Building a Movie Search Tool with the TMDB API

Contents

Chapter 1. Prerequisites

Want to learn to use use a movie search API and create a web app like this? Let's go through it together.

Note:

Some intermediate knowledge of HTML, inline CSS, and JavaScript is needed.

Step 0 - Getting an API key

Follow this link to create an account on themoviedb.org. After signing up, you can register for an API key by clicking the API link from within your account settings page.

After that's done, open the full documentation in another tab for later reference.

Chapter 2. HTML

Step 1 - Set Up HTML and CSS

Create a new .html file and start with this code

```
<!DOCTYPE html>
          <html>
            <head>
              <meta charset="UTF-8">
              <title>Movie Search</title>
              <style>
                .enclosure {
                  background-color: midnightblue;
                  color: aliceblue;
                  border: 5px groove;
                }
                </style>
            </head>
```

Note:
The entire `<style>` section is completely optional, or customizable to your liking. They are merely style elements for visual interest and do not affect the functionality at all.

Step 2 - Implement Search Functionality

Add this HTML code below your previous HTML code.

```
        <body>
          <div class="enclosure">
          <form>
              Movie:  <input id="movieTitle" size="25px" placeholder="Type the name of a movie" />
              <button type="submit" id="submit">Submit</ button>
              </form>
              <p id="title">Title:  </p>
              <p id="overview">Overview:  </p>
              <p id="release-date">Release  date: </p>
              <img id="poster">
```

This code block contains the visible DOM text elements and the search bar. All the functionality will be added later.

Chapter 3. JavaScript

Step 3 - Implement API Call and Fetch Results

All of this JavaScript code is written inside the same HTML file as above, inside `<script>` tags.

Add this JavaScript code to the bottom of the `<body>` section, just above the `</body>` tag.

```
<script>
          var apiKey = 'YOUR_API_KEY_HERE';
          var button = document.getElementById('submit');
          button.addEventListener('click', (e) => {
            e.preventDefault();
            movieSearch();
          });

          async function movieSearch() {
            var userQuery = document.getElementById('movieTitle').value;
            var url = `https://api.themoviedb.org/3/search/movie?api_key=${apiKey}&query=${userQuery}`;
            let response = await fetch(url);
            if (!response.ok) {
                console.log('HTTP error; status: ', response.status);
            } else {
              let data = await response.json();
              console.log(data);
```

This code creates the submit button functionality, defines the asynchronous function that makes the API call, and produces a JSON object of the response.

Step 4 - Display Movie Details on the Web Page

Add this JavaScript code immediately after the previous code block:

```
        document.getElementById('title').textContent = `🎬Title:  ${data.results[0].title}`

        document.getElementById('overview').textContent = `🎬Overview:  ${data.results[0].overview}`

        document.getElementById('release-date').textContent = `🎬Release  date: ${data.results[0].release_date}`
```

```
      document.getElementById('poster').src = `https://image.tmdb.org/t/p/w342${data.results[0].poster_path}`;
    }
  }
  </script>
```

This code renders the relevant data from the API response to the DOM.

The magenta-colored code fragments represent **title**, **overview**, and **release date**. However, they can be replaced with whichever data elements in the response object you want (see below).

Data elements

The table below contains all the different data elements returned by our API call .

Note:
All response data can be found in the relevant documentation, underneath the **Responses** heading.

Element	JSON Path	Description	Data type	Note
Poster Path	`.poster_path`	The url fragment that accesses an image of the movie poster	string	See this page for more information on how to build a complete image URL
Overview	`.overview`	Provides a plot summary of the movie	string	
Release date	`.release_date`	Returns the date the movie was released	string	Format is YYYY-MM-DD
Original language	`.original_language`	Indicates the original primary language of the movie	string	Format is a two letter language code (click here for a full list of ISO

Element	JSON Path	Description	Data type	Note
				639-1 language codes)
Title	`.title`	The title of the movie	string	
Popularity	`.popularity`	Indicates the average popularity of the movie	float	This popularity element has a range of 0-100

Important:
Each of the **JSON path** code fragments must be preceded by `data.results[0]`. The data object is the JSON-formatted version of the API response, and `results[0]`accesses the data of the first search result in the response.

Using the example code and table above, customize your app to display the information you want it to display.

Chapter 4. Finishing touches

Step 5 - Finish document

Add this code to the end of your project to close it up:

```
    </div>
  </body>
</html>
```

Your app should be functioning as expected now! If you are experiencing issues, consult the full code implementation below:

```
<div class="enclosure">
<form>
      📽Movie:  <input id="movieTitle" size="25px" placeholder="Type the name of a movie" />
      <button type="submit" id="submit">📽Submit</ button>
      </form>
      <p id="title">🎬Title:  </p>
      <p id="overview">🎬Overview:  </p>
      <p id="release-date">🎬Release  date: </p>
      <img id="poster">
      </div>
      <script>
  var apiKey = 'YOUR_API_KEY_HERE';
  var button = document.getElementById('submit');
  button.addEventListener('click', (e) => {
    e.preventDefault();
    movieSearch();
  });

  async function movieSearch() {
    var userQuery = document.getElementById('movieTitle').value;
    var url = `https://api.themoviedb.org/3/search/movie?api_key=${apiKey}&query=${userQuery}`;
    let response = await fetch(url);
    if (!response.ok) {
        console.log('HTTP error; status: ', response.status);
    } else {
```

```
            let data = await response.json();
            console.log(data);

            document.getElementById('title').textContent = `🎬Title:  ${data.results[0].title}`;

            document.getElementById('overview').textContent = `🎬Overview:  ${data.results[0].overview}`;

            document.getElementById('release-date').textContent = `🎬Release  date: ${data.results[0].release_date}`;

            document.getElementById('poster').src = `https://image.tmdb.org/t/p/w342${data.results[0].poster_path}`;
          }
        }
        </script>
        </div>
        </body>
        </html>
```

Author: Kevin Picado