



ECOLE MAROCAINE DES
SCIENCES DE L'INGENIEUR

Membre de
HONORIS UNITED UNIVERSITIES

23/10/2022

Langages de Scripts



AMEKSA Mohammed
emsi.ameksa@gmail.com



ECOLE MAROCAINE DES
SCIENCES DE L'INGENIEUR

Membre de
HONORIS UNITED UNIVERSITIES



Apprendre JavaScript

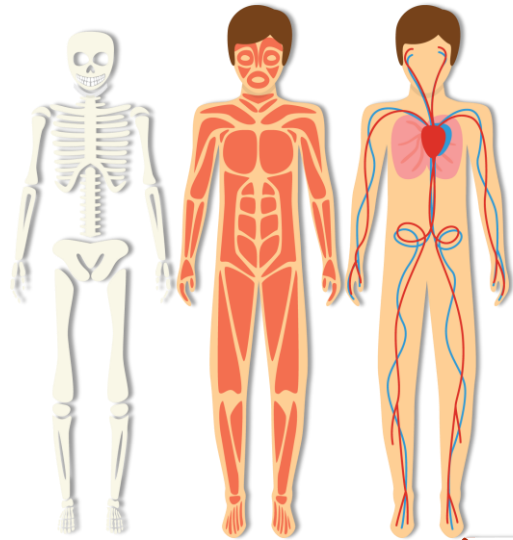
- ✓ Les types des données et les variables
- ✓ Les opérations et les instructions
- ✓ Les fonctions
- ✓ Les objets JavaScript
- ✓ La gestion des évènements
- ✓ Les objets standards du navigateur {DOM – DHTML}



Qu'est-ce que le JavaScript?

JavaScript est un langage de programmation ou de script qui permet d'intégrer des fonctionnalités dans les pages web.

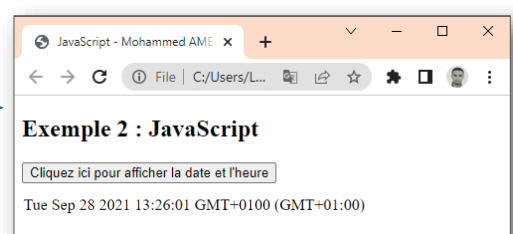
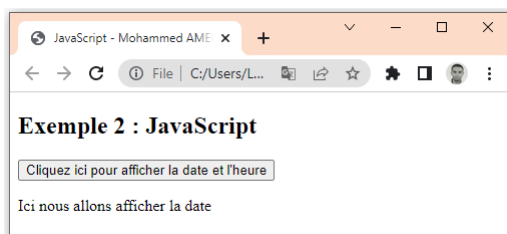
- **HTML** pour définir le contenu des pages Web
- **CSS** pour spécifier la mise en page des pages Web
- **JavaScript** pour programmer le comportement des pages Web



Qu'est-ce que le JavaScript?

Exemple 1

```
<body>  
  <h2>Exemple 2 : JavaScript</h2>  
  <button type="button" onclick="document.getElementById('demo').innerHTML=Date()">  
    Cliquez ici pour afficher la date et l'heure  
  </button>  
  <p>  
    Ici nous allons afficher la date  
  </p>  
</body>
```



Qu'est-ce que le JavaScript?

Exemple 2

```
<!DOCTYPE html>
<html lang="fr">

<head>
  <meta charset="UTF-8">
  <title>JavaScript - Mohammed AMEKSA</title>
</head>

<body>
  <button id="btn">Affichez un alert</button>
</body>
</html>
```

```
<style>
  button {
    font-family: sans-serif;
    border: none;
    padding: 15px 30px;
    font-size: 15px;
    outline: none;
    margin: 10px;
  }

  #btn {
    background-color: rgb(23, 52, 89);
    color: #f8f9f9;
    box-shadow: .5px .5px 1px 2px #000;
  }
</style>
```

JS

```
<script>
  document.getElementById("btn").onclick = function() {
    alert("Voici une boîte de dialogue")
  }
</script>
```

Affichez un alert

Affichez un alert

Affichez un alert

2022 / 2023

Langage de Script - Mohammed AMEKSA

5

Qu'est-ce que le JavaScript?

- Le **JavaScript** est un langage de programmation créé en **1995**.
- Le **JS** va nous permettre de créer des pages **interactives** et « **vivantes** » à l'aide de scripts.
- Un **script**, c'est tout simplement une suite d'instructions qui vont être interprétées par un programme.
- Le JavaScript est un langage de programmation dit « **client-side** »: Tout comme le HTML, le JavaScript est généralement exécuté par le navigateur de l'internaute.
- JavaScript est un langage de scripts incorporé aux balises Html, qui permet d'améliorer la présentation et l'interactivité des pages Web.

