Introduction au JavaScript

**HISTORIQUE DU DOCUMENT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Indice** | **Auteurs** | **Modification** |
| 13/05/14 | 00 | A. RMINECHE | Création du document |
|  |  |  |  |
|  |  |  |  |

# INTRODUCTION

## But du document

Ce document a pour but de résumer les principales bases à connaitre afin de programmer en Javascript.

## Documentation

### Documents applicables

|  |  |  |
| --- | --- | --- |
| Nom du document | Description | Version |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

### Documents de référence

|  |  |  |
| --- | --- | --- |
| Nom du document | Description | Version |
|  |  |  |
|  |  |  |
|  |  |  |

# Le Javascript :

### Déclaration des variables en Javascript :

La déclaration des variables en javascript est précédée du mot « var », le nom de la variable et l’initialisation qui définit le type de la variable.

**var** myVariable1, myVariable2 = 4, myVariable3;

**var** text1 = "Mon premier texte"; *// Avec des guillemets*

**var** text2 = 'Mon deuxième texte'; *// Avec des apostrophes*

**var** hi = 'Bonjour', name = 'toi', result;

result = hi + name;

alert(result); *// Affiche : « Bonjourtoi »*

### La conversion des variables :

Convertir une variable de type char en int :

**var** text = '1337', number;

number = parseInt(text);

Convertir une chaîne de caractère en nombre :

* parseInt(chaine,base)
* parseFloat()

### Saisir avec le clavier :

La fonction qui permet de récupérer les données saisies au clavier est la suivante :

prompt('Entrez un prénom :');

### Les objets et les tableaux :

#### Les tableaux :

La déclaration des tableaux s’effectue de la manière suivante :

**var** myString = 'Ceci est une chaîne de caractères';

**var** myArray = ['Sébastien', 'Laurence', 'Ludovic', 'Pauline',

'Guillaume'];

**var** myArray\_a = [42, 12, 6, 3];

**var** myArray\_b = [42, 'Sébastien', 12, 'Laurence'];

**var** myArray = **new** Array('Sébastien', 'Laurence', 'Ludovic','Pauline', 'Guillaume');

alert(myArray[1]);

**for** (**var** i = 0; i < myArray.length; i++) {

alert(myArray[i]);

}

Déclaration d’un tableau de donnée :

**var** myArray = [

{

nick: 'Sébastien',

age: 23,

sex: 'm',

parent: 'aîné',

work: 'Javascripteur'

},

{

nick: 'Laurence',

age: 19,

sex: 'f',

parent: 'soeur',

work: 'Sous-officier'

},

*// et ainsi de suite…*

];

**var** myArray = [

**new** Person('Sébastien', 23, 'm', 'aîné', 'Javascripteur', []),

**new** Person('Laurence', 19, 'f', 'soeur', 'Sous-officier', []),

**new** Person('Ludovic', 9, 'm', 'frère', 'Etudiant', []),

**new** Person('Pauline', 16, 'f', 'cousine', 'Etudiante', []),

**new** Person('Guillaume', 16, 'm', 'cousin', 'Dessinateur', []),

];

Récupérer un caractère en fonction de sa position dans une chaine de caractère :

**var** first = myString.charAt(0);

**var** myArray = **new** Array(longueurdutableau);

rajouter un élément dans un tableau : push()

retire et retourne le dernier élément d’un tableau : pop()

ajout un élément au début du tableau : unshift()

retirer le premier élément d’un tableau :shift()

#### Les objets littéraux :

Exemple de création des objets :

**var** family = {

self: 'Sébastien',

sister: 'Laurence',

brother: 'Ludovic',

cousin\_1: 'Pauline',

cousin\_2: 'Guillaume'

};

family.sister;

family.sister;

alert(family[id]);

**for** (**var** id **in** family) { *// On stocke l'identifiant dans « id »*

*pour parcourir l'objet « family »*

alert(family[id]);

}

**var** check = {};

check['login'] = **function**() {

**var** login = document.getElementById('login'),

tooltipStyle = getTooltip(login).style;

**if** (login.value.length >= 4) {

login.className = 'correct';

tooltipStyle.display = 'none';

