Carnet de Laboratoire

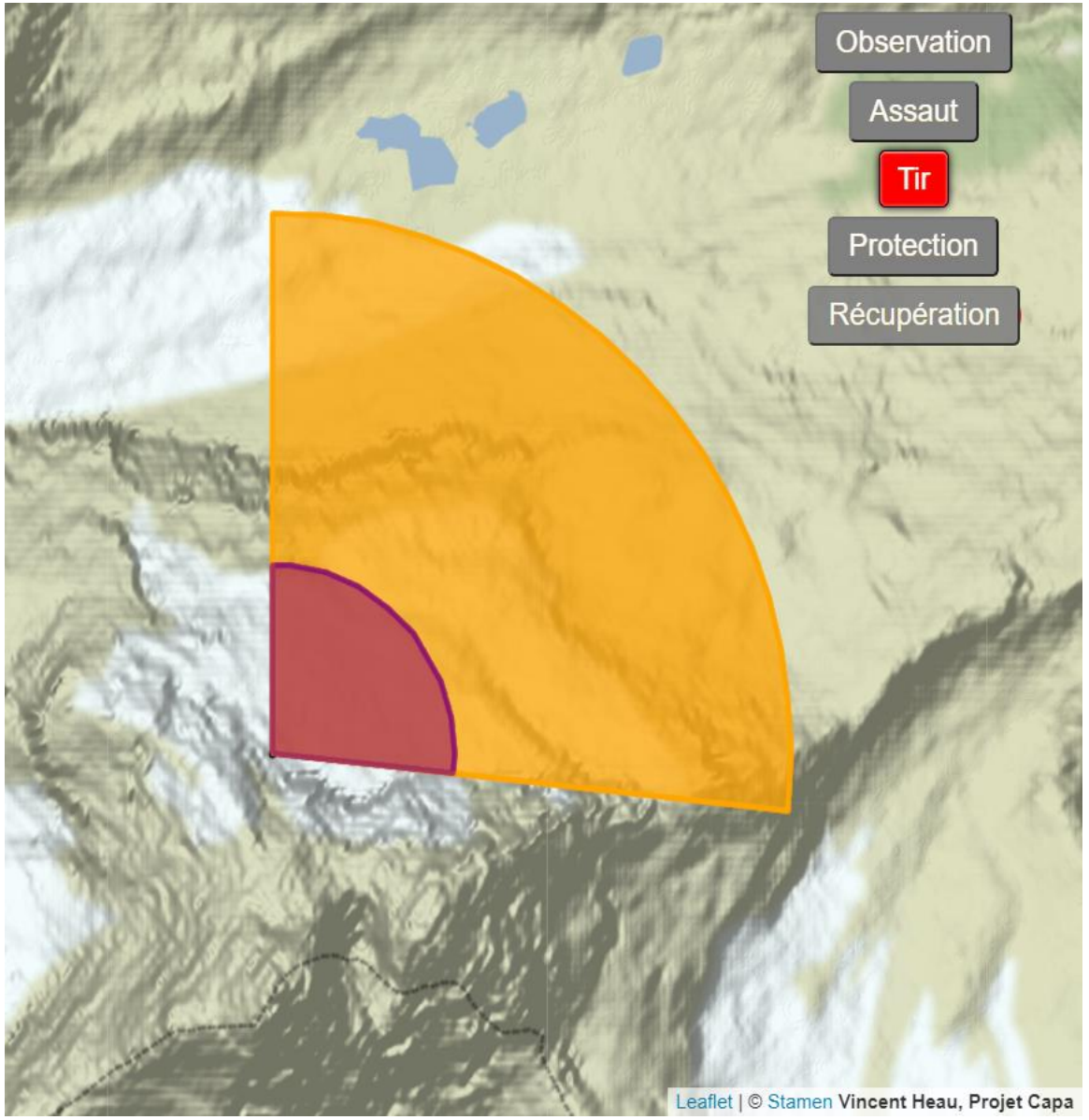
Projet CAPA

Sujet 2 :

Préparation de la requête de tir et extension aux autres fonctionnalités du jeu (Observation, Assaut, Protection, Récupération)

HEAU Vincent

13/12/2021 - 01/02/2022



Lundi 13 décembre 2021

matin

Découverte du sujet et du groupe

(chef de projet: Quentin)

Prise en main des différentes fonctionnalités de jeu, on se rend compte que le jeu devient difficilement jouable si on utilise simplement des requêtes ajax. En effet, les effets des actions des joueurs doivent être immédiat et on doit passer à une communication bidirectionnelle entre client et serveur Le choix fait par le groupe précédent d'utiliser les web sockets semble être donc judicieux car les websockets permettent une mise à jour en temps réel.

