Interface Homme Machine sur le Web
26 Mars 2024

1 Introduction

Jusqu'à maintenant, nous nous sommes familiarisé avec le couple HTML-CSS dans le développement web. Néanmoins, ce couple est plutôt un trio. Aujourd'hui, le développement web ne peut pas se passer de l'utilisation de JavaScript.

Notre but ici n'est pas d'apprendre un nouveau langage de programmation parfaitement, mais d'étudier quelques exemples d'utilisation de JavaScript, notamment dans les interactions entre l'utilisateur et une page web.

JavaScript a été créé en dix jours par Brendan Eich en 1995. Malgré son nom, JavaScript n'a rien à voir avec le langage Java, même si Brendan Eich affirme s'être inspiré de nombreux langage, dont Java, pour mettre au point JavaScript. Après des débuts chaotiques, il est, comme déjà dit plus haut, devenu incontournable dans le développement web.

2 JavaScript et son intégration

JavaScript est un langage de programmation qui permet d'implémenter des mécanismes complexes sur une page web. À chaque fois qu'une page web fait plus que simplement afficher du contenu statique — afficher du contenu mis à jour à des temps déterminés, des cartes interactives, des animations 2D/3D, des menus vidéo défilants, ou autre, JavaScript a de bonnes chances d'être impliqué. C'est la troisième couche des technologies standards du web, les deux premières étant HTML et CSS.

En résumé, JavaScript est un langage de programmation permettant :

— de créer du contenu mis à jour de façon dynamique,
— de contrôler le contenu multimédia,
— d'animer des images,
— etc.

Pour être exécuté, JavaScript doit être intégré à la page HTML. Il existe plusieurs méthodes permettant de l'intégrer et nous allons nous intéresser à deux d'entre elle.

— Intégration par un fichier externe (fichier avec l'extension .js),
— Intégration par balise.

2.1 Intégration par un fichier externe

Cette première méthode consiste à écrire notre code dans un fichier avec l'extension .js (par exemple script.js). Il faut ensuite ajouter une ligne dans le code HTML afin de lier les deux ensembles. (ligne 11)

Fichier index.html

```
<!doctype html>
<html lang="fr">
    <head>
      <meta charset="utf-8">
      <title>Introduction a JavaScript</title>
      <link rel="stylesheet" href="style.css">
    </head>
    <body>
      ...
    </body>
  <script src="script.js"></script>
</html>
```

2.2 Intégration par balise

La seconde méthode utilisée consiste à écrire notre programme dans un fichier .js puis d'y faire un lien via l'attribut "src" de la balise "script". Nous allons par la suite directement écrire notre programme à l'intérieur de cette balise. (Ligne 11 à 13) Nous utiliserons la première méthode dans la suite du cours.

Fichier index.html

```
<!doctype html>
<html lang="fr">
    <head>
      <meta charset="utf-8">
      <title>Introduction a JavaScript</title>
      <link rel="stylesheet" href="style.css">
    </head>
    <body>
      ...
    </body>
  <script>
    ICI notre programme
  </script>
</html>
```

2.3 Fonctionnement

Le JavaScript est exécuté par le moteur JavaScript du navigateur, après que le HTML et le CSS ont été assemblés et combinés en une page web. Cet enchaînement est nécessaire pour être sûr que la structure et le style de la page sont déjà en place quand le JavaScript commence son exécution.

C'est une bonne chose, étant donné qu'un usage fréquent de JavaScript est de modifier dynamiquement le HTML et le CSS pour mettre à jour l'interface utilisateur, via l'API DOM comme nous l'évoquerons plus tard. Charger le JavaScript et essayer de l'exécuter avant que le HTML et le CSS ne soient en place mènerait à des erreurs.

Quand le navigateur rencontre un bloc de JavaScript, il l'exécute généralement dans l'ordre, de haut en bas. Vous devrez donc faire attention à l'ordre dans lequel vous écrivez les choses.

