# **Jalon 3**

## **Rapport de développement/test :**

* **Ce qui a été développé :**

La partie IHM n’est pas terminée, le jeu ne tourne qu’en console. En mode graphique, on peut quand même lancer l’application : cela ouvre le menu principal où l’on peut jouer, voir les scores, les règles, les options et quitter. Lorsque l’on joue, cela affiche une fenêtre où l’on peut choisir le niveau de difficulté et le pseudo. Ensuite on peut cliquer sur une tour (en bas) puis sur la case où l’on veut la placer pour la poser. Elle s’affiche donc sur la carte. On peut ensuite lancer une vague, mais dès que l'un des ennemis meurt ou arrive a passer, il y a une erreur. Il manque donc la fin de l’implémentation du jeu en lui-même. De plus, nous aurions voulu créer un autre package pour mettre notre IHM (et en faire une bonne gestion avec les interfaces), mais nous n'en avions pas compris le fonctionnement au moment où nous avons commencé à la créer. En console, le jeu fonctionne mais il y a des problèmes dans l’algorithme de chemin.

* **Tests unitaires :**

Explication de la méthode lancer un tour

On répète les opérations qui suivent jusqu'à ce que le joueur ai des vies ou que le tour soit fini. Un tour est fini s'il n'y a plus d'ennemi.

A chaque début de boucle on teste s'il y a des ennemis sur la carte et s'il reste des ennemis dans la vague, si les deux listes sont vides, le tour est terminé.

Sinon à chaque tour pair on ajoute un ennemi sur la carte, on teste si les tours peuvent attaquer un ennemi. Par la suite on teste si l'ennemi est sur la case de fin (si c'est le cas il est retiré de la liste d'ennemi présent sur la carte). Enfin on met à jour la carte, on fait avancer les ennemis présents sur la carte et on incrémente le nombre de tours de 1.

**Classe JUnit**: *EnnemiTest*

Explication de la méthode trouver case adjacentes

On prend une case de la carte et on lui demande une liste des cases adjacentes avec une portée. Cependant cette liste ne doit pas comporter de cases qui ne sont pas sur la carte. Par exemple la case en bas à droite ne doit pas renvoyer, avec une portée de 3, des cases inexistantes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Tour en rouge avec une portée de 1.

Création de tour

On crée des tours sur des cases, mais on ne peut créer une tour que sur une case vide et si on a suffisamment d’argent.

|  |  |  |
| --- | --- | --- |
| Donnée | résultat | Résultat attendu |
| Etat case : vide  Argent : supérieur au prix de la tour | Création de la tour | Création de la tour |
| Etat case : Chemin Argent : supérieur au prix de la tour | Soulève une exception CasePleinException | Soulève une exception CasePleinException |
| Etat case : vide Argent : inférieur au prix de la tour | Soulève une exception ArgentInsufisant | Soulève une exception ArgentInsufisant |

Afficher Menu (Tower Defense)

|  |  |  |
| --- | --- | --- |
| Donnée | résultat | Résultat attendu |
| 1 | Lancement d’une partie | Lancement d’une partie |
| 2 | Affichage des règles | Affichage des règles |
| 3 | Affichage des options | Affichage des options |
| 4 | Affichage des scores | Affichage des scores |
| 5 | Quitter | Quitter |
| « autre caractère ou suite de caractères « | « message d’erreur et question reposée » | « message d’erreur et question reposée » |

Scores (Table Score)

Si aucun fichier n'existe, on créer un fichier avec seulement les en-têtes de colonne. Ok

J'ai tester l'ajout dans la table des scores avec les cas suivants :

Table score vide. Ok

Table score non vide, avec score plus élevé, moins élevé, et au milieu de score déjà existant. Ok

Table score plein, avec score plus élevé, moins élevé, et au milieu de score déjà existant. Ok

* **Bilan personnel/technique :**

L’équipe à bien fonctionné, nous avons travaillé tous ensemble aux plages horaires indiquées sur le planning. L’ambiance dans le groupe était bonne, ce qui nous a permis de travailler de manière coordonnée. Les difficultés techniques rencontrées ont été des difficultés liées au codage et à la conception (quelles classes utiliser…). Ce projet nous a apporté de l’expérience supplémentaire concernant le respect de contraintes telles qu’une date limite, le travail en équipe ou la gestion de projet (partie économique). Nous avons aussi dû apprendre à gérer Github.

**Lien GitHub**:[*https://github.com/lucasmd/iutvalence-projetS2-2013-2014-groupe10*](https://github.com/lucasmd/iutvalence-projetS2-2013-2014-groupe10)

*Planning du 7/04 au 30/05*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Membres/Jours* | *Jeudi 8/04* | *Jeudi 15/04* | *Jeudi 22/04* | *Jeudi 15/05* | *Jeudi 22/05* |
| *Héloïse* | *13h - 14h30* | *13h - 14h30* | *12h30 - 14h30* | *12h30 - 15h* | *13h - 17h30* |
| ***Maxime*** | *13h - 14h30* | *13h - 14h30* | *12h30 - 14h30* | *12h30 - 15h* | *13h - 17h30* |
| ***Nathan*** | *13h - 14h30* | *13h - 14h30* | *12h30 - 14h30* | *12h30 - 15h* | *13h - 17h30* |
| ***Lucas*** | *13h - 14h30* | *13h - 14h30* | *13h - 14h30* | *12h30 - 15h* | *13h - 17h30* |
| *Alexandre* | *13h - 14h30* | *13h - 14h30* | *12h30 - 14h30* | *12h30 - 15h* | *13h - 17h30* |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Membres/Jours* | *Vendredi 23/05* | *Lundi 26/05* | *Mardi 27/04* | *Mercredi 28/05* |
| *Héloïse* | *13h30 - 18h* | *15h30 - 18h30* | *13h30 - 17h30* | *14h30 - 17h30* |
| *Maxime* | *13h30 - 18h* | *15h30 - 18h30* | *13h30 - 17h30* | *14h30 - 17h30* |
| *Nathan* | *13h30 - 18h* | *15h30 - 18h30* | *13h30 - 17h30* | *14h30 - 17h30* |
| *Lucas* | *13h30 - 18h* | *15h30 - 18h30* | *13h30 - 17h30* | *14h30 - 17h30* |
| *Alexandre* | *13h30 - 18h* | *15h30 - 18h30* | *13h30 - 17h30* | *14h30 - 17h30* |