Programmation Internet

Partie II

JavaScript

Le script côté client

Ivan Madjarov , IUT-R&T, 2002-2013

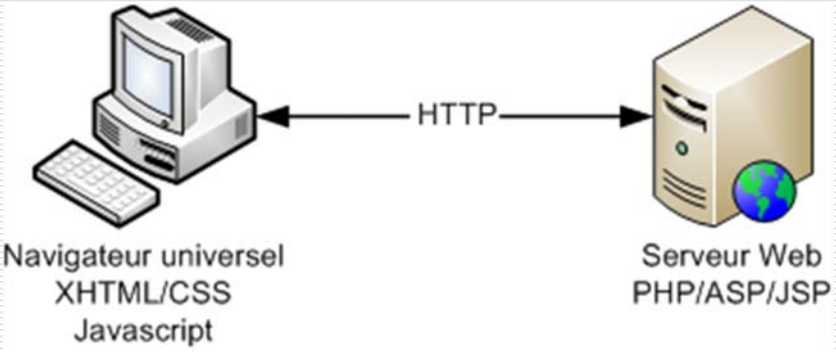
DHTML - Le HTML dynamique

- Le **DHTML** n'est pas un standard!
  - C'est une appellation qui désigne un rassemblement de techniques.
- Le **DHTML** combine:
  - **feuilles de styles** *(CSS)*;
  - langages de **scripts** *(JavaScript, VBScript, JScript)*;
  - **objets** *(DOM)*.

JavaScript - Le langage

Le modèle Client-Serveur par rapport aux scripts et les langages interprétés

1. Côté Client: *HTML/CSS/JavaScript*
2. Côté Serveur: *PHP/ASP/JSP*



JavaScript - Le langage

- JavaScript est un langage de programmation introduit dans le code HTML5 (HTML4).
- Le navigateur Web du côté client interprète le code HTML et le script JavaScript de manière native.
- La qualité d'interprétation dépend du type de navigateur (*IE, Firefox, Opera, Safari ou autre*).
- JavaScript est un langage objet et événementiel.
- Le développeur peut créer des objets interactifs avec des propriétés et des méthodes et leur associer des actions en fonction d'événements déclenchés par le client (*passage de souris, clic, saisie clavier*)

JavaScript - Intégrer

- Intégrer du code JavaScript dans une page HTML5
  - dans le corps de la page <body>
  - dans la partie entête <head>

```
<script language="JavaScript"> … </script>
```

  - dans un événement d'un objet de la page.

```
onClick="alert('Vous avez cliqué')"
```

  - faire appel à un script au clique sur un lien:

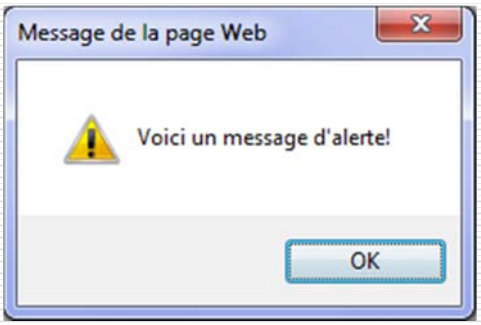
```
<A HREF="javascript:ma_fonction()">Cliquez
ici</A>
```

  - depuis un fichier externe :

```
<script language = "Javascript"
src="url/fichier.js"> </script>
```


JavaScript - Intégrer

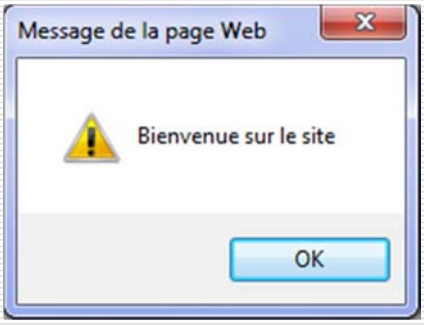

```
<!DOCTYPE html>
<HTML>
<HEAD>
<meta charset="utf-8" />
<TITLE> Voici une page contenant du JavaScript </TITLE>
</HEAD>
<BODY>
<SCRIPT language="Javascript">
<!-- on met le code en commentaire
alert("Voici un message d\'alerte!");
// -->
</SCRIPT>
</BODY>
</HTML>
```



TEST

JavaScript - Appeler

```
<HTML>
<HEAD>
<SCRIPT language="Javascript">
<!--
function Charger() {
        alert('Bienvenue sur le site');
}
//-->
</SCRIPT>
</HEAD>
<BODY onLoad="Charger();" >
<p>Texte HTML dans le corps de la page Web...</p>
</BODY>
</HTML>
```



(Test)

JavaScript - Les Variables

Instruction var pour la déclaration. Toute nouvelle variable doit être initialisée ce qui définie son type:

```
var prenom_visiteur = "Marcel";   // type string
var nom_visiteur = "Dupond";      // type string
var age_visiteur = 29;            // type numérique
```

Une variable déjà déclarée s'utilise ensuite comme d'habitude:

```
var accueil="Bonjour " + prenom_visiteur + " " +
nom_visiteur;
// Concaténation directe
```

