NSI - Première

IHM sur le web - introduction à JavaScript

qkzk

2021/07/27

JavaScript

Avant d'entrer dans le vif du sujet, un petit historique :

- 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.
- JavaScript a été conçu pour être exécuté directement *par le navigateur* et côté client.

  Quand vous ouvrez une page web contenant du JS, il sera exécuté par *votre* machine et non par le serveur. C'est très important, car cela permet au serveur de limiter sa *charge*. 100 clients en même temps ? Il sert *une fois* la page par client et les calculs sont effectués chez ceux-ci.
- HTML5 est une norme qui voit le jour en 2014 décrivant le web moderne. Par abus de langage, HTML5 désigne le trio 'html, css, js'.
- Depuis quelques années JavaScript a beaucoup évolué et il existe de nombreux projets majeurs :
  - possibilité d'exécuter un **serveur** qui fonctionne en JS : c'est **node.js** (2009)
  - possibilité de créer des applications de bureau qui fonctionnent sur un serveur **node.js** et s'exécutent dans "chromium" (partie open source de Google Chrome). Par exemple : `atom` (présent sur vos machines) qui utilise la technologie **electron (2013/2016)**.
- JavaScript est le langage informatique le plus populaire depuis quelques années, il a surpassé Java, C/C++ avec l'essor du web. Il est talonné par Python. En 2019, presque tous les éléments "dynamiques" d'une page web moderne sont programmés en JavaScript.

Inclure un script JS dans une page web

Un script JavaScript est chargé et exécuté dans une balise `<script> </script>`.

Il peut être soit inclu entre les balises soit chargé depuis un fichier extérieur.

Considérons l'exemple suivant :

```
<html>
  <head>
    <title>Exemple JS</title>
    <body>
      <h1 id="titre">Titre</h1>
      <p id="para">Paragraphe</p>
    </body>
</html>
```

On insère maintenant une balise script à la dernière ligne du body :

Exercice : écrire le rendu de cette page.

```
<html>
  <head>
    <title>Exemple JS</title>
    <body>
      <h1 id="titre">Titre</h1>
      <p id="para">Paragraphe</p>
      <script>
        document.getElementById("titre").innerHTML = "Nouveau titre";
      </script>
    </body>
</html>
```

Que fait-elle ?

- `document` désigne l'ensemble du document, de la page web.
- `getElementById("titre")` est un *sélecteur* qui va récupérer la première balise dont l'id est `"titre"`.
- `.innerHTML = "Nouveau titre"` va remplacer le contenu html de la balise sélectionnée par "Nouveau titre".

Ainsi JS permet de changer le contenu d'une balise.

Il existe de nombreuses méthodes permettant de faire ce genre de choses.

Modifier le style

Une fois qu'on a récupéré un *élément html*, on peut lui attribuer du style, par exemple :

```
var element = document.getElementById("titre");
element.style.color = "red";
```

Exercice : lire attentivement et écrire le CSS équivalent à ce script.

Éléments de syntaxe

- Commentaire

  ```
  // ceci es un commentaire
  ```

- Variable

  ```
  a = 1; // une variable globale
  var b = 2; // une variable locale
  let c = 3; // une variable dont la portée est limitée au bloc courant
  ```

- conditions

  ```
  if (condition) {
      inscrution;
    }
  ```

- boucles

  ```
  for (let i = 0; i < 10; i++) {
      // corps de la boucle
  }
  ```

  Pour `i` entre 0 et 9 et progressant de 1.

  Équivalent à :

```
for i in range(10):
  blablabla
```

- Afficher un message dans la console de développeur

```
var prenom = "Nina";
console.log("bonjour", prenom);
```

Va afficher “bonjour nina” dans la console.

Exercice. Afficher “J’ai fait 10 pompes !”, “J’ai fait 20 pompes”, etc. jusque “J’ai fait 200 pompes !” dans la console.

