JavaScript - AJAX

APPLICATION PROGRAMMING INTERFACE & FETCH

Léo LAROU-CHALOT
IPSSI | 25 RUE CLAUDE TILLIER, 75012 PARIS

Table des matières

Le contexte

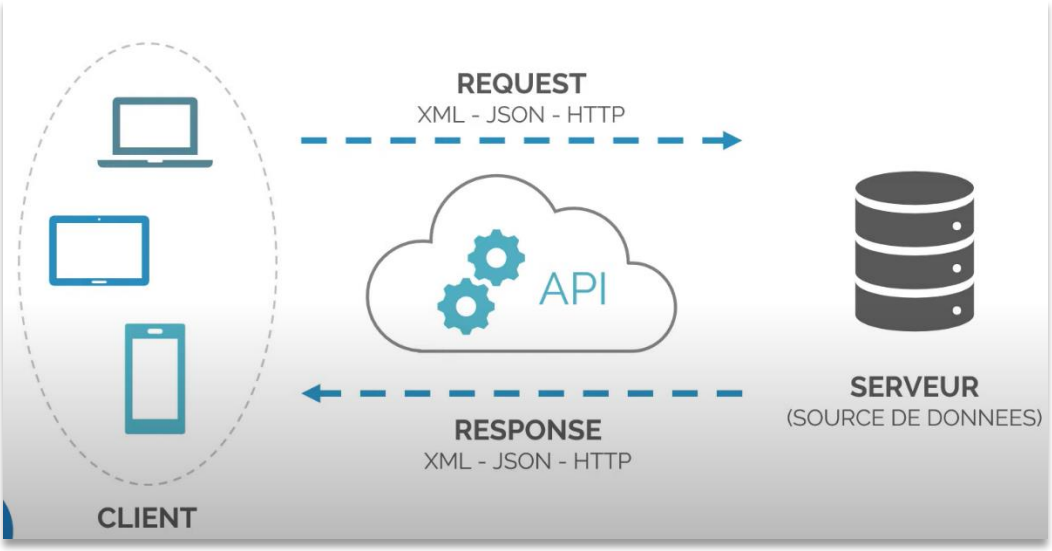
Le projet

Dans cet exemple nous allons réaliser une recherche de ville/population par code postales, cela nous permettra d'utiliser une API publique afin de parvenir au bon résultat

Qu'est-ce qu'une API ?

Selon une définition trouvée lors de mes recherches.

Une API (Application Programming Interface) est un ensemble normalisé de classes, de méthodes, de fonctions et de constantes qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels.



Source : JavaScript | API & Fetch (youtube)

Grâce à JavaScript, il nous est possible d'utiliser ces API pour mettre à jour du contenu sur nos page web, et ce, en évitant d'avoir à recharger ou rafraîchir notre navigateur. C'est avec l'utilisation d'AJAX (Asynchronous JavaScript + XML) que cela sera possible.

Fetch

Dans ce cours, nous apprendrons à utiliser l'API Fetch.

L'API Fetch fournit une interface JavaScript pour l'accès et la manipulation du pipeline HTTP, comme les requêtes et les réponses.

Fetch est une nouvelle méthode apparue avec ES6, et qui nous permet l'utilisation de l'AJAX.

Cela va nous permettre de créer des requêtes http, et de gérer les réponses en retour.

L'écriture de la méthode Fetch

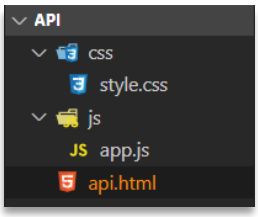
Voici le schéma simplifié de l'écriture de cette méthode :

