Projet Programmation avancée : Création d'un jeu de gestion de colonie

Rapport

LEGER Sébastien

sleger003@ensc.fr

RENAUD Thomas

thomarenaud@ensc.fr

Année : 2021/2022

Table des matières

1. Introduction

Ce projet nous a été présenté le lundi 7 mars 2022 et devait être rendu le jeudi 21 avril 2022. L'objectif de ce projet était d'imaginer et de programmer un jeu de gestion de colonie en mode Console en utilisant le langage C#, avec une approche "POO". Nous avons eu 12h de cours pour mettre en place notre projet, auxquelles s'ajoutent des heures supplémentaires de travail en autonomie.

En ce qui concerne notre groupe, nous avons choisi de former une équipe de deux. Le travail s'est fait sous forme de phases de recherches groupées pour définir le plan d'action, puis de phases de programmation individuelles où chacun s'occupait d'un morceau du code en essayant toujours de rester le plus équitable possible.

Le jeu que nous avons décidé de coder s'appelle **CITADELLE**, le but du jeu est de construire (en un nombre de tours limité) une cité et de la compléter, mais attention aux catastrophes naturelles qui viendront complexifier cette quête !

Les contraintes techniques :

La programmation a été réalisée en C#, en mode Console, dans les conditions habituelles des TP et en utilisant, si besoin, des bibliothèques de fonctions externes. Contrairement au projet du S5, la Programmation Orientée Objet (POO) est autorisée et, est même au cœur de ce projet. Le code source respecte les règles d'indentations, la norme camelCase et des commentaires sont présents pour expliquer des parties de code complexes.

2. Gestion de projet

Au lancement du projet, nous avons décidé de mettre en place un planning. En effet, en voyant l'ampleur du projet et le fait que celui-ci utilise la POO nous avions du mal à nous projeter concrètement dans le projet. Pour nous permettre de nous lancer rapidement et de ne pas perdre des heures précieuses à travailler, nous avons listé les tâches à réaliser. Nous avons donc pris chaque problème l'un après l'autre et essayé, contrairement au projet précédent, de penser à la structure du code avant de commencer à coder. En effet, en POO, un intérêt tout particulier à l'organisation, et donc à la factorisation du code, doit être porté.

Tâche	Description	Temps estimé
Découverte du sujet	Lire attentivement l'intégralité du sujet en repérant les éléments clés à mettre en place.	1h
Règles du jeu	Réflexion sur l'organisation de notre jeu, quelles sont les différentes manières de gagner et de perdre ? Quel est le but du jeu ?	2h
Définitions des classes	Mise en place des différentes classes qui nous seront utiles dans le jeu. Organisations des relations entre les classes.	2h
Création d'un diagramme de classe	Mise au propre des relations entre les différentes classes en utilisant l'application diagrams.net.	2h
Jouabilité du jeu	Choix de la manière dont le jeu sera jouable, entrées de l'utilisateur, touches spécifiques du clavier, curseur ?	1h
Mise en place de l'interface graphique	Pour éviter de coder sans tester les fonctionnalités, créer en premier lieu l'affichage de la carte et des informations.	3h
Curseur	Mise en place de la classe curseur permettant de se déplacer et de sélectionner un élément du jeu.	2h
Implémentation des bâtiments	Création des classes bâtiments avec la possibilité de les construire et de les améliorer.	5h
Finition des règles du jeu	Choix arbitraire des valeurs prises par les différentes caractéristiques des bâtiments suivant leur niveau. Ainsi que le coût de production.	1h
Implémentation des personnages	Création de la classe colon avec la possibilité de se déplacer dans la carte.	4h

Tâche	Description	Temps estimé
Classe cité	Création d'une classe regroupant toutes les informations sur l'état d'avancement du jeu.	1h
Relation Personnage/Bâtiment	Mettre en place le déplacement d'un colon quand un bâtiment se construit ou s'améliore (suivant le coût de production).	3h
Gérer les choix	Créer une classe permettant de limiter les entrées du joueur à des éléments acceptables.	3h
Messages d'erreurs	Création de messages d'erreurs pour aider le joueur à comprendre son erreur. Ils gèrent toutes les exceptions telles qu'une mauvaise saisie d'entrée ou bien si les conditions ne sont pas remplies (ex : ressources insuffisantes pour améliorer un bâtiment).	2h
Classe Catastrophe naturelle	Dernière classe à implémenter : celle qui fera trembler plus d'un joueur. A chaque fin de tour, l'aventure peut brutalement s'arrêter.	3h
Accessibilité du jeu	Vérifier les contrastes.	1h
Tests d'intégration	Vérification du bon fonctionnement de l'enchaînement des classes et des méthodes.	3h
Rapport	Rédaction pendant toute la durée du projet.	8h
Total		**47h**

Nom	***Temps estimé***
Thomas RENAUD	*24h**
Sébastien LEGER	*23h**
Total	***47h***

**Dans l'idéal, 50% de la charge de travail pour chaque membre du binôme mais au fur et à mesure de l'avancée du projet, nous adapterons cette planification. Un membre pourra être plus à l'aise sur du code pur, l'autre sur la rédaction du rapport ou sur la création des règles du jeu par exemple.*

Fig. 1 : Planification détaillé du projet de programmation avancé

Tâche	Temps estimé	Temps réel	Acteur
Découverte du sujet	1h	1h	Thomas et Sébastien
Règles du jeu	2h	3h	Sébastien
Définitions des classes	2h	4h	Thomas et Sébastien
Création d'un diagramme de classes	2h	4h	Sébastien
Jouabilité du jeu	1h	1h	Thomas
Mise en place de l'interface graphique	3h	4h	Thomas
Curseur	2h	3h	Sébastien
Implémentation des bâtiments	5h	6h	Thomas et Sébastien
Finition des règles du jeu	1h	1h	Sébastien
Implémentation des personnages	4h	3h	Sébastien
Classe cité	1h	1h	Thomas
Relation Personnage/Bâtiment	3h	3h	Thomas
Gérer les choix	3h	2h	Sébastien
Messages d'erreurs	2h	2h	Thomas
Classe Catastrophe naturelle	3h	3h	Thomas et Sébastien
Accessibilité du jeu	1h	1h	Thomas
Tests d'intégration	3h	3h	Thomas
Rapport	8h	8h	Thomas et Sébastien
Total	**47h**	**53h**	-

Fig. 2 : Matrice des temps réel avec acteur par tâche.

Le projet nous a été présenté le lundi 7 mars 2022 et devait être rendu le jeudi 21 avril 2022. Nous avons eu 12h de TP réparties dans l'emploi du temps pour mener à bien ce projet. Cependant, des heures de travail supplémentaires en autonomie ont été nécessaires pour le terminer. Vous trouverez ci-dessus nôtre matrice de tâches et d'implication détaillant

les différentes phases de notre projet qu'elles furent traitées individuellement ou collectivement.

Concernant l'organisation, nous n'avons pas créé de compte GitHub. Thomas s'est occupé en grande majorité du code tandis que Sébastien s'est davantage focalisé sur le rapport et les diagrams.net en aidant Thomas à réfléchir à comment bien organiser le code en classe. Sébastien s'est imprégné au fur et à mesure du code et s'occupait de repérer les coquilles dans le code et/ou les commentaires internes à celui-ci. Le choix a d'ores et déjà été fait d'utiliser GitHub pour le projet de Communication Web afin de mieux équilibrer l'apport amené par chacun dans la réalisation du projet mais surtout pour faciliter l'accès à un code régulièrement mis à jour.

