Chapitre 2 : JavaScript
Interaction entre l'homme et la machine sur le Web

I. Introduction

A. Qu'est-ce que JavaScript ?

JavaScript a été créé en 1995 par Brendan Eich.

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives mais aussi pour les serveurs avec l'utilisation (par exemple) de Node.js.
On abrège souvent JavaScript par JS, ce qui sera parfois fait par la suite.
Avec les technologies HTML et CSS, JavaScript est parfois considéré comme l'une des technologies cœur du World Wide Web. Le langage JavaScript permet des pages web interactives, et à ce titre est une partie essentielle des applications web. Une grande majorité des sites web l'utilisent, et la majorité des navigateurs web disposent d'un moteur JavaScript dédié pour l'interpréter, indépendamment des considérations de sécurité qui peuvent se poser le cas échéant.
Du code JavaScript peut être intégré directement au sein des pages web, pour y être exécuté sur le poste client. C'est alors le navigateur web qui prend en charge l'exécution de ces programmes appelés scripts.

Dans une page Web (au programme de 1ère NSI)

Généralement, JavaScript sert à contrôler les données saisies dans des formulaires HTML, ou à interagir avec le document HTML via l'interface Document Object Model, fournie par le navigateur (on parle alors parfois de HTML dynamique ou DHTML). Il est aussi utilisé pour réaliser des applications dynamiques, des transitions, des animations ou manipuler des données réactives, à des fins ergonomiques ou cosmétiques.
JavaScript n'est pas limité à la manipulation de documents HTML et peut aussi servir à manipuler des documents SVG, XUL et autres dialectes XML.

Sur un serveur Web

JavaScript peut également être utilisé comme langage de programmation sur un serveur HTTP à l'image des langages comme PHP, Python, ASP, etc. D'ailleurs le projet CommonJS travaille dans le but de spécifier un écosystème pour JavaScript en dehors du navigateur (par exemple sur le serveur ou pour les applications de bureau natives).

B. Premiers pas avec JavaScript

Comme indiqué précédemment, le JavaScript est un langage essentiellement utilisé avec le HTML, vous allez donc apprendre dans ce chapitre comment intégrer ce langage à vos pages Web, découvrir sa syntaxe de base et afficher un message sur l'écran de l'utilisateur.

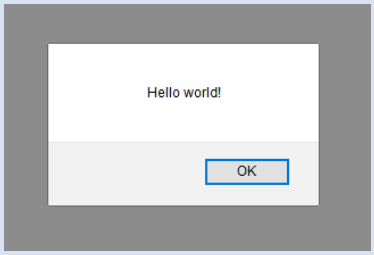
Ne dérogeons pas à la règle traditionnelle qui veut que tous les tutoriels de programmation commencent par afficher le texte « Hello World! » (« Bonjour le monde ! » en français) à l'utilisateur.

1ère méthode : écrire le JavaScript directement dans le code HTML

Germain BECKER & Sébastien POINT, Lycée Mounier, ANGERS

Pour écrire du JavaScript dans le code HTML, il suffit de l'écrire entre les balises `<script>` et `</script>`. Le code ci-dessous permet d'afficher le message « Hello World » à l'utilisateur grâce à la fonction `alert`.

```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <title>Hello World</title>
</head>
<body>
      <script>
            alert("Hello World !");
      </script>
</body>
</html>
```



À faire : écrivez ce fichier HTML avec un éditeur de texte puis ouvrez-le avec un navigateur. A l'ouverture, le navigateur analysera le code et affichera une fenêtre « pop-up » contenant le message « Hello World ! » comme ci-dessus.

Remarque : chaque instruction JavaScript doit se terminer par un point-virgule.

Il est donc possible d'intégrer le code JavaScript directement dans le code HTML. Cependant, les bonnes pratiques actuelles recommandent de séparer complètement le JavaScript du HTML (comme le CSS). Voici donc ce qu'il est recommandé de faire à la place.

2ème méthode (recommandée) : séparer le JavaScript du HTML

Il est préférable d'externaliser le code JS du code HTML, on devra donc utiliser deux fichiers séparés. Pour afficher le message précédent de cette façon, il est nécessaire de :

- Créer un fichier JS, par exemple appelé « script.js » ;
- Associer ce fichier JS au fichier HTML comme ci-dessous.

