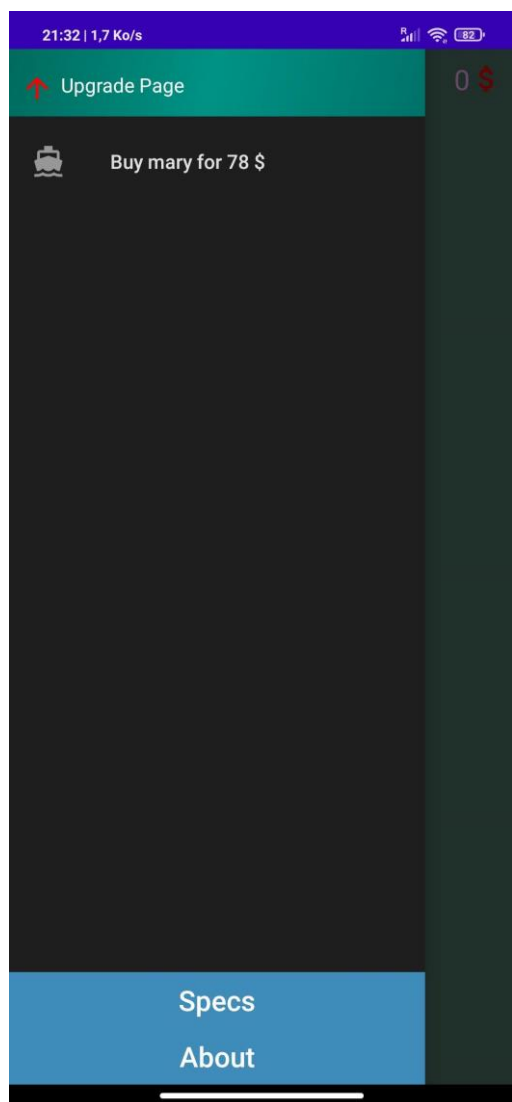


Figure 4-visuel "neuf" de la page d'achat

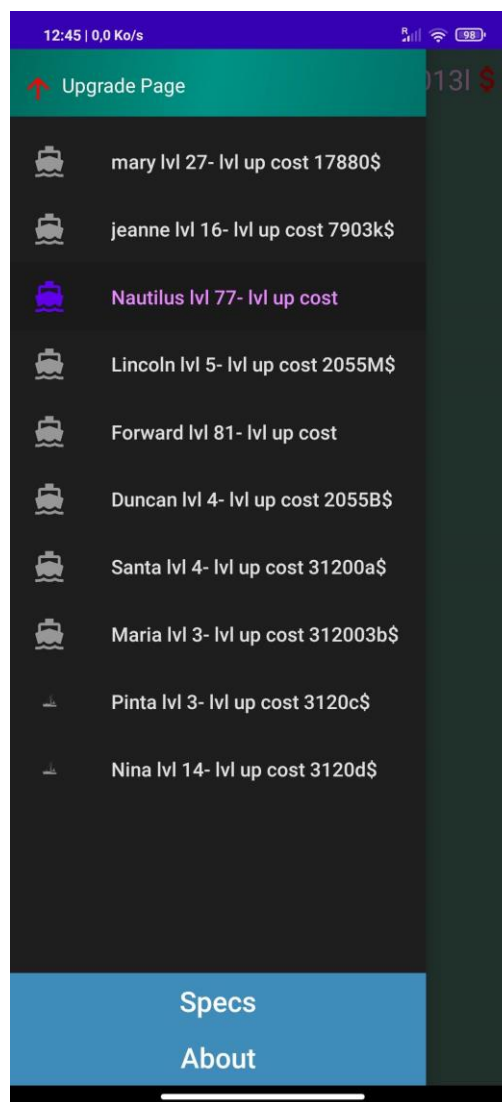


Figure 3-visuel après usage de la page d'achat

Ci-dessus, à gauche le menu en début de partie lorsque l'on n'a pas de bateau ; le premier est alors proposé à l'achat. Ci-dessus à droite, le menu une fois que tous les bateaux possibles dans notre version 1.2 sont achetés. Plus les bateaux achetés dans le menu sont bas, plus ils permettent au joueur de gagner beaucoup d'argent, pour savoir combien le joueur gagne à la seconde, il peut utiliser le menu Specs détaillé plus bas. Un seul bateau est achetable à la fois, à chaque fois que le joueur achète un bateau, le suivant devient visible et donc libre à l'achat.

