Prog web client riche - JavaScript
IUT Montpellier-Sète – Département Informatique
TD1
Thème : dynamiser une page web

INTRODUCTION

Dans ce TD, nous allons découvrir et manipuler un objet très important en JavaScript : l'objet XMLHttpRequest.

Cet objet est une interface JavaScript permettant de communiquer avec un serveur pour recueillir des données.

Il permet par exemple de lancer des requêtes sur une base de données, mais aussi, plus simplement, de recueillir des données affichées par un simple echo PHP.

Vous utiliserez le format JSON, qui offre un format très lisible et compréhensible d'échange de données, et que vous pouvez utiliser en PHP et en JavaScript. Ce format sera utilisé simplement au travers de fonctions PHP (json_encode) et JS (JSON.parse).

EXERCICE 1 – premier exemple simple

1. Ecrivez un fichier coucou.php dont le seul objectif est d'afficher « coucou », au niveau de votre dossier public_html/TD5.

   ```
   <?php echo "coucou"; ?>
   ```

2. DE votre navigateur, faites une requête http sur la page coucou.php puis ouvrez la console JavaScript (F12, onglet console).

3. Entrez la commande suivante

   ```
   let xhr = new XMLHttpRequest();
   ```

   Cette commande instancie un nouvel objet JavaScript, stocké dans la variable xhr.

4. Cet objet sert à lancer des requêtes au serveur. Nous allons lancer, grâce à la variable xhr, une requête à la page coucou.php. Pour cela, entrez les commandes suivantes dans la console :

   ```
   xhr.readyState;
   xhr.responseText;
   xhr.open("GET","votre url vers coucou.php",true);
   xhr.readyState;
   xhr.responseText;
   xhr.send(null);
   xhr.readyState;
   xhr.responseText;
   ```

- readyState donne l'état d'avancement de la requête, voir cours plus tard (valeurs de 0 à 4)
- responseText donne l'état actuel de la réponse textuelle à la requête
- la méthode open donne à xhr tous les éléments pour lancer la requête.
- la méthode send envoie la requête (le paramètre null vient du fait que la méthode est GET, voir plus tard dans le cours).
- le paramètre « true » de la méthode open sera détaillé dans le cours.

5. Commentez l'évolution des attributs readystate et responseText

EXERCICE 2 – Utilisation de json_encode (PHP) et de JSON.parse (JS)

1. Examinez le contenu du fichier src/haddock_v1.php puis appelez-le dans le navigateur. Vérifiez qu'à l'affichage, vous obtenez bien une chaîne de caractères correspondant à un tableau équivalent à celui stocké dans la variable $haddock.

2. comme dans le premier exemple, créez un objet XMLHttpRequest puis lancez les commandes suivantes :

```
let xhr = new XMLHttpRequest();
xhr.open("GET","votre url vers haddock_v1.php",true);
xhr.send(null);
xhr.readyState;
xhr.responseText;
let resultat = xhr.responseText;
resultat;
let tab = JSON.parse(resultat);
tab;
```

EXERCICE 3 – Utilisation de json_encode (PHP) et de JSON.parse (JS)

3. Examinez le contenu du fichier src/haddock_v2.php puis appelez-le dans le navigateur. Vérifiez qu'à l'affichage, vous obtenez bien une chaîne de caractères correspondant à un tableau équivalent à celui stocké dans la variable $haddock.

4. comme dans le premier exemple, créez un objet XMLHttpRequest puis lancez les commandes suivantes :

```
let xhr = new XMLHttpRequest();
xhr.open("GET","votre url vers haddock_v2.php",true);
xhr.send(null);
xhr.readyState;
xhr.responseText;
let resultat = xhr.responseText;
resultat;
let obj = JSON.parse(resultat);
obj;
```

5. Remarques sur la variable obj ?

EXERCICE 4 – chargement long d'un fichier texte

Il peut arriver que le chargement des données à recueillir soit assez long. Dans ce cas l'attribut readystate ne passe pas immédiatement à la valeur 4.

L'événement load traduit la fin du chargement des données.

On va donc mettre l'objet xhr en état d'écoute de cet événement.

Voici les commandes :

```
let xhr = new XMLHttpRequest();
xhr.open("GET","un_certain_fichier.txt",true);
xhr.addEventListener("load",function() {
      console.log("chargement terminé !!!");
});
xhr.send(null);
let texte = xhr.responseText;
texte.length;
```

remarque : évitez d'afficher directement dans la console la variable `texte`. Sa taille importante peut faire planter le navigateur...

Détails

- on crée un objet XMLHttpRequest ;
- on lui donne les éléments pour lancer la requête
- on le met en écoute de l'événement « load », et quand cet événement se produit, on lance une fonction anonyme qui affiche dans la console « chargement terminé » et on lance la requête.

Cette fonction lancée après la fin du chargement est habituellement appelée fonction « callback ».

Vous allez mettre en œuvre ces commandes en chargeant divers fichiers txt de tailles variées (voir le zip sur l'ENT) :

mobydick.txt (environ 1,2 Mo)
bible.txt (environ 4,4 Mo)
bible2.txt (environ 45 Mo) obtenue par recopie du précédent

voyez-vous un délai pour le lancement de la fonction callback ?