



# **Langage javascript**

## **1ère Partie**

1



## **Plan**

- Connaître l'intérêt du développement côté client
- Le langage javascript
- Les fonction et objets built-in
- Les expressions régulières
- Les objets du navigateur
- Les cookies

2

# Présentation

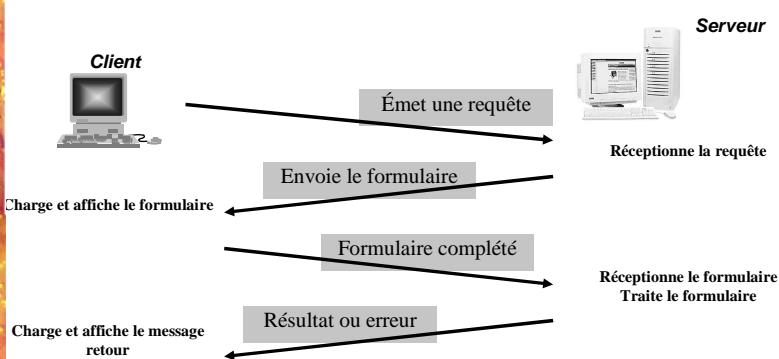
- JavaScript langage interprété
- Pas d'interprète en dehors du browser
- Développé initialement par NetScape (Livescript)
- Adopté en 1995 par Sun
- Versions

| Versions       | Apparue dans                    |
|----------------|---------------------------------|
| Javascript 1.0 | Netscape 2.0                    |
| Javascript 1.1 | Netscape 3.0                    |
| Javascript 1.2 | Netscape 4.0-4.05(Communicator) |
| Javascript 1.3 | Netscape 4.06-4.5               |
| Javascript1.4  | Netscape 7                      |
| Javascript1.5  | Netscape 8                      |

|   |              |
|---|--------------|
| Quelque chose qui ressemble à du Javascript 1.0 | Explorer 3.0 |
| Javascript 1.2                                  | Explorer 4.0 |
| Jscript5  | Explorer 5.0 |

3

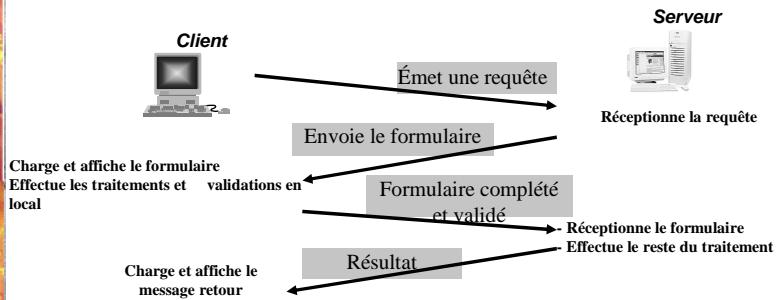
# Avant Javascript



- Majorité des traitements chez le serveur
- Client inactif

4

## Arrivée du Javascript



- Alléger les serveurs et le réseau en déportant le traitement sur le client
- Rendre le poste client intelligent, autonome
- Rendre les pages HTML plus dynamiques.

5

## Fonctionnalités

- Les applications nécessitant du calcul en local (Total d'une note de frais, Statistique....)
- Les pages web plus conviviales ( Animations, interactivité...)
- La gestion des évènements (Click, Double click, Passage de la souris sur une zone,...)
- La maîtrise de tous les éléments du navigateur(Window, Bouton,.....)
- La génération dynamique du code HTML
- Tous les contrôles de saisie (Champs obligatoires, taille des champs)
- Manipulation facile des cookies

6

## Javascript n'est pas Java

| Javascript  | Java                                     |
|---|--|
| Code intégré dans la page Html  | Module (applet) distinct de la page Html |
| Code interprété par le browser au moment de l'exécution                 | Code source compilé avant l'exécution    |
| Codes simples<br>Applications limitées                                  | Langage complexe mais plus performant    |
| accès aux objets du navigateur  | Pas d'accès aux objets du navigateur     |
| La majorité de code source est visible (il faut utiliser les biblio.js) | Sécurité (code source compilé)           |

7

## Outils pour écrire du code JavaScript

- Beaucoup d'éditeurs HTML intègrent parfaitement JavaScript (Visuel Studio Code, Dreamweaver, etc.)
- Pour apprendre à programmer « proprement », utiliser un simple traitement de texte (TextPad ou NotePad).

8

# Intégration du JavaScript dans HTML

Méthode 1 : Balise ajoutée au langage HTML <SCRIPT>

- Syntaxe générale

```
<SCRIPT LANGUAGE="nom_langage" [SRC="URL"]>  
//code du script  
</SCRIPT>
```

\* nom\_langage : JavaScript, JavaScript1.1, JavaScript1.2, ..  
\* URL : fichier javascript d'extension '.js'  
\* ID et Archive : optionnels, signature digitale des scripts

- Exemple

```
<HTML>  
<HEAD>  
<!-- Insertion d'une librairie JavaScript  
-->  
<SCRIPT SRC="lib.js"></SCRIPT>  
</HEAD>  
<BODY>  
.....
```

Fichier Lib.js

```
//Librairie de fonction  
//javascript à partir de laquelle  
//cette fonction est appellée  
//fonction qui ferme le browser  
function fin()  
{ wwindow.close()  
}  
.....
```

9

# Intégration du JavaScript dans HTML

Méthode 2 : Utilisation de l'attribut HREF de la balise <A> ...

