JavaScript

1. Introduction : notions de base

- Langage "case sensitive"

- Langage objet, par exemple, l'objet **window** correspond à la fenêtre affichée dans le navigateur, l'objet **location,** à l'URL courante, …

- Certains objets possèdent des propriétés, par exemple dans un formulaire comprenant une zone de texte de nom "login", la propriété value fait référence au contenu de cette zone. (**form.prenom.value**)

- Délimiteur de code javascript :

```
<SCRIPT LANGUAGE="JAVASCRIPT">
 code javascript
</SCRIPT>
```

- Il existe des fonctions en javascript :

```
function nom([param1,… paramN])
{
Instructions
}
```

nom est le nom de la fonction à appeler
param1 à paramN, sont les éventuels paramètres passés à la fonction
Voici un exemple où une fonction JavaScript est appelée dans le corps de la page HTML:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE =”JavaScript”>
        function afficheLeCarre(x)
        {
        document.write (“le carré de “,x, “ est “,x*x ,“.<br>“) ;
        }
</SCRIPT>
</HEAD>
<BODY>
        Test de la fonction <B> carre</B> : <BR>
        <SCRIPT LANGUAGE =”JavaScript”>
                afficheLeCarre (7);
                afficheLeCarre (12);
        </SCRIPT>
</BODY>
</HTML>
```

NB : L'instruction **document.write** fait référence à l'objet document (courant) et utilise la méthode write() pour écrire le texte sur l'objet document.
Le résultat dans notre navigateur est :

```
Test de la fonction carre :
le carré de 7 est 49.
le carré de 12 est 144.
```

- Gestion des événements :

Certaines actions effectuées par un utilisateur dans son navigateur donnent lieu à des événements. La syntaxe utilisée sera :

<Marqueur NomEvenement = "*Code JavaScript*">

Exemple d'utilisation d'événement à partir de la fonction "**carre"**:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE =”JavaScript”>
        function carre()
        {
        document.forms['form1'].resultat.value =
document.forms['form1'].resultat.value*
document.forms['form1'].resultat.value;
        }
</SCRIPT>
</HEAD>
<BODY>
        Test de la fonction <B> carre</B> : <BR>
        <form name="form1">
        <input type ="text" name="resultat" size="25">
        <input type ="button" value="carré" onclick="carre()">
        </form>
</BODY>
</HTML>
```

L'objet *this.form* fait référence au formulaire défini dans le document, chaque clic sur le bouton provoquera l'exécution de la fonction "**carre"**.

- Commentaires entre /* et */

- Objets liés au navigateur :

Window : fenêtre affichée

Par exemple : window.close(); fermera la fenêtre courante.

Location : URL courante et ses propriétés

History : URL déjà visitées

Par exemple : *history.back()* est équivalent au bouton précédent de votre navigateur.

Document : Accès aux propriétés du document courant (titre, couleur d'arrière plan,…)

Par exemple : *document.bgColor = "000000"; mettre un fond d'écran noir*

La syntaxe Javascript est très précise, par exemple,

pour accéder à un objet de type button il faudra écrire :

document.NomFormulaire.nombouton.name →pour obtenir son nom

document.NomFormulaire.nombouton.value →pour obtenir la légende du bouton

pour accéder à un objet de type checkbox, il faudra écrire:

document.NomFormulaire.nombouton.checked →pour obtenir l'état du bouton radio

document.NomFormulaire.nombouton.defaultChecked →pour obtenir l'état par défaut du bouton

document.NomFormulaire.nombouton.name →pour obtenir le nom du bouton radio

document.NomFormulaire.nombouton.value →pour obtenir la valeur du bouton radio

- Les autres objets utiles :
    - Date

      getDate(), getDay(), getMinutes(), getMonth, getSeconds(), getTime(),getYear, setDate(), setDay(), setMinutes(), setMonth, setSeconds(), setTime(),setYear.

