



Javascript API call workshop

⌚ Date de création @January 19, 2023 4:57 PM

↗ Module

▼ Installer Node

▼ Pour Ubuntu :

Ouvrez un terminal et mettez à jour les paquets en utilisant la commande :

```
sudo apt update
```

Installez les dépendances nécessaires pour installer Node.js en utilisant la commande :

```
sudo apt install curl software-properties-common
```

Ajoutez le dépôt de Node.js en utilisant la commande :

```
curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash -
```

Installez Node.js en utilisant la commande :

```
sudo apt install nodejs
```

Vérifiez que Node.js est installé correctement en utilisant la commande :

```
node -v
```

▼ Pour Fedora :

Ouvrez un terminal et mettez à jour les paquets en utilisant la commande :

```
sudo dnf update
```

Ajoutez le dépôt de Node.js en utilisant la commande :

```
curl -sL https://rpm.nodesource.com/setup_14.x | sudo bash -
```

Installez Node.js en utilisant la commande :

```
sudo dnf install nodejs
```

Vérifiez que Node.js est installé correctement en utilisant la commande :

```
node -v
```

Il est important de noter que ces instructions d'installation sont pour Node.js version 14.x , si vous voulez installer une version différente, il suffit de remplacer le numéro de version dans les commandes.

▼ Exercices

▼ Exercice 1: Récupérer des informations météos

Weather Forecast API

Open-Meteo weather forecast APIs use weather models from multiple national weather providers. For each location worldwide, the best models will be combined to provide the best possible forecast. Weather models cover different geographic areas at different resolutions and provide different weather variables.

 <https://open-meteo.com/en/docs>

L'objectif de cet exercice est de vous familiariser avec axios afin de pouvoir récupérer des informations sur la météo. Pour valider cet exercice, vous devrez utiliser l'api du lien ci-dessus dans votre script afin d'afficher les informations météorologiques de votre ville. Voici une courte définition d'axios et de son utilité lors de cet exercice :

Axios est un module JavaScript qui permet de faire des requêtes HTTP (comme des requêtes GET, POST, PUT, etc.) facilement. Il est basé sur la bibliothèque Promise et prend en charge les navigateurs modernes ainsi que les environnements Node.js. Axios peut être utilisé pour envoyer des requêtes à une API REST, pour télécharger ou envoyer des fichiers, pour gérer les erreurs, les intercepteurs, etc. Il est très populaire et facile à utiliser.

Getting Started

Promise based HTTP client for the browser and node.js Axios is a promise-based HTTP Client for and the browser. It is isomorphic (= it can run in the browser and nodejs with the same codebase). On the server-side it uses the native node.js http module, while on the client (browser) it uses XMLHttpRequests.

 <https://axios-http.com/docs/intro>

▼ Réponse

```
import axios from 'axios';

const url = 'https://api.open-meteo.com/v1/forecast?latitude=43.70&longitude=7.27&hourly=temperature_2m';
const data = {
  title: 'My title',
  body: 'Premier Post de Morgan Wolff',
  userId: 1
};

async function getinfos() {
  axios.get(url)
    .then(response => {
      console.log("data: " + JSON.stringify(response.data));
    })
    .catch(error => {
      console.log("Error: " + error);
    });
}

async function launchScript() {
  await getinfos();
}

launchScript();
```

▼ Exercice 2: Poster des informations sur une base de données

json-server

Get a full fake REST API with zero coding in less than 30 seconds. Latest version: 0.17.1, last published: 3 months ago. Start using json-server in your project by running `npm i json-server`. There are 300 other projects in the npm registry using

 <https://www.npmjs.com/package/json-server>



Ce deuxième exercice nécessite l'utilisation d'une nouvelle requête d'axios, vous allez découvrir le POST. La méthode "post" d'Axios permet de faire une requête HTTP de type "post" qui envoie des données au serveur pour être traitées. Cette méthode prend en entrée l'URL cible, les données à envoyer et des options (comme les entêtes) et renvoie une promesse qui se résout avec la réponse du serveur. Elle est utilisée pour envoyer des données au serveur pour créer ou mettre à jour des ressources.

1. Assurez-vous d'avoir Node.js et npm (le gestionnaire de paquets de Node.js) installés sur votre ordinateur.
2. Créez un nouveau répertoire pour votre projet et naviguez jusqu'à ce répertoire dans votre terminal.
3. Installez le package `json-server` en utilisant la commande `npm install -g json-server`. Cela installera le serveur JSON globalement sur votre ordinateur, de sorte que vous pourrez l'utiliser pour n'importe quel projet.
4. Créez un fichier `db.json` dans votre répertoire de projet. Ce fichier sera utilisé comme base de données pour votre serveur JSON. Il peut contenir toutes les données que vous souhaitez exposer via l'API.
5. Lancez le serveur JSON en utilisant la commande `json-server --watch db.json` dans votre terminal. Cela démarra le serveur sur `http://localhost:3000` et il utilisera votre fichier `db.json` comme base de données.

```
{
  "posts": [
    { "id": 1, "title": "json-server", "author": "typicode" }
  ],
  "comments": [
    { "id": 1, "body": "some comment", "postId": 1 }
  ],
  "profile": { "name": "typicode" }
}
```

6. Utilisez Axios pour effectuer des requêtes HTTP au serveur JSON. Par exemple, pour effectuer une requête `POST` pour envoyer des éléments dans votre base de données, vous pourriez utiliser la commande suivante:

```
axios.post(url, data)
  .then(response => {
    console.log(response.data);
  })
  .catch(error => {
    console.log(error);
  });

```

▼ Réponse:

```
import axios from 'axios';

const url = 'http://localhost:3000/posts/';
const data = {
  title: 'My title',
  body: 'Premier Post de Morgan Wolff',
  userId: 1
};

async function postinfos() {
  axios.post(url, data);
  .then(response => {
    console.log("Successfully Posted");
  })
  .catch(error => {
    console.log(error);
  });
}

async function launchScript() {
  await postinfos();
}

await launchScript();
```

▼ Exercice 3: Supprimer un post sur la base de données

Maintenant que vous avez posté avec succès vos infos sur la base de données, il vous reste plus qu'à le supprimer. Pour ça, vous devez vous y prendre exactement pareil que pour les exercices précédents, mais vous allez devoir utiliser une autre requête d'Axios vous permettant de supprimer votre précédent post.



Pensez à indiquer l'id du post que vous souhaitez supprimer.

▼ réponse

```
import axios from 'axios';

const idtosupp = POST_ID
const url = 'http://localhost:3000/posts/' + idtosupp;
const data = {
    title: 'My title',
    body: 'Premier Post de Morgan Wolff',
    userId: 1
};

async function deleteinfos() {
    axios.delete(url);
    .then(response => {
        console.log("Successfully Deleted");
    })
    .catch(error => {
        console.log(error);
    });
}

async function launchScript() {
    await deleteinfos();
}

await launchScript();
```