307 - Réaliser des pages Web interactives

Rapport personnel

Date de création : 15.05.2023  
Version 1 du 16.06.2023

Eldi Kabashi

|  |  |  |
| --- | --- | --- |
| S:\EMF\CPROF\Identite_EMF\2015-Identites\Logos des sections COULEURS\Logo_EMF-Informatique_FR_RVB_25.jpg |  | Module du 15.05.2023 au 13.06.2023 |

Table des matières

[1 Introduction 5](#_Toc137377771)

[2 Exercices 5](#_Toc137377772)

[2.1 Exercice 1 5](#_Toc137377773)

[2.2 Exercice 2 5](#_Toc137377774)

[2.3 Exercice 3 6](#_Toc137377775)

[2.4 Exercice 4 7](#_Toc137377776)

[2.5 Exercice 5 7](#_Toc137377777)

[2.6 Exercice 6 8](#_Toc137377778)

[2.7 Exercice 7 8](#_Toc137377779)

[2.8 Exercice 8 9](#_Toc137377780)

[2.9 Exercice 9 9](#_Toc137377781)

[2.10 Exercice 10 9](#_Toc137377782)

[2.11 Exercice 11 9](#_Toc137377783)

[2.12 Exercice 12 9](#_Toc137377784)

[2.13 Exercice 13 9](#_Toc137377785)

[2.14 Exercice 14 9](#_Toc137377786)

[2.15 Exercice 15 9](#_Toc137377787)

[2.16 Exercice 16 10](#_Toc137377788)

[2.17 Exercice 17 10](#_Toc137377789)

[2.17.1 Généralité API 10](#_Toc137377790)

[2.17.2 REST 10](#_Toc137377791)

[2.17.3 GET 10](#_Toc137377792)

[2.17.4 POST 10](#_Toc137377793)

[2.17.5 PUT 10](#_Toc137377794)

[2.17.6 DELETE 10](#_Toc137377795)

[2.17.7 SOAP 10](#_Toc137377796)

[2.18 Exercice 18 10](#_Toc137377797)

[2.19 Exercice 20 10](#_Toc137377798)

[3 Introduction du projet 10](#_Toc137377799)

[3.1 Définition 10](#_Toc137377800)

[3.2 Explications API 11](#_Toc137377801)

[3.2.1 Site de référence 11](#_Toc137377802)

[3.2.2 Fonctionnement 11](#_Toc137377803)

[3.2.3 Exemple d’utilisation 11](#_Toc137377804)

[4 Analyse 11](#_Toc137377805)

[4.1 Diagramme 11](#_Toc137377806)

[4.2 Maquette 11](#_Toc137377807)

[5 Conception 13](#_Toc137377808)

[5.1 Diagramme de navigation 13](#_Toc137377809)

[6 Implémentation 14](#_Toc137377810)

[6.1 HTML / CSS 14](#_Toc137377811)

[6.2 Javascript général 14](#_Toc137377812)

[6.3 Javascript spécifiques 14](#_Toc137377813)

[6.4 Descente de code complète pour une action qui fait appel à un service Web 14](#_Toc137377814)

[7 Conclusion 14](#_Toc137377815)

[7.1 Ce que j’ai aimé 14](#_Toc137377816)

[7.2 Ce que j’ai moins aimé 14](#_Toc137377817)

[7.3 Ce que j’ai appris 14](#_Toc137377818)

# Introduction

* Développer le caractère fonctionnel des pages Web interactives conformément aux données du problème.
* Développer une maquette pour la saisie et la présentation des données compte tenu des aspects ergonomiques.
* Choisir les éléments de formulaire appropriés pour la réalisation des données du problème, et garantir la validation des données entrées.
* Programmer l’application de manière modulaire et conformément aux directives de codification.
* Définir et en mettre œuvre des cas de tests appropriés pour des pages Web interactives et documenter dans le procès-verbal de tests.

# Exercices

## Exercice 1

### Objectif de l’exercice

Le but était de faire un bouton qui allait afficher dans un paragraphe le message « C’est Eldi Kabashi qui a pressé le bouton ! ».

### Explications

Comment ajouter un écouteur "click" via Javascript.

Il faut utiliser la commande « addEventListener », le premier paramètre est l’évènement, ici ce sera « click » et le deuxième paramètre est la fonction que va exécuter l’évènement, ici ce sera « testez ».

Comment ajouter directement un écouteur "click" sur un bouton.

Il faut rajouter un attribut sur notre balise « button » dans le fichier html. L’attribut se nomme « onclick », nous ajoutons ensuite la fonction qu’exécutera le bouton en click, il faut rajouter les parenthèses après le nom de la fonction.

Expliquez comment exécuter du code Javascript lorsque la page HTML a fini d'être chargée.

Pour exécuter du code js dans la page html, il faut utiliser l’attribut « onload » dans la balise « body », nous mettrons ensuite notre fonction js principale.

Expliquez ce que signifie DOM.

Document Object Model est le modèle des objets du fichier html, ce modèle se construit sous la forme d’arbre. Cela permet de retrouver et d’interagir avec les éléments html depuis notre fichier js et rendre dynamique l’affichage en html.

Où le script Javascript peuvent être chargés et pourquoi.

Dans la balise body car cela permet de charger tous les éléments de notre page avant de pouvoir exécuter notre code js qui fera référence à certains éléments de notre page. Il faut aussi préciser notre script js dans la balise « head » à l’aide de la balise « script ».

### Extrait de code

// Ajout de l'écouteur du bouton "Testez-moi" via programmation

function initCtrl() {

document.getElementById("testez").addEventListener("click", testez);

}

// Appel de cette méthode par la ligne ci-dessus, via un écouteur.

function testez() {

document.getElementById("info").innerHTML = "C'est Eldi qui a pressé le bouton !";

}

### Capture

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

## Exercice 2

### Objectif de l’exercice

Le but était de mettre en place une identification avec un nom d’utilisateur et un mot de passe.

### Explications

À quoi sert l’attribut « placeholder » ?

Il sert à avoir du texte dans un textfield qui disparait lorsqu’on commence à écrire dans le textfield.

À quoi sert l’attribut « autofocus » ?

Il permet d’avoir le focus lors du chargement de la page. Il n’est pas conseillé de le mettre sur plusieurs éléments.

Y a-t-il déjà du JavaScript dans ce code ?

Oui, nous avons l’attribut « onclick » suivi de la fonction « validerUtilisateur() » dans notre bouton.