- Exécution du code en un simple click sur le lien
- Utile aux appels des fonctions définies dans le document
- L'URL courante est remplacée par le script
- Syntaxe
- <A HREF= "JavaScript : codescript ">texte </A>

Exemple

```
<HTML><HEAD>  
<TITLE>Mon premier Javascript</TITLE>  
<SCRIPT LANGUAGE="Javascript">  
function fin(){  
window.close()  
}  
</SCRIPT>  
</HEAD>  
<BODY>  
<A HREF="JavaScript:fin()>">ici</A>  
<BR>  
</BODY></HTML>
```

10

## Intégration du JavaScript dans HTML

- Méthode 3 : Utilisation de nouveaux attributs de gestion d'événements ( onMouseOver, onClick...)
- Syntaxe :
  - <BALISE onEvenement="CodeJavaScript">
- Exemple

```
<HTML><HEAD>
<TITLE>Mon premier Javascript</TITLE>
<SCRIPT LANGUAGE="Javascript">

function fin(){
    window.close()
}
</SCRIPT>
</HEAD>
<BODY>
En passant la souris au-dessus de
<A HREF="" onMouseOver="fin()">cela</A>
</BODY></HTML>
```
- <NOSCRIPT> ... </NOSCRIPT> pour les anciens browser

11

## Les commentaires

- Code lisible et facile à corriger
- Utilisation des conventions de C et C++
  - Utilisation de deux barre obliques //
- Insertion de plusieurs lignes de commentaire /\* .... \*/

```
<SCRIPT LANGUAGE = "Javascript">
// on écrit bonjour
document.write('bonjour et ') ;
// on écrit au revoir
document.write('Au revoir') ;
</SCRIPT>
```

```
<SCRIPT LANGUAGE = "Javascript">
/*on écrit bonjour
puis
on écrit Au revoir
*/
document.write('bonjour et ') ;
document.write('Au revoir') ;
</SCRIPT>
```

12

## Les opérateurs

- Ces opérateurs ont la même signification que dans C
- Liste des opérateurs du degré de priorité le plus faible au plus élevé

| Opération        | Opérateur                      | Exemple  |
|------------------|--------------------------------|--|
| ,                | virgule ou séparateur de liste | var a,b,c;   |
| = += -= *= /= %= | affectation                    | x += y équivaut à x= x+y   |
| ? :              | opérateur conditionnel         | (condition) ? valeur_si_vrai : valeur_si_faux  |
|                  | ou logique                     | (true)    (false)  |
| &&               | et logique                     | (true) && (false)  |
| = !=             | égalité                        | if (a==b) équivaut à si a égale à b<br>if (a!=b) équivaut à si a est différent de    |
| < <= > >=        | relationnel                    | if (i>30) instruction;   |
| + -              | addition soustraction          | a= 25+10;  |
| * /              | multiplier diviser             | b=10/5;  |
| ! - ++ --        | unaire                         | vrai = !(false)<br>x=++y équivaut à y= y+1 et x=y ;<br>x=y++ équivaut à x=y et y=y+1 |
| ()               | parenthèses                    | a=(25+2)*5   |

13

## Les types de base

### ■ Syntaxe de C

#### ■ Les nombre : number

##### Entiers

Base 16 : 0x14  
Base 10 : 20  
Base 8 : 010

##### Réels

var Pi =3.141592654  
var reel = 6.25E-3

#### ■ Les booléens : boolean

*Il\_fait\_beauc = true ;  
Fin\_du\_cours = false ;*

#### ■ Les chaîne de caractère

*"Salut tout le monde"  
'c'est une simple apostrophe : |' ↲→ "c'est une simple apostrophe"*

14

## Déclaration des variables

- Une variable référence :
  - des nombres
  - des chaînes de caractères
  - des booléens
  - des objets
- Déclaration optionnelle
- Elle se fait par le mot var
- Javascript est faiblement typé :
  - Type de variable imprécis : var toto ; (type undefined)
  - Type définie par la valeur d'initialisation : var entier = 10 ;
  - Changement du type de variable
- JavaScript fait la différence entre Maj et Min
- La fonction typeof
  - typeof 'bonjour' retourne String
  - typeof (3.14) retourne number
  - typeof (false) retourne boolean
  - typeof (nimportequoi) retourne undefined

15

## L' instruction: if ....else

- Choix entre deux blocs d'instructions
- Syntaxe :

```
if (condition) {  
    Instruction1  
    Instruction N  
} else {  
    Instruction 1  
    Instruction M  
}
```

```
if (condition1) {  
    bloc d'instructions1  
} else if(condition2) {  
    bloc d'instruction2  
} else {  
    bloc d'instruction3  
}
```

### Exemple

```
if (age >=18) {  
    document.write('adulte');  
} else {  
    document.write('mineur');  
}
```

16

## L' instruction: switch

- Exécution d'un bloc selon l'expression donnée
- Comparaison de l'expression aux #tes valeurs constantes
- Pas d'opérateur relationnel à l'intérieur de switch
- Pas d'égalité avec une variable ou une expression.
- break : Provoque l'exécution de la première instruction suivant le bloc de switch.
- Syntaxe :

```
switch (expression){  
    case const1 :      instruction1;  
        break;  
    case const2 :      instruction2  
        break  
    default :         instruction3  
}
```

17

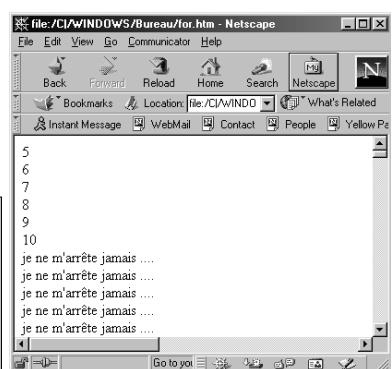
## Instruction for

- Répétition d'une série d'instruction
- Syntaxe

```
for (expr_initiale; condition; expr_repetee)  
{    Instruction1  
.....  
    InstructionN  
}
```

- Exemple

```
<script language="JavaScript1.2"></script>  
for (i = 5 ; i <= 10 ; i++){  
    document.write(i + '<BR>')  
}  
// boucle infini  
for ( ; ; ) {  
    document.write('je ne m\'arrête jamais ....' + '<BR>')  
}  
</script>
```



