Rapport



**Java & LibGDX**

Présenté à :

**RIZZOTTI Aïcha**

**PASIN Marcelo**

Étudiants :

**VAUCHER Joël**

**OMBANG NDO Charles**

**RODRIGUES LOURENÇO Daniel**

Abstract

Ce rapport présente notre projet académique P2 réalisé durant l’HES d’été.

Table des matières

[1. Introduction 3](#_Toc460929556)

[2. Conception 3](#_Toc460929557)

[2.1. Les Uses Cases 3](#_Toc460929558)

[2.2. Diagrammes de séquence 3](#_Toc460929559)

[2.3. Schéma de classes 5](#_Toc460929560)

[2.4. Architecture de logicielle 6](#_Toc460929561)

[3. Outils de conceptions et développement 6](#_Toc460929562)

[4. Fonctionnement 6](#_Toc460929563)

[5. Tiled 6](#_Toc460929564)

[6. LibGdx 7](#_Toc460929565)

[6.1. PlayScreen, World & JdcGame 7](#_Toc460929566)

[6.2. Sprite 7](#_Toc460929567)

[6.3. InteractiveTileObject 7](#_Toc460929568)

[6.4. Collision & Mask 7](#_Toc460929569)

[6.5. Son 7](#_Toc460929570)

[6.6. Changement de Screen 8](#_Toc460929571)

[7. Problèmes rencontrés et solutions 8](#_Toc460929572)

[8. Améliorations possibles 8](#_Toc460929573)

[9. Conclusion 9](#_Toc460929574)

[10. Bibliographie 10](#_Toc460929575)

# Introduction

# Conception

Ce chapitre présente les éléments importants de la phase de conception.

## Les Uses Cases

Le diagramme des cas d’utilisations métier du projet est représenté ci-dessous.

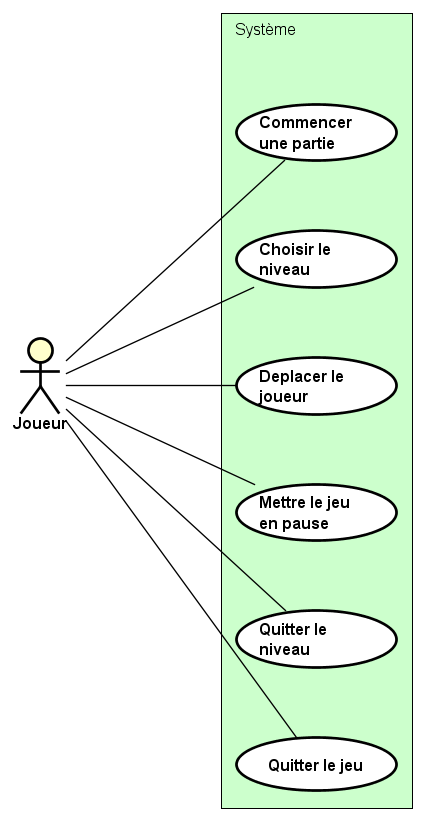


Figure 1 - Diagramme des cas d'utilisation

## Diagrammes de séquence

Le diagramme d’Uses cases ci-dessous montre les différentes fonctionnalités que le logiciel doit offrir à ses utilisateurs.

### Séquence dans le menu

Ce schéma présente le déroulement des interactions lorsque l’utilisateur est dans le menu principal. Le déroulement des interactions dans les sous-menus est le même donc un seul diagramme est présenté.

