

Conception Logicielle - TD 3

† † †

Jeu des personnages – V2.0 : Patrons de conception

Once we have tests in place, we can now, in a more serene way, evolve our requirements document for the characters.

I - Les patterns de comportement

Let's start by looking at the concept of *troupe*.

Exercice I.1

Troupes... En marche !

With the observer pattern, we want to model the concept of troupe: more precisely we want to regroup the ensemble of characters and elements that can move (i.e. the worker and the warrior who walk, the dragon who flies and the catapult that rolls if it is linked to a warrior).

1. Modify your UML class diagram to make the new concept appear.

Then reflect:

- possibility to make a troupe movable
- and therefore possibility to manipulate a troupe within another troupe...

2. Implement these evolutions in your Java project

3. Test.

Exercice I.2

Évolution... des personnages et de leurs comportements

We now want to add some modifications concerning the movement and attack behaviors of our characters.

More precisely, characters and objects will evolve over time in a way that, at the beginning:

→ always moves while flying (so no modification)

→ attacks by spitting fire

and, **at the beginning**:

→ a worker:

→ will evolve to move while running

→ a warrior:

→ will evolve to move while running

→ will evolve to attack with a sword

→ a mage:

→ will evolve to move while floating

→ will evolve to attack by launching a spell

→ a worker

→ moves while walking

→ never attacks and will never attack

→ a warrior

→ moves while running

→ attacks with a sword

→ a small mage, the mage

→ moves while walking

→ never attacks as he does not know magic

→ a dragon

and, **at the beginning**:

1. Modify your UML class diagram to take into account this evolution. You will use the strategy pattern for the modeling of new movement and attack functions of characters.

2. Implement, in your Java project, the changes made previously in the UML design.

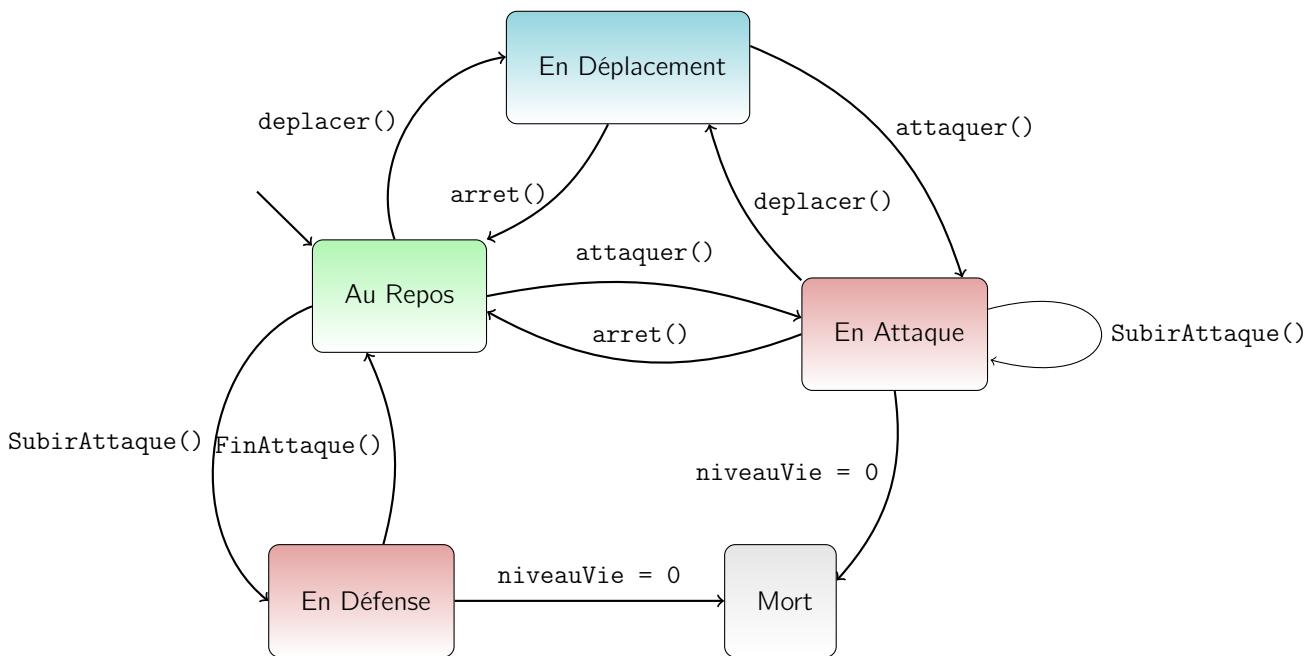
Exercice I.3

Gestion des états

Now that our characters have movement and attack behaviors, we must manage their states.

The automaton defining the different states taken by a character and the associated transitions is given below.





1. En utilisant notamment le pattern état, modifier votre diagramme de classe UML afin de prendre en compte la gestion des états des personnages.
2. Implémenter votre nouveau diagramme de classes. On réfléchira à l'implémentation à prévoir lorsqu'une transition n'est pas tirable depuis un état donné.

II - Les patterns de structure

Exercice II.1

L'équipement du guerrier ...

Afin de remplir la lourde tâche qui leur incombe, les guerriers peuvent, au cours du jeu, gagner ou acheter des options. On décide ici d'implémenter les options suivantes :

- une armure permettant de diminuer par 2 les dommages des coups reçus par le personnage
- un casque permettant de diminuer par 2 les dommages des coups reçus par le personnage
- une amulette doublant la puissance d'attaque
- un pouvoir de double attaque donnant la possibilité d'attaquer avec deux armes à la fois.

1. *Conception UML* : modifier votre diagramme de classes afin de prendre en compte ces nouvelles fonctionnalités en vous inspirant du pattern décorateur.
2. *Implémentation Java* : ajouter les classes java nécessaires à la gestion de ces options au jeu des personnages.

III - Les patterns de création

Nous allons maintenant essayer de contrôler et/ou faciliter la création d'instance des différents éléments de notre jeu. L'objectif du jeu des personnages est d'être le premier joueur à trouver et mettre la main sur le fabuleux et *unique* trésor, objet de toutes les convoitises.

Exercice III.1

Un unique trésor

1. *Conception UML* : modéliser cette fonctionnalité dans votre diagramme de classes.
2. *Implémentation Java* : À l'aide du pattern singleton, implémenter une classe modélisant le trésor telle qu'elle respecte, par construction, la règle énoncée ci-dessus : le trésor doit être *unique*.
3. À l'aide des tests unitaires, assurez-vous que cette règle est bien respectée.

Exercice III.2

La fabrique... de personnages !

Tout est dans le titre!!

En effet, nous souhaitons faciliter la vie des utilisateurs de nos différents personnages.

1. *Conception UML* : en utilisant les pattern fabrique et fabrique abstraite, proposer une nouvelle version de votre diagramme de classes afin de prendre en compte cette fonctionnalité.
2. *Implémentation Java* : mettre en œuvre cette évolution dans votre code java.

À suivre... .

‡ ‡ ‡