

Hitty Knife Redux

Projet de programmation

ALVES VIEIRA Lionel | COULOMBEL Corentin | FACCINI BANDEIRA Guilherme | LES BIENS Cédric | LOTHSAVAN Hugo

# Sommaire :

## [Introduction](#_Introduction_:)

## [description du jeu](#_Description_du_jeu)

## [mÉthodologie de dÉvelopement](#_Méthodologie_de_développement)

## Évolution du jeu et valeur ajoutÉe

## conception et architecture du jeu

## points algorithmiques et techniques

## [conclusion](#_Conclusion_:)

## [annexe](#_Annexes_:)

# Introduction :

*"Hitty Knife Redux"* est une réinterprétation du jeu *"Hitty Knife"*, dans lequel les joueurs doivent détruire des cibles mobiles en maîtrisant l’art du lancer de couteau.  
Initié par M. Yan Jurski, ce projet offre une expérience de jeu immersive et divertissante, tout en offrant des possibilités infinies de variations et d'améliorations.

Notre objectif initial est séparé en deux parties distinctes : une première partie où l’on devait créer des bases solides, proches de la description initiale du jeu et une seconde pour apporter une plus-value au jeu, des ajouts intéressants et pertinents, tout en faisant certaines abstractions.

Le jeu repose sur des mécanismes simples et addictifs où les joueurs doivent ajuster leur lancer pour atteindre des cibles, tout en affrontant des Boss redoutables.

Cette version repensée propose une expérience de jeu enrichie, avec des vagues de défis de plus en plus ardus, comprenant des Boss uniques dans leur conception et des environnements variés.

De plus, les joueurs peuvent choisir parmi une sélection de skins de couteaux pour personnaliser leur expérience de jeu, et s'affronter dans un tout nouveau mode versus !

Dans ce rapport, nous explorerons en détail les différentes phases du développement du projet *"Hitty Knife Redux"*,en commençant pas une description approfondie du jeu initial. Nous discuterons ensuite de la méthodologie de développement utilisée. En outre, nous présenterons les différentes variations et améliorations implémentées pour enrichir l'expérience de jeu, ainsi que les choix de conception et d’architecture qui ont guidé le développement.

Enfin, nous dresserons un bilan du projet et proposerons des pistes d’améliorations pour l’avenir. Bonne lecture !

# Description du jeu :

Le jeu *"Hitty Knife Redux"* propose une expérience amusante où les joueurs incarnent des lanceurs de couteaux talentueux, chargés de détruire des cibles mobiles avec précision. Inspiré du jeu original *"Hitty Knife"*, cette version revisitée propose de nouvelles fonctionnalités et améliorations pour enrichir l'expérience de jeu.

Dans ce jeu, les joueurs sont confrontés à des cibles flottantes en mouvement, représentées sous forme de cibles circulaires de différentes tailles. Ils doivent ajuster leur lancer de couteau pour atteindre ces cibles tout en évitant les obstacles et en tenant compte de la trajectoire balistique de leurs lancers. Chaque cible atteinte rapporte des points, tandis que les échecs réduisent le nombre de vies du joueur. Arrivé à trois lancers loupés, le joueur doit recommencer le niveau.

Les niveaux sont quant à eux divisés en 3 à 5 vagues : 3 vagues avec des cibles « normales », de plus en plus difficiles (plus de cibles, mouvements plus rapides, etc.), tandis que pour les niveaux à 5 vagues, 4 sont composées de cibles normales mais la dernière vague est un boss. Les boss, contrairement aux cibles usuelles, sont dotés de trajectoires complexes mais régulières, différentes pour chacun des boss et rendant la visée plus difficile. Certains d’entre eux ont des fonctionnalités spécifiques que nous aborderons plus tard dans ce rapport. Ces confrontations ajoutent un défi supplémentaire au jeu, testant les compétences de précision et de timing du joueur.