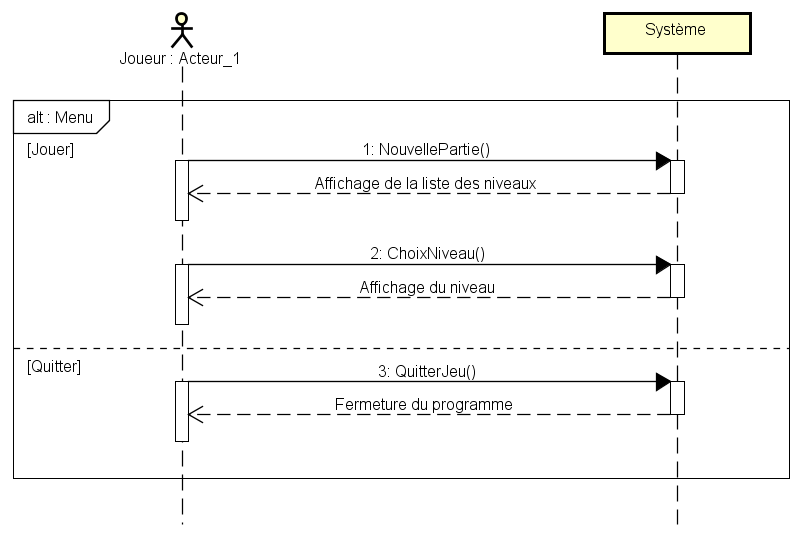


Figure 2 - Diagramme de séquence dans le menu

### Séquence durant le jeu

Voici le diagramme de séquence lors du déroulement d’une partie.

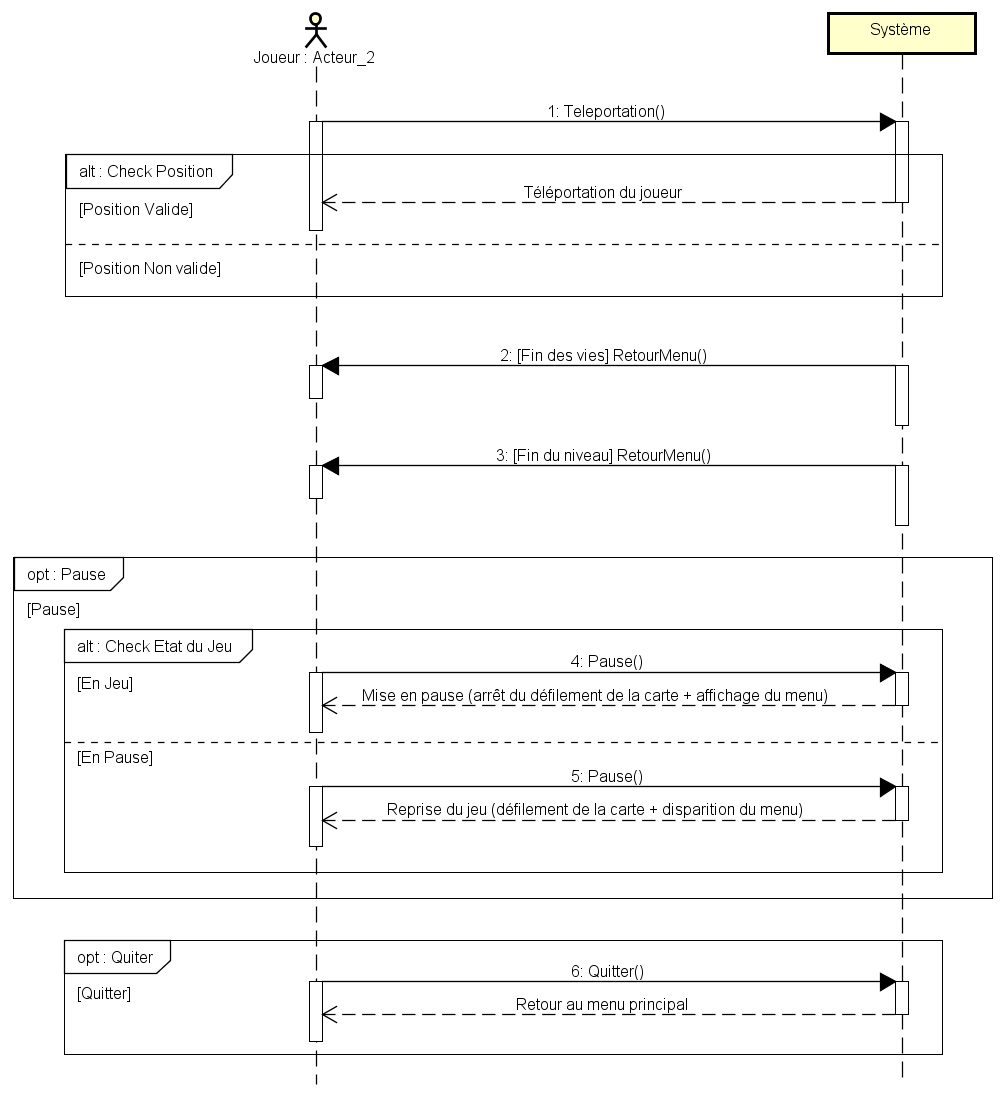


Figure 3 - Diagramme de séquence durant le jeu

## Schéma de classes

Voici le schéma de classes du projet. La version de ce schéma correspond au projet dans son état actuel. Le schéma prévu initialement se trouve en annexe.