Exemple :

```
<SCRIPT LANGUAGE =”JavaScript”>
Aujourdhui = new Date ();
Jour = Aujourdhui.getDay();
Mois = Aujourdhui.getMonth();
Anne = Aujourdhui.getYear();
Document.write("Aujourd'hui, nous sommes le" + Jour + " / " + Mois + " / " + Annee);
</SCRIPT>
```

- Math

  Constantes :

  E, LN2,LN10,LOG2E,LOG10E,PI,SQRT1_2,SQRT2

  disponible via Math.PI ou dans un bloc :

```
with (Math)
    {
      PI…
    }
```

Méthodes :

abs(),acos(),asin(),atan(),ceil(),cos(),sin(),exp(),floor(),log(),max(),min(), pow(),random(),round(),sqrt(),tan()

- String

  Méthodes :

  big(),blink(),bold(),charAt(),fontcolor(),fontsize(),italics(),small(),substring(), toLowerCase(),toUpperCase(),…

Pour manipuler les objets window, vous utiliserez les méthodes suivantes :

alert() : (popup avec message)

close() : fermeture de la fenêtre spécifiée

confirm() : affiche une popup de confirmation contenant un message, un bouton ok et un bouton annuler

open() : ouverture d'une nouvelle fenêtre

prompt() : saisie de donnée dans une popup contenant un message, un bouton ok et un bouton annuler

setTimeout() : Exécution d'une expression après un délai

ID = setTimeout(instruction,valeur);

clearTimeout() : suppression du délai.

clearTimeout(ID);

Il existe énormément de site internet parlant de JavaScript et proposant des scripts tout fait… Mais avant tout, il faudra localiser le fichier JSDOC.ZIP et le télécharger, celui-ci contient toute la documentation spécifique du javascript.

De plus vous trouverez à l'adresse suivante un bon nombre d'objets et de méthodes expliqués en long et en large. http://www.allhtml.com/javascript/index.php

Exercice 1. 1
Réalisez une calculatrice de base (addition, soustraction, multiplication, division) en javascript.

*** Exercice 1.2**
Réalisez, à l'aide de l'objet math une calculatrice scientifique. (Testez-la, à l'aide de la calculatrice sous window, attention aux erreurs d'arrondis!)

2. Les conditions (if then else/switch)

difference entre le "if then else" et le "switch":

```
switch(nombre){
        case "1":
                case "2":
                case "3":document.write("le nombre est < que 4");
                    break;
                case "4":
                case "5":
                case "6": document.write("le nombre est >3 et < 7");
                        break;
                case "7": document.write("le nombre est 7");
                        break;
                case "8": document.write("le nombre est 8");
                        break;
                case "9": document.write("le nombre est 9");
                        break;
                default: document.write("le nombre est >9");
                        break;
                }
```

```
if (nombre<4){
        document.write("le nombre est < que 4");
        }else{
        if (nombre<7){
                document.write("le nombre est >3 et < 7");
                }else{
                if (nombre=7){
                        document.write("le nombre est 7");
                        }else{
                        if (nombre=8){
                                document.write("le nombre est 8");
                                }else{
                                if (nombre=9){
                                        document.write("le nombre est 9");
                                        }else{
                                        if (nombre>9){
                                                document.write("le nombre est >9");
                                        }
                                }
                        }
                }
        }
}
```

Exercice 2.1
Ecrivez un programme qui, en fonction d'une moyenne affichera dans une "popup" le grade obtenu de l'étudiant.
Exemple :

0≤moyenne <12 : refusé
12≤moyenne <14 : satisfaction
14≤moyenne <16 : distinction
16≤moyenne <18 : grande distinction
18≤moyenne <20 : la plus grande distinction

*** Exercice 2.1**
Même question que le 2.1 mais calculez la moyenne vous-même en fonction d'une série de cote et de leur pondération.

3. Validation d'un formulaire en javascript

4. Validation d'un formulaire plus complexe en javascript

Expression régulière

Une expression régulière permet de vérifier si une chaîne de caractères encodée par l'utilisateur correspond bien à une chaîne de caractères prédéfinie.
On utilisera l'objet **RefExp.** Par exemple :

```
MonExpressionReguliere = new RegExp("^a","i")
```

Cette instruction crée une expression régulière, MonExpressionReguliere, qui doit commencer par la lettre "a" ou "A".
Le deuxième paramètre ("i"), facultatif, signifie qu'on ne tient pas compte de majuscule ou minuscule. (si on ne met rien, il fera la différence)
Il faut utiliser un 'backslash" devant les caractères spéciaux, par exemple, la tabulation "\t".

```
MonExpressionReguliere = new RegExp("\\t")
```

Cette instruction crée une expression régulière, MonExpressionReguliere, qui doit contenir un caractère de tabulation.

