Rabhi mohamed seghir

Contact :mohamedrbh519@gmail.com.

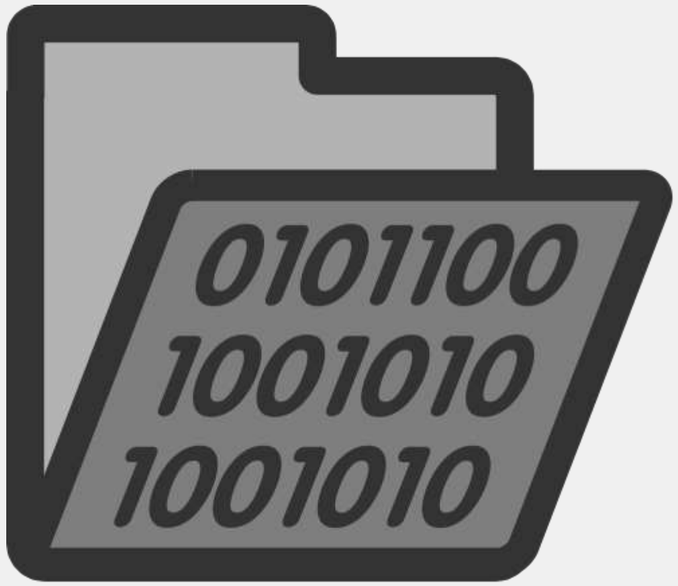
01 INITIATION À JAVASCRIPT

INTRODUCTION À L'HISTOIRE DE JAVASCRIPT

Javascript est un langage de programmation de scripts. Il est utilisé pour rendre les pages web *dynamiques / interactives.*

Javascript est un langage orienté objet à prototype. L’orienté objet est un paradigme de programmation sur lequel on s'attardera plus tard.

MAIS AVANT, QU'EST-CE QU'UN LANGAGE DE PROGRAMMATION ?



UN OUTIL DE RÉSOLUTION DE PROBLÈMES

C’est la manière que nous, humains, utilisons pour traduire des instructions à un ordinateur.

JavaScript est un langage interprété.

Il existe deux types de langage de programmation.
Les langages de programmations compilés et les langages de programmations interprétés.

INTERPRÉTÉ VS COMPILÉ

Dans le cas d'un langage interprété, toutes les commandes sont analysées et exécutées une à une.
Dans le cas langage compilé, le code est analysé et traduit en code source qui sera exécuté par l'OS directement.

06

DU COUP, OÙ TROUVER L'INTERPRÉTEUR DE JAVASCRIPT?

Les navigateurs (Google Chrome, Mozilla, Safari etc…) ont un interpréteur de JS intégré.

02

COMMENCER À CODER EN JS

Les outils et notions de base

Les outils pour coder en JavaScript

Vous aurez besoin de deux outils :

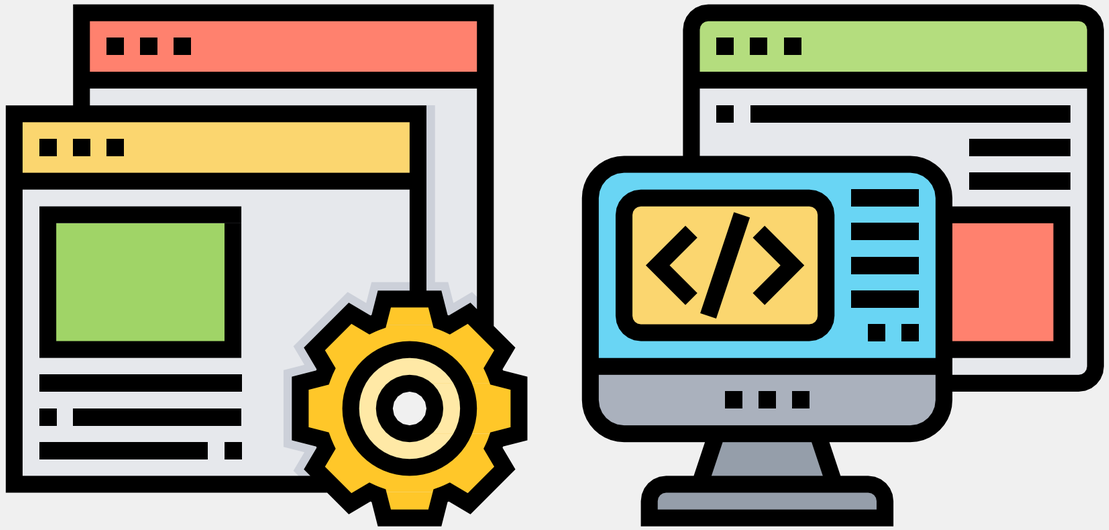
- Un éditeur de texte
- Un navigateur web