Une fois un bateau acheté, il devient visible sur la page de jeu principale :

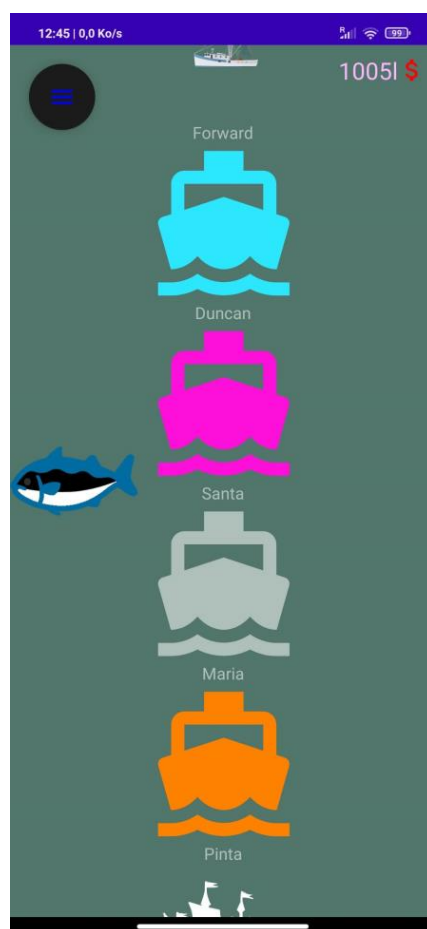


Figure 5-Visuel de la page principale en fonctionnement.

Comme vous pouvez le voir un bateau, en plus de posséder une image le représentant sur la page principale et dans le menu pour l'acheter, possède aussi d'autres attributs. On voit ici son nom, visible aussi dans le menu. Chaque bateau possède un prix d'achat, un prix d'amélioration, un niveau et enfin une efficacité qui est le gain par seconde que rapporte le bateau au joueur.

Les bateaux sont semi-génériques, car la plupart de leurs attributs sont lus dans un fichier texte pourvus et stocké par les assets de l'application. Chaque ligne correspond à un bateau sous le format « *nom;efficacité;id ;prix_d'achat* » cela permet en tant que développeur de facilement modifier ces différentes valeurs.

L'id permet de lier le bateau à l'item dans le menu auquel il correspond et au drawables qui le représente. Le prix d'amélioration quant à lui est calculé à partir du prix d'achat. A chaque amélioration le prix d'amélioration du prochain niveau de bateau correspond au prix payé multiplié par 1.5. Mais cela ne rend pas les bateaux complètement génériques car il faut ajouter les items dans le menu ainsi que les drawables directement dans le code métier. Enfin, les bateaux achetés sont sauvegardés afin de ne pas perdre la progression du joueur dans le cas de l'arrêt de l'application.

Niveaux

Le joueur augmente passivement de niveaux par palier. Afin de passer au niveau suivant, le joueur doit amasser une somme d'argent. Cette somme d'argent à atteindre est uniquement fonction du niveau de la partie actuel. Cela permet d'équilibrer quelque peu la valeur du clic par rapport aux gains engrangés via les bateaux. La valeur du gain par clic augmente également en fonction du niveau du joueur.

De manière générale il est actuellement plus efficace avec beaucoup de bateaux de ne plus cliquer, mais cela permet de rendre le clic intéressant plus longtemps dans la partie pour le joueur.

