

Les formulaires, le langage JavaScript

Walid Belkhir

Université de Provence

`belkhir@cmi.univ-mrs.fr`

`http://www.lif.univ-mrs.fr/~belkhir/`

Plan

- 1 Les formulaires en HTML
- 2 Le langage JavaScript
 - Le noyau du JavaScript
 - Les classes prédéfinis
 - Les événements

Pourquoi les formulaires

- Inter-activité avec l'utilisateur en proposant des zones de dialogue
- Selon le choix de l'utilisateur, un traitement est associé aux zones de dialogue :
 - au niveau client avec **JavaScript**, ...
 - au niveau du serveur avec **PHP**, ...

Exemple de formulaire : champs texte, cases à cocher, ...

Principe du formulaire

- Les différents champs de saisie sont décrites à l'aide des balises HTML
- Chaque zone est identifiée par un **nom** auquel sera associée une **valeur** (par l'utilisateur)
- A chaque zone de saisie peut être associé un traitement au niveau clients à l'aide d'un événement JavaScript
- Quand le formulaire est soumis, les couples (nom/valeur) de toutes les zones sont transmis dans la requête HTTP au serveur

Les éléments d'un formulaire

Trois catégories :

- **input** : champs de saisie de texte et divers types de boutons :
 - type="text" : zone de texte
 - type="password" : zone de texte caché
 - type="checkbox" : cases à cocher
 - type="radio" : minimum 2, un seul sélectionnable
 - type="submit" : bouton de soumission du formulaire
 - type="reset" : bouton de remise à zéro des champs
 - type="hidden" : bouton caché
- **select** : menus déroulant, listes à ascenceurs
 - size="1" : un seul élément sélectionnable
 - size="n", $n > 1$: liste à choix multiples
- **textarea** : zone de saisie d'un texte long.

La balise <FORM>

- <FORM> ... </FORM>
- Les champs (de type input, select et textarea) ne seront visible que s'ils sont à l'intérieur d'une balise <FORM>
- Attributs : **METHOD**, **NAME**, **ACTION**, **TARGET**
 - **METHOD** : valeurs GET ou POST qui indiquent la façon dont les données sont transmises au programme.
 - **ACTION** : URL du programme qui sera exécuté quand l'utilisateur clique sur un bouton de soumission
 - **NAME** : identifiant pour distinguer les différents formulaires
 - **TARGET** : cible dans laquelle la réponse du programme sera affichée

La balise <FORM>

Propriétés de l'objet FORM

- **action** : accès à l'attribut ACTION
`<FORM name="f1" action="/bin/prog1" >...</FORM>`
`<SCRIPT>document.f1.action="/bin/prog2" </SCRIPT>`
- **method** : accès à l'attribut METHOD
- **target** : accès à l'attribut TARGET
- **enctype** : type d'encodage des données transmises vers le serveur avec la méthode POST
- **elements** : accès aux *objets du formulaires*
 - `elements.length` : nombre d'objets du formulaire
 - `elements[n].name` : nom du $n^{\text{ième}} + 1$ objet du formulaire
 - `elements[n].value` : valeur du $n^{\text{ième}} + 1$ objet du formulaire

La balise <FORM>

- Méthode de l'objet FORM : **submit()** :
⇒ déclenche l'envoi du formulaire comme si l'utilisateur avait appuyé sur un bouton de soumission

```
<SCRIPT> document.f1.submit()</SCRIPT>
```

- Événement JS associé à l'objet FORM : **onSubmit()**
permet l'exécution de code JS avant l'envoi du formulaire :
<FORM name="f1" method="post" action="/bin/prog1" target="_blank" onSubmit=" ma_methode(this)" >

<INPUT type="TEXT" >

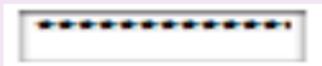
- Attributs : NAME, VALUE, SIZE, MAXLENGTH :
 - SIZE : taille d'affichage de la zone (en caractères)
 - MAXLENGTH : taille de remplissage de la zone (en caractère)

```
< INPUT TYPE="TEXT" NAME="nom"  
VALUE="entrer votre nom ici..." SIZE="20"  
MAXLENGTH="50" >
```

