

Résumé JavaScript

Les bases.

Définition :

JavaScript est un langage de script, multiplateforme et orienté objet C'est un langage léger qui doit faire partie d'un environnement hôte (un navigateur web par exemple) pour qu'il puisse être utilisé sur les objets de cet environnement. (MDN)

Popularité :

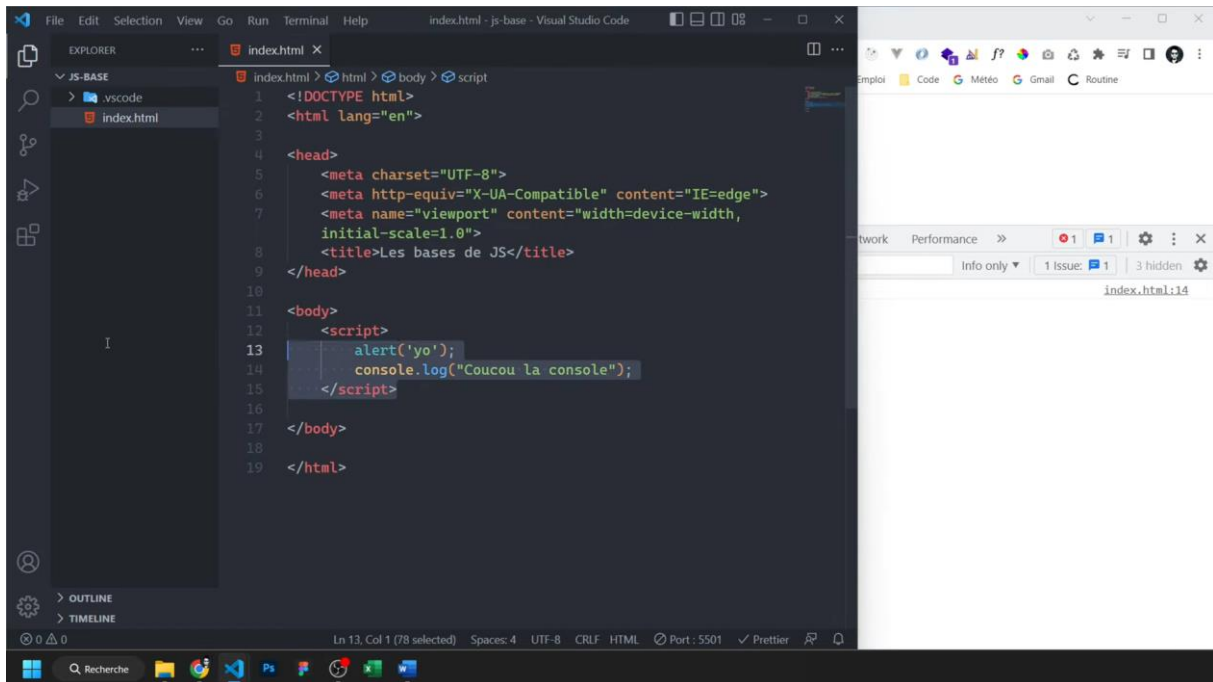
Selon l'extrapolation réalisée par SlashData, JavaScript serait le langage de programmation le plus utilisé au monde, avec près de 12,7 millions d'utilisateurs. Il devance Python et Java dans le classement. Sa progression est remarquable, avec 5 millions de nouveaux développeurs en seulement 3 ans.

Voici quelques grosses entreprises qui utilise du JavaScript



Où s'écrit le JS :

Le code JavaScript peut être écrit directement dans les fichiers HTML, avant la fin de la balise **body**, entre les balises `<script>...</script>`, comme le montre l'exemple ci-dessous.