Liste des caractères spéciaux utilisés dans les expressions régulières:

^	Caractère utilisé pour faire correspondre une lettre en début de chaîne. Par exemple avec "Association malfaiteurs" l'expression régulière "^A" donnera une réponse positive tandis que dans "Membre de l'Association" pas.
$	Caractère utilisé pour faire correspondre une lettre en fin de chaîne.
*****	L'astérisque est utilisé pour faire correspondre 0 ou plusieurs occurrences du caractère qui précède. Par exemple "astro*" renverra une valeur différente de **null** si la chaîne est "astrologue" ou "astrologie" ou encore "astro".
+	Le signe + est utilisé pour faire correspondre 1 ou plusieurs occurrences du caractère qui précède.
.	Le point fait correspondre un seul caractère (sauf le passage à la ligne)
?	Le point d'interrogation est utilisé pour faire correspondre 0 ou 1 fois le caractère qui le précède.
x|y	Fait correspondre soit x soit y .
{n}	Fait correspondre n occurrence de la lettre qui précède. Par exemple "s{2}" fait correspondre les deux "ss" de "mission".
[ ]	Une séries de caractère peut être séparé par des crochets, par exemple [abcde] ou [a-e].
[^xyz]	Cette expression fait correspondre la lettre suivante d'un chaîne de caractère qui commencerait par xyz. Par exemple dans "blanche" l'expression régulière /[^bla]/ fait correspondre la lettre "n".
[\b]	L'expression fait correspondre un caractère d'espacement.
\n	Fait correspondre une nouvelle ligne

\r	Fait correspondre un passage à la ligne
\t	Fait correspondre une tabulation
\w	Fait correspondre tous les caractères alpha-numérique et l'underscore. C'est équivalant à [a-zA-Z0-9_].

```
function verifnombre(x)
{
	expression = "[^0-9]";
	nombreregexp = new RegExp(expression);
	result = x.value.match(nombreregexp);
	if (result !=null )
	{
		return false;
	}
	else
	{
		return true;
	}
}
```

Exemple de validation d'Email en JavaScript

```
function isValidEmail(e)
{
	exp="[a-zA-Z0-9_]+@[a-zA-Z0-9_]+.[a-zA-Z0-9_][a-zA-Z0-9_][a-zA-Z0-9_]?$"
	emailregexp = new RegExp(exp);
	result = e.match(emailregexp);
	if (result != null)
	{
		return true;
	}
	else
	{
		return false;
	}
}
```

4.1 Testez la fonction verifnombre() ci-dessus dans l'exercice 1.1.
4.2 Testez la fonction isValidEmail () ci-dessus dans un formulaire où vous demandez à l'utilisateur d'encoder son adresse Email.
4.3 Créez une fonction qui vérifie qu'une chaîne de caractères ne contient que des lettres.
4.4 Créez une fonction qui vérifie à l'aide d'une expression régulière, un numéro de téléphone encodé.

5. Les répétitives

for

```
for (i=0; i<10 ; i++)
{
```

```
        instruction A
}
```

L'instruction A sera exécutée pour des valeurs allant de i=0 jusqu'à 9, soit 10 fois.

while

```
while (condition){
        instruction B
}
```

L'instruction B sera exécutée tant que la condition est satisfaite.

do while

```
do{
    instruction C
} while (condition)
```

L'instruction C sera exécutée au moins une fois, ensuite tant que la condition est satisfaite.

5.1 Affichez en javascript la table de multiplication par 8 (en utilisant une répétitive!)
5.2 Affichez en javascript la table de multiplication de n'importe quel nombre encodé par l'utilisateur.
5.3 Créez le tableau suivant en HTML à l'aide d'une répétitive en Javascript.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

5.4 Créez un tableau en HTML dans lequel vous affichez 100 couleurs différentes.
5.5 Affichez, en fonction d'un nombre entier n, les différents dessin suivants : (ici n = 4)

```
XXXX
XXXX
XXXX
XXXX
```

```
X
XX
XXX
XXXX
```

```
XXXX
XXX
XX
X
```

```
XXXXXXXX
XXX  XXX
XX    XX
X      X
XX    XX
XXX  XXX
XXXXXXXX
```

```
   X
  XXX
 XXXXX
XXXXXXX
```

5.6 Affichez le calendrier d'un mois donné. Chaque semaine doit être affichée sur une ligne.

Exemple :

Lu	Ma	Me	Je	Ve	Sa	Di
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28				

6. Variables :

var x = "essai"
var y = "essai <B>concluant</B>"
x contient la chaîne de caractère "essai"
y contient la chaîne de caractère "essai **concluant**"

7. Conversion de types :

eval() = évaluation et conversion numérique d'une chaîne.
parseInt() = conversion d'une chaîne en utilisant une base de numération donnée.
parseFloat() = Conversion d'une chaîne en nombre réel.

