JavaScript

Interactions sur le Web

Introduction

JavaScript est le langage de programmation des navigateurs : il rend les pages Web dynamiques en réagissant aux actions de l'utilisateur et en modifiant la page grâce au DOM. On pourra consulter ce site pour un aperçu de ses possibilités.

Contrairement à Python qui utilise les indentations, JavaScript délimite les blocs par des accolades (les indentations ne servent qu'à la lisibilité). Voici la même fonction dans les deux langages :

Python

```python
def ma_fonction():
    for i in range(5):
        print(i)
```

JavaScript

```javascript
function maFonction() {
  for (let i = 0; i < 5; i++) {
    console.log(i);
  }
}
```

Pour les noms de variables et de fonctions, Python utilise la convention `snake_case` (mots en minuscules séparés par des tirets bas) et JavaScript la convention `camelCase` (chaque mot après le premier commence par une majuscule).

Les scripts s'accumulent vite et rendent le code HTML difficile à lire. Il est donc d'usage de les écrire dans un fichier `script.js` (`js` est l'extension JavaScript) et de l'appeler depuis la balise `head`, comme on le fait pour le CSS. L'attribut `defer` demande au navigateur d'exécuter le script une fois la page construite, sans quoi `getElementById` ne trouverait pas encore les éléments :

```html
<script src="script.js" defer></script>
```

Les événements

Un **événement** est un fait qui se produit dans la page : un clic, une touche enfoncée, une saisie dans un champ, la fin du chargement. Un **gestionnaire d'événement** est une fonction que le navigateur exécute quand l'événement se produit. Deux écritures :

— dans le HTML, avec un attribut de la balise : `<button onclick="maFonction()">` ;

— dans le script, avec `addEventListener("click", maFonction)` sur l'élément : le HTML ne contient plus de code et un même élément peut avoir plusieurs gestionnaires.

Les événements les plus utilisés sont `click`, `input` (le contenu d'un champ change), `change`, `submit`, `keydown` et `load`. La fonction reçoit un objet `event` : `event.target` désigne l'élément concerné et `event.preventDefault()` annule l'action par défaut (par exemple l'envoi d'un formulaire). Les exemples 1 à 7 utilisent l'attribut `onclick`, l'exemple 8 utilise `addEventListener`.

Recopier chacun des exemples suivants en créant un fichier `html` pour chacun d'entre eux afin d'en tester le contenu. Sous chaque code, le résultat est affiché et fonctionne.

Exemple 1

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Exemple 1</title>
  </head>
  <body>
    <button type="button" onclick="document.getElementById('demo1').innerHTML = Date()">
      Cliquez-moi pour afficher la date et l'heure.
    </button>

    <p id="demo1"></p>
  </body>
</html>
```

Exemple 2

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Exemple 2</title>
  </head>
  <body>
    <p id="demo2">JavaScript peut changer du contenu HTML.</p>

    <button type="button" onclick='document.getElementById("demo2").innerHTML = "Bonjour JavaScript !"'>
      Cliquez-moi !
    </button>
  </body>
</html>
```

Exemple 3

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
```

```
    <title>Exemple 3</title>
  </head>
  <body>
    <p>JavaScript peut changer la valeur de l'attribut src (source) d'une image.</p>
    <button onclick="document.getElementById('lampe').src = 'pic_bulbon.gif'">
      Allumer la lumière
    </button>
    <img id="lampe" src="pic_bulboff.gif" style="width:100px">
    <button onclick="document.getElementById('lampe').src = 'pic_bulboff.gif'">
      Éteindre la lumière
    </button>
  </body>
</html>
```

Les deux images sont à télécharger : pic_bulbon.gif et pic_bulboff.gif.

Exemple 4

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Exemple 4</title>
  </head>
  <body>
    <p id="demo3">JavaScript peut changer le style d'un élément HTML.</p>

    <button type="button" onclick="document.getElementById('demo3').style.fontSize = '35
    px'">
      Cliquez-moi !
    </button>
  </body>
</html>
```

Exemple 5

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Exemple 5</title>
    <style>
      #maDIV {
        width: 100%;
        padding: 50px 0;
        text-align: center;
        background-color: lightblue;
        margin-top: 20px;
      }
    </style>
    <script>
      function basculer() {
        const x = document.getElementById("maDIV");
        if (x.style.display === "none") {
```

```
          x.style.display = "block";
        } else {
          x.style.display = "none";
        }
      }
    </script>
  </head>
  <body>
    <p>JavaScript peut cacher ou faire apparaître du contenu.</p>

    <button onclick="basculer()">Cliquez-moi !</button>

    <div id="maDIV">Ceci est le contenu d'une balise DIV.</div>
  </body>
</html>
```

Exemple 6

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Exemple 6</title>
    <script>
      function recupValeurForm() {
        const formulaire = document.getElementById("form1");
        for (const champ of formulaire.elements) {
          if (champ.type === "text") {
            alert(champ.name + " : " + champ.value);
          }
        }
      }
    </script>
  </head>
  <body>
    <p>JavaScript peut exploiter un formulaire.</p>

    <form id="form1">
      <label>Prénom : <input type="text" name="prenom" value="Leo"></label>

      <label>Nom : <input type="text" name="nom" value="Messi"></label>

      <button type="button" onclick="recupValeurForm()">Afficher</button>
    </form>
  </body>
</html>
```

Le bouton est de type `button` et non `submit` : le formulaire n'est pas envoyé et la page n'est pas rechargée.

Exemple 7

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Exemple 7</title>
    <script>
      function listerOptions() {
        const liste = document.getElementById("couleurs");
        let texte = "Nombre d'éléments : " + liste.length;
        for (const option of liste.options) {
          texte = texte + "\n" + option.text;
        }
        alert(texte);
      }
    </script>
  </head>
  <body>
    <form>
      <label>Sélectionnez votre couleur préférée :
        <select id="couleurs">
          <option>Rouge</option>
          <option>Vert</option>
          <option>Bleu</option>
          <option>Blanc</option>
        </select>
      </label>
      <button type="button" onclick="listerOptions()">Compter et afficher tous les élé
    ments</button>
    </form>
  </body>
</html>
```

Exemple 8 : avec addEventListener

Le même compteur, écrit sans aucun code dans le HTML : le script attache les gestionnaires aux éléments.

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Exemple 8</title>
    <script src="script.js" defer></script>
  </head>
  <body>
    <p>Valeur : <span id="valeur">0</span></p>
    <button id="plus">+1</button>
    <button id="zero">Remise à zéro</button>
    <p>Saisir un prénom : <input type="text" id="prenom"> <span id="salut"></span></p>
  </body>
</html>
```

```js
// script.js
const affichage = document.getElementById("valeur");

document.getElementById("plus").addEventListener("click", function () {
  affichage.textContent = Number(affichage.textContent) + 1;
});

document.getElementById("zero").addEventListener("click", function () {
  affichage.textContent = 0;
});

document.getElementById("prenom").addEventListener("input", function (event) {
  document.getElementById("salut").textContent = "Bonjour " + event.target.value + " !";
});
```

Pour s'entraîner : fiche d'exercices 24 - JavaScript.