

# Dictionnaire JavaScript Complet

## Guide de Référence pour Apprenants

Real Conseil - Formation DWWM

# Introduction

Ce dictionnaire JavaScript contient plus de 150 termes essentiels expliqués en détail. Chaque terme comprend :

- Une définition complète et claire
- La syntaxe exacte
- Des exemples pratiques commentés
- Des explications sur ce que ça fait concrètement
- Des notes importantes quand nécessaire

Ce document est conçu pour les apprenants de la formation DWWM de Real Conseil.

# A

## addEventListener()

Méthode DOM

**Définition :** Attache un gestionnaire d'événement à un élément HTML. Écoute un événement spécifique et exécute une fonction.

**Syntaxe :**

```
element.addEventListener(type, fonction);
```

**Exemple :**

```
bouton.addEventListener("click", () => { alert("Cliqué!"); });
```

## alert()

Fonction globale

**Définition :** Affiche une fenêtre popup avec un message. Bloque l'exécution jusqu'au clic sur OK.

**Syntaxe :**

```
alert(message);
```

**Exemple :**

```
alert("Bienvenue!");
```

## Array

Type de données

**Définition :** Structure qui stocke plusieurs valeurs dans une liste ordonnée. Index commence à 0.

**Syntaxe :**

```
let tab = [val1, val2];
```

**Exemple :**

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

## async/await

Mots-clés ES8

**Définition :** Syntaxe moderne pour gérer du code asynchrone. Plus lisible que les Promises.

**Syntaxe :**

```
async function nom() { await promesse; }
```

**Exemple :**

```
async function getData() {  
  let r = await fetch(url);  
  let data = await r.json();  
}
```

## appendChild()

Méthode DOM

**Définition :** Ajoute un élément enfant à un parent dans le DOM.

**Syntaxe :**

```
parent.appendChild(enfant);
```

**Exemple :**

```
let div = document.createElement("div");  
document.body.appendChild(div);
```

# B

## Boolean

Type primitif

**Définition :** Type qui ne peut être que true ou false. Utilisé pour la logique et les conditions.

**Syntaxe :**

```
let variable = true;
```

**Exemple :**

```
let majeur = age >= 18;  
if (majeur) { console.log("OK"); }
```

## break

Mot-clé

**Définition :** Sort immédiatement d'une boucle ou d'un switch.

**Syntaxe :**

```
break;
```

**Exemple :**

```
for (let i = 0; i < 10; i++) {  
  if (i === 5) break;  
  console.log(i);  
}
```

# C

## console.log()

### Méthode

**Définition :** Affiche des messages dans la console du navigateur (F12). Essentiel pour déboguer.

**Syntaxe :**

```
console.log(valeur);
```

**Exemple :**

```
console.log("Hello");  
console.log({nom: "Alice"});
```

## const

### Mot-clé

**Définition :** Déclare une constante non réassignable. Portée de bloc.

**Syntaxe :**

```
const nom = valeur;
```

**Exemple :**

```
const PI = 3.14;  
// PI = 3; // ERREUR
```

## continue

### Mot-clé

**Définition :** Passe à l'itération suivante d'une boucle sans sortir.

**Syntaxe :**

```
continue;
```

**Exemple :**

```
for (let i = 0; i < 5; i++) {  
  if (i === 2) continue;  
  console.log(i); // 0,1,3,4  
}
```

## classList

Propriété DOM

**Définition :** Gère les classes CSS d'un élément (add, remove, toggle, contains).

**Syntaxe :**

```
element.classList.add("classe");
```

**Exemple :**

```
div.classList.add("active");
```

```
div.classList.remove("hidden");
```

```
div.classList.toggle("visible");
```