JavaScript – La notion d'Objets

- Imaginez un arbre dans un jardin comportant une branche sur laquelle se trouve un nid.
  La hiérarchie d'objets est définie comme ceci :
- jardin
  - arbre
    - branche
      - feuille
      - nid
        - largeur: 20
        - couleur: jaune
        - hauteur: 4

JavaScript – La notion d'Objets

Notation

- Le nid sur l'arbre est donc désigné comme suit :
  `jardin.arbre.branche.nid`
- Pour changer la couleur du nid :
  `jardin.arbre.branche.nid.couleur = vert;`

Les objets du navigateur

- L'objet de la racine est l'objet fenêtre (***window***)
- Dans la fenêtre s'affiche une page (***document***)
- Cette page peut contenir plusieurs objets, comme des formulaires, des images, etc.
  `window.document.title = "titre de la page web";`

JavaScript - Les Objets

- La déclaration se fait avec var.
- Pour créer un objet, il faut utiliser le mot-clé new suivi du type d'objet.
- *Le respect des majuscules/minuscule est obligatoire*.

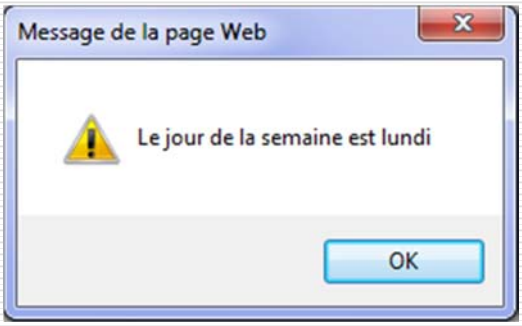
```
var datejour = new Date();
var unedate = new Date(a,  m,  j,  h,  m, s);
```

- Cela correspond à la création d'un type objet et donne accès aux méthodes et propriétés prédéfinies

```
var a = datejour.getDay();
// le jour de la semaine  (Txt)
```


JavaScript - Les Objets

```
<HTML><HEAD>
<TITLE> Voici une page contenant du code Javascript</TITLE>
</HEAD>
<BODY>
<SCRIPT language="Javascript">
<!--
var T = new Array (
"dimanche", "lundi", "mardi", "mercredi",
"jeudi", "vendredi", "samedi" );
var dj = new Date();
alert("Le jour de la semaine est "+T[dj.getDay()]);
// -->
</SCRIPT>
</BODY>
</HTML>
```



JavaScript - Les Tableaux

- Un tableau, en Javascript, est une variable pouvant contenir plusieurs données indépendantes, indexées par un numéro, appelé *indice*, qui assure l'accès aux données.

Indice	0	1	2	3
Données	donnée 1	donnée 2	donnée 3	donnée 4

- Tableaux associatifs
  - indices personnalisés pour indexer les valeurs du tableau.

Indice	"Paul"	"André"	"Marie"	"Jean"
Données	1000	500	1200	4500

JavaScript - Les Tableaux

- En JavaScript, les tableaux sont des objets:

```
var T = new Array(10); // Initialise un tableau de 10 el.
```

- Le premier élément est indexé à 0.
- Il est possible de déclarer un tableau sans dimension fixe. La taille du tableau s'adapte en fonction du contenu (taille variable).
- Pour accéder aux éléments du tableau:

```
T[0] = 10;  T[9] = 5;  T[2] = "Toto";
```

- Des propriétés associées à l'objet : *T.length*
  - Retourne le nombre d'éléments de l'objet Array.

JavaScript - Les Fonctions

- Les fonctions et leurs instructions sont déclarées et codées dans l'entête de la page (<head>) et peuvent être appelées ensuite dans le corps de la page (<body>).
  - *function Nom (argument1, argument2, ...)*
    { liste d'instructions }
  - On l'appelle par: *Nom_De_La_Fonction();*
  - Une variable déclarée dans la fonction (non précédée du mot-clé *var*) sera globale et accessible après exécution de la fonction.
  - Une variable déclarée avec le mot-clé *var* sera locale, accessible uniquement dans la fonction.

JavaScript - Les Instructions

- Le test conditionnel:

```
if (condition) { … } else { … }
```

- Les boucles:

```
For (initialisation; condition; opération) {
    // Vos instructions
}
while (condition) {
    // Les instructions de la boucle
}
```

- Un commentaire:

```
// commentaire sur une ligne
/* des commentaires sur plusieurs lignes */
```

JavaScript - Les méthodes

- Une méthode est une fonction associée à un objet
- Appel d'une méthode:
  - *window.objet1.objet2.methode();*
    window étant l'objet de base du navigateur
- Exemple:
  - une page HTML est composée d'un objet appelé *document*.
  - A l'objet *document* est associée une méthode *write(liste de paramètres)*.
    - La méthode permet de modifier de façon dynamique le contenu de la page.
  - *window.document.write("bonjour");* (*Test*, *Txt*)
  - La méthode *writeln()* ajoute un retour chariot à la fin.