Enfin, ce changement de niveau entraine un changement de couleur de fond. Pour ce faire, nous avons à notre disposition 17 couleurs de fond qui bouclent, c'est-à-dire que le 18^{ème} niveau à la même couleur que le premier. Cela permet d'ajouter du dynamisme au jeu, un simple fond fixe étant ennuyant. Voici les couleurs choisies :




















						0 -> R.color.colorMainBG0
						1 -> R.color.colorMainBG1
						2 -> R.color.colorMainBG2
						3 -> R.color.colorMainBG3
						4 -> R.color.colorMainBG4
						5 -> R.color.colorMainBG5
						6 -> R.color.colorMainBG6
						7 -> R.color.colorMainBG7
						8 -> R.color.colorMainBG8
						9 -> R.color.colorMainBG9
						10 -> R.color.colorMainBG10
						11 -> R.color.colorMainBG11
						12 -> R.color.colorMainBG12
						13 -> R.color.colorMainBG13
						14 -> R.color.colorMainBG14
						15 -> R.color.colorMainBG15
						16 -> R.color.colorMainBG16
						else -> R.color.colorMainBG0

Figure 6-Choix des couleurs pour le fond du jeu.

Poissons

Le jeu met en place trois poissons, deux sont positifs et un négatif. Le poisson bleu permet de multiplier son argent par 10 si on clique dessus pendant qu'il est visible :

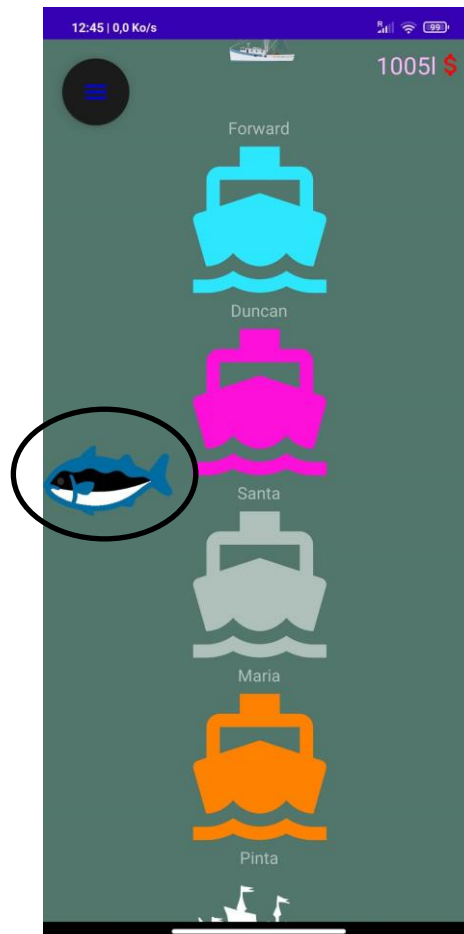


Figure 7-Exemple d'un poisson

Le poisson doré est un autre bonus pour le joueur et lance l'utilisation d'un capteur. Lorsqu'il est cliqué, une page apparaît, demandant à l'utilisateur de secouer son téléphone. Une fois ceci fait, un gain en pourcent apparaît :

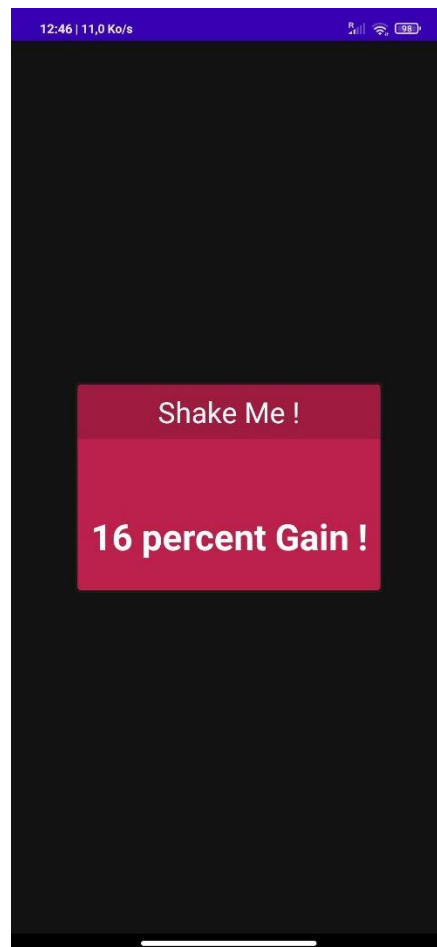


Figure 8-page du Shake Event

Dans le cas où la personne ne secoue pas, le gain est de minimum 1%, nous n'avons jamais réussi à monter au-dessus de 17% de gain, nous vous invitons donc à faire plus (sans casser le portable). Ce gain est un bonus d'argent, ici, l'argent est multiplié par 116%. Le poisson doré est le poisson le plus rare même si les trois poissons sont visibles relativement souvent afin de dynamiser le jeu et de pousser l'utilisateur à plus d'interactions. Cette fonctionnalité utilise donc l'accéléromètre du téléphone.

