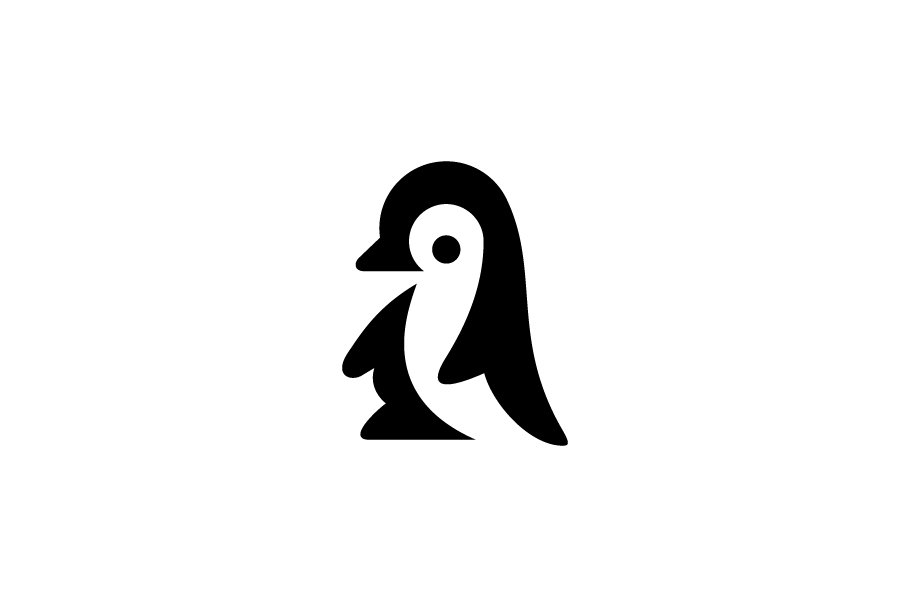
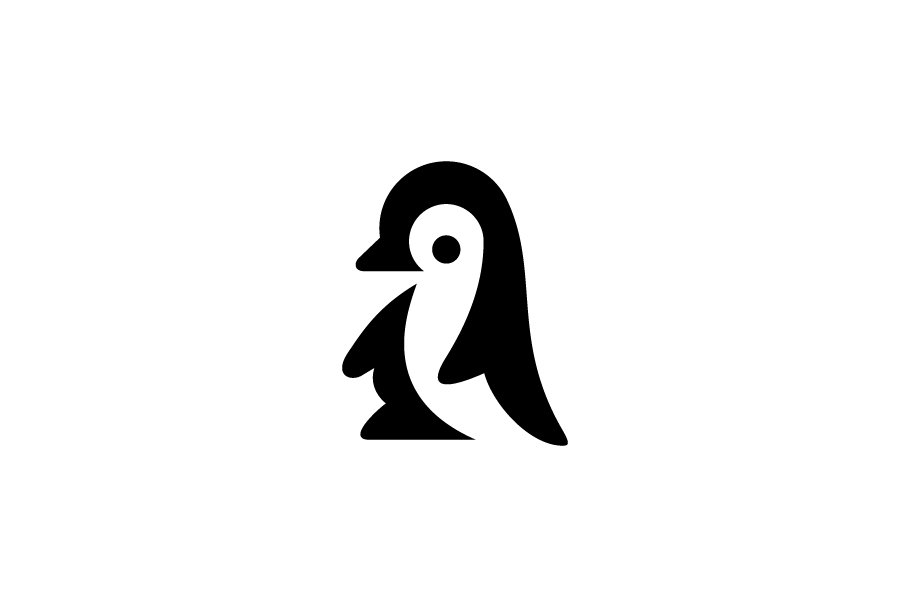
Rapport personnel

307

Réaliser des pages Web interactives

Version 2 du 05.10.2021

Kevin Morattel

Module du 17.05.2021 au 15.06.2021

Table des matières

[1 Introduction 1](#_Toc84322275)

[2 Projet 1](#_Toc84322276)

[2.1 Contexte 1](#_Toc84322277)

[2.2 API 1](#_Toc84322278)

[3 Tests technologiques 2](#_Toc84322279)

