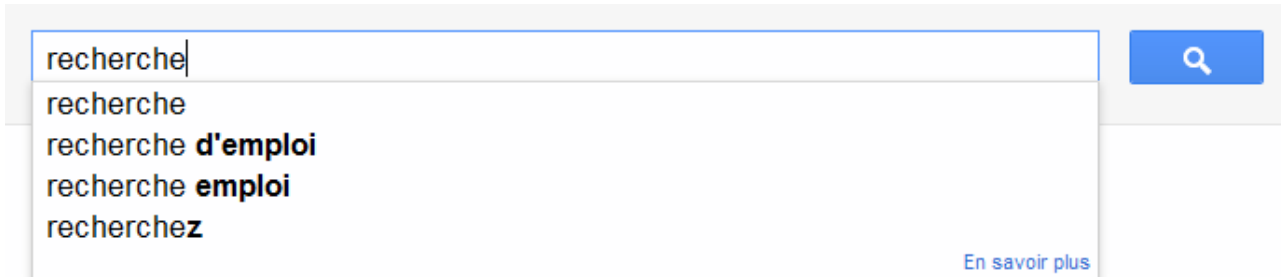


TP : AJAX – Auto-complétion

Nous allons implémenter un champ de recherche avec auto-complétion. Il utilisera 4 fichiers. **autocomplete.html** et **autocomplete.js** implémenteront la page web sur laquelle se trouve le champ de recherche. Cette page interrogera en AJAX **towns.php** qui cherchera les villes suggérées dans le fichier **towns.txt**.

Exemple d'auto-complétion :



Les questions les plus importantes pour votre progression sont numérotées **6 et 8**. Ce sont elles qui appliquent le cours sur AJAX et JSON. Celles d'avant sont importantes pour comprendre le fonctionnement global d'AJAX.

Côté serveur

1) unserialize() (temps conseillé : 10')

Ecris un fichier **towns.php** qui affiche la liste des villes contenues dans **towns.txt**. Tu pourras lire ce fichier avec **file_get_contents()**, puis convertir son contenu en tant que tableau PHP grâce à la fonction **unserialize()**.

Vérifie ton travail en affichant towns.php avec ton navigateur.

PHP n'est pas connu par le navigateur. Il faut passer par un serveur web : xampp par exemple.

2) Liste de suggestions (temps conseillé : 15')

towns.php recevra une chaîne de caractères dans `$_GET[]`. Ce sera ce que l'utilisateur tape dans le champ de recherche. Parcours le tableau des villes à la recherche de suggestions qui commencent par cette chaîne de caractères.

Tu pourras utiliser la fonction **strpos()**, qui permet de vérifier si une chaîne de caractères contient certains caractères, et ce sans tenir compte de la casse. Les suggestions trouvées seront mis dans un tableau grâce à la fonction **array_push()**.

Trie les suggestions avec la fonction **sort()**.

Vérifie ton travail en affichant towns.php dans ton navigateur et en lui fournissant différentes chaînes de caractères.

Il suffit pour cela de taper : `http://localhost/towns.php?s=ce_que_je_cherche`

3) Format des données (temps conseillé : 10')

Quel format choisir pour envoyer les données à **autocomplete.html** ? XML ? JSON ?

Ni l'un ni l'autre, nous allons commencer par le plus simple ici : du texte.

Pourquoi ? Pour une raison simple : le XML et le JSON sont utiles pour renvoyer des données qui ont besoin d'être structurées. Si nous avions eu besoin de renvoyer, en plus des noms de ville, le nombre d'habitants, de commerces et d'administrations françaises, alors nous aurions pu envisager l'utilisation du XML ou du JSON afin de structurer tout ça. Mais dans notre cas cela n'est pas utile, car nous ne faisons que renvoyer le nom de chaque ville trouvée.

Alors comment renvoyer tout ça sous forme de texte brut ? Nous pourrions faire un saut de ligne entre chaque ville retournée, mais ce n'est pas spécialement pratique à analyser pour le JavaScript. Nous allons donc devoir choisir un caractère de séparation qui n'est jamais utilisé dans le nom d'une ville. Dans ce TP, nous allons donc utiliser la barre verticale |, ce qui devrait nous permettre de retourner du texte brut sous cette forme :

```
Paris|Perpignan|Patelin-Paumé-Sur-Toise|Etc.
```

Tout comme **join()** en JavaScript, il existe une fonction PHP qui permet de concaténer toutes les valeurs d'un tableau dans une chaîne de caractères avec un ou plusieurs caractères de séparation : il s'agit de la fonction **implode()**. Une fois la fonction utilisée, il ne te reste plus qu'à retourner le tout au client avec un bon vieux `echo`.

