Langages Applications et Internet

Sadok Ben Yahia, PhD
Faculté des Sciences de Tunis
sadok.benyahia@fst.rnu.tn

Le langage Javascript

Introduction

- Programme Java s'exécutant **côté client Web** (cad dans le navigateur)
- Créé par Netscape (1er nom : LiveScript 1.0)
- Applet: prog. "autonome" stocké dans un fichier **.class**
- JavaScript: prog. **source** embarqué dans une page **.html**
- (quasiment) même syntaxe que Java mais
  - *langage interprété (par le navigateur)*
  - *pas de classe, pas d'héritage, pas de typage, API moins riche*

	côté client	côté serveur
.class autonome	applet	servlet
embarqué dans .html	JavaScript	JSP

Exemple

- Java **embarqué** dans une page HTML entre les balises <SCRIPT> et </SCRIPT>
- le **chargement** de la page provoque l'**exécution** du code JavaScript
- le script JavaScript génère dynamiquement du **code HTML**

```
<HTML> <BODY>
<H1>Table des factorielles</H1>
<SCRIPT LANGUAGE="JavaScript">
for ( i=1 , fact=1 ; i<4 ; i++ , fact *= i )
{
document.write( i + "! = " + fact + "<BR>" );
}
</SCRIPT>
</BODY> </HTML>
```

Principe de fonctionnement

ce qu'affiche le navigateur

```
<HTML> <BODY>
<H1> Table des factorielles</ H1>
1! = 1< BR>
2! = 2< BR>
3! = 6< BR>
</ BODY> </ HTML>
```

- **attribut language**
  - *définit la version du langage de script utilisé (VBScript, JavaScript)*
- **attribut spécifique : src**
  - *pour charger du javascript dans un fichier séparé*
  - *src="nomfichierscript.js"*
  - *spécifique à Netscape -> déconseillé*
- **Position du code dans un fichier HTML**
  - *dans l'en-tête : pour définir des fonctions dès le chargement ;*
  - *entre l'en-tête et le corps ;*
  - *dans le corps : pour gérer les événements.*

Insertion de code dans une URL d'un élément HTML

- **Balise A**
  - *Pour exécuter du code JavaScript il est possible d'utiliser :*
  - *<A href="JavaScript:code">texte</A>*
  - *Quand on clique sur texte le code s'exécute (le plus souvent c'est un appel de fonction).*

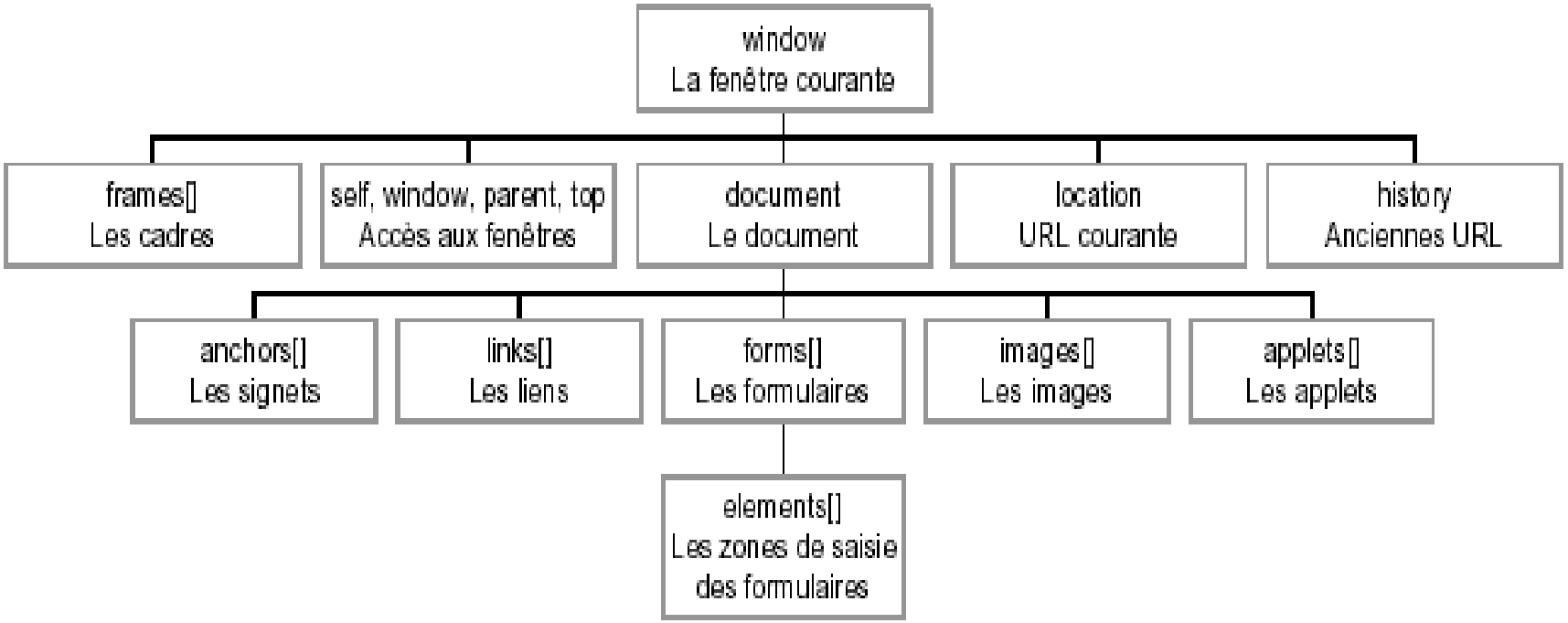
jsurl.html

```
<HTML>
<HEAD>
   <TITLE>afficher l'heure</TITLE>
</HEAD>

<SCRIPT language="JavaScript1.2">
<!--
function heure()
{
   var ch;
   var today = new Date();

   ch = "il est " + today.getHours() + " heures ";
   ch += today.getMinutes() + " minutes et ";
   ch += today.getSeconds() + " secondes";

   window.alert(ch);
}
// -->
</SCRIPT>

<BODY BGCOLOR="#FFFFCC">

<P>Bonjour, voulez-vous conna&icirc;tre l'<A href="javascript:heure()">heure</A> ?</P>

<P>Vous avez ici un <A href="javascript:void(0)">lien</A> qui ne charge rien.</P>

</BODY>
</HTML>
```

le "JavaScript:void(0)" empêche le navigateur de faire un chargement quand on clique sur le lien

Hiérarchie des éléments d'une page HTML

- hiérarchie de (pseudo)-objets JavaScript ➡ représentation du contenu HTML
- consultables / modifiables ➡ génération dynamique de contenu HTML



Hiérarchie des éléments d'une page HTML

- Exemple d'accès aux éléments d'une page HTML

<FORM ACTION="..." **NAME="formulaire"** METHOD=POST>
Nom <INPUT **NAME="nom"** SIZE=46> <P>
Prénom <INPUT **NAME="prenom"** SIZE=40> <P>
<INPUT TYPE=SUBMIT VALUE="Envoi">
<INPUT TYPE=RESET VALUE="Remise à zéro"> <P>
</FORM>

- Lecture/écriture de la variable **document.forms[0].elements[1].value**
- ≡ consultation/modification automatique du contenu du champ **prenom**
- Utilisation de façon alternative des attributs **NAME** **document.formulaire.prenom.value**

Hiérarchie des éléments d'une page HTML

- Manipulation des éléments d'une page HTML
- Nombreuses variables et fonctions disponibles sur **chaque élément**
- Quelques exemples :
  - **Form.*submit()*** *: force l'envoi des données du formulaire (≡ clic sur bouton* submit*)*
  - **Input.*focus()***: *amène le curseur de saisie dans le champ*
  - ***window.setInterval(*** **fonction*, nbMilliSecondes)*** *: exécute un traitement périodiquement*
  - ***window.alert("message")*** *: affiche un message dans une boîte de dialogue*
  - ***window.prompt("message", "valeur par défaut")*** *: affiche une boîte de dialogue permettant de saisir une valeur*

