LOUP GAROU

Projet mené par Nina Durin, Lucas Valladon et Adriana Girol

SOMMAIRE

INTRODUCTION

A l'issue de la première S, mes camarades et moi avons choisis de poursuivre la terminale S avec l'option Informatique et Sciences du Numérique. Un enjeu de cette option était d'être capable de mener à bien un projet. Nina, Lucas et moi avons décidé de former un groupe afin de réaliser un jeu, le loup garou.

PRESENTATION DU PROJET

Le loup garou est un jeu de société où les joueurs se voient attribués un groupe. Soit loup garou, soit villageois. Une partie consiste en plusieurs tours à l'issue desquelles un ou plusieurs joueurs sont éliminés. Le but étant pour les villageois d'éliminer tous les loup garous pour remporter la partie, et vice versa.

Nous avons décidé de mettre le jeu à disposition de n'importe quel joueur depuis un site internet. Ainsi nous avons eu recours non seulement à python pour coder le jeu mais aussi au html afin de réaliser notre site internet.

REALISATION DU PROJET

Organisation du groupe

A partir du moment où nous avions décidé ce que nous voulions le réaliser, il a fallu se répartir les tâches. Le codage en python demandant plus de volume de travail nous pensions juste que deux personnes s'en occupent pour ne pas avoir de charge trop lourde. Lucas et moi nous sommes donc divisés le travail en python. Dès lors je me suis occupée de l'affichage du programme, Lucas s'est occupé de la partie code pur et du déroulement du jeu, et Adriana me suis occupée du site internet. Le travail s'est fait en deux phases, d'abord une phase de recherches et ensuite une phase de réalisation. Aussi nous n'avions pas spécialement de deadlines à respecter, nous estimions plutôt le nombre de temps à consacrer à chaque tâche.

	PHASE DE RECHERCHES	PHASE DE REALISATION
LUCAS	Recherches associées à la création de fichier exe à partir de fichier python Dimensions juridiques propres à la licence Loup-Garou	Création de la structure de base du jeu Adaptation des règles pour que le jeu fonctionne en tant que programme
NINA	Recherche des règles : http://jeuxstrategie.free.fr/Loups_garous_complet.php mise en place de l'organisation et division du travail Recherche d'images utilisables vis à vis de la loi	Travail en division par répartition Cours de python donnés à Lucas Reprise du travail mené par Lucas pour ajouter une couche d'affichage dans le projet. Dernières modifications

ADRIANA	Étude de l'architecture de : https://www.minecraft.net/fr-fr/ https://www.drakensang.com/fr Listing des éléments essentiels Recherche d'images utilisables vis à vis de la loi Visionnages de cours vidéos, prise de notes	Codage de chaque éléments Modifications si nécessaires à l'aide de l'outil de développement du navigateur

Pour être tenu au courant du travail de chacun nous avions aussi créé un Github qui nous permettait d'ajouter au fur et à mesure nos avancée dans un dossier accessible par tous. Github est un service de Versionning de fichiers de programmation, c'est-à-dire un service cloud donnant accès à toutes les versions précédentes du dossier. Cela est important en programmation d'avoir accès à un service comme celui-ci, et ce gratuitement, car une perte du fichier code n'est plus grave. De plus, ce Github permet de voir l'avancée précise du projet car nous avons mis à jour le fichier une trentaine de fois. Github est un service en ligne, ce qui rend possible le visionnage précis de l'avancée du projet en python.
Le lien pour y accéder est le suivant : https://github.com/Phoboe/Projet-ISN. Dans la partie « commits », on peut accéder à l'historique d'ajouts au fur et à mesure.

