AJAX

I. Objectifs

1. Comprendre le principe de fonctionnement d'AJAX
2. Mettre en œuvre (via le framework JQuery) une application Web utilisant AJAX

II. Définition

AJAX signifie *Asynchronous Javascript And XML*. Il s'agit d'un objet natif de Javascript (comme *Math*, *Window*...). Cependant, sa mise en oeuvre nécessite de nombreuses lignes de code).

La librairie **jQuery** prévoit un ensemble de fonctions Ajax qui facilite la tâche.

III. Cheminement

III.1. Phase 1 - Découverte

Lire le document 01 – Présentation AJAX

Reprenez les exemples de l'archive AJAX_demo.

Si vous utilisez Bootstrap, vous chargez déjà jQuery mais le fichier appelé doit être remplacé car il s'agit d'une version allégée de jQuery (*slim*) qui n'inclut pas les fonctions Ajax. Allez sur le site jQuery pour charger une version non (*slim*).

III.2. Phase 2 - Travailler avec JSON

Lire le document 03 – JSON.pdf

IV. Phase 3 - Prototype d'une requête Ajax avec Jquery

Le code Javascript/Jquery :

```
$(document).ready(function()
{
   $("#bouton1").click(function()
  {
        $.post({
            url: "post.php",
            data: { pro_id:7 },
            success: function(resultat)
        {
                $("#div1").html(resultat.pro_libelle);
            }
         });

        return false;
   });
});
```

Explication du code :

- `$.post` : cette méthode Jquery spécifie que l'on envoie une requête HTTP Post. Permet de se passer de l'attribut `type`.
- `url` : l'url vers laquelle la requête est envoyée
- `data` : paramètres (tableau si plusieurs) à transmettre au fichier de destination; ici on tramset une variable `pro_id` qui a pour valeur *7*.
- `success` : ce que le script doit faire si la requête est réussie : ici le contenu du `<div>` qui a pour attribut `id="div1"` sera remplacé par ce que vaut la variable `resultat`.
- `return : false;` est utilisé pour empêcher - le cas échéant - l'action par défaut (envoi d'un formulaire, d'un lien...) par HTTP quand l'utilisateur clique sur le bouton *submit* (on peut aussi utiliser `e.preventDefault()`.

Contenu du fichier PHP `post.php` :

```
$sql = "SELECT * FROM produits WHERE pro_id=".$_POST['id'];
$result = $db->query($sql);
$oProduit = $db->query($sql)->fetch(PDO::FETCH_OBJ));
echo json_encode($oProduit);
```

Explication du code PHP :

- les 2 premières ligne effectuent une requête SQL classique (dans l'exemple la requête SQL est non préparée mais pensez à le faire bien sûr).
- la fonction `json_encode()` va encoder la variable `oProduit` au format JSON.
- `echo` stoppe le script PHP pour renvoyer une valeur côté Javascript.

Dans le code Javascript ci-dessus, la valeur écrite par `echo` du PHP est réceptionnée dans la variable `resultat`, laquelle permet d'accéder aux propriétés de l'objet JSON par notation pointée ; ces propriétés correspondent aux colonnes de la table SQL interrogée.

V. Phase 4 - Mise en œuvre

1. A partir de l'archive *AJAX_demo*, reproduisez l'exemple régions/départements (base sql).
2. Dans le projet Jarditou, sur le formulaire d'ajout d'un produit, faites-en sorte que le choix d'une catégorie charge les sous-catégories dans une seconde liste.

VI. Ressources complémentaires

- Méthode $.ajax (JQuery)
- AJAX avec jQuery (OpenClassrooms)
- AJAX (MDN),
- Les entêtes 'Cross-Origin'