18

## Instruction for...in

- Variable balayant les index d'un tableau ou les propriétés d'un objet

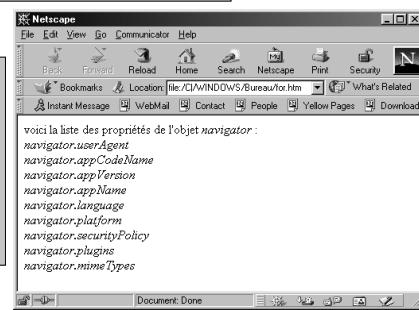
- Syntaxe

```
//boucle pour un tableau  
for (variable in tableau){  
    Instruction 1  
    .....  
    Instruction N  
}
```

```
//boucle pour un objet  
for (variable in objet) {  
    Instruction 1  
    .....  
    Instruction N  
}
```

- Exemple

```
<script language =JavaScript1.2>  
document.write('voici la liste des propriétés de l\'objet <I>  
navigator</I> : <BR>')  
for (prop in navigator){  
document.write('navigator.'+italics() + prop.italics() +  
'<BR>')  
}  
</script>
```



19

## Instruction while

- Répétition d'un bloc d'instruction

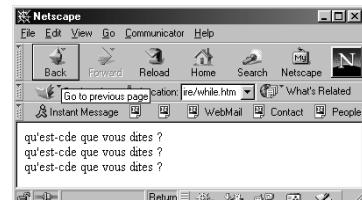
- Syntaxe

```
do {  
    instruction 1  
    .....  
    instruction N  
} while (condition)
```

```
While (condition) {  
    instruction 1  
    .....  
    instruction N  
}
```

- Exemple

```
<script language ="JavaScript1.2">  
//on exécute 3 fois les mêmes commandes  
var compteur = 0;  
while (compteur<3) {  
document.write('qu'est-ce que vous dites ?' + '<BR>')  
compteur = compteur+1  
}  
</script>
```



- Instruction continue : Interruption de l'itération courante et passage à la suivante

20

# Les fonctions

- Ensemble d'instruction réunies pour une action
- Passage des paramètres
  - Les chaînes de caractères, nombres et booléens passent en valeur
  - Les tableaux et les objets passent en références
- Syntaxe

```
//fonction sans valeur en retour
function nom_Function (Arg1, Arg2,...,ArgN)
{
    instruction1
    ...
    InstructionN
}
```

```
//fonction retournant une valeur
function nom_Function (Arg1, Arg2,...,ArgN)
{
    instruction1
    ...
    InstructionN
    Return valeur
}
```

21

# Les fonctions

## Exemple

```
<script></script>
var chaîne = 'bonjour';
function Change(param) {
    //on change la valeur du paramètre
    param = 'Au revoir'
}
//la valeur de la chaîne avant l'appel
document.write('la valeur de la chaîne avant l'appel : ' + chaîne + '<BR>');
//appel de la fonction
Change(chaîne);
// la valeur de la chaîne après l'appel
document.write('la valeur de la chaîne après l'appel : ' + chaîne + '<BR>');
</script>
```



22

## JavaScript et la notion d'objet

- Javascript n'est pas un véritable langage OO :
  - La notion d'héritage entre les classes manque
  - La définition des classe est simpliste
- Un objet est une structure comportant des champs :
  - attributs
  - méthodes
- Syntaxe

```
//Méthodes
function nom_methode1(arg1,arg2,...,argN)
{
    bloc d'instructions
}
// Constructeur
function constructeur_objet{
    this.methode1 = nom_methode1 ;
    this.attribut1= attribut1 ;
    this.attribut1= attribut1 ;
}
```

- Création : var instance\_objet = new constructeur\_objet();
- Appel des méthodes :

```
Instance_objet.nom_methode1(arg1,arg2,...,argn) ;
```

23

## JavaScript et la notion d'objet

```
<SCRIPT LANGUAGE="JavaScript">
function str2Date(strDate)
{
    TDate=strDate.split("/");
    return new Date(TDate[2], TDate[1]-1, TDate[0])
}

function classPersonne(nom, prenom,dateNaissance)
{
    this.Nom=nom;
    this.Prenom=prenom;
    this.DateNaissance=new str2Date(dateNaissance);
    this.Age= function(){
        dateSysteme=new Date();
        return dateSysteme.getFullYear()-this.DateNaissance.getFullYear();}
}

function classFonctionnaire(nom, prenom, dateNaissance, dateEmbauche){
    this.base=classPersonne;
    this.base(nom, prenom, dateNaissance);
    this.DateEmbauche=new str2Date(dateEmbauche);
    this.Anciennete=function(){
        dateSysteme=new Date();
        return dateSysteme.getFullYear()- this.DateEmbauche.getFullYear(); }
}

oFonctionnaire = new classFonctionnaire("BOUSSALAM", "Yasser",
"29/10/1969","02/06/2004");
document.write(oFonctionnaire.Nom+ " " + oFonctionnaire.Prenom+ " " +
oFonctionnaire.Age()+ " " + oFonctionnaire.Anciennete());
</SCRIPT>
```



## Les fonctions built-in de JavaScript

### Fonctions prédéfinies dans javascript

- `parseFloat(chaine)` : Convertit chaîne de caractères en valeur entière.
  - Exemple : `parseFloat('1.414')` retourne le réel 1.414.
- `parseInt(chaine)` : Convertit une chaîne de caractères en entier.
  - Exemple : `parseInt('1.414')` retourne 1.
- `isNaN(expression)` : Vérifie la validité de l'expression.
  - Exemple : `isNaN(parseInt('rien'))` retourne false.
- `eval(chaine)` : Evalue la chaîne de caractères comme une expression.
  - Exemple : `var x =5`
    - `document.write('la valeur de X2+3x-3 est :'+eval('x*x+3*x-3'))`
    - la valeur retournée de `X2+3x-3` est : 37
- `escape(caractere)` : convertit le caractère de ISO Latin en code ASCII
  - Exemple : `escape("é")` retourne %E9
- `unescape(chaine)` : convertit la chaîne en caractères ISO Latin
  - Exemple : `unescape("%E9")` retourne 'é'

25



## Les objets de JavaScript

- Deux types d'objets prédéfinis :
- Objets built-in définis par l'interpréteur Javascript
  - Gestion facile des types de données les plus courants
  - Liste complète des objets built-in : `String()`, `Date()`, `Image()`, `Option()`, `Math`, `Array()`
- Objets du navigateur client (Netscape, Internet Explorer,...)
  - Gestion des objets du navigateur
  - Liste des objets du navigateur : `window`, `navigator`, `document`, `history`, `forms`, ....