ÉDITEURDETEXTE

NAVIGATEURWEB

08

LE DÉVELOPPEMENT WEB FRONT ET BACK END



Le front end c’est l’interface avec laquelle l’utilisateur interagit.

Le back end est la partie qui gère les données, interagit avec le serveur et produit des résultats.

Javascript est l’un des trois outils principaux du développement web frontal aux côtés d’HTML et CSS.

- **HTML sert de structure à une page web**
- **CSS permet de gérer l’aspect et le design d’une page web**
- **JS permet de gérer le comportement des composants d’une page web**

Il est temps de découvrir les principes de programmation de base et leur syntaxe en JS.

03

A VOS CLAVIERS !

Affichages

Pour afficher des valeurs et des résultats, on peut les afficher au niveau de la console. Pour cela, on utilise console.log()

```
<script>
    console.log("Hello World!");
</script>
```

Commentaires

Un commentaire est une ligne de code qui est ignorée par l'interpréteur.

```
<script>
    //  ceci est un commentaire

    /* ceci est églament
    un commentaire */
</script>
```

VARIABLES

Les variables sont utilisées pour manipuler des données
Il existe 5types de variables :

- number (nombres entiers ou flottants)
- boolean (booléens)
- string (chaîne de caractères)
- undefined (variable indéfinie)
- null (variable définie, mais sans valeur)

VARIABLES

Les variables sont typés dynamiquement par JavaScript.
Pour déclarer une variable, il faut utiliser le mot clé var.

```
<script>
    var a = 0;
    var b = "bonjour";
    var c = true;
</script>
```

VARIABLES

Il est possible de voir le type d'une variable en utilisant la fonction typeof(). Pour l'afficher, on utilise console.log().

```
<script>
    var a = 0;
    console.log(typeof(a))
</script>
```

VARIABLES

Il existe une deuxième manière de déclarer les variables, qui est : let.
let permet de déclarer des variables dont la portée réduite. Elle est limitée à un bloc d'instructions. Un bloc d'instructions est défini par deux acolades.
Par exemple, on utilise des blocs d'instructions pour les branchements conditionnels.

```
<script>
    {
        let a = 0;
        let b = 'anis';
    }
</script>
```

NUMBER

Le type Number permet de stocker des nombres allant jusqu'à 2^53.

Il permet de stocker les nombres naturels et flottants.

Il possède trois signes :

- +Infinity
- -Infinity
- NaN

NUMBER

```
let a = 1;
let b = 1.75;
let c = +Infinity;
let d = -Infinity;
let e = NaN;
```

EXERCICE

Affichez dans votre console l'addition de deux nombres entiers.

PS : l'addition se fait avec l'opérateur +.

BOOLEAN

Le type boolean permet de donner une valeur de vérité à une variable.

Il a deux symboles spéciaux :

- True
- False

BOOLEAN

```
let x = true;
let y = false;
```

STRING

Le type string permet de stocker des chaines de caractères. En d'autres termes, du texte.

- Concaténation à l’aide de l’opérateur +.
- Longueur de la chaîne à l’aide de la propriété length.
- Récupération d’un caractère à l’aide de [n] ou charAt().
- Caractère d’échappement :\

String


```
let test = "true";
let x = 'jazz';
```

EXERCICE

Quelles sont les types des variables suivantes ?

```
let a = "Javascript";
let b = "true";
let c = true;
let d = 100;
```

EXERCICE

Que valent les variables suivantes ?

```
let a = "Javascript";
let b = "true";
let c = a + b;
let d = 100;
let e = a + d;
```

EXERCICE

Affichez les chaines de caractères et leur longueur sur la console.

Affichez le caractère à l'indice 5 pour chaque chaine. Que remarquez-vous ?

```
<script>
    {
        var a="Bonjour";
        var b="Je m'appelle";
        var c= "Yasmine";
    }
</script>
```

UNDEFINED & NULL

En plus des nombres, des types primitifs vu auparavant, nous avons undefined et null.

