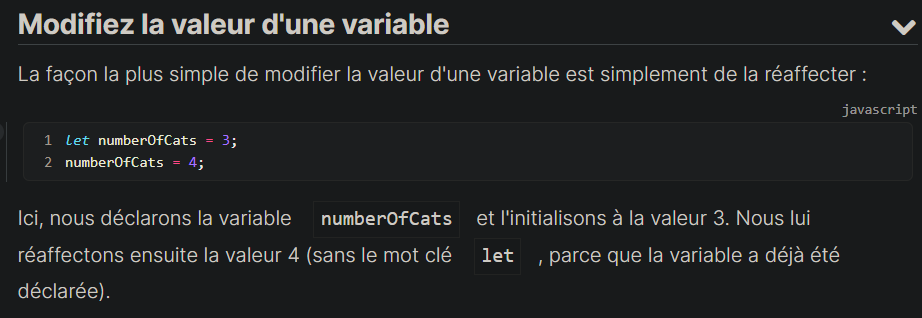
Javascript :

Déclarer une variable :

Let variableName = value ;

Une image contenant texte, intérieur, orange, sombre

Description générée automatiquement



Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Déclarer une constante :



Une image contenant texte

Description générée automatiquement

Object :

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Console.log() == print()

**== and ===**

When the equality operators, == and ===, are used on reference-type variables, they check the reference. If the variables contain a reference to the same item, the comparison will result in true.

var arrRef = [’Hi!’];  
var arrRef2 = arrRef;console.log(arrRef === arrRef2); // -> true

If they’re distinct objects, even if they contain identical properties, the comparison will result in false.

var arr1 = ['Hi!'];  
var arr2 = ['Hi!'];console.log(arr1 === arr2); // -> false

== when comparing primitive types, it compares the value, but when comparing non primitive types, it checks the reference.

If we have two distinct objects and want to see if their properties are the same, the easiest way to do so is to turn them both into strings and then compare the strings. When the equality operators are comparing primitives, they simply check if the values are the same.

var arr1str = JSON.stringify(arr1);  
var arr2str = JSON.stringify(arr2);console.log(arr1str === arr2str); // true

function changeAgePure(person) {  
 var newPersonObj = JSON.parse(JSON.stringify(person));  
 newPersonObj.age = 25;  
 return newPersonObj;  
}var alex = {  
 name: 'Alex',  
 age: 30  
};var alexChanged = changeAgePure(alex);console.log(alex); // -> { name: 'Alex', age: 30 }  
console.log(alexChanged); // -> { name: 'Alex', age: 25 }

In this function, we use JSON.stringify to transform the object we’re passed into a string, and then parse it back into an object with JSON.parse. By performing this transformation and storing the result in a new variable, we’ve created a new object.

A = [] -> let A = [];

Len(A) -> A.length;

A.append(x) ->A.push(x);

A.insert(0,x) -> A.unshift(x); #add x on the front of the array.

Delete last element; A.pop();

EQUIVALENT DE LA FONCTION MAIN EN JAVASCRIPT :

En JS BIN :

JSBin parcourt alors vos lignes de code et les exécute l'une après l'autre. L'ordre est donc important.

En page web :

Une image contenant texte, écran, capture d’écran

Description générée automatiquement

C'est plus ou moins de cette façon que JavaScript fonctionne sur le web. Vous écrivez du code, l'enregistrez dans un ou plusieurs fichiers, puis **importez** ces fichiers par une balise spéciale dans votre fichier HTML. Le navigateur **exécute** ensuite automatiquement le code dans ces fichiers, en général dans l'ordre de leur importation.

If/else comme dans java :

If(condition){

}else if{

}else{

}

Une image contenant texte

Description générée automatiquement

Une image contenant texte, capture d’écran, orange, fermer

Description générée automatiquement

Difference between var and let :

var and let are both used for variable declaration in javascript but the difference between them is that var is function scoped and let is block scoped. Variable declared by let cannot be redeclared and must be declared before use whereas variables declared with var keyword are hoisted.

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Une image contenant texte

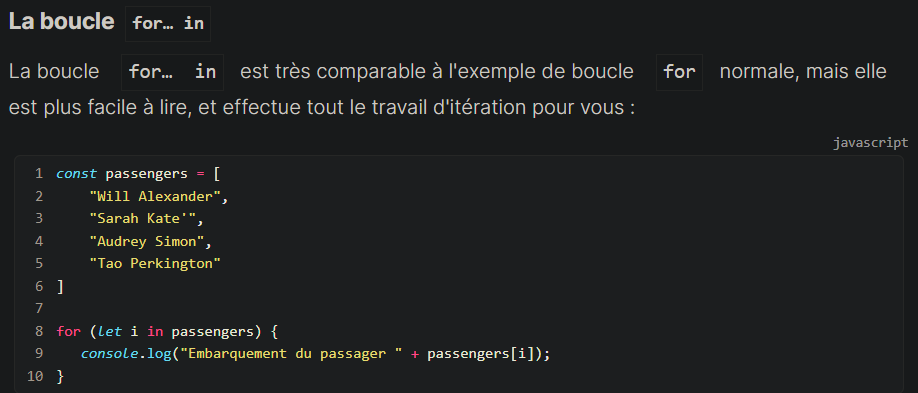
Description générée automatiquement

Utiliser switch :

Une image contenant texte

Description générée automatiquement

La boucle for .. in



La boucle for .. of

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

La boucle while :

While(condition){

Bloc instruction;

}

Gérer les erreurs:

Une image contenant texte, capture d’écran, écran, noir

Description générée automatiquement

Les fonctions en javascript:

Une image contenant texte

Description générée automatiquement

Méthode d’instance :

Une image contenant texte

Description générée automatiquement

exemple

Une image contenant texte

Description générée automatiquement

Une image contenant texte, capture d’écran, écran

Description générée automatiquement

Méthodes statiques :elles ne sont pas liées à une instance particulière d’une classe, mais à la classe elle-même. On peut les utiliser sans instancier la classe.

Syntaxe : nomDeLaClasse.nomMethodeStatique(‘arg’) ;

Une image contenant texte

Description générée automatiquement

Une fonction propre = simple, facile à comprendre, n’effectue qu’une seule tâche.

Une image contenant texte

Description générée automatiquement

Une image contenant texte, capture d’écran, écran

Description générée automatiquement

Test unitaire :

Une image contenant texte

Description générée automatiquement

On teste souvent pour chaque fontion le cas normal et des cas limites. Exemple :

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Les tests d’intégration :

Une image contenant texte

Description générée automatiquement

Test fonctionnels : Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Déboguer des fonctions :

Une image contenant texte

Description générée automatiquement

Recherche binaire : Une image contenant texte

Description générée automatiquement