Logotipo, Icono

Descripción generada automáticamente

**ECOLE TECHNIQUE  
ECOLE DES MÉTIERS LAUSANNE**

07 novembre 2023

Adrian toledo

Enseignant : Aurélie Curchod

« J-Snake »

Support de clase : P\_Bulles

Faire un réplica du célèbre jeu Snake

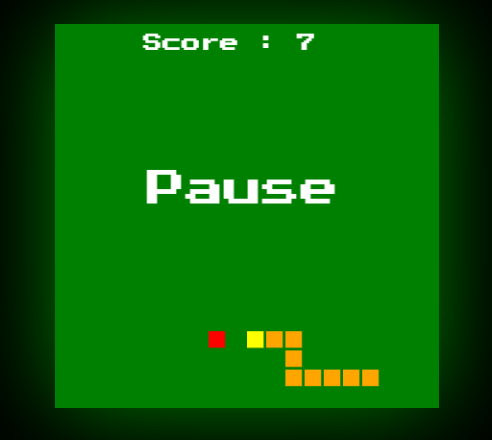


Table des matières

[1. P\_Bulle : Snake 3](#_Toc153888075)

[1.1. Introductio 3](#_Toc153888076)

[1.2. Objectif 3](#_Toc153888077)

[2. JavaScript 3](#_Toc153888078)

[2.1. IDE 3](#_Toc153888079)

[2.2. Introduction « Hello World » 3](#_Toc153888080)

[2.3. Fundamentos de JavaScript 3](#_Toc153888081)

[2.3.1. Variables 3](#_Toc153888082)

[2.3.2. Tipos de datos 4](#_Toc153888083)

[2.3.3. Array 4](#_Toc153888084)

[2.3.4. Conversiones 4](#_Toc153888085)

[2.3.5. Mateaticas - Operadores clasicos 4](#_Toc153888086)

[2.3.6. Comparadores , conditionales y operadores logicos 5](#_Toc153888087)

[2.3.7. Funciones 5](#_Toc153888088)

[2.3.8. Condiciones 5](#_Toc153888089)

[2.3.9. Bucles 5](#_Toc153888090)

[2.4. Interacciones 5](#_Toc153888091)

[2.5. Objects 5](#_Toc153888092)

[2.6. Inputs 6](#_Toc153888093)

[2.7. Import/export 6](#_Toc153888094)

[2.8. Tableau et rest 6](#_Toc153888095)

[2.9. Coding style 6](#_Toc153888096)

[3. J-Snake Game 7](#_Toc153888097)

[3.1. Menus de jeux vidéo 7](#_Toc153888098)

[3.2. Analyse du jeu en code 9](#_Toc153888099)

[3.2.1. Emulator – JS 9](#_Toc153888100)

[3.2.2. FruitClass - JS 9](#_Toc153888101)

[3.2.3. SnakeClass - JS 10](#_Toc153888102)

[3.2.4. Index -JS 11](#_Toc153888103)

[4. Conclusions 11](#_Toc153888104)

[4.1. Idées 11](#_Toc153888105)

[4.2. Améliorations 11](#_Toc153888106)

[5. Sources 11](#_Toc153888107)

[6. Annex 11](#_Toc153888108)

# P\_Bulle : Snake

## Introduction

El famoso videojuego Snake, lleva anos conviviendo en nuestra sociedad como un videojugo de culto por su impacto en la sociedad, simplidez de juego y adaptibilidad en cualquier dispositivo. A continuacion, este proyecto ofrezera una version simple del juego adaptado a una pagina web con JavaScript

## Objectif

Durante 8 semanas con un total de 5 periodes de 45 minutos, se debe realizar un proyecto libre en JavaScript que represente el famoso videojuego Snake. Este juego debe ser desarrollado con los siguiente elementos basico : movimiento, collisiones, score, interaccion de teclados y una syntaxis moderna y optimal para el juego.

# JavaScript

JavaScript es un lenguaje de programacion interpretados que permite inetrectuar y trabajar sobre los codigos HTML de las paginas web. Su origen viene del dialecto ECMAScripti qeu se s define como codigo orientado a obketos, basados en prototipos, débilmente tipado y dinámico.

