

Portrait Florian

Raad Josué

Projet IHM

Rapport de projet IHM jeu d'exploration

The Shadow Lair



Université de Poitiers - L3 informatique - Programmation des
interfaces homme-machine

Année universitaire 2020 - 2021

Sommaire

Sommaire	2
Présentation du projet	3
Comment lancer le jeu :	3
Présentation de l'interface	3
Les boutons	4
Les onglets	5
Les champs texte	5
Design de l'application	5
Le modèle	6
La vue	6
Contrôleur	6
Contrôleurs de la vue	6
Contrôleurs de l'application	6
Structure de l'application	11
Résolutions des problèmes rencontrés	14
Améliorations possibles pour l'application rêvée	14



Présentation du projet

Ce projet poursuit le projet de Programmation Orientée Objet que nous avons réalisé en 2020. Il s'agit de reprendre la même base de code et de réaliser une interface graphique à partir de ce modèle.

Nous avons créé un jeu d'exploration et cette nouvelle version propose une version améliorée reposant sur les mêmes comptes.

Il s'agit d'une petite aventure au sein d'un sinistre donjon, où notre héros va devoir survivre et terrasser toutes sortes d'adversaires pour tenter de venir à bout du mal qui occupe les lieux et libérer les pauvres âmes sous son emprise.

Comment lancer le jeu :

Pour exécuter le jeu : se placer au niveau du dossier source et exécuter :

```
javac shadowLair/ShadowLairApplication.java
```

Ensuite, exécuter : `java shadowLair.ShadowLairApplication`.

Le projet a été testé sous java 1.8 avec javaFX inclus.

Pour bénéficier des polices personnalisées, une connexion en ligne est requise au moment de lancer le jeu. Sinon ce n'est pas bien grave : la police par défaut sera utilisée.

Présentation de l'interface

Notre interface se décompose en plusieurs panneaux.

Chacune des commandes du modèle textuel a été intégrée à l'interface et les descriptions textuelles sont toutes présentes.

Parmi les composants nous trouvons entre autre :

Les boutons

Attack : Il vous est possible d'attaquer tous les pnj du jeu (hostile/agressif/passif), pour cela vous devez sélectionner le pnj souhaité et cliquer sur Attack.

Talk : Vous avez la possibilité de discuter avec les pnj passifs, pour cela sélectionnez le pnj en question et cliquez sur Talk.

Look : Le bouton est disponible lorsque vous sélectionnez un pnj ou un coffre, il permet d'avoir une description de l'entité en question.

Take : Lorsque l'onglet Loot est actif, vous pouvez récupérer les objets disponibles, il vous suffit de sélectionner l'objet et de cliquer sur Take pour le récupérer.

Buy : Lorsque vous êtes dans l'onglet trade avec un pnj vous pouvez acheter les objets se trouvant dans son inventaire (moyennant des pièces d'or). Il vous faut sélectionner l'objet qui vous intéresse et cliquer sur Buy.

Sell : Lorsque vous êtes dans l'onglet trade avec un pnj vous pouvez vendre les objets se trouvant dans votre inventaire (si le pnj en question a assez d'or). Il vous faut sélectionner l'objet que vous souhaitez vendre et cliquer sur Sell.

Use : Use permet d'utiliser les objets de votre inventaire, il vous faut sélectionner l'objet à utiliser et cliquer sur Use. Certains objets s'utilisent avec ou sur d'autres objets, par exemple une clé avec une porte, ou une flèche avec un arc, pour cela il vous faut sélectionner l'objet, à utiliser, sélectionnez ensuite l'objet ou la porte qui correspond dans le menu déroulant "**Use on**", et enfin cliquer sur Use pour l'utiliser.

Drop : Il vous permet de jeter un objet de votre inventaire (celui-ci se retrouvera sur le sol de la salle actuelle). Pour cela vous devez sélectionner un objet dans votre inventaire et ensuite cliquer sur Drop.

Equip : Vous permet d'équiper une arme de votre inventaire (veillez à bien Use vos flèches sur l'arc avant de l'équiper). Pour cela vous devez sélectionner une arme dans votre inventaire et cliquer sur Equip.

Unequip : Vous permet de déséquiper votre arme, il vous suffit de cliquer sur Unequip.

Look Place : Cela vous affiche dans "Information" la description de la salle actuelle.