-undefined est le type qui est attribué implicitement à une variable déclarée sans valeur.
-null est le type que l'on attribue explicitement à une variable lorsque l'on souhaite la définir sans valeur.
Typiquement, null est utilisée pour la gestion d'erreurs.

UNDEFINED & NULL

En plus des nombres, des types primitifs vu auparavant, nous avons undefined et null.

```
<script>
    {
        var a; // type undefined
        var b=null; // type null
    }
</script>
```

CONSTANTES

Les constants sont des variables déclarées en utilisant le mot clé **CONST**, comme leur noms l'indique, la valeur d'une constante ne peut pas être modifié après leur avoir attribué explicitement une valeur.

```
<script>
    const pi = 3.14;
    const avogadro  = 6.02214
</script>
```

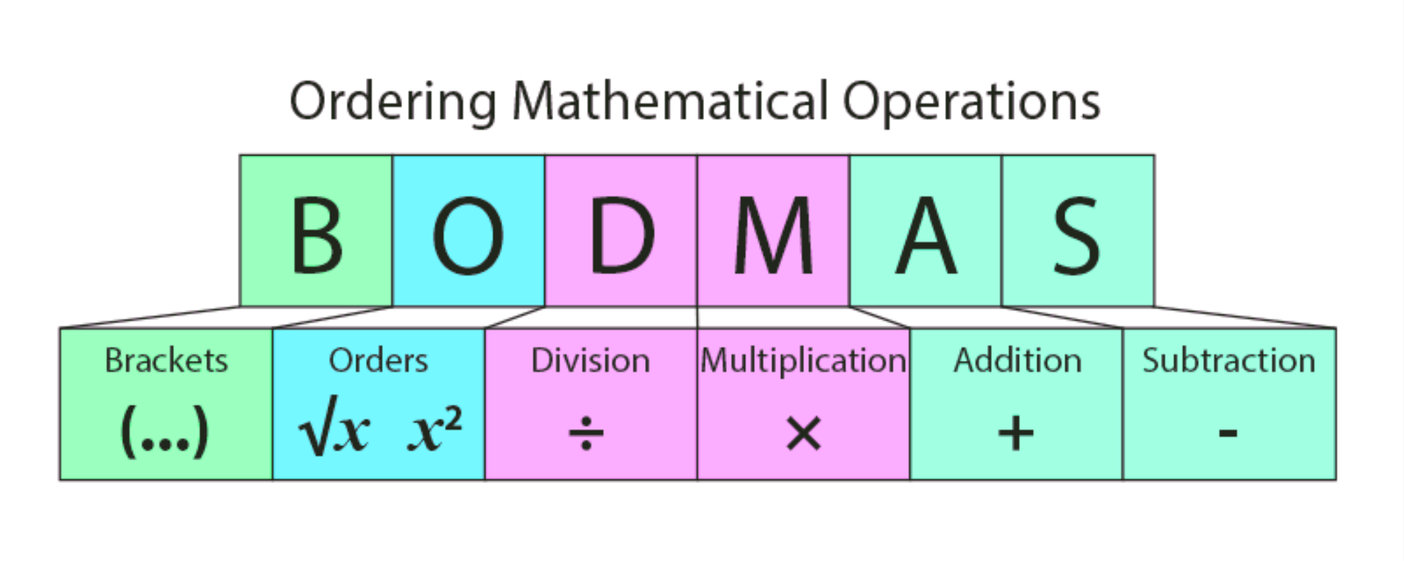
EXERCICE

Transcrivez les informations suivantes à l'aide de variable ou de constantes.

```
1   Le système solaire est consituté de 9 planètes.
2   L'hydrogène a un électron.
3   En ce moment, il y a 12 personnes dans la salle d'attentes.
4   Au jour d'aujourd'hui, il y a 1420 satellites starling opérationnels.
```

OPÉRATEURS ARITHMÉTIQUES

Passons à présent aux opérateurs arithmétiques.
le plus +, le moins -, la multiplication * et la division /.
l'ordre de priorités est :
* et / d'abord, puis + et le - , de gauche a droite, les parenthèses les plus internes en premier.