Actualmente, los navegadores modernos soportan el dialecto ECMAScript5.1 para poder soportar los archivos javascirpt.

## IDE

Un IDE (Intergrated Development Enviroment) es un editor que soporta los proyectos completos y permite navegar entre archivos, provee autompletados, ambiente de pruebas y mas caracteristicas.

Para el uso de JavaScipt se requiere editores que puedan controlarel archivo y su contenido en el proyecto. Para este caso, usaremos un Microsoft Visual Code como IDE par construir el codigo Snake a medidas.

## Introduction « Hello World »

Declarar en el HTML

<script></script>

En caso de un archivo externo :

<script src="/path/to/script.js"></script>

Para mostrar un mensaje :

Alert(``Hello’’)

## Fundamentos de JavaScript

### Variables

* + - * + Let : se usa la declaracion LET para guardar variables. Este puede registrar tanto numeros como texto. Ex « let name = ‘adri’ »
        + Const : se usa la declaracion constante CONST para crear variables que su valor no vaya ser cambiado. Ejemplo :

Const COLRO\_RED= ‘#F00’

* + - * + Var : las declaraciones con VAR se utilizave recurrentemente para la declaracion de variables, debido a su falta de definicion y visibilidad exacta, fue remplazado por el LET. VAR tiene la desventaja que su uso dentro de bloques o methodos seran leido de forma global, por lo que no existe variables temporales. Su uso actual es para crear variables que formen parte del inicio del codigo. Ejemplo : « var name = ‘adri’ »

### Tipos de datos

* Numeros « let n=123 ; »
* Numero decimales « let n=12.3 ; »
* Numeros mayores a (253-1) « let number=254n ;»
* String, cadena de caracteres « let name = ‘adri’ » « let nameAge = ‘name adrian age ${age}’ //Se muestra name adrian age 24 »
* Boolean « let eating = false »
* Nul et undefined « let age = null ; let age2 = undefined »
* Typeof para conocer el tipo de una variable ex : «typeof age  // ‘number’ »

### Array

* Crear : « Let array = [1,2] »; « let array = new Array(item1, item2)
* Insertar : « array[1] = 3 » ;
* Longitud de la cadena : « array.length »
* Conocer el ultimo valor : « array.at(-1) »
* Pop – elimina el ultimo valor : « array.pop()»
* Push – anade un valor al final de la cadena : « array.push(7) »
* Shift – elimina el primer valor de la cadena : « array.shift() »
* Unishift – aande un valor al principio de la cadena : « array.unshift(5) »
* Matriz - let matrix = [[1, 2, 3], [4, 5, 6],[7, 8, 9]]; alert(matrix[1][1])
* ForEach : arr.forEach(function(item, index, array) { // ... do something with item });

### Conversiones

* ToString cambia un valor a tipo texto ejemplo : « let value = true ; value = String(value) »
* ToNumber cambia un valor a tipo numero ejemplo : « let value = ‘123’; value = Number(value) »
* ToNumber cambia un valor a tipo bolean ; « let value = 1 ; value=Boolean(value)

### Mateaticas - Operadores clasicos

* « let suma = 2+1 »
* « let resta = 2-1 »
* « let multi = 2\*2 »
* « let division = 2/1 »
* « let resto = 2%1 »
* « let exponencias = 2\*\*2 »
* «  let n +=5 »
* « let n++ ; »
* Random
* Math(). floor

### Comparadores , conditionales y operadores logicos

* Estructura de « if (){}else{} »
* Mayor que : « a>b » O mayor o igual que « a>=b »
* Menor que « b<a » o menor que « a <=b »
* Iugal « a==b » o no es iugal « a+=b »
* Operador OR « a||b »
* Operador AND « a %%b »
* Operador NOT « !a »
* Operador Nullish Coalescing « a ?? b », si el primer valor es nullo o desconocido, volvera el seugndo valor.