Gestion d'événements

- Interaction avec l'utilisateur via des **gestionnaires d'événements**
- événements gérables : clavier/souris
- **ajout de gestionnaires sur des éléments du document HTML**
- ⇨ ajout d'attributs (onClick="...", keyPressed="...", ...) sur des balises HTML (<INPUT TYPE=SUBMIT onClick="...">, ...)
- fonctions JavaScript (**définies par le programmeur**) associée aux gestionnaires
  - *<INPUT TYPE=SUBMIT onClick="**onAClique()**">: ⇨ appel de la fonction lorsque l'événement survient*
  - *Exécution:Si la fonction retourne*
    - *true : le **comportement par défaut est exécuté** après la fonction*
    - *false : le comportement par défaut est **ignoré***
- ⇨ permet de **redéfinir certains comportements par défaut** du navigateur

Type d'événements

- **Souris** : onClick, onDblClick, onMouseDown, onMouseUp onMouseOver, onMouseOut
  - *onMouseDown : appui sur le bouton de la souris*
  - *onMouseUp : relâchement du bouton de la souris*
  - *onMouseOver : arrivée sur une zone*
  - *onMouseOut : départ de la zone*

- **Clavier**: onKeyPressed, onKeyDown, onKeyUp
  - *onFocus : lorsque le curseur de saisie arrive dans une zone*
  - *onBlur : lorsque le curseur de saisie repart d'une zone*
- **Mixte**: onFocus, onBlur
- **Pour les formulaires**: onSubmit, onReset
- **Pour les zones de saisie des formulaires**: onChange
- **Pour les documents et les images**: onLoad, onUnload, onAbort

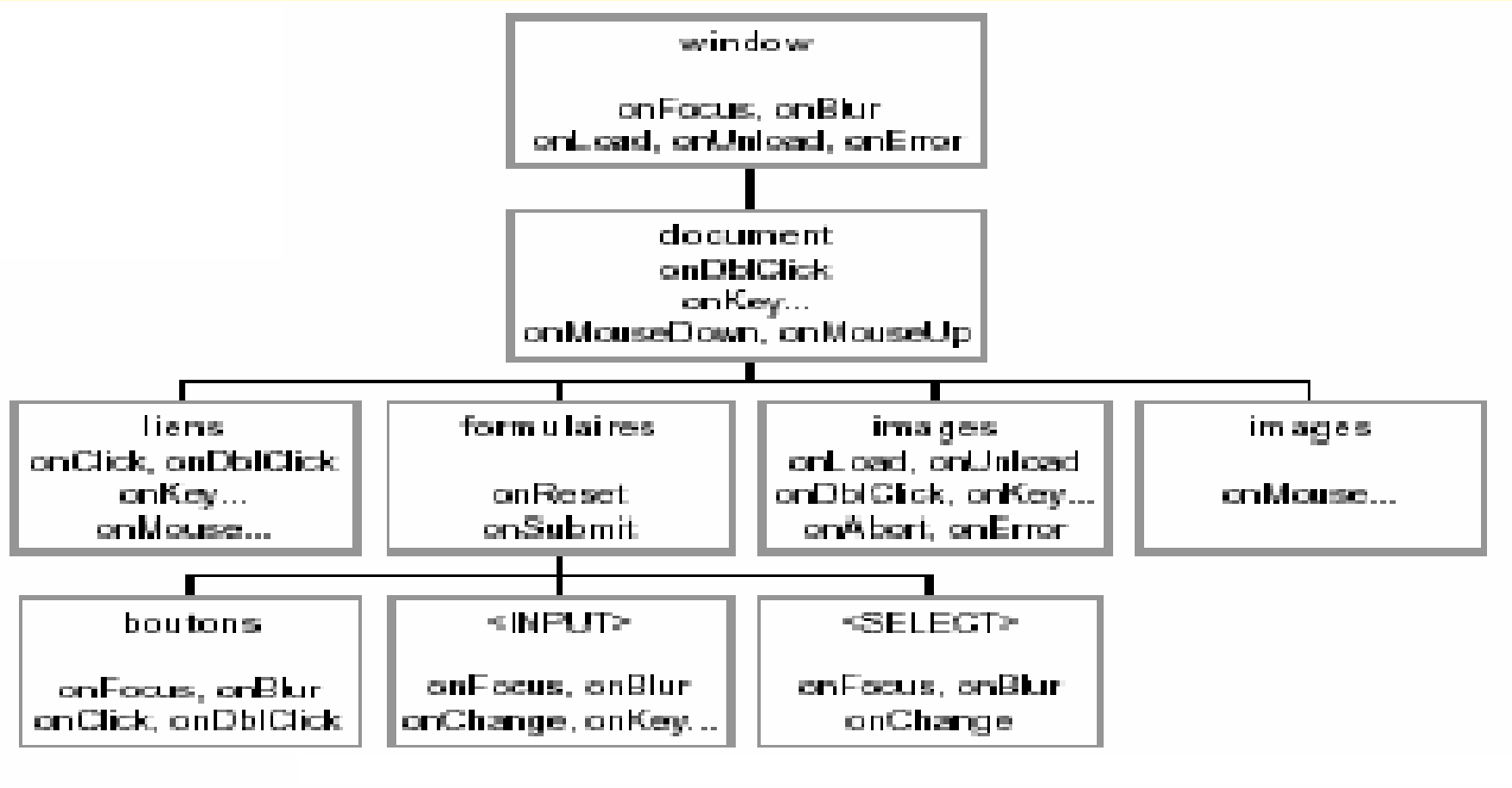
Récapitulatif

nom	événement	s'applique à
onAbort	interruption de chargement	Image
onBlur	perte du focus	champs de saisie, window
onFocus	attribution du focus	
onClick	clic sur un objet	button, checkbox, radio, reset, submit, lien retourne *false pour annuler l'action*
onChange	changement de champ de saisie	fileupload, password, select, text, textarea
onDblClick	double clic	lien, image, bouton
onError	erreur JS ou de chargement d'une image	image, window
onKeyDown	touche enfoncée	document, image, lien, text retourne *false pour annuler*
onKeyPress	l'utilisateur a appuyé sur une touche	
onKeyUp	touche relâchée	

Récapitulatif

nom	événement	s'applique à
onLoad	chargement	image, window
onMouseDown	un bouton de la souris est enfoncé	document, lien, image, button retourne *false pour annuler*
onMouseUp	le bouton de la souris est relâché	
onMouseMove	déplacement de la souris	
onMouseOut	la souris vient de quitter une zone	lien,image: retourne *true pour empêcher l'affichage de l'URL*
onMouseOver	la souris entre sur une zone	
onReset	annulation des données saisies	formulaire ; retourne *false pour annuler,* *<FORM action="fich.php3" onSubmit="return Verif()">*
onSubmit	soumission du formulaire	
onSelect	sélection d'un champ de texte	champs de saisie
onUnload	déchargement d'un document	image, window

- Type d'événements : En fonction de l'élément envisagé



La classe WINDOW de JavaScript

- Il s'agit de la classe située au sommet de la hiérarchie.
- de façon générale, comme tout se déroule dans une fenêtre du navigateur, le nom de la fenêtre est implicite, le préfixe ***window.*** peut être omis pour désigner un objet ou une méthode de la fenêtre courante (sauf dans un gestionnaire d'événement dont l'objet courant étant un document, il faut préciser la fenêtre de ce document).
- Bien entendu si la propriété ou la méthode s'adresse à une fenêtre définie par la programmeur à l'aide de la méthode ***open()***, il faut préfixer par son nom.

