JavaScript: Convivialité & Interactivité Dans Un Site Web Pour Commerce Électronique

2, 3 et 10 Mai 2000

Dr. Osmar R. Zaïane

University of Alberta, Canada

Objectifs du Cours

- Introduction a JavaScript;
- Comprendre l'utilité d'un langage comme JavaScript dans l'élaboration d'un site Web interactif;
- Apprendre quelques notions de programmation;
- Voir quelques exemple utiles.

Plan du Cours

I. Présentation de JavaScript	• Rapel sur quelques balises HTML; • L'architecture client-serveur et le protocol HTTP; • Qu'est ce que JavaScript en bref; • La difference HTML/**JavaScript**/Java; • JavaScript: le pourquoi et comment ça marche; • Quelques exemples.
II. JavaScript en Detail	
III. Programmation Evénementielle en JavaScript	
IV. Exemples Pratiques	

1:30

Plan du Cours

I. Présentation de JavaScript	• Les identificateurs de variables et leurs types • La notion d'objet • Les tableaux • Les structures de contrôle • Condition et selection • Repetition • Les procédures et fonctions
II. JavaScript en Detail	
III. Programmation Evénementielle en JavaScript	
IV. Exemples Pratiques	

4:00

Plan du Cours

I. Présentation de JavaScript II. JavaScript en Detail **III. Programmation Evénementielle en JavaScript** IV. Exemples Pratiques	• Qu'est ce qu'un événement? • Quels sont les événements reconnus. • Saisir et gérer les événements.

2:00

Plan du Cours

I. Présentation de JavaScript II. JavaScript en Detail III. Programmation Evénementielle en JavaScript **IV. Exemples Pratiques**	• Vérification des données dans un formulaire de saisie; • Animation des images dans une page HTML; • Utilisation des cookies pour stoker des information sur le client; • Petit site de commerce electronique avec chariot de transactions

1:30

I. Présentation de JavaScript

Objectif: Comprendre le context et avoir une idée ce qu'est JavaScript et ce que l'on peut faire avec JavaScript.

Premier jour – Durée: ≈1 heure 30

Contenu

- **Rapel sur quelques balises HTML**;
- L'architecture client-serveur et le protocole HTTP;
- Qu'est ce que JavaScript en bref;
- La difference HTML/JavaScript/Java;
- JavaScript: le pourquoi et comment ça marche;
- Quelques exemples.

Rapel sur les Balises HTML

```
<HTML>
<HEAD>
   <TITLE>Texte du titre ici</TITLE>
</HEAD>

<BODY BGCOLOR="#00FF00">

   <! Texte et balises viennent ici>
   contenu de ma page

</BODY>
</HTML>
```

Rapel sur les Balises HTML (suite)

Quelques balises utiles:

Sous-titres:
<**H1**>Sous-titre 1</**H1**>
<**H2**>Sous-titre 2</**H2**>
<**H3**>Sous-titre 3</**H3**>
<**H4**>Sous-titre 4</**H4**>
<**H5**>Sous-titre 5</**H5**>
<**H6**>Sous-titre 6</**H6**>

Ligne horizontales:
<**HR**>
<**HR SIZE=4**>
<**HR WIDTH=50%**>

Texte du titre ici

Sous-titre 1
Sous-titre 2
Sous-titre 3
Sous-titre 4
Sous-titre 5
Sous-titre 6

Rapel sur les Balises HTML (suite)

Quelques balises utiles:

Nouvelles lignes and Paragraphes:
Ligne 1 et <**BR**>Ligne 2
<**P**>Paragraphe</**P**>

Images:
<**IMG SRC**=lune.gif **WIDTH**=100 **HEIGHT**=200>
<**IMG SRC**=lune.gif **ALIGN**=right>

Texte gras et texte souligné:
<**B**>Texte Gras</**B**>
<**STRONG**>Text gras</**STRONG**>
<**U**>Texte Souligné</**U**>

Texte du titre ici

Ligne 1 et
Ligne 2

Paragraphe

Texte Gras
Text gras
Texte souligné

Rapel sur les Balises HTML (suite)

Quelques balises utiles:

Hyperliens:
<**A HREF**=*http://www.ualberta.ca*>This a link to UofA</**a**>

Taille de Fonte et couleurs:
<**FONT SIZE=4**>Ceci est de taille 4,</**FONT**>
<**FONT SIZE=5**>Parcontre ceci est de taille 5</**FONT**>

<**FONT COLOR=red**>Ceci est rouge</**FONT**>
<**FONT COLOR=white**>et ceci est blanc</**FONT**>

Texte du titre ici

This a link to UofA

Ceci est de taille 4,
Parcontre
Ceci est de taille 5

Ceci est rouge et ceci est blanc

Rapel sur les Balises HTML (suite)

Quelques balises utiles:

Listes Ordonnées:
<**OL**>
<**LI**>élément 1
<**LI**>élément 2
<**LI**>élément 3
</**OL**>

Listes non ordonnées:
<**UL**>
<**LI**>élément 1
<**LI**>élément 2
<**LI**>élément 3
</**UL**>

Texte du titre ici

1. élément 1
2. élément 2
3. élément 3

- élément 1
- élément 2
- élément 3

Rapel sur les Balises HTML (suite)

Quelques balises utiles:

Tableaux:

```
<TABLE BORDER=1 WIDTH=100%>
<TR>
   <TH>Titre1</TH><TH>Titre 2</TH>
</TR>
<TR>
   <TD>data 1</TD><TD>data 2</TD>
</TR>
<TR>
   <TD>data 3</TD><TD>data 4</TD>
</TR>
</TABLE>
```

Texte du titre ici

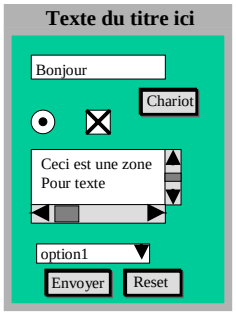
Titre 1	Titre 2
data 1	data 2
data 3	data 4

Rapel sur les Balises HTML (suite)

Quelques balises utiles:

Formulaires:

```
<FORM ACTION="url" METHOD=POST>
<INPUT name=var1 type=text value="Bonjour">
<INPUT name=var2 type=button value="Chariot">
<INPUT name=var3 type=hidden value=23>
<INPUT name=var4 type=radio>
<INPUT name=var5 type=checklist>
<TEXTAREA name=var6>Ceci est une zone
pour texte</TEXTAREA>
<SELECT name=var7>
   <OPTION> option 1</OPTION><OPTION>option 2</OPTION>
</SELECT>
<INPUT type=submit value="Envoyer"><INPUT type=reset>
</FORM>
```



I. Présentation de JavaScript

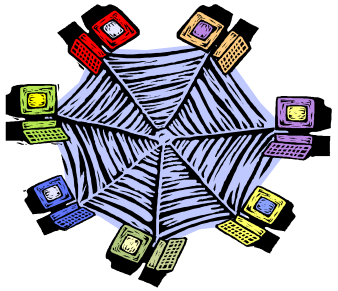
Objectif: Comprendre le context et avoir une idée ce qu'est JavaScript et ce que l'on peut faire avec JavaScript.

Premier jour – Durée: ≈1 heure 30

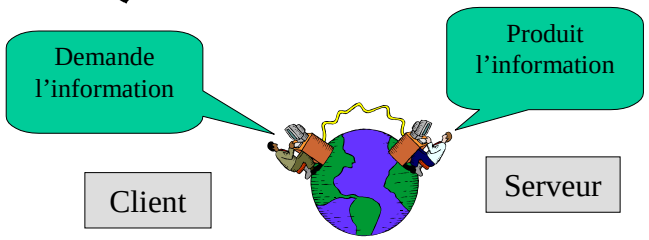
Contenu

- Rapel sur quelques balises HTML;
- **L'architecture client-serveur et le protocole HTTP;**
- Qu'est ce que JavaScript en bref;
- La difference HTML/JavaScript/Java;
- JavaScript: le pourquoi et comment ça marche;
- Quelques exemples.

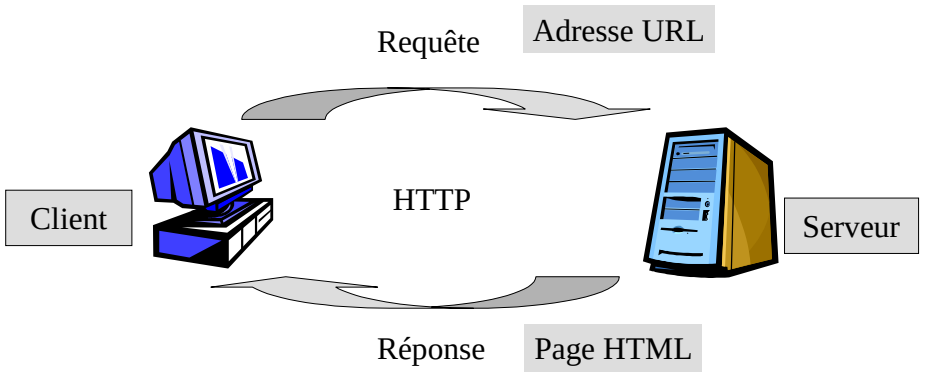
Architecture Client-Serveur



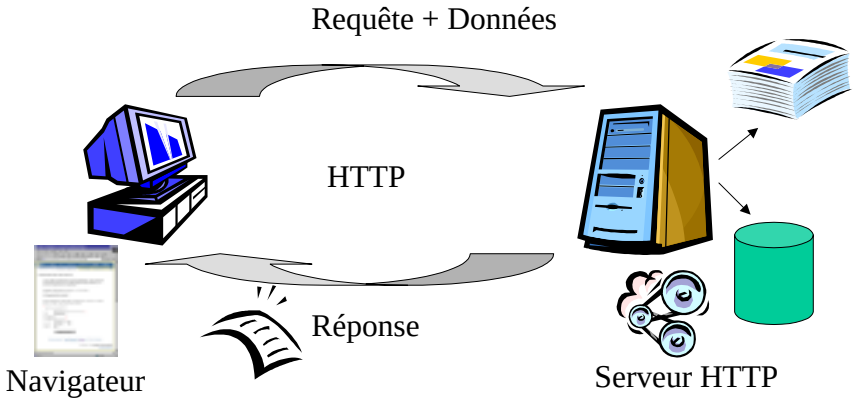
La toile mondiale (WWW) est un assortiment de machines interconnectées. Ces machines mettent des données à la disposition d'autres machines.



