CoGrammar

Welcome to this session:

Task Walkthrough - JSON and AJAX with Fetch

The session will start shortly...

Questions? Drop them in the chat. We'll have dedicated moderators answering questions.

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar HyperionDev

Skills Bootcamp Cloud Web Development

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

Skills Bootcamp Cloud Web Development

- For all **non-academic questions**, please submit a query: ***www.hyperiondev.com/support***

- **Report a safeguarding incident: *www.hyperiondev.com/safeguardreporting***

- We would love your feedback on lectures: ***Feedback on Lectures***

- If you are hearing impaired, please kindly use your computer's function through Google chrome to enable captions.

Learning Outcomes

- ❖ **Create and manipulate JavaScript objects and JSON data** for storing structured information.
- ❖ **Use session storage to save and retrieve user data** in a web application.
- ❖ **Fetch and display real-time data using AJAX and JSON APIs**, such as for motivational quotes.
- ❖ **Implement interval-based functions** to update real-time information on a webpage, such as goal progress.
- ❖ **Develop simple applications** that combine data storage, dynamic updates, and asynchronous fetching for engaging user experiences.

Lecture Overview

- → Presentation of the Task
- → JSON
- → AJAX with Fetch
- → Task Walkthrough

JSON Task

Imagine having a daily goal tracker that reminds you of your progress and keeps you motivated with personalized messages. 🎯 In this task, you'll create a Goal Tracker that stores your goals for the day, shows your current progress, and gives motivational tips when you hit key milestones. You'll use JSON to structure goal data and session storage to save and retrieve this data during the session.

This task will help you master JSON data handling and storing user-specific information, turning your website into an interactive goal-checker!

AJAX Task

Need a quick dose of inspiration? 📝 In this task, you'll create a Daily Inspiration Generator that fetches a random motivational quote or advice from an external API. Each time the user clicks for inspiration, your app will use AJAX and Fetch to retrieve a new quote in JSON format, bringing fresh motivation to the screen.

This task will strengthen your understanding of AJAX with Fetch and JSON handling, while making your application dynamic and engaging.

What does JSON.stringify() do in JavaScript?

A. Fetches JSON data from an API
B. Stores JSON in session storage
C. Converts a JSON string into a JavaScript object
D. Converts a JavaScript object into a JSON string

CoGrammar

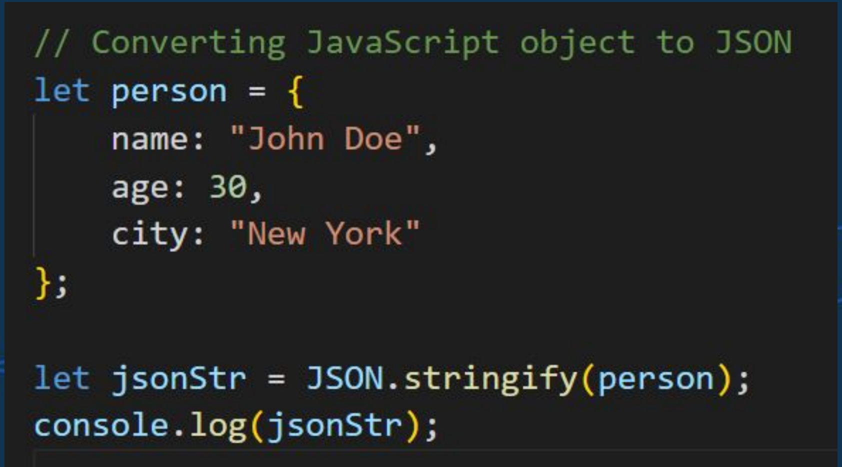
What does the .then() method in a fetch request do?

A. Executes code immediately
B. Specifies what to do once the promise is resolved
C. Loops through the JSON response
D. Clears the session storage

CoGrammar

Object Serialization with JSON

- **JSON** is a lightweight data interchange format inspired by JavaScript object literal syntax.
- It's commonly used for **serializing** and **transmitting** structured **data** over a **network** connection.
- JSON is **language-independent**, making it easy to work with in various programming languages.



