Kyllian CELISSE Licence 3 informatique

Technologie du Web Année Universitaire : 2025/2026

TD 2- Javascript

Exercice de révision

Kyllian CELISSE Licence 3 informatique

Date de remise : 28/09/2025 Mr Tomczak

Table des matières

1. Introduction....3
2. Fonctionnalités implémentées....4
2.1 modification dynamique d’un lien....4
2.2 Vérification de la saisie utilisateur....4
2.3 Gestion des boutons radio....5
2.4 Contrôle du volume....5
2.5 Gestion du menu....6
2.6 Gestion de la date et récupération de l’année....6
2.7 Animation des barres de progression....6
3. Convertisseurs de devises....7
3.1 Première version....7
3.2 Deuxième version....8
4. Formulaires....9
4.1 Fichier HTML de Base....9
4.2 Fichier JavaScript....10
5. Difficultés rencontrées et solutions....11
6. Commentaires et suggestions....11
7. Auto-évaluation....12
8. Code et GitHub....12

1. Introduction

Le but de ce TP est de se familiariser avec la manipulation de JavaScript afin de comprendre comment interagir avec le style ou encore les attributs d'une page HTML.

Il a pour but principal de :

Modifier et ajouter des attributs dans la page HTML sans y toucher directement, gérer les actions faites par l'utilisateur, contrôler l'affichage dynamiquement ou encore automatiser certaines tâches.

Les technologies utilisées sont :

L'utilisation de HTML pour structurer la page, le CSS pour la mise en forme et principalement le JavaScript pour tout l'aspect interactivité et logique de la page.

Enfin, les outils utilisés sont :

L'éditeur de code Vscode qui permet d'écrire le code HTML/CSS/JS et l'utilisation d'un navigateur web pour pouvoir tester et vérifier le code.

2. Fonctionnalités implémentées

2.1 modification dynamique d'un lien

Le lien de base est celui de Wikipédia anglais qui a été modifié automatiquement pour renvoyer vers le lien Wikipédia français.

L'utilité de cette fonctionnalité implémentée est pour montrer la capacité de changer dynamiquement les attributs HTML sans en modifier son contenu

Lien dans le HTML source :

```
<a href="https://en.wikipedia.org/" target="_blank">La page principale de Wikipedia</a>
```

Changement via JavaScript grâce au **querySelector** :

```
// 1. Modifier le lien Wikipédia
const lienWiki = document.querySelector('a');
lienWiki.href = "https://fr.wikipedia.org";
```

Changement effectif sur le navigateur :

```
<a href="https://fr.wikipedia.org" target="_blank">
Wikipedia</a> == $0
```

2.2 Vérification de la saisie utilisateur

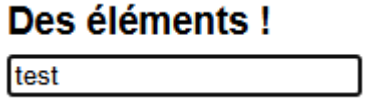
Lors de la saisie du texte si l'utilisateur n'a pas saisi « oui » ou « non » un message remplace notre saisie pour : « Il faut mettre Oui ou Non ».

Cette fonctionnalité illustre la validation de saisie côté client, afin d'éviter des entrées incorrectes.

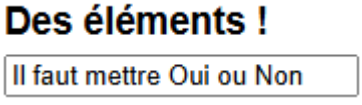
Utilisation d'un gestionnaire d'événement **click** sur le bouton et test conditionnel de la valeur avec **if**.

```
// 2. Vérifier si le texte est "Oui" ou "Non" au clic du bouton "Ok"
const bouton = document.querySelector('button');

const inputTexte = document.querySelector('input[type="text"]');

bouton.addEventListener('click', function() {
    const valeur = inputTexte.value.trim();

    if (valeur !== "Oui" && valeur !== "Non") {
        inputTexte.value = "Il faut mettre Oui ou Non";
    }
});
```

Avant d'appuyer sur le bouton « Ok » :



Après :



L'une des difficultés rencontrées a été de gérer les espaces involontaires, tout cela a pu être réglé grâce à l'utilisation de **trim().**

2.3 Gestion des boutons radio

Le texte des trois boutons radio sont remplacés dynamiquement par « HP », « Casque » et « Bluetooth » et change aussi le texte du volume en fonction du choix de l'utilisateur.