Le fichier HTML

```
<!doctype html>
<html lang="fr">
<head>
  <meta charset="utf-8">
  <title>Hello World</title>
</head>
<body>
  <!-- contenu HTML -->
  <script src="script.js"></script>
</body>
</html>
```

Le fichier `script.js` associé

```
alert("Hello World !");
```

Dans le HTML, on écrit la ligne surlignée en jaune pour faire référence au code JS, on utilise la balise `<script>` avec l'attribut `src` dont la valeur est le chemin du fichier dans lequel on écrit tout le code JS. **Cette ligne est à écrire juste avant la fermeture du `<body>`.**

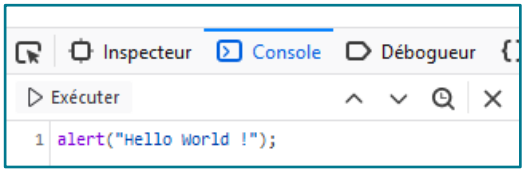
Germain BECKER & Sébastien POINT, Lycée Mounier, ANGERS

L'extension d'un fichier JS est « `.js` ». Ici, seul le nom du fichier a été écrit, il faut donc que le fichier `script.js` soit dans le même répertoire que le fichier HTML.

C. Comment tester du code JavaScript ?

On peut bien entendu utiliser un **éditeur de texte** pour écrire le code HTML et le code JavaScript (et le CSS), comme présenté dans le paragraphe précédent. C'est d'ailleurs ce qu'il faut faire lorsque l'on a projet Web complet à développer. Il en existe plein, par exemple Notepad++ sur Windows, Geany sous Linux, etc.

On peut également utiliser la **console JavaScript d'un navigateur**. Dans Firefox, on peut ouvrir la console Web par le raccourci Ctrl + Maj + K (ou dans Menu puis Développement web puis Console web). Il suffit alors d'écrire le code JS dans cette console puis l'exécuter.



Enfin, on peut utiliser un **bac à sable en ligne** comme CodePen (https://codepen.io/pen/) ou jsFiddle (https://jsfiddle.net/). Ces éditeurs permettent de tester des codes directement en ligne via un navigateur, sans enregistrer de fichiers sur votre ordinateur. (on peut bien entendu copier-coller dans un éditeur de texte, le travail fait en ligne pour enregistrer les fichiers sur son ordinateur). Ils ont l'avantage de présenter directement quatre fenêtres à l'écran : une pour le HTML, une pour le CSS, une pour le JS et une dernière pour l'affichage de la page Web. Leur utilisation est largement suffisante pour ce que vous avez à connaître cette année, et nous utiliserons l'éditeur CodePen.

II. Interaction avec l'utilisateur dans une page Web

A faire **Activité : JS et interaction avec l'utilisateur**

A. Les événements et leurs utilisations

Les événements permettent de déclencher une fonction selon qu'une action s'est produite ou non. Ces événements peuvent être associés à n'importe quel élément HTML (un bouton <button>, un paragraphe <p>, un titre <h.>, un bloc etc.).
Il existe beaucoup d'événements, en voici quelques-uns :

Evénement	Description
`click`	Cliquer sur l'élément
`dblclick`	Double-cliquer sur l'élément
`mouseover`	Faire entrer le curseur sur l'élément
`mouseout`	Faire sortir le curseur de l'élément
`keydown`	Appuyer (sans relâcher) sur une touche de clavier sur l'élément
`keyup`	Relâcher une touche du clavier
`change`	Changer la valeur d'un champ de formulaire

Germain BECKER & Sébastien POINT, Lycée Mounier, ANGERS

Exemple : On souhaite associer à un bouton, l'événement consistant à afficher le message « Hello World ! » dès lors que l'on clique sur ce bouton. Nous proposons ci-dessous les trois méthodes permettant de le faire. La première étant obsolète elle ne doit plus être utilisée ! Pour gagner en temps et en place, on ne présente que le contenu du `<body>` du code HTML.

1ère méthode : Utiliser les attributs HTML du gestionnaire d'événements (très ancienne → OBSOLETE donc à ne plus utiliser car obsolète !)

Codes HTML et JS

```
<body>
  <button onclick="printMsg()">Clic</button>
  <script>
    function printMsg() {
      alert("Hello World !");
    }
  </script>
</body>
```

Un attribut `onclick` est ajouté au bouton et sa valeur est la fonction JavaScript contenant le code à exécuter dès que l'événement survient. Cette fonction est directement définie dans le code HTML dans un élément `<script>`. On ne procède plus ainsi pour plein de raisons que l'on ne détaillera pas ici. Retenez juste qu'il faut séparer le code JS du code HTML !

2ème méthode : Utiliser les propriétés du gestionnaire d'événements (ancienne mais peut être utilisée)

Code HTML

```
<body>
  <button id='btn'>Clic</button>
  <script src="script.js"></script>
</body>
```

Code JS (fichier `script.js`)

```
var bouton = document.querySelector('#btn');

function printMsg() {
  alert("Hello World !");
}

bouton.onclick = printMsg;
```

Ou de manière plus synthétique :

```
var bouton = document.querySelector('#btn');

bouton.onclick = function() {
  alert("Hello World !");
}
```

Dans le code HTML, on a ajouté un attribut `id` au bouton afin de pouvoir sélectionner cet élément HTML en JavaScript. En effet, la première ligne du code JS sélectionne notre bouton (l'élément HTML dont l'`id` vaut `'btn'`) et stocke cet élément dans une variable appelée `bouton`. La dernière ligne indique que c'est la propriété `onclick` du gestionnaire d'événement qui est utilisée : dès que l'on clique sur le bouton la fonction `printMsg` est appelée et s'exécute.