3. Wiki du jeu

3.1. Présentation du jeu

Envie de partir à la conquête de la Grèce antique ? Vous voulez marcher dans les pas d'Ulysse, de Leonidas, de la belle Hélène et d'autres mythiques héros grecs ? Alors rejoignez-nous dans une aventure unique qui vous emmènera au cœur de la Grèce antique !

Jeu en local gratuitement sur vôtre ordinateur, **CITADELLE** met particulièrement l'accent sur la stratégie et réveille chez les joueurs leur âme et instinct de bâtisseur ! Votre objectif sera de transformer une petite ville en une immense métropole. Recrutez des personnages puissants composés actuellement uniquement de colons.

Bâtissez soigneusement vôtre ville ou vous le regretterez. Dans votre quête de gloire et de prestige, n'oubliez pas de gagner les faveurs des divinités grecques pour échapper aux catastrophes naturelles. Construisez vôtre cité et imposez-vous comme le plus grand souverain de la Grèce antique !

3.1.1. Fondez une puissante métropole

Votre aventure commence sur une petite île paisible, sur laquelle vous pourrez déjà bâtir vos bâtiments. Votre objectif est d'établir une nouvelle ville et d'y rassembler vos premières ressources. Aidez votre cité à se transformer en une magnifique métropole et développez là avec 4 différents bâtiments.

3.1.2. Bravez les catastrophes naturelles

Vôtre aventure ne sera pas sans embûche, vous devrez affronter les forces de la Nature. Même si vous atterrissez sur une île paisible où la population ne connaît pas la guerre, vous n'échapperez pas à la colère de Zeus et Poséidon qui viendront endommager voire même détruire les avancées de vôtre cité.

3.2. Bâtiments et unité

3.2.1. Sénat

- **Description du bâtiment :**

Grâce au sénat, vous pouvez construire de nouveaux bâtiments ou agrandir les bâtiments déjà construits. Le Sénat peut stocker une quantité finie d'or et de population (100 et 3 respectivement) quel que soit son niveau. Le niveau de développement maximal de ce bâtiment est le niveau 4.

- **Coordonnées : [5,4]**

- **Prérequis de construction :**

Ce bâtiment n'a aucun prérequis de construction, il est présent au lancement de la partie.

- **Onglet du Sénat :**

1) Onglet de construction
2) Onglet d'amélioration
3) Onglet de recrutement

- **Information d'amélioration :**

Niveau	[Or]	[Populations utilisées]	Points
1	-	-	110
2	10	3	121
3	25	5	133
4	49	8	146

3.2.2. Mine d'or

- **Description du bâtiment :**

Dans la mine d'or, vos ouvriers extraient le précieux minerai d'or à partir duquel est frappée la monnaie de la cité. Plus le niveau de développement de la mine est élevé, plus la quantité de monnaie frappée par tour sera importante. Le niveau de développement maximal de ce bâtiment est le niveau 4.

- **Coordonnées : [9,9]**

- **Prérequis de construction :**

Sénat construit.

- **Onglet de la Mine :**

Onglet d'amélioration

- **Information d'amélioration :**

Niveau	[Or]	[Populations utilisées]	Rendement par tour	Points
1	4	1	8	22
2	14	3	12	24
3	29	4	18	27
4	49	6	24	30

3.2.3. Entrepôt

- **Description du bâtiment :**

Les ressources de votre ville sont stockées dans l'entrepôt. Plus le niveau de développement de l'entrepôt est élevé, plus vous pouvez y stocker de ressources. Le niveau de développement maximal de ce bâtiment est le niveau 4.

- **Coordonnées : [7,6]**

- **Prérequis de construction :**

Sénat construit.

- **Onglet de l'entrepôt :**

Onglet d'amélioration

- **Information d'amélioration :**

Niveau	[Or]	[Populations utilisées]	Stockage d'or	Points
1	-	-	300	15
2	47	0	700	17
3	93	2	1200	19
4	150	3	1700	22

3.2.4. Ferme

- **Description du bâtiment :**

La ferme nourrit les colons. Votre ville ne peut pas grandir sans développer votre ferme. Plus le niveau de la ferme est élevé, plus le nombre d'habitants que peut accueillir votre cité augmente. Le niveau de développement maximal de ce bâtiment est le niveau 4.

- **Coordonnées : [0,1]**

- **Prérequis de construction :**

Sénat construit.

- **Onglet de la ferme:**

Onglet d'amélioration

- **Information d'amélioration :**

Niveau	[Or]	[Populations utilisées]	Population maximale	Points
1	-	-	14	14
2	1	0	38	17
3	5	0	69	19
4	26	0	105	22

3.2.5. Colon

- **Description de l'unité:**

Le colon permet de construire les bâtiments de la cité en s'y déplaçant.

- **Coordonnées : [5,4]**

- **Prérequis de construction :**

Sénat construit.

- **Information d'amélioration :**

[Or]	[Populations utilisées]
5	1
1	0
5	0
26	0

3.3. Déroulement d'une partie

Fig. 3 : Capture d'écran de l'affichage de l'écran d'accueil

Une partie se déroule de la manière suivante :

Tout d'abord, une fois le code lancé, vous vous retrouvez sur une page d'accueil vous proposant de jouer, de connaître les règles du jeu ou de le quitter. En choisissant de regarder les règles du jeu, une nouvelle page apparaîtra dans la console vous expliquant, pas à pas, comment prendre en main le jeu : les grands principes du jeu (déplacement via un curseur clavier, construire un bâtiment, améliorer un bâtiment, système de "tour", etc.) et les différentes façons de remporter et perdre la partie. En choisissant de quitter le jeu, la console se ferme. Et enfin le plus important : le lancement du jeu. Une fois le jeu lancé vous pouvez vous déplacer librement sur l'interface du jeu. Différentes possibilités s'offrent à vous, vous pouvez sélectionner un bâtiment pour l'améliorer en utilisant le bouton de la barre latérale. Ou bien construire, recruter via le menu de caractéristiques issu de la sélection du Sénat.

Une fois que le joueur est satisfait, il peut passer son tour en utilisant le bouton PASSER SON TOUR présent en bas à droite de l'interface. Une catastrophe naturelle se produit alors. Au début du tour suivant un récapitulatif de sa cité et de sa production lui est proposé suite au passage de la catastrophe..

Une fois que la cité est complète (Victoire) ou que le Sénat est détruit (Défaite) le jeu s'arrête.

3.4. Interface du jeu

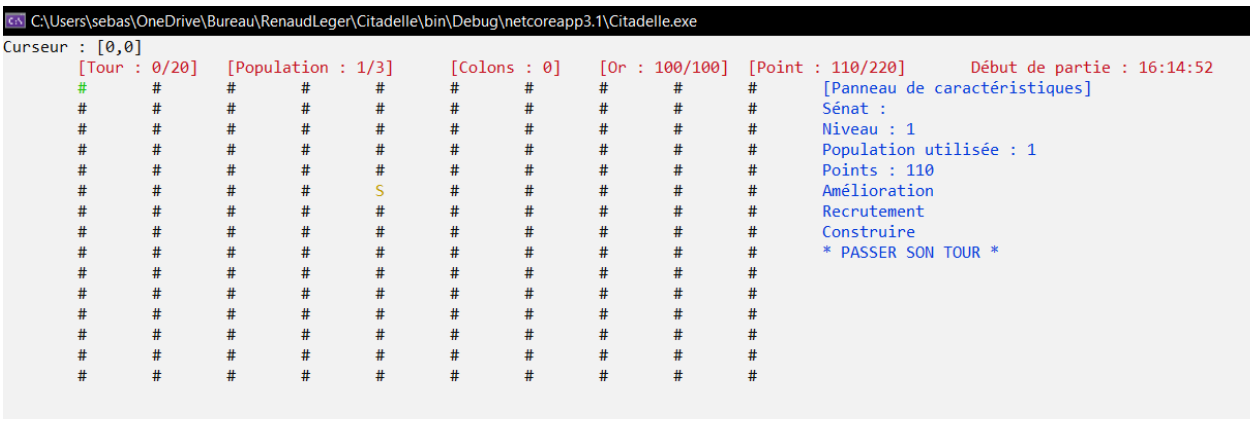


Fig. 4 : Capture d'écran de l'interface du jeu

Comme vous pouvez le voir sur l'image ci-dessus, la plupart de l'écran est dédié à la vue de la carte. Grâce à des barres latérales et des barres d'outils en haut de l'écran, le joueur peut accéder à toutes les informations qui lui sont nécessaires pour qu'il puisse jouer.