Architecture Client-Serveur (suite)

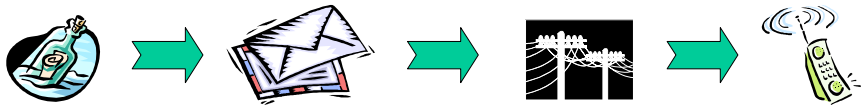


Architecture Client-Serveur (suite)

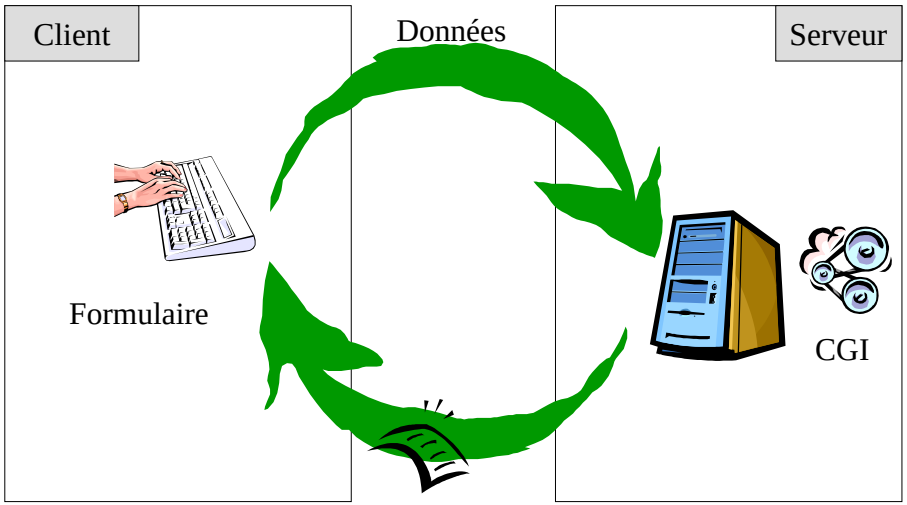


Architecture Client-Serveur (suite)

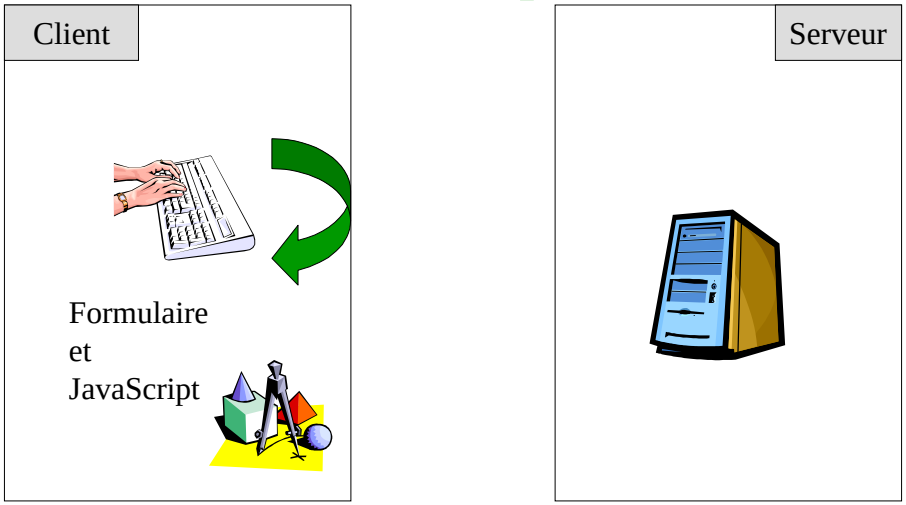
- Avec le protocole HTTP, l'interaction de l'utilisateur avec l'application se fait par une série de requêtes-réponses entre le client et le serveur.
- Le lapse de temps entre l'envoie de la requête et la réception de la réponse dépend de la disponibilité et la largeur de bande du réseau.



Validation des Données



Validation des Données avec JavaScript



I. Présentation de JavaScript



Objectif: Comprendre le context et avoir une idée ce qu'est JavaScript et ce que l'on peut faire avec JavaScript.

Premier jour – Durée: ≈1 heure 30

Contenu

- Rapel sur quelques balises HTML;
- L'architecture client-serveur et le protocole HTTP;
- **Qu'est ce que JavaScript en bref;**
- La difference HTML/JavaScript/Java;
- JavaScript: le pourquoi et comment ça marche;
- Quelques exemples.

Qu'est ce que JavaScript?

- JavaScript est un langage de commandes interprétées;
- Les commandes JavaScript (programme) sont insérées avec les balises HTML;
- JavaScript est interpété par le navigateur;
- JavaScript dans un document HTML permet d'intercepter et de traiter **localement** les événements générés par l'utilisateur;

Qu'est ce que JavaScript? (suite)

- JavaScript ne nécessite pas une connaissance particulière en programmation objet;
- JavaScript est un langage simple avec une syntaxe simple;
- JavaScript peut être utilisé à des fins simples (pour non programmeurs) ou pour des problèmes plus complexes (programmeurs zélés);
- JavaScript n'a de raison d'être en dehors d'un navigateur;

I. Présentation de JavaScript

Objectif: Comprendre le context et avoir une idée ce qu'est JavaScript et ce que l'on peut faire avec JavaScript.

Premier jour – Durée: ≈1 heure 30

Contenu

- Rapel sur quelques balises HTML;
- L'architecture client-serveur et le protocole HTTP;
- Qu'est ce que JavaScript en bref;
- **La difference HTML/JavaScript/Java;**
- JavaScript: le pourquoi et comment ça marche;
- Quelques exemples.

La Différence HTML/JavaScript/Java (HTML)

- HTML est un ensemple de balises et de règles syntaxiques pour définir la composition et l'apparence d'un document multimédia.
- Les balises permettent de contrôler l'aspet du texte, des images et insertions diverses.
- Avec les formulaires de saisie et les boutons, il est possible de simuler l'interface utilisateur d'une application sophistiquée.

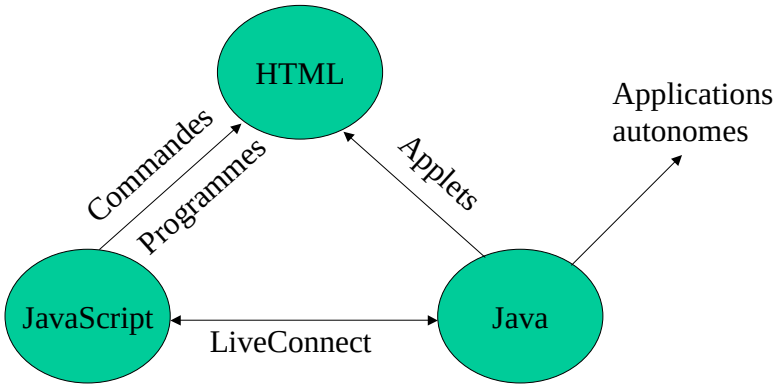
La Différence HTML/JavaScript/Java (Java)

- Java est un langage de programmation orienté-Objet.
- Java permet de concevoir de véritables applications complilées.
- Les programmes Java sont exécutées dans l'environnement du navigateur en tant qu'Applets, ou comme applications autonomes.
- Les programmes Java sont compilées sur le serveur avant d'être téléchargés et exécutés.

La Différence HTML/JavaScript/Java (JavaScript)

- JavaScriptest un langage de command interprétées.
- Le code JavaScript est transmis sans compilation préalable.
- Le but de JavaScript est d'augmenter la capacité de traitement du poste client affichant des documents HTML.
- JavaScript améliore les capacités du navigateur.

HTML - JavaScript - Java



I. Présentation de JavaScript

Objectif: Comprendre le context et avoir une idée ce qu'est JavaScript et ce que l'on peut faire avec JavaScript.

Premier jour – Durée: ≈1 heure 30

Contenu

- Rapel sur quelques balises HTML;
- L'architecture client-serveur et le protocole HTTP;
- Qu'est ce que JavaScript en bref;
- La difference HTML/JavaScript/Java;
- **JavaScript: le pourquoi et comment ça marche;**
- Quelques exemples.

Pourquoi Vouloir Utiliser JavaScript?

- Utiliser ou pas JavaScript depend de l'application et de l'interaction que l'on veut avoir avec l'utilisateur de l'application.
- Utiliser ou pas JavaScript depend des fonctionnalités offertes par JavaScript.
- JavaScript ajoute des fonctionnalités et plus d'interactivité au navigateur.
- JavaScript 1.3 peut même changer le menu de Netscape 6.0.

JavaScript: Le Pourquoi

- Analyse les données collectées avec les formulaires HTML sur la machine client.
- Valide les données saisies sans devoir accéder au serveur.
- Crée et sauvegarde des données directement sur la machine client.
- Ajoute de l'interactivité aux graphiques.
- Change des éléments de documents à la volé.
- Résous des problèmes de surchargement du serveur et de temps d'accès au serveur.
- Facilite la connection entre divers technologies comme les applets Java et les composants ActiveX.

Comment ça marche?

- Les commandes JavaScript sont insérées avec les balises HTML.
- Les commandes sont téléchargées avec le document HTML.
- Le navigateur interprète, commande par commande, le programme JavaScript.
- Le navigateur agit en conséquense.

Compatibilité

JavaScript n'est pas du Java. Mais JavaScript n'est pas du JavaScript non plus.
Il y a souvent des problèmes de compatibilité entre navigateurs et entre versions.

- JavaScript 1.0 ➔Netscape 2.0
- JavaScript 1.1 ➔ Netscape 3.x
- JavaScript 1.2 ➔ Netscape Communicator 4.0 à 4.04
- JavaScript 1.3 ➔ Netscape Communicator 4.06 à 6.0
- Microsoft Jscript ~ JavaScript 1.1

JavaScript dans HTML

JavaScript peut etre insere dans un document HTML a plusieurs niveaux.