Un des atouts majeurs de *"Hitty Knife Redux"* est l'introduction de différents univers et environnements. La direction artistique a été entièrement repensée et les joueurs peuvent désormais choisir parmi une variété de décors et d'ambiances, allant des paysages urbains aux jungles exotiques, ajoutant ainsi de la diversité visuelle et thématique au jeu.

En outre, les joueurs peuvent personnaliser leur expérience de jeu en sélectionnant des skins de couteaux uniques, leur permettant d'exprimer leur style et leur personnalité. Tous ces ajouts cosmétiques sont accessibles par un « shop » implémenté dans le jeu, permettant au joueur de dépenser de la monnaie virtuelle pour acheter des environnements, des skins de couteau, mais aussi différentes musiques ! L'argent est accessible simplement en jouant, chaque niveau remporté rapportant au joueur un certain montant d'argent.

Enfin, un tout nouveau mode versus permet aux joueurs de s'affronter dans des duels palpitants, testant ainsi leurs compétences contre celles de leurs adversaires. Les joueurs doivent vaincre les vagues de cibles et les boss avant leur adversaire. Pour rendre ce mode plus amusant, des cibles dites « bonus » et « malus » sont déposées sur chacun des écrans de jeu ! En les touchant, les joueurs peuvent recevoir un avantage comme un ralentissement de la trajectoire du couteau, ou bien infliger un malus à leur adversaire comme une immobilisation de quelques secondes !

# Méthodologie de développement :

Le développement du jeu *"Hitty Knife Redux"* s'est déroulé selon une méthodologie AGILE, avec une forte collaboration et flexibilité pour s'adapter aux évolutions du projet. Nous avons utilisé GitLab comme système de contrôle de versions, ce qui nous a permis de suivre efficacement l'avancement des tâches, de gérer les problèmes et de collaborer sur le code source.

Notre approche AGILE s'est articulée autour de sprints hebdomadaires. Au début de chaque sprint, une réunion était organisée où chaque membre de l'équipe présentait ce qu'il avait accompli lors du sprint précédent et discutait des tâches à réaliser pour le sprint en cours. Cette réunion était également l'occasion de planifier les objectifs et les priorités pour le sprint à venir, ainsi que d’échanger avec le Chargé de Projet, M. Jurski, dont l’écoute, l’aide et les conseils ont été d’une importance cruciale.

L'utilisation d'issues sur GitLab a permis à chaque membre de l'équipe de proposer des tâches et de se les assigner ou les laisser à d’autres membres en fonction de leur disponibilité et de leurs compétences. Cette approche a favorisé une répartition équilibrée du travail et une collaboration fluide au sein de l'équipe. De plus, la flexibilité offerte par cette méthode a permis d'ajuster les priorités et les objectifs en fonction des besoins du projet.

Les 6 premiers sprints ont été consacrés à l'établissement d'une base solide pour le jeu et la direction artistique, en mettant en place les fonctionnalités de base et en stabilisant le code. Les 6 sprints suivants ont été consacrés à l'ajout de fonctionnalités avancées et à la perfection du jeu. Cela inclut la conception et l'implémentation de nouveaux boss uniques, une refonte complète du GUI –dont on parlera plus en détails dans la partie « Évolution du jeu et valeur ajoutée »  -, une perfection du code source (refactorisation, JavaDoc…) et l’ajout du mode Versus.

En résumé, notre méthodologie de développement agile et notre utilisation de GitLab nous ont permis de travailler de manière collaborative, flexible et efficace pour donner vie au jeu !

# Évolution du jeu et valeur ajoutée :

* Explication des variantes et des idées d'amélioration proposées dans la fiche de projet.
* Développement des nouveaux univers imaginés pour enrichir le jeu.
* Discussion sur l'importance des abstractions et de la conception pour permettre ces variations

# Conception et Architecture du jeu :

Nous avons utilisé Maven pour gérer les dépendances, la compilation ainsi que l’exécution du projet. Doté d’une facilité d’utilisation, la configuration de Maven a accéléré le processus de développement, en particulier la sauvegarde de fichier du joueur en local.

