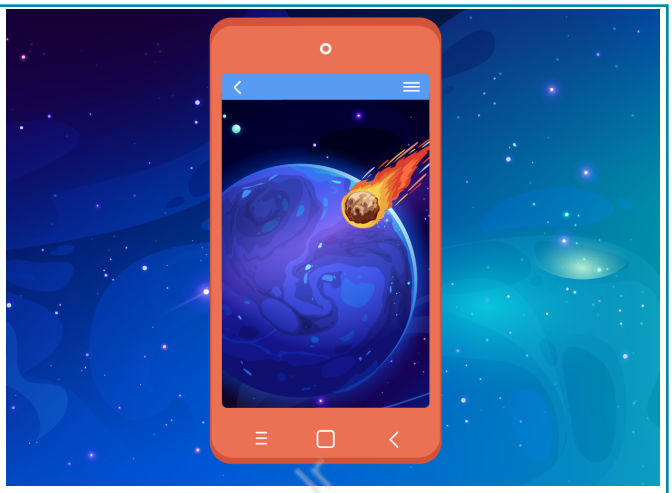
CS-PRO-C79(V3)

WhiteHat Jr
Live Online Coding for Kids

METEOR SCREEN 1



What is our GOAL for this MODULE?

We used the NASA API to get data about different asteroids/meteors.

What did we ACHIEVE in the class TODAY?

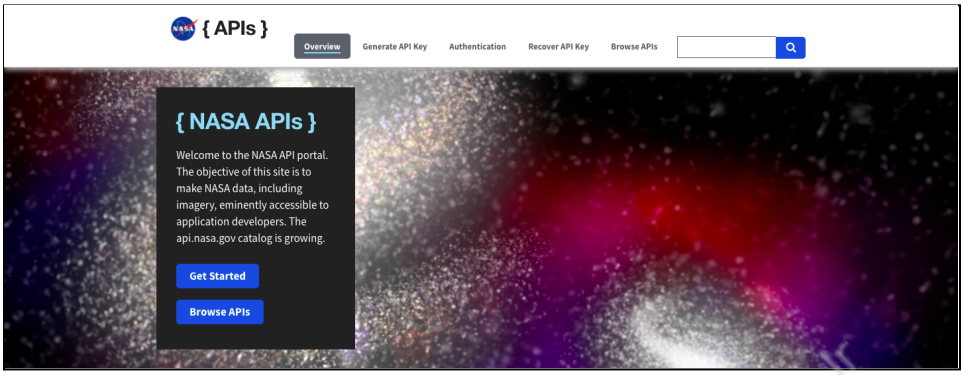
- Generated the API key to access the data.
- Used the API key to get data from the asteroids API.
- Generate threat scores for the meteors

Which CONCEPTS/ CODING BLOCKS did we cover today?

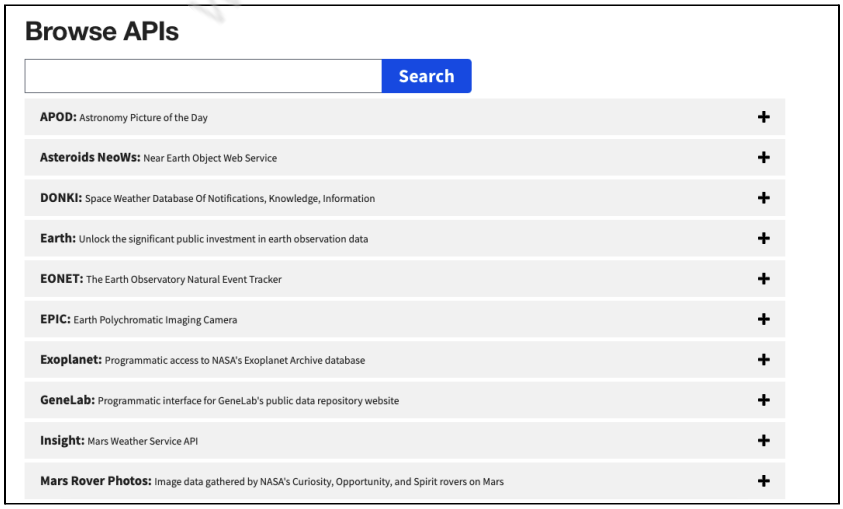
- Usage of API.
- Advanced JavaScript concepts to perform array operations

How did we DO the activities?