- Propriétés :
name, value, defaultValue, type, form (le nom du formulaire qui contient l'élément INPUT)
- Méthodes :
focus(), blur(), select()
- Événements :
onBlur, onChange, onFocus, onSelect

INPUT type="PASSWORD"

- Attribut : NAME, VALUE, SIZE, MAXLENGTH
`<INPUT type="PASSWORD" NAME="pass"
VALUE="entrer votre passwd ici" SIZE="8" >`



- Propriétés :
name, value, defaultValue, type, form
- Méthodes :
focus(), blur(), select()
- Pas d'événement associé

INPUT type="CHECKBOX"

- Cases à cocher à choix multiple
- Attributs : NAME, VALUE, CHECKED
`<INPUT type="CHECKBOX" NAME="cours" VALUE="1" CHECKED>` OW : `
`
`<INPUT type="CHECKBOX" NAME="cours" VALUE="2" CHECKED>` HTML `
`
`<INPUT type="CHECKBOX" NAME="cours" VALUE="4">` JS



- Propriétés : name, value, type, form, checked
- Méthode :
`document.f1.cours[1].click()` : coche/ décoche la case OW
- Événement :
`onClick` : quand l'utilisateur coche la case

<INPUT type="RADIO" >

- Choix **d'une et une seule** option parmi n

- Attributs : NAME, VALUE, CHECKED

```
<INPUT type="RADIO" NAME="cours" VALUE="1" > OW
```

```
<INPUT type="RADIO" NAME="cours" VALUE="3" CHECKED > HTML
```

```
<INPUT type="RADIO" NAME="cours" VALUE="4" > JS
```

OW HTML JS

- Propriétés :

name, value, type, form, checked, index (le rang du bouton sélectionné), length

- Méthode

document.f1.cours[2].click() : sélectionne la case JS

- Événement :

onClick : quand l'utilisateur coche la case

<INPUT type="SUBMIT">

- Envoi des données et exécution du programme spécifié par l'attribut ACTION de <FORM>
- Attributs : NAME, VALUE
`<INPUT type="SUBMIT" NAME="su" VALUE="login" >`
`<INPUT type="SUBMIT" NAME="su" VALUE="logout" >`



- Propriétés :
name, value, type, form
- Méthode :
click(), soumet le formulaire
- Événement :
onClick

<INPUT type="RESET" >

- Recharge tous les champs du formulaire à leur valeur par défaut
- Attributs : NAME, VALUE

```
<INPUT type="RESET" NAME="eff" VALUE="Effacer" >
```



- Propriétés :
name, value, type, form
- Méthode
click() : réinitialise le formulaire
- Événement :
onClick()

<INPUT type="BUTTON" >

- N'a de sens que dans un contexte JavaScript
- Attributs : NAME, VALUE

```
<INPUT type="BUTTON" name="b" VALUE="Help"  
onClick=" AideEnLigne()" >
```



- Propriétés :
name, value, type, form
- Méthode :
click() : simule un click de l'utilisateur
- Événement :
onClick()

<SELECT> et <OPTION>

- Attributs de <SELECT> : NAME, SIZE, MULTIPLE
 - SIZE : taille de la liste (nbr d'éléments)
 - MULTIPLE : autorise la sélection multiple si *SIZE* > 1
- Attributs de <OPTION> : VALUE, SELECTED

```
<SELECT NAME="pop" >
```

```
<OPTION VALUE="v1" > a </OPTION>
```

```
<OPTION VALUE="v2" SELECTED> b </OPTION>
```

```
<OPTION VALUE="v3" > c </OPTION>
```

```
</SELECT>
```

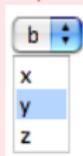
```
<SELECT NAME="mul" size="3" MULTIPLE> <OPTION>
```

```
<OPTION VALUE="v1" > x </OPTION>
```

```
<OPTION VALUE="v2" SELECTED> y </OPTION>
```

```
<OPTION VALUE="v3" > z </OPTION>
```

```
</SELECT>
```