[3.1 Exercice 1 : HTML5 base 2](#_Toc84322280)

[3.1.1 But 2](#_Toc84322281)

[3.1.2 Code à effectuer/changer 2](#_Toc84322282)

[3.1.3 Ce que j'ai appris 2](#_Toc84322283)

[3.2 Exercice 2 : HTML5 from login 3](#_Toc84322284)

[3.2.1 But 3](#_Toc84322285)

[3.2.2 Code à effectuer/changer 3](#_Toc84322286)

[3.2.3 Ce que j'ai appris 3](#_Toc84322287)

[3.3 Exercice 3 : html5 from php 3](#_Toc84322288)

[3.3.1 But 3](#_Toc84322289)

[3.3.2 Code à effectuer/changer 3](#_Toc84322290)

[3.3.3 Ce que j'ai appris 4](#_Toc84322291)

[3.4 Exercices 4 à 7 : Javascript de base 4](#_Toc84322292)

[3.4.1 Buts 4](#_Toc84322293)

[3.4.2 Codes à effectuer/changer 4](#_Toc84322294)

[3.4.3 Ce que j'ai appris 5](#_Toc84322295)

[3.5 Exercices 8 et 9 : Javascript de traitement 5](#_Toc84322296)

[3.5.1 Buts 5](#_Toc84322297)

[3.5.2 Code à effectuer/changer 6](#_Toc84322298)

[3.5.3 Ce que j'ai appris 6](#_Toc84322299)

[3.6 Exercices 10 à 13 : Javascript en OO 6](#_Toc84322300)

[3.6.1 Buts 6](#_Toc84322301)

[3.6.2 Codes à effectuer/changer 6](#_Toc84322302)

[3.6.3 Ce que j'ai appris 8](#_Toc84322303)

[3.7 Exercices 14 à 16 : jQuerry (bootstrap) 8](#_Toc84322304)

[3.7.1 But 8](#_Toc84322305)

[3.7.2 Code à effectuer/changer 8](#_Toc84322306)

[3.7.3 Ce que j'ai appris 9](#_Toc84322307)

[3.8 Exercices 17 à 20 : web services 9](#_Toc84322308)

[3.8.1 But 9](#_Toc84322309)

[3.8.2 Code à effectuer/changer 9](#_Toc84322310)

[3.8.3 Ce que j'ai appris 10](#_Toc84322311)

[4 Analyse 10](#_Toc84322312)

[4.1 Use case 10](#_Toc84322313)

[4.2 Activity 11](#_Toc84322314)

[4.3 MVC 11](#_Toc84322315)

[5 Conception 11](#_Toc84322316)

[6 Implémentation 12](#_Toc84322317)

[6.1 Code 12](#_Toc84322318)

[7 Hébergement 16](#_Toc84322319)

[8 Fonctionnement 16](#_Toc84322320)

[9 Conclusion 17](#_Toc84322321)

# Introduction

Durant ce module, nous allons faire un projet. Ce projet consiste à créer un site internet utilisant du Javascript et une API. On apprendra aussi à utiliser du php.

# Projet

## Contexte

Mon projet utilise l'api kitsu, qui répertorie une majorité des œuvres d'animations japonaises sorties et à venir. Mon projet consiste à récupérer tout ces animes et de les lister et d'afficher leur notes. On peut aussi trier la liste et faire des recherches. En cliquant sur un des animes de la liste, on peut accéder à ses informations comme : son synopsis, ces tags, sur quels sites de streaming les trouver, un trailer, ….

## API

L'API que j'ai utilisé s'appelle [Kitsu](https://kitsu.docs.apiary.io/%23introduction). Cet API permet de récupérer les animes sous forme de liste. Cette liste peut être filtrée, triée, …. On peut aussi faire des recherches.

