



## Cours Web n°9

### Introduction à JavaScript

Pierre Senellart ([pierre@senellart.com](mailto:pierre@senellart.com))

Pierre Yger ([yger@unic.cnrs-gif.fr](mailto:yger@unic.cnrs-gif.fr))



**IFIPS** Institut  
de Formation  
d'Ingénieurs

UNIVERSITÉ **PARIS-SUD 11**

26 novembre et 7 décembre 2007

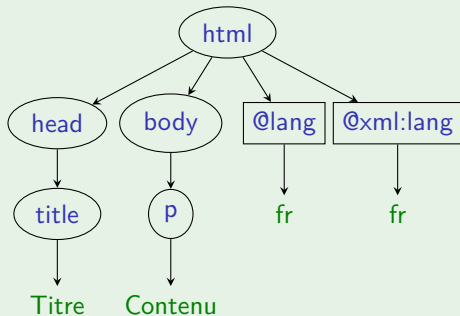
# Plan du cours

- 1 Introduction
- 2 L'objet Node
- 3 Fonctions utiles de JavaScript
- 4 Gestionnaire d'événements
- 5 Références
- 6 Application

- PHP, CGI... : permettent des comportements dynamiques **côté serveur**. Nécessitent un échange entre le navigateur et le serveur Web (soumission d'un formulaire, clic sur un lien) pour chaque comportement dynamique souhaité.
- JavaScript : permet des comportements dynamiques **côté client** : manipulation des fenêtres, changement dynamique du code HTML/CSS, interaction fine avec les formulaires...
- Permet la manipulation du DOM (**D**ocument **O**bject **M**odel), la représentation du document HTML comme un arbre, les balises étant les nœuds de l'arbre.
- « Dynamic HTML » (**DHTML**) : JavaScript + DOM + CSS
- **Alternatives** : VBScript (Internet Explorer uniquement), Java (plus complet, mais plus lourd, plug-in nécessaire), Flash (nécessite un logiciel propriétaire pour le développement, plug-in nécessaire).
- Rien à voir avec Java !

## Example

```
<html lang="fr" xml:lang="fr" xmlns="...">
  <head>
    <title>Titre</title>
  </head>
  <body>
    <p>Contenu</p>
  </body>
</html>
```



- Langage de Programmation
- Normalisé sous le nom d'**EcmaScript**
- Syntaxe très proche de PHP. Différences :

	PHP	JavaScript
Variables	<code>\$toto</code>	<code>toto</code>
Concaténation	<code>\$chaine1.\$chaine2</code>	<code>chaine1+chaine2</code>
Interpolation	oui ( <code>"\$toto"</code> )	non
Tableaux	<code>\$t=array(1,'two')</code>	<code>t=new Array(1,'two')</code>
Taille tableau	<code>count(\$t)</code>	<code>t.length</code>

- Liaison d'un script JavaScript avec un document XHTML

- ▶ `toto.js` contenant des fonctions JavaScript (**function**)
- ▶ Dans le `<head>` du XHTML :

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

- ▶ Gestionnaires d'évènement comme attributs des balises XHTML (cf plus loin).

- JavaScript basé sur le **modèle objet**.
- Variables : objets complexes, ayant des propriétés (**membres**) et des fonctionnalités (**fonctions membres, méthodes**).
- En JavaScript, on accède au membre blah de l'objet toto avec **toto.blah**, et on utilise la méthode bouh de l'objet toto avec **toto.bouh(arguments)**.

## Exemple

Par exemple, un objet **voiture** pourrait avoir une propriété **couleur** et des fonctionnalités **tourneGauche()**, **tourneDroite()** ou **avance(distance)**.

On pourrait alors marquer :

```
voiture.couleur="bleu";  
voiture.avance(100);  
voiture.tourneGauche();
```

- En pratique, les objets JavaScript qu'on utilisera représenteront le document XHTML, les nœuds du documents, la fenêtre...



## Plan du cours

- 1 Introduction
- 2 L'objet Node**
- 3 Fonctions utiles de JavaScript
- 4 Gestionnaire d'événements
- 5 Références
- 6 Application

- L'objet **node** est l'objet central du modèle DOM (Document Object Model).
- Chaque élément, chaque attribut et chaque donnée en caractères représentent des nœuds distincts. Ces nœuds forment une arborescence.
- L'objet node dispose de propriétés et de méthodes pour accéder aux différents nœuds, peu importe s'ils sont placés très bas dans l'arborescence.



