Use the Fetch API to GET Data Labs

Perform these labs on your own computer using VS Code or any editor and command prompt to ensure you understand the lessons presented in the corresponding videos and lectures.

Lab 1: Create Website Using lite-server

Open the folder where you like to create your projects in a terminal window and create a new folder named **FetchAPISamples**.

```
mkdir FetchAPISamples
```

Create a new project in this folder.

```
npm init -y
```

Next, install lite-server.

```
npm install lite-server --save-dev
```

Open folder in VS Code

Open the **package.json** file and modify the "start" property to use node**mon** instead of just **node**.

```
"scripts": {
  "start": "lite-server"
},
```

Create a file named **index.html**.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Fetch API Samples</title>
  </head>
  <body>
    <h1>Fetch API Samples</h1>
  </body>
</html>
```

Try it Out

Open a terminal window in VS Code and type in

```
npm start
```

Your browser should open, and you should see the <h1> header appear.

Lab 2: View the Response Object

Open the **index.html** file and below the </h1> tag add the following code.

```
<div>
  <button onclick="getAllData();">Get All</button>
</div>

<script>
  const SERVER_URI = "http://localhost:5000/api";

  function getAllData() {
    fetch(`${SERVER_URI}/people`)
      .then(response => console.log(response))
      .catch(error => console.error(error));
  }
</script>
```

Try it Out

Save all your changes.

Make sure your Web API node server is still running.

Copyright © 2024 by Paul D. Sheriff
All rights reserved worldwide. Reproduction is strictly prohibited.

If it has not already, refresh your **http://localhost:3000** page.

Click on the **Get All** button. Bring up the Browser's Console Window and you should see the JSON response object.

NOTE: If it does not refresh with the data, restart both web servers.

Lab 3: Get body Using .json() Method

Open the **index.html** file and within the getAllData() function add the line shown in bold below.

```
function getAllData() {
  fetch(`${SERVER_URI}/people`)
    .then(response => response.json())
    .then(response => console.log(response))
    .catch(error => console.error(error));
}
```

Try it Out

Save all your changes.

Make sure your Web API node server is still running.

If it has not already, refresh your **http://localhost:3000** page. Click on the **Get All** button. Bring up the Browser's Console Window and you should see a JSON response object with the **data** property set to an array of JSON data.

NOTE: If it does not refresh with the data, restart both web servers.

Lab 4: Get a Single JSON Object Using the Fetch API

Open the **index.html** file and just above the <div> you created in the last lab, add the following.

```
<div>
  <label for="personId">Person ID</label>
  <input type="text" id="personId" value="1" />
</div>
```

Copyright © 2024 by Paul D. Sheriff
All rights reserved. Reproduction is strictly prohibited.

Add a new button within the <div> where the Get All button is located.

```
<div>
  <button onclick="getAllData();">Get All</button>
  <button onclick="getRow();">Get a Row</button>
</div>
```

Just below the getAllData() function, add a new function.

```
function getRow() {
  let id = document.getElementById("personId").value;
  fetch(`${SERVER_URI}/people/${id}`)
    .then(response => response.json())
    .then(response => console.log(response))
    .catch(error => console.error(error));
}
```

Try It Out

Save all your changes.

If it has not already, refresh your **http://localhost:3000** page. Click on the **Get a Row** button. Bring up the Brower's Console Window and you should see a JSON response object with the **data** property set to a single JSON object with the people data.

Lab 5: Working With Status Codes

Enter the number 111 into the Product ID input field.

Click on the **Get a Row** button.

Bring up the Brower's Console Window and you should see a JSON response object with the **status** property set to 404 and an **error** property is showing a JSON error object.

Lab 6: Viewing Errors in the catch() Method

Open the **index.html** file and modify the port number from 5000 to 50.

Copyright © 2024 by Paul D. Sheriff
All rights reserved worldwide. Reproduction is strictly prohibited.

```
http://localhost:50/api/
```

Try It Out

Save your change and run the project. Open your browser tools (F12) to get to the console window and you should see an error message that looks like the following if you are using the Chrome browser.

```
Failed to load resource: net::ERR_CONNECTION_REFUSED
```

In the error message label on the index page, you should see something that looks like the following message.

```
TypeError: Failed to fetch
```

Because the port number does not exist, a network error is detected by the Fetch API. Since the fetch() function is unable to reach the Web API server the .catch() method is called.

Copyright © 2024 by Paul D. Sheriff
All rights reserved. Reproduction is strictly prohibited.