Rapport de Projet JAVA

Hermel Jean-Alexis et GHYS Robin

Sujet : Western

Introduction :

Nous avons choisi de prendre le sujet sur le Western car nous avons trouvé que c'était un sujet très ouvert et possiblement plus créatif que les autres, mais aussi parce que nous avions une petite base dans le cours pour nous lancer et voir ce qu'il était possible de faire grâce au langage JAVA.

Pour débuter nous avons d'abord, avant de coder, mis en place ce que nous voulions qu'il y ait dans notre projet sur papier. Le but étant de savoir où l'on va lorsqu'on commence à écrire du code et de ne pas avancer à l'aveugle. Ainsi nous avons réalisé un premier diagramme de classes (disponible en PDF), peaufiné au fil du temps, qui nous a été très utile pour nous repérer à travers nos différentes classes. Le Western que nous avons créé reflète l'histoire classique qui se déroule dans les films de western, avec des personnages de type cowboy, brigand, banquier, barman, indien… Ainsi il est possible de générer quelques histoires en fonction des différents choix qui s'offrent à l'utilisateur comme nous allons l'expliquer.

Sujet et Description :

Le but du projet est de créer un environnement de classes dans lequel il va être possible de générer des histoires selon les décisions prises par l'appel de méthodes de ces différentes classes. Notre projet s'articule autour d'un genre de village Western, où des lieux vont être créés (Classe Enumération), sur cette base de lieux, nous allons construire des personnages de différents types. Certains auront un lieu attitré, d'autres non. Un personnage verra sa classe héritée d'une ou plusieurs autre classe, en fonction de s'il possède des propriétés communes avec d'autres personnages. Exemple avec le personnage indien qui hérite de la classe Homme (arme, tirer, force…) qui elle-même hérite de la classe Humain (nom, prénom, Position, âge...).

Nous avons fait le choix de nous restreindre dans le nombre et les différences entre les personnages, cela malgré nos nombreuses idées, car sinon le temps nécessaire au projet n'aurait pas suffi pour avoir un programme avec un fonctionnement correct. Finalement nous avons 7 types de personnages pouvant interagir dans 7 lieux différents.

Les personnages pourront être : Banquier, Cowboy, Barman, Indien, Dame, Brigand, Sherif.

Les lieux sont : Le campement, la rue, la prison, la maison, le désert, le bar, la banque.

Un personnage va être décomposé en plusieurs classes (héritées), dont certaines abstraites. Comme dit précédemment les caractéristiques partagées par différents personnages sont réunies dans des classes mère (cf. exemple humain>homme>civil…) afin de factoriser le code. Le barman et le banquier partagent une interface commune, la classe du barman est composée avec une classe Boisson pour

pouvoir gérer et entretenir un stock. Enfin les lieux sont regroupés dans une classe Enumération pour y accéder facilement dans le code.

Le principe de notre projet va être d'instancier des personnages avec leurs caractéristiques propres (cf. diagramme) dans le *main*, pour ensuite pouvoir leur faire effectuer des actions et interagir entre eux. Ceci va avoir la forme ci-contre, pour connaître les options possible pour chaque personnage créé, une méthode *help(personnage)* renvoie chaque action disponible.

```
joe.demanderPret(cresus, 200);
joe.braquer(cresus);
joe.seDeplacer(Position.MAISON);
joe.kidnapper(robinne);
robinne.annoncerPosition();
bill.seDeplacer(Position.RUE);
bill.PayerBoisson(bob, 3);
```

Notre programme est fait de sorte que l'utilisateur ne puisse pas faire ce qu'il veut avec toutes les méthodes, afin de garder un aspect cohérent et logique à la génération de l'histoire. Par exemple si l'utilisateur souhaite commander une boisson avec un personnage, il devra être vivant, et se trouver au bar. Pour cela il devra s'y déplacer et discuter avec le barman en particulier. Nous avons voulu impliquer l'utilisateur dans les choix s'offrant à lui plutôt que simplement appeler une méthode « d'affichage d'information » lorsqu'il effectue une action avec un personnage.