### Funciones

* Function showMessage(){}
* Let sum = (a,b) => a+b ;

### Condiciones

* If(condition){}
* Switch(xxxx){ case ‘xxxx’ : \_\_\_ [break] default : \_\_\_[break]}

### Bucles

* « While (condition) {} » Mientras la condicion sea cierta, se seguira procesando las iteraciones.
* « Do {} While (condition) » Primero ejecutara el contendio de la funcion y luego verificara si la condicion es verdadera para ejecutar de nuevo la funcion
* « For (let i=0 ; i >3 ; i++ ){} » en este bucle, se trabaja en funcion de la variables creada « i », donde hay que crearla « let i=0 », poner su condicion « i>3 » y la modificacion para el siguiente bucle « i++ »

## Interacciones

* Alert : se muestra un mensaje al usuario hasta que acepte

« alert(« ‘hello world’) »

* Prompt : acepta la introducciond de dos parametros donde el primero srivira para mostrar un mensaje y el seguno para registarar lo que el usuario quiera escribir.

« let test = prompt(‘Cuantos anos tienes ?’,’[respuesta]’)

* Confirm : muestra un mensaje y espera que el usuario elgia entre confirmar o cancelar.

## Objects

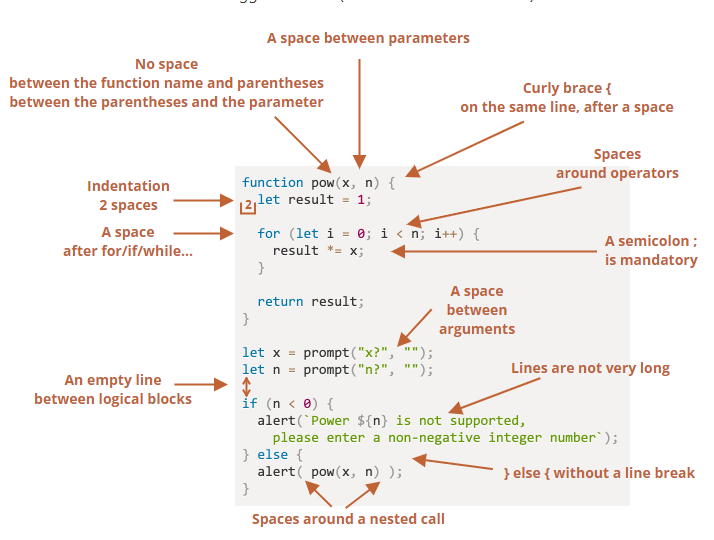
* Declarar : Let user = new Object() ; Let user = {}
* Propiedades : let user = {name : ‘Fede’, age : 30}
* Set propiedad : user[‘celibataire’] = true ;
* Get propiedad : user[‘age’] ;
* Delate propiedad : delate user [‘age’];

## Inputs

## Import/export

## Tableau et rest

## Coding style



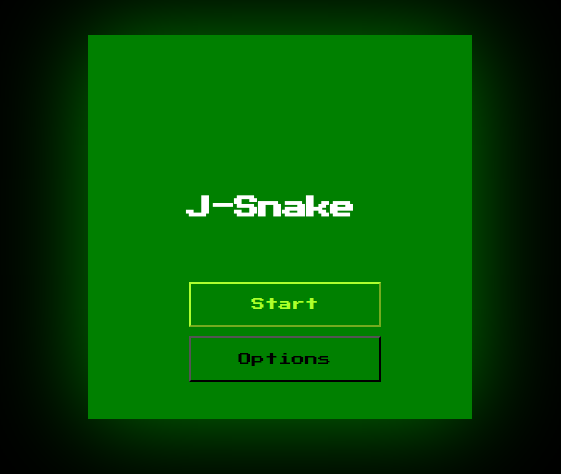
# J-Snake Game

J-Snake est une réplique du célèbre jeu du serpent sorti dans les années 1960. Dans ce jeu, le joueur contrôle un serpent dont l'objectif est de manger autant de pommes que possible sans se manger lui-même ou sortir des limites.

Ce jeu est connu pour être sorti sur une multitude d'appareils et de consoles, notamment le fameux Nokia 3310. Pour cette raison, ce site web offrira au joueur l'expérience de pouvoir choisir un émulateur comme le Nokia ou un GameBoy avec des boutons fonctionnels pour pouvoir jouer à J-Snake.

## Menus de J-Snake

J-Snake est présenté sur un écran 400x400 avec une couleur verte et un fond noir. Son premier écran affiche le titre du jeu et deux options, « Start » ou « Options ».