EXERCICE

Donner le résultat des opérations suivantes

```
<script>
    const pi = 3.14;
    var r = 4;
    let res = pi * r * r - 5;
    // res = ?
    const res2 = 5 * 4 - 2 * (12 - (7 * 2));
    // res2 = ?
</script>
```

LES CONVERSIONS IMPLICITE

La conversion implicite se fait lors de l'utilisation d'opérateurs arithmétiques entre variables de types différents.

- La concaténation d'une chaine de caractères avec un nombre donne une chaine de caractères
- Dans le cas de nombres stockés sous forme de chaines de caractères, ils sont traités comme nombre avec les autres opérateurs arithmétiques.

```
<script>
    {
        x="11"+2; // 112
        x="11"-2; // 9
    }
</script>
```

LES OPÉRATEURS LOGIQUES

Les opérateurs logiques nous permettent de faire des comparaisons entre des variables et avoir un résultat sous forme d'un boolean.

&& ET	! NON	> SUPERIEUR
|| OU	< INFERIEUR	!= NEGALITÉ

== VS ===

== et === sert tous les deux à étudier l'égalité entre deux opérandes. Néanmoins, ils ne sont pas équivalent.
== est utilisée pour l'égalité des valeurs, et === est utilisé pour légalité des valeurs et du type.

x	y	==	===
undefined	undefined	true	true
null	null	true	true
true	true	true	true
false	false	true	true
'foo'	'foo'	true	true
0	0	true	true
+0	-0	true	true
+0	0	true	true
-0	0	true	true
0	false	true	false
""	false	true	false
""	0	true	false
'0'	0	true	false
'17'	17	true	false
[1, 2]	'1,2'	true	false
new String('foo')	'foo'	true	false
null	undefined	true	false

EXERCICE

Donner le résultat des opérations suivantes

```
<script>
    var a = true && false;
    let b = false || true;
    var z = 0 && false;
    var x = 10 && true;
    const cond = 5 === 6;
    let cond2 = 5 == '5';
</script>
```

Passons aux structures de contrôles. Les structures qui permettent de répéter des instructions, gérer des conditions, etc...

04 LES STRUCTURES DE CONTROLE

LES CONDITIONS

Les conditions sont des instructions qui nous permettent d’exécuter un bout de code seulement et seulement si la condition est vérifiée, ou d'exécuter un autre bout de code dans le cas contraire.

```
<script>
    if (condition) { // la condition doit avoir une valeu booléene
        //intructions
    }
    else { // le else est optionnel
        //instructions
    }
</script>
```

EXERCICE

Donner les scripts qui permettent :

-d'afficher "possibilité d'avoir le permis de conduire" si la variable âge est supérieure ou égale à 18.

- d'afficher le signe de la variable x (négatif ou positif)

fg

LE SWITCH CASE

Pour éviter d'imbriquer beaucoup de conditions, on peut utiliser le switch case

```
switch (expression) {
    case x:
        // instructions
        break;
    case y:
        // instructions
        break;
    default:
    // instructions

}
```

EXERCICE

Donner les scripts qui permettent :

- d'afficher le mois si son nombre est donné

LES BOUCLES

Avant de définir les boucles, imaginons qu'on ait besoin d'afficher tous les nombres entre 1 et 100, ou que vous vouliez exécuter des instructions similaires n fois.

Comment allez-vous procéder ?

LES BOUCLES

Les boucles remédient à cette problématique. Une boucle vous permet d'exécuter des instruction n fois, sans pour autant les re-écrire plusieurs fois.

Pour le moment, on va voir uniquement deux types de boucle, for

et while. Plus tard nous verons aussi la boucle foreach.

LA BOUCLE FOR

```
<script>
    for (let index = 0; condition; index++) {
        //initialisation de l'index
        // la condition d'aret( par rapport a l'index)
        // l'incrementation ou decremntation de l'index


        //instructions
    }
</script>
```

LA BOUCLE WHILE

```
<script>
    while (conditions) {
        //boucle tantque la condition est vrai


        //instructions
    }
</script>
```

EXERCICE

Donner le script qui font le traitement suivant:

- affiche "itération x" sur la console, avec x allant de 0 à 99.
- affiche tous les nombres premiers existanst entre 0 et 400.
- affiche les nombres dans un ordre décroissant (120, 119.... 3, 2, 1, 0)