Pour tester le programme, nous avons mit en place un *input* donnant accès à 2 scénarios différents, ce sont des exemples d'utilisation possible des méthodes de chaque personnage. Libre à l'utilisateur d'employer les actions qu'il souhaite dans son scénario, situé dans le *main.*

Ainsi, nous avons dans quelques classes et méthodes introduit le hasard, qui fait varier et augmente considérablement les issues des histoires possibles. Il est envisageable que si, par exemple, le brigand décide de combattre le cowboy, celui-ci réplique et tue le brigand. Et ce à la place d'une méthode qui tuerait automatiquement le personnage qui se fait tirer dessus. Notre programme est donc plus vivant et permet une plus grand durée d'utilisation.

Parmi les différentes méthodes et classes disponibles, certaines sont au cœur du projet et ce sont celles-ci qui vont augmenter les interactions et scénarios possibles :

- La fonction *tirer (Homme)*, agit avec un système de force (0-10) des personnages de type Homme, l'issue du *combat (Homme1, Homme2)* va être décidée par le hasard, pondérée par la force de chaque personnage. L'utilisateur va pouvoir découvrir qu'il ne peut pas faire ce qu'il veut et doit respecter des conditions s'il veut faire combattre des personnages (ex. le cowboy est gentil et ne tire donc que sur les méchants).

- La fonction *négocier (Homme)*, est en lien avec la fonction *tirer* car elle va avoir des effets principalement sur le Cowboy et le Brigand, pouvant négocier avec un l'Indien pour augmenter un peu leur force. Des variantes existent aussi si le cowboy négocie avec le brigand pour en devenir un, ou si le brigand essaie avec le shérif il va en prison.

- Le Barman, qui va avoir un vrai stock de boissons avec quantité et prix va pouvoir en proposer et servir les clients qui lui en demandent via *demanderBoisson (Civil, Boisson).* C'est grâce au système d'argent que nous avons mit en place pour les personnages de type Civil, nous allons le détailler par la suite.

- Un système de « crimes » que va pouvoir commettre le brigand qui vont pouvoir être déjoués/contrés par le cowboy ou le shérif, comme exemple, la Dame va se faire enlever, le cowboy va pouvoir la libérer et le shérif emprisonner le brigand.

- Un système de classe interface pour les personnages commerçants (Banquier et Barman), qui va déclencher une réaction du personnage à la suite d'un braquage par le brigand. Ceux-ci auront une certaine chance de vider leur réserve d'alcool et peut être de tomber en dépression à la suite de ça.

- Enfin un système d'argent pour les personnages de Type Civil. Ceux-ci auront un somme d'argent de base dès leur instanciation, cette somme pourra varier selon ce qu'ils décideront de faire, par exemple tous pourront choisir de donner de l'argent à un autre personnage via *donnerArgent(Civil,argent)* et auront la possibilité de faire un prêt au banquier (donc en se rendant à la banque). Ce prêt pourra être accordé si celui qui le demande rempli quelques conditions sur la somme qu'il demande. Par exemple un personnage ne peut pas demander plus de 45% de ce qu'il a déjà. Ce système rajoute un plus au programme en augmentant les interactions possibles entre personnages, c'est le cas avec les commande de boissons au Barman.

Il y a également d'autres méthodes permettant de faire quelques petites choses comme jouer d'un instrument pour le Barman, se présenter (pour tous les personnages), ou encore s'échapper de prison pour le Brigand.

Toutes ces classes mettent en œuvre un environnement dans lequel l'utilisateur va pouvoir bouger et agir, il reste très basique mais pose les bases d'un jeu vidéo RPG. Nous avons mis en place un système de sauvegarde partie, qui va enregistrer dans un fichier toutes les répliques et commentaire qui se sont déroulés dans le scénario.