Pour finir, le troisième poisson est un malus. Il s'agit d'un requin prêt à dévorer 90% de notre argent. Lorsqu'il est cliqué (attention sa hitbox prend quasiment toute la page, difficile de l'éviter), il lance cette page :



Figure 9-page Blow Event

Afin de sauver notre argent, il faut faire fuir le requin. Pour ce faire, un grand bruit dans le microphone fonctionne, ou pour plus de discrétion (si vous êtes dans le train au moment de jouer), soufflez dans votre micro. Si le microphone détecte suffisamment de bruit, votre argent est sauvé, sinon il ne vous en restera plus que 10%. Il s'agit donc là de notre deuxième capteur utilisé sur le téléphone.

Pour résumer, le poisson doré utilise l'accéléromètre pour un bonus et le requin utilise le microphone pour éviter le malus.

Persistence

Nonobstant notre amour de la difficulté, il est important pour l'utilisateur d'avoir sa progression sauvegardée et ce même lors de l'arrêt de l'application. De ce fait, on sauvegarde les données de l'utilisateur ainsi que tous les bateaux achetés. On sauvegarde via les *sharedpreferences*.

Pour l'utilisateur sont sauvegardé :

- Son niveau
- Son argent
- La valeur de son clic

Pour les bateaux :

- Leur efficacité
- L'id qui permet de les lier à leur item et drawables
- Le prix d'achat
- Leur niveau

Ensuite on reconstruit la liste des bateaux a afficher dès que l'utilisateur revient sur l'application, sans oublier si celui-ci n'a pas acheté tous les bateaux disponibles, de rajouter le prochain bateau comme possibilité d'achat.

Page About

Afin de permettre un référencement direct du nom de l'école ainsi que des membres de l'équipe directement au sien de l'application, une page présentant ces informations a été réalisée. Cette page présente le visuel de la figure suivante :



Figure 10-Visuel de la page About.

Cette page est accessible à tout moment depuis le menu d'achat. En outre, elle permet une navigation aisée, de part la présence d'un bouton de retour en arrière.

Page Spécifications

Dans l'optique de permettre à l'utilisateur une consultation aisée de la quantité de poissons pêchés, une page permettant de mettre en valeur le rendement à la seconde de la partie ainsi que le niveau atteint a été mise en place. La figure suivante présente la page de spécifications.

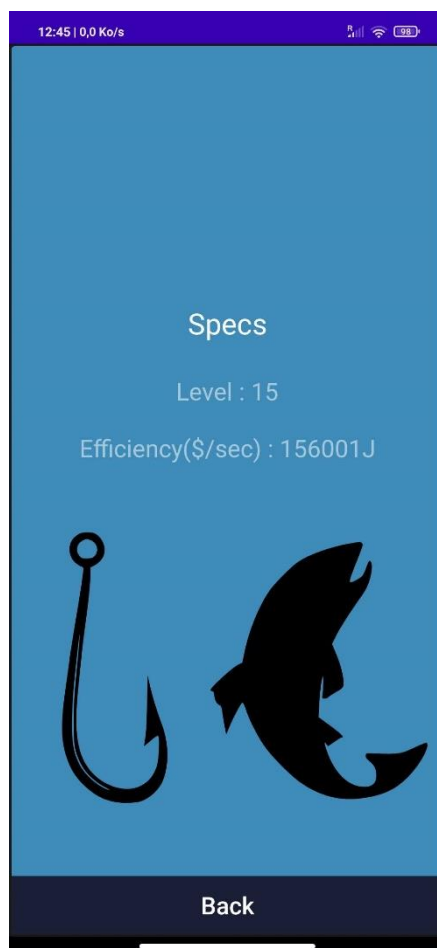


Figure 11-Visuel de la page de spécifications

Dans l'exemple de la figure ci-dessus, le niveau atteint est de 15, est la production à la seconde de poisson est équivalente à 156 001 J.