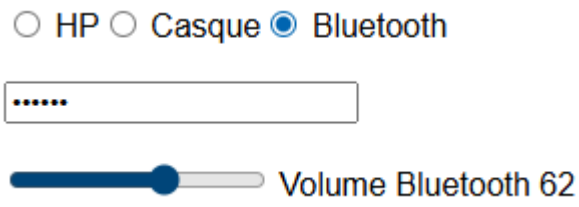
L'utilité de cette fonction est de l'impact direct qu'un choix utilisateur peut faire sur la page HTML

L'utilisation de getElementById et nextSibling m'ont permis la modification du texte et l'ajout d'un événement change m'a permis de mettre à jour l'intitulé du volume.

```
// 3. changment de textes dans les labels avec nextSiblings
const radio1 = document.getElementById('choix1');
const radio2 = document.getElementById('choix2');
const radio3 = document.getElementById('choix3');

radio1.nextSibling.textContent = ' HP';
radio2.nextSibling.textContent = ' Casque';
radio3.nextSibling.textContent = ' Bluetooth';
```

```
// 4. modifier le texte du volume selon notre choix avec l'est input radio
const spanVolume = document.querySelector('input[type="range"] + span');

radios.forEach(radio => {
    radio.addEventListener('change', function() {
        if (this.value === "1") {
            spanVolume.textContent = "Volume HP";
        } else if (this.value === "2") {
            spanVolume.textContent = "Volume Casque";
        } else if (this.value === "3") {
            spanVolume.textContent = "Volume Bluetooth";
        }
    });
});
```

Exemple sur le site :



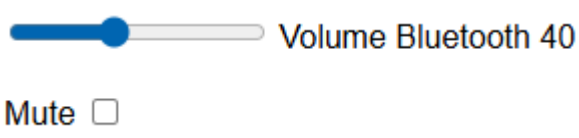
2.4 Contrôle du volume

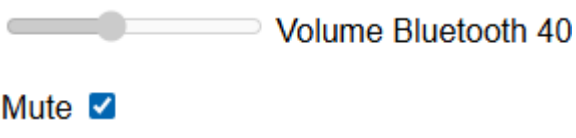
Le son est modifié au maximum de 100, la valeur du volume s'affiche en direct et la case à cocher permet de mettre en sourdine le son et désactiver le slider.

L'utilité est de montrer le changement dynamique sur le HTML via le JS et l'interaction utilisateur pour modifier ou « muter » le son

utilisation de l'événement **input** pour afficher sa valeur en temps réel et l'utilisation de **checked** et **disabled** pour contrôler les fonctionnalités de la case « mute »

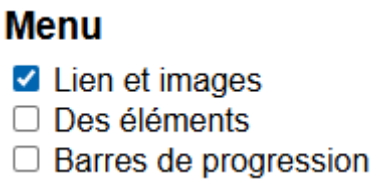
Voici comment la valeur du volume est affiché : voici le slider quand la case mute est coché





2.5 Gestion du menu

Le but est d'avoir des cases à cocher permettant d'afficher ou de masquer les différentes sections de la page. Celles-ci sont cachées au démarrage de la page.



L'utilité est de montrer le contrôle de la visibilité des sections selon le choix de l'utilisateur.

L'utilisation d'une boucle **forEach** sur les cases du menu pour ajouter des événements **change**, modification de **style.display** afin d'afficher ou cacher les sections voulu.

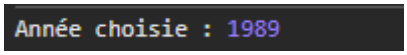
2.6 Gestion de la date et récupération de l'année

Lors de la sélection d'une date par l'utilisateur, l'année est affichée dans la console.

L'utilité est de montrer l'utilisation de l'objet date afin de récupérer la saisie de l'utilisateur

La récupération de la valeur a été faite avec **this.value** puis a été convertie en objet **Date**, puis extraite avec **getFullYear().**

Voici ce qui est affiché dans la console après que l'utilisateur ait choisi une date de l'année 1989 :



2.7 Animation des barres de progression

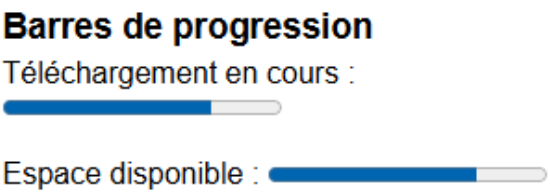
Lors du démarrage de la page, les deux barres de progression repartent à zéro puis avancent automatiquement de 5 % chaque seconde.

L'utilité de cette fonctionnalité est de montrer l'usage de setInterval

l'animation est faite grâce à une boucle forEach sur les barres

Initialisation de value à 0, puis incrémentation régulière tant que le value est plus petit que le max de la barre.