Création d'un Github pour le partage de code.

Nous passons donc un peu de temps ensemble à essayer de comprendre le fonctionnement des websockets.

Lundi 13 décembre 2021

après-midi

Nous décidons toujours de travailler tous ensembles en créant une base de donnée en ligne pour notre projet. Pour cela, nous utilisons Always DATA, ce qui nous permet non seulement d'héberger notre site en ligne en ligne mais aussi d'héberger la base de donnée.

– Création de la base de données selon l'architecture que nous discutons cette après-midi.

Pour prendre en main le code des websockets, nous décidons de créer un chat d'après un exercice de Vincent Da Olivera (https://ensg_dei.gitlab.io/web-az/js/nodejs/). Nous installons NodeJS sur nos machines.

Mardi 14 décembre 2021

matin

Le chat crée avec nodeJS hier est fonctionnel. Cependant, il marche en local sur nos machines (il est possible de voir les requêtes envoyées par le serveur en communiquant avec plusieurs onglets localhost)

Pour que le jeu puisse fonctionner et qui plus est sur téléphone, il faut que ce site node Js soit en ligne. Nous décidons donc ensemble de mettre le site en ligne sur Always Data.

Mais avant cela et après avoir passé une petite heure à définir clairement les 4 sujets, nous décidons de nous répartir le travail. Je travaillerai donc sur le sujet2 (requête de tir). Cependant, nous décidons de ne pas encore commencer à coder ou à mettre en place l'architecture de nos parties car nous voulons d'abord mettre notre chat en ligne sur Always data.

Mardi 14 décembre 2021

après-midi

Avant d'attaquer la mise en ligne, nous passons une bonne heure à regarder les codes des élèves des années précédentes. Nous préférons tout recommencer.

La mise en ligne s'avère particulièrement difficile, plein de bug, nous passons tous notre après-midi sur StackOverflow ou à regarder des vidéos sur YouTube pour réussir la manipulation. A la fin de la journée, il y a toujours un bug, pourtant nous sentons que nous sommes tous proches.

Vendredi 17 décembre 2021

matin

On reprend à zéro la mise en ligne de notre chat sur Always Data et par miracle, celle-ci fonctionne. C'est le moment de commencer à coder.

Pour moi la première chose c'est de créer une interface de jeu. Alexandre doit s'occuper de l'interface et le jeu final comprendra son interface mais je ne peux coder sans interface. Je décide donc de me lancer dans une interface.

Vacances

- Gros problème de la séparation de des différents fichiers de codes (à expliquer en détail) je compte aller voir Vincent Da Olivera à la rentrée pour mieux comprendre ce problème.

- Il n'est pas dérangeant en soi mais il handicape beaucoup la lisibilité du code et sa réutilisabilité ce qui pose problème. De même, une séparation des fichiers permet de mieux débugger.

Malgré tout, je décide après avoir passé deux journées sur ce problème de ne plus perdre de temps et d'attendre de voir Vincent Da Olivera à la rentrée. Je code donc sur le fichier index.html directement

Vacances

- Mise en place du CSS selon la grille suivante

(le css n'est pas à priori ma partie dans ce projet mais j'ai besoin d'avoir une interface graphique assez poussée pour pouvoir bien tester mes fonctions)

Le plan du css sous forme de schéma figure dans la documentation programmeur.

Vacances

- Je décide de coder le secteur de tir. Pour fabriquer ce secteur, j'envisage plusieurs possibilités
- La première, trouver de quoi faire un polygone sur leaflet, ce que je ne trouve malheureusement pas.
- J'examine la méthode de Nathan Wollf réalisée deux ans auparavant. Elle est judicieuse dans la mesure où son polygone ressemble quasiment à un cône mais le code est assez lourd (il doit être le plus fin possible dans la partie arrondie du secteur) et les calculs mathématiques le sont aussi.

