|  |  |
| --- | --- |
| **BTS SIO** | **BTS SIO (SLAM)** |
|  | **Bloc 3** |
|  |  |
| **SUPPORT DE COURS Bloc 1** | |



**JavaScript**

|  |  |  |  |
| --- | --- | --- | --- |
| **date** | **révision** | | |
|  | Timothée Robert | | |
|  |  | | |
|  |  | | |
|  |  | | |
|  |  | | |
|  |  | | |
|  |  | | |
| **BTS SIO** | |  | **BTS SIO (SLAM)** |
|  | | |  |
| INTRODUCTION | | |  |

JavaScript est un langage inventé par Brendan Eich en 1995 qui sert à programmer des traitements côté "client" (on dit Front pour frontend en anglais) dans les développements web. La première version de ce langage avait été baptisée LiveScript. Après avoir été renommé en Javascript il a été standardisé par l’ECMA (<http://www.ecma-international.org/> organisation industrielle chargée de standardiser les systèmes de communication et d’information).

# FONCTIONNEMENT

## Utilisation

### En complémentarité du HTML

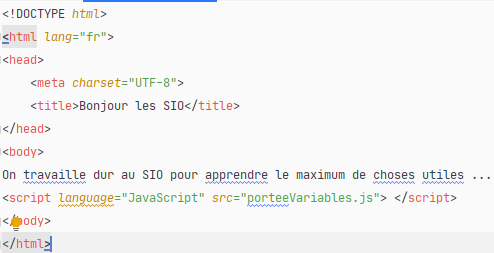
Le langage JavaScript s’utilise à l’intérieur de pages HTML. C’est un langage qui n’est pas typé par nature, même si une de ses extensions les plus utilisées, TypeScript, propose un typage strict.

C’est le rôle de la balise SCRIPT qui indique au navigateur Web l’utilisation du langage JavaScript.

### Emplacement de la balise SCRIPT

La balise peut être placée à l’intérieur de la balise HEAD ou de la balise BODY.

Le meilleur emplacement est dans le BODY juste avant la balise de fin, comme ci-dessous.



### Code dans un fichier séparé ou dans la page

Comme pour le CSS (feuilles de style) on peut écrire le code à l’intérieur de la page HTML ou dans un fichier séparé (comme dans l’exemple précédent).

## Utilité

L’intérêt du langage JavaScript est d’apporter des possibilités d’interaction et de dynamisme à une page Web sans aller-retour vers un serveur, comme c’est le cas avec le langage PHP.

L’ensemble du code est chargé dans la page Web et permet d’éviter un serveur d’application.

De nombreuses librairies (bibliothèque en français[[1]](#footnote-1)) et frameworks [[2]](#footnote-2) ont été développés avec le langage JavaScript.

# Le langage JavaScript

## Les commentaires

Comme en Java et en PHP :

* // pour commenter la ligne uniquement
* Bloc de commentaire encadré par /\* (début) et \*/ (fin) pour mettre en commentaire une ou plusieurs lignes contiguës

## Déclaration de variables

JavaScript distingue 3 types de variable, les références constantes, les variables locales et globales.

### Variable locale

L’instruction **const** sert à déclarer une référence constante dont la ***portée*** est locale au bloc ou la fonction dans laquelle elle est utilisée.

L’instruction **let** permet de déclarer une variable dont la ***portée*** est locale au bloc ou la fonction dans laquelle elle est utilisée.

L’instruction **var** permet de déclare une variable de ***portée*** globale c’est-à-dire qu’elle sera connue et accessible dans tout le script où elle est déclarée.

En pratique on privilégie d’abord **const** puis **let** puis **var** : <https://code-garage.fr/blog/quelles-sont-les-differences-entre-var-let-et-const-en-javascript>

**Exemple :** cf le script porteeVariable.js

Le mot clef ***this*** permet de désigner explicitement une variable globale.

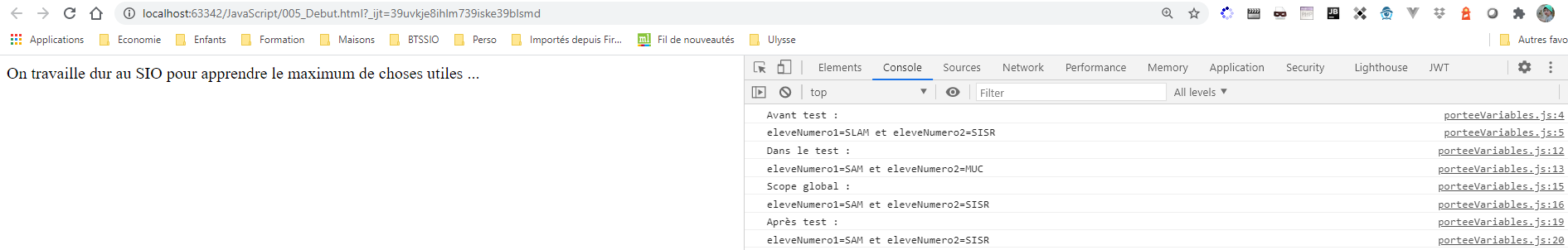
### Ecriture en sortie

#### Console

L’accès à la console se fait vie la méthode (fonction) log de l’objet Console : **console.log** permet d’écrire un message dans la console du navigateur Web.

