JAVASCRIPT

Tableaux

INTRODUCTION

Les tableaux (ou arrays) sont des structures de données fondamentales en programmation qui permettent de stocker plusieurs valeurs dans une seule variable.

Ils sont particulièrement utiles pour gérer des collections d'éléments, qu'ils soient de types similaires ou différents.

```
let tableauVide = [];
let fruits = ["pomme", "banane", "cerise"];
let tableauMixte = [1, "texte", true, null];
```

ACCÈS AUX ÉLÉMENTS

Les éléments d'un tableau sont accessibles par leur index, qui commence à 0.

```
let tableau = [17, 43, 22, 34, 32];
```

Valeur	17	43	22	34	32
Indice	0	1	2	3	4

```
// Imprimer le deuxième élément du tableau
console.log(tableau[1]); // 43
```

CRÉATION DES TABLEAUX

En JavaScript, les tableaux sont des objets (Array) et peuvent être créés de différentes manières. Bien que l'on puisse utiliser le constructeur Array() avec l'opérateur new, il existe des méthodes plus modernes et concises pour créer des tableaux.

Création d'un Tableau Vide

```
let tableauVide = new Array(); // Création d'un tableau vide
```

Création d'un Tableau avec des Valeurs Initiales

```
let fruits = new Array("Pomme", "Banane", "Orange", "Cerise", "Avocat");
let notes = new Array(12, 15, 12, 17, 14, 18);
let mixte = new Array("Bonjour", 15, 17, "Hello");
```

LA TAILLE D'UN TABLEAU

La taille d'un tableau (le nombre d'éléments qu'il comporte) est obtenu avec la propriété length.

```
let fruits = new Array("Pomme", "Banane", "Orange", "Cerise", "Avocat");
console.log(fruits.length); // 5
```

MODIFICATION DE VALEURS D'UN TABLEAU

Pour modifier la valeur d'un élément d'un tableau, on utilise l'indice de cet élément.

```
let fruits = new Array("Pomme", "Banane", "Orange", "Cerise", "Avocat");
console.log(fruits[1]); // Banane
fruits[1] = "Mangue";
console.log(fruits[1]); // Mangue
```

PARCOURIR UN TABLEAU

Les boucles (spécialement la boucle for) permettent de parcourir un tableau et par suite d'effectuer des opérations sur toutes les éléments du tableau.

```
let fruits = new Array("Pomme", "Banane", "Orange", "Cerise", "Avocat");
for(let i = 0; i < fruits.length; i++){
  console.log(fruits[i]);
}

let notes = new Array(12, 15, 12, 17, 14, 18);
let somme = 0;
for(let i = 0; i < notes.length; i++){
  somme += notes[i];
}
console.log(somme); // 88
```

PARCOURIR UN TABLEAU

Exemple: Augmenter toutes les notes de 10%

```
let notes = new Array(12, 15, 12, 17, 14, 18);
for(let i = 0; i < notes.length; i++){
  notes[i] = notes[i] * 1.1;
}
```

PARCOURIR UN TABLEAU AVEC FOR…OF

La boucle for…of est une boucle spéciale permettant de parcourir un tableau d'une manière plus simple. Il s'agit d'une boucle for avec la différence que l'initialisation, la condition de la boucle et l'incrémentation sont implicitement définies.

```
let notes = new Array(12, 15, 12, 17, 14, 18);
for(let note of notes){
  console.log(note);
}
```

PARCOURIR UN TABLEAU AVEC FOR...IN

Bien que for...in soit principalement conçu pour parcourir les propriétés des objets, il peut également être utilisé pour parcourir les indices des tableaux. Cependant, il n'est pas recommandé d'utiliser for...in pour les tableaux en raison de son comportement particulier et des risques associés, comme le parcours des propriétés héritées.
Pour parcourir les tableaux, la boucle for ou for...of sont généralement préférées.

```
let fruits = ["Pomme", "Banane", "Cerise"];

for (let index in fruits) {
  console.log("Indice " + index + ": " + fruits[index]);
}
```

GÉRER LES ÉLÉMENTS D'UN TABLEAU

JavaScript offre plusieurs méthodes pour ajouter, retirer et manipuler les éléments d'un tableau. Voici un aperçu des principales méthodes : push(), unshift(), pop(), et shift().