Voici un aperçu de ce que le programme renvoi comme scénario :

```
Joe se deplace de Prison a Bar
Voici la carte des boissons :
Leffe au prix de 3€
Wisky au prix de 8€
cognac au prix de 8€
Rhum au prix de 8€
```

```
joe.seDeplacer(Position.BAR);
joe.demanderCarte(bob);
joe.demanderBoisson(bob, leffe);
joe.demanderBoisson(bob, wisky);
```

```
Joe Dalton : Un Leffe siou'plaît m'sieur !
Bob Sauler : Je n'ai plus de Leffe voulez-vous autre chose ?
Joe Dalton : Ah ! c'est plus ce que c'était ce bar ! je vais regarder.
Joe Dalton : Un Wisky siou'plaît m'sieur !
Bob Sauler : Vous avez de la chance, il m'en reste 10
Bob Sauler : Et voici votre boisson, un bon Wisky, ça vous fera 8€
Joe Dalton : Tennez, voilà 8€.
L'argent de Bob passe de 1500 à 1508
L'argent de Joe passe de 400 à 392
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
joe.seDeplacer(Position.MAISON);
joe.kidnapper(robinne);
rody.seDeplacer(Position.DESERT);
joe.annoncerArme();
combat(rody, joe);
rody.sauverDame(robinne);
afficherCimmetiere();
```

```
Joe se deplace de Prison a Maison
Joe Dalton : Viens la ma jolie Dame Robinne !
Robinne Ghyse : A L'aiiiide, on me kidnappe !!
Dame Robinne est enlevee jusqu'au désert par Joe

Rody se déplace de Campement à Desert
Joe Dalton : Attention ! Je suis en possession d'un revolver
Rody a voulu tirer sur Joe, qui s'est defendu, à l'issue du combat Joe est mort !
Rody Allen : Me voici dame Robinne! Je viens vous sauver !
Robinne Ghyse : Oh, quel heros, Rody a reussi à me liberer de cette ordure, je ne l'en remercierai jamais assez
Robinne retourne à sa maison.

Liste des personnes au cimmetiere :
Joe
```

Travail effectué :

Concernant le travail commun : nous avons élaboré le plan du projet et son diagramme de classes ensemble pour être sûr que chacun soit d'accord sur la conduite à adopter. Ensuite, lorsque nous avons commencé à coder nous nous sommes réparti le travail en deux concernant l'écriture des *getters* et *setters* de chaque classe, chose assez répétitive. Il en va de même pour l'écriture des constructeurs, nous avons pour ça décider de l'ordre des paramètres de ceux-ci pour s'y retrouver plus facilement. Enfin nous avons mis en place le système de force pour avoir des combats qui ne soient pas toujours prédéterminés, puis déterminé les conditions que l'utilisateur devait remplir s'il voulait utiliser certaines actions d'un personnages. Comme exemple cité précédemment, se trouver au même endroit pour interagir et être en vie. Il est possible de retrouver l'intégralité de l'histoire, générée dans un fichier .txt (à ouvrir sous WordPad), et enregistrée dans le même dossier que le dossier *src,* ceci permet de conserver une trace, « une sauvegarde » de la partie jouée.

Concernant Jean-Alexis :

J'ai pu réaliser le système d'argent et tout ce qui va s'y allier, c'est-à-dire l'écriture des méthodes de la classe Barman et Banquier, le système de prêts, la classe Boisson, la classe PeutBoireBar. Mon objectif était que cette gestion d'argent des personnages se rapproche de ce qu'il peut se faire dans les jeux vidéo mais en restant assez simple. Nous aurions pu développer d'autres méthodes similaire d'achats/vente mais elles n'auraient rien apporté de plus au contenu du projet en termes de richesse de code.

J'ai eu l'occasion de concevoir, une fois l'idée mise en place, le système de combats entre personnages, celui reste basique mais est fonctionnel, par exemple les personnages ont une arme mais la nature de celle-ci n'influe pas sur le résultat du combat. La force et le hasard décideront de l'issue. Pour ajouter des fonctionnalités à cette méthode *tirer* j'ai également conçu la méthode *combat* et *afficherCimmetière* pour simplifier la lecture du jeu à l'utilisateur.