Afin d’apporter une documentation claire et compréhensible des fonctionnalités et de l’architecture du code, nous avons employé JavaDoc. En effet, chaque classe, chaque fonction/méthode importante a été documenté de sorte à ce qu’une personne externe comprenne parfaitement l’utilité de celle-ci, comme ce qui peut nous être demandé en entreprise.

Contenant toutes les descriptions de classes, de fonctions importantes.

* Diagramme architectural présentant les grandes lignes de la structure du jeu.
* Explication des choix architecturaux et des abstractions utilisées.
* Discussion sur la refactorisation et l'amélioration du code au fil du développement.

# Points algorithmiques et techniques :

## Sauvegardes :

La gestion des sauvegardes dans *"Hitty Knife Redux"* est un élément crucial pour offrir aux joueurs une expérience fluide et personnalisée. Nous avons mis en place un système de sauvegarde efficace, basé sur une structure de données sérialisée, permettant de conserver l'état du jeu pour chaque utilisateur.

### Implémentation :

L'implémentation des sauvegardes repose sur l'utilisation de fichiers JSON pour stocker les informations des utilisateurs et leur progression dans le jeu. Nous avons développé une classe User qui représente les utilisateurs, avec des attributs tels que le nom d'utilisateur, le mot de passe, le chemin de sauvegarde et le montant d'argent. Cette classe est sérialisée dans un fichier JSON pour chaque utilisateur.

La classe *UserManager* quant à elle joue un rôle central dans la gestion des utilisateurs et de leurs données dans *"Hitty Knife Redux"*. Elle simule une base de données en chargeant toujours la même instance de *UserManager* au lancement du jeu.

#### Simulation de base de données

Le *UserManager* est implémenté selon le modèle Singleton, assurant qu'une seule instance de la classe est utilisée dans toute l'application. Cela permet de centraliser la gestion des utilisateurs et de maintenir la cohérence des données.

La simulation de base de données est réalisée en chargeant les informations des utilisateurs à partir d'un fichier JSON lors du démarrage de l'application. Les utilisateurs sont stockés dans une structure de données HashMap, où chaque utilisateur est associé à son nom d'utilisateur. Cela permet un accès rapide et efficace aux données utilisateur tout au long de l'exécution du jeu.

La classe UserManager gère la création, la connexion et la sauvegarde des utilisateurs. Elle utilise un algorithme de hachage SHA-256 pour sécuriser les mots de passe stockés.

Lorsqu'un utilisateur se connecte, le système charge son état de jeu à partir du fichier JSON correspondant à son nom d'utilisateur.

#### Gestion des sauvegardes

Lorsque l'utilisateur effectue des actions importantes dans le jeu, telles que la progression de niveau ou l'achat d'éléments cosmétiques, l'état du jeu est sauvegardé automatiquement.

La classe Game gère quant à elle la sauvegarde de l'état du jeu.   
Cette approche garantit que les données des utilisateurs sont persistantes et accessibles à chaque exécution du jeu

#### Gestion des mouvements / géométrie

Tous les mouvements dans le jeu sont gérés à partir de la classe Geometry. Cette dernière comporte plusieurs fonctions décrivant différents types de mouvements (horizontaux, verticaux, etc.), également des mouvements soumis à la gravité, comme le saut du couteau.

La classe Geometry utilise deux autres classes, la classe Coordinate qui décrit un tuple de double x, y correspondant aux coordonnées théoriques d’un objet, et la classe Loop qui est la boucle principale du jeu et qui actualise le jeu.

Ce qui est important pour Geometry dans la classe Loop c’est la valeur delta qui permet de calculer avec précision les nouveaux emplacements des objets en mouvement, en tenant compte du temps écoulé entre les mises à jour. Cela permet de créer des simulations plus réalistes et fluides dans des environnements de jeu ou des applications impliquant des objets géométriques en mouvement.