2022 / 2023

Langage de Script - Mohammed AMEKSA

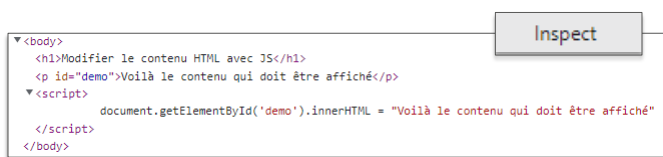
6

Que peut faire le Javascript ?



JS peut Modifier le contenu HTML

```
<body>  
  <h1>Modifier le contenu HTML avec JS</h1>  
  <p id="demo">Cet paragraphe avec HTML simple</p>  
  <script>  
    document.getElementById('demo').innerHTML = "Voilà le contenu qui doit être affiché"  
  </script>  
</body>
```



2022 / 2023

Langage de Script - Mohammed AMEKSA

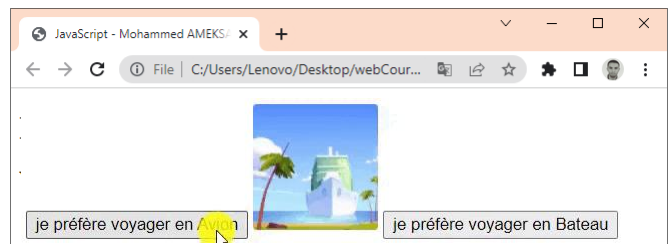
7

Que peut faire le Javascript ?



JS peut Modifier les valeurs des attributs HTML

```
<div style="text-align:center">  
  <button onclick="document.getElementById('imageID').src='images/avion.png'">  
    je préfère voyager en Avion  
  </button>  
    
  <button onclick="document.getElementById('imageID').src='images/bateau.png'">  
    je préfère voyager en Bateau  
  </button>  
</div>
```



2022 / 2023

Langage de Script - Mohammed AMEKSA

8

Que peut faire le Javascript ?



JS peut Modifier le style CSS

```
<!-- change style CSS -->
<div>
  <p id="pID">
    Texte peut changer son style avec JS
  </p>
  <button onclick="document.getElementById('pID').style.fontSize = '35px'">
    changer la taille du font
  </button>
</div>
```

Texte peut changer son style avec JS

changer la taille du font

Texte peut changer son style avec JS

changer la taille du font

Que peut faire le Javascript ?



Masquer / Afficher les éléments HTML

```
<!-- masquer un element -->
<div>
  <p id="p1">
    1er paragraphe
  </p>
  <p id="p2">
    2ème paragraphe
  </p>
  <p id="p3">
    3ème paragraphe
  </p>
  <button onclick="document.getElementById('p2').style.display = 'none'">
    Masquer la 2ème paragraphe
  </button>
</div>
```

1er paragraphe
2ème paragraphe
3ème paragraphe

Masquer la 2ème paragraphe

```
<!-- afficher un element masqué -->
<div>
  <p id="p1">
    1er paragraphe
  </p>
  <p id="p2" style="display: none;">
    2ème paragraphe
  </p>
  <p id="p3">
    3ème paragraphe
  </p>
  <button onclick="document.getElementById('p2').style.display = 'block'">
    Afficher la 2ème paragraphe
  </button>
</div>
```

1er paragraphe
3ème paragraphe

Afficher la 2ème paragraphe

1er paragraphe
3ème paragraphe

Masquer la 2ème paragraphe

1er paragraphe
2ème paragraphe
3ème paragraphe

Afficher la 2ème paragraphe

Qu'est-ce que le JavaScript?

- **JavaScript** et **Java** sont des langages complètement **différents**
- **Java** a été développé par **Sun**.
- **JavaScript** a été développé par **Netscape** en 1995.
- **Java** est **compilé** (applets), **JavaScript** est **interprété** (scripts)
- **Java** permet de créer des **applications** qui sont exécutées sur une machine tandis que le code **JavaScript** est exécuté uniquement sur un **navigateur**.
- Avec JS ne peut pas **lire/écrire** dans les fichiers
- Avec JS ne peut pas **exécuter** d'autres programmes



Insertion du code JavaScript?

- Le code JavaScript doit être inséré entre `<script>` et `</script>`
- Il existe deux méthodes d'insertion: **Interne** et **externe**
- JavaScript dans `<head>` ou `<body>` ?

Les scripts peuvent être placés dans le `<body>`, ou dans le `<head>` d'une page HTML, ou dans les deux.