JavaScript - L'Objet

- Le mot-clé this
  - Lorsqu'on fait appel à une fonction à partir d'un objet (formulaire), le mot clé *this* fait référence à l'objet en cours et se substitue à l'appel complet: *window.objet1.objet2...* ainsi lorsque l'on passe l'objet en cours en paramètre d'une fonction, il suffit de taper *nom_de_la_fonction(this)* pour pouvoir manipuler cet objet à partir de la fonction.
  - Pour manipuler les propriétés de l'objet il suffira de taper *this.propriete* (où *propriete* représente le nom de la propriété).

JavaScript – Objets prédéfinis

Les chaînes de caractères

- La déclaration

  var ch1="Bonjour";

- Les opérations:

  Concaténation:

  var chaine1="Vive le ";

  var chaine2="JavaScript";

  var chaine=chaine1+chaine2;

  La variable chaine contient après cette concaténation "Vive le JavaScript".

JavaScript - Les chaînes de caractères

o *La longueur d'une chaîne*

- Une chaîne de caractères en JavaScript est un objet string sur lequel s'appliquent des propriétés et des méthodes
- La propriété length indique le nombre de caractères de la chaîne
- var chaine="azerty"; 6 <- *chaine.length*

o *Récupérer le n^{ième} caractère*

- La méthode *charAt(n)* récupère le caractère *n* : var chaine="azerty";
  "z" <- chaine.charAt(1)

JavaScript - Les chaînes de caractères

- *MAJUSCULES / minuscules*

```
var chaine="Ceci est un texte";
var maj=chaine.toUpperCase();
var min=chaine.toLowerCase();
```

- *Une sous-chaîne dans une chaîne*

```
var domaine = "www.javascript.com";
var extension = domaine.substring( domaine.lastIndexOf(".") );
```

  - *substring* extrait une sous-chaîne à partir de l'indice retournée par *lastIndexOf* qui retrouve la dernière occurrence de sous-chaine

JavaScript – L'objet String

Méthode	description
Chaine.anchor("nom_a_donner");	Transforme le texte *Chaine* en ancrage HTML.
Chaine.big()	Augmente la taille de la police.
Chaine.blink()	Transforme la chaîne en texte clignotant.
Chaine.bold()	Met le texte en **gras** (balise <B>).
Chaine.charAt(position**)**	Retourne le caractère situé à la position donnée en paramètre
Chaine.charCodeAt(position**)**	Renvoie le code *Unicode* du caractère situé à la position donnée en paramètre
concat(chaîne1, chaîne2[, ...]**)**	Permet de concaténer les chaînes passées en paramètre, c'est-à-dire de les joindre bout à bout.
Chaine.fixed()	Transforme la Chaine en caractères de `police fixe` (balise <TT>)
Chaine.fontcolor(couleur**)**	Modifie la couleur du texte (admet comme argument la couleur en hexadécimal ou en valeur littérale)
Chaine.fontsize(Size**)**	Modifie la taille de la police, en afectant la valeur passée en paramètre
Chaine.fromCharCode(code1[, code2, ..]**)**	Renvoie une chaîne de caractères composée de caractères correspondant au(x) code(s) *Unicode* donné(s) en paramètre.
Chaine.indexOf(sous-chaîne, position**)**	Retourne la position d'une sous-chaîne (lettre ou groupe de lettres) dans une chaîne de caractère, en effectuant la recherche de gauche à droite, à partir de la position spécifiée en paramètre.
Chaine.italics()	Transforme le texte en *italique* (balise <I>)

JavaScript – L'objet String

Chaine.lastIndexOf(sous-chaîne, position**)**	La méthode est similaire à *indexOf()*, à la différence que la recherche se fait de droite à gauche : Retourne la position d'une sous-chaîne (lettre ou groupe de lettres) dans une chaîne de caractère, en effectuant la recherche de droite à gauche, à partir de la position spécifiée en paramètre.
Chaine.link(URL**)**	Transforme le texte en hypertexte (balise <A href>)
Chaine.small()	Diminue la taille de la police
Chaine.strike()	Transforme le texte en ~~texte barré~~ (balise <strike>)
Chaine.sub()	Transforme le texte en $_{\text{indice}}$ (balise <sub>)
Chaine.substr(position1, longueur**)**	La méthode retourne une sous-chaîne commençant à l'index dont la position est donnée en argument et de la longueur donnée en paramètre.
Chaine.substring(position1, position2**)**	La méthode retourne la sous-chaîne (lettre ou groupe de lettres) comprise entre les positions 1 et 2 données en paramètre.
Chaine.sup()	Transforme le texte en $^{\text{exposant}}$ (balise <sup>).
Chaine.toLowerCase()	Convertit tous les caractères d'une chaîne en minuscule.
Chaine.toSource()	Renvoie le code source de création de l'objet.
Chaine.toUpperCase()	Convertit tous les caractères d'une chaîne en majuscule.
Chaine.valueOf()	Renvoie la valeur de l'objet String.

