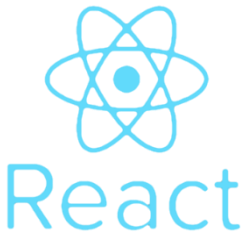
Compte Rendu final
Projet JS

YOFCLAN



Sommaire

Sommaire

Introduction :

Introduction au projet.

Le challenge JS a commencé le 3 avril 2023 avec pour seul consigne de créer un jeu avec les langages suivant : HTML/CSS/JAVASCRIPT.

Ainsi nous avons donc dû réaliser notre propre cahier des charges.

Rappel du cahier des charges.

Le projet YofClan avait pour objectif d'être un jeu de création d'un village. Le village devait être évolutif et devait remplir différents objectifs :

- Pouvoir acheter plusieurs bâtiments
- Plusieurs types de bâtiment
- Pouvoir améliorer les bâtiments
- Avoir la possibilité d'attaquer
- La création de différentes troupes
- L'affichage des bâtiments
- Une sauvegarde fonctionnelle du village
- Un affichage des ressources

Problématique :

Comment réalisé un jeu qui utilise du HTM/CSS/JavaScript sur un délaie d'1 mois tout en respectant les différentes contraintes ?

Rappel travail effectuer première semaine 07/04/2023

Objectif :

- Création des menus et différents boutons
- structure de html/css
- implémentation des bâtiments.

Objectif Annex :

- Création des bâtiments
- Faire la musique (chez moi)

Ainsi lors de la première semaine ont été créer les bases du jeu.

Le html / css a donc été implémenté avec un début de structure du village.

Le village se situe dans un canvas qui sera amené à être rempli par les bâtiments de l'utilisateurs

La première semaine a été aussi celle de l'implémentation de bouton qui provienne du frameWork JS Mantine.

Les boutons sont affichés et caché en fonction de si on clique dessus.

La création du shop a également été faites.

L'implémentation de barres qui contiennent les valeurs des différentes ressources et qui est progressive.

Rappel travail semaine2 16/04/2023

Objectif à atteindre :

- Création des instances.

-Bâtiments.

-Troupes

- Gestion des différentes

-Limitations et fonctions

- Limites de ressource

- Ajout de ressource en fonction du temps

- Limites des bâtiments.

-Ressource récoltée

Objectif bis

-Faire la musique (chez moi)

La deuxième semaine a donc été celle de la création de toutes les instances et des prémices des fonction principales.

On retrouve donc l'architecture de fichier suivante.

On retrouve donc toutes les fichiers css omis ceux qui sont liées a

L'App.js.

On retrouve également un dossier élément qui contient un premier dossier allInstance qui sert a initié et créer toutes les instances

Des bâtiments (le types, leurs couts, la production, la défense …)

De la mairie, des coordonnée des bâtiments et l'instances des

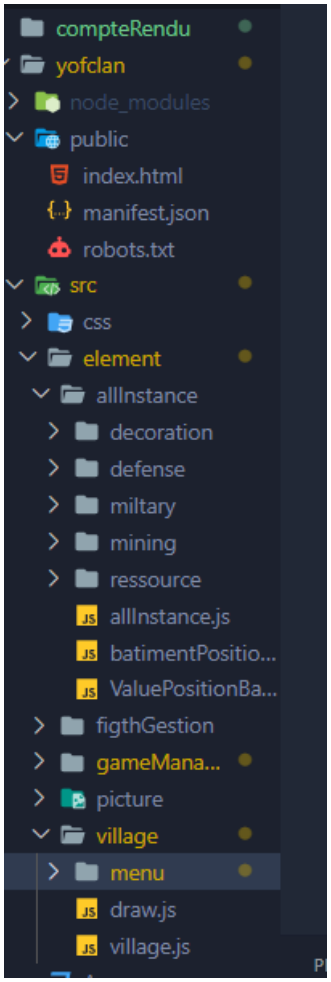
Différentes troupes.

Ensuite on retrouvera un dossier pour gérer le fight.

