**Le langage JavaScript**

# Bien que le HTML soit reconnu comme étant un langage pour la mise en forme de documents Web, il présente les limites suivantes :

* Absence de structures de contrôle de flux (structures algorithmiques).
* Aucune connectivité avec les serveurs de base de données.
* Pas d'interactions avec l'utilisateur.

Pour étendre les capacités du langage HTML deux solutions ont été mises en oeuvre :

* Utilisation d'un **langage coté client** (JavaScript, VbScript)
* Utilisation d'un **langage coté serveur** (PHP, ASP/ASP .NET, JSP, Cold Fusion, Zope, …)

**Langages coté client**

Les langages coté client permettent quand à eux d'utiliser les ressources du navigateur pour : contrôler la validité du contenu saisit par l'utilisateur avant soumission du formulaire, enrichir le contenu des pages Web avec un contenu interactif et/ou animé. Ces langages possèdent l'avantage de ne pas nécessiter d'installations supplémentaires.

**Langages coté serveur**

Les langages coté serveur possèdent l'avantage de connectivité aux bases de données, transparence du code source, et d'autres fonctionnalités liées principalement à la sécurité de données. Le test de ces Scripts avant de leur publication nécessite l'installation d'un serveur Web.

**Code JavaScript Intégré dans le fichier HTML**

Pour intégrer JavaScript dans une page Web il suffira d'utiliser la balise suivante :

<script language="JavaScript"> // votre Script ici </script>

**Code JavaScript commun à plusieurs pages**

Pour intégrer un même script dans plusieurs pages Web. Il faudra :

1. Ecrire le script dans un fichier indépendant.
2. Enregistrer ce fichier avec une extension ".js".
3. Ajouter le code HTML suivant dans toutes les pages qui devront inclure ce script :

<script language="JavaScript" src="fichier.js"></script>

**Catégories**

Les objets JavaScript sont classés en deux catégories :

* Objets d'interface : window, document,…
* Objets Prédéfinis : String, Math, Date,…

# Déclaration des variables en JavaScript

|  |  |
| --- | --- |
| ***fonctionalité*** | ***Signe/Fonction en JavaScript***  Déclaration |
| Déclaration implicite  (obligatoirement avec affectation) | **x = 10**  **x= x+1**  Déclaration |
| Déclaration explicite avec affectation | **var x = 10**  **x= x+1** |
| Déclaration explicite, sans affectation | **var x**  Déclaration  **x=10**  **x= x+1** |

# Affectation, affichage et saisie

|  |  |  |
| --- | --- | --- |
| ***fonctionalité*** | ***Signe/Fonction en javascript*** | ***Exemples*** |
| Affectation | **=** | email**=** "ahmed@gmail.com" |
| Affichage dans une alerte | **alert(…)** | **alert**("Bonjour") |
| saisie d'une valeur donnée par l’utilisateur | **variable = prompt(…)** | n = **prompt**('Donner un nombre') |

# Opérateurs de comparaison

|  |  |  |
| --- | --- | --- |
| ***fonctionalité*** | ***Signe en javascript*** | ***Exemples*** |
| Egalité | **==** | if(x**==** 5) … |
| Différent | **!=** | if(x **!=** 5) … |
| Supérieur ou égale | **>=** | if(x **>=** 5) … |
| … | … | … |

# Opérateurs logiques

|  |  |  |
| --- | --- | --- |
| ***fonctionalité*** | ***Signe en javascript*** | ***Exemples*** |
| Et | **&&** | if(x ==true**&&**y == false ) … |
| Ou | **||** | if(x ==true**||** y == false) … |
| Non | **!** | if( ! x  ) … |

# Opérateurs arithmétiques

|  |  |  |  |
| --- | --- | --- | --- |
| ***fonctionalité*** | ***Signe en javascript*** | ***Exemples*** | ***Résultat*** |
| division réelle | / | x = 5 **/** 3 | x vaut1.66 |
| Reste de la division entière (MOD) | % | x = 5 **%**3 | x vaut2 |
| Résultat (Quotient)de la division entière (DIV) | *1ère méthode :*  Math.trunc(.../...)  *2ème méthode :*  ~~(… / …) | 1ère méthode :  x = **Math.trunc**(5 **/** 3)  2ème méthode :  x = **~~**(5 **/** 3) | x vaut1 |

