Initiation à Javascript

Utilisation d'Ajax avec Axios

I - Quelques éléments de culture

A. Ajax

Lorsque vous utilisez *Deezer*, *Facebook* ou *Twitter*, vous constatez qu'une petite partie de la page web est modifiée sans que le navigateur ait besoin de recharger l'ensemble de la page.

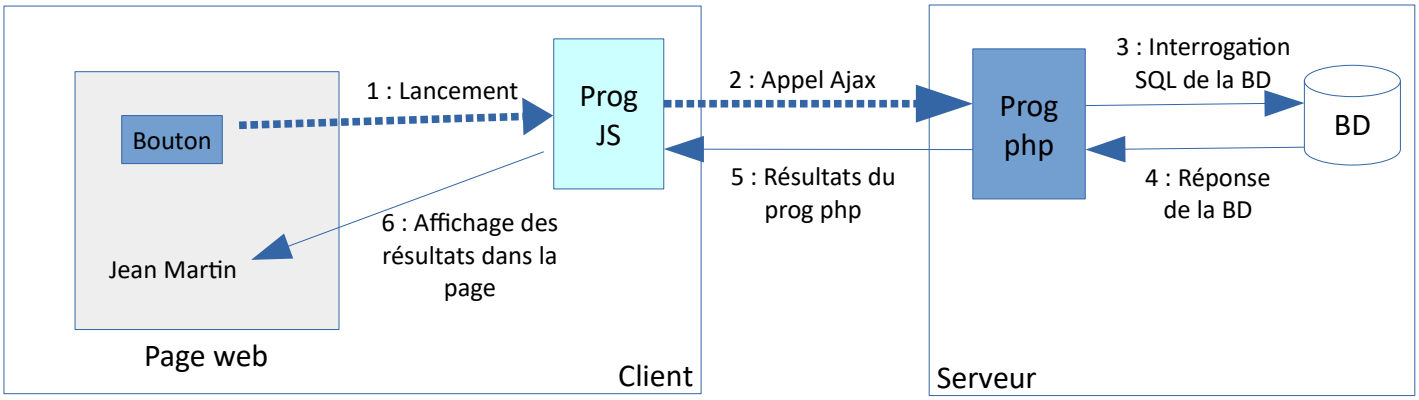
Ajax (Asynchronous Javascript And Xml) est utilisé dans ce but.

Ajax regroupe un ensemble de techniques permettant à un programme js de modifier seulement une partie d'une page web en y incorporant des données reçues au format XML ou JSON.
Sans Ajax, pour modifier le contenu d'une page, le navigateur client doit recharger toute la page.
Avec Ajax, clients et serveurs peuvent communiquer de manière plus souple, un navigateur peut ainsi demander à ce que soit modifiés seulement certains éléments de la page web.

Dans notre programme Javascript, les gestionnaires d'événements récupéreront des données au format JSON générées par des programmes php exécutés par le serveur.

B. Architecture mise en jeu

Ici, notre page web contiendra un bouton permettant au programme javascript de lancer l'exécution d'un appel Ajax, ce qui déclenchera l'exécution d'un programme php sur le serveur. Le programme php ira chercher des informations dans une base de données. Ces informations seront récupérées par le programme javascript comme résultat de l'appel Ajax.



C. API

Application Programmer Interface : ensemble de ressources permettant à un programme de dialoguer avec un autre programme.
Les API permettent à un programme *Javascript* de récolter des informations délivrées par un autre serveur web.
Rien de tel qu'un bon exemple. Visualisez les données servant à créer les marqueurs sur notre carte : https://data.toulouse-metropole.fr/api/records/1.0/search?dataset=defibrillateurs&rows=5.
Le résultat reçu par votre navigateur est un texte au format JSON, qui décrit un ensemble de données structurées.
Si votre navigateur n'est pas capable d'afficher correctement ces informations, **ajoutez-lui une extension d'affichage des textes JSON**.

