Javascript

Où je peux écrire javascript :

Mon premier code en js :

La differnce entre window ,document et console :

La recherche d'erreur en js :

Datatypes en js (primitive et non primitive) :

Les variables (let et var) et les constantes (counst) :

La concaténation en js ;

Les opérations arithmitéques en js :

La conversion de string en number :

La converion avec d'autres méthodes (Number , parseInt , paeseFloat) :

L'objet Number en js (MAX_NUMBER_VALUE , MAX_SAFE_INTEGER , isInetger , isSafeInteger,isNan):

L'objet math en js :

La conversion du number en string (string() , toString()) :

L'objet string en js :

```
      let name='ali ';

console.log(name.repeat(3));
// ==> ali ali ali
console.log(name.length);
// ==> 4
//string is a sequence values in js
//but number isn't
//let x=100;
//console.log(x.length)
// ==> Error
//so
//let name='ali' est équivalent à (name[0]='a' ;; name[1]='a' ;; name[2]='a')
et équivalent à (name[-3]='a' ;; name[-2]='a' ;; name[-1]='a')
console.log(name.charAt(1));
// ==> l
console.log(name[10]);
// ==> undefined
console.log(name.charAt(10));
```

```
// ==> Rien n'affiche dans la console
console.log(charAt('l'));
// ==> 1
```

```
let name = 'I love javascript';
//I love javascript <==>name[0]='I';name[1]='
';name[2]='l';name[3]='o';name[4]='v';name[5]='e';name[6]='
';name[7]='j';name[8]='a';name[9]='v';name[10]='a';name[11]='s';
//name[12]='c';name[13]='r';name[14]='i';name[15]='p';name[16]='t';
console.log(name.indexOf('o',2));
// ==> 3
//la recherche débute de l'index 2
console.log(name.indexOf('java'));
// ==> 7
// l'index de 'java' est l'index du premier caractère dans le mot 'java' ,donc
l'index du 'java' est l'index du 'j'
console.log(name.lastIndexOf('i'));
// ==> 14
// lastIndexOf fait la recherche du la fin vers le début
console.log(name.slice(2,5));
// ==> lov
// la fonction slice(a,b) coupe string du (a) vers (b-1)
console.log(name.split(' '));
// la fonction split return un tableau
// ['I', 'love', 'javascript']
console.log(name.split(' ',2));
// ==> ['I', 'love']
console.log(name.split(''));
// ==> ['I', ' ', 'l', 'o', 'v', 'e', ' ', 'j', 'a', 'v', 'a', 's', 'c', 'r',
'i', 'p', 't']
```

Découpage du string en js (slice() , sustring() , substr()) :

```
let name = 'I love javascript';
//I love javascript <==>name[0]='I';name[1]='
';name[2]='l';name[3]='o';name[4]='v';name[5]='e';name[6]='
';name[7]='j';name[8]='a';name[9]='v';name[10]='a';name[11]='s';
//name[12]='c';name[13]='r';name[14]='i';name[15]='p';name[16]='t';
//I love javascript <==>name[-17]='I';name[-16]=' ';name[-15]='l';name[-
14]='o';name[-13]='v';name[-12]='e';name[-11]=' ';name[-10]='j';name[-
9]='a';name[-8]='v';name[-7]='a';name[-6]='s';
//name[-5]='c';name[-4]='r';name[-3]='i';name[-2]='p';name[-1]='t';
console.log(name.slice(-6,-4));
// ==> sc
console.log(name.substring(2,6));
// ==> love
```

```
//la difference entre entre slice() et substring() et que slice peut avoir des index positives ou négatives , mais substring() peut avoir juste les index positives
console.log(name.substr(2,10));
// ==> love javas
// la fonction substr(a,b) coupe string du le charactère (a) au caractère (a+b)
```

La recherche dans string en js :

On a découvert 2 méthode pour la recherche dans string (indexOf() et lastIndexOf()) .

Et maintenant on va découvrir d'autres méthodes comme , includes() qui retourne true ou false ; true s'il existe le string qu'on cherche , et false s'il n'éxiste pas .