Sous Options, le joueur pourra choisir l'arrière-plan et la manette de jeu (Nokia, Gameboy ou aucune).



La première option initialise l'écran de jeu et n'est activée que lorsque le joueur décide de la direction du serpent. Au cours du jeu, le joueur doit manger le fruit qui lui rapportera 1 point et le score sera affiché.



Enfin, lorsque le jeu est perdu, le joueur peut redémarrer le jeu ou revenir au menu principal.



J-Snake se caractérise par une grande accessibilité des commandes de jeu. Dans les menus, vous pouvez utiliser les claviers pour sélectionner les options désirées, ou vous pouvez utiliser la souris ou les boutons de la manette de jeu du joueur.

Pour vous déplacer dans le jeu, vous pouvez utiliser les touches fléchées, les touches WASD ou cliquer sur les boutons de la manette de jeu.

## Analyse du jeu en code

J-Snake est un site web conçu avec Java Script, HTML et CSS. L'utilisation de Java Script a été concentrée sur le développement d'un moteur de jeu et de boutons fonctionnels. HTML a été utilisé pour la création de différents éléments qui structurent le site web et CSS a été utilisé pour gérer les couleurs, les positions, les interfaces et l'art visuel du site web.

|  |  |  |
| --- | --- | --- |
| HTML | CSS | JS |
| Index.html | css/style.css | js/emulator.js  js/fruitClass.js  js/snakeClass.js  js/index.js |

### Emulator – JS

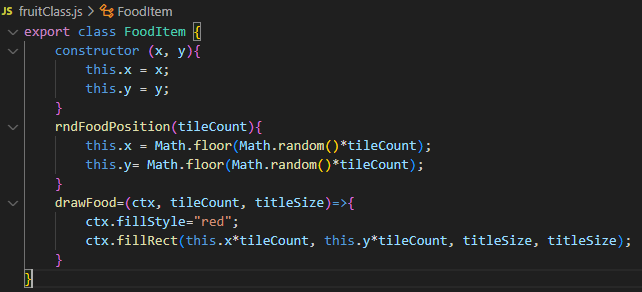
Le fichier emulator.js stocke et contrôle toutes les données relatives aux contrôleurs de jeu, à leurs boutons et à leurs images.

|  |  |
| --- | --- |
| emulator.js | Description |
| gameboyMode()  nokiaMode()  gameboyModeInOption()  nokiaModeInOptions()  noneModeInOptions() | Ces méthodes ont pour fonction d'afficher les images avec les boutons correspondants sur l'écran. Selon le mode choisi, l'un ou l'autre mode sera activé. |
| allHiddenAssets()  allDisplayAssets() | Méthodes pour cacher tous les éléments visuels indésirables ou les faire réapparaître |

### FruitClass - JS

Le fichier fruitClass.js contient les spécifications pour la création d'un élément comestible qui interagit avec le jeu.

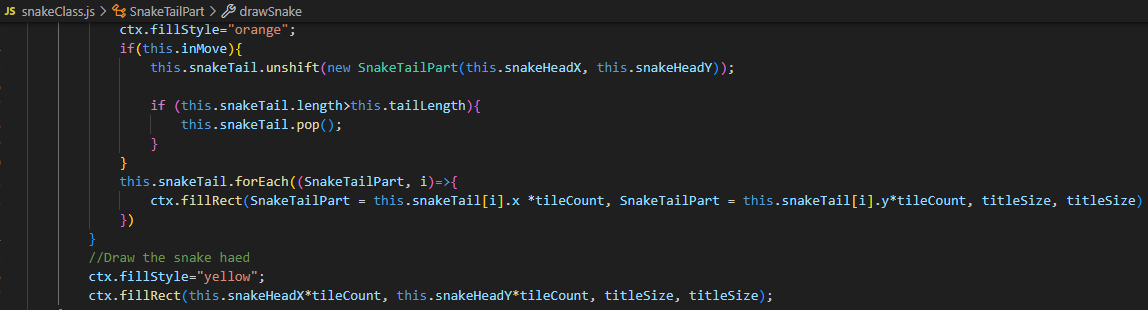
|  |  |
| --- | --- |
| fruitClass.js | Description |
| Class FoodItem   * Constructor (x, y) | Création de la classe fruit de avec ses éléments de position X et Y. |
| rndFoodPosition() | Méthode pour changer la position des fruits de manière aléatoire avec  Math.floor(Math.random()\*tileCount) |
| drawFood() | Méthode pour dessiner l'élément |