Loot Place : Cela vous ouvre l'onglet Loot, qui affiche les objets se trouvant au sol dans la salle actuelle.

Quit Game : Vous permet de quitter l'application.

Help : Ouvre une nouvelle fenêtre vous proposant un visuel et récapitulatif du fonctionnement de l'interface.(voir image dessous*)

Boutons des salles : Vous pouvez trouver jusqu'à 4 boutons de direction dans la salle courante, s'il s'agit d'une simple porte il vous suffit de cliquer sur le bouton de direction pour

changer de salle. Si le bouton est coloré, c'est qu'il s'agit d'une porte verrouillée et qu'il vous faut alors trouver une clé de même couleur pour la déverrouiller.

Les onglets

Inventory : C'est ici que vous retrouverez les items que vous allez récolter tout au long de la partie, l'inventaire est tout le temps visible. Vous pouvez y voir vos pièces d'or, le type d'objets possédés, leur nom, valeurs et poids.

Trade : Cet onglet s'activera seulement après une interaction avec un pnj passif, il permet d'acheter ou vendre des items à celui-ci.

Loot : Cette onglet s'activera seulement après une interaction avec un coffre ou en Lootant la salle actuelle. Les objets que vous jetez iront ici.

Les champs texte

Dialogue : Ici s'affiche le dialogue avec les pnj.

Information : Ici s'affiche la description des salles ou pnj, ainsi que les erreurs (si vous utilisez un objet sur un mauvais élément, ou si vous n'avez plus de places dans votre inventaire par exemple).

Description : Lorsque vous sélectionnez un objet dans votre inventaire, dans l'onglet Trade ou l'onglet Loot, sa description apparaît ici (puissance, vie rendue, ...).

Nom de la salle : Au-dessus de la salle courante, s'affiche le nom de celle-ci, sympa pour se repérer.

Barre de personnage : Vous pouvez voir au dessus du champ de dialogue votre barre de personnage, celle-ci indique votre niveau, vos pièces d'or, votre état de santé, l'arme actuellement équipé ainsi que vos dégâts, et le poids total des objets que vous posséder avec la valeur max qu'ils vous est possible de porter.

Les pnj ont également une barre de personnage, celle-ci apparaît lorsque vous sélectionnez un pnj.

Avant de passer au code nous avons d'abord travaillé et pensé des maquettes afin d'imaginer une interface pour notre futur jeu.

Design de l'application

Dans un premier temps nous avons commencé par régler les soucis qui étaient présents au niveau du modèle. Quelques bugs nous sont apparus et leur résolution était la plupart du temps trivial avec un recul.

Une fois le noyau fonctionnel opérationnel nous avons commencé à designer des mockups d'interface à l'aide de logiciels de dessins. Cette approche nous a permis de visualiser plus aisément comment sera construite notre interface par la suite.

Également cela a aidé à nous rendre rapidement compte des défauts de notre interface et de les corriger avant même de commencer à écrire la moindre ligne de code.

Le modèle

Notre projet POO s'apparente déjà à un modèle MVC. En effet nous avons de classes responsables de l'affichage (Printer.java) et d'autres responsables de l'exécution (Execute.java) des commandes utilisateur. Celles-ci faisaient la liaison avec le modèle à proprement dit. Ainsi pour adapter notre projet de POO à javaFX, nous avons simplement désactivé l'affichage dans le Printer (seul les erreurs restent), et nous envoyons à l'exécuteur des commandes afin d'interagir avec le modèle.

Le reste du code est quasiment identique, hormis quelques assesseurs supplémentaires et la récupération de Strings en résultat des commandes exécutées.

Après que l'application soit devenue complètement fonctionnelle, nous avons ajouté beaucoup de contenu dans le jeu, de nouvelles salles, monstres, objets etc ...

La vue

La vue se compose exclusivement de fichiers FXML, à l'exception des popups qui ont leur propre classe et contiennent uniquement une image avec une action.

La plupart de ces fichiers FXML ont été créés avec SceneBuilder. Mais nous avons trouvé qu'il était plus efficace de créer un brouillon rapide avec cet outil, puis de raffiner à la main le code produit ensuite.

Contrôleur

Contrôleurs de la vue

Presque chaque fichier de FXML de la vue est associé à un contrôleur qui lui correspond.

Nous avons mis en place un héritage simple sur les contrôleurs de la vue, mais nous ne l'avons que peu utilisé au final.

