CODE EXERCISE

JAVASCRIPT PROMISES

In JavaScript, a promise is an object that represents the eventual completion (or failure) of an asynchronous operation, and its resulting value. Promises are a way to handle asynchronous operations in JavaScript, such as making HTTP requests, in a more elegant and easier to reason about way than using callbacks. Promises were introduced in ECMAScript 6 and are now widely supported in modern browsers and JavaScript environments.

A promise can be in one of three states:

- pending: The initial state, indicating that the promise is still in progress.
- fulfilled: The state indicating that the promise has completed successfully and has a resulting value.
- rejected: The state indicating that the promise has failed and has a reason for the failure.

Promises have two main methods: then and catch. The then method is used to attach callbacks for the fulfillment of the promise and the catch method is used to attach callbacks for the rejection of the promise.

Here is an example of a simple promise:

```
let promise = new Promise((resolve, reject) => {
     //do some async operation
     setTimeout(() => resolve("Hello from the promise"), 3000);
 });

 promise
     .then((response) => console.log(response))
     .catch((error) => console.log(error));
```

Here, a promise is created with a constructor, this constructor takes one argument, a function that takes two arguments, a resolve and a reject. Inside that function we can put the asynchronous operation. In this case, the operation is just a setTimeout to wait 3 seconds. After the time passed the resolve function is called with the result. We can attach callbacks for the fulfillment of the promise using the then method and callbacks for the rejection of the promise using the catch method.

Promise also allows you to chain multiple then methods, so that you can execute multiple asynchronous operations one after the other.

SUBMITTING EXERCISES

To earn certification all exercises must be submitted and accepted.

1) When you've completed all of the HTML exercises, please zip them into a single file and submit to our Dropbox at http://bit.ly/CWDP2324.

2) Next, fill out the certification completion form at https://forms.gle/5EiUGCM6dGd1F2Py6.

Remember, that all of your exercises for each module should be included in a separate zip file.

GETTING HELP

We always want to ensure that your questions are answered. There are a number of ways to get in touch.

1) We operate a lively Discord server. Join us at https://discord.gg/tgxX2fCrv5 and you can ask your question on Discord. Mark and our team of instructional assistants monitor this Discord and answer questions ASAP.

2) This certification program is offered on several platforms. Most platforms have a Q & A section where you can post questions. We monitor these and respond as quickly as we can.

3) You may use our question hotline email at questions@dollardesignschool.com.

Remember to always to include which section your question is from and send any code you're working on!

CODE EXERCISE

JAVASCRIPT PROMISES

```
let promise1 = new Promise((resolve) => {
     setTimeout(() => resolve('Hello'), 2000);
 });
 let promise2 = new Promise((resolve) => {
     setTimeout(() =>
```

1) Create a new JavaScript file called "promise-example.js"

2) In the file, use the fetch() function to make a GET request to the JSONPlaceholder API (https://jsonplaceholder.typicode.com/), which returns sample data for testing and prototyping. For example:

```
fetch('https://jsonplaceholder.typicode.com/todos/1')
    .then(response => response.json())
    .then(data => console.log(data))
    .catch(error => console.error(error))
```

3) Run the code and you should see the first to-do item in the API logged to the console.

4) Play with the endpoint. Try to fetch different resources

```
fetch('https://jsonplaceholder.typicode.com/posts/1')
    .then(response => response.json())
    .then(data => console.log(data))
    .catch(error => console.error(error))
```

5) To demonstrate how to use Promises to handle multiple asynchronous operations, you can use the Promise.all() method. In this step you will fetch multiple post from the API

```
let postPromise1 = fetch('https://jsonplaceholder.typicode.com/posts/1').then(response
=> response.json())
let postPromise2 = fetch('https://jsonplaceholder.typicode.com/posts/2').then(response
=> response.json())
let postPromise3 = fetch('https://jsonplaceholder.typicode.com/posts/3').then(response
=> response.json())
Promise.all([postPromise1, postPromise2, postPromise3])
    .then(data => console.log(data))
    .catch(error => console.error(error))
```

SUBMITTING EXERCISES

To earn certification all exercises must be submitted and accepted.

1) When you've completed all of the HTML exercises, please zip them into a single file and submit to our Dropbox at http://bit.ly/CWDP2324.

2) Next, fill out the certification completion form at https://forms.gle/5EiUGCM6dGd1F2Py6.

Remember, that all of your exercises for each module should be included in a separate zip file.

GETTING HELP

We always want to ensure that your questions are answered. There are a number of ways to get in touch.

1) We operate a lively Discord server. Join us at https://discord.gg/tgxX2fCrv5 and you can ask your question on Discord. Mark and our team of instructional assistants monitor this Discord and answer questions ASAP.

2) This certification program is offered on several platforms. Most platforms have a Q & A section where you can post questions. We monitor these and respond as quickly as we can.

3) You may use our question hotline email at questions@dollardesignschool.com.

Remember to always to include which section your question is from and send any code you're working on!

CODE EXERCISE

JAVASCRIPT PROMISES

6) Now you should have an idea of how you can use promises to handle API calls, you can practice by extending this example by adding more functionality, for example adding an error handling for non-200 status codes, displaying the data in an html page, fetching resources using query parameters and so on.

Note:

- Make sure to check if the browser supports fetch, you can do that by checking for the existence of the Headers, Request, Response, and Fetch classes on the window object.
- The jsonplaceholder API might be rate-limited, please be aware of this while testing
- Keep in mind that fetch is just a way to make http request and many other libraries such as axios, superagent use promises as well.

Have fun!

SUBMITTING EXERCISES

To earn certification all exercises must be submitted and accepted.

1) When you've completed all of the HTML exercises, please zip them into a single file and submit to our Dropbox at http://bit.ly/CWDP2324.

2) Next, fill out the certification completion form at https://forms.gle/5EiUGCM6dGd1F2Py6.

Remember, that all of your exercises for each module should be included in a separate zip file.

GETTING HELP

We always want to ensure that your questions are answered. There are a number of ways to get in touch.

1) We operate a lively Discord server. Join us at https://discord.gg/tgxX2fCrv5 and you can ask your question on Discord. Mark and our team of instructional assistants monitor this Discord and answer questions ASAP.

2) This certification program is offered on several platforms. Most platforms have a Q & A section where you can post questions. We monitor these and respond as quickly as we can.

3) You may use our question hotline email at questions@dollardesignschool.com.

Remember to always to include which section your question is from and send any code you're working on!

CODE EXERCISE

JAVASCRIPT PROMISES

SUBMITTING EXERCISES

To earn certification all exercises must be submitted and accepted.

1) When you've completed all of the HTML exercises, please zip them into a single file and submit to our Dropbox at http://bit.ly/CWDP2324.

2) Next, fill out the certification completion form at https://forms.gle/5EiUGCM6dGd1F2Py6.

Remember, that all of your exercises for each module should be included in a separate zip file.

GETTING HELP

We always want to ensure that your questions are answered. There are a number of ways to get in touch.

1) We operate a lively Discord server. Join us at https://discord.gg/tgxX2fCrv5 and you can ask your question on Discord. Mark and our team of instructional assistants monitor this Discord and answer questions ASAP.

2) This certification program is offered on several platforms. Most platforms have a Q & A section where you can post questions. We monitor these and respond as quickly as we can.

3) You may use our question hotline email at questions@dollardesignschool.com.

Remember to always to include which section your question is from and send any code you're working on!