Quelle est la différence entre le bouton de l'exercice 1 et celui-ci ?

Le bouton de l’exercice 1 était un élément « button », ici nous avons l’élément « input » qui se trouve dans un « fieldset » se trouvant dans un « form ».

Expliquer les fonctions alert(), confirm() et prompt().

La fonction alert() sert à faire apparaitre une popup avec un message. Confirm() permet de faire apparaitre une popup avec deux boutons : « OK » et « cancel ». Prompt() permet de faire apparaitre une popup où l’utilisateur peut écrire dans un textfield, il peut ensuite appuyer sur le bouton « OK ».

Quelle est la différence entre bouton "button" et un bouton "input" ?

Nous ne pouvons pas mettre de contenu dans un input.

Une explication pour récupérer un élément HTML depuis Javascript grâce à son id.

Pour récupérer un élément HTML, il faut utiliser la commande « document.getElementById() ».

Comment récupérer le texte contenu dans un champ de formulaire ?

Il suffit de rajouter « .value » à la fin de la commande précédente et mettre dans comme paramètre l’id du champ.

Comment écrire sur la console de votre navigateur ? Quelle est l'utilité ? Où se trouve la console de débogage de Chrome ?

Il faut utiliser la fonction « console.log() » et mettre notre message en paramètre. La console se trouve dans l’onglet « console » lorsqu’on inspecte la page.

Comment définit-on une « fonction » (méthode) comme « validerUtilisateur() » ?

function nomDeLaFonction(paramètres){}

Comment définit-on une « variable » (par exemple pour « username » et « password ») ?

let/const nomDeLaVariable = valeur.

Let est utilisé lorsqu’on peut changer la valeur de la variable, const lorsque la valeur est constante.

Les variables sont-elles « typées » (String par exemple) ?

Oui mais nous ne déclarons pas leur type.

Comment convertir un texte en minuscule ?

A l’aide de la fonction toLowerCase().

### Extrait de code

function validerUtilisateur() {

let username = "admin";

let password = "emf123";

if (username == document.getElementById("username").value) {

if (password == document.getElementById("password").value) {

window.alert("Mot de passe juste");

} else {

window.alert("Mot de passe incorrect");

}

} else {

window.alert("Login inexistant");

}

}

### Capture

Une image contenant texte, logiciel, Icône d’ordinateur, Page web

Description générée automatiquement

## Exercice 3

### Objectif de l’exercice

Le but est de faire un login à nouveau, mais cette fois-ci en reprenant les données sur un script en PHP.

### Explications

Dans un formulaire html, il existe plusieurs types de boutons :

* + « button » : bouton normal
  + « submit » : bouton qui envoie un formulaire à un serveur, le client est redirigé à la page indiqué dans l’attribut « action ».
  + « reset » : bouton qui réinitialise les valeurs d’un formulaire avec leur valeurs par défaut.
  + « image » : bouton représenté par une image.
  + « checkbox » : bouton qui peut être coché ou décoché.
  + « radio » : bouton qui peut être sélectionné dans un groupe de boutons.
  + « file » : bouton qui permet de sélectionner un fichier.

Un code PHP commence par « <?PHP » et fini par « ?> ».

Pour récupérer des données sur PHP avec les méthodes POST ou GET, il faut utiliser la commande $\_POST/GET[« »] et mettre dans les crochets le nom de notre textfield.

Pour concaténer des strings en PHP, il suffit de mettre un point entre les strings.

Pour convertir un string en minuscule, il faut utiliser la méthode « strtolower ».

Pour renvoyer quelque chose au client depuis le PHP, il faut utiliser la commande « echo ».

### Extrait de code

<?PHP

// test si on a reçu une donnée de formulaire nommée "username"

if (isset($\_POST['username'])) {

// récupération des données transmises dans des variables locales

$username = strtolower($\_POST['username']);

$password = $\_POST['password'];

// affichage des infos reçues

echo "username: ".$username."</br>";

echo "password: ".$password."</br>";

// test username et mot de passe

if (($username == "admin") && ($password == "emf123")) {

echo "<script>alert('Validation OK');</script>";

} else {

echo "<script>alert('Utilisateur ou mot de passe incorrect !!!');</script>";

}

}

?>

### Capture

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

## Exercice 4

JSFiddle est un outil permettant de tester du code js dans notre navigateur sans devoir associer des pages html ou css. Nous avons juste à mettre notre bout de code html, css et mettre le code que nous voulons tester en js. Cela permet de ne pas avoir à toujours associer des pages html à notre code js juste pour tester une petite partie.

Une image contenant capture d’écran, texte, logiciel, Logiciel multimédia

Description générée automatiquement

## Exercice 5

### Objectif de l’exercice

Le but était de réaliser une liste de commandes sur JSFiddle. Elles se basaient sur la notion de variable en Javascript.

### Explications

Comment déclarez-vous une variable en Javascript ?

On peut utiliser les mots-clés « let/const », let permet de faire des variables dont la valeur peut changer et const permet de garder toujours la même valeur.

Comment déclarez-vous une constante et quelle convention est utilisée ?

Pour faire une constante il faut donc utiliser le mot-clé « const ». Comme en Java, il faut mettre le nom de la constante en majuscules.

Quelle est la différence entre les mots clés var et let ?

« var » était utilisé avant 2015, il n’est plus utilisé car il laisse trop de liberté.

Quelle particularité y a-t-il à déclarer une variable sans les mots clés var et let ?

Cela permet de faire des variables globales.

Combien y a-t-il de types primitifs en Javascript et quels sont-ils ?

Il y en a 7 : string, number, bigint, boolean, undefined, symbol et null.

Quelle instruction permet de connaître le type d'une variable ? Donnez un exemple.

On peut utiliser le mot-clé « typeof » suivi de la valeur ou la variable. Exemple : console.log(typeof variable) ;

## Exercice 6

### Objectif de l’exercice

Le but était de reprendre l’exercice 1, mais cette fois-ci d’utiliser un switch en affichant le jour de la semaine de la date courante. Nous avions deux boutons, un qui permettait de faire cela avec un switch, et l’autre qui utilisait un tableau.

### Explications

Comment récupérez-vous le numéro du jour actuel ?

Pour cela, il faut créer une variable qui aura comme valeur « new Date() », il faudra ensuite utiliser la méthode « getDay() ».

Comment pouvez-vous insérer du texte dans une balise html ?

Il faut retrouver la balise, puis utiliser la commande « innerText » suivi du texte que l’on veut ajouter.