La valeur de x est : 3.6

```
<html>
<head>
  <title>Insertion JS</title>
  <script>
    var x = 3.6;
    document.write('La valeur de x est : ' + x);
  </script>
</head>
<body>
</body>
</html>
```

```
<html>
<head>
  <title>Insertion JS</title>
</head>
<body>
  <script>
    var x = 3.6;
    document.write('La valeur de x est : ' + x);
  </script>
</body>
</html>
```

Placer des scripts au bas de l'élément `<body>` améliore la vitesse d'affichage, car l'interprétation des scripts ralentit l'affichage

Insertion du code JavaScript?



<head> ou <body> ? quelle est la différence

- lorsque le script se trouve dans l'en-tête, il se charge en premier avant le contenu de la page.

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>JavaScript - Mohammed AMEKSA</title>
  <script>
    document.getElementById('btn').addEventListener("click", function(){
      document.getElementById('demo').innerHTML = "Le contenu à afficher";
    });
  </script>
</head>
<body>
  <h1>Emplacement du script JS</h1>
  <p id="demo">
    cela ne peut pas fonctionner car le gestionnaire d'événements
    se charge d'abord avant les éléments HTML.
  </p>
  <button id="btn">Changer le contenu</button>
</body>
</html>
```



Insertion du code JavaScript?



<head> ou <body> ? quelle est la différence

- lorsqu'il se trouve au <body>, il se charge après le chargement du contenu de la page (éléments HTML).

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>JavaScript - Mohammed AMEKSA</title>
</head>
<body>
  <h1>Emplacement du script JS</h1>
  <p id="demo">
    Maintenant doit être fonctionner car le gestionnaire
    d'événements se charge d'abord après les éléments HTML.
  </p>
  <button id="btn">Changer le contenu</button>

  <script>
    document.getElementById('btn').addEventListener("click", function() {
      document.getElementById('demo').innerHTML = "Le contenu à afficher";
    });
  </script>
</body>
</html>
```



Insertion du code JavaScript?



Interne

- Consiste à faire **appel** à l'élément script et à **insérer** le code JavaScript dans le **même** fichier HTML.

```
<> insertionInterne.html > ...
1  <!DOCTYPE html>
2  <html>
3  <body>
4      <p>
5          On cherche à insérer une autre paragraphe
6          avec un Script dans le meme fichier.
7      </p>
8      <p id="demo"></p>
9      <button type="button" onclick="jsFonction()">
10         JS fonction
11     </button>
12     <script>
13         function jsFonction() {
14             document.getElementById("demo").innerHTML =
15                 "Ce texte est inséré par un script dans le meme fichier";
16         }
17     </script>
18 </body>
19 </html>
```

On cherche à insérer une autre paragraphe avec un Script dans le meme fichier.

JS fonction

On cherche à insérer une autre paragraphe avec un Script dans le meme fichier.

Ce texte est inséré par un script dans le meme fichier

JS fonction

2022 / 2023

Langage de Script - Mohammed AMEKSA

15

Insertion du code JavaScript?



externe

- Consiste à écrire le code JavaScript dans un ou plusieurs fichiers **externes** portant l'extension « .js ». Et l'appeler par la suite dans le fichier HTML.

```
<> insertionJS.html > ...
1  <!DOCTYPE html>
2  <html>
3  <body>
4      <p>
5          On cherche à afficher une autre paragraphe,
6          avec un fichier JS externe
7      </p>
8      <p id="demo"></p>
9      <button type="button" onclick="jsFonction()">
10         JS externe
11     </button>
12     <script src="script.js"></script>
13 </body>
14 </html>
```

```
JS script.js > ...
1  function jsFonction() {
2      document.getElementById("demo").innerHTML =
3          "Ce texte est inséré par un script dans un fichier externe";
4  }
```

On cherche à afficher une autre paragraphe, avec un fichier JS externe

JS externe

On cherche à afficher une autre paragraphe, avec un fichier JS externe

Ce texte est inséré par un script dans un fichier externe

JS externe

2022 / 2023

Langage de Script - Mohammed AMEKSA

16

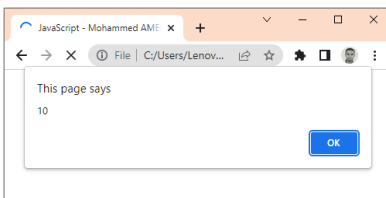
- **innerHTML** : Écrire dans un élément HTML

```
<body>
<h1>JavaScript Affichage</h1>
<p id="demo"></p>
<script>
  var x = 5;
  var y = 5;
  var somme = x + y;
  document.getElementById("demo").innerHTML = somme;
</script>
</body>
```



- **alert()** : Écriture dans une boîte d'alerte

```
<script>
  var x = 5;
  var y = 5;
  var somme = x + y;
  alert(somme);
</script>
```



