

# Ajax

## Asynchronous Javascript And XML

Denis Szalkowski Formateur Consultant  
<http://www.dsfc.net/>

# I. Présentation

## A. Définition

Cette technologie s'insère dans un ensemble beaucoup plus large dénommé Web 2.0. Ajax signifie Asynchronous Javascript And Xml. Les sites utilisant cette technologie sont encore peu nombreux. On peut citer toutefois :

<a href="#">Google Suggest</a>	Outil de recherche sur le Web
<a href="#">Google Maps</a>	Outil de cartographie
<a href="#">Gmail</a>	Outil de messagerie Google
<a href="#">Google Groups</a>	Outil de recherche sur les newsgroups
<a href="#">Orkut</a>	Réseau social

## B. Technologies employées

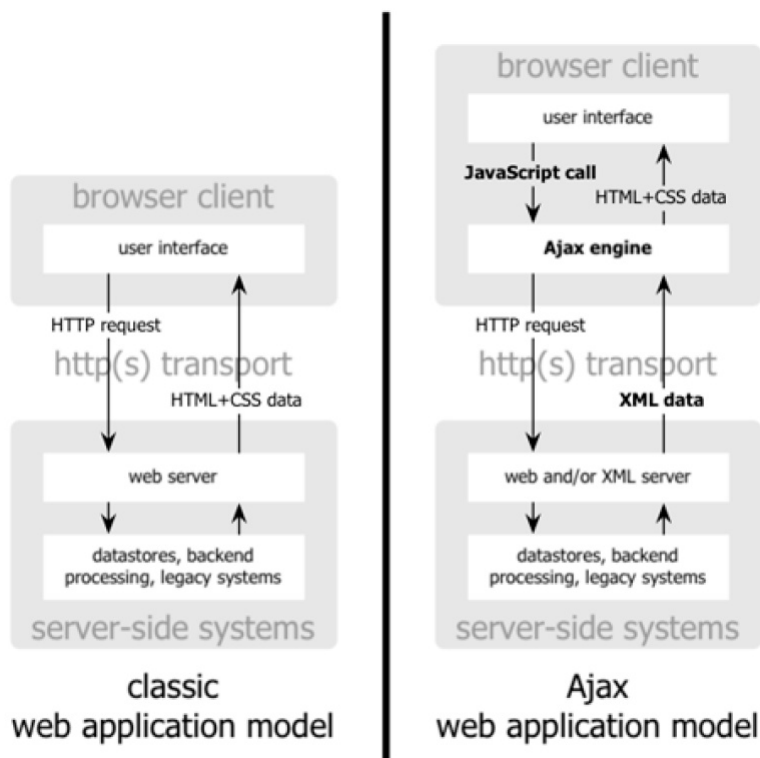
La présentation est réalisée avec Xhtml et Css.

L'affichage dynamique et l'interactivité utilisent le DOM (Document Object Model).

L'échange de données et leur manipulation s'appuient sur XML et XSLT.

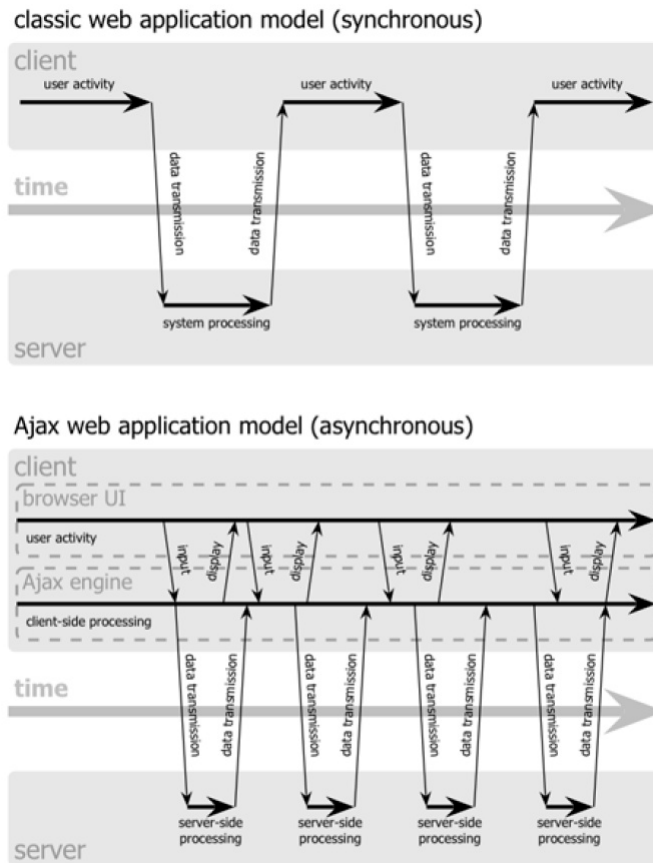
Les requêtes asynchrones utilisent XMLHttpRequest.

