Programmation Web client riche

Auteur : **Nassim Hadj-rabia**

Contenu du cours

- langage javascript
- Utilisation d'un framework : Jquery
- JSON, XML
- Ajax (web asynchrone)

De JavaScript… à jQuery (1)

- **La naissance de JavaScript**
  - 1995: **Brendan Eich** développe pour *Netscape Communications Corporation*, un langage de script, appelé Mocha, puis LiveScript et finalement JavaScript
- **La guerre des navigateurs**
  - Concurrence de *Netscape* et *Microsoft* qui ont développé leur propre variante de JavaScript avec des fonctionnalités supplémentaires et incompatibles, notamment dans la manipulation du DOM (Document Object Model)
  - 1997 : Adoption du standard ECMAScript.

Différence d'implémentation des spécifications suivant le type de navigateur.

De JavaScript… à jQuery (2)

- Les spécifications **ECMAScript** ont permis de pérenniser **JavaScript**
- JavaScript permet le contrôle d'une grande partie des paramètres d'une page WEB
- Avec l'arrivée de l'objet *XMLHttpRequest* permettant le développement d'applications **AJAX** (Asynchronous JavaScript and XML), **JavaScript** est devenu incontournable dans le développement d'interfaces WEB évoluées (WEB2.0)

De JavaScript… à jQuery (3)

- Existence de *Framework* permettant une indépendance vis à vis du navigateur
  - PrototypeJS => www.prototypejs.org
  - Jquery => jquery.com
  - Mootools => mootools.net
  - DoJo Toolkit => www.dojotoolkit.org
  - Angular JS => angularjs.org
  - BackBone JS => backbonejs.org
  - ...

Le langage Javascript

Langage de scripts (i.e. langage de programmation interprété)

- un script est envoyé au navigateur (par un serveur web) et est exécuté sur celui-ci
- javascript est un standard de fait des langages de script coté client
- principales utilisations :

  a) interagir avec un document html (en particulier le modifier) en utilisant le DOM (Document Object Model)

  b) limiter les échanges entre client et serveur web :

  - contrôle des données d'un formulaire avant transmission à un serveur (moins vrai depuis HTML 5)
  - envoi de requêtes http sans rechargement de la page (AJAX :Asynchronous JavaScript And XML)

Association HTML – Javascript

- **à l'aide de la balise <script> suivi du code javascript**

  ```
  <script type="text/javascript">
  <!-- mon code Javascript -->
  </script>
  ```

- **dans un fichier externe dans l'entête html**

  ```
  <script type="text/javascript" src="fichier.js"></script>
  ```

Premier exemple

Langage: orienté objet, faiblement typé, syntaxe proche java

```
< !DOCTYPE html>
<html> <head> <meta charset= "utf-8 "><title>Premier exemple </title>
<script type="text/javascript">
function affiche() {
var laDate = new Date();
var message = " Bonjour ! Nous sommes le " + laDate;
window.alert(message); }
</script>
</head><body onload="affiche();">
</body> </html>
```

Type de données de base en javascript

Booléen=> True, False

Nombre => exemple : 1234 ou 56,89

Chaîne de caractères => exemple: "coucou"

Objet => exemple: maDate=new date()

Variables

- sensibles à la casse
- déclaration de variable => **var** i=3; (var non obligatoire mais conseillé)
- Les variables en javascript sont dynamiques et faiblement typées.
- les variables peuvent être globales ou locales (déclarées AVEC var dans une fonction)

```
var i=0; // variable globale

function test() {
for (i=0; i<5; i++) {
...}  }
test();
alert(i); // i=5
```

Variables

- Une variable peut changer de type:

```
<script
type="text/javascript">
a= 'salut';
a= 3;
a= a + 2;
a= a+7;
</script>
```

```
<button onclick="alert('valeur de a '+ a)">montrer
a</button>
```

Quelques opérations sur les types de bases

- **Les nombres :** on dispose entre autres des opérateurs +, -, *, /, % (modulo)

pour la division euclidienne de x par y, on fera :q= Math.floor(x/y);// Calcul de partie entière

Quand un résultat n'est pas calculable (racine carrée de -1 ou 0/0), on obtient NaN ("Not a number").

NaN est le seul nombre... pas égal à lui même. Pour le tester, il faut utiliser isNaN() :

a= 0/0 ; alert (a==NaN); // affiche false

alert(isNaN(a)); // affiche true

- **Les booléens :** Opérateurs booléens usuels && (et), ! (not), || (ou).
- **Les strings :** La déclaration de string peut se faire en utilisant des guillements simples '...' ou doubles "....".

L'opérateur "+" permet de concaténer deux string, voire une string et autre chose (un nombre par exemple).

Comparaisons

Égalité : Deux comparateurs disponibles : "==" qui sait faire des conversions, et "**===**" (trois signe égal) qui n'en **fait pas.**

a === b

si a et b ont le même type, et correspondent à la même valeur (pour les objets, s'ils ont la même adresse).

a == b

si, après conversion éventuelle, a et b ont la même valeur. Les règles de conversions sont complexes.

si a ou b est un nombre, et que l'autre est une string, on converti la string en nombre, et on compare les valeurs

ainsi, les comparaisons suivantes sont vraies :

"01" == "01" (deux chaînes identiques)

1 == "01" (la chaîne "01" est convertie en nombre, et vaut 1)

En revanche : "01" != "1" (deux chaînes différentes)

Conversions automatiques

- Règles complexes et peu intuitives :

  1 + '0' -> '10'

  1 * '0' -> 0

  1+ true -> 2

Suggestion: se contenter de la concaténation; pour les autres : utiliser les conversions explicites

```
s="15";
a= 2 * parseInt(s);
alert(a); // affiche 30
```

Quelques opérations sur les types de bases

les chaînes

- sont des objets
- constantes déclarées entre "..." *ou* entre '...'.
- ont une propriété, length (leur longueur)
- se comparent avec "==" !!!
- de nombreuses méthodes:
  - **charAt**(pos) retourne une String
  - **charCodeAt**(pos) retourne le code **unicode** du caractère (si inférieur à 65535)
  - **split**(separateur) : renvoie le tableau obtenu en découpant la chaîne.
  - **indexOf**(sousChaine,index), indexOf(sousChaine): premier indice de la sous-chaîne, à partir de l'index donné, ou -1.
  - **substr**(debut,longueur) : renvoie la sous-chaîne démarrant à "debut", de longueur "longueur".
  - **toLowerCase**() : la chaîne en minuscules ; et toUpperCase() : la chaîne en majuscules.
  - String.**fromCharCode**(c1,c2....) : permet de passer d'une suite de codes ascii à la chaîne correspondante.

