Guide des Fonctions JavaScript Essentielles

📊 Fonctions pour les Nombres

Math.round()

Description : Arrondit un nombre à l'entier le plus proche.

```javascript
Math.round(4.7); // 5
Math.round(4.3); // 4
Math.round(-4.5); // -4
```

Math.floor()

Description : Arrondit un nombre vers le bas (vers l'entier inférieur).

```javascript
Math.floor(4.9); // 4
Math.floor(-4.1); // -5
```

Math.ceil()

Description : Arrondit un nombre vers le haut (vers l'entier supérieur).

```javascript
Math.ceil(4.1); // 5
Math.ceil(-4.9); // -4
```

Math.max()

Description : Retourne le plus grand nombre parmi les arguments.

```javascript
Math.max(1, 5, 3, 9, 2); // 9
Math.max(...[1, 5, 3, 9, 2]); // 9 (avec spread operator)
```

Math.min()

Description : Retourne le plus petit nombre parmi les arguments.

```javascript
```

```
Math.min(1, 5, 3, 9, 2); // 1
Math.min(...[1, 5, 3, 9, 2]); // 1
```

Math.random()

Description : Génère un nombre aléatoire entre 0 (inclus) et 1 (exclus).

```javascript
Math.random(); // 0.7834592837465
Math.floor(Math.random() * 10); // Nombre entre 0 et 9
Math.floor(Math.random() * 100) + 1; // Nombre entre 1 et 100
```

parseFloat()

Description : Convertit une chaîne en nombre décimal.

```javascript
parseFloat("3.14"); // 3.14
parseFloat("3.14sometext"); // 3.14
parseFloat("text"); // NaN
```

parseInt()

Description : Convertit une chaîne en nombre entier.

```javascript
parseInt("10"); // 10
parseInt("10.5"); // 10
parseInt("10", 2); // 2 (base 2)
parseInt("FF", 16); // 255 (base 16)
```

Number.isNaN()

Description : Vérifie si une valeur est NaN (Not a Number).

```javascript
Number.isNaN(NaN); // true
Number.isNaN(10); // false
Number.isNaN("hello"); // false
```

toFixed()

Description : Formate un nombre avec un nombre spécifique de décimales.

```javascript
let num = 3.14159;
num.toFixed(2); // "3.14"
num.toFixed(0); // "3"
```

🔤 Fonctions pour les Chaînes

charAt()

Description : Retourne le caractère à l'index spécifié.

```javascript
let str = "Hello";
str.charAt(0); // "H"
str.charAt(4); // "o"
```

indexOf()

Description : Retourne l'index de la première occurrence d'une sous-chaîne.

```javascript
let str = "Hello World";
str.indexOf("o"); // 4
str.indexOf("World"); // 6
str.indexOf("xyz"); // -1 (non trouvé)
```

lastIndexOf()

Description : Retourne l'index de la dernière occurrence d'une sous-chaîne.

```javascript
let str = "Hello World Hello";
str.lastIndexOf("Hello"); // 12
str.lastIndexOf("o"); // 15
```

substring()

Description : Extrait une partie d'une chaîne entre deux index.

```javascript
```

```javascript
let str = "Hello World";
str.substring(0, 5); // "Hello"
str.substring(6); // "World"
```

slice()

Description : Extrait une section d'une chaîne (accepte les index négatifs).

```javascript
let str = "Hello World";
str.slice(0, 5); // "Hello"
str.slice(-5); // "World"
str.slice(-5, -1); // "Worl"
```

toLowerCase()

Description : Convertit une chaîne en minuscules.

```javascript
let str = "Hello World";
str.toLowerCase(); // "hello world"
```

toUpperCase()

Description : Convertit une chaîne en majuscules.

```javascript
let str = "Hello World";
str.toUpperCase(); // "HELLO WORLD"
```

trim()

Description : Supprime les espaces en début et fin de chaîne.

```javascript
let str = "  Hello World  ";
str.trim(); // "Hello World"
```

replace()

Description : Remplace une partie d'une chaîne par une autre.

```javascript
```

```javascript
let str = "Hello World";
str.replace("World", "JavaScript"); // "Hello JavaScript"
str.replace(/o/g, "0"); // "Hell0 W0rld" (regex global)
```

split()

