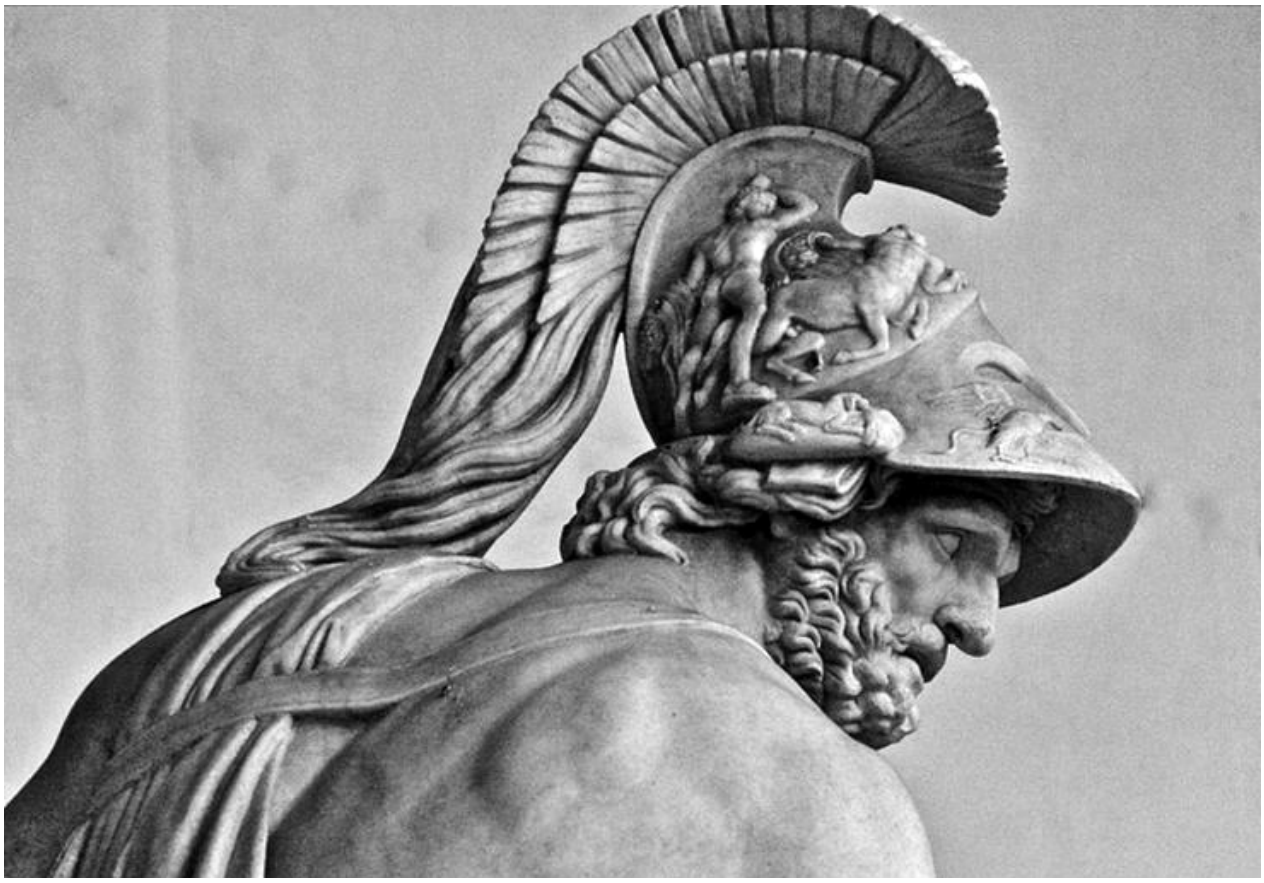
AJAX : *Asynchonous JavaScript And XML*

Par Sayah El Yatim (@sqyqh)



- **Objectifs :** Mettre en place un chargement asynchrone des données.
- **Prérequis :** Savoir analyser et rédiger.
- **Public :** Débutants avec des notions de JavaScript, jQuery et Git.

Introduction

AJAX (*Asynchonous JavaScript And XML*) n'est pas un langage de programmation mais plutôt une approche, une méthode, qui utilise un ensemble de technologies web. Une fois combinées ensemble avec une approche AJAX au sein d'une application, l'application pourra émettre des requêtes serveurs et mettre à jour l'interface utilisateurs rapidement sans forcer le rechargement de toute une page.

Ajax peut tout aussi bien fonctionner avec du json, du yaml etc…

Une première chose à **comprendre** une requête AJAX fonctionne comme n'importe quelle requête à l'exception près qu'elle ne nécessite pas un rechargement de la page entière.

