02_Fondamentaux JavaScript

Table des Matières

Qu'est-ce qu'un Algorithme ?

Un algorithme est une série d'étapes précises permettant de résoudre un problème ou de réaliser une tâche. Chaque étape doit être claire et non ambiguë, et l'algorithme doit se terminer après un nombre fini d'étapes.

Les Variables

Qu'est-ce qu'une Variable ?

Une variable est un espace mémoire qui stocke une valeur. Cette valeur peut être modifiée au cours de l'exécution de l'algorithme.

Déclaration et Initialisation

En JavaScript, ils est possible de déclarer une variable avec `var`, `let`, ou `const`.

```
let age = 33;
const prenom = "Damien";
```

- `var` permet de déclarer des variables dont la valeur peut changer mais a une portée (scope) **fonctionnelle** ou **globale**. Si `var` est déclaré dans une fonction, il sera accessible uniquement dans cette fonction. S'il est déclaré en dehors d'une fonction, il devient **global** et est accessible partout dans le code. Mais si `var` est déclaré dans un bloc de code comme une boucle ou une condition, il sera toujours accessible en dehors de ce bloc.
- `let` permet de déclarer des variables dont la valeur peut changer mais a une portée **de bloc**. Cela signifie que les variables ne sont accessibles que dans le bloc `{}` où elles sont définies, comme dans les boucles ou les conditions.
- `const` permet de déclarer des variables dont la valeur ne peut pas changer après initialisation. Pour idem que `let` pour la portée.

Exemple

```
let temperature = 22;
const PI = 3.14159;
```

Les Types de Données

Types de Données en JavaScript

- **Number**: représente des nombres.

```
let age = 30;
let prix = 99.99;
```

- **String**: représente des chaînes de caractères.

```
let prenom = "Damien";
let message = "Bonjour, monde!";
```

- **Boolean**: représente des valeurs vraies ou fausses.

```
let estVrai = true;
let estFaux = false;
```

- **Array**: représente des listes de valeurs.

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

- **Object**: représente des collections de paires clé/valeur.

```
let personne = { prenom: "Damien", age: 33 };
```

Les Opérateurs

Opérateurs Arithmétiques

```
let a = 10;
let b = 5;

console.log(a + b); // Addition
console.log(a - b); // Soustraction
console.log(a * b); // Multiplication
console.log(a / b); // Division
console.log(a % b); // Modulo
```

Opérateurs de Comparaison

```
let a = 10;
let b = 5;

console.log(a == b); // Égalité
console.log(a == b); // Égalité strict (valeur & type)
console.log(a != b); // Inégalité
console.log(a !== b); // Inégalité strict (valeur & type)
console.log(a > b);  // Supérieur à
console.log(a < b);  // Inférieur à
console.log(a >= b); // Supérieur ou égal à
console.log(a <= b); // Inférieur ou égal à
```

Opérateurs Logiques

```
let vrai = true;
let faux = false;

console.log(vrai && faux); // ET logique
console.log(vrai || faux); // OU logique
console.log(!vrai);        // NON logique
```

Les Conditions

Qu'est-ce qu'une Condition ?

Une condition permet d'exécuter différentes parties du code en fonction de certaines conditions.

If...Else

```
let age = 18;

if (age >= 18) {
    console.log("Vous êtes majeur.");
} else {
    console.log("Vous êtes mineur.");
}
```

Switch

```
let fruit = "Pomme";

switch (fruit) {
    case "Pomme":
        console.log("C'est une pomme.");
        break;
    case "Banane":
        console.log("C'est une banane.");
        break;
    default:
        console.log("Fruit inconnu.");
}
```

Les Boucles

Qu'est-ce qu'une Boucle ?

Une boucle permet de répéter une série d'instructions plusieurs fois.

For

La boucle `for` est idéale lorsque vous savez combien de fois vous souhaitez exécuter le bloc de code.

```
for (let i = 0; i < 5; i++) {
    console.log("i est égal à " + i);
}
```

For ... in

La boucle `for...in` est utilisée pour parcourir les **propriétés d'un objet**. Elle permet d'itérer sur les clés d'un objet, mais elle peut aussi être utilisée sur des tableaux (ce qui est déconseillé, car l'ordre des éléments dans les tableaux n'est pas garanti).

```
const personne = { nom: "Damien", age: 33, ville: "Montpellie
r" };
for (let cle in personne) {
    console.log(cle + ": " + personne[cle]);
}
```

For ... of

La boucle `for...of` est utilisée pour parcourir les **valeurs** d'objets itérables tels que les tableaux, les chaînes de caractères, les objets `Set`, les objets `Map`, etc. Contrairement à `for...in` qui itère sur les clés (ou index), `for...of` parcourt les **valeurs** directement.

```
const nombres = [10, 20, 30];
for (let nombre of nombres) {
    console.log(nombre);
}
```

While

La boucle `while` exécute le bloc de code tant que la condition donnée est vraie. Elle est utilisée lorsque vous ne savez pas à l'avance combien d'itérations sont nécessaires, mais que vous voulez continuer à boucler tant qu'une condition est vraie.

```
let i = 0;
while (i < 5) {
    console.log("i est égal à " + i);
    i++;
}
```

Do...While

La boucle `do...while` est similaire à la boucle `while`, à la différence près que le bloc de code est exécuté **au moins une fois**, même si la condition est fausse dès le début. Après la première exécution, la condition est ensuite évaluée pour décider si la boucle doit continuer.

```
let i = 0;
do {
    console.log("i est égal à " + i);
    i++;
} while (i < 5);
```

Les Fonctions

Qu'est-ce qu'une Fonction ?

Une fonction est un bloc de code qui effectue une tâche particulière. Une fonction peut prendre des paramètres et retourner une valeur.

Déclaration et Appel de Fonction

```
function saluer(prenom) {
    return "Bonjour, " + prenom + "!";
}

console.log(saluer("Damien"));
```

Fonctions Anonymes et Fléchées

```
// Fonction anonyme
let saluer = function(prenom) {
    return "Bonjour, " + prenom + "!";
};

// Fonction fléchée
let saluer = (prenom) => "Bonjour, " + prenom + "!";
```

Les Tableaux

Qu'est-ce qu'un Tableau ?

Un tableau est une collection de valeurs. Chaque valeur est accessible via un indice.

Déclaration et Accès aux Éléments

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

console.log(fruits[0]); // Pomme
console.log(fruits[1]); // Banane
console.log(fruits[2]); // Kiwi
```

Boucle sur un Tableau

```javascript
let fruits = ["Pomme", "Banane", "Kiwi"];

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

Les Algorithmes Courants

Recherche Linéaire

```javascript
function recherche(tableau, element) {
    for (let i = 0; i < tableau.length; i++) {
        if (tableau[i] === element) {
            return i;
        }
    }
    return -1;
}

let index = recherche([1, 2, 3, 4, 5], 3);
console.log(index); // 2
```

Tri par Insertion

```javascript
function triInsertion(tableau) {
    for (let i = 1; i < tableau.length; i++) {
        let cle = tableau[i];
        let j = i - 1;
        while (j >= 0 && tableau[j] > cle) {
            tableau[j + 1] = tableau[j];
            j = j - 1;
        }
```

```
            tableau[j + 1] = cle;
        }
        return tableau;
    }

    let tableauTrie = triInsertion([5, 2, 4, 6, 1, 3]);
    console.log(tableauTrie); // [1, 2, 3, 4, 5, 6]
```

Ressources Utiles

- MDN Web Docs
- W3Schools
- JavaScript.info