

LES TABLEAUX

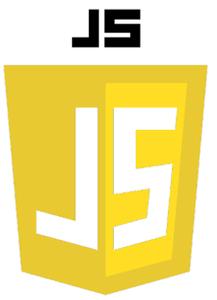
Un tableau (array) est une structure de données qui peut contenir plus d'une valeur à la fois. Ces valeurs peuvent être de n'importe quel type : nombres, chaînes de caractères, objets, autres tableaux, etc.

Création d'un tableau :

Vous pouvez créer un tableau en JavaScript de deux façons principales :

- En utilisant la notation littérale avec des crochets [] :

```
let tableau = []; // Création d'un tableau vide
let nombres = [1, 2, 3, 4, 5]; // Création d'un tableau avec des nombres
```



- En utilisant le constructeur Array :



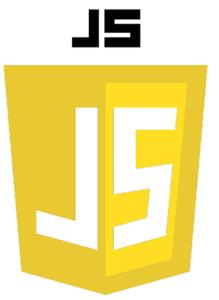
```
let tableau = new Array(); // Création d'un tableau vide
let nombres = new Array(1, 2, 3, 4, 5); // Création d'un tableau avec des nombres
```

Accès aux éléments d'un tableau :

Vous pouvez accéder à un élément d'un tableau en utilisant son index (les indices commencent à 0 en JavaScript) :



```
let fruits = ["pomme", "banane", "cerise"];
console.log(fruits[0]); // Affiche "pomme"
console.log(fruits[1]); // Affiche "banane"
console.log(fruits[2]); // Affiche "cerise"
```



Modification d'un tableau :

Vous pouvez modifier le contenu d'un tableau à un indice spécifique :



```
let fruits = ["pomme", "banane", "cerise"];  
fruits[0] = "kiwi";  
console.log(fruits[0]); // Affiche "kiwi"
```

Propriété 'length' :

La propriété 'length' donne la taille d'un tableau :



```
let fruits = ["pomme", "banane", "cerise"];  
console.log(fruits.length); // Affiche 3
```



Méthodes de tableau :

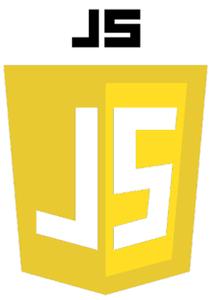
JavaScript offre de nombreuses méthodes pour travailler avec des tableaux, comme `push()` (pour ajouter des éléments à la fin), `pop()` (pour supprimer le dernier élément), `shift()` (pour supprimer le premier élément), `unshift()` (pour ajouter des éléments au début), `join()` (pour convertir un tableau en chaîne de caractères), `splice()` (pour ajouter/supprimer des éléments à un indice spécifique), `slice()` (pour créer un nouveau tableau à partir d'une section), `sort()` (pour trier les éléments), et bien d'autres.

push() :

La méthode `push()` ajoute un ou plusieurs éléments à la fin d'un tableau et renvoie la nouvelle longueur du tableau.



```
let fruits = ["pomme", "banane"];
fruits.push("cerise");
console.log(fruits); // Affiche ["pomme", "banane", "cerise"]
```



MÉTHODES DE TABLEAU (SUITES)

pop() :

La méthode pop() supprime le dernier élément d'un tableau et le retourne.



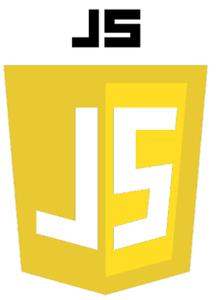
```
let fruits = ["pomme", "banane", "cerise"];  
let dernierFruit = fruits.pop();  
console.log(dernierFruit); // Affiche "cerise"  
console.log(fruits); // Affiche ["pomme", "banane"]
```

shift() :

La méthode shift() supprime le premier élément d'un tableau et le retourne.



```
let fruits = ["pomme", "banane", "cerise"];  
let premierFruit = fruits.shift();  
console.log(premierFruit); // Affiche "pomme"  
console.log(fruits); // Affiche ["banane", "cerise"]
```



MÉTHODES DE TABLEAU (SUITES)

unshift() :

La méthode unshift() ajoute un ou plusieurs éléments au début d'un tableau et renvoie la nouvelle longueur du tableau.



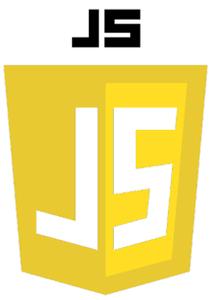
```
let fruits = ["banane", "cerise"];
fruits.unshift("pomme");
console.log(fruits); // Affiche ["pomme", "banane", "cerise"]
```

splice() :

La méthode splice() change le contenu d'un tableau en supprimant, remplaçant ou ajoutant des éléments.



```
let fruits = ["pomme", "banane", "cerise"];
fruits.splice(1, 0, "kiwi"); // Insère "kiwi" à l'indice 1 sans
supprimer d'éléments
console.log(fruits); // Affiche ["pomme", "kiwi", "banane", "cerise"]
```



MÉTHODES DE TABLEAU (SUITES)

slice():

La méthode slice() renvoie une copie superficielle d'une portion de tableau.