- **les tableaux (Array) :**
  - doivent être créés, mais on n'a pas besoin de fixer leur taille:

```
tab= new Array(); // Création du tableau
tab[0]= 2; tab[1]= 3; s= 0;
for (i= 0; i < tab.length; i++) {  s= s + tab[i]; }
```

  - peuvent être initialisés à la création : tab= new Array("un", "deux", "trois");
  - longueur donnée par l'attribut **length**
  - passage d'un tableau à une chaîne de caractères par "join": l'argument de join est inséré entre les éléments.

    s= tab.join(""); // s vaut **undeuxtrois**

    s= tab.join(":"); // s vaut un**:**deux**:**trois
  - Push : Ajout d'une case au tableau : tab.push("quatre") ;
  - Pop : supprime le dernier élément du tableau : tab.pop()

- **Tableaux associatifs :**

```
var t= {};
t["java"]= "langage de programmation orienté objet";
 t["javac"]= "compilation d un programme java ";
```

Ou

```
var t= {"java": "langage de programmation orienté objet",
        "javac": "compilation d un programme java"};
```

Structures de contrôle

- **If**
- **for : Deux syntaxes possibles**

```
s= 0;
for (var i= 0; i < tab.length; i++) {
s= s+ tab[i];
}
```

```
var t= new Array("un",
"deux", "trois");
for (var i in t) {
alert( i + " : " + t[i]);
}
```

- **while, do...while, switch**

Fonctions

- **les fonctions**
  - Bout de code réutilisable, avec un nom.
  - Déclarées à l'aide du mot-clef function.
  - Une fonction *peut* retourner une valeur, à l'aide du mot-clef return

```
<script type="text/javascript">
function somme(a,b) {
return a+b;
}
</script>
<button onclick="alert(somme(3,2))"> montrer somme (entiers) </button>
<button onclick="alert(somme('bonjour ','monde'))"> montrer somme (chaînes)</button>
```

Objets

- objet => propriétés + fonctions spécialisées dans la manipulation de l'objet
- utilise la notation pointée
- Exemple :

  - **window.alert**("ok");

  - **window.location.href**="http://www.univ-nantes.fr";

Types d'objet Javascript

- objets **core** = ceux de ECMAscript
- objets du **BOM** (Browser Objet Model) = ceux du navigateur
- objets du **DOM** (Document Objet Model) = ceux permettant d'accéder aux éléments du document html

ECMA – objets core

- Number
- Boolean
- String
- Date
- Math

BOM

les objets mis à disposition par le navigateur

PAS de norme les décrivant => un usage

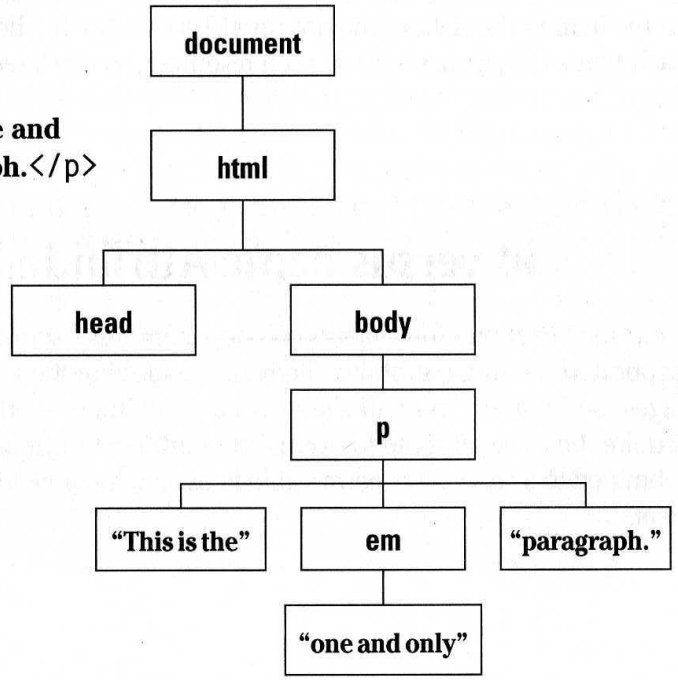
- window
- navigator
- screen
- history
- location
- ...