Voici comment cela rend sur la page :



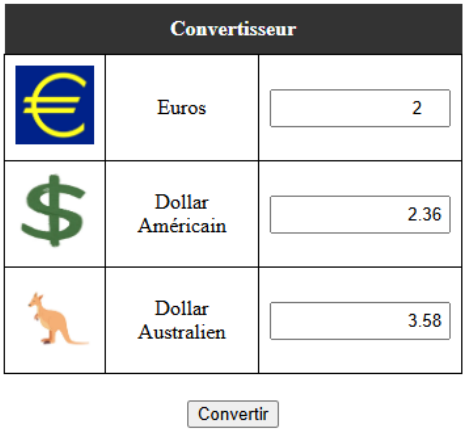
3. Convertisseurs de devises

3.1 Première version

Le but étant d'utiliser la fonction Convertir sur le bouton afin de faire la conversion en USD et AUD on récupère dans une variable euros l'élément que l'utilisateur inscrit dans la case afin de pouvoir procéder à la conversion.

```
<div class="center">
  <button onclick="convertir()">Convertir</button>
</div>

<script>
  function convertir() {
    let euros = parseFloat(document.getElementById("euros").value);

    // Taux actuel USD / AUD  le 23/09/2025 pour 1 EUR
    let tauxUSD = 1.18;
    let tauxAUD = 1.79;

    document.getElementById("usd").value = (euros * tauxUSD).toFixed(2);
    document.getElementById("aud").value = (euros * tauxAUD).toFixed(2);
  }

  // Conversion automatique au chargement
  window.onload = convertir;
</script>
```

Voici le rendu de la page :



3.2 Deuxième version

Le rendu de la page est similaire sauf que le bouton convertir n’est plus là puisque le résultat voulu change dynamiquement sans avoir besoin de cliquer sur un bouton. Tout cela sera récupéré encore une fois grâce à des id, sauf que cette fois-ci nous pourrons le faire que ce soit par l’**input** ayant l’id EUR, USD ou encore AUD.

Voici le code. Il y a des écouteurs d'événements afin de changer en temps réel les autres inputs quand l’utilisateur change les valeurs d’un des inputs

```
<script>
  // Taux actuel USD / AUD  le 23/09/2025 pour 1 EUR
  let tauxUSD = 1.18;
  let tauxAUD = 1.79;

  const eurosInput = document.getElementById("euros");
  const usdInput = document.getElementById("usd");
  const audInput = document.getElementById("aud");

  function fromEuros() {
    let eur = parseFloat(eurosInput.value) || 0;
    usdInput.value = (eur * tauxUSD).toFixed(2);
    audInput.value = (eur * tauxAUD).toFixed(2);
  }
```

```
  function fromUSD() {
    let usd = parseFloat(usdInput.value) || 0;
    let eur = usd / tauxUSD;
    eurosInput.value = eur.toFixed(2);
    audInput.value = (eur * tauxAUD).toFixed(2);
  }

  function fromAUD() {
    let aud = parseFloat(audInput.value) || 0;
    let eur = aud / tauxAUD;
    eurosInput.value = eur.toFixed(2);
    usdInput.value = (eur * tauxUSD).toFixed(2);
  }

  // Ajout d'écouteurs d’évènements
  eurosInput.addEventListener("input", fromEuros);
  usdInput.addEventListener("input", fromUSD);
  audInput.addEventListener("input", fromAUD);


  fromEuros();
</script>
```

4. Formulaires

4.1 Fichier HTML de Base

J'ai essayé de me rapprocher le plus possible de ce qui à été donnée dans le TP voici le rendu :


```css
body {
    font-family: Arial, sans-serif;
    width: 300px;
    margin: 20px auto;
    border: 1px solid rgb(15, 212, 212);
    padding: 10px;
}
h3 {
    margin: 5px 0;
}
img {
    display: block;
    margin: 10px auto;
    width: 200px;
}
.section {
    border-top: 1px solid black;
    margin-top: 10px;
    padding-top: 10px;
}
progress {
    width: 150px;
}
```

Voici la structure de la section Lien et Images

```html
<div id="liens">
    <h3>Lien et images</h3>
    <a href="https://en.wikipedia.org/" target="_blank">La page principale de Wikipedia</a>
    <br>
    <img src="https://www.google.fr/images/branding/googlelogo/2x/googlelogo_color_272x92dp.png" alt="Google">
</div>
```

Voici la structure de la section Des éléments :

```html
<div id="elements" class="section">
    <h3>Des éléments !</h3>
    <input type="text" value="Premiers pas avec Javascript">
    <br><br>

    <input type="radio" name="choix" id="choix1" value="1"> Choix N°1
    <input type="radio" name="choix" id="choix2" value="2" checked> Choix N°2
    <input type="radio" name="choix"id="choix3" value="3"> Choix N°3
    <br><br>

    <input type="password" value="123456">
    <br><br>

    <input type="range" min="0" max="100" value="50"> <span>Volume</span>
    <output> 0 </output>
    <br><br>

    <label for="mute">Une case à cocher</label>
    <input type="checkbox" id="mute">
    <br><br>

    Date : <input type="date">
    <br><br>

    <button>Ok</button>
</div>
```

