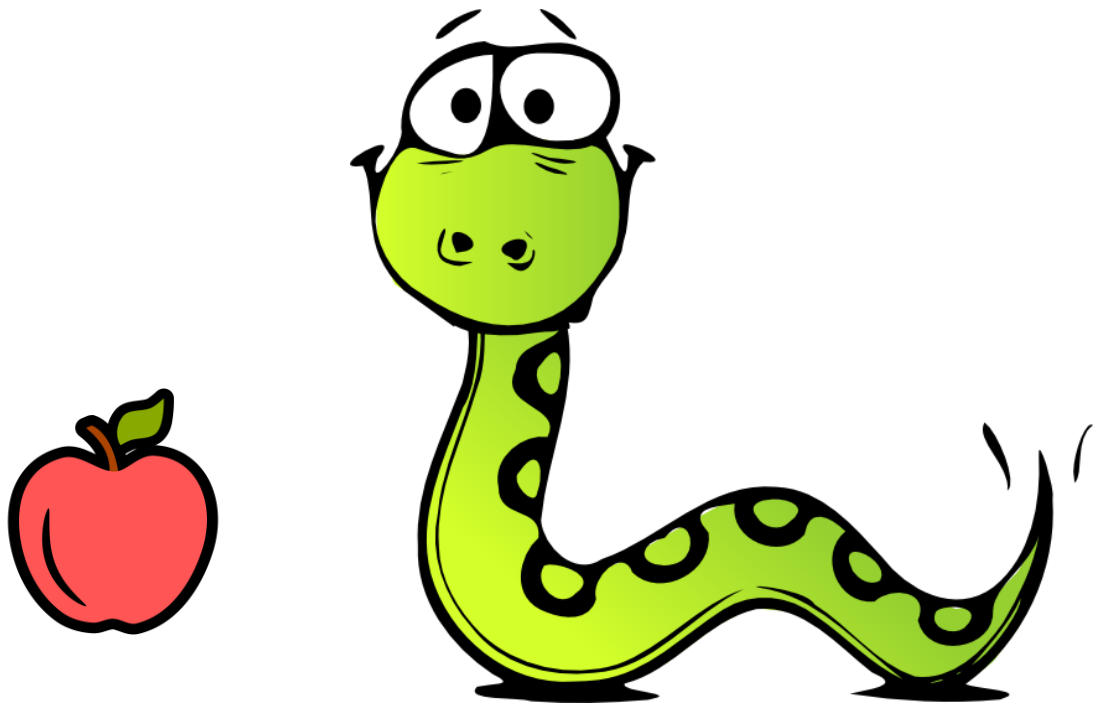


ETML

ETML

# Rapport de Projet

Snake en JS



Evin Paramanathan  
09/01/2024

# Table des matières

.....	0
Table des matières.....	1
A. Introduction .....	2
B. Apprentissage JS .....	2
a) Variables .....	2
b) Classes .....	2
c) If.....	2
d) Import / Export.....	2
e) Math.....	3
f) ArrowkeyPressed.....	3
g) Fonction fléchées .....	3
h) ECMAScript.....	4
C.    Projet Snake .....	4
a) Snake .....	4
b) Pomme .....	5
c) End Game .....	6
D.    Test .....	7
E.    ChatGPT .....	7
F.    Conclusion .....	7
G.    Sources .....	8

## A. Introduction

Dans le cadre de ce projet, nous devons créer un jeu de Snake en JavaScript afin de développer nos compétences dans ce langage. Cela nous préparera aux modules avancés de JavaScript à venir. Pour atteindre nos objectifs, nous avons la possibilité de nous appuyer sur Internet pour effectuer des recherches sur des forums ou des sites dédiés au JavaScript, ainsi que d'utiliser ChatGPT, dont l'utilisation a été décrite dans le point 3.f.

En plus de réaliser le projet Snake, nous devons également créer un aide-mémoire personnel. Ce document contiendra des informations sur JavaScript qui nous semblent importantes pour nos apprentissages.