7.1 Convertir FF en base 16
7.2 Convertir 12 en base 8
7.3 Convertir "essaie" en base 10
7.4 Ecrivez un programme qui lit un en entier, qui vérifie s'il est bien compris entre 1 et 99, et qui, si c'est le cas, affiche à l'écran ce nombre écrit en toutes lettres en français. (Tachez d'éviter un immense switch à 99 cas!)

8. L'instruction for …in

L'instruction "for in" accède séquentiellement aux propriétés d'un objet.

```
For (var in objet)
{
instruction1;
…
instructionN;
}
```

8.1 Affichez toutes les propriétés de l'objet "navigator"
8.2 Affichez toutes les propriétés d'un objet bouton que vous avez créé dans un formulaire.

9. Instruction continue

L'instruction continue interdit l'exécution d'une boucle lorsqu'une condition logique est satisfaite(et passe directement au pas suivant.)

9.1 Ecrivez une répétitive avec un **for** qui va de 1 à 10 par pas de un, et qui affiche à chaque fois le nombre, sauf pour le nombre 7.
9.2 Ecrivez une répétitive avec un **while** qui va de 1 à 10 par pas de un, et qui affiche à chaque fois le nombre, sauf pour le nombre 5.

10. Instruction break

Arrête l'exécution d'une boucle lorsqu'une condition logique est satisfaite.

10.1 Ecrivez une répétitive qui va de 1 à 10 par pas de un, et qui affiche à chaque fois le nombre, mais qui s'arrêtera au nombre 7 et affichera un message de d'arrêt de la répétitive.

11. Instruction return

Permet de renvoyer une valeur dans une fonction. Par exemple la fonction **Carre** vue au premier cours deviendrait :

```
<SCRIPT LANGUAGE =”JavaScript”>
        function carre(x)
        {
        return x*x;
        }
</SCRIPT>
```

Et on l'appellera non plus

carre (7);

mais on placera le résultat dans une variable:

var x;
x = carre(7);

La variable x contiendra la valeur 49.

11.1 Créez une fonction qui renvoie une chaîne de caractère en fonction de l'heure ("bonjour", "bon après-midi", "bonsoir", "bonne nuit", …)

12. Manipulation de chaînes de caractères

Les chaînes de caractères sont des objets de type **string.**

Par exemple dans l'instruction suivante chaine est un objet de type string:

```
chaine = "une chaîne de caractères"
```

l'objet string possède plusieurs propriétés, comme par exemple la longueur d'une chaîne :

```
chaine.length
```

renvoie la valeur 24, car la chaîne contient 24 caractères.

Il existe aussi de nombreuses méthodes associées aux string :

big()	Caractère de grande taille *chaine = "une chaîne de caractère";* *document.write(chaine.big()*)
blink()	Affecte l'attribut clignotant au texte
bold()	Affecte l'attribut gras au texte
charAt()	Renvoie le caractère qui se trouve à la position indiquée *chaine = "une chaîne de caractère";* *document.write("la caractère en position 3 est :" + chaine.charAt(3)*); *document.write("la caractère en position 9 est :" + chaine.charAt(9)*); **nb la position du premier caractère est 0 et pas 1!**
fontcolor()	Définit la couleur des caractères *chaine = "une chaîne de caractère";* *document.write(chaine.fontcolor("#0000F5")*)
fontsize()	Définit la taille des caractères
italics()	Affecte l'attribut italique au texte
indexOf()	Première position d'une sous-chaîne dans une chaîne *chaine = "une chaîne de caractère";* *document.write("le premier e occupe la position" + chaine.indexOf("e")*);

lastIndexOf()	Dernière position d'une sous-chaîne dans une chaîne *chaine = "une chaîne de caractère";* *document.write("le dernier e occupe la position" + chaine.lastIndexOf("e")*);
small()	Caractère de petite taille
strike()	Affecte l'attribut barré au texte
sub()	Affecte l'attribut indice au texte
sup()	Affecte l'attribut exposant au texte
substring()	Extraction d'une sous-chaine *chaine = "une chaîne de caractère";* *document.write(chaine.substring(4,7)*); **affiche"cha"**
toLowerCase()	Conversion en minuscule
toUpperCase()	Conversion en majuscule