Comment pouvez-vous insérer du code html dans une balise html ? Par exemple "<b>On est mardi</b>"

Avec la fonction « innerHTML ».

Expliquez comment créer un tableau, vide ou avec des valeurs.

Pour créer un tableau, il suffit de créer une variable et de lui donner comme valeur « [] ». Nous pouvons mettre des valeurs dans les crochets.

Expliquez comment ajouter et supprimer des valeurs, au début ou à fin, d'un tableau.

Pour ajouter une valeur à la fin, il faut utiliser la commande « push() » avec la valeur entre les parenthèses, pour rajouter au début « unshift() ». Pour supprimer la dernière valeur « pop() » et pour la première « shift ».

Expliquez comment récupérer une valeur, dans un tableau, à position donnée.

Comme en Java, il faut mettre le nom du tableau et mettre entre crochet la position de la valeur que l’on veut.

Expliquez comment afficher l'ensemble des valeurs d'un tableau.

Pour cela, il faut faire une boucle qui affichera tous les éléments. Exemple :

noms.forEach(function(nom) {

console.log(nom);

});

### Extrait de code

Switch :

let date = new Date();

const s = "On est ";

switch (date.getDay()) {

case 0:

document.getElementById("info").innerHTML = (s + "Dimanche");

break;

case 1:

document.getElementById("info").innerHTML = (s + "Lundi");

break;

case 2:

document.getElementById("info").innerHTML = (s + "Mardi");

break;

case 3:

document.getElementById("info").innerHTML = (s + "Mercredi");

break;

case 4:

document.getElementById("info").innerHTML = (s + "Jeudi");

break;

case 5:

document.getElementById("info").innerHTML = (s + "Vendredi");

break;

case 6:

document.getElementById("info").innerHTML = (s + "Samedi");

break;

}

Tableau :

let date = new Date();

const jours = [

"Dimanche",

"Lundi",

"Mardi",

"Mercredi",

"Jeudi",

"Vendredi",

"Samedi",

];

document.getElementById("info").innerHTML = "On est "+jours[date.getDay()]+" tableau";

### Capture

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

## Exercice 7

### Objectif de l’exercice

Le but était de tester les différentes boucles en Javascript et d’afficher des décomptes.

### Explications

Les boucles for sont utilisées lorsqu’on connait le nombre d’itérations. Pour la faire, il faut utiliser la commande suivante :

for (let i=0; i<5; i++) {

console.log(i);

}

Ici le nombre d’itération est de 5.

Les boucles while servent lorsqu’on ne connait pas le nombre d’itérations, elles vont exécuter des instructions tant qu’une condition est vraie.

while(i<5){

console.log(i);

i++;

}

Les boucles do while agissent comme les boucles while, cependant, elles vont exécuter une fois les instructions avant de vérifier que la condition est vraie.

### Extrait de code

function testerFor() {

let s = "for (let i=0; i<5; i++) {...}<br/>"

for (let i=0; i<5; i++) {

s+="i = "+i+"<br/>";

}

s+="--> À utiliser si on sait que l'on veut itérer x fois (x connu avant de commencer la boucle)";

document.getElementById("info").innerHTML = s;

}

function testerWhile() {

let s = "while (i<5) {...}<br/>"

let i = 0;

while(i<5){

s+="i = "+i+"<br/>";

i++;

}

s+="--> À utiliser si on ne sait pas le nombre d'itérations au démarrage de la boucle";

document.getElementById("info").innerHTML = s;

}

function testerDoWhile() {

let s = "do {...} while (i<5)<br/>"

let i = 0;

do {

s+="i = "+i+"<br/>";

i++;

}while(i<5)

s+="--> À utiliser si on ne sait pas le nombre d'itérations au démarrage de la boucle mais avec un passage obligatoire";

document.getElementById("info").innerHTML = s;

}

### Capture

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

## Exercice 8

### Objectif de l’exercice

Le but était de parcourir un tableau d’objet en JSON et de les afficher.

### Explications

En javascript, nous pouvons utiliser une boucle « for » en utilisant la taille du tableau présent dans le json pour trouver le nombre d’itérations. Ensuite, pour récupérer la valeur des propriétés des objets présents dans le json, nous pouvons utiliser ce qui s’apparente à une boucle « for-each » en java. Il va s’écrire comme cela :

for (const f in objetDuJson) {

s+= objetDuJson[f] + " ";

}

La constante f permet de passer à travers chaque propriété de notre objet.

### Extrait de code

function parcourirUnTableauJSON() {

let s ="";

const json = {

personnes: [

{ prenom: "John", nom: "Doe", age: 44 },

{ prenom: "Anna", nom: "Smith", age: 32 },

{ prenom: "Peter", nom: "Jones", age: 29 },

],

};

for (let i = 0; i < json.personnes.length; i++) {

let personne = json.personnes[i];

s+=i+". "

for (const f in personne) {

s+=personne[f] + " ";

}

s+="<br/>";

}

document.getElementById("info").innerHTML=s

}

### Capture

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

## Exercice 9

### Objectif de l’exercice

Le but était de reprendre le projet et de tester les différentes manières de créer un objet en Javascript.

### Explications

En javascript, nous pouvons créer des objets à l’aide de classe exactement comme en java. Pour cela, il nous faudra créer la classe, elle comportera un constructeur, des attributs et des fonctions. Pour créer la classe, il nous suffit de créer une variable avec un nom et lui donner comme valeur « new NotreClasse() ». Exemple :

class Eleve {

constructor(prenom, nom, age) {

this.prenom = prenom;

this.nom = nom;

this.age = age;

}

toString() {

return this.prenom + " " + this.nom + " (" + this.age + ")";

}

}

### Capture

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

## Exercice 10

### Objectif de l’exercice

Le but était de créer un objet Javascript à l’aide plusieurs champs et de l’afficher en bas de la page. En cliquant sur un des objets, ses propriétés seront mises dans les champs.

### Explications

Pour cela nous avons d’abord créer un fichier js nommé « personne.js » qui sera notre bean. Nous avons aussi un worker.js et enfin notre indexCtrl. Dans le worker nous avons mis des fonctions permettant d’ajouter des personnes à l’aide des champs en reprenant leurs valeurs. L’indexCtrl va permettre d’afficher les informations.