Pour conclure cette introduction, si nous terminions le projet Snake en avance, comme c'était le cas dans notre classe, nous avons la possibilité de travailler sur un second projet. Ce projet était similaire à un futur projet consistant à développer une machine de self-service. L'objectif était de concevoir une borne automatique pour un célèbre fast-food, en réalisant toute la partie JavaScript nécessaire : sélection de produits, suppression de produits, calcul du total des prix, etc.

## B. Apprentissage JS

### a) Variables

Il y a 4 façons de déclarer une variable :

- Var : cette variable permet de déclarer une variable et même de lui donner une valeur.
- Let : Permet de déclarer une variable dont la portée est limitée celle du bloque ou elles sont déclarées
- Const : permet de déclarer une constante nommée qui est seulement accessible en lecture.
- Automatiquement : Ça crée automatiquement une variable

Mais le mieux c'est d'utiliser des const et let.

### b) Classes

Les classes sont juste des fonctions spéciales. Pour définir des classes il suffit de mettre class suivie par le nom que l'on souhaite donner.

```
export class Snake {}
```

Il y a un export juste avant classe parce que comme cité de le point 4 cela va nous permettre de lier avec une autre page comme la main

### c) If

Un if permet d'effectuer différents type d'actions en fonction d'une condition

#### Opérateur ternaire(?)

L'opérateur ternaire ou point d'interrogation permet de faire une condition comme le for mais de façon plus simple et plus rapide

### d) Import / Export

En js on utilise des imports et export pour lier les différentes pages de js, dans le cadre du projet snake on utilise l'export sur la class snake `export class Snake {}` et pour l'importer dans la page main grâce à `import {Snake} from './Snake.js';`

## e) Math.

Math n'est pas un constructeur, après le point on peut mettre de propriétés statique.

Telle que :

- PI : 3.14
- LN10 : logarithme naturel de 10 = 2.303
- Ln2 : logarithme naturel de 2 = 0.693
- SQRT2 : racine carrée de 2 = 1.414

Sinon on peut aussi mettre des méthodes statiques

- Abs : renvoie la valeur absolue de x
- Ceil : renvoie le plus petit entier supérieur ou égal à x
- Floor : Renvoie le plus grand entier inférieur ou égal à x
- Random : renvoie une valeur aléatoire

Et il y en a plein d'autre qui permette de faire différents opérations


## f) ArrowkeyPressed

Pour le projet on doit utiliser les flèches pour déplacer le snake pour cela on doit c'est comme en C# on utilise des Switch Case

## g) Fonction fléchées

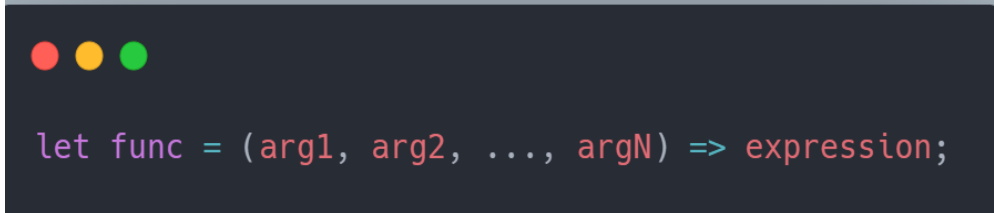
C'est une syntaxe plus courte et plus simple pour créer des fonctions.

Par exemple au lieu d'écrire une fonction comme ça :



```
let func = function(arg1, arg2, ..., argN) {  
    return expression;  
};
```

On va raccourcir en une seule ligne comme cela :



```
let func = (arg1, arg2, ..., argN) => expression;
```

(Exemple repris sur le site [developer.mozilla.org](https://developer.mozilla.org))

Tout ça va créer une fonction qui accepte les arguments Arg1, Arg2, ..., ArgN. Ensuite elle va évaluer l'expression et va retourner le résultat.

Si on souhaite faire des fonctions plus compliquées on peut créer une fonction ensuite on met les arguments et ensuite mets des accolades, cela va permettre d'ouvrir une fonction fléchée multi ligne. Et si on utilise des fonctions fléchées avec des accolades on doit utiliser un Return à l'intérieur de celles-ci.

En gros il y a deux types de fonction fléchée, une avec des accolades ou sur une ligne il y a les arguments à gauche et à droite une expression (ça évalue et renvoie le résultat.) Et la deuxième est celui avec les accolades, on met les arguments ensuite on met les accolades pour créer une fonction multi ligne, et on n'oublie pas de mettre un return.