1. Dans l'entête du document HTML (entre <**HEAD**> et </**HEAD**>)
2. Dans le corps du document HTML (entre <**BODY**> et </**BODY**>)
3. À l'intérieur d'une balise HTML

- Dans l'entête et le corps, utiliser une nouvelle balise HTML

  <SCRIPT LANGUAGE="JavaScript"> ……</SCRIPT>

- À l'intérieur d'une balise, utiliser un capteur d'évènement.
- On verra les évènements un peu plus tard .

Mon Premier Exemple (1)

```
<HTML>
<HEAD>
<TITLE>Mon premier JavaScript!</TITLE>
</HEAD>
<BODY>
<BR>
Ceci est un document HTML normal.
<BR>
  <script language="JavaScript">
    document.write("<B>Ceci est du JavaScript!</B><BR>")
  </script>
Ceci est du texte normale
</BODY>
</HTML>
```

Mon Premier Exemple (2)

```
<HTML>
<HEAD>
<TITLE>Mon premier JavaScript!</TITLE>
</HEAD>
<BODY>
<BR>
Ceci est un document HTML normal.
<BR>
  <script language="JavaScript">
    document.write("<B>Ceci est du JavaScript!</B><BR>")
    alert("Bonjour! Ceci est aussi du JavaScript.");
  </script>
Ceci est du texte normale
</BODY>
</HTML>
```


Mon Premier Exemple (3)

```
<HTML>
<HEAD>
<TITLE>Mon premier JavaScript!</TITLE>
  <script language="JavaScript">
    var monNom=prompt("Quel est ton nom?");
  </script>
</HEAD>
<BODY>
Un exemple simple:
<HR>
  <script language="JavaScript">
    document.write("<B>Bonjour" + MonNom + "!</B><BR>");
  </script>
<HR>
</BODY>
</HTML>
```


Mon Premier Exemple (4)

```
<HTML>
<HEAD>
<TITLE>Mon premier JavaScript!</TITLE>
</HEAD>
<BODY>
Un autre exemple simple. Clicker sur un lien.
<HR>
<A href="#">Ancre 1</a> -
<A href="#" onClick="confirm('Ceci est le second
hyperlien')">Ancre 2 </a> -
<A href="#" onMouseOver="alert('Ca me chatouille')">
Ancre 3</a>
<HR>
</BODY>
</HTML>
```


Comment cacher JavaScript

- Et si le navigateur n'interprète pas JavaScript?
  - Il faut cacher le code avec des commentaires

```
<HTML>
<HEAD>
<TITLE>Mon titre de page</TITLE>
  <script language="JavaScript">
  <!-- cacher le script
          MON SCRIPT VIENT ICI
  // fin de cache -->
  </script>
</HEAD>
<BODY>
etc…
```

En HTML
<! Ceci est un commentaire>
En JavaScript
// **Ceci est un commentaire**
/***Ceci est un commentaire***/*/

Où Obtenir des Programmes?

- Dois-je toujours écrire un programme JavaScript à partir de zéro?
  - On peut "voler" le code:
    - Voir le source d'un document contenant JavaScript;
    - Identifier le code nécessaire;
    - Copier le code.
  - Modifier et adapter un programme existant.
- Dois-je re-écrire le même code pour différentes pages Web?
  - Utiliser un fichier JavaScript (*.js)
  - <SCRIPT LANGUAGE="JavaScript" SRC="monScript.js">

I. Présentation de JavaScript

Objectif: Comprendre le context et avoir une idée ce qu'est JavaScript et ce que l'on peut faire avec JavaScript.

Premier jour – Durée: ≈1 heure 30

Contenu

- Rapel sur quelques balises HTML;
- L'architecture client-serveur et le protocole HTTP;
- Qu'est ce que JavaScript en bref;
- La difference HTML/JavaScript/Java;
- JavaScript: le pourquoi et comment ça marche;
- **Quelques exemples.**

Exemples de Programmes JavaScript

- Hello!
- Hello World
- Petit Quiz
- Editeur HTML
- Editeur HTML 2
- Menu avec images (rollover)
- Menu avec pointeur (rollover2)
- Bande déroulante (scroller)
- Bande déroulante 2
- Fenetre
- Multi-Moteurs de Recherche
- Convertir
- Formulaire de saisie
- Note de Frais
- Menu glissant (avec DHTML)
- Tabs (avec DHTML)
- Ordonner éléments de tableau

Résumé Partie I

- Revue générale des balises HTML
- L'architecture Client-Serveur
- JavaScript en bref
  - Langage simple de commandes interprétées
  - Commandes insérées avec balises HTML
  - Syntaxe simplifiée
  - Pas de raison d'être en dehors d'un navigateur
- Quelques spécificités et utilités de JavaScript
  - Que peut faire JavaScript
  - Comment insérer JavaScript dans un document HTML
- Exemples de ce qu'on peut faire avec JavaScript

II. JavaScript en Détail

Objectif: Apprendre comment JavaScript sauvegarde des données, comment un document est structuré vis-à-vis JavaScript, et voir l'essentiel de la programmation.

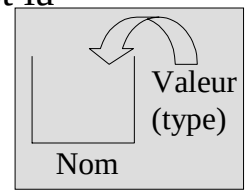
Premier et deuxième jour – Durée: ≈4 heures.

Contenu

- Les identificateurs de variables et leurs types
- La notion d'objet
- Les tableaux
- Les structures de contrôle
  - Conditions et sélections
  - Répétitions
- Les procédures et fonctions

Introduction aux Variables

- Une variable est un contenant pour une donnée.
- Une variable a un nom et ce nom est la référence pour la variable.
- Une variable a un type:
  - Nombre (entiers ou non)
  - Chaîne de caractères
  - Booléan (valeurs logiques *true* ou *false*)
  - Valeur unique null
- JavaScript est faiblement typé.



Declaration de Variables

La premiere fois qu'une variable est utilisée elle doit être déclarée par le mot clé **'var'.**

var identificateur = valeur;

L'identificateur est un nom de variable qui doit commencé par une letter ou un souligné '_', et peut avoir autant de caractères que nécessaire (letter, chiffre, souligné).

Les majuscules sont importantes et distinguées.
Ainsi, *mavariable* est differente de *MaVariable* et x ≠ X

Conversion de type à la volé

- Parceque JavaScript est faiblement typé, il est possible de:
  - Changer le type d'une variable;
  - Faire des operations sur des variables de differents types.
  - Le type majeur est le type chaîne de caractères.

Exemple de Variables

```
<HTML>
<HEAD>
<TITLE>Mon premier JavaScript avec variables</TITLE>
<script language="JavaScript">
 <!-- cacher le script
    var monNombre=35;
    var maChaine="2000";
    var monAutreChaine="TICE";
    var monAddition = monNombre+monNombre;
    var maConcatenation = maChaine + monAutreChaine;
    var monErreur = monNombre + monAutreChaine;
    var monCalcul = monNombre + maChaine;
    var monReve = monAutreChaine + maChaine;
 // fin de cache -->
 </script>
</HEAD>
```


Exemple de Variables (suite)

```
<BODY>
<script language="JavaScript">
 <!-- cacher le script
        document.write("monAddition="+monAddition+"<BR>");
        document.write("maConcatenation="+maConcatenation+"<BR>");
        document.write("monErreur="+monErreur+"<BR>");
        document.write("monReve="+monReve+"<BR>");
        monErreur = monNombre * 3;
        document.write("monErreur="+monErreur+"<BR>");
        monNombre="Salut";
        document.write("monNombre="+monNombre+"<BR>");
 // fin de cache -->
 </script>
</BODY>
</HTML>
```

```
monAddition=70
maConcatenation=2000TICE
monErreur=35TICE
monReve=TICE2000
monErreur=105
monNombre=Salut
```


Encore sur les Variables

- Il n'est pas toujours nécessaire de déclarer les variables.
- On peut assigner une valeur à une nouvelle variable sans la declarée avec **var**.
- Parcontre, on ne peut pas utiliser une variable sans la declarer ou l'initialiser au préalable.

Exemple avec Erreur

```
<HTML>
<HEAD>
<TITLE>Mon deuxieme JavaScript avec variables</TITLE>
<script language="JavaScript">
    var monNombre=35;
    var monAutreNombre=200;
</script>
</HEAD><BODY>
<script language="JavaScript">
        document.write("monNombre="+monNombre+"<BR>");
        document.write("monAutreNombre="+monAutreNombre+"<BR>");
        monGrandNombre=monNombre*monAutreNombre;
        document.write("monGrandNombre="+monGrandNombre+"<BR>");
        monErreur=monNombre*monQuelquechose;
        document.write("monErreur="+monErreur+"<BR>");
 </script>
</BODY></HTML>
```

var monQuelquechose=2;

Instruction JavaScript

- Comme une phrase en langage naturel se termine toujours par un point, une instruction JavaScript se termine toujours par un point virgule.
- instructionJavaScript**;**
- Exemple:
  - var monNombre=35;
  - document.write("monGrandNombre="+monGrandNombre+"
");
  - monErreur=monNombre*monQuelquechose;

II. JavaScript en Détail



Objectif: Apprendre comment JavaScript sauvegarde des données, comment un document est structuré vis-à-vis JavaScript, et voir l'essentiel de la programmation.

Premier et deuxième jour – Durée: ≈4 heures.

Contenu

JavaScript et la Notion d'Objet

- JavaScript n'est pas un langage orienté-objet à part entière.
- JavaScript is un langage à base d'objet.
- JavaScript utilise abondamment la notion d'objet.
- Il y a des objets prédéfinis mais le programmeur peut créer ses propres objets.

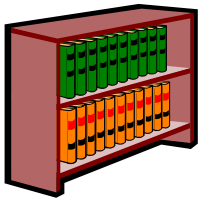
Qu'est ce qu'un Objet?

