Rapport TP2
INFO - OBJET - 2021/2022
Bodet Alexandre (Rémondeau Paul)

Objectif du TP : Comprendre les diagrammes de classe UML. Utiliser la génération de nombre pseudo-aléatoires ainsi que créer différentes classes en utilisant l'héritage et la création de méthodes de bases automatiquement.

1ere partie : création d'un projet JAVA

On recopie les deux classes du TP précédent puis on run project avec TestPoint2D. On obtient les résultats attendus du TP précédent. Les coordonnées sont affichées comme voulu et modifiées quand et comme on le souhaitait avec les méthodes set et translates.

```
----------------------< edu.centralenantes:Projet >----------------------
Building Projet 0.1
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ Projet ---
[0;0]
[5;10]
[5;10]

Test de setX :
Avant:
[0;0]
Après:
[10;0]

Test de setY :
Avant:
[10;0]
Après:
[10;10]

Test de getX :
10

Test de getY :
10

Test de setPosition :
Avant:
[10;10]
Après:
[0;0]

Test de translate :
Avant:
[0;0]
Après:
[5;5]
------------------------------------------------------------------------
BUILD SUCCESS
```

2eme partie : compréhension d'un diagramme de classe UML

Description des différentes classes :

- TestSeance1 : classe pour effectuer les tests
- Point2D : décrite précédemment dans le sujet, point à coordonnées entières
- Personnage : personnage du jeu. Il possède un nom, des points de vie, du mana, différentes statistiques pour le combat, et une position
- Archer : sous-classe de personnage qui peut tirer des flèches
- Paysan : sous-classe de personnage
- Monstre : monstre présent dans le jeu qui a des points de vie, des statistiques pour le combat et une position
- Lapin : sous-classe de monstre
- World : décrite précédemment, représente le monde du jeu

Paysan et Archer sont des sous-classes de Personnage car ils ont des attributs communs mis à part quelques différences. On les sépare car ils ont des méthodes différentes.
Les monstres n'ont pas de mana et certains autres attributs. Ils ne sont pas assimilés aux personnages, on crée une classe monstre pour ça. Les types de monstres seront tous des sous-classes.

On pourrait définir Personnage comme une sous-classe de Monstre puisqu'elle possède tous les attributs de Monstre. En revanche on pourrait être tenté d'ajouter certaines choses à Monstre et pas Personnage par la suite. On pourrait par contre créer une super-classe Créature de Personnage et de Monstre.

Pas besoin de supprimer la classe Point2D, elle fonctionne bien. On va en revanche enlever la classe TestPoint2D qui contient une méthode main pour tester Point2D, on va utiliser une seule classe main à la fois, ici on utilise celle de TestSeance1.

Pour simplifier, on redéfinit la méthod toString de Point2D pour rendre son utilisation plus propre dans la fonction affiche() de personnage.

Les captures d'écrans des tests montrant le bon fonctionnement de la fonction principale se trouvent dans les deux pages suivantes.

On a bien défini l'archer, le paysan et les lapins avec les attributs qu'on souhaitait. La fonction creeMondeAlea() permet de bien créer les 2 personnages et 2 monstres sur des positions différentes, aléatoires et éloignées de 5 au maximum entre deux personnages ou monstres successifs.

La fonction affiche() nous permet de voir tous les attributs d'un Personnage ou d'un Monstre. On peut utiliser les setters et getters comme montré pour les flèches de robin. On ne teste pas toutes les fonctions car c'est un peu répétitif.

La fonction deplace() fonctionne bien également car la position du lapin est modifiée.

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ ProjetTP ---
Le personnage s'appelle Robin et est situé à la position [81;17].
Robin a 10 points de vie.
Robin a 0 points de mana.
Robin a 10 points de dégats physique.
Robin a 0 points de dégats magique.
Robin a une portée de 20.
Robin a un pourcentage d'attaque de 20%.
Robin a 5% de chance de parer.
Robin a un pourcentage d'attaque magique de 0%.
Robin a 5% de résistance magique.
Robin a 32 flèches.


Le personnage s'appelle Peon et est situé à la position [79;15].
Peon a 5 points de vie.
Peon a 0 points de mana.
Peon a 5 points de dégats physique.
Peon a 0 points de dégats magique.
Peon a une portée de 5.
Peon a un pourcentage d'attaque de 10%.
Peon a 5% de chance de parer.
Peon a un pourcentage d'attaque magique de 0%.
Peon a 5% de résistance magique.


Le monstre a 1 points de vie et se situe en [81;11].
Le monstre a un pourcentage d'attaque de 1%.
Le monstre a un pourcentage de parade de 1%.
Le monstre a 1 points d'attaque.


Le monstre a 1 points de vie et se situe en [79;14].
Le monstre a un pourcentage d'attaque de 1%.
Le monstre a un pourcentage de parade de 1%.
Le monstre a 1 points d'attaque.
```

```
Robin a utilisé 2 flèches.
Le personnage s'appelle Robin et est situé à la position [81;17].
Robin a 10 points de vie.
Robin a 0 points de mana.
Robin a 10 points de dégats physique.
Robin a 0 points de dégats magique.
Robin a une portée de 20.
Robin a un pourcentage d'attaque de 20%.
Robin a 5% de chance de parer.
Robin a un pourcentage d'attaque magique de 0%.
Robin a 5% de résistance magique.
Robin a 30 flèches.


le monstre s'est déplacé aléatoirement pour fuir robin.
Le monstre a 1 points de vie et se situe en [59;72].
Le monstre a un pourcentage d'attaque de 1%.
Le monstre a un pourcentage de parade de 1%.
Le monstre a 1 points d'attaque.


------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  2.016 s
Finished at: 2021-09-14T09:10:06+02:00
------------------------------------------------------------------------
```

Conclusion :

Le diagramme de classe UML est pratique pour comprendre les classes et leurs relations, ainsi que leurs attributs et méthodes.

Utiliser la création automatique de constructeurs, setters et getters fait gagner du temps mais il faut faire attention aux attributs et à si on veut renvoyer un objet ou la référence dans le cas d'un objet.

L'utilisation du générateur de nombre aléatoire a aussi été effectuée sans trop de problème. On a pu générer des positions aléatoires pour les Personnage et Monstre avec succès.