Les liens utiles pour faire ces actions sont :

* Pour naviguer dans la liste : [https://kitsu.io/api/edge/anime?page[limit]=5&page[offset]=0](https://kitsu.io/api/edge/anime?page%5blimit%5d=5&page%5boffset%5d=0)

ici, anime veut dire que l'on veut récupérer tout les animes. Par défaut, on aura accès à 10 pages (1 page = un anime et toutes ces informations) et aucun décalage. On peut incrémenter la limite pour avoir jusqu'à 20 pages en une fois. L'offset permet de décaler ces pages. Donc si on a une limite de 10 pages et que l'on décale de 10, on aura comme résultat les Pages 11 à 20.

* Pour tirer : <https://kitsu.io/api/edge/anime?sort=-slug>

Pour trier, il faut ajouter un sort. Après le "=" on met le nom d'un des attributs de l'anime. Si le après le "=" il y a un signe "-" c'est pour trier de manière décroissante ou Z-A.

* Pour filtrer : [https://kitsu.io/api/edge/anime?filter[categories]=action](https://kitsu.io/api/edge/anime?filter%5bcategories%5d=action)

Ici, on met le mot clé filter puis avec quoi l'on veut filtrer (on pourrait mettre d'autres que catégories comme voice actor ou studio d'animations). On peut donner plusieurs filtres en mettant une virgule entre les différentes choses (ici : action,adventure).

* Pour chercher : [https://kitsu.io/api/edge/anime?filtre[text]=nom](https://kitsu.io/api/edge/anime?filtre%5btext%5d=nom)

Pour rechercher par rapport au nom, on va juste utiliser entre crochets le mot "text".

# Tests technologiques

## Exercice 1 : HTML5 base

### But

Cet exercice à pour but de voir certaines bases en HTML et vite fait en JavaScript. On a dû voir comment marche le code donné puis essayé d'implémenter un écouteur directement dans une balise html.

Le but principal était d'introduire deux façons différentes d'appeler des écouteurs depuis le code html.

### Code à effectuer/changer

Il n'y a pas eu grand-chose à modifier ou à ajouter dans cet exercice. La seule chose à faire était d'ajouter un écouteur onclick="" dans la balise du bouton en html.

### Ce que j'ai appris

#### Fonctions

On peut voir, dans le fichier Javascript, comment est créée une fonction.

Voilà un exemple :

function test() {

  // code

}

#### Lier un fichier javascript au fichier html

Il y a deux façons de lier son fichier html à un code Javascript.

1ère (incorrecte mais fonctionnelle) :  
ajouter le code directement dans une balise <script>.

2ème (correcte) :  
ajouter dans la balise head, une balise script qui contient en paramètres les informations de votre fichier Javascript :  
<script type="text/javascript" src="[chemin du Javascript].js" async></script>

#### Écouteur Javascript

On peut ajouter un écouteur à un élément grâce à différentes fonctions Javascript.

Document : est une variable contenant la page web (c'est un point d'entrée qui permet d'interagir avec la page Web)

getElementById("[id]") : est une méthode qui permet de trouver un élément (de document par exemple) en lui donnant une id. (par exemple trouver un bouton avec un id bouton).

addEventListener("[quoi écouter]", [fonction]) : est une fonction qui permet

#### Écouteur html

Element.addEventListener(Déclencheur, méthode à effectuer)

Exemple :

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

## Exercice 2 : HTML5 from login

### But

Le but de cet exercice est d'utiliser des fonctions javascript pour se login.

### Code à effectuer/changer

Pour cet exercice, il a fallu aller chercher des valeurs venant de champs. Pour ce faire nous avons utiliser les deux fonctions suivantes :

let username = document.getElementById("username").value;

let password = document.getElementById("password").value;

on va ensuite utiliser un bout de code pour faire apparaître une popup si les valeurs de login sont correct :

alert("Validation ok")



### Ce que j'ai appris

Durant cet exercice, j'ai appris à utiliser les informations données par l'utilisateur et comment créer des champs à cet effet.

<input>

Permet de créer un objet qui demandera une valeur à l'utilisateur. On pourra ensuite la récupérer pour diverses choses.

<input type="text" size="30" id="username" placeholder="un nom svp" autofocus/>

On peut voir ici différentes choses. Déjà la type. Il permet de définir quel type de champ sera à cet endroit :

* Text : permet d'entrer du texte
* Password : permet d'écrire du texte qui sera caché par es points (pour les mots de passes)
* Button : permet d'afficher un bouton sur lequel on peut cliquer.