Vérifie ton travail en affichant `towns.php` dans ton navigateur et en lui fournissant différentes chaînes de caractères.

Il suffit pour cela de taper : `http://localhost/towns.php?s=ce_que_je_cherche`

Côté client

4) `autocomplete.html` (temps conseillé : 10')

Crée un fichier **`autocomplete.html`** qui contient simplement un champ texte pour écrire les mots-clés. Cependant, ce dernier va nous poser problème, car le navigateur enregistre généralement ce qui a été écrit dans les champs de texte afin de vous le re-proposer plus tard sous forme d'auto-complétion, ce qui va donc faire doublon avec notre système... Heureusement, il est possible de désactiver cette auto-complétion en utilisant l'attribut `autocomplete` de cette manière :

```
<input type="text" autocomplete="off" >
```

Sous ce champ texte, ajoute une `<div id='suggestions'>` qui contiendra les suggestions.

En ce qui concerne un éventuel bouton de type `submit`, nous allons nous en passer, car notre but n'est pas de lancer la recherche, mais seulement d'afficher une auto-complétion.

5) `autocomplete.js` : réagir à chaque relâchement de touche (temps conseillé : 10')

Crée un fichier `autocomplete.js` lié à `autocomplete.html`.

Fais en sorte que lorsque on tape des lettres dans le champ texte, ces lettres soient copiées au fur et à mesure dans `<div id="suggestions">`.

Exemple :



1. Utilise l'événement « `keyup` »

6) `XMLHttpRequest` (temps conseillé : 20')

Au lieu d'afficher les caractères tapés, nous voulons maintenant afficher le résultat brut de la recherche :

p
Palaiseau|Pantin|Paris|Pau|Perigueux|Perpignan|Plaisir|Poitiers|Pontault-Combault|Puteaux

1. Lis l'exemple simple du cours
2. Ecris une fonction `getSuggestions(lettres_tapées)` qui va interroger `towns.php` en AJAX, puis afficher la réponse.
3. `getSuggestions()` est appelée par le gestionnaire d'évènement de `keyup`.

7) Les suggestions listées verticalement (temps conseillé : 10')

Affiche la liste des villes dans `<div id='suggestions'>`. Sous forme d'une succession de `div`. Pour obtenir par exemple :

ni|
Nice
Nîmes

1. Utilise `split('|')` pour découper la réponse fournie par `towns.php`.
2. Si tu tapes « `nic` », est-ce que « `Nîmes` » disparaît ?

8) JSON (temps conseillé : 20')

Sauvegarde ton travail puis modifie `towns.php` puis `autocomplete.js` pour que les suggestions soient maintenant transmises sous forme de JSON.

Vérifie ton travail côté `towns.php` en affichant par exemple : <http://localhost/at03/towns.php?s=pa>

9) Flèches « haut » et « bas » (optionnel)

Quel format préfères-tu ici ? JSON ou texte brut ? Choisis le format que tu préfères.

Fais maintenant en sorte que lorsqu'une liste de suggestions est proposée, on puisse en mettre une en gras en appuyant sur les flèches « haut » ou « bas ». Exemple :

or|
Orleans
Only
Orvault

1. Tu pourras utiliser une variable `maSelection` qui indique quelle suggestion est sélectionnée :
 - 2 signifie qu'il n'y a pas de liste affichée
 - 1 signifie "aucune sélection"
 - 0 signifie "premier élément de la liste sélectionné"
2. Il faut ajouter des conditions dans le gestionnaire d'évènement de « `keyup` ».
3. Comment savoir quelle touche a été relâchée ?
4. Relis le cours sur les évènements (« Cours 2 – JavaScript et HTML »).
5. Il faut utiliser `keyCode`. Code pour les flèches : « haut » : 38 et « bas » : 40.
6. Pour mettre en gras une `<div>` il suffit de la mettre dans la classe CSS « `selection` »
7. On change la classe CSS avec l'attribut `className`.

10) Sélection finale (optionnel)

Lorsque l'utilisateur presse la touche « Droite » ou clique avec la souris sur une suggestion, on veut que la suggestion soit copiée dans le champ texte et que les suggestions disparaissent.

Après avoir pressé la touche « droite », si l'utilisateur continue à écrire, d'autres suggestions doivent apparaître.

Ce TP est largement inspiré du MOOC d'OpenClassrooms « Dynamisez vos sites web avec JavaScript »

<https://openclassrooms.com/courses/dynamisez-vos-sites-web-avec-javascript/tp-un-systeme-d-auto-completion-1>

