Cheatsheet: JavaScript Async

JavaScript Promises, Callback, Fetch and Axios Terminologies	Description	Code Example

JSON	It is a text-based format used for structuring data in a way that is both human-readable and machine-readable.	`{`
		`  "name": "John Doe",`
		`  "age": 30,`
		`  "city": "New York",`
		`  "email": "johndoe@email.com",`
		`  "hobbies": ["Reading", "Hiking", "Cooking"]`
		`}`
		Copied!

Callback	A callback in JavaScript is a function passed as an argument to another function, which is then executed at a later time or under certain conditions.	`function greet(name, callback) {`
		``  console.log(`Hello, ${name}!`); ``
		`  callback(); // Executes the callback function`
		`}`
		
		`function sayGoodbye() {`
		`  console.log('How are you!');`
		`}`
		
		`greet('John Doe', sayGoodbye); // Passing sayGoodbye function as a callback`
		Copied!

XMLHttpRequest Object	It is used to create an instance of the XMLHttpRequest object to initiate an HTTP request.	`var xhr = new XMLHttpRequest();` Copied!
XMLHttpRequest Open Methods	The open() method sets up the request, specifying the HTTP method (GET, POST, and so on) and the URL.	`xhr.open('GET', 'https://api.example.com/data', true);` Copied!
send() Method	The send() method is invoked to send the request to the specified URL.	`xhr.send();` Copied!
Load Data Using XMLHttpRequest	This code describes that data can be loaded using Ajax methods.	

```
<!DOCTYPE html>
<html>
<head>
  <title>AJAX Example</title>
</head>
<body>
  <button id="loadUsersBtn">Load Users</button>
  <div id="userList"></div>

  <script>
    // JavaScript for AJAX functionality
    document.getElementById('loadUsersBtn').addEventListener('click', function() {
      // Creating an XMLHttpRequest object
      var xhr = new XMLHttpRequest();

      // Define the request
      xhr.open('GET', 'https://jsonplaceholder.typicode.com/users', true);

      // Handle the response
      xhr.onload = function() {
        if (xhr.status >= 200 && xhr.status < 400) {
          var users = JSON.parse(xhr.responseText);
          displayUsers(users);
        } else {
          console.error('Error fetching data');
        }
      };

      // Handle network errors
      xhr.onerror = function() {
        console.error('Network error');
      };

      // Send the request
      xhr.send();
    });

    // Function to display users on the page
    function displayUsers(users) {
      var userListDiv = document.getElementById('userList');
      userListDiv.innerHTML = '<h2>User List</h2>';
      var ul = document.createElement('ul');

      users.forEach(function(user) {
        var li = document.createElement('li');
        li.textContent = user.name;
        ul.appendChild(li);
      });

      userListDiv.appendChild(ul);
    }
  </script>
</body>
</html>
```

Copied!

Promise Syntax

Promises are used for tasks like fetching data from a server, reading files, or performing other operations that may take some time to complete.

```
const myPromise = new Promise((resolve, reject) => {
  // Asynchronous operation goes here
  // If successful, call resolve with the result
  // If an error occurs, call reject with an error
});
```

		Copied!

Promise with .then and .catch	Promises are used for tasks like fetching data from a server, reading files, or performing using \`.then()\` method and caught error using \`.catch()\` method.	`const myPromise = new Promise((resolve, reject) => {`
		`  // Simulated asynchronous operation (e.g., making an API request)`
		`  setTimeout(() => {`
		`    const success = true; // Simulating a successful operation`
		`    if (success) {`
		`      resolve('Data successfully fetched');`
		`    } else {`
		`      reject('Error: Failed to fetch data');`
		`    }`
		`  }, 1000);`
		`});`
		
		`myPromise.then(`
		`  (result) => {`
		`    // Handle the successful result (e.g., update UI with the data)`
		`    console.log(result);`
		`  },`
		`  (error) => {`
		`    // Handle the error (e.g., log the error or show an error message)`
		`    console.error(error);`
		`  }`
		`);`
		
		Copied!

Fetch API Syntax	It is used for fetching resources from the web, such as data from a server or an API.	`fetch(url, options)`
		`  .then(response => {`
		`    // Handle the response`
		`  })`
		`  .catch(error => {`
		`    // Handle any errors that occurred during the fetch`
		`  });`
		
		Copied!