La classe WINDOW de JavaScript

- **Propriétés simples**
  - ***defaultStatus*** *: représente le message de défaut qui sera affiché dans la barre de statut <body onLoad="defaultStatus='Bonjour à tous'">*
  - ***status*** *est un message affiché dans la barre de statut de la fenêtre. window.status="N'oubliez pas de fermer vos fenêtres !"*
  - ***length*** *représente le nombre de cadres dans la fenêtre parente (0 sinon).*
  - ***name*** *représente le nom de la fenêtre*
  - ***opener*** *spécifie le nom de la fenêtre parent, qui l'a créée dynamiquement, avec open().*
  - ***parent*** *est le nom de la frame où se trouve éventuellement la fenêtre*
  - ***self*** *est un synonyme pour le nom de la fenêtre et fait référence à la fenêtre courante*
  - ***top*** *fait référence à la fenêtre principale du navigateur.*
  - ***window*** *est un synonyme pour la fenêtre courante*
  - ***closed*** *booléen qui indique si la fenêtre a été fermée. utilisation : if (! fen.closed) fen.close();*

Les Propriétés objets

- Ces propriétés sont en fait elle-même des objets dotées de propriétés et de méthodes
- **document** c'est le document HTML étudié
- **history** liste des documents chargés dans la fenêtre (simule l'action des boutons suivant et précédent)
  - *history.back() : document précédent*
  - *history.forward() : le suivant*
  - *history.go(n) : recharge le document situé à n étapes du présent document (n est de signe qcq)*
- **location**
  - *location.href : chaîne contenant l'url*
  - *location.hash : partie de l'url après # (lien interne)*
  - *location.host : nom serveur et port*
  - *location.protocol : http, ftp, ..*
  - *location.reload() : recharge le document courant = history.go(0)*
  - *location.replace(url)*
- **frames[]** est un tableau représentant tous les cadres dans une fenêtre Les fenêtres-cadres peuvent être nommées au moment de leur définition par un nom, sinon on peut les référencer par frames[index]

Méthodes de gestion d'objets fenêtres

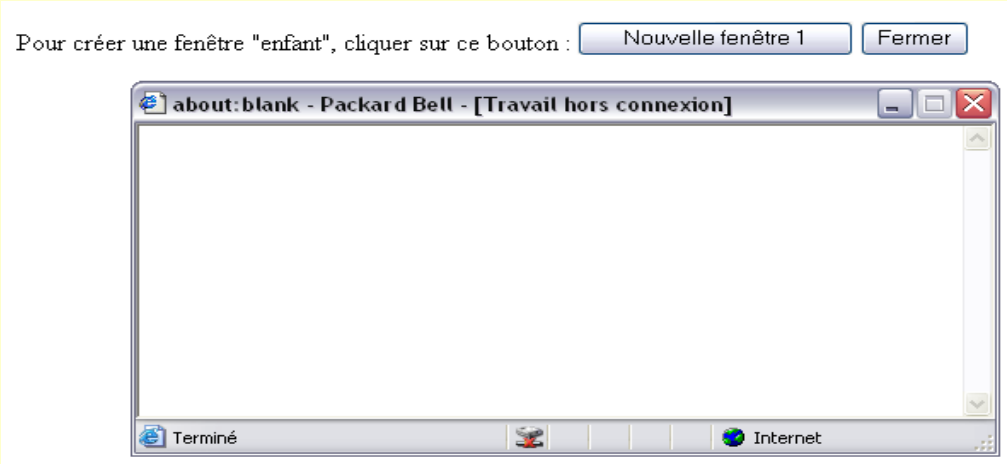
- Avec *open()* et *close()*, le programmeur dispose de moyens très souples avec les gestionnaires d'événements, d'ouvrir et de fermer des fenêtres auxiliaires.
  nomFenetre =open("URL","nom_fenetre", "options"); ouvre une nouvelle fenêtre
  - *"URL" est l'adresse du document à charger dans la nouvelle fenêtre.*
  - *"nom_fenetre" est le nom de la fenêtre, à donner à l'attribut TARGET des balises <FORM> ou <A>*
  - *"options" est une liste d'éléments optionnels qui précisent l'aspect de la fenêtre.*
- On utilise ensuite **nomFenetre** pour faire référence à un objet ou une propriété de la nouvelle fenêtre.
  Par exemple, pour écrire dans le composant message du formulaire formu situé dans le document de la fenêtre fen, créee par fen=open(...), on écrit :
  fen.document.formu.message.value= "...";

Paramètres de la partie OPTIONS

- On peut contrôler l'apparence de la nouvelle fenêtre à l'aide du 3ème paramètre de la méthode open() qui accepte une liste d'options séparées par des virgules
  - *barre d'outils* — *toolbar[=yes|no] | [=1|0]*
  - *url visible* — *location[=yes|no] | [=1|0]*
  - *répertoire ?* — *directories[=yes|no] | [=1|0]*
  - *barre d'état* — *status[=yes|no] | [=1|0]*
  - *barre de menu* — *menubar[=yes|no] | [=1|0]*
  - *barre de défilement* — *scrollbars[=yes|no] | [=1|0]*
  - *Redimensionnement* — *resizable[=yes|no] | [=1|0]*
  - *visibilité permanente ?* — *alwaysRaised[=yes|no] | [=1|0]*
  - *fermée avec la fen. Parente* — *dependant[=yes|no] | [=1|0]*
  - *Hauteur* — *height= dimension en pixels*
  - *Largeur* — *width= dimension en pixels*
  - *positionnement X* — *screenX = nb pixels*
  - *positionnement Y* — *screenY = nb pixels*

Exemple

```
<SCRIPT>
 var hauteur=200; // hauteur de la fenêtre à créer
var largeur=500; options="width="+largeur+",height="+hauteur+"toolbar=yes,
   directories=no, menubar=no,scrollbars=yes,status=yes";
function OuvrirFenetre1()
 { fen1 =open("","Nouvelle_fenetre",options); } </SCRIPT>
 <body>
<form>
 Pour créer une fenêtre "enfant", cliquer sur ce bouton :
<INPUT type =Button value="Nouvelle fenêtre 1" onClick="OuvrirFenetre1()">
<INPUT type =Button value="Fermer" onClick="fen1.close()">
</form>
</body>
```



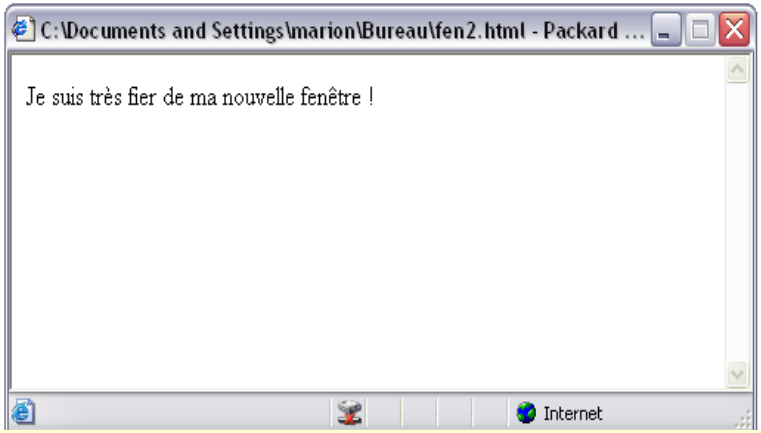
Ecriture dans la nouvelle fenêtre après sa création

```
function EcrireFenetre3(message)
 { fen3 =open("","Nouvelle_fenetre",options);
if (message !="") fen3.document.write(message+"</BR>");
else alert ("message vide !"); }
<FORM>
Ecrire un court texte dans le champ de saisie ci-dessous
<TEXTAREA NAME="zone" ROWS=8 COLS=55 VALUE="" > Je suis très
  fier de ma nouvelle fenêtre !
</TEXTAREA>
puis le faire afficher dans <TT>MaFenêtre </TT> en cliquant sur :
<INPUT type =Button name="Ecrire" value="Ecrire"
  onClick="EcrireFenetre3(this.form.zone.value)">
 </FORM>
```

Ecrire un court texte dans le champ de saisie ci-dessous

```
Je suis très fier de ma nouvelle fenêtre !
```

puis le faire afficher dans `MaFenêtre` en cliquant sur : Ecrire



Le langage: Syntaxe

- Types de base : nombre (pas de distinction entier, réel), chaîne de caractères, booléen
- Langage faiblement typé
- une variable peut à tout moment changer de type
- le type est déterminé au moment de l'affectation
- Syntaxe proche de Java : quasiment les mêmes mots-clés et opérateurs
- **mais pas** de classe, ni d'héritage, ni d'exception, ni d'interface
- Quelques pseudo-objets prédéfinis (manipulation de dates, tableaux, calculs)
- Entrées/sorties rudimentaires (clavier, écran), pas de fichiers, ni de réseau
- Langage sensible à la casse (var ≠ VAR)
- Commentaires (idem Java, C/C++) /* */ ou //