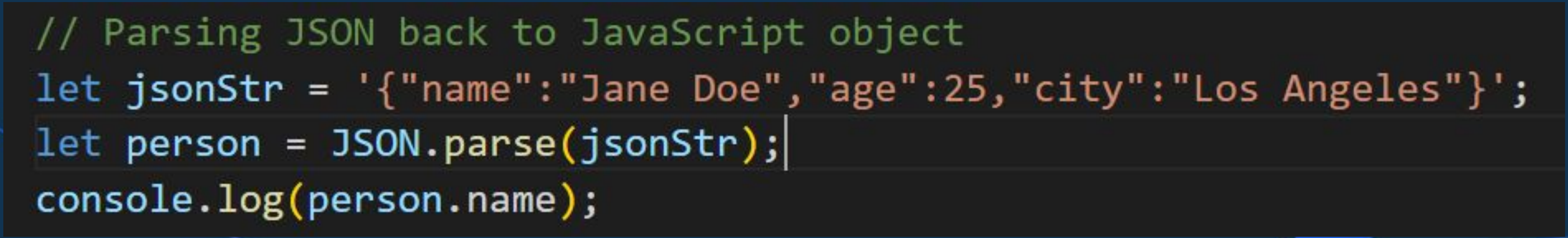
```
// Converting JavaScript object to JSON
let person = {
    name: "John Doe",
    age: 30,
    city: "New York"
};

let jsonStr = JSON.stringify(person);
console.log(jsonStr);
```

CoGrammar

Parsing JSON

- JSON can be parsed back into JavaScript objects using the **JSON.parse()** method.
- This allows us to work with **JSON** data received from external sources.



```
// Parsing JSON back to JavaScript object
let jsonStr = '{"name":"Jane Doe","age":25,"city":"Los Angeles"}';
let person = JSON.parse(jsonStr);
console.log(person.name);
```