26

# Les objets built-in de JavaScript

## ▪ L'objet String

- Pas nécessaire pour la création d'une chaîne  
( var chaine='bonjour')
- But : respecter la syntaxe objet
- Syntaxe :

```
var maChaine = new String()  
var maChaine = new String(chaine)
```
- Exemple
- var pays = new String('Maroc') ⇔ var pays ='Maroc'

27

## Méthodes de l'objet String

| Méthodes         | Description   |
|------------------|---|
| Anchor(name)     | renvoie la chaîne de caractères sous forme d'un lien interne avec le name exprimé en argument |
| bold             | renvoie la chaîne de caractères sous format gras  |
| Fontcolor(color) | renvoie la chaîne de caractères avec une couleur exprimée en argument                         |
| FontSize(val)    | renvoie la chaîne de caractères avec une taille exprimée en argument                          |
| italics          | renvoie la chaîne de caractères sous forme italique   |
| Link(URL)        | renvoie la chaîne de caractères sous forme de lien dans l'URL est exprimé en argument         |
| sub              | renvoie la chaîne de caractères sous forme d'indice   |
| sup              | renvoie la chaîne de caractères sous forme d'exposant   |
| toLowerCase      | renvoie la chaîne de caractères convertie en minuscules                                       |
| toUpperCase      | renvoie la chaîne de caractères convertie en majuscules                                       |

28

## Traitement des chaînes de caractères

| Méthodes             | Description   |
|----------------------|---|
| charAt(pos)          | Retourne le caractère de la chaîne, situé à la position 'pos'                                   |
| Contact(ch)          | Concaténation de chaîne   |
| indexOf(ch, pos)     | Renvoie la position de la première occurrence de 'ch' en commençant de la position 'pos'        |
| lastIndexOf(ch, pos) | Renvoie la position de la dernière occurrence de 'ch' en commençant de la position 'pos'        |
| substr(pos,long)     | retourne la sous débutant de la position 'pos' et de longueur 'long'                            |
| Slice(debut, fin)    | Retourne la sous-chaîne débutant à la position 'debut' et se terminant un caractère avant 'fin' |
| split(sep)           | Retourne un tableau de sous-chaîne obtenues en découplant la chaîne selon le séparateur 'sep'   |
| substr(pos1,pos2)    | retourne la sous chaîne entre 'pos1' et 'pos2'  |

29

## Les objets built-in de JavaScript

- L 'objet Date : Manipulation des dates avec javascript
- Syntaxe :
  - var nomObjet = new Date()
  - var nomObjet = new Date('mois jourDuMois,année heures:minutes:secondes')
  - var nomObjet = new Date(année, mois, jours)
  - var nomObjet = new Date(année, mois, jours, heure, minute, seconde)

Exemple :

- var dateduJour =Date() // dateduJour vaut : 20/06/2000
- var MaDate = new Date("August 29, 1969 12:00:00");
- var MaDate = new Date(1969,7,29);

30

## Méthodes pour les dates

| Méthodes             | Description   |
|----------------------|---|
| getDate()            | revoit le numéro du jour (1 à 31) dans le mois de la date considérée  |
| getDay()             | revoit le numéro du jour (0 pour Dimanche, à 6 pour Samedi) dans la semaine de la date considérée.  |
| getHours()           | revoit le nombre d'heures passées (0 à 23) dans le jour de la date sur laquelle la méthode est appelée.   |
| getMinutes()         | revoit le nombre de minutes passées (0 à 59) dans l'heure de la date sur laquelle la méthode est appelée.   |
| getMonth             | retourne le numéro du mois (0 à 11) de la date sur laquelle la méthode est appelée.   |
| getTime()            | retourne la date en unités internes de temps (soit en nombre de millisecondes écoulées depuis le 1er janvier 1970). Ceci est utile pour faire des comparaisons entre différentes dates : t1.getTime() < t2.getTime() => t1 précède t2). |
| getFullYear()        | retourne l'année de la date sur laquelle la méthode est appelée.  |
| setDay, setDate, ... | Positionne respectivement le numéro du jour de la semaine, numéro du jour du mois, ....   |

31

## Les objets built-in de JavaScript

### Exemple

```
<script></script>
function Semaine(){
    this[0] = "Dimanche"; this[1] = "Lundi";
    this[2] = "Mardi";   this[3] = "Mercredi";
    this[4] = "Jeudi";   this[5] = "Vendredi";
    this[6] = "Samedi";
}
function Mois(){
    this[0] = "Janvier";  this[1] = "Fevrier";
    this[2] = "Mars";     this[3] = "Avril";
    this[4] = "Mai";      this[5] = "Juin";
    this[6] = "Juillet";  this[7] = "Aout";
    this[8] = "Septembre"; this[9] = "Octobre";
    this[10] = "Novembre"; this[11] = "Decembre";
}

var semaine=new Semaine(); // Tableau des noms des jours
var mois=new Mois();      // Tableau des noms des mois
var myDate=new Date();    // On récupère la date actuelle
var result=semaine[myDate.getDay()]+''+myDate.getDate()+' '+
mois[myDate.getMonth()]+''+(myDate.getFullYear());

document.write("la date du système est : " + result)
</Script>
```



32

## Les objets built-in de JavaScript

- L'objet Math : Manipulation des opérations mathématiques
- Ne possède pas de constructeur (pas d'utilisation de new)
- Classe instanciée
- Propriétés : Les principales constantes mathématiques (E, PI,...)
  - Syntaxe Math.NomDeLaConstante
- Méthodes : Les fonctions mathématiques (abs,acos,...)
  - Syntaxe Math.NomDeLaMethode (Arg1,Arg2,...)