Description : Divise une chaîne en tableau selon un délimiteur.

```javascript
let str = "pomme,banane,orange";
str.split(","); // ["pomme", "banane", "orange"]
str.split(""); // ["p", "o", "m", "m", "e", ",", "b", ...]
```

includes()

Description : Vérifie si une chaîne contient une sous-chaîne.

```javascript
let str = "Hello World";
str.includes("World"); // true
str.includes("world"); // false (sensible à la casse)
```

startsWith()

Description : Vérifie si une chaîne commence par une sous-chaîne.

```javascript
let str = "Hello World";
str.startsWith("Hello"); // true
str.startsWith("World"); // false
```

endsWith()

Description : Vérifie si une chaîne se termine par une sous-chaîne.

```javascript
let str = "Hello World";
str.endsWith("World"); // true
str.endsWith("Hello"); // false
```

📋 Fonctions pour les Tableaux

push()

Description : Ajoute un ou plusieurs éléments à la fin du tableau.

```javascript
let arr = [1, 2, 3];
arr.push(4); // [1, 2, 3, 4]
arr.push(5, 6); // [1, 2, 3, 4, 5, 6]
```

pop()

Description : Supprime et retourne le dernier élément du tableau.

```javascript
let arr = [1, 2, 3];
arr.pop(); // 3, arr devient [1, 2]
```

unshift()

Description : Ajoute un ou plusieurs éléments au début du tableau.

```javascript
let arr = [2, 3];
arr.unshift(1); // [1, 2, 3]
arr.unshift(-1, 0); // [-1, 0, 1, 2, 3]
```

shift()

Description : Supprime et retourne le premier élément du tableau.

```javascript
let arr = [1, 2, 3];
arr.shift(); // 1, arr devient [2, 3]
```

slice()

Description : Retourne une copie partielle du tableau.

```javascript
let arr = [1, 2, 3, 4, 5];
arr.slice(1, 3); // [2, 3]
arr.slice(-2); // [4, 5]
```

splice()

Description : Modifie le tableau en supprimant/ajoutant des éléments.

```javascript
let arr = [1, 2, 3, 4, 5];
arr.splice(2, 1); // Supprime 1 élément à l'index 2: [1, 2, 4, 5]
arr.splice(2, 0, 'a', 'b'); // Ajoute 'a' et 'b' à l'index 2: [1, 2, 'a', 'b', 4, 5]
```

concat()

Description : Combine deux ou plusieurs tableaux.

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

join()

Description : Joint tous les éléments du tableau en une chaîne.

```javascript
let arr = ["Hello", "World"];
arr.join(" "); // "Hello World"
arr.join("-"); // "Hello-World"
```

indexOf()

Description : Retourne l'index de la première occurrence d'un élément.

```javascript
let arr = [1, 2, 3, 2, 4];
arr.indexOf(2); // 1
arr.indexOf(5); // -1 (non trouvé)
```

includes()

Description : Vérifie si un tableau contient un élément.

```javascript
```

```javascript
let arr = [1, 2, 3];
arr.includes(2); // true
arr.includes(4); // false
```

reverse()

Description : Inverse l'ordre des éléments du tableau.

```javascript
let arr = [1, 2, 3];
arr.reverse(); // [3, 2, 1]
```

sort()

Description : Trie les éléments du tableau.

```javascript
let arr = [3, 1, 4, 1, 5];
arr.sort(); // [1, 1, 3, 4, 5]
arr.sort((a, b) => b - a); // [5, 4, 3, 1, 1] (ordre décroissant)
```

forEach()

Description : Exécute une fonction pour chaque élément du tableau.

```javascript
let arr = [1, 2, 3];
arr.forEach((element, index) => {
    console.log(`Index ${index}: ${element}`);
});
// Affiche: Index 0: 1, Index 1: 2, Index 2: 3
```

map()

Description : Crée un nouveau tableau avec les résultats d'une fonction appliquée à chaque élément.

```javascript
let arr = [1, 2, 3];
let doubled = arr.map(x => x * 2); // [2, 4, 6]
let strings = arr.map(x => `Number: ${x}`); // ["Number: 1", "Number: 2", "Number: 3"]
```

filter()

Description : Crée un nouveau tableau avec les éléments qui passent un test.

```javascript
let arr = [1, 2, 3, 4, 5];
let evens = arr.filter(x => x % 2 === 0); // [2, 4]
let greaterThan2 = arr.filter(x => x > 2); // [3, 4, 5]
```

reduce()

