Pense-bête javascript

Louis Arys

20 novembre 2018

Table des matières

1 Introduction : **1**

2 Les spécificités du langage Javascript : **1**
2.1 Les expressions True/False : 2
2.2 Les déclarations : 2
2.3 Les fonctions : 3

3 Fonctions Utiles : **4**
3.1 les fonctions de casts : 4
3.2 les fonctions de "Print" : 4

4 Les formulaires : **4**

1 Introduction :

Ce document à pour but de synthétiser les différentes notions nécessaire pour le cours "logique et programmation" de première bac à l'Ephec en Technologie de l'informatique. Ce n'est en aucun cas un document professionnel, mais seulement une synthèse. Bonne lecture.

2 Les spécificités du langage Javascript :

Le langage Javascript est très permissif en comparaison d'autres langages tel que le Java ou le Python. Pour faire simple, lorsq'une variable est initialisée, on ne lui attribue pas de type. La variable ne sera typée que lorqu'on lui assignera une valeur. Ainsi donc, le type d'une variable en Javascript peut aussi changer en cours d'éxécution du programme.

Par exemple : si j'assigne la valeur 5 à la variable "test", elle aura pour type *number*. Par contre si en cours de programme je lui donne comme valeur "framboise", la variable "test" sera de type *String*.

Cela n'aurait pas été possible en Java par exemple.

Pour vérifier le type d'une variable, il faut passé par l'opérateur **typeof**.
exemple :

```
function printAll(message){
        console.log(typeof message); //affichera en console le type du paramet
}
```

2.1 Les expressions True/False :

En Javascript, il existe plusieurs expressions particulières pouvant être évaluées telles quelles comme des booléens.

Les expressions False :

— *Le number 0*
— *Le String vide ""*
— *Le boolean false*

Les expressions True :

— *Les numbers autres que 0*
— *Les strings non-vide*
— *le boolean true*

Voici la fin de ce petit résumé sur les expressions true/false.

2.2 Les déclarations :

Pour utiliser une variable ou une fonction, il faut tout d'abord la déclarer (l'initialiser) dans le code. La manière de déclarer une variable change en fonction du langage de programmation utiliser. En javascript, ce sera fait de la manière suivante :

```
// Cette declaration permet d'eviter de redeclarer la variable plus loin dans
// Son utilisation est fortement conseillee.
let variable;
// Cette declaration est beaucoup plus permissive que l'autre, a eviter.
var otherVariable;

function maFonction(parametre1, parametre2, parametre3){
        //contenu de la fonction
}
```

Il peut parfois être nécessaire de déclarer une variable dont la valeur ne doit pas changer durant l'exécution du programme à savoir une **constante**. Pour se faire, il faut utiliser le mot-clé *const*.

Exemple :

```
const CONSTANTE = 25;
```

Lorsqu'une variable est déclaré comme constante, si vous essayez de modifier la valeur qu'elle contient, une erreur sera affichée en console, et la valeur contenue dans la variable restera inchangée.

2.3 Les fonctions :

Une fonction est composée de 3 parties : son nom, ses paramètres, et son contenu.

- Le nom : le nom de la fonction permet de plus tard faire appel à celle-ci et ainsi de l'exécuter dans le script.
- Les paramètres : ce sont des données passée à la fonction lors de son appel dans le script. En Javascript, les paramètres sont passé par adresse. Cela veut dire que si on *lie* un paramètre à une variable, si on modifie le paramètre, on modifie aussi la variable. Vous pouvez voir les paramètres comme des références vers d'autres variables (Ce n'est peut être qu'une approximation, mais ça me semble plus facile pour la visualisation). Un paramètre n'est visible que dans la fonction elle-même et sera initialisé automatiquement lors de l'appel de la fonction.
- Le contenu : l'ensemble des instructions se trouvant dans la fonction.

Ci-dessous se trouve une exemple de la déclaration d'une fonction et de son contenu, ainsi qu'un appel classique de fonction :

```
/*
* Le nom de ma fonction est : maFonction. Ce nom sera utile pour faire appel
* Les parametres de ma fonction sont : roger, norbert, josephine.
Ils seront initialises lors de l'appel futur de la fonction, et c'est aussi a
* Le contenu de cette fonction est tout le reste ; ce qui se trouve entre les
*/

function maFonction(roger, norbert, josephine){
        let uneVariable = 0;
        let uneAutreVariable = "bonjour";

        for(let i = 0 ; i < 10 ; i++){
                uneVariable++;
        }

        console.log(uneAutreVariable);
        console.log(roger+norbert+josephine);

        return uneVariable;
}

// Ci-dessus, on retrouve deja un appel de fonction avec "console.log(parametr
// Pour donner un exemple plus concret, ci-dessous je vais faire appel a ma fo
```

```
// Je declare une variable que je compte passer a ma fonction
let roberto = "Hello world";

maFonction(roberto, " it's a great ", "sunny day");

// lors de cette appel, on va attribuer les valeurs ci-dessus aux 3 parametres
Dans le cas de roger, on va en fait avoir : roger = roberto = "hello world".
```

3 Fonctions Utiles :

3.1 les fonctions de casts :

Le casting est le fait de changer le type d'une variable, en un autre type. Par exemple tranformer un "5" (`type String`) en 5 (`type number`). Pour se faire, il existe plusieurs fonctions en Javascript, ainsi que des mécanismes de casts implicite.

Liste des fonctions de casts :

—

—

3.2 les fonctions de "Print" :

Différentes fonctions peuvent être utilisées pour afficher une information à l'écran :

- La fonction `console.log()` : permet d'afficher le message passé en paramètre dans la console de dévellopement du navigateur.
- La fonction `alert()` : permet d'afficher le message passé en paramètre dans un pop-up.
- Les `formulaires` : permet d'afficher le message directement dans la page web, mais nécessite un peu plus de manipulations pour avoir un résultat potable.

4 Les formulaires :