L'écran de jeu est composé de deux barres d'outils :

Depuis la barre latérale de droite, le joueur peut accéder à différentes fonctionnalités clés du jeu. En accédant aux caractéristiques d'un bâtiment, le joueur peut décider de réaliser un certain nombre d'actions.

La barre d'outils du haut affiche la population qui compose la cité, les ressources présentent ainsi que l'heure du début de partie.

3.3.1. Ressources

Fig. 5 : Capture d'écran de l'affichage des ressources dans le jeu

Ceci affiche la quantité de chaque ressource dont vous disposez actuellement dans l'entrepôt, ainsi que la capacité de stockage de celui-ci. Vous pouvez aussi voir le nombre d'habitants actuellement dans la ville, et combien elle peut en contenir au maximum.

3.3.2. Boutons

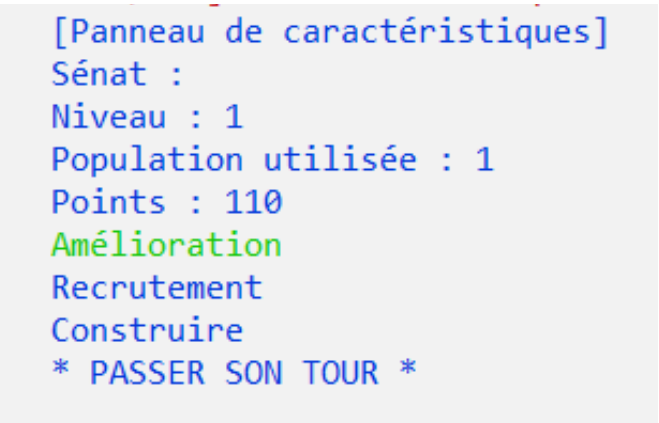


Fig. 6 : Capture d'écran de l'affichage des boutons d'action

Nom	Description
Construire	Le bouton construire amène à un menu permettant de choisir le bâtiment à bâtir. L'affichage des ressources nécessaires est pris en charge.
Amélioration	Ce bouton permet d'accéder aux ressources nécessaires à l'amélioration d'un bâtiment.
Recrutement	Menu de recrutement de colons, la quantité peut être choisie par le joueur.
PASSER SON TOUR	Menu pour passer son tour.

3.3.3. Écran principal

La partie principale de votre écran est la zone de jeu. Lorsque tous les menus sont fermés, vous pouvez voir votre île. Vous pouvez utiliser les flèches du clavier pour naviguer sur le monde. C'est dans cette vue que vous pouvez interagir avec vos bâtiments et vôtre cité.

Lorsque vous cliquez pour ouvrir un menu, il s'ouvrira comme un pop-up. Cliquez sur "Entrer" ou la touche demandée pour fermer le pop-up.

3.3.4. Détails de la ville

Elément	Description
S	Sénat : bâtiment principal de la ville
E	Entrepôt : lieu de stockage des ressources
F	Ferme : hébergement de la population
M	Mine : production d'or
c	Colon : bâtisseur de la cité
#	Élément vierge

3.3.5. Carte du monde

	0	Axe des Y	9
0			
Axe des X			
14			

```
# F # # # # # # # #
# # # # # # # # # #
# # # # # # # # # #
# # # # # # # # # #
# # # # # # # # # #
# # # # S # # # # #
# # # # # # # # # #
# # # # # # E # # #
# # # # # # # # # #
# # # # # # # # # M
# # # # # # # # # #
# # # # # # # # # #
# # # # # # # # # #
# # # # # # # # # #
# # # # # # # # # #
```

Le monde est divisé en 150 cases dans une disposition en grille, et chaque case est numérotée allant de (0,0) à (14,9). Le numéro de la case correspond aux coordonnées (X, Y) dans le monde.

3.4. Guide de prise en main du jeu

3.4.1. Ressources

L'exploitation de vos ressources est la base de la croissance de votre ville afin de devenir une cité puissante. Dans **CITADELLE**, il existe une seule ressource : l'or. Toute construction de bâtiments ou création d'unités nécessite des ressources. Les ressources sont récoltées automatiquement et produites par la mine d'or.

3.4.2. Premier pas

Au départ, votre cité requiert une chose plus qu'une autre : des ressources. La mine d'or produit de la monnaie. Plus vous investissez vos ressources dans les bâtiments de production, plus votre production au tour par tour sera grande. Ce développement est prioritaire et vous devrez y accorder toute votre attention lors des premiers tours.

En parallèle avec le développement des bâtiments de production, vous aurez besoin de développer votre ferme. À chaque fois qu'un bâtiment est amélioré, ce dernier mobilisera des habitants supplémentaires afin de garder un rythme de travail correct. Au fur et à mesure que votre production de ressources augmentera, vous aurez besoin d'améliorer votre entrepôt afin de stocker les ressources produites.

3.4.3. Construction de bâtiment

Pour construire un bâtiment, déplacer votre curseur au niveau du Sénat. Appuyer sur la touche "ENTER" puis rendez vous sur la barre latérale des caractéristiques. Quand vous y êtes, il vous reste uniquement à choisir le bouton construire.

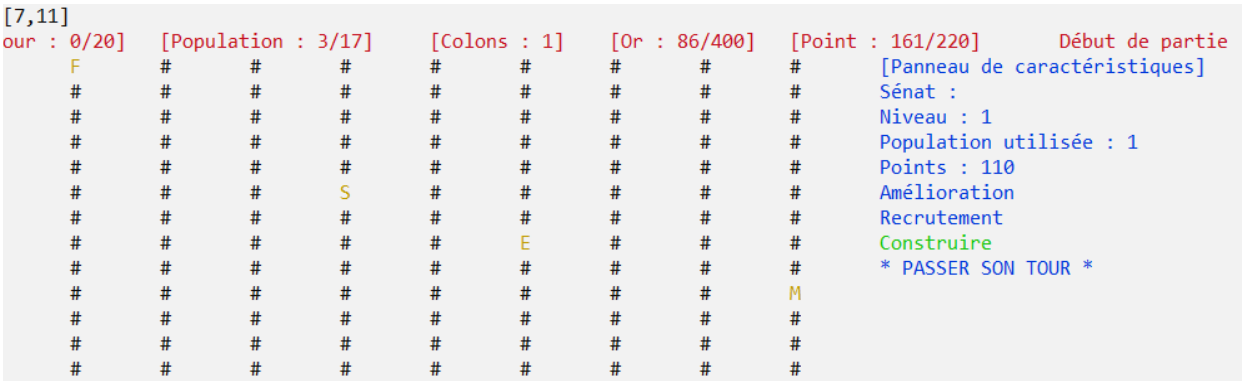


Fig. 7 : Capture d'écran de l'affichage lors de la sélection de Construire

Un pop-up s'affiche alors, vous avez le choix entre 3 bâtiments à construire. Tapez alors 1 ou 2 ou 3.