## Exercice 3 : html5 from php

### But

Le but de cet exercice est de comprendre à quoi sert et comment utiliser php.

### Code à effectuer/changer

Pour cet exercice, il faut ajouter une ligne de formulaire pour envoyer des informations au php :

<form class="user-form" action="login\_POST.php" method="POST">

La variable action permet de lier à un php pour envoyer ou recevoir des données. La variable method permet de donner l'informations de si la on GET (récupère) des informations, ou si l'on POST (publie) des informations.

### Ce que j'ai appris

Durant cet exercice, j'ai appris à utiliser les méthodes GET et POST de php.

## Exercices 4 à 7 : Javascript de base

### Buts

Le but de ces exercices est de comprendre comment créer des nouveaux projets et certaines commandes de bases tels que les conditions, boucles et switchs

### Codes à effectuer/changer

#### Bases du Javascript

Créer une variable nommée *« a »*

let a

Afficher le contenu de *« a »*avec console.log;

console.log(a);

Stocker la valeur 15 dans cette variable

a = 15;

Afficher le contenu de cette variable dans la console sous la forme *« Ma variable a = ? »*

console.log("Ma variable a = "+ a);

Créer une variable nommée *« b »* et lui assigner directement la valeur 9 ;

let b = 9

Afficher le contenu de cette variable dans la console sous la forme *« Ma variable b = ? »*

console.log("Ma variable b = "+ b);

Faire l’addition de ces 2 variables en affichant directement le résultat dans la console sous cette forme : « 15 + 9 = ? » ; (essayer d’utiliser un littéral avec `…${…}…`)

console.log("15 + 9 = "+b+a);

Compléter en faisant de même pour une soustraction, une multiplication et une division des deux variables ;

console.log("15 + 9 = "+b-a);

console.log("15 + 9 = "+b\*a);

console.log("15 + 9 = "+b/a);

Stocker *« Bonjour »* dans la variable a ;

a = "Bonjours"

Stocker *« les amis »* dans la variable b ;

b = "les amis"

Afficher *« bonjour les amis »* dans la console en concaténant les variables ;

console.log(a+b);

Stocker *« true »* dans la variable a ;

a = true

Stocker *« false »* dans la variable b ;

b = false

Effectuer une opération AND entre les 2 variables et afficher le résultat sous cette forme *« true AND false = ? » ;*

console.log("true AND false = "+a && b);

Effectuer une opération OR entre les 2 variables et afficher le résultat sous cette forme *« true OR false = ? » ;*

console.log("true AND false = "+a || b);

Stocker la date du jour dans la variable a avec new Date();

a = new Date(Date.now());

Calculer une nouvelle date dans la variable b qui est 61 jours avant la date courante (utilisation getDate, setDate)

let test = (24\*60\*60\*1000)\*61;

b = new Date();

b.setTime(a.getTime()- test);

Afficher les dates suivante en vous aidant de *« toLocaleString() et toLocaleDateString() »* :

console.log("Date du jour: "+a.toLocaleDateString())

console.log(" - 61 jours: "+b.toLocaleDateString())

#### Switchs

function afficherJourSemaine(){

permet d'avoir le jour de la semaine en nombre

var d = new Date().getDay();

var joursemain;

  vas vérifier le jour actuel et si c'est le cas change le joursemain avec le jour de la semaine en string

switch(d){

  case 0 :

  joursemain = "Dimanche"

  break

. . .

document.getElementById("info").innerHTML= "On est "+joursemain;

}

#### Boucles

function forfun() {

let tempo = "let index = 0; index < 5; index++";

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

tempo += "index = " + index + "</br>";

}

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

}

### Ce que j'ai appris

J'ai pu comprendre les variables javascript, apprendre les switchs et les boucles.

## Exercices 8 et 9 : Javascript de traitement

### Buts

Le but de cet exercice est de créer un objet JSON et l'utiliser, et de comprendre comment marche le js poo.

