



# Cours JavaScript

Date : 03/04/2023

## Les boucles en JavaScript

En JavaScript, il existe plusieurs types de boucles pour itérer sur des tableaux, des objets ou des séquences de données. Les plus courantes sont `for`, `for...in`, `for...of` et `forEach`.

### La boucle For

La boucle for est la boucle la plus courante en JavaScript. Elle est utilisée pour itérer un nombre spécifié de fois sur un bloc de code. Elle prend trois arguments : l'initialisation de la variable de contrôle de boucle, la condition d'arrêt de la boucle et l'expression de mise à jour de la variable de contrôle de boucle.

```
1 for (let i = 0; i < 5; i++) {  
2   console.log(i);  
3 }
```

Dans cet exemple, la boucle for va itérer cinq fois, de 0 à 4 inclus, et afficher chaque valeur de i.

Imaginons donc un tableau qui contient 5 valeurs comme suit:

```
1 let monTableau = ['Cedric', 'Angelika', 'Marie-Claire', 'Bessem', 'David'];
```

Le premier élément de mon tableau 'Cedric' aura l'index '0', pour y accéder nous écrirons le nom du tableau suivi directement de l'index entre crochets: `monTableau[0]`; Le second élément de mon tableau 'Angelika' aura l'index '1', pour y accéder nous écrirons le nom du tableau suivi directement de l'index entre crochets: `monTableau[1]`; et ainsi de suite ...

Nous savons que notre tableau contient 5 éléments, nous pourrions donc itérer sur le tableau comme nous l'avons vu dans l'exemple précédent, ainsi au lieu d'accéder à chaque élément du tableau en écrivant tous les index comme nous le ferions sans boucle nous pouvons remplacer l'index par i.

```
1 let monTableau = ['Cedric', 'Angelika', 'Marie-Claire', 'Bessem', 'David'];  
2  
3 for (let i = 0; i < 5; i++) {  
4   console.log(monTableau[i]);  
5 }
```

Nous venons de nous épargner d'écrire une ligne de code par élément du tableau. Mais nous pouvons améliorer notre boucle en prenant en compte que le tableau est une variable puisqu'il est instancié par le mots clé 'let'. Si je rajoute un élément dans mon tableau et que je laisse la boucle tel quel celle-ci ne prendra en compte que les cinq premiers éléments:

```
1 let monTableau = ['Cedric', 'Angelika', 'Marie-Claire', 'Bessem', 'David', 'Lisa'];  
2  
3 for (let i = 0; i < 5; i++) {  
4   console.log(monTableau[i]);  
5 }
```

Dans cet exemple, tous les noms des élèves sont affichés dans la console jusqu'à David car celui-ci est le cinquième élément du tableau et que la condition d'arrêt de la boucle est 'i < 5'. Nous pouvons donc modifier la condition d'arrêt en écrivant tant que i est inférieur à la longueur du tableau comme suit:

```
1 let monTableau = ['Cedric', 'Angelika', 'Marie-Claire', 'Bessem', 'David', 'Lisa'];  
2  
3 for (let i = 0; i < monTableau.length; i++) {  
4   console.log(monTableau[i]);  
5 }
```

De cette façon, peu importe le nombre d'élèves que nous ajouterons au tableau, ceux-ci seront tous affichés dans la console.

## La boucle for...in

La boucle for...in est utilisée pour itérer sur les propriétés d'un objet. Elle permet d'itérer sur les noms des propriétés de l'objet, plutôt que sur les valeurs elles-mêmes.

```
1  const eleve = {  
2    name: 'Remi',  
3    age: 30,  
4    ecole: 'Simplon'  
5  };  
6  
7  for (let property in eleve) {  
8    console.log(`${property}: ${eleve[property]}`);  
9  }
```

Dans cet exemple, la boucle for...in va itérer sur les propriétés de l'objet 'eleve' et afficher leur nom et leur valeur.

## La boucle for...of

La boucle for...of est utilisée pour itérer sur les éléments d'un tableau ou d'un autre objet itérable, tels que des chaînes de caractères.


```
1  let eleves = ['Cedric', 'Angelika', 'Marie-Claire', 'Bessem', 'David', 'Lisa'];  
2  
3  for (const eleve of eleves) {  
4    console.log(eleve);  
5  }
```

Dans cet exemple, la boucle for...of va itérer sur les éléments du tableau 'eleves' et afficher chaque élément.