D. JSON

JavaScript Object Notation est une manière de décrire une donnée structurée.
Le texte JSON présenté à droite décrit le tableau "musiciens" contenant deux personnes pour lesquelles nous possédons trois informations.
nom et *prenom* sont des informations de type texte, *anneeNaiss* est une donnée de type numérique.
La structure de cette information la rend similaire à une table de base de données :

nom	prenom	anneeNaiss
"Patitucci"	"John"	1959
"Page"	"Jimmy"	1944

```
{"musiciens" :
[
   {
         "nom" : "Patitucci",
         "prenom" : "John",
         "anneeNaiss":1959
   },
   {
         "nom" :   "Page",
         "prenom" : "Jimmy",
         "anneeNaiss" :1944
   }
]
}
```

Exemple de texte json

Les symboles *[* et *]* délimitent un **tableau** : un ensemble d'éléments possédant une structure similaire (ou une sémantique équivalente). Ici, le tableau contient deux éléments similaires : deux personnes.
En JSON, la syntaxe générale d'un tableau est : *[élément1 , élément2, ... , dernierElement]*
Les symboles *{* et *}* délimitent un **objet**. Un objet regroupe généralement plusieurs champs.
En JSON, la syntaxe générale d'un objet est :

```
{
  "nom du champ 1" : "valeur du champ 1",
  "nom du champ 2" : "valeur du champ 2",
  …
  "nom du dernier champ " : "valeur du dernier champ"
}
```

Les valeurs textuelles sont entourées par des guillemets contrairement aux valeurs numériques.

Remarque : Un champ peut contenir un objet ou un tableau, un tableau peut contenir des objets ...

Exercice :

- Allez ici pour voir le résultat d'une requête soumise à l'API d'open data soft
- L'objet principal comporte trois champs, pour chacun d'entre eux, donnez son nom et sa nature (texte, nombre ou objet)
- Le champ *records* contient-il un objet ou un tableau ?
- Si l'on stocke l'objet présenté ici dans la variable *resultat*, le chemin pour atteindre le champ *recordid* du premier élément des *records* est *resultat.records[0].recordid.* Donnez le chemin permettant d'atteindre le premier prénom de cette donnée.
- Observez l'URL présent dans la barre d'adresse de votre navigateur. Pouvez-vous modifier cette adresse pour obtenir 20 résultats ?

II - Manipulation

A. Traitement d'un objet

Dans le site fourni, remarquez que le fichier *index.html* fait appel à *axios.js* et *comportements.js* grâce à deux balises script placées dans la balise *head*.

Comportements.js utilise la fonction *get* pour exécuter un appel Ajax et récupérer ainsi des données fournies par un serveur web.

Les données sont récupérées à l'adresse :

https://prodrigu.lpmiaw.univ-lr.fr/miaw/js_wdi/api_personnes/recupererUnePersAuHasard/

Allez à cette adresse et observez la structure de la données récupérée.

Mettez à jour les données en utilisant la touche F5, et vérifiez que la donnée fournie n'est pas toujours la même.

