

Activity 1:

Fetch API is an interface for fetching objects. It is an alternate to XMLHttpRequest.

The Fetch is used as below,

Fetch (URL).then

```
(function(response){  
    console.log("fetched "+response);  
})
```

If response is received,
then do something

```
.catch(function(error){  
    console.log("Error: "+error);  
});
```

If error is returned, Display
appropriate message

By default, the HTTP request is a GET. We can use other types of request's perusal. The method name should be passed in a JSON with method as key and its value as the method name.

References: "[Working with the Fetch API | Web | Google Developers](#)"

Activity 2

In this activity, I learnt what asynchronous programming means.

I understood that this is a concept where the execution of one statement is not dependent on successful execution of a previous statement.

This paradigm was demonstrated using an example which helped me understand the concept clearly. Below I have explained my takeaways from the example.

I have divided the program into segments.

```
Activity2-ajax-JSON.html X
Activity2-ajax-JSON.html > html > head > script > openJSON
1 <html>
2 <head>
3 <title> ITC5202 - Week 10 </title>
4 <script>
5 function openJSON(){
6     xmlhttp = new XMLHttpRequest();
7     xmlhttp.onreadystatechange = function() {
8         if (this.readyState == 4 && this.status == 200)
9             myFunction(this);
10    }
11    xmlhttp.open("GET", "Activity2-json_data.json", true);
12    xmlhttp.send();
13 }
14 function myFunction(Data) {
15     var myObj = JSON.parse(Data.responseText);
16     document.getElementById("demo").innerHTML = myObj.name;
17 }
18 }
19 </script>
20 </head>
21
22
23 <body onload=openJSON()>
24 <p id="demo"></p>
25
26 </body>
27
28 </html>
29
```

Once the XMLHttpRequest object is created at line 6, the readystate attribute value is initialized to 0.

Next, the control goes to line 11 because Line 7 does not execute the inline function as the onreadystatechange event has not been triggered.

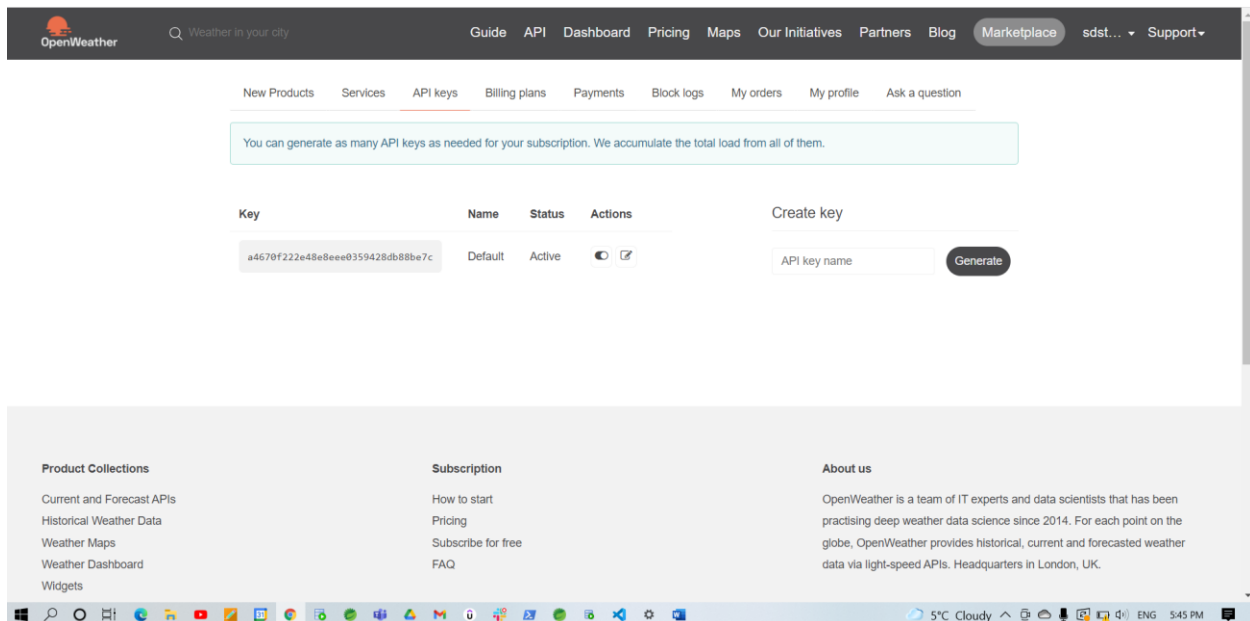
At line 11, when the open function is called, the readystate value changes to 1 and the event onreadystatechange is triggered so control goes to line 7. The if condition is not satisfied so control returns to line 12 now.

Once send function is executed, the readystate value changes to 2 and the event onreadystatechange is triggered so control goes to line 7. The if condition is not satisfied so control returns to line 13 now.

Once the response body is being received, the readystate value changes to 3 and the event onreadystatechange is triggered so control goes to line 7. The if condition is not satisfied so control returns back to line 13 now.

Now once the entire response body is received, readystate value changes to 4 and the event onreadystatechange is triggered so control goes to line 7. The if condition is satisfied now so line 9 is executed. The xhr object is passed to myFunction and the function is executed.