Pour faire des intersections, je découvre turf mais je ne peux malheureusement pas intersecter un cercle avec un triangle (qui est un polygone), la fonction intersect ne marche qu'avec des polygones.

- J'ai beaucoup de chance et je me rend compte que j'aurai du faire cela en premier, en parcourant la librairie turf, je découvre le fonction sector faite pour cela

Vacances

- Envoie de la requête de tir lorsque le bouton de tir est relâché.

Partie importante car très calculatoire et qui comporte tous les aspects mathématiques du jeu.

- Création rapide d'un logo et de deux couleurs (histoire d'harmoniser les choses)

Jeudi 6 janvier 2021

- Première journée après les vacances où nous pouvons travailler ensemble. Un des membres du groupe n'est malheureusement pas présent à cause du covid. Nous décidons à ce moment de légèrement changer nos plans. Je rajoute à ma partie le code de la fonction protection en plus du tir. De plus , nous avons également convenu que je m'occupait des affichages pour les parties assaut et protection.
- Nous n'arrivons pas à séparer le code avec nodeJS. (voir annexe)

Jeudi 13 janvier 2021

- Deuxième séance de travail en groupe. De mon côté, j'ai été comprendre l'histoire des chemins en allant voir Victor Coindet et Vincent Da Olivera. Grâce à leur conseil et l'aide de StackOverFlow, je suis parvenu à mettre en place un code permettant de lire les fichiers css et js depuis le fichier html. Le problème est enfin réglé et le code va maintenant pouvoir être plus propre et organisé. Par dessus tout, il va être possible de rajouter des images ce qui est un avantage majeur.
- Ce jeudi après-midi, je partage donc mon code aux autres membres de mon groupe.

Jeudi 20 janvier 2021

- Ma partie est quasiment finie, j'ai ajouté un mode de jeu permettant de tester le tir lorsque l'orientation n'est pas récupérée.
- Pour l'assaut, j'ai mis en place l'affichage par un rond rouge qui correspond à la zone d'assaut.
- Je code également la protection de telle sorte que lorsque l'on appuie sur le bouton, il devienne vert et lorsque l'on appuie dessus il redevient gris. (pour la partie affichage). De plus la protection est matérialisée par un cône bleu de 88m orienté dans la direction du joueur.

En ce qui concerne le code, je ne sais pas pour le moment si la version qui reconnaît l'orientation du téléphone fonctionne. Je décide donc pour les jours qui suivent, de coder une « version de secours » qui permette de visualiser la protection sur les ordinateurs de l'ENSG par exemple.

Février 2022

- Difficulté pour mettre en commun nos parties car les avancements des membres du groupe sont différents.
- Une autre difficulté est l'installation en local sur les machines de l'IGN. Nous allons voir Vincent Da Olivera pour comprendre l'erreur lors de l'exécution de la commande

node capa.js

indispensable au lancement du serveur qui permet d'afficher notre jeu dans un navigateur en tapant

localhost/capa.html

- Vincent Da Olivera nous conseille de ne pas passer trop de temps à chercher à régler ce problème. Il faut donc procéder à l'installation sur un ordinateur portable.

- Nous parvenons finalement à séparer le code avec nodeJS pour le rendre plus présentable. La syntaxe présente au début du fichier capa.js permet de faire comprendre à Node que certains fichiers ne sont pas du javascript car sa configuration lui empêchait de lire des images et autres fichiers.

Finalisation du projet, derniers jours de travail

Nous avançons chacun dans nos parties mais la mise en commun est assez compliquée du fait de l'utilisation de nodeJS. Le passage du travail en local à un site entièrement en ligne est également une difficulté.

