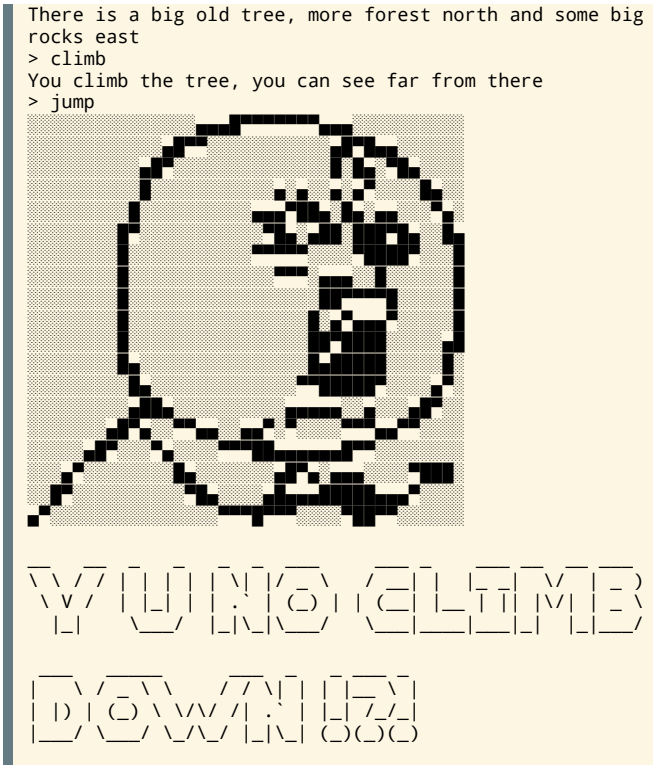
Projet : RPG

Programmation Orienté Objet

Salima HASSAS, Laëtitia MATIGNON, Antoine GRÉA

☘ Compétences À Acquérir

- ✔ Concevoir un projet complexe en groupe
- ✔ Résoudre les problèmes d'implémentation efficacement
- ✔ Apprendre à travailler à plusieurs sur le même code