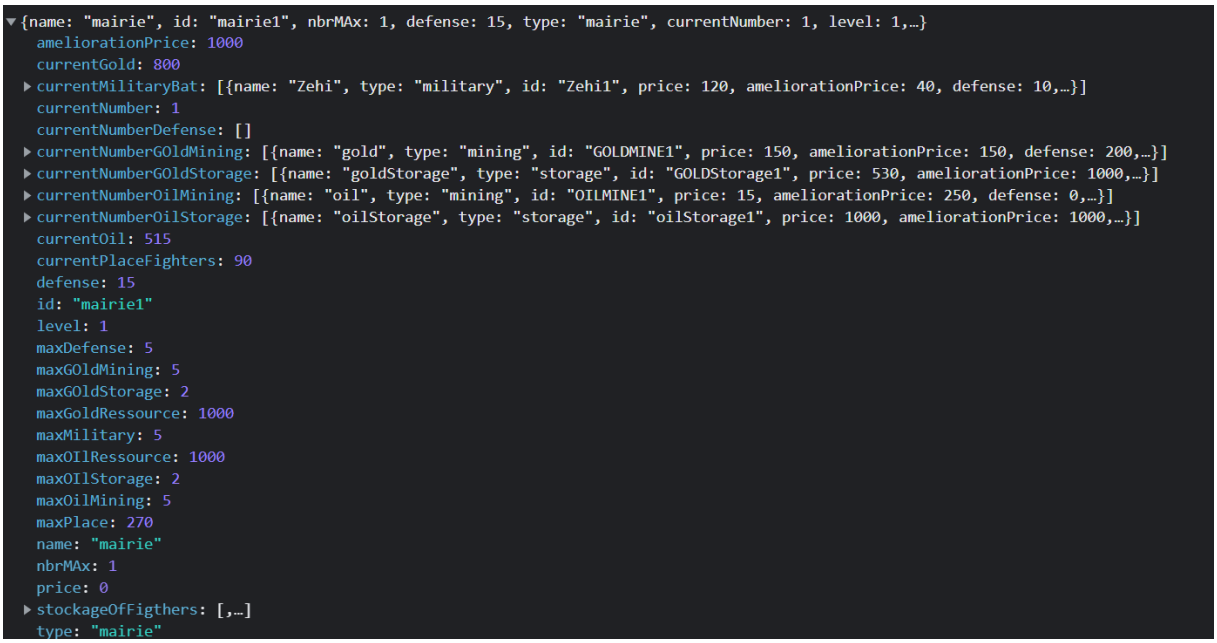
Un dossier gameManager afin de gérer tout le jeu.

Un dossier menu qui contient tous les éléments du village.

Et les fichiers pour dessiner et gérer le village.



L’implémentation de la sauvegarde via le LocalStorage a été faites. J’ai choisi le localStroage pour ça simplicité d’accès et d’utilisation.



On retrouve donc un localStorage comme ci-dessus pour la mairie.
Il sert également pour gérer le temps entre deux connexions.

Ainsi on stockera la date depuis le 1er Janvier 1970 00:00:00 UTC. Quand on appuie sur le bouton add ressource on comparera ensuite la date stockée dans le localeStorage et la date du jour.

Travail 3ème semaine :

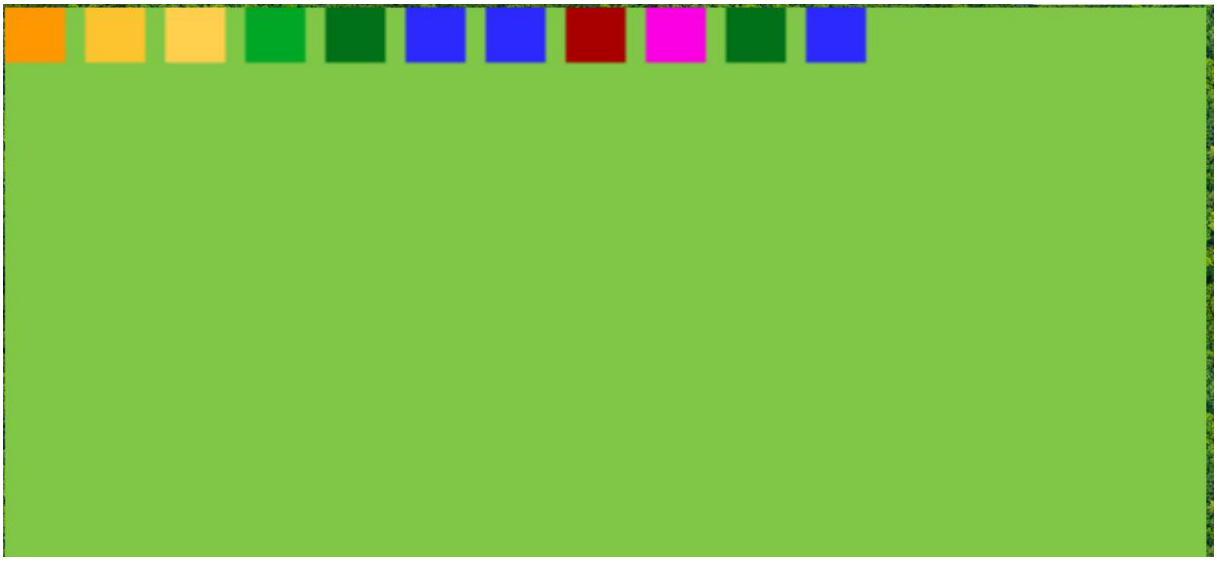
La troisième semaine et jusqu'à la fin du projet n'avais pas d'objectif propre et consistait à développer les différentes fonctions.

