1 API RESTFUL

1.1 UTILISATION D'UNE API RESTFUL SIMPLE

L'application suivante va simplement se connecter à l'API REST du site www.prevision-meteo.ch

Le site fournit une documentation de son service : https://www.prevision-meteo.ch/services

Parmi les méthodes disponibles :

1.1.1 Récupération d'une image

En indiquant la ville retenue, le site fournit une image par le biais d'un code HTML

```
<img src="https://www.prevision-meteo.ch/uploads/widget/grenoble_0.png" width="650"
height="250" />
```

L'inconvénient est que les données de températures ou de précipitations ne sont pas récupérables par un logiciel.

1.1.2 Récupération des données JSON

Cette méthode propose uniquement un lien et nécessite de lire le guide d'utilisation, mais est beaucoup plus intéressante :

https://www.prevision-meteo.ch/services/json/grenoble

Les données JSON sont généralement lisibles de manière simple dans tous les navigateurs :

Les

- city_info:
  - name: "Grenoble"
  - country: "France"
  - latitude: "45.1850014"
  - longitude: "5.7227778"
  - elevation: "212"
  - sunrise: "07:32"
  - sunset: "19:22"
- forecast_info: {…}
- current_condition:
  - date: "30.09.2019"
  - hour: "10:00"
  - tmp: 19
  - wnd_spd: 5
  - wnd_gust: 9
  - wnd_dir: "N"
  - pressure: 1012.4
  - humidity: 68
  - condition: "Ensoleillé"
  - condition_key: "ensoleille"
  - icon: "https://www.prevision-meteo.ch/style/images/icon/ensoleille.png"
  - icon_big: "https://www.prevision-meteo.ch/style/images/icon/ensoleille-big.png"

données brutes seront utilisées par les applications :


```
{"city_info":
{"name":"Grenoble","country":"France","latitude":"45.1850014","longitude":"5.7227778","ele
vation":"212","sunrise":"07:32","sunset":"19:22"},"forecast_info":
{"latitude":null,"longitude":null,"elevation":"478.0"},"current_condition":
{"date":"30.09.2019","hour":"10:00","tmp":19,"wnd_spd":5,"wnd_gust":9,"wnd_dir":"N","press
ure":1012.4,"humidity":68,"condition":"Ensoleill\
u00e9","condition_key":"ensoleille","icon":"https:\/\/www.prevision-meteo.ch\/style\/
images\/icon\/ensoleille.png","icon_big":"https:\/\/www.prevision-meteo.ch\/style\/
images\/icon\/ensoleille-big.png"},"fcst_day_0":
{"date":"30.09.2019","day_short":"Lun.","day_long":"Lundi","tmin":15,"tmax":24,"condition"
:"Ensoleill\u00e9","condition_key":"ensoleille","icon":"https:\/\/www.prevision-
meteo.ch\/style\/images\/icon\/ensoleille.png","icon_big":"https:\/\/www.prevision-
meteo.ch\/style\/images\/icon\/ensoleille-big.png","hourly_data":{"0H00":
{"ICON":"https:\/\/www.prevision-meteo.ch\/style\/images\/icon\/nuit-
claire.png","CONDITION":"Nuit claire","CONDITION_KEY":"nuit-
```

1.1.3 Création de l'API en Node.JS

Node.JS permet de se connecter (en tant que client) à une API web. Il faut utiliser un module appelé node-rest-client, dont le rôle est de fournir les méthodes d'accès aux serveurs web.

```
npm install node-rest-client
```

Ensuite, vous pouvez tester ce code :

```
let Client = require('node-rest-client').Client;
/*
  Zone d'insertion du code pour l'intégration du proxy
*/
let client = new Client();

client.registerMethod("jsonMethod",
"https://www.prevision-meteo.ch/services/json/seyssinet-pariset", "GET");

client.methods.jsonMethod(function (data, response) {
    // parsed response body as json object
    console.log(data);
      console.log('====================================');
      console.log(`actuellement à ${data.city_info.name} il fait $
{data.current_condition.condition}.`);
    // raw response
    // console.log(response);
});
```

Si le code ne fonctionne pas au lycée, c'est parce qu'il y a un serveur proxy, il faut donc modifier le code et y inclure le code suivant :

```
let Client = require('node-rest-client').Client;
// configure proxy
var options_proxy = {
    proxy: {
        host: "172.16.0.1",
        port: 3128,
        tunnel: true  // false = direct connexion
    }
};

let client = new Client(options_proxy);
```

1.2 Création d'une API RESTFUL

Le tutoriel suivant permet de comprendre comment générer des données au format JSON et les renvoyer au client.

Le plus simple est de créer des routes ayant un chemin racine avec le mot-clé 'api'. Voici un exemple simple :

```
const users = [
 { id: 1, name: "Jane Doe", age: "22" },
 { id: 2, name: "John Doe", age: "31" }
];

app.get("API/users", (req, res) => {
  try {
    res.status(200).json({
    users
    });
  } catch (error) {
    res.status(500).json({
    message: "Failed to retrieve all users",
    });
  }
});
```

Il y a deux informations à retourner :

- Les données au format JSON
- Le statut de la requête

Dans le cas où la requête fonctionne, on renvoie les données ET son statut (200 = succès).

Si un problème survient (accès trop long à la base par exemple), c'est un message ET le statut d'erreur serveur (500).

Pour aller plus loin sur les différentes routes, voici deux tutoriels intéressant à étudier.

https://medium.com/swlh/how-to-create-a-simple-restful-api-in-node-js-ae4bfddea158

https://www.makeuseof.com/beginners-guide-to-restful-apis-in-nodejs/