Déclaration de variables

- affectation d'une valeur à un identificateur : utilisation du mot clé **var**
  - *x = 24.5; var y;*
- Caractères autorisés : lettres, chiffres (sauf le premier caractère), et souligné _
- Nombres entiers signés, hexadécimaux (0x..), octal (0..), réels
- Booléens true ou false
- Chaînes '...' ou "..." ASCII 8 bits + \" \' \n \t \\
- Opérations sur les chaînes de caractères (+ nombreuses autres)
  - *maChaine.length : longueur de la chaîne*
  - *maChaine.charAt(*i*) :* ième *caractère de la chaîne*
  - *maChaine.indexOf(*ch,start*) : indice de* ch *à partir de* start
  - *maChaine.substring(*from,to*) : sous-chaîne de* from *(inclus) à* to *(exclus)*
  - *maChaine.split(*delim*) : tableau de sous-chaînes séparées par* delim

Récapitulatif

charAt(i)	caractère qui correspond à l'index i
charCodeAt(i)	entier correspondant au code ascii du caractère en i
+	chaine = chaine1 + chaine2
indexOf(ch,i)	index de la première occurence de ch à partir de i (optionnel)
lastIndexOf(ch,i)	index de la dernière occurence de ch à partir de i (recherche arrière)
match(exp)	applique l'expression rationnelle exp et renvoie les correspondances
replace(exp,ch)	remplace les correspondances de exp et de la chaine par ch
search(exp)	index de la première correspondance entre exp et chaine
slice(d,f)	donne la sous-chaîne qui commence en d et finit en f
split(sep)	donne les sous-chaînes quand la chaîne est découpée avec sep
substr(d,l)	sous-chaîne de longueur l commençant en d
substring(d,f)	sous-chaîne entre d et f
toLowerCase()	donne la chaîne en minuscules
toUpperCase()	donne la chaîne en majuscules
anchor(ancre)	insère la chaîne dans les éléments <A NAME="ancre">...</A>
big()	insère la chaîne dans les éléments <BIG>...</BIG>

Récapitulatif

blink()	insère la chaîne dans les éléments <BLINK>...</BLINK>
bold()	insère la chaîne dans les éléments <B>...</B>
fixed()	insère la chaîne dans les éléments <TT>...</TT>
fontcolor(couleur)	insère la chaîne dans les éléments <FONT color="couleur">...</FONT>
fontsize(taille)	insère la chaîne dans les éléments <FONT size="taille">...</FONT>
italics()	insère la chaîne dans les éléments <I>...</I>
link(URL)	insère la chaîne dans les éléments <A HREF="URL">...</A>
small()	insère la chaîne dans les éléments <SMALL>...</SMALL>
strike()	insère la chaîne dans les éléments <STRIKE>...</STRIKE>
sub()	insère la chaîne dans les éléments _{...}

Exemples

instruction	résultat
var ch = "bonjour"	Bonjour
ch.charAt(3)	J
ch.charCodeAt(3)	106
ch += " tout le monde"	bonjour tout le monde
ch.indexOf("ou")	4
ch.indexOf("ou",5)	9
ch.lastIndexOf("ou")	9
ch.lastIndexOf("ou",8)	4
ch.slice(3, 7)	Jour
tab = ch.split(' ')	Tableau
	case 0 = bonjour
	case 1 = tout
	case 2 = le
	case 3 = monde

Exemples

ch.substr(3, 7)	jour to
ch.toUpperCase()	BONJOUR TOUT LE MONDE
ch.toLowerCase()	bonjour tout le monde
ch.fontcolor("red")	<FONT color="red">bonjour tout le monde</FONT>

Tableaux

- commencent à l'indice 0
- accès aux éléments avec [] monTab[12]
- type Array
- peuvent contenir des éléments de types hétérogènes
- leur taille peut être augmentée à la demande
- Déclaration
  - *t = new Array('garage', '', 2);*
  - *≡ t = ['garage', '' , 2];*
  - *≡ t = new Array(); t[0]='garage'; t[1]=''; t[2]=2;*
  - *≡ t = new Array(3); t[0]='garage'; t[1]=''; t[2]=2;*
- Tableaux multi-dimensionnels
  - *matrice[i][j] = ...*

Tableaux

- Opérations sur les tableaux
  - *t.length : taille du tableau*
  - *t.concat(*t1,*...) : concaténation de* t, t1, *...*
  - *t.join(*sep*) : concaténation des éléments de* t *séparés par* sep
  - *t.reverse() : renverse les éléments de* t
  - *t.slice(*from,to*) : sous-tableau de* from *(inclus) à* to *(exclus)*
  - *t.sort() : tri*
- ces opérations retournent un résultat
- le tableau initial (*t*) est inchangé (ce n'est pas un objet au sens Java)

Opérateurs

- Manipulation de bits & | ^ ~ << >> >>>
- ^ XOR
- ~ NOT
- >> décalage à droite en conservant le signe
- >>> décalage à droite en ajoutant des 0
- Affectations = += -= *= /= <<= >>= >>>= &= ~= != ternaire ?:
- Logiques: && || !
- Sur les chaînes: + +=
- Sur les variables typeof *variable*
- retourne 'number', boolean', string', object', function'
- ou 'undefined' selon le **type de la variable**

Structures de contrôle

- Identiques Java, C/C++
- Conditions:

  if (*condition*) { ... } else { ... /* facultatif */ }
- switch (*expression*) {
  - *case* constante *: ... break;*
  - *...*
  - *default : ...*
  - *}*

- Boucles :
  - *for (* initialisation *;* test *;* increment *) { ... }*
  - *while (*condition*) { ... }*
  - *do { ... } while (*condition*);*
- Déroutements **break** : interruption for, case, while ou do/while
- **continue** : passage itération suivante for
- Fonction prédéfinie **eval**(*'expression JavaScript'*): évalue

  l'expression JavaScript passée en paramètre
- **with(objet par défaut pour le bloc) :** Permet d'éviter de répéter l'objet tout le long du bloc (le code devient plus lisible).

Variantes de for

Version 1

```
var fruits = new Array('pomme', 'poire', 'fraise', 'abricot');

for(i = 0 ; i < 4 ; i++)

if(fruits[i] == fruit_cherche)

window.alert("nous avons ce fruit en stock");
```

Version 2

```
var fruits = new Array('pomme', 'poire', 'fraise', 'abricot');
for (i in fruits)
if(fruits[i] == fruit_cherche)
window.alert("nous avons ce fruit en stock");
```

- Définition identique aux fonctions C
- **mais** pas de typage des arguments, ni de la valeur de retour
  - *function factorielle(n) {if (n<2) return 1; else return n*factorielle(n-1);}*
- Non déclaration exhaustive des arguments possible
- ⇨ accès par le tableau prédéfini **arguments**
  - *function plus() {*
    for (i=0,s=0 ; i<arguments.length ; i++) {
    s += arguments[i];
    }
  return s;
    }
- Panachage déclaration arg. / non déclaration possible

Pseudo-objets JavaScript

- But identique aux structures (struct) C
- ⇨ stocker des données dans un enregistrement
- Déclaration
- *objet* = { *propriete1*: *valeur1*, ... , *proprieten*: *valeurn* };
- monRectangle = { longueur: 4.5, largeur: 2 };
- Accès aux propriétés des objets *objet*.*propriete*
- Itération sur les propriétés d'un objet
- for (*variable* in *objet*) { ... }
- for (i in monRectangle) { document.write(i); }
- ⇨ affichage : longueur largeur

- **Constantes mathématiques *: Math.constante***

E	exponentielle de 1
LN10	ln(10)
LN2	ln(2)
PI	pi
SQRT1_2	racine carrée de 1/2
SQRT2	racine carrée de 2

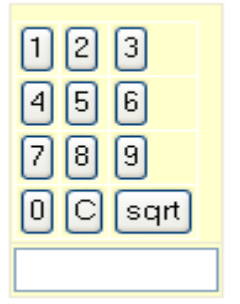
Méthodes mathématiques *: Math.fonction(paramètres)*

abs(n)	valeur absolue
acos(n)	arc cosinus (n entre -1 et 1)
asin(n)	arc sinus (n entre -1 et 1)
atan(n)	arc tangente
atan2(x,y)	angle associé aux coordonnées polaires du point x,y
ceil(n)	entier supérieur ou égal à n
cos(n)	Cosinus
exp(n)	Exponentielle
floor(n)	entier inférieur à n
log(n)	Logarithme
max(a, b)	retourne le paramètre le plus grand
min(a, b)	retourne le paramètre le plus petit