Caractère spéciaux :

\b backspace	\f form feed	\n line feed
\r carriage return	\t tabulation	

12.1 Créez une fonction qui va lire une lettre d'un champ texte dans un formulaire et qui vous renvoie un message précisant si la lettre a été encodée en minuscule ou en majuscule.

12.2 Ecrivez une fonction qui reçoit une String et qui la modifie en normalisant les espaces entre les mots : toute séquence de plusieurs espaces doit être remplacée par un seul espace.

Exemple:

String en entrée : "Le JavaScript c'est très simple non ?"

String après modification : "Le JavaScript c'est très simple non ?"

13 Définition d'un nouveau type d'objet

En JavaScript, il est possible de définir ses propres objets (comme l'objet String, Math, …). La création d'un nouvel objet se fait en deux temps :

- définition du type de l'objet à l'aide d'une fonction
- instanciation de l'objet avec l'opérateur **new**

Exemple :

Soit un libraire qui voudrait référencer tous les livres de son magasin, il pourra définir un nouveau type d'objet avec la fonction suivante :

```
function reference(nom, collection, prix, quantite)
{
        this.nom = nom;
        this.collection = collection;
        this.prix = prix;
        this.quantite = quantite;
}
```

Cette fonction affecte 4 propriétés à un nouveau type d'objet appelé référence.

Maintenant pour définir un livre à l'aide de ce nouvel objet, on utilisera, par exemple :

```
S4582 = new reference("Le Monde de Sophie","Seuil",12.5,15);
```

Cette instruction définit l'objet S4582 de type référence et initialise ses propriétés à "Le Monde de Sophie","Seuil",12.5,15

Pour connaître la valeur de la propriété nom (par exemple), on utilisera la syntaxe **S4582.nom.value**

14. Les tableaux

En javascript, aucune instruction n'a été définie pour déclarer un tableau! Mais il très simple de créer un objet dont l'utilisation s'apparente à celle des tableaux dans les autres langages:

```
function DefinitTableau(Nbelem)
{
this.length = NbElem;
var i;
for (i = 1; i<=nbElem; i++)
        {
        this[I] = 0;
        }
return(this);
}
```

Le paramètre passé à la fonction (NbElem) définit le nombre d'éléments du tableau. Ces éléments seront numérotés de 1 à NbElem. En sortie, la fonction renvoie un objet tableau de NbElem initialisé à 0.
Maintenant pour définir un tableau de 20 éléments à l'aide de cette fonction, on utilisera la syntaxe suivante :

```
MonTableau = new DefinitTableau(20)
```

On peut dès lors remplir le tableau en affectant des valeurs à l'objet MonTableau :

```
MonTableau[1] ="Le premier élément";
MonTableau[2] = 3.1415;
MonTableau[3] = "une autre chaîne"
…
```

14.1 Stocker dans un tableau les factorielles de 1 à 10, et utilisez une boucle for pour les afficher.
14.2 On veut savoir si la fonction Math.random() donnant un nombre au hasard fonctionne bien. A cet effet, on va tirer successivement au hasard une grande série d'entiers compris entre 0 et 20, et on va compter le nombre de fois que le 0 est sorti, le nombre de fois que le 1 est sorti, et ainsi de suite jusqu'à 20. Ensuite on va afficher tous ces compteurs. Puis on affichera l'écart maximal entre deux de ces compteurs (la plus petite valeur et la plus grande) et on affichera un message d'avertissement si cet écart maximum/moyenne est supérieur à 1%.

15. L'objet Date

Les objets Date permettent de travailler avec les dates (jours, mois, année) et les heures (heures, minutes, secondes).
Pour définir un objet de type Date:

```
MaDate = new Date();
```

Ou encore :

```
MaDate = new Date(mois, jours, année);
```

Et même :

```
MaDate = new Date(mois, jours, année, heures, minutes, secondes);
```

Il existe aussi de nombreuses méthodes associées à l'objet Date :
getDate() getDay() getMinutes() getMonth getSeconds() getTime() getYear.