Les contrôleurs de la vue se contentent de recevoir des signaux de mise à jour de leur interface depuis les contrôleurs de l'application. Et si besoin ils notifient également les contrôleurs de l'application d'un changement interne (une sélection d'un objet de la part de l'utilisateur par exemple)

Contrôleurs de l'application

Avec le travail précédemment établi le rôle du contrôleur de l'application est relativement simple, il doit :

- Lors de l'appui sur un bouton, Générer une commande correspondant à l'état de l'interface et au bouton (par exemple "GO BARN")

Ainsi le contrôleur de l'application se décompose en deux fichiers :

- “MainController” est chargé de la récupération de l’état de l’interface et de sa mise à jour
- “ExecutionController” est chargé de l’exécution de commandes à la demande

Ces deux classes sont statiques afin de gérer plus simplement la communication au sein de l'application.

Première interface envisagée

Ruined Temple

Look place

Loot place

Quit

Forest

Barn

	You	golds	Life <div><div></div></div>	equiped (5 dmg)	15 --- 17
	Scandlaf	golds	Life <div><div></div></div>	attaque	

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla quam velit, vulputate eu pharetra nec, mattis ac neque. Duis vulputate commodo lectus.

Inventory

trade

loot

▲ Golds : 5

🗝 key

⚔ weapon

🍖 food

Label

Label

Label

▼

Gold : 3

Label

Label

☒ Label

Label

Label

Label

Label

Label

Sell ->

<- Buy

Drop

Describe itm

What to do with Scandlaf :

Talk

Attack

Look

Description:

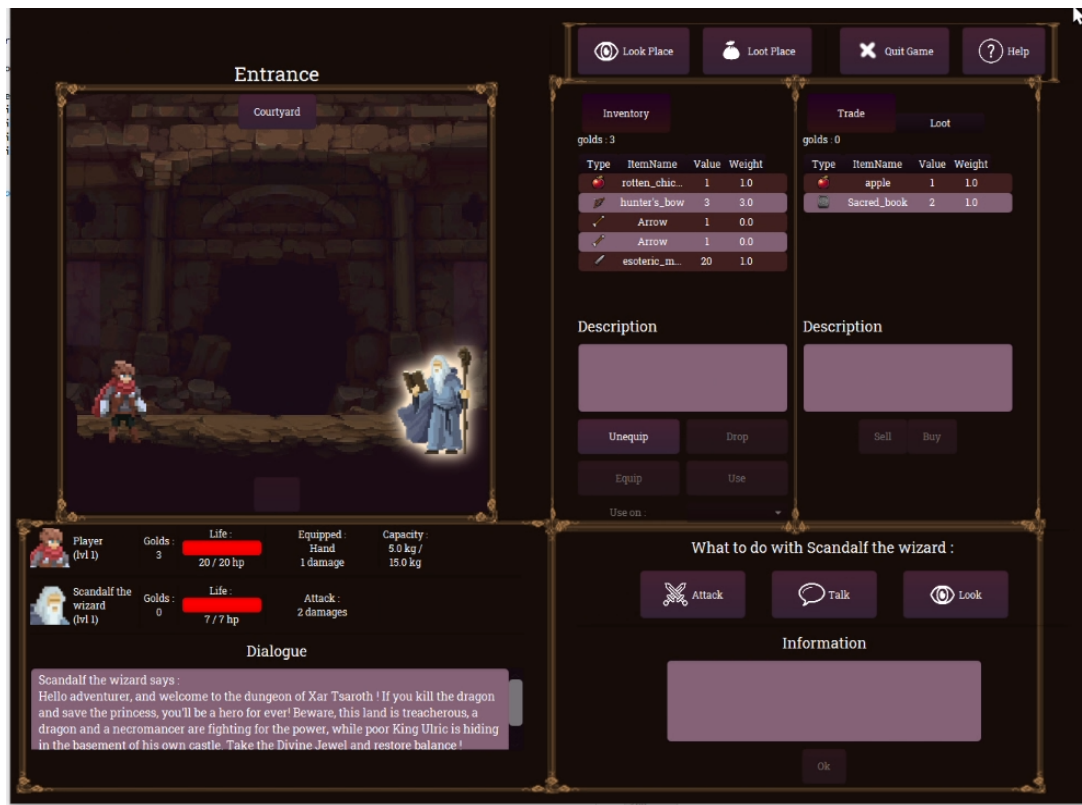
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla

name	Label
value	5 golds
damage /	2
weight	0.7 kg

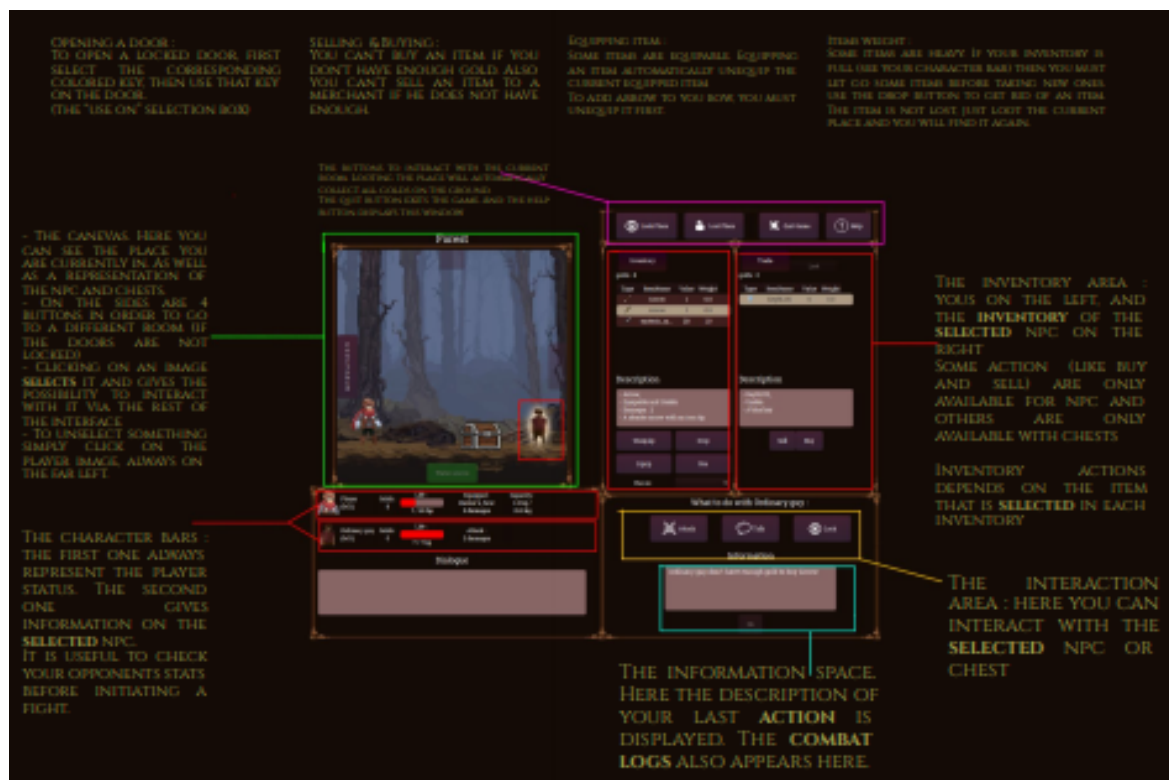
Message :

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla quam velit, vulputate eu pharetra nec, mattis ac neque. Duis vulputate commodo lectus.

Itération finale de l'interface :



Écran d'aide :



Structure de l'application

Dans la vue nous trouvons les ressources nécessaires pour l'affinage des éléments graphiques et les feuilles de style dans le dossier "view/ressources/"

les classes javaFX du nœud root et des fenêtres pop ups se trouvent respectivement dans "view/ui/" et "vue/stage/".

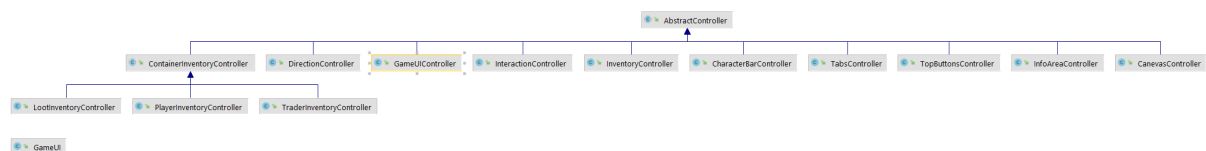
Les fichiers FXML sont dans "vue/ui/".