<SELECT> et <OPTION>

- Propriétés d'un objet <SELECT>
 - name, type (*select/select-one/select-multiple*), form, length
 - selectedIndex : rang de l'option sélectionnée (dans le cas d'une liste multiple, rang de la première option sélectionnée)
- Méthodes d'un objet <SELECT> : focus(), blur()
- Propriétés relative aux options
 - defaultSelected, selected, text, value
document.f1.mul.options[2].text : vaut z
 - on peut modifier, ajouter, supprimer des items de la liste

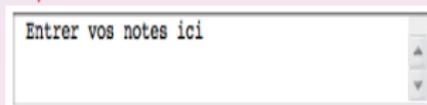
<TEXTAREA>

- Zone de saisie de texte libre
- Attributs : NAME, ROWS, COLS

```
<TEXTAREA NAME="t1" ROWS="3" COLS="44" >
```

Entrer vos notes ici

```
</TEXTAREA>
```

A screenshot of a web browser showing a text area input field. The text area is rectangular with a light gray border and contains the text "Entrer vos notes ici". To the right of the text area is a vertical scrollbar with up and down arrow buttons.

- Propriétés :
name, value, defaultValue, type, form, cols, rows
- Méthode :
focus(), blur(), select()
- Événements :
onBlur, onChange, onFocus, onSelect

Plan

- 1 Les formulaires en HTML
- 2 Le langage JavaScript
 - Le noyau du JavaScript
 - Les classes prédéfinis
 - Les événements

JavaScript : les principes

- Insertion des scripts (programmes) directement dans le code des pages HTML
- Exécution de code du côté client (temps de réponse plus court)
- Exemples : test d'un formulaire avant envoi, affichage dynamique, ...
- Conçu pour traitement local des événements clients (click, déplacement souris, ...)
- JavaScript est un langage
 - interprété (le code est analysé et exécuté au fur et à mesure par le navigateur)
 - à base d'objets

Le langage JavaScript

JavaScript permet

- d'effectuer des calculs (comme n'importe quel autre langage)
- de programmer des actions en fonction des **événements**
ex. si la zone de texte contient une adresse mail *valide* alors l'enregistrer dans un fichiers sinon afficher un message d'erreur

Le langage JavaScript

Que peut-on faire avec JavaScript ?

- petites applications : calculettes, éditions de texte, jeu, ...
- aspect graphiques : modification d'images, gestion de fenêtres, modification locale de la page HTML, ...
- test de validité des données saisies dans les formulaires : vérifier si une zone de saisie a le bon format, ...

Le noyau JavaScript

- Au niveau du langage, il y a
 - le noyau JavaScript : opérateurs de base, objets prédéfinis, structures, ...
 - un ensemble d'objets associés au navigateur :
 - fenêtres (objet *window*)
 - documents (objet *document*),
 - images, ...
- JavaScript n'est pas Java

Insertion de code JavaScript dans un document HTML

3 méthodes :

- 1 utilisation de la balise `<script>...</script>`
 - déclaration de fonction dans l'en-tête entre `<head>...</head>`
 - appel de fonction ou exécution d'une commande JavaScript dans `<body>...</body>`
 - insertion d'un fichier JavaScript *externe*
- 2 utilisation d'une URL :
 - une URL peut être une exécution de fonction JavaScript (entre `<a>...` ou `<form>...</form>`)
- 3 utilisation des attributs de balises qui gèrent les événements utilisateur :
 - `<BALISE onEvenement="code JavaScript">`

Insertion de code JavaScript

```
<HTML><HEAD>  
  <SCRIPT LANGUAGE=" JAVASCRIPT" >  
    fonction fermer() { window.close();}  
  </SCRIPT>  
</HEAD><BODY>  
  <SCRIPT LANGUAGE=" JAVASCRIPT" >  
    document.write(" pour fermer la fenetre");  
  <SCRIPT>  
    <br><a href="javascript :fermer();" cliquer ici</a>  
    <br> ou passer la souris sur  
    <a href="" onMouseOver=" fermer();" ce lien </a>  
</BODY></BODY>
```