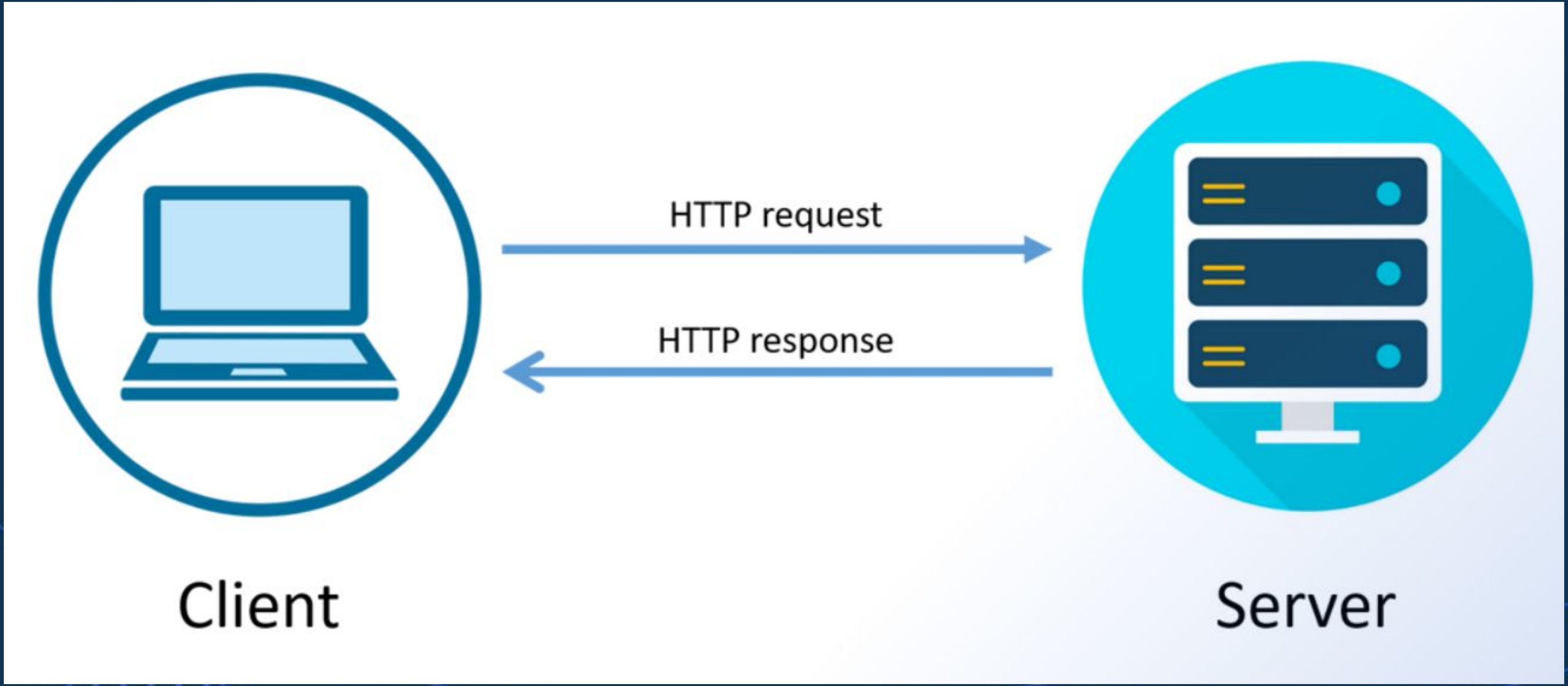
CoGrammar

Introduction to Web Requests

- ❖ Web requests, also known as HTTP requests, are the foundation of communication between clients (such as web browsers) and servers over the World Wide Web.
- ❖ They facilitate the exchange of data and resources, allowing users to access and interact with web applications, websites, and services.
- ❖ HTTP (Hypertext Transfer Protocol) is the underlying protocol used for sending and receiving web requests and responses.
- ❖ It operates as a request-response protocol, where clients send requests to servers, and servers respond with the requested resources or information.

CoGrammar

Introduction to Web Requests



CoGrammar

Components of an HTTP Request

- An HTTP request consists of several components, including:
  - Request Method (e.g., GET, POST, PUT, DELETE)
    - The request method indicates the desired action to be performed on the specified resource.
  - URL (Uniform Resource Locator) or URI (Uniform Resource Identifier)
    - The URL or URI specifies the location of the resource being requested by the client.
  - Headers (optional metadata)
  - Body (optional data for POST and PUT requests)

Components of an HTTP Response

- An HTTP response consists of several components, including:
  - Status Code (e.g., 200 OK, 404 Not Found)
    - The status code indicates the outcome of the server's attempt to fulfill the client's request.
  - Headers (metadata): HTTP headers provide metadata about the response, including information about the server, content type, content length, caching directives, and more.
  - Body (response data): The response body contains the actual data or resource requested by the client, such as HTML content, JSON data, images, or files.

CoGrammar

Common Request Methods

- GET: Retrieves data from the server.
- POST: Submits data to the server to create or update a resource.
- PUT: Updates an existing resource on the server.
- DELETE: Removes a resource from the server.

CoGrammar

What are APIs?

- ❖ An API (Application Programming Interface) allows different software systems to communicate with each other.
- ❖ It defines the methods and data formats that applications can use to request and exchange information.
- ❖ APIs are fundamental for web development, enabling interaction with remote servers and services.

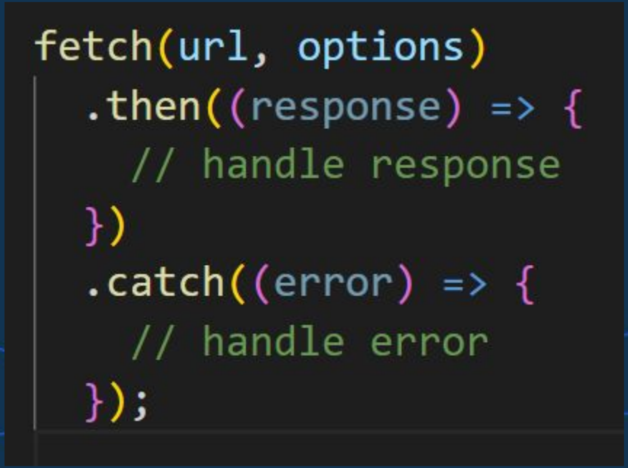
CoGrammar

Overview of Fetch API

- ❖ The Fetch API is a modern JavaScript interface for making asynchronous HTTP requests.
- ❖ Fetch follows a promise-based approach, which simplifies the handling of asynchronous operations, making code more readable and maintainable.
- ❖ Fetch returns a Promise that resolves to a Response object, allowing for seamless chaining of asynchronous operations using .then() and .catch() methods.

