README - Projet Don’t Starve

Benjamin Delbos - Tiphaine Diot

10/06/2018

Table des matières

1 Comment lancer le jeu ?

1.1 Bibliothèques nécessaires

Dans ce programme nous utilisons la bibliothèque graphique de SFML et time.h.

1.2 Compilation

Pour compiler le programme il suffit d'utiliser :

```
make
```

Cette commande va compiler le programme grâce au makefile présent dans le dossier code.

Pour lancer le jeu, il suffit de rentrer dans la console la commande suivante :

```
./simulation
```

2 Le jeu : Don't Starve

2.1 Le principe du jeu

Le principe du jeu est simple : il faut garder en vie son personnage le plus longtemps possible.

Nous avons implémenté deux modes de jeu : un mode solo et un mode à deux joueurs.

2.1.1 Déroulement

Le jeu alterne entre deux états : le jour et la nuit. Toutes les 10 minutes (valeur que nous avons choisi) l'état du jeu change et le personnage doit s'adapter.

Le personnage peut se déplacer dans toute la carte et peut interagir avec les éléments présents dans l'environnement s'il possède les bons outils. Cette partie sera détaillée dans la partie *L'environnement* ci-dessous.

Chaque personnage commence avec 2 ressources Bois et 2 ressources Pierre afin de pouvoir au début créer un outil de son choix.

Le jeu s'arrête quand un des joueurs n'a plus de vie (voir partie *Comment rester en vie ?*). Nous souhaitions mettre en place une particularité, obligeant les joueurs a s'entraider. En effet, si le but est de gagner seul, si un des joueurs meure alors que l'autre n'a pas toute sa vie, les deux ont perdus.

2.1.2 L'environnement

L'environnement est composé de différents éléments avec lesquels le personnage peut interagir. Le fond de la carte n'est pas sujet à des interactions.

On distingue différents types d'éléments :

1. Les arbres
2. Les roches
3. Le feu
4. Les arbres à baies

Les arbres peuvent être coupés si le joueur possède une hache dans sa main. Les roches peuvent être cassées grâce à une pioche.

A chaque coup porté par le joueur, grâce à l'action interagir, les arbres et les roches peuvent un point de vie. Quand le joueur porte le coup final, il récupère respectivement du bois et de la

pierre. Il peut alors utiliser ces ressources pour construire de nouveaux outils, ou allumer un feu par exemple.

Le feu et les arbres à baies sont des éléments un peu différents. Le feu peut être raviver et les arbres à baies ne peuvent être coupés en revanche, comme leur nom l'indique, ils donnent des baies permettant au joueur de se nourrir.

2.1.3 Comment rester en vie ?

Pour rester en vie, deux grandes étapes sont importantes dans cette version simplifiée de Don't Starve.

La nuit, le personnage doit rester prêt du feu pour ne pas prendre peur et ne pas perdre la tête. En journée, il doit se nourrir et essayer de se reposer.

Ainsi, le but du jeu est d'interagir avec son environnement afin de créer de nouveaux outils, de se nourrir et d'allumer son feu (un seul feu peut être créer dans l'environnement).

2.2 Comment jouer ?

2.2.1 Lancement du jeu

Cette partie ne sert pas à lancer le programme mais a expliqué les différentes étapes avant le lancement du jeu à proprement parlé.

La page d'accueil permet de passer deux actions, le passage vers la page du choix de nombre de joueurs ou de quitter. En cliquant sur Play à cette étape, le joueur peut alors choisir s'il veut jouer seul ou à deux. Le bouton "VS IA" n'est pas implémenté car l'intelligence artificielle crée pour le moment ne fait qu'imiter le joueur principal.

Après le choix du nombre de joueurs, la carte s'affiche avec les différents éléments. Chaque joueur alors dispose d'un certain nombre de commandes au clavier lui permettant d'interagir avec l'environnement.

Si un (ou deux) joueur meure, une page finale s'affiche. La seule action possible est alors de quitter le jeu.

2.2.2 Commandes

Chaque joueur dispose de son panel de commandes, elles sont attribuer au moment du choix des joueurs. Sur la page affichant le choix du mode de jeu, les commandes sont montrées.

Pour le joueur principal (ou le joueur 1)

1. Les 4 flèches : se déplacer
2. Agir: m
3. Changer d'outil : l
4. Manger : k
5. Creer Hache : o

6. Creer Pioche : p
7. Creer Feu: i
8. Dormir : j

Pour le deuxième joueur :

1. R: Haut, F: Bas, G: Droite, D: Gauche
2. Agir: E
3. Changer d'outil : s
4. Manger : z
5. Creer Hache : x
6. Creer Pioche : c
7. Creer Feu: V
8. Dormir : w

3 Conception du programme

3.1 Ce qui fonctionne

Nous avons décidé d'implémenter le principe de base du jeu Don't Starve, c'est à dire que nous avons décidé de n'implémenter que deux outils (la hache et la pioche) et les éléments fondamentaux permettant de jouer (le feu, les arbres, les roches et les baies).