Javascript assure le lien entre toutes ces technologies.



*Illustration Séquence Illustration: Schéma issu du site AdaptivePath*

### C. Illustration du mécanisme asynchrone



*Illustration Séquence Illustration: Schéma issu du site AdaptivePath*

---

## II. Le XHTML

---

### A. Différences avec le Html 4.0

Les éléments,attributs doivent être saisis en minuscules, y compris dans les feuilles Css.

Tous les éléments doivent être emboîtés.

Les noms d'éléments et d'attributs doivent être emboîtés.

Pour les éléments non vides, une balise de fin est obligatoire.

Les valeurs d'attributs doivent toujours être mises entre guillemets.

Tout attribut doit posséder une valeur.

Les éléments vides doivent avoir une balise de fin ou se terminer par />.

Les espacements de début et de fin doivent être retirer des valeurs d'attributs

Utilisez les attributs lang ou xml:lang pour préciser la langue d'un attribut..

### B. Entête de la page Xhtml

```
<!doctype html public "-//w3c//dtd/xhtml1.0 transitional//en" "dtd/xhtml1-  
transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">
```

### C. Valideur Xhtml

[W3C](#)

[Tidy](#)

### D. Bibliographie

[La-Grange.Net](#)

[W3C](#)

## III. Javascript

### A. Écriture en mode objet

```
<html>
<head>
</head>
<body>
<script type="text/javascript">
function classPersonne(strNom,strPrenom,strDateNaissance)
{
    this.propNom=strNom;
    this.propPrenom=strPrenom;
    this.propDateNaissance=new Date(strDateNaissance);
    this.propAge=function()
    {
        dateDateDuJour=new Date();
        return dateDateDuJour.getYear()-this.propDateNaissance.getYear();
    }
}
function classSalarie(strNom,strPrenom,strDateNaissance,strDateEmbauche)
{
    this.base = classPersonne;
    this.base(strNom,strPrenom,strDateNaissance);
    this.propDateEmbauche=new Date(strDateEmbauche);
    this.propAnciennete=function()
    {
        dateDateDuJour=new Date();
        return dateDateDuJour.getYear()-this.propDateEmbauche.getYear();
    }
}
oSalarie=new classSalarie("Szalkowski","Denis","21/1/1964","1/2/1989");
document.write(oSalarie.propNom," ",oSalarie.propPrenom," ",oSalarie.propAge(),"
",oSalarie.propAnciennete());
</script>
</body>
</html>
```

### B. Gestion événementielle

```
<html id="idDoc">
<head>
<script type="text/javascript">
var docHtml=document.getElementById('idDoc');
function processMove(event)
{
    if (!event)
    {
        event = window.event;// pour IE
    }
    //var e=event || window.event;
    var X=parseInt(event.clientX);
    var Y=parseInt(event.clientY);
    window.defaultStatus=X + " x " + Y;
}
docHtml.onclick=processMove;
</script>
</head>
<body>
</body>
</html>
```

### C. Dom (Document Object Model)

#### 1. Énumération de la structure du document : dom1.html

```
<html>
<head>
<script type="text/javascript">
```

```

function modifDiv()
{
  var docElement = document.documentElement;
  var childNode = docElement.childNodes;
  var bodyElement;
  for (i = 0; i < childNode.length; i++)
  {
    if (childNode[i].nodeName == 'BODY')
    {
      bodyElement = childNode[i];
      break;
    }
  }
  childNode = bodyElement.childNodes;
  var divElement;
  for (i = 0; i < childNode.length; i++)
  {
    if (childNode[i].nodeName == 'DIV')
    {
      divElement = childNode[i];
      break;
    }
  }
  var strMess=divElement.childNodes[0].nodeValue;
  if(strMess=='Bonjour le monde')
  {
    strMess='Au revoir le monde';
  }
  else
  {
    strMess='Bonjour le monde';
  }

  //divElement.replaceChild(document.createTextNode(strMess),divElement.childNodes[0]);
  divElement.childNodes[0].nodeValue=strMess;
}
</script>
</head>
<body>
  <div>Bonjour le monde</div>
  <button onclick="modifDiv()">Voir</button>
</body>
</html>