JavaScript - Conversions

- Les fonctions de conversion
  - transformer une chaîne en un entier ou un réel:

```
var chaîne = "3.14";
var entier = parseInt(chaîne);
var reel = parseFloat(chaîne);
```

- Est-ce un nombre
  - Pour détecter qu'une *chaîne* a le format d'un nombre, on applique la fonction *isNaN(valeur)* qui renvoie :
    - *true* si valeur n'est pas un nombre
    - *false* si valeur est un nombre

JavaScript – L'objet Math

- Utilisation:

x = Math.propriété; x = Math.méthode(paramètre);

Méthode/Param	Valeur	Méthode/param	
E	constante d'Euler	sqrt()	Racine carrée
LN2	logarithme naturel de 2	exp()	valeur exponentielle
LN10	logarithme naturel de 10	max()	le plus grand de deux chiffres
PI	constante PI	min()	le plus petit de deux chiffres
abs()	valeur positive	pow()	puissance exposant
cos()	cosinus	random()	0 ou 1 aléatoire
sin()	sinus	round()	arrondi d'un nombre

JavaScript et les Maths

La plupart des fonctions de base mathématiques sont des méthodes de l'objet Math:

Math.abs(a)	Retourne la valeur absolue de a
Math.round(a)	Retourne l'entier arrondi le plus proche de a *Pour avoir un arrondi deux chiffres après la virgule, il faut utiliser Math.round(variable*100)/100;*
Math.ceil(a)	Retourne l'entier immédiatement supérieur (ou égal) à a
Math.floor(a)	Retourne l'entier immédiatement inférieur (ou égal) à a
Math.sqrt(a)	Retourne la racine carrée de a
Math.log(a)	Retourne le logarithme de a
Math.ln(a)	Retourne le logarithme népérien de a
Math.exp(a)	Retourne l'exponentielle de a
Math.pow(a,b)	Retourne a à la puissance b
Math.min(a,b)	Retourne le plus petit des paramètres a ou b
Math.max(a,b)	Retourne le plus grand des paramètres a ou b

Un tableau HTML en JavaScript

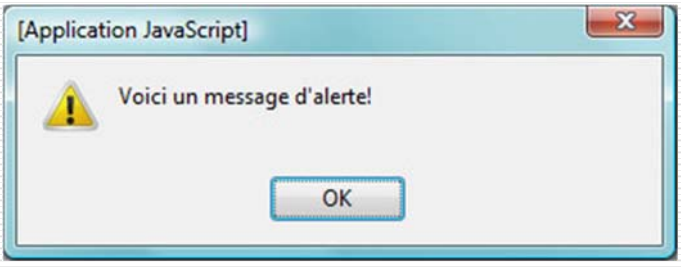
```
<SCRIPT language=javascript>
   document.write("<TABLE border='1'><TR>");
   for (var i=0; i<5; i++) {
       document.write("<TD>"+ 10*(Math.random())+"</TD>");
   }
   document.write("</TR></TABLE>");
</SCRIPT>
```

(Test)

- Ce script commence par initialiser le tableau.
- La boucle de 5 itérations crée 5 cellules de tableau et affiche à l'intérieur un nombre aléatoire.
- Le script clôt le tableau en fermant la balise </TABLE>

JavaScript - les boîtes de dialogue

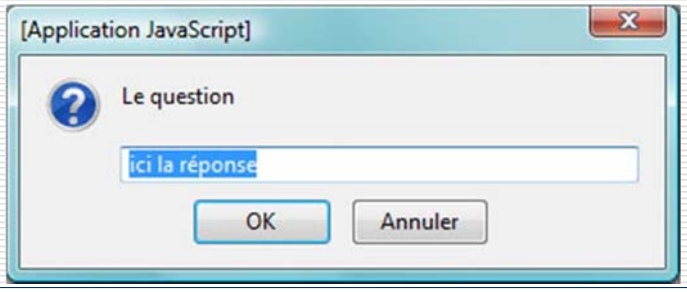
- La méthode *alert()* permet d'afficher dans une boîte composée d'une *fenêtre* et d'un *bouton OK un* texte fournit en paramètre. Exemple.



- La méthode *confirm()* est similaire à la méthode alert(), si ce n'est qu'elle permet un choix entre "OK" et "Annuler". Lorsque l'utilisateur appuie sur "OK" la méthode renvoie la valeur *true*. Elle renvoie *false* dans le cas contraire... Exemple.

JavaScript - les boîtes de dialogue

- La méthode *prompt()*
- La méthode prompt fournit un moyen simple de récupérer une information provenant de l'utilisateur.
- La méthode *prompt()* requiert deux arguments :
  - le texte d'invite
  - Le texte par défaut dans le champ de saisie



JavaScript - Formulaires

- Les éléments de formulaire sont des objets JavaScript. Soit le formulaire:

```
<FORM name="general“>
<INPUT type="text" name="champ1" value="Valeur">
</FORM>
```

- Accéder au formulaire:
  - Un formulaire est un élément de l'objet document

```
document.forms["general"]
document.forms[0]
document.general
```

  - forms est le tableau des formulaires de document

JavaScript - Formulaires

- Accéder à un élément:
  ```
  document.forms["general"].elements["champ1"]
  document.forms["general"].elements[0]
  document.forms["general"].champ1
  ```
  - "*elements*" est le nom du tableau de tous les éléments du formulaire.
- On peut atteindre un élément par :
  - son nom,
  - son indice,
  - son nom.
  - *Utilisez le nom des éléments, plutôt que les indices. Les noms sont indépendants du formulaire.*