### Extrait de code

Worker :

// fonction privée pour retrouver l'index d'une personne dans le tableau, -1 autrement

// il faut comparer avec toString()

function \_trouverPersonne(p) {

let idx = -1;

for (let i = 0; i < personnes.length; i++) {

if (p.toString() == personnes[i].toString()) {

idx = i;

}

}

return idx;

}

// ajouter une personne dans la liste des personnes si pas trouvée

function ajouterPersonne(p) {

let idx = \_trouverPersonne(p);

if (idx == -1) {

personnes.push(p);

}

}

// supprimer une personne dans la liste des personnes si trouvée

function supprimerPersonne(p) {

let idx = \_trouverPersonne(p);

if(idx!=-1){

personnes.splice(idx);

}

}

### Capture

Une image contenant texte, capture d’écran, logiciel, affichage

Description générée automatiquement

## Exercice 11

### Objectif de l’exercice

Le but était de reprendre l’exercice 10 mais cette fois-ci en utilisant des classes.

### Explications

En Javascript, la création de class est pratiquement identique à la création de class en Java.

class NomDeClass{

constructor(){

}

nomDeFonction(){}

}

Le constructor permet d’exécuter des instructions lors de la création d’un objet utilisant cette class. Nous pouvons aussi créer des fonctions que seuls les objets de cette class peuvent utiliser.

Pour instancier un objet directement pour qu’il soit accessible dans le HTML, il faut utiliser la commande « document.onreadystatechange=function(){} » et mettre dans cette fonction la création de notre objet donc « nomObjet = new Class() ; ».

document.onreadystatechange = function () {

if (document.readyState === "complete") {

window.ctrl = new Ctrl(); // ou ctrl = new Ctrl();

ctrl.afficherPersonnes();

}

};

### Extrait de code

class Worker{

// ajouter une personne dans la liste des personnes si pas trouvée

ajouterPersonne(p) {

let idx = this.trouverPersonne(p);

if (idx == -1) {

this.personnes.push(p);

}

}

// supprimer une personne dans la liste des personnes si trouvée

supprimerPersonne(p) {

let idx = this.trouverPersonne(p);

if(idx!=-1){

this.personnes.splice(idx);

}

}

}

### Capture

Une image contenant texte, capture d’écran, logiciel, Page web

Description générée automatiquement

## Exercice 12

### Objectif de l’exercice

Le but était de reprendre les différentes fonctions fournies pour tester les différentes manières de déclarer des fonctions

### Explications

Il existe trois façons de déclarer une fonction :

La fonction classique, nous allons ensuite appeler la fonction a() comme cela : « a() ; ».

function a() {

let val = 1;

console.log(val) ;

}

La fonction anonyme, nous allons appeler la fonction b() comme cela : « b() ; ».

let b = function() {

let val = 2;

console.log(val) ;

} ;

La arrow function, la fonction utilise les deux paramètres entre parenthèses. On peut l’appeler et l’afficher en faisant cela : « console.log(e(5,7)) ; ».

let e = (a,b) => { // avec plusieurs instructions

let somme = a+b;

return somme;

};

## Exercice 13

### Objectif de l’exercice

Le but était de mettre en place un cookie qui reprend les informations de l’utilisateur pour les garder et les remettre dans les champs.

### Explications

Un cookie est un petit fichier texte stocké sur notre pc par un site web. Il permet de contenir des informations utilisées par le site, il permet par exemple de contenir le login et mot de passe d’un utilisateur.

Pour créer un cookie et stocker une chaine de caractère, il faut utiliser la commande suivante :

document.cookie = "cookie\_exercice\_13=" + contenu + "; expires=" + dateExpiration.toUTCString() + "; path=/";

Pour récupérer le contenu d’un cookie, on peut par exemple faire cela :

// Tester la présence du cookie. Pour cela, il faut tester si la taille du cookie est supérieure à 0.

if (document.cookie.length > 0){

// Récupérer le contenu du cookie et en extraire la partie qui se situe après le "="; Vous pouvez utiliser la méthode "split"

// liée aux chaînes de caractères.

let cookieS = document.cookie.split("=")[1];

// Convertir la chaîne de caractères en objet Json.

let personne = JSON.parse(cookieS);

// Charger les deux champs avec la valeur du prénom et du du nom contenu dans l'objet json.

document.getElementById("prenom").value = personne.prenom;

document.getElementById("nom").value=personne.nom;

console.log("Data retrieved from cookie");

}

Pour convertir un objet JSON en chaine de caractère, on peut utiliser la fonction « JSON.stringify() » et mettre en paramètre notre objet.

let personneJsonS = JSON.stringify(personneJson);

// Appeler la méthode "setCookie" en passant en paramètre l'objet "personneJson" converti et en précisant le délai de 3 minutes.

setCookie(personneJsonS, 3);

Pour convertir une chaine de caractères en objet JSON, il faut utiliser la fonction « JSON.parse() » et mettre en paramètre notre chaine.

let personne = JSON.parse(cookieS);

// Charger les deux champs avec la valeur du prénom et du du nom contenu dans l'objet json.

document.getElementById("prenom").value = personne.prenom;

document.getElementById("nom").value=personne.nom;

console.log("Data retrieved from cookie");

### Extrait de code

function saveData() {

// Récupérer les valeurs contenues dans les champs "prenom" et "nom".

let prenomCookie = document.getElementById("prenom").value;

let nomCookie = document.getElementById("nom").value;

// Créer un objet json "personneJson" contenant le prénom et le "nom".

let personneJson={

prenom: prenomCookie,

nom: nomCookie

};

// Convertir l'objet json "personneJson" en chaine de caractère.

let personneJsonS = JSON.stringify(personneJson);

// Appeler la méthode "setCookie" en passant en paramètre l'objet "personneJson" converti et en précisant le délai de 3 minutes.

setCookie(personneJsonS, 3);

}

### Capture

Une image contenant texte, capture d’écran, Rectangle, conception

Description générée automatiquement

## Exercice 14

Jquery est une bibliothèque Javascript très populaire et utile. Elle permet d’interagir avec les éléments de nos pages HTML bien plus facilement. Avec jquery, nous pouvons sélectionner les éléments, manipuler le DOM et mettre en place des effets et des animations bien plus facilement.

Nous pouvons reconnaitre du code jquery grâce au symbole « $ », très présent, il est utilisé pour sélectionner des éléments.

Les deux tâches de jquery sont de simplifier la manipulation du DOM et de faciliter les interactions côté client.

Pour trouver un élément en jquery, il nous faut utiliser le symbole « $ » suivie de notre élément entre parenthèses, la sélection de l’élément marche comme en css.