fetch() Method

- The fetch() function is used to initiate a request to the specified url.
- It returns a promise that resolves to the Response object representing the response to the request.
- The options parameter is an optional object containing settings for the request such as method, headers, body, etc.



```
fetch(url, options)
  .then((response) => {
    // handle response
  })
  .catch((error) => {
    // handle error
  });
```

CoGrammar

fetch() Method Parameters

- URL (required):
  - Specifies the URL to which the request is made.

- Options (optional): An object containing various settings for the request such as:
  - method: HTTP method (GET, POST, PUT, DELETE, etc.)
  - headers: Headers to include in the request
  - body: Data to send with the request (e.g., JSON, FormData)

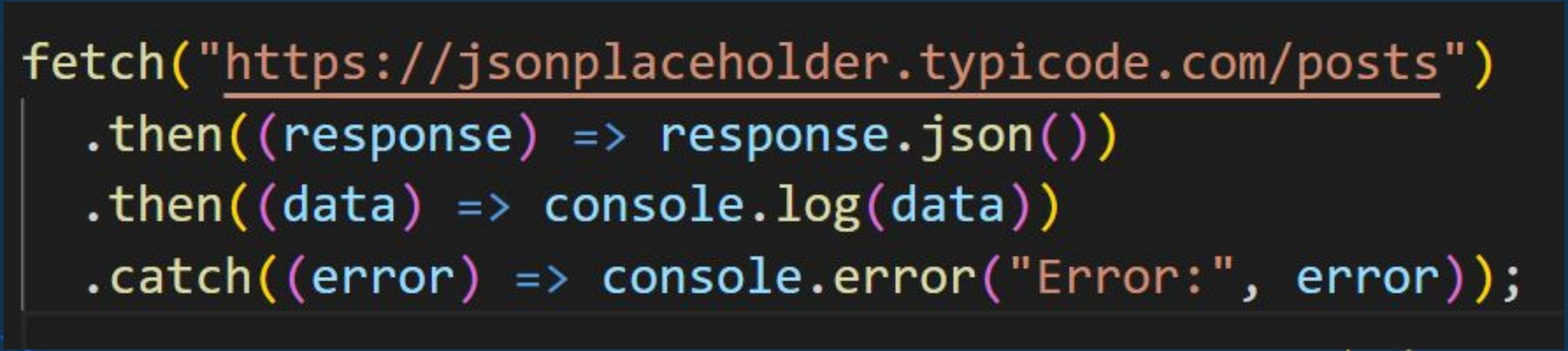
CoGrammar

Response Handling

- Response Object:
  - Represents the response to the request made by fetch().
  - Contains properties and methods to access response data and metadata.
- .json(): Parses the response body as JSON.