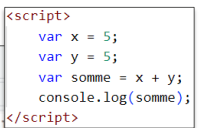
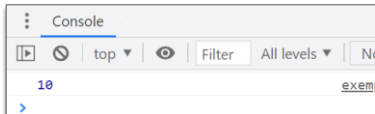
- **document.write()** : écrire dans les fenêtres du navigateur

```
<script>
  var x = 5;
  var y = 5;
  var somme = x + y;
  document.write(somme);
</script>
```



- **console.log()** : Écriture dans la console du navigateur

```
<script>
  var x = 5;
  var y = 5;
  var somme = x + y;
  console.log(somme);
</script>
```



- Une langage de programmation lié à HTML (*intégré aux pages HTML*).
- Code interprété par le navigateur client \neq code PHP (interprété du côté serveur).
- JavaScript \neq Java
- deux méthodes d'insertion: *Interne* et *externe* (dans l'entête et/ou dans le corps).
- JavaScript peut Modifier le contenu HTML, les valeurs des attributs HTML, les styles (CSS), et Masquer ou Afficher les éléments HTML.
- Affichage des résultats de la sortie : InnerHTML, document.write(), window.alert(), et console.log().

Apprendre JavaScript

Déclarations Et les variables



Instructions Javascript

programme informatique est une liste d'instructions à exécuter par un ordinateur.



Un script JS est une liste d'instructions à interpréter par un navigateur Web.

- Une seule instruction Javascript doit être **terminée** ou **séparée** par un point-virgule (;)
 - Les instructions Javascript peuvent être regroupées à l'intérieur de crochets {...}.
- **blocs de code** : sont utilisés pour que les déclarations soient exécutées ensemble.

✓ Une seule instruction JS simple :

```
document.getElementById("demo").innerHTML = "Contenu à afficher";
```

✓ Une seule instruction JS simple dans deux lignes :

```
document.getElementById("demo").innerHTML =  
    "Contenu à afficher";
```

✓ Bloc d'instruction :

```
function afficherMsg() {  
    document.getElementById('demo').innerHTML = "Hello world!";  
}
```

Instructions JS

mots-clés, expressions, opérateurs, Valeurs, ; ,
et/ou commentaires.

```
<html>
<body>
  <h2>JavaScript Instruction</h2>
  <p id="demo"></p>
  <script>
    document.getElementById("demo").innerHTML = "Je suis etudiant a EMSI";
  </script>
</body>
</html>
```

Points-virgules ;

- En JavaScript, chaque instruction de code doit se terminer par un point virgule.
- Des points-virgules séparent les instructions exécutables JavaScript.

Mots clés en JS

- Les instructions JavaScript commencent généralement par un mot-clé
- Chaque mot-clé en JavaScript a un sens particulier.
- par exemple, pour déclarer une variable, il faut utiliser le mot-clé var.

Syntaxe du Javascript

Syntaxe JS

- JavaScript suit un ensemble de règles appelées syntaxe.
- La syntaxe JS doit être respectée.

```
<html>
<body>
  <h2>JavaScript Instruction</h2>
  <p id="demo"></p>
  <script>
    document.getElementById("demo").innerHTML = "Je suis etudiant a EMSI";
  </script>
</body>
</html>
```

Commentaires JS

- Un commentaire permet de placer du texte en-dehors du script : il n'est alors pas interprété.
- Deux Syntaxes sont possibles :
 1. une seule ligne: // commentaire
 2. Plusieurs lignes: /* commentaires */

```
<script>
  z = 2 + 3; // une expression JS
  y = z + 2;
  /* une autre expression JS
     on assigne la valeur de z à y
     puis on ajoute un 2 au résultat
  */
</script>
```

Les variables en Javascript



Note générale

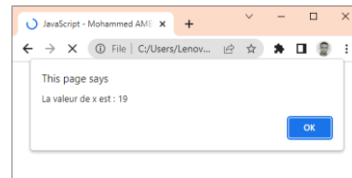
- Dans un langage de programmation, ou même en algèbre, les variables sont utilisées pour stocker des valeurs de données, (conteneurs).
- pareil en javascript, les variables sont utilisées pour stocker des données.
- les données stockées peuvent être un texte, un nombre ou autres.

déclaration d'une variable

- l'instruction « **var** » permet de créer ou de déclarer une variable.

```
var x = 19;  
alert('La valeur de x est : ' + x);
```

Ici la variable x est déclarée et initialisée dans une ligne



```
var x;  
x = 19;  
alert('La valeur de x est : ' + x);
```

On peut aussi déclarer ou créer une variable d'abord, puis lui attribuer une valeur plus tard (l'initialiser)