- En JavaScript, l'information est organisée et gérée en termes d'objets.
- Un objet représente un concept ou une chose que l'on veut modéliser.
- Un objet a des attributs (propriétés spécifiques à l'objet) ainsi que des méthodes (comportements propres à l'objet).
- Un attribut peut etre une valeur, un ensemble de valeurs, ou un objet.

Exemple: Un Livre est un Objet

Titre
Auteurs
Editeur
Nombre de pages
Prix
Ensemble de chapitres
Ensemble de figures et images
etc.

Chaque livre a les
même attributs
avec différentes valeurs



Quels sont les Objets, Quelles sont les Propriétés?

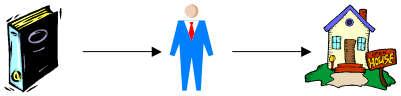
?

Accéder aux Propriétés d'un Objet

monObjet.unePropriete

Non de l'objet . Nom de l'atribut
Trouve une propriété du nom **unePropriete** appartenant à l'objet du nom **monObjet.**
Si l'attribut est aussi un objet, accéder à l'attribut de l'attribut:

monObjet.unePropriete.uneProprieteDeLaPropriete
Ex: Livre.Editeur.Adresse



document.MonFormulaire.Nom.value

Accéder aux Méthodes d'un Objet

monObjet.uneMethode(parametres)

Non de l'objet . Nom de la méthode (paramètres si existent)

Les paramètres sont les données de la méthode. S'il n'y a pas de paramètres, il faut comme même mettre les parenthèses.

monObjet.uneMethode()

Ex: document.write("Bonjour!")

Classes Objets Prédéfinis

- Il y a plusieurs objets intrinsèque prédéfinis dans le langage JavaScript:
  - –Date
  - –String
  - –Math
  - –Window
  - –Document
  - –Navigator
  - –History
  - –Location
  - –Form
  - etc…
- Ces objets ont leurs attributs et méthodes prédéfinis.

L'Objet Date

- L'objet Date nécessite une instanciation (création de l'objet) avec le mot clé ***new***.

  var aujourdhui= new date();
- La classe Date n'a pas de propriétés mais a une liste de méthodes:
  - •getDate()
  - •getDay()
  - •getHours()
  - •getMinutes()
  - •getMonth()
  - •getSeconds()
  - •getTime();
  - •getTimezoneOffset()
  - •getYear()
  - •setDate()
  - •setHours()
  - •setMinutes()
  - •setMonth()
  - •getSeconds()
  - •setTime();
  - •setYear()

  etc…

Exemple de Manipulation de Date

```
<HTML>
<HEAD>
<TITLE>Mon essai avec les dates</TITLE><script language="JavaScript">
      var maintenant=new Date();
      var dateNaissance = new Date(60,05,18);
</script></HEAD>
<BODY> <script language="JavaScript">
    document.write("Aujourd'hui nous sommes le:="+maintenant+"<BR>");
    document.write("La date de naissance de toto est le="+dateNaissance+"<BR>");
    document.write("la Date:"+dateNaissance.getDate()+ "/" +
                                (dateNaissance.getMonth()+1) + "/" +
                                (dateNaissance.getYear()+1900)+"<BR>");
     document.write("Il est maintenant:" + maintenant.getHours() + ":" +
                                          maintenant.getMinutes() + ":" +
                                          maintenant.getSeconds()+"<BR>");
     maintenant.setYear(2010);
     document.write("On s'est transporté au futur. La nouvelle date est:<br>"+maintenant);
 </script></BODY></HTML>
```


L'Objet String

- Lorsqu'on définit une constante ou une variable chaîne de caractères, JavaScript crée d'une façon transparente une instance String.
- L'objet String a une unique propriété: ***lenght***, mais plusieures méthodes:

•anchor()	chaine.anchor(ancre)➔<A name="ancre">chaine</A>
•big()	chaine.big()➔<BIG>chaine</BIG>
•blink()	chaine.bink()➔<BLINK>chaine</BLINK>
•bold()	chaine.bold()➔<BOLD>chaine</BOLD>
•fontcolor()	chaine.fontcolor(#FF0000)➔<FONT color="#FF0000">chaine</FONT>
•fontsize()	chaine.fontsize(5)➔<FONT size=5>chaine</FONT>
•italics();	chaine.italics() ➔<I>chaine</I>
•small()	chaine.small()➔<SMALL>chaine</SMALL>
•sub()	chaine.sub() ➔_{chaine}
•sup()	chaine.sup() ➔^{chaine}

L'Objet String (suite)

- Autres méthodes pour l'objet String:

•link()	
•toUpperCase()	chaine.toUpperCase() renvoie la chaîne en caractères majuscules
•toLowerCase()	chaine.toLowerCase() renvoie la chaîne en caractère minuscules
•substring()	chaine.substring(3,5) renvoie une sous-chaîne allant du 3ème caractère au 5ème caractère.
•indexOf()	chaine.indexOf(uneAutre) renvoie la premiere position à laquelle uneAutre apparait dans chaîne.
•charAt()	chaine.charAt(4) renvoie le caractère à la position 4.

Les positions dans une chaîne de caracterès commencent à 0. Pour la chaîne "Je suis alle a l'ecole" 'J' se trouve à la position 0, alors que 'a' se trouve à la position 8.

Exemple de manipulation d'objet String

```
<HTML>
<HEAD>
<TITLE>Mon essai avec les chaines</TITLE>
<script language="JavaScript">
var maChaine=prompt("Donnez moi une phrase","Je suis alle a l'ecole");
</script>
</HEAD><BODY>
<script language="JavaScript">
        document.write(maChaine+"<BR>");
        document.write(maChaine.bold()+"<BR>");
        document.write(maChaine.italics()+"<BR>");
        document.write(maChaine.fontcolor("red").blink()+"<BR>");
        document.write(maChaine.link("http://www.cnn.com")+"<BR>");
        document.write(maChaine+"<BR>");
 </script>
</BODY></HTML>
```


L'Objet Math

- La classe Math possède des propriétés qui revoient les valeurs de quelques constantes mathématiques telles que:

  Math.PI et **Math.E** qui revoient respectivement **3.1415926535897931** et **2.7182818284590451**
- Quelques méthodes utiles:

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

L'Objet Window

- JavaScript nous procure par défaut un objet nomé ***window*** représentant la fenêtre courante. Cet objet est situé à la racine de la hiérarchy décrivant les objets JavaScript.
- Cet objet a plusieurs propriétés:

•defaultStatus	message affiché par défaut dans la barre d'état
•frames	ensemble des divisions de la fenêtre du navigateur
•length	nombre de frames présent dans la fenêtre parent
•name	nom de la fenêtre
•parent	fenêtre parent
•self	fenêtre active
•status	message affiché dans la barre d'état
•top	sommet de la hierarchie objet définie sur le navigateur
•window	fenêtre active

Les Méthodes de l'Objet Window

- alert() fenêtre (modale) pour afficher un message.
- confirm() fenêtre (modale) pour faire un choix.
- prompt() fenêtre (modale) pour saisie d'une valeur.
- clear() efface le contenu de la fenêtre .
- open() ouvre une nouvelle fenêtre .
- close() ferme la fenêtre courante.

Pour ouvrir une fenêtre:
window.open("URL","NomdeFenetre","options");
Les option sont:

•menubar	•resizable	•toolbar
•status	•width	•location
•scrollbars	•height	•directories

Exemple avec les Fenêtres

```
<HTML>
<HEAD>
<TITLE>Mon essai avec les fenetres</TITLE>
</HEAD>
<BODY>
<A HREF="#" onMouseOver="window.status='bonjour!';return true;">Salut</A>
<BR>
<A HREF="test.html" TARGET="nouvelle">Ouvre moi</A>
<BR>
</BODY>
</HTML>
```


Ouvrir une Fenêtre

```
<HTML>
<HEAD>
<TITLE>Mon autre essai avec les fenetres</TITLE>
</HEAD>
<BODY>
<a href="#" onClick="maFenetre=window.open(
        'test.html','monTest','width=100,height=200,scrollbars,resizable');
         return true;">Ouvre moi</a><br>
<a href="#" onClick="maFenetre.focus();return true;">Montre moi</a><br>
<a href="#" onClick="maFenetre.blur();">Cache moi </a><br>
<a href="#" onClick="maFenetre.close();return true;">Ferme moi</a><br>
</BODY>
</HTML>
```

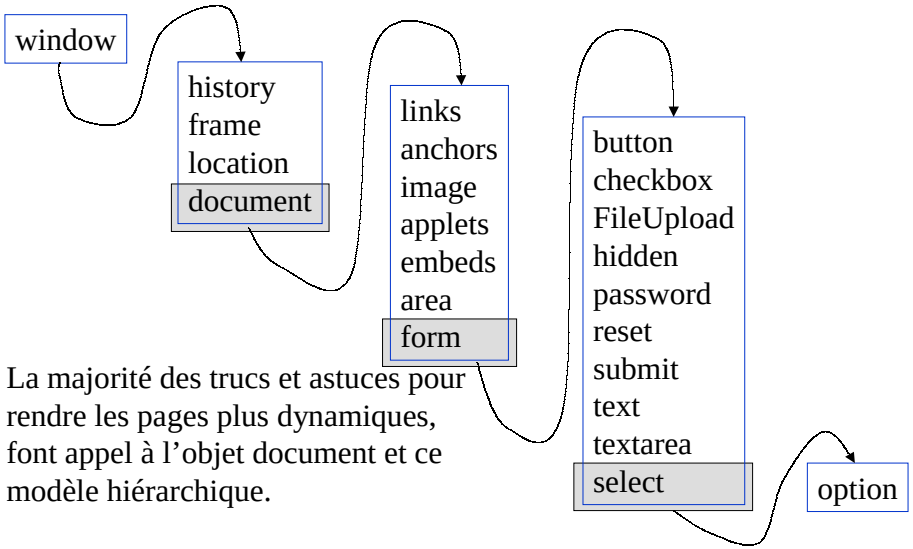

Exemple avec Options de Fenêtre

```
<HTML>
<HEAD>
<TITLE>Encore un essai avec les fenetres</TITLE>
</HEAD>
<BODY>
menubar,status,scrollbars,resizable,location,directories,width,height
<BR>
<a href="#" onClick="option=prompt('Quelles sont les options:');
                maFenetre=window.open('','monTest',option);
                return true;">Ouvre moi</a>
<BR>
<a href="#" onClick="maFenetre.close();
        return true;">Ferme moi</a>
<BR>
</BODY>
</HTML>
```

Document Object Model (DOM)

- Maintenant que l'on sait ce qu'est un objet en JavaScript et que l'on sait comment ouvrir une fenêtre, il est temps d'apprendre à propos du modèle document objet.
- JavaScript inclus des objets prédéfinis comme *window*. Ces objets on des propriétés et des méthodes prédéfinis.
- La propriété d'un objet peut aussi être un objet.