DOM

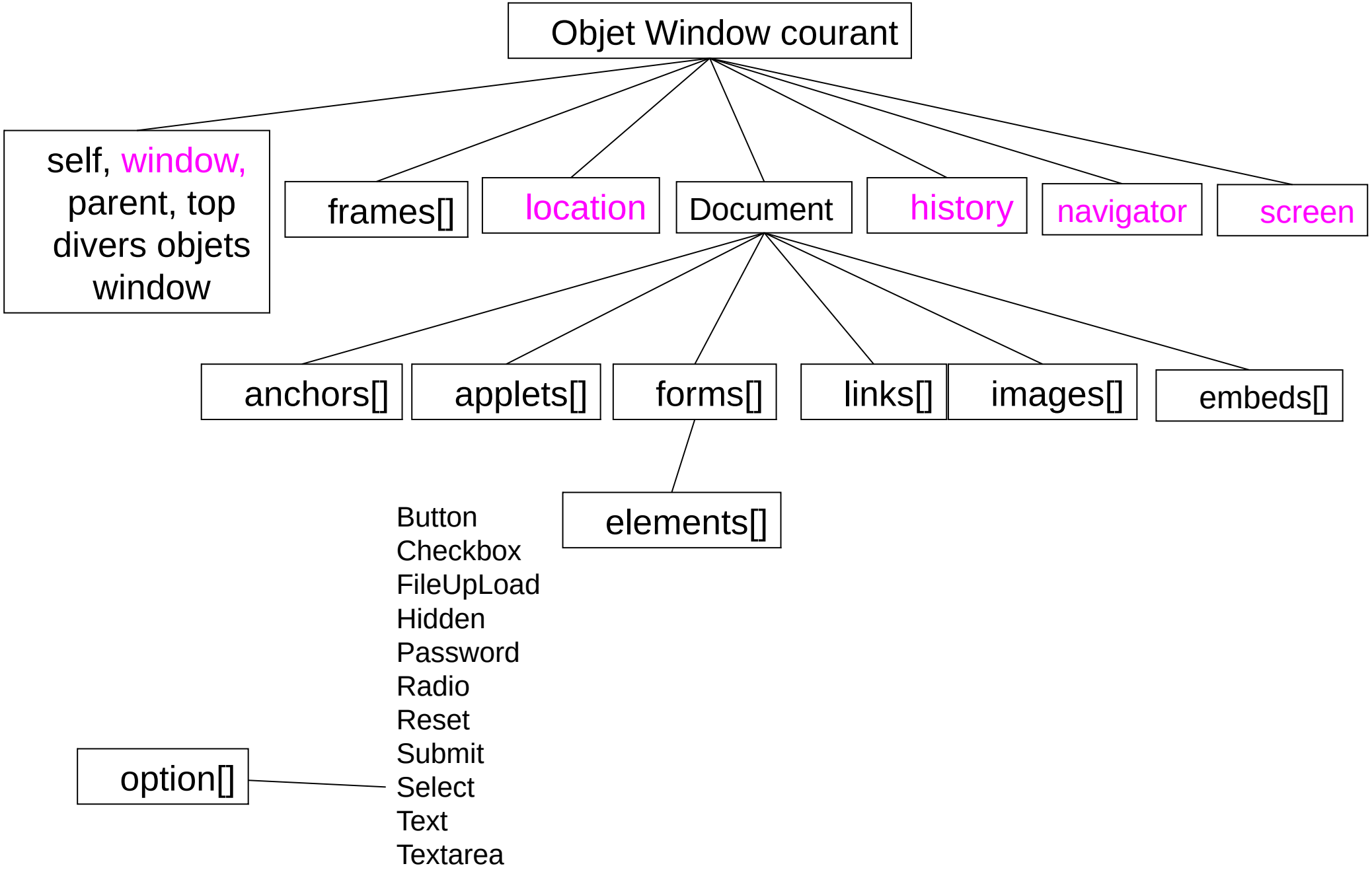
Le Document Object Model (DOM) est une convention indépendante du langage pour représenter et interagir avec des objets dans des documents en HTML, XHTML ou XML.

A simple HTML document node tree.

```
<html>
  <head></head>
  <body>
      <p>This is the <em>one and
        only</em> paragraph.</p>
  </body>
</html>
```