Développement du projet : le Jeu

La réalisation du projet s'est donc réalisée en 2 étapes.
La première : créer un fichier en python lisible par l'ordinateur dans une boite MsDOS nous permettant de jouer au jeu. Le jeu est donc jouable à ce moment là uniquement en lignes écrite sur la command-line. Aucun graphisme n'est donc disponible mais passer par cela est nécessaire pour créer une base forte du jeu, certes sans graphisme mais avec par exemple une attente entre les interactions parfaites et une base utilisable. Cette étape a été réalisée par Lucas pour un projet stable dès son début.
La deuxième : ajouter une couche de code sur le fichier pour rendre le programme utilisable plus facilement et plus instinctivement. Cette tâche réalisée par Nina a plusieurs objectifs. Il faut d'abord que le jeu reste utilisable et que toutes les fonctionnalités initiales soient respectées, alors il a été écrite une démarche générale du jeu, à suivre, et permettant de guider la réalisation du programme. Il fallait ensuite que le programme, à terme, affiche des choses cohérentes et facile d'accès pour que le jeu soit le plus agréable possible. Il a donc été fait un certain travail de graphisme et la majorité des images ont été faites à la main avec des photos personnelles, donc des images sans droits interposés.
Mais certaines images ont été prises d'internet, notamment les icônes du jeu, pour lesquelles nous avons fait appel à : https://icon-icons.com/fr/. Ce site est un recueil d'icônes gratuites qui fait uniquement appel au don des designers, donc des images libres de droit car données par les créateurs sans conditions.

Articulations principales du jeu

Le projet est codé en Python, qui est un langage orienté-objet. Cela signifie que l'on peut concaténer les objets les uns dans les autres. J'ai donc divisé le projet en plusieurs instances placées dans de fonctions et dans des classes.

Et on utilise la librairie Pygame qui permet l'affichage avec Python, une librairie bien documentée et libre de droit.

Cette approche permet une clarté exceptionnelle en programmation et la reprise d'un programme dans un tel langage est très facilitée.

Nous commençons notre programme en définissant les fonctions et les classes qui seront nécessaires pour le lancement du jeu. Lucas s'est occupé de cette partie en divisant le tout en deux fonctions principales : menu() et play(). J'ai modifié ces fonctions de manière significatives afin que l'affichage puisse être plus rapide, ce que l'on appelle de l'optimisation. Mais de fait, elles restent sous le même nom. J'ai modifié les noms de variables, ajouté des classes et des fonctions, modifié la structure de menu() et aussi de play().

Ensuite, j'ai commencé l'affichage. Mon choix artistique a été la simplicité. De ce fait j'ai dessiné et choisies des images simples et épurées. Mais il a fallu les introduire.

Le menu, tout d'abord, et le jeu ensuite, car on peut le voir dans le code : le menu est lancé avant le jeu.

```
1256
1257  display.stopALL = False
1258  while display.stopALL is False:
1259      menu()
1260      play()
1261      pass
1262
```

Donc commençons par le menu.

```
113
114  def menu():
115      win = display.win
116 ›    def Rules():⋯
282      '''Fin Rules'''
283
```

Le menu commence par Rules qui est une fonction que l'on appellera plus tard, qui affiche les règles, tel une visionneuse de photos. On a plusieurs slides, et on peut passer aux autres avec les flèches sur les côtés de l'écran.

Ensuite, on a une fonction permettant de faire un dégradé visuel en fondu au noir faisant apparaître le menu tout doucement. Cet effet visuel suit ce que j'ai souhaité faire avec ce projet : un projet fluide. J'ai souhaité dès le début que le projet sois le plus doux possible pour l'utilisateur et non brut.