# D

## document

Objet DOM

**Définition :** Représente la page HTML. Point d'entrée pour accéder et manipuler les éléments.

**Syntaxe :**

```
document.methode()
```

**Exemple :**

```
let elem = document.getElementById("id");  
document.title = "Nouveau titre";
```

## do...while

Boucle

**Définition :** Boucle qui s'exécute au moins une fois avant de tester la condition.

**Syntaxe :**

```
do { code } while (condition);
```

**Exemple :**

```
let i = 0;  
do {  
  console.log(i++);  
} while (i < 3);
```



# E

## else

### Mot-clé

**Définition :** Exécute du code si la condition du if est fausse.

**Syntaxe :**

```
if (cond) {...} else {...}
```

**Exemple :**

```
if (age >= 18) {  
  console.log("Majeur");  
} else {  
  console.log("Mineur");  
}
```

## event

### Objet

**Définition :** Contient les informations sur un événement (clic, touche, etc.).

**Syntaxe :**

```
function(event) {...}
```

**Exemple :**

```
btn.addEventListener("click", (e) => {  
  console.log(e.target);  
});
```

# F

## fetch()

Fonction API

**Définition :** Fait des requêtes HTTP pour récupérer/envoyer des données. Retourne une Promise.

**Syntaxe :**

```
fetch(url).then(r => r.json())
```

**Exemple :**

```
fetch("api/users")  
.then(r => r.json())  
.then(data => console.log(data));
```

## filter()

Méthode Array

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

**Syntaxe :**

```
tableau.filter(fonction)
```

**Exemple :**

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

## for

Boucle

**Définition :** Boucle avec compteur. Idéale quand on connaît le nombre d'itérations.

**Syntaxe :**

```
for (init; cond; incr) {...}
```

**Exemple :**

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

## forEach()

Méthode Array

**Définition :** Exécute une fonction pour chaque élément d'un tableau.

**Syntaxe :**

```
tableau.forEach(fonction)
```

**Exemple :**

```
fruits.forEach(f => console.log(f));
```

## function

Mot-clé

**Définition :** Déclare une fonction réutilisable.

**Syntaxe :**

```
function nom(params) {...}
```

**Exemple :**

```
function saluer(nom) {  
  return "Bonjour " + nom;  
}
```



## getElementById()

Méthode DOM

**Définition :** Sélectionne un élément HTML par son ID.

**Syntaxe :**

```
document.getElementById(id)
```

**Exemple :**

```
let elem = document.getElementById("monDiv");  
elem.style.color = "red";
```

## getElementsByClassName()

Méthode DOM

**Définition :** Sélectionne tous les éléments avec une classe donnée. Retourne une HTMLCollection.

**Syntaxe :**

```
document.getElementsByClassName(classe)
```

**Exemple :**

```
let elems = document.getElementsByClassName("maClasse");  
elems[0].textContent = "Modifié";
```

## if

### Mot-clé

**Définition :** Exécute du code si une condition est vraie.

**Syntaxe :**

```
if (condition) {...}
```

**Exemple :**

```
if (age >= 18) {  
  console.log("Majeur");  
}
```

## innerHTML

### Propriété DOM

**Définition :** Récupère ou modifie le contenu HTML d'un élément.

**Syntaxe :**

```
element.innerHTML = "html"
```

**Exemple :**

```
div.innerHTML = "Gras";
```

## includes()

### Méthode Array/String

**Définition :** Vérifie si un élément/texte existe. Retourne true/false.

**Syntaxe :**

```
tableau.includes(element)
```

**Exemple :**

```
let fruits = ["pomme", "banane"];  
fruits.includes("pomme"); // true  
"hello".includes("ell"); // true
```