Par la suite j'ai réalisé l'interface *SestFaitFaucher* qui intervient lors d'un braquage du brigand, le but ici était d'utiliser le principe des interfaces pour appliquer un même comportement à des classes qui n'ont pas le même héritage. Ici, après avoir écrit les méthodes concernant le braquage (*seFaireBraquer*, *braquer*), j'ai codé celles conséquentes au braquage, *videReserveAlcool* et *depression* toujours avec le hasard qui vient augmenter les possibilités de scénario. Par exemple le fait de vider la réserve d'alcool peut entrainer une dépression du personnage (avec plus de chance s'il s'est fait braquer juste avant).

Enfin j'ai rajouté des exceptions dans le code pour prévenir l'utilisateur d'une mauvaise saisie, notamment pour un âge ou une somme d'argent négative. Ceci avec l'utilisation de blocs *Try* et *Catch*.

Concernant Robin :

En début de projet et après la création du diagramme de classe, je me suis chargé de compléter les squelettes de classes créées avec les différentes méthodes que nous avions déterminées auparavant, sans forcément les implémenter.

Une fois tous les squelettes de classe créés, je me suis chargé d'implémenter les méthodes de différentes classes dans le détail, notamment les classes Dame, Brigand ou encore Shérif.

Et c'est en implémentant ces différentes méthodes que je me suis rendu compte de la complexité des fonctions et des différents cas qui pouvaient se présenter lors de l'appel d'une fonction. En effet, le résultat, le texte affiché dans la console ou encore les conséquences directes dans le jeu pouvaient être différentes selon le type de personnage (Cow-boy, Indien, Brigand …) qui était à l'origine de l'appel d'une fonction. C'est pourquoi j'ai décidé par la suite d'étudier tous les différents cas qui pouvaient se présenter lors de l'appel d'une fonction, et cela m'a en effet pris un certain temps.

Traiter tous les cas était indispensable pour éviter de mauvaises surprises et pouvoir anticiper les choix des utilisateurs du programme et éviter les « choses absurdes » dans nos histoires. L'étude des différents cas que j'ai pu mener est nettement observable notamment dans les méthodes de *tirer()* et *négocier()* disponibles dans la classe Humain.

Nous souhaitions également conserver une trace de l'histoire générée et avons donc décidé d'enregistrer celle-ci dans un fichier .txt. Bien que cela ne fut pas compliqué en termes de difficulté, cela pris à nouveau du temps car il m'obligea à reparcourir le code du début à la fin pour faire appel à ma fonction *ecrireFichier()* (également disponible dans la classe Humain.)

Difficultés rencontrées :

Concernant les difficultés que nous avons rencontrées, les premières furent dès la conception du diagramme de classes, nous avions nos personnages en tête mais il fallut bien réfléchir à comment factoriser, via des héritages, tous les attributs communs à chacun. Nous avons réglé ce problème en créant des classes telles que Humain, Homme, le seul bémol reste que la femme ne peut pas bénéficier des attributs et méthodes de la classe Civil dans notre système actuel (voir diagramme).

Vient ensuite un petit soucis concernant l'usage de l'attribut force, comment rendre notre projet moins prédictible et plus axé sur des « rebondissements ». Nous avons donc pour cela fait appel au hasard qui prend une grande place dans notre simulation de Western car c'est grâce à lui que les scénarios sont plus variés qu'ils ne le seraient s'ils n'étaient basés que sur un simple modèle action/conséquence.

