



## Ajax

Formation

### Développeur Web et Web Mobile

Module : 03

#### Développer la partie front-end d'une application web


Séquence : 03

#### Développer une interface utilisateur web dynamique

Séance : 03

#### Développer des scripts clients dans une page web


|                      |            |
|----------------------|------------|
| Libellé réduit:      | AJAX       |
| Code produit:        |            |
| Type de document:    | Ressource  |
| Version:             | 1          |
| Référence            | 1-1-1      |
| Date de validation:  | 14/02/2013 |
| Date de mise à jour: | 26/10/2022 |

|   |                       |                          |             |                  |                            |
|---|-----------------------|--------------------------|-------------|------------------|----------------------------|
|  | Auteur                | <i>Centre de Créteil</i> | Formation   | Date Mise à jour |                            |
|   | <i>Didier Bonneau</i> | <i>GRN 164</i>           | <i>DWWM</i> | 26/10/2022       | support_de_cours-ajax.docx |

# Ajax

## Sommaire

|      |                                     |   |
|------|-------------------------------------|---|
| I.   | Introduction.....                   | 1 |
| II.  | avantages d'ajax.....               | 2 |
| A.   | Application web classique.....      | 2 |
| B.   | Appllication WEB Ajax .....         | 2 |
| III. | Reponse Ajax .....                  | 3 |
| A.   | Texte simple .....                  | 3 |
| B.   | Html .....                          | 3 |
| C.   | XML .....                           | 3 |
| D.   | JSON .....                          | 3 |
| IV.  | l'objet XMLHttpRequest.....         | 4 |
| A.   | Introduction.....                   | 4 |
| B.   | Propriétés et méthodes.....         | 4 |
| 1.   | Les propriétés.....                 | 4 |
| 2.   | Les valeurs du « readyState » ..... | 4 |
| 1.   | Les méthodes.....                   | 5 |
| 2.   | Les valeurs du « status http »..... | 5 |
| C.   | Envoi d'une requête.....            | 5 |
| D.   | Traitement du coté client .....     | 6 |

|   |                |                   |           |                  |                            |
|---|----------------|-------------------|-----------|------------------|----------------------------|
|  | Auteur         | Centre de Créteil | Formation | Date Mise à jour |                            |
|   | Didier Bonneau | GRN 164           | DWWM      | 26/10/2022       | support_de_cours-ajax.docx |

# La technologie AJAX

## I. INTRODUCTION

Le terme AJAX est l'acronyme de "Asynchronous Javascript and Xml".

C'est Microsoft qui a été pionnière en implémentant l'objet « ajax » dans IE (en tant qu'ActiveX). Le projet Mozilla a rapidement suivi en intégrant directement l'objet dans le code de son navigateur.

La technologie Ajax permet d'envoyer des requêtes à un serveur, en obtenir des données puis modifier en conséquence la page WEB, en manipulant dynamiquement son arbre DOM. Il est ainsi possible d'afficher un résultat de requête sur une base de données ou autre en restant dans la même page.

La technologie Ajax repose sur un ensemble de techniques :

- Une page web faisant usage de codes HTML, CSS et Javascript
- Un objet de communication asynchrone avec un serveur pour obtenir des données
- Une manipulation de l'arbre DOM de la page web qui a lancé la requête pour permettre sa mise à jour


Le langage Javascript permettra de réaliser les différentes actions côté client :

- Envoi de la requête
- Réception de la réponse
- Manipulation du Dom pour l'affichage

Attention la technologie Ajax n'est possible que dans le même domaine (site web) pour des raisons de sécurité. Il existe maintenant de l'Ajx inter domaine mais tous les navigateurs ne l'intègrent pas encore.