33

### Méthodes pour Math

| Méthodes            | Description   |
|---------------------|---|
| Math.E              | retourne la valeur mathématique e (~2.71828).   |
| Math.PI             | retourne la valeur de Pi (~3.141592654).  |
| Math.pow(val,expo)  | calcule la valeur de celle du premier paramètre (value) élevée à la puissance de la valeur du second paramètre (exp). |
| Math.sqrt(value)    | calcul la racine carrée de la valeur passée en paramètre.   |
| Math.exp(value)     | calcule la valeur exponentielle de celle passée en paramètre.   |
| Math.floor(value)   | retourne l'entier le plus proche par défaut de la valeur passée en paramètre  |
| Math.cos(value)     | calcule le cosinus de l'angle passé en paramètre (en radian).   |
| Math.log(value)     | calcule le logarithme de la valeur passée en paramètre.   |
| Math.max(val1,val2) | retourne la plus grande des deux valeurs passées en paramètres.   |
| Math.min(val1,val2) | retourne la plus petite des deux valeurs passées en paramètres.   |
| Math.random()       | retourne une valeur aléatoire comprise entre 0 et 1.  |
| Math.tan(value)     | calcule la tangente de l'angle passé en paramètre (en radian).  |
| Math.abs(value)     | calcule la valeur absolue du nombre passé en paramètre.   |

## Les objets built-in de JavaScript

- L'objet Image : Manipulation des images avec javascript
  - **Gestion dynamique et simple des graphiques**
  - **Charge les images à travers le réseau (équivaut à <IMG>)**
  - **Remplace une image insérée par <IMG> sans recharger le doc**
  - **Différence avec <IMG> : l'image stockée en mémoire sans affichage**
- Syntaxe :
  - nom\_variable = new Images()

Propriétés :

| Propriétés | Description  |
|------------|--|
| scr        | Chaîne de caractères contenant l'URL du fichier à charger (le chargement n'est pas bloquant) |
| height     | Hauteur de l'image en pixels   |
| width      | Largeur de l'image en pixels   |
| complete   | Booléen indiquant si le chargement de l'image est terminé                                    |

35

## Les objets de JavaScript

Exemple :

```
<html><head>
<script language ="JavaScript1.2">
var imageVide = new Image() ;var imagePrintemps = new Image() ;
var imageEte = new Image() ;var imageAutomne = new Image() ;
var imageHiver = new Image() ;
imageVide.src = "/images/photo_vide.jpg";
imagePrintemps.src = "/images/photo_printemps.jpg"
imageEte.src = "/images/photo_ete.jpg"
imageAutomne.src = "/images/photo_automne.jpg"
imageHiver.src = "/images/photo_hiver.jpg"
function afficheImage(saison){
switch (saison) {
case 'printemps' :
document.images[0].src = imagePrintempsVide.src
break
case 'ete' :
document.images[0].src = imageEte.src
break
case 'automne' :
document.images[0].src = imageAutomne.src
break
case 'hiver' :
document.images[0].src = imageHiver.src
break
default :
document.images[0].src = imageVide.src
}
}
</script>
```



36

## Les objets built-in de JavaScript

- L'objet Array : Création d'un tableau d'entité (nombre, chaîne, objet, ou tableau.)
- Syntaxe :
  - Var nomTableau = new Array();
  - Var nomTableau = new Array(10)
  - Var nomTableau = new Array(valeur1, valeur2,...valeurN)
- Propriétés :
  - length : taille du tableau
- Méthodes :

| Méthode               | Description   |
|-----------------------|---|
| concat(tableau1)      | Concatène deux tableau  |
| sort(fonctionCompare) | Classe les éléments du tableau selon la fonction fonction_compare   |
| join(separateur)      | Retourne une chaîne de caractères composée de tous les éléments du tableau séparés par la chaîne séparateur             |
| reverse()             | Inverse l'ordre des éléments du tableau   |
| slice(debut,fin)      | Retourne un nouveau tableau composé des éléments à partir de l'index debut et s'arrêtant à un élément avant l'index fin |

37

## Les expressions régulières

- Objectif : Simplifient grandement le traitement des chaînes (recherche et/ou substitution)
- Largement utilisées dans le monde d'Unix (awk, Perl, ....)
- Principe
  - Utilisation d'un modèle décrivant toutes les chaînes à rechercher
- Syntaxe
  - Il existe deux types de syntaxe :
    - Reg=/model/
    - Reg = new RegExp('modele') // propre à Javascript
  - Les options :
    - g : indique la recherche de toutes les correspondances
      - Reg=/model/g ou Reg = new RegExp('modele', 'g')
    - i : faire la recherche sans tenir compte de la casse
      - Reg=/model/i ou Reg = new RegExp('modele', 'g')
    - On peut combiner les deux options
      - Reg=/model/gi ou Reg = new RegExp('modele', 'gi')

38

## Les expressions régulières

- Méthodes :

| Méthode        | Description  |
|----------------|--|
| exec(chaine)   | Exécute une recherche du modèle dans la chaîne chaîne et renvoie un tableau :<br>A l'index 0 : la dernière correspondance du modèle<br>Au index 1 à n : les chaînes mémorisées correspondantes |
| test(chaine)   | Renvoie un booléen indiquant si une correspondance du modèle a pu être trouvée dans la chaîne chaîne   |
| search(chaine) | retourne l'indice dans la chaîne, auquel l'expression régulière a été trouvée. S'il ne la trouve pas, la méthode retourne -1.  |