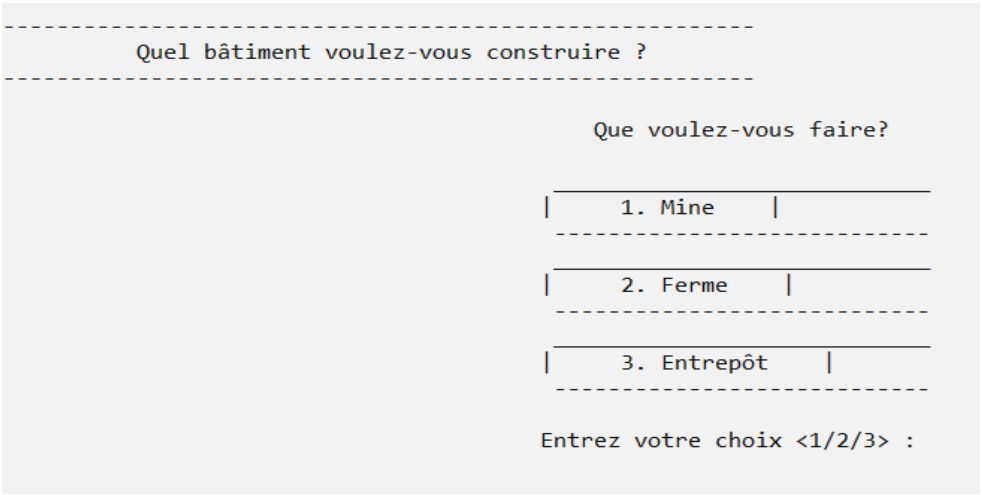


Fig. 8 : Capture d'écran du pop-up de construction

L'onglet de construction du bâtiment choisi s'ouvre. Vous pouvez valider ou non votre choix au vu des ressources nécessaires. Les colons se mettent en route.

3.4.4. Recrutement des unités

Pour recruter des unités, déplacer votre curseur au niveau du Sénat. Appuyer sur la touche "ENTER" puis rendez vous sur la barre latérale des caractéristiques. Quand vous y êtes, il vous reste uniquement à choisir le bouton recrutement.

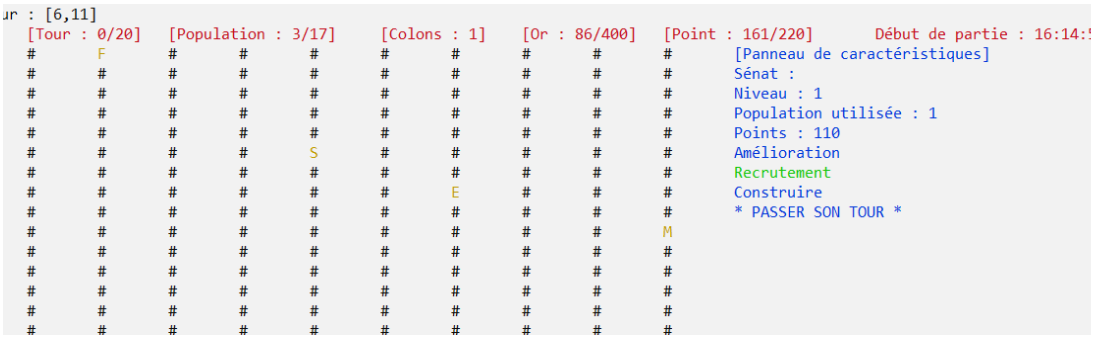


Fig. 9 : Capture d'écran de l'affichage de la sélection de l'action Recrutement

Un pop-up s'affiche alors, vous avez le choix entre recruter des unités ou bien ne rien faire. Tapez alors 1 ou 2.

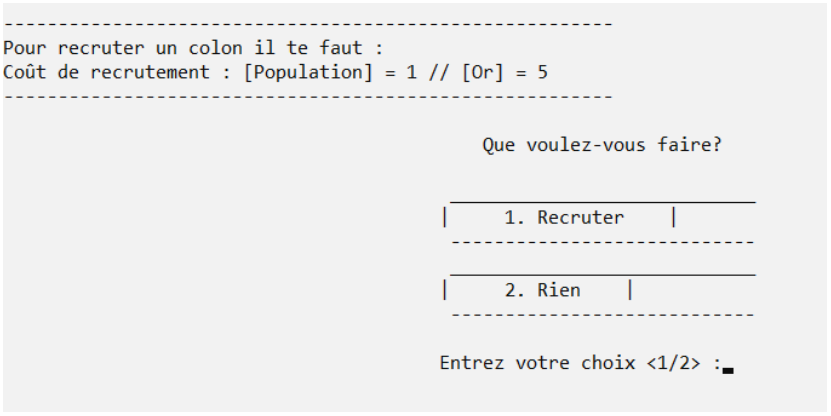


Fig. 10 : Capture d'écran de l'affichage du pop-up de recrutement

Suivant votre choix, vous pouvez entrer le nombre de colons à recruter ou vous êtes redirigé vers l'interface principale du jeu.

3.4.5. Amélioration

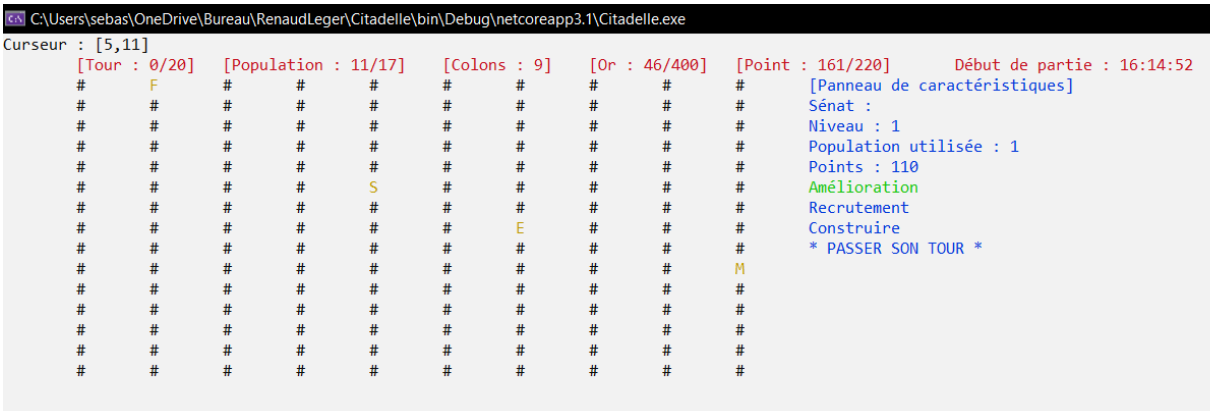


Fig 11. : Capture d'écran de l'affichage de la sélection de l'action Amélioration

Pour améliorer un bâtiment, déplacer votre curseur au niveau du bâtiment souhaité. Appuyer sur la touche "ENTER" puis rendez-vous sur la barre latérale des caractéristiques. Quand vous y êtes, il vous reste uniquement à choisir le bouton amélioration.

