JavaScript

1. Le formalisme de base du JavaScript

```
<html>
<head>
<title>Premier code en javascript</title>

<script language="javascript">
alert("Bienvenue");
//alert permet d'afficher une fenêtre de message.
 </script>

</head>

<body>
<h1><center>Premier exemple en JavaScript
</center></h1>
<h1><center>Fin du premier script</center></h1>
</body>
</html>
```

```
<html>
<head>
<title>Premier code en javascript</title>
</head>

<body>
<h1><center>Premier exemple en
JavaScript </center></h1>

<script language="javascript">
alert("Bienvenue");
//alert permet d'afficher une fenêtre de message.
</script>

<h1><center>Fin du premier script</center></h1>
</body>
</html>
```

Constatations :

- Les solutions permettant d'insérer du code JavaScript dans une page web sont :
  - Insérer des instructions JavaScript entre <script>…</script> dans <body> et </body> ;
  - Déclarer les fonctions puis les appeler dans <body> et </body> ;
  - Utiliser un des gestionnaires d'évènements rattachés aux balises html pour appeler des fonctions préalablement définies « onClik » et « onMouseOver ».
- La déclaration de fonctions JavaScript peut se faire :
  - Entre <head> et </head>
  - Entre <body> et </body>
  - Dans un fichier externe avec l'extension « .js » à inclure dans les fichiers HTML.

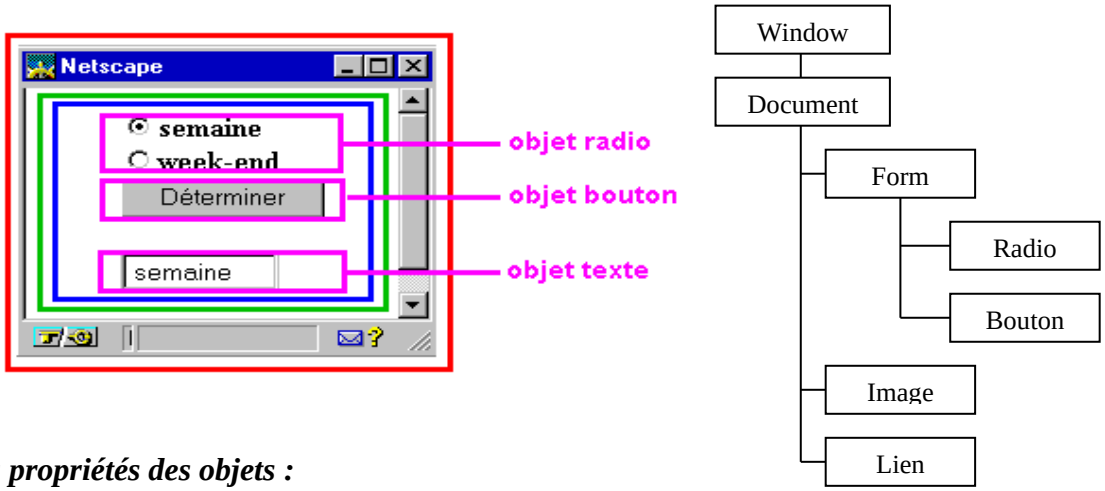
2. Les objets JavaScript et leurs hiérarchies

JavaScript divise une page web en objets et permet d'accéder à ces objets de tirer des informations et de les manipuler. On distingue deux catégories d'objets :

- *Les objets d'interface* : permettent de gérer les aspects visuels des différents contrôles graphiques : L'objet Window, document, button, radio, checkbox, etc.
- *Les objets des propriétés et des fonctions prédéfinies* : fournissent les différentes ressources requises pour la programmation : l'objet String, math, date, navigator, array et object.

a- La hiérarchie des objets d'interfaces :

Exp. :



b- Les propriétés des objets :

➔ Pour accéder à une propriété, il faut donner le chemin complet de l'objet :

Syntaxe : **nom_objet.nom_propriété**

Exp. :

document.form.zone.value c'est le contenu (propriété *value*) de l'élément appelé « zone » du formulaire « form ».