Ainsi a été effectué le dessin de bâtiments.
Initialement les bâtiments devait être des images qui les représentaient. Suite à des difficulté technique les bâtiments correspondent a des carrés de différentes couleurs.

Ils sont également stockés dans le localStorage.


```
{array: [{xAxes: 0, yAxes: 0, size1: 15, size2: 15, color: "#FF9700", name: "mairie", idColor: 1},…]}
 array: [{xAxes: 0, yAxes: 0, size1: 15, size2: 15, color: "#FF9700", name: "mairie", idColor: 1},…]
  0: {xAxes: 0, yAxes: 0, size1: 15, size2: 15, color: "#FF9700", name: "mairie", idColor: 1}
     color: "#FF9700"
     idColor: 1
     name: "mairie"
     size1: 15
     size2: 15
     xAxes: 0
     yAxes: 0
  1: {xAxes: 20, yAxes: 0, size1: 15, size2: 15, color: "#FCC42E", name: "goldStorage", idColor: 1}
  2: {xAxes: 40, yAxes: 0, size1: 15, size2: 15, color: "#FFCF4D", name: "oilStorage", idColor: 1}
  3: {xAxes: 60, yAxes: 0, size1: 15, size2: 15, color: "#00A726", name: "goldMine", idColor: 1}
  4: {xAxes: 80, yAxes: 0, size1: 15, size2: 15, color: "#007019", name: "oilMine", idColor: 1}
  5: {xAxes: 100, yAxes: 0, size1: 15, size2: 15, color: "#2C29FF", name: "zehi", idColor: 1}
```


Ainsi on stocke les coordonnées, sa taille, sa couleur, le nom du bâtiment et un idColor qui correspond au nombre de bâtiment créer par catégorie de bâtiment.



Ainsi on retrouve donc un village comme ci-dessus.

Le localStorage est également important pour permettre de ne pas redessiner sur un carrées.

Ainsi la fonction drawLastVillage (qui sert a dessiner le village si il existe) et draw (fonction qui dessine les carrées) parcours les coordonnées des bâtiments stockées dans le localStorage.

```
let newValue = localStorage.getItem("batimentArray");
 let batiment = JSON.parse(newValue);
 let i = batiment.array.length - 1;
```

```
  // *Parcours of tiotu my buildings in order to avoid the drawing of two
buildings on top of each other
  ctx.fillStyle = batiment.array[i].color;
  ctx.fillRect(
    batiment.array[i].xAxes,
    batiment.array[i].yAxes,
    batiment.array[i].size1,
    batiment.array[i].size2
  );
```

Codes issus de draw ligne93-104.

La fonction drawLastVilage fonctionne sur le même principe.

Sauvegarde

Le localstorage permet également une limite de bâtiment dessinable sur l'écran.

axeY	0
firstDate	1683805073830
limite	15
batimentArray	{"array":[{"xAxes":0,"yAxes":0,"size1":15,"size2":15,"color":"#FF9700","name":"mairie","idCol...
myMairie	{"name":"mairie","id":"mairie1","nbrMAx":1,"defense":15,"type":"mairie","currentNumber":1,..