3 Parallèle entre JavaScript et Python

JavaScript (JS) et Python sont deux langages de programmation largement utilisés dans divers domaines de développement logiciel. Bien qu'ils aient des syntaxes distinctes et des uti-

lisations spécifiques, ils partagent de nombreuses similitudes dans leurs structures de contrôle. Ces structures, telles que les boucles, les variables, les fonctions, les conditions, etc. sont des éléments fondamentaux de la programmation.

Comparer la manière dont JavaScript et Python implémentent ces structures de contrôle peut aider à mieux comprendre leurs fonctionnalités et à faciliter la transition entre les deux langages.

Créer une variable	`a = 2`	`var a = 2;`
Fin d'instruction	retour à la ligne	point virugle `;`
indentation	définit la structure	optionnelle
fonction	`def f(x): ...`	`function f(x) { expr }`
condition	`if condition: expression`	`if (condition) { expr }`
boucle for	`for i in range(10): ...`	`for (var i=0; i<10; i++) { expr }`
boucle while	`while condition: expr`	`while (condition) { expr }`
commentaire	`# un commentaire`	`// un commentaire`
écrire dans la console	`print( expr1, expr2 )`	`console.log( expr1, expr2 );`

4 Exercice 1 : Application simple

1. Créer un fichier "exercice1.html" et un fichier "exercice1.js". Vous pouvez reprendre la structure de code vu plus haut pour le fichier HTML.
2. Intégrer le fichier JS au fichier HTML.
3. Déclarer une variable somme initialisé à 0.
4. Déclarer une variable nombreBoucle initialisé à 20.
5. Faîtes la somme de tous les entiers de 0 à nombreBoucle.
6. Afficher la somme.

Note : La fonction "prompt()" fonctionne comme la fonction "input()" en Python.

1. Modifier votre programme pour demander à l'utilisateur le nombre de boucle.

Pour tester votre code, ouvrez le fichier exercice1.html dans votre navigateur.

5 Événements et fonctions

Il est possible d'associer des événements aux différentes balises HTML, ces événements sont très souvent des actions réalisées pour l'utilisateur de la page :

— l'événement onclik correspond à un clic de souris (bouton gauche) sur un élément HTML donné,

— l'événement onmouseover correspond au survol d'un élément HTML par le curseur de la souris.

Il existe beaucoup d'autres événements, dans le cadre de ce cours nous utiliserons uniquement les deux cités ci-dessus.

5.1 Événement sur un bouton

Ici, nous avons une balise bouton sans aucun événement. Dans le cas où un clic est effectué dessus, rien ne vas se passer.

Fichier index.html

```
<!doctype html>
<html lang="fr">
    <head></head>
    <body>
        <h1 id="titre">Interaction Homme Machine sur le Web</h1>
        <button>Changer titre</button>
    </body>
    <script src="script.js"></script>
</html>
```

Pour se faire, nous allons ajouter un attribut "onclick" au bouton que nous pouvons ainsi que notre fonction.

Fichier index.html

```
<!doctype html>
<html lang="fr">
    <head></head>
    <body>
        <h1 id="titre">Interaction Homme Machine sur le Web</h1>
        <button onclick="fonctionClic()">Changer titre</button>
    </body>
    <script src="script.js"></script>
</html>
```

La fonction que nous allons placer en paramètre doit être défini dans le fichier JavaScript associé à notre page HTML. Avec cet ajout, lorsqu'un clic sera effectué sur le bouton, la fonction "fonctionClic" de notre programme JavaScript sera exécutée.

6 Définir une fonction JavaScript

Une définition de fonction (aussi appelée déclaration de fonction ou instruction de fonction) est construite avec le mot-clé function, suivi par :

— Le **nom** de la fonction.
— Une **liste d'arguments** à passer à la fonction, entre parenthèses et séparés par des virgules.
— Les **instructions** JavaScript définissant la fonction, entre accolades, { }.