➔ La propriété *innerHTML* permet d'accéder au contenu d'une balise identifiée par son attribut « *id* ».

Syntaxe : **document.getElementById('idelement').innerHTML = "contenu";**

Exp. :

```
document.getElementById("paragraphe").innerHTML = "Texte changé !";
```

c- Les méthodes des objets :

➔ Le langage JavaScript a prévu un ensemble de méthodes (fonctions) pour chaque objet.

syntaxe : **nom_objet.nom_methode()**

Par exemple pour l'objet document on dispose de la méthode **write ()** ou **writeln()** « Ecrire dans le document ».

- L'opérateur « + » joue le rôle de concaténation lorsqu'il est utilisé avec la méthode « write ».
- Il est possible de générer du code HTML lors de l'utilisation de la méthode « write ».
- Si le code HTML contient (") et pour ne pas confondre avec les guillemets de write il sera mieux de les remplacer par des apostrophes (').

d- Accès aux éléments d'une page web

La méthode ***getElementById()*** permet de récupérer les informations d'une balise identifiée par son ***id***.

Exp. :

- Modifier la contenu d'une zone de texte :

```
document.getElementById("age").value = 25;
```

- Modifier la couleur du texte :

```
document.getElementById("paragraphe").style.color = 'blue';
```

- Modifier le fond du titre du texte :
  document.getElementById("titre").style.backgroundColor = "#FF0000";
- Modifier le source d'une image :
  document.getElementById('idImage').src='images/photo2.jpg';

3. Les Entrées/Sorties en JavaScript

- L'entrée (lecture) : est faisable avec la méthode « prompt » de l'objet « window » ou à l'aide d'objets graphiques du formulaire HTML ;
- La sortie : est possible en utilisant la méthode « write » de l'objet « document », la méthode « alert » de l'objet « window » ou à l'aide d'objets graphique du formulaire HTML. Ou avec l'objet innerHTML contient le code HTML intérieur à l'objet.

Remarque : innerHTML :
Cette propriété est accessible en écriture : il est donc possible de changer le contenu d'un objet.

```
Exemple :  x=document.getElementById("element1").innerHTML;
           document.getElementById("element2").innerHTML=5+x;
           contenu=document.getElementById("eltformulaire").value ;
```

4. Les opérateurs :

4.1. Les Opérateurs de Calcul :

Dans les exemples, la valeur initiale de x sera toujours égale à 11

Signe	Nom	Signification	Exemple	Résultat
+	plus	addition	x + 3	14
-	moins	soustraction	x - 3	8
*	multiplié par	multiplication	x*2	22
/	divisé	par division	x /2	5.5
%	modulo	reste de la division par	x%5	1
=	a la valeur	affectation	x=5	5

4.2.Les Opérateurs de Comparaison :

Signe	Nom	Exemple	Résultat
==	égal	x==11	true
<	inférieur	x<11	false
<=	inférieur ou égal	x<=11	true
>	supérieur	x>11	false
=<	supérieur ou égal	x>=11	true
!=	différent	x!=11	false

IMPORTANT :

On confond souvent le **=** et le **==** (deux signes =). Le = est un opérateur d'attribution de valeur tandis que le == est un opérateur de comparaison. Cette confusion est une source classique d'erreur de programmation.

4.3.Les Opérateurs Associatifs :

Dans les exemples suivants x vaut toujours 11 et y aura comme valeur 5.

Signe	Description	Exemple	Signification	Résultat
+=	plus égal	x += y	x = x + y	16
-=	moins égal	x -= y	x = x - y	6
*=	multiplié égal	x *= y	x = x * y	55
/=	divisé égal	x /= y	x = x / y	2.2

4.4.Les Opérateurs Logiques :

Aussi appelés opérateurs booléens, ses opérateurs servent à vérifier deux ou plusieurs conditions.

Signe	Nom	Exemple	Signification
&&	et	(condition1) && (condition2)	condition1 et condition2
||	ou	(condition1) || (condition2)	condition1 ou condition2

4.5.Les opérateurs d'incrémentation

Dans les exemples x vaut 3.