Fetch API Get Methods	The GET method is used to retrieve data from the specified resource.	`fetch('https://jsonplaceholder.typicode.com/posts')`
		`  .then(handleResponse)`
		`  .then(data => {`
		`    console.log('GET Request Result:', data);`
		`  })`
		`  .catch(error => {`
		`    console.error('Error:', error);`
		`  });`
		
		Copied!

Fetch API POST Method	The POST method is used to submit data to be processed to a specified resource.	

		`const newPost = {`
		`  title: 'New Post',`
		`  body: 'This is a new post.',`
		`  userId: 1`
		`};`
		
		`fetch('https://jsonplaceholder.typicode.com/posts', {`
		`  method: 'POST',`
		`  headers: {`
		`    'Content-Type': 'application/json'`
		`  },`
		`  body: JSON.stringify(newPost)`
		`})`
		`  .then(handleResponse)`
		`  .then(data => {`
		`    console.log('POST Request Result:', data);`
		`  })`
		`  .catch(error => {`
		`    console.error('Error:', error);`
		`  });`
		
		Copied!

Fetch API PUT Method	The PUT method is used to update or replace data at the specified resource. It is typically used to update existing records on the server.	`const updatedPost = {`
		`  id: 1,`
		`  title: 'Updated Post',`
		`  body: 'This post has been updated.',`
		`  userId: 1`
		`};`
		
		`fetch('https://jsonplaceholder.typicode.com/posts/1', {`
		`  method: 'PUT',`
		`  headers: {`
		`    'Content-Type': 'application/json'`
		`  },`
		`  body: JSON.stringify(updatedPost)`
		`})`
		`  .then(handleResponse)`
		`  .then(data => {`
		`    console.log('PUT Request Result:', data);`
		`  })`
		`  .catch(error => {`
		`    console.error('Error:', error);`
		`  });`
		
		Copied!

Fetch API PATCH Method	The PATCH method is used to apply partial modifications to a resource. It is typically used to update parts of a resource while leaving the rest of the resource unchanged.	

		`const updatedData = {` `  title: 'Updated Title'` `};` `fetch('https://jsonplaceholder.typicode.com/posts/1', {` `  method: 'PATCH',` `  headers: {` `    'Content-Type': 'application/json'` `  },` `  body: JSON.stringify(updatedData)` `})` `  .then(handleResponse)` `  .then(data => {` `    console.log('PATCH Request Result:', data);` `  })` `  .catch(error => {` `    console.error('Error:', error);` `  });` Copied!
Fetch API DELETE Method	The DELETE method is used to request the removal of a resource from the server. It is used to delete records or resources.	`fetch('https://jsonplaceholder.typicode.com/posts/1', {` `  method: 'DELETE'` `})` `  .then(response => {` `    if (response.ok) {` `      console.log('DELETE Request Successful');` `    } else {` `      throw new Error('DELETE request failed');` `    }` `  })` `  .catch(error => {` `    console.error('Error:', error);` `  });` Copied!
Axios Library Syntax	It provides a consistent way for making asynchronous HTTP requests to interact with RESTful APIs or other web services.	`axios({` `  method: 'HTTP_METHOD',` `  url: 'URL',` `  headers: {` `    // Headers (optional)` `  },` `  data: {` `    // Request data (optional)` `  }` `})` `  .then(response => {` `    // Handle the successful response` `  })` `  .catch(error => {` `    // Handle errors` `  });` Copied!
install axios	You can install axios using npm in the terminal after installing node.	`npm install axios`

		Copied!

Axios Methods	Axios have HTTP method for the request such as 'GET', 'POST', 'PUT', 'DELETE'.	`1`
		`2`
		`3`
		`4`
		`5`
		`6`
		`7`
		`8`
		`9`
		`10`
		`11`
		`12`
		`13`
		`14`
		`15`
		`16`
		`17`
		
		`axios({`
		`  method: 'HTTP_METHOD',`
		`  url: 'URL',`
		`  headers: {`
		`    // Headers (optional)`
		`  },`
		`  data: {`
		`    // Request data (optional)`
		`  }`
		`})`
		`  .then(response => {`
		`    // Handle the successful response`
		`  })`
		`  .catch(error => {`
		`    // Handle errors`
		`  });`
		
		Copied!

Skills Network