```

## 2. Utilisation de getElementByTagName : dom2.html

```

<html>
  <head>
    <script type="text/javascript">
      function modifDiv()
      {
        var divElements = document.getElementsByTagName("DIV");
        var divElement = divElements[0];
        var strMess=divElement.childNodes[0].nodeValue;
        if(strMess=='Bonjour le monde')
        {
          strMess='Au revoir le monde';
        }
        else
        {
          strMess='Bonjour le monde';
        }
        divElement.childNodes[0].nodeValue=strMess;
      }
    </script>
  </head>
  <body>
    <div>Bonjour le monde</div>
    <button onclick="modifDiv()">Voir</button>
  </body>
</html>

```

### 3. Utilisation de getElementById et setAttribute : dom3.html

```

<html>
<head>
<script type="text/javascript">
function modifDiv()
{
var divElement = document.getElementById("idMess");
var strMess=divElement.innerHTML;
var strCouleur;
if(strMess=='Bonjour le Monde')
{
strMess='Au revoir le Monde';
strCouleur='blue';
}
else
{
strMess='Bonjour le Monde';
strCouleur='red';
}
divElement.setAttribute('style','color:'+strCouleur);
divElement.innerHTML=strMess;
}
</script>
</head>
<body>
<div id="idMess" style="color: red;">Bonjour le Monde</div>
<button onclick="modifDiv()">Voir</button>
</body>
</html>

```

## D. XMLHttpRequest

### 1. Méthodes et propriétés

Propriétés	Description
.onreadystatechange	Gestionnaire d'événements pour les changements d'état.
.readyState	Statut de l'objet.
.responseText	Réponse sous forme de chaîne de caractères.
.responseXML	Réponse sous forme d'objet DOM.
.status	Code numérique de réponse du serveur HTTP : 200, 403, 404, 500
.statusText	Message accompagnant le code de réponse.

Méthode	Description
.abort()	Abandonne la requête.
.getAllResponseHeaders()	Renvoie l'ensemble de l'entête de la réponse sous forme de chaîne de caractères.
.getResponseHeader(champEntete)	Renvoie la valeur d'un champ d'entête HTTP.
.open(method, URL[, asyncFlag[, userName[, password]])	Prépare une requête en indiquant la méthode, l'URL, la drapeau de synchronisation, le nom d'utilisateur et le mot de passe.
.send(contenu)	Effectue la requête, éventuellement en envoyant les données.
.setRequestHeader(champ, valeur)	Assigne une valeur à un champ d'entête HTTP qui sera envoyé lors de la requête.

## 2. Propriété readyState

Code	Signification
0	non initialisé
1	ouverture. La méthode open() a été appelée avec succès
2	envoyé. La méthode send() a été appelée avec succès
3	réception en cours. Des données sont en train d'être transférées, mais le transfert n'est pas terminé
4	terminé. Les données sont chargées

## 3. Côté serveur : le script php now.php

```
<?php
echo date("d/m/Y H:i:s");
?>
```

## 4. Spécificités Firefox

A partir de l'url about:config, passez la valeur du paramètre signed.applets.codebase\_principal\_support à true.

## 5. Côté client : la page html xhr1.html (mode procédural)

```
<html>
<head>
<script type="text/javascript">
var xhr;
function modifDiv()
{
try
{
xhr = new ActiveXObject("Msxml2.XMLHTTP");
}
catch (e)
{
try
{
xhr = new ActiveXObject("Microsoft.XMLHTTP");
}
catch (E)
{
xhr = false;
}
}
if (!xhr && typeof XMLHttpRequest != 'undefined')
{
xhr = new XMLHttpRequest();
}
xhr.open("GET", "now.php");
xhr.onreadystatechange=function()
{
if (xhr.readyState != 4)
{
return;
}
document.getElementById("idMess").innerHTML = xhr.responseText;
}
xhr.send(null);
}
</script>
</head>
<body>
<div id="idMess"></div>
<button onclick="modifDiv()">Voir</button>
</body>
</html>
```