Signe	Description	Exemple	Signification	Résultat
x++	incrémentation (x++ est le même que x=x+1)	y = x++	3 puis plus 1	4
x--	décrémentation (x-- est le même que x=x-1)	y= x--	3 puis moins 1	2

5. Les variables

5.1. La déclaration des variables

Les variables peuvent se déclarer de deux façons :

- Façon explicite : en utilisant la commande « var »
  Exemple : var Numero =1 var Prenom= "Yassine"
- Façon implicite : Décrire directement le nom de la variable suivi de la valeur attribuée :
  Exemple : Numero = 1

5.2. La visibilité des variables

- Les variables déclarées (de façon explicite ou implicite) au début du script, en dehors et avant toutes fonctions, sont des variables globales ;
- Dans une fonction :
  - Une variable déclarée par le mot clé Var est locale à cette fonction ;
  - Une variable déclarée sans le mot clé Var, sa portée sera globale.
- JavaScript utilise 4 types de données :
  - Des nombres (entier et réel) ;
  - Des chaînes de caractères ;
  - Des booléens : (True pour vrai et False pour faux) ;
  - Le mot null : Mot spécial qui indique l'absence d'une valeur.

Il ne faut pas déclarer le type de données d'une variable.

Activité2 : Créer un fichier nommé « tpjs4bis.html » et saisir le code suivant :

```
<html><body>
<script language="javascript">
```

```
b=prompt ("saisissez une valeur");
if(isNaN(b))
 {alert("conversion impossible");}
  else
  {
   b=Number (b);
   b=b+1;
   document.write("la valeur de b est :"+b);
  }</script>
</body></html>
```

Remarque :

- La Fonction « **isNaN** : is Not a Number » est une fonction booléenne permettant de vérifier si le contenu d'une variable donnée est numérique ou non. ;
- **Alert** et **Prompt** : sont deux méthodes de l'objet « window » permettant l'affichage et la saisie dans des boites de dialogue.
- Il existe des fonctions de conversion de type : String et Number
  - Var a = String (21.34) ; → a = "21.34"
  - Var a = Number ("10.5") ; → a = 10.5
- La fonction « **eval** » évalue une chaine de caractères sous forme de valeur numérique :
  Exemple : ch="5+10" ; x=eval (ch) ; → x=15 ;
- Il est préférable de précéder toute conversion avec la fonction « Number » par un test de validité avec la fonction « isNaN ».
- **Confirm** :donne une fenetre avec deux choix ok et annuler

```
exemple :  if(confirm("vous etes bac info?"))  alert("commencer la programmation");
                 else   alert(("commencer le traitement de texte");
```

6. Les types

6.1. Objet string

x=variable.length;	Donne la longueur d'une chaine	a= « informatique » ; x=a.length ; x =…….

Les méthodes de l'objet String

Syntaxe	Role	Exemples
chaine.charAt(pos)	Donne lecaractere correspondant a la position dans la chaine	a= informatique ; a. charAt(3) donne « f »
P=Ch.indexOf(Sch,pos); **ou** **P=Ch.indexOf(Sch);**	Retourne la position d'une sous-chaîne souschaine dans une chaîne, à partir de la position pos sinonelle retourne -1	Ch="Javascript" ; Sch="script" ; P=Ch.indexOf(Sch,0); P=…….
P=Ch.lastIndexOf(Sch,pos); **P=Ch.lastIndexOf(Sch);**	Retourne l'indice de la dernière occurence de Sch Si souschaine n'est pas trouvée, lastIndexOf() retourne -1.	Ch="Javascript" ; P=Ch.LastIndexOf("a"); P=……..

