

Projet de Programmation Avancée

2016-2017

Synthèse Technique

Développeurs	Encadrants
Sylvain Duhau-Marmon	Baptiste Pesquet
Valentin Mancier	Pierre-Alexandre Favier
	Edwige Clermont

Préface

Ce livrable reprend la structure du document explicitant les spécifications prévues. La conception de chaque objet annoncé y est détaillée, les choix concernant son développement et l'architecture mise en place y sont justifiés.

Veuillez trouver le diagramme des classes correspondant à notre projet en pièce jointe.

Table des matières

I - Scénario	4
Univers	4
Héros	4
II - Gameplay	5
Déplacements	5
Déplacements dans le monde	5
Déplacements dans les villes	6
Déplacements dans les donjons	6
Personnalisation du héros	7
Statistiques	7
Equipements	7
Classes	8
Objets	8
Or	8
Consommables	9
Villageois	9
Alchimiste	9
Forgeron	9
Ennemis	9
Combat	10
Actions	10
Récompenses	10
Aide	10
III - Exigences fonctionnelles	11
IV - Tests	11

I - Scénario

1. Univers

Le jeu est un RPG de type donjons & dragons. Celui-ci prend place dans le monde fantastique médiéval d'Ivalice. Ce monde est corrompu par les sbires du dragon Adrammelech semant le chaos dans toutes les contrées d'Ivalice. Les habitants d'Ivalice se sont organisés pour lutter contre l'invasion de monstres et ont érigé des murs pour protéger leurs cités. Cependant, des donjons infestés de créatures maléfiques rendent les voyages entre les cités périlleux.

Le jeu exploite la construction d'un objet de la classe **World**, dans lequel évolue le joueur (attribut).

Le monde est statique et est composé de salles élémentaires représentées par la classe **Room** (voir [§II.1](#)). Une initialisation aléatoire du monde (notamment des donjons), peut être envisagée mais n'a pas été retenue pour faciliter l'environnement de test.

2. Héros

Le héros vient tout juste de terminer son initiation auprès des chevaliers Hokuten. Il choisit son nom de guerrier et sa spécialité et se lance dans son aventure. Son but est de répandre ses idéaux de justice et d'honneur, et surtout se libérer Ivalice des griffes du dragon Adrammelech. Son aventure commence dans la cité de Cyril.

En début de partie, le joueur choisit son nom et sa classe, qui seront fixés pour le reste de l'aventure. La partie prend fin lorsque le joueur parvient à terrasser le dragon Adrammelech. Il peut tout de même continuer à évoluer dans le monde s'il le souhaite.

La classe abstraite **Entity** représente les unités combattantes du jeu. Celle-ci se décline en deux sous-classes **Player** et **Enemy**, représentant respectivement le héros et les monstres. En effet, ceux-ci partagent de nombreuses statistiques et méthodes communes (voir [§II.2.a](#) et [§II.5](#)).

II - Gameplay

1. Déplacements

Le joueur choisit s'il veut se déplacer vers le Nord, le Sud, l'Est ou l'Ouest en entrant le chiffre correspondant via le pavé numérique du clavier puis en appuyant sur la touche "Entrée".


Si aucune action spéciale n'est déclenchée sur la case (par un PNJ, ou par un monstre), alors le joueur a accès à la méthode `Player.Move()`, lui permettant entre autres d'effectuer un déplacement vers une des 4 directions cardinales.

L'affichage de ces commandes se fait similairement à une rose des vents de la sorte :

	1 : Nord	
2 : Ouest		3 : Est
	4 : Sud	

a. Déplacements dans le monde

Le monde est rond, comporte 5 villes et donjons et s'organise de la sorte :

Ville de Cyril	
Donjon des Gobelins	
Ville de Sprohm	
Donjon des Trolls	
Ville de Cadoan	
Donjon des Centaures	
Ville de Baguba	
Donjon des Orcs	
Ville de Muscadet	
Donjon des Dragons	
Ville de Cyril	
...	

Les **Room** sont quadruplement chaînées, c'est à dire que chaque Room connaît les Room qui lui sont adjacentes et indique donc au joueur dans lesquelles le déplacement est possible.

b. Déplacements dans les villes

Les villes comportent 9 cases et leur disposition est celle-ci :

<p><i>Forgeron</i> → Achat d'équipements</p>	<p><i>Entrée du donjon suivant</i></p>
<p><i>Sortie du donjon précédent</i></p>	<p><i>Alchimiste</i> → Achat de potions</p>

Une carte de la ville peut être affichée par le joueur.

La méthode Player.Move() propose la commande Carte (menu n°5), permettant d'afficher une représentation graphique de ces 4 cases.

c. Déplacements dans les donjons

Les donjons sont composés de plusieurs salles. Le nombre de salles dépend de la difficulté du donjon. Certaines cases déclenchent un combat contre un monstre du donjon. La dernière salle mène au boss du donjon.

