

# Cours

## 1) introduction

Nous avons déjà pu nous familiariser avec le couple HTML-CSS, en fait, le couple est plutôt un trio, car aujourd'hui un développeur web ne peut pas faire l'impasse sur le JavaScript .

Notre but ici n'est pas d'apprendre un nouveau langage de programmation, mais juste d'étudier quelques exemples d'utilisation du JavaScript, notamment dans le cas des interactions entre un utilisateur et une page web.

Avant d'entrer dans le vif du sujet, un petit rappel 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 langages, 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) intégrer le JavaScript

Pour être exécuté, le JavaScript doit être intégré à la page HTML. Il existe différentes méthodes pour intégrer le JavaScript, nous allons en étudier une : le code JavaScript est écrit dans un fichier avec l'extension .js (par exemple script.js). Il faut ensuite ajouter une ligne dans le code HTML afin de lier ce dernier au fichier *script.js* :

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

l'attribut *src* de la balise *script* doit être égal au chemin vers le fichier qui contient le JavaScript.

Voici un exemple de fichier HTML associé avec un fichier *script.js* :

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

Comme vous pouvez le constater ci-dessus la balise *script* est placée après la balise *body* fermante (il est aussi possible de placer la balise *script* dans la partie *head*, mais ce choix peut entraîner certains problèmes si on ne prend pas certaines précautions, on préférera donc placer la balise *script* après le *body*).

### 3) le JavaScript pour répondre à des événements

#### a) notion d'événements

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 *onclick* 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.

#### b) réponse à un événement

Un clic de souris vient d'être effectué sur un élément HTML, que va-t-il se passer ?

Dans la plupart des cas rien...si le développeur n'a pas prévu de gérer cet événement, il ne se passera rien.

Comment gérer un événement ?

Il suffit d'ajouter un attribut à la balise HTML concernée, prenons un exemple :

```
<button onclick="maFonction()">Cliquer ici</button>
```

Nous avons ici une balise *button* qui possède un attribut *onclick*. Cet attribut *onclick* est égal à *maFonction()*. *maFonction* est une fonction défini dans le fichier JavaScript qui est associé à la page HTML (par exemple le fichier *script.js*). En cas de clic sur le bouton défini ci-dessus, la fonction JavaScript *maFonction* sera exécutée.

#### c) les fonctions en JavaScript

Comment écrire une fonction en JavaScript ?

Il suffit d'utiliser le mot clé *function* suivi du nom de la fonction :

```
function Mafonction(){
    .....
}
```

Les limites de la fonction sont définies par l'accolade ouvrante et l'accolade fermante.

## 4) Modifier un élément HTML avec le JavaScript

En cas de clic sur le bouton défini ci-dessus, la fonction JavaScript *maFonction* sera exécutée, mais qu'est-il possible de faire avec le JavaScript ?

Aujourd'hui, presque tout : il est possible de coder des applications extrêmement complexes en JavaScript. Nous allons rester modestes, nous allons uniquement modifier un élément HTML.

Pour modifier un élément HTML il est tout d'abord nécessaire de le sélectionner : c'est le rôle des *querySelector*.

Considérons le code HTML suivant :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Le trio</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <h1>Le trio : HTML, CSS et JavaScript</h1>
    <p id="monPara">Voici une page web qui ne fait pas grand chose</p>
    <button onclick="maFonction()">Cliquer ici</button>
  </body>
  <script src="script.js"></script>
</html>
```

et le code JavaScript suivant (fichier *script.js*) :

```
function maFonction() {
    let maBalise = document.querySelector("#monPara");
}
```

Le mot clé *let* permet de définir une variable, dans l'exemple ci-dessus, la variable a pour nom *maBalise* (on aurait évidemment pu choisir un autre nom). Cette variable *maBalise* va nous permettre de manipuler l'élément HTML qui a pour *id monPara* (c'est-à-dire la balise *p* présente dans le code HTML ci-dessus).

Maintenant que nous avons "accès" à une balise HTML depuis le code JavaScript, il est possible de modifier le contenu de cette balise ou encore le style associé à cette balise comme bon nous semble :

```
function maFonction() {
    let maBalise = document.querySelector("#monPara");
    maBalise.innerHTML="coucou"
}
```

dans le cas ci-dessus nous remplacerons le texte présent dans la balise `p` ("Voici une page web qui ne fait pas grand chose") par "coucou".

Autre exemple :

```
function maFonction() {
    let maBalise = document.querySelector("#monPara");
    maBalise.style.color="red";
}
```

dans le cas ci-dessus, le texte du paragraphe sera écrit en rouge.

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

soit le code HTML :

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="utf-8">
    <title>Le trio</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <h1>Le trio : HTML, CSS et JavaScript</h1>
    <p id="monPara">Voici une page web qui ne fait pas grand-chose</p>
    <button onclick="foncRouge()">Rouge</button>
  </body>
  <script src="script.js"></script>
</html>
```

soit le code CSS (fichier "style.css") :

```
h1{
    text-align: center;
}
.rou {
    color: red;
}
```

soit le code JavaScript (fichier "script.js") :

```
function foncRouge() {
    let maBalise = document.querySelector("#monPara");
    maBalise.classList.add("rou");
}
```

```
}
```

Résumons ce qui peut se passer avec la page HTML ci-dessus :

- un utilisateur clique sur le bouton *Rouge*, comme un événement *onclick* est associé à ce bouton, la fonction JavaScript *foncRouge* est donc exécutée
- la variable *maBalise* correspond à la balise *p* du code HTML (*let maBalise = document.querySelector("#monPara");*)
- on associe ensuite cette balise *p* à la classe CSS *rouge* : le texte contenu dans la balise *p* devient donc rouge (*maBalise.classList.add("rou");*)