AJAX - SOP

Asynchronous JavaScript And XML

Same Origin Policy

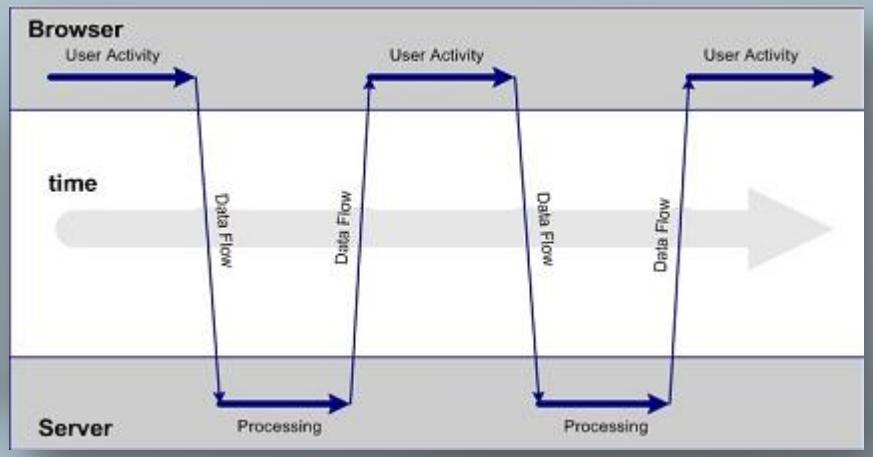
Sommaire

- Vue globale d'AJAX
  - Principe
  - XMLHttpRequest
- Utilisation courante
  - Framework JavaScript
  - JSON
- Problématique SOP
  - Origine
  - JSONP
- Conclusion

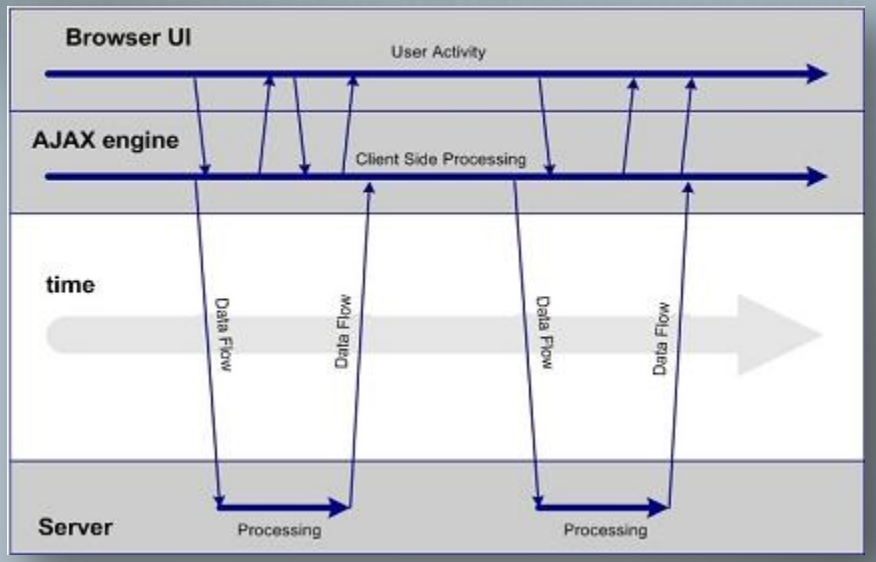
AJAX - Principe

- Origines
  - 2005 : J.J. Garret [Adaptive Path]
    - « Ajax, a new approach to Web App »
    - ➔ « plusieurs technologies se développant chacune de leur côté, combinées ensemble pour donner des résultats aussi nouveaux que puissants. »
  - Google donne l'exemple
    - Google Search (autocomplétion)
    - Google Map/Reader/Calendar

AJAX - Principe



Fonctionnement classique d'une application Web



Fonctionnement d'une application Web avec AJAX

AJAX - Principe

- Définition
  - (**A**)synchronous
  - **J**avaScript (**A**nd)
  - **X**ML (Text)
- Composition
  - Présentation : DOM (layout) / CSS (L'n'F)
  - Données : XML / Text
  - Dialogue : XMLHttpRequest

➔ XMLHttpRequest (client) ⇔ scripts (serveur)

XMLHttpRequest

- Extérieur au contexte de la page
  - prompt() / window.open() / …
- gestion des requêtes HTTP
- Implémentation dépendante du navigateur

```
function getXMLHttpRequest() {
        var xmlhttp = null;

        if (window.ActiveXObject)
                xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");  IE6 –
        else
                xmlhttp = new XMLHttpRequest();  Firefox/Safari/Opera/IE7+ …

        return xmlhttp;
}
```

XMLHttpRequest - méthodes

- **open**()
  - GET / POST
  - URL du serveur
  - Asynchrone / Synchrone
- **send**(Postdata)

- abort()
- get[All]ResponseHeader[s](header)
- setRequestHeader()

XMLHttpRequest - propriétés

- onreadystatechange
- readyState
  - 0: uninitialized,
  - 1: loading,
  - 2: loaded,
  - 3: interactive,
  - **4**: complete
- status[Text]
- responseText [XML]

