Mise en place du projet GitHub

Installation de node.js et des modules nécéssaires (express...)

Mise en place d'un serveur simple via node.js

Réorganisation du dossier GitHub, création de dossiers public/private

Passage du site en https, génération des clés et certificats, installation des modules fs, https et openssl à l'aide de cygwin et console.

_____ TP2 _____

Exclusion du dossier private (clés/certificats) de GitHub pour raisons de sécurité, tâche assez rapide.

Installation de paper.js dans le projet, consultation de tutoriels sur le site officiel de paper.js dans le but d'en comprendre le fonctionnement.

Correction d'un bug dans le code du serveur réussie, on obtient désormais une page avec un canvas paper.js.

Intégration de fonctions de "dessin" paper.js dans index.html.

Test non fructueux, recherche de solutions. Adaptation de l'élément canvas à toute la page pendant ce temps.

Dessin opérationnel, début du travail d'animation de déplacement vectoriel. Expérimentation de features mineures (changement de couleur en temps réel avec onFrame)

Déplacement fonctionnel, une partie assez ardue mais il est désormais possible de déplacer le snake à vitesse constante via clics de souris comme attendu. Le snake est composé de nombreux cercles copiés du cercle originel qui sont rafraîchis régulièrement.

Cleanup de fonctions obsolètes commentées pour éclaircir le code. Modifications du CSS pour définir un cadre autour du canvas.

Début de la mise en place du server WebSocket.

_____ TP3 _____

Cleanup rapide du code du fichier server.js via JSLint.

Revert de quelques modifications apportées (on déplacera le snake à l'aide de clics de souris et non pas via la position temps réel de la souris pour un souci de simplicité dans les échanges client/serveur)

Suppression du dossier private sur GitHub (il avait été exclu des commits lors du TP

précédent mais pas retiré)

Consultation de divers tutoriels WebSockets, questions sur StackOverflow... pour mieux comprendre son potentiel fonctionnement dans le projet.

Mise en place de la possibilité de communiquer entre le client et le serveur via WebSocket, réflexion sur l'implémentation de l'animation avec cette nouvelle méthode.

Ajouts de quelques petites fonctions pour une meilleure intégration multi-joueur (différenciation des différents snakes avec couleur aléatoire, départ à un point aléatoire du canvas), des difficultés à trouver la manière pour réussir à implanter correctement le mouvement de plusieurs joueurs côté serveur.

Ajout de nombreux commentaires dans le but d'éclaircir le code. Restructuration majeure des fonctions, mise en place d'un "échaufaudage" pour le code client/serveur encore non fonctionnel.

On peut désormais passer les données de position des serpents et sa direction entre les clients et le serveur, grâce à un broadcast timé via setInterval, qui envoie à tous les clients un tableau avec les données de chaque serpent.Un des problêmes que j'ai rencontré était le fait de ne pas pouvoir envoyer de tableau entre le client et le serveur, mais grâce aux fonctions JSON.stringify et JSON.parse, j'ai pu résoudre cette issue. Reste à résoudre le problême de l'affichage de ceux-ci car les objets, transformés en JSON et reformés semblent ne pas pouvoir être clonés, mais globalement le problême semble être facilement gérable avec plus de recherches. Cette partie s'est avérée difficile à matérialiser dans l'idée mais une fois le principe compris, la réalisation en trial&error était assez triviale bien que assez longue pour obtenir un fonctionnement sans problêmes.