Exercise Monday

Intermediate JavaScript (/intermediate-javascript) / Asynchrony and APIs (/intermediate-javascript/asynchrony-and-apis) / OpenWeather API, Giphy API

Text

Goal: Practice making API calls in JavaScript. Start by following along with the pre-work and then build out a basic application that uses an API call. Your projects should include full development environments, a `.env` file for sensitive API variables, and complete README instructions, including instructions for setting up an API key as needed. Make sure to practice parsing JSON and API call error handling, too.

Warm Up

- What is the difference between synchronous and asynchronous code?
- What does it mean for JavaScript to be single-threaded and non-blocking?
- What does API stand for?
- What are some differences between Web APIs and third-party APIs?
- What is an HTTP status code?
- Name two tools that we can use to add error handling for API calls.

Code

OpenWeather API

Follow along with the weekend homework to build out a basic application that works with the OpenWeather API. Once you've finished the weekend homework, incorporate the following functionality in your application:

- Add additional information beyond temperature and humidity so that users get a full picture of the weather forecast. Convert the temperature into Fahrenheit, too.
- Allow users to search by other options beyond `city`.
- Check out other OpenWeather API endpoints and add additional code to your application so that users can query the API for other interesting data.
- Use the DevTools *Network* tab to look at the API response object.
- **Optional:** Try adding a `try...catch` block to handle API errors. *Don't* look at future lessons on how to do this. A big part of being a developer is figuring out how to combine different tools in an application. This is a good opportunity to practice! **Hint:** You'll want to throw an error if the API call is not successful.

Giphy API

Create an application that allows users to search Giphy (https://developers.giphy.com/docs/api/) for interesting GIFs. Make sure to carefully read Giphy's documentation as you build out your application.

Project Requirements

Make sure that your project has the following features:

- A fully functioning webpack environment.

- Testing the API via Postman before coding.
- An API key protected via a `.env` file.
- Error handling for the API call.

Functionality

- First, build out your application to allow users to search for GIFs using a keyword.
- At some point, use the DevTools *Network* tab at least once to look at the API response object.
- Next, add functionality so your users can see trending content from Giphy.
- Finally, add a button that users can click to randomly see a GIF from Giphy.
- **Very challenging:** Add functionality so a user can upload content to Giphy. Note that the free tier only allows users to upload 10 files per day. You can apply for a production key if you'd like the option to add more.
- **Optional:** Try adding a `try...catch` block to handle API errors without looking at future lessons on how to do this. Once again, this is good practice.

Trivia Flash Cards

Create a trivia flash card application using a trivia API listed here (https://www.programmableweb.com/category/trivia/api) or another API such as the Open Trivia Database (https://opentdb.com/api_config.php). Note that these APIs haven't been vetted... This is where you need to do some research on your own, try out API calls with Postman, and determine whether an API is a good fit for your project. Once you've found an API and verified that you can make API calls with Postman, build out the project to include the following:

Project Requirements

Make sure that your project has the following features:

- A fully functioning webpack environment.
- An API key protected via a `.env` file.
- Testing the API key via Postman before coding.
- Error handling for the API call.
- **Optional:** Try adding a `try...catch` block to handle API errors without looking at future lessons on how to do this. Once again, this is good practice.

Functionality

- First, query the API to get one (or multiple) trivia questions and answers.
- Next, populate a series of cards with trivia questions. A user should be able to click on a trivia question to see the answer.
- Add other functionality as you see fit. For instance, allow a user to mark whether they answered a question correctly or incorrectly — and then tally the total number of correct and incorrect answers once they are finished.

Peer Code Review

- Does the application have a fully functioning development environment?
- Does the application successfully make an API call?
- Are API keys stored in a `.env` file and protected?
- Does the README include instructions for setting up the project including getting an API key and adding it to a `.env` file?

Previous (/intermediate-javascript/asynchrony-and-apis/journal-6-discussion)
Next (/intermediate-javascript/asynchrony-and-apis/tools-for-handling-async-code)

Lesson 15 of 33
Last updated more than 3 months ago.

disable dark mode

(http://www.epicodus.com)