Description : Réduit le tableau à une seule valeur en appliquant une fonction.

```javascript
let arr = [1, 2, 3, 4];
let sum = arr.reduce((acc, curr) => acc + curr, 0); // 10
let product = arr.reduce((acc, curr) => acc * curr, 1); // 24
```

find()

Description : Retourne le premier élément qui satisfait une condition.

```javascript
let arr = [1, 2, 3, 4, 5];
let found = arr.find(x => x > 3); // 4
let notFound = arr.find(x => x > 10); // undefined
```

some()

Description : Teste si au moins un élément satisfait une condition.

```javascript
let arr = [1, 2, 3, 4, 5];
arr.some(x => x > 3); // true
arr.some(x => x > 10); // false
```

every()

Description : Teste si tous les éléments satisfont une condition.

```javascript
let arr = [2, 4, 6, 8];
arr.every(x => x % 2 === 0); // true (tous pairs)
arr.every(x => x > 5); // false
```

🗂️ Fonctions pour les Objets

Object.keys()

Description : Retourne un tableau des clés énumérables d'un objet.

```javascript
let obj = {name: "John", age: 30, city: "Paris"};
Object.keys(obj); // ["name", "age", "city"]
```

Object.values()

Description : Retourne un tableau des valeurs énumérables d'un objet.

```javascript
let obj = {name: "John", age: 30, city: "Paris"};
Object.values(obj); // ["John", 30, "Paris"]
```

Object.entries()

Description : Retourne un tableau des paires [clé, valeur] d'un objet.

```javascript
let obj = {name: "John", age: 30};
Object.entries(obj); // [["name", "John"], ["age", 30]]
```

Object.assign()

Description : Copie les propriétés d'un ou plusieurs objets sources vers un objet cible.

```javascript
let target = {a: 1};
let source = {b: 2, c: 3};
Object.assign(target, source); // {a: 1, b: 2, c: 3}

// Copie d'objet
let copy = Object.assign({}, obj);
```

hasOwnProperty()

Description : Vérifie si un objet possède une propriété spécifique.

```javascript
```

```javascript
let obj = {name: "John", age: 30};
obj.hasOwnProperty("name"); // true
obj.hasOwnProperty("height"); // false
```

Object.freeze()

Description : Rend un objet immutable (non modifiable).

```javascript
let obj = {name: "John"};
Object.freeze(obj);
obj.name = "Jane"; // Ignoré en mode non-strict, erreur en mode strict
console.log(obj.name); // "John"
```

Object.seal()

Description : Empêche l'ajout ou la suppression de propriétés, mais permet la modification des valeurs existantes.

```javascript
let obj = {name: "John"};
Object.seal(obj);
obj.name = "Jane"; // Autorisé
obj.age = 30; // Ignoré
delete obj.name; // Ignoré
```

Object.create()

Description : Crée un nouvel objet avec le prototype spécifié.

```javascript
let prototype = {greet: function() { return "Hello!"; }};
let obj = Object.create(prototype);
obj.greet(); // "Hello!"
```

JSON.stringify()

Description : Convertit un objet JavaScript en chaîne JSON.

```javascript
```

```javascript
let obj = {name: "John", age: 30};
JSON.stringify(obj); // '{"name":"John","age":30}'
JSON.stringify(obj, null, 2); // Format indenté
```

JSON.parse()

Description : Convertit une chaîne JSON en objet JavaScript.

```javascript
let jsonString = '{"name":"John","age":30}';
let obj = JSON.parse(jsonString); // {name: "John", age: 30}
```

delete

Description : Supprime une propriété d'un objet.

```javascript
let obj = {name: "John", age: 30};
delete obj.age;
console.log(obj); // {name: "John"}
```

in

Description : Vérifie si une propriété existe dans un objet (y compris dans la chaîne de prototypes).

```javascript
let obj = {name: "John", age: 30};
"name" in obj; // true
"height" in obj; // false
"toString" in obj; // true (hérité du prototype Object)
```

🔗 Méthodes de Chaînage Courantes

Exemple avec les tableaux

```javascript
let nombres = [1, 2, 3, 4, 5, 6];
let resultat = nombres
  .filter(n => n % 2 === 0)  // [2, 4, 6]
  .map(n => n * 2)           // [4, 8, 12]
  .reduce((sum, n) => sum + n, 0); // 24
```