Sch=Ch.substr(P,Nbr);	permet d'extraire d'une chaîne **Ch**, une sous chaîne **Sch** à partir d'une position **P** et d'une longueur **Nbr**. Nbr est facultatif.	Ch="Javascript" ; Sch1 = Ch.substr(3,3); Sch2 = Ch.substr(1); Sch1 ="asc" Sch2 = "avascript"
R=Ch.toLowerCase();	transforme une chaîne de caractères **Ch** en minuscule	
R=Ch.toUpperCase();	transforme une chaîne de caractères **Ch** en majuscule	
Ch=Ch1.concat(Ch2);	concaténer une chaîne de caractères **Ch1** avec une autre **Ch2**	Ch1="java" Ch2="script" Ch=Ch1.concat(Ch2); Ch="javascript"
X=chaine.charCodeAt(pos) ;	Retourne le code ASCII du caractère de chaine à la position pos	x= chaine.charCodeAt(1) ; **x = 65**

6.2.L'objet Array :

a- Declaration

Schéma	Syntaxe	Exemple
1	NomObjet= new Array();	T=new Array();
2	NomObjet= new Array(nombre);	T=new Array(10);
3	NomObjet= new Array(val1, val2);	T=new Array("A","B","C");

Taille=Nomtableau.length; permet de calculer le nombre d'éléments dans un tableau. Le remplissage se fait avec **nomtableau[position]=valeur ;**

b- Les méthodes de l'objet Array

ObjetTab=NTab1.concat(NTab2);	former un tableau en provenance de la concaténation de deux autres	Var A= new Array(1,2); var B = new Array(10,30); C= A.concat(B); **C =(1, 2, 10, 20, 30)**
Ch=T.join("Separateur");	transforme un tableau T en chaîne de caractères Ch et exploite un séparateur par lequel les éléments du tableau doivent être séparés dans la chaîne de caractères.	Var T = new Array(1,2,4,8,16,32); var ch = T.join(":"); **Ch = "1:2:4:8:16:32"**
T.pop();	permet d'effacer le dernier élément d'un tableau T	Var T = new Array(64,128,256); T.pop(); **T=(64,128)**

Taille=T.push("élt1","éltn");	ajoute plusieurs éléments à la fin d'un tableau T et retourne la nouvelle taille du tableau.	T=new Array("UN"); N=T.Push("10","Info"); **N=3**
T.reverse()	permet d'inverser l'ordre des éléments d'un tableau T	T=new Array("R","I","P","T"); T.reverse() ; T=("T","P","I","R")
element = T.shift();	enlever le premier élément d'un tableau T et le mettre dans le resultat	T=new Array("R","I","P","T"); A=T.shift(); **T=("I","P","T")** **A="R"**
R = T.slice(indexdébut,indexfin);	Extraire une partie d'un tableau T allant de indexdebut a indexfin Elle renvoie les éléments extraits sous forme d'un nouveau tableau	T=new Array("R","I","P","T"); R=T.slice(1,2); **R=("R","I")**
T.sort(); permet de trier les éléments d'un tableau. Si vous ne transmettez aucun paramètre, le tri se fait alphabétiquement, les valeurs numériques sont transformées automatiquement en chaînes de caractères et triées en tant que telles. Dans le cas où vous voulez trier des valeurs numériques, vous pouvez définir une fonction de comparaison et transmettre son nom comme paramètre. Cette méthode s'applique avec la syntaxe suivante **function compare(a,b)** **{return a-b}**		Exemple1 T=new Array("R","I","P","T"); T.sort(); **T("I","P","R","T")** Exemple2 T=new Array(14,7,66,19,33); T.sort(**compare**); **T=(7,14,19,33,66)**

6.3.L'objet Math :

a- Les méthodes de l'objet Math

Méthode	Résultat
Math.abs(nombre)	Valeur absolue du nombre.
Math.min(nombre1,nombre2)	Plus petit des deux nombres.
Math.max(nombre1,nombre2)	Plus grand des deux nombres.
Math.sin(nombre)	Sinus de l'angle nombre.
Math.cos(nombre)	Cosinus de l'angle nombre.
Math.tan(nombre)	Tangente de l'angle nombre.
Math.asin(nombre)	Arc-sinus du nombre.
Math.acos(nombre)	Arc-cosinus du nombre.
Math.atan(nombre)	Arc-tangente du nombre.
Math.atan2(y,x)	Arc-tangente dont les projections sur les axes Y et X sont les nombres y et x.
Math.sqrt(nombre)	Racine carrée du nombre.
Math.ceil(nombre)	Arrondi du nombre à l'entier supérieur.