# Fonctions sur les nombres

|  |  |  |  |
| --- | --- | --- | --- |
| ***fonctionnalité*** | ***Fonction en JavaScript*** | ***Exemples*** | ***Résultat*** |
| Verifier si une valeur n’est pas un nombre  (is Not a Number) | **isNaN( … )** | a = isNaN("Bonjour")  b = isNaN("3")  c = isNaN("3 ")  d = isNaN("3 6")  e = isNaN(3) f = isNaN(-1.23) | a vaut true  b vaut false  c vaut false  d vaut true  e vaut false  f vaut false |
| Convertir une chaine en un nombre | **Number(ch)** | n1 = Number("14.50")  n2 = Number(" 14.50 ")  n3 = Number(" 14.50 5 ")  n4 = Number("**14.50 abc**") | n1 vaut 14.5  n2 vaut 14.5  n3 vaut**NaN**  n4 vaut**NaN** |
| Convertir un nombre en une **chaine** | **n.toString( )** | x = 5  ch1 = x.toString( )  y = 18.75  ch2 = y.toString( ) | ch1 vaut**"**5**"**  ch2 vaut**"**18.75**"** |
| Convertir une chaine en un **entier** | **parseInt(ch)** | a = parseInt("10")  b = parseInt(" 10 ")  c = parseInt("10.33")  d = parseInt("34 45 66")  e = parseInt("**30dinars**")  f = parseInt("**30.5 dinars**")  g = parseInt("**j’ai30 dinars**") | a vaut 10  b vaut 10  c vaut 10  d vaut 34  e vaut**30**  f vaut **30**  g vaut **NaN** |
| Convertir une chaine en un **réel** | **parseFloat(ch)** | a = parseFloat ("10")  b = parseFloat (" 10 ")  c = parseFloat ("10.33")  d = parseFloat ("34 45 66")  e = parseFloat ("**30dinars**")  f = parseFloat ("**30.5 dinars**")  g = parseFloat ("**j’ai30 dinars**") | a vaut 10  b vaut 10  c vaut**10.33**  d vaut 34  e vaut**30**  f vaut **30.5**  g vaut **NaN** |