Indexof() et lastIndexOf() retourne -1 s'il n'éxiste pas le string qu'on cherche .

```
let name = 'I love javascript';
//I love javascript <==>name[0]='I';name[1]=' ';name[2]='l';name[3]='o';name[4]='v';name[5]='e';name[6]=' ';name[7]='j';name[8]='a';name[9]='v';name[10]='a';name[11]='s';
//name[12]='c';name[13]='r';name[14]='i';name[15]='p';name[16]='t';
//I love javascript <==>name[-17]='I';name[-16]=' ';name[-15]='l';name[-14]='o';name[-13]='v';name[-12]='e';name[-11]=' ';name[-10]='j';name[-9]='a';name[-8]='v';name[-7]='a';name[-6]='s';
//name[-5]='c';name[-4]='r';name[-3]='i';name[-2]='p';name[-1]='t';
console.log(name.includes('o'));
// ==> true
console.log(name.includes('o',4));
// ==> false
console.log(name.startsWith('I'));
// ==> true
console.log(name.startsWith('i'));
// ==> false
console.log(name.toLowerCase().startsWith('i'));
// ==> true
console.log(name.startsWith('l',2));
// ==> true
console.log(name.startsWith('love'));
// ==> true
console.log(name.endsWith('t'));
// ==> true
console.log(name.endsWith('e',6));
// ==> true
```

Array en js :

Array is a sequence of values in js .

```
names = ['ahmed','ali','mazen','omar'];
console.log(names[1]);
// ==> ali
console.log(names.length);
// ==> 4
// on peut mettre different types de données dans un tableau
// Exemple :: let objet=[1,2,3,'ahmed',true,null,undefined,[1,2,3]];
let object=[1,2,3,[4,5,6]];
console.log(object[3][1]);
// ==> 5
```

Ajouter et supprimer des éléments dans tableau en js :

```
let names =['ahmed','ali','mazen'];
names.push('gamal');
console.log(names);
// ==> ['ahmed', 'ali', 'mazen', 'gamal']
names.push('gamal','taha');
console.log(names);
// ==> ['ahmed', 'ali', 'mazen', 'gamal', 'gamal', 'taha']
names.unshift('fadel');
console.log(names);
// ==> ['fadel', 'ahmed', 'ali', 'mazen', 'gamal', 'gamal', 'taha']
names.shift();
console.log(names);
// ==> ['ahmed', 'ali', 'mazen', 'gamal', 'gamal', 'taha']
names.pop();
console.log(names);
// ==> ['ahmed', 'ali', 'mazen', 'gamal', 'gamal']
```

Slice( ) et splice( ) dans tableau en js :

La fonction splice( ) est très important dans les tableaus , car elle nous permet de supprimer les éléments par leurs position dans le tableau .

```
names =['ahmed','ali','mazen','gamal'];
// supprimer le premier élément
names.splice(0,1);
console.log(names);
// ==> ['ali', 'mazen', 'gamal']
```

La fonction splice( ) nous permet aussi d'ajouter un ou des éléments .

```
names =['ahmed','ali','mazen','gamal'];


(names.splice(2,0,'fadel'));
console.log(names);
➔ ['ahmed', 'ali', 'fadel', 'mazen', 'gamal']

// supprimer le premier élément , et ajouter ('mona','gamal') à la place du
premier élément
(names.splice(0,1,'mona','gamal'));
console.log(names);
// ==> ['mona', 'gamal', 'ali', 'mazen', 'gamal']


//Ajouter ('mona','gamal') au début du tableau
(names.splice(0,0,'mona','gamal'));
console.log(names);
// ==> ['mona', 'gamal', 'mona', 'gamal', 'ali', 'mazen', 'gamal']

// ==> Ajouter ('mona','gamal') dans l'index 1
(names.splice(1,0,'mona','gamal'));
console.log(names);
// ==> ['mona', 'mona', 'gamal', 'gamal', 'mona', 'gamal', 'ali', 'mazen',
'gamal']
```

La fonction slice( ) a même rôle que la fonction( ) ,mais la fonction slice n'affecte pas sur le tabeau .

```
      names =['ahmed','ali','mazen','gamal'];


console.log(names.slice(0,3));
// ==> ['ahmed', 'ali', 'mazen']
console.log(names);
// ==> ['ahmed', 'ali', 'mazen', 'gamal']
console.log(names.slice(-2));
// ==> ['mazen', 'gamal']
```

La recherche dans les tableau en js :

Pour la recherche dans les tableaux on utilse indexOf( ) et lastIndexOf( ) comme dans string .