### Exemples d'utilisation d'Ajx

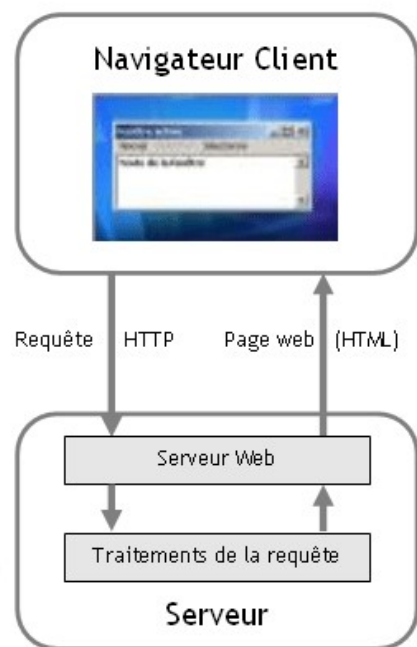
- Google suggest
- Auto complétion ou suggestion d'une zone de saisie
- Gestion de tableaux dynamiques

|   |                |                   |           |                  |                            |
|---|----------------|-------------------|-----------|------------------|----------------------------|
|  | Auteur         | Centre de Créteil | Formation | Date Mise à jour | Page 1                     |
|   | Didier Bonneau | GRN 164           | DWWM      | 26/10/2022       | support_de_cours-ajax.docx |

## II. AVANTAGES D'AJAX

### A. APPLICATION WEB CLASSIQUE

Dans une application web classique, les échanges de données entre le client et le serveur sont réalisés de manière synchrone. Chaque requête au serveur nécessite un traitement côté serveur et impose un rechargement et un affichage complet de la page.



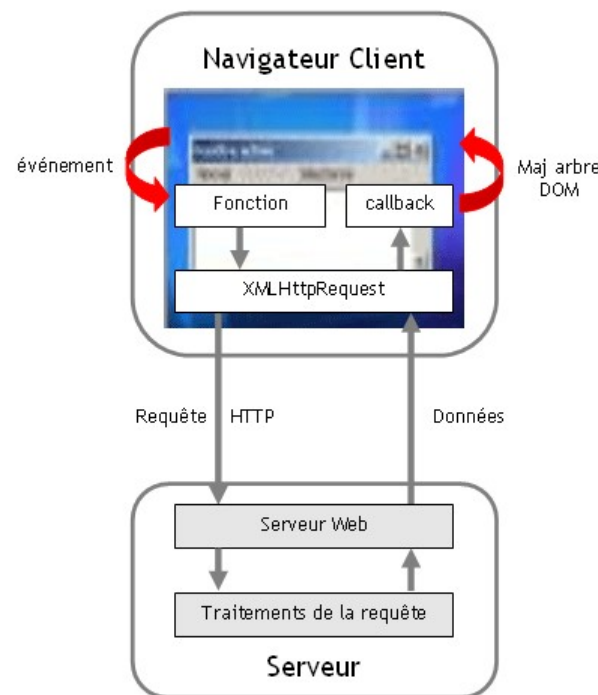
#### Inconvénients :

- ✓ Allongement de la durée des échanges
- ✓ Pour l'utilisateur : obligation d'attendre
- ✓ Pour le serveur : surcharge de travail inutile
- ✓ Pour le réseau : encombrement du trafic

### B. APPLICATION WEB AJAX

A l'opposé, Ajax propose des échanges *asynchrones* entre le client et le serveur.

Le rafraîchissement *partiel* de la page est réalisé par l'écriture dans le DOM des données reçues.



#### Avantages :

Le rôle du (ou des) serveur(s) WEB + langage de scripts + base de données, est réduit

#### Inconvénients :

- ✓ Utilisation de Javascript qui doit être activé sur le navigateur
- ✓ Différences d'implémentation suivant les navigateurs
- ✓ Déroutant pour l'utilisateur : impossibilité de faire un favori vers une page dans un certain état, le bouton "back" ne permet plus de réafficher la page dans son état précédent la dernière action

### III. REPONSE AJAX

Le navigateur et le serveur ne peuvent se parler que via un format de données bien défini. Plusieurs formats sont possibles :

#### A. TEXTE SIMPLE

Recevoir des informations sous forme de texte est souvent mal adapté si ces informations doivent être traitées pour être insérées dans des balises de l'arbre Dom.

#### B. HTML

Le HTML semble intéressant car il suffira de l'insérer directement dans la page avec la propriété innerHTML. C'est rapide. Cela dit la création des balises avec les données se fait côté serveur ce qui peut se révéler assez lourd quand il s'agit d'un grand volume de données.

#### C. XML

Il est possible de récupérer des données sous forme de XML (à l'intérieur de balises spécifiques) et de l'interpréter comme tel, ce qui permet de manipuler les données avec les fonctions DOM.