JavaScript – Manipuler des formulaires

- Manipuler les propriétés d'un élément
  - Pour placer dans la zone de texte le mot "NOUVEAU", il faut juste écrire :
  - ```
    document.forms["general"].elements["champ1"].
    value="NOUVEAU"
    ```
- Appeler une méthode sur un élément
  - Pour donner le focus à un champ texte il faut appeler la méthode *focus()* sur cet élément.
  - ```
    document.forms["general"].elements["champ1"].
    focus()
    ```

Formulaires - *onClick*

Intégrer du JavaScript dans un événement

- L'événement le plus classique est le clic sur un bouton appelé onClick

```
<FORM name="changer">
<INPUT type="text" name="zonetexte" value="Valeur
initiale"><br/>
<INPUT type="button" value="Changer la zone de
texte" onClick = '
document.forms["changer"].elements["zonetexte"].
value="NOUVEAU"'>
</FORM>
```

[Exécuter]

Formulaires - *alert*

- Au clic d'un bouton un événement est appelé de type *onClick* qui peut faire appel à une fonction ou directement à un code JavaScript.
- Le code JavaScript doit se mettre entre " ou entre '. Il faut donc faire très attention de les alterner correctement!

```
onClick='alert("Bonjour")'
```

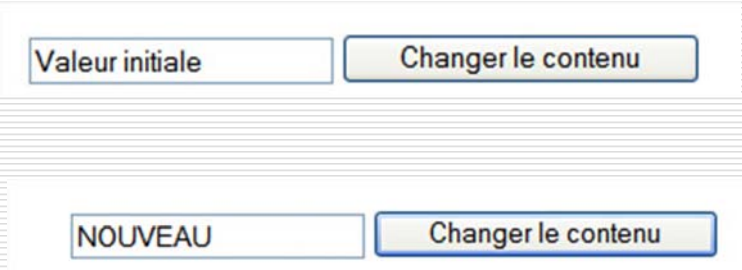
ou

```
onClick="alert('Bonjour')"
```

Formulaires - *this*

Le mot clé this représente l'objet JavaScript en cours.

```
<FORM>
<INPUT type="text" name="zonedetexte"
value="Valeur initiale">
<INPUT type="button" value="Changer le contenu"
onClick='this.form.zonedetexte.value="nouveau"'>
</FORM>
```



JavaScript - méthodes d'accès

- Méthode : *window.document.getElementById()*
  - Retourne un objet HTML à partir de son *id*
- On peut ainsi modifier les *input* dans un formulaire:

```
<html><head><script>
function f() {
   var obj = document.getElementById("champ_input");
   alert('le champ a pour valeur: "'+obj.value+'"')
   obj.value="autre valeur";
   alert('maintenant il contient: "'+obj.value+'"');
}
</script>
</head><body>
<form>
<input type="text" id="champ_input"><br>
<input type="button" onclick="f()" value="modifier">
</form>
</body></html>
```

[Tester]

JavaScript - méthodes d'accès

- Méthode : *window.document.getElementsByName()*
  - Retourne un objet HTML à partir de son nom:

```
<FORM name="form_fruit">
<INPUT type="checkbox" name="fruit" value="Fraise"> Fraise
<BR>
<INPUT type="checkbox" name="fruit" value="Banane"> Banane
<BR>
<INPUT type="checkbox" name="fruit" value="Pomme"> Pomme
<BR>
</FORM>
<SCRIPT language="javascript">
   document.getElementsByName("fruit")[0].checked = true;
   document.getElementsByName("fruit")[1].checked = true;
   document.getElementsByName("fruit")[2].checked = true;
</script>
```

  - Les cases sont repérées et cochées par programmation:

☑ Fraise
☑ Banane
☑ Pomme

[Tester]

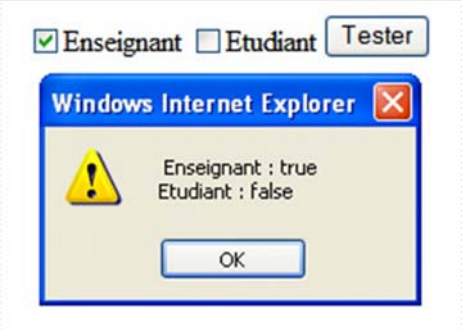
Formulaires - *input*

- Les zones de texte <input>
  - La principale action sur une zone de texte est de manipuler son contenu.
  - Un formulaire "monform" possède un champ texte "monchamp".
  - On accède au contenu du champ par :
  - document.forms["monform"].elements["monchamp"].value
  - Il faut ajouter la propriété .value pour accéder au contenu du champ!
  - Alors il faut penser aux opérations sur les chaînes de caractères.

