Projet Bebou

Présentation Générale:

Notre projet est un jeu de plateforme dans le style des jeux Mario du studio Nintendo. Il a été codé en python et rentre dans le chapitre "programmation objet" du programme de NSI de Terminale.

Histoire du jeu:

Pour avoir un thème au projet, nous avons créé une histoire. Nous jouons une lanterne appelée Bébou. Elle est à la recherche d'une autre lanterne, sa bébou, qui elle est perdu au fin fond d'une mine.

Pour s'adapter à cette petite histoire, nous avons alors adapté les graphismes: un fond avec des pierres, des plateformes de pierres, de la lave, des champignons de décoration, etc. *(voir annexe)*

Fonctionnement du jeu:

Le fonctionnement du jeu est simplement: il suffit de sauter de plateforme en plateforme à l'aide de la touche "ESPACE" pour sauter, la touche "D" pour aller vers la droite et la touche "Q" pour aller vers la gauche.

Mais attention! Il faut absolument éviter les piques et éviter de tomber dans la lave ou le vide sinon vous perdez! *(Toutes ces instructions sont notés dans l'écran titre d'aide, accessible dans le jeu (voir annexe))* Il existe aussi des vortex, vous permettant de faire un double saut!

Pas de panique il existe un écran titre vous permettant de retourner au menu et recommencer le jeu.

Pour finir le jeu, il suffit de rejoindre la deuxième lanterne, rose, qui se situe à la fin du niveau. Une fois la lanterne touchée, vous avec accès à deux possibilités, soit revenir au menu et recommencer le jeu, ou accéder au jeu bonus (un jeu inspiré du jeu dinosaure de chrome) et faire le meilleur score! *(attention pour ré accéder au vrai jeu, il faut fermer la fenêtre et relancer le jeu)*

Pour savoir comment lancer le jeu, un document *fonctionnement.txt* explique cela dans le dossier "_Documentation" du projet.

Origines et intérêts:

Nous voulions coder un jeu de plateforme car cela nous a permis autant de s'entraîner à la logique et à la façon de coder (en python et en programmation objet) lors de la création d'un jeu vidéo mais aussi de nous "former" au module pygame permettant l'implémentation d'une interface graphique.
Comme autre motivation, nous avions aussi promis à notre professeur de NSI de lui rendre, avant la fin de l'année, un projet de jeu de plateforme.

Organisation du Travail:

L'équipe du projet est composée de deux personnes:
Elliot Barthélémy (Graphiste/Développeur) et Maëlys Boissezon (Développeuse).

La répartition des tâches s'est faite au fur et à mesure en se concertant. Nous utilisions un serveur discord créé pour le projet, nous permettant de communiquer mais aussi de pouvoir nous envoyer les différentes partie du code mais aussi les différents graphismes du jeu.

L'organisation du projet a été faite en suivant un cahier des charges.

Cahier des charges:

-Semaine 1:
(En classe et en travail à la maison)

-Formation au module python: pygame (Elliot/Maëlys)
-Mise à l'écrit des idées pour l'histoire du jeu et différentes classes et méthodes à mettre dans le jeu; (Maëlys)
-Élaboration des premiers graphismes: différentes sprites du personnage, plateformes, sols, fond d'écran; (Elliot)

-Semaine 2:
(En classe et en travail à la maison)

-Codage de la base du jeu en programmation objet:
→ animation du personnage: implémentation et animation des sprites; (Maëlys)
→ le niveau: → mécaniques de jeu, gravité (Maëlys)
→ collisions, défilement droite/gauche; (Elliot)
-Création des graphismes supplémentaires: décorations, lave, piques, vortex; (Elliot)

-Semaine 3:
(Travail à la maison et regroupement)

-Fin de codage:
→ collisions avec les vortex, la lave (avec animation) et les piques; (Elliot)
→ implémentation de la mort du joueur; (Maëlys)
→ ajout des différents écrans titres; (Maëlys)
→ création du jeu bonus; (Elliot/Maëlys)
→ level design du niveau; (Elliot/Maëlys)
→ ajout du défilement vers le haut; (Elliot)

Fonctionnement et Opérationalité:

Avancement du Projet:

Le Projet est fini. Il manque néanmoins quelques petites choses à faire.

