# Mini Games

M. Friedli, A. Gillioz, J. Guerne
He-Arc Ingénierie
2000 Neuchatel

11 juillet 2017

**Résumé**

La HES d'été permet aux étudiants de deuxième année d'étude dans le domaine de l'informatique la possibilité de travailler sur un projet libre dans le but d'approfondir leurs connaissances.

Ce rapport décrit, explique les choix d'implémentations pris dans la réalisation de notre projet Mini Games.

Une planification des tâches ainsi qu'une spécification du travaille ont été réalisés dans le but d'organiser au mieux le temps à disposition.

# Table des matières

# Chapitre 1

# Introduction

Mini Gamesest une application offrant la possibilité de jouer à des minis jeux très classique tel que le morpion ou la bataille navale en réseau et en multi-platforme. C'est à dire que deux personnes l'une sur son téléphone android et l'autre sur son ordinateur auront la possibilité de se défier à une partie de jeu en ligne. Deux grands outils on été utilisé pour faciliter l'implémentation de ce projet, il sagit du framework Libgdx, qui a servit à deployer le même programme sur différentes platforme, et de la libraire kryonet, qui a elle facilitée les échanges réseau.

# Chapitre 2

# Planification

# Chapitre 3

# Conventions

# Chapitre 4

# Identité graphique

Le jeu possède un style fortement inspiré de l'univers des comic book (bande dessinée principalement américaine) ce choix d'inspiration est principalement lié à l'utilisation d'un très bon thème graphique pour les éléments d'interface de LigGDX. Le thème nous plaisant nous avons choisi d'associer notre jeu à cet univers en adaptant l'interface, les textes et les images. C'est de plus un choix qui sort de l'ordinaire car il est plutôt rare de voir des jeux reprenant ce sytle graphique ce qui est à notre avis un point positif pour notre projet.

# Chapitre 5

# LibGDX

Dès le lancemment du projet, nous nous sommes orienté vers libGDX qui est un framework Java gratuit et open source permettant la conception de jeux vidéo. Nous avons fait le choix de travailler avec ce framework en particulier, car nous avions découvert son existence quelque temps auparavant et, en apprenant à le connaître, nous avons découvert à quel point il facilite le déploiement multi-platforme. LibGDX nous a permis de gagner en temps précieux au niveau de l'implémentation puisqu'il propose nativement des fonctionnalités comme la gestion de stages ou de cameras qui sinon aurait dû être créés à la main. Étant un framework connu et grandement utilisé, il est simple de trouver des renseignements ou de l'aide concernant sa façon de fonctionner. Nous n'avions aucune expérience dans son utilisation pourtant il ne nous a fallu que très peu de temps avant de commencer à d'obtenir de bons résultats.

# Chapitre 6

# Kryonet

Ayant travaillé cette année sur des échanges réseau en Java sans utiliser de libraire externe nous avons pu réaliser que la tâche était fastidieuse, c'est donc naturellement que nous nous sommes tourné vers Kryonet qui est une librairie open source permettant de faciliter la communication entre différents clients. Un des points essentiel du projet était de pouvoir garentir que LibGDX et Kryonet pouvaient cohabiter, après quelques tests et recherches nous avons pu réaliser que c'était bel et bien ée cas ( du moins pour le déploiment sur ordiateur et android).

Chapitre 7

# Architecture logicielle

# Chapitre 8

# Communication réseau

Comme dit plus haut, le programme est une collection de minis jeux auxquels les clients auront la possibilités de jouer à plusieurs. Le programme est décomposé en deux parties : la partie client et la partie serveur. La communication entre ces deux parties est facilitée par l'utilisation de Kryonet une librairie Java open source conçu pour gérer les échanges réseau.

## 8.1 Les packets

Les données envoyées entre le client et le serveur transitent sous la forme de Packets, les packets sont des classes présentent chez le client et chez le serveur enregistrer au lancement du service.

Pour un client, si une connexion à été établie, il lui est possible d'envoyer par TCP ou UDP des Packets au serveur. Dans le cas complémentaire, pour recevoir les différents Packets émmanent du serveur le client met en place un listener qui va automatiquement gérer la réception des Packets et analyser leur contenu. L'implémentaion du serveur est identique à la nuance près qu'elle laisse le choit de la personne (de l'adresse) à qui sera envoyé le Packets. En effet toutes les infos, tous les Packets, transitent par le serveur, c'est lui ensuite qui les traitent et les renvoient aux différents clients conscernés.