Math.floor(nombre)	Arrondi du nombre à l'entier inférieur.
Math.round(nombre)	Arrondi du nombre à l'entier le plus proche.
Math.exp(nombre)	enombre
Math.log(nombre)	Logarithme naturel du nombre
Math.pow(base,exposant)	baseexposant
Math.random()	Nombre pseudo-aléatoire compris entre 0 et 1 inclus. Le générateur de nombres aléatoires prend automatiquement une valeur initiale lors du chargement de JavaScript.
X= Math.parseFloat(chaine);	essayent de convertir une chaîne numérique en en réel retourne NaN si le 1er caractere n'est pas numerique
x= Math.parseInt(chaine);	essayent de convertir une chaîne numérique en entier retourne NaN si le 1er caractere n'est pas numerique

parseInt(chaine, b) :permet de convertir la chaine en nombre entier a base 10 avec **b** est la base de chaine exemple parseInt("ABC7F", 16) donne 703615.
nb.toString(b) permet de convertir le nombre decimal nb a la base b
Exemple nb=25 ; V= nb.toString(2) ;//convertit nb a la base binaire V=11001
V = nb.toString(16)// convertit nb a la base hexadecimal V=19

6.4.L'objet Date :

a- Les méthodes de l'objet Date

Syntaxe	Role
variable = new Date();	Cette méthode renvoie la date et l'heure système. Ces informations sont enregistrées par Javascript sous le format : "Mon Dec 10 09:23:30 2014"
variable_date = new Date(); **an = variable_date.getYear();**	Retourne les deux derniers chiffres de l'année dans variable_date. Soit ici 14.
variable_date = new Date(); **mois =variable_date.getMonth();**	Retourne le mois dans variable_date sous forme d'un entier compris entre 0 et 11 (0 pour janvier, 1 pour février, 2 pour mars, etc.).
variable_date = new Date(); **jourm = variable_date.getDate();**	Retourne le jour du mois dans variable_date sous forme d'un entier compris entre 1 et 31.

variable_date = new Date(); jours = variable_date.getDay();	Retourne le jour de la semaine dans variable_date sous forme d'un entier compris entre 0 et 6 (0 pour dimanche, 1 pour lundi, 2 pour mardi, etc.).
variable_date = new Date(); HRS = variable_date.getHours();	Retourne l'heure dans variable_date sous forme d'un entier compris entre 0 et 23.
variable_date=new Date(); min=variable_date.getMinutes();	Retourne les minutes dans variable_date sous forme d'un entier compris entre 0 et 59
variable_date=new Date(); sec=variable_date.getSeconds();	Retourne les secondes dans variable_date sous forme d'un entier compris entre 0 et 59.

7. Les Structures de contrôle :

7.1. Les structures conditionnelles

La Forme réduite	La Forme complète

if (condition vraie)
{Une ou plusieurs instructions ;}
	if (condition vraie)
	{Instructions 1 ;}
	else
	{Instructions 2 ;}

Remarques:

- Dans le cas d'une seule instruction, les accolades sont facultatives;
- Il est possible d'imbriquer des structures conditionnelles (forme généralisée) ;
- Autre syntaxe (expression) ? instruction 1 : instruction 2.