Les variables en Javascript

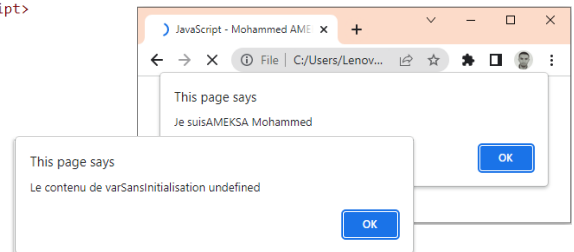


déclaration d'une variable

- JS utilise les mots-clés **var**, **let** et **const** pour déclarer des variables.
- Chaque nouvelle variable doit avoir un nom unique qu'on appelle « **identifiant** ».
- Un signe égal (=) est utilisé pour attribuer des valeurs aux variables.
- On peut déclarer plusieurs variables à la fois, en utilisant une seule instruction. Il suffit de séparer les variables par une virgule.

NB : Si vous n'assignez pas une valeur à la variable, la valeur sera indéfinie (**undefined**).

```
<script>  
var prenom = "Mohammed",  
    nom = "AMEKSA";  
var varSansInitialisation;  
alert("Je suis" + nom + " " + prenom);  
alert("Le contenu de varSansInitialisation " + varSansInitialisation);  
</script>
```



IMPORTANT

- ✓ Identifiants JavaScript sensibles à la casse

Déclaration, de plusieurs variables

```
<script>
  var pi = 3.14;
  var x = 4;
  var nomPrenom = "AMCF Creative";
  var nomPrenom2 = 'AMCF Pro';
</script>
```

```
<script>
  var pi, x, nomPrenom, nomPrenom2;
  pi = 3.14;
  x = 4;
  nomPrenom = "AMCF Creative";
  nomPrenom2 = 'AMCF Pro';
</script>
```

```
<script>
  var pi = 3.14,
  x = 4,
  nomPrenom = "AMCF Creative",
  nomPrenom2 = 'AMCF Pro';
</script>
```

```
<script>
  var pi = 3.14, x = 4, nomPrenom = "AMCF Creative", nomPrenom2 = 'AMCF Pro';
</script>
```

Déclaration, de plusieurs variables

```
<script>
  var cour = "JavaScript";
  var cour;
  document.getElementById('demo').innerHTML = cour;
</script>
```

↓
JavaScript

```
<script>
  var cour = "JavaScript";
  var cour = "PHP";
  document.getElementById('demo').innerHTML = cour;
</script>
```

↓
PHP

- Les variables définies avec **let** ne peuvent pas être redéclarées
- Les variables déclarées à l'intérieur d'un bloc avec le mot-clé **let** ou **const** ne sont pas accessibles depuis l'extérieur du bloc

Type de données en Javascript



- En JS ou dans n'importe quel langage de programmation, le concept de « **data types** » est important.
- Les variables JavaScript peuvent contenir des **nombres** ainsi du **texte**.

Définition

« **data types** » : les différents type de données qui peuvent être stockées / utilisées dans un script JS.

Les variables JavaScript peuvent être de type:

Chaînes de caractères (**string**), Nombres (**number**), Booléens (**boolean**), Indéfinis (**undefined**), Nuls (**null**), Objets (**object**), ou Tableaux (**arrays**).

String

String est une séquence de caractères utilisée pour représenter du texte. Ce type de données peut être écrit avec des guillemets simples ou doubles.

number

le type de données numériques est un nombre entier ou un nombre à virgule flottante.

String & number

La combinaison de deux chaînes de caractères renvoie une chaîne de caractères

La combinaison de deux nombres renvoie un nombre

La combinaison d'un nombre et d'une chaîne de caractères renvoie une chaîne de caractères

2022 / 2023

Langage de Script - Mohammed AMEKSA

27

Type de données en Javascript



boolean

Les booléens sont un type de données logique qui ne peut avoir que les valeurs **true** ou **false**.

- Ils sont utilisés pour les tests conditionnels
- Ils sont généralement la sortie des opérateurs de comparaison

undefined

lorsqu'une variable n'a pas de valeur assignée, ou n'est pas déclarée, la valeur est indéfinie.

null

null est un type de données spécial indiquant une valeur nulle, et non une chaîne vide "" ou 0.

object

Un objet est une collection de données liées, il contient des propriétés sous forme des paires **clé : valeur**, placées entre accolades {}.

Chaque paire est séparée par une virgule (,).

```
var personne = {  
  nomPrenom: "Mohammed AMEKSA",  
  age: 26,  
  profession: 'Enseignant',  
  score: 5.5,  
  acces: true  
};
```

Dans l'exemple, l'objet personne a 5 propriétés : nomPrenom, age, profession, score et acces.

2022 / 2023

Langage de Script - Mohammed AMEKSA

28

Type de données en Javascript



array

Un tableau JS est une collection de valeurs, séparées par des virgules (,). Et le tout s'écrit entre crochets [].