En cours de réalisation:

-Pouvoir revenir à l'écran du menu du jeu, après avoir eu accès au mini jeu

Reste à faire:

-composer/ajouter de la musique au jeu

-améliorer les différentes collisions entre le joueur et les plateformes et objets

Vérification de l'absence de bugs et Facilité d'utilisation du projet:

Pour vérifier l'absence de bugs nous avons:

- testé et retesté les collisions joueur/plateforme afin qu'elles soient le mieux possible mais aussi l'affichage et la position des différents graphismes

-affichage de certaines variables dans le terminal pour voir si les fonctions fonctionnaient:

exemple: Pour la fonction disponible dans le fichier"support.py", permettant un accès aux chemins des images plus efficace, on a du afficher la liste pour voir si tous les chemins étaient bien importés

Difficultés rencontrées et solutions:

Lors du développement du jeu, nous avons rencontré plusieurs difficultés.

Tout d'abord une difficulté de test: en effet lors du "level design", tester les nouvelles combinaisons de plateforme était fastidieux. Nous avons alors créer une fonction permettant de recentrer l'écran sur l'image du joueur. Nous n'avions plus qu'à placer le joueur où nous le voulions et appuyer sur r pour passer de l'image du début du jeu à l'endroit où se trouvait le joueur. Cette fonction n'est pas disponible dans le code du jeu car elle est inutile au bon fonctionnement de celui-ci. Cependant, si elle vous intéresse, elle disponible à la fin de l'annexe en dernière page.

Nous avons aussi eu des difficultés du au matériel que nous avons. En effet, nous utilisons L'Ordi de la région Occitanie, et les graphismes que nous avons fait en premier faisaient laguer nos ordinateurs. Nous avons donc du modifier ces graphismes afin qu'ils soient moins "lourds".

Ouverture:

Idées d'amélioration:

Nous avons plusieurs idées d'amélioration:

-Ajouter des nouveaux **niveaux** avec différents thèmes. A chaque niveau, la lanterne "sortirait" de sous-terre (et gravirait une montagne): par exemple, le niveau 1 dans une mine profonde, le 2 en haut d'une mine, le 3 sur terre, etc jusqu'à arriver à la lanterne.

-Créer des **ennemis**: Ils seraient sous forme de cristaux (nous avons déjà les graphismes *(voir annexe)*, et bougeraient de gauche à droite à des endroits définis mais n'attaqueraient pas. Pour les combattre, nous ajouterons une pioche à la lanterne (le joueur) lui permettant de leur donner un coup et de les tuer en appuyant sur une touche du clavier. De ces ennemis nous pourrions aussi créer différents "boss" à chaque fin de niveau, qui eux mourraient en plusieurs coups et pourraient même attaquer.

-Implémenter un **défilement** de l'écran vers le bas (nous avons pour l'instant un défilement haut, gauche et droite): il permettrait d'éviter que le joueur meurt quand il tombe du plateforme mais qu'il y a d'autres plateformes en dessous que l'on ne voit pas à l'écran.

Stratégie de diffusion:

Pour diffuser notre projet à un large public, nous pourrions:

- Sponsoriser un équipe de foot nationale et un bateau du Vendée Globe;
- Payer une pub à la mi temps des matchs sportifs;
- Organiser des concours sur internet et faire des partenariats avec des influenceurs;
- En faire parler dans les journaux et plateaux télévisés;
- Mettre le jeu dans des bundle (plateforme d'achat de jeux vidéos collectant des dons pour des œuvres caritatives, en fonction du lot);
- Faire des compétitions de speedrun du jeu;
- Mettre le jeu en code libre pour attirer les moddeurs;

Analyse critique du résultat:

Au niveau de l'organisation: nous aurions pu nous prendre plus à l'avance pour éviter de se former sur pygame en vitesse par exemple (nous avons dû regarder plusieurs vidéos/tuto lors d'une semaine de cours); nous avons réparti les tâches au fur et à mesure de l'avancée du projet, nous aurions pu se répartir mieux les tâches et surtout à l'avance.

