

TP 5 JavaScript & JSON

Exercice 1. Création d'un tableau de bord à partir de données Json

- Télécharger le fichier "TP5-Json-Population-etu.html" sur Moodle, dossier TP5

Sujet : l'objectif est de télécharger des données Json à partir d'un site web et créer dynamiquement une page web qui affiche et met en forme ces données.

Les données sont disponibles sur le site gouvernemental « geo.api.gouv.fr », les données json concernant les populations par département : <https://geo.api.gouv.fr/departements/>

Le fichier *TP4 -Json-Population-etu.html* contient une description du css permettant la mise en forme de la page web, ainsi qu'un tableau html qui va accueillir le résultat de la requête. L'objectif est de permettre à l'utilisateur de taper le numéro d'un département en France, et qu'en cliquant sur le bouton API population, s'affiche toutes les communes du département avec la population de la commune (*cf. résultat attendu page suivante*).

Compléter donc ce fichier html avec un script javascript pour :

- Étape 1 :** Dans la **fonction rest()**, créer la requete *XMLHttpRequest* afin de se connecter au site et lancer la requête (cf. cours BUT3_VCOD R5.VCOD.07_ProgWebVis-Json.pdf – Diapo 11)
- Étape 2 :** Créer une **fonction gestion_requete()** qui va dynamiquement :
 - Vérifier si la requête a donné est OK au niveau de l'exécution (**.readyState**) et au niveau du résultat (**.status**), en cas de problème afficher : **alert("problème !")** ; :


```
if (vRequete.readyState === XMLHttpRequest.DONE) {
    //Test le status de la requête, si 200 --> OK, si 404 --> problème
    if (vRequete.status === 200) {
```
 - Récupérer le contenu de la requête dans une variable **var donnees** puis parser le résultat (**.parse**)
 - Trier par ordre décroissant de population les données récupérées après le parse :


```
// tri les données de la réponse par ordre décroissant de population
donnees.sort(function(key1,key2){return key1.population <
key2.population;})
```
 - Insérer les données récupérées triées dans le **<tbody id="rendu">**. Pour cela, faire une boucle sur toutes les communes du département avec :
 - . Insertion de nouvelle ligne : **.insertRow()** ;
 - . Insertion de nouvelle cellule : **.insertCell()** ;
 - . Mettre le contenu dans les nouvelles cellules (nom, codepostal et population) :
Exemple : **.textContent = donnees[i].nom** ;

3. Étape 3 : Compléter le script afin de :

- Limiter l'affichage des communes dans le tableau uniquement aux communes de moins de 3000 habitants.
- Afficher dans la zone de texte `<input type="text" id='popu_globale'/>` la valeur de la population totale du département sélectionné.

file:///Users/youzrout/Documents/Yacine/IUT/YO-Cours/IUT/SD/_Cours/_2023_Programmation Web Visualisation ☆ 🔍 Rechercher

API population 69
1883437

ville	classement	code postaux	population
Lyon	1	69001,69002,69003,69004,69005,69006,69007,69008,69009	522228
Villeurbanne	2	69100	154781
Vénissieux	3	69200	66765
Vaulx-en-Velin	4	69120	51761
Saint-Priest	5	69800	48318
Caluire-et-Cuire	6	69300	43355
Bron	7	69500	42442
Villefranche-sur-Saône	8	69400	36009
Meyzieu	9	69330	35134
Rillieux-la-Pape	10	69140	30887
Décines-Charpieu	11	69150	28913
Oullins	12	69600	26994
Tassin-la-Demi-Lune	13	69160	22475
Sainte-Foy-lès-Lyon	14	69110	22077
Saint-Genis-Laval	15	69230	20971
Givors	16	69700	20672
Saint-Fons	17	69190	19500
Écully	18	69130	18789
Francheville	19	69340	14821
Mions	20	69780	13684
Belleville-en-Beaujolais	21	69220	13336
Genas	22	69740	13195
Brignais	23	69530	12403
Craponne	24	69290	11453
Corbas	25	69960	11161
Chassieu	26	69680	10844
Tarare	27	69170	10572
Pierre-Bénite	28	69310	10508
Feyzin	29	69320	9926
Grigny	30	69520	9739
Chaponost	31	69630	8887
Dardilly	32	69570	8829
Irigny	33	69540	8805
Anse	34	69480	7754
Neuville-sur-Saône	35	69250	7635
Gleizé	36	69400	7495
Saint-Didier-au-Mont-d'Or	37	69370	7124
Fontaines-sur-Saône	38	69270	7065
Saint-Bonnet-de-Mure	39	69720	6876
Sathonay-Camp	40	69580	6678