### Code à effectuer/changer

L'objet JSON :

const json = {

  personnes: [

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

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

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

  ]

}

Le code pour chercher les informations de cet objet :

function parcourirUnTableauJSON() {

let tempo = "";

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

let personne = json.personnes[index]

tempo += index + ". " + personne.nom + " " + personne.prenom + " " + personne.age + "</br>";

}

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

}

### Ce que j'ai appris

Durant ces exercices j'ai pu apprendre à utiliser des objets JSON.

## Exercices 10 à 13 : Javascript en OO

### Buts

Le but de ces exercices est de commencer à créer des objets Javascript, de modifier le code créer pour le faire correspondre à un schéma MVC puis finalement apprendre le principe des closures. On voit aussi différentes manières d'appeler des fonctions.

### Codes à effectuer/changer

#### Beans

function Personne(prenom, nom, age) {

this.prenom = prenom ;

this.nom = nom;

this.age = age;

}

Personne.prototype.toString = function() {

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

};

#### Constantes et différentes fonctions

const personnes = [

  new Personne("John", "Doe", 44),

  new Personne("Anna", "Smith", 32),

  new Personne("Peter", "Jones", 29),

];

personnes.sort();

// cherche si la personne est dans le tableau

function \_trouverPersonne(p) {

  let idx = -1;

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

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

      idx = i;

      break;

    }

  }

  return idx;

}

// ajoute la personne est dans le tableau

function ajouterPersonne(p) {

  let idx = \_trouverPersonne(p);

  if (idx == -1) {

    personnes.push(p);

  }

}

// supprime la personne est dans le tableau

function supprimerPersonne(p) {

  let idx = \_trouverPersonne(p);

  if (idx > -1) {

    personnes.splice(idx, 1);

  }

}

#### indexCtrl.js (Closures)

const tab =[

compteur(),compteur(),compteur(),

];

function actionCompteur(idx) {

let val = tab[idx-1].call(); // appelle explicite de la fonction du tableau

document.getElementById("button"+idx).innerHTML = "Compteur No "+idx+": "+val;

}

function compteur() {

let count = 0;

return function() {

return count++;

};

}

#### Appels de fonctions

// Déclaration d’une fonction

function a(nb) {

let val = 1 + nb;

console.log(val) ;

}

a(10);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Déclaration d’une expression fonction (fonction anonyme)

let b = function(nb) {

let val = 2 + nb;

console.log(val) ;

} ;

b(10);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Fonction IIFE

(function(nb) {

let val = 3 + nb;

console.log(val) ;

})(10) ;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Fonction IIFE (flèche)

((nb) => {

let val = 4 + nb;

console.log(val) ;

})(10) ;

### Ce que j'ai appris

J'ai appris à créer et manipuler des objets en Javascript et à adapter un code en modèle MVC. J'ai aussi appris le principe des js closures et comment appeler les fonctions de différentes manières

## Exercices 14 à 16 : jQuerry (bootstrap)

### But

Le but de cet exercice était de se documenter et d'utiliser ce qu'on a appris pour créer notre première page utilisant jQuerry.

### Code à effectuer/changer

Dans l'indexCtrlJQ :

$().ready(function () {

ctrl = new Ctrl();

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

console.log(couleur);

ctrl.changerCouleur(couleur);

});

class Ctrl {

constructor() {}

changerCouleur(couleur) {  
//sortie du container

$("#container").fadeOut(1500, function () {

//Changement de background

$("#container").css("background-color", couleur);

//si le background choisi est rouge, bleu ou vert le text est en blanc

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

$("#container").css("color", "white");

} else {

$("#container").css("color", "black");

}

//le retour du container

$("#container").slideToggle(1000);

//$("#container").fadeIn(1000);

});

}

}

Dans l'HTML :