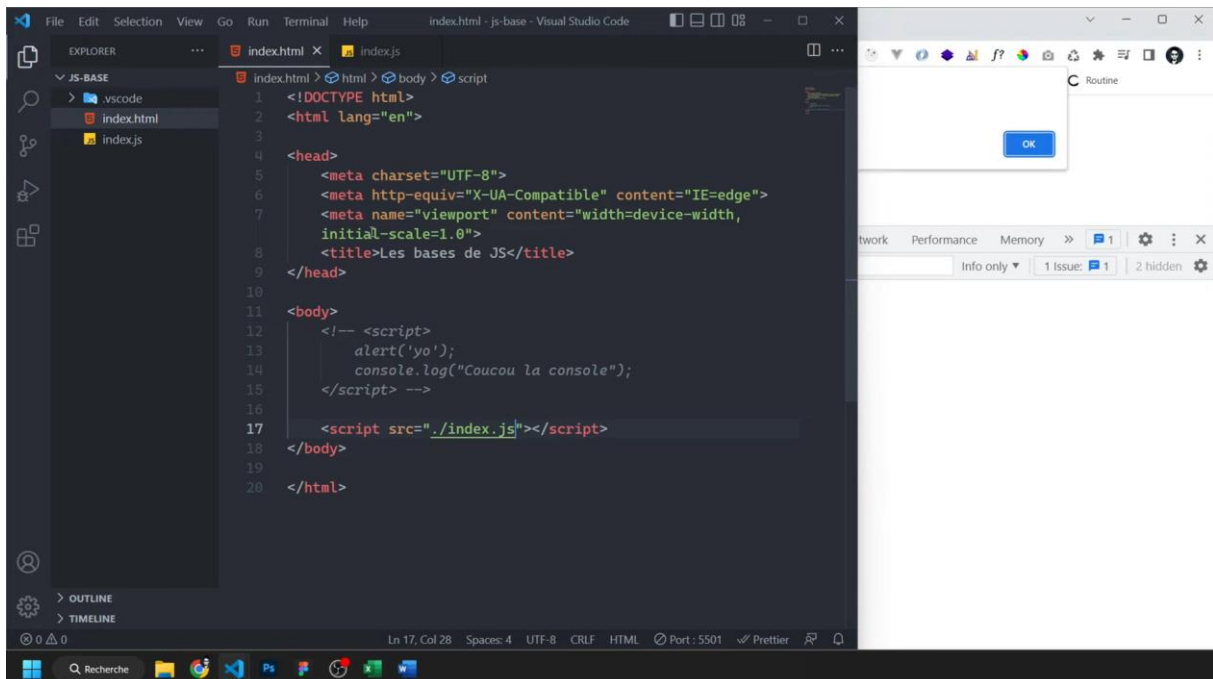


The screenshot shows the Visual Studio Code editor with a file named `index.html` open. The code is as follows:

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width,
8     initial-scale=1.0">
9   <title>Les bases de JS</title>
10
11 </head>
12
13 <body>
14   <script>
15     alert('yo');
16     console.log("Coucou la console");
17   </script>
18
19 </body>
20 </html>
```

The right-hand pane shows a web browser displaying the rendered HTML page, which is currently blank.

Ou il peut être écrit dans des fichiers JavaScript séparés et inclus dans les pages HTML à l'aide de la balise `<script src="chemin/vers/le/fichier.js"></script>`. Par exemple :



The screenshot shows the Visual Studio Code editor with two files open: `index.html` and `index.js`. The `index.html` file contains the following code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width,
8     initial-scale=1.0">
9   <title>Les bases de JS</title>
10
11 </head>
12
13 <body>
14   <!-- <script>
15     alert('yo');
16     console.log("Coucou la console");
17   </script> -->
18   <script src="./index.js"></script>
19
20 </body>
21 </html>
```

The `index.js` file is shown in the Explorer sidebar but its content is not visible in the editor. The right-hand pane shows a web browser displaying the rendered HTML page, which is currently blank. A small dialog box with an "OK" button is visible in the browser window.

Syntaxe :

La syntaxe de base du JavaScript (JS) comprend plusieurs éléments essentiels. Voici une introduction aux principaux aspects de la syntaxe JS :

1. Le point-virgule (;) :

En JavaScript, le point-virgule (;) est utilisé pour terminer une instruction. Cependant, il n'est pas strictement obligatoire dans tous les cas. Le langage JavaScript dispose d'un mécanisme appelé "Automatic Semicolon Insertion" (ASI) qui permet d'insérer automatiquement des points-virgules dans certaines situations.

2. Commentaires :

- Utilisez // pour commenter une ligne de code et /* ... */ pour commenter plusieurs lignes. Par exemple :

```
// Ceci est un commentaire sur une ligne

/*
Ceci est un commentaire
sur plusieurs lignes
*/
```

3. Camel case :

Le camel case est une convention couramment utilisée pour nommer des variables, des fonctions et des identifiants. Dans le camel case, chaque mot après le premier commence par une majuscule, sans espaces ni caractères spéciaux entre les mots. Voici un exemple :

```
let monNom = "John";
let ageUtilisateur = 30;
```

Les Variables :

En JavaScript, les variables sont utilisées pour stocker des valeurs. Voici comment déclarer une variable en JavaScript :

```
var maVariable = 10;
let autreVariable = "Bonjour";
const PI = 3.14;
```

Dans l'exemple ci-dessus, "maVariable" est une variable de type nombre (integer), "autreVariable" est une variable de type chaîne de caractères (string), et "PI" est une constante dont la valeur ne peut pas être modifiée.

Concaténation :

En JavaScript, la concaténation est utilisée pour combiner des chaînes de caractères. Il existe plusieurs façons de réaliser une concaténation en JavaScript :

1. Opérateur de concaténation (+) :

- Vous pouvez utiliser l'opérateur de concaténation (+) pour concaténer des chaînes de caractères. Par exemple :

```
var texte1 = "Bonjour";  
var texte2 = "Monde";  
var resultat = texte1 + " " + texte2; // "Bonjour Monde"
```

2. Template literals (ES6+) :

- Les template literals, également appelés littéraux de gabarits ou backticks (``), permettent une concaténation plus simple en utilisant la notation \${} pour insérer des variables ou des expressions dans une chaîne de caractères. Par exemple :

```
var texte1 = "Bonjour";  
var texte2 = "Monde";  
var resultat = `${texte1} ${texte2}`; // "Bonjour Monde"
```

Les types de données :

Il est important de noter que JavaScript est un langage de typage dynamique, ce qui signifie que vous n'avez pas besoin de spécifier explicitement le type de données lors de la déclaration de variable. Les variables peuvent contenir des valeurs de différents types de données, et elles peuvent également changer de type au cours de l'exécution du programme.