## 6. Parser un document Xml

```

<html>
<head>
<script type="text/javascript">
function getXhr()
{
if(window.XMLHttpRequest)
{
return new XMLHttpRequest();
}
else
{
if(window.ActiveXObject)
{
return new ActiveXObject("Microsoft.XMLHTTP");
}
}
}
function contenuXhr(strUrl)
{
var xmlDoc;var strContenu="";
var xhr=getXhr();
xhr.onreadystatechange=function()
{
if (xhr.readyState == 4)
{
xmlDoc=xhr.responseXML;
/*
var elChannel=xmlDoc.getElementsByTagName('item')[0];
for(i=0;i<elChannel.childNodes.length;i++)
{
var elItem=xmlDoc.getElementsByTagName('item')[i];
}
*/
var elTitle=xmlDoc.getElementsByTagName('title')[0].firstChild.nodeValue;
var elLink=xmlDoc.getElementsByTagName('link')[0].firstChild.nodeValue;
var
elDesc=xmlDoc.getElementsByTagName('description')[0].firstChild.nodeValue;

strContenu+="<h1>"+elTitle+"</h1>"+"<h1>"+elLink+"</h1>"+"<h1>"+elDesc+"</h1>";
var elItem=xmlDoc.getElementsByTagName('item')[0];
for(i=0;i<elItem.childNodes.length;i++)
{
var
elItemTitle=elItem.getElementsByTagName('title')[0].firstChild.nodeValue;
var
elItemLink=elItem.getElementsByTagName('link')[0].firstChild.nodeValue;
var
elItemDesc=elItem.getElementsByTagName('description')[0].firstChild.nodeValue;

strContenu+="<h2>"+elItemTitle+"</h2>"+"<h2>"+elItemLink+"</h2>"+"<h2>"+elItemDescription
+"</h2>";
}
var
elItem=xmlDoc.getElementsByTagName('description')[0].firstChild.nodeValue;
document.getElementById("idMess").innerHTML=strContenu;
}
}
xhr.open("GET",strUrl,true);
xhr.send(null);
}
</script>
</head>
<body>
<div id="idMess"></div>
<button onclick="contenuXhr('rss.xml');">Voir</button>
</body>
</html>

```

## 7. Framework Javascript Ajax

<a href="#">Dojo</a>	Gratuit	Javascript
<a href="#">ActiveWidgets</a>	Gratuit / Payant	?
<a href="#">SmartClient</a>	Gratuit	.Net
<a href="#">BackBase</a>	Gratuit/Payant	?
<a href="#">TinyAjax</a>	Gratuit	Php
<a href="#">AjaxAc</a>	Gratuit	Php
<a href="#">Xajax</a>	Gratuit	Php
<a href="#">PEAR::HTML::Ajax</a>	Gratuit	Php
<a href="#">Bindows</a>	Gratuit	Javascript
<a href="#">Rico</a>	Gratuit	Javascript
<a href="#">Qooxdoo</a>	Gratuit	Javascript
<a href="#">Prototype</a>	Gratuit	Javascript
<a href="#">Script.Aculo.Us</a>	Gratuit	Javascript
<a href="#">DWR</a>	Gratuit	Java
<a href="#">Ajax.Net</a>	Gratuit/Payant	.Net
<a href="#">Sajax</a>	Gratuit	Perl, Php, Python
<a href="#">Json-Rpc</a>	Gratuit	Java

## IV. Php

### A. En-tête Php

```
header("Content-type: application/xml");
echo "<?xml version=\"1.0\" encoding=\"UTF-8\" ?>\n";
```

### B. Créer un flux Rss

```
<?php
class RSS
{
    function RSS($CHANNEL=null,$ITEMS=null)
    {
        echo '<?xml version="1.0" encoding="ISO-8859-1"?>'. "\n";
        echo '<rss version="0.91">'. "\n";
        echo '<channel>'. "\n";
        echo '<title>'. $CHANNEL['title']. '</title>'. "\n";
        echo '<link>'. htmlspecialchars($CHANNEL['link'], ENT_QUOTES). '</link>'. "\n";
        echo '<description>'. $CHANNEL['description']. '</description>'. "\n";
        echo '<language>'. $CHANNEL['language']. '</language>'. "\n";
        echo '<copyright>'. $CHANNEL['copyright']. '</copyright>'. "\n";
        echo '<pubDate>'. date('D, j M Y G:i:s'). ' GMT'. '</pubDate>'. "\n";
        foreach($ITEMS as $ITEM)
        {
            echo '<item>'. "\n";
            echo '<title>'. stripslashes($ITEM['title']). '</title>'. "\n";
            echo '<link>'. htmlspecialchars($ITEM['link'], ENT_QUOTES). '</link>'. "\n";
            echo '<category>'. $ITEM['category']. '</category>'. "\n";
            echo '<description>'. $ITEM['description']. '</description>'. "\n";
            echo '<pubDate>'. date('D, d M Y G:i:s', $ITEM['pubDate']). ' GMT'. '</pubDate>'. "\n";
            echo '</item>'. "\n";
        }
        echo '</channel>'. "\n";
        echo '</rss>'. "\n";
    }
}
?>
```

