TP6: introduction aux generics en Java

Introduction

Comme vu en cours, la notion de *generics* (génériques) en Java permet d'avoir du polymorphisme au niveau des types et donc d'écrire du code (par exemple une interface) qui s'applique à n'importe quel type (classe).

Dans ce TP, nous allons implémenter quelque chose de similaire au password generator présenté en cours, mais cette fois nous n'allons pas générer des Strings, mais des objets d'une certaine classe que nous aurons défini au préalable (classe `Item`).

Dans le contexte du jeu de rôle, ces objets aléatoires pourront être des potions magiques ou des pièces d'équipement (armes, armures) par exemple. Lorsque un joueur tue un monstre (pas encore implémenté dans ce TP) ils vont pouvoir *looter* ("piller" litteralement, "rammasser un butin") que le montre aura *droppé*, en franglais, i.e., laissé tombé. C'est une récompense pour le joueur. On peut aussi considérer que certaines quêtes donnent un *loot* lorsque un joueur les a réussi.

Vous pouvez soit reprendre le dernier TP que vous avez fait et le continuer ou alors en faire un nouveau avec juste le minimum pour implémenter ce qu'on vous demande dans le TP6 et qui sera décrit dans la section Exercices.

Exercice 1

Au lieu de l'interface `Equipment`, nous allons créer une interface `Item` plus générale (pour toutes sortes d'objets pouvant être mis dans un sac) que voici:

```
1 public interface Item {
2     public int getWeight();
3     public String getName();
4     public int getMinLvl();
5 }
```

Une potion, une arme ou une armure sont tous des `Items`[1]. Ces derniers ont un *poids* (on pourrait imaginer que ce sont des grammes par exemple, mais peu importe l'unité), un *nom* et un *niveau minimum* pour pouvoir être utilisé.

Sur la base de cette interface, créez quelque classes qui l'implémentent. On vous propose:

- HealingPotion: doit avoir un champ `HP` pour indiquer le nb de points de vie rendus
- ManaPotion: doit avoir un champ `mana` pour indiquer le nb de mana de vie rendus
- Armor: doit avoir un champ `protection`
- Weapon: doit avoir un champ `damage`

Vous pouvez soit utiliser la classe `Damage` des TPs précédents si vous voulez l'intégrer au TP précédent ou sinon simplement le type primitif `int` si vous faites un TP à part.

Pour tester vous pouvez ensuite en instancier quelqueuns comme par exemple:

```
var potion = new HealingPotion("Lesser healing potion", 10,
    5,1);
var chainMailArmor = new Armor("Chain mail armor", 75,
    5000,5);
var sword = new Weapon("Sword", 25, 900,3);
```

où le dernier objet représente une épée qui cause 25 dégats, a un poids de 900 et est utilisable à partir du niveau 3. C'est pour vous donner une idée...

Remarque: il serait plus simple de considerer les champs `mana`, `HP`, `protection` et `damage` comme des `int`, en tout cas dans un premier temps, pour tester le générateur aléatoire.

Exercice 2

Pour commencer, nous n'allons qu'implémenter un générateur aléatoire de potions. Celui-ci va se baser sur un fichier texte contenant les lignes suivantes :

```
HealingPotion, Lesser Healing Potion, 1, 10, 1
HealingPotion, Healing Potion, 11, 25, 5
HealingPotion, Major Healing Potion, 26, 50, 10
```

[1]Par la suite (mais pas dans ce TP), on peut imaginer que les armes et armures implémentent une autre interface (en plus de "Item") leur permettant d'être équipées, par exemple.

```
ManaPotion, Lesser Mana Potion, 1, 5, 1
ManaPotion, Mana Potion, 6, 10, 5
ManaPotion, Major Mana Potion, 11, 20, 10
```

Il faudra générer un numéro de ligne au hasard (entre 1 et 6), lire la ligne et générer un objet potion correspondant aux donnée de la ligne. Voici comment interpréter chaque colonne:

1. Nom de la classe de l'objet à générer (HealingPotion ou ManaPotion)
2. Nom (champ `name`) de l'instance
3. valeur minimale de mana ou de vie
4. valeur maximale de mana ou de vie
5. niveau mini pour utiliser la potion

Les colonnes 3 et 4 sont à utiliser pour générer une valeur de mana ou de vie comprise entre les min et max de la ligne lue. Vous allez ensuite faire renvoyer au générateur un objet instancié avec ces valeurs aléatoires et un poids calculé comme $\lceil \frac{x}{5} \rceil$, où x est soit la valeur de mana ou de vie. On va utiliser la fonction ceiling pour éviter d'avoir un poids nul (arrondi à 0).

Tests

Implémenter un test qui vérifie la propriété suivante:

- toutes les potions de vie ayant le mot "Lesser" dans leur nom doivent avoir une valeur de vie comprise entre 1 et 10 (comme spécifié dans le fichier)

Pour cela, il faudra générer un grand nombre de potions de vie et tester cette propriété.

Exercice bonus

Si vous avez le temps et l'envie ... essayez d'implémenter un générateur d'Items qui serait capable de générer toutes sorte d'objets y compris des armes et des armures. (Il n'est cependant pas sûr que nous ayons le temps de proposer un corrigé de cet exercice d'ici la semaine prochaine).

Rendu

Ce TP n'est pas noté. Néanmoins, nous vous demandons de le déposer sur Moodle via le widget prévu à cet effet. Archivez votre projet tout

entier dans **un seul** fichier .zip avec la convention de nommage suivante: **prenomNomTP6.zip**.