- Les caractères spéciaux :

| Syntaxe | Description   |
|---------|---|
| .       | Correspond à n'importe quel caractère   |
| \       | Indique que le caractère qui suit ne doit pas être considéré comme un caractère spécial |
| \f      | Correspond à un saut de page  |
| \n      | Correspond à un saut de ligne   |
| \r      | Correspond à un retour chariot  |
| \t      | Correspond à une tabulation   |

- Gestion des positions

| Syntaxe   | Description   |
|-----------|---|
| ^         | Indique que la correspondance doit avoir lieu en début de la chaîne de caractères |
| \$        | Indique que la correspondance doit avoir lieu en fin de chaîne                    |
| Mot1 mot2 | Indique que la correspondance peut se faire sur mot1 ou mot2                      |

## Les expressions régulières

- Les classes de caractères :

| Syntaxe | Description   |
|---------|---|
| [abc]   | Correspond à tous les caractères présents entre les deux crochets.<br>Pour indiquer une suite de caractères on utilise le caractère - |
| [^abc]  | Correspond à tous les caractères non présents entre les deux crochets.  |
| \d      | Correspond à un chiffre   |
| \D      | Correspond à tout ce qui n'est pas un chiffre   |
| \w      | Correspond à tout ce qui est composé de lettres, chiffres ou de caractère _   |
| \W      | Correspond à tout ce qui n'est pas composé de lettres, chiffres ou de caractère _   |
| \s      | Correspond à un espace, retour à la ligne, retour chariot ou tabulation   |
| \S      | Correspond à tout ce qui n'est pas un espace, retour à la ligne, retour chariot ou tabulation   |

- Gestion du nombre d'occurrences

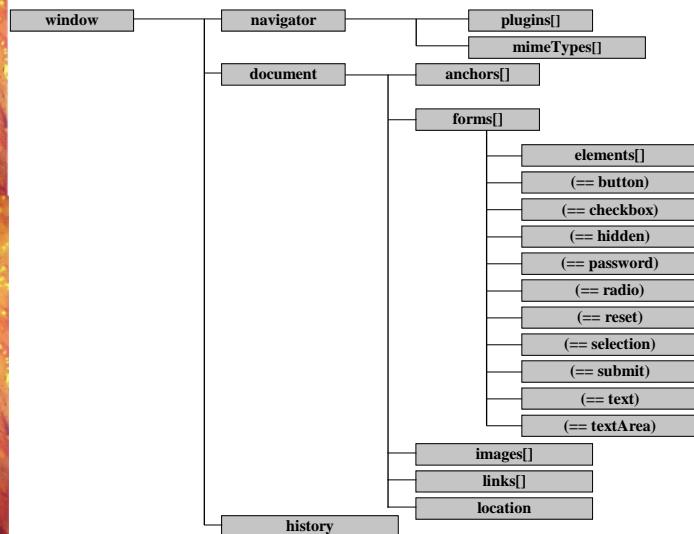
| Syntaxe   | Description  |
|-----------|--|
| *         | Le caractère doit apparaître 0 ou n fois                           |
| +         | Le caractère doit apparaître au moins une fois                     |
| ?         | Le caractère doit apparaître 0 ou une fois                         |
| (numOcc)  | Le caractère doit apparaître exactement numOcc fois                |
| (min,)    | Le caractère doit apparaître au moins min fois                     |
| (min,max) | Le caractère doit apparaître plus de min fois et moins de max fois |

## Les objets du navigator

- Au chargement du navigateur , instantiation automatiquement de ses objets
- Agie sur l'état du browser et sur les zones des Documents
- Les descendants d'un objet correspondent à ses propriétés
- Pas d'héritage comme dans l'O.O

41

## Structure des objets du navigator



42

## Les objets du Navigateur

- Objet window
  - De niveau hiérarchique plus élevé.
  - Toute fenêtre ou frame est associée à un objet window
- Propriétés :
  - Soit des chaînes de caractères (ex : la propriété status)
  - Soit des objets (ex : l'objet document)
- Méthodes
  - Méthodes de manipulation ( clear(), open(), close(),...)
  - Méthodes interactives (alert, confirm,...)
- Événements
  - Exemple

```
<HTML>
<HEAD>
<TITLE> Création d'une nouvelle fenêtre </TITLE>
<SCRIPT LANGUAGE="JavaScript1.2">
<!--
toto = open("", "", "height=200,width=550,status=yes");
toto.defaultStatus="Bienvenu sur ma modeste page d'accueil";
//--
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF" onload="window.status =
'Ecriture d\'un message
dans la barre de status de la fenêtre'" onUnload = "toto.close()">
</BODY></HTML>
```

43

## Méthodes Window

| Méthodes     | Description   |
|--------------|---|
| back()       | Equivalent au bouton 'page précédente' du navigateur                                |
| blur()       | Retire le focus à une fenêtre   |
| find(chaine) | Recherche une chaîne de caractères dans le document                                 |
| focus()      | Donne le focus à une fenêtre  |
| forward()    | Equivalent au bouton 'page suivante' du navigateur                                  |
| home()       | Provoque le chargement de la page d'accueil par défaut du browser                   |
| alert        | Crée une fenêtre de dialogue modale où s'inscrit un message                         |
| confirm      | Crée une fenêtre de dialogue modale qui permet à l'utilisateur de faire un choix    |
| prompt       | Crée une fenêtre de dialogue modale qui permet à l'utilisateur de saisir une valeur |
| clear        | Efface le contenu de la fenêtre   |