Les fonctions draw et drawLastVillage permettent également d'appeler la fonction checkColorAtClick (checkColorAtClikc.js) qui permet la possibilité d'amélioré un bâtiment.

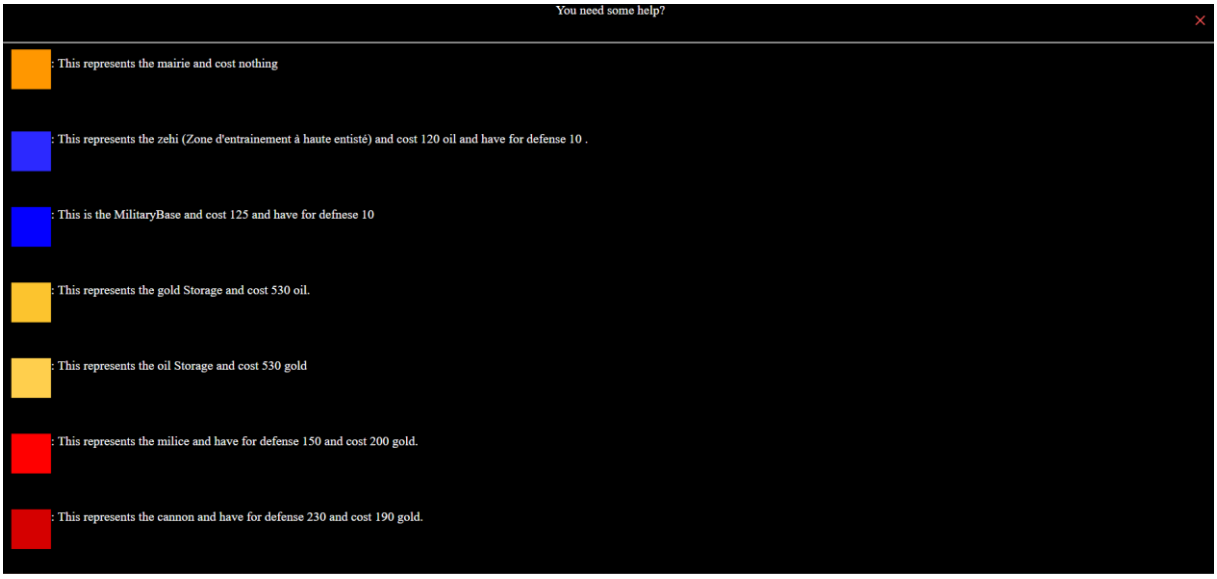
La fonction récupère les coordonnées du click au pixel près et compare la couleur en hexadécimale du click à celle qui est stocké dans le localStorage.

Après le click, elle affiche une div qui propose d'améliorer le bâtiment cliqué, ainsi que récupéré des ressources.

Help

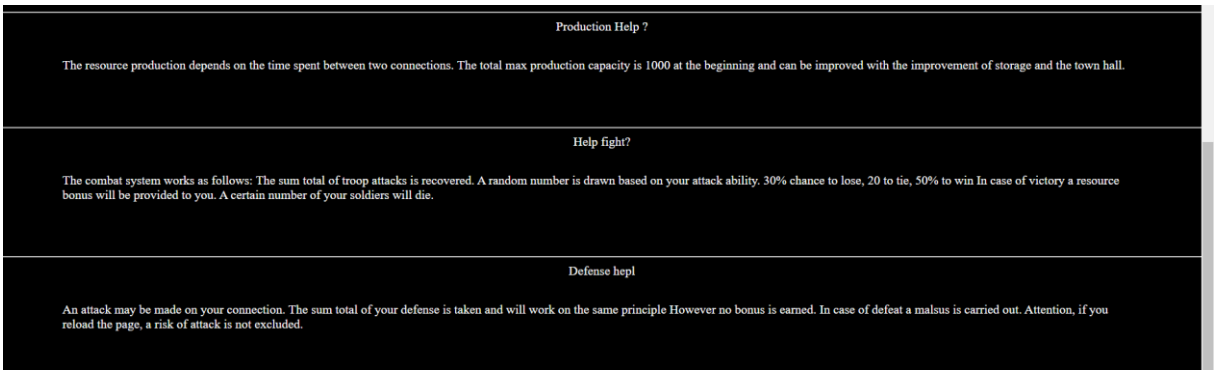
Compte tenu de la difficulté de reconnaitre un bâtiment, une fenêtre help a donc été crée et s'affiche quand on clique sur le bouton help.