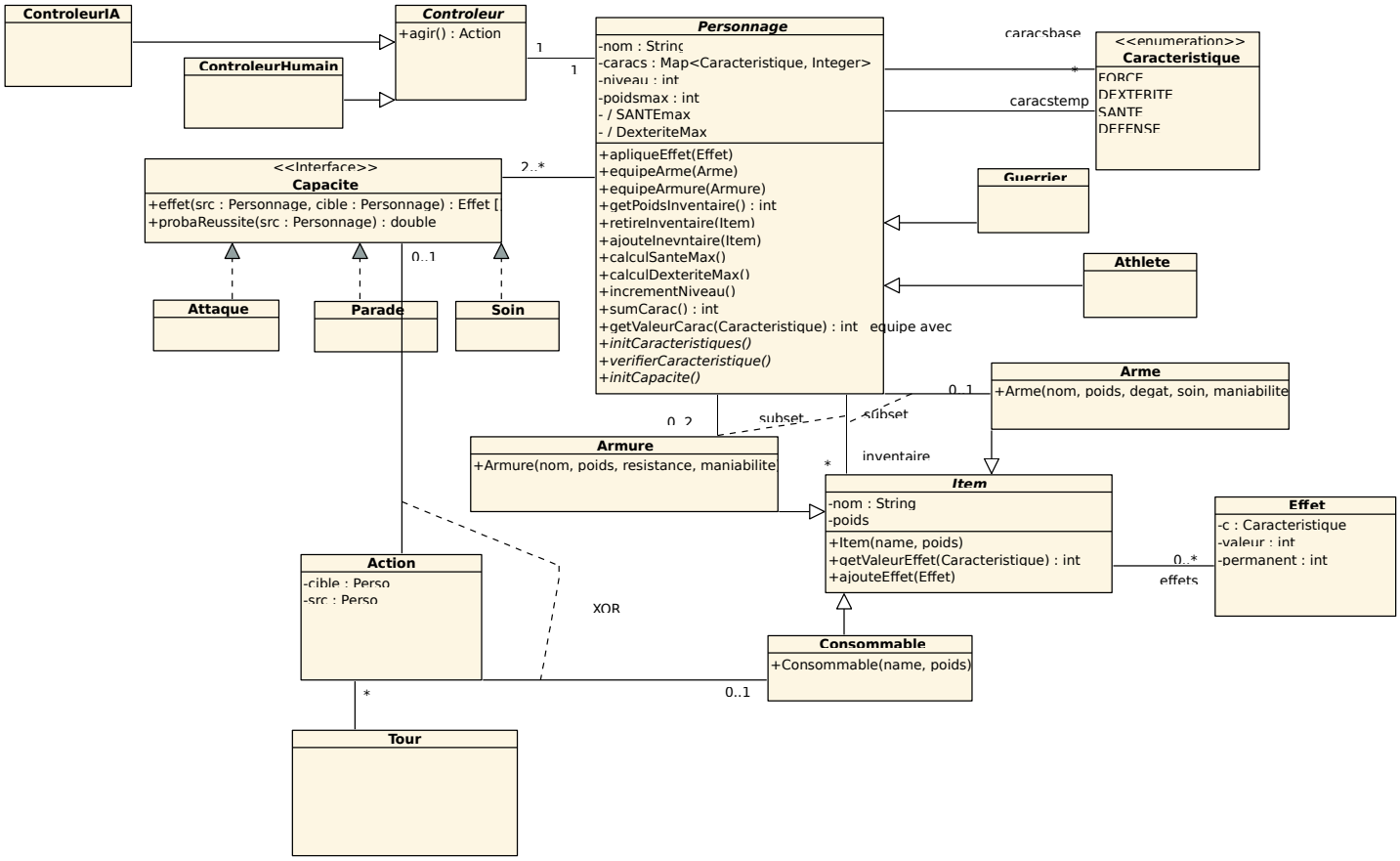


⊙ Objectifs Du Projet

- › Réaliser un schéma UML compréhensible
- › Spécifier chaque fonctionnalité à l'avance
- › Factoriser tout code factorisable (ne jamais se répéter dans le code !)
- › Privilégier le code minimal au code optimal
- › Noter les points de difficultés
- › Utiliser un outil de gestion de version collaboratif ainsi que l'IDE au maximum.
- › Faire un scénario, un monde et des personnages cohérent et originaux
- › Présenter son travail en mettant l'accent sur les points d'implémentation pertinents, sur les spécificités de vos fonctionnalités et sur vos difficultés.

i Un RPG (Role Playing Game) est un jeu de rôle qui consiste à mettre en œuvre des combats entre deux personnages ou deux groupes de personnages, en mode tour par tour. Ce style de jeu possède de nombreuses variations. Vous devez en proposer une ici.

> *Suite au TD vous devez avoir une implémentation de base. Si ce n'est pas le cas vous pouvez vous inspirer de ce schéma UML.*



1 Description Du Jeu

> *Petit rappel des bases du jeu demandé.*

1.1 Caractéristiques Des Personnages

> *Chaque personnage possède un **nom** et un ensemble de **caractéristiques** de base : force, dextérité, défense ; qui ont des valeurs entières. Chaque personnage a aussi un certain nombre de points de vie (sa santé) et un niveau d'expérience. Vous pourrez ajouter d'autres caractéristiques de base au besoin.*

- ✔ La **santé** diminue lorsqu'un personnage est blessé au combat. Elle peut augmenter si le personnage se soigne par exemple.
- ✔ Le **niveau** d'expérience augmente lorsqu'un combat est gagné. À chaque fois

qu'un nouveau niveau est atteint, il est possible d'augmenter de 1 point une des caractéristiques de base du personnage.

Vous devez définir des **contraintes** sur les caractéristiques permettant de rendre votre jeu équitable. Par exemple, une contrainte commune à tous les personnages pourrait être que la somme des valeurs des caractéristiques de base ne devra pas excéder. La santé doit être bornée. Sa valeur maximale sera calculée en fonction de la formule de votre choix.

Par exemple : $200 - \sum (valeurs_{caracteristiques_{de\ base}}) + experience \times 3$.

1.2 Classes De Personnage

> *Chaque personnage appartient à une **classe** (par exemple, magicien, guerrier ou athlète). Les classes permettent notamment de spécifier des **contraintes** sur les caractéristiques de base des personnages de la classe. Par exemple, l'aptitude d'un guerrier réside plus dans sa force physique. On peut donc imposer les contraintes suivantes à tous les guerriers :* $force \geq dexterite + 10 \geq defense$.

1.3 Équipement

> *Les personnages peuvent récupérer des **éléments** dans l'environnement : armes, nourriture, potions, armures, sortilèges... Chaque élément a un poids et chaque personnage peut porter un poids maximum, ce qui limite le nombre d'éléments présents dans l'inventaire d'un personnage.*

Certains éléments sont **consommables**, i.e. si le personnage les utilise, ils disparaîtront de l'inventaire.

D'autres éléments doivent être **équipés** par le personnage pour qu'il puisse les utiliser (équipement). Le nombre d'éléments avec lesquels le personnage peut s'équiper est limité. Par exemple, les personnages ne peuvent s'équiper que d'une seule arme et de deux armures.