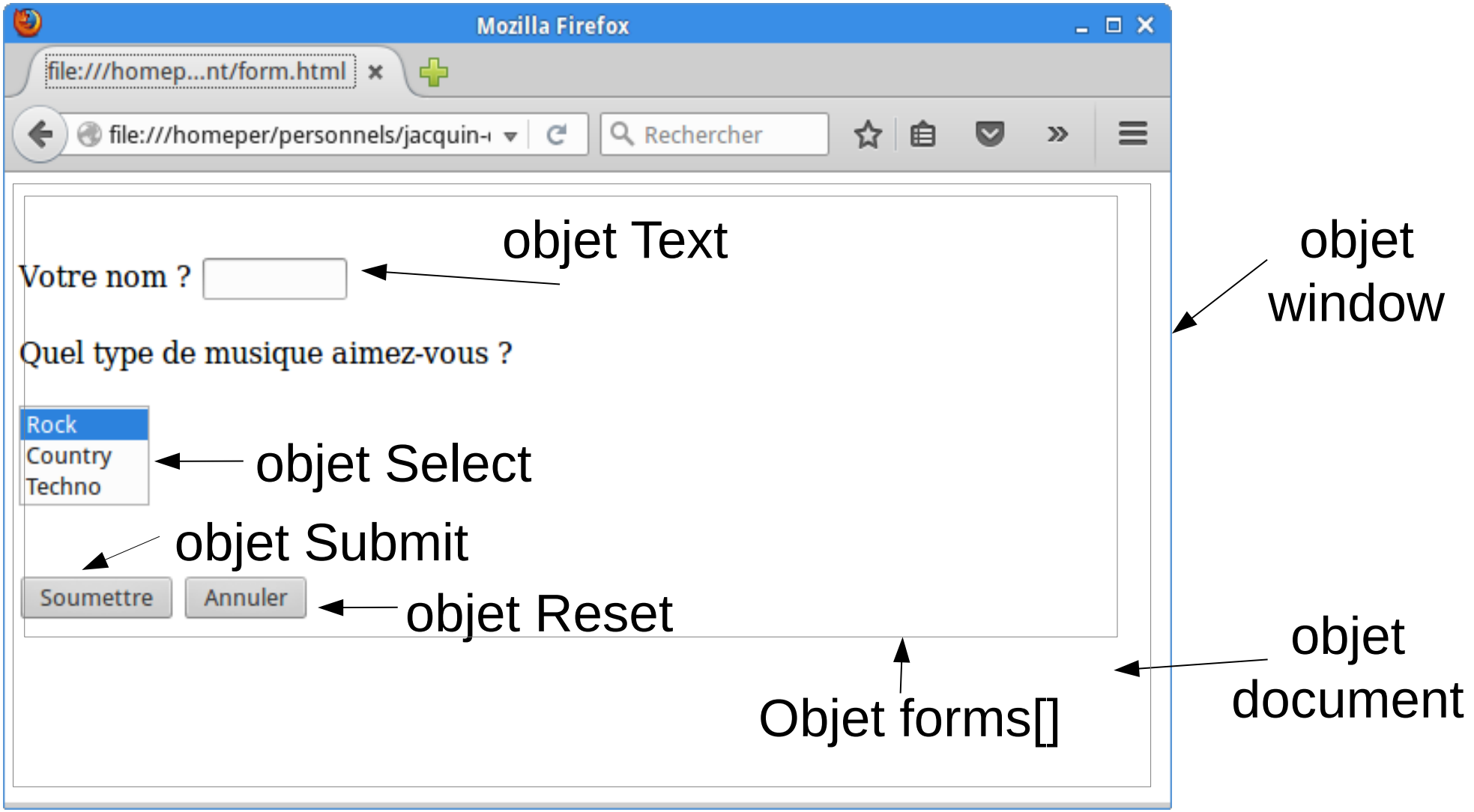
La hiérarchie d'objets javascript



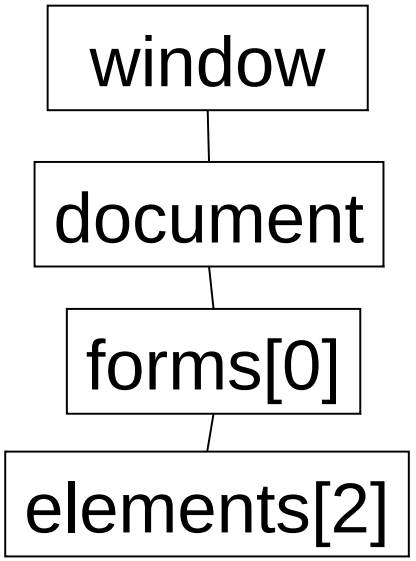
Exemple de code HTML et de référencement d’objets

```
<!DOCTYPE html>
<html>
<body>
<form name= "formulaire" action="http://mapage.php" method="get">
Votre nom ?
<input name="nom" size="8" type="text" >
Quel type de musique aimez-vous ?
<select name="musique" size="3" >
<option> Rock </option>
<option> Country</option>
<option> Techno </option>
</select>
<input name="soumettre" type="submit" value="Soumettre">
<input name="annuler" type="reset" value="Annuler">
</form>
</body>
</html>
```

La hiérarchie des objets graphiques



La hiérarchie associée



Pour référencer, un élément d'un formulaire dans un script javascript, on donne le chemin dans la hiérarchie:
Exemple pour l'objet de type Submit (2 manières)

- window.document.forms[0].elements[2]

- window.document.forms['formulaire'].elements['soumettre']

Pour accéder aux propriétés des objets graphiques: *objet.nom_propriété*

window.document.forms[0].elements[1].length (longueur de la liste)

Possibilité de nommage des objets

• On peut nommer les objets par le nom qui leur a été attribué via l'attribut *name* des balises (si cet objet dispose d'un attribut *name* !)

Dans l'exemple précédent:

window.document.forms[0].elements[1]

équivaut à

window.document.formulaire.musique

• Si deux éléments ont le même nom alors un tableau est automatiquement créé avec comme nom le nom des éléments

• Nommage de l'objet courant par ***this***.

Accès aux éléments d'une page (à privilégier)

- Pour manipuler un élément, on lui donne un **identifiant**.

  Pour éviter une navigation complexe dans l'arbre DOM, les deux méthodes à privilégier sont :

- **getElementById()**: renvoie l'élément correspondant à cet élément
- **getElementsByTagName()**: renvoie un tableau des éléments de ce nom

<p **id="dom1"**>Un texte</p>

<button onclick="document.getElementById('**dom1**').style.color='red';">peindre</button>

*var x=document.getElementsByTagName('LI');//accés avec x[**i**]*

var nbLI=document.getElementsByTagName("LI").length;

http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_document_getelementsbytagname_length

Propriétés et méthodes

Tous les objets prédéfinis dans le langage ont des propriétés et des méthodes prédéfinies ainsi que des événements qui leur sont associés.

Exemple window:

- quelques propriétés: width, height, resizable, status,…
- quelques méthodes: alert, prompt, open, close
- un événement associé: onUnload

DOM et CSS (Cascading Style Sheets)

- Les attributs de style CSS d'un élément du DOM permettent de changer son aspect.
- Ils sont accessibles par la propriété "**style**" des éléments(Exemple DOCSS.html).

```
<!DOCTYPE html>
<html>
<body>

<h1 id="id1">My Heading 1</h1>

<button type="button"  onclick="document.getElementById('id1').style.color = 'red'">
Click Me!</button>

</body>
</html>
```

http://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_color2

DOM et CSS

On utilise en particulier la propriété "**display**", et ses valeur "**none**" ou "**block**", pour faire apparaître et disparaître des éléments.

```
<button
onclick="document.getElementById('coucou').style.display='none';">cacher</button>
<button onclick="document.getElementById('coucou').style.display='block';"> non
cacher</button>

<div style="border: 2px solid red" id="coucou">
Vous le voyez ?
</div>
```

- **Manipulation des images**

On peut changer ce qu'une image affiche en modifiant sa propriété "src":

```
<img id="img1" src="image1.png"
onmouseover="document.getElementById('img1').src='image2.png'"
onmouseout="document.getElementById('img1').src='image3.png'"
/>
```

La gestion basique des événements (à ne plus utiliser)

- des événements peuvent être liés à certains éléments HTML.
- le nom de l'attribut concerné se construit en préfixant par **on** le nom de l'événement.

```
<a href="#" onMouseOver="coucou()"> message important</a>
```