# indexOf()

Méthode Array/String

**Définition :** Retourne l'index d'un élément. -1 si non trouvé.

**Syntaxe :**

`tableau.indexOf(element)`

**Exemple :**

```
let index = [1,2,3].indexOf(2); // 1
```

```
"hello".indexOf("l"); // 2
```

# J

## join()

Méthode Array

**Définition :** Combine tous les éléments d'un tableau en une chaîne.

**Syntaxe :**

```
tableau.join(separateur)
```

**Exemple :**

```
let fruits = ["pomme", "banane"];  
fruits.join(", "); // "pomme, banane"
```

## JSON.parse()

Méthode JSON

**Définition :** Convertit une chaîne JSON en objet JavaScript.

**Syntaxe :**

```
JSON.parse(chaine)
```

**Exemple :**

```
let obj = JSON.parse('{"nom":"Alice"}');  
console.log(obj.nom); // "Alice"
```

## JSON.stringify()

Méthode JSON

**Définition :** Convertit un objet JavaScript en chaîne JSON.

**Syntaxe :**

```
JSON.stringify(objet)
```

**Exemple :**

```
let json = JSON.stringify({nom: "Alice"});  
// '{"nom":"Alice"}'
```



## length

### Propriété

**Définition :** Retourne la longueur d'un tableau ou d'une chaîne.

**Syntaxe :**

tableau.length ou chaine.length

**Exemple :**

```
let fruits = ["a", "b", "c"];  
console.log(fruits.length); // 3  
"hello".length; // 5
```

## let

### Mot-clé

**Définition :** Déclare une variable modifiable. Portée de bloc.

**Syntaxe :**

```
let nom = valeur;
```

**Exemple :**

```
let age = 25;  
age = 26; // OK
```

## localStorage

### Objet

**Définition :** Stocke des données dans le navigateur de façon permanente.

**Syntaxe :**

```
localStorage.setItem(cle, valeur)
```

**Exemple :**

```
localStorage.setItem("nom", "Alice");  
let nom = localStorage.getItem("nom");
```



# M

## map()

Méthode Array

**Définition :** Crée un nouveau tableau en transformant chaque élément.

**Syntaxe :**

```
tableau.map(fonction)
```

**Exemple :**

```
let doubles = [1,2,3].map(x => x * 2);  
// [2, 4, 6]
```

## Math

Objet

**Définition :** Fournit des fonctions mathématiques.

**Syntaxe :**

```
Math.methode()
```

**Exemple :**

```
Math.round(3.7); // 4  
Math.random(); // 0.xxx  
Math.max(5, 10); // 10
```

## match()

Méthode String

**Définition :** Recherche une correspondance avec une regex. Retourne un tableau.

**Syntaxe :**

```
chaine.match(regex)
```

**Exemple :**

```
let resultat = "hello123".match(/\d+/);  
// ["123"]
```

# N

## new

Opérateur

**Définition :** Crée une nouvelle instance d'un objet.

**Syntaxe :**

```
new Constructeur()
```

**Exemple :**

```
let date = new Date();  
let arr = new Array(5);
```

## null

Valeur

**Définition :** Représente l'absence intentionnelle de valeur.

**Syntaxe :**

```
let variable = null;
```

**Exemple :**

```
let user = null; // Pas d'utilisateur
```

## Number()

Fonction

**Définition :** Convertit une valeur en nombre.

**Syntaxe :**

```
Number(valeur)
```

**Exemple :**

```
Number("42"); // 42  
Number("3.14"); // 3.14  
Number("abc"); // NaN
```



## Object

Type

**Définition :** Type de données pour stocker des paires clé-valeur.

**Syntaxe :**

```
let obj = {cle: valeur};
```

**Exemple :**

```
let user = {nom: "Alice", age: 25};  
console.log(user.nom); // "Alice"
```

## Object.keys()

Méthode statique

**Définition :** Retourne un tableau des clés d'un objet.

**Syntaxe :**

```
Object.keys(objet)
```

**Exemple :**

```
let obj = {a: 1, b: 2};  
Object.keys(obj); // ["a", "b"]
```

## Object.values()

Méthode statique