Sort( ) et reversce( ) dans les tableau en js :

```
arr =['ahmed','zyad','salem','mazen'];
```

```
console.log(arr.reverse());
// ==> ['mazen', 'salem', 'zyad', 'ahmed']
console.log(arr.sort());
// ==> ['ahmed', 'mazen', 'salem', 'zyad']
```

Concat et join dasn les tableaux en js :

```
arr1 =['ahmed','ali','mazen'];
arr2 =['gamal','karem','omar'];
arr1 += arr2 ;


console.log(arr1);
// ==> ahmed,ali,mazengamal,karem,omar
console.log(typeof(arr1));
// ==> string
```

```
arr1 =['ahmed','ali','mazen'];
arr2 =['gamal','karem','omar'];

console.log(arr1.concat(arr2));
// ==> ['ahmed', 'ali', 'mazen', 'gamal', 'karem', 'omar']
console.log(typeof(arr1.concat(arr2)));
// ==> object
```

```
arr1 =['ahmed','ali','mazen'];
arr2 =['gamal','karem','omar'];


console.log(arr1.join(arr2));
// ==> ahmedgamal,karem,omaraligamal,karem,omarmazen
console.log(typeof(arr1.concat(arr2)));
// ==> string

console.log(arr1.join());
console.log(typeof(arr1.join()));
// ==> string

console.log(arr1.join('-'));
// ==> ahmed-ali-mazen
console.log(arr1.join(';;;;;;;;;;;;;;;;;'));
// ==> ahmed;;;;;;;;;;;;;;;;;ali;;;;;;;;;;;;;;;;;mazen
```

La comparaison en js :

```
let x='ahmed';
let y='ahmed';

console.log(x==y);
// ==> true

let a='ahmed';
let b='ali';
console.log(a=b);
// ==> ali
console.log(a==b);
// ==> false

console.log(5==5);
// ==> true
console.log(5=='5');
// ==> true
//  ( == ) fait la comparaison par la valeur ,n'est par le type de donnée ,(
=== ) qui fait la comparaison par par la valeur et par le type de donnée .
console.log(5==='5');
// ==> false
// En js on a d'autres opérations pour la comparaison ( < ; > ; <= ; >= ; != )
et d'autres opérations logiques ( && (et logique) et || (ou logique))
```

La condistion (if) en js :

```
age=18;
if (age>18){
    console.log('hello user');
}
else if(age ==18 ){
    console.log('you\'re 18');
}
else{
    console.log('you\'re so young ');
}
// ==>you're 18
```

Est équivalent à

```
age=18;
age>18?console.log('hello user'):age ==18 ?console.log('you\'re
18'):console.log('you\'re so young ')

// ==>you're 18
```

Loop en js :

Nasted loop en js :

Break et continue en js :

While loop et doWhile loop :

Function :

Hoisting :

```
console.log(x);
var x = 100;
// ==> undefined .
// js voit ce code comme
//var x;
//console.log(x);
//x=100;

// ==> si pour cela la console nous donne undefined car il n'y a aucune valeur affecté au variable x ;
// si même pour les fonctions
hello();
function hello(){
    console.log('hello');
}
// ==> hello

// pour éviter ce problème (hoisting) ;pour les variables on utilise let ; et pour les fonctions on utilise 'let NomDeLaFonction = function(){}'
// Exemple
console.log(x);
let x=100;
// => Error

hello();
let hello = function(){
    console.log('hello');
}
// ==> Error
```