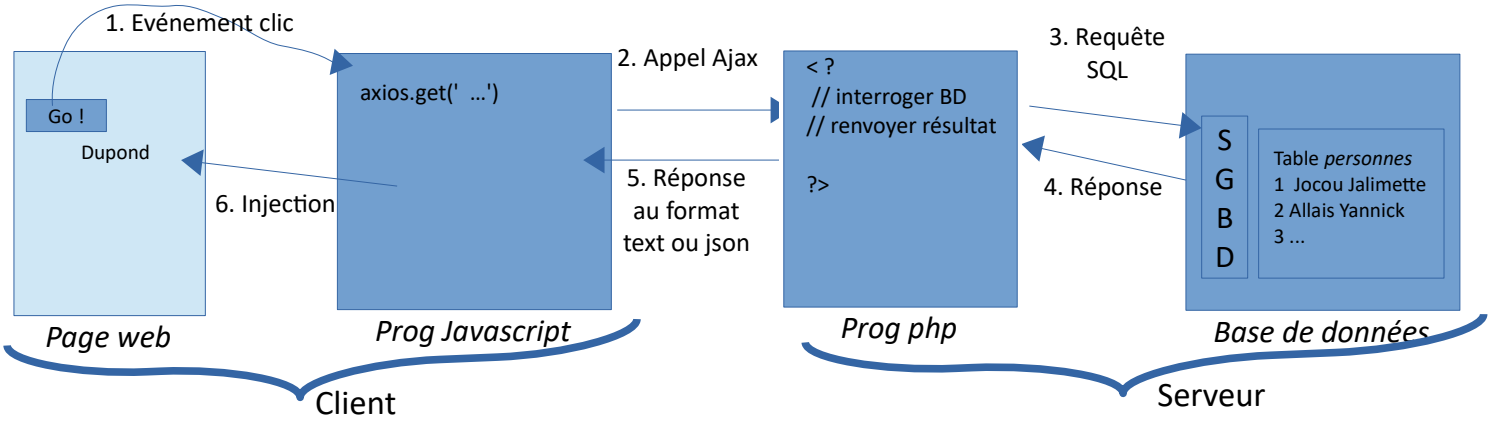
Vous allez compléter la fonction de traitement de la donnée reçue par le programme javascript en résultat du *axios.get*, à partir de la ligne 17.

Complétez le programme pour qu'il puisse :

- afficher la donnée reçue dans la console
- afficher seulement le nom de la personne dans la console
- placer ce nom dans la zone de résultat du premier article de la page web.

Modifiez le programme pour que l'appel Ajax soit déclenché seulement lorsque l'internaute appuie sur le premier bouton "Go !".

Le schéma suivant montre l'architecture de notre application et la chronologie des événements :



1. Un clic sur le bouton déclenche l'exécution d'une fonction qui effectue le *axios.get*
2. La fonction *get* lance l'exécution d'un programme php situé sur le serveur.
3. Le serveur interroge une base de données grâce à une requête SQL
4. Le Système de Gestion de Base de Données fournit une réponse à la requête
5. Le programme php récupère cette réponse et la met en forme, par exemple pour la présenter au format JSON
6. Le programme javascript récupère le résultat crée par le programme php et injecte ces informations dans la page web

B. Traitement d'un tableau d'objets

On veut qu'un clic sur le second bouton "Go !" permette d'afficher la liste des personnes fournie par https://prodrigu.lpmiaw.univ-lr.fr/miaw/js_wdi/api_personnes/recupererLesPers/

Avec votre navigateur, allez à cette adresse pour visualiser les données à traiter.

Mettez en place l'écouteur d'événement permettant de gérer le clic sur le bouton.

Créez la fonction de gestion de cet événement.

Dans cette fonction, effectuez l'appel à la fonction *axios.get* permettant d'aller chercher les données, et affichez le résultat dans la console.

Nous nommerons *lesPersonnes*, le résultat Json de cet appel Ajax.

Ce résultat est un tableau de données.

L'accès à la première personne du tableau s'effectue avec *donnees[0]*.

Affichez dans la console, puis dans la page, le nom et le prénom de la première personne de la liste.

Utilisez la fonction *forEach* pour parcourir l'ensemble des personnes, dans un premier temps, vous chercherez à afficher le nom de chaque personne dans la console.

Complétez le programme pour qu'il afficher les noms des personnes dans un paragraphe unique (vous pouvez ajouter du texte dans un élément avec *element.textcontent += "nouveau texte"*)

Faites en sorte que le programme crée une balise *<li>* pour chaque personne, contenant le prénom et le nom de la personne, et qu'il ajoute cette balise à la liste positionnée à la fin du second article.

C. Création d'une carte

Vous allez modifier le programme *carte.js* qui permet de créer une carte dans la page *carte.html.*

Le programme *carte.js* effectue un appel Ajax pour récupérer des données concernant les défibrillateurs de la ville de Toulouse.

Il crée une carte dans la page web, place un marqueur sur celle-ci.

Vous devez compléter ce programme afin qu'il ajoute à la carte, des marqueurs représentant les défibrillateurs récupérés grâce à l'appel Ajax.