Le Sujet 2 sur lequel j'ai travaillé demandais de mettre en place la requête de tir. L'affichage a été fait avec l'utilisation du module Turf (option cône) pour éviter de créer dans le code un énorme polygone.
Pour essayer au maximum d'avoir un jeu qui fonctionne, nous avons décidé d'étendre ma partie à la gestion du tir. J'ai donc utilisé les Websockets pour mettre à jour la base de données lors d'un tir. J'ai également réalisé une interface graphique permettant de visualiser les résultats de mon code. Cette interface doit s'apparenter au maximum à celle du jeu Capa futur.

Deux difficultés majeures :

- Bien que responsive, la jouabilité sur téléphone parce que les boutons fonctionnent mal en tactile. L'idéal serait de coder avec **Android Studio** en java comme nous l'avons vu lors des cours d'application mobile en février.
- La précision de la position doit être suffisamment bonne pour que le jeu soit jouable car sinon les tir à seulement 80m n'ont plus de sens.

Amélioration possibles des règles du jeu : A l'entrée du jeu, le participant pourrait choisir un emplacement relatif et ainsi les gens pourraient bouger selon leur emplacement relatif ce qui permettrait de jouer ensemble même à grande distance.

Annexe A

13 Answers

Active | Oldest | Votes

This code:

60

```
app.all('*', function (req, res) {
  res.sendFile(__dirname+'/index.html') /* <= Where my ng-view is located */
})
```

tells Express that no matter what the browser requests, your server should return `index.html`. So when the browser requests JavaScript files like `jquery-x.y.z.main.js` or `angular.min.js`, your server is returning the contents of `index.html`, which start with `<!DOCTYPE html>`, which causes the JavaScript error.

Your code inside the callback should be looking at the request to determine which file to send back, and/or you should be using a different path pattern with `app.all`. See the routing guide for details.

Share Improve this answer Follow

edited Mar 6 '15 at 7:40

answered Mar 6 '15 at 7:33

T.J. Crowder

925k ● 169 ● 1702 ● 1715

Uncaught SyntaxError: expected expression, got '<' [Learn More] carte.js:1

ERREUR Obtenue lorsqu'on tente de séparer le js, du html et du css.

Cela s'explique par le fait que Node ne comprend que le js par défaut. Lorsqu'on met des fichiers qui ne sont pas du js, il faut le préciser sinon, il affiche des erreurs.

Par exemple l'erreur ci-contre crée par un document html.

Le problème est réglé via le début du code capa.js

Annexe B

TURF

GETTING STARTED

Search modules

MEASUREMENT
along
area
bbox
bboxPolygon
bearing
center
centerOfMass
centroid
destination
distance
envelope
length
midpoint
pointOnFeature
polygonTangents
pointToLineDistance
rhumbBearing
rhumbDestination
rhumbDistance
square
greatCircle

COORDINATE MUTATION
cleanCoords
flip
rewind
round
truncate

TRANSFORMATION
bboxClip
bezierSpline
buffer

sector

Creates a circular sector of a circle of given radius and center Point , between (clockwise) bearing1 and bearing2; 0 bearing is North of center point, positive clockwise.

Arguments

Argument	Type	Description
center	Coord	center point
radius	number	radius of the circle
bearing1	number	angle, in decimal degrees, of the first radius of the sector
bearing2	number	angle, in decimal degrees, of the second radius of the sector
options	Object	Optional parameters: see below

Options

Prop	Type	Default	Description
units	string	'kilometers'	miles, kilometers, degrees, or radians
steps	number	64	number of steps
properties	Properties	{}	Translate properties to Feature Polygon

Returns

Feature <Polygon> - sector polygon

Example

```
var center = turf.point([-75, 40]);
var radius = 5;
var bearing1 = 25;
var bearing2 = 45;

var sector = turf.sector(center, radius, bearing1, bearing2);
```

Références et documentations utilisées

liste non exhaustive

- Turf.js [https://turfjs.org/docs/]
- NodeJS [https://nodejs.org/en/]
- AlwaysData [https://www.alwaysdata.com/fr/]
- PostgreSQL [https://www.postgresql.org/]

Le code ainsi que les documentations sont présentes à l'adresse:
https://github.com/VincentHeau/CAPA2021_HEAU_Sujet2.git

La version en ligne du site se trouve à l'adresse:
https://vincentheau-capa.alwaysdata.net/capa.html

CAPA