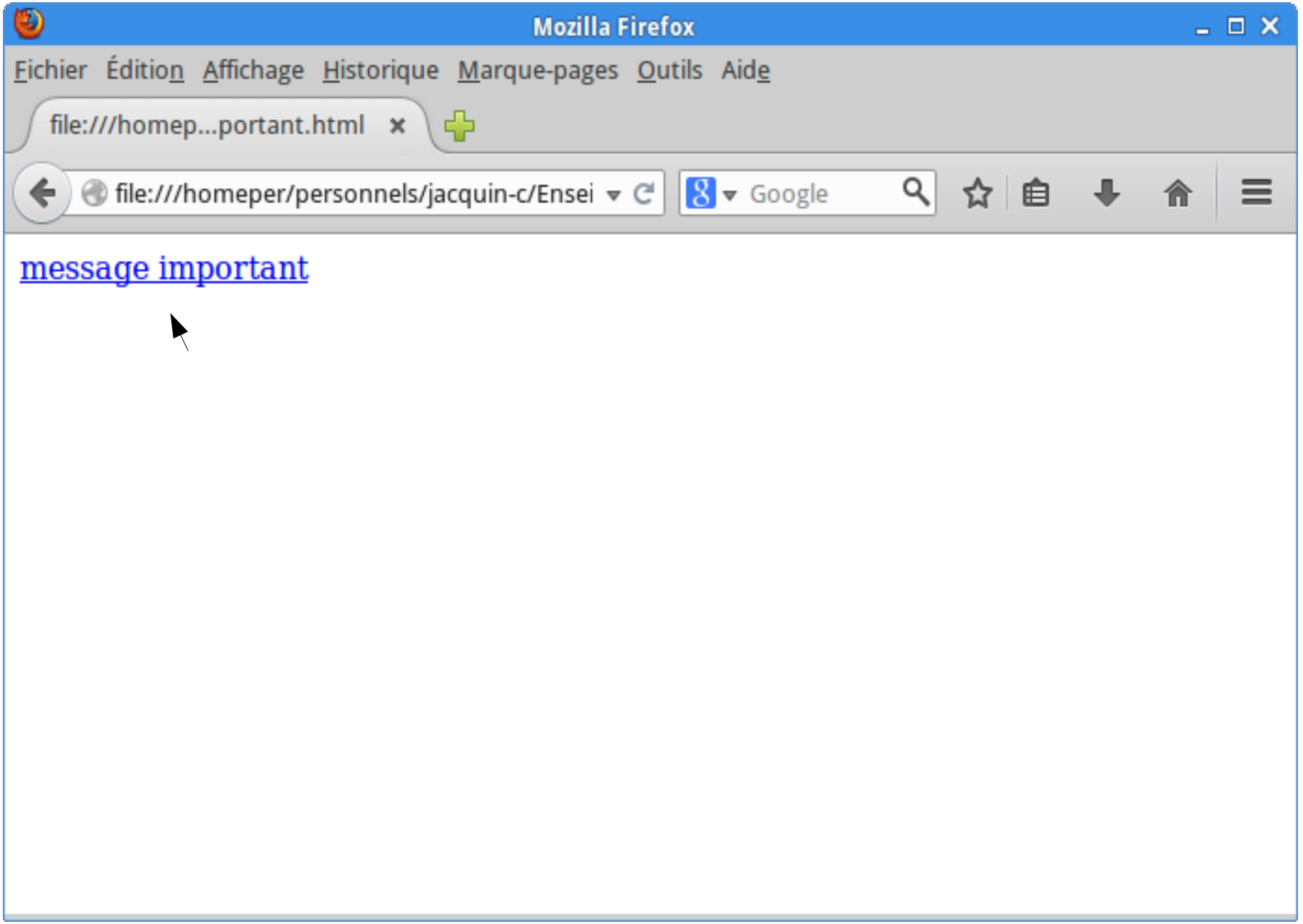
Limitations

- non séparation du code html/javascript
- pas d'accès à l'événement (disponible sous forme d'un objet)

Exemple d’un script avec gestion d’évènement (obsolète)

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript”>
function coucou(){
window.alert(“coucou !!!”);
}
</script>
</head>
<body>
<a href="#" onMouseOver=“coucou()”> message important</a>
</body>
</html>
```

https://www.w3schools.com/css/tryit.asp?filename=trycss_display


Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils Aide
file:///homep...portant.html
file:///homeper/personnels/jacquin-c/Ensei
Google
message important

Les événements DOM-0

Des événements peuvent être interceptés par l'utilisation d'une propriété d'un élément.

* passer avec la souris *

```
document.getElementById('exemple').onmouseover = changer ;

function changer(){
this.style.fontWeight="bold" ;//accès à l'objet courant
this.style.color="red" ;}
```

Avantages :

- séparation code html/javascript
- accès à un objet événement Event qui peut être passé en paramètre de la fonction associé à l'écouteur d'événement.

Inconvénients :

- ne permet pas d'ajouter plusieurs gestionnaires d'événement de même type à un même nœud.
- l'objet Event est géré différemment sur IE et les autres navigateurs

Exemple 1 de gestion d'événement via une propriété (DOM-0)

```
<!DOCTYPE html>
<html> <head>
<script type="text/javascript">
function init(){
document.getElementById('exemple').onmouseover = changer ;}