Remarque : on peut remplacer la propriété `onclick` par une autre : par exemple, en la remplaçant par la propriété `ondblclick` la fonction serait exécutée dès que l'on double-clique sur le bouton.

Germain BECKER & Sébastien POINT, Lycée Mounier, ANGERS

3ème méthode : Utiliser la fonction `addEventListener()` (la plus récente donc celle à privilégier)

C'est le mécanisme d'événement le plus récent (défini dans le Document Object Model, ou DOM) et le plus flexible (on ne rentrera pas dans les détails ici). C'est celui qui a été présenté dans les vidéos de l'activité.
Celui-ci fournit aux navigateurs une nouvelle fonction appelée `addEventListener()` qui fonctionne de la même manière que les propriétés du gestionnaire d'événement, mais la syntaxe est légèrement différente.

Code HTML

```
<body>
  <button id='btn'>Clic</button>
  <script src="script.js"></script>
</body>
```

Code JS (fichier `script.js`)

```
var bouton = document.querySelector('#btn');

function printMsg() {
      alert("Hello World !");
}

bouton.addEventListener('click', printMsg);
```

Ou de manière plus synthétique :

```
var bouton = document.querySelector('#btn');

bouton.addEventListener('click', function() {
  alert("Hello World !");
}
```

Dans la fonction `addEventListener()`, il faut spécifier deux paramètres : le nom de l'événement (ici `click` mais on peut remplacer par `mouseover` ou un autre) et le nom de la fonction à exécuter en réponse à cet événement.

B. Modifier les éléments de la page

On considérera la page HTML ci-dessous pour les exemples qui suivent.

```
<body>
  <h1 id="monTitre">Généralités sur JS</h1>
  <p id="monPara">Voici une page Web
interactive grâce à JavaScript.</p>
  <button id="btn">Cliquez ici</button>
</body>
```

Voici quelques exemples de fonctions permettant de modifier les propriétés des éléments de la page Web. On suppose que ces fonctions sont appelées lors du clic sur le bouton de la page comme on l'a vu dans le paragraphe précédent.

Modifier le style d'un élément : la propriété `style`

On peut modifier le style d'un élément HTML en utilisant la propriété `style` de cet élément. Par exemple, cette fonction Javascript

```
function changeStyles() {
  var para = document.querySelector("#monPara");
  var corps = document.querySelector("body");
  para.style.color="red";
  para.style.fontWeight="bold";
  corps.style.backgroundColor="rgb(255,255,0)";
}
```

permet de modifier le style du paragraphe : le texte passe en rouge et en gras ; le style du corps de la page : la couleur de fond passe en jaune.

Changer le texte d'une balise : la propriété `innerHTML`

On peut changer le texte d'un élément HTML en utilisant la propriété `innerHTML` de cet élément. Par exemple, cette fonction Javascript

```
function changeTexte() {
  var para = document.querySelector("#monPara");
  para.innerHTML = "Bonjour tout le monde !";
}
```

permet de modifier le texte du paragraphe.

Utiliser des champs de saisie : la propriété value

On ajoute dans le code HTML un élément `<input>` (=champ de saisie) avec la ligne

```
<input type="text" id="zoneDeSaisie"/>
```

dans lequel on demande à l'utilisateur d'écrire son prénom.
Il est alors possible de récupérer le prénom saisi en utilisant la propriété `value` de cet élément. Par exemple, si l'utilisateur a saisi le prénom « Brendan » alors cette fonction Javascript

```
function disBonjour() {
  var saisie = document.querySelector("#zoneDeSaisie");
  var para = document.querySelector("#monPara");
  let prenom = saisie.value;
  let texte = "Bonjour " + prenom;
  para.innerHTML = texte;
}
```

permet de récupérer le texte saisi grâce à la propriété value de l'élément <input> puis de construire la chaîne de caractères « Bonjour Brendan » et de l'écrire dans la paragraphe.

Ressources :

- Introduction aux événements, MDN Mozilla : https://developer.mozilla.org/fr/docs/Apprendre/JavaScript/Building_blocks/Ev%C3%A8nements
- Wikipédia : https://fr.wikipedia.org/wiki/JavaScript
- Cours OpenClassrooms, Premiers pas en JavaScript : https://bit.ly/2UtkVOb