Fichier script.js

```
function carre(nombre) {
  return nombre * nombre;
}
```

7 Exercice 2 : Application fonction

1. Créer un fichier ”exercice2.js”.
2. Réutiliser le fichier ”exercice1.html” et remplacer le fichier ”exercice1.js” par ”exercice2.js”
3. Déclarer une variable ”mot” initialisé à ”Numerique”.
4. Définir une fonction ”inverserMot” prenant en paramètre un mot et retournant son inverse. (euqiremuN)
5. Définir une fonction ”afficherMotInverse” prenant en paramètre le mot inversé et l'affichant dans la console.
6. Appeler la fonction ”inverserMot” avec ”mot” puis l'afficher grâce à la fonction ”afficherMotInverse”.

Note : Comme pour le premier exercice, ouvrez le fichier exercice1.html dans votre navigateur pour tester votre programme.

8 Interactions

Maintenant que nous avons vu comment définir une fonction en JavaScript, nous allons étudier ce qu'il est possible de faire de façon avancée. Pour celà, nous allons avoir besoin des API de navigateur. Les API de navigateur font partie intégrante de votre navigateur web, et peuvent accéder à des données de l'environnement informatique (l'ordinateur), ou faire d'autres choses complexes.

Dans notre cas nous voulons modifier un élément de notre page HTML grâce à notre fonction. Nous allons donc avoir recours à l'API DOM (Document Object Model) qui permet de manipuler du HTML et du CSS (créer, supprimer et modifier du HTML, appliquer de nouveaux styles à la page de façon dynamique, etc.). Chaque fois que vous voyez une fenêtre popup sur une page ou du nouveau contenu apparaître (comme dans notre démonstration plus haut), il s'agit d'une action du DOM.

8.1 Modifications HTML

Pour modifier un élément HTML il est tout d'abord nécessaire de comprendre le fonctionnement de l'API DOM. La première étape est de comprendre qu'avec un programme JavaScript nous allons travailler sur notre page html. Pour le spécifier à JavaScript nous allons utiliser ”document”. (Ligne 2 du fichier JS). Ensuite nous allons utiliser les fonctions de l'API. Pour sélectionner un élément de notre page nous allons utiliser les ”querySelector”.

Il existe de nombreuses fonctions :
— document.createElement(”p”) ; permettant de créer un nouveau paragraphe,
— document.body.appendChild(para) ; permettant d'ajouter le nouveau paragraphe à notre page,
— element.addEventListener(”click”, fonction) ; permettant d'ajouter un événement de clic sur ”element”.
— document.querySelectorAll(”element”) ; permettant de sélectionner tous les éléments de type ”element”,
— document.querySelector(”element”) ; permettant de sélectionner l'élément ”element”.

Fichier index.html

```
1   <!doctype html>
2   <html lang="fr">
3       <head></head>
```

```
    <body>
        <h1 id="titre">Interaction Homme Machine sur le Web</h1>
        <button onclick="maFonction()">Changer titre</button>
    </body>
    <script src="script.js"></script>
</html>
```

Fichier script.js

```
function maFonction() {
      let monTitre = document.querySelector("#titre");
}
```

Le mot clé let permet de définir une variable, dans l'exemple ci-dessous, la variable a pour nom "monTitre". Cette variable va nous permettre de manipuler l'élément HTML qui a pour id "titre" (c'est-à-dire la balise h1 présente dans le programme HTML).

Maintenant que nous avons "accès" à une balise HTML depuis le programme JavaScript, il est possible de modifier le contenu de cette balise.

Fichier script.js

```
function maFonction() {
    let monTitre = document.querySelector("#monPara");
    monTitre.innerHTML="Changement de titre"
}
```

Ici, nous avons changé le titre par "Changement de titre".

8.2 Modifications CSS

Il est aussi possible de modifier les classes CSS associées à une balise HTML, prenons un exemple :

Fichier index.html

```
<!doctype html>
<html lang="fr">
    <head></head>
  <link rel="stylesheet" href="style.css">
    <body>
        <h1 id="titre">Interaction Homme Machine sur le Web</h1>
        <button onclick="changeCouleur()">Changer couleur</button>
    </body>
    <script src="script.js"></script>
</html>
```

Fichier script.js

```
.vert {
      color: green;
}
```

Fichier style.css

```
function changeCouleur() {
      let monTitre = document.querySelector("#titre");
  monTitre.classList.add("vert");
}
```