

Création d’un jeu snake

Technique Web



03 février 2020

Boinega Antonin, Dautheville Dylan

# Table des matières

[I. Table des matières 1](#_Toc31661226)

[II. Introduction 1](#_Toc31661227)

[III. Le principe d’un snake 1](#_Toc31661228)

[IV. Le modèle MVC 2](#_Toc31661229)

[V. Déroulement d’une partie 2](#_Toc31661230)

[VI. Choix dans le code 3](#_Toc31661231)

[VII. Amélioration possible 3](#_Toc31661232)

# Introduction

Ce document traitera de la réalisation d’un jeu « Snake », à l’aide du langage Web Javascript.

Il fait suite au cours de Monsieur Jean, « Technique WEB », qui concernais l’utilisation des langages WEB et plus précisément le langage Javascript.

Ce TP à étais réalisé par Monsieur Boinega et Monsieur Dautheville qui ont travaillé en collaboration dans la réalisation de ce projet.

# Le principe d’un snake

D’après wikipédia :

« Le snake, de l'anglais signifiant « serpent », est un genre de jeu vidéo dans lequel le joueur dirige une ligne qui grandit et constitue ainsi elle-même un obstacle. Bien que le concept tire son origine du jeu vidéo d'arcade Blockade, il n'existe pas de version standard. Son concept simple l'a amené à être porté sur l'ensemble des plates-formes de jeu existantes sous des noms de clone. »

Le snake est un jeu très simple et très rependu, ou le joueur incarne un serpent dont le but et de manger des fruits pour grandir (et faire monter le score), plus le serpent mange, plus il grandit et plus cela devient compliqué de rester en vie. En effet si le serpent se cogne la tête contre le mur ou qu’il se mange la queue, le serpent meurt et la partie est perdu.

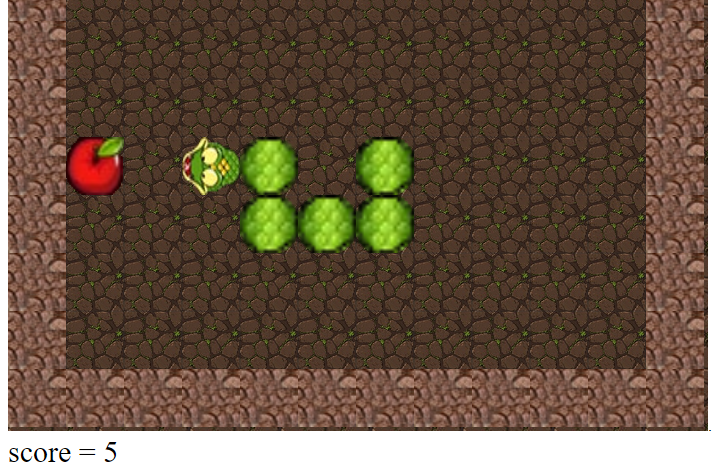


Figure 1 Capture d’écran de l'application

# Le modèle MVC

Pour la réalisation de ce TP, une des seules contraintes fonctionnelles étais l’utilisation du modèle MVC (model view Controller). C’est une organisation très répandue qui consiste à répartir les classes selon leurs rôles, ainsi nous avons :

* Des classes Model, ce sont les classes métier, ils représentent l’objet et la data et seras appelé par le Controller, dans notre exemple nous avons deux classes Model, une représentante plusieurs objet tel que la grille, la position de la nourriture, la position de la tête, le premier élément de la queue et une autre classe Model Queue représentation un élément de la queue.
* Une View, qui représentent l’affichage, c’est dans cette classes on gère l’affichage de l’interface avec lequel le joueur devra interagir pour jouer.
* Une classe Controller, les classes Controller ont pour rôle de faire les appel au deux classes cité précédemment (Model et View), et de gérer le déroulement de la partie, il réceptionneras les demandes de l’utilisateur (via des appuies sur les touches directionnel du clavier) et les retransmettras dans son code pour faire respecter les demandes.

# Déroulement d’une partie

# Choix dans le code

Dans le code nous avons fait plusieurs choix, tout d’abord lorsque le serpent mange un fruit un nouvel élément découlant de ModelQueue est rattacher directement à la tête si le serpent avait déjà une queue le reste de la queue se rattache à ce nouvel élément.

A chaque frame la queue doit s’actualiser, la méthode que nous avons utilisé un plutôt optimisé puisque à part pour le premier élément de la queue qui doit s’adapter à la tête les autres éléments n’ont qu’a prendre la position de son élément précédent pour s’actualiser.

Nous avons décidé dans le Model de ne pas faire apparaitre la tête du serpent sur la grille pour ne pas a voir a gérer le cas de la tête de serpent qui prend la position d’un mur ou d’un fruit ou de sa queue même si c’est relativement facile à développer.

Également nous avons décidé de charger toutes les images aux préalables avant le lancement de l’application grâce à une fonction de callback.

Nous avons essayé de mettre en œuvre la plupart des pratiques que vous nous avez enseigné en cours

# Amélioration possible

Séparer encore plus en profondeur les classes Model en dissociant la grille du serpent et de la nourriture.

# Conclusion

Ce projet nous a permis de pratique plus en profondeur le javascript, qui est une initiation pour nous, nous avons essayer de respecter l’ensemble du projet mais nous regrettons également de ne pas avoir fait une architecture parfaite.

Dans l’ensemble nous sommes plutôt satisfait du travail réalisé.