# Fonctions sur les chaines

|  |  |  |  |
| --- | --- | --- | --- |
| ***fonctionnalité*** | ***Fonction en JavaScript*** | ***Exemples*** | ***Résultat*** |
| Longueur d’une chaine ch | **ch.length** | ch="bjr"  n = ch.length | n vaut 3 |
| Le caractère d’indice i de la chaine ch.  Renvoi une chaine vide dans le cas l’indice déborde | ch.charAt( i ) | ch="bjr"  c1 = ch.charAt(1)  ch="bjr"  c2 = ch.charAt(10) | c1 vaut'j'  c2 vaut'' |
| Retourne la première position d'une sous-chaîne **Sch**dans la chaîne **Ch**, en commençant la recherche à partir de la position **debut**.Elle renvoi -1 dans le cas contraire ou il n’existe pas.  Si l’argument **debut** est omis, la recherche débutera à la position 0 | **P=Ch.indexOf(Sch, debut)** | Ch="Javascript"  Sch="a"  P1=Ch.indexOf(Sch)  Ch="position"  Sch="o"  P2=Ch.indexOf(Sch,3)  Ch="Javascript"  Sch="@"  P3=Ch.indexOf(Sch)  Ch="Javascript"  Sch="script"  P4=Ch.indexOf(Sch) | P1 vaut 1  P2 vaut 6  P3 vaut -1  P4 vaut 4 |
| Retourne la dernière position d'une sous-chaîne **Sch**dans la chaîne **Ch**, en commençant la recherche à partir 0 jusqu'à la position **fin**.Elle renvoi -1 dans le cas contraire ou il n’existe pas.  Si l’argument **debut** est omis, la recherche débutera à la position 0 | **P=Ch.lastIndexOf(Sch,fin)** | Ch="Javascript"  Sch="a"  P3=Ch.lastIndexOf(Sch)  Ch="Javascript"  Sch="i"  P4=Ch.lastIndexOf(Sch, 5)  Ch="Javascript"  Sch="a"  P1=Ch.lastIndexOf(Sch,2)  Ch="Javascript"  Sch="a" ;  P2=Ch.lastIndexOf(Sch,Ch.length) | P3 vaut 3  P4 vaut -1  P1 vaut 1  P2 vaut 3 |
| permet d'extraire d'une chaîne donnée Ch, une sous chaîne de caractères Sch  à partir d'une position déterminée **debut** et d'une longueur déterminée **Nbr**.  Le paramètre **Nbr** peut être omis de façon à inclure tous les caractères jusqu’à la fin de la chaîne. | **Sch=Ch.substr(debut,Nbr)** | Ch="Javascript" ;  Sch1 = Ch.substr(3,4);  Ch="Javascript" ;  Sch2 = Ch.substr(3); | Sch1 vaut "ascr"  Sch2 vaut "ascript" |
| permet d'extraire d'une chaîne donnée Ch, une sous chaîne de caractères Sch  à partir d'une position **debut** jusqu’à la position **fin**.  Le paramètre **fin** peut être omis de façon à inclure tous les caractères jusqu’à la fin de la chaîne. | **Sch=Ch.substring(debut,fin\_exclu)** | Ch="Javascript" ;  Sch3 = Ch.**substring** (0,2);  Ch="Javascript" ;  Sch1 = Ch.**substring** (3,4);  Ch="Javascript" ;  Sch4 = Ch.**substring** (3,3);  Ch="Javascript" ;  Sch2 = Ch.**substring** (3); | Sch3 vaut'Ja'  Sch1 vaut'a'  Sch4 vaut ''  Sch2 vaut'ascript' |
| Concaténer ch1 et ch2 | **Ch=Ch1.concat(Ch2)** | Ch1="bon"  Ch2="jour"  Ch3=Ch1.concat(Ch2);  Ch4=Ch2.concat(Ch1); | Ch3 vaut "bonjour"  Ch4 vaut "jourbon" |
| Concaténer ch1 et ch2 | **Ch=Ch1 + Ch2** | Ch1="bon"  Ch2="jour"  Ch3=Ch1 + Ch2  Ch4= Ch2 + Ch1 | Ch3 vaut "bonjour"  Ch4 vaut "jourbon" |
| transforme une chaîne de caractères Ch en majuscule | Ch.toUpperCase( ) | Ch="**JAVA**script"  Ch1=Ch.toUpperCase() | Ch1 vaut"**JAVASCRIPT**" |
| transforme une chaîne de caractères Ch en minuscule | Ch.toLowerCase( ) | Ch="JAVAscript"  Ch1=Ch.toLowerCase( ) | Ch1 vaut "javascript" |
| Enlève les espaces du début et de la fin de la chaine ch | Ch.trim( ) | Ch="bjr "  Ch1=Ch.trim( ) | Ch1 vaut "bjr" |
| Remplace, dans ch, la **premiere** occurrence de ch1 par ch2 | ch.replace(ch1,ch2) | ch = "Mr **Blue** has a **blue** house and a **blue** car";  result = ch.replace("blue", "red"); | result vaut "Mr **Blue** has a **red**house and a **blue** car" |
| Découpe une chaine ch en se basant sur la chaine ch1 et range le résultat dans un tableau | Ch.split(ch1) | ch = "Mr Blue has a **blue** house and a **blue** car";  t = ch.split('**blue'**) | t vaut  Array(  "Mr Blue has a ",  " house and a ",  " car"  ) |

# Fonctions diverses

|  |  |  |  |
| --- | --- | --- | --- |
| ***fonctionnalité*** | ***Fonction en JavaScript*** | ***Exemples*** | ***Résultat*** |
| évalue une expression écrite sous forme de chaîne de caractères | **eval(ch)** | a = 5.2  msg = "a+3"  x=eval(msg)  b = '5.2'  msg = " parseInt(b) +3 "  y=eval(msg)  msg = "alert('Bonjour')"  eval(msg) | x vaut **8.2**  y vaut **8**  Le navigateur va afficher une alerte contenant le message "Bonjour " |

Déclaration d’une fonction La déclaration d’une fonction dans l'en-tête de la page, entre les balises <Head> et </Head>

function nomfonction( paramètres)

{ Traitement ; }

Structures Conditionnelles

If (condition)

{ }

Else

{ }

Rq sur les conditions : On note : a==b si a et b sont égaux

a!=b si a et b sont différents

&& pour ET

|| Pour OU

**Les boucles**

* while (condition)

{ instructions }

* for(initialisation ; condition ; incrémentation)

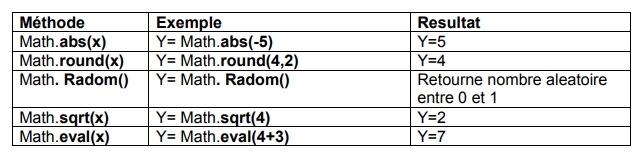
{ instructions }

* Do

{ instructions }

while(condition);

**Les méthodes de l’objet Maths :**

****