→ Le type de données d'un tableau est un objet

```
var fruits = ["pomme", "orange", "fraise", "kiwi"];
```

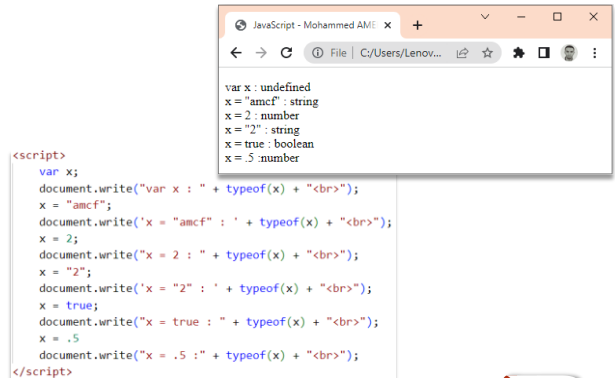
IMPORTANT

- JavaScript est un langage dynamique.
- Les variables ne sont pas directement associées à un type de données unique, contrairement à d'autres langages de programmation.
- Toute variable en JS peut être réaffectée à des valeurs de tous types

Opérateur typeof

pour connaître le type d'une valeur stockée dans une variable, on utilise généralement l'opérateur **typeof()**

→ Le type de données d'un tableau est un objet



```
<script>
var x;
document.write("var x : " + typeof(x) + "<br>");
x = "amcf";
document.write('x = "amcf" : ' + typeof(x) + "<br>");
x = 2;
document.write("x = 2 : " + typeof(x) + "<br>");
x = "2";
document.write('x = "2" : ' + typeof(x) + "<br>");
x = true;
document.write("x = true : " + typeof(x) + "<br>");
x = .5;
document.write("x = .5 : " + typeof(x) + "<br>");
</script>
```

2022 / 2023

Langage de Script - Mohammed AMEKSA

29

Manipulation – variables en JavaScript



- Créez une variable appelée **nomPrenom** attribuez-lui votre **nom** et **prénom**
- Créez une variable appelée **x**, affectez-lui la valeur **50**
- Sur une seule ligne, déclarez trois variables avec les noms et valeurs suivants :

premier = AMCF

deuxieme = business

age = 26

- Quel est la valeur de x et y ?

```
let x = 1.5 + 2 + "AMCF";
let y = "AMCF" + 1.5 + 2;
```
- Qu'affiche le script suivant ?

```
<body>
<p id="demo"></p>
<script>
var x = 2;
var x = 3;
document.getElementById("demo").innerHTML = x;
</script>
</body>
```



2022 / 2023

Langage de Script - Mohammed AMEKSA

30

Les opérateurs en JavaScript

- Les opérateurs JavaScript sont utilisés pour effectuer certaines actions particulières
- ils peuvent être utilisés pour attribuer des valeurs, comparer des valeurs, combiner des valeurs, effectuer des opérations arithmétiques et bien d'autres choses encore.
- Les opérateurs JS sont des symboles (ou un ensemble de symboles) ou des mots-clés
- il existe différents types d'opérateurs

Les opérateurs en JavaScript

L'affectation

L'opérateur d'affectation est utilisé pour affecter une valeur à une variable, en utilisant le symbole égal (=).

Opérateur	Syntaxe	Identique à	Exemple
=	x = y	x = y	a = 2
+=	x += y	x = x + y	a += 5
-=	x -= y	x = x - y	a -= 4
*=	x *= y	x = x * y	a *= 3
/=	x /= y	x = x / y	a /= 3
%=	x %= y	x = x % y	a %= 2
**=	x **= y	x = x ** y	a ** 20

```
<script>
var x = 5,
    y = 10,
    z = -2;

x += 2;
x -= 3;
y *= x;
y /= 2;
y %= x;
</script>
```


L'arithmétique

les opérateurs arithmétiques sont utilisés pour effectuer des opérations arithmétiques (l'addition, la soustraction, etc.) entre des valeurs.

Soient a = 20 et b = 10

Opérateur	Signe	Exemple
Addition	+	a + b = 30
Soustraction	-	a - b = 10
Multiplication	*	a * b = 200
Division	/	a / b = 2
Exponentiation	**	a ** b = 10240*10^9
Incrémentation	++	a ++ = 21
Décrémentation	--	a -- = 19
Modulo (reste de la division)	%	a % 3 = 2

```
<!-- autre exemple -->
<script>
  var divisor = 3,
      result1, result2, result3;
  result1 = (16 + 8) / 2 - 2;
  result2 = result1 / divisor;
  result3 = result1 % divisor;
  alert(result2);
  alert(result3);
</script>
```

Les opérateurs en JavaScript

Opérateurs de chaîne de caractères

L'opérateur de chaîne est utilisé pour concaténer ou ajouter des chaînes de caractères.