15.1 Affichez la date et l'heure dans le document en cours.
15.2 Affichez le nombre de jours restant entre aujourd'hui et votre prochain anniversaire.
15.3 Programmer une fonction qui compare deux dates et renvoie 1 si la première est plus grande que la 2ème, renvoie 0 si elles sont égales, et renvoie –1 si la 2ème est plus grande que la 1er.
15.4 Réalisez un chrono

16. Gestion des images et animations (rollover)

L'animation d'une image est souvent automatisée par des éditeurs d'HTML. Dreaweaver par exemple proposera une insertion d'image "rollover" (une image qui change quand on va dessus avec la souris) automatisé… mais quand on regarde le code que celui-ci génère:

```
<html>
<head>
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_swapImgRestore() { //v3.0
  var i,x,a=document.MM_sr; for(i=0;a&&i<a.length&&(x=a[i])&&x.oSrc;i++)
x.src=x.oSrc;
}

function MM_preloadImages() { //v3.0
  var d=document; if(d.images){ if(!d.MM_p) d.MM_p=new Array();
    var i,j=d.MM_p.length,a=MM_preloadImages.arguments; for(i=0; i<a.length; i++)
    if (a[i].indexOf("#")!=0){ d.MM_p[j]=new Image; d.MM_p[j++].src=a[i];}}
}

function MM_findObj(n, d) { //v4.01
  var p,i,x;  if(!d) d=document; if((p=n.indexOf("?"))>0&&parent.frames.length) {
    d=parent.frames[n.substring(p+1)].document; n=n.substring(0,p);}
  if(!(x=d[n])&&d.all) x=d.all[n]; for (i=0;!x&&i<d.forms.length;i++) x=d.forms[i][n];
  for(i=0;!x&&d.layers&&i<d.layers.length;i++) x=MM_findObj(n,d.layers[i].document);
  if(!x && d.getElementById) x=d.getElementById(n); return x;
}

function MM_swapImage() { //v3.0
  var i,j=0,x,a=MM_swapImage.arguments; document.MM_sr=new Array;
for(i=0;i<(a.length-2);i+=3)
   if ((x=MM_findObj(a[i]))!=null){document.MM_sr[j++]=x; if(!x.oSrc) x.oSrc=x.src;
x.src=a[i+2];}
}
//-->
</script>
</head>

<body>
<a href="#" onMouseOut="MM_swapImgRestore()"
onMouseOver="MM_swapImage('Image1','','image1.jpg',1)"><img src="image2.jpg"
name="Image1" width="120" height="112" border="0"></a>
</body>
</html>
```

Il devient malheureusement fort lourd et difficile à comprendre. Voici un exemple ci-dessous qui effectue exactement la même animation:

```
<html>
<head>
<SCRIPT LANGUAGE = "JavaScript">
   function imgOn() {
       document.MonImage.src = "MonImage_on.jpg";
   }

   function imgOff() {
       document.MonImage.src = "MonImage_off.jpg";
   }
</SCRIPT>
</head>
<body>
<a href="#" onMouseOver = "imgOn()" onMouseOut = "imgOff()"><img
src="MonImage_off.jpg" width="80" height="11" name="MonImage" border="0"></a>
</body>
</html>
```

D'où l'importance de se méfier des générateurs de code Javascript!

16.1 Réaliser dans un même document 5 images disposant d'un **rollover**.

17. Les menus

Il n'existe pas de théorie toute faite pour les menus, on peut simplement créer une série d'images (gif animé par exemple) avec un lien dans différentes rubriques. Ou encore avec des évènements **rollover** sur chacune d'entre elle. Mais cette démarche est un peu longue si on doit souvent ajouter des rubriques. La solution sera donc de regarder ce qui existe déjà sur le net! Le site http://www.editeurjavascript.com/ semble être le plus riche pour vous aider à créer votre propre menu. Le problème sera de bien vérifier la compatibilité avec les différents navigateurs.