Scoop et self invoked en js :

```
// en js on a 2 types des variables ,global et local .

//global
//Exemple
```

```
var x=5;
console.log(x);
hello();
function hello(){
    console.log(x);
}
console.log(x);
// ==> 5
// ==> 5
// ==> 5
```

```
// mais si je mais la variable à l'intéreur de la fonction ,je ne peux
l'utiliser à l'éxtérieur de la fonction
//local
// Exemple

function hello(){
    var x=10;
}
console.log(x);
```

```
// les fonctions sont des blocs d'instructions ,et aussi les bouvles sont des
blocs d'instructions
// mais c'est différent que les fonctions , à la déclaration des variables
// Exemple
for(let x=0 ;x<2;x++){
    var y=10;
}
console.log(y);
// ==> 10
console.log(x);
// ==>  Error
// alors dans les boucles var est un variable globale et let est un variable
locale
```

```
function hello(){
    console.log('hello');
}
// ==> Rien n'affiche car on n'a pas fait l'appèlle à la fonction
// alors pour écire une fonction et l'appellé en même temps
//Exemple
(function (){
    console.log('hello');
})();
// ==> hello
```

Arrow function en js :

```
let x = function(){
    return 1;
}
//Arrow function
let x = () => {
    return 1;
}
// si j'ai une seule instruction
let x = () => 1;
// si vous n'avez pas de paramétres
let x =_ => 1;
// et si vous avez juste un paramétre
let x =(num) => num * 2 ;
```

L'objet en js :

Nasted objet en js :

Create object :

This keyword (this):

Créer un objet à base d'un autre objet :

```
let user1 ={
    name :'fadel ellah',

    getName:function(){
        return `hello ${user1.name}`;
    }
};
let user2= Object.create(user1,{
    agee:{value:20}
});
console.log(user2.name);
// ==> fadel ellah

console.log(user2.getName());
// ==> hello fadel ellah

user2.name='ali';
console.log(user2.name);
```

```
// ==> ali
console.log(user2.getName());
// ==> hello fadel ellah

// ==> pour éviter ce problème ,il faut remplacer à la méthode getName ,user1
par this

console.log(user1.name);
// ==> fadel ellah

user2.age=21;
console.log(user2.age);
// ==> 21

user2.agee=21;
console.log(user2.agee);
// ==> 21
```

Assign object en js :

```
let a1={
    num1 :1,
    hello:function(){
        console.log('hello');
    }
};
let a2={
    num2 :2,
};
let a3={
    num3 :3,
};
let a4 =Object.assign(a1,a2,a3,{
    num4 : 4,
});
```

Dom en js :

getElementById() – getElementsById() – getElementByClassName() – getElementsByClassName() – getElementByTagName() -getElementsByTagName() – querySelector() .

L'ajout et la modification d'un attribut :

```
let img = document.getElementById('img');
console.log(img.hasAttributes());
// ==> true
console.log(img.hasAttributes('src'));
// ==> true
console.log(img.hasAttributes('title'));
// ==> false
console.log(img.attributes);
// ==> NamedNodeMap {0: id, 1: src, 2: alt, id: id, src: src, alt: alt,
length: 3}
console.log(img.attributes[0]);
// ==> id='img'
img.setAttribute('alt','ahmed');
console.log(img);
// ==> <img id="img" src="" alt="ahmed" ></img>
console.log(img.alt);
// ==> ahmed
img.removeAttribute('alt');
console.log(img);
// ==> <img id="img" src="" ></img>
```

Inner et outer en js : (innerHTML() ,outerHTML() , innerText() ,outerText (),contentText()) :

Sibling et parent en js :

Dans ce chapitre on va voir deux fonctions pour sibler un élément html en js , previousElementSibling() et nextElementSibling() ,et pour sibler n'importe qu'ils éléments(quelque soit un élément html ,un commentaire ,un mot(string)) on utilise previousSibling() et nextSibling() . et pour sibler le père d'un élément j'utilise la fonction parentElement() .