Hiérarchie des Objets



La majorité des trucs et astuces pour rendre les pages plus dynamiques, font appel à l'objet document et ce modèle hiérarchique.

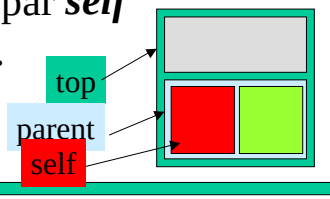
Quelques Autres Exemples d'Objets prédéfinis

- History contient l'historique complet des URL des sites visités durant la session courante.

```
<HTML>
<HEAD><TITLE>essai avec l'historique</TITLE></HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
  document.write("Nombre d'URL="+window.history.length);
</SCRIPT><FORM>
<INPUT TYPE=button VALUE="Back"
        onClick="window.history.back();return true;">
<INPUT TYPE=button VALUE="Retour 3 pages"
        onClick="window.history.go(-3);return true;">
</FORM>
</BODY></HTML>
```


Les Frames

- L'important à savoir au sujet des cadres (Frames), c'est que chaque cadre à l'intérieur d'une fenêtre (window) est considéré comme une autre fenêtre.
- Cela engendre la possibilité de fenêtres imbriquées.
- Il y a donc une fenêtre de sommet indiquée par ***top***, et chaque fenêtre a un parent indiqué par ***parent***. Une autoréférence est indiquée par ***self*** (interchangeable avec ***window***).



JavaScript et les Frames

```
<HTML><HEAD><TITLE>essai avec les cadres</TITLE>
<SCRIPT LANGUAGE="JavaScript">
var FrameVide = '<html><body bgcolor="#FFFFFF"></body></html>';
</SCRIPT>
</HEAD>
<FRAMESET rows="25%,*">
<FRAME SRC="EX16.HTM" name="controle">
<FRAME SRC="javascript:parent.FrameVide" name="cadre_cible">
</FRAMESET>
</HTML>
```

EX16.HTM

```
<HTML>
<HEAD><TITLE>cadre de controle</TITLE></HEAD>
<BODY>
<a href="#" onClick="top.cadre_cible.document.writeln('Bonjour!<br>');">Bonjour!</a>
<BR><a href="#" onClick="top.cadre_cible.document.writeln('Salut!<br>');">Salut!</a>
</BODY></HTML>
```


JavaScript et les Frames (bis)

```
<HTML><HEAD><TITLE>encore un essai avec les cadres</TITLE>
<SCRIPT LANGUAGE="JavaScript">
var FrameVide = '<html><body bgcolor="#FFFFFF"></body></html>';
</SCRIPT>
</HEAD>
<FRAMESET rows="25%,*">
<FRAME SRC="EX16b.HTM" name="controle">
<FRAME SRC="javascript:parent.FrameVide" name="cadre_cible">
</FRAMESET>
</HTML>
```

EX16b.HTM

```
<HTML>
<HEAD><TITLE>cadre de controle</TITLE></HEAD>
<BODY>
<a href="#" onClick="top.cadre_cible. document.bgColor='#00FF00';">Vert</a>
<BR><a href="#" onClick="top.cadre_cible. document.bgColor='yellow';">Jaune</a>
</BODY></HTML>
```


L'Objet Navigator

Il n y a pas de méthodes associées mais quelques attributs:

```
<html>
<head><title>Version du Navigateur</title></head>
<body>
<h1> Voyons de quel navigateur s'agit il</h1>
<hr>
<script language="javascript">
document.write("La version de ce navigateur est " + navigator.appVersion);
document.write ("<br>le Navigateur est <B>" + navigator.appName + "</B>");
document.write("<br>Le nom de code est " + navigator.appCodeName);
document.write ("<br>La plate-forme cliente est " + navigator.userAgent);
</script>
</body></html>
```


II. JavaScript en Détail



Objectif: Apprendre comment JavaScript sauvegarde des données, comment un document est structuré vis-à-vis JavaScript, et voir l'essentiel de la programmation.

Premier et deuxième jour – Durée: ≈4 heures.

Contenu

- Les identificateurs de variables et leurs types
- La notion d'objet
- Les tableaux
- Les structures de contrôle
  - Conditions et sélections
  - Répétitions
- Les procédures et fonctions

Les Tableaux

- Nous avons vu que les variables peuvent contenir des nombres, des chaînes de caractères et des références d'objets. Il y a un autre type d'information que JavaScript peut manipuler: Les **Tableaux**.

```
var produits = new Array("chaise", "table",  "armoire");
```

produit[0] ➔chaise

produit[1] ➔table

produit[2] ➔armoire

produit.length ➔ 3

Exemple Simple avec Tableau

```
<HTML>
<HEAD>
<TITLE>essai avec les tableaux</TITLE>
<SCRIPT LANGUAGE="JavaScript">
        var couleurs=new Array("red","blue","green","white");
</SCRIPT>
</HEAD>
<BODY>
<a href="#" onClick="document.bgColor=couleurs[0];return true;">Couleur1</a>
<BR>
<a href="#" onClick="document.bgColor=couleurs[1];return true;">Couleur2</a>
<BR>
<a href="#" onClick="document.bgColor=couleurs[2];return true;">Couleur3</a>
<BR>
<a href="#" onClick="document.bgColor=couleurs[3];return true;">Couleur4</a>
</BODY>
</HTML>
```


Un Autre Exemple avec Tableau

```
<HTML>
<HEAD>
<TITLE>Un autre essai avec les tableaux</TITLE>
<SCRIPT LANGUAGE="JavaScript">
        var processeurs=new Array("Intel PIII","AMD K7","Cirex");
</SCRIPT>
</HEAD><BODY>
<FORM NAME=formulaire>
<INPUT TYPE=text NAME=typeAchat VALUE=""><BR>
<INPUT TYPE=button VALUE="Intel"
   onClick="document.formulaire.typeAchat.value=processeurs[0];return true;">
<INPUT TYPE=button VALUE="AMD"
   onClick="document.formulaire.typeAchat.value=processeurs[1];return true;">
<INPUT TYPE=button VALUE="Cirex"
   onClick="document.formulaire.typeAchat.value=processeurs[2];return true;">
</FORM>
</BODY></HTML>
```


Publicité Alléatoire

```
<HTML><HEAD>
<TITLE>essai avec les tableaux</TITLE>
<SCRIPT LANGUAGE="JavaScript">
    var pubs=new Array("car01.jpg","car02.jpg","car03.jpg","car04.jpg",
                       "car05.jpg","car06.jpg","car07.jpg");
    var nombPub=pubs.length;
    var maintenant = new Date();
    var x = maintenant.getSeconds() % nombPub;
</SCRIPT>
</HEAD><BODY>
<H1>Voiture du jour</H1>
<SCRIPT LANGUAGE="JavaScript">
    document.write("Voiture numero "+ x+"<br>");
    document.write("<IMG WIDTH=320 HEIGHT=240 SRC="+pubs[x]+">");
</SCRIPT></BODY></HTML>
```

DOM Revisité

- Malgrès que l'on n'ait pas vu tous les détails des objets dans DOM, on a maintenant une idée sur comment ça marche.
- Après avoir vu les tableaux on sait qu'un document est structuré ainsi:

document

links[]
anchors[]
images[]
area[]
form[]

```
document.image[2].src="…";
```

Changer la 3ème image du document.

II. JavaScript en Détail

Objectif: Apprendre comment JavaScript sauvegarde des données, comment un document est structuré vis-à-vis JavaScript, et voir l'essentiel de la programmation.

Premier et deuxième jour – Durée: ≈4 heures.

Contenu

- Les identificateurs de variables et leurs types
- La notion d'objet
- Les tableaux
- Les structures de contrôle
  - Conditions et sélections
  - Répétitions
- Les procédures et fonctions

Les Conditionnelles

- JavaScript offre une structure de contrôle qui permet d'exécuter une série d'instructions seulement si une conditions est vérifiée.
- Si une condition est vraie, exécuter des commandes, sinon exécuter d'autres commandes.
- Exécuter des commandes seulement si condition est vraie.

if - then - else

Si condition alors instructions

```
if (condition)  instruction;
```

```
if (condition) {
        instruction1;
        instruction2;
        …
}
```

Si condition alors instructions sinon autresInstructions;

```
if (condition)  instruction;
else autreInstruction;
```

```
if (condition) {
        instruction1;
        instruction2;
        …
} else {
        instructionA;
        instructionB;
        …
}
```

Séléction ou Conditions Imbriquées

- On peut avoir plusieurs conditions imbriquées: si condition1 alors faire série1, sinon, si condition2 faire série2, sinon, si condition3 faire série3, sinon…

```
if (condition1)  {instructions1;}
else if (condition2) {instructions2;}
else if (condition3) {instructions3;}
else if (condition4) {instructions4;}
….
else {autresInstructions;}
```


Opérateurs Logiques

- Conjonction:
  - ET **&&**
  - exemple: (prix>=200 && membre==true)
- Disjonction
  - OU **||**
  - exemple: (age>26 || total==1000)
- Négation
  - NON !
  - exemple: (!finale && !(nombre==50))

Exemple de Conditionnelle

```
<HTML>
<HEAD><TITLE>conditionnelle en action</TITLE></HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
        var pays=prompt("Quel est ton pays d'origine?");
        if (pays.toUpperCase() == "TUNISIE")
                alert("Moi aussi");
        else
                alert("Je viendrai visiter un jour");
</SCRIPT>
</BODY>
</HTML>
```


Les Itérations

- JavaScript offre une structure de contrôle qui permet d'exécuter une série d'instructions plusieurs fois.
- Cette structure de contrôle est utile pour les opérations répétitives.
- Répéter x fois ou répéter jusqu'à ce qu'une certaine condition est vérifiée.

