# **Browser game : Snake Game**



Le but de ce projet est de créer une application sur un Browser Web en utilisant comme interface utilisateur deux boîtiers Nordic Thingy. Le jeu choisi est une réplique du célèbre « Snake ».

## Concept du jeu

Le joueur incarne un serpent affamé qui se déplace dans un jardin entouré de buissons à la recherche de pommes. Le but du jeu est de manger un maximum de pommes, qui apparaissent aléatoirement, sans jamais se mordre la queue ou percuter un buisson.

L’utilisateur dirige le serpent à l’aide de 2 Thingies. L’un fait tourner le serpent dans le sens horaire (Thingy rouge) et l’autre dans le sens anti-horaire (Thingy vert). Chaque pomme mangée incrémente le score et la vitesse de déplacement du serpent accélère toutes les 2 pommes.

## Design

L’écran de jeu est divisé en 2 zones :

* Un bandeau supérieur, contenant les boutons « Start/Pause », « Restart », le titre du jeu, le score et le meilleur score de la session
* La zone de jeu, qui est constituée de 14x23 cases avec une résolution de 80x80 pixels

Au démarrage, le serpent n’est composé que de 3 parties (tête-corps-queue), l’utilisateur doit cliquer sur le bouton « Start » pour lancer le jeu. Lorsque le serpent mange une pomme, il faut attendre qu’elle arrive jusqu’au bout de la queue pour qu’il grandisse. On peut d’ailleurs suivre la progression de la pomme dans tout le corps du serpent pendant qu’il se déplace.

Figure  : Exemple d’une partie de Snake

Les scores (actuel et meilleur) restent affichés en permanence en haut à droite de la page de jeu.

En cas de *game over*, pour relancer une partie il suffit de cliquer sur le bouton « restart » pour réinitialiser le jeu, puis sur le bouton « start » pour démarrer une nouvelle partie.

Il est possible de mettre le jeu en pause à tout moment en cliquant sur le bouton « pause ». Dans ce cas les LED des Thingies passent au bleu. Pour relancer le jeu il suffit de cliquer à nouveau sur « start », et les Thingies reprennent leur couleurs d’origine.

#### Résumé des codes couleurs

|  |  |
| --- | --- |
| Couleur des Thingies | Signification |
| 2x bleu | mode « Pause » activé |
| 1x rouge + 1x vert | jeu en cours |
| 2x éteins | mode « Game Over » |
| 2x violet | chargement de la page en cours |
| Thingy rouge | rotation sens anti-horaire |
| Thingy vert | rotation sens horaire |

## Implémentation

Le code est séparé en 3 fichiers

#### client.html

Ce fichier gère la partie browser de l’application. Il définit les différentes zones d’affichage et leurs tailles, mais également les styles des boutons et de l’écriture des scores. Les scripts contenant le code de l’application sont implémentés dans les 2 autres fichiers.

#### client.js

Ce fichier, écrit en javascript, contient tout le code concernant la communication avec le broker MQTT. Les fonctions contenues dans ce fichier sont :

* **client.onMessageArrived = function(message)**

est appelée lors du modification du broker MQTT, cela permet d’informer le jeu lors d’un appui sur les boutons des Thingies.

* **client.connect**

connecte la page internet avec le broker MQTT

* **function setLED(n,red,green,blue)**

permet de modifier les couleurs des Thingies via des data envoyées au broker MQTT

#### game.js

Ce fichier, écrit en javascript, contient tout le code concernant le jeu. Les paramètres du jeu sont stockés ici. Les fonctions implémentées par ce fichier sont :

* **function startStop()**

permet de démarrer le jeu et de le mettre en pause, lors de la mise en pause le symbole «  » apparaît au centre de l’écran. met également à jour les couleurs des Thingies en fonction de l’état du jeu (run/pause)

* **function reStart()**

permet de réinitialiser toutes les variables du jeu afin de commencer/recommencer une nouvelle partie

* **function runGame()**

est appelé en boucle afin de faire tourner le jeu

* **function checkCollision()**

vérifie si le serpent n’enfreint pas les règles permettant de continue la partie, teste également s’il va manger une pomme

* **function move()**

si le serpent n’a pas enfreint de règles, effectue le déplacement du serpent et met à jour les parties de son corps avec les bonnes images à la bonne place

* **function turnL()**

modifie le sens du déplacement du serpent de 90° dans le sens anti-horaire

* **function turnR()**

modifie le sens du déplacement du serpent de 90° dans le sens horaire

* **function addMiam(miamPos)**

place une nouvelle pomme aléatoirement dans la zone de jeu

* **function drawGame()**

affiche les images du jeu

* **function drawRotatedImage ( image , x , y , angle )**

permet d’afficher une image à une position définie et avec un angle de rotation, l’image sera centrée sur cette position. L’image aura une hauteur et une largeur de la résolution du jeu

* **function drawImage ( image , x , y )**

permet d’afficher une image à une position définie, l’image sera centrée sur cette position

* **function upVitesse()**

diminue le temps entre 2 animations de 20ms

* **function theEnd()**

est appelé en cas d’infraction aux règles. Mets fin à la partie

## Problèmes rencontrés

Au départ je n’avais pas prévu de faire les sprites de rotation du serpent, mais lors d’un virage du serpent l’animation n’était pas à la hauteur de mes attentes. J’ai donc dû redessiner des versions en rotation et les implémenter à mon code cela a créé beaucoup de modifications afin de prendre en compte tous les cas possibles.

Les caractéristiques inhérentes du codage en javascript, telles que le non-typage des variables, des fonctions usuelles non disponibles, ou encore la définition des fonctions et de leur paramètres m’ont posé quelques problèmes car elles n’utilisent pas les mêmes standards que tous les principaux langage de programmation.

## Instructions de paramétrage

1. modifier dans le fichier client.js les adresses des Thingies, le username et le password qui vous sont associé
2. connecter les Thingies au broker MQTT via le Raspberry Pi
3. lancer le jeu en double cliquant sur le fichier client.html
4. Amusez-vous bien !

## Conclusion

Le jeu est fonctionnel et réagit parfaitement aux commandes. Il m’a permis d’apprendre à coder en langage javascript et HTML, je suis à présent un peu plus à l’aise avec ces langages.

Malgré les nombreux problèmes rencontrés lors du développement de mon jeu, je suis fier du résultat final.

Figure : Exemple de Game Over