Ensuite nous aurions aimé concevoir une méthode permettant d'instancier un type de personnage choisi avec son nom, du type *creerCowboy('rody',24,'Revolver'…)* mais le problème, non résolu, est que dans ce cas, on ne peut pas donner le nom du personnage à l'objet lui-même, tel qu'on le fait lorsqu'on en créé un dans le main. Par exemple *bill = new Cowboy ('bill'…)* ne fonctionnera pas, on devra au mieux se contenter d'un numéro dans une liste de personnages, ce qui rend l'utilisation des méthodes moins compréhensible. On ne pourrait plus faire *bill.negocier(bob)* mais *persos[1].negocier(persos[2])* qui n'est pas ce que l'on souhaite.

Au terme du projet, nous voulions que l'utilisateur puisse, via une interface type *scanner,* pouvoir créer un personnage en rentrant ses caractéristique, mais nous nous sommes heurtés au problème de la *reflection* en java et du *invok* que nous n'avons pas su gérer finalement.

Enfin un dernier petit problème fut la façon dont on allait ajouter un stock de boissons au barman. Au début nous avions comme idée de créer une ArrayList de ArrayList mais on ne peut pas faire de Tuple en Java comparé à Python. C'est pourquoi nous avons créer une classe Boisson, possédant des attributs comme quantité, type, prix, pour ajouter une boisson il suffit de l'instancier dans le main et de l'ajouter au Barman. Cette solution permet une meilleure flexibilité et utilise le principe même de Java à savoir créer des classes d'objets.

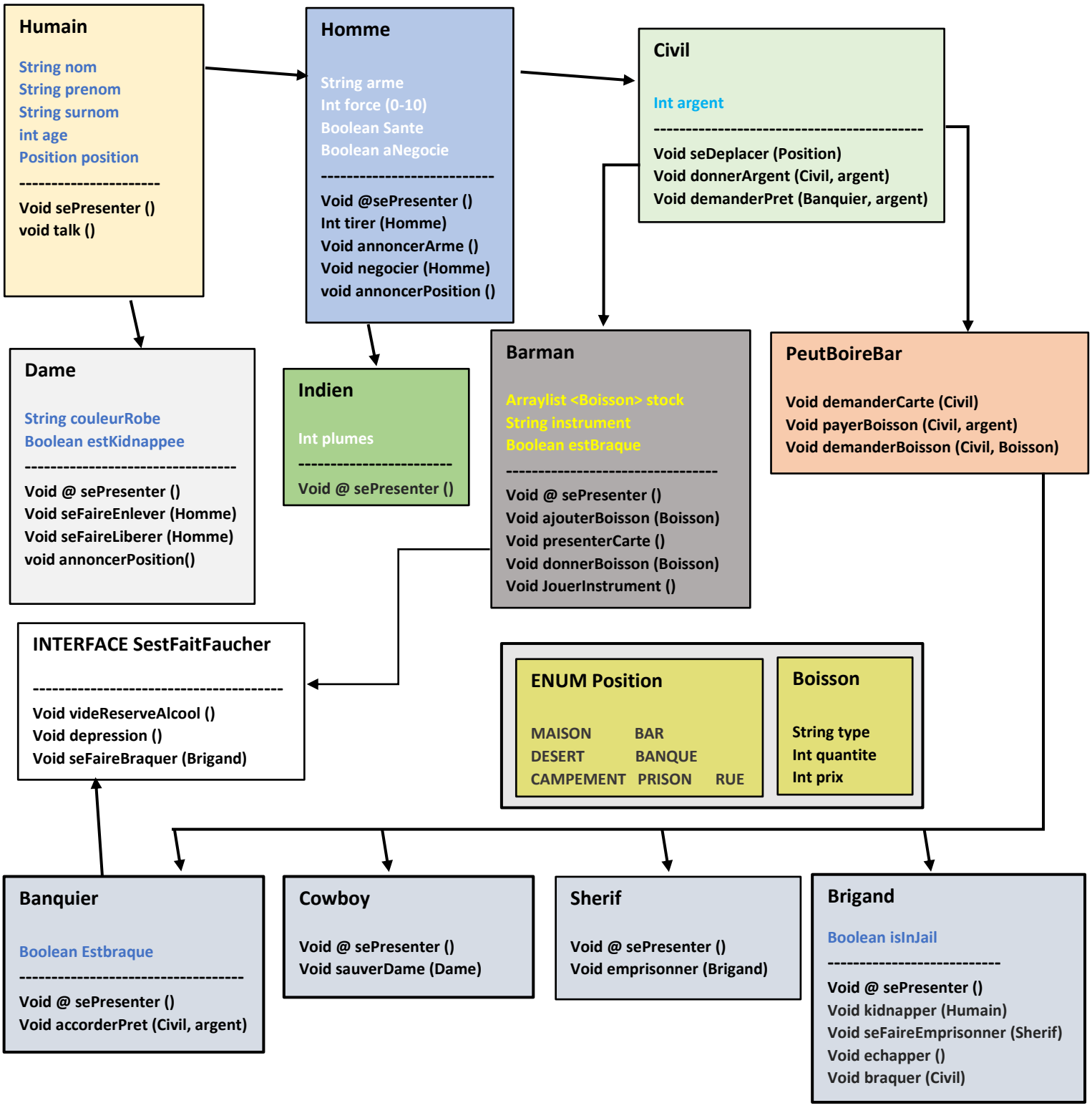
Conclusion :