## La méthode forEach


La méthode `forEach` est utilisée pour itérer sur les éléments d'un tableau. Elle est appelée pour chaque élément du tableau et peut prendre une fonction en argument qui sera exécutée pour chaque élément.

Exemple avec une fonction fléchée:

A code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains five lines of JavaScript code:

```
1 let eleves = ['Cedric', 'Angelika', 'Marie-Claire', 'Bessem', 'David', 'Lisa'];
2
3 eleves.forEach(eleve => {
4   console.log(eleve);
5 });
```

Exemple avec une fonction simple:

A code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains five lines of JavaScript code:

```
1 let eleves = ['Cedric', 'Angelika', 'Marie-Claire', 'Bessem', 'David', 'Lisa'];
2
3 eleves.forEach(function(eleve) {
4   console.log(eleve);
5 });
```

Dans ces exemples, la méthode `forEach` va itérer sur les éléments du tableau 'eleves' et exécuter la fonction pour chaque élément.

## La méthode map

La méthode `map` est utilisée pour transformer les éléments d'un tableau. Elle prend une fonction en argument qui sera exécutée pour chaque élément du tableau et qui retourne la nouvelle valeur de l'élément. La méthode `map` retourne un nouveau tableau contenant les nouvelles valeurs.

La syntaxe de la méthode `map()` est la suivante :

```
1 const nouveauTableau = tableauOriginal.map(function(elementCourant, index, tableau) {  
2   // Instructions à exécuter pour chaque élément du tableau  
3 });
```

- `nouveauTableau` est le tableau qui sera créé à partir des éléments du tableau original ;
- `tableauOriginal` est le tableau à transformer ;
- `elementCourant` est l'élément en cours de traitement dans le tableau ;
- `index` est l'index de l'élément en cours de traitement dans le tableau ;
- `tableau` est le tableau original sur lequel la méthode `map()` a été appelée.

La méthode `map()` retourne un nouveau tableau, contenant les éléments transformés. Le tableau original n'est pas modifié.

Voici un exemple d'utilisation de la méthode `map()` :

```
1 const numbers = [1, 2, 3];  
2  
3 const doubledNumbers = numbers.map(function(number) {  
4   return number * 2;  
5 });  
6  
7 console.log(doubledNumbers);
```

Dans cet exemple, la méthode `map` va transformer chaque élément du tableau `numbers` en le multipliant par 2 et retourner un nouveau tableau contenant les nouvelles valeurs. Le tableau résultant sera `[2, 4, 6]`.

## En résumé

Pour les boucles avec les objets, on utilise souvent la boucle `for...in` ou la méthode `Object.keys()` pour parcourir les propriétés de l'objet.

- La boucle `for...in` : elle permet de parcourir toutes les propriétés énumérables d'un objet et de les traiter une à une.

Voici un exemple :



```
1  const obj = { a: 1, b: 2, c: 3 };
2
3  for (let prop in obj) {
4      console.log(`${prop} = ${obj[prop]}`);
5  }
6  // Output :
7  // a = 1
8  // b = 2
9  // c = 3
```

- La méthode `Object.keys()` : elle retourne un tableau des propriétés énumérables d'un objet. On peut ensuite utiliser une boucle pour parcourir ce tableau et traiter chaque propriété.

Voici un exemple :

```
1  const obj = { a: 1, b: 2, c: 3 };
2
3  const keys = Object.keys(obj);
4
5  for (let i = 0; i < keys.length; i++) {
6    console.log(`${keys[i]} = ${obj[keys[i]]}`);
7  }
8  // Output :
9  // a = 1
10 // b = 2
11 // c = 3
```

Enfin, la méthode `forEach()` est une méthode disponible pour les tableaux en JavaScript, qui permet de parcourir chaque élément du tableau et de leur appliquer une action. Contrairement aux autres boucles, elle ne retourne pas une nouvelle valeur, mais modifie directement le tableau existant.