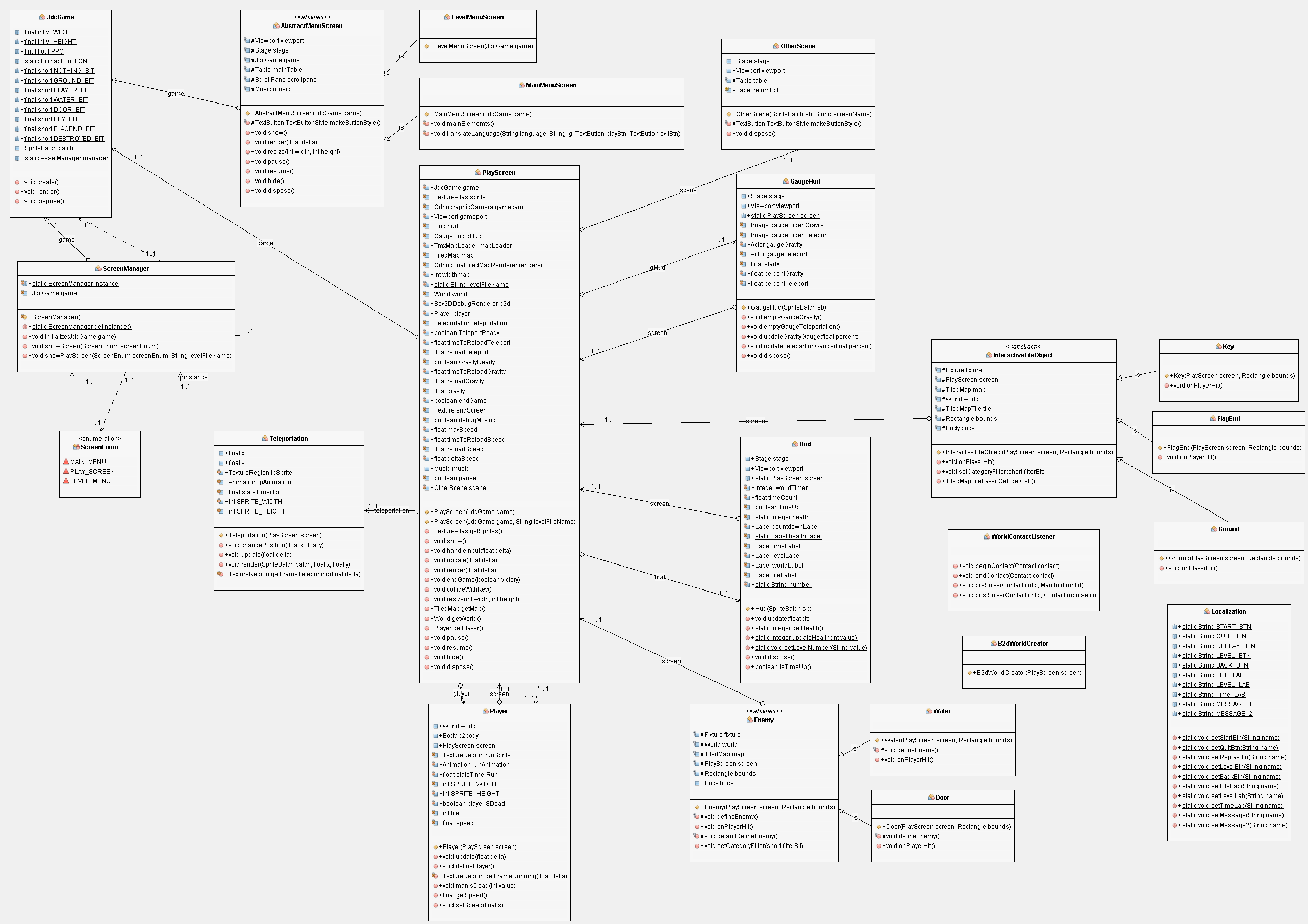


Figure 4 - Schéma de classes

## Architecture de logicielle

# Outils de conceptions et développement

La mise en place de ce projet a nécessité des outils assez simples d’utilisation et surtout, des outils utilisés quotidiennement pour notre formation.

