Sébastien Lafontaine

Version 1.1.0

22/01/2018

Javascript

Javascript = code client side

Concaténation = ajout de valeur d’une variable l’une à l’autre

(ex : var mot1 = chat ;

 var mot2 = chien ;

 var mot3 = mot1 + mot2 ;

)

|| = ou

&& = et

!qqc = l’inverse

(ex : var un = 1 ;

 var deux = 2 ;

 var trois = un + deux ;

 var condition = un > deux ; 🡪 false

 var resultat = !condition ; 🡪 true

)

Les conditions : possibilité d’exécution du code différemment selon ce que l’on veut

var uneVariable = ‘Bonjour’ ;

if(‘uneVariable’) {

 alert(‘ceci est plein’) ;

} else {

 alert(‘ceci est vide’) ;

}

Cette structure conditionnelle est correcte mais va poser problème lorsque l’on ajoutera des ‘else if’.

Switch : la structure conditionnelle qui ne fait pas des rallonges

var mot4 = message ;

**switch(**uneVariable**) {**

 **case ‘**chat**’ :**

 **message = ‘**joli les petits chats**’ ;**

 **break ;**

 **default :**

 **message = ‘**je ne connais pas cet animal**’ ;**

**alert(message) ;**

**if(**uneVariable **== ‘**chat**’) { alert((**uneVariable **== ‘**chat**’) ?**

 **alert(‘**comme moi**’) ; 🡪 ‘**comme moi**’ : ‘**pas comme moi**’**

**} else { ) ;**

 **alert(‘**pas comme moi**’) ;**

**}**

Les boucles : répétition de code et de condition sur une durée déterminée

++ = incrément de +1

-- = incrément de -1

Var nb = 1 ;

**while (**nb < 30**) {**

 Nb ++ ;

**}**

NaN = not a number

**for (**var repete = 0 ; repete < 10 ; repete++**) { 🡪** on remplace très souvent cette

alert**(‘**coucou**’) ;** valeur de répétition par ‘i’**.**

**}**

**for (**var i = 0 ; i < 10 ; i++**) {**

alert**(‘**coucou**’) ;**

**}**

Les tableaux : insérer des données dans des tableaux afin de les utiliser par la suite

var unTableau = **[** ‘chien’, ‘chat’, ‘lapin’, ‘canard’ **] ;**

 0 1 2 3

unTableau**[**2**]** = ‘vache’ ; 🡪 la troisième case du tableau va être remplacée (‘lapin’ par ‘vache’)

**Ajouter des valeurs:**

unTableau**.push(‘**cochon**’, ‘**renard**’) ; 🡪** ajout de ‘cochon’ et ‘renard’ au tableau

 après ‘canard’

unTableau**.pop() ; 🡪** enlève la dernière valeur du tableau, donc renard

(ex :

var animaux = **[]** ;

while (true) {

 var promptText = prompt(‘entrer un animal’) ;

 if(promptText != ‘ ‘) {

 animaux**.push(**promptText**) ;**

 } else {

 break ;

 }

}

alert(unTableau**.indexOf(‘**chat**’)**);

**🡪** Afficher une valeur du tableau par son index (1)

**Supprimer des valeurs et remplacer :**

unTableau**.splice(**placement**,** nbélemsupp, ‘élemajouté**’, ‘**élemajouté**’) ;**

On peut initialiser le nombre d’éléments à 0 pour ne rien supprimer.

ex : unTableau**.splice(**1**,** 2, ‘mouton**’, ‘**poule**’) ;**

[‘chien’, ‘chat’, ’mouton’, ‘poule’] On remplace ‘lapin’ et ‘canard’ du premier tableau par ‘mouton’ et ‘poule’ après ‘chat’.

Les fonctions : suite d’opération retournant des résultats configurés.

Affichage :

**function** bonjour**(){**

 alert(‘bonjour tout le monde’) ;

**}**

bonjour**() ;**

Addition :

var result(number(nb1) + number(nb2)) ;

**Function** addition**(**nb1, nb2**){**

**return** result

**}**

var prompt1 = prompt(‘un premier nombre : ‘) ;

var prompt2 = prompt(‘n deuxième nombre : ‘) ;

addtion(prompt1, prompt2) ;

Les objets : ensemble des données utiles du code.

var utilisateur **= {**nom : ‘Jean’**,** age : ‘22’**,** sexe : ‘M’**} ;**

**console.log(**utilisateur**.**nom**) ;**

var utilisateur **= {**nom : {prenom : ‘Jean’, famille : ‘Stalin’}**,** age : ‘22’**,** sexe : ‘M’**} ;**

**console.log(**utilisateur**.**nom**) ;**

**🡪** objet imbriqué