Méthodes mathématiques *: Math.fonction(paramètres)*

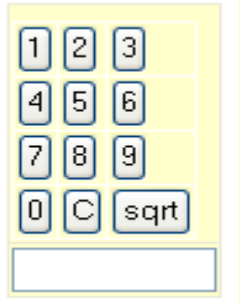
pow(x, y)	donne x puissance y
random()	nombre pseudo-aléatoire (0..1)
round(n)	entier le plus proche de n
sin(n)	sinus
sqrt(n)	racine carrée
tan(n)	tangente

Atelier: une petite Calculette



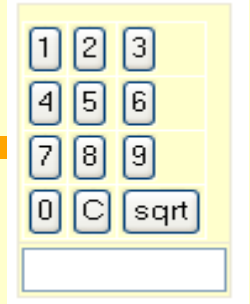

```
<BODY bgcolor="#FFFFCC">

<FORM action="" name="calc">

<TABLE border="0">    <TR>
     <TD><INPUT type="button" value="1" onClick="ValeurEntree('1')"></TD>
        <TD><INPUT type="button" value="2"
    onClick="ValeurEntree('2')"></TD>       <TD><INPUT type="button"
    value="3" onClick="ValeurEntree('3')"></TD>
 </TR>    <TR>
  <TD><INPUT type="button" value="4" onClick="ValeurEntree('4')"></TD>
     <TD><INPUT type="button" value="5" onClick="ValeurEntree('5')"></TD>
        <TD><INPUT type="button" value="6"
    onClick="ValeurEntree('6')"></TD>
</TR>    <TR>
```

Atelier: une petite Calculette...



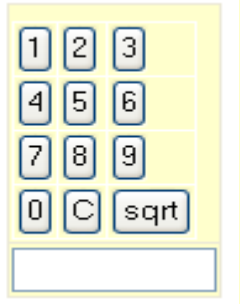

```
<TD><INPUT type="button" value="7" onClick="ValeurEntree('7')"></TD>

<TD><INPUT type="button" value="8" onClick="ValeurEntree('8')"></TD>
<TD><INPUT type="button" value="9" onClick="ValeurEntree('9')"></TD>
</TR>  <TR>
<TD><INPUT type="button" value="0" onClick="ValeurEntree('0')"></TD>
<TD><INPUT type="reset" value="C" onClick="nb.value=""></TD>
<TD><INPUT type="button" value="sqrt" onClick="Calcule()"></TD>
</TR>
</TABLE>
<INPUT type="text" size="10" maxlength="10" name="nb" value="">
```

Atelier: une petite Calculette...



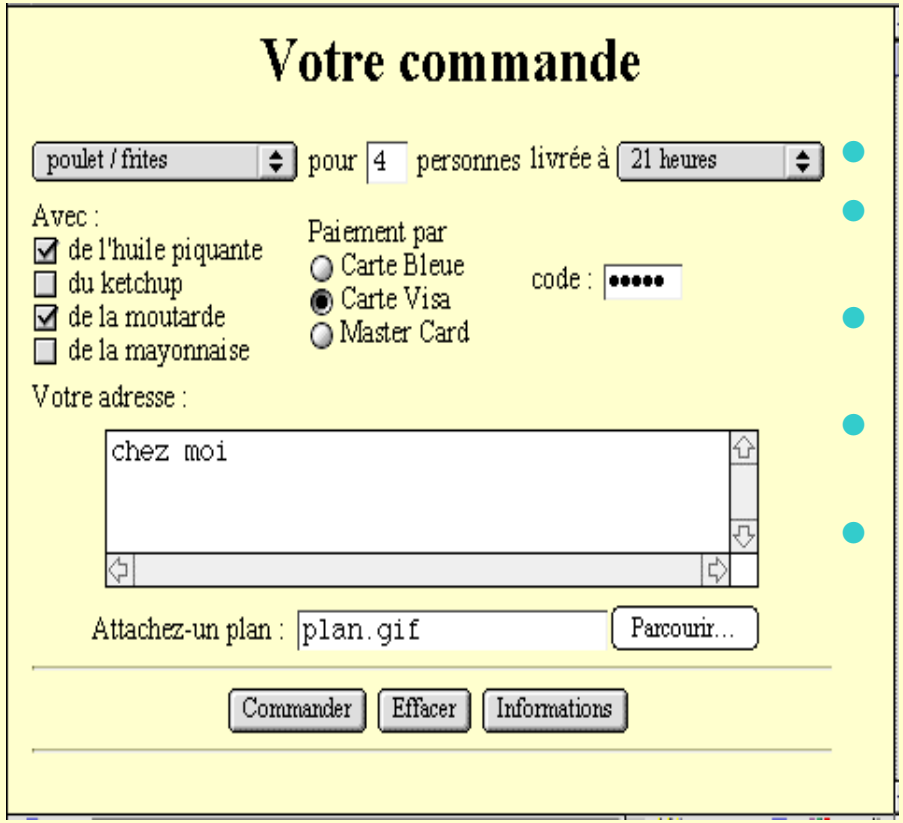

```
<HTML><HEAD>
 <TITLE>Une petite calculette </TITLE></HEAD>
<SCRIPT language="Javascript1.2">
<!—
var flagsqrt;
  function Calcule()   {
    with(window.document.calc)
        nb.value = Math.sqrt(nb.value);
     flagsqrt = 1;
   }
```

Atelier: une petite Calculette...




```
function ValeurEntree(valeur)
 {      if(flagsqrt) /* si on vient d'obtenir un resultat  */
  {          window.document.calc.nb.value="";
    flagsqrt = 0;       }
   with(window.document.calc)
 {      switch (valeur)          {
  case '0':     if(nb.value.length != 0)                nb.value += '0';                break;
          case '1': nb.value += '1'; break;          case '2': nb.value += '2'; break;
     case '3': nb.value += '3'; break;          case '4': nb.value += '4'; break;
  case '5': nb.value += '5'; break;          case '6': nb.value += '6'; break;
  case '7': nb.value += '7'; break;          case '8': nb.value += '8'; break;
  case '9': nb.value += '9'; break;          default: window.alert("pb dans switch");
       }      }   }
// -->
</SCRIPT>
```

Gestion des formulaires de données



Gestion du texte

- **boîtes de saisie de texte (INPUT type="text")**
- **saisie de textes sur plusieurs lignes (TEXTAREA)**
- **boîtes de saisie masquées (INPUT type="password")**
- **boîtes de soumission de fichier (INPUT type="file")**
- **données masquées (INPUT type="hidden")**

Lecture : **window.document.nom_formulaire.nom_element.value**

<INPUT type="text" size="2" maxlength="2" name="nombre_personnes"> personnes

ch = window.document.commande.nombre_personnes.value;

Ecriture : **window.document.nom_formulaire.nom_element.value = texte;**

window.document.commande.adresse.value = "adresse non conforme";

ch = window.document.commande.nombre_personnes.value;

Accès aux choix cochés par le client

- **Les éléments de formulaire concernés sont :**
  - ***cases à cocher (INPUT type="checkbox")***
  - ***boutons radio (INPUT type="radio")***
- ***Recherche des choix cochés***
  - ***window.document.nom_formulaire.nom_element[num_case].checked (retourne une valeur true/false)***
    window.document.nom_formulaire.nom_element[num_case].value (retourne le nom donné à la boîte)

Exemple:

```
Paiement par<BR>
<INPUT type="radio" name="paiement" value="carte_bleue">Carte Bleue<BR>
<INPUT type="radio" name="paiement" value="visa">Carte Visa<BR>
<INPUT type="radio" name="paiement" value="master_card" checked>Master Card
```

instruction JavaScript pour rechercher qui est coché (un seul)

```
with(window.document.commande){
 for(i in paiement)
   if(paiement[i].checked == true)   window.alert("radio = "+paiement[i].value);
}
```

***Avec :
***
***<INPUT type="checkbox" name="assaisoner" value="huile">de l'huile piquante
***
***<INPUT type="checkbox" name="assaisoner" value="ketchup">du ketchup
***
***<INPUT type="checkbox" name="assaisoner" value="moutarde" checked>de la moutarde
***
<INPUT type="checkbox" name="assaisoner" value="mayonnaise">de la mayonnaise

```
with(window.document.commande)
{
   var liste="";
   for(i in assaisoner)
  if(assaisoner[i].checked == true)
 liste += assaisoner[i].value + ", ";
 window.alert("checkbox = "+liste);
}
```