- Si vous avez un menu présent sur toutes les pages de votre site, il est clair qu'à chaque nouvelle rubrique que vous ajoutez, vous serez obligé de retoucher toutes vos pages. Il existe cependant plusieurs moyens d'éviter cela. Pour commencer, placez votre menu dans un fichier "menu.html" par exemple. Si votre hébergeur supporte le PHP il vous suffit d'insérer cette ligne dans vos pages (qui doivent être en .php ou .php3)

```
<?
require("menu.html");
?>
```

- Si votre hébergeur supporte le SSI (ASP) il vous suffit d'insérer cette ligne dans vos pages (qui doivent être en .asp ou .shtml)

```
<!-- #include file ="menu.html" -->
```

- Si votre hébergeur ne supporte ni le PHP ni le SSI il faudra insérer

```
<SCRIPT LANGUAGE="JAVASCRIPT" SRC="menu.html"></SCRIPT>
```

Mais malheureusement, le fichier menu.html devra obligatoirement être en javascript

```
document.write('<A HREF="lien1.html">lien</a>');
document.write('<A HREF="lien2.html">autre lien</a>');
…
```

Ce qui est long et fastidieux, il sera donc plus sage de changer d'hébergeur…

17.1 Faites une recherche sur le mot "menu" dans le site http://www.editeurjavascript.com/ et essayez au moins 3 menus différents.

18. Obtenir des informations chez les clients :

Dans l'objet navigateur, vous pouvez trouver toutes les informations nécessaires chez le client, comme par exemple détecter son navigateur :

```
document.write(navigator.appName);
```

la propriété appName donne le nom du navigateur, appCodeName donne le nom de code du navigateur, appVersion donne sa version et userAgent, les informations "UseAgent".

L'objet navigator permet aussi de détecter les plugin installés chez le client :

```
<script language="javascript">
var i,p
p = navigator.plugins.length
document.write("Liste des plugins installés : <br>")
for (i=0 ; i<p ; i++)
        {
        document.write(navigator.plugins[i].name + "<br>")
        }
</script>
```

Les cookies Définition :

Un cookie est un enregistrement d'informations par le serveur dans un fichier texte situé sur l'ordinateur client, informations que ce même serveur (et lui seul) peut aller relire et modifier ultérieurement. Un cookie est obligatoirement rattaché à un nom de domaine et un ensemble d'URL de telle sorte que seule une requête provenant du même serveur pourra y accéder.

Exemple de création d'un cookie en javascript:

```
<script language="javascript">
function creationCookie (nom,prenom) {
date = new Date;
date.setMonth(date.getMonth()+3);
document.cookie = "nom =" + nom + "; expires=" + date.toGMTString();
document.cookie = "prenom =" + prenom + "; expires=" + date.toGMTString();

}
</script>
```

remarque :

expires : durée de vie du cookie.(3 mois à partir d'aujourd'hui)

Pour savoir lire dans un cookie, il faut d'abord comprendre d'instruction split : l'instruction "split" construit un tableau de sous-chaîne à partir d'une chaîne de caractère selon un "délimiteur" :

```
myString = new String("A,B,C,D")
splitString = myString.split(",")
document.write("<BR>" + splitString[0])
document.write("<BR>" + splitString[1])
document.write("<BR>" + splitString[2])
document.write("<BR>" + splitString[3])
```

Nous pouvons maintenant décomposer notre cookie :

Exemple de lecture de cookies en javascript:

```
Montableau = document.cookie.split(";")
nom = Montableau[0].split("=")[1]
prenom = Montableau[1].split("=")[1]
```

```
document.write("nom : " + nom +"<br>");
document.write("prénom : " + prenom +"<br>");
```

Liens utiles :
http://www.anguenot.com/cookies.php
http://www.dreamweaver-forum.net/viewtopic.php?t=293
http://kridamaury.membres.jexiste.org/java/cookies.php3

Exercice :

18.1 Comment vérifier si le client accepte les coockies ?
18.2 Testez si le client possède un plug-in flash, sinon envoyez-le vers une page html (noflash.html) avec un message d'avertissement.
18.3 Créez un cookie chez un client
18.4 Créez un cookie chez un client afin de compter son nombre de visite.
18.5 testez le débit de connexion du client
http://www.toulouse-renaissance.net/c_outils/c_debit.htm

19. Utilisation de fonctions existantes

19.1 Recherchez insérez et utilisez la fonction overlib sur internet.
19.2 Recherchez et insérez un script d'image aléatoire.
19.3 Faire défiler un texte horizontalement dans un input text.
19.4 Faire défiler un texte dans la barre de status.
19.5 Faire tomber des flocons de neige sur votre écran.
19.6 Entourer le pointeur de la souris d'images