**return true**;

} **else** {

login.className = 'incorrect';

tooltipStyle.display = 'inline-block';

**return false**;

}

};

check['lastName'] = **function**(id) {

**var** name = document.getElementById(id),

tooltipStyle = getTooltip(name).style;

**if** (name.value.length >= 2) {

name.className = 'correct';

tooltipStyle.display = 'none';

**return true**;

} **else** {

name.className = 'incorrect';

tooltipStyle.display = 'inline-block';

**return false**;

}

};

check[‘firstName’] = check[‘lastName’] ;

La fonction check['firstName'](), implicitement ce sera la fonction check['lastName']() qui sera appelée.

### Découper des chaines de caractère en un tableau :

**var** cousinsString = 'Pauline Guillaume Clarisse',

cousinsArray = cousinsString.split(' ');

### Créer une chaine de caractère depuis un tableau :

Cette fonction permet de créer une chaine de caractère à partir d’un tableau

**var** cousinsString\_2 = cousinsArray.join('-');

### L’accès aux éléments :

L’objet document possède trois méthodes principales :

getElementById() : Permet d’accéder à un élément en connaissant son ID qui est simplement l’attribut id de l’élément.

getElementsByTagName() : Cette méthode permet de récupérer sous la forme d’un tableau, tous les éléments de la famille.

getElementsByName() : Cette méthode est semblable à getElementsByTagName() et permet de ne récupérer que les éléments qui possèdent un attribut name que vous spécifiez. L'attribut name n'est utilisé qu'au sein des formulaires, et est déprécié depuis la spécification

HTML5 dans tout autre élément que celui d'un formulaire. Par exemple, vous pouvez vous en servir pour un élément **<input>** mais pas pour un élément **<map>**.

### Les événements :

#### Les événements à appliquer aux champs de texte :

**var** inputs = document.getElementsByTagName('input'),

inputsLength = inputs.length;

**for** (**var** i = 0 ; i < inputsLength ; i++) {

**if** (inputs[i].type == 'text' || inputs[i].type == 'password') {

inputs[i].onkeyup = **function**() {

check[**this**.id](**this**.id); *// « this » représente l'input*

*actuellement modifié*

};

}

}

#### Les événements à appliquer aux boutons :

myForm.onsubmit = **function**() {

**var** result = **true**;

**for** (**var** i **in** check) {

result = check[i](i) && result;

}

**if** (result) {

alert('Le formulaire est bien rempli.');

}

**return false**;

};

myForm.onreset = **function**() {

**for** (**var** i = 0 ; i < inputsLength ; i++) {

**if** (inputs[i].type == 'text' || inputs[i].type ==

'password') {

inputs[i].className = '';

}

}

deactivateTooltips();

};

})();

### Créer une chaîne de caractères depuis une chaîne ASCII

**var** myString = String.fromCharCode(74, 97, 118, 97, 83, 99, 114,105, 112, 116);

Détecter les touches clavier :

listenKey()

KeyCode retourne le code ASCII du caractère

Couper une chaine de caractère à chaque fois qu’une virgule est rencontrée :

**var** splitted = mychaine.split(',');

### Les expressions régulières :

Les regex ne s'utilisent pas seules, et il y a deux manières de s'en servir : soit par le biais de RegExp qui est l'objet qui gère les expressions régulières, soit par le biais de certaines méthodes de l'objet String :

* match() : retourne un tableau contenant toutes les occurrences recherchées ;
* search() : retourne la position d'une portion de texte (semblable à indexOf() mais avec une regex);
* split() : la fameuse méthode split(), mais avec une regex en paramètre ;
* replace() : effectue un rechercher/remplacer.

### Les données numériques :

Tester si une opération n’est pas infini ou NaN :

isFinite()

isNaN()

### La gestion du temps :

Mesurer le temps d’exécution d’une fonction en millisecondes**:** **getTime();**

setTimout() : déclencher un code au bout d’un temps donné,

setInterval() :déclencher un code à un intervalle régulier,

L’AJAX :

**Les chapitres pas très clairs :**

Les Regex