Découverte de JQuery

Pour ce projet, vous allez créer un fichier index.html dans le répetoire /SEQ4/S01/tp-decouverte/

Il va vous permettre de découvrir le framework JQuery et l’utiliser de manière basique

« Write less, do more »

Vous pouvez trouver le framework sur l’adresse suivante : <https://jquery.com/>

Préparation de votre projet

Créez un fichier unique « index.html », un fichier « index.js » et un fichier « index.css ». Ajoutez dans votre répertoire une image de taille conséquente.

**index.html**

<!DOCTYPE html>

<html lang="fr">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>Document</title>

</head>

<body>

    <div id="chargement">

    </div>

    <div id="contenu">

        <nav></nav>

        <main></main>

    </div>

</body>

</html>

Vous voilà paré !

1. Installer JQuery

Vous pouvez utiliser deux solutions :

* Télécharger le fichier js et utiliser la balise script pour l’intégréer
* Utiliser un CDN pour utiliser les serveurs d’autrui

Les deux méthodes ont leur avantage et inconvénient. Nous allons utilisé le CDN.

Ajoutez la ligne suivante dans l’entête du document HTML (<HEAD>)

 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>

Testons son fonctionnement ? Et si cela faisait un message dans le console.log pour vérifier ?

1. Tester le fonctionnement de JQuery

Ajoutez votre fichier JS et mettez dedans la commande suivante.

jQuery(document).ready(function(){

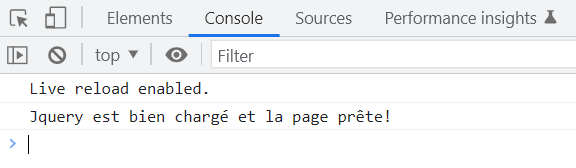
    /\* notre code ici\*/

    console.log('Jquery est bien chargé et la page prête!');

}

    );

Voici ce que vous devez obtenir :



**Principes de base**

Deux écritures équivalentes à la base de tout :

- jQuery()

- $()

La deuxième est bien un raccourci de la première.

**Chargement de la page**

Pour lancer du code au moment ou la page a fini de charger, pour faire l’équivalent d’un addEventListener(‘loaded’,….)

Deux écritures équivalentes :

jQuery(document).ready(function(){/\* notre code ici\*/ });

jQuery(function(){ /\* notre code ici\*/});

1. Ajouter une image et cacher jusqu’au chargement

Il arrive que notre page puisse être un peu lente à charger (même si ce n’est pas souhaitable !) et l’on veut parfois tout cacher tant que tout n’est pas chargé complétement : cela fait plus propre.

En premier, on va ralentir le chargement de la page : ajoutez une image lourde (comme les photos de classe ?)

 <img src="image-lourde.jpg" alt="une image lente à charger!">

Pour être précis, disposez-la dans le <main></main> !

Dans le fichier css, il va falloir définir une taille minimum pour le div chargement et dissimuler aux regards le div contenu. Nous allons également mettre des couleurs pour mieux voir.

#chargement {

    background-color: lightblue;

    min-height: 100vh;

}

#contenu {

    background-color: whitesmoke;

    display: none;

}

Maintenant, on ne voit que le fond de couleur bleu clair.

Plus qu’à changer ce qui se passe quand le document est chargé :

jQuery(document).ready(function(){

    /\* notre code ici\*/

    console.log('Jquery est bien chargé et la page prête!');

    $('#chargement').hide();

    $('#contenu').show();

}

    );

**Et si on bidouille le CSS** ?

On peut aussi faire ainsi en modifiant nous-même l’attribut CSS display

  $("#chargement").css("display", "none");

  $("#contenu").css("display", "block");

A ce stade, vous commencer à cerner l’intérêt !

1. Ajout de boutons de zoom et dezoom

On va maintenant ajouter deux boutons : un bouton « + » et un bouton « - »

<nav><button id="zoom">+</button><button id="dezoom">-</button></nav>

Une fois cela fait, nous allons ajouter des événements au click sur ces boutons : pour les viser, c’est comme pour le CSS et pour querySelector et querySelectorAll : #zoom et #dezoom !

Pour ajouter l’événement, il faut faire comme en javascript vanilla dans le principe : abonner au click !

jQuery('#zoom').click(function(){

        console.log('click sur zoom');

        $('img').each(function()

        {

            console.log('une image trouvee! a zoommer');

            $(this).css('height','100vh');

        })

    });

Attention qu’il faut bien ajouter le « defer », sinon cela ne trouvera pas le bouton !

**Sélection d’un ou plusieurs éléments** :

Lors de la sélection des éléments img, on peut avoir plusieurs images ! On peut boucler sur tous les éléments trouvé avec

$('img').each()

Dans cette boucle foreach, $(this) représente l’élément courant de l’itération en cours.

On l’appelait « élément » rappelez-vous.

Pour abonner au dezoom, on fait la même chose avec #dezoom et une taille de 50vh plutôt !

   jQuery('#dezoom').click(function(){

        console.log('click sur dezoom');

        $('img').each(function()

        {

            console.log('une image trouvee!a dezoommer');

            $(this).css('height','25vh');

        })

    });

1. Améliorer les interactions

Maintenant, nous allons ajouter une ligne de code pour désactiver le bouton sur lequel on vient de cliquer et activer l’autre bouton.

Pour ce faire, dans la fonction click :

* Sur #zoom, il faut désactiver le bouton #zoom et activer le bouton #dezoom
* Sur #dezoom, il faut désactiver le bouton #dezoom et activer le bouton #zoom

Pour désactiver ou activer ?

   $('#zoom').prop('disabled', true);

   $('#zoom').prop('disabled', false);

Plus qu’à placer correctement ces instructions : il faut bien quand on clique sur #zoom, désactiver #zoom et activer #dezoom ! et inversement !

1. Rendre le + et le – actifs

Côté ergonomie, ça serait pas mal que le clavier soit reconnu !

Pour tester ; affichons dans la console la touche qui est tapée !

Ajoutez le code suivant !

$(document).keyup(function(touche){

        // on écoute l'évènement keyup()

           var appui = touche.which || touche.keyCode;

       // le code est compatible tous navigateurs grâce à ces deux propriétés

      console.log(appui);

       })

Vous pouvez maintenant repérer le code correspondant aux touches « + » et « - ».  
Ajoutez maintenant le code pour faire l’effet de zoom et dezoom avec les mêmes fonctionnalités que précédemment (on veut que les boutons + et – s’activent et se désactivent !