Ceci peut être intéressant mais XML souffre du même problème que le HTML : il est verbeux.

#### D. JSON

Le JSON est une manière de structurer l'information en utilisant la syntaxe objet de JavaScript – des objets et des tableaux. JSON est très léger. L'évaluation se fait via eval pour les navigateurs obsolètes ou via la méthode parse de l'objet natif JSON.

Le JSON est donc le format travaillant de pair avec AJAX quand il s'agit de recevoir des données classées et structurées.

*"nom\_club1, president1, telephone1, mail1,  
nom\_club2, president2, telephone2, mail2"*

```
<tr>
  <td>nom_club1</td><td> president1</td>
  <td>telephone1</td><td>mail1</td>
</tr>
<tr>
  <td>nom_clu2b</td><td> president2</td>
  <td>telephone2</td><td>mail2</td>
</tr>
```

```
<club>
  <nom>nom_club1</nom><pres> president1</pres>
  <tel>telephone1</tel><mail>mail1</mail>
</club>
<club>
  <nom>nom_clu2b</nom><pres> president2</pres>
  <tel>telephone2</tel><mail>mail2</mail>
</club>
```

```
[
  {"nom": "nom_club1", "pres": "president1",
   "tel": "telephone1", "mail": "mail1"},
  {"nom": "nom_club2", "pres": "president2",
   "tel": "telephone2", "mail": "mail2"}
]
```

## IV. L'OBJET XMLHttpRequest

### A. INTRODUCTION

La communication Ajax entre le client (page web) et le serveur web repose sur l'utilisation de l'objet « **XMLHttpRequest** ».

### B. PROPRIETES ET METHODES

Comme toutes les classes, XMLHttpRequest possède des propriétés et des méthodes.

#### 1. Les propriétés

| Propriétés         | Description   |
|--------------------|---|
| onreadystatechange | Désigne une fonction de rappel qui sera appelée à chaque fois que l'état du traitement de la requête changera d'état (readyState).                      |
| readyState         | Etat du traitement de la requête (valeur numérique de 0 à 4, voir le tableau ci-après pour connaître les différents états et leur signification).       |
| responseText       | Contient le résultat du serveur sous forme de texte.  |
| responseXml        | Contient le résultat du serveur sous forme XML.   |
| status             | Contient le code de statut HTTP (exemple : 200, voir tableau ci-après). Ce code est disponible en lecture uniquement lorsque la réponse a été reçue.    |
| statusText         | Contient le message de statut HTTP (exemple : OK, voir tableau ci-après). Ce texte est disponible en lecture uniquement lorsque la réponse a été reçue. |

Nous serons obligés de prendre en charge la propriété « readyState » car c'est elle qui nous informera que la requête est terminée.

Le status http doit être vérifié pour s'assurer que tout c'est bien déroulé.

#### 2. Les valeurs du « readyState »

| Valeur | Etat          | Signification   |
|--------|---------------|---|
| 0      | Uninitialized | L'objet n'a pas encore été initialisé et donc la méthode open() n'a pas encore été appelée.   |
| 1      | Loading       | L'objet a été initialisé mais la requête n'a pas encore été envoyée à l'aide de la méthode send().  |
| 2      | Loaded        | La requête a été envoyée à l'aide de la méthode send().   |
| 3      | Interactive   | La réponse est en cours de réception.   |
| 4      | Completed     | La réponse du serveur est complètement réceptionnée. Les données sont disponibles dans la propriété.responseText (s'il s'agit de données texte) ou dans responseXML (s'il s'agit de données XML). |

On voit que tant que la valeur du readyState est différente de 4, les données ne sont pas fiables.

C'est donc cette valeur que nous testerons.