- ***Accès aux choix dans les listes de choix***

•*window.document.nom_formulaire.nom_element.options[num_case].**selected***

(retourne une valeur true/false)

• *window.document.nom_formulaire.nom_element.options[num_case].**value***

(retourne le nom donné au choix)

• *window.document.nom_formulaire.nom_element.**length** (retourne le nombre de choix possibles dans la liste)*

•*window.document.nom_formulaire.nom_element.options[num_case].**text***

(retourne le texte HTML associé)

- ***Accès aux choix dans les listes de choix***

```
<SELECT name="nourriture">
<OPTION value="pizza">pizza
<OPTION value="poulet_frites">poulet / frites
<OPTION value="moules_frites">moules / frites
</SELECT>
```

```
with(window.document.commande)
{    var liste = "";
for(i=0; i<nourriture.length; i++)        if(nourriture.options[i].selected ==
true)
{              liste += nourriture.options[i].value;
              liste += " (texte HTML : "+nourriture.options[i].text+"), ";        }
   window.alert("liste de choix = "+liste);}
```

- ***Accès aux textes sur les boutons***

Les éléments de formulaire concernés sont :

•boutons de soumission (INPUT type="submit")

•boutons de soumission avec image(INPUT type="image")

•boutons d'annulation (INPUT type="reset")

•boutons simples (INPUT type="button")

window.document.nom_formulaire.nom_element.value

•***Exemple***

<INPUT type="button" value="Informations" name="binfo">

texte = window.document.binfo.value;

- ***Accès par tableau: par l'intermédiaire du tableau forms[num_form]***

  - Nombre d'éléments dans le formulaire:

    window.document.forms[0].elements.length

  - type de l'élément i: ***window.document.forms[0].elements[i].type***

  - text, password, hidden, file, textarea :

    window.document.forms[0].elements[i].name (nom donné à l'élément)

    window.document.forms[0].elements[i].value (texte dans l'élément)

Gestion des formulaires de données

- ***Accès par tableau: par l'intermédiaire du tableau forms[num_form]***

  •radio, checkbox :

  window.document.forms[0].elements[i].ckecked (pour savoir si l'élément est coché)

  window.document.forms[0].elements[i].value (valeur de l'élément)

  •select unique et multiple :

  window.document.forms[0].elements[i].options[j].selected (pour savoir si une option est sélectionnée)

  window.document.forms[0].elements[i].options[j].value (valeur de l'option)

  • button, submit, reset :

  window.document.forms[0].elements[i].value (texte sur l'élément bouton);

Gestion des formulaires de données: Exemple

```
with(window.document.forms[0]) //pour éviter les répétitions

{    for (i=0; i<elements.length; i++) //pour tous les éléments

 {       type = elements[i].type; //récupérer le type de l'élément i
                 switch (type)

  {    //données textuelles


case "text" :  case "password" :  case "hidden" :   case "file" :   case
"textarea" :

     texte += "<BR>"+elements[i].name+" = "+elements[i].value;
break;          //choix par cochage

case "radio" :          case "checkbox" :

   if(elements[i].checked == true)                texte +=
"<BR>"+elements[i].name+" = "+elements[i].value;              break;
```

Gestion des formulaires de données: Exemple…

```
//choix unique ou multiple dans une liste

case "select-one" :          case "select-multiple" :

for(j=0; j<elements[i].options.length; j++)
if(elements[i].options[j].selected == true)

texte += "<BR>select = "+elements[i].options[j].value;          break;
      //boutons

case "submit" :    case "reset" :          case "image" :          case
"button" :              texte += "<BR>"+elements[i].type+" =
"+elements[i].value;              break;              default : texte +=
"<BR><B>cas "+type+" non traite</B>";

 }

   }

}
```

Événements dans un formulaire

• Lorsque le client clique dans le formulaire, modifie des champs, coche des boutons radio..., ⇨ exécuter du code JavaScript.

⇨ Vérifier la validité des données entrées (pas de lettres dans un numéro de fax)

⇨ donner des informations (affichage d'un message dans la barre d'état lorsqu'on clique dans un champ du formulaire)

• Les événements se placent dans l'élément de formulaire comme des attributs.

⇨ De la forme nom_evenement = "code JavaScript“

⇨ le code JavaScript étant la plupart du temps un appel à une fonction JavaScript définie plus haut dans la page.

ex :

<INPUT type="button" value="Informations" name="binfo" onClick="Info()">

onClick (clic sur un élément)

Éléments concernés :

- **cases à cocher (INPUT type="checkbox")**
- **boutons radio (INPUT type="radio")**
- **boutons de soumission (INPUT type="submit")**
- **boutons de soumission avec image (INPUT type="image")**
- **boutons d'annulation (INPUT type="reset")**
- **boutons simples (INPUT type="button")**

onClick (clic sur un élément)

```
<HTML><HEAD>...</HEAD> <SCRIPT language="JavaScript1.2"> <!—

function Info(){ window.alert("Nos produits sont en general non perimes");}

function Confirmer(){ return window.confirm("Voulez-vous vraiment soumettre ?");    }

function AffStatus(choix)

{ switch (choix)   {

case 1 : window.status = "avez-vous vraiment une carte bleue ?"; break;

case 2 : window.status = "avez-vous vraiment une carte visa ?"; break;

 case 3 : window.status = "avez-vous vraiment une master card ?"; break;       default : ;
  }}

// --></SCRIPT><BODY> <FORM ...>...

<INPUT type="radio" name="paiement" value="carte_bleue" onClick="AffStatus(1)"
>Carte Bleue
<INPUT type="radio" name="paiement" value="visa" onClick="AffStatus(2)"> Carte Visa
<INPUT type="radio" name="paiement" value="master_card" checked
onClick="AffStatus(3)">Master Card

<INPUT type="button" value="Informations" name="binfo" onClick="Info()">...

<INPUT type="submit" value="Commander" onClick="return Confirmer()">

</FORM></BODY></HTML>
```

onChange (modification dans élément)

Éléments concernés :

- *boîtes de saisie de texte (INPUT type="text")*
- *saisie de textes sur plusieurs lignes (TEXTAREA)*
- *boîtes de saisie masquées (INPUT type="password")*
- *boîtes de soumission de fichier (INPUT type="file")*
- *listes (SELECT)*

- *Exemple:*

```
<INPUT type="password" size="6" maxlength="6" name="pwd"
onChange="window.status=this.value">
```

onFocus, onBlur (attribution, perte du focus)

Éléments concernés :

- *boîtes de saisie de texte (INPUT type="text")*
- *saisie de textes sur plusieurs lignes (TEXTAREA)*
- *boîtes de saisie masquées (INPUT type="password")*
- *boîtes de soumission de fichier (INPUT type="file")*
- *listes (SELECT)*

Exemple:

```
<INPUT type="password" size="6" maxlength="6" name="pwd"
onFocus="window.status='Entrez votre code'">
```

onSelect (sélection dans un champ)

Éléments concernés :

- *boîtes de saisie de texte (INPUT type="text")*
- *saisie de textes sur plusieurs lignes (TEXTAREA)*
- *boîtes de saisie masquées (INPUT type="password")*
- *boîtes de soumission de fichier (INPUT type="file")*

Exemple:

```
<INPUT type="password" size="6" maxlength="6" name="pwd"
onSelect="window.status='champ password selectionne'">
```

onReset, onSubmit

Lorsque le formulaire est soumis en appuyant sur le bouton de soumission il est possible d'appeler un événement onSubmit (pour vérifier par exemple la validité des données avant d'envoyer le formulaire, pour demander la confirmation de la soumission).

L'événement reset est appelé lorsqu'on a cliqué sur le bouton annuler

Exemple:

<FORM name="commande" action="script.php3" onSubmit="return Verif()">

lorsque le client clique sur le bouton Commander, une fonction de vérification Verif() est appelée, si elle retourne la valeur true, le formulaire est envoyé au serveur, sinon la soumission est annulée.