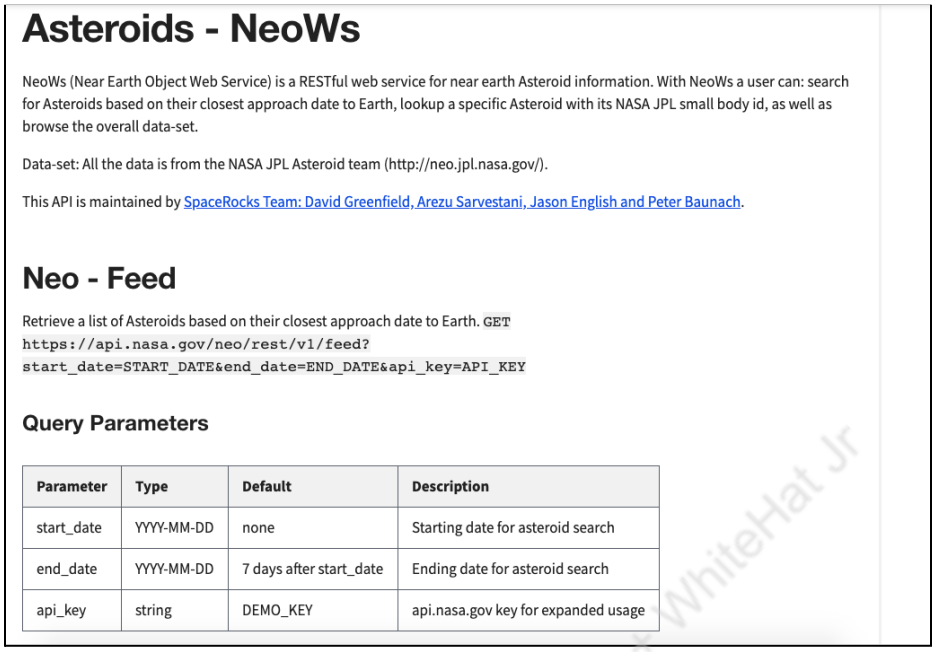
1. Sign up on the NASA website to generate the API key.



2. Browse through all the APIs provided by NASA to get the asteroids API.



3. Get the Asteroids NeoWs API.



Asteroids - NeoWs

NeoWs (Near Earth Object Web Service) is a RESTful web service for near earth Asteroid information. With NeoWs a user can: search for Asteroids based on their closest approach date to Earth, lookup a specific Asteroid with its NASA JPL small body id, as well as browse the overall data-set.

Data-set: All the data is from the NASA JPL Asteroid team (http://neo.jpl.nasa.gov/).

This API is maintained by SpaceRocks Team: David Greenfield, Arezu Sarvestani, Jason English and Peter Baunach.

Neo - Feed

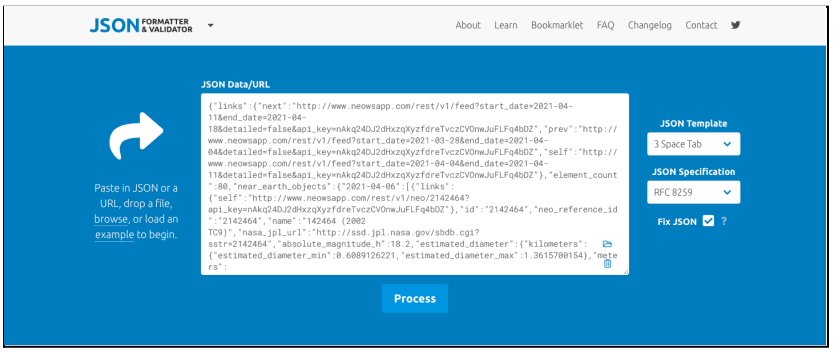
Retrieve a list of Asteroids based on their closest approach date to Earth. `GET https://api.nasa.gov/neo/rest/v1/feed?start_date=START_DATE&end_date=END_DATE&api_key=API_KEY`

Query Parameters

Parameter	Type	Default	Description
start_date	YYYY-MM-DD	none	Starting date for asteroid search
end_date	YYYY-MM-DD	7 days after start_date	Ending date for asteroid search
api_key	string	DEMO_KEY	api.nasa.gov key for expanded usage

4. Check for the data on the asteroids API.

5. Use the JSON formatter to make it readable.



6. Write a getMeteor function to get the data using the asteroids API.

```
import React, { Component } from 'react';
import { Text, View } from 'react-native';

import axios from "axios";

export default class MeteorScreen extends Component {

    getMeteors = () => {
        axios
            .get("https://api.nasa.gov/neo/rest/v1/feed?api_key=nAkq24DJ2dHxzqXyzfdreTvczCVOnwJ
            .then(response => {
                this.setState({ meteors: response.data.near_earth_objects })
            })
            .catch(error => {
                Alert.alert(error.message)
            })
    }

    render() {
        return (
            <View
                style={{
                    flex: 1,
                    justifyContent: "center",
                    alignItems: "center"
                }}>
                <Text>Meteor Screen!</Text>
            </View>
        )
    }
```

7. Call the function in the ComponentDidMount function.

```
constructor(props) {
    super(props);
    this.state = {
        meteors: {},
    };
}

componentDidMount() {
    this.getMeteors()
}
```

8. Write code to get the specific needed data from the provided data.

```
let meteor_arr = Object.keys(this.state.meteors).map(meteor_date => {
        return this.state.meteors[meteor_date]
    })
    let meteors = [].concat.apply([], meteor_arr);

    meteors.forEach(function (element) {
        let diameter =
(element.estimated_diameter.kilometers.estimated_diameter_min +
element.estimated_diameter.kilometers.estimated_diameter_max) / 2
        let threatScore = (diameter /
element.close_approach_data[0].miss_distance.kilometers) * 1000000000
        element.threat_score = threatScore;
    });
```


What's NEXT?

In the next class, we will work on completing our meteor screen and with it, our ISS Tracker app will get completed.

EXTEND YOUR KNOWLEDGE

1. Learn & Experiment more with Arrays and operations : https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array