## 1. Les méthodes

| Méthodes                                      | Description  |
|---|--|
| <code>open("methode","url","async")</code>    | Initialise l'objet en précisant certains paramètres de la requête <i>methode</i> : la méthode utilisée GET ou POST, <i>url</i> : l'URL du script côté serveur (fichier.php par exemple) et <i>async</i> : paramètre optionnel indiquant si la communication doit être asynchrone (true) ou synchrone (false).  |
| <code>send("queryString")</code>              | Envoie la requête.<br><i>queryString</i> : pourra prendre la valeur null dans le cas d'une requête initialisée avec la méthode GET ou une chaîne de requête dans le cas d'une méthode POST. Évidemment cette méthode ne peut être utilisée que lorsque la méthode <code>open()</code> a déjà été configurée et donc que lorsque l'état du traitement ( <code>readyState</code> ) est égal à 1 (Loading). |
| <code>setRequestHeader("nom","valeur")</code> | Attribue une valeur ( <i>valeur</i> ) à l'en-tête de la requête dont le nom ( <i>nom</i> ) est spécifié dans le premier paramètre. Cette méthode n'est utilisable que lorsque l'état du traitement ( <code>readyState</code> ) est égal à 1 (Loading).   |

## 2. Les valeurs du « status http »

| Code de statut | Signification         | Exemple/Message                                  |
|----------------|-----------------------|--|
| 200 à 299      | Succès de l'opération | 200/OK : Requête accomplie avec succès           |
| 300 à 399      | Redirection           | 301/Moved Permanently : Redirection permanente   |
| 400 à 499      | Erreur client         | 404/Not Found : Document non trouvé              |
| 500 à 599      | Erreur serveur        | 503/Service Unavailable : Service non disponible |

## C. ENVOI D'UNE REQUETE

Il faut créer un objet « XMLHttpRequest »

```
monXHR= new XMLHttpRequest();
```

Sur cet objet, il faut lui indiquer quelle est la fonction de traitement de la requête :

```
monXHR.onreadystatechange = monTraitementResultat ;
```

Maintenant il faut ouvrir l'objet en lui précisant la méthode, l'URL et si on travaille en synchrone ou asynchrone :

```
monXHR.open("get/post","monScript.php",false/true);
```

Reste plus qu'à envoyer la requête :

```
monXHR.send(null/ queryString);
```



Exemples de différentes requêtes :

- requête GET asynchrone sans paramètre

La méthode send() reçoit « null »

- requête GET asynchrone avec paramètre

Les paramètres sont donnés dans l'URL. La méthode send() reçoit « null »

- requête POST asynchrone avec paramètre

En méthode POST, il faut indiquer le type des données envoyées par le biais de la méthode setRequestHeader qui affectera le type correspondant à l'en-tête Content-Type avant d'envoyer la requête.

Les données seront définies dans la méthode send().

#### D. TRAITEMENT DU COTE CLIENT

La fonction de traitement du résultat de la requête est précisée dans la propriété « onreadystatechange ».

La propriété readyState correspond aux différents états de traitement de la requête Ajax (voir tableau), elle changera donc plusieurs fois d'état au cours du cycle de la requête. Comme nous ne désirons récupérer le résultat que lorsque les données sont fiables (complètement réceptionnées), il faut un test dans la fonction de rappel afin de s'assurer que la propriété readyState est égale à la valeur 4.

De plus il faut ajouter un second test afin de s'assurer que tout s'est bien passé au niveau du protocole http avant d'exploiter les données (dans ce cas la propriété status doit être égale à 200).

```
objetXHR=new XMLHttpRequest();
objetXHR.onreadystatechange = traitementResultat ;
```


```
objetXHR.open("get","script.php",true);
objetXHR.send(null);
```

```
objetXHR.open("get","script.php?param=valeur",true);
objetXHR.send(null);
```

```
objetXHR.open("post","script.php",true);
objetXHR.setRequestHeader(
"Content-Type","application/x-www-form-urlencoded")
objetXHR.send("param1=valeur1&
                parm2=valeur2&...");
```

#### Traitement du résultat coté client :

```
function traitementResultat() {
  if(objetXHR.readyState==4) {
    if(objetXHR.status==200) {
      var resultat = objetXHR.responseText
      ou
      var resultat = objetXHR.responseXML
      //Le résultat peut maintenant être pris en compte
    }else {
      alert("Erreur HTTP N°"+ objetXHR.status);
    }
  }
}
```

|   |                |                   |           |                  |                            |
|---|----------------|-------------------|-----------|------------------|----------------------------|
|  | Auteur         | Centre de Créteil | Formation | Date Mise à jour | Page 6                     |
|   | Didier Bonneau | GRN 164           | DWWW      | 26/10/2022       | support_de_cours-ajax.docx |