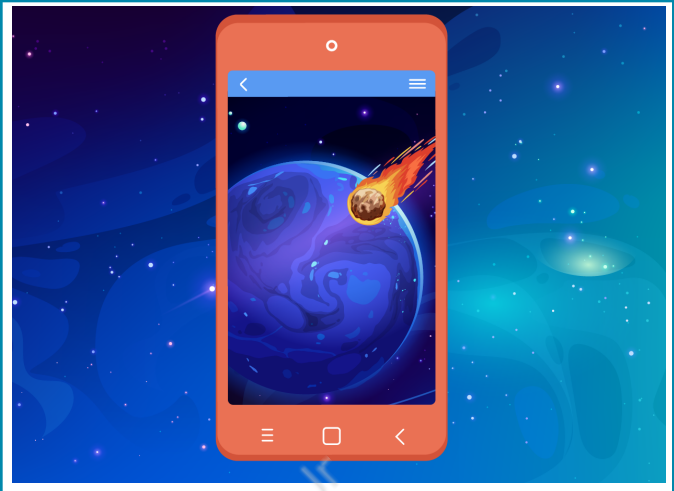


METEOR SCREEN 1



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?

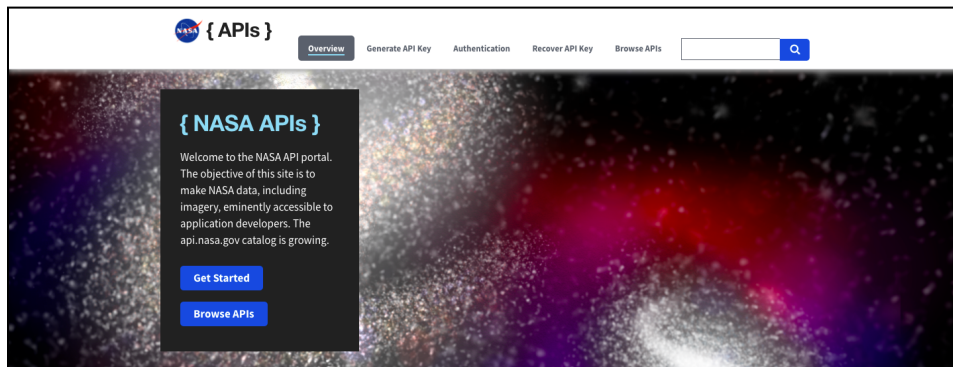
- Generated the API key to access the data.
- Used the API key to get data from the asteroids API.
- Generate threat scores for the meteors

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.



Generate API Key


Sign up for an application programming interface (API) key to access and use web services available on the Data.gov developer network.

* Required fields

* First Name

* Last Name

* Email

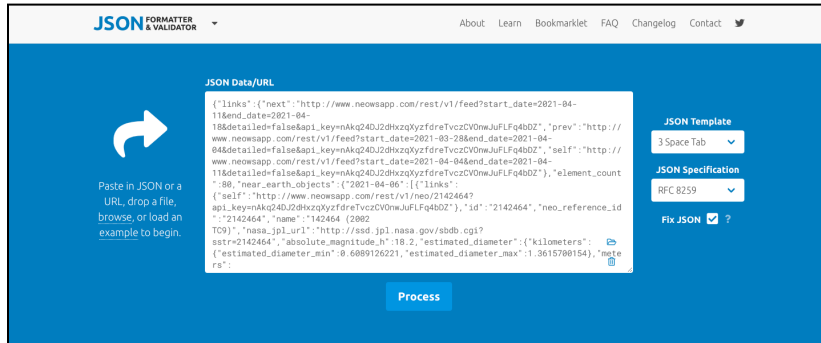
Application URL  (optional):

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

Browse APIs

APOD: Astronomy Picture of the Day	+
Asteroids NeoWs: Near Earth Object Web Service	+
DONKI: Space Weather Database Of Notifications, Knowledge, Information	+
Earth: Unlock the significant public investment in earth observation data	+
EONET: The Earth Observatory Natural Event Tracker	+
EPIC: Earth Polychromatic Imaging Camera	+
Exoplanet: Programmatic access to NASA's Exoplanet Archive database	+
GeneLab: Programmatic interface for GeneLab's public data repository website	+
Insight: Mars Weather Service API	+
Mars Rover Photos: Image data gathered by NASA's Curiosity, Opportunity, and Spirit rovers on Mars	+

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=nAkq24DJ2dHxqXyzfdreTvczCV0nwJuF4b0Z")
      .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) * 100000000  
  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