Formulaires - Les cases à cocher

Détecter une case cochée. On utilise la propriété *checked* qui est de type booléen (true / false) pour vrai ou faux.

```
<FORM>
   <INPUT type="checkbox" name="majeur">Enseignant
   <INPUT type="checkbox" name="etudiant">Etudiant
   <INPUT type="button" value="Tester"
onClick="alert(' Enseignant : '
+this.form.majeur.checked+
'\nEtudiant : '
+this.form.etudiant.checked);">
</FORM>
```



Formulaires - Les radio-boutons

La gestion des radio-boutons est assez complexe.

```
<FORM>
   <INPUT type="radio" name="os" value="Windows 95"
checked>Windows 95
   <INPUT type="radio" name="os" value="Windows 98">Windows 98
   <INPUT type="radio" name="os" value="Windows NT">Windows NT
   <INPUT type="radio" name="os" value="Linux">Linux
   <INPUT type="radio" name="os" value="Autre">Autre
   <INPUT type="button" value="Tester"
onClick="testerRadio(this.form.os)">
<SCRIPT language="javascript">
     function testerRadio(radio) {
        for (var i=0; i<radio.length; i++)  {
          if (radio[i].checked)  {  alert("Système =
"+radio[i].value)  }     }    }
</SCRIPT>
```

Un groupe de radio-boutons liés est créé sous le nom de 'os'. La fonction a comme paramètre *le groupe des radio-boutons.* On repère en boucle la propriété checked à true et on affiche la valeur correspondante.

Formulaires - Les radio-boutons

Formulaires - SELECT

La structure d'un élément de type SELECT (voir Tableau)

Pour récupérer l'indice la ligne sélectionnée :
this.form.elements['liste'].selectedIndex
Pour récupérer le nombre de lignes :
this.form.elements['liste'].options.length
Pour récupérer la valeur de la ligne sélectionnée:
this.form.elements['liste'].options[this.form.elements['liste'].selectedIndex].value

Nom	Nom de la liste
selectedIndex	Indice de la ligne sélectionnée (ligne 1 : indice=0)
options	Tableau des lignes
length	Nombre de lignes
value	Valeur d'une ligne
text	Libellé d'une ligne

Formulaires - TEXTAREA

Une zone de texte multi-lignes a comme propriété principale .value qui contient le texte de la zone. Pour récupérer le contenu:

```
document.forms["nom"].elements["zone"].value
```


JavaScript - Changer un contenu

```
<!DOCTYPE html>
<html>
<head>
<title>Changer un contenu</title>
<script>
  function uneFunction() {
    x = document.getElementById("demo");     // Trouver l'élément
    x.innerHTML = "Bonjour de JavaScript!"; // Changer le contenu
}  // innerHTML défini le contenu
</script>
</head>
<body>
<h1>JavaScript</h1>
<p id="demo">
JavaScript peux changer le contenu d'un paragraphe HTML.
</p>
<button type="button" onclick="uneFunction()">Clic</button>
</body>
</html>
```

[Test]

JavaScript - Définir un objet

```
<!DOCTYPE html>
<html>
<body>
<script>
  var person=new Object();
  person.firstname="John";
  person.lastname="Despere";
  person.age=45;
  person.eyecolor="blue";
  document.write(person.firstname + " is "
            + person.age + " years old.");
</script>
</body>
</html>
```

[Test]

John is 45 years old.

JavaScript - Les événements

- Les évènements sont des actions de l'utilisateur, qui donnent lieu à une interactivité.
- Ainsi, il est possible d'associer des fonctions, des méthodes à des événements:
  - le passage de la souris au-dessus d'une zone, le changement d'une valeur, …
- Les gestionnaires d'événements permettent d'associer une action à un événement.
  - *onEvenement="Action_Javascript_ou_Fonction() ";*
- Les gestionnaires d'événements sont associés à des objets …

JavaScript - Liste des événements (1)

Evénement	Se produit...	S'applique à :
onabort	lorsque l'utilisateur a stoppé le chargement de l'image.	Objet javascript : image Balises HTML : img
onblur	lors de la perte du Focus.	Objet javascript : form, window, frame Balises HTML : input, textarea, select, body, frame, frameset
onchange	quand on change le contenu.	Objet javascript : form Balises HTML : input, textarea,select
onclick	quand on clique.	Objet javascript : link,document, form Balises HTML : a, body,area
ondlclick	quand on fait un double clique.	Objet javascript : link, document,area Balises HTML : a, body, area
ondragdrop	quand on déplace un objet dans une fenêtre.	Objet javascript : window, frame Balises HTML : body, frame, frameset
onerror	lorsqu'il se produit une erreur de script.	Objet javascript : image, window, frame Balises HTML : img, body, frame, frameset
onfocus	quand un élément prend le focus.	Objet javascript : window, frame, form Balises HTML : body, frameset, frame, input
onkeydown	quand une touche du clavier est enfoncée.	Objet javascript : link, image, document, form Balises HTML : a, img, body, textarea
onkeypress	quand on appuie sur une touche.	Objet javascript : link, image, document, form Balises HTML : a, img, body, textarea
onkeyup	quand on lâche une touche du clavier.	Objet javascript : link, image, document, form Balises HTML : a, img, body, textarea