### C. SimpleXML (Php 5)

```
<?php
$strUrl="http://www.zdnet.fr/feeds/rss/actualites/?l=5";
$xmlDoc = simplexml_load_file($strUrl);
echo "<h1>". $xmlDoc->channel->title. "</h1>";
foreach($xmlDoc->channel->item as $elItem)
{
    $a = utf8_decode($elItem->title);
    echo "$a<br/>";
}
?>
```

## V. Annexe : le Dom

Attr	représente un attribut, défini dans la Définition de Type de Document, d'un objet Element.	name, ownerElement, specified, value attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
CDATASection	représente une section de données textuelles (Character DATA Section).	data, length, attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendData, deleteData, insertData, replaceData, substringData, appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild, splitText
CharacterData	est une extension de l'interface Node qui permet d'accéder aux données textuelles dans le modèle d'objet.	data, length attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendData, deleteData, insertData, replaceData, substringData, appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
Comment	représente un commentaire dans un document XML ou HTML : <!-- Commentaire -->.	data, length, attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendData, deleteData, insertData, replaceData, substringData, appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
Document	représente la totalité d'un document XML ou HTML. En fait il est la racine (root) de l'arborescence d'un document.	doctype, documentElement, implementation attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	createAttribute, createAttributeNS, createCDATASection, createComment, createDocumentFragment, createElement, createElementNS, createEntityReference, createProcessingInstruction, createTextNode, getElementById, getElementsByTagName, getElementsByTagNameNS, importNode, appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
DocumentFragment	représente une partie de l'arborescence d'un document. Ce fragment pouvant ne pas être bien formé, est utilisé généralement pour des opérations d'insertion.	attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
DocumentType	représente la déclaration de type de document indiqué par	entities, internalSubset, name, notations, publicId, systemId attributes, childNodes, firstChild,	appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported,

	la balise <DOCTYPE>.	lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	normalize, removeChild, replaceChild
DOMImplementation	fournit des méthodes qui sont indépendantes de n'importe quelles instances particulières du Modèle d'Objet du Document	Aucune	createDocument, createDocumentType, hasFeature
Element	représente tous les noeuds communs, à l'exception des noeuds textuelles, dans les documents XML.	tagName attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	getAttribute, getAttributeNS, getAttributeNode, getAttributeNodeNS, getElementsByTagName, getElementsByTagNameNS, hasAttribute, hasAttributeNS, removeAttribute, removeAttributeNS, removeAttributeNode, setAttribute, setAttributeNS, setAttributeNode, setAttributeNodeNS appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
Entity	représente une entité analysée ou non-analysée dans un document XML.	notationName, publicId, systemId attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
EntityReference	contient le nom de l'entité <!ENTITY Nom SYSTEM Valeur>.	attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
NamedNodeMap	représente des collections de noeuds qui peuvent être accédées par un nom.	length	getNamedItem, getNamedItemNS, item, removeNamedItem, removeNamedItemNS, setNamedItem, setNamedItemNS
Node	représente un noeud unique dans l'arborescence d'un document XML.	attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
NodeList	représente une collection de noeuds ordonnés.	length	item
Notation	représente une notation <!NOTATION> déclarée dans la DTD.	publicId, systemId attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument,	appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild

		parentNode, prefix, previousSibling	
ProcessingInstruction	représente une instruction de traitement <?Nom_Instruction Contenu?>.	data, target attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild
Text	représente le contenu textuel d'un attribut ou d'un élément.	data, length, attributes, childNodes, firstChild, lastChild, localName, namespaceURI, nextSibling, nodeName, nodeType, nodeValue, ownerDocument, parentNode, prefix, previousSibling	splitText appendData, deleteData, insertData, replaceData, substringData, appendChild, cloneNode, hasAttributes, hasChildNodes, insertBefore, isSupported, normalize, removeChild, replaceChild

---

## VI. Bibliographie

---

### A. Ouvrages en anglais

Pragmatic Ajax – Justin Gehrtland, Ben Galbraith, Dion Almaer (2005) – The Pragmatic Programmers  
Ajax in action – Dave Crane, Eric Pascarello, Darren Jones (2005) – Manning Publications

### B. Remarques

Ces ouvrages offrent hélas qu'un intérêt très relatif pour deux raisons. La première est qu'il s'appuie sur des exemples très lourds. La deuxième est qu'il s'appuie sur des frameworks sans réellement expliqués le code utilisé.