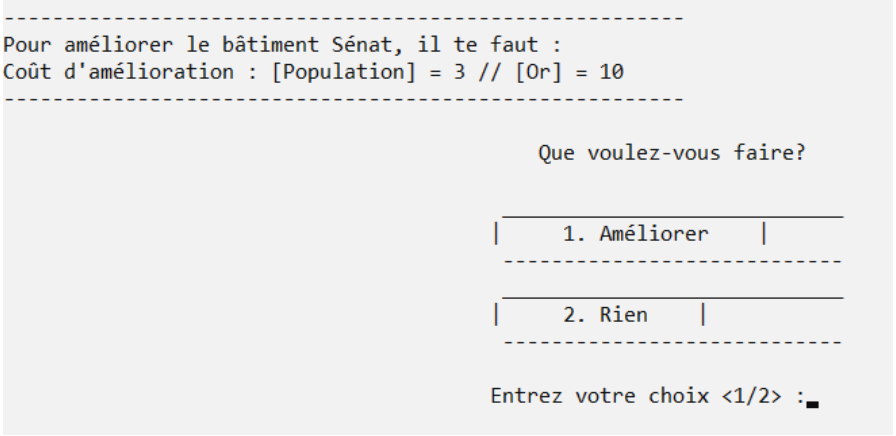


Fig. 12 : Capture d'écran de l'affichage du pop-up d'amélioration

L'onglet d'amélioration du bâtiment choisi s'ouvre. Vous pouvez valider ou non votre choix au vu des ressources nécessaires. Les colons se mettent en route.