La Répétition *while*

Tantque condition faire instructions

```
while (condition)  instruction;
```

```
while (condition) {
        instruction1;
        instruction2;
        …
}
```

L'exécution des instructions est répétée tant que la condition est vraie (ou jusqu'à ce que la condition devienne fausse.

Jeu de Devinette avec Boucle

```
<HTML>
<HEAD><TITLE>devinette</TITLE>
<SCRIPT LANGUAGE="JavaScript">
        var maintenant = new Date();
        var x = maintenant.getSeconds();
        var reponse=prompt("devinez les secondes");
</SCRIPT></HEAD><BODY>
<SCRIPT LANGUAGE="JavaScript">
        while (reponse != x) {
                document.write( reponse+"<br>");
                if (x<reponse) alert("Trop grand");
                else alert("Trop petit");
                reponse=prompt("devinez les secondes");
        }
        document.write("Bravo!");
</SCRIPT></BODY></HTML>
```


La Répétition *for*

De début jusqu'à fin faire instructions
Exemple: de 1 jusqu'à N faire instructions

```
for (init;condition;incrémentation)  instruction;
```

```
for (init;condition;incrémentation) {
        instruction1;
        instruction2;
        …
}
```

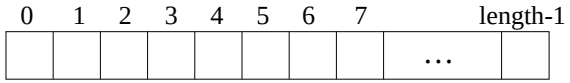
for (i=0;i<20;i++) document.write("*");

Exemple de Boucle *for*

```
<html>
<head><title>Table des Factoriels</title></head>
<body>
<h1>Factoriels de 1 a 9</h1>
<script language = "JavaScript">
<! - - cache moi
for (i=1, fact=1; i<10;i++, fact=fact *i){
        document.write (i + "! = " + fact);
        document.write ("<br>");
}
//  fin de cache -->
</script>
</body>
</html>
```

Les Boucles et les Tableaux

- Les boucles for et while sont très utiles pour manipuler les éléments d'un tableau.
- Les indices des éléments d'un tableau vont de 0 à length-1 (taille du tableau –1).
- Il est facile de parcourir les élément d'un tableau en faisant une boucle for de 0 à length-1.



Exemple de Boucle *for* avec Tableau

```
<html>
<head><title>Messagerie</title>
<script language = "JavaScript">
        var listeMessage=new Array("ALI.HTM","SALAH.HTM",
                                   "ZOHRA.HTM","SELMA.HTM");
</script></head>
<body>
<h1>Messages de coll&egrave;gues</h1>
<script language = "JavaScript">
for (i=0;i<listeMessage.length;i++){
        var fenetre=open(listeMessage[i],"NouveauMessage",
                         "width=300,height=200,scrollbars");
        document.write("Message "+(i+1)+ " lu: "+listeMessage[i]+"<BR>");
        alert("Confirmer pour aller au message suivant");
}
</script>
</body></html>
```


Tableaux et DOM

- Comme nous l'avons vu, un document peut avoir une série d'image. Ces images peuvent avoir des noms: <IMG SRC="image.gif" **NAME=monNom**>
- Le nom d'une image peut être utilisé pour manipuler l'image: document.monNom.src="car.gif";
- Avec DOM, un document a un tableau d'images: document.images[0], document.images[1],…
- On peut aussi manipuler les images on accedant au tableau: document.images[3].src="car.gif";

Un Example avec images[]

```
<html>
<head><title>Les images du document</title>
</head><BODY>
<IMG SRC=dblace1.jpg><IMG SRC=dblace2.jpg>
<IMG SRC=dblace3.jpg><IMG SRC=dblace4.jpg>
<IMG SRC=dblace5.jpg>
<SCRIPT LANGUAGE="JavaScript">
var laquelle=100;
while(laquelle<0 || laquelle>4)
        laquelle=prompt('Changer quelle image? (0 .. 4)','1');
window.document.images[laquelle].src="car01.jpg";
</SCRIPT>
</BODY>
</html>
```

II. JavaScript en Détail



Objectif: Apprendre comment JavaScript sauvegarde des données, comment un document est structuré vis-à-vis JavaScript, et voir l'essentiel de la programmation.

Premier et deuxième jour – Durée: ≈4 heures.

Contenu

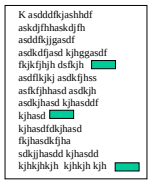
- Les identificateurs de variables et leurs types
- La notion d'objet
- Les tableaux
- Les structures de contrôle
  - Conditions et sélections
  - Répétitions
- Les procédures et fonctions

Les Fonctions et Procédures

- Les fonctions et procédures, appelées aussi sous-routines, sont fondamentales dans la programmation en JavaScript.
- Une sous-routine est un ensemble d'instructions sémantiquement liées et appelées à mainte reprises dans un programme.
- Grouper ces instructions dans une sous-routine évite de les re-écrire, et s'il y a un changement a faire dans le script, il suffit de le faire dans une seule place: la sous-routine.

Petite Application

- Supposons que l'on veuille tester la vitesses de lecture d'un lecteur. On pourrait insérer des boutons tout au long du texte pour afficher le temps courant lorsqu'un boutons est clické.



Le bouton pourrait être:

```
<FORM>
<INPUT TYPE=button VALUE=Temps onClick="
        var laDate=new Date();
        var heure=laDate.getHours();
        var minutes=laDate.getMinutes();
        var secondes=laDate.getSeconds();
        var leTemps=heure + ':' + minutes + ':' + secondes;
        alert(leTemps);">
</FORM>
```

Il y a un Problème

- Le code défini précédemment peut être mis à plusieurs endroits dans le texte à lire.
- Remarque: si les minutes ou les secondes sont inférieures à 10, un seul digit est imprimé (0,1, 2, 3, …9) et non deux (00, 01, 02,…09).
- On doit donc changer partout ou on a inséré le code.
- Meilleure solution: écrire une sous-routine pour calculer le temps et appeler la même routine pour chaque bouton dans le texte.

Fonction sans Paramètres

```
<SCRIPT LANGUAGE="JavaScript">
<!-- Cache moi
    function annonceTemps() {
        var laDate=new Date();
        var heure=laDate.getHours();
        var minutes=laDate.getMinutes();
        var secondes=laDate.getSeconds();
        var leTemps=heure + ':' + minutes + ':' + secondes;
        alert(leTemps);
    }
// fin de cache -->
</SCRIPT>
```

```
<FORM>
<INPUT TYPE=button VALUE=Temps
        onClick="annonceTemps();">
</FORM>
```


Changement dans la Fonction

```
function annonceTemps() {
        var laDate=new Date();
        var heure=laDate.getHours();
        var minutes=laDate.getMinutes();
        if (minutes<10) minutes="0"+ minutes;
        var secondes=laDate.getSeconds();
        if (secondes<10) secondes="0"+ secondes;
        var leTemps=heure + ':' + minutes + ':' + secondes;
        alert(leTemps);
}
```

Le changement est fait une seule fois et tous les boutons changent en conséquence.

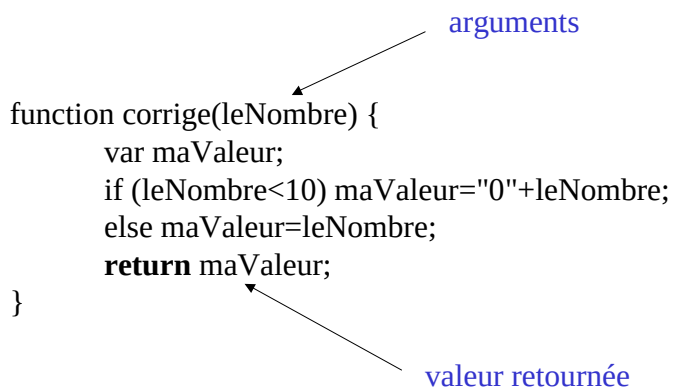
Fonctions avec Paramètres et Valeur de Retour

```
if (minutes<10) minutes="0"+ minutes;
if (secondes<10) secondes="0"+ secondes;
```

Opérations très similaires → on peut créer une nouvelle fonction pour effectuer cette opération.

On lui envoie comme donnée les minutes ou les secondes et la fonction retourne les mêmes minutes ou secondes préfixées par un 0 si nécessaire.

Fonctions avec Paramètres et Valeur de Retour (suite)



La Fonction annonceTemps Mise à Jour

```
function annonceTemps() {
        var laDate=new Date();
        var heure=laDate.getHours();
        var minutes=laDate.getMinutes();
        var corrigeMin = corrige(minutes);
        var secondes=laDate.getSeconds();
        var corrigeSec = corrige(secondes);
        var leTemps=heure + ':' + corrigeMin + ':' + corrigeSec;
        alert(leTemps);
}
```


Création d'Objet

- Les fonctions sont aussi utilisées pour définir et créer de nouveaux objets.

```
function maison(chbs, an, gar){
   this.chambres = chbs;
   this.annee = an;
   this.aGarage = gar;
}
var maMaison = new maison(8, 1990, true)
```

chambres
annee
aGarage

Objet - Tableau

- Chaque objet est en fait un tableau de valeurs de ses propriétés.
- Donc, on peut aussi écrire:

```
maMaison[0] = 8;
maMaison[1] = 1990;
maMaison[2] = true;
```

Index par Propriété

- Un tableau est indéxé par un entier (0 à N).
- Un tableau représentant un objet est aussi indéxé par les attribut de l'objet.
- Ainsi, on peut aussi écrire:

maMaison["chambres"] = 8;

maMaison["annee"] = 1990;

maMaison["aGarage"] = true;

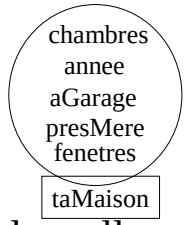
Extension Dynamique d'Objet

- On peut dynamiquement ajouter des attributs à un objet.

taMaison = new maison(5, 1974, true);

taMaison.presMere = "false";

taMaison.fenetres = 18;



- L'extension ne touche que l'instance à laquelle les attributs ont été ajouté. Les autres instances de la même classe restent inchangées.