Lorsqu'ils sont utilisés par un personnage, les éléments ont une influence sur les caractéristiques du personnage. Par exemple, si un personnage utilise une potion de santé, sa santé augmentera. Si un personnage utilise son arme pour attaquer un autre personnage (cf. les capacités), le personnage attaqué pourra voir sa santé diminuée.

1.4 Capacités

> *Les personnages ont un ensemble de **capacités** qui leur permettent de porter des coups à d'autre personnage (attaque), de parer ou esquiver des coups (parade), et de soigner (soin). Vous pouvez ajouter d'autres capacités au besoin. Chaque personnage doit avoir au moins deux capacités de base. Ces capacités de base seront déterminées en fonction de la classe du personnage. Par exemple, un guerrier aura comme capacités de base l'attaque et la parade.*

Les équipements du personnage ont un impact sur les effets d'une capacité mise en œuvre. Par exemple, un personnage armé d'une épée qui attaque, occasionnera plus de dégâts sur son adversaire qu'un personnage non armé. Un personnage équipé d'un bouclier subira moins de dégâts que s'il n'a pas d'armures.

L'utilisation d'une capacité se fait en deux temps :

- ✔ Vérifier sa réussite – la probabilité de réussite de la mise-en-œuvre d'une capacité est fonction des caractéristiques du personnage et de l'arme portée. Par exemple, pour une attaque, la probabilité de réussite pourra être fonction de la dextérité du personnage attaquant et de la maniabilité de son arme.
- ✔ Mesurer son impact – si la mise en œuvre de la capacité a réussi, son impact est fonction des caractéristiques des personnages impliqués et de l'équipement utilisé. Par exemple, pour calculer l'impact d'une attaque, on pourra :
  - › calculer les dégâts occasionnés comme la somme de la force de l'attaquant et de la valeur de dégât de son arme s'il en est équipé ;
  - › calculer la défense du personnage attaqué comme la somme de sa valeur de défense et des valeurs de résistance des armures s'il en est équipé ;
  - › calculer le dommage comme la différence entre les dégâts et la défense ;
  - › l'impact de l'attaque sera alors de soustraire le dommage à la santé du personnage attaqué.
  - › La mise en œuvre réussie d'une parade aura pour effet d'augmenter la défense du personnage s'il est attaqué. La mise en œuvre réussie d'un soin augmentera la santé du personnage d'une valeur fonction de la capacité de soin de ses équipements.

1.5 Déroulement D'un Combat

> *Un combat se déroule tour par tour et oppose deux groupes de personnages. Avant chaque combat, la santé des personnages est ré-initialisée à sa valeur maximale. À chaque tour de jeu, tous les personnages de l'équipe qui joue proposent une action. L'ordre peut être fixe ou déterminé au hasard ou en fonction des capacités et attributs.*

Une **action** pour un personnage correspond à la mise en œuvre d'une de ses capacités sur une cible choisie ou à l'utilisation d'un consommable.

À chaque attaque portée la cible peut perdre de la santé. Si la santé du personnage cible tombe à zéro, il est vaincu et ne peut plus agir dans les tours de jeu suivant. Si tous les personnages d'une équipe sont vaincus, l'autre équipe remporte la victoire. Tous les personnages non vaincus de l'équipe gagnante gagnent alors un niveau.

1.6 Événements

> *Un événement est un élément narratif de votre jeux. Les combats sont des événements (possiblement aléatoires). Les événements peuvent aussi porter sur l'ajout de personnages, la découverte d'éléments dans l'environnement, etc.*

2 Conseils D'implémentation

> *Vous devez implémenter le projet selon votre diagramme de classes. Vous pouvez utiliser easyUML pour aller plus vite si vous le souhaitez.*

2.1 NE PANIQUEZ PAS !

i Pour toutes questions sur Java vous pouvez consulter l'aide mémoire sur spiral ou ici. Je suis aussi joignable par mail pour toute question à antoine.grea@univ-lyon1.fr

2.2 Bibliothèque D'Utilitaires

> *Vous avez à votre disposition une bibliothèque d'utilitaires que vous retrouverez sur spiral et ici.*

! Vous devez utiliser un système de Log dans votre projet, cette bibliothèque en contient un mais vous pouvez aussi opter pour log4j.

2.3 Les Meilleures Solutions Sont Les Plus Simples

i Si vous pensez que vous faites un code trop compliqué c'est que c'est le cas. Prenez le problème autrement ou tenter de l'expliquer à quelqu'un verbalement. Ne vous répétez pas et tenter de faire en sorte que votre code ne nécessite aucune explications supplémentaire. Faites en le moins possible et soyez fainéant : le plus simple et le plus court est le meilleur.