La balise <SCRIPT>

- Permet l'exécution de code par le navigateur
- Syntaxe :
`<SCRIPT LANGUAGE="nom" SRC="URL" > ... </SCRIPT>`
- Attribut LANGUAGE :
 - "JavaScript" par défaut
- Attribut SRC
 - pour charger du code présent dans un autre fichier

Insertion d'un fichier externe : un exemple

fichier index.html

```
<HTML><HEAD>
  <SCRIPT LANGUAGE=" JAVASCRIPT" SCR=" fermer.js" >
  </SCRIPT>
</HEAD><BODY>
  <SCRIPT LANGUAGE=" JAVASCRIPT" >
    document.write('pour fermer la fenetre');
  </SCRIPT>
  <br><a href="javascript :fin();" cliquer ici </a>
  <br> ou passer la souris sur
  <a href="" onMouseOver=" fin();" ce lien </a>
</BODY></HTML>
```

fichier fermer.js

```
function fin(){
  window.close();}
```

Le langage JavaScript

- Typage faible
- Opérateurs et instructions : ceux du langage C
- Méthodes
 - globales (prédéfinies, associées à tous les objets)
 - fonctions (définies par l'utilisateur)
- Objets
 - prédéfinis : *String, Date, Math, ...*
 - spécifiques au navigateur : *window, document, ...*
- Commentaires : comme en langage C :
//..... ou bien */**/*
- Séparateur d'instruction : ;

Opérateurs JavaScript

Ceux du langage C :

- arithmétiques : +, -, *, %
- incrémentation (pré/post indexée) : ++i , i++
- logique : && (et), || (ou), ! (non)
- comparaison : ==, !=, <=, ...
- concaténation de chaînes : +
- affectation : **variable=valeur**

NB. + peut être un opérateur d'addition ou de concaténation selon le type des opérandes.

Variables JavaScript

- Déclaration de variables :
 - optionnelle mais conseillée
 - avec **var**
 - le type n'est pas précisé lors de la déclaration
- Utilisation d'une variable globale d'un autre document :
window.parent.NomVar
- Types :
 - *Number* : $-2.4E-90$
 - *Boolean* : *true*, *false*
 - *String* : "chaine" ou 'chaine'

Instructions de bases

- Conditionnelle :
`if (condition) {Suite_Instructions;} [else {Suite_Instructions;}]`
- Boucles
 - `for(i=0 ;i<V ;i++){instructions ; }`
 - `while (condition){instructions ; }`
 - `do {instructions} while (condition)`
- Sortie d'une boucle : `break ;`
- Passer à l'itération suivante d'une boucle : `continue ;`
- `typeof(qq_chose)` : retourne le type de qq_chose

Instructions de base

- **switch** :

```
switch(variable){
```

```
  case 'valeur_1' :
```

```
    Instructions à exécuter si variable==valeur_1 ;
```

```
    break ;
```

```
  case 'valeur_2' :
```

```
    Instructions à exécuter si variable==valeur_2 ;
```

```
    break ;
```

```
  default :
```

```
    Instructions à exécuter si toutes les conditions précédentes ne  
    sont pas vérifiées ;
```

```
}
```

Les fonctions

```
function nom(arg_1, ..., arg_n){  
  Instructions ;  
  return valeur ;  
}
```

- pas de spécification de type dans la liste d'arguments

Fonctions et formulaires : un exemple

```
<html> <head>  
<SCRIPT LANGUAGE=" JAVASCRIPT" SRC=" FACT.js" >  
</SCRIPT>  
</HEAD><BODY>  
<FORM name=" MonForm"  
  onSubmit=" alert('la facotielle de ' + this.nombre.value + 'est' +  
  fact(nombre.value) );" >
```