4. Choix techniques

4.1. Structure générale du jeu

4.1.1. Structure globale

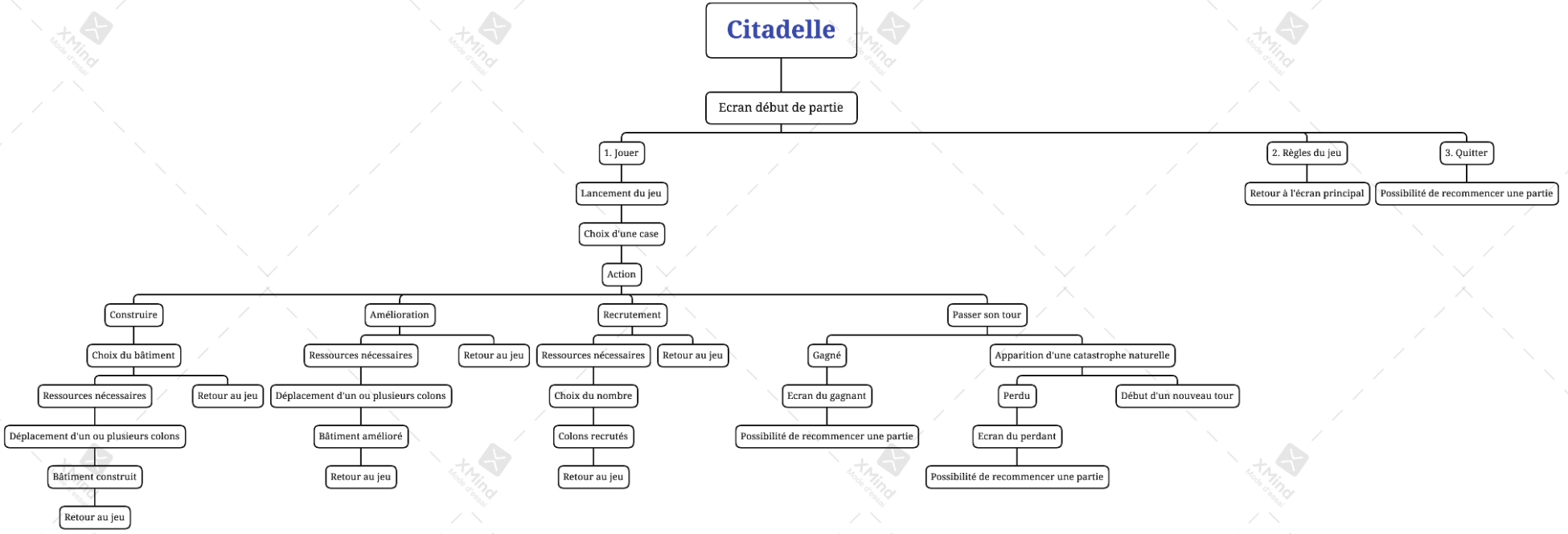


Fig. 13 : Carte mentale de la structure globale du code

4.1.2. Diagrammes de classes

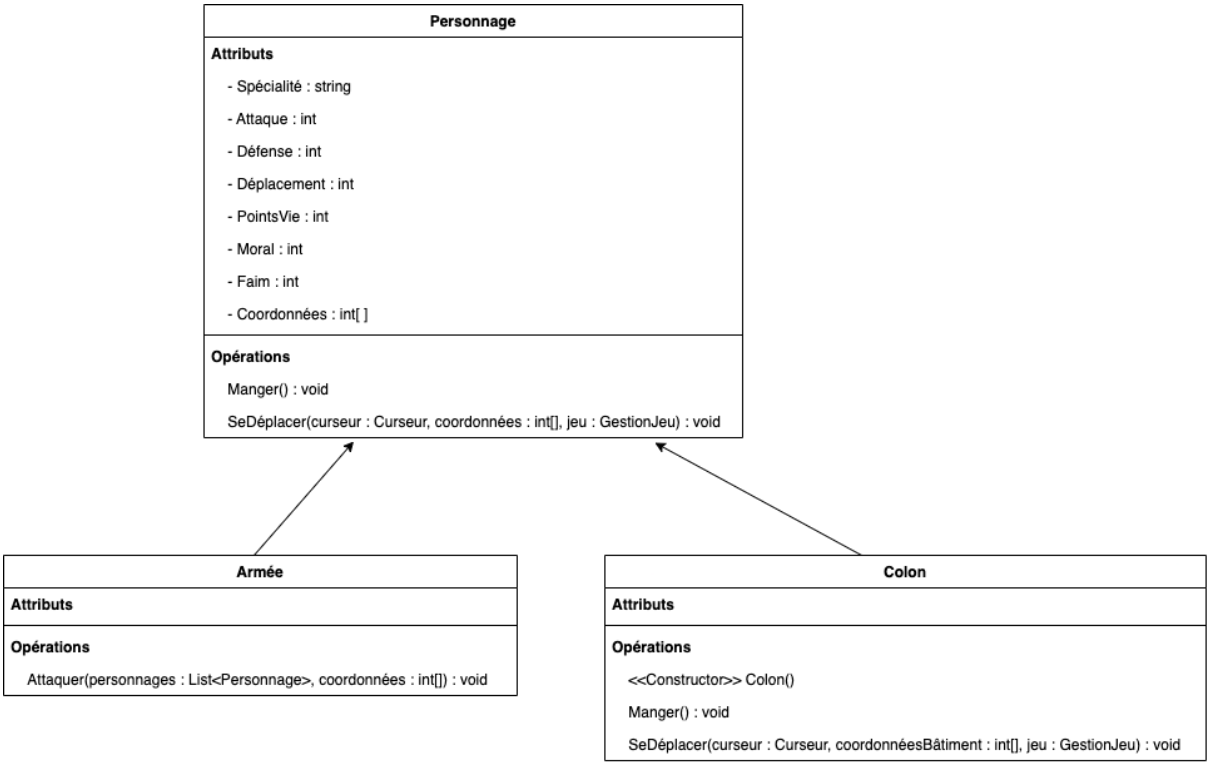


Fig. 14 : Diagramme de la classe Personnage

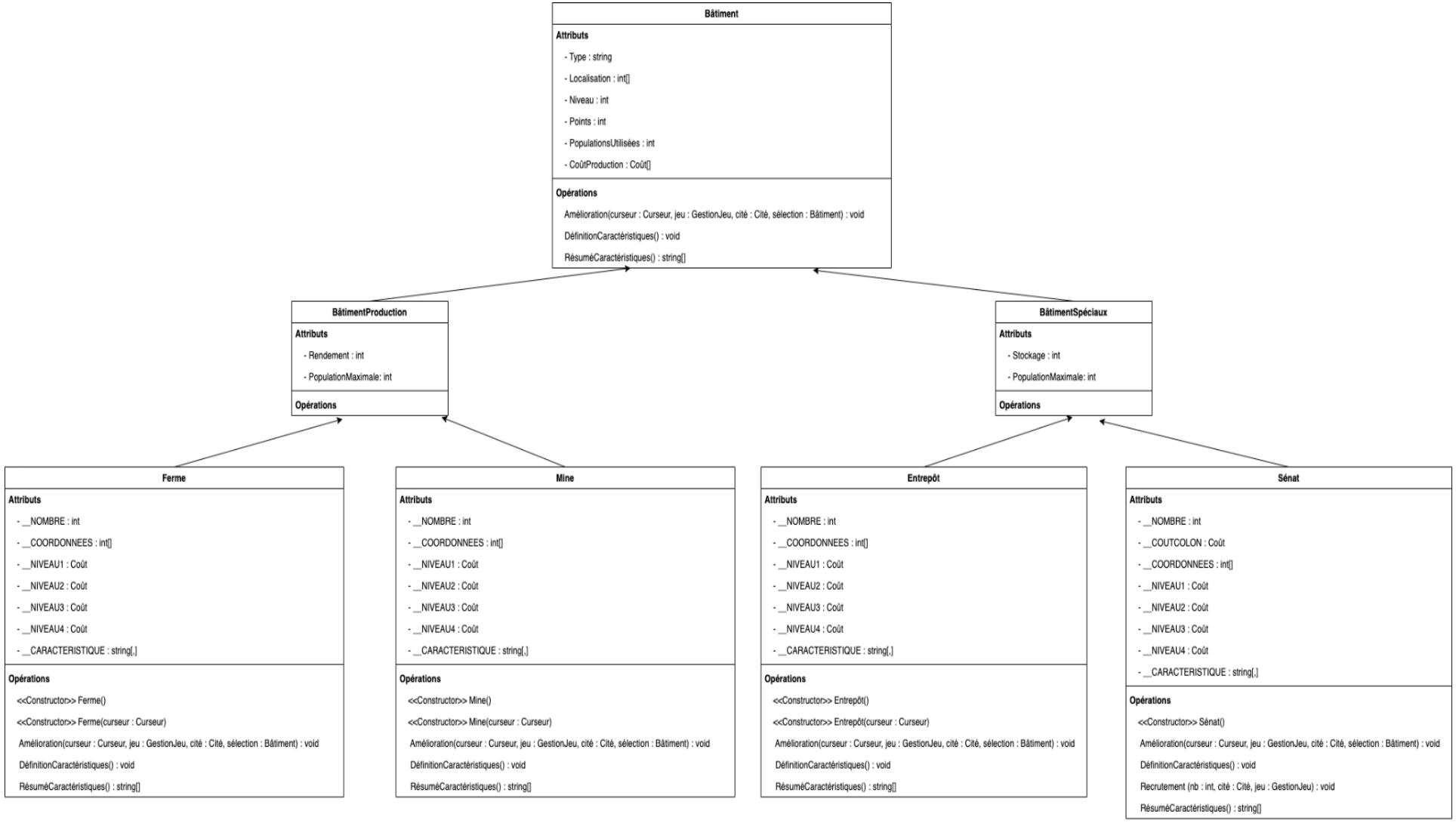


Fig. 15 : Diagramme de la classe Bâtiment

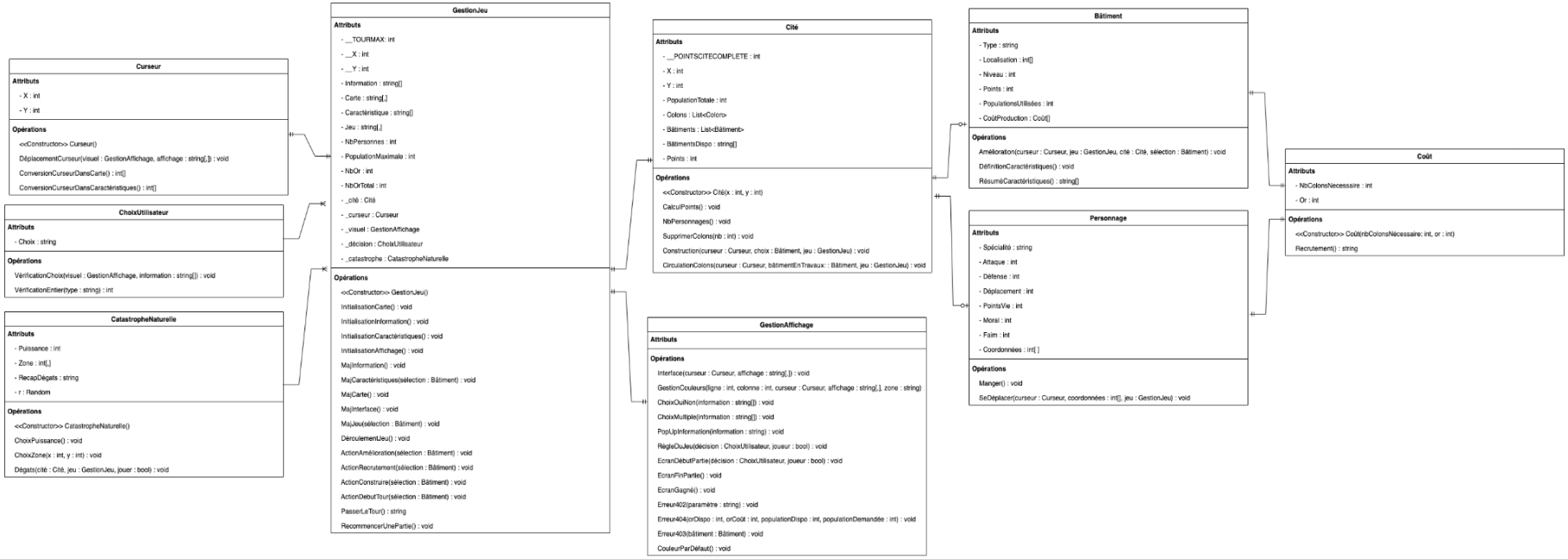
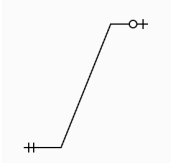
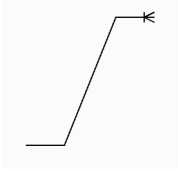
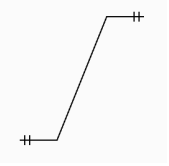


Fig. 16 : Diagramme de relation entre les classes.

Relation d'association	Relation de composition	Relation d'agrégation
		

4.1.3. Explication classe par classe

- **Armée :** Cette spécialisation de la classe Personnage n'est pas implémentée dans la version actuelle. Cette classe hébergera toutes les troupes formant la milice de votre cité tels que des combattants à l'épée, des cavaliers, etc...

- **Bâtiment :** Classe abstract créant le modèle d'un bâtiment. Elle possède deux classes filles, la classe bâtiments de production et la classe bâtiments spéciaux.

- **BâtimentProduction :** Elle hérite de la classe Bâtiment. Ces bâtiments (Mine et Ferme) produisent des ressources. La quantité de ressource produite est stockée dans la propriété rendement.