L’accès à la console se fait vit la touche F12 en haut du clavier.

Exemple : affichage dans la console accédée avec la touche F12 dans le navigateur Chrome



#### Avertissement utilisateur

La fonction ***alert****(texte)* (raccourci de window.alert) affiche un dialogue d'alerte contenant le texte spécifié.

#### Ecriture dans un document HTML

L’accès au document HTML courant se fait via l’instruction document.write.

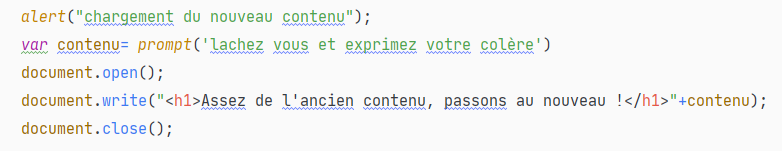
Exemple : cf le script ecrireDocument.html issu de la référence <https://developer.mozilla.org/fr/docs/Web/API/Document/write>

#### Récupération de données saisies par l’utilisateur

La méthode (fonction) prompt permet de récupérer des données saisies par l’utilisateur.

Le résultat peut être placé dans une variable.

Exemple :



## Tableaux

### Introduction

On retrouve les mêmes types de tableaux qu’en PHP. La principale différence est que la boucle foreach a une syntaxe légèrement différente en JavaScript.

Les tableaux s’ils ne sont pas modifiés peuvent être déclarés comme constante vie l’instruction ***const***

### Tableaux simples

L’accès à chaque donnée du tableau se fait au moyen d’un indice, qui comme en Java et PHP, commence à la valeur 0.

Le langage JavaScript fournit plusieurs façons de créer un tableau :

* la syntaxe littérale,
* la syntaxe dite "Programmation orientée objet".

Avec une syntaxe littérale, la déclaration se fait comme suit :



Avec une syntaxe "Programmation orientée objet", on écrit :



Rmq : items est automatiquement affiché par l’IDE PHPStorms, c’est absent du langage, c’est un helper, une fonctionnalité proposée par un IDE pour aider le programmeur à comprendre, documenter et maintenir son programme.

### Tableaux associatifs

Un tableau associatif est une liste d’association de clefs et de valeur. La clef est unique.

Un tableau associatif JavaScript est très proche du format JSON.

La clef est entre double quotes et séparée de sa valeur associée par le caractère :

Chaque couple clef-valeur est séparé par une ,

Exemple : tableau associatif JavaScript codePostal



### Parcours de tableaux

#### Introduction

On retrouve les outils habituels de parcours de tableaux : boucle for, boucle while et boucle de type foreach qui ici aussi s’appelle for avec une syntaxe différente

#### Boucle for traditionnelle

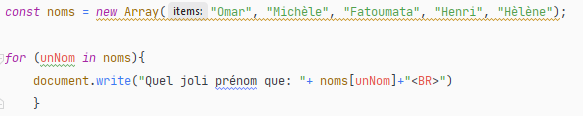
Exemple :



Remarques : const déclare une constante. Length est la propriété d’un tableau qui indique le nombre d’éléments présents dans ce tableau.

#### Boucle for type foreach

Exemple : tableau simple



La syntaxe est différente de celle du PHP et de Java. On trouve le mot clef in à l’intérieur de l’instruction for. La variable à gauche du in contiendra l’indice ou bien la clef si c’est un tableau associatif.

Exemple : tableau associatif



## Les instructions conditionnelles

### Opérateurs de comparaison

Les opérateurs de comparaison permettent de comparer 2 termes nommés opérandes. Ils sont utilisés à l’intérieur de conditions.

Principaux opérateurs de comparaison :

|  |  |
| --- | --- |
| Opérateur | Signification |
| < ou > | Plus petit ou plus grand : if (a < b) ou bien if (a > b) |
| <= | Plus petit ou égal : if (a <= b) |
| >= | Plus grand ou égal |
| == | Égal : if (a == b) Renvoie true si les opérandes sont égaux après conversion en valeurs de mêmes types. |
| != | Différent : if (a != b) |
| === | Égal en valeur ET en type : if (age === 50) Renvoie true si les opérandes sont égaux et de même type |

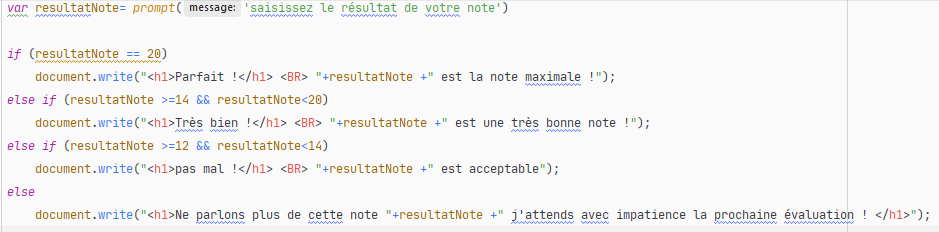
### Condition if … else

Comme dans les autres langages.