Modifier les attributs html

Considérons ce contenu html

```
<html>
  <body>
    <p class="monPara">bonjour</p>
  </body>
</html>
```

Insérons maintenant un script JS à la fin du body et le fichier css suivant :

```
#rouge {
  color: red;
}
#green {
  color: green;
}
```

```
<html>
  <head>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <p id="monPara">bonjour</p>
    <p class="rouge">rouge</p>
    <p class="vert">vert</p>
    <script>
      var monPara = document.getElementById("monPara");
    </script>
  </body>
</html>
```

Exercice : écrire le rendu de cette page. *Clic Clac c’est le son du 4 Couleurs...*

Maintenant nous allons ajouter à l’élément sélectionné dans le script :

```html
<html>
  <head>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <p id="monPara">bonjour</p>
    <p class="rouge">rouge</p>
    <p class="vert">vert</p>
    <script>
      var monPara = document.getElementById("monPara");
      monPara.classList.add("rouge");
    </script>
  </body>
</html>
```

Et javascript va ajouter la classe `"rouge"` et son style à notre paragraphe.

Intéractivité

Ajoutons maintenant des boutons...

```html
<html>
  <head>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <p id="monPara">bonjour</p>
    <p class="rouge">rouge</p>
    <p class="vert">vert</p>
    <button onclick="foncRouge()">Rouge</button>
    <button onclick="foncVert()">Vert</button>
    <script>
      var monPara = document.getElementById("monPara");
      monPara.classList.add("rouge");
    </script>
  </body>
</html>
```

Que font-ils ? Rien.

Ou plutôt, lorsqu'on clique dessus, JS déclanche un événément "onclick" qui à son tour, *tente* d'exécuter une fonction `foncRouge` ou `foncVert` mais celles-ci n'étant pas définies... il ne se passe rien.

Définissons les !

```html
<html>
  <head>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <p id="monPara">bonjour</p>
    <p class="rouge">rouge</p>
    <p class="vert">vert</p>
    <button onclick="foncRouge()">Rouge</button>
    <button onclick="foncVert()">Vert</button>
    <script>
      var monPara = document.getElementById("monPara");
      monPara.classList.add("rouge");

      function foncRouge() {
          monPara.classList.remove("vert");
          monPara.classList.add("rouge");
      }
      function foncVert() {
          monPara.classList.remove("rouge");
          monPara.classList.add("vert");
      }
    </script>
  </body>
</html>
```

Remarquons d'abord la syntaxe des fonctions (il en existe d'autres) :

```
function maFonction() {
    inscrutions;
  }
```

Que font ces fonctions :

- la première retire la classe `"vert"` et ajoute la classe `"rouge"`
- la seconde fait le contraire.

Résumons, lorsqu'on appuie sur le bouton "rouge", on colorie le paragraphe "monPara" en rouge, et lorsqu'on appuie sur le bouton "vert"... Je vous laisse deviner.

Exercice : Modifier les fonctions afin de changer aussi le contenu du paragraphe lorsqu'on clique sur les boutons.

Programmation *événementielle*

Python utilise la programmation *séquentielle* où les lignes sont exécutées les une après les autres. L'exécution d'une fonction est donc prédite par le contenu du programme.

Le programme sera principalement défini par ses réactions aux différents événements qui peuvent se produire, c'est-à-dire des changements d'état de variable, par exemple l'incrémentation d'une liste, un déplacement ou un click de souris, une saisie au clavier...

Un autre exemple d'événement :

- `onmouseover` : lorsque la souris passe sur un élément
- `onmouseout` : lorsque la souris quitte un élément

Exercice : remplacer les boutons `rouge` et `vert` afin que le texte du paragraphe soit toujours vert, sauf quand la souris le survole auquel cas il devient rouge.

Il existe beaucoup d'autres événements que nous n'aborderons pas ici. Si vous voulez en savoir plus, vous pouvez consulter ce site.

En résumé, le code HTML permet de générer des éléments graphiques qui seront affichés par un navigateur web, mais pas seulement : il est aussi possible de mettre en place dans le code HTML des événements. Un événement donné pourra déclencher l'exécution d'instructions JavaScript.