Inculquer une méthode à un Objet

- Comme les langage orienté-objet, JavaScript permet d'ajouter des methodes aux objets que l'on crée.

```
function maison(chbs, an, gar){
  this.chambres = chbs;
  this.annee = an;
  this.aGarage = gar;
  this.affiche = afficherMaison;
}
```


Appeler la Méthode

```
function afficheMaison(){
  document.write("Maison a " + this.chambres);
  if (this.aGarage)
    document.write(" et un garage.");
  document.write("Construite en "+ this.annee);
}
```

maMaison=new maison(8,1990,true);

maMaison.affiche();

Résumé Partie II

• On a vu les bases du langage JavaScript et on a appris à écrire des scripts et à interpréter des scripts écrits par d'autres.
•On a exploré le concept objet en JavaScript et on a découvert des objets prédéfinis comme String, Date, Math, Window, etc., ainsi que leurs propriétés et méthodes.
• On a vu le modèle document objet (DOM).
• On a appris a stoker des données dans des variables pour des nombres, des chaînes de caractères, des références à des objets ainsi que pour des tableaux.
• On a appris une structure de contrôle qui permet de faire des conditionnelles, ainsi qu'une structure de contrôle qui permet les itérations.
• On a vu les fonctions, comment les définir et les utliser

III. Programmation Evénemencielle en JavaScript

Objectif: Apprendre comment JavaScript gère des événements particuliers liés aux actions de l'utilisateur. Apprendre a tenir compte de ces evénements dans notre programmation

Deuxième jour – Durée: ≈2 heures.

Contenu

- **Qu'est ce qu'un événement?**
- Quels sont les événements reconnus.
- Saisir et gérer les événements.

Application Interactive

- Pour avoir une application interactive il faut pouvoir reagir aux actions des utilisateurs
- Il faut offrir les moyens à l'utilisateur d'introduire des données, de clicker sur des objets, de choisir des options, etc.
- Il faut que l'application puisse capter les actions de l'utilisateur et réagir et s'adapter en conséquence.
- ➔ Identifier des événements et y répondre.

Qu'est ce qu'un Événement?

- Un événement est un changement dans l'environnement due à une action généralement produite par l'utilisateur.
- Ces action sont liées à la sourie ou au clavier.
- Une perturbation dans le document ou les objets du document représente un événement.
- Exemple: movement ou click de la sourie, changement de valeur d'un champ de données, etc.

III. Programmation Evénemencielle en JavaScript

Objectif: Apprendre comment JavaScript gère des événements particuliers liés aux actions de l'utilisateur. Apprendre a tenir compte de ces evénements dans notre programmation

Deuxième jour – Durée: ≈2 heures.

Contenu

- Qu'est ce qu'un événement?
- **Quels sont les événements reconnus.**
- Saisir et gérer les evénements.

Quels sont les Événements Reconnus par JavaScript?

- Le clic du bouton (gauche) de la sourie.
- Le passage du pointeur de la sourie sur un objet.
- La sélection ou désélection d'un champs de saisie.
- La modification du contenu d'un champ de saisie.
- La soumission d'un formulaire.
- Le chargement d'un nouveau document.
- La sortie du navigateur ou d'un document.

Événement en JavaScript

•Nous avons déjà vu un échantillon d'événements dans les exemples de JavaScript utilisés pour illuster les concepts étudies.
•JavaScript capture et gère 9 types d'événements répertoriés ci-dessous:

• onClick	un clic du bouton gauche de la sourie sur une cible.
• onMouseOver	un passage du pointeur de la sourie sur une cible.
• onBlur	une perte de focus d'une cible.
• onFocus	une activation d'une cible.
• onSelect	une selection d'une cible.
• onChange	une modification du contenue d'une cible.
• onSubmit	une soumission d'un formulaire.
• onLoad	un chargement d'une page.
• onUnload	la femeture d'une fenetre ou le chargement d'une autre page autre que la courante.

Les Cibles des Événements

Les cibles sont en fait les capteurs des événements; des objets appartenant au modèle document objet (DOM). Les objets ne captent pas les même événements.

• onClick	A HREF, BUTTON, CHECKBOX, RADIO, RESET, SUBMIT
• onMouseOver	A HREF
• onBlur	PASSWORD, SELECT, TEXT, TEXTAREA
• onFocus	PASSWORD, SELECT, TEXT, TEXTAREA
• onSelect	PASSWORD, TEXT, TEXTAREA
• onChange	PASSWORD, SELECT, TEXT, TEXTAREA
• onSubmit	FORM
• onLoad	BODY (window), IMAGE
• onUnload	BODY (window)

D'Autres Événements que vous Pouvez Rencontrer

- onError
- onAbort
- Ces événements sont utilisés avec window et image et proviennent d'une interruption du téléchargement d'une page ou d'une image (interruption manuelle, abort, ou pas).

Comment ça Marche?

- Le navigateur intercèpte les événements (aussi appelés interruptions) et agit en conséquence.
- Action ➔ Événement ➔Capture ➔Action
- Les actions sont associées aux cibles au moyen des balises HTML.
- <BALISE surEvenement="Action">
- Exemple: **<A href**="#" **onClick**="alert('Merci!');">

Un Autre Événement: Fin de Compte à Rebours

- L'objet window a deux méthodes spécifiques pour la gestion d'un compte à rebours.
- ***setTimeout()*** permet de spécifier un compteur de millisecondes associé à une instruction. Après l'intervale de temps spécifié, une iterruption est produite et l'instruction est évaluée.
- ***clearTimeout()*** annule le compte à rebours.

Une Horloge Vivante

- On a vu comment afficher le temps courant dans un exemple précédent.
- Pour avoir une horloge, il suffit d'afficher le temps régulièrement à la même place.
- Pour ceci, il faut rappeler automatiquement à intervalle régulier la fonction qui affiche l'heure.
- Définir un compte a rebours a l'intérieur de la fonction d'affichage fait l'affaire.

Exemple d'Horloge

```
<html><head><title>Horloge</title>
<SCRIPT LANGUAGE="JavaScript">
function horloge () {
        var maintenant = new Date();
        var heures = maintenant.getHours();
        var minutes = maintenant.getMinutes();
        var secondes = maintenant.getSeconds();
        var resultat = "" + heures+ ((minutes < 10)?":0":":")
        + minutes + ((secondes < 10)?":0":":") + secondes;
        return resultat;
}
function quelleHeure() {
  var heure=horloge();
  document.forms[0].temps.value=heure;
  setTimeout('quelleHeure()',1000);
}
</SCRIPT></head>
```

```
<BODY onLoad="quelleHeure()">
<FORM>
<INPUT TYPE=TEXT SIZE=8
        NAME=temps>
</FORM>
</BODY></html>
```


Diapos avec setTimeout()

Continuellement superposer des images on rappelant la sous-routine d'affichage d'image après un certain delai.

```
<html>
<head><title>Les images a compte a rebours</title>
<SCRIPT LANGUAGE="JavaScript">
var diapos=new Array("jump1","jump2","jump3","jump4",
                     "jump5","jump6","jump7","jump8");
var nombre=diapos.length;
var numero=0;
var timer=0;
```


Diapos (suite)

```
function avant() {
        stop();
        numero = (numero+1) %nombre;
        window.document.images[0].src=diapos[numero] + ".jpg";
        window.document.forms[0].image.value=numero+1;
}

function arriere() {
        stop();
        if (numero==0) numero=nombre-1; else numero--;
        document.images[0].src=diapos[numero] + ".jpg";
        window.document.forms[0].image.value=numero+1;
}
```


Diapos (suite)

```
function suivant() {
        avant();
        timer=setTimeout("suivant()",500);
}

function stop() {
        clearTimeout(timer);
}

function reprendre() {
        timer=setTimeout("suivant()",500);
}
```

Diapos (suite)

```
</SCRIPT>
</head>
<BODY>
<CENTER>
<IMG SRC=jump1.jpg>
<FORM>
<INPUT TYPE=BUTTON value=&lt;&lt; onClick='arriere();'>
<INPUT TYPE=BUTTON value=Stop onClick='stop();'>
<INPUT TYPE=BUTTON value=&gt;&gt; onClick='avant();'>
<INPUT TYPE=BUTTON value=Show onClick='reprendre();'>
Image:<INPUT TYPE=TEXT SIZE=1 NAME=image>
</FORM>
</CENTER>
</BODY>
</html>
```


III. Programmation Evénementielle en JavaScript

Objectif: Apprendre comment JavaScript gère des événements particuliers liés aux actions de l'utilisateur. Apprendre a tenir compte de ces evénements dans notre programmation

Deuxième jour – Durée: ≈2 heures.

Contenu

- Qu'est ce qu'un événement?
- Quels sont les événements reconnus.
- **Saisir et gérer les événements.**

Ordrer d'Exécution

- En programmation traditionnelle, l'ordre d'exécution des instructions d'un programme est dicte par l'ordre d'apparition de ces instructions dans le programme. L'exécution se fait instruction par instruction.
- En programmation événementielle, l'ordre d'exécution dépend des avènements.
- Un événement intercepte engendre l'exécution des instructions correspondantes à l'évènement.