**Définition :** Retourne un tableau des valeurs d'un objet.

**Syntaxe :**

```
Object.values(objet)
```

**Exemple :**

```
let obj = {a: 1, b: 2};  
Object.values(obj); // [1, 2]
```

# P

## parseInt()

Fonction

**Définition :** Convertit une chaîne en nombre entier.

**Syntaxe :**

```
parseInt(chaine, base)
```

**Exemple :**

```
parseInt("42"); // 42  
parseInt("3.99"); // 3  
parseInt("10", 2); // 2 (binaire)
```

## parseFloat()

Fonction

**Définition :** Convertit une chaîne en nombre décimal.

**Syntaxe :**

```
parseFloat(chaine)
```

**Exemple :**

```
parseFloat("3.14"); // 3.14  
parseFloat("2.5e3"); // 2500
```

## pop()

Méthode Array

**Définition :** Supprime et retourne le dernier élément.

**Syntaxe :**

```
tableau.pop()
```

**Exemple :**

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

## push()

Méthode Array

**Définition :** Ajoute un/des élément(s) à la fin.

**Syntaxe :**

```
tableau.push(element)
```

**Exemple :**

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

## Promise

Objet

**Définition :** Représente une valeur qui sera disponible plus tard.

**Syntaxe :**

```
new Promise((resolve, reject) => {...})
```

**Exemple :**

```
let p = new Promise((resolve) => {  
  setTimeout(() => resolve("OK"), 1000);  
});
```

## preventDefault()

Méthode Event

**Définition :** Empêche le comportement par défaut d'un événement.

**Syntaxe :**

```
event.preventDefault()
```

**Exemple :**

```
form.addEventListener("submit", (e) => {  
  e.preventDefault();  
});
```



## querySelector()

Méthode DOM

**Définition :** Sélectionne le premier élément correspondant au sélecteur CSS.

**Syntaxe :**

```
document.querySelector(selecteur)
```

**Exemple :**

```
let elem = document.querySelector(".maClasse");  
let btn = document.querySelector("#monBtn");
```

## querySelectorAll()

Méthode DOM

**Définition :** Sélectionne tous les éléments correspondant au sélecteur.

**Syntaxe :**

```
document.querySelectorAll(selecteur)
```

**Exemple :**

```
let paragraphes = document.querySelectorAll("p");  
paragraphes.forEach(p => console.log(p));
```

# R

## reduce()

Méthode Array

**Définition :** Réduit un tableau à une seule valeur en appliquant une fonction.

**Syntaxe :**

```
tableau.reduce((acc, val) => {...}, init)
```

**Exemple :**

```
let somme = [1,2,3,4].reduce((acc, val) => acc + val, 0);  
// 10
```

## return

Mot-clé

**Définition :** Renvoie une valeur depuis une fonction et termine son exécution.

**Syntaxe :**

```
return valeur;
```

**Exemple :**

```
function additionner(a, b) {  
  return a + b;  
}  
  
let r = additionner(5, 3); // 8
```

## removeChild()

Méthode DOM

**Définition :** Supprime un élément enfant.

**Syntaxe :**

```
parent.removeChild(enfant)
```

**Exemple :**

```
let enfant = document.querySelector("#elem");  
enfant.parentNode.removeChild(enfant);
```

## replace()

Méthode String

**Définition :** Remplace du texte dans une chaîne.

**Syntaxe :**

```
chaîne.replace(recherche, remplacement)
```

**Exemple :**

```
"hello".replace("h", "H"); // "Hello"
```

```
"hello".replace(/l/g, "L"); // "heLLo"
```



# S

## setInterval()

Fonction

**Définition :** Exécute une fonction de façon répétée à intervalle régulier.

**Syntaxe :**

```
setInterval(fonction, delai)
```

**Exemple :**

```
let id = setInterval(() => {  
  console.log("Tick");  
}, 1000);  
// clearInterval(id) pour arrêter
```