20. Exercices récapitulatifs

20.1 Créez le calendrier de l'année en fonction d'une date choisie.
20.2 Réalisez un pendu.

21. DHTML : Les calques

Définition : un calque est une partie d'une page web manipulable comme un élément séparé : il peut-être mis à jour, repositionné, affiché ou masqué. C'est donc encore un nouveau moyen de rendre les pages plus dynamiques chez le client. Plusieurs calques peuvent se chevaucher, les calques d'arrière-plan étant visibles à travers les calques de premier plan transparents.

La balise <DIV></DIV> :

Exemple :

```
<DIV ID="calque1" STYLE="position:absolute; left:100;top:100">
voici le contenu du calque.
</DIV>
```

Il s'agit ici d'un calque situé à 100 pixels au-dessous et à droite du coin supérieur gauche de la fenêtre du navigateur.

Propriétés de l'attribut STYLE :

- position :
  - static : éléments affichés de manière normale, ils ne peuvent pas être déplacés.
  - absolute : élément décalé en fonction des coordonnées spécifiées.

    - relative : élément décalé en fonction de sa position static (c'est à dire la position qu'il occuperait s'il s'agissait d'un élément html "normal")
- left et top : coordonnées de l'élément
- width et height : largeur et hauteur de l'élément
- z-index : manière dont les éléments se superposent
- visibility : active ou désactive la visibilité de l'élément
    - visible (visible)
    - hidden (caché)
    - inherit (hérite de la visibilité de l'élément parent; un calque, un tableau, un paragraphe…)
- background-color : couleur de l'arrière plan du texte du calque
- layer-background-color : couleur de l'arrière plan du calque, qu'il contienne du texte ou non.
- background-image : image d'arrière plan pour le texte du calque
- layer-backgroud-image : image d'arrière plan pour le calque, qu'il contienne du texte ou non.

La balise <NOLAYER></NOLAYER>

Permet d'afficher un message pour les navigateurs qui ne prennent pas en charge les calques.

Exemple :

```
<NOLAYER>
Cette page fait appel à des calques et nécessite l'emploi de Netscape 4 et Internet Explorer 4 ou ultérieur.
</NOLAYER>
```

remarque:

Netscape traite les calques de la même manière, qu'ils soient créés à l'aide de la balise <DIV> ou de la balise <LAYER>; ces deux types de calques seront représentés par l'objet layer. Mais sous Netscape, avec <DIV>, le navigateur Netscape a tendance à "se planter", donc il vaut mieux préconiser <LAYER>.

Mais les propriétés de l'attribut style ne seront pas les même d'un navigateur à l'autre; on aura par exemple pour Explorer :

visibility:hidden

et pour netscape :

visibility:hide

Exemple d'un script qui permet d'afficher ou de masquer un calque aussi bien sur explorer que netscape:

```
<html>
<head>
<SCRIPT LANGUAGE="JavaScript">
function afficherMasquer(valeur)
{
	if (valeur==0){// on masque le calque
		if (document.layers)
			{document.layers["calque1"].visibility = "hide";}
		else
			{document.all["calque1"].style.visibility = "hidden";}
	}
```

```
        else
        {//on affiche le masque
                if (document.layers)
                        {document.layers["calque1"].visibility = "show";}
                else
                        {document.all["calque1"].style.visibility = "visible";}
        }
}
</SCRIPT>
</head>
<body>
<form name="form1">
<input type="button" value="afficher le calque" onClick="afficherMasquer(1);">
<input type="button" value="masquer le calque" onClick="afficherMasquer(0);">
</form>
<div id="calque1" style="postion:absolute; left:35;top:210;background-
color:orange;width:250;height:50;visibility:hidden">
blabla blabla blabla blabla blabla blabla blabla blabla blabla <br>
blabla blabla blabla blabla blabla blabla blabla blabla blabla <br>
</div>
</body>
</html>
```

Liens utiles :
http://www.commentcamarche.net/dhtml/dhtmllayer.php3
http://www.selfhtml.com.fr/articles/dhtml/positionnement/

21.1 Réalisez un formulaire dans lequel vous créez un calque visible ou non en fonction d'une liste déroulante.