JavaScript - Liste des événements (2)

onload	lors du chargement.	Objet javascript : image, window, frame img, body, frameset, frame
onmousedown	quand le bouton de la sourie est appuyé.	Objet javascript : link, document, form Balises HTML : a, body, form
onmousemove	quand le curseur bouge.	Objet javascript : link, document, form Balises HTML : a, body, form
onmouseout	quand le curseur sort de l'objet.	Objet javascript : link, area Balises HTML : a, area
onmouseover	quand le curseur passe au dessus de l'objet	Objet javascript : link, area Balises HTML : a, area
onmouseup	quand le bouton de la sourie est relâché.	Objet javascript : link, document, form Balises HTML : a, body, input
onmove	quand on déplace la fenêtre.	Objet javascript : window, frame Balises HTML : body, frameset, frame
onreset	quand on réinitialise.	Objet javascript : form Balises HTML : form
onresize	quand on redimensionne.	Objet javascript : window, frame Balises HTML : body, frameset, frame
onselect	quand on sélectionne.	Objet javascript : form Balises HTML : input, textarea
onsubmit	quand on envoie un formulaire.	Objet javascript : form Balises HTML : form
onunload	quand on ferme la fenêtre.	Objet javascript : window, frame Balises HTML : body, frameset, frame

JavaScript - Liste des événements

- Chaque événement ne peut pas être associé à n'importe quel objet... il est évident qu'un événement *OnChange* ne peut pas s'appliquer à un lien hypertexte par exemple, ...
- Objets auxquels on peut associer des événements:

Objet	Evénements associables
Lien hypertexte	onClick, onMouseOver, onMouseOut
Page du navigateur	onLoad, onUnload
Bouton, Case à cocher, Bouton radio, Bouton Reset	onClick
Liste de sélection d'un formulaire	onBlur, onChange, onFocus
Bouton Submit	onSubmit
Champ de texte et zone de texte	onBlur, onChange, onFocus, onSelect

JavaScript - Liste des événements

- Comparaison de deux champs mail:

```
<form><p><label>Adresse e-mail: </label><input type="email"
id="email_addr" name="email_addr" required><br /><br />
<label>Confirmez l'adresse e-mail: </label><input type="email"
id="email_addr_repeat" name="email_addr_repeat" required
oninput="check(this)"></p>
</form><script>function check(input) { if (input.value !=
document.getElementById('email_addr').value) {
input.setCustomValidity('Les deux e-mail ne correspondent pas.');
} else {
// le champ est valide : on réinitialise le message d'erreur
input.setCustomValidity(''); } }
</script>
```

- La méthode *setCustomValidity* prend une chaîne vide pour valider l'élément, sinon, il sera marqué invalide et la chaîne sera utilisée dans l'aide apparaissant pour l'utilis

JavaScript et les Maths

Générer un nombre aléatoire entier entre 1 et N:

```
function aleatoire(N) {
        return (Math.floor((N)*Math.random()+1));
}
```

Convertisseur Euros-Francs:

```
<FORM>
<INPUT type="text" name="franc" size=10
onBlur="convF(this.form)" value="0"> FF
<INPUT type="button" value="&lt; Convertir > ">
<INPUT type="text" name="euro" size=10"
onBlur="convE(this.form)" value="0"> Euros
</FORM>
```

JavaScript et les Maths

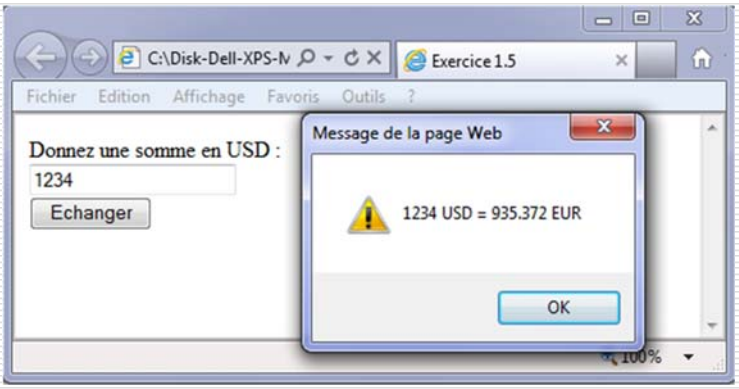

```
<SCRIPT LANGUAGE="JavaScript">
  var taux=6.55957;

  function convF(f) {
        var E=Math.round(100*parseFloat(f.franc.value)/taux)/100;
        if (isNaN(E)) { alert("Montant incorrect"); f.franc.focus();
        } else { f.euro.value=E;  }
  }

  function convE(f) {
        var F=Math.round(100*parseFloat(f.euro.value)*taux)/100;
        if (isNaN(F)) { alert("Montant incorrect"); f.euro.focus();
        } else { f.franc.value=F; }
  }
</SCRIPT>
```

Formulaires - Un exemple

```
<html><head><title>Exercice 1.5</title>
<script language="javascript">
function controle(form1) {
var inpt = document.form1.input.value;
var tsts = parseFloat(inpt);
alert(inpt+" usd = "+(tsts*0.758)+" eur");
}
</script></head>
<body>
<form name="form1">
<p>Donnez une somme en USD :<br />
<INPUT TYPE="text" NAME="input" VALUE="0"><BR />
<INPUT TYPE="button" NAME="bouton" VALUE="Echanger" onClick="controle(form1)"></p>
</FORM></BODY></HTML>
```