Voici la structure de la section Barres de progression :

```html
<div id="progress" class="section">
    <h3>Barres de progression</h3>
    Téléchargement en cours :
    <progress value="30" max="100"></progress>
    <br><br>
    Espace disponible :
    <progress value="70" max="100"></progress>
</div>
```

4.2 Fichier JavaScript

Toutes les réponses ont été détaillées dans l'intitulé **2. fonction implémentée**. Je vais seulement me pencher sur l'ajout dynamique d'une image qui, elle, n'a pas été détaillée.

L'objectif était d'insérer automatiquement le logo UPHF à la fin de la section « lien et images »

L'utilité étant de montrer comment insérer de nouveau élément dans le DOM grâce au JavaScript

pour ce faire j'ai créé un élément **<img>** avec **createElement** puis l'insertion à la fin de la section avec **appenChild**

L'une des difficultés a été de bien identifier le conteneur où l'image doit être ajoutée. Pour ce faire, j'ai utilisé un id permettant de facilement cibler la section.

Voici comment rend le code :

```
// 7. Ajout d'une image à la fin de la section "Lien et Images"
const sectionLiens = document.getElementById("liens");

const nouvelleImage = document.createElement("img");
nouvelleImage.src = "https://upload.wikimedia.org/wikipedia/commons/b/bd/UPHF_logo.svg";
nouvelleImage.width = 200;

sectionLiens.appendChild(nouvelleImage);
```

Voici le rendu sur la section :

5. Difficultés rencontrées et solutions

Voici quelques difficultés que j'ai rencontrées :

-- J'ai pu rencontrer une difficulté sur quel chose mettre dans le querySelector afin d'être précis dans ce que je veux récupérer dans certaines constantes.

- La vérification de Oui ou Non qui a posé quelques soucis liés aux espaces, l'utilisation de **trim()** a permis de régler ce problème.

- La barre de progression a pu aussi me poser des problèmes sur le fait de la faire repartir de zéro à chaque actualisation de page et pour que la progression s'arrête au maximum donné dans la structure du HTML. Pour régler ce problème j'ai utilisé le **setInterval** avec un **if value < max.**

Les ressources mobilisées afin de régler les difficultés rencontrées sont les documentations données dans le TP, W3Schools, MDN Web Docs, StackOverflow et l'utilisation de prompt. L'utilisation de toutes ces documentations m'a permis d'avoir des réponses précises sur les problèmes que j'ai rencontrés.

6. Commentaires et suggestions

Le TP a été en globalité intéressant puisque il a permis de manipuler une grande variété de choses sur les bases du JavaScript car il permet d'initier sur comment gérer des événements simples et en temps réel, modifier l'affichage dynamiquement de la page ou encore la mise en place d'une mise à jour automatique de la page avec les barres de progression.

Ce qui pourrait être plus accentué est la documentation donnée durant tout le TP afin de rendre plus simple la recherche d'éléments.

Ce que j'ai appris/accentué grâce à ce projet est principalement le fait de procéder par étape durant un projet comme par exemple dans celui-ci où premièrement on commence par le HTML/CSS, puis on continue par la suite l'implémentation des fonctionnalités une par une avec des tests réguliers durant l'implémentation du JavaScript.

7. Auto-évaluation

J’ai été satisfait sur plusieurs éléments durant l’ensemble de ce TP. La structure du compte rendu a été assez bien respectée, j’ai essayé de ne pas trop me répéter durant la rédaction, les consignes ont été assez bien respectées permettant un bon suivi de la résolution des exercices. Cependant, je trouve que j’ai pris pas mal de temps pour rédiger ce compte rendu, j’ai donc préparé pour les comptes rendus futurs une template afin de gagner du temps dans la rédaction. Pour ce qui est du code et de la qualité du code, pour le HTML/CSS, je suis assez satisfait du code niveau structure, il manque peut-être un peu de commentaire dans le code. Pour ce qui est du Javascript il manque peut-être un peu de structure de code afin de mieux s’y retrouver et principalement beaucoup de commentaire à ajouter afin de clarifier ce que fait la fonctionnalité implémenté dans le code.

8. Code et GitHub

Pour en savoir plus, vous pouvez consulter mon code via ce lien GitHub qui permettra d’en savoir un peu plus sur le travail que j’ai effectué :

https://github.com/KyllianCel/Exercice_Initiation_TD