function changer(){
this.style.fontWeight="bold" ; this.style.color="red" ;}
</script> </head>
<body onload="init()">
<div id="exemple">passer avec la souris</div>
</body></html>
```

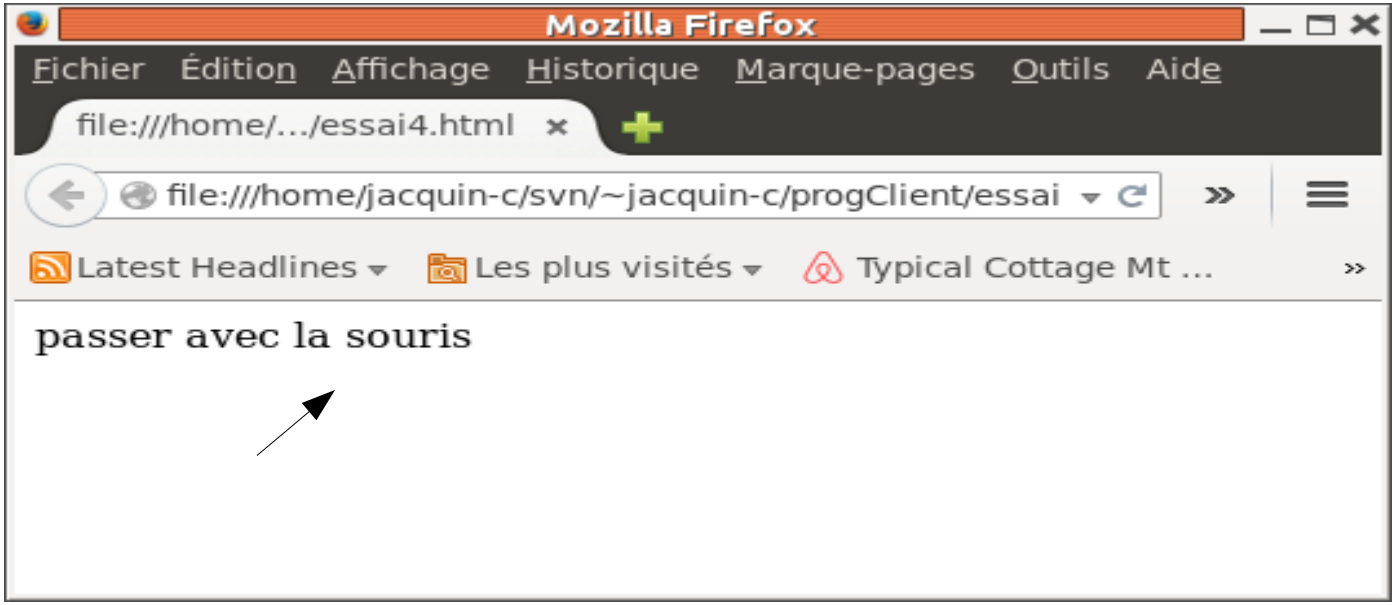
La fonction ***onload*** *est appelée quand le corps de la page a été chargé.* C'est l'endroit normal pour placer tous les traitements qui doivent se faire quand la page a été chargée.

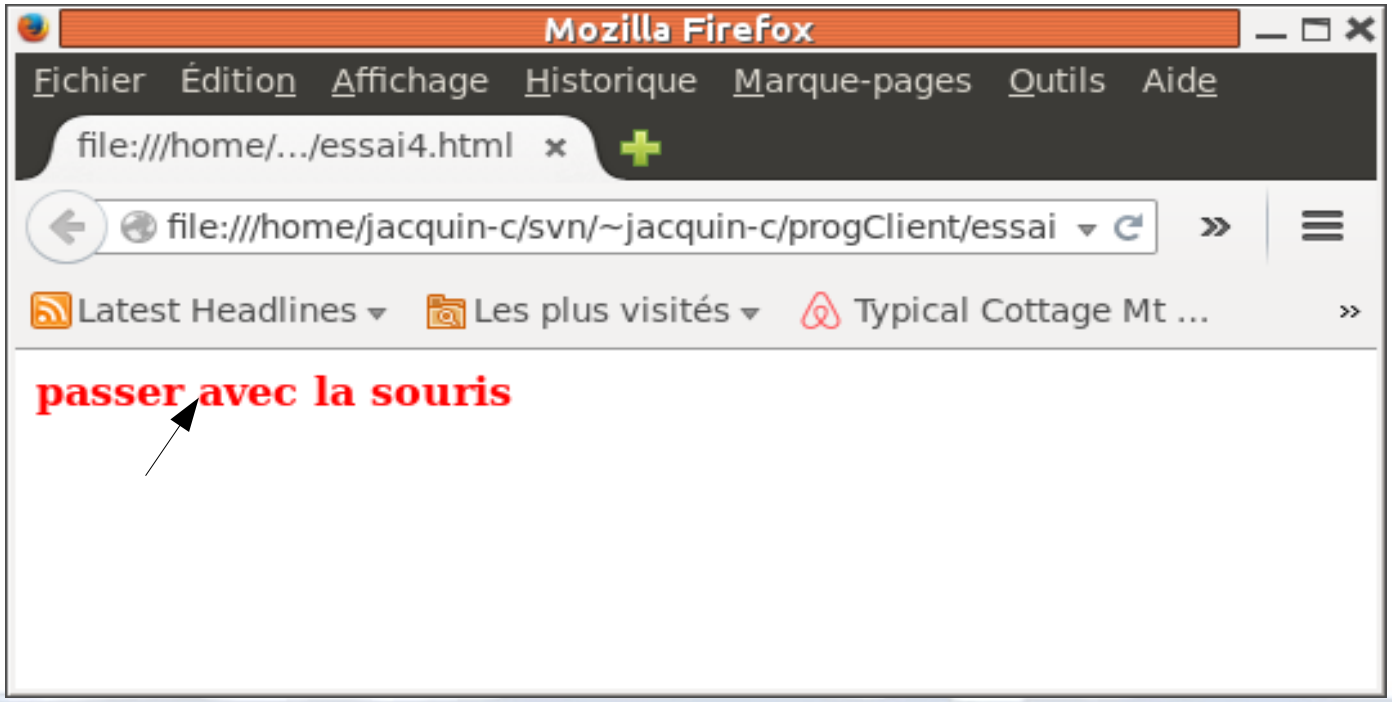
Exemple 2 de gestion d'événement via une propriété (DOM-0)

```
<!DOCTYPE html>
<html> <head>
<script type="text/javascript">
function init(){
document.getElementById('exemple').onmouseover = function(){
this.style.fontWeight="bold" ;
this.style.color="red" ;}
};

</script>
</head>

<body onload="init()">
<div id="exemple">passer avec la souris</div>
</body>
</html>
```

fonction anonyme

Exemple


Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils Aide
file:///home/.../essai4.html
file:///home/jacquin-c/svn/~jacquin-c/progClient/essai
Latest Headlines
Les plus visités
Typical Cottage Mt ...
passer avec la souris

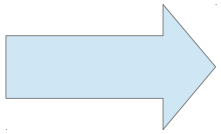

Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils Aide
file:///home/.../essai4.html
file:///home/jacquin-c/svn/~jacquin-c/progClient/essai
Latest Headlines
Les plus visités
Typical Cottage Mt ...
passer avec la souris

Exemple 2 d'utilisation de l'objet événement en DOM-0

```
<!DOCTYPE html>
<html> <head>
<script type="text/javascript">
function init(){
document.getElementById('envoyer').onsubmit = tester ;
}

function tester(event){
if (document.getElementById('nom').value== " "){
// évite que les données du formulaire soient envoyées au serveur
event.preventDefault() ;
}
}
</script></head>

<body onload="init()">
<form action= "script.php " method= "post ">
nom : <input id="nom" type="text">
<input id= "envoyer" type= "submit" value="envoyer">
</form>
</body></html>
```

Les événements DOM-2

Des événements peuvent être interceptés par des gestionnaires d'événement (spécification DOM 2)



ajout de gestionnaire d'événements aux éléments

* passer avec la souris *