<head>

<meta charset="UTF-8" />

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

<link rel="icon" href="#"/>

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

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

<!-- <script src="js/indexCtrlJS.js"></script> -->

<script src="js/indexCtrlJQ.js"></script>

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

<title>Ex15 — JQuery - premiers pas</title>

</head>

<!-- corps de la page -->

<body >

<div id ="container">

<form>

<p id="titre">

Choisir une couleur pour changer le fond du conteneur

</p>

<select size="9" id="couleurs"

onChange="ctrl.changerCouleur(couleurs.value);">

<option value="white">Blanc</option>

<option value="red">Rouge</option>

<option value="green">Vert</option>

<option value="blue">Bleu</option>

<option selected value="yellow">Jaune</option>

<option value="#F0F0F0">Gris</option>

<option value="#0FABCD">Bleu clair</option>

<option value="#FB33BB">Violet</option>

<option value="#CDB0DC">Violet Clair</option>

</select>

</form>

</div>

</body>

### Ce que j'ai appris

J'ai appris à utiliser jQuerry sur une page internet.

Voilà des sites utiles pour se documenter :

* [Jquery Original](https://johnresig.com/apps/workshop/intro/)
* [Jquery FR](https://www.jcsinfo.ch/demo/jquery/)
* [Complément Jquery](https://www.jcsinfo.ch/demo/jquery/ajax_jquery_complements-js.pdf)
* [Api Jquery](https://api.jquery.com/)
* [W3C Jquery](https://www.w3schools.com/jquery/default.asp)
* [Github Jquery](https://github.com/jeresig/jquery-workshop)

## Exercices 17 à 20 : web services

### But

Comprendre comment aller chercher des informations d'APIs en Json et de les mettre en page sur un site internet. On va aussi voir une démo d'une street map.

### Code à effectuer/changer

#### Requête à un web service

// envoi de la requête

$.ajax(url, {

type: "POST",

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

data: param,

success: successCallback

});

#### 2 exemples de fonctions

Celcius2Farenheit

celcius2Fahrenheit(degres, successCallback) {

let url = "http://mettrauxpa.emf-informatique.ch/convert\_temp\_p\_xml.php";

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

// envoi de la requête

$.ajax(url, {

type: "POST",

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

data: param,

success: successCallback

});

}

Farenheit2Celcius

fahrenheit2Celcius(degres, successCallback) {

let url = "http://mettrauxpa.emf-informatique.ch/convert\_temp\_g\_json.php";

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

// envoi de la requête

$.ajax(url, {

type: "GET",

data: param,

success: successCallback

});

}

### Ce que j'ai appris

J'ai appris à faire des requêtes à des web services

# Analyse

## Use case



## Activity



## MVC



# Conception

Voilà la maquette de mon site internet :



# Implémentation

L'implémentation de mon code est faite en modèle MVC.

* Les ihm sont les fichiers html qui appellent les méthodes du controller.
* Le controller est une classe javascript contenant tous les écouteurs et peu de choses faites directement dedans.
* Le worker contient toutes les méthodes devant communiquer avec l'API.

## Code

Voilà une suite de code qui amènera la liste des animes sur la page web.

Appel du controller depuis la vue.

<body onload="initCtrl()" onscroll="buttonToTop()">

//initialisation de la page

function initCtrl(){

    if (document.getElementById("chercher").hasAttribute("disabled")) {

        document.getElementById("chercher").removeAttribute("disabled");

    }

    scrollUp();

    buttonToTop();

    let page = document.getElementById("nbPage").innerText;

    if (page == 1) {

        document.getElementById("prev").style.visibility = "hidden";

    }else{

        document.getElementById("prev").style.visibility = "visible";

    }

    traiterAnimes(page, afficherAnimes)

}

Traitement des animes depuis le worker

//permet de traiter la liste des animes données par rapport à une recherche ou à un triage spécifique.

function traiterAnimes(page, callback){

    let liste= "";

    let lien ="";

    let nbAnimes;

    page--;

    //si la barre de recherche n'est pas vide, on va changer le lien.

    if (document.getElementById("chercher").value != "") {

        document.getElementById("sort").setAttribute("disabled", "disabled");

        lien = "https://kitsu.io/api/edge/anime?page[offset]="+page\*20+"&filter[text]="+document.getElementById("chercher").value+"&page[limit]=20";

    }else{

        if (document.getElementById("sort").hasAttribute("disabled")) {

            document.getElementById("sort").removeAttribute("disabled");

