Fetch the Weather with OpenWeatherMap API and JavaScript

As a web developer, grabbing data from `API` 's is something you are going to do often. Fetching weather data is a perfect way to get your feet wet. In this article we are going to use the browsers built in `fetch` with JavaScript to grab data from OpenWeatherMap's API.

OpenWeatherMap API

OpenWeatherMap has a very good API. They offer a free tier that allows up to 60 calls per minute along with 5 day and 3 hour forecasts.

Create an OpenWeather account

To get started we need an account. Head over to Open Weather to Sign up.

Generate a Key

After sign up, a key automatically gets created for you. The API keys are accessed in your account settings. This key will be used when we fetch the weather which acts as an identifier so OpenWeatherMap knows who is asking for data.

Endpoints

The API call we are going to use has a few different options to get weather. To test parameter shown below.

by City ID (Recommended)

https://api.openweathermap.org/data/2.5/weather?id=6167865&appid={yourkey}

Grab the current weather by using an explicit `city ID`. A list of cities is available here.

by City Name

https://api.openweathermap.org/data/2.5/weather?q=Toronto,CA&appid={yourkey}

Where `q` is our desired city, along with optional country code in ISO 3166 format.

by Latitude, Longitude

https://api.openweathermap.org/data/2.5/weather?lat=43.700111&lon=-79.416298&appid={yourkey}

More documentation is available on OpenWeatherMap's documentation page.

HTML

Pretty straight forward so far, now let's get into some code. Create a new file called `weather.html` in your favourite editor and start with the following HTML:

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Current Weather</title>
</head>
<body>
 <div id="weather">
  <div id="description"></div>
```

```
      <h1 id="temp"></h1>
      <div id="location"></div>
    </div>
    <script lang="text/javascript">
      // future javascript
    </script>
  </body>
  </html>
```

The Fetch API

To get the actual data we will being using the Fetch API. If you haven't used fetch before it is a way to make simple get requests to data like API's. It has good browser support with Chrome, Firefox and Edge however it does not work Internet Explorer.

Check out more extensive information on the support for the Fetch API.

Fetch the Weather

The Fetch API is dead simple. It works with a function called `fetch` that takes your `url` as the only parameter. The initial response is technically a stream of content, so to transform it we call use a `.then` to return the `json` response. Wrap it in a function and you have your very own weather balloon.

```
  function weatherBalloon( cityID ) {
    var key = '{yourkey}';
    fetch('https://api.openweathermap.org/data/2.5/weather
    .then(function(resp) { return resp.json() }) // Conver
    .then(function(data) {
      console.log(data);
    })
    .catch(function() {
      // catch any errors
    });
}
```

window.onload = function() {

```
        weatherBallon( 6167865 );
    }
```

Place this inside the `script` tag and open your `developer console` and you if you see the following response you have weather!

(Hit F12 in Chrome/Firefox/Edge to open the developer console)

```
    {
       "coord":{
          "lon":-79.42,
          "lat":43.7
       },
       "weather":[
          {
             "id":804,
             "main":"Clouds",
             "description":"overcast clouds",
             "icon":"04n"
          }
       ],
       "base":"stations",
       "main":{
          "temp":292.15,
          "pressure":1023,
          "humidity":72,
          "temp_min":291.15,
          "temp_max":293.15
       },
       "visibility":14484,
       "wind":{
          "speed":5.7,
          "deg":130
       },
       "clouds":{
          "all":90
       },
       "dt":1537837200,
       "sys":{
          "type":1,
          "id":3721,
          "message":0.0039,
          "country":"CA",
    "id":6167865,
    "name":"Toronto",
    "cod":200
}
```

Display the Weather

To a programmer, a nicely organized JSON structure holds its own beauty, but everyone else may not agree. We need to actually put this data on the page in a meaningful way.

Weather Markup

Time to add a few html tags to the page that will have data inserted into. In your body add the following placeholder.