```
var el=document.getElementById("exemple") ;
el.addEventListener("mouseover", changer, false) ;
el.addEventListener("mouseout", normal, false) ;
```

- le second paramètre est une fonction javascript
- Le 3ème paramètre de type booléen correspond à la phase dans laquelle l'évènement sera **traité ou propagé** (false : bouillonnement, true : capture)

Avantages et inconvénients des gestionnaires d'événement DOM-2

addEventListener est la manière d'enregistrer un écouteur telle que spécifiée dans le DOM du W3C.

element.addEventListener(event, function, useCapture)

Avantages:

- séparation code html/javascript
- accès à l’objet événement Event
- prise en compte de 2 phases pour la propagation des événements
- possibilité d'associer plusieurs gestionnaires pour un même événement
- Elle donne un contrôle plus fin sur la phase d'activation de l'écouteur (capture ou propagation)

Inconvénients :

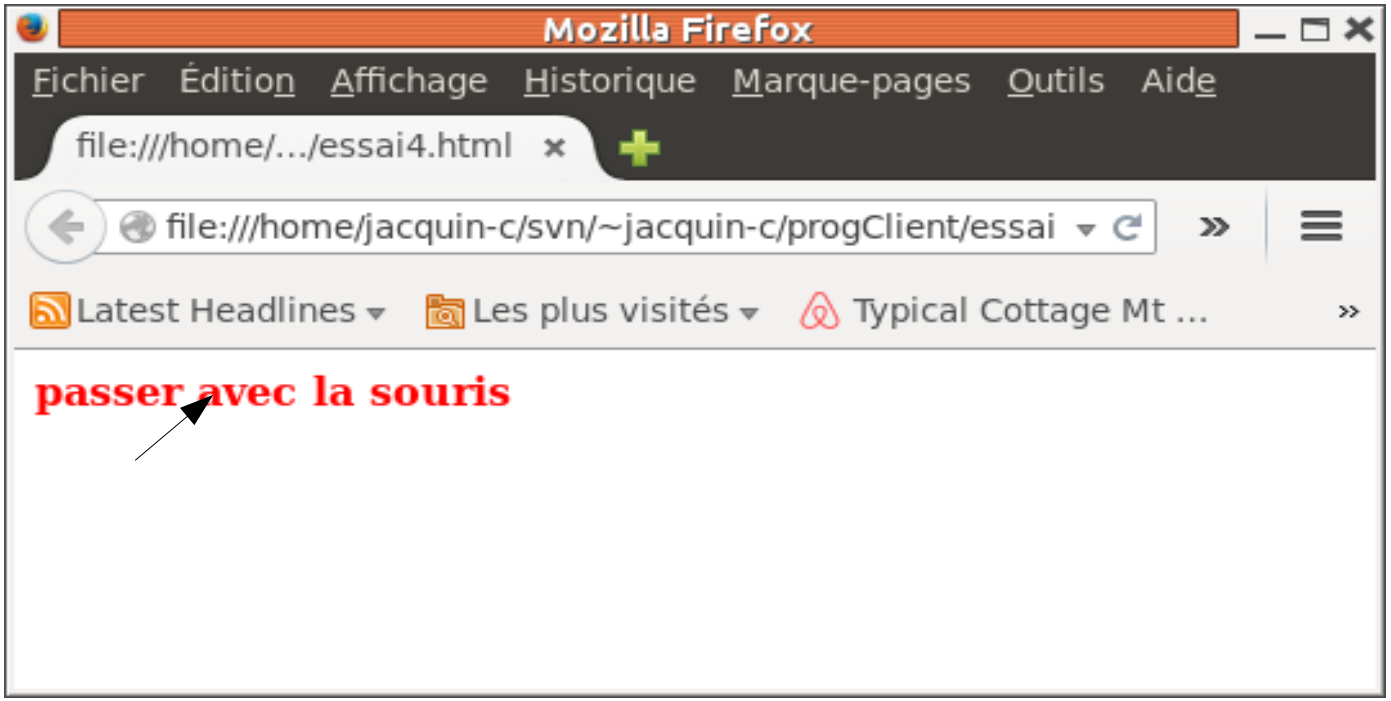
- méthode d’association différente sur IE et les autres navigateurs...

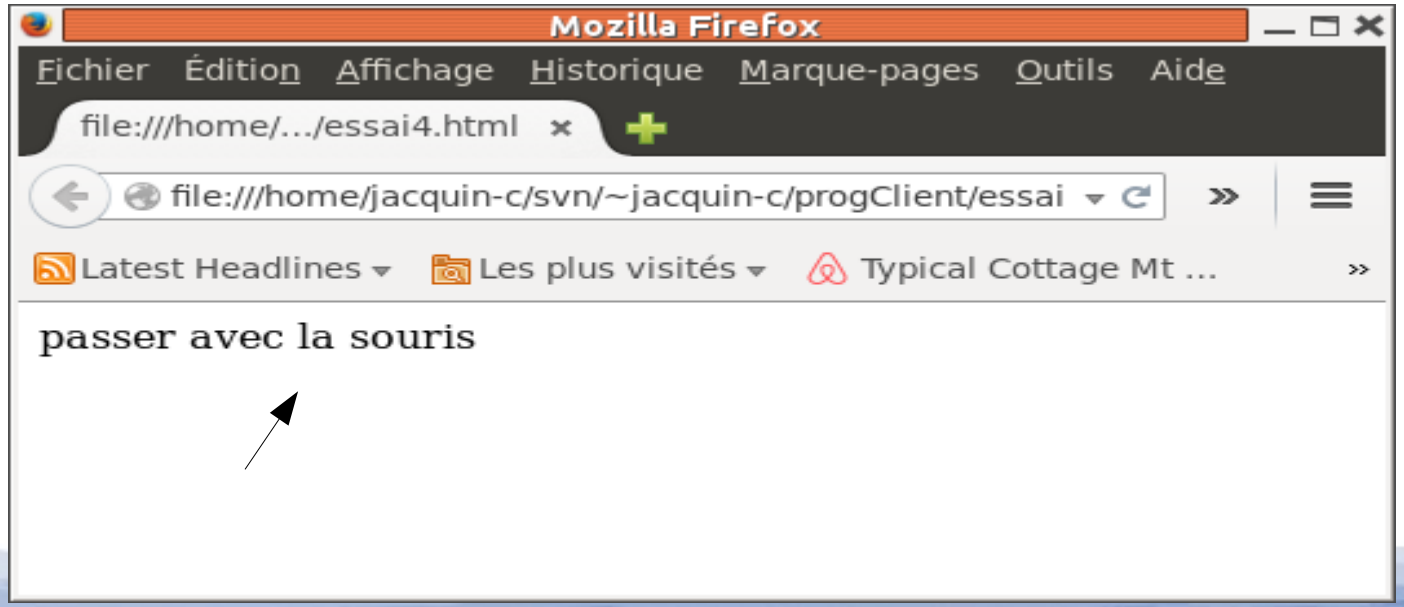
Exemple de gestion d'événement via un gestionnaire d'événement (DOM-2)