Nous souhaitions que le jeu soit fonctionnel et que toutes les fonctionnalités simples soient opérationnelles.

Ainsi les interactions basiques du jeu (couper, casser, cueillir, allumer/raviver le feu) sont fonctionnelles.

L'alternance jour/nuit est également opérationnelle avec un traitement visuel différent pour la nuit (où l'environnement visible est uniquement autour du feu).

L'évolution du niveau de vie des personnages est relativement basique pour le moment. Elle baisse pour les deux personnages toutes les 4 minutes.

3.2 Les fonctionnalités intéressantes

Nous avons décidé d'implémenter des structures compliquées comme les vecteurs et les HashMap.

Ce dernier type de structure nous a permis de décomposer la correspondance entre les touches appuyées et les actions. A chaque fois qu'une touche est appuyée pendant la simulation du jeu, le programme va identifier ou non la touche et dans le cas échéant lui associé un string correspondant à une action. Dans une autre fonction, cette string va permettre de lancer l'action sur le bon joueur.

Nous avons aussi choisi de créer une classe *ToDraw* dont nos classes ElemEnv et Personnage héritent. Elle se charge de l'affichage des sprites associés à chaque élément.

L'affichage des différents éléments nous a également pris beaucoup de temps. Nous souhaitions charger le moins de sprites possibles pour une question de fluidité et de mémoire. Nous avons donc eu un travail à faire sur le découpage des sprites le personnage notamment en fonction de ses déplacements par exemple.

Un point crucial de notre projet a été de permettre l'extension de notre projet notamment par le nombre d'éléments de l'environnement (sans compter la classe personnage). Tous les éléments (arbres, roches, feu et arbres à baies) avec lesquels le joueur peut interagir hérite de la classe *ElemEnv* qui implémente des fonctions permettant de récupérer le nom de la ressource associé à l'élément, le nombre de ressource récupéré à chaque interaction etc... Cette méthode nous a

permis de considérablement simplifié la fonction *Personnage::interagir(ElemEnv e)* qui est alors devenue beaucoup plus générale.

3.3 Ce que nous aurions aimé mettre en place

Nous aurions aimé implémenter plusieurs choses : notamment la possibilité de créer une torche à partir d'un feu, mais également un environnement animal.

L'environnement animal aurait été constitué de plusieurs bêtes dangereuses ou non pour le joueur. Idéalement, nous aurions aimé créer une classe *Lucioles* qui aurait permis de voir la nuit en plus du feu, et également une classe *Loup* qui aurait ajouté un danger supplémentaire la nuit.

Nous aurions aimé créer un mode multijoueur plus étendu, via un serveur réseau (le nombre de touches sur un clavier étant limité, nous ne pouvions que créer deux joueurs maximum sur un même ordinateur).

Nous aurions aimé améliorer le fait qu'un seul feu ne peut être créer. En effet, nous aurions voulu en créer plusieurs et créer un effet shader autour de chaque.

4 Notre organisation

Nous avons essayé de nous dispatcher au mieux le travail, mais les tâches se rejoignant assez rapidement le découpage n'a pas été très net.

Tout d'abord, l'UML a été decidé à deux. La première réflexion c'est déroulée en cours. Nous avons essayé de mettre en place une réflexion a deux sur l'ensemble de la mise en place des classes.

Puis notre travail a dû être séparé en grandes étapes :

1. Le graphisme : la création et la gestion de l'affichage
2. Les différents éléments de l'environnement
3. Les interactions
4. Les ressources
5. Les commandes des joueurs
6. Le mouvement des personnages
7. La gestion du temps

La réalisation des sprites/textures de l'environnement et des différentes pages hors jeu a été réalisé par Tiphaine. La grosse partie de la gestion de l'affichage dans le jeu a été géré par Benjamin.

Tiphaine a beaucoup travaillé sur les différentes classes des éléments de l'environnement, les différents états du jeu (avec les fonctions isClic[...]), la gestion de la HashMap de commandes liées à la partie des interactions, d'une partie des interactions et des tests unitaires présents dans le dossier *Tests*.

Benjamin a notamment travaillé sur le mouvement des personnages et de les associer au bon affichage. Il s'est également chargé de la création de l'environnement et de mettre en place la mise à jour toutes les nuits des éléments de l'environnement. Il s'est chargé de la mise en place du fonctionnement global à partir des différentes fonctions, et notamment de la mise à jour de la partie graphique avec l'évolution du jeu. Il s'est chargé de la gestion du temps.