### SnakeClass - JS

Le fichier snakeClass.js contient les fonctionnalités nécessaires à la création du serpent contrôlé par le joueur. Ce serpent est divisé en deux parties, la tête et la queue, dans un tableau. En outre, il fournit d'autres fonctions pour le positionnement, le dessin et ses valeurs de base.

|  |  |
| --- | --- |
| snakeClass.js | Description |
| Class FoodItem   * Constructor (x, y) | Création de la classe Snake de avec ses éléments de position X et Y |
| snakePosition () | Méthode pour changer la position des fruits pour chaque update |
| drawSnake(ctx, tileCount, titleSize) | Méthode pour dessiner le snake. L'utilisation d'une boucle forEach() permet de mettre à jour la position de tous les corps dans la liste. Une fois la nouvelle position de la tête détectée, le jeu utilise unshift() pour ajouter de nouvelles informations pour la prochaine position prévue et la fonction pop() pour supprimer la dernière case dessinée. |
| SnakePauseMode() | Enregistre les informations de position du serpent lorsqu'il est en mode Pause |
| SnakeOrigin() | Rétablit les valeurs par défaut du serpent |



### Index -JS

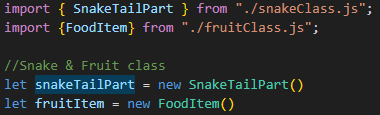
Le fichier index.js contient à la fois les fonctions destinées à faire fonctionner le moteur du jeu, les collisions et les interactions du HUD du jeu. En outre, il rejoint et interagit avec les autres fichiers à utiliser dans le jeu.

|  |  |
| --- | --- |
| indexClass.js | Description |
| GameEngine() | Dans cette méthode, il est chargé de développer le jeu. Son fonctionnement consiste à gérer l'état du jeu, qu'il soit en menu, en options, en pause ou en jeu, à mettre à jour les valeurs et à afficher chaque élément. |
| Menu() | Cette méthode définit le statut du jeu sur Menu uniquement et n'affiche que les éléments du menu : le titre et les boutons de lecture ou d'options. |
| Options() | Cette méthode permet d'ajuster l'état du jeu en mode Options où les éléments de sélection de l'arrière-plan, les manettes, les images et les boutons seront affichés. |
| IsGameOver() | Est une méthode qui renvoie l'état du jeu en fonction des conditions de défaite. Si le joueur entre en collision avec lui-même ou avec les limites de la carte, le jeu s'arrête et affiche Game Over, Score et deux boutons pour rejouer ou retourner au menu de démarrage. |
| PauseMode() | Dans ce mode, le jeu est mis en pause en enregistrant la dernière position du serpent et en affichant le message Pause. |
| Movement() | Dans cette méthode, le mouvement du joueur est déterminé en fonction de la direction donnée. |
| CheckCollision() | Pour contrôler si le joueur touche le fruit ou si le fruit apparaît à l'intérieur du serpent et doit être repositionné. |
| ChangeButtonMenu() | Méthode qui change la couleur des boutons sur l'écran pour savoir sur quelle option nous nous trouvons. |
| GameEnterBtn() | Méthode qui accepte l'option lorsqu'elle est sélectionnée. Selon l'état du jeu, vous pouvez accéder à l'option ou la confirmer. |
| ClearScreen() | Méthode chargé de nettoyer l'écran de tous ses éléments. |
| DrawScore() | Affiche le score du joueur à l'écran |
| BtnActions() | Liste d'actions pour les boutons du jeu ou les boutons de la manette utilisée. |

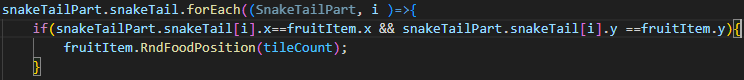
### JavaScript caractéristiques

JavaScript possède des fonctionnalités uniques qui permettent d’adapter le code pour les web site et de développer un code plus optimal avec une variété d’outils uniques. Beaucoup d’entre eux ont été expliqués dans la section 2 de ce document, et ci-dessous, une explication détaillée de la façon dont ces fonctionnalités ont été appliquées dans ce jeu :