DOM CSS STYLE :

```
let container = document.getElementById('container');

container.innerHTML = 'hello world';
// element.style.proprety = value ;
container.style.backgroundColor = '#444';
container.style.color = '#fa0';
container.style.padding = '10px';
container.style.barderLesf = '4px solid #fa0';
//element.style.cssText =
container.style.cssText = `
    background : '#444';
```

```
    color:'white';
`
let container = document.getElementById('container');

container.innerHTML = 'hello world';

container.setProperty('color','red','important');
```

```
let container = document.getElementById('container');

container.innerHTML = 'hello world';
// element.style.proprety = value ;
container.style.backgroundColor = '#444';
container.style.color = '#fa0';
container.style.padding = '10px';
container.style.barderLesf = '4px solid #fa0';
// remove property
container.style.removeProperty('color');
container.style.removeProperty('backgroundColor');
```

Créer des élément en js :

```
// Pour écrire html dans le code js on peut utilser plusieurs méthode
// La première méthode
document.body.innerHTML = `
    <h1> Hello world <h1>
    <h2> Hello world <h2>
    <p> Hello world <p>
`
// La deuxiéme méthode
//  1 - Création d'un élément
let container = document.createElement('div');
console.log(container);
// ==> <div></div>
let head = document.createElement('h1');
console.log(head);
// ==> <h1></h1>
let img = document.createElement('img');
console.log(img);
// ==> <img>

//  2 - L'ajout des éléments

let content = document.createTextNode('Hello Fadel Ellah ERRAMI');
head.appendChild(content);
console.log(head);
// ==> <h1>Hello World</h1>
```

```
img.src = 'inscription.png';
console.log(img);
// ==>  <img src="inscription.png">

// 3 - Lajout dés élémens dans un container

container.appendChild(head);
container.appendChild(img);
console.log(container);
// ==> <div><h1>Hello world</h1><img src="inscription.png"></div>

//  3 - L'ajout de container dans le body :

document.body.appendChild(container);
console.log(document.body);
//==>
// <body>
//    <h1> Hello world </h1><h1>
//    </h1><h2> Hello world </h2><h2>
//    <p> Hello world </p><p>
//</p></h2><div><h1>Hello Fadel Ellah ERRAMI</h1><img
src="inscription.png"></div>
//</body>
// <==
// On peut aussi ajouter du css
container.style.background='#666';
img.style.width='500px';
img.style.marginLeft='300px';
```

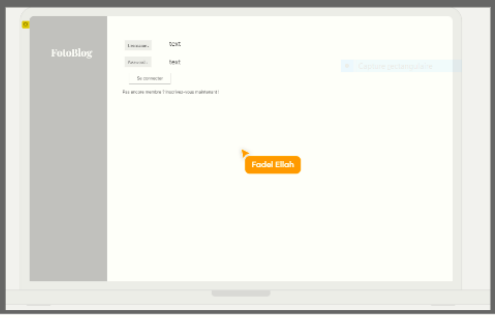
Resultat :