Dans le modèle il y a le code du projet de POO corrigé et mis à jour pour javaFX. ("./model/")

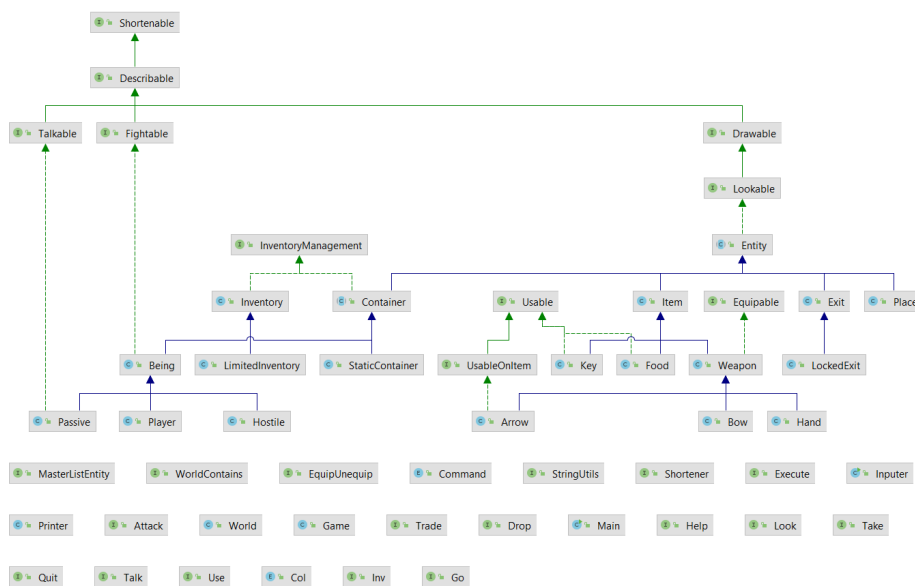
Les contrôleurs liés aux fichiers FXML se trouvent dans "/controllers/ui/"

Et les contrôleurs de l'application sont dans "/controller/" .


Voici le diagramme de classe des contrôleurs liés aux fichiers FXML :





Du modèle :




Et du contrôleur de l'application:

 CommandGenerator

 ExecutionController

 MainController

 Direction

Résolutions des problèmes rencontrés

Récupérer les résultats des exécutions s'est avéré plus compliqué que prévu dans le cas des utilisations des items. En effet des commandes use envoient déjà des booléens pour savoir si l'item avait été utilisé et auquel cas le retirer de l'inventaire du joueur. Mais nous souhaitions récupérer des Strings afin de pouvoir afficher les résultats des exécutions dans l'interface. Pour les commandes qui renvoient "void" , il suffisait de récupérer la partie pertinente de leur affichage. Mais dans le cas des items nous avons dû définir un couple <Boolean, String> comme valeur de retour de la commande et filtrer la partie qui est nécessaire à chaque composant.

Nous avons eu un problème de combats instantanés à l'entrée dans une salle, il s'affichait bien mais nous n'avions pas le temps de voir le pnj à l'écran (car les pnj agressifs attaquent dès l'arrivée dans la salle). Nous avons éclaté une partie de la logique de la commande GO qui exécutait les vérifications d'agression au moment de l'exécution. C'est désormais le contrôleur qui se charge de vérifier les agressions. Et c'est la boucle principale du jeu textuel dans le terminal qui se charge également de cette vérification. Maintenant lors d'une agression l'attaquant reste visible et le résultat du combat également jusqu'à ce que l'utilisateur décide d'appuyer sur "ok" sous le champ information pour débloquer l'interface et poursuivre .

Améliorations possibles pour l'application rêvée

Nous pourrions ajouter des bandes son tout au long du jeu avec différents thèmes(pour les combats, suivant les salles, ajout de bruitages, ...).

L'ajout d'animations pour chaque action (attaquer, déverrouiller des portes, parler, mourir (actuellement on change de png lors de la mort du joueur), ...).

L'univers en 2D et pixel art est un style qui nous plait beaucoup, mais le passage du jeu vers un visuel 3D pourrait être très intéressant également.

Ajoutez un bouton "reset" pour relancer la partie sans quitter l'application.

Éventuellement améliorer notre "Help" en ajoutant un tutoriel interactif étape par étape avec des "highlights" pour expliquer chaque bouton / champ en début de partie.