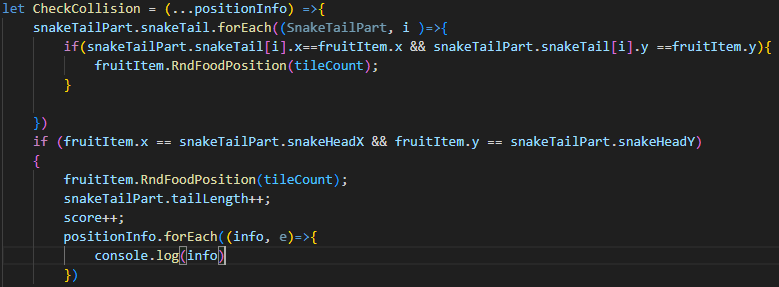
* Syntaxe moderne avec les variables const et let : En raison de l’intégration des variables Const et Let, il n’a pas été nécessaire de créer des variables de style Var dans tous les cas du code.
* Les classes Snake et Fruit ont été saisies avec leurs fichiers de type module correspondants afin qu’elles puissent être exportées et importées dans le code principal.

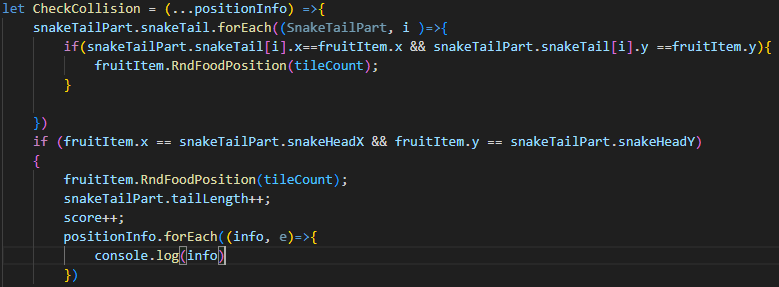


* Seules les boucles ForEach ont été utilisées pour le développement du jeu et les contrôles d’état. De même, les fonctions ont également été optimisées pour utiliser le système de fonctions avec des flèches afin d’optimiser le code.



* Enfin, l’opérateur Rest a été utilisé comme élément pour pouvoir afficher les données dans la console en format de tableau.





# Conclusions

J-Snake est mon deuxième projet de site web en java script, où j'ai pu apprendre de nouvelles fonctionnalités de ce langage et compléter mes connaissances antérieures.

La construction du jeu Snake a été développée facilement grâce à mes connaissances antérieures de ce jeu en C#. En raison de l'utilisation de javascript, l'adaptation de ce code a présenté des difficultés telles que les interactions avec les boutons, un ordre correct dans la structure du code, l'utilisation de différentes fonctions et de nouveaux outils pour les tableaux. Mais une fois que j'ai appris à utiliser ces nouveaux éléments, j'ai pu développer le jeu complètement et librement selon mes idées.

C'est pour cette raison que le développement d'un jeu simple avec une liberté totale de réalisation m'a rendu curieux de ce langage et m'a donné envie de développer plus d'éléments dans le jeu et d'apporter de l'originalité.

## Améliorations

Voici une liste d'idées et d'améliorations diverses que, par manque de temps, je n'ai pas pu appliquer à mon projet :

* Créer une classe pour chaque élément : Score, boutons de contrôle, HUD et UI.
* Utiliser RequestAnimationFrame comme alternative à SetTimeout()
* Avoir un site web responsive qui soit correct et adaptable à l'écran.
* Bugs : en mode pause, un élément peut être temporairement supprimé à l'écran.

# Sources

* W3School : <https://www.w3schools.com/>
* Javascript : [www.javascript.info](http://www.javascript.info)
* Youtube - Pokemon JavaScript : <https://www.youtube.com/watch?v=yP5DKzriqXA&t=7928s>
* Chat GPT n'a pas été utilisé pour la réalisation de ce projet