Documentation ChaDu Arena

Plan

1. Contexte
2. Description
3. Les patrons rencontrés
4. Diagramme de cas d'utilisation
5. Diagramme UML

1. Contexte

ChaDu Arena est un jeu de survie dans une arène ayant une taille limitée. Le joueur devra survivre à des vagues de monstres (chatons) de différents types et il pourra se défendre en leur jetant des pics. La vie du joueur, le numéro de la vague ainsi que la durée survécu depuis le début en secondes sont affichés à l'écran. Le but est de survivre le plus de temps possible.

Une fois mort, le joueur a la possibilité d'enregistrer son score s'il le souhaite. Tous les scores triés du meilleurs au moins bon sont sauvegardés et peuvent être vus par le joueur. La légende raconte que lorsqu'il y a dans le classement un score avec un pseudo comportant le nom de famille d'un fameux tueur de chatons de l'IUT, une mystérieuse image apparaît lorsque le classement est affiché.

2. Description

Ce projet contient :
- une documentation en pdf comportant les différents diagrammes (ce fichier)
- une Javadoc générée à partir du projet
- un fichier contenant le projet netbeans
- un fichier txt contenant les preuves des compétences (dont certaines renvoient ici)

Ce projet a été réalisé dans le but de découvrir JavaFX et les outils / possibilités qu'il donne. Il a également été réalisé lors de l'apprentissage de principes de qualités. C'est pour cela qu'il a été réalisé dans l'optique de se rapprocher le plus possible des principes S.O.L. de SOLID.

3. Les patrons rencontrés

Lors de la création de ce projet, nous avons utilisé, consciemment ou non, des patrons de conception.

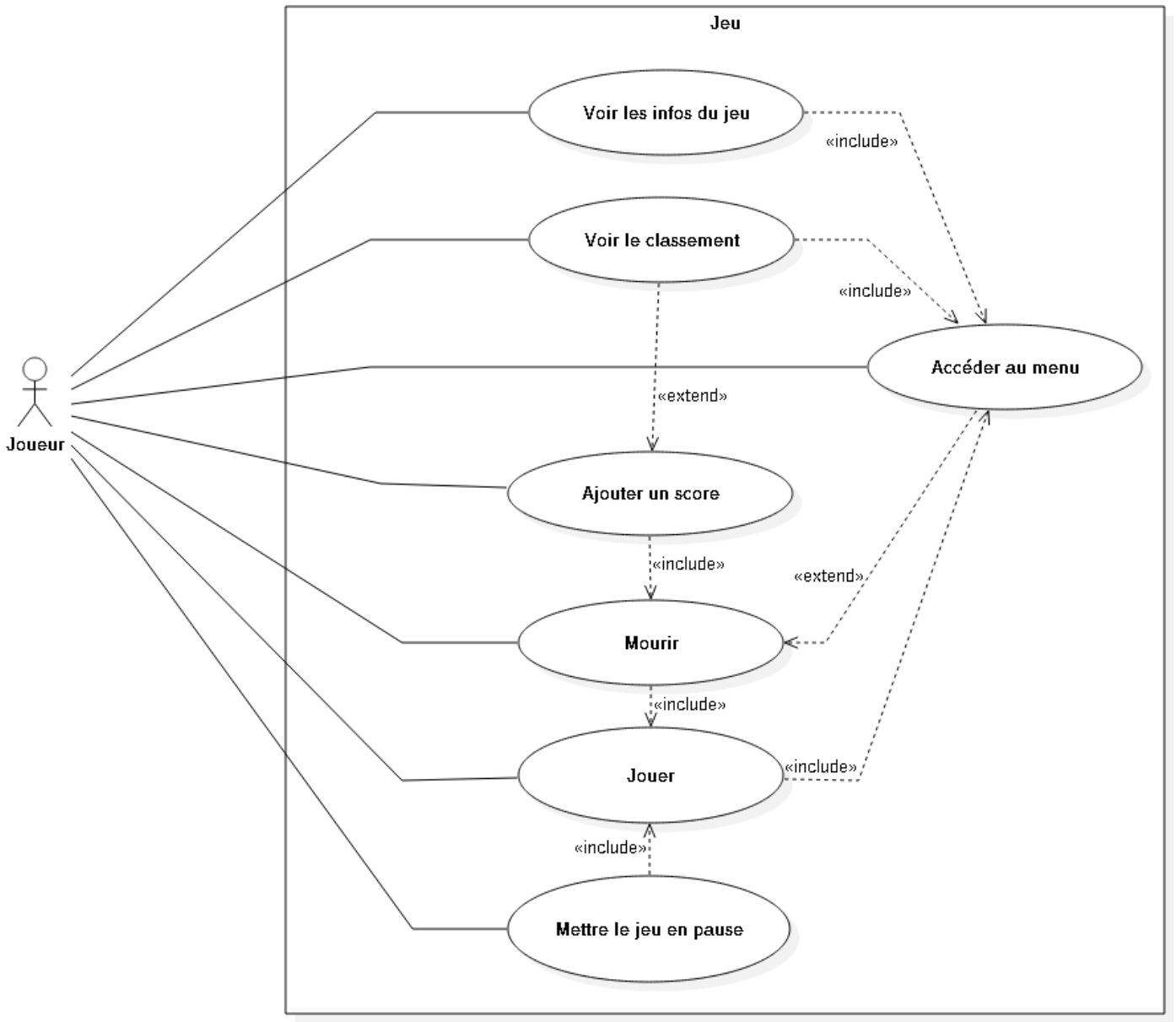
Premièrement, on a utilisé le **pattern stratégie** pour la partie persistance qui permet de sauvegarder et charger la liste des scores. Ce pattern permet de switcher d'une technique à l'autre (binaire ou XML) et permet d'implémenter d'autres façons de gérer la persistance si on le souhaite.

Ensuite, on a pu rencontrer le **patron fabrique** pour créer des entités & des projectiles. Il est utile car il permet si on le souhaite de changer de façon de les créer. L'idée générale de la fabrique abstraite (et donc de la concrète) est de stocker les infos permettant de fabriquer des ensembles différents en fonction notamment de la valeur de la vague indiquée pour la création des entités monstres.

On a également décidé de reprendre l'idée du patron ***poids mouche***, bien que l'on ne l'a pas utilisé en lui-même, car on utilise les mêmes images pour créer plusieurs entités d'un même type. Cela évite d'en créer inutilement car une par type de classe suffit.

On a également un **patron Façade** avec la classe Jeu qui permet de gérer l'ensemble du jeu avec des sous systèmes.

4. Diagramme de cas d'utilisation



5. Diagramme UML