Chaque condition, à l’intérieur des parenthèses des instructions if, else if … est évaluée successivement et dès qu’une condition est vraie l’instruction correspondante est exécutée et le programme passe à la suite.

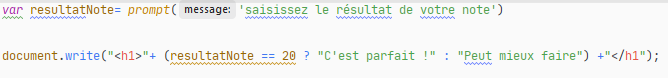
Dès qu’une condition est vraie les suivantes (else if et else) ne sont pas évaluées.

Exemple : instructions if … else if … et else



### Opérateur ternaire

Comme en PHP on trouve également l’opérateur ternaire de la forme condition ? resultat1 : resultat2



Explication :

Si l’utilisateur saisit la valeur 20 l’affichage sera « C’est parfait » sinon ce sera « Peut mieux faire ».

### Autres opérateurs

Opérateurs logiques

|  |  |  |
| --- | --- | --- |
| Opérateur | Usage | Signification |
| && | expr1 && expr2 | ET logique. Renvoie true si expr1 et expr2 valent true, false sinon |
| || | expr1 || expr2 | || renvoie true si l'un des opérandes vaut true, si les deux valent false, il renvoie false |
| ! | ! Expr | Renvoie false si son unique opérande peut être converti en true, sinon il renvoie true |

Opérateurs Arithmétiques

|  |  |  |
| --- | --- | --- |
| Opérateur | Usage | Signification |
| % | expr1 % expr2 | Reste. Renvoie le reste entier de la division entre les deux opérandes. |
| ++ | ++ index | Opérateur unaire. Ajoute un à son opérande. S'il est utilisé en préfixe (++x), il renvoie la valeur de l'opérande après avoir ajouté un, s'il est utilisé comme opérateur de suffixe (x++), il renvoie la valeur de l'opérande avant d'ajouter un. |
| -- | -- index | Opérateur unaire. Il soustrait un à son opérande. Il fonctionne de manière analogue à l'opérateur d'incrément. |
| \*\* | base \*\* puissance | Calcule un nombre (base) élevé à une puissance donnée |

## Fonctions JavaScript

### Introduction

Comme en PHP, les fonctions JavaScript peuvent ne rien retourner, on dit que ce sont des procédures ou alors renvoyer une variable, ce sont alors de véritables fonction au sens informatique du terme.

### Fonction simple

La syntaxe est la suivante :

function nomDeFonction(parametres) {

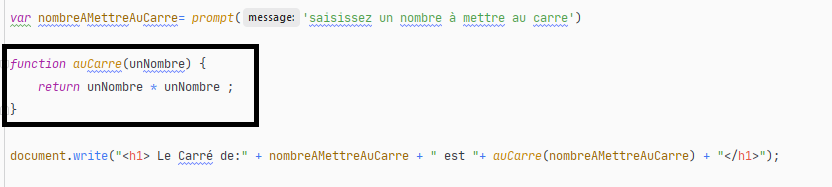
// instructions

// return éventuel

}

Les parametres éventuels sont séparés par des virgules.

Exemple :



La fonction auCarre prend en paramètre un nombre et renvoie son carré

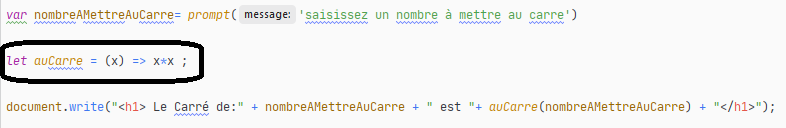
### Fonction fléchée

Les nouvelles versions de JavaScript proposent une notation plus allégée et permettant de proposer des fonctions anonymes si elles n’ont pas vocation à être utilisées partout dans le programme.

Syntaxe :

let nomFonction = (paramètres) => resultat ;

Exemple : carré d’un nombre



C’est la même fonction que précédemment écrite sous forme de fonction fléchée.

1. En [informatique](https://fr.wikipedia.org/wiki/Informatique), une **bibliothèque logicielle** est une collection de [routines](https://fr.wikipedia.org/wiki/Routine_(informatique)), qui peuvent être déjà [compilées](https://fr.wikipedia.org/wiki/Compilateur) et prêtes à être utilisées par des [programmes](https://fr.wikipedia.org/wiki/Programme_informatique) : source Wikipedia [↑](#footnote-ref-1)
2. un ***framework*** (appelé aussi **infrastructure logicielle**, **socle d'applications**, **infrastructure de développement**, ou **cadre d'applications** ) désigne un ensemble cohérent de [composants logiciels](https://fr.wikipedia.org/wiki/Composant_logiciel) structurels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d’une partie d'un [logiciel](https://fr.wikipedia.org/wiki/Logiciel) : source Wikipedia [↑](#footnote-ref-2)