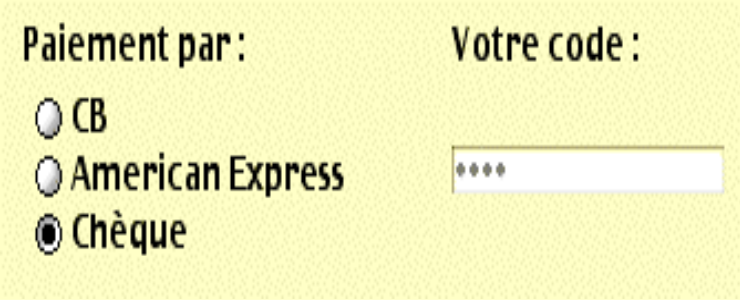
Les méthodes

nom	Rôle
blur()	enlève le focus d'un champ du formulaire
click()	simule un clic de souris sur le champ du formulaire
focus()	attribue le focus à un champ du formulaire
select()	sélectionne le contenu d'un champ du formulaire

window.document.nom_form.nom_element.focus()

Activer ou désactiver des contrôles

Il est possible de désactiver ou d'activer des contrôles d'un formulaire pour les éléments INPUT, BUTTON, TEXTAREA et SELECT en réponse à un événement.



<Form name =“formul”>

<INPUT type="radio" name="paiement" value="cb" onClick="Activer()" >Carte Bleue
<INPUT type="radio" name="paiement" value="visa" onClick="Activer()">
Carte Visa
<INPUT type="radio" name="paiement" value=“CH" checked
onClick=“Desactiver()">Chéque

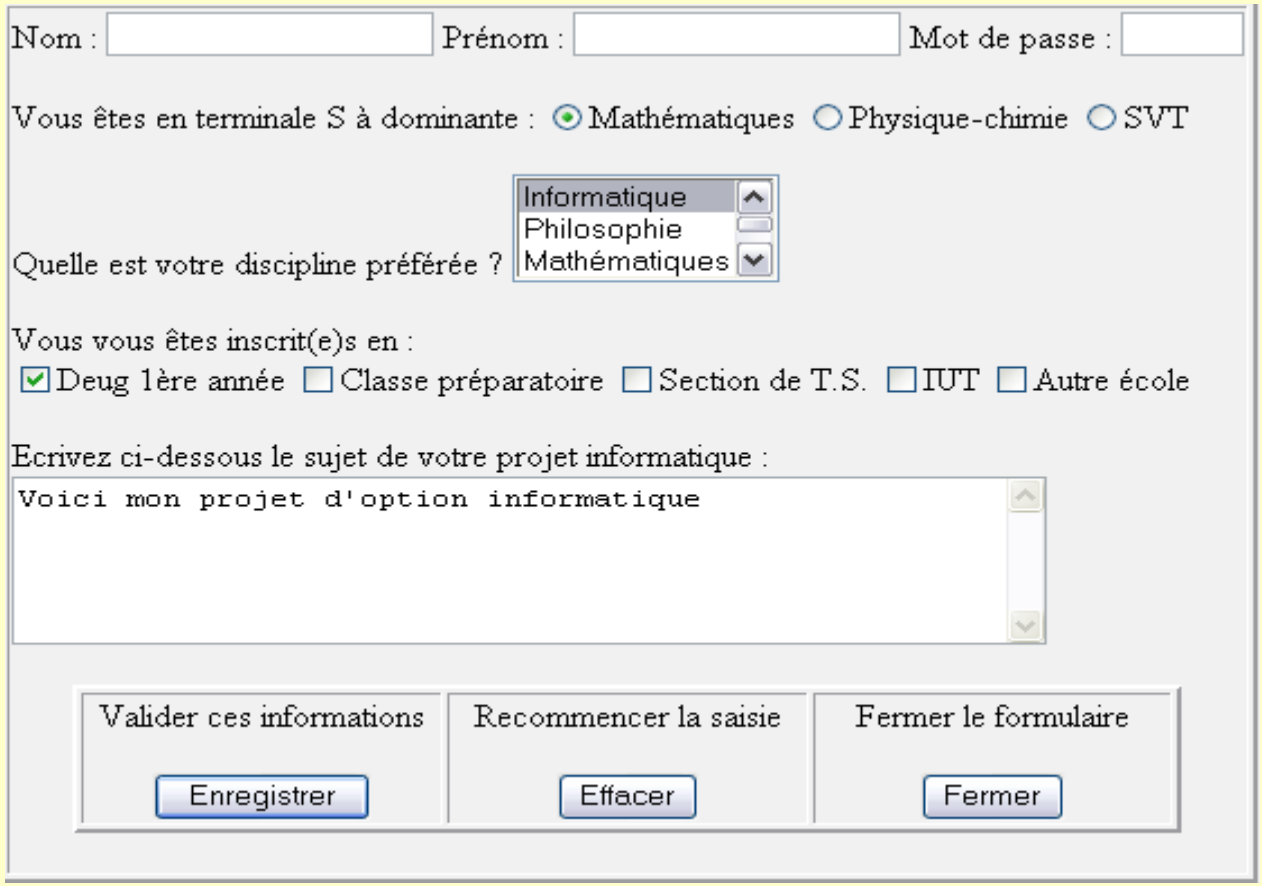
<INPUT type=“password" name=“code" disabled">

....

Activer ou désactiver des contrôles

```
<SCRIPT language="JavaScript1.2">
<!—
function activer()
{ window.document.formul.code.disabled=« false »;
}
function desactiver()
{ window.document.formul.code.disabled=« true »;
}
// -->
```

Atelier 1: Recap



```
<BODY BGCOLOR="#c0c0c0" TEXT="#000000" LINK="#0000A0" VLINK="#009000"
   ALINK="#FF00FF">
<H1>Fiche d'inscription</H1>
<FORM NAME="Fiche">
 Nom : <INPUT TYPE ="text" NAME="Nom" VALUE="" >
Prénom : <INPUT TYPE ="text" NAME="Prenom" VALUE="" >
 Mot de passe : <INPUT TYPE="password" NAME="secret" SIZE=5>
<P> Vous êtes en terminale S à dominante :
<INPUT TYPE="radio" NAME="specialites" VALUE="Math" checked>Mathématiques
   <INPUT TYPE="radio" NAME="specialites" VALUE="PC">Physique-chimie <INPUT
   TYPE="radio" NAME="specialites" VALUE="SVT">SVT
<P> Quelle est votre discipline préférée ?
 <SELECT NAME="disciplines" SIZE=3>
<OPTION>Physique <OPTION SELECTED> Informatique <OPTION>Philosophie
   <OPTION>Mathématiques <OPTION>Langues <OPTION>SVT <OPTION>Histoire
   <OPTION>EPS </SELECT> <P>
Vous vous êtes inscrit(e)s en :<BR> <INPUT TYPE=CHECKBOX CHECKED
   NAME="deug" value="Deug 1ère année">Deug 1ère année <INPUT
   TYPE=CHECKBOX NAME="cpge" value="Classe préparatoire">Classe préparatoire
   <INPUT TYPE=CHECKBOX NAME="bts" value="Section de T.S.">Section de T.S.
   <INPUT TYPE=CHECKBOX NAME="iut" value="IUT">IUT <INPUT
   TYPE=CHECKBOX NAME="autre" value="Autre école">Autre école <P>
```

```
Vous vous êtes inscrit(e)s en :<BR> <INPUT TYPE=CHECKBOX CHECKED
   NAME="deug" value="Deug 1ère année">Deug 1ère année <INPUT
   TYPE=CHECKBOX NAME="cpge" value="Classe préparatoire">Classe préparatoire
   <INPUT TYPE=CHECKBOX NAME="bts" value="Section de T.S.">Section de T.S.
   <INPUT TYPE=CHECKBOX NAME="iut" value="IUT">IUT
 <INPUT TYPE=CHECKBOX NAME="autre" value="Autre école">Autre école
<P>
Ecrivez ci-dessous le sujet de votre projet informatique :<BR>
<TEXTAREA NAME="Projet" ROWS=5 COLS=55>Voici mon projet d'option
   informatique</TEXTAREA><P>
```