44

| Méthodes Window                         |  |
|---|--|
| Méthodes                                | Description  |
| open(URL,<br>nomFenetre,<br>Paramètres) | Ouvre une nouvelle fenêtre du navigateur.<br>URL : l'URL du document à présenter dans cette fenêtre.<br>nomFenetre : Nom de la fenêtre.<br>Paramètres : directories=yes/no (affiche la barre d'outils personnel), height=value (hauteur en pixels de la fenêtre), location=yes/no (affiche la barre d'adresses), enubar=yes/no (la barre du menu), resizable=yes/no (droit de modification des dimensions), scrollbars=yes/no (autorise l'affichage des barres de défilement), status=yes/no (crée la barre de status), toolbar=yes/no (affichage de la barre des icônes), width=value (largeur de la fenêtre) |
| close                                   | Ferme une fenêtre du navigateur sur le poste client  |
| Print()                                 | Impression du contenu de la fenêtre  |
| resizeBy(X,Y)                           | Modifie la position du coin inférieur droit de la fenêtre par rapport à son coin supérieur gauche de X et Y pixels   |
| scrollBy(P1,P2)                         | Décale le contenu d'une fenêtre de P1 pixels de horizontalement et P2 pixels verticalement (maFenetre.scrollBy(150,150))   |
| scrollTo(P1,P2)                         | Fixe une nouvelle origine au coin supérieur gauche de la fenêtre (maFenetre.scrollTo(150,150))   |
| stop()                                  | Arrêt du chargement du contenu d'une fenêtre (maFenetre.stop())  |

| Méthodes Window |  |
|-----------------|--|
| Méthodes        | Description  |
| onLoad          | Au chargement de la page                                     |
| onUnload        | En quittant la page  |
| onDragDrop      | En glissant un objet (ex : icône du bureau ) dans la fenêtre |
| onMove          | En déplaçant la fenêtre                                      |
| onResize        | En modifiant la dimension de la fenêtre                      |

■ Exemple

```
<HTML>
<HEAD><SCRIPT LANGUAGE="JavaScript">
<!--
var chaine;
function scrolling(chaine){
status=chaine;
defillement=chaine.substring(1,chaine.length)+chaine.charAt(0);
setTimeout('scrolling(defillement)',200);
}
scrolling('mon message qui fait un scrolling infini ..... ');
//-->
</SCRIPT></HEAD>
<BODY>
<H2><B><CENTER>Utilisation de la barre d'état pour afficher un message qui défile</CENTER></B></H2>
</BODY></HTML>
```



The screenshot shows a Microsoft Internet Explorer window with the title bar "D:\DevWeb\2008\_2009\DevWeb\3\1js\scrolling....". The status bar at the bottom displays the message "Utilisation de la barre d'état pour afficher un message qui défile". The main content area of the browser is mostly blank.

## L'objet Navigator

- Donne des informations sur le navigateur

| Méthodes                | Description  |
|-------------------------|--|
| navigator.appName       | cet attribut contient le nom du navigateur utilisé   |
| navigator.appVersion    | retourne le numéro de version du navigateur utilisé  |
| navigator.javaEnabled() | cette méthode permet de savoir si les applets Java sont autorisées ou non. Fort utile pour prévenir l'utilisateur non averti |
| navigator.language      | nom de la langue utilisée par le navigateur pour ses affichages  |
| navigator.mimeTypes     | un tableau de tous les types mimes définis au sein du navigateur   |
| navigator.platform      | contient le nom de la plate-forme sur laquelle tourne le navigateur  |
| navigator.plugins       | contient un tableau des plugins installés  |
| navigator.javaEnabled() | Indique si Les applets java sont autorisées à s'exécuter   |
| Refresh()               | Rend disponible les nouveaux pluggins fraîchement installés  |

## L'objet History

- Permet de contrôler le déplacement dans l'historique des documents déjà visités

| Méthodes  | Description   |
|-----------|---|
| length    | Contient le nombre d'URL de documents chargés au cours de 1 session   |
| back      | Permet de revenir au document dont l'URL est située immédiatement avant (l'URLen cours) dans l'historique des URL   |
| forward() | au contraire de la précédente, cette méthode fait aller au document suivant s'il existe (ceci implique que vous ayez déjà effectué des back())                  |
| go(value) | Cette méthode permet d'accéder à un document dont l'URL se tient dans l'historique des URL, history.go(-1) permet de revenir à l'avant dernier document chargé. |

## L'objet Link

- Ces objets javascript sont associés aux adresses URL du document : qu'il s'agisse des attributs location, ou bien des liens hypertextes présent

| Méthodes | Description   |
|----------|---|
| hash     | Le nom de l'étiquette HTML ciblée par le lien             |
| host     | Partie 'nom d'hôte' de l'URL comprenant le numéro du port |
| hostname | host moins le numéro du port                              |
| href     | L'URL complète  |
| pathname | La localisation du fichier sur la machine                 |
| port     | Le numéro de port à utiliser                              |
| protocol | La définition du protocole utilisé ('http', 'ftp', ...).  |
| search   | Chaîne de caractères de spécification de recherche        |
| target   | Le nom de la fenêtre qui reçoit l'URL                     |

49

## L'objet document

- Décrit le contenu du document visualisé dans une fenêtre du navigateur

| Attributs    | Description  |
|--------------|--|
| alinkColor   | contient la couleur (RGB) utilisée pour les liens actifs                                       |
| anchors      | Ce tableau permet d'accéder aux ancrées contenues dans le document                             |
| bgColor      | Correspond à la couleur (RGB) de fond du document  |
| cookie       | Correspond à la chaîne de caractères contenant le cookie associé au document                   |
| fgColor      | Correspond à la couleur (RGB) des caractères du document                                       |
| forms        | Ce tableau permet d'accéder à tous les formulaires du document                                 |
| lastModified | la date de dernière modification du fichier contenant le document                              |
| linkColor    | Correspond à la couleur (RGB) des liens hypertexte   |
| links        | Ce tableau permet d'accéder aux liens hypertexte du document                                   |
| location     | Correspond à l'adresse (URL) complète du document  |
| referrer     | Correspond à l'adresse (URL) complète du document qui permet de consulter le document en cours |
| title        | Correspond au titre du document en cours   |
| vlinkColor   | Correspond à la couleur (RGB) des liens déjà visités   |