Hello world

Hello world

Hello world

Hello Fadel Ellah ERRAMI

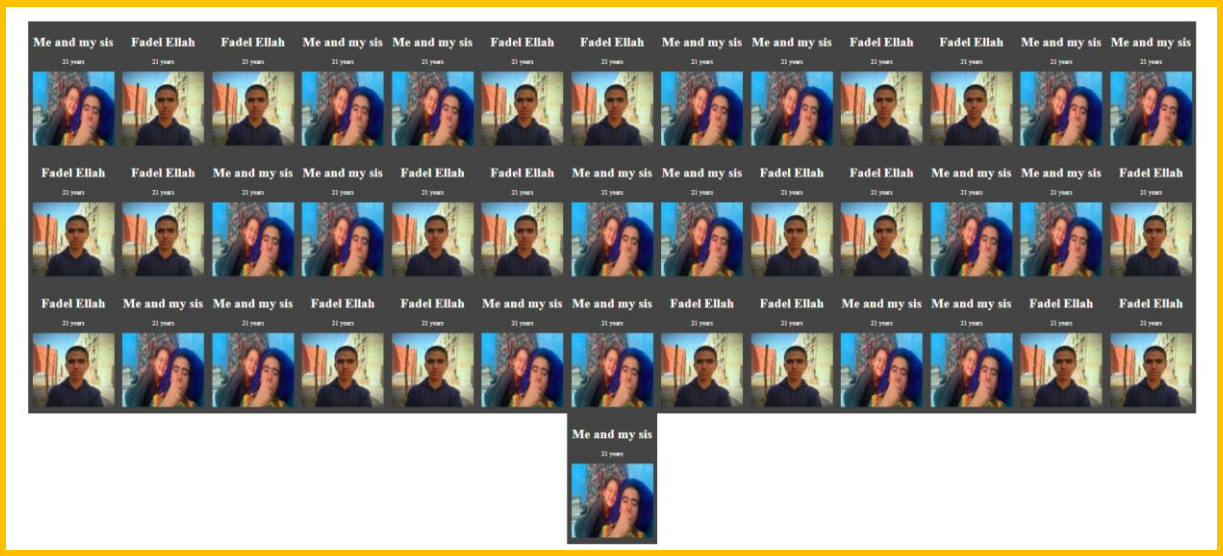


Un petit projet en js :

```
let container = document.createElement('div');
document.body.appendChild(container);
container.style.textAlign ='center';
function element(Head,ContentAge,ImgSRC){
    //elements
    let cards = document.createElement('div');

    let title = document.createElement('h1');
    let age = document.createElement('p');
    let img = document.createElement('img');
    cards.appendChild(title);
    cards.appendChild(age);
    cards.appendChild(img);
    //content
    head = document.createTextNode(Head);
    contentAge = document.createTextNode(ContentAge);
    img.src = ImgSRC;


    //Add content to elements
    title.appendChild(head);
    age.appendChild(contentAge);
```

```
    // add cards to container
    container.appendChild(cards);
    // style
    document.body.style.padding ='0px';
    document.body.style.margin='0px';
    cards.style.width='200px';
    cards.style.height='300px';
    cards.style.background='#444';
    cards.style.color='#fff';
    cards.style.padding='10px';
    //cards.style.margin='2px';
    cards.style.display='inline-Block';
    img.style.width='100%';
    img.style.height='60%';


}
for(let i =  0 ; i<10 ; i++)
{
    element('Me and my sis','21 years','me and my sis.jpg');
    element('Fadel Ellah','21 years','erramii.jpeg');
    element('Fadel Ellah','21 years','erramii.jpeg');
    element('Me and my sis','21 years','me and my sis.jpg');
}
```

Résultat :



Events en js (addEventListener(Event) vs onEvent) :

Mouse Events :

onclick() ; onmouseup() ; onmousedown() ; onmouseover() ; onmouseout() ; onmousemove() ; ...

Keyboard Events :

onkeyup() ; onkeydown ; onfocus() ; onblur() ; ...

Window Events :

onload() ; resize() ; ...

Un petit projet en js :

Code html :

```
<body>
    <input type="number" placeholder="USD" id="dollar"></input>
    <input type="number" placeholder="MAD" id="dirham"></input>

 <script src="main.js"></script>
</body>
```

Code js :

```
let dollar = document.getElementById('dollar');
let dirham = document.getElementById('dirham');

dollar.onkeyup = function(){
    dirham.value = dollar.value * 10;
}
dirham.onkeyup = function(){
    dollar.value = dirham.value / 10;
}
```

Résultat :

Before et after et append en js :

Code html :

```
<body>
    <button id="after" >after</button>
    <button id="before" >before</button>
    <button id="inside" >inside</button>

    <p id="content" >content</p>

    <div id="container">

    </div>

 <script
```

Code js :

```
after = document.getElementById('after');
before = document.getElementById('before');
inside = document.getElementById('inside');
content=document.getElementById('content');
container=document.getElementById('container');

container.style.background='#ffa';
container.style.height='50px';
after.onclick=function(){
    container.after(content);
}
before.onclick=function(){
    container.before(content);
}
inside.onclick=function(){
```

```
    container.append(content);
}
```

Add et remove et toggle une classe en js :

Aut event en js :

BOM en js :

Scroll en js :

Screen en js :

Location object en js :

SetTimeOut et setInterval en js :

LocalStotage et seesionStorage en js :