Entrer un nombre :

```
<input type="text" name=" nombre" >  
<input type="submit" value=" calcule" >  
</FORM>  
</BODY> </html>
```

Entrer un nombre :

Les tableaux

- Construction d'un tableau sans préciser le contenu
`var Tab = new Array();`
- Construction d'un tableau en précisant la taille
`var Tab = new Array(3);`
- Création + initialisation du tableau
`var Tab = new Array(e_1 , ..., e_n);`
 - les e_i n'ont pas forcément le même type
 - les indicent varient de 0 à $n - 1$
- propriété `length` : taille du tableau : `Tab.length`

Tableaux et Objets

- Un tableau est un objet prédéfini qui possède des propriétés et des méthodes
- Tableaux **associatifs** : l'indice peut être une chaîne de caractères :

Tab['nom'] ⇔ **Tab.nom**

- Parcours de l'ensemble des propriétés d'un objet
for (p in window){
 document.write(window[p]); }



Les classes prédéfinis

- 1 **Array** : manipulation des tableaux
- 2 **Boolean**
- 3 **Date** : traitement de la date
- 4 **Function** : création de fonctions
- 5 **Math** : (sinus, cosinus, racine carrée ...)
- 6 **Image** : gestion dynamique des images
- 7 **Option** : gestion des listes créées avec `<SELECT>`
- 8 **RegExp** : manipulation des expressions régulières
- 9 **String** : manipulation des chaînes

La classe String

- Propriété : length
- Méthodes principales :
 - indexOf(chaine,index)
 - substring(début,fin+1), charAt(n)
 - lastIndexOf(chaine), toLowerCase()
 - toUpperCase(), split(motif)

ex.

```
var Ch=" Salut Salut" ;  
Ch.indexOf(" S" );    —> 0  
Ch.lastIndexOf(" S" ); —> 6  
Ch.charAt(4);        —> t  
Ch.substring(6,11);  —> Salut
```



La classe Math

- Propriétés : constantes
 - E, PI, ...
- Méthodes :
 - cos, sin, tg, acos, atan, asin, ...
 - abs : valeur absolut
 - ceil (partie entière sup), floor (partie entière inf)
 - exp, log, sqrt, pow(x,n)
 - max(x,y), min(x,y)
 - random $\in [0, 1]$



La classe Array

- Propriété : length
- Méthodes principales :
 - **join** : concatène tous les éléments en une chaîne de caractères séparés par une virgule
 - **reverse** : transpose le tableau
 - **sort** : trie selon la relation d'ordre passée en argument (fonction de comparaison fournie par l'utilisateur), ordre lexicographique par défaut



Les événements

- Des événement sont associés aux actions de l'utilisateur et à certaines balises
 - correspondance *objet, événement*
 - le code JavaScript est exécuté quand l'événement se produit sur l'objet associé
- Exemple d'événement :
 - passage de la souris sur un lien hyper-texte
 - soumission d'un formulaire

Les événements

2 manières pour associer une action à un couple (objet, événement)

- 1 utiliser des attributs de balises spécifiques :

```
<a href="index.html" onMouseOver="alert('Salut');">
```

cliquer ici

```
</a>
```

- 2 spécifier pour un élément et un événement la fonction qui sera exécutée :

```
<body><a href="index.html" >
```

cliquer ici </body>

```
<script>
```

```
function MaFonc(){ alert('Salut');};
```

```
document.links[0].onmouseover=MaFonction;
```

```
</script>
```



Événement	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
ondblclick	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
onload	lors du chargement.	Objet javascript : Image, window, frame img, body, frameset, frame



Les événements

onmousedown	quand le bouton de la souris 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 souris 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

Les événements

La classe *event*

- Quand un événement se produit, le navigateur crée un objet *event* donc les propriétés sont
 - *type*
 - *target* objet sur lequel l'événement s'est produit
 - *layerX*, *layerY* position de la souris en pixels dans la page
 - *screenX*, *screenY* position de la souris en pixels dans l'écran