7.2. Les structures itératives

a- La structure For

Permet de répéter l'exécution d'un bloc d'instructions un certain nombre de fois connu d'avance :

```
For (initialisation ; condition ; progression)
{
 Instructions ;
}
```

Initialisation : les instructions d'initialisations nécessaires ;
Condition : la condition de continuité (bouclage) ;
Progression : définit le pas du compteur (pas forcément + ou – 1)

Activité : Ecrire un code JavaScript permettant d'afficher tous les nombres parfaits compris entre 2 et 1000 ; sachant qu'un nombre N est dit parfait s'il est égal à la somme de ses diviseurs sauf lui même.
Exemple pour N=6 ➔ 1+2+3=6

```
<html>
<head>
<title>boucle pour</title>
</head>
<body>
<script language="javascript">
for (n=2;n<=1000;n++)
{
 s=0;
 for(i=1;i<=n/2;i++)
  if(n%i==0) s+=i;
 if (s==n) document.write(n+"<br>");
}
</script>
</body>
</html>
```

b- La structure do…while

Permet de répéter l'exécution des instructions tant que la condition est vérifiée. L'équivalent en langage pascal est repeat…until (à la seule différence que la condition utilisée après until est une condition d'arrêt alors que la condition utilisée après while est une condition de continuité)

```
Do
{
Instructions;
} while (condition(s));
```

Activité : Ecrire un code JavaScript permettant d'afficher tous les nombres premiers compris entre deux entiers (a>1 et a<b et b<1001). Un nombre est dit premier s'il n'est divisible que par 1 et par lui-même.

```
<html>
<head>
<title>do while</title>
</head>
<body>
<script language="javascript">
do
{
a=Number(window.prompt("donner un entier: ",""));
b=Number(window.prompt("donner un autre entier:",""));
}while(a<=1||a>=b||b>=1001);
for (n=a;n<=b;n++)
{i=1;
 do
  {i=i+1;
  }while (n%i!=0 && i<=n/2);
if (i>n/2) document.writeln(n);}
</script>
</body>
</html>
```

c- La structure while

Lorsque le nombre de répétitions n'est pas connu d'avance on utilise la structure itérative while.

```
While (condition(s))
{
 Instructions ;
}
```

Activité : Ecrire un code JavaScript permettant d'afficher le PGCD de deux entiers donnés a et b tels que (a>1 et b>1) en utilisant la méthode de différences.

```
<html>
<head>
<title>boucle while</title>
</head>
<body>
<script language="javascript">
var a,b;
```

```
do
{
a=Number(prompt("donner un entier",""));
b=Number(prompt("donner un autre entier",""));
}while (a<=1||b<=1);
while (a!=b)
if (a>b)
a-=b;
else
b-=a;
alert("le pgcd est "+a);
</script></body>
</html>
```

8. Les fonctions en javascript

Une fonction est un groupe de ligne(s) de code de programmation destiné à exécuter une tâche bien spécifique et que l'on pourra utiliser à plusieurs reprises. De plus, l'usage des fonctions améliorera grandement la lisibilité de notre script.
En Javascript, il existe deux types de fonctions :

- les fonctions propres à Javascript. On les appelle des "méthodes". Elles sont associées à un objet bien particulier comme c'était le cas de la méthode Alert() avec l'objet window.
- les fonctions écrites par le programmeur pour les besoins de son script.

8.1.Déclaration des fonctions

La syntaxe d'une déclaration de fonction est la suivante :

```
Function  nom_de_la_fonction(arguments) {
... code des instructions ...
}
```

La mention des arguments est facultative mais dans ce cas les parenthèses doivent rester. C'est d'ailleurs grâce à ces parenthèses que l'interpréteur Javascript distingue les variables des fonctions.
C'est lors de l'appel de la fonction que le code de programme est exécuté.

8.2.L'appel d'une fonction

L'appel d'une fonction se fait par le nom de la fonction (avec les parenthèses).
Soit par exemple **nom_de_la_fonction();**

La fonction doit etre définie avant d'être appelée.Il est donc prudent de placer toutes les déclarations de fonctions dans la dans la balise <HEAD> ... <HEAD>.