- `node = document.getElementById("titi")` accède à un élément XHTML qui possède un attribut `id` valant `titi`.
- `valeur = node.nodeValue` sauvegarde la valeur ou le contenu d'un nœud :
  - ▶ pour les nœuds texte, c'est le texte,
  - ▶ pour les nœuds attribut la valeur affectée à l'attribut

- `node.className="nouvelle_classe"` pour changer le nom de la classe CSS à laquelle appartient le nœud.
- `node.style.borderStyle="valeur"` pour changer le style de bordure d'un nœud.
- `node.style.visibility="valeur"` pour changer la visibilité d'un nœud
- `node.style.display="valeur"` pour changer la propriété CSS `display` d'un nœud.

## Règle générale

On peut changer de cette façon n'importe quelle propriété CSS, d'un nœud. Le nom de la propriété en JavaScript est identique au nom CSS, sauf que les traits d'unions sont remplacés par une majuscule sur la lettre suivante. Les valeurs des propriétés en JavaScript sont identiques aux valeurs CSS (mais doivent être mis entre guillemets).

## Exemple

JavaScript :

```
function Test() {  
    document.getElementById("paragraphe").style.color = "blue";  
}
```

XHTML :

```
<p id="paragraphe" style="color: red;">un texte</p>  
<a href="" onclick="Test()">Test</a>
```

**Explication :** L'exemple contient un paragraphe avec le nom id *paragraphe* et un lien qui si on le clique appelle la fonction *Test()*. Cette fonction change la propriété CSS *color* du paragraphe, de telle sorte que le paragraphe perde sa couleur rouge et devienne bleu.

`node.parentNode` nœud parent d'un nœud.

`node.childNodes` tableau de tous les nœuds enfant disponibles d'un nœud.

Quand un nœud n'a pas de nœud enfant, *childNodes* a la valeur **null**.

`node.nodeName` le nom d'un nœud.

`node.appendChild(enfant)` ajoute un nœud créé auparavant à la structure de nœuds existante, et cela de façon à ce qu'il soit inséré comme dernier nœud enfant.

`node.removeChild(enfant)` efface un nœud enfant d'un élément.

`node.cloneNode()` construit une copie à l'identique d'un nœud, avec ou sans la structure de sous-nœuds qui en fait partie.

`node.setAttribute("name","value")` fixe à nouveau une valeur d'attribut dans un élément. Si l'attribut existe déjà, son ancienne valeur sera remplacée par la nouvelle. Si ce n'est pas le cas, il est créé et la nouvelle valeur lui est affectée.

`node.getAttribute("name")` recherche la valeur d'un attribut déterminé dans un élément.

# Plan du cours

- 1 Introduction
- 2 L'objet Node
- 3 Fonctions utiles de JavaScript**
- 4 Gestionnaire d'événements
- 5 Références
- 6 Application

`alert("m")` crée une fenêtre de dialogue dans laquelle le message *m* est affiché

`back()` permet de retourner à la dernière page visitée

`close("nom_de_la_fenetre")` détruit une fenêtre du client

`confirm("m")` crée une fenêtre de dialogue pour confirmer une action : elle permet le choix entre *OK* et *Annuler*

`open("URL","nom_de_la_fenetre","options_de_la_fenetre")` crée une nouvelle fenêtre client

`prompt("m","par défaut")` crée une fenêtre de dialogue permettant la saisie et dans laquelle le message *m* est affiché

## Exemple

JavaScript :

```
function OpenWindow() {  
    Info = open("fichier.htm", "secondefenetre");  
}
```

XHTML :

```
<body onload="OpenWindow()">  
    <p><a href="" onclick="Info.close()">Fermer la fenêtre</a></p>  
</body>
```

**Explication :** L'exemple ouvre à la lecture du fichier une deuxième fenêtre du nom de *Info*. Dans le fichier est défini un bouton. Si l'utilisateur clique sur le lien, la deuxième fenêtre est fermée.



