

TP4 : Développement web AJAX

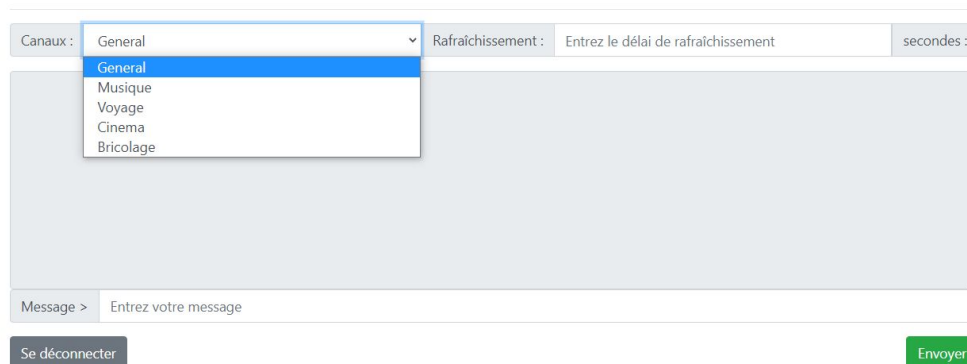
Ayoub KARINE (ayoub.karine@isen-ouest.yncrea.fr)

Objectifs

Dans ce TP vous allez mettre en place la fonction JavaScript permettant au client de faire des requêtes au serveur. Pour cela vous utiliserez la technologie AJAX qui permet de faire des requêtes asynchrones, autrement dit non bloquantes. Pour tester cette nouvelle fonction, vous mettez en place la requête nécessaire à l'affichage des salons disponibles dans le chat.

Ce TP est une amélioration du TP3. Le résultat final consiste à insérer le résultat de la requête demandée au serveur dans le formulaire de chat créé dans le TP1 :

Chat



Préparation de l'environnement de travail

1. Via la WSL, déplacez vous vers le chemin du serveur apache et tapez `code` .

```
akarine@DESKTOP-FTU8G8F:/var/www/html$ code .
```

2. Créer un dossier TP4
3. Déplacez les fichiers/dossiers :
 - a. html et JS du TP1 dans TP4
 - b. php du TP3 dans TP4

4. Démarrez le serveur apache

```
sudo service apache2 start
```

5. Démarrez postgresql

```
sudo service postgresql start
```

Requête AJAX et affichage dans la console :

6. Écrivez, dans le fichier "utils.js", une fonction nommée ajaxRequest. C'est une fonction "générique" qui doit permettre de faire des requêtes AJAX en prenant les 2 paramètres suivants en arguments :

- a. **type** : le type de la requête (GET, POST...);
- b. **url** : l'url du serveur avec la ressource demandée (par exemple le chemin du script PHP);

Attention : le résultat de la requête ne doit être traité par la fonction callback que si la réponse du serveur est correcte (code HTTP 200 ou 201). De plus, décodez la réponse JSON en JavaScript avec :

```
1 JSON.parse(xhr.responseText);
```

Pour afficher le résultat dans la console JavaScript, ajoutez un affichage dans votre fonction « ajaxRequest » grâce à :

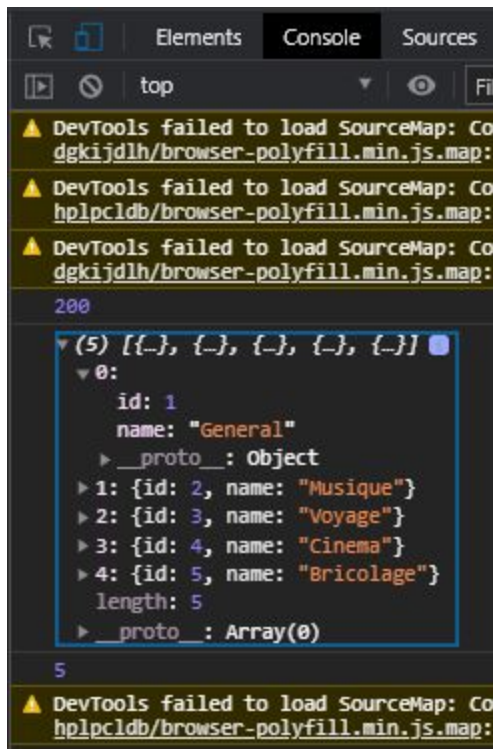
```
1 console.log('texte');  
2 console.log(variable);
```