Usage des ressources

Dans le cadre de notre projet nous utilisons le plus possible les ressources du projet. Pour les images, nous utilisons des images en format svg, c'est-à-dire des vecteurs pour ne pas devoir gérer plusieurs dossiers avec les différentes tailles d'images png. De plus nous utilisons un grand nombre d'images provenant directement de material.io accessibles depuis Android studio, c'est-à-dire les icônes fournies par google sous licence Apache 2.0. Nos couleurs sont répertoriées dans le fichier color.xml.

Enfin toutes les chaînes de caractères visibles par l'utilisateur et utilisées dans les layouts ou menu en xml sont répertoriées dans le fichier string.xml. Cela a pour avantages majeurs de pouvoir facilement traduire l'application dans sa globalité.

Musique

La musique était un objectif secondaire, mais au vu de la simplicité d'implémentation (simplement un objet mediaPlayer) nous avons décidé de le mettre en place.

Malgré tout, sous cette apparente simplicité, se cache diverses contraintes dues aux différents intents qui sont lancés avec les poissons ou aux déplacements dans les pages specs ou about. Pour résoudre ces problèmes le mediaPlayer est un companion object ce qui permet d'en avoir qu'une seule instance même parmi les classes enfants de MainActivity.

La musique est en boucle et ne s'arrête que dans le onDestroy() de l'application c'est-à-dire lorsqu'on quitte l'application. Lorsque l'application est mise en arrière-plan, la musique continue, comme une invitation pour revenir jouer.

Pour le contrôle du volume, c'est à l'utilisateur de le faire directement via son bouton physique.

Cette musique est libre de droit puisque composée par M.Monnet.

Discussion

Conclusion

Ce projet nous a permis de réinvestir au travers d'une démarche d'approfondissement les connaissances nouvelles abordées durant le cours de développement mobile.

Ainsi nous avons pu répondre à tous les objectifs principaux ainsi qu'une majorité des objectifs secondaires. Subsidiairement, l'objectif facultatif a été réalisé. Afin de détailler quelque peu, un seul objectif secondaire n'a pas du tout été réalisé, il s'agit de l'ajout d'objectifs journaliers afin de pousser le joueur à revenir jouer chaque jour. De surcroît, un objectif secondaire, l'animation du jeu a été initié via l'animation lorsque l'on rentre ou quitte le menu Specs ou About. Cependant, cet aspect du jeu est encore inachevé, le but était d'animer aussi les bateaux et les poissons pour rajouter du dynamisme au jeu. Malgré cela, étant secondaires, on peut dire que le cahier des charges a bel et bien été complété. Le résultat de ce semestre de projet est donc positif, avec beaucoup de nouvelles connaissances.

Pour aller plus loin

Afin de poursuivre l'aventure Fishing Clicker, de nombreuses voies s'offrent à nous. Les plus évidentes seraient de développer l'interface utilisateur par des ajouts de détails. L'ergonomie du système d'achat ainsi que des informations pour chaque bateau pourrait également être optimisée. De plus, notre objectif secondaire d'ajouter des tâches journalières pour fidéliser le joueur nous semble toujours une très bonne idée.

Références

- i. La musique du jeu a été créée par Joris Monnet.
- ii. Toutes les images proviennent soit de notre applications, soit des pages Wikipédia de Kotlin, Android ou Android Studio.
- iii. Ces différents tutoriels nous ont permis d'utiliser aisément les capteurs :
 - <https://johncodeos.com/how-to-create-a-popup-window-in-android-using-kotlin/>
 - <https://gist.github.com/h4ck4life/6433506>
 - <https://www.tutorialspoint.com/how-to-detect-shake-events-in-kotlin>

Annexes

Diagramme de classe