# Plan du cours



- 1 Introduction
- 2 L'objet Node
- 3 Fonctions utiles de JavaScript
- 4 Gestionnaire d'événements**
- 5 Références
- 6 Application

- Les événements sont des actions de l'utilisateur, qui vont pouvoir donner lieu à une **interactivité**.
- L'événement correspond à un clic de souris, une sélection d'un champ, une touche pressée au clavier. . .
- Il est possible d'associer des fonctions à des événements.
- La syntaxe est la suivante :

```
onevenement="Action_Javascript_ou_Fonction()"
```

- onblur** se produit lorsque l'élément ( `<input>` , `<textarea>` , `<select>` ) perd le focus, c'est-à-dire que l'utilisateur clique hors de cet élément, celui-ci n'est alors plus sélectionné comme étant l'élément actif.
- onchange** se produit lorsque l'utilisateur modifie le contenu d'un champ de données ( `<input>` , `<textarea>` , `<select>` ).
- onclick** se produit lorsque l'utilisateur clique sur l'élément ( `<a>` , `<input type="radio">` , `<input type="checkbox">` ) associé à l'événement. Si le gestionnaire d'évènement **renvoie** false, le clic n'est pas effectué.
- onfocus** se produit lorsque l'utilisateur donne le focus à un élément ( `<input>` , `<textarea>` , `<select>` ), c'est-à-dire que cet élément est sélectionné comme étant l'élément actif.

- onload** se produit lorsque le navigateur de l'utilisateur charge la page en cours ( `<body>` ).
- onreset** se produit lorsque l'utilisateur efface les données d'un formulaire ( `<form>` ) à l'aide du bouton Reset.
- onsubmit** se produit lorsque l'utilisateur clique sur le bouton de soumission d'un formulaire ( `<form>` ). Si le gestionnaire d'évènement **renvoie** `false`, le formulaire n'est pas soumis.

## Exemple

```
<form name="Test" action="">
  <input type="text"
    onfocus="this.value='Entrez votre nom ici'" />
  <input type="text"
    onfocus="this.value='Entrez votre adresse ici'" />
  <input type="text"
    onfocus="this.value='Entrez votre âge ici'" />
</form>
```

**Explication :** Dans l'exemple un formulaire est défini et contient trois champs de saisie. Étant donné que les champs ne portent pas d'inscription, l'utilisateur ne sait pas ce qu'il doit entrer dans les différents champs. Pourtant, s'il déplace le curseur par curiosité dans l'un des champs de saisie, le gestionnaire d'événement `onFocus=` du champ concerné deviendra actif. Alors s'inscrira dans le champ concerné une invite de ce qu'il faut y mettre.

- ondblclick** se produit lorsque l'utilisateur double-clique sur l'élément associé à l'événement (un lien hypertexte ou un élément de formulaire).
- onmouseover** se produit lorsque l'utilisateur positionne le curseur de la souris au-dessus d'un élément.
- onresize** se produit lorsque l'utilisateur redimensionne la fenêtre du navigateur.
- onselect** se produit lorsque l'utilisateur sélectionne un texte (ou une partie d'un texte) dans un champ de type "text" ou "textarea"

# Plan du cours

- 1 Introduction
- 2 L'objet Node
- 3 Fonctions utiles de JavaScript
- 4 Gestionnaire d'événements
- 5 Références**
- 6 Application

- Standards
  - ▶ Langage ECMAScript, <http://www.ecma-international.org/publications/standards/Ecma-262.htm>
  - ▶ Spécifications DOM, <http://www.w3.org/DOM/DOMTR>
- Pages Web
  - ▶ Comparatif syntaxe PHP/JavaScript, <http://phplens.com/phpeverywhere/node/view/30>
  - ▶ Tutorial JavaScript, <http://fr.selfhtml.org/javascript/>
- *JavaScript, la Référence*, O'Reilly



# Plan du cours

- 1 Introduction
- 2 L'objet Node
- 3 Fonctions utiles de JavaScript
- 4 Gestionnaire d'événements
- 5 Références
- 6 Application**

Reproduire les exemples de la page Web du cours :

- Vérification qu'un champ texte est bien rempli à la soumission d'un formulaire.
- Champs de formulaires apparaissant ou disparaissant suivant qu'un bouton radio est sélectionné ou non.
- Mise à jour dynamique de la couleur d'une boîte en fonctions de champs textes indiquant les pourcentages de Rouge, Vert, Bleu.