## 8.2 Initialisation de la connexion

A l'ouverture du programme le client est invité à entrer une adresse de serveur et un pseudo pour tenter ensuite de se connecter. Afin de faciliter l'entrée de l'adresse du serveur la fonction de découverte des hôtes fournie par Kryonet a été utilisée, elle va fournir une liste d'adresse qui seront ensuite présentées à l'utilisateur sous la forme d'une liste déroulante si l'utilisateur sélectionne un élément de la liste l'adresse est automatiquement copié dans le champs de l'adresse du serveur.

Une fois que le serveur à reçu un Packet de login (une tentative de connexion a été envoyée) le serveur stocke le nouveau joueur dans une liste, il comfirmera ensuite le bon déroulement des opération au client en lui envoyant un Packet de confirmation. Ce Packet de confirmation le client l'utilise comme signal pour changer d'écran et afficher le menu de sélection des jeux.

## 8.3 Communication en jeu

Quand un joueur décide de lancer une nouvelle partie d'un jeu il transmet un Packet au serveur donnant comme information son ID et le jeu auquel il shouaite jouer. Le serveur va, une fois le Packet reçu, verifier si quelqu'un est déjà en train d'attendre de démarrer un partie de ce jeu ou non. Comme ce projet n'implémente que des jeux à deux joueurs la "salle d'attente" pour jouer à un jeu ne sera jamais composée de plus d'une personne, ce joueur (ou plutôt son ID) est stocké dans un variable faisant office de salle d'attente du jeu désiré.

Quand un second joueur se connecte la pertie peu commencer ! le serveur prépare donc un Packet de création de partie contenant l'ID et le nom de tous les joueurs de la partie, mais également le numéro (également appelé ID) de la partie en elle même (utile plus tard).

Tour à tour les joueurs envoient des informations aux serveurs propre au jeu auquel ils sont en train de jouer, le serveur va ensuite lui se charger de tester si la partie est terminée ou non et enverra les bons Packets en conscéquence.

Quand une partie est finie un message s'affiche chez les joueurs leur idiquant s'ils ont gagné, perdu ou encore fait un match nul (morpion). Ils sont ensuite invité à cliquer sur l'écran pour revenir à la sélection des minis jeux.

Si un joueur quitte la partie alors qu'elle n'est pas terminé le serveur en est informé. Il récupère ensuite les informations de la partie que ce joueur était en train de faire et contact son adversaire pour lui signaler l'abandon de son adversaire.

# Chapitre 9

# Minis jeux

Dans ce chapitre seront présentés les différents minis jeux mis en place dans ce projet aisni que leur implémentation. Le serveur garde un historique des jeux sous la forme d'une liste d'objets contenant les informations sur les différents joueurs opposés durant la partie.

## 9.1 Morpion

Le morpion est un jeu très simple dans lequel deux joueurs s'affronte sur un plateau de 3 x 3 cases. Chaque joueur possède des caractère (joueur 1 'x' et joueur 2 'o') tour à tour ils vont devoir placer ces cartacères dans le plateau de jeu dans le but de faire une ligne horizontale,verticale ou encore une diagonale. Si la partie se finit sans qu'aucun joueur n'est réussit à remplir une ligne/ une diagonale c'est un match nul.

## 9.2 Bataille navale

# Chapitre 10

# Conclusion

Ce chapitre propose une vue d'ensemble récapitulative sur le travaille effectué. Les probématiques encore présentent dans le programme à ce jour ainsi que les différentes idées d'améliorations future seront exposées et détaillées.

## 10.1 Problematique

Une des problématique rencontrée a été la mise en place d'un serveur sur android. En effet, LibGDX "facile" le portage de l'application client vers les périphériques android mais le serveur à lui été dévelopé sans libgdx. Une ébauche d'application android oppérant de la même manière que le serveur Java classique a été mise en place, l'implémentation c'est avérée très lourdes et peu fructueuse dans le sens où lorsqu'un client se connectait à un serveur android celui-ci crashait immédiatement.

## 10.2 Améliorations

La communication entre pc et android est fonctionnelle mais, comme expliqué plus haut, notre shouaite était d'étendre notre proramme à encore deux autres plate-formes : IOS et HTML5 (navigateur web). Certain problème d'implpémentation pourrait survenir (utilisation d'outil non supportés sur certaines plate-formes, etc.) mais avec du temps l'application pourrait être totalement multi-plateforme.

# Chapitre 11

# Bibliographie

image pirate
Lien de l'image