Dedans on y retrouve différentes aides comme



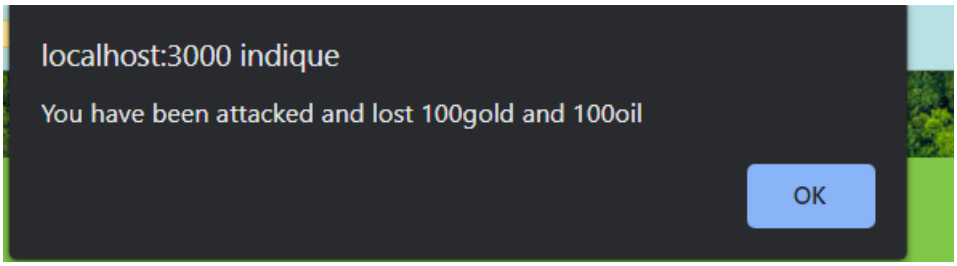
L'identification des bâtiments.

Et comment fonctionne les différentes fonctions.



Défense

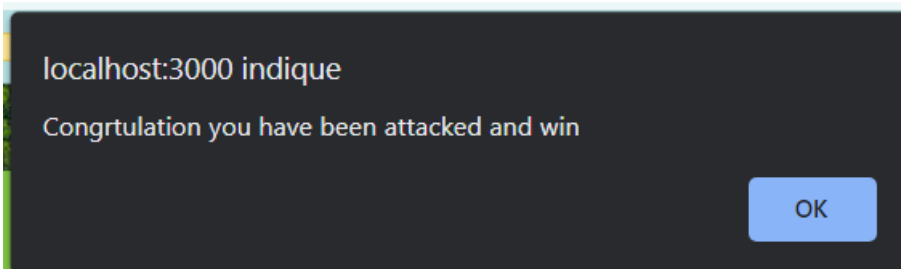
Il y’a également eu l’implémentation d’un système d’attaque par l’ordinateurs qui est aléatoire.



Elle s’effectue au chargement du village et possède 30% de chance de perdre.

La perte dépend du niveau de la mairie (100 * leNiveauDeLaMairie).

En cas de victoire une fenêtre l’averti de l’attaque.



Attaque

Le fight à également été crée et fonctionne sur le même principe que la défense.

On récupère la puissance d'attaque de nos troupes, leurs défenses et on fait aléatoirement l'attaque.

On prend un nombre au hasard entre la somme totale de la défense ou de l'attaque +1 et ensuite on regarde si le résultat est supérieur a une certaine valeur.

En fonction de celle-ci on perd ou on gagne.

Le bouton fight sert également à créer des troupes.

En cas de victoire, un gain de ressource est effectué de la même manière que la défense.

Conclusion :

En conclusion nous avons un jeu qui a été développé dans le temps imparti et qui correspond aux critères donnés et aux cahier des charges.

-Le jeu fonction sur un serveur en react js et utilise les technologies html / css / js

-Le cahier des charges est respecté dans sa globalité.

Nous avons une création de bâtiments fonctionnel et qui correspond aux critères du cahier des charges. On pourrait améliorer le village en intégrant des images à la place des carrés et également amélioré la responsivité du village.

Nous avons l'affichage du shop et qui est également fonctionnel. On pourrait penser en axe d'amélioration une amélioration de l'affichage des boutons qui est assez basique et qui ne correspond pas forcément au style du village.
On pourrait également améliorer le style générale des contenus.

Nous avons la créations et l'utilisation fonctionnelle des troupes et du fight. Là encore on pourrait améliorer l'affichage générale des boutons et également inclure la possibilité d'améliorés les troupes.

Nous avons un élément supplémentaire qui est de se faire attaquer.

Au niveau des objectifs annexe, la création de la musique a été réalisé mais pas implémenté à cause d'un mauvais rendu. La responsivité est également minim.