## L'objet document

- Décrit le contenu du document visualisé dans une fenêtre du navigateur

| Méthodes        | Description  |
|-----------------|--|
| Clear()         | Efface le contenu du document                                  |
| Close()         | Ferme le tampon du document                                    |
| Open()          | Ouvre le tampon du document                                    |
| Write(chaine)   | Permet d'écrire dans le document                               |
| Writeln(chaine) | Permet d'écrire dans le document avec un retour chariot en fin |

51

## Les objets du Navigateur

- L'objet Form : Associé aux Formulaires HTML
- Rappel objectif du formulaire :
  - L'interactivité
  - Envoi des informations au serveur
- Propriétés

| Propriétés | Description  |
|------------|--|
| action     | L'action à exécuter lors de la soumission du formulaire                            |
| elements[] | Tableau de tous les éléments du formulaire   |
| encoding   | La spécification d'encodage spécifiée par le paramètre ENCTYPE de la marque <FORM> |
| length     | Le nombre d'objets du tableau elements[]   |
| method     | Le type de traitement du formulaire(POST ou GET)                                   |
| target     | Le nom de la fenêtre (frame) cible, où l'information sera employée                 |

- Méthode
  - submit() : associée à l'événement onSubmit
- Propriété
  - onSubmit : Soumission du formulaire

52

## Les objets du Navigateur

- Objets liés aux éléments du formulaire

| Balise HTML              | Objet JavaScript | Rôle                          |
|--------------------------|------------------|-------------------------------|
| <INPUT TYPE="button">    | button           | Commande de l'utilisateur     |
| <INPUT TYPE="checkbox">  | Checkbox         | Case à cocher                 |
| <INPUT TYPE="file">      | Fileupload       | Choix d'un fichier            |
| <INPUT TYPE="hidden">    | Hidden           | Champ caché                   |
| <INPUT TYPE="password">  | Passord          | Saisie d'un password          |
| <INPUT TYPE="radio">     | Radio            | Bouton radio                  |
| <INPUT TYPE="reset">     | Reset            | Réinitialisation des éléments |
| <INPUT TYPE="submit">    | Submit           | Envoi des saisies au serveur  |
| <INPUT TYPE="text">      | Text             | Zone d'entrée ou d'affichage  |
| <SELECT>.....</SELECT>   | Select           | Zone de sélection             |
| <TEXTAREA>...</TEXTAREA> | Textarea         | Zone d'entrée ou d'affichage  |

- Utiles pour le contrôle des données utilisateur( ex : vérification de la syntaxe d'un mail, champs obligatoire,...)

53

## Les Cookies et JavaScript

- Objets liés aux éléments du formulaire
- Stocker des informations persistantes sur le poste du client, accessibles à partir d'un serveur : nombre de connexions, les rubrique les plus visitées, date de la dernière visite, adresse mail,....
  - Appel à des données placées antérieurement sur le disque du client
  - Pas d'appel aux ressources serveur (données)
  - Prendre des décisions adaptées au traitement client
    - Des informations à transférer d'une page à l'autre du site.
- Un cookie est un ensemble d'infos. C'est une chaîne de caractères contenant les informations concaténées
- Un serveur ne peut pas placer plus de 20 cookies par client
- Mémorisation allant jusqu'à 300 cookies
- La taille d'un cookie <= 4 Ko
- Seul le serveur propriétaire d'un cookie peut le mettre à jour ou le supprimer

54

# Les Cookies et JavaScript

- Directive SetCookie() de HTTP : fonction permettant de stocker des infos dans un cookie
- Arguments

| Arguments   | Description   |
|-------------|---|
| nomDuCookie | Nom du cookie. (obligatoire)  |
| valeur      | valeur à stocker est associée à un nom de cookie  |
| expires     | Date d'effacement du cookie ( facultatif )  |
| path        | Chemin correspondant aux documents qui sont autorisés à accéder au cookie. ( facultatif )<br>Si le chemin est :<br>/ : toutes les pages du serveur pourront accéder<br>repertoire : toutes les pages se trouvant dans le répertoire correspondant et ses sous répertoire peuvent accéder<br>repertoire/nomPage : seule la page spécifiée peut accéder |
| domain      | Chemin correspondant au serveur http qui est à l'origine du cookie. ( facultatif ).   |
| secure      | Champ ( facultatif ), le cookie ne sera transmis que si la connexion et sécurisée ( par ex HTTPPs ) ; il prend les valeur soit TRUE ou FALSE.   |

55

# Les Cookies et JavaScript

## ▪ Exemples

### Écriture d'un cookie

```
function CreeCookie (nom,valeur) {  
    document.cookie = nom + "=" + escape(valeur)  
}  
  
function CreeCookieDate(nom,valeur,dateExpire)  
{  
if (dateExpire == null) expiration ="";  
else expiration = " ; expires=" + dateExpire.toGMTString();  
document.cookie =  
nom + "="  
+ valeur  
+ expiration ;  
}  
  
maDate=new Date(2008,11,31);  
CreeCookieDate("login", "B.yasser",maDate)
```

### Lecture d'un cookie

```
function LitCookie(nom)  
{  
var search = new String(nom + "=" );  
//y a t-il des cookies ?  
if (document.cookie.length >0)  
{  
//le cookie existe-t-il ?  
debut = document.cookie.indexOf(search) ;  
if (debut != -1)  
{  
    // fin de la chaîne  
fin = document.cookie.indexOf(";", debut) ;  
  
if (fin == -1) fin = document.cookie.length ;  
return document.cookie.substring(debut,fin);  
}  
else return "" ;  
}  
else return "" ;  
}
```

56