        }

        lien = "https://kitsu.io/api/edge/anime?sort="+document.getElementById("sort").value+"&page[offset]="+page\*20+"&page[limit]=20";

    }

    //récupération des animes et création de la liste

     $.getJSON(lien,undefined,function(animes){

        nbAnimes = animes.meta.count;

        console.log(nbAnimes);

        if ((page+1) == Math.ceil(nbAnimes/20)) {

            hideNext(true);

        }else{

            hideNext(false);

        }

        if (animes.meta.count > 20) {

            nbAnimes = 20;

        }else{

            nbanimes = animes.meta.count;

        }

        for (let index = 0; index < nbAnimes; index++) {

            //on vérifie si l'anime a un rank

            if(animes.data[index].attributes.ratingRank !== null){

                //on vérifie si l'anime a une image de poster

                if (animes.data[index].attributes.posterImage !== null) {

                    //on vérifie si l'anime a un titre en anglais

                    if(animes.data[index].attributes.titles.en\_us !== undefined){

                        liste=liste+"<ul class='anime'><img class='animage' src='"+animes.data[index].attributes.posterImage.tiny+"'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en\_us+"</a><a class='rank'> "+animes.data[index].attributes.averageRating+" </a></ul>"

                    }else if(animes.data[index].attributes.titles.en\_jp !== undefined){

                        liste=liste+"<ul class='anime'><img class='animage' src='"+animes.data[index].attributes.posterImage.tiny+"'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en\_jp+"</a><a class='rank'> "+animes.data[index].attributes.averageRating+" </a></ul>"

                    }else{

                        liste=liste+"<ul class='anime'><img class='animage' src='"+animes.data[index].attributes.posterImage.tiny+"'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en+"</a><a class='rank'> "+animes.data[index].attributes.averageRating+" </a></ul>"

                    }

                }else{

                   if(animes.data[index].attributes.titles.en\_us !== undefined){

                        liste=liste+"<ul class='anime'><img class='animage' src='images/pasDimage.png'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en\_us+"</a><a class='rank'> "+animes.data[index].attributes.averageRating+" </a></ul>"

                    }else if(animes.data[index].attributes.titles.en\_jp !== undefined){

                        liste=liste+"<ul class='anime'><img class='animage' src='images/pasDimage.png'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en\_jp+"</a><a class='rank'> "+animes.data[index].attributes.averageRating+" </a></ul>"

                    }else{

                        liste=liste+"<ul class='anime'><img class='animage' src='images/pasDimage.png'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en+"</a><a class='rank'> "+animes.data[index].attributes.averageRating+" </a></ul>"

                    }

                }

            }else{

                if (animes.data[index].attributes.posterImage !== null) {

                    if(animes.data[index].attributes.titles.en\_us !== undefined){

                        liste=liste+"<ul class='anime'><img class='animage' src='"+animes.data[index].attributes.posterImage.tiny+"'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en\_us+"</a><a class='rank'> N/A </a></ul>"

                    }else if(animes.data[index].attributes.titles.en\_jp !== undefined){

                        liste=liste+"<ul class='anime'><img class='animage' src='"+animes.data[index].attributes.posterImage.tiny+"'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en\_jp+"</a><a class='rank'> N/A </a></ul>"

                    }else{

                        liste=liste+"<ul class='anime'><img class='animage' src='"+animes.data[index].attributes.posterImage.tiny+"'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en+"</a><a class='rank'> N/A </a></ul>"

                    }

                } else {

                   if(animes.data[index].attributes.titles.en\_us !== undefined){

                        liste=liste+"<ul class='anime'><img class='animage' src='images/pasDimage.png'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en\_us+"</a><a class='rank'> N/A </a></ul>"

                    }else if(animes.data[index].attributes.titles.en\_jp !== undefined){

                        liste=liste+"<ul class='anime'><img class='animage' src='images/pasDimage.png'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en\_jp+"</a><a class='rank'> N/A </a></ul>"

                    }else{

                        liste=liste+"<ul class='anime'><img class='animage' src='images/pasDimage.png'><a class='titre' href='pages/anime.html?id="+animes.data[index].id+"'>"+animes.data[index].attributes.titles.en+"</a><a class='rank'> N/A </a></ul>"

                    }

                }

            }

        }

        //retour de la liste d'animes

        callback(liste);

    });

}

Retour de la liste des animes sous forme de string html

//affiche la liste des animes sur la page

function afficherAnimes(animes){

    document.getElementById("list").innerHTML = animes;

}

Résultat.