Voici quelques types de données courants en JavaScript :

- **Number** (nombre) : 3, 3.14, -10, etc.
- **String** (chaîne de caractères) : "Bonjour", 'Hello', etc.
- **Boolean** (booléen) : true, false
- **Array** (tableau) : [1, 2, 3], ['a', 'b', 'c'], etc.
- **Object** (objet) : {nom: 'John', age: 30}, etc.
- **Null** (valeur nulle) : null
- **Undefined** (valeur non définie) : undefined

Les Opérateurs :

JavaScript propose différents opérateurs qui permettent d'effectuer des opérations sur les données. Voici une liste des opérateurs les plus couramment utilisés en JavaScript :

1. Opérateurs arithmétiques :

- + : Addition
- - : Soustraction
- * : Multiplication
- / : Division
- % : Modulo (reste de la division)
- ** : Exponentiation (puissance)

2. Opérateurs d'assignation :

- = : Assignation de valeur
- += : Addition et assignation
- -= : Soustraction et assignation
- *= : Multiplication et assignation
- /= : Division et assignation

3. Opérateurs d'incrément et de décrémentation :

- ++ : Incrément de 1
- -- : Décrément de 1

Les structures de contrôle :

En JavaScript, les structures de contrôle permettent de prendre des décisions conditionnelles et de répéter des blocs de code. Voici les principales structures de contrôle en JavaScript :

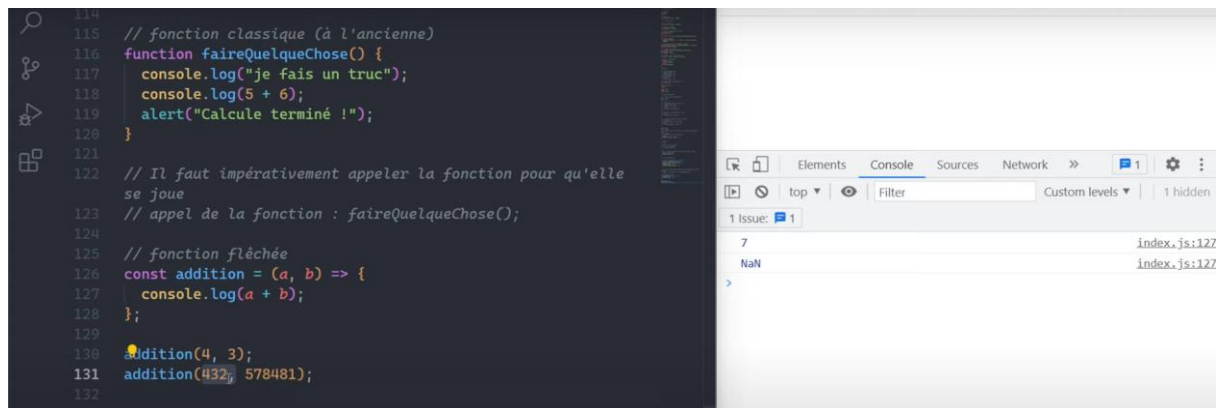
1. Structure de contrôle conditionnelle avec "if", "else if" et "else" :

```
if (condition) {  
    // Code à exécuter si la condition est vraie  
} else if (autreCondition) {  
    // Code à exécuter si la première condition est fausse et que l'autre cond  
} else {  
    // Code à exécuter si toutes les conditions précédentes sont fausses  
}
```

Les Fonctions :

En JavaScript, les fonctions sont des blocs de code réutilisables qui effectuent une tâche spécifique. Elles permettent d'organiser et d'encapsuler la logique du programme.

Voici comment déclarer et utiliser des fonctions en JavaScript :



La portée des variables :

En JavaScript, la portée des variables définit la visibilité et l'accès aux variables dans différentes parties du code. La portée des variables est déterminée par le contexte dans lequel elles sont déclarées. Il existe principalement deux types de portée en JavaScript :

1. Portée locale (fonctionnelle) :

- Les variables déclarées à l'intérieur d'une fonction sont dites locales et ne sont accessibles qu'à l'intérieur de cette fonction.
- Elles sont créées lorsque la fonction est appelée et détruites lorsque la fonction se termine.
- Les variables locales ont une portée limitée aux accolades ({}) de la fonction dans laquelle elles sont déclarées.
- Exemple :

```
function maFonction() {  
  var x = 5; // Variable locale  
  console.log(x); // Accessible ici  
}  
  
console.log(x); // Erreur : x n'est pas accessible à ce niveau
```

2. Portée globale :

- Les variables déclarées en dehors de toutes les fonctions sont dites globales et sont accessibles depuis n'importe où dans le code.

- Elles sont créées lorsque le script démarre et restent en mémoire jusqu'à la fin de l'exécution.
- Les variables globales ont une portée qui s'étend à tout le script.
- Exemple :

```
var y = 10; // Variable globale

function maFonction() {
  console.log(y); // Accessible ici
}

console.log(y); // Accessible ici également
```