Apprendre JavaScript - Les boucles

Table des matières

Introduction

Les boucles sont des structures de contrôle qui permettent de répéter une ou plusieurs instructions plusieurs fois. Il existe plusieurs types de boucles en JavaScript, et chacune a ses particularités. Nous regarderons les boucles `for`, `while`, `forEach` et `for...in`.

Il existe beaucoup d'autres méthodes pour boucler plus spécialisées, mais nous verrons cela dans un prochain cours. Ex: map, filter, reduce, some, every, etc.

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Statements/for
https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Statements/while
https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach
https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Statements/for...in

La boucle `for`

La boucle `for` est la plus utilisée en JavaScript. Elle permet de répéter une ou plusieurs instructions un nombre de fois déterminé. La syntaxe de la boucle `for` est la suivante :

```
for (initialisation; condition; incrémentation) {
    // Instructions à répéter
}
```

- `initialisation` : permet de déclarer et d'initialiser une variable qui servira de compteur.
- `condition` : permet de définir la condition qui doit être vraie pour que la boucle continue à s'exécuter.
- `incrémentation` : permet de modifier la valeur du compteur à chaque itération.

Donc pour exécuter une instruction 10 fois, on peut écrire :

```
for (let i = 0; i < 10; i++) {
    console.log("Instruction " + i);
```

```
}
```

La variable `i` est initialisée à 0. Tant que `i` est inférieur à 10, la boucle continue à s'exécuter. À chaque itération, la variable `i` est incrémentée de 1. Elle n'est pas disponible en dehors de la boucle.

Chaque paramètre de la boucle `for` est séparé par un point-virgule.

La boucle `while`

La boucle `while` est une autre structure de contrôle qui permet de répéter une ou plusieurs instructions tant qu'une condition est vraie. La boucle while est souvent utilisée lorsqu'on ne connaît pas à l'avance le nombre d'itérations à effectuer, "Tant que" une condition est vraie, on continue à exécuter les instructions.

Comme le risque de boucle infinie est élevé, il est important de s'assurer que la condition finira par devenir fausse à un moment donné.

La syntaxe de la boucle `while` est la suivante :

```
while (condition) {
    // Instructions à répéter
}
```

Par exemple, pour exécuter une instruction 10 fois, on peut écrire :

```
let i = 0;

while (i < 10) {
    console.log("Instruction " + i);
    i++; // Ne pas oublier d'incrémenter la variable, sinon la boucle sera infinie
}
```

foreach

La méthode `forEach` est une méthode qui permet de parcourir un tableau. Elle prend en argument une fonction qui sera exécutée pour chaque élément du tableau. Elle est plus élégante que la boucle `for`, et elle est souvent préférée par les développeurs JavaScript pour parcourir un tableau.

La syntaxe de la méthode `forEach` est la suivante :

```
tableau.forEach(function (element, index, tableau) {
    // Instructions à exécuter
});
```

for...in

La boucle `for...in` est une autre structure de contrôle qui permet de parcourir les propriétés d'un objet. Elle est souvent utilisée pour parcourir les clés d'un objet. Nous n'avons pas vu les objets pour le moment, mais nous verrons cela dans un prochain cours. C'est un peu comme un boucle `foreach` pour les clés des objets.

La syntaxe de la boucle `for...in` est la suivante :

```
let objet = {
    propriete1: "valeur1",
    propriete2: "valeur2",
    propriete3: "valeur3",
};

for (let propriete in objet) {
    // Instructions à exécuter
    let valeur = objet[propriete];
    console.log(propriete + " : " + valeur);
}
```

Boucler sur un tableau d'éléments HTML

Il est possible de boucler sur un tableau d'éléments HTML pour appliquer une action à chaque élément. Par exemple, pour ajouter une classe à chaque élément d'une liste.

Pour récupérer un groupe d'élements HTML, on utilise la méthode `querySelectorAll`. Cette méthode retourne un tableau d'éléments HTML.

```
let elements = document.querySelectorAll("li");

elements.forEach(function (element) {
    element.classList.add("classe");
});

// ou

for (let i = 0; i < elements.length; i++) {
    elements[i].classList.add("classe");
}
```