Une image contenant texte

Description générée automatiquement

# Hébergement

Mon site est hébergé sur <https://morattelk.emf-informatique.ch/animust>

# Fonctionnement

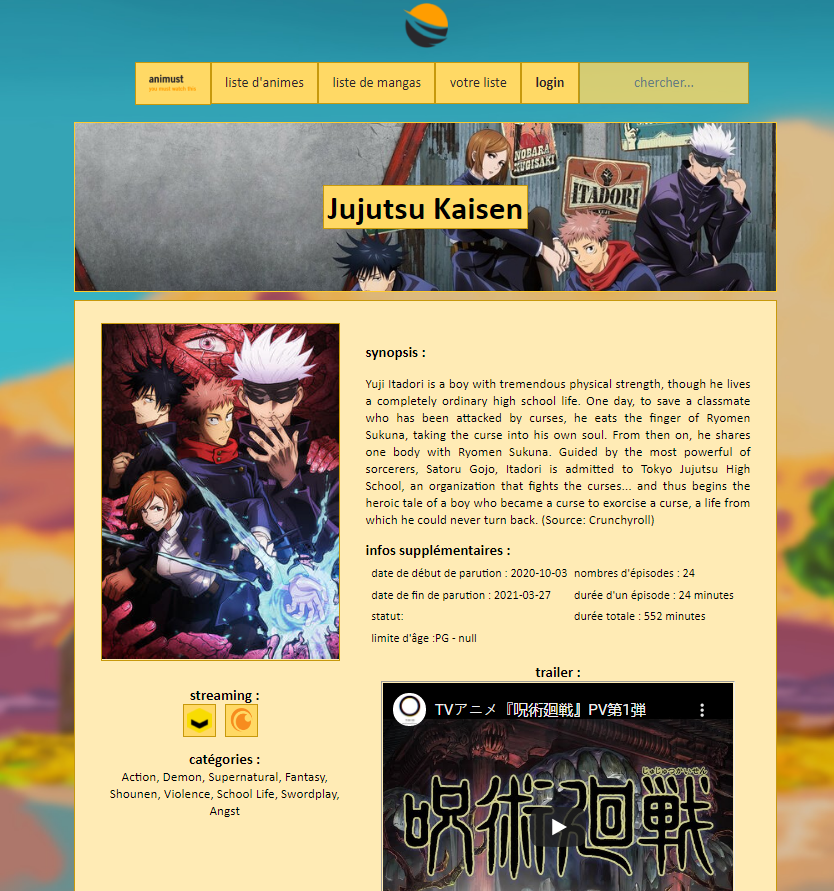
Tout n'est pas encore fonctionnel sur mon site. Donc voilà ce que peut faire mon site :

* Afficher la liste des animes du mieux noté au moins bien noté
* Changer le tri
* Rechercher un anime par rapport à son nom
* Accéder aux informations d'un anime et afficher celle-ci
* Jouer un trailer depuis le site
* Accéder aux sites de streaming

Voilà les deux types de pages que vous allez avoir en allant sur on site :

Une image contenant texte

Description générée automatiquement



Ce qui n'as pas encore été implémenté :

* Filtrer les animes en fonction de leurs catégories
* Faire des recherches pas uniquement sur le titre
* Une option de login
* Création d'une liste d'animes personnelle (disponible si connecté)
* Liste de mangas

# Conclusion

J'ai beaucoup aimé ce module et beaucoup aimé travailler sur mon projet. J'ai eu beaucoup d'idées dont certaines que je n'ai pas pu implémenter mais à cause de ça je n'ai pas assez pris le temps de documenter et créer les diagrammes nécessaires.