63-21 Approfondissement de la programmation
Ramiqi Andi

h e g | Haute école de gestion de Genève

# TP09

**Gestion des Personnages de Jeu Vidéo**

Vous travaillez sur un jeu vidéo où différents types de personnages peuvent effectuer diverses actions basiques comme attaquer, défendre ainsi que des actions spéciales comme utiliser des compétences spéciales. Vous devez développer une application permettant de gérer ces actions en utilisant des interfaces pour regrouper les fonctionnalités communes.

Chaque personnage possède un nom, un niveau, des points de vie (PV), des points de mana (PM), des points d'attaque (PA) et des points de défense (PD). Les personnages peuvent être de différents types :

1. Guerrier : Un guerrier peut attaquer et défendre. Sa compétence spéciale double sa défense. En plus de cela, il possède un attribut de plus que les autres classes : de la **résistance** supplémentaire lorsqu'il se défend
2. Mage : Un mage peut attaquer, défendre et utiliser une compétence spéciale (sort). Le mage a de la **puissanceMagique** qui s'ajoute à chacune de ses attaques.
3. Soigneur : Un soigneur peut attaquer faiblement, défendre et utiliser une compétence spéciale (soin). Le soigneur a un passif **recupPV** lui permettant de récupérer naturellement des points de vie après chaque action.

Les personnages sont actuellement uniquement stockés dans un fichier CSV, contenant les informations suivantes :

- Type de personnage : la lettre G, M ou S indique si le personnage est un Guerrier, un Mage ou un Soigneur
- Nom du personnage (String).
- Niveau du personnage (int).
- Points de vie (int).
- Points de mana (int)
- Points d'attaque (int)
- Points de défense (int)
- Le dernier attribut est propre à chaque classe (**resistance**, **puissanceMagique**, **recupPV**)

Avant de charger les données, structurez correctement ce projet en utilisant des interfaces, des classes abstraites et des classes concrètes.
En cas d'erreur de type (autre lettre que G, M ou S), il faut pouvoir relever l'erreur à l'aide d'un affichage, mais le reste des données doit tout de même être chargé.

**Fonctionnalités à implémenter :**

- Charger les données à partir du CSV et stocker les personnages dans une liste
- Permettre l'ajout d'un nouveau personnage à la liste des combattants.
- Afficher la liste de tous les combattants avec leurs détails.
- Lancer un combat entre deux personnages au hasard dans la liste
- Afficher le vainqueur d'un combat.

**Détails d'un combat :**

1. Initialisation du Combat : Deux personnages sont sélectionnés pour le combat.

2. Tours Alternés : Chaque personnage agit à tour de rôle.
   Le premier personnage effectue une action.
   Ensuite, c'est au tour du second personnage.

3. Actions Disponibles : (pour faire simple le choix est effectué aléatoirement, mais vous pouvez essayer d'implémenter une logique de combat…)
   Attaquer : Le personnage attaque son adversaire, infligeant des dégâts basés sur ses points d'attaque (PA).
   Défendre : Le personnage augmente ses points de défense de 1 pour réduire les dégâts des attaques reçues jusqu'à la fin du combat. (Les guerriers se voient ajouté leur **résistance** en plus)
   Utiliser une Compétence Spéciale : Si le personnage est un Mage ou un Soigneur, il peut utiliser une compétence spéciale pour doubler ses points d'attaque (le mana diminue de 10) ou soigner (augmente les points de vie de 10 pour le Soigneur, le mana diminue de 10).

4. Mise à Jour des PV : Après chaque action, les points de vie des personnages sont mis à jour. Pour la réduction des points de vie, il faut bien penser à prendre en compte la défense du personnage. Lors des échanges, les attributs propres à chaque classe (resistance, puissanceMagique, recupPV) influent aussi les échanges.

5. Vérification des PV : À la fin de chaque tour, on vérifie si un des personnages a ses PV réduits à zéro.
   Si un personnage n'a plus de PV, il est déclaré vaincu.
   Si le combat dure plus que 10 tours, il faudra afficher un message d'erreur et arrêter le combat

Réfléchissez bien aux responsabilités de classes, la méthode permettant de faire combattre deux personnages doit être allégé le plus possible, il faut ainsi coder correctement les classes du domaine ainsi que leurs différentes actions possibles.

La classe **GestionPersonnage** se charge du chargement et du stockage des données.
La classe **CombatPersonnage** prend en paramètre les deux personnages pour l'affrontement et disposera d'une méthode combat().

L'affichage doit se faire via les toStrings, la méthode combat ne sert qu'à lancer le combat et effectuer les actions tour par tour jusqu'à ce qu'un des deux personnages n'a plus de points de vie.

**Affichages demandés :**

Si un personnage n'est pas de type Guerrier, Mage ou Soigneur dans le chargement demandé :

```
------Chargement des personnages------
Le personnage Gollum n'est pas reconnu
```

Affichage de tous les personnages :

```
------Affichage des personnages------
Conan est de niveau 10, il s'agit d'un guerrier
Gandalf est de niveau 12, il s'agit d'un mage
Merlin est de niveau 8, il s'agit d'un soigneur
Thor est de niveau 15, il s'agit d'un guerrier
Saroumane est de niveau 14, il s'agit d'un mage
Medic est de niveau 7, il s'agit d'un soigneur
Achilles est de niveau 20, il s'agit d'un guerrier
Dumbledore est de niveau 18, il s'agit d'un mage
Priest est de niveau 10, il s'agit d'un soigneur
```

Affichage d'un combat (ceci est un exemple) :

```
------Lancement d'un combat aléatoire------
Combat entre les deux personnages suivants :
Priest est de niveau 10, il s'agit d'un soigneur
VS
Gandalf est de niveau 12, il s'agit d'un mage

Priest lance une attaque magique sur Gandalf
Gandalf utilise un sort puissant
Priest se protège avec un bouclier magique
Gandalf lance une attaque magique sur Priest
Priest se protège avec un bouclier magique
Gandalf lance une attaque magique sur Priest
Fin du combat
Victoire de Gandalf
```