#### h) ECMAScript

ECAMSCRIPT est une norme de programmation sur laquelle le JavaScript est basé. ECMA est l'abréviation de European, Computer, Manufactures, Association.

C'est une organisation qui développe de norme. Fun facts ECMA se situe à Genève.

## C. Projet Snake

### a) Snake

Pour les Snake il y a plusieurs méthodes la première est le

- **Constructor**

Cette méthode va nous permettre d'initier la positions de base du snake donc  $x = 10$  et  $y = 10$ . Ça va aussi nous permettre d'afficher la taille du snake  $this.size = 40$ . Mais le constructor vas aussi nous servir pour déclarer une variable pour le canva

```
constructor() {  
    const canvas = document.querySelector('canvas');  
    this.ctx = canvas.getContext('2d');  
    this.x = 10;  
    this.y = 10;  
    this.size = 40;  
    this.corps=[];  
}
```

- **Move**

Dans cette méthode on va faire deux types déplacements du snake le premier c'est pour la tête du snake et le deuxième est pour le corps. Pour la tête du snake on va faire que quand on appuie sur une des flèches que cela fasse que la tête du Snake va dans la direction indiquer. Pour cela on va utiliser comme au C# des switches case, d'abord on mettre le switch avec une variable qu'on va appeler direction, ensuite on met case avec la direction « Gauche » et ensuite on met la direction ( $x--$ ) et on finit par un break. Il faut faire ça avec toute les directions

```
switch (this.direction) {  
    case 'left':  
        this.x--;  
        break;
```

Ensuite il y a partie pour le corps. Pour le déplacement du corps

- **Grow**

Pour le grow il faut d'abord faire en sorte que chaque fois que le Snake mange une pomme une case s'ajoute et que cette case suit la tête du Snake, ensuite le reste du corps suit à chaque fois la case qui le précède. Pour ajouter la case on utilise un « push » cela permet d'ajouter un ou plusieurs éléments à la fin du tableau.

```
grow() {  
    this.corps.push({ x: this.x, y: this.y });  
}
```

b) Pomme

- **Constructor**

La pomme aussi a un constructeur ou il y a le canvas et la taille mais aussi une variable pour générer une position pour la pomme.

```
constructor() {  
    const canvas = document.querySelector('canvas');  
    const ctx = canvas.getContext('2d');  
    //taille de la pomme  
    this.size = 40;  
    this.GenerateRandomPosition();  
}
```

- **Positionnement**

Il y a une méthode qui va permettre de d'afficher la pomme de façon aléatoire sur le terrain de jeu. Pour cela il faut qu'on utilise math.random afin de créer une position x et y on va aussi utiliser un math.floor pour que la position aléatoire soit un nombre entier et s'il y a des virgules c'est arrondi au chiffre inférieur.

```
GenerateRandomPosition() {  
    //genere une position aleatoire pour la position x  
    this.x = Math.floor(Math.random() * (800 / this.size)) * this.size;  
    this.y = Math.floor(Math.random() * (800 / this.size)) * this.size;  
}
```

## c) End Game

• **Mur**

Pour perdre il faut que le Snake touche un mur. Donc pour cela il faut faire une boucle if qui fait que si la position de la tête du snake est égal aux bordures ça arrête le jeu. Et ça affiche Game Over avec le score du joueur et ça dit d'appuyer sur F5 pour rafraîchir la page et recommencer le jeu.

```
if (snake.x < 0 || snake.x >= 800 / snake.size || snake.y < 0 || snake.y >= 800 / snake.size) {  
  // Ecrire game Over en rouge au milieu de l'ecran  
  ctx.font = '50px Arial';  
  ctx.fillStyle = 'Red';  
  ctx.fillText('Game Over', 270, 300);  
  //afficher le score en blanc avec les points  
  ctx.font = '35px Arial';  
  ctx.fillStyle = 'white';  
  ctx.fillText('Score : ' + Point, 325, 360);  
  //Afficher pour recommencer  
  ctx.font = '25px Arial';  
  ctx.fillStyle = 'white';  
  ctx.fillText('Appuyer sur F5 pour rejouer', 240, 420);  
  
  return;  
}
```