La fonction jquery qui est appelé lorsque la page a été chargé est la fonction « $(document).ready() », les instructions de la fonction va s’exécuter lorsque notre page HTML se charge. Un exemple :

$(document).ready(function () {

let couleur = $("#couleurs").val();

changerCouleur(couleur);

});

Pour retrouver un élément parent, on utilise la fonction « .parent() », pour retrouver l’élément suivant, on utilise « .next() », pour le précédent « .prev() », pour les enfants d’un élément « .children() », pour retrouver les éléments précédents frères « .prevAll() » et les suivants « .nextAll() ».

Pour modifier le CSS d’un élément, on utilise la fonction « .css(« propriété », « valeur ») », pour ajouter un élément avant ou après l’élément sélectionné, on utilise les fonctions « .before(« élément ») » et « after(« élément ») », pour supprimer un élément on utilise « .remove() », pour gérer les événement on peut utiliser la fonction « .on(« click », function(){}) » et mettre les instructions que l’on veut exécuter lorsqu’on click sur l’élément. Pour cacher ou afficher un élément, on utilise les fonctions « .hide() » et « .show() ».

Pour lire un fichier avec la fonction « .ajax() », il faut préciser l’url du fichier et le retour si le fichier a pu être lu. Exemple :

$.ajax({

url: "file.xml",

success: function( xml ) {

$(xml).find("tab").each(function(){

$("ul").append(

"<li>" + $(this).text() + "</li>");

});

}

});

Pour lire des fichier JSON, nous pouvons utiliser la méthode « .getJSON ». Nous mettrons en paramètres le fichier et une fonction où on utilise l’objet JSO.

$.getJSON("file.json", function( obj ) {

for ( var prop in obj ) {

$("ul").append(

"<li>" + prop + ": " + obj[prop] + "</li>");

}

});

Pour charger un fichier HTML dans un div existant, nous pouvons utiliser la fonction « .load() » et mettre en paramètre notre fichier HTML.

$("div.load").load("file.html");

Pour intégrer jquery à nos pages, nous devons rajouter une balise « script » dans notre HTML et mettre ce lien :

script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

## Exercice 15

### Objectif de l’exercice

Le but était de mettre en place une liste de couleur qui changera la couleur de fond quand on en choisit une (en jquery).

### Explications

Pour récupérer la valeur d’un élément avec jquery, il faut utiliser la fonction « .val() ». Elle va nous retourner la valeur de notre élément sélectionné.

Pour modifier une propriété css avec jquery, il nous faut utiliser la fonction « .css() » et mettre en paramètres d’abord la propriété à modifier et ensuite la valeur.

Pour utiliser une animation avec jquery, il faut utiliser la fonction « .animate() » et mettre en paramètre une fonction qui définit les propriétés CSS à animer et leur valeurs, et en deuxième la durée de l’animation en millisecondes.

$("#monElement").animate({ left: "200px" }, 1000);

### Extrait de code

$(document).ready(function () {

let couleur = $("#couleurs").val();

changerCouleur(couleur);

});

function changerCouleur(couleur){

$("#container").css({"background-color":couleur});

if(couleur=="red"||couleur=="blue"||couleur=="green"){

$("#titre").css({color:"white"});

}else{

$("#titre").css({color:"black"});

}

}

### Capture

Une image contenant texte, capture d’écran, vert, logiciel

Description générée automatiquement

## Exercice 16

### Objectif de l’exercice

Le but était de tester plusieurs fonctions jquery pour différentes situations.

### Explications

Pour ajouter un écouteur via jquery, il faut utiliser la fonction « .on() ». Le premier paramètre sera l’évenement et le deuxième sera la fonction avec les instructions à exécuter.

$("#btnShow").on("click", function(){

ctrl.afficher();

});

Pour sélectionner des éléments du DOM via jquery, il suffit de mettre le symbole « $ » et mettre entre parenthèses notre élément.

Pour ajouter une classe CSS à un élément, il faut utiliser la fonction « .addClass() » en mettant en paramètre notre classe, pour la supprimer, il faut utiliser « .removeClass() ». Nous pouvons aussi utiliser la fonction « toggleClass() » qui ajoute la classe si il n’y est pas et la supprime si il y est.

Pour récupérer les valeurs d’un formulaire via jquery, il faut utiliser la fonction « .val() » qui va nous retourner la valeur d’un élément, on pourra stocker cette valeur dans une variable et l’utiliser.

Pour ajouter des éléments dans notre DOM en jquery, nous pouvons utiliser les fonctions « .append() », « .prepend() », « .after() », « .before() » et « .html() ». « append() » va ajouter l’élément à la fin de notre élément, « prepend() » va l’ajouter au début de l’élément, « after() » va l’ajouter après un élément, « before() » va l’ajouter avant un élément et « html() » permet de remplacer un élément par un élément que nous mettons en paramètre.

Pour extraire la valeur d’un attribut d’un élément HTML, il faut utiliser la fonction « .attr() » en mettant en paramètre l’attribut.

### Extrait de code

class Ctrl {

constructor(){}

afficher() {

$("#div\_1").show();

}

cacher() {

$("#div\_1").hide();

}

afficherCacher() {

$("#div\_1").toggle();

}

plier(){

$("#div\_2").slideUp();

}

deplier(){

$("#div\_2").slideDown();

}

fonduAfficher(){

$("#div\_3").fadeIn();

}

fonduCacher(){

$("#div\_3").fadeOut();

}

setClass(){

$("div>ol>li:first-of-type").addClass("boldBlueText");

}

removeClass(){

$("div>ol>li:first-of-type").removeClass("boldBlueText");

}

toggleClass(){

$("div>ol>li:last-of-type").toggleClass("boldBlueText");

}

pairesClass(){

$("div>ol>li:nth-child(odd)").toggleClass("underlineClass");

}

insererTexte(){

$("#textes").append($("#textToInsert").value());

if($("#textBold")==true){

}

}

}

### Capture

Une image contenant texte, capture d’écran, logiciel, Page web

Description générée automatiquement

## Exercice 17

### Généralité API

Une API permet de faire le lien entre le client et le serveur en lui permettant d’interagir avec les données de la base de données du serveur. Le client peut alors faire des requêtes pour récupérer, ajouter, modifier ou supprimer ces données.