Activity 3:



The screenshot shows the OpenWeather API dashboard. At the top, there's a navigation bar with links like Guide, API, Dashboard, Pricing, Maps, Our Initiatives, Partners, Blog, Marketplace, and Support. Below this, there's a section for API keys with a table listing generated keys. The table has columns for Key, Name, Status, and Actions. One key is listed with the name 'Default' and status 'Active'. To the right of the table is a 'Create key' button. Below the table, there's a 'Product Collections' section with links to Current and Forecast APIs, Historical Weather Data, Weather Maps, Weather Dashboard, and Widgets. There's also a 'Subscription' section with links to How to start, Pricing, Subscribe for free, and FAQ. Finally, there's an 'About us' section with a brief description of OpenWeather.

Key	Name	Status	Actions
a4670f222e48e8eee0359428db88be7c	Default	Active	Edit

Create key

API key name

Product Collections

- Current and Forecast APIs
- Historical Weather Data
- Weather Maps
- Weather Dashboard
- Widgets

Subscription

- How to start
- Pricing
- Subscribe for free
- FAQ

About us

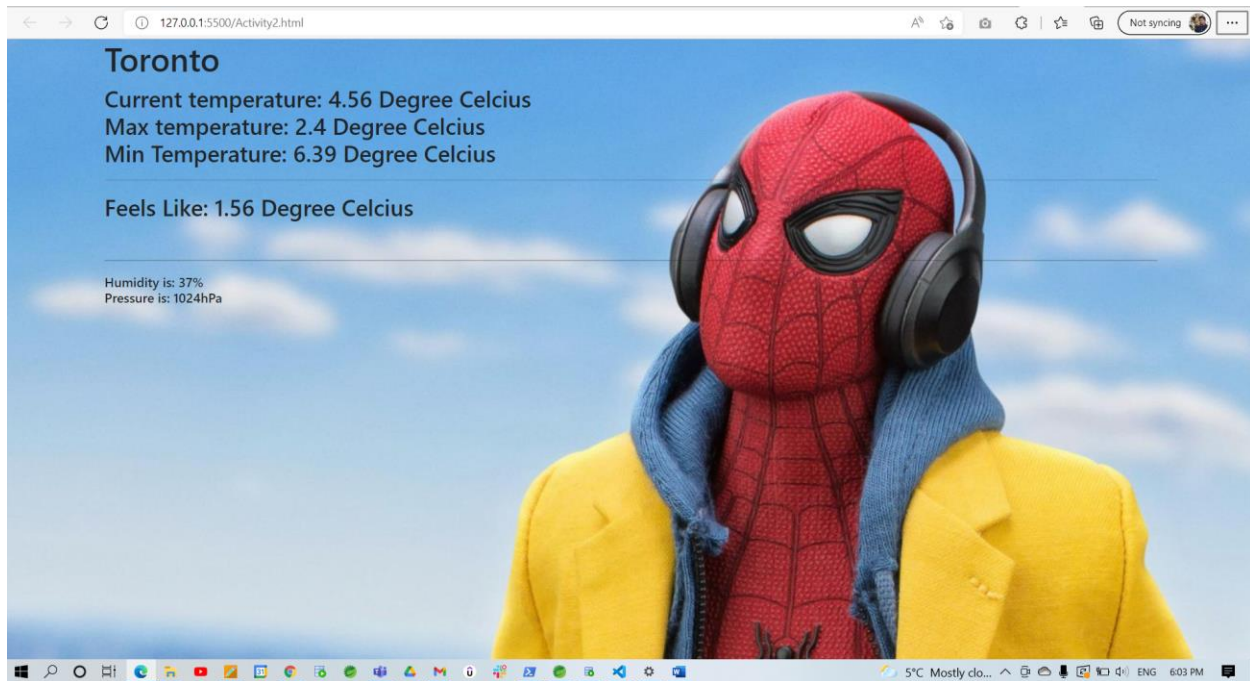
OpenWeather is a team of IT experts and data scientists that has been practising deep weather data science since 2014. For each point on the globe, OpenWeather provides historical, current and forecasted weather data via light-speed APIs. Headquarters in London, UK.

I created an account with openweather. And I received my API key.

I used the API to get the weather of Toronto in XML format.

The endpoint I used is:

<http://api.openweathermap.org/data/2.5/weather?q=Toronto&APPID=2e59fd0b2130921b24b3cee7923ba4d5&mode=xml&units=metric>



The screenshot shows a web application displaying weather data for Toronto. The data includes current temperature (4.56 Degree Celcius), max temperature (2.4 Degree Celcius), min temperature (6.39 Degree Celcius), and feels like (1.56 Degree Celcius). It also shows humidity (37%) and pressure (1024hPa). The background features a Spider-Man character wearing a yellow jacket and headphones.

Toronto

Current temperature: 4.56 Degree Celcius
Max temperature: 2.4 Degree Celcius
Min Temperature: 6.39 Degree Celcius

Feels Like: 1.56 Degree Celcius

Humidity is: 37%
Pressure is: 1024hPa