Ajouter des Éléments avec **push**(): Ajouter des Éléments à la Fin

```
let fruits = ["Pomme", "Banane"];
fruits.push("Cerise", "Avocat");

console.log(fruits);
// Sortie : ["Pomme", "Banane", "Cerise", "Avocat"]
```

GÉRER LES ÉLÉMENTS D'UN TABLEAU

Ajouter des Éléments au Début avec **unshift**()

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

```
let fruits = ["Banane", "Cerise"];
fruits.unshift("Pomme", "Avocat");

console.log(fruits);
// Sortie : ["Pomme", "Avocat", "Banane", "Cerise"]
```

GÉRER LES ÉLÉMENTS D'UN TABLEAU

Retirer le Dernier Élément avec **pop**()

La méthode pop() retire le dernier élément d'un tableau et le retourne. Le tableau est modifié en place.

```
let fruits = ["Pomme", "Banane", "Cerise"];
let dernierFruit = fruits.pop();

console.log(dernierFruit); // Sortie : "Cerise"
console.log(fruits); // Sortie : ["Pomme", "Banane"]
```

GÉRER LES ÉLÉMENTS D'UN TABLEAU

Retirer le Premier Élément avec **shift**()

La méthode shift() retire le premier élément d'un tableau et le retourne. Le tableau est modifié en place.

```
let fruits = ["Pomme", "Banane", "Cerise"];
let premierFruit = fruits.shift();

console.log(premierFruit); // Sortie : "Pomme"
console.log(fruits); // Sortie : ["Banane", "Cerise"]
```

LA MÉTHODE SPLICE

La méthode **splice**() est une méthode puissante pour manipuler des tableaux en JavaScript. Elle permet de modifier un tableau en ajoutant, supprimant ou remplaçant des éléments à une position spécifiée.

```
let fruits = ["Pomme", "Banane", "Cerise", "Avocat"];
fruits.splice(1, 2); // À partir de l'indice 1, supprimer 2 éléments

console.log(fruits);
// Sortie : ["Pomme", "Avocat"]
```

LES TABLEAUX MULTIDIMENSIONNELS

Les tableaux multidimensionnels sont des tableaux contenant d'autres tableaux comme éléments. Ils sont utilisés pour représenter des structures de données en plusieurs dimensions, comme les matrices en mathématiques. En JavaScript, les tableaux multidimensionnels sont simplement des tableaux imbriqués.

Création d'un Tableau Multidimensionnel

```
let matrice = [
  [1, 2, 3],
  [4, 5, 6],
  [7, 8, 9]
];
```

```
let matrice = new Array(
  new Array(1, 2, 3),
  new Array(4, 5, 6),
  new Array(7, 8, 9),
);
```

ACCÉDER AUX ÉLÉMENTS

```
let matrice = [
  [1, 2, 3],
  [4, 5, 6],
  [7, 8, 9]
];

console.log(matrice[0][1]);
// Affiche 2 (élément de la première ligne, deuxième colonne)
console.log(matrice[2][0]);
// Affiche 7 (élément de la troisième ligne, première colonne)
```

MODIFIER LES ÉLÉMENTS

```
let matrice = [
  [1, 2, 3],
  [4, 5, 6],
  [7, 8, 9]
];

matrice[1][2] = 10;
// Modifie l'élément de la deuxième ligne, troisième colonne

console.log(matrice);
// Sortie : [[1, 2, 3], [4, 5, 10], [7, 8, 9]]
```

PARCOURIR UN TABLEAU MULTIDIME

```
let matrice = [
  [1, 2, 3],
  [4, 5, 6],
  [7, 8, 9]
];

for (let i = 0; i < matrice.length; i++) {
  for (let j = 0; j < matrice[i].length; j++) {
    console.log(matrice[i][j]);
  }
}
```

LES TABLEAUX CLÉS-VALEURS EN JAVASCRIPT

En JavaScript, les tableaux sont généralement indexés par des nombres. Cependant, il est souvent plus pratique d'utiliser des objets pour stocker des paires clé-valeur, où chaque clé est associée à une valeur spécifique.

Bien que JavaScript ne dispose pas de tableaux associatifs comme certains autres langages, vous pouvez utiliser des objets pour stocker des paires clé-valeur. Les objets permettent d'associer des clés (qui sont des chaînes de caractères) à des valeurs.

```
let personne = {
  nom: "Alice",
  âge: 30,
  profession: "Développeuse"
};

// Accès aux valeurs par clé
console.log(personne.nom); // Affiche "Alice"
console.log(personne["âge"]); // Affiche 30
```

PARCOURIR UN OBJET AVEC FOR...IN

La boucle for...in est la méthode principale pour parcourir les propriétés d'un objet en JavaScript. Elle itère sur les clés de l'objet, permettant d'accéder à chaque propriété et à sa valeur.

```
// Création d'un objet avec des clés et des valeurs
let personne = {
  nom: "Alice",
  âge: 30,
  profession: "Développeuse"
};

for (let clé in personne) {
  console.log(clé + ": " + personne[clé]);
  // Affiche chaque clé et sa valeur
}
```