#### Affichage théorique et réel

Deux types de coordonnée sont distinct dans le jeu, les coordonnée théorique et réel. En effet, les coordonnées théoriques sont utilisées pour faire interagir les différents objets géométriques entre eux, tandis que les coordonnées réelles prennent en compte la taille de l’écran et gardent le même ratio entre les différents éléments du jeu.

Cela a pour avantage de pouvoir choisir quelle zone du jeu « théorique » on veut afficher, et donc pouvoir faire sortir certains éléments de l’écran, une mécanique qui est d’ailleurs utilisé dans les combats de boss.

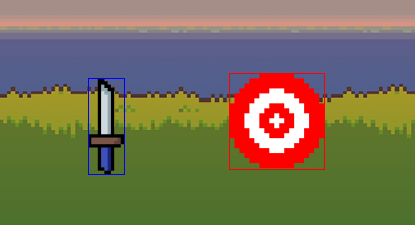
Cette façon de gérer les coordonnées est très pratique puisqu’elle permet d’afficher le jeu uniquement dans le terminal en se passant de l’affichage graphique. C’était très pratique lors du début du développement du jeu quand on n’avait pas encore d’interface graphique.

#### Gestion des collisions

La gestion des collisions se fait principalement en trois étapes :

La première étape consiste à créer un CollisionMask à partir de l’image pour générer un objet type Shape.

Pendant longtemps on a gardé une Shape de type Rectangle :



Pour ensuite partir sur des formes plus complexes en utilisant un Area. Pour cela on parcourt l’image et dès qu’on tombe sur un pixel qui n’est pas transparent, on l’ajoute à l’Area. Ce qui donne une hitbox plus détaillé :

Enfin on utilise la méthode intersects de java.awt.Shape qui retourne True si les deux Shape passé en paramètre sont en collision.

* Mise en avant des aspects algorithmiques importants du jeu.
* Discussion sur les techniques de programmation utilisées pour implémenter les fonctionnalités.

# Conclusion :

Le développement de « *Hitty Knife Redux* » a été une aventure riche en défis et en apprentissages, tant dans le plan humain que dans le plan académique. Ce projet a non seulement permis de créer un jeu en partant de rien - ce qui nous a amené à vite tenir à cœur à ce projet - mais il a également été l'occasion de développer et de concrétiser les compétences acquises depuis le début de notre cursus.

Les difficultés ont quant à elle su nous rassurer dans nos compétences, tant en Java que dans le travail d’équipe, car aucun problème n’a pas pu être résolu.

Durant ce projet, toutes les fonctionnalités clé ont été implémentées, reflétant le respect du cahier des charges, tout en intégrant d’autres éléments. Jamais « *Hitty Knife Redux* » n’a cessé de se réinventer. Les défis techniques et de conception rencontrés ont été, bien que parfois éprouvants, de précieuses opportunités pour apprendre et s'adapter, contribuant ainsi à un constant affinage de nos capacités.

L'exploration des pistes d'extensions futures pour « Heroes and Defenders » ouvre la voie à de nouvelles améliorations et à une envie partagée de continuer ce projet tout au long de notre licence.

[PISTES D’AMÉLIORATIONS]

Enfin, toute l’équipe souhaite remercier M. Yan Jurski pour son accompagnement et ses conseils tout au long du développement de ce projet, mais aussi vous Mesdames et Messieurs les Jurys pour l’attention portée à notre projet et notre rapport.

# Annexes :

Vous trouverez dans cette partie toutes les sources qui ont aidées au développement de Hitty Knife Redux ainsi que les documentations des outils utilisés par nous-mêmes :

* [Documentation de Jackson (JSON)](https://github.com/FasterXML/jackson-docs)
* [Tutoriel des basiques de la programmation de Jeu Vidéos en Java](https://zetcode.com/javagames/basics/)