7. Écrivez, dans le fichier "chat.js", une fonction sans arguments nommée getChannels permettant d'envoyer une requête AJAX qui récupère la liste des salons (channels). Ainsi la requête à envoyer est (c'est pas du tout le lien qu'il faut mettre dans le navigateur) :

```
GET php/chat.php?request=channels Liste des salons en JSON
```

8. Appelez la fonction getChannels au début du fichier "chat.js".

9. Dans le navigateur, lancez la page HTML puis clic droit > inspecter > console. Le résultat doit être comme suit :



Requête AJAX et affichage dans la page HTML :

10. Pour afficher la liste des salons dans la page HTML plutôt que dans la console JavaScript, écrivez la fonction `displayChannels`. Celle-ci prend en paramètre la réponse du serveur (le résultat de la question 9) et l'injecte dans la liste déroulante de la partie chat de la page HTML. Vous prendrez soin de formater en HTML, avec la balise `<option>`, la réponse du serveur et d'ajouter un id à la liste déroulante pour pouvoir la modifier en JavaScript. Utilisez `innerHTML`.
11. Ajoutez un argument "callback" à la fonction `ajaxRequest`. Ce callback, qui aura comme argument la réponse JSON de la requête AJAX, sera appelé une fois que la réponse du serveur est obtenue (appel asynchrone). Lors de l'appel de la fonction `ajaxRequest`, cet argument sera la fonction `displayChannels`.
- Rappel :** un callback permet d'appeler une fonction à l'intérieur d'une autre fonction.
Exemple : https://www.w3schools.com/js/tryit.asp?filename=tryjs_callback4
12. Modifiez le callback de la requête AJAX de la fonction `getChannels` pour appeler votre fonction d'affichage.

Gestion des erreurs de la requête AJAX :

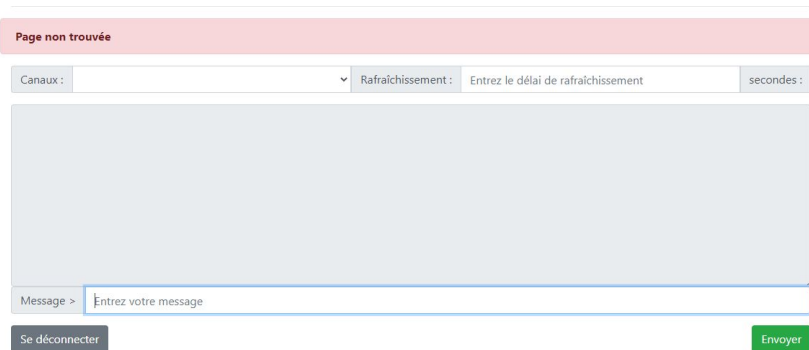
13. Lorsqu'une erreur survient lors de la requête AJAX, l'utilisateur doit en être informé. Pour ce faire, écrivez dans le fichier « `utils.js` » une fonction nommée `displayErrors` qui prend en paramètre un statut HTTP. Cette fonction doit afficher un message d'erreur dans la

div d'id errors en se basant sur le tableau suivant (utilisez la fonction showhide déjà programmée) :

```
1 let messages = {  
2   400: 'Requête incorrecte',  
3   401: 'Authentifiez-vous',  
4   404: 'Page non trouvée',  
5   500: 'Erreur interne du serveur',  
6   503: 'Service indisponible'  
7 };
```

14. Testez la fonction en envoyant une requête de type GET au fichier chatpp.php et vérifiez si le message d'erreur va s'afficher dans la page index.html. L'affichage dans la page HTML doit être comme suit :

Chat



The screenshot shows a web chat interface. At the top, there is a pink banner with the text "Page non trouvée". Below this, there is a header area with a dropdown menu labeled "Canaux :", a refresh button labeled "Rafraichissement :", a text input for "Entrez le délai de rafraichissement", and a label "secondes :". The main chat area is a large, empty light blue rectangle. At the bottom, there is a message input area with a label "Message >" and a text input "Entrez votre message". Below the input area, there are two buttons: "Se déconnecter" and "Envoyer".

15. Pour le moment, le message d'erreur reste affiché en permanence. Pour éviter cela, ajoutez dans la fonction displayErrors un timeout qui cachera le message d'erreur au bout de 5 secondes. Pour ce faire, utilisez la fonction JS [setTimeout](#)