L'opérateur d'affectation par addition (+=) est l'opérateur de chaîne de caractères.

Opérateur	Description	Exemple
+	Ajouter (concaténer) des chaînes	<pre><script> let text1 = "Je suis étudiant "; let text2 = "à EMSI"; let text = text1 + text2; document.write(text); </script></pre> <div>Je suis étudiant à EMSI</div>
+=	Ajouter (concaténer) des chaînes	<pre><script> let text = "Je suis étudiant "; text += "à EMSI"; document.write(text); </script></pre> <div>Je suis étudiant à EMSI</div>

1. Réalisez un script dont vous Multipliez 6 par 5, et alertez le résultat
2. Utilisez l'opérateur d'affectation convenable afin d'obtenir la valeur 30 en x.

```
<script>
  var x = 25;
  var y = 5
  var x ... y;
  document.write(x);
</script>
```

3. A votre avis qu'affiche le programme suivant ? Et pourquoi ?

```
<script>
  let x = 5;
  x++;
  let z = x;
  document.write(z);
</script>
```



Les opérateurs en JavaScript

Opérateurs de comparaison et logiques

1. Les opérateurs de comparaison comparent leurs opérandes et renvoient un booléen selon que la comparaison est vraie ou fausse.
2. Les opérateurs logiques sont utilisés avec des valeurs booléennes, servent à déterminer la logique entre leurs opérandes.

Opérateur	Description
==	égal à
===	égale à valeur et égale à type
!=	différent de
!==	différent de (valeur ou type)
>	supérieur à
<	inférieur à
>=	supérieur ou égal à
<=	inférieur ou égal à

Opérateur	Description
&&	Et logique
	Ou logique
!	Non logique

IMPORTANT

Les opérateurs de comparaison peuvent être utilisés dans les instructions conditionnelles pour comparer des valeurs et prendre des mesures en fonction du résultat

opérateur conditionnel (ternaire)

L'opérateur conditionnel ou ternaire est le seul opérateur qui prend trois opérandes.

Syntaxe : `condition ? valeur1 : valeur2`

Si la condition est vraie, l'opérateur renvoie la valeur1 sinon il renvoie la valeur2.
Cet opérateur généralement utilisé avec des variables

Exemple :

```
<body>
  <p id='p1'></p>
  <p id='p2'></p>

  <script>
    let x = 15;
    let y = -20;
    document.getElementById('p1').innerHTML =
      x >= 10 ? 'x supérieur à 10' : 'x stric. inférieur à 10';

    document.getElementById('p2').innerHTML =
      y >= 10 ? 'y supérieur à 10' : 'y stric. inférieur à 10';
  </script>
</body>
```

Les fonctions en JavaScript



Les fonctions en JavaScript



Généralité

- Les fonctions sont l'un des éléments fondamentaux de type blocs en javascript.
- Une fonction en JS est une méthode.
- Il s'agit d'un ensemble d'instructions qui exécutent une tâche ou produisent une valeur.
- Elle est exécutée lorsqu'elle est invoquée (appelée) par quelque chose.

Syntaxe

```
function nomFonction(parametre1, parametre2, parametre3) {  
    // codes ou instructions à exécuter  
}
```

Une fonction javascript est définie ou déclarée à l'aide du mot clé **function**, suivi de :

- Le nom de la fonction
- Une liste de paramètres (facultatifs) entre parenthèses (). Les paramètres multiples sont séparés par des virgules (,).
- Le bloc de codes ou d'instructions entouré par des accolades {}.

Les fonctions en JavaScript



Exemples

```
<body>  
<h1>Les fonctions en JS </h1>  
<p id="demo"></p>  
<p id="demo2"></p>  
<script>  
    function ecrireDuTexte(str) {  
        document.getElementById('demo').innerHTML =  
            "Le texte donnée en paramètre est : " + str;  
    }  
  
    function ajoutNombres(nombre1, nombre2) {  
        var somme = nombre1 + nombre2;  
        document.getElementById("demo2").innerHTML =  
            "La somme de " + nombre1 + " et " + nombre2 + " est : " + somme;  
    }  
    /* appelant des fonctions */  
    ecrireDuTexte("Mohammed AMEKSA");  
    ajoutNombres(5, 7.5);  
</script>  
</body>
```

- Les fonctions ne seront exécutées que lorsqu'elles seront appelées ou invoquées.
- Une fonction peut être appelée à l'intérieur d'un script JS, tout comme les exemples.
- Une fonction peut être appelée lorsqu'un événement se produit, le plus souvent lorsqu'un bouton est cliqué.