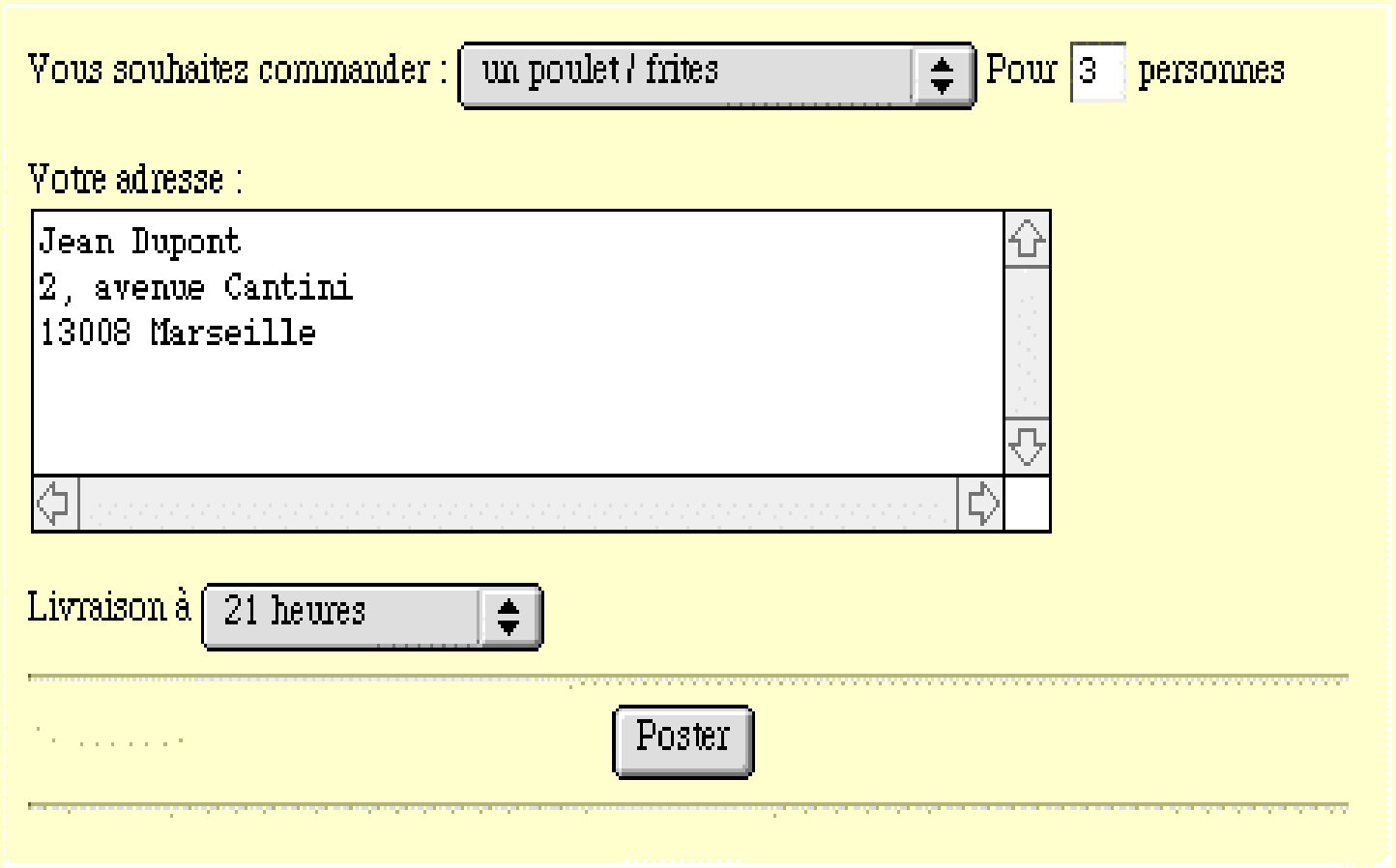
```
<CENTER>
<TABLE width=90% BORDER=2>
 <TR ALIGN=center>
   <TD width=33%>Valider ces informations<P>;
INPUT TYPE=BUTTON      VALUE="Enregistrer"
   onClick='traiter_info(Fiche)'>
 </TD>
   <TD width=33%>Recommencer la saisie<P>
<INPUT TYPE=RESET      VALUE="Effacer">
  </TD>    <TD width=33%>Fermer le formulaire<P>
<INPUT TYPE=BUTTON     VALUE="Fermer" onClick='fermer()'>
   </TD>
  </TR>
</TABLE>
</CENTER>
</FORM>
```

Atelier 1: les scripts

```
<SCRIPT LANGUAGE="JavaScript1.1">
var hauteur=300; var largeur=300;
   options="width="+largeur+",height="+hauteur+"toolbar=yes,
directories=no, menubar=no,scrollbars=yes,status=yes";
function traiter_info(formulaire)
{w open("","Fiche_de_renseignements",options);
nom=formulaire.Nom.value;
prenom=formulaire.Prenom.value;
motpass=formulaire.secret.value;
discipline=formulaire.disciplines.options[formulaire.disciplines.selectedIndex].text;
for (i=0;i<formulaire.specialites.length;i++)
{  if (formulaire.specialites[i].checked)
   specialite=formulaire.specialites[i].value;}w.document.write("<font
   size=+1><u><center>Fiche de renseignements</center></u></font><p>");
```

```
w.document.write("Votre nom : ", nom,"<br>");
w.document.write("Votre prénom : ", prenom,"<br>");
w.document.write("Votre mot de passe : ", motpass,"<P>");
w.document.write("Votre spécialité de TS : ", specialite,"<br>");
w.document.write("Votre discipline  favorite : ",discipline,"<P>");
w.document.write("Vous êtes inscrit(e) en : <UL>");
if  (formulaire.deug.checked)
   w.document.write("<LI>"+formulaire.deug.value+"<br>");
if  (formulaire.cpge.checked)
   w.document.write("<LI>"+formulaire.cpge.value+"<br>");
if  (formulaire.iut.checked)  w.document.write("<LI>"+formulaire.iut.value+"<br>");
if  (formulaire.bts.checked) w.document.write("<LI>"+formulaire.bts.value+"<br>");
if  (formulaire.autre.checked)
   w.document.write("<LI>"+formulaire.autre.value+"<br>");
   w.document.write("</ul>");
w.document.write("<br>");}
function fermer(){w.close();}
</SCRIPT>
```

Atelier:vérifier et envoyer par mail la commande



Atelier:vérifier et envoyer par mail la commande

```
<BODY BGCOLOR="#FFFFCC">
<FORM name="bondecommande" action="" method="get" enctype="text/plain" onSubmit="return Poster()">
    Vous souhaitez commander :
    <SELECT name="nourriture">
        <OPTION value="pizza"> une pizza
        <OPTION value="poulet_frites"> un poulet / frites
        <OPTION value="moules_frites"> des moules / frites
    </SELECT>
    Pour <INPUT type="text" size="2" maxlength="2" name="nombre_personnes"> personnes
    <P>Votre adresse :<BR>
    <TEXTAREA name="adresse" rows="5" cols="50" wrap></TEXTAREA>
    <P>Livraison &agrave;
    <SELECT name="heure_livraison">
        <OPTION value="20"> 20 heures
        <OPTION value="21"> 21 heures
        <OPTION value="22"> 22 heures
    </SELECT>
    <HR>
    <DIV align="center">
        <INPUT TYPE="submit" value="Poster">
    </DIV>
    <HR>
</FORM>
</BODY>
```

Atelier:vérifier et envoyer par mail la commande

```
<HEAD>
    <TITLE>Formulaire postable</TITLE>
    <SCRIPT>
    function Poster(){
        var subject="";
        var i = 0;

        with(window.document.bondecommande){ // objet par defaut pour le bloc

            //verifie que le champ adresse n'est pas vide
            if(adresse.value.length == 0){
                alert("entrez une adresse");
                return false;
            }

            //verifie le champ nombre_personnes
            if(isNaN(nombre_personnes.value) || nombre_personnes.value < 1){
                alert("entrez un nombre de personnes en chiffres > 0");
                return false;
            }

            //creation du subject du mail
            for(i=0 ; i < nourriture.length; i++)
                if(nourriture.options[i].selected == true)
                        subject = nourriture.options[i].value;

            //action du formulaire
            action = 'mailto:personne_bidule@free.fr?content-type=text/html&subject=';
            action += 'commande : ' + subject ;
        }
        return true;
    }
    </SCRIPT>
</HEAD>
```