Au niveau du jeu en lui-même: nous aurions plus créer une histoire plus étoffée, afin de le rendre plus intéressant pour le joueur; nous pourrions changer les graphismes des plateformes et du sol afin de rendre le jeu moins "cubique", moins "plat" et moins monotone.

Annexes:

Personnages et Objets du jeu:


Joueur


Lanterne


Pique

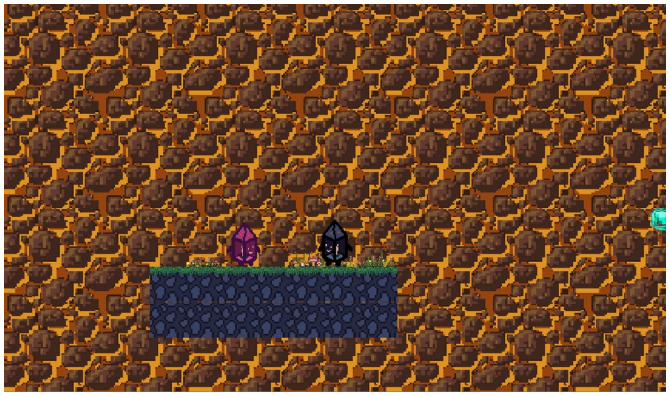

Lave


Vortex


Exemples de décoration

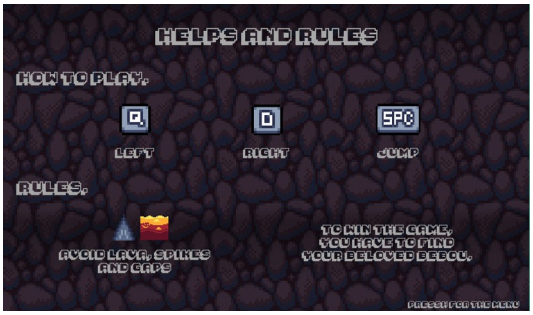
Images du jeu:


Début

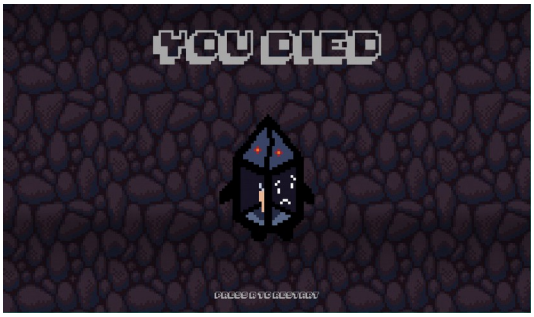

Fin

Écrans titres:

Menu



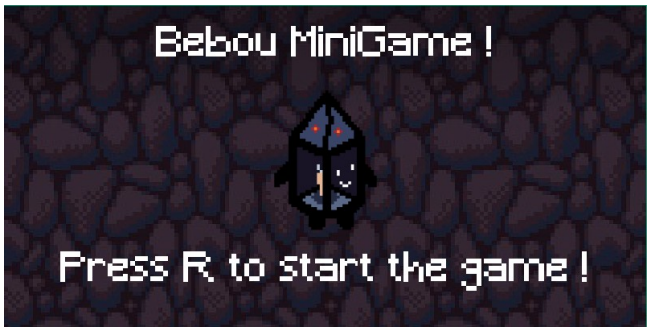
Aides



Mort

Fin

Mini-Jeu:



Écran Titre de début

Écran Titre de fin

Images du jeu

Fonction d'aide au level design:

```
def fixplayer(self):
    player = self.player.sprite
    if player.rect.x != screen_width/2:
        for sprite in self.tiles.sprites():
            sprite.rect.x -= player.rect.x - 8*tile_size
        player.rect.x = screen_width/2
    if player.rect.y != screen_height/2:
        for sprite in self.tiles.sprites():
            sprite.rect.y -= player.rect.y - 5*tile_size
        player.rect.y = screen_height/2
```

Appel de la fonction dans la fonction qui lance les méthodes de la classe Player

```
def run(self,x):

    keys = pygame.key.get_pressed()
    if keys[pygame.K_r]:
        self.fixplayer()
```

Graphismes des futurs ennemis: