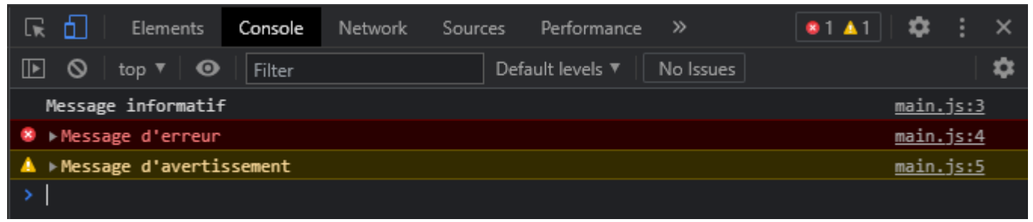
MÉMO JAVASCRIPT

EXÉCUTER DU JAVASCRIPT

```html
<!doctype html>
<html lang="fr">
<head>
    <title>Mémo Javascript</title>

    <!-- 3 méthodes au choix : -->

    <!-- Dans la balise <head> : -->
    <!-- Fichier exécuté PENDANT le chargement de la page -->
    <script src="main.js"></script>
</head>
<body>
    <!-- Juste avant la balise fermante </body> : -->
    <!-- Fichier exécuté APRES le chargement de la page -->
    <script src="main.js"></script>

    <!-- Dans les balises <script> : -->
    <!-- Code exécuté APRES le chargement de la page -->
    <script>
        console.log('Ceci est du code JavaScript');
    </script>
</body>
</html>
```

LIRE DU JAVASCRIPT DANS LA CONSOLE

```javascript
// Ces messages s'afficheront dans la console :

console.log('Message informatif');
console.error('Message d\'erreur');
console.warn('Message d\'avertissement');
```



DÉCLARER UNE VARIABLE OU UNE CONSTANTE

```
let myVariable = 'Hello World!';
const myConst = 'Hello World!';
```

ÉCRIRE DES COMMENTAIRES

```
// Commentaire sur une ligne

/*
Commentaire sur
plusieurs
lignes
*/
```

FONCTIONS NORMALES ET FLÉCHÉES

```
// Déclaration de la fonction
function addition(numberA, numberB) {
    return numberA + numberB;
}

// Affichage du résultat dans la console
console.log(addition(42, 128));
```

```
// Déclaration d'une fonction fléchée :
const addition = (numberA, numberB) => {
    return numberA + numberB;
}

// Qui est équivalente à :
const addition = (numberA, numberB) => numberA + numberB;
```

CONDITIONS

```
const age = 42;

// Condition en if...else if...else
if (age <= 18) {
    console.log('You are young');
} else if (age < 60) {
    console.log('You are an adult');
} else {
    console.log('You are old');
}

// Condition en switch
switch (age) {
    case 18:
        console.log('You are 18');
        break;
    case 42:
        console.log('You are 42');
        break;
    default:
        console.log('You are neither 18 nor 42');
}
```

BOUCLES

```
let age = 18;

// Boucle while
while (age < 30) {
    console.log('You are too young');
    age++;
}

// Boucle for
for (let i = 0; i < 42; i++) {
    console.log('Let\s repeat this sentence 42 times!');
}

// Boucle do...while
do {
    // Cette phrase sera affichée au moins une fois
    console.log('This sentence will be displayed');
    age++;
} while (age < 30);
```

TABLEAUX

```
// Déclarer un tableau :
const myArray = ['Lyon', 42, false];
// Autre manière :
const mySecondArray = new Array('Lyon', 42, false);

// Afficher "Lyon" dans la console :
console.log(myArray[0]);
```

GESTION DES ERREURS

```
try {
    // Code susceptible de provoquer une erreur
} catch(error) {
    console.log(error); // Affiche le contenu de l'erreur
}
```

OBJETS LITTÉRAUX

```
const myLiteralObject = {
    name: 'Toto',
    age: 42,
    administrator: false
}

// Afficher "Toto" dans la console :
console.log(myLiteralObject.name);
// Autre manière :
console.log(myLiteralObject['name']);
```

OPÉRATEUR TERNAIRE

```
const age = 42;

// Affiche "Plus de 40"
console.log(age <= 40 ? 'Moins de 40' : 'Plus de 40');
```

CLASSES

```
class User {
    #age; // Champ privé (avec le symbole #)
    name; // Champ publique (sans le symbole #)
    ville = 'Montpellier'; // Valeur par défaut

    constructor(name, age) {
        this.name = name;
        this.#age = age;
    }

    displayName() {
        // this correspond à l'instance de la classe
        console.log('Je m\'appelle ' + this.name);
    }

    // Une méthode statique s'appelle sur la classe elle-même
    static displayType() {
        console.log('Je suis un utilisateur');
    }
}

const toto = new User('Toto', 42);
toto.displayName(); // Affiche "Je m'appelle Toto"
User.displayType(); // Affiche "Je suis un utilisateur"
```

HÉRITAGE DE CLASSES

```
class User {
    constructor(name) {
        this.name = name;
    }
}

class Admin extends User {
    constructor(name, role) {
        // "super()" appelle le constructeur de la classe parente

        // super.maMethode() appellerait la méthode "maMéthode()"
        // de la classe parente
        super(name);
        this.role = role;
    }
}

const toto = new Admin('Toto', 'Développeur');
console.log(toto.role); // Affiche "Développeur"
```

PROMISES (ASYNCHRONE)

```
function myAsyncFunction(isOk) {
    return new Promise((resolve, reject) => {
        // setTimeout() exécutera le code 1 seconde
        // après avoir été lancé
        setTimeout(() => {
            if (isOk) {
                console.log('Resolve');
                resolve();
            } else {
                reject('Une erreur est survenue');
            }
        }, 1000);
    });
}

// Affiche "Resolve" après 1 seconde
myAsyncFunction(true).then();

// Affiche "Une erreur est survenue" après 1 seconde
myAsyncFunction(false).catch(errorMsg => {
    console.log(errorMsg);
});
```

CONCATÉNATION

```
let name = 'Toto';

// Concaténation basique :
// Affiche "Je m'appelle Toto"
console.log('Je m\'appelle ' + name);

// Version "template strings" :
// Affiche également "Je m'appelle Toto"
console.log(`Je m'appelle ${name}`);
```

OPÉRATIONS RACCOURCIES

```
let age = 42;
age++; // age = age + 1 : 43
age--; // age = age - 1 : 42
age += 5; // age = age + 5 : 47
age -= 7; // age = age - 7 : 40
age /= 2; // age = age / 2 : 20
age *= 3; // age = age * 3 : 60
```

INTERACTIONS

```
// Ouvre une popup avec bouton "Ok"
alert('Message');

// Ouvre une popup avec les boutons "Ok" et "Annuler"
// et retourne l'action effectuée sous forme de booléen
let yourAction = confirm('Message');

// Ouvre une popup demandant d'entrer du texte
// et retourne le texte entré dans la variable yourText
let yourText = prompt('Message');
```

AJAX : RÉCUPÉRER DES DONNÉES DISTANTES

```
fetch('http://url-cible.com').then(response => {
    // Affiche la réponse dans la console
    console.log(response);

    // Vérifie si la réponse est ok, puis retourne
    // ses données au format json
    if (response.ok) {
        return response.json();
    }
}).then(data => {
    // Affiche les données s'il n'y a pas d'erreur
    console.log(data);
}).catch(error => {
    // Affiche l'erreur s'il y en a une
    console.log(error);
});
```

AJAX - ENVOYER DES DONNÉES

```
fetch('http://url-cible.com', {
    method: 'POST', // Méthode (GET, POST, PUT, PATCH, DELETE)
    headers: { // Headers
        'Accept': 'application/json',
        'Content-Type': 'application/json'
    },
    body: 'Corps de la requête' // Données à envoyer
});
```

LOCALSTORAGE

```
// Stocke "Toto" dans la clé "user" du localstorage
localStorage.setItem('user', 'Toto');

// Récupère la valeur de la clé "user" (affiche "Toto")
localStorage.getItem('user');

// Supprime l'entrée pour la clé "user"
localStorage.removeItem('user');

// Vide totalement le localstorage
localStorage.clear();
```

FONCTIONS NATIVES COURANTES

STRING

```
let test = 'Test';

// Retourne la longueur de la chaîne
test.length; // Retourne 4

// Retourne le caractère situé à l'index 2
test.charAt(2); // Retourne "s"

// Teste si la chaîne contient la valeur "es"
test.includes('es') // Retourne true

// Retourne la chaîne selon un indice de début et de fin
// (L'indice de fin est optionnel)
test.slice(1); // Retourne "est"
test.slice(2, 3); // Retourne "s"

// Découpe la chaîne en tableau selon le séparateur "s"
test.split('s'); // Retourne ['Te', 't']
```

```
// Retourne la chaîne en minuscules
test.toLowerCase(); // Retourne "test"

// Retourne la chaîne en majuscules
test.toUpperCase(); // Retourne "TEST"

// Retire les blancs en début et fin de chaîne
'       test  '.trim(); // Retourne "test"
```

FONCTIONS NATIVES COURANTES

NUMBER

```
let test = 42;

// Ajoute une virgule et 2 décimales
test.toFixed(2); // Retourne 42.00

// Transforme un number en string
test.toString(); // Retourne "42"

// Teste si le nombre est un entier
Number.isInteger(test); // Retourne true

// Teste si le nombre est NaN (Not a Number)
Number.isNaN(test); // Retourne false

// Convertit un float en integer
Number.parseInt(42.3); // Retourne 42
```