### REST

Les Webservices REST fonctionne avec une architecture clien-serveur. Ils utilisent des méthodes HTTP : GET, POST, PUT, PATCH, DELETE. Ils représentent les données dans un format JSON, XML ou HTML. Les méthodes doivent être accessible par les clients, elles doivent donc être publiques.

### GET

Les requêtes GET permettent de récupérer des données. Lors d’une requête GET, l’API retourne les données dans un format JSON ou XML. Dans les requêtes, nous retrouvons le lien https suivi de paramètres permettant de retrouver certaines données. Après la requête, l’API nous retourne un code HTTP pour nous indiquer des informations sur la requête, si la requête a abouti, on reçoit « 200 ».

### POST

Les requêtes POST permettent d’ajouter des objets dans une API. Ces requêtes ne possèdent pas de paramètres. Dans la requête, nous allons préciser la destination de la requête, le format et les objets à ajouter. Nous pouvons aussi définir un token pour avoir les droits d’ajouter des données sur l’API. Le champ « content-type » nous permet de définir le type du contenu.

### PUT

Les requêtes PUT permettent de mettre à jour ou de remplacer des données sur des entrées de notre API. Pour cela nous allons utiliser l’URI pour retrouver l’entrée et nous allons indiquer quoi modifier, cela sera souvent en format JSON. Nous recevrons ensuite un code pour nous indiquer si la requête à marcher.

### DELETE

Les requêtes DELETE permettent de supprimer des entrées dans notre API. Pour cela, il suffit d’indiquer le lien de notre entrée et envoyer la requête. Nous pouvons utiliser des paramètres pour retrouver les entrées. Nous pouvons aussi ajouter des autorisations pour pouvoir avoir le droit de supprimer des entrées dans l’API.

### SOAP

Simple Object Access Protocol est un protocole de communication utilisant des fichiers XML pour pouvoir échanger des données. Les requêtes SOAP sont structurées comme une enveloppe qui possède un « header » et un « body ». La requête est envoyée sous forme d’enveloppe au serveur avec le protocole http, le serveur va extraire les informations de l’enveloppe et va les traiter, il va renvoyer une enveloppe SOAP au client et il reçoit les informations. Ce protocole utilise WSDL qui est un schéma XML pour pouvoir spécifier ce que le message doit contenir.

## Exercice 18

### Objectif de l’exercice

Le but était de mettre en place un convertisseur de degrés en fahrenheit à l’aide de requêtes faites sur un serveur PHP.

### Explications

Pour faire un appel GET ou POST avec Ajax, il faut qu’elle soit structurée de la sorte :

let url = "https://kabashiel.emf-informatique.ch/307/Exercices/Exercice\_18/php/convert-temp\_g\_json.PHP";

let param = "Temperature=" + degres + "&FromUnit=C&ToUnit=F";

$.ajax(url, {

type: "GET",

contentType: "application/x-www-form-urlencoded; charset=UTF-8",

data: param,

success: successCallback,

error: errorCallback

});

Nous allons d’abord indiquer le type de la requête (GET ou POST), puis nous allons le type de contenu et l’encodage utilisé, nous pouvons définir les paramètres avec la propriété « data », pour retourner une méthode en cas de succès, nous avons la propriété « success », la variable utilisée se trouve en paramètre de la fonction contenant la fonction ajax, comme cela nous peut utiliser les données reçues en cas de succès lors de l’appel de la fonction. En cas d’erreur, nous avons la propriété « error » qui marche de la même façon que le success mais qui s’exécutera en cas d’erreur.

Sur Postman, les requêtes vont ressembler à cela :

Une image contenant texte, capture d’écran, logiciel, nombre

Description générée automatiquement

L’onglet « Params » va nous permettre d’y entrer nos paramètres, les données vont ensuite s’afficher dans le « body », nous pouvons choisir la forme que prendra le body (« pretty », « raw », etc…).

### Extrait de code

celcius2Fahrenheit(degres, successCallback, errorCallback) {

let url = "https://kabashiel.emf-informatique.ch/307/Exercices/Exercice\_18/php/convert-temp\_g\_json.PHP";

let param = "Temperature=" + degres + "&FromUnit=C&ToUnit=F";

// envoi de la requête

$.ajax(url, {

type: "GET",

contentType: "application/x-www-form-urlencoded; charset=UTF-8",

data: param,

success: successCallback,

error: errorCallback

});

}

### Capture

Une image contenant texte, capture d’écran, Police, ligne

Description générée automatiquement

## Exercice 20

### Objectif de l’exercice

Le but était de créer une application web en SPA où l’on peut se loguer ou créer un compte.

### Explications

SPA (Single Page Application) est un modèle d’application web qui se base sur le fait d’avoir une seule page HTML qui va charger d’autre fichiers HTML dans un de ses éléments, les fichiers HTML chargés seront des « vues » ou « views » en anglais. Cela permet de ne pas avoir à naviguer entre plusieurs pages de site, cela permet de charger qu’une seule fois notre site car il n’y a qu’une seule page, cela permet donc aussi d’avoir un seul URL pour notre site.

Pour pouvoir charger du code HTML dans un élément d’une autre page, nous pouvons utiliser la fonction jquery « .load() » avec en paramètre notre fichier HTML.

loadLogin() {

this.vue.chargerVue("login", () => new LoginCtrl());

}

La fonction « $.ajaxSetup() » permet de configurer les options par défaut pour toutes les requêtes ajax que nous ferons dans notre projet. Ces options vont s’appliquer à toutes les requêtes appart si nous déclarons explicitement une option dans notre requête ajax.

### Extrait de code

VueService.js :

class VueService {

constructor() {}

chargerVue(vue, callback) {

// charger la vue demandee

$("#view").load("views/" + vue + ".html", function () {

// si une fonction de callback est spécifiée, on l'appelle ici

if (typeof callback !== "undefined") {

callback();

}

});

}

}

indexCtrl :

$().ready(function () {

// service et indexCtrl sont des variables globales qui doivent être accessible depuis partout => pas de mot clé devant ou window.xxx

http = new HttpService();

indexCtrl = new IndexCtrl(); // ctrl principal

http.centraliserErreurHttp(indexCtrl.afficherErreurHttp);

});

class IndexCtrl {

constructor() {

this.vue = new VueService();

this.loadLogin();

}

afficherErreurHttp(msg) {

alert(msg);

}

// avec arrow function

loadLogin() {

this.vue.chargerVue("login", () => new LoginCtrl());

}

// avec function classique

loadAccueil(langue) {

this.vue.chargerVue("accueil", function() {

new AccueilCtrl(langue);