```
<div id="weather">
    <div id="description"></div>
    <h1 id="temp"></h1>
    <div id="location"></div>
</div>
```

Now that our markup is in place, we need to insert the weather data we fetched above into the placeholders.

drawWeather

To do this, we will create a new `drawWeather` function, add this under your `weatherBalloon` . This will take the weather data as a parameter and insert it into the

```
function drawWeather( d ) {
        var celcius = Math.round(parseFloat(d.main.temp)
        var fahrenheit = Math.round(((parseFloat(d.main.

        document.getElementById('description').innerHTML
        document.getElementById('temp').innerHTML = celc
        document.getElementById('location').innerHTML =
}
```

Most of the data is a straight replace, with the exception of the temperature. OpenWeatherMap gives us temperature in Kelvin, because they don't mess around, so we have a two different conversions happening.

use drawWeather

Of course the function is half the battle. We haven't called it yet from our `weatherBalloon` function. Instead of console.log, call `drawWeather` .

```
function weatherBalloon( cityID ) {
        var key = '{yourkey}';
        fetch('https://api.openweathermap.org/data/2.5/w
        .then(function(resp) { return resp.json() }) //
        .then(function(data) {
                drawWeather(data); // Call drawWeather
        })
        .catch(function() {
                // catch any errors
        });
}
```

Your page should now look like this:

```
few clouds
9°
Toronto
```

Make it pretty

We could stop here but what fun would that be. We need to up our font game and change the background a bit depending on the weather description. Add a the following Google Font and `style` tag in the `<head>` tag of your html:

```
<link href="https://fonts.googleapis.com/css?family=Mont

<style>
body {
    font-family: 'Montserrat', sans-serif;
    font-weight: 400;
    font-size: 1.3em;
    color: #fff;
    text-shadow: .1em .1em 0 rgba(0,0,0,0.3);
    height: 100vh;
    display: flex;
    align-items: center;
    justify-content: center;
    background-image: linear-gradient(to right top, #99b
}
h1 {
    margin: 0 auto;
    font-size: 2.2em;
    text-align: center;
    color: #fff;
    font-size: 5em;
}
body.sunny {
    background-image: linear-gradient(to right top, #ff4e
}
body.cloudy {
    background-image: linear-gradient(to right top, #637
}
body.rainy {
    background-image: linear-gradient(to right top, #637
}
```

We are using flexbox to vertically and horizontally center our content on the page, but we also need to add a height to the body. By default the body is only as large as its content so we will make sure it is `100vh` which is `100%` of the viewport height.

<body> class swapping

Our `drawWeather` function will now need updates to change swap the body class between `.cloudy` , `.rainy` and `.sunny` . We are going to look at the weather description given to use by the `API` to look for a few keywords and simply adjust the class if we find anything.

```
function drawWeather( d ) {
    var celcius = Math.round(parseFloat(d.main.temp)
    var fahrenheit = Math.round(((parseFloat(d.main.
    var description = d.weather[0].description;

    document.getElementById('description').innerHTML
    document.getElementById('temp').innerHTML = celc
    document.getElementById('location').innerHTML =

    if( description.indexOf('rain') > 0 ) {
    document.body.className = 'rainy';
 } else if( description.indexOf('cloud') > 0 ) {
    document.body.className = 'cloudy';
 } else if( description.indexOf('sunny') > 0 ) {
    document.body.className = 'sunny';
 }
}
```

Clear skies ahead

You now have a functioning weather page. We only scratched the surface on the around with the rest of the data! If you just want to play around check out the complete version on CodePen.

CodePen

What do you think?

Enter your thoughts...

6 comments

Bhuvaneswari a year

This post was really helpful. Thanks, byte master.

REPLY

tibogen a year

nice ^^

REPLY

AWESOME!!!

REPLY

Glen 4 months

awesome work!

REPLY

Yannick 4 months

Awesome

REPLY

maru 4 months

I learned so mcuh thank you very much. it was easy to follow

REPLY