FONCTIONS NATIVES COURANTES

ARRAY

```
let test = ['a', 'b', 'c'];
let test2 = ['d', 'e'];

// Retourne le nombre d'éléments dans le tableau
test.length; // Retourne 3

// Parcourt le tableau en itérant sur chaque élément
test.forEach((element, index) => {
    console.log(element); // Retourne 'a', puis 'b', etc...
    console.log(index); // Retourne 0, puis 1, etc...
});

// Ajoute un ou plusieurs éléments à la fin du tableau
test2.push('f'); // Retourne 3 (nouvelle taille du tableau)

// Ajoute un ou plusieurs éléments au début du tableau
test2.unshift('g'); // Retourne 4 (nouvelle taille du tableau)

// Supprime le dernier élément du tableau et retourne celui-ci
test2.pop(); // Retourne 'f';
```

```
// Supprime le premier élément du tableau et retourne celui-ci
test.shift(); // Retourne 'g'

// Teste si le tableau contient la valeur indiquée
test.includes('c'); // Retourne true

// Filtre un tableau selon une condition
test.filter(element => {
    if (element === 'c') {
        return element;
    }
}); // Retourne ['c']

// Retourne le premier élément correspondant à la condition
test.find(element => {
    if (element === 'c') {
        return element;
    }
}); // Retourne "c"
```

FONCTIONS NATIVES COURANTES

ARRAY

```
// Comme find(), mais retourne l'index de l'élément
test.findIndex(element => {
    if (element === 'c') {
        return element;
    }
}); // Retourne 2

// Retourne l'index du premier élément dont la valeur correspond
// à celle indiquée
test.indexOf('c'); // Retourne 2

// Parcourt le tableau en itérant sur chaque élément et permet
// la modifications de ceux-ci, tout en retournant un nouveau
// tableau avec les éléments modifiés
test.map((element, index) => {
    return element + index;
}); // Retourne ['a0', 'b1', 'c2']
```

```
// Retourne un tableau trié selon une comparaison 2 par 2
// Ici, le tableau est trié pour afficher les éléments
// par ordre croissant
[15, 42, 12].sort((elementA, elementB) => {
    if (elementA < elementB) {
        return -1;
    } else if (elementA > elementB) {
        return 1;
    } else {
        return 0;
    }
}); // Retourne [12, 15, 42]
```

JAVASCRIPT POUR LE WEB

ACCÉDER AUX ÉLÉMENTS DU DOM

```
// Récupère l'élément par son ID
document.getElementById('mon-id');

// Récupère le premier élément par son sélecteur CSS
document.querySelector('#mon-id');

// Récupère un tableau d'éléments par leur sélecteur CSS
document.querySelectorAll('.ma-classe');
```

CRÉER UN NOUVEL ÉLÉMENT HTML

```
// Crée un élément HTML de balise <h1>
document.createElement('h1');
```

AJOUTER DU CONTENU TEXTUEL OU HTML À UN ÉLÉMENT

```
let title = document.createElement('h1');
// Ajoute le texte "Mon titre principal" dans l'élément <h1>
title.textContent = 'Mon titre principal';

let nav = document.createElement('nav');
// Ajoute une liste non ordonnée dans l'élément <nav>
nav.innerHTML = '<ul><li>Accueil</li><li>Livre d\'or</li></ul>';
```

MODIFIEZ LES ATTRIBUTS DES ÉLÉMENTS HTML

```
let title = document.createElement('h1');

// Crée et assigne l'attribut "mon-attribut"
title.setAttribute('mon-attribut', 'Valeur de l\'attribut');
// Récupère la valeur de l'attribut
title.getAttribute('mon-attribut');
// Supprime l'attribut
title.removeAttribute('mon-attribut');
```

JAVASCRIPT POUR LE WEB

MODIFIER LES CLASSES

```
let title = document.createElement('h1');

// Ajoute une classe "classe-1" à l'élément <h1>
title.classList.add('classe-1');
// Retourne true si la classe "classe-1" existe
title.classList.contains('classe-1');
// Remplace la classe "classe-1" par "classe-2"
title.classList.replace('classe-1', 'classe-2');
// Supprime la classe "classe-2"
title.classList.remove('classe-2');
```

MODIFIER LES STYLES

```
let title = document.createElement('h1');

// Ajoute une marge au titre
title.style.margin = '20px 40px';
// Ajoute une couleur
title.style.color = '#fff';
// Ajoute une couleur de fond
title.style.backgroundColor = '#000';
```

JAVASCRIPT POUR LE WEB

AJOUTER OU SUPPRIMER DES ÉLÉMENTS ENFANTS

```
let body = document.querySelector('body');

let title = document.createElement('h1');
title.textContent = 'Mon titre';
// Ajoute l'élément title (<h1>) dans l'élément <body>
body.appendChild(title);

let newTitle = document.createElement('h1');
newTitle.textContent = 'Mon nouveau titre';
// Remplace l'élément title par l'élément newTitle
body.replaceChild(newTitle, title);

// Supprime l'élément newTitle
body.removeChild(newTitle);
```

ÉCOUTER ET RÉAGIR À UN ÉVÉNEMENT

```
let link = document.createElement('a');
link.textContent = 'Je suis un lien';
link.setAttribute('href', 'http://google.fr');
document.querySelector('body').appendChild(link);

// Création d'un listener sur l'événement click
link.addEventListener('click', (event) => {
    // Empêche le comportement par défaut
    // Ici, il s'agit de la soumission du lien
    event.preventDefault();

    // Affiche des informations sur l'événement
    console.log(event);
});
```