Ordrer d'Exécution (suite)

- Les fonctions doivent être définies avant d'être invoquées.
- **Attention**: Un événement peut se produire avant même que le document soit téléchargé en entier. En effet, l'utilisateur peut déjà actionner la sourie alors que seulement une partie de la page est chargée et présentée à l'écran.
- Avant d'invoquer une fonction, il faut être sur qu'elle est définie.
- Mettre les définitions de fonctions dans <HEAD>

Associer une Fonction à un Événement

- Après avoir défini les fonctions, il faut les associer aux divers événements que l'on veut capturer.
- Pour ceci, on associe les mots clé des événements (event handlers), vu précédemment, aux diverses balises HTML.

```
<BODY onLoad="Instructions JavaScript">
<FRAMESET onLoad="Instructions JavaScript">
window.onLoad=referenceFonction;
```


Exemple d'Association

- Un formulaire **<FORM METHOD=…>…</FORM>** permet de saisir des données. Le navigateur envoie ces données au server lorsque le bouton de soumission est clické.
- En ajoutant onSubmit="**return verifierForm(this)**" à la balise <FORM>, la fonction verifierForm est exécutée avant que le navigateur ne soumette les données au serveur, et ainsi intercepter et vérifier les données à temps.

Les Événements par Objet JavaScript

- On a vu la liste des événements interceptés par JavaScript.
- Voyons maintenant, objet par objet, les événements les plus courants ainsi que la syntaxe HTML.
- Nous mettrons l'emphase sur les événements liés aux objets des formulaires

Événements Courants et *window*

- onLoad invoqué lorsque la page est chargée.
- onUnload invoqué lorsque la pae n'est plus.
- Syntaxe HTML
  - **<BODY**
    [BACKGROUND= "imageURL"] [BGCOLOR = "color"]
    [TEXT = "color"] [LINK = "color"] [ALINK = "color"] [VLINK = "color"]
    [onLoad = "handler"] [onUnload = "handler"] >
  - **<FRAMESET**
    [ROWS="lignes"] [COLS="colonnes"]
    [onLoad="handler"] [onUnload="handler"]>

Événements Courants et *link*

- onClick invoqué lorsque le lien est clické.
- onMouseOver invoqué lorsque survolé par la sourie.
- Syntaxe HTML
  - **<A**
    HREF = "URL"
    [TARGET = "fenetre_cible"]
    [onClick = "handler"]
    [onMouseOver = "handler"] >

Événements Courants et *form*

- onReset invoqué juste avant re-initialization.
- onSubmit invoqué juste avant soumission.
- Syntaxe HTML
  - **<FORM**
    [NAME = "nom"]
    [TARGET = "fenetre_cible"]
    [ACTION = "URL"]
    [METHOD = GET/POST]
    [ENCTYPE = "encryption"]
    [onReset = "handler"]
    [onSubmit = "handler"] >

Événements Courants et *button*

- onClick invoqué lorsque le bouton est clické.
- Syntaxe HTML
  - **<INPUT**
    TYPE = button
    VALUE= "libelle"
    [NAME = "nom"]
    [onClick = "handler"] >

Attributs

name
form
type
value

Événements Courants et *checkbox*

- onClick invoqué lorsque la case à cocher est clické.
- Syntaxe HTML
  - **<INPUT**
    TYPE = checkbox
    [VALUE= "libelle"]
    [NAME = "nom"]
    [CHECKED]
    [onClick = "handler"] >

Attributs

name
form
checked
defaultChecked
type
value

Événements Courants et *radio*

- onClick invoqué lorsque la case à cocher est clické.
- Syntaxe HTML
  - **<INPUT**
    TYPE = radio
    [VALUE= "libelle"]
    [NAME = "nom"]
    [CHECKED]
    [onClick = "handler"] >

Attributs

name
form
checked
defaultChecked
type
value

Événements Courants et *select*

- onChange invoqué lors d'un changement
- onBlur invoqué lorsque select pert le focus
- onFocus invoqué lorsque select gagne le focus
- Syntaxe HTML
  - **<SELECT**
    NAME = "nom"
    [SIZE= taille]
    [MULTIPLE]
    [onChange = "handler"]
    [onBlur = "handler"]
    [onFocus = "handler"] >

Attributs

name
form
length
options
selectedIndex
type

Événements Courants et *text*

- onChange invoqué lors d'un changement
- onBlur invoqué lorsque le texte pert le focus
- onFocus invoqué lorsque le texte gagne le focus
- Syntaxe HTML
  - **<INPUT**
    TYPE=text
    [NAME = "nom"]
    [VALUE= "defaut"]
    [SIZE=taille] [MAXLENGTH=taillemax]
    [onChange = "handler"]
    [onBlur = "handler"]
    [onFocus = "handler"] >

Attributs

name
form
value
defaultValue
type

Événements Courants et *textarea*

- onChange invoqué lors d'un changement
- onBlur invoqué lorsque le texte pert le focus
- onFocus invoqué lorsque le texte gagne le focus
- Syntaxe HTML
  - **<TEXTAREA**
    [NAME = "nom"]
    [ROWS= "ranges"] [COLUMNS="colonnes"]
    [WRAP=OFF/VIRTUAL/PHYSICAL]
    [onChange = "handler"]
    [onBlur = "handler"]
    [onFocus = "handler"] >

Attributs

name
form
value
defaultValue
type

Événements Courants et *submit*

- onClick invoqué lorsque le bouton est clické.

- Syntaxe HTML
  - **<INPUT**
    TYPE = submit
    [VALUE= "libelle"]
    [NAME = "nom"]
    [onClick = "handler"] >

Attributs

name
form
type
value

Événements Courants et *reset*

- onClick invoqué lorsque le bouton est clické.

- Syntaxe HTML
  - **<INPUT**
    TYPE = reset
    [VALUE= "libelle"]
    [NAME = "nom"]
    [onClick = "handler"] >

Attributs

name
form
type
value

Événements Courants et *image*

- onClick invoqué lorsque le bouton est clické.

- Syntaxe HTML
  - **<INPUT**
    TYPE = image
    SRC= "URL"
    [NAME = "nom"]
    [onClick = "handler"] >

Attributs

name
form
type
src

Saisir une Adresse e-mail

```
<html>
<head><title>Valider e-mail</title>
<SCRIPT LANGUAGE="JavaScript">
 ...

</script>
<BODY onLoad="aZero();">
<FORM name="monAdresse" onsubmit="return verifierEmail();">
E-Mail: <input type="text" name="email" size="25"
        onchange="this.value=checkemail();"><br>
<input type="submit" value="Soumet" name="B1">
<input type="reset" value="Init" name="B2" onclick="aZero();">
</FORM>
</BODY>
</html>
```

Valider e-mail

```
function checkemail(){
  var ladresse=document.monAdresse.email.value;
  var nouvelle = "";
  for (k = 0; k < ladresse.length; k++){
    ch = ladresse.substring(k, k+1);
    if ((ch >= "A" && ch <= "Z") || (ch >= "a" && ch <= "z") || (ch == "@") ||
        (ch == "[") || (ch == "]") || (ch == ".") || (ch == "_") || (ch == "-") ||
        (ch >= "0" && ch <= "9"))
      nouvelle += ch;
  }
  if (ladresse!= nouvelle) {
    if (confirm("Vous avez entre des espaces ou des caracteres non valides.\n\n
       Clickez OK pour fixer ceci.\n\nClickez CANCEL pour garder inchange."))
      return nouvelle;
    return ladresse;
  }
  return ladresse;
}
```


Valider e-mail (suite)

```
function verifierEmail(){
  var ladresse = document.monAdresse.email.value
  if (document.monAdresse.email.value == "") {
    alert("Entrez votre e-mail S.V.P.")
    document.monAdresse.email.focus();
    return false;
  }
  ladresse=document.monAdresse.email.value;
  b = ladresse.substring(0,1)
  if (b == '@') {
    alert("Verifiez votre e-mail. Il doit y a voir un prefix avant '@'\n\n
          Exemple: jha@alibaba.tn")
    document.monAdresse.email.select();
    document.monAdresse.email.focus();
    return false;
  }
```


Valider e-mail (suite)

```
  if ((ladresse.indexOf("@") == -1) || (ladresse.indexOf(".") == -1)) {
     alert("Verifiez votre e-mail.  Une adresse doit inclure les signes '@' et '.'\n\n
           Exemple: jha@alibaba.tn");
     document.monAdresse.email.select();
     document.monAdresse.email.focus();
     return false;
   }
   c = ladresse.indexOf("@")
   d = ladresse.indexOf(".");
   e = ladresse.substring(c,d);
   if (e.length < 2) {
     alert("Vous devez introduire quelque chose entre les signes \"@\" et \".\"")
     document.monAdresse.email.select();
     document.monAdresse.email.focus();
     return false;
   }
```


Valider e-mail (suite)

```
b = ladresse.indexOf(".")
  ladresse= ladresse.substring(b,ladresse.length);
  if (ladresse.length <2) {
    alert("Vous devez introduire au moins un caractere apres le signe '.'")
    document.monAdresse.email.select();
    document.monAdresse.email.focus();
    return false;
  }
  alert("Merci");
  return false;
}

function aZero(){
   document.monAdress.email.value = "";
   document.monAdress.email.focus();
}
```

Résumé Partie III

- On a vu ce qu'est un événement en JavaScript.
- On a vu comment intercepter un événement.
- On a énuméré la list des événements reconnus.
- On a passer en revue les élements des formulaires et leurs événements associés.
- On a vu de quoi consiste la programmation événementcielle et comment se déroule l'exécution.
- On a illustré les conceptes d'évenement par des exemples simple.

IV. Exemples Pratiques

Objectif: Voir et disséquer quelques exemples concret de programmes JavaScript pratiques.

Trosième jour – Durée: ≈1 heure 30

Contenu

- Vérification des données dans un formulaire de saisie;
- Animation des images dans une page HTML;
- Utilisation des cookies pour stoker des information sur le client;
- Petit site de commerce électronique avec chariot de transactions

Quelques Références Utiles

- JavaScript avec Netscape
http://developer.netscape.com/docs/manuals/index.html?content=javascript.html
- Jscript chez Microsoft http://msdn.microsoft.com/scripting/default.htm
- Builder.com http://www.builder.com
- Designing with JavaScript http://www.webcoder.com/
- Ask the JavaScript Pro http://www.inquiry.com/techtips/js_pro/
- WebMonkey http://hotwired.lycos.com/webmonkey/programming/javascript/
- Yahoo
http://dir.yahoo.com/Computers_and_Internet/Programming_Languages/JavaScript/