3 Exemple

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="Javascript">
function message() {
document.write("Bienvenue à ma page");}
</SCRIPT>
</HEAD>
```

<BODY onLoad="message()">

Description	Evénement
Lorsque l'utilisateur clique sur un bouton, un lien ou tout autre élément.	Clik
Lorsque la page est chargée par le browser ou le navigateur.	Load
Lorsque l'utilisateur place le pointeur de la souris sur un lien ou tout autre élément.	MouseOver
Lorsque un élément de formulaire a le focus c-à-d devient la zone d'entrée active.	Focus
Lorsque un élément de formulaire perd le focus c-à-d que l'utilisateur clique hors du champs et que la zone d'entrée n'est plus active.	Blur
Lorsque la valeur d'un champ de formulaire est modifiée.	Change
Lorsque l'utilisateur clique sur le bouton Submit pour envoyer un formulaire.	Submit
Lorsque l'utilisateur écrit dans une zone de formulaire	input

</BODY></HTML>

8.3.Passer des valeurs à une fonction

On peut passer des paramètres aux fonctions Javascript
function Exemple(Texte) {
alert(texte); }
Et dans l'appel de la fonction : Exemple("Salut à tous");

Syntaxe	**exemple**

function nom_de_la_fonction(arg1, arg2, arg3) {
... code des instructions ...
}
	function Exemplebis(Texte1, Texte2){
	alert(texte1,texte2);
	}
	et pour l'appel de la fonction
	Exemplebis("Salut à tous", "Signé directeur")

8.4.Retourner une valeur

Pour renvoyer un résultat, il suffit d'écrire le mot clé return suivi de l'expression à renvoyer.

syntaxe	**exemple :**

Function nom_de_la_fonction(arguments) {
... code des instructions ...
Return non_vble ; }
	function cube(nombre) {
	var C = nombre*nombre*nombre
	return C;
	}

Remarque L'instruction return est facultative et on peut trouver plusieurs return dans une même fonction.

9. La gestion des événements en javascript :

Les gestionnaires d'événements

les événements doivent etre associées a des actions prévues par le programme. C'est le rôle des gestionnaires d'événements. La syntaxe est

<nonbalise onévénement="fonction()">

exemple

<input type="button" name="b" value="executer" onClick="execution()">

Objets	Gestionnaires d'événement disponibles
Formulaire	onSubmit
Fenêtre	onLoad,
Lien hypertexte	onClick, onmouseOver
Elément de texte	onBlur, onChange, onFocus,

Elément de zone de texte	onBlur, onChange, onFocus, oninput
Elément bouton	onClick
Case à cocher	onClick
Bouton Radio	onClick
Liste de sélectionon	onBlur, onChange, onFocus
Bouton Submit	onClick
Bouton Reset	onClick
Clavier (saisie d'un texte)	onKeyUp, onkeypress

Exemple <A HREF="" onmouseOut="action()">lien</A>

Application1

Utiliser le gestionnaire d'evenement pour le code suivant:

```
<HTML>
<HEAD>
<SCRIPT language="Javascript">
function message(){
alert("evenementchoisi")
}
</SCRIPT>
</HEAD>
<BODY>
<FORM>
<INPUT TYPE=text ……..="message( )">
<INPUT TYPE="button" VALUE="Cliquezici"
……="message()">
</FORM>
<A HREF=""…….="message( )">
message important</A>
</BODY></HTML>
```

Application2