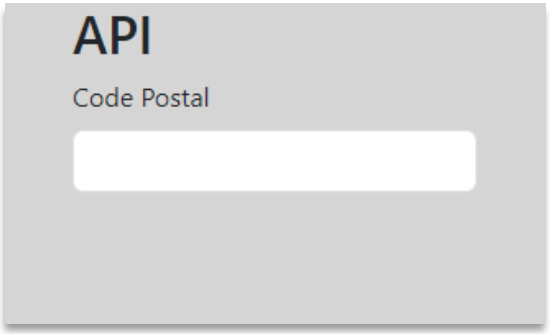
```
fetch('http://www.site.com/users')
  .then(function(response){
    return response.json()
  })
  .then(function(data){
    console.log(data);
  })
```

1) Fetch prend l'URL à contacter pour recueillir des datas.
2) Une fois l'URL contactée, on récupèrera une « Promesse », Cela explique l'utilisation de « .then » (Cf cours sur les promesses).
3) Les données sont ensuite formatées (ou typées) au format JSON (Javascript Object Notation) puis retournées.
4) Le deuxième « .then » est une nouvelle promesse contenant les « datas » qui vont pouvoir être traitées.

Codes, explications et manipulations

Présentation de l'espace de travail






```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD" crossorigin="anonymous">
  <link rel="stylesheet" href="css/style.css">
  <title>API - CP</title>
</head>
<body>

  <div class="container">
    <div class="row justify-content-center">
      <div class="col-12 col-sm-6">
        <h1>API</h1>
        <form>
          <div class="form-group">
            <label for="cp" class="form-label">Code Postal</label>
            <input type="text" maxlength="5" class="form-control" name="cp">
          </div>
        </form>
      </div>
    </div>
  </div>

  <script src="js/app.js"></script>
</body>
</html>
```

Source | Github API publiques

Source | API Gouv

Après passage sur le site cité ci-dessus : nous obtenons un résultat de requête de ce genre, notre matière pour commencer le travail ! :

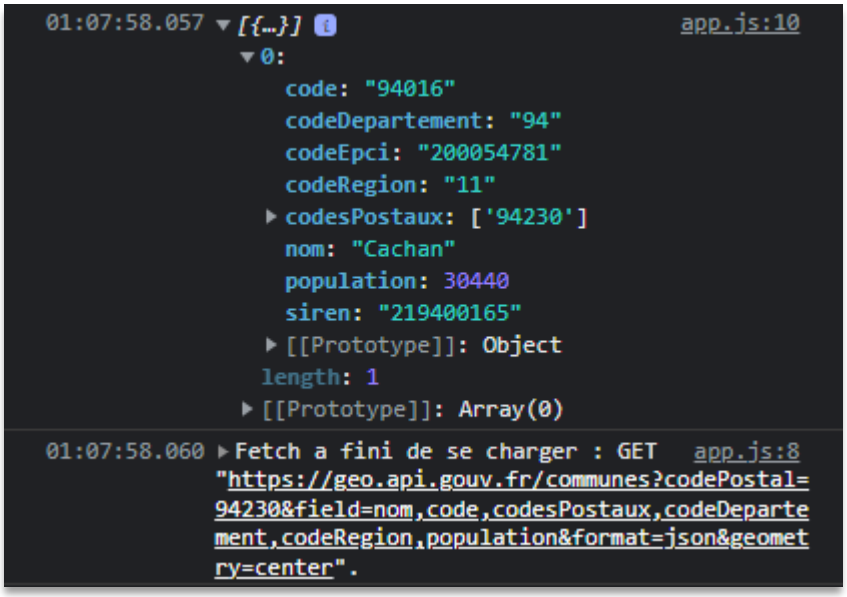

```
[{"nom":"Cachan","code":"94016","codeDepartement":"94","siren":"219400165","codeEpci":"200054781","codeRegion":"11","codesPostaux":["94230"],"population":30440}]
```

Réalisation

Une fois la mise en place de l'espace de travail, notre fichier « app.js » ressemblera à cela :

```
// ? Utilisation de l'écoute "input" pour chaque frappe dans la barre de recherche
document.querySelector("#cp").addEventListener("input", function () {
  // ? Si les 5 caractères du CP on été saisis
  if (this.value.length == 5) {
    // ? Début de l'échange avec l'API
    let url = `https://geo.api.gouv.fr/communes?codePostal=${this.value}&field=nom,
    code,codesPostaux,codeDepartement,codeRegion,population&format=json&
    geometry=center`;

    // ? Début de la communication avec le serveur
    fetch(url)
      .then(response => response.json())
      .then(data => console.log(data));
  }
});
```

Via le navigateur, on remarquera la récupération de notre réponse !