Les fonctions en JavaScript

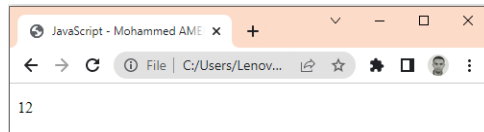


NB : veuillez noter que les fonctions ne nécessitent pas de paramètres.

Le mot-clé return

Utilisé dans une fonction pour arrêter son exécution et retourner une valeur à celui qui la demande.

```
<p id="demo"></p>
<script>
  function ajoutNombres(nombre1, nombre2) {
    var somme = nombre1 + nombre2;
    return somme
  }
  document.getElementById("demo").innerHTML =
    ajoutNombres(4, 8);
</script>
```



Les fonctions en JavaScript

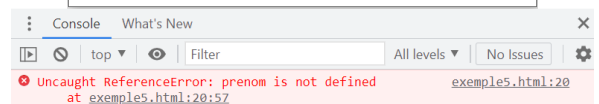
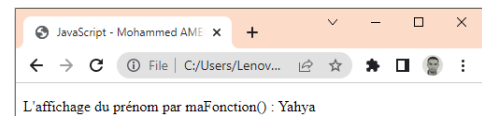


Variable locale

- Les variables déclarées à l'intérieur d'une fonction sont appelées des variables locales.
- Les variables locales ne peuvent être utilisées qu'à l'intérieur de la fonction où elles ont été déclarées.
- Si une variable locale est utilisée à l'extérieur, le type de données de la valeur est indéfini.

```
<script>
  function afficheMessage() {
    let message = "le contenu de message!";
    alert(message);
  }
  afficheMessage();
  alert(message);
</script>
```

```
<p id="demo"></p>
<script>
  function getAge(age) {
    // nomPrenom peut être utilisé ici
    var nomPrenom = "Amina AMEKSA";
    document.getElementById('demo').innerHTML =
      nomPrenom + ' a ' + age + ' ans.'
  }
  // nomPrenom ne peut pas être utilisé ici
  document.getElementById("demo").innerHTML =
    nomPrenom;
</script>
```



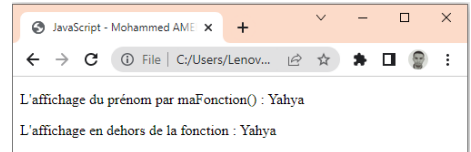
Variable globale

- Les variables globales peuvent être utilisées par n'importe quelle fonction
- Les variables globales sont des variables déclarées en dehors d'une fonction

```
<script>
  let nom = 'AMCF';
  function bienvenuMsg() {
    let message = 'Bienvenu, ' + nom;
    alert(message);
  }
  bienvenuMsg();
</script>
```

```
<p id="demo"></p>
<p id="demo2"></p>
<script>
  var prenom = "Yahya";

  function maFonction() {
    document.getElementById("demo").innerHTML =
      "L'affichage du prénom par maFonction() : " + prenom;
  }
  maFonction()
  document.getElementById("demo2").innerHTML =
    "L'affichage en dehors de la fonction : " + prenom;
</script>
```



Manipulation des fonctions en JavaScript

- Copier la fonction suivant dans un script JS

```
function maFonction() {
  alert("Bonjour tout le monde !");
}
```

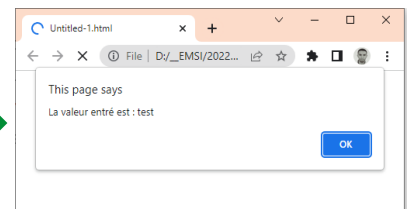
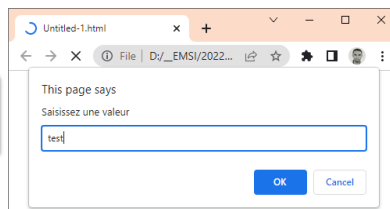
Puis essayez de l'appeler une fois l'utilisateur click sur un Button

- Créez une fonction dont vous demandez à l'utilisateur de saisir leurs nom et prénom.

- Puis affichez le msg de bienvenu 'Bienvenu nom prénom chez nous!'
- Pour récupérer les données dans cette exercice veuillez utiliser prompt()



```
<script>
  let valeur = prompt("Saisissez une valeur");
  alert("La valeur entré est : " + valeur);
</script>
```



Un autre exemple de manipulation des fonction

1. Dans un script externe, toujours à l'aide de `prompt()` vous demandez à l'utilisateur l'accord d'afficher une image de votre choix.
2. Et à l'aide des conditions ternaires,
 - s'il donne l'accord (o) vous lui affichez l'image,
 - sinon vous (n) lui affichez un message se forme d'une boîte de dialogue.