CoGrammar

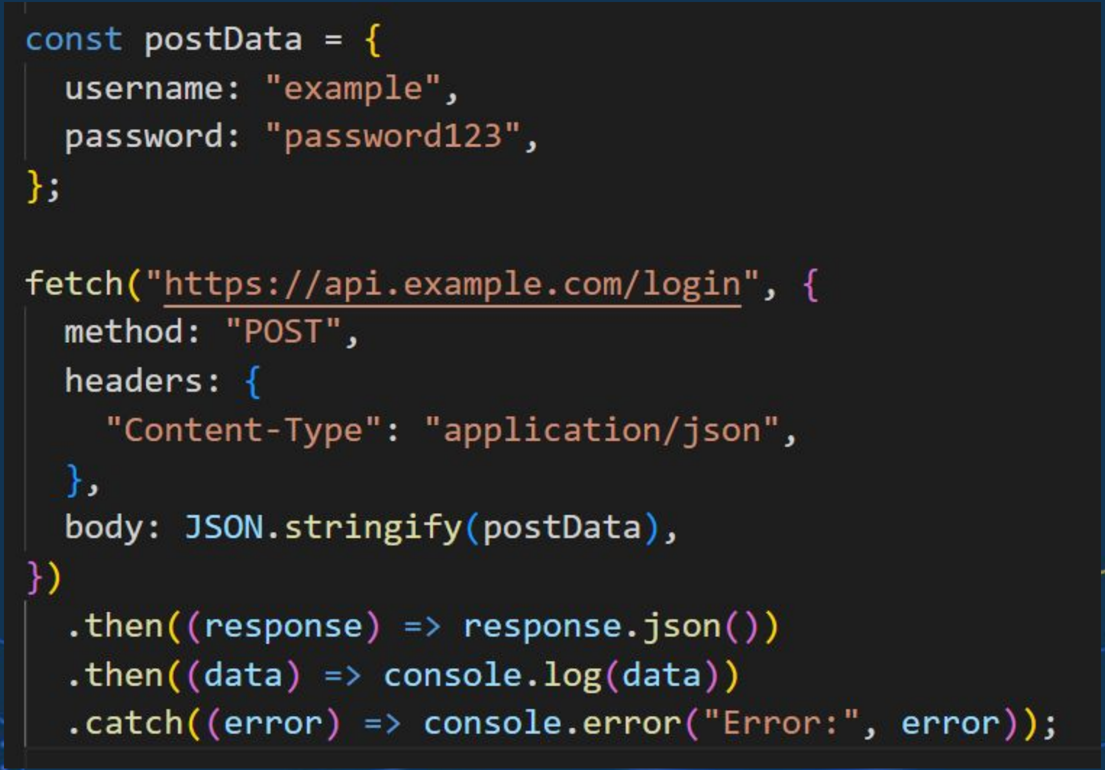
Making a GET Request



```
fetch("https://jsonplaceholder.typicode.com/posts")
  .then((response) => response.json())
  .then((data) => console.log(data))
  .catch((error) => console.error("Error:", error));
```

