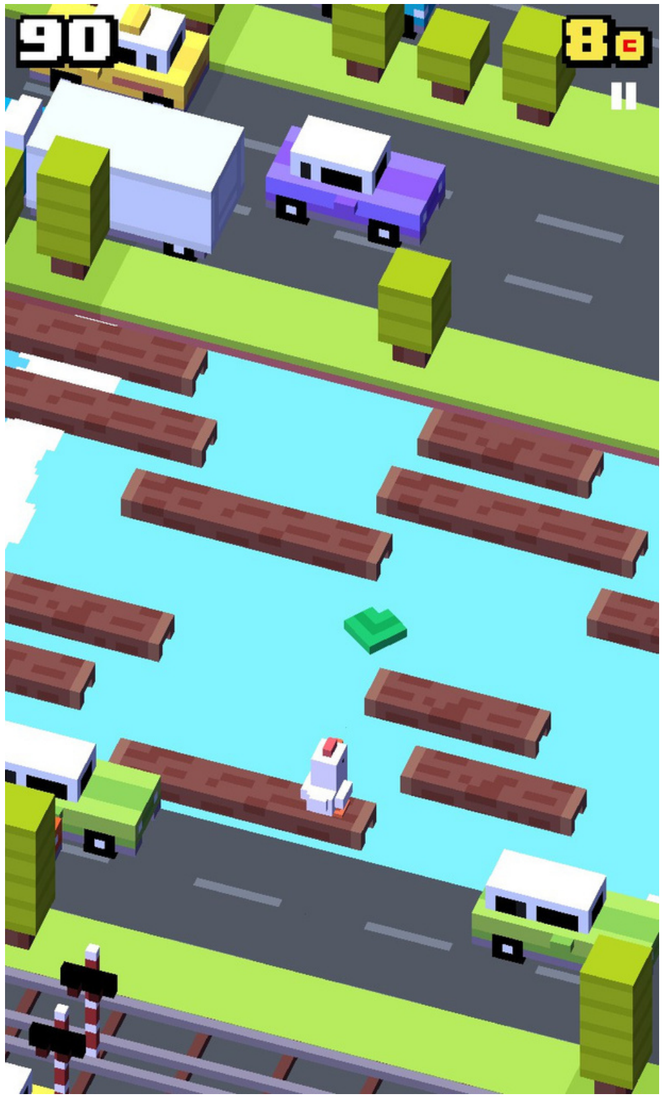
Groupe 27 : Pauline Elizalde, Basile Heurtault, Ruben Cizallet, Chenghao Lyu, Arthur Leene, Diane Dufetelle

LIEN VERS LE DÉPÔT GITLAB

PRINCIPE DU JEU CROSSY ROADS

Contrôler un personnage avançant case par case devant éviter un certain nombre d'obstacles (arbres qui bloquent, traverser une route sans se faire écraser, traverser une rivière sans tomber dans l'eau en passant sur des rondins). L'écran avance de plus en plus vite pour forcer le personnage à avancer, s'il n'avance pas assez vite et sort de l'écran il meurt.



DESCRIPTION GLOBALE DE NOTRE PRODUIT

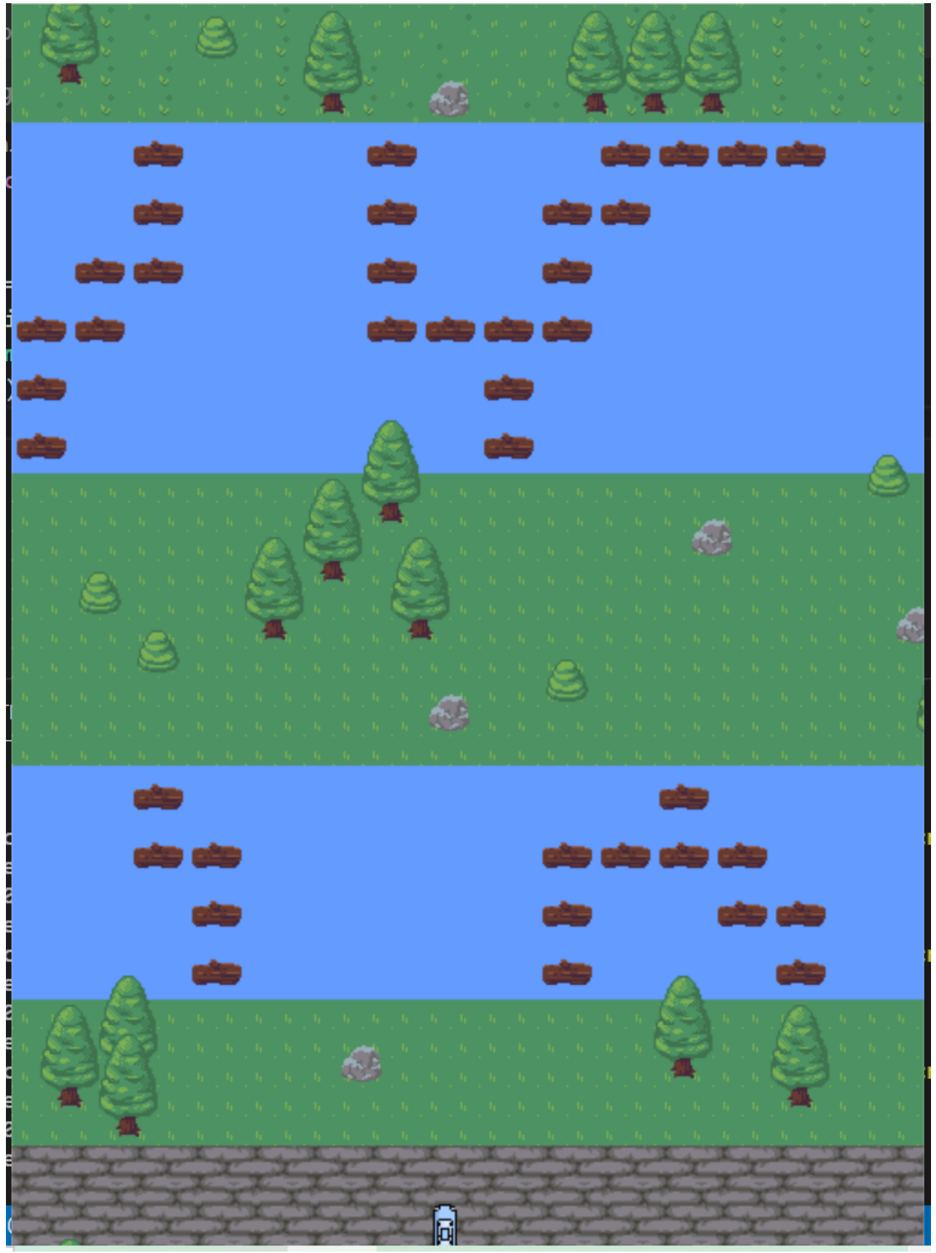
Quoi?