Tableau 1 - Liste des outils utilisés

|  |  |
| --- | --- |
| Nom | Description |
| NetBeans | C’est l’IDE retenue pour la programmation de notre logiciel. |
| LibGdx | FrameWork pour les jeux 2D/3D en Java |
| GitHub | Comme outils de collaboration. |
| Astah | Pour les diagrammes de modélisation utilisés. |

Nous avons également pu compter sur un tutoriel expliquant la base de LibGdx :

<https://www.youtube.com/playlist?list=PLZm85UZQLd2SXQzsF-a0-pPF6IWDDdrXt> (Aureli, 2015)

# Fonctionnement

# Tiled

Tiled est un logiciel de création de map possédant différent niveau d’image (background, foreground) et des objets de familles différentes (mur, clé, sol). Le logiciel exporte les fichiers en .tmx qui grâce à une librairie de LibGdx peuvent être très facilement récupérer dans le code Java.

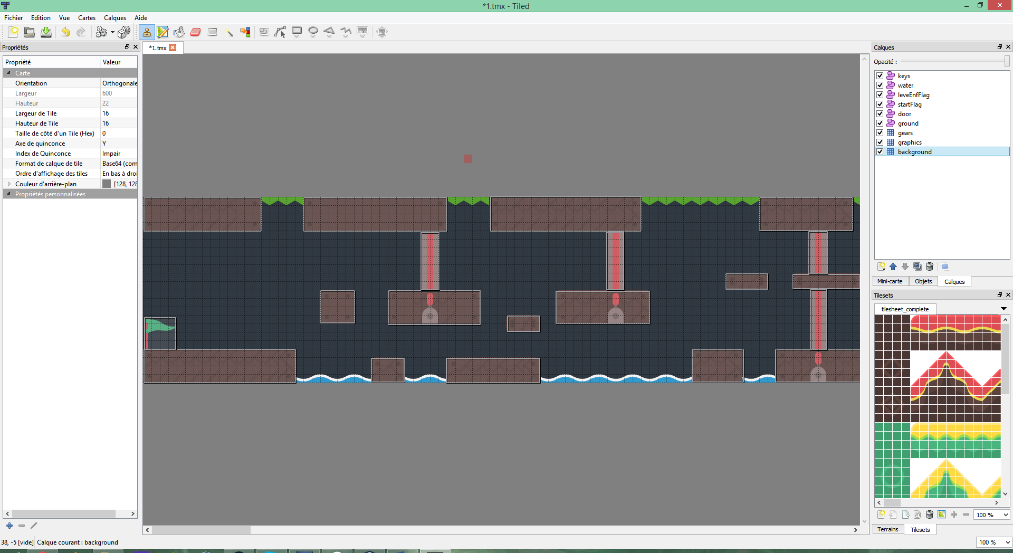


Figure 5 - Interface de Tiled Map

Le fichier contient différents calques (background, graphics & gear), ainsi que des sélections (ground, door, startflag, endflag, water, keys).

# LibGdx

## PlayScreen, World & JdcGame

Nous avons suivi la structure proposée par le tutoriel qui conseillait d’utiliser les classes de com.badlogic.gdx Screen, World et Game.

Screen représente notre application est sert de « main ». Elle crée les objets world et game ainsi que viewport qui est la fenêtre qui apparait à l’écran et gamecam qui est la position logique de ce qui apparaitra dans le viewport.

World est contient les objets qui ont une apparence dans le jeu ou une interaction avec (comme des senseurs). Il ne faut pas le confondre avec la gamecam qui est juste la proportion du monde qu’on affiche à l’écran. Chaque corps qui est ajouté dans le monde doit posséder un Body contient des coordonnées formant un Polygon ainsi qu’un type (StaticBody ou DynamicBody, KinematicBody n’a pas été utilisé).

Game qu’on a appelé JdcGame pour Joel-Daniel-Charles-Game contient les paramètres du jeu et ces ressources. On y trouve les constantes de proportions, les sons ainsi que les bits de mask.

## Sprite

Un Sprite est une image contenant toutes les différents frames d’une animation.

C:\Joel_Vaucher\INF2-b\HesETE-p2-Vaucher\hesETE-jdcGame\core\assets\sprites\Run_sprite.png

Figure 6 - Exemple de Sprites

LibGdx possède une classe Sprite afin de simplifier leurs intégrations au jeu.

## InteractiveTileObject

Parmi les objets qu’on a créé dans le monde, certain devront effectuer certaines actions au contact du joueur (mort, bonus).

Le tutoriel proposait une classe abstraite laissant une méthode virtuelle pour que les différentes spécifications fassent eux-mêmes les conséquences d’une collision au joueur.

Nous avons fait deux classes abstraites Enemy et InteractiveTileObject qui fonctionne de la même manière et pour les mêmes raisons, mais dont la famille d’objet est différente et dont les extensions futures pourraient être problématique s’il n’y avait pas deux classes.

## Collision & Mask

Une gestion des collisions assez courante dans les IDE de 2D/3D est la gestion par calque. Cela signifie qu’on définit que certains objets font partie d’un groupe (ou plusieurs) et qu’il ne peut entrer en collision qu’avec seulement certains autres groupes.

## Son

## Changement de Screen

Ce projet est principalement composé de screens. Chaque menu est un screen ainsi que l’interface de jeu. Afin de manager les screens, une classe ScreenManager a été créée.

### ScreenManager

La classe ScreenManager permet de contrôler le changement de screen. Cette classe utilise le patron Singleton afin de garantir qu’il n’existe qu’une seule instance de celle-ci et ainsi éviter tous problèmes liés au changement de screen (deux screens superposés, par exemple).

Afin de rendre la réalisation et la visualisation du code, cette classe utilise une énumération qui s’occupe de retourner le bon screen en fonction d’un nom.

#### ScreenEnum

Cette énumération permet de récupérer le screen souhaité grâce à un nom. Cette énumération possède deux méthodes : getScreen() et getPlayScreen(). La première, abstraite, permet de récupérer n’importe quel screen (chaque élément de l’énumération définira le screen à retourner). La deuxième (concrète) permet quant à elle de récupérer l’interface de jeu. Celle-ci se différencie de l’autre méthode par le fait qu’elle reçoit un paramètre en plus représentant le niveau du jeu.

Voici un petit exemple de déclaration d’un élément :

public enum ScreenEnum **{**

MAIN\_MENU **{**

@Override

public Screen getScreen**(**JdcGame game**)** **{**

**return** **new** MainMenuScreen**(**game**);**

**}**

**};**

//... }

#### showScreen()

Cette méthode permet d’afficher le screen souhaité. Elle reçoit comme paramètre l’élément de l’énumération souhaité. Cette fonction s’occupe d’initialisé le nouveau screen et de détruire l’ancient si nécessaire.

#### showPlayScreen()

Comme pour l’énumération, il existe une fonction dédier pour l’interface de jeu. Cette fonction reçoit comme deuxième paramètre le nom du fichier contenant le niveau.

Voici un exemple de code pour changer de screen :

// Appel du screen LEVEL\_MENU

ScreenManager**.**getInstance**().**showScreen**(**ScreenEnum**.**LEVEL\_MENU**);**

# Problèmes rencontrés et solutions

Ce chapitre présente quelques problèmes rencontrés et leurs solutions.

# Améliorations possibles

Ce chapitre présente quelques améliorations possibles du projet.

# Conclusion

Neuchâtel, le 9 septembre 2016

Vaucher Joël

Ombang Ndo Charles

Rodrigues Lourenço Daniel

# Bibliographie

**Aureli, Brent. 2015.** LibGDX - Super Mario Bros. *Youtube.* [En ligne] 22 10 2015. https://www.youtube.com/playlist?list=PLZm85UZQLd2SXQzsF-a0-pPF6IWDDdrXt.

**libgdx.badlogicgames. 2013.** Getting started. *libGDX.* [En ligne] 2013. http://libgdx.badlogicgames.com/documentation.html.