Exemple 1 - Hello World

- HTML

```
<div id="target">
</div>
<input type="button" value="hello world" onclick="processAjax();return;">
```

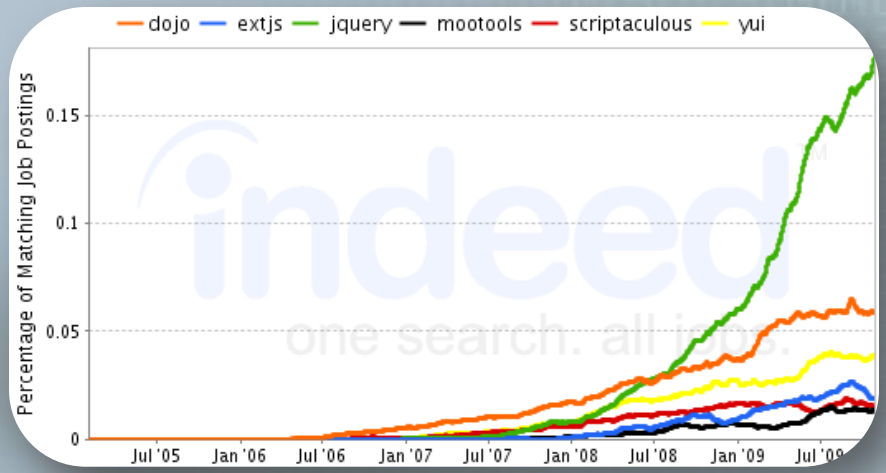
- JavaScript

```
function getXMLHttpRequest() {
        // recupere instance XMLHttpRequest
}
var xmlhttp = getXmlHTTPRequest();
function processAjax() {
    var dest = document.getElementById("target");
    xmlhttp.open("GET", "http://localhost/xpose/server.php?name=jean");
    xmlhttp.onreadystatechange = function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200)  // success
            dest.innerHTML = xmlhttp.responseText;
    }
    xmlhttp.send(null);
}
```

```
echo "Hello ".$_GET["name"];
```
server.php

AJAX - Frameworks

- Nombreux Frameworks JavaScript
  - JQuery
  - Mootools
  - Prototype
  - …

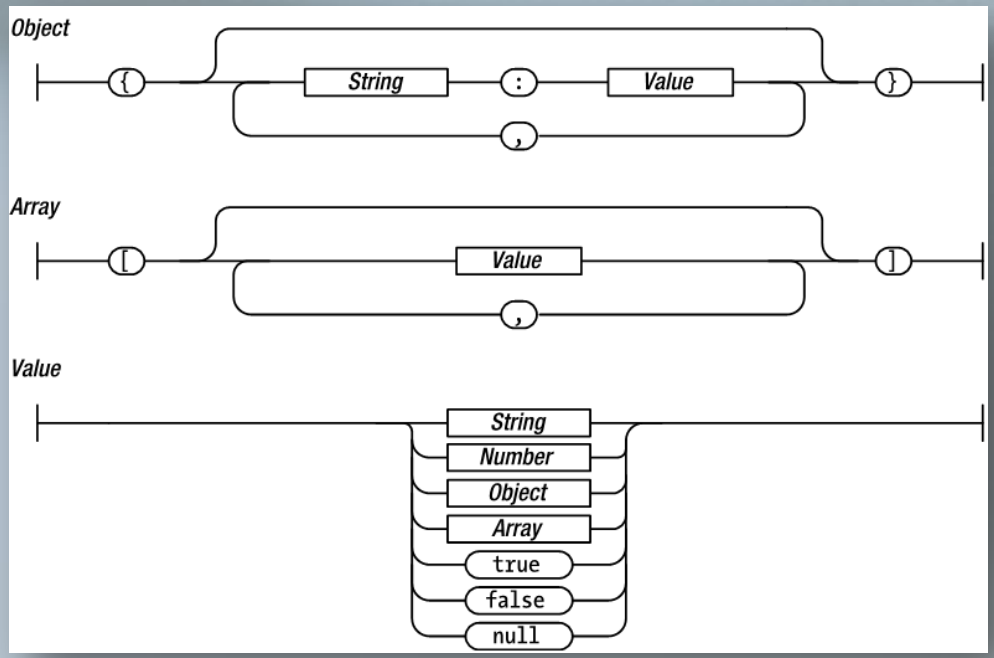


- Intérêt
  - code non intrusif
    - Séparation code HTML / JavaScript
    - Gestion désactivation JavaScript
  - code plus lisible / maintenable
  - DOM adapté à l'ensemble des navigateurs

AJAX - JSON

Java**S**cript **O**bject **N**otation

- Alternative au XML pour échanger des données
- Construit sur deux structures de données :
  - Une liste de paires « clé / valeur »
  - Une liste ordonnée de valeurs (tableaux / objets / génériques)



AJAX - JSON

- Instance pour un étudiant :

```
var student = {
        "firstName" : "jean" ,
        "lastName" : "eymar" ,
        "studentId" : 222
}
```