- Ecran de jeu dans lequel seront placés différents obstacles (arbres, voitures, rivière) à franchir
- L'utilisateur joue via le déplacement d'un personnage dans cet écran en lui demandant une direction parmi (bas, haut, droite, gauche)
- Le jeu se termine lorsque le personnage entre en collision avec une voiture ou tombe dans la rivière

Pourquoi? jouer à un jeu de type arcade

Pour qui? Utilisateur souhaitant se distraire en jouant à un jeu de type arcade

Readme



NOS DIFFÉRENTS MVP

- **Premier MVP** : une grille de jeu type 2048 où le personnage est représenté par une croix qui se déplace, une nouvelle colonne apparaît dès que la croix s'approche d'un bord (droite ou gauche), pas encore d'obstacles. Si le personnage avance trop vers le haut de la grille alors il a perdu
- **Deuxième MVP** : sans tiled et sans carte graphique mais avec des voitures se déplaçant et la possibilité de déplacer le personnage, génère des routes de manière aléatoire et il y a du son (déplacement, collision avec une voiture)
- **Troisième MVP** : en utilisant tiled et une carte graphique que nous avons nous-mêmes générée (création de routes, de rivières)

ELÉMENTS SUR LA QUALITÉ DU CODE

- Utilisation de la bibliothèque **pygame**
- Dossier "graphic" contenant les différents éléments graphiques, le joueur, les voitures
- Dossier "game_crossyCS" avec les différents fichiers : __init__.py, game.py, main.py, player.py, textual_crossyCS.py, car.py, map.py, menu.py
- Dossier sounds avec les différents sons de notre jeu (game over, won!, crashsound, jumpsound)
- Différentes **classes** (player, car)
- Utilisation de **sprites** (mouvement, gérer les collisions avec rect *(rectangle de l'espace occupé par le sprite)*)
- On a généré une carte tmx (package pytmx)
- Package pyscroll pour le déplacement du joueur
- Utilisation de **tiled** pour gérer la position, la taille des différents éléments et le déplacement dans la map
- Différents niveaux de jeu à choisir dans le menu d'accueil ainsi que différents personnages
- Pour lancer le code : aller sur le fichier main.py et lancer. Pour le MVP aller sur le dossier game_crossyCS_MVP

COMMENT?

Découpage en sous-tâches :

- Carte graphique
- Apparition et déplacement du joueur
- Différents obstacles et collision
- Déplacement des voitures et collision
- Si le joueur entre en collision avec une voiture, tombe dans la rivière alors il y a game over (apparition d'un écran "game over")
- Ecran d'accueil à presser pour pouvoir jouer avec différentes options : choix du niveau, du personnage

Différentes pistes au départ :

- Map infinie mais compliqué à générer
- **Différents niveaux : à choisir dans le menu d'accueil**

RÉPARTITION DU TRAVAIL DANS LE GROUPE

Nous avons tous réfléchi au départ du projet à la définition des besoins, aux différents MVP que nous pourrions mettre en place. Il fallait également définir les règles du jeu, les différents obstacles et s'interroger dans chaque cas au code python à mettre en place (collisions, fin de jeu).

MVP : Ruben
Carte graphique : Basile et Arthur
Déplacement du joueur : Arthur
Collision obstacles : Pauline et Basile
Déplacement des voitures et collision : Basile et Arthur
Ecran d'accueil et game over : Pauline, Chenghao, Diane
Musiques : Ruben
Readme : Diane

DES PISTES D'AMÉLIORATION

- Caractère infini et aléatoire
- Carte qui bouge en fonction du temps
- Rondins de bois qui bougent
- Score avec timer pour établir des records