- **BâtimentsSpéciaux :** Elle hérite de la classe Bâtiment. Ces bâtiments (Entrepôt et Sénat) ont des capacités particulières. Comme recruter des troupes, stocker des ressources. C'est dans cette classe que le bâtiment caserne qui formera l'armée de la cité pourra être implémenté.

- **CatstropheNaturelle :** Classe gérant l'apparition de catastrophe naturelle à chaque fin de tour dans le jeu avec une puissance et une zone touchée variable. Un récapitulatif des dégâts y est intégré.

- **ChoixUtilisateur :** Tous les choix entrés par l'utilisateur doivent être vérifiés pour ajouter de la robustesse au jeu. Ainsi, cette classe va gérer toutes les exceptions liées aux entrées utilisateurs.

- **Cité :** Classe qui va contenir toutes les informations importantes du jeu en stockant les bâtiments et les personnages. C'est la cité qui va gérer le déplacement et la suppression des troupes, la construction des bâtiments et le calcul des points et du nombre de personnes peuplant la cité.

- Colon : Spécialisation de la classe Personnage. Les colons vont se charger de bâtir la cité, et de s'occuper du fonctionnement des bâtiments. En effet, dans chaque bâtiment un nombre de colons sera attribué pour effectuer les fonctions du bâtiment.

- Coût : Classe regroupant le nombre de ressources nécessaires à l'amélioration d'un élément de la cité. Elle prodigue aussi deux méthodes qui transmettent les informations nécessaires à l'amélioration.

- Curseur : La classe curseur permet au joueur d'avoir une jouabilité plus importante en pouvant réaliser l'ensemble des actions en se déplaçant avec les flèches de son clavier.

- Entrepôt : Spécialisation de la classe BâtimentSpéciaux. Les ressources de votre ville sont stockées dans l'entrepôt. Plus le niveau de développement de l'entrepôt est élevé, plus vous pouvez y stocker de ressources. Le niveau de développement maximal de ce bâtiment est le niveau 4.

- Ferme : Spécialisation de la classe BâtimentProduction. La ferme nourrit les colons. Votre ville ne peut pas grandir sans développer votre ferme. Plus le niveau de la ferme est élevé, plus le nombre d'habitants augmente. Le niveau de développement maximal de ce bâtiment est le niveau 4.

- GestionAffichage : Classe gérant uniquement l'affichage, elle va recevoir des données qu'elle va mettre en forme. L'interface avec le choix des couleurs est défini dans cette classe, ainsi que toutes les demandes auxquelles l'utilisateur doit répondre. Cette classe contient uniquement un constructeur implicite. En effet, cette classe nécessite uniquement des méthodes.

- GestionJeu : Classe qui fait tourner le jeu, stocke toutes les informations relatives au jeu.

- Mine : Spécialisation de la classe BâtimentProduction. Dans la mine d'or, vos ouvriers extraient le précieux minerai d'or à partir duquel sont frappées la monnaie de la cité. Plus le niveau de développement de la mine est élevé, plus la quantité de monnaie frappée par tour sera importante. Le niveau de développement maximal de ce bâtiment est le niveau 4.

- Personnage : Cette classe définie en abstract permet de regrouper tous les personnages qui pourront être créés au cours de la partie. Dans la version du jeu actuelle, la diversité des personnages est faible mais cette organisation du code permettra un maintien et une amélioration du jeu de manière efficace.

  Les Propriétés présentent regroupent les caractéristiques générales des personnages :

  - La nature du personnage (colon, combattant à l'épée, etc...)
  - Les caractéristiques du personnage (Attaque, défense, etc...)
  - Les coordonnées

  Les méthodes communes au personnages (pas toutes implémentés dans cette version)

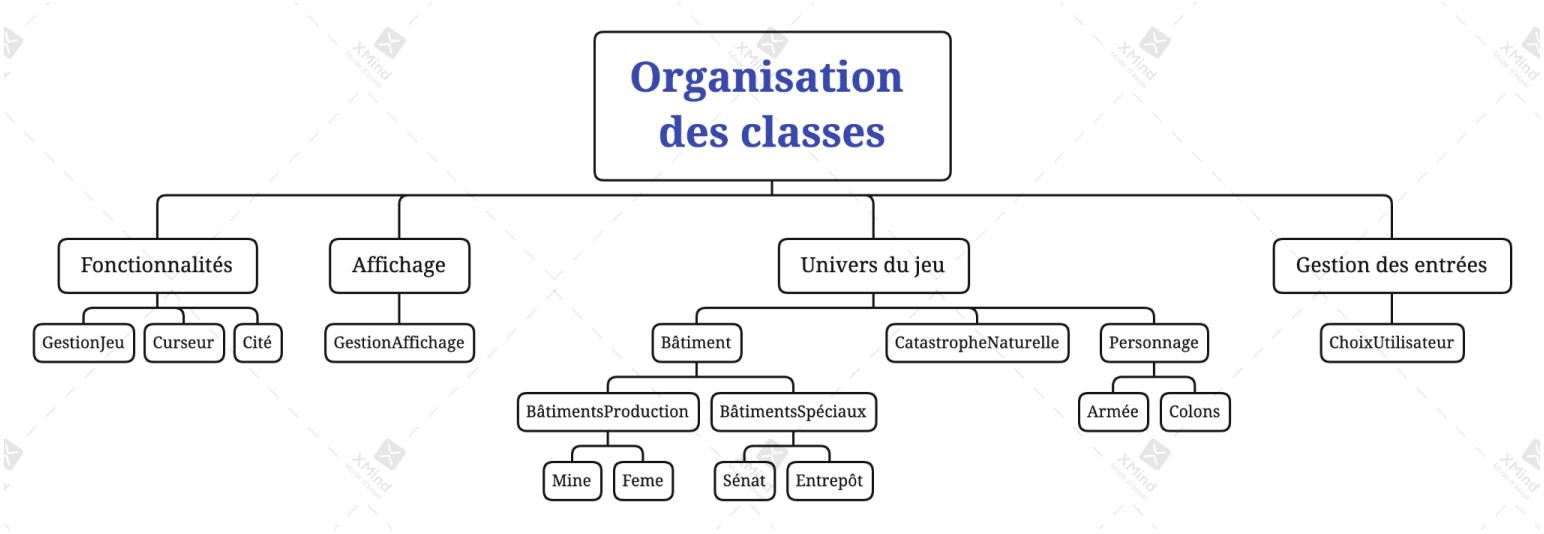
  - Une méthode permettant de manger
  - Une méthode permettant le déplacement de troupe

- Sénat : Spécialisation de la classe BâtimentSpéciaux. Grâce au sénat, vous pouvez construire de nouveaux bâtiments ou agrandir les bâtiments déjà construits. Le Sénat peut stocker une quantité finie d'or et de population (100 et 3 respectivement). Le niveau de développement maximal de ce bâtiment est le niveau 4.

4.2. Catégorisation des classes

Nous avons divisé notre jeu en quatre grandes catégories de classes :

- Nous avons tout d'abord toutes les classes qui vont permettre de gérer les éléments du jeu. C'est dans ces classes que nous avons pu définir tous les éléments que l'on souhaitait retrouver dans notre jeu, comme les bâtiments, les colons et les catastrophes naturelles.
- Nous avons ensuite les classes relatives aux règles et fonctionnements internes au jeu. Avec la définition des coûts, des actions possibles, et si la partie est gagnée ou perdue.
- Nous avons enfin la classe gérant l'affichage du jeu.
- Et finalement une classe qui permet de tester les entrées de l'utilisateur.