- Accès / modification des champs :

```
var name = student.lastName;
student.studentId = 111;
```

- Représentation XML :

```
<student>
            <firstName>jean</firstName>
            <lastName>eymar</lastName>
            <studentId>222</studentId>
</student>
```

Exemple 2 - JQuery / JSON

- HTML

```
<div id="target">
</div>
<input type="button" value="hello world" id="trigger"/>
```

- JavaScript : actionEvent.js

```
$(document).ready ( function() {
        $("#trigger").click(
                function() {
                    $.ajax({
                        url: "server.php",
                        dataType: "json",
                        data: "name="+name,
                        success: function(data) {
                            var res = '<`+data.balise+' style="'+data.style +'">` +
                                data.message +
                                        '</'+data.balise+'>';
                            $("#target").html(res);
                        }
                    });
                }
        );
```

server.php

```
echo '{
        "balise": "h3",
        "style": "color:#FF0000",
        "message": "Hello $_GET['name']"
}';
```

AJAX - Besoin

- Pouvoir requêter sur des API distantes
  - Google
  - Yahoo
  - Facebook
  - …

➔ système de WebService pour AJAX

AJAX - Problème : SOP

Same **O**rigin **P**olicy

- Restriction sur l'origine des scripts requêtés :
  - Serveur
  - Protocole
  - Port

Exemple de détermination de l'Origine

URL requêtée via XMLHttpRequest	SOP	Raison
http://www.exemple.com/dir/page.php	OK	
http://www.exemple.com/dir2/index.php	OK	
http://www.exemple.com**:81**/dir2/index.php	KO	port
https://www.exemple.com/dir2/index.php	KO	protocole
http://**fr.**exemple.com/dir2/index.php	KO	serveur

SOP - Origine

- Netscape 2.0
  - ➔l'intégrité d'un document d'un domaine A ne peut être modifiée par un document d'un domaine B
- Politique étendue
  - Navigateurs (IE, Firefox, Opera, ...)
  - Langages de script (Adobe Flash)

SOP - Une solution ?

Balise script : aucune restriction sur l'origine

```
<script type="text/javascript" src="http://site.com/api.php?arg=bob">
```

Problème : appel non réactif

➔ création de la balise script dynamiquement

```
Function getRemoteData(url) {
        var script = document.createElement("script");
        script.type = "text/javascript";
        script.src  = url;
        $("head")[0].appendChild(script);
}
```

Problème : traitement du retour impossible

➔ JSONP

SOP - JSONP

JSON with **P**adding

- ➔ Gestion de la fonction de retour (callback)

url = http://www.sitedistant.com/api.php?param=hello&**callback**=myFunction

client :

```
Function getRemoteData(url) {
        // création d'une balise script avec l'URL&callback=myFunction
        // ajout dans la partie <head></head> via le DOM
}
Function myFunction(data) {
        // traitement des données JSON
}
```

serveur :

```
echo $_GET['callback']. '(' .$json. ');' ;
        // ➔ myFunction( {"name" : "Jean" , "value" : 10} );
```

SOP - JSONP

JQuery 1.2 : intégration de JSONP

```
$.ajax({
	dataType: 'jsonp',
	url: 'http://www.sitedistant.com/api.php?name='+name,
	success: function (data) {
		alert(data.message);
	}
});
```

Fonctionnement de JSONP avec JQuery :

- création et inclusion de la **balise script**
- création d'une fonction de type **jsonp12345**
- ajout du paramètre &**callback=jsonp12345**

Exemple 3 - JSONP / FlickR

- HTML

```
<div id="target">
</div>
<input type="button" value="hello world" id="trigger"/>
```

- JavaScript : actionEvent.js

```
$.ajax({
        dataType: 'jsonp',
        jsponp: 'jsoncallback',
        url: 'http://api.flickr.com/services/photos.gne? tagmode=any&format=json,
        success: function (data) {
                $.each(data.items, function(i, item){
                        $("<img/>").attr("src", item.media.m)
                                .wrap("<a href='" + item.link + "'></a>")
                                .appendTo("#target");
                        if ( i == 3 )
                                return;
                });
        }
});
```

```
echo $_GET['jsoncallback'].'({
        "items" : [
                "link" : - url - ,
                "media" : {"m" : -url .jpg-}
        …]
})';
```

api.flickr.com/photos.gne

Conclusion

- AJAX (+JSON)
  - ➔Récupération dynamique de données
  - ➔Transparent pour l'utilisateur
  - ➔Limitation SOP
- AJAX + JSON + JSONP
  - ➔Aucune contrainte sur l'URL requêtée
  - ➔nombreuses API implémentant JSONP :
    - API Google (search/map/youtube…)
    - API Facebook
    - API FlickR

Bibliographie

- Publications
  - Ajax in action

    Dave Crane, Eric Pascarello
  - Foundations of Ajax

    Ryan Asleson
- Web
  - Blog Jaysalvat *http://is.gd/5gSjb*
  - Jquery doc *http://is.gd/5hUMR*
  - API flickr *http://is.gd/5gSAY*

AJAX - SOP

Merci

--

Questions ?