La topologie du donjon est inconnue au joueur. Le premier donjon comporte 5 salles. Ce nombre augmente d'une unité par donjon. Lorsqu'un monstre vivant est présent dans la **Room** courante (attribut non nul), celle où se trouve le joueur, un combat s'engage automatiquement entre les 2 entités.

2. Personnalisation du héros

a. Statistiques

Les statistiques du héros sont les suivantes :

- **PV** (Points de Vie) → Vitalité du joueur, si celle-ci tombe à zéro, le héros se réveille à son point de respawn
- **Mana** → Énergie spirituelle du joueur, permet d'effectuer des sorts spéciaux
- **Attaque**
- **Défense**
- **Vitesse**
- **Critique** → Permet d'activer des capacités passives
- **Niveau** → Monter de niveau augmente toutes les statistiques précédentes

- **Expérience** → *Vaincre un monstre permet de recevoir de l'expérience*

Toutes ces statistiques, excepté l'expérience (attribut de la classe **Player**), sont communes aux monstres et au héros & sont donc des attributs de la classe **Entity**. Les statistiques boostées par les équipements du joueur, dites effectives, sont également attributs de la classe **Player**.

Cela permet de différencier les statistiques effectives du joueur en combat et ses statistiques de base, par exemple pour le passage à un niveau supérieur. Cela demande aussi l'utilisation de méthodes différentes pour la régénération d'un monstre (HeathMax), ou d'un héros (EffectiveHealth), en définissant une méthode "virtual" dans **Entity**.

b. Equipements

Le héros peut s'équiper d'un élément de chacune de ces catégories :

- **Arme** → *Augmente l'attaque*
 - *Epée courte : +10 attaque*
 - *Sabre vicié : +20 attaque*
 - *Ragnarok : +30 attaque*
 - *Claymore : +40 attaque*
 - *Masamune : +50 attaque*
 - *Excalibur : +75 attaque*
- **Armure** → *Augmente la défense*
 - *Cuirasse : +10 défense*
 - *Cotte de fer : +20 défense*
 - *Carapaçon : +30 défense*
 - *Côte d'or : +40 défense*
 - *Plastron diamant : +50 défense*
 - *Maximilien : +75 défense*
- **Casque** → *Augmente les PV*
 - *Melon plume : +10PV*
 - *Casque romain : +20 PV*
 - *Casque de fer : +30 PV*
 - *Casque genji : +40PV*
 - *Casque diamant : +50PV*
 - *Casque vangaa : +75PV*
- **Gants** → *Augmente le critique*
 - *Gantelet : +10 Critique*
 - *Gants ivoire : +20 Critique*
 - *Bracers : +30 Critique*
 - *Gants genji : +40 Critique*
 - *Mitaines de feu : +50 Critique*
 - *Gants mog : +75 Critique*
- **Bottes** → *Augmente la vitesse*
 - *Bottes cloutées : +10 vitesse*

- *Bottes de combat* : +20 vitesse
- *Bottes en plume* : +30 Vitesse
- *Caligula* : +40 vitesse
- *Tabi ninja* : +50 vitesse
- *Bottes de 7 lieues* : +75 vitesse

Les **Item** sont construits à partir d'une fonction statique de la classe faisant appel au constructeur puis celle-ci renvoie l'objet créé.

Le joueur possède un équipement de chaque type. Chaque **Equipment**, sous-classe d'**Item** possède un niveau ce qui permet de remplacer un équipement de niveau inférieur par un équipement de niveau supérieur lorsque cela est possible.

c. Classes de jeu

Le héros choisit une classe avec des statistiques spécifiques et un sort parmi ces quatre au début de son aventure :

- **Écuyer** → Généraliste
 - *"Gloire du Juste"* : inflige des dégâts supplémentaires
- **Spadassin** → Spécialisé dans l'attaque
 - *"Coups Critiques"* : double les dégâts infligés
- **Paladin** → Spécialisé dans la défense
 - *"Drain"* : soigne des dégâts infligés
- **Ninja** → Spécialisé dans la vitesse
 - *"Attaque Furtive"* : l'attaque de base ne prend pas en compte l'armure

Les 4 classes de jeu sont des sous-classes de **Player**. Elles disposent d'un constructeur accordant des statistiques de base spécialisées, mais également des attaques spéciales sous la forme d'une méthode "override" `Spell()` reprenant la méthodes abstraites définie dans **Player**.

3. Objets

a. Or

La monnaie du jeu est l'or, elle permet d'acheter et de vendre des objets aux différents commerçants.

Les **Enemy** et le **Player** possèdent de l'or, ce qui permet au joueur de gagner l'or de l'ennemi lorsqu'il l'a vaincu pour le dépenser ensuite dans des **Item** chez les **Villager**.

b. Consommables

Les consommables sont :

- *Potions* → Permet de restaurer ses points de vie même à l'extérieur du village
 - *Potion* : +10PV

- *Super Potion* : +25PV
- *Hyper Potion* : +50PV
- *Potion X* : +100PV
- *Parchemins* → Permet de se téléporter jusqu'à notre point de respawn

Les **Consumable** sont des **Items** créés de la même manière que les **Equipment**. Le joueur possède un inventaire (inventory) qui est une liste de consommables qu'il pourra ensuite utiliser en combat.

4. Villageois

Les villageois sont des PNJ instanciés par la classe **Villager**.

a. Alchimiste

L'alchimiste se trouve dans les villes. Il est chargé de vendre les potions et parchemins listés plus haut au joueur.

L'**Alchemist** est un **Villager** vendant des **Consumable** (**Potion & Scroll**).

b. Forgeron

Le forgeron se trouve dans les villes. Il est chargé de vendre les différents équipements au joueur. Le niveau des équipements qu'il vend dépendent de son propre niveau.

Le **BlackSmith** est un **Villager** vendant des **Equipment** (**Armor, Weapon, Gloves, Boots & Helmet**).

5. Ennemis

Il est possible de combattre les ennemis suivants :

- *Gobelins*
- *Boss Gobelins*
- *Trolls*
- *Boss Troll*
- *Centaures*
- *Boss Centaur*
- *Orcs*
- *Boss Orc*
- *Dragons*
- *Boss Dragon*

Les **Enemy** sont créés de la même manière que les **Item**. Ils possèdent de l'or et un **Equipment** aléatoire qui pourra être récupéré par le **Player** après l'avoir vaincu. En combat, ils possèdent seulement une attaque de base sous la forme d'une méthode `HitPlayer()`.

6. Combat

a. Actions

Il existe 3 actions possibles lors d'un combat :

- **Attaquer** → Inflige des dégâts au points de vie adverses
- **Lancer un sort** → Lance le sort de la classe que le joueur a sélectionné
- **Utiliser une potion** -> affiche la liste des potions disponibles et demande au joueur d'en choisir une

Le combat s'effectue en tour par tour et le joueur le plus rapide commence (celui avec le plus de vitesse).

L'attribut `_speed` de la classe **Entity** définit qui du **Player** ou de l'**Enemy** commence à attaquer. Le choix des actions se fait grâce à un "switch", en considérant les commandes du joueur sous la forme de chiffres.

b. Récompenses

Vaincre un monstre dans un donjon permet d'obtenir de l'or et parfois un équipement. Si l'équipement ramassé est de plus haut niveau que celui équipé, il est automatiquement remplacé.

La méthode `Player.Looting()` permet au joueur d'effectuer cette action.

7. Aide

Le RPG est textuel et ne se joue qu'au clavier.

La commande aide permet d'afficher les commandes disponibles en jeu.

La méthode `DisplayHelp()` est le 9ème menu de la méthode `Player.Move()`.

III - Exigences fonctionnelles

- *Le joueur ne doit utiliser que le clavier pour effectuer ses actions en jeu. OK*
- *Le joueur disposera d'une commande "aide" permet de lister toutes les actions possibles en jeu. OK*
- *Le joueur peut se déplacer dans les 4 directions cardinales. OK*
- *Le joueur peut afficher une carte de la ville. OK*
- *Le joueur peut revenir à un point de sauvegarde lorsqu'il meurt. OK*
- *Le joueur doit pouvoir combattre des ennemis dans les donjons. OK*
- *Le joueur doit pouvoir choisir entre plusieurs classes de personnages. OK*
- *Le joueur doit pouvoir nommer son personnage. OK*
- *Le joueur doit pouvoir porter des équipements lui conférant des bonus. OK*
- *Le joueur doit pouvoir utiliser des objets consommables. OK*
- *Le joueur doit pouvoir acheter des pièces d'équipement auprès d'un PNJ. OK*
- *Le joueur doit pouvoir restaurer sa vie et son mana auprès de PNJ.*
- *Le joueur doit pouvoir effectuer des quêtes lui donnant des récompenses.*
- *Le joueur doit pouvoir monter de niveau pour augmenter ses statistiques de base. OK*
- *Chaque classe de personnage possède des statistiques de base différentes. OK*
- *Chaque classe de personnage possède un sort unique. OK*
- *Les ennemis vaincus octroient des récompenses. OK*
- *Le monde doit posséder plusieurs villes et donjons. OK*

Il est possible d'implémenter une classe Auberge héritant de **Villager** qui permettrait de régénérer son Mana et une classe Castle héritant elle aussi de **Villager** qui permettrait au chef du village de donner des quêtes à notre héros.

IV - Tests

Des tests fonctionnels et ergonomiques ont été conduits en binôme après chaque séance de développement. L'approche TDD n'a pas été utilisée pour ce projet. Nous sommes toutefois enthousiastes d'essayer cette approche pour des développements futurs, permettant d'assurer une certaine robustesse du code et de permettre une intégration continue.