Pour conclure, nous sommes contents du résultat obtenu, et des scénarios en tant que tels produits par notre jeu. Ce projet fut pour nous l'occasion de revoir l'ensemble du cours de Java et de le mettre en application, avec plus ou moins de difficultés. Le fait de d'avoir choisi un sujet avec plus de libertés nous a forcé à mettre de la rigueur dans ce que l'ont faisait dès le début. C'est grâce à cela que nous n'avons pas vraiment eu de gros soucis de code une fois que nous avons commencé à nous plonger dans l'écriture. Les concepts de classe abstraite et d'héritage sont au cœur de notre projet Western et donc bien acquises. Il a parfois été nécessaire de contourner des choix de jeu que nous avions du fait de la difficulté ou l'infaisabilité des idées mais dans l'ensemble nous avons réussi à respecter le cahier des charges fixé au début, surtout concernant la partie combats et argent. Pour terminer, nous sommes conscients que le jeu n'est pas vraiment jouable comme un jeu normal mais c'est justement un des points d'amélioration que nous avons en tête si on a l'occasion de reprendre le projet. Malgré le bilan positif que nous tirons de ce projet, nous sommes tout de même déçus de ne pas pouvoir vous présenter une version de notre projet avec une interface Homme Machine qui apporterait beaucoup plus de liberté à l'utilisateur.

HERMEL Jean-Alexis

WESTERN

GHYS Robin



TOUS LES CONSTRUCTEURS PEUVENT AVOIR OU NON UNE POSITION (après Age)

New Dame (Nom, Prenom, Surnom, Age, CouleurRobe)
New Cowboy (Nom, Prenom, Surnom, Position, Age, Arme, Force, Argent)
New Banquier (Nom, Prenom, Surnom, Position, Age, Arme, Force, Argent)
New Brigand (Nom, Prenom, Surnom, Position, Age, Arme, Force, Argent)
New Barman (Nom, Prenom, Surnom, Position, Age, Arme, Force, Argent, Instrument)
New Indien (Nom, Prenom, Surnom, Position, Age, Arme, Force, Plumes)
New Sherif (Nom, Prenom, Surnom, Position, Age, Arme, Force, Argent)

##############################----DOCUMENTATION----################################

Généralités :

-help(humain)
- un personnage ne peut pas interagir s'il ne se trouve pas au même endroit que l'autre personnage
- un personnage ne peut interagir avec lui-même
- un personnage doit être vivant pour faire une action
- la force d'un personnage se trouve entre 0-10, 9 correspond à l'invincibilité - un combat entre deux personnages se fait avec la fonction combat(homme, homme)
- a l'issue d'un combat, la personne tuée va au cimetière, on peut voir les personnes au cimetière avec afficherCimmetiere().
- On peut faire renaitre un perso avec jesusDeNazareth(perso).
- un personnage crée sans position spawn dans la RUE.
- Un civil qui commande peut commander(et payer en même temps) sa boisson, tout dépend de si toutes les conditions pour qu'ils puissent boire soient remplies.
- les personnages Civils peuvent se donner de l'argent.
- un personnage ne peut négocier avec un autre qu'une seule fois
- Tous les personnages peuvent annoncer leur position

Le barman :

Il possède nom, prénom, surnom, Age, position, arme, force, argent, stock de boissons, instrument.
Sa position de base est le BAR
Il peut se présenter, se déplacer, il ne sait pas tirer ni négocier
Il peut donner de l'argent et faire un prêt au banquier
Il peut ajouter des boissons a son stock, présenter sa carte, donner une boisson, jouer un instrument il peut se faire braquer et perdre tout son argent, dans ce cas il peut vider sa réserve d'alcool et a plus de chance de faire une depression
Il peut décider de vider sa réserve d'alcool et a une faible chance de faire une depression

L'indien :

Il possède nom, prénom, surnom, Age, position, arme, force, argent, plumes
Sa position est le CAMPEMENT
Il peut se présenter, il reste au campement, il ne peut tirer que sur un cowboy ou brigand, il ne peut initier une négociation
Le nombre de plumes d'un indien varie entre 1 et 20, son pouvoir d'augmentation de force est fonction de son nombre de plumes (+1 force si plume <13, +2 sinon), il y a 30% de chance que la négociation échoue.

La Dame :

Elle possède un nom, prénom, surnom, âge, position, couleur de robe
Sa position de base est la MAISON
Elle peut se présenter, se faire enlever par un brigand, se faire libérer par un homme, et se faire sauver par un cowboy ou shérif

Le Brigand :

Il possède nom, prénom, surnom, âge, position, arme, force, argent
Sa position de base est le DESERT
Il peut se présenter, se déplacer, il peut tirer sur n'importe quel homme
il peut donner demander un prêt au banquier, il peut donner de l'argent a un civil
il peut demander la carte au barman, commander/payer une boisson
il peut négocier une seule fois, avec un indien pour augmenter sa force, ou un shérif (il ira en prison)
Il peut kidnapper une dame, braquer un banquier ou barman (récupère 30% du butin), se faire emprisonner par le shérif (va en prison), peut s'echapper de prison (se retrouve a RUE)

Le Banquier :

Il possède nom, prénom, surnom, âge, position, arme, force, argent
Sa position de base est la BANQUE
Il peut se présenter, se déplacer, il ne sait pas tirer ni négocier
il peut accorder un prêt a un civil (sous certaines conditions)
il peut donner demander un prêt au banquier, il peut donner de l'argent a un civil
il peut demander la carte au barman, commander/payer une boisson
il peut se faire braquer et perdre tout son argent, dans ce cas il peut vider sa réserve d'alcool et a plus de chance de faire une depression
Il peut décider de vider sa réserve d'alcool et a une faible chance de faire une depression

Le shérif :

Il possède nom, prénom, surnom, âge, position, arme, force, argent
Sa position de base est la PRISON
Il peut se présenter, se déplacer, il ne peut tirer que sur un brigand, il ne sait pas négocier
il peut donner demander un prêt au banquier, il peut donner de l'argent a un civil
il peut demander la carte au barman, commander/payer une boisson
Il peut emprisonner un brigand

Le cowboy :

Il possède nom, prénom, surnom, âge, position, arme, force, argent
Sa position de base est la RUE
Il peut se présenter, se déplacer, il ne peut tirer que sur un brigand, il peut négocier avec l'indien pour augmenter sa force
il peut donner demander un prêt au banquier, il peut donner de l'argent a un civil
il peut demander la carte au barman, commander/payer une boisson
Il peut sauver une Dame