• **Corps**

Une autre façon de mourir est que la tête du snake touche son propre corps. Dès que la tête du snake touche son propre corps, automatiquement le jeu s'arrête et il l'affichage Game-over comme pour l'End-Game précédant.

```
for (let i = 0; i < snake.corps.length; i++) {  
  if (snake.x === snake.corps[i].x && snake.y === snake.corps[i].y) {  
    // Ecrire game Over en rouge au milieu de l'ecran  
    ctx.font = '50px Arial';  
    ctx.fillStyle = 'Red';  
    ctx.fillText('Game Over', 270, 300);  
    ...  
    return;  
  }  
}
```

## D. Test

Nom du test	Description	Etape de test	Résultat Attendu	Résultat actuelle
Direction	Changement de direction	1) Presser sur l'une des flèches de direction	Le snake doit aller du coté ou le testeur a appuyé	En appuyant sur l'une des flèches le snake va bien dans la direction souhaitée.
Score	Lorsque le snake mange une pomme les points augmente dans le conteur.	1) déplacer le snake 2) manger une pomme	Le conteur qui se situe en haut à gauche de l'écran augmente	Le conteur augment bien à chaque fois qu'une pomme est mangé
Grow	Lorsque le snake mange une pomme le snake grandi d'un bloc	1) déplacer le snake 2) manger une pomme	Dès que le snake mange la pomme son corps grandi d'un bloc	Le snake grandi bien d'un bloc en manger une pomme
End-Game (mur)	Lorsque le snake touche les bords du terrain de jeu, le jeu s'arrête et affiche Game over avec le score	1) déplacer le snake 2) toucher le bord du terrain	Lorsque le snake touche le bord, le jeu s'arrête et au milieu de l'écran il y a le message Game-over et le score et aussi le moyen de recommencer le jeu	Le jeu s'arrête et le Game-Over est bien afficher avec le score et le moyen de relancer une partie.
End-Game (mur)	Lorsque la tête du snake touche son propre corps , le jeu s'arrête et affiche Game over avec le score	1) déplacer le snake 2) manger 5-6 pommes 3) Foncer avec la tête du snake directement sur le corps	Dès que le snake touche son propres corps avec sa tête, le jeu s'arrête et au milieu de l'écran il y a le message Game-over et le score et aussi le moyen de recommencer le jeu	Le jeu s'arrête et le Game-Over est bien afficher avec le score et le moyen de relancer une partie.

## E. ChatGPT

Dans le cadre de ce projet les IA plus précisément ChatGPT ma surtout servit pour m'aider à comprendre des choses en JavaScript que je ne comprenais pas. Par exemple en cherchant comment faire grandir mon Snake j'ai trouvé une méthode qui s'appelle push, et sur internet plusieurs personnes l'utiliser donc j'ai demandé à ChatGPT qu'est-ce que ça fait et comment l'utiliser. Cela ma permit de comprendre un peu comment utiliser le push.

## F. Conclusion

Ce projet a permis d'apprendre le node JS afin de développer le jeu Snake. Pour cela on a dû faire pas mal de recherches, les sites javascript.info et developer.mozilla.org ont pour ma part beaucoup aidé à comprendre et découvrir de nouvelle chose. Comme citer dans le point précédant ChatGPT a pu aider à comprendre certaines choses plus facilement et plus rapidement donc ce fut un gain de temps. La seule chose qui m'a posé quelques problèmes c'était pour que le corps puisse grandir, mais après quelques recherches j'ai réussi à trouver



une méthode (push) qui ma permit de débloquer ce problème. Mais à part ça je n'ai pas eu de gros problème.

Pour ma part j'ai trouvé ce projet très sympa parce que je ne connaissais pas le JavaScript et ce projet m'a permis d'apprendre ce langage de façon assez ludique, car à la fin du projet on pouvait tester/jouer notre propre jeu. Et l'aide-mémoire va sûrement nous servir dans la future.

## G. Sources

<https://fr.javascript.info/>

<https://developer.mozilla.org/fr/docs/Web/JavaScript>

ChatGPT