```
<html>
<head>
<script language="javascript">
function lon( ){
np=document.f.a.value;
document.f.d.value=np.length
}
</script>
</head>
<body>
<form method="POST" name="f">
<p>nom
<input type="text" name="a" size="20" onkeyup="lon( )">
</p>
<p>nombre de caracteres tapés
<input type="text" name="d" size="20"></p>
</form>
</body></html>
```

10.Manipulation des formulaires

A tout élément de formulaire on peut lui associe les méthodes suivantes :

nom	rôle	contrôles concernés
blur()	enlève le focus d'un champ du formulaire	input, select, textarea
click()	simule un clic de souris sur le champ du formulaire (input,)	input
focus()	attribue le focus à un champ du formulaire	input, select, textarea
select()	sélectionne le contenu d'un champ du formulaire	input, textarea

il existe également les méthodes submit() et reset() pour l'élément form

- **Zone de texte**

recuperer son contenu :

Nomvariable= document.nomformulaire.nomzonedesaisie.value ;

Nomvariable= document. getElementById('idtexte').value ;

modifier son contenu :

document.nomformulaire.nomzonedesaisie.value=expression ;

document. getElementById('idtexte').value =expression ;

Les controles password ,email,tel,number, textarea sont traitées avec la meme façon .

Les boutons radio

tester si la case est cocher

document.nomformulaire.nomcase[indice].checked==true ;

document. getElementById('idradio').checked==true ;

recuperer la valeur d'un bouton radio

nomvariable = document.nomformulaire.nomcase[indice].value ;

nomvariable = document. getElementById('idradio').value ;

recuperer le nombre d'options :

nomvariable = document.nomformulaire.nomcase.length ;

nomvariable = document. getElementById('idradio'). length ;

Les checkbox

recuperer la valeur d'un bouton checkbox

nomvariable = document.nomformulaire.nomcase.Value ;

nomvariable = document. getElementById('idchek').value ;

Verifier si une case est cocher

nomvariable = document.nomformulaire.nomcase.checked ;

nomvariable = document. getElementById('idchek'). checked ;

Liste de selection

la gestion des listes déroulantes se fait à travers ***document.nomformulaire.nomliste ou document. getElementById('idliste')***

document. getElementById('idliste'). **length** :/

retourne le nombre d'éléments de la liste

document. getElementById('idliste').**selectedIndex** : /

permet de retourner l'indice de l'élément sélectionné dans une liste déroulante à sélection unique.

NB1 : L'indice d'une liste déroulante commence à partir de 0.

Les propriétés JavaScript des éléments d'un objet options sont :

document. getElementById('idliste').options[i].**selected** : renvoie true si l'option est sélectionnée, false sinon.

document. getElementById('idliste').options[i].**text** : renvoie le texte de l'élément sélectionné.

document. getElementById('idliste').options[i].**value** : renvoie la valeur de l'élément sélectionné.

L'ajout d'une nouvelle option, se fait à travers la syntaxe :

nomvariable= new Option(texte, valeur);

document.nomformulaire.nomliste.options[taille] = nomvariable;

document. getElementById('idliste').options[taille] = nomvariable;

texte : le texte de l'élément à ajouter.
valeur : la valeur de l'élément à ajouter.
taille : la position de l'élément à ajouter dans la liste.(length de la liste)

❖ La suppression d'une option, se fait à travers la syntaxe :

document. nomformulaire.nomliste.options[i] = null;

document. getElementById('idliste'). options[i]=null ;

avec :

i : la position de l'élément à supprimer de la liste.

Exemple

Créer un formulaire qui contient une liste deroulante appelee eleve ,contenant 4 nom d'eleves ,si on choisit un et on clique sur un bouton il affiche le nom d'eleve choisit .ajouter 3 bouton pour ajouter,supprimer,

```
<html>
<head>
<script language="javascript">
function changer()
{i=document.f.choix.options.selectedIndex;
alert(document.f.choix.options[i].value);
}
function ajout()
{x=prompt("le texte");
y=prompt("valeur");
v=new Option(x,y);
t=document.f.choix.options.length;
document.f.choix.options[t]=v;
alert("on ajout elelt "+(t+1));
}
function supp()
{i=prompt("donner la pos a sup");
document.f.choix.options[i]=null;
}
function vider()
{document.f.choix.options.length=null;
}
</script>
</head>
<body>
<form name="f">
<select name="choix" size="1"
onchange="changer()">
<option value="chaine1">ch1</option>
<option value="chaine2">ch2</option>
<option value="chaine3">ch3</option>
</select>
<input type=”button” name=”b1”
value=”executer” onclick="changer()">
<input type=”button” name=”b2”
value=”ajouter” onclick="ajout()">
<input type=”button” name=”b3”
value=”supprimer” onclick="supp()">
<input type=”button” name=”b4”
value=”vider” onclick="vider()">
</form>
</body>
</html>
```