-

4.3. Structure détaillée

L'intégralité des fonctions est commentée dans le code, en explicitant leur finalité, les paramètres d'entrée et la sortie attendue.

4.3.1. Tests réalisés

Nous avons sous-évalué et jugé peu digne d'intérêt cette partie du projet en Introduction à la programmation. Ceci nous a valu des heures de débogage de notre code. Le test est un pilier fondamental sans lequel notre projet n'aurait pas eu la qualité attendue. En programmation orientée objet le test des classes implémentées est beaucoup plus simple. Nous avons mis en place deux types de tests unitaires, un test uniquement graphique et un autre numérique. Au début du projet, nous étions sur l'implémentation une par une des classes en les testant numériquement, on vérifiait qu'à la suite de l'appel à une fonction donnée on obtenait bien le résultat attendu. Cette méthode était très frustrante car nous avions l'impression que nous n'avançions pas suffisamment vite et que nous n'arriverions pas à finir le projet. Donc nous avons mis en place l'interface graphique du jeu, ce qui nous permettait de visualiser nos fonctions et donc notre avancement. Nous avons ensuite repris l'implémentation des classes, donc des fonctionnalités du jeu que nous pouvions tester simplement en observant les résultats obtenus.

On a ensuite fini par une grosse phase de tests d'intégration. La particularité de ces tests est que ceux-ci visent à tester la mise en commun de plusieurs composants et l'enchaînement du processus complets de notre jeu.

4.3.3. Propriétés récurrentes

Nom	Type	Description
Colon.Coordonnées ou Bâtiment.Localisation	int[]	*Tableau stockant l'abscisse et l'ordonnée d'un élément du jeu.*
Bâtiment.CoûtProduction	Coût[]	*Tableau stockant le coût en or et population pour chacun des niveaux d'un bâtiment.*
Mine.Rendement	int	*Stocke le nombre d'or produit par la mine.*
Ferme.PopulationMaximale	int	*Stocke le nombre d'habitants que la cité peut accueillir.*
Entrepôt.Stockage	int	*Capacité de stockage de l'entrepôt.*
GestionJeu.Information	string[]	*Tableau contenant les informations de l'inventaire du jeu, tel que le nombre de population, le nombre d'or, etc...*
GestionJeu.Carte	string[,]	*Tableau regroupant tous les bâtiments construits et les personnages.*
GestionJeu.Caractéristique	string[]	*Tableau stockant les caractéristiques de l'élément sélectionné et les actions possibles.*
GestionJeu.Jeu	string[,]	*Tableau qui va rassembler tous les éléments des tableaux susmentionnés. Il nous permettra un affichage par thématique.*
GestionJeu.NbPersonnes	int	*Nombre de personnes peuplant la cité.*
GestionJeu.NbOr	int	*Nombre d'or disponible dans la cité.*
Cité.Colons	List<Colon>	*Liste des colons présents dans la ville.*
Cité.Bâtiments	List<Bâtiment>	*Liste des bâtiments de la cité.*
Cité.Points	int	*Nombre de points de la cité.*

ChoixUtilisateur.Choix	string	*Décision prise par l'utilisateur*
CatastropheNaturelle.Puissance	int	*Nombre de niveau que vont perdre les bâtiment s'ils sont touchés*
CatastropheNaturelle.Zone	int[,]	*Coordonnées en haut à gauche et en bas à droite de la zone touché par la catastrophe*

4.4. Conception Centrée Utilisateur et Utilisabilité

Pour l'esthétique et le développement de ce jeu, nous avons essayé de faire une interface agréable mais qui respecte au maximum la norme ISO 9241-210 développée durant le cours de Mme. LESPINET. En effet, notre jeu se doit d'être accessible, ergonomique, utilisable et attractif. Même si les possibilités de design sont assez limitées pour une application console, nous avons essayé de respecter ces 4 contraintes. Pour les titres des différentes parties du jeu, nous avons utilisé un générateur d'Ascii, nous avons recherché ce qui pour nous était le meilleur compromis entre lisibilité et design. Pour les couleurs utilisées, nous avons respecté le code couleur communément utilisé. Par exemple, nous avons affiché les messages d'erreurs en rouge, les informations en bleu, etc. Enfin, les différentes indications données à l'utilisateur pour entrer un choix sont en noir. Nous avons aussi repris l'interface de la majorité des jeux, avec une carte au centre et des bandeaux d'informations fixes en haut et à droite de la carte. Pour améliorer l'expérience de l'utilisateur nous avons rendu notre jeu robuste ; en effet même si l'utilisateur rentre une information qui ne convient pas (pas le bon type, choix incorrect, …) le programme ne plante pas. Il continue de tourner et affiche à l'utilisateur un message d'erreur lui expliquant assez précisément la source du problème et l'invitant à réitérer sa demande de façon correcte. De plus, nous avons opté pour l'utilisation d'un curseur pour permettre à l'utilisateur de se balader librement sur l'interface sans constamment entrer les coordonnées de la case qu'il veut visiter.

4.5. Fonctionnalités bonus

Au vu du travail à effectuer tout au long du projet, nous avons mis l'accent sur l'objectif minimal attendu. Nous avons aussi veillé à ce que notre jeu soit le plus robuste et avec le moins de bugs possibles. Nous nous sommes aussi attardés à trouver un moyen ergonomique et pratique de jouer à notre jeu (en utilisant la classe Curseur). L'unique fonctionnalité avancée est donc la présence de catastrophes naturelles qui touchent la cité à la fin de chaque tour.

5. Futurs évolutions du jeu

On a vraiment essayé de penser le jeu d'une manière que le maintien et la reprise du code par une personne extérieure soit possible. Durant la période de réflexion sur le sujet et sur les règles du jeu, nous avions vu très large avec un jeu vraiment complet. Donc nos diagrammes de classes sont pensés pour implémenter des améliorations. Par exemple, on pourrait imaginer l'implémentation de l'académie qui héritera de la classe BâtimentSpéciaux ou bien d'une caserne. On peut aussi rajouter des ressources, telles que du bois et de la pierre, qui seraient récoltées respectivement dans une carrière et une scierie. Et enfin, ce qui ajouterait une véritable touche finale à notre jeu, serait l'implémentation d'une armée qui servirait à coloniser d'autre cité ou de se défendre d'un envahisseur extérieur (monstre ou autre armée).

6. Conclusion

Nous avons donc réalisé notre propre version de jeu de gestion de colonie en respectant les règles et/ou consignes imposées.

Ce projet fut plaisant à mener car nous étions assez libres quant au format du jeu, aux améliorations que nous pouvions y apporter et ainsi implémenter des fonctionnalités nous tenant à coeur.

Cependant, mener à bien ce projet ne fut pas de tout repos. Nombreux ont été les doutes, réflexions et difficultés tout au long de l'avancée du projet. En effet, se lancer dans un projet d'une telle ampleur nous a parfois donné un peu le vertige. En parvenir au bout est donc d'autant plus satisfaisant.

D'un point de vue organisationnel, et comme nous nous y attendions, le fonctionnement en binôme s'est avéré efficace. Bien que Thomas ait davantage codé que Sébastien, chacun a mis la main à la pâte et fourni les efforts nécessaires à la réalisation du projet. Si nous avions eu plus de temps, nous aurions voulu intégrer un historique des scores à notre jeu, un moyen de sauvegarder sa partie en cours et aussi et surtout implémenter davantage de classes afin de développer davantage l'univers autour du jeu, les stratégies ou encore sa complexité.