Par exemple, au lieu d'envoyer une requête POST avec un formulaire, on peut à la place se servir d'AJAX pour envoyer le formulaire sans avoir à naviguer sur une autre page.

Pour commencer

Installation du dépôt

Pour cloner le dépôt du cours (celui-ci même) nous utilisons la commande :

```
git clone https://github.com/sqyqh/cours-ajax
```

Pour mettre à jour votre dossier, vous pouvez directement *pull* (tirer) les sources du dépôt d'origine vers votre copie locale avec la commande :

```
git pull
```

Attention : lancez la commande depuis le dossier de votre copie locale.

Les méthodes jQuery Ajax

Méthode load()

La méthode *load()* permet de charger du contenu externe. Elle nécessite jusqu'à 3 paramètres : URL, data et callback.

- **URL** spécifie l'adresse du contenu,
- **data** (paramètre optionnel) spécifie un jeu de couples clés/valeurs à envoyer avec la requête,
- **callback** (paramètre optionnel) spécifie le nom de la fonction qui sera exécutée une fois que la méthode load() est complète.

Requêtes GET et POST

Les méthodes jQuery *ajax()*, *get()* et *post()* sont utilisées pour demander des données du serveur avec un requête HTTP GET ou POST.

- **GET** - requête de données depuis une ressource spécifiée
- **POST** - envoi de données pour traitement depuis une ressource spécifiée

Exemple de requête AJAX de type GET utilisant la méthode *ajax()* (documentation):

```
$.ajax({
    type: "GET",
    url: 'test.php',
    data: {prenom: 'Philippe', nom: 'ROBERT', age: 38},
    success: function(data){
        alert(data);
    }
});
```

Dans ce code, il y a quatre paramètres/options :

- **type :** c'est le type de la requête http qu'on souhaite envoyer. Dans notre exemple, on envoie donc une requête de type GET. Pour envoyer une requête POST, il faut simplement le 'GET' en 'POST'.
- **url :** c'est l'URL à laquelle on envoie notre requête AJAX. Dans notre exemple, c'est "test.php". Vous pouvez changer cela par "autreTest.php" ou dans un dossier "monDossier/monAutreTest.php" ou n'importe quelle autre fichier php qui traitera la requête. Ne surtout pas oublier que cette adresse est relative à l'adresse page dans laquelle l'utilisateur se trouve.
- **data** est un objet JavaScript qui contient les données qui seront envoyées avec notre requête. Dans notre exemple, nous envoyons avec notre requête une chaîne de caractères qui va s'ajouter à la chaîne la requête comme suit : **test.php?prenom=Philippe&nom=ROBERT&age=38**
- **success :** c'est la fonction qui est appelée une fois que la requête est réussie. Cette fonction a pour paramètre *data* qui contient ce qui est retourné par *test.php*. Si par exemple, *test.php* imprime la chaîne de caractères "Super !", alors la variable *data* contiendra la chaîne "Super !".

Exemple de requête POST avec la méthode ajax() :

```
$.ajax({
    type: "POST",
    url: 'envoi.php',
    data: {prenom: 'Gérard', nom: 'NOAS', age: 55},
    success: function(data){
        alert("Super ! L'inscription des données suivantes est réalisée : " + data);
    }
});
```

Dans *envoi.php* on a par exemple :

```
$prenom = $_POST['prenom'];
$nom = $_POST['nom'];
$age = $_POST['age'];
```

Comme on le constate, **ces variables sont accessibles de la même manière qu'avec requêtes GET ou POST habituelles**.

On peut modifier la fonction **success** pour notamment afficher un message à l'intérieur de la page concernée. En bref, modifier l'UI (*User Interface* soit Interface Utilisateur) pour que l'utilisateur ait une forte impression de réactivité et qu'il puisse lire un message qui lui indique que sa requête est bien validé.

En plus de la méthode *ajax()* il existe aussi les méthodes *get()* et *post()* qui permettent la même chose mais avec une syntaxe différente. Si on reprend les exemples précédents on aura donc, pour les mêmes requêtes :

Dans le cas de la méthode *get()*, la requête peut se faire comme suit :

```
$.get("test.php",
    {
            prenom: 'Philippe',
            nom: 'ROBERT',
            age: 38
    },
    function(response, status){
        alert("Data: " + response + "\nStatus: " + status)
    }
)
```

Dans le cas de la méthode *post()*, la requête peut se faire comme suit :

```
$.post("envoi.php",
    {
            prenom: 'Gérard',
            nom: 'NOAS',
            age: 55
    },
    function(response, status){
        alert("Data: " + response + "\nStatus: " + status)
    }
)
```

Sources

jQuery Ajax documentation
HTTP Methods