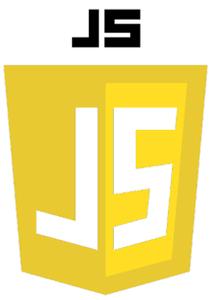
```
let fruits = ["pomme", "kiwi", "banane", "cerise"];  
let certainsFruits = fruits.slice(1, 3);  
console.log(certainsFruits); // Affiche ["kiwi", "banane"]
```

join():

La méthode join() crée et renvoie une nouvelle chaîne en concaténant tous les éléments d'un tableau (ou un objet semblable à un tableau) séparés par des virgules ou une autre chaîne spécifiée.



```
let fruits = ["pomme", "kiwi", "banane", "cerise"];  
let stringFruits = fruits.join(", ");  
console.log(stringFruits); // Affiche "pomme, kiwi, banane, cerise"
```



MÉTHODES DE TABLEAU (SUITES)

sort() :

La méthode `sort()` trie les éléments d'un tableau in-place et renvoie le tableau.



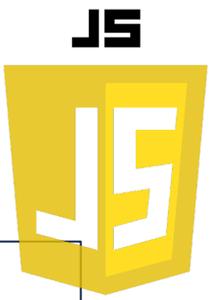
```
let fruits = ["pomme", "kiwi", "banane", "cerise"];  
fruits.sort();  
console.log(fruits); // Affiche ["banane", "cerise", "kiwi", "pomme"]
```

reverse() :

La méthode `reverse()` inverse l'ordre des éléments d'un tableau in-place.



```
let fruits = ["pomme", "kiwi", "banane", "cerise"];  
fruits.reverse();  
console.log(fruits); // Affiche ["cerise", "banane", "kiwi", "pomme"]
```



concat() :

La méthode `concat()` est utilisée pour fusionner deux ou plusieurs tableaux. Cette méthode ne modifie pas les tableaux existants, mais retourne un nouveau tableau.



```
let arr1 = [1, 2, 3];  
let arr2 = [4, 5, 6];  
let arr3 = arr1.concat(arr2);  
console.log(arr3); // Affiche [1, 2, 3, 4, 5, 6]
```

forEach() :

La méthode `forEach()` exécute une fonction donnée sur chaque élément du tableau.



```
let arr = [1, 2, 3, 4, 5];  
arr.forEach(function(element) {  
    console.log(element * 2); // Multiplie chaque élément par 2  
});
```



MÉTHODES DE TABLEAU (SUITES)

map() :

La méthode `map()` crée un nouveau tableau avec les résultats de l'appel d'une fonction pour chaque élément du tableau appelant.

```
●●●
let arr = [1, 2, 3, 4, 5];
let newArr = arr.map(function(element) {
    return element * 2; // Multiplie chaque élément par 2
});
console.log(newArr); // Affiche [2, 4, 6, 8, 10]
```

filter() :

La méthode `filter()` crée un nouveau tableau avec tous les éléments qui passent le test implémenté par la fonction fournie.

```
●●●
let arr = [1, 2, 3, 4, 5];
let newArr = arr.filter(function(element) {
    return element > 2; // Retourne les éléments supérieurs à 2
});
console.log(newArr); // Affiche [3, 4, 5]
```



MÉTHODES DE TABLEAU (SUITES)

reduce() :

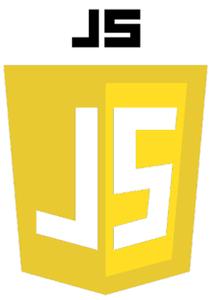
La méthode `reduce()` applique une fonction qui est un "réducteur" aux éléments de chaque tableau (de gauche à droite) afin de les réduire à une seule valeur.

```
●●● let arr = [1, 2, 3, 4, 5];
    let sum = arr.reduce(function(acc, curr) {
        return acc + curr; // Additionne les éléments du tableau
    }, 0);
    console.log(sum); // Affiche 15
```

find() :

La méthode `find()` renvoie la valeur du premier élément dans le tableau qui satisfait la fonction de test fournie. Sinon `undefined` est renvoyé.

```
●●● let arr = [1, 2, 3, 4, 5];
    let first = arr.find(function(element) {
        return element > 2; // Trouve le premier élément supérieur à 2
    });
    console.log(first); // Affiche 3
```



MÉTHODES DE TABLEAU (SUITES)

includes() :

La méthode includes() détermine si un tableau contient une certaine valeur parmi ses entrées, renvoyant true ou false.



```
let arr = [1, 2, 3, 4, 5];  
let isIncluded = arr.includes(2); // Vérifie si le tableau contient 2  
console.log(isIncluded); // Affiche true
```



Boucles et tableaux :

Vous pouvez utiliser des boucles pour parcourir les éléments d'un tableau. Voici un exemple utilisant une boucle for :



```
let fruits = ["pomme", "banane", "cerise"];
for (let i = 0; i < fruits.length; i++) {
    console.log(fruits[i]);
}
```

Ou avec une boucle for...of :



```
let fruits = ["pomme", "banane", "cerise"];
for (let fruit of fruits) {
    console.log(fruit);
}
```