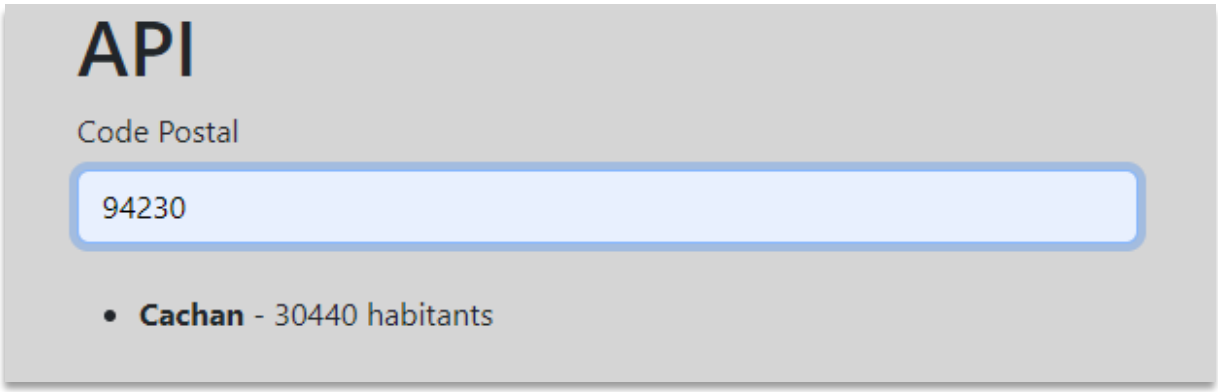


Codes source et résultat

Ci-joint le code source et le rendu final

```html
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.
      css"
      rel="stylesheet"
      integrity="sha384-GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/
      Jr59b6EGGoI1aFkw7cmDA6j6gD"
      crossorigin="anonymous"
    />
    <link rel="stylesheet" href="css/style.css" />
    <title>API - CP</title>
  </head>
  <body>
    <div class="container">
      <div class="row justify-content-center">
        <div class="col-12 col-sm-6">
          <h1>API</h1>
          <form>
            <div class="form-group">
              <label for="cp" class="form-label">Code Postal</label>
              <input type="text" maxlength="5" class="form-control mb-4" name="cp"
              id="cp" />
            </div>
          </form>
          <div id="ville"></div>
        </div>
      </div>
    </div>

    <script src="js/app.js"></script>
  </body>
</html>
```

```js
// ? Utilisation de l'écoute "input" pour chaque frappe dans la barre de recherche
document.querySelector("#cp").addEventListener("input", function () {

  // ? Si les 5 caractères du CP on été saisis
  if (this.value.length == 5) {

    // ? Début de l'échange avec L'API
    let url = `https://geo.api.gouv.fr/communes?codePostal=${this.value}&field=nom,
    code,codesPostaux,codeDepartement,codeRegion,population&format=json&
    geometry=center`;

    // ? Début de la communication avec le serveur
    fetch(url)
      .then((response) =>
        response.json().then((data) => {
          console.log(data);
          let affichage = "<ul>";
          for (let ville of data) {
            affichage += `<li><strong>${ville.nom}</strong> - ${ville.population}
            habitants</li>`;
          }
          affichage += "</ul>";
          document.querySelector("#ville").innerHTML = affichage;
        })
      )
      .catch((erreur) => console.log("Erreur : " + erreur));
  }
});
```



Conclusion

Fetch a permis l'utilisation de l'AJAX pour ce projet, il s'agit là de la suite logique du cours sur les « promesses » en JavaScript