```
<!DOCTYPE html>
<html> <head>
<script type="text/javascript">
function init(){
var el=document.getElementById("exemple") ;
el.addEventListener("mouseover", changer, false) ;//false par défaut
el.addEventListener("mouseout", normal, false) ;
}

function changer(){
this.style.fontWeight="bold" ;
this.style.color="red" ;}

function normal(){
this.style.fontWeight="normal" ;
this.style.color="black" ;}
</script>
</head>

<body onload="init()">
<div id="exemple">passer avec la souris</div>
</body>
</html>
```

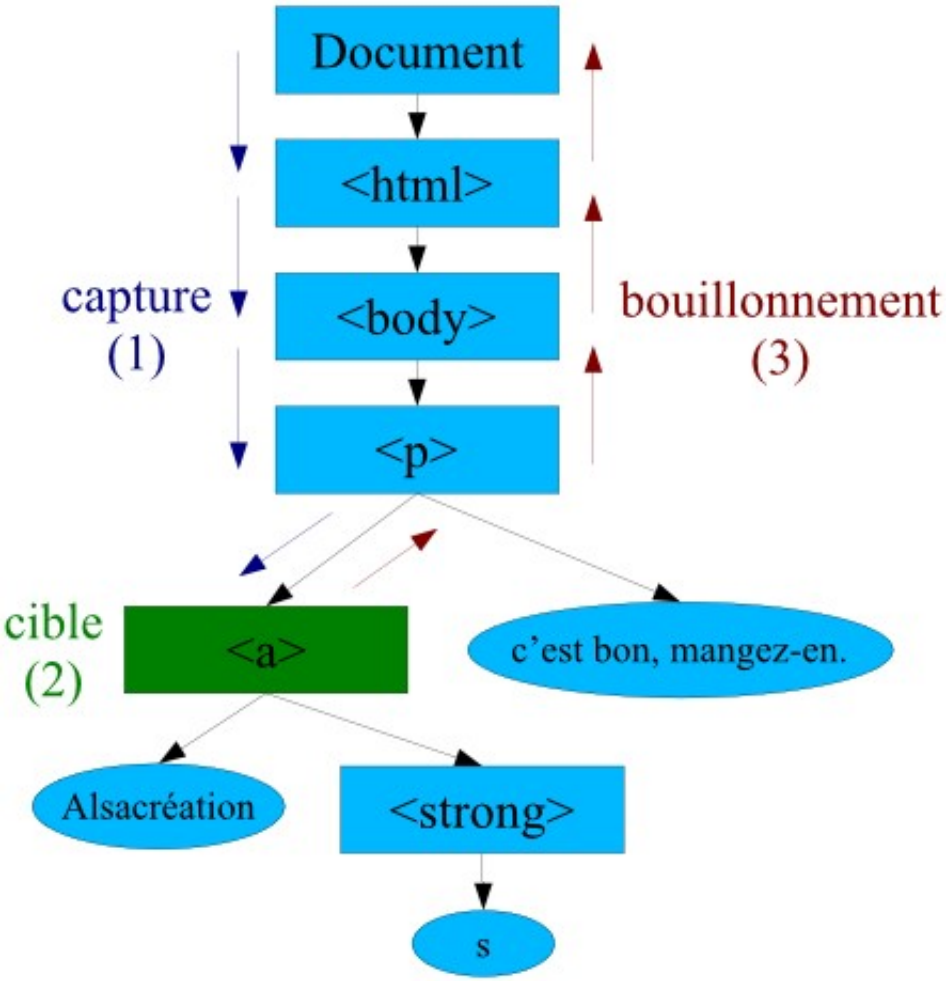
this utilisé dans le code des gestionnaires d'événements correspond à l'objet à partir duquel la méthode addEventListener() a été invoquée

Exemple


Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils Aide
file:///home/.../essai4.html
file:///home/jacquin-c/svn/~jacquin-c/progClient/essai
Latest Headlines
Les plus visités
Typical Cottage Mt ...
passer avec la souris


Mozilla Firefox
Fichier Édition Affichage Historique Marque-pages Outils Aide
file:///home/.../essai4.html
file:///home/jacquin-c/svn/~jacquin-c/progClient/essai
Latest Headlines
Les plus visités
Typical Cottage Mt ...
passer avec la souris

Gestionnaire d'évènement (DOM-2)




```
<html>
<head>
  <meta http-equiv="Content-Type"
    content="text/html; charset=UTF-8" />
  <title>Publicité</title>
</head>
<body>
  <p><a href="http://www.alsacreations.com/">
Alsacréation<strong>s</strong></a>
c'est bon, mangez-en.
  </p>
</body>
</html>
```

Phase de capture : l'événement se propage de la racine du document (incluse) à la cible (exclue).

L'événement atteint la cible.

Phase de bouillonnement (bubbling) : l'événement se propage dans le sens inverse : de la cible (exclue) à la racine du document (incluse).

Remarques :

- Certains événements comme load ne "bouillonnent" pas.
- On peut interrompre le flux de propagation
  event.stopPropagation() ;

BC

Les événements et les objets(1)

Événement	Objet(s) concerné(s)
abort	image
blur	button, checkbox, fileUpload, layer, password, radio, reset, select, submit, text, textarea, window
change	fileUpload, select, submit, text, textarea
click	button, document, checkbox, link, radio, reset, select, submit
dblclick	document, link
dragdrop	window
error	image, window
focus	button, checkbox, fileUpload, layer, password, radio, reset, select, submit, text, textarea, window
keydown	document, image, link, textarea
keypress	document, image, link, textarea
keyup	document, image, link, textarea

Les événements et les objets(2)

Événement	Objet(s) concerné(s)
load	image, layer, window
mousedown	button, document, link
mousemove	aucun spécifiquement
mouseout	layer, link
mouseover	area, layer, link
mouseup	button, document, link
move	window
reset	form
resize	window
select	text, textarea
submit	form
unload	window

Bibliographie

- Cours de première année de DUT informatique de l'IUT de Nantes, 2013-2014, J F Remm
- Cours de deuxième année de DUT informatique de l'IUT de Nantes, 2015-2016, C. Jacquin

- http://www.gchagnon.fr/cours/dhtml/

- http://www.w3schools.com/ pour une référence au langage

- Alsacreation, http://www.alsacreations.com/