});

}

loadCompte() {

this.vue.chargerVue("compte", () => new CompteCtrl());

}

}

Httpservice.js :

class HttpService {

constructor() {}

/\*

\*\* $.ajaxSetup permet de définir une fois un élément sans le refaire par la suite. Ici cela se fait l'error

\*/

centraliserErreurHttp(httpErrorCallbackFn) {

$.ajaxSetup({

error: function (xhr, exception) {

let msg;

if (xhr.status === 0) {

msg = "Pas d'accès à la ressource serveur demandée !";

} else if (xhr.status === 404) {

msg = "Page demandée non trouvée [404] !";

} else if (xhr.status === 500) {

msg = "Erreur interne sur le serveur [500] !";

} else if (exception === "parsererror") {

msg = "Erreur de parcours dans le JSON !";

} else if (exception === "timeout") {

msg = "Erreur de délai dépassé [Time out] !";

} else if (exception === "abort") {

msg = "Requête Ajax stoppée !";

} else {

msg = "Erreur inconnue : \n" + xhr.responseText;

}

httpErrorCallbackFn(msg);

},

});

}

/\*

\*/

login(identifiant, successCallback) {

// Uploade votre propre fichier PHP et adaptez l'URL ci-dessous.

let url = "https://kabashiel.emf-informatique.ch/307/Exercices/Exercice\_20/php/login20.php";

let param = "username=" + identifiant.username +

"&password="+identifiant.password + "&domaine=" + identifiant.domaine +

"&mail="+identifiant.mail+ "&langue="+ identifiant.langue;

// envoi de la requête

$.ajax(url, {

type: "POST",

contentType: "application/x-www-form-urlencoded; charset=UTF-8",

data: param,

success: successCallback

});

}

}

### Capture

Une image contenant texte, capture d’écran, affichage, logiciel

Description générée automatiquement

Une image contenant texte, capture d’écran, affichage, logiciel

Description générée automatiquement

# Introduction du projet

## Définition

Mon site web permet de répertorier les jeux gratuits et de les trier par catégorie, par plateforme et par développeur.

## Explications API

### Site de référence

Le site utilisé est « freetogame » qui fournit une API avec des jeux gratuits sur les plateformes « PC » ou « Web Browser ». Le lien de la documentation se trouve ici :

<https://www.freetogame.com/api-doc>

### Fonctionnement

L’URL utilisé pour retrouver les jeux et celui-ci « https://www.freetogame.com/api/games », il va nous retourner tous les jeux sous format JSON, ces jeux vont posséder plusieurs propriétés comme un titre, une image, un genre, la plateforme sur lesquels ils sont, etc…

Nous pouvons ajouter des paramètres à cette requête pour définir la plateforme, le genre ou bien pour trier les jeux par leur date de sortie.

Nous avons aussi cette URL : « https://www.freetogame.com/api/game?id= » qui nous retourne un seul jeu. Le JSON possède plus de propriétés. Nous devons définir l’id du jeu pour le retrouver.

### Exemple d’utilisation

<https://www.freetogame.com/api/games?category=shooter>

Une image contenant texte, capture d’écran, document, Police

Description générée automatiquement

<https://www.freetogame.com/api/game?id=452>

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement

# Analyse

## Diagramme

Une image contenant texte, diagramme, ligne, capture d’écran

Description générée automatiquement

## Maquette

Une image contenant diagramme, Rectangle, texte, Parallèle

Description générée automatiquement

Une image contenant texte, capture d’écran, Rectangle, diagramme

Description générée automatiquement

Une image contenant texte, capture d’écran

Description générée automatiquement

# Conception

## Diagramme de navigation

Une image contenant texte, diagramme, capture d’écran, ligne

Description générée automatiquement

# Implémentation

## HTML / CSS

Index.html

<!DOCTYPE html>

<html>

<!--

But : Index du site

Auteur : Eldi Kabashi

Date : 06.06.2023 / V1.0

-->

<!-- entête de la page -->

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<link

rel="stylesheet"

href="https://unpkg.com/leaflet@1.6.0/dist/leaflet.css"

integrity="sha512-xwE/Az9zrjBIphAcBb3F6JVqxf46+CDLwfLMHloNu6KEQCAWi6HcDUbeOfBIptF7tcCzusKFjFw2yuvEpDL9wQ=="

crossorigin />

<script

src="https://unpkg.com/leaflet@1.6.0/dist/leaflet.js"

integrity="sha512-gZwIG9x3wUXg2hdXF6+rVkLF/0Vi9U8D2Ntg4Ga5I5BZpVkVxlJWbSQtXPSiUTtC0TjtGOmxa1AJPuV0CPthew=="

crossorigin></script>

<script src="https://cdn.jsdelivr.net/npm/leaflet@1.7.1/dist/leaflet.js"></script>