Formulaires - Contrôler la saisie

- Un nombre:
- Il peut être utile de vérifier que la saisie dans un champ de formulaire est bien un nombre : saisie de quantités, de prix...
- Soit nb le champ à tester. Si !(isNaN(nb)) retourne true, alors nb est un nombre.

```
<SCRIPT language="javascript">
     function CheckNombre(nb) {
   // retourne true si c'est un nombre
   // et false sinon
   return !(isNaN(nb));
     }
</SCRIPT>
```

Formulaires - Contrôler la saisie

- Valider un formulaire par JavaScript :
  - Il est souvent utile de vérifier la saisie d'un formulaire avant de le valider.
  - L'idéal est de créer un bouton (de type "*button*" et pas "*submit*") qui appelle une fonction JavaScript qui contrôle la saisie et soumet ou non le formulaire.

```
<FORM name="form4">
   <label>Adresse mail</label>
   <INPUT type="text" name="mail"> <br />
   <INPUT type="button" name="bouton" value="Valider"
onClick="ValiderMail(this.form)">
</FORM>
```


Formulaires - Contrôler la saisie

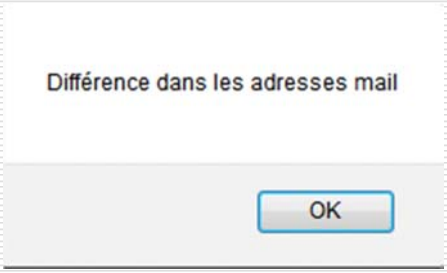
- Pour vérifier qu'un mail est valide, il suffit de tester la présence de @ et du point.

```
<SCRIPT language="javascript" method="POST">
   function ValiderMail(formulaire) {
     if (formulaire.mail.value.indexOf("@")<0 &&
        (formulaire.mail.value.indexOf(".")<0)) {
            alert("Adresse mail invalide.")
     } else {
            alert("Formulaire validé");
            // Pour valider le formulaire
            // en JavaScript
            formulaire.submit()
     }
</SCRIPT>
```

Formulaires - Contrôler la saisie

- Pour vérifier la validité de deux champs qui contiennent deux adresses mail apriori identique (comparer si deux chaines de caractères sont égales)

```
<SCRIPT language="javascript">
   function ComparerMail() {
     var M1 = "iv@iut.fr";
     var M2 = "iv@iut.net";
     if (M1 == M2) {
         alert("Equivalence des adresses mail")
     } else {
         alert("Différence dans les adresses mail");
     }
</SCRIPT>
```



Les pop-ups

- Le mot *popup* peut être traduit par *fenêtre surgissante*.
- L'événement est déclenchée par l'utilisateur (clic, ouverture de site, minuterie, ...) via un code JavaScript.
- L'objet *window* possède la méthode *open* qui attend 3 paramètres chaînes de caractères :
  window.open(page [,nom] [,options])
  - *page* contient l'adresse de la page à afficher.
  - *nom* du popup qui va être ouvert.
  - *options* paramétrage du popup.

Les pop-ups – '*page*' et '*nom*'

- Pour ouvrir un popup sur un lien, voici la syntaxe HTML :

```
<A  href = "javascript:popup('popup.html')">
Ouverture popup basique</A>
```

- Déclaration la fonction popup() :

```
<SCRIPT language="javascript">
   function popup(page) {
     window.open(page);
   }
 </SCRIPT>
```


Les pop-ups – '*options*'

La chaîne d'options d'affichage:

Propriétés	Effets	Valeurs possibles
directories	Affichage de la barre de liens	yes | no
menubar	Affichage de la barre de menu	yes | no
status	Affichage de la barre de statut	yes | no
location	Affichage de la zone d'adresse	yes | no
scrollbars	Affichage des barres de scrolling	yes | no | auto
resizable	Autorise le redimensionnement du popup	yes | no
height	Hauteur en pixels	nombre entier
width	Largeur en pixels	nombre entier
left	Position horizontale en pixels sur l'écran	nombre entier
top	Position verticale en pixels sur l'écran	nombre entier
fullscreen	Popup en plein écran (version 5 et +)	yes | no

Aucune barre de menu, taille fixe :

OuvrirPopup('popup.html', '', 'resizable=no, location=no, width=200, height=100, menubar=no, status=no, scrollbars=no, menubar=no')

Les pop-ups – '*options*'

Ouverture d'une fenêtre *popup* en fonction

```
<script LANGUAGE="JavaScript">
  <!--
  function ShowWindow(cURL) {
   var ControlWindow;
   ControlWindow = window.open(cURL, "RFC", "status,
height=480, width=640, status=no, scrollbars=yes,
resizable=yes, toolbar=0");
   if (parseInt(navigator.appVersion) >= 3)    {
     ControlWindow.focus();
   }
 }
 //-->
</script>
```

[Test]