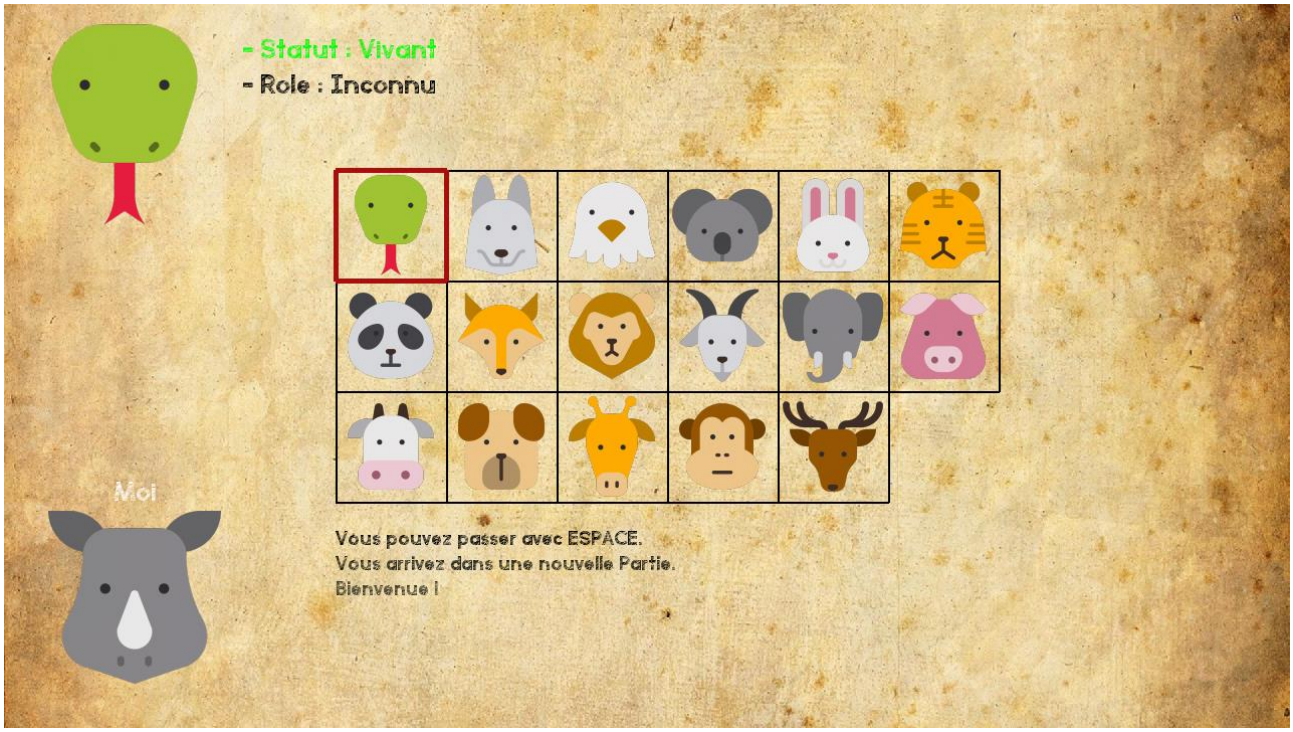
Après ce dégradé visuel, on arrive sur une interface accueillante avec des boutons interactif au nombre de 3 : le bouton permettant de jouer, le bouton pour quitter et celui permettant de voir les règles du jeu. Le menu est présenté comme sur la capture d'écran qui suit :

Chacun des boutons est dit interactif car il grossit doucement lorsqu'on passe la souris dessus jusqu'à être 1.5x plus grand pour sa taille maximale. On utilise la fonction sinus en facteur d'agrandissement car cette fonction passe par l'origine et car elle varie de manière douce.

```
381         # ---------Gestion de l'agrandissement des boutons--------- #
382         # Pour playButton :
383         playButton.factor = 1 + ( (sin(playButton.sub)**2)*0.2 )
384         if (mouse_x >= playButton.x and mouse_x <= playButton.x+playButton.w and
385            mouse_y >= playButton.y and mouse_y <= playButton.y+playButton.h)     :
386             if playButton.sub <= (pi/2):
387                 playButton.sub += pi/60
388         else:
389             if playButton.sub >= 0:
390                 playButton.sub -= pi/60 # on rétrécit l'image jusqu'à sa taille normale
```

Il n'y a plus qu'à cliquer pour accéder à la partie que l'on souhaite : jeu, règles ou quitter. De cette manière à chaque tour de boucle on affiche les boutons à leur taille modifiée et on attend que le joueur clique pour sortir de la boucle.

