Aide-Mémoire JS



Lucas Simões Pólvora

MID2

2023-2024

Table des matières

[1. Introduction 3](#_Toc150860330)

[2. Syntaxe 3](#_Toc150860331)

[2.1 Mettre un script JS dans un fichier HTML 3](#_Toc150860332)

[2.2 Commentaires 3](#_Toc150860333)

[2.3 Variables et Constantes 3](#_Toc150860334)

[2.4 Interactions : Alert, prompt, confirm 3](#_Toc150860335)

[2.5 Conversion de types 3](#_Toc150860336)

[2.6 Opérateurs de bases de mathématique 3](#_Toc150860337)

[2.7 Les conditions (if, else, opérateur ternaire) 3](#_Toc150860338)

[2.8 Opérateurs logiques 3](#_Toc150860339)

# Introduction

# Fondamentaux

## Mettre un script JS dans un fichier HTML

Nous avons 2 choix :

* Utiliser la balise script (Exemple : <Script> le code </script>)
* Utiliser un lien (Exemple : <Script src= « /chemin/vers/script.js »></script>

## Commentaires

Les commentaires sont écrits comme en C#. Ce qui veut dire qu’il faut introduire par // et ensuite écrire le commentaire.

## Variables et Constantes

Le JavaScript est un langage typé dynamiquement, ce qui veut dire qu’il existe des types de données, mais que les variables ne sont liées à aucun d’entre eux, donc on utilise « let » pour déclarer une variable. Le nom d’une variable doit être précis et doit montrer à quoi elle sert. Une variable peut être utilisée comme en C#.

Une constante est exactement comme en C#, elle ne peut pas changer de valeur, sa valeur est déclarée avec la constante. On utilise une écriture en majuscules seulement pour montrer que c’est une constante.

Exemples :

Variable : let jeSuisUneVariable ;

Constante : const JE\_SUIS\_UNE\_CONSTANTE ;

## Interactions : Alert, prompt, confirm

Alert(Message) : Affiche une fenêtre modale avec un message et attend que l’utilisateur appuie sur « OK »

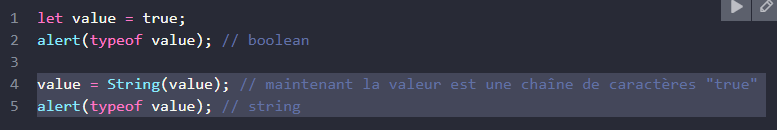
Prompt (Titre, valeur initiale du champ d’entrée(facultatif)) : Affiche une fenêtre avec un titre, un champ de saisie qui peut ensuite être dans une variable (let variable = prompt(titre)).

Confirm(question) : Affiche une fenêtre modale avec une question et deux boutons : « OK » et « annuler ». Le résultat est « true » si on appuie sur « OK » et « false » si on clique sur annuler.

## Conversion de types

### Conversion en string

### On peut convertir un bool en string. Cependant elle sera égale soit à « true », soit à « false »

Exemple : 

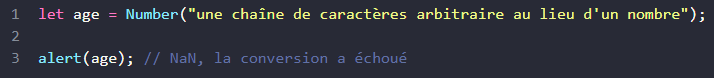
### Conversion numérique

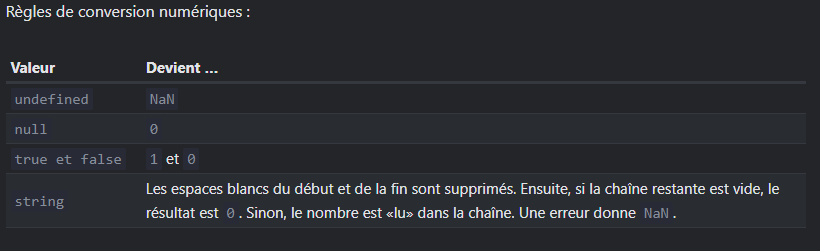
La conversion numérique dans les fonctions et expressions mathématiques s’effectue automatiquement.

Lorsque la division est appliquée à des non-numéros :



Cependant, si la chaîne de caractères est invalide, le résultat de la conversion sera NaN (Not a Number)

Exemple :



### Conversion Booléenne

Pour faire simple, si la chaîne de caractères est vide («») la valeur est « false », s’il y a qqch comme un espace ou n’importe quel caractère c’est « true »

## Opérateurs de bases de mathématique

Les opérateurs mathématiques de base sont les mêmes qu’en C#.

+ => Additionner

- => Soustraire

\* => Multiplier

/ => Diviser

% => Modulo

Il y a juste l’exponentiation qui se note « \*\* » c’est la même chose que les puissances en mathématiques

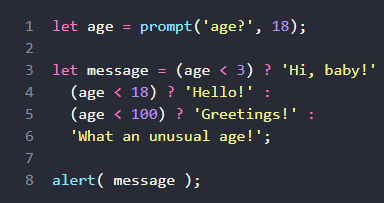
## Les conditions (if, else if, else, opérateur ternaire)

La règle est la même qu’en C#.

If(condition){Code} else if(condition){code} else{code}

Let result = condition ? valeur 1(si vrai) : valeur 2(si faux)

Nous pouvons enchaîner les opérateurs ternaire si besoin :  
Exemple :



## Opérateurs logiques

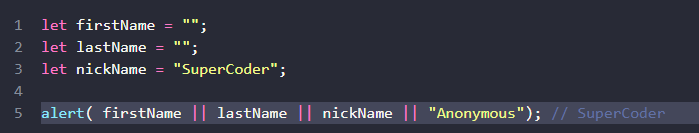
Même chose qu’en C#

|| = ou

&& = et

Elle peut aussi être utilisé pour obtenir la première valeur vraie

Exemple :



## Opérateur de coalescence des nuls (??)

Prend la première valeur « true »

Exemple : a ?? b (si a est défini alors a, si a n’est pas défini, alors b)

## Boucles (while, for, do…while)

La boucle while et do…while est comme en C#, il s’écrit de la manière suivante :

While (condition) {code (appelé Loop body)}

Do { code } while (condition)