Voici un exemple :

```
1  const arr = [1, 2, 3];
2
3  arr.forEach(function(element) {
4    console.log(element);
5  });
6  // Output :
7  // 1
8  // 2
9  // 3
```

## Conclusion

Les boucles et la méthode `map()` sont des outils indispensables en JavaScript. Il est important de bien comprendre leur fonctionnement pour pouvoir manipuler les tableaux



efficacement. **N'hésitez pas à pratiquer régulièrement pour vous exercer et améliorer vos compétences en JavaScript.**



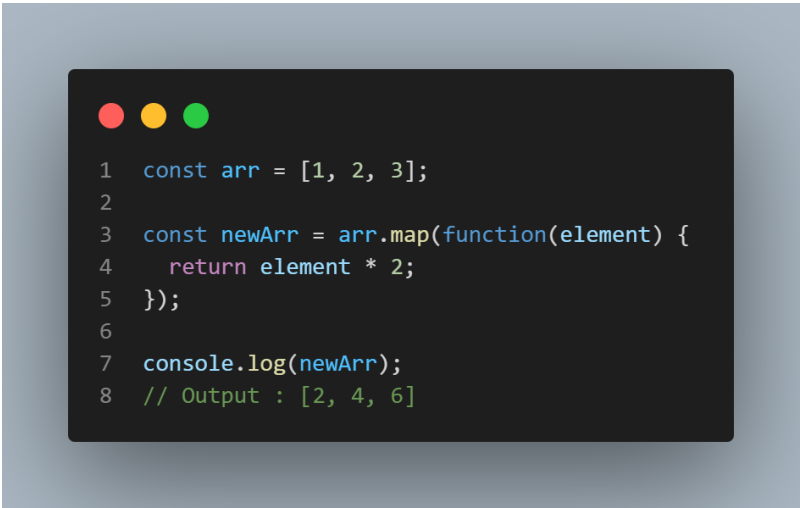
## Exercices

1. Écrire une boucle for qui affiche chaque lettre d'une chaîne de caractères.
2. Écrire une boucle for...of qui calcule la somme des éléments d'un tableau de nombres.
3. Écrire une boucle for...in qui affiche chaque propriété d'un objet.
4. Utiliser la méthode forEach pour ajouter 10 à chaque élément d'un tableau de nombres.

Maintenant, passons à la méthode map.

Rappel: La méthode map() est une méthode de tableau qui crée un nouveau tableau avec les résultats d'une fonction appelée sur chaque élément du tableau d'origine. Elle retourne toujours un tableau de la même longueur que l'original, avec des éléments transformés par la fonction.

Voici un exemple :



```
1  const arr = [1, 2, 3];
2
3  const newArr = arr.map(function(element) {
4    return element * 2;
5  });
6
7  console.log(newArr);
8  // Output : [2, 4, 6]
```

1. Écrire une fonction qui prend un tableau de nombres en entrée et renvoie un nouveau tableau avec les éléments multipliés par 10.
2. Écrire une fonction qui prend un tableau de chaînes de caractères en entrée et renvoie un nouveau tableau avec les chaînes de caractères en majuscules.
3. Écrire une fonction qui prend un tableau d'objets en entrée et renvoie un nouveau tableau avec les propriétés "name" de chaque objet.
4. Écrire une fonction qui prend un tableau d'objets en entrée et renvoie un nouveau tableau avec les objets filtrés selon une propriété. Ex: nous filtrons tous les objets qui ont une propriété age dont la valeur est supérieure à 30.

## Exercice 1 : Boucles

- Créez un tableau contenant les nombres 1 à 5.
- Utilisez une boucle for pour afficher chaque élément du tableau.
- Utilisez une boucle for...in pour afficher chaque index du tableau et sa valeur.
- Utilisez une boucle for...of pour afficher chaque valeur du tableau.
- Utilisez la méthode forEach() pour afficher chaque valeur du tableau.

## Exercice 2 : Méthode map

- Créez un tableau contenant les nombres 1 à 5.

- 
- Utilisez la méthode `map()` pour créer un nouveau tableau contenant chaque élément multiplié par 2.
  - Utilisez la méthode `map()` pour créer un nouveau tableau contenant chaque élément converti en chaîne de caractères. (ex: 5 doit devenir "5")
  - Utilisez la méthode `map()` pour créer un nouveau tableau contenant chaque élément au carré.
  - Utilisez la méthode `map()` pour créer un nouveau tableau contenant chaque élément inversé (ex: 1 devient 1/1, 2 devient 1/2, etc.).