Cette même boucle étant la seule dans le menu qui gère cela, la quitter revient à atterrir dans le jeu. On arrive alors dans une interface comme celle-ci qui marque le début d'une partie :



On a plusieurs choses :
- Notre avatar situé en bas à gauche,
- La grille des autres avatars au milieu,
- Les détails de l'avatar souhaités,
- Le texte de la partie en bas.

La case rouge est celle sur laquelle on a la souris, sans cliquer. Cela permet d'accéder aux informations des personnages facilement et rapidement.

Les paramètres relatifs au joueur « sélectionné » s'affichent sous la même forme et selon les conjonctures de la partie, on peut accéder ou non à certaines informations. Lorsqu'un joueur est mort, on voit une croix sur son avatar.

Le texte est affiché succinctement sur 7 lignes, de plus en plus transparentes. Et à chaque nouvelle ligne, les autres sont déplacées vers le bas et un son de notification est là. On a pour cela la fonction text() qui s'en occupe.

```
533     def text(message):
534         ''' écris le message dans texts et sera affiché ensuite. '''
535         sounds.newText.play()
536         display.texts.pop(7)
537         display.texts.insert(0, message)
538         pass
```

Avec texts une variable de type liste qui contient les phrases précédentes et la phrase actuelle.

Concernant l'affichage pur, on a plusieurs fonctions venues de Pygame qui agissent différemment sur la fenêtre ou qui créent des choses :

Tout d'abord la création de la fenêtre :

```
88  display.win = pygame.display.set_mode((display.width, display.height))
```

Cette fonction crée une surface de la taille demandée qui correspond à la fenêtre.

Ensuite :

```
94  pygame.display.set_caption("Les loup-Garous de Thiercelieux")
95  icon = pygame.image.load("TEXTURES/icon.png")
96  pygame.display.set_icon(icon)
97  clock = pygame.time.Clock()
98  clock.tick(display.fps)
99  pygame.key.set_repeat(400, 30)
00  pygame.mouse.set_visible(True)
```

On change le nom de la fenêtre

On charge l'icône,

On la change

On définit un objet de temps
Et on l'initialise

On définit le temps entre la répétition des touches et on définit la souris comme visible.

Donc on importe une image, avec Pygame, avec la fonction `pygame.image.load()`.
On peut aussi la transformer ensuite, avec `pygame.transform.smoothscale()`.
On l'affiche ensuite avec `window.blit()`.

Ces trois fonctions sont les fonctions de base de Pygame et je me suis servie majoritairement de celles-ci pour créer la couche d'affichage.

Contraintes rencontrées

- fonctions difficiles à appréhender
- Problèmes de droits, il fallait s'assurer que les images n'étaient pas protégées ou utilisables à condition de créditer la source.
- Choix d'hébergement, réseau distant ou local (avantage = non intervention de cookies)
- durée de programmation longue car le projet est très complexe et se compose de milliers de lignes de code.

CONCLUSION, Personnelle :

L'année d'ISN a été une année très stimulante et m'a apporté beaucoup d'expérience en programmation en plus de celle que j'ai pu acquérir. Elle m'a donnée de l'assurance dans la programmation car j'ai été en groupe et j'ai reçu des conseils de mes camarades. La programmation de jeux ou autres programmes est une activité passionnante pour moi, j'aime y passer du temps et j'aime aller plus loin dans l'apprentissage du code. De plus l'ajout d'une dimension mathématique est intéressante et j'aime jouer avec les fonctions que j'apprend en math et en physique pour l'implémenter dans un code et faire des simulations diverses et visuelles.

Les améliorations possibles de notre projet aurait pu être d'ajouter une sorte d'intelligence artifielle pour qu'on puisse dialoguer avec les autres joueurs, ou alors même ajouter la possibilité de jouer avec de réelles personnes en ligne. Nous aurions pu de même ajouter des rajouts aux jeu qui existent dans certaines éditions.