CoGrammar

Making a POST Request

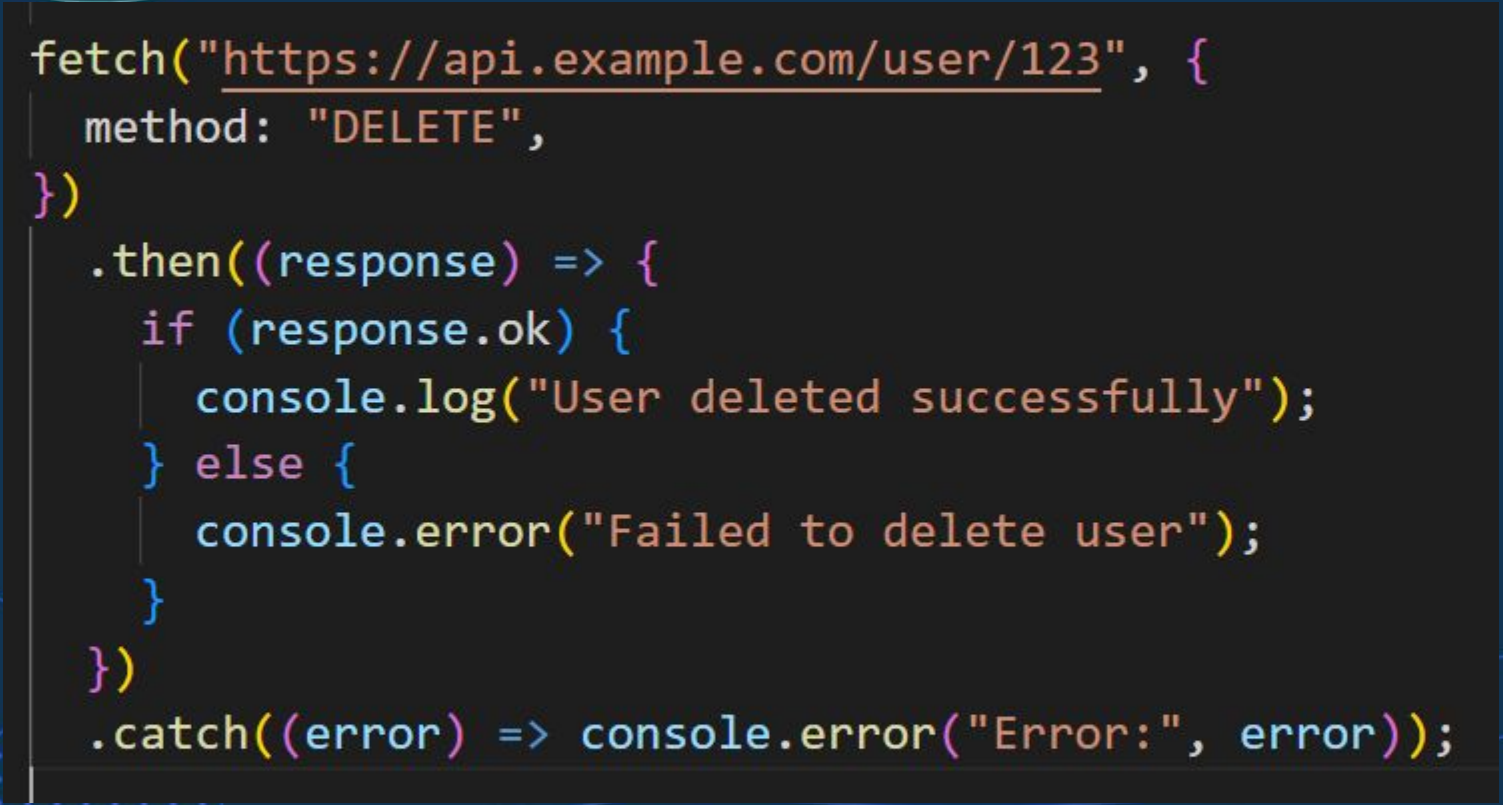


```
const postData = {
  username: "example",
  password: "password123",
};

fetch("https://api.example.com/login", {
  method: "POST",
  headers: {
    "Content-Type": "application/json",
  },
  body: JSON.stringify(postData),
})
  .then((response) => response.json())
  .then((data) => console.log(data))
  .catch((error) => console.error("Error:", error));
```

CoGrammar

Making a DELETE Request



```
fetch("https://api.example.com/user/123", {
  method: "DELETE",
})
  .then((response) => {
    if (response.ok) {
      console.log("User deleted successfully");
    } else {
      console.error("Failed to delete user");
    }
  })
  .catch((error) => console.error("Error:", error));
```

CoGrammar

AJAX Task

Need a quick dose of inspiration? 📝 In this task, you'll create a Daily Inspiration Generator that fetches a random motivational quote or advice from an external API. Each time the user clicks for inspiration, your app will use AJAX and Fetch to retrieve a new quote in JSON format, bringing fresh motivation to the screen.

This task will strengthen your understanding of AJAX with Fetch and JSON handling, while making your application dynamic and engaging.

JSON Task

Imagine having a daily goal tracker that reminds you of your progress and keeps you motivated with personalized messages. 🎯 In this task, you'll create a Goal Tracker that stores your goals for the day, shows your current progress, and gives motivational tips when you hit key milestones. You'll use JSON to structure goal data and session storage to save and retrieve this data during the session.

This task will help you master JSON data handling and storing user-specific information, turning your website into an interactive goal-checker!

What is JSON commonly used for in web development?

A. Storing styles for a webpage
B. Structuring data to send and receive with APIs
C. Running server-side scripts
D. Adding animations to the webpage

CoGrammar

What is an advantage of using AJAX with Fetch?

A. It allows asynchronous data requests without reloading the page
B. It clears the browser’s cache
C. It manages CSS files
D. It adds styles to HTML

CoGrammar

Q & A SECTION

Please use this time to ask any questions relating to the topic, should you have any.

Thank you for attending

CoGrammar

SKILLS FOR LIFE SKILLS BOOTCAMPS

Department for Education