<link rel="icon" type="image/png" href="images/emf-info.png" />

<link rel="stylesheet" href="css/main.css" />

<link rel="stylesheet" href="css/details.css" />

<link rel="stylesheet" href="css/jeux.css" />

<link rel="stylesheet" href="css/carte.css" />

<script src="https://code.jquery.com/jquery-3.7.0.min.js"></script>

<script type="text/javascript" src="js/indexCtrl.js"></script>

<script type="text/javascript" src="js/httpService.js"></script>

<script type="text/javascript" src="js/vueService.js"></script>

<script type="text/javascript" src="js/jeuxCtrl.js"></script>

<script type="text/javascript" src="js/detailsCtrl.js"></script>

<script type="text/javascript" src="js/carteCtrl.js"></script>

<title>JeuxGratuits - Kabashi - 307</title>

</head>

<!-- corps de la page -->

<body>

<div id="container">

<!-- calque pour les differentes vues -->

<div id="view"></div>

</div>

<!-- fin du conteneur -->

</body>

</html>

Jeux.html

<header>

<h1>

Jeux Gratuits

</h1>

<div>

<div>

<select name="Genre" id="cmboxGenre">

</select>

</div>

<div>

<select name="Plateforme" id="cmboxPlateforme">

</select>

</div>

<input type="button" value="Loupe" id="buttonFiltreJeu">

</div>

<div>

<input type="text" name="nomJeu" id="textfNomJeu" placeholder="nom">

<input type="button" value="Loupe" id="buttonNomJeu">

</div>

</div>

</header>

<div id="jeux">

</div>

<footer>

<div>Eldi Kabashi</div>

<button id="ouvrirCarte">Bureau des developpeurs</button>

<div>300223</div>

</footer>

Details.html

<header class="entete">

<h1 id="accueil">Jeux Gratuits</h1>

</header>

<div id="contenuDetails">

<table>

<tr>

<td>

<div id="infosJeu">

<p id="titreJeu"></p>

<p id="descriptionJeu"></p>

<p id="genreJeu"></p>

<p id="plateformeJeu"> </p>

<p id="developpeursJeu"></p>

<p id="datesortieJeu"></p>

<a id="lienJeu"></a>

</div>

</td>

<td>

<div id="imagesJeu">

<img id="imageJeu">

<div class="slideshow-container">

<!-- Full-width images with number and caption text -->

<div class="mySlides fade">

<div class="numbertext">1 / 3</div>

<img id="imageJeu1">

</div>

<div class="mySlides fade">

<div class="numbertext">2 / 3</div>

<img id="imageJeu2">

</div>

<div class="mySlides fade">

<div class="numbertext">3 / 3</div>

<img id="imageJeu3">

</div>

<!-- Next and previous buttons -->

<a class="prev" id="prev">&#10094;</a>

<a class="next" id="next">&#10095;</a>

</div>

<br>

<div style="text-align:center" class="circles">

<span class="dot" id="dot1"></span>

<span class="dot" id="dot2"></span>

<span class="dot" id="dot3"></span>

</div>

</div>

</td>

</tr>

</table>

</div>

<footer>

<div>Eldi Kabashi</div>

<button id="ouvrirCarte">Bureau des developpeurs</button>

<div>300223</div>

</footer>

Carte.html

<header>

<h1 id="accueil">Jeux Gratuits</h1>

</header>

<div id="mapid">

</div>

<footer>

<div>Eldi Kabashi</div>

<button id="ouvrirCarte">Bureau des developpeurs</button>

<div>300223</div>

</footer>

Main.css

/\*

But : Css principale de mon projet

Auteur : Eldi Kabashi

Date : 14.06.2023 / V1.0

\*/

header{

display: flex;

justify-content: space-between;

background-color: #002244;

align-items: center;

border-bottom: solid;

margin-bottom: 10px;

}

footer{

display: flex;

justify-content: space-between;

position: fixed;

left: 0;

bottom: 0;

width: 100%;

margin-top: 10px;

background-color: #002244;

border-top: solid;

}

footer > \*{

margin: 10px;

}

body{

background-color: #00416A;

}

div{

color: white;

font-family: Arial, Helvetica, sans-serif;

}

Jeux.css

header div{

display: flex;

height: fit-content;

align-items: center;

}

header div > \*{

margin-right: 10px;

margin-left: 10px;

}

.jeu{

border: solid 2px;

border-color: black;

margin: 3px;

border-radius: 10px;

background-color: #002244;

}

.jeu img{

height: 195px;

margin: 10px;

margin-bottom: 0px;

}

.jeu p{

padding-left: 10px;

padding-right: 10px;

}

#jeux{

display: flex;

flex-wrap: wrap;

margin-bottom: 50px;

}

.genreEtPlat{

display: flex;

justify-content: space-between;

}

Details.css

.entete{

text-align: center;

}

#lienJeu{

color: red;

}

.mySlides img {

height: 200px;

}

#titreJeu{

font-size: 30px;

font-weight: bold;

}

\* {box-sizing:border-box}

/\* Slideshow container \*/

.slideshow-container {

max-width: 1000px;

position: relative;

margin: auto;

}

/\* Hide the images by default \*/

.mySlides {

display: none;

}

/\* Next & previous buttons \*/

.prev, .next {

cursor: pointer;

position: absolute;

top: 50%;

width: auto;

margin-top: -22px;

padding: 16px;

color: white;

font-weight: bold;

font-size: 18px;

transition: 0.6s ease;

border-radius: 0 3px 3px 0;

user-select: none;

}

/\* Position the "next button" to the right \*/

.next {

right: 0;

border-radius: 3px 0 0 3px;

}

/\* On hover, add a black background color with a little bit see-through \*/

.prev:hover, .next:hover {

background-color: rgba(0,0,0,0.8);

}

/\* Caption text \*/

.text {

color: #f2f2f2;

font-size: 15px;

padding: 8px 12px;

position: absolute;

bottom: 8px;

width: 100%;

text-align: center;

}

/\* Number text (1/3 etc) \*/

.numbertext {

color: #f2f2f2;

font-size: 12px;

padding: 8px 12px;

position: absolute;

top: 0;

}

/\* The dots/bullets/indicators \*/

.dot {

cursor: pointer;

height: 10px;

width: 10px;

margin: 0 2px;

background-color: #bbb;

border-radius: 50%;

display: inline-block;

transition: background-color 0.6s ease;

}

.active, .dot:hover {

background-color: #717171;

}

/\* Fading animation \*/

.fade {

animation-name: fade;

animation-duration: 1s;

}

@keyframes fade {

from {opacity: .4}

to {opacity: 1}

}

Carte.css

#mapid {

height: 680px;

}

div.leaflet-popup-content{

color: black;

}

## Javascript général

## Javascript spécifiques

## Descente de code complète pour une action qui fait appel à un service Web

getJeuxFiltre(genre, plateforme, successCallBack){

if(genre!="all"){

$.ajax({

url: "https://www.freetogame.com/api/games?category=" + genre + "&platform="+plateforme,

type: "GET",

contentType: "application/x-www-form-urlencoded; charset=UTF-8",

success: successCallBack,

});

}else{

$.ajax({

url: "https://www.freetogame.com/api/games?platform="+plateforme,

type: "GET",

contentType: "application/x-www-form-urlencoded; charset=UTF-8",

success: successCallBack,

});

}

if(genre==null && plateforme==null){

this.getAllJeux;

}

}

# Conclusion

## Ce que j’ai aimé

J’ai beaucoup aimé le langage Javascript, je me suis beaucoup amusé dessus. J’ai adoré faire mon site, je trouvais cela très agréable d’écrire du code Javascript et de le voir fonctionner sur son site.

## Ce que j’ai moins aimé

Je trouve qu’il y avait beaucoup d’exercices, ce qui nous a pris du temps et donc nous n’avions pas énormément de temps pour notre site. La plupart des notions de javascript se basent sur des principes venant de java que nous connaissons déjà.

## Ce que j’ai appris

J’ai appris à coder en Javascript, à utiliser la bibliothèque jquery, à faire des requêtes sur une API en ligne.