college CDI

Data processing technologies (TTD)

Class 08

Third party APIs

As we said earlier, third party APIs are provided by companies allowing to access some of their data using JavaScript on a web page. A very good example is Mapquest or Google map making it possible to show custom maps on a web page.

Accessing third party APIs

Although most of the APIs have their particularities, they also generally offer common features and work more or less the same way. Here is a guideline to use most of the available APIs.

Find an API

Of course, you will first have to explore and find the proper API for your needs. For this example, we will use a current weather API from https://openweathermap.org/.

Get a developer key

Most APIs require you to use an ID key for security reasons and accountability. You may have to create an account first in order to get your API key.

An API key will look like something like this :
k40b8d48e1yr53a86e25a3382957f5cf

Test the endpoint

The end point is the page containing the data you will be able to use (usually in JSON). Test the end point in a browser.

APIs usually have different endpoints and different ways for reaching them. For instance, the API we are now testing, among others, has one endpoint for each city. You can get all the forecasts using a city name, geographical coordinates, etc.

Error message :

{"cod": 429,"message": "Your account is temporary blocked due to exceeding of requests limitation of your subscription type. Please choose the proper subscription http://openweathermap.org/price"
}

JSON format content :

{"coord":{"lon":-73.59,"lat":45.51},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"}],"main":{"temp":269.45,"feels_like":266.05,"temp_min":268.15,"temp_max":270.93,"pressure":1010,"humidity":85},"visibility":14484,"wind":{"speed":1,"deg":90},"clouds":{"all":90},"dt":1580814983,"sys":{"type":1,"id":820,"country":"CA","sunrise":1580818346,"sunset":1580853840},"timezone":-18000,"id":6077243,"name":"Montreal","cod":200}

Parsing content in your browser

Reading raw data can be quite awful. In order to have your JSON data look good on screen, you can use JSONview extension (https://jsonview.com/).

Endpoint for general forecasts :

http://api.openweathermap.org/data/2.5/forecast?id=524901&APPID=c40bk-40b8d48e1yr53a86e25a3382957f5cf

Endpoint for Montreal forecast using city name :

http://api.openweathermap.org/data/2.5/**weather?q=montreal**,ca&APPID=k-40b8d48e1yr53a86e25a3382957f5cf

Endpoint for Montreal forecast specifying the metric units:

http://api.openweathermap.org/data/2.5/weather?q=montreal,ca**&units=metric**&APPID=k40b8d48e1yr53a86e25a3382957f5cf

Parsed data :


```
{
"coord": {
"lon": -73.59,
"lat": 45.51
},
"weather": [
{
"id": 804,
"main": "Clouds",
"description": "overcast clouds",
"icon": "04n"
}
],
"base": "stations",
"main": {
"temp": -3.75,
"feels_like": -7.15,
"temp_min": -5,
"temp_max": -2.78,
"pressure": 1010,
"humidity": 85
},
"visibility": 14484,
"wind": {
"speed": 1,
"deg": 90
},
"clouds": {
"all": 90
},
"dt": 1580816231,
"sys": {
"type": 1,
"id": 820,
"country": "CA",
"sunrise": 1580818346,
"sunset": 1580853840
},
"timezone": -18000,
"id": 6077243,
"name": "Montreal",
"cod": 200
}
```


Find and read the documentation

When you want to use a third party's API, one of the very first step to take is to find and read it's documentation. This is how you will find out about the API's features and how to use it.

Loading JSON data into a web document

As you already know, importing data from another server requires an AJAX request. Although, using jQuery, it is possible to use the *getJSON()* function.

Loading JSON data and displaying text content to a web document

As you already know, importing data from another server requires an AJAX request. Although, using jQuery, it is possible to use the *getJSON()* function. So, in your web document containing the jQuery CDN and between <script> tags or using an external JavaScript file, make you request.

```
<div id="map"> </div>

<script type="text/javascript">
$.getJSON("http://api.openweathermap.org/data/2.5/weather?q=montreal,ca&units=metric&APPID=c40b9d48e1ae53a86e25a3382957f4cf", function(data) {
        $("#map").html(data.name);
});
</script>
```

Explanation :
In the example above, *getJSON()* function is used to make an *AJAX request* allowing to import the content and to store it in the variable *data*. Using this last variable's name as prefix, it becomes easy to access the available content. The jQuery instruction finally writes the data in the tag containing *id="map"*.

Displaying an icon to a web document

If you take a look to the image beside, you will notice the second element ***(weather)*** stores an array with only one object ***(weather[0])*** containing four elements *(id, main, description* and *icon)*. It's the icon that interests us here.

If you code ***weather[0].icon*** to display the content, the result will be a string, the name of the PNG icon : *04n*.


```
{
  ▾ coord: {
        lon: -73.59,
        lat: 45.51
    },
  ▾ weather: [
      ▾ {
            id: 804,
            main: "Clouds",
            description: "overcast clouds",
            icon: "04n"
        }
    ],
    base: "stations",
```


Knowing that icons are stored at the URL *http://openweathermap.org/img/wn/*, you should then store the entire path to the icon in a variable and use concatenation to code the image in your web document like in the example below.

```
let icon = "http://openweathermap.org/img/wn/" + data.weather[0].icon + "@2x.png";

$("#map").append('<img  src="' + icon + '" />');
```