## setTimeout()

Fonction

**Définition :** Exécute une fonction après un délai (en millisecondes).

**Syntaxe :**

```
setTimeout(fonction, delai)
```

**Exemple :**

```
setTimeout(() => {  
  console.log("Après 3s");  
}, 3000);
```

## shift()

Méthode Array

**Définition :** Supprime et retourne le premier élément.

**Syntaxe :**

```
tableau.shift()
```

**Exemple :**

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



Cofinancé par  
l'Union européenne

```
// arr = [2, 3]
```

## slice()

Méthode Array/String

**Définition :** Extrait une portion sans modifier l'original.

**Syntaxe :**

```
tableau.slice(debut, fin)
```

**Exemple :**

```
let arr = [1,2,3,4,5];  
arr.slice(1, 3); // [2, 3]  
"hello".slice(1, 4); // "ell"
```

## sort()

Méthode Array

**Définition :** Trie les éléments du tableau.

**Syntaxe :**

```
tableau.sort(fonction)
```

**Exemple :**

```
let nombres = [10, 5, 40, 25];  
nombres.sort((a, b) => a - b);  
// [5, 10, 25, 40]
```

## split()

Méthode String

**Définition :** Divise une chaîne en tableau.

**Syntaxe :**

```
chaine.split(separateur)
```

**Exemple :**

```
"a,b,c".split(","); // ["a", "b", "c"]  
"hello".split(""); // ["h","e","l","l","o"]
```

# String

## Type primitif

**Définition :** Type pour les chaînes de caractères (texte).

**Syntaxe :**

```
let chaine = "texte";
```

**Exemple :**

```
let nom = "Alice";  
let msg = `Bonjour ${nom}`;
```

# switch

## Structure

**Définition :** Évalue une expression et exécute le code du cas correspondant.

**Syntaxe :**

```
switch (expr) { case val: ... }
```

**Exemple :**

```
switch (jour) {  
  case "lundi":  
    console.log("Début");  
    break;  
  default:  
    console.log("Autre");  
}
```

# T

## textContent

Propriété DOM

**Définition :** Récupère ou modifie le texte d'un élément.

**Syntaxe :**

```
element.textContent = "texte"
```

**Exemple :**

```
let div = document.querySelector("#div");  
div.textContent = "Nouveau texte";
```

## then()

Méthode Promise

**Définition :** Exécute du code quand une Promise est résolue.

**Syntaxe :**

```
promise.then(fonction)
```

**Exemple :**

```
fetch(url)  
.then(r => r.json())  
.then(data => console.log(data));
```

## toString()

Méthode

**Définition :** Convertit une valeur en chaîne de caractères.

**Syntaxe :**

```
valeur.toString()
```

**Exemple :**

```
let num = 42;  
num.toString(); // "42"  
[1,2,3].toString(); // "1,2,3"
```

## trim()

Méthode String

**Définition :** Enlève les espaces au début et à la fin.

**Syntaxe :**

```
chaine.trim()
```

**Exemple :**

```
let text = " hello ";  
text.trim(); // "hello"
```

## try...catch

Structure

**Définition :** Gère les erreurs sans bloquer le code.

**Syntaxe :**

```
try {...} catch (e) {...}
```

**Exemple :**

```
try {  
  riskyOperation();  
} catch (error) {  
  console.error("Erreur:", error);  
}
```

## typeof

Opérateur

**Définition :** Retourne le type d'une variable.

**Syntaxe :**

```
typeof variable
```

**Exemple :**

```
typeof "hello"; // "string"  
typeof 42; // "number"  
typeof true; // "boolean"
```



Cofinancé par  
l'Union européenne

# U

## undefined

### Valeur

**Définition :** Valeur d'une variable déclarée mais non assignée.

**Syntaxe :**

```
let x; // x est undefined
```

**Exemple :**

```
let x;  
console.log(x); // undefined  
let obj = {};  
console.log(obj.prop); // undefined
```

## unshift()

### Méthode Array

**Définition :** Ajoute un/des élément(s) au début.

**Syntaxe :**

```
tableau.unshift(element)
```

**Exemple :**

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



Cofinancé par  
l'Union européenne



## var

Mot-clé (ancien)

**Définition :** Ancienne façon de déclarer une variable. À éviter.

**Syntaxe :**

```
var nom = valeur;
```

**Exemple :**

```
var age = 25; // Préférer let ou const
```

## value

Propriété DOM

**Définition :** Récupère ou modifie la valeur d'un input.

**Syntaxe :**

```
input.value
```

**Exemple :**

```
let input = document.querySelector("#email");  
console.log(input.value);
```





## while

Boucle

**Définition :** Répète un code tant qu'une condition est vraie.

**Syntaxe :**

```
while (condition) {...}
```

**Exemple :**

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