CoGrammar

Welcome to this session: Asynchronous JavaScript (Promises & Async/Await)

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.

What is an application?

A. A physical device
B. A software program
C. A JavaScript method
D. An API

CoGrammar

What is an interface?

A. A browser
B. A point of interaction between systems
C. A user input method
D. A type of API

What does API stand for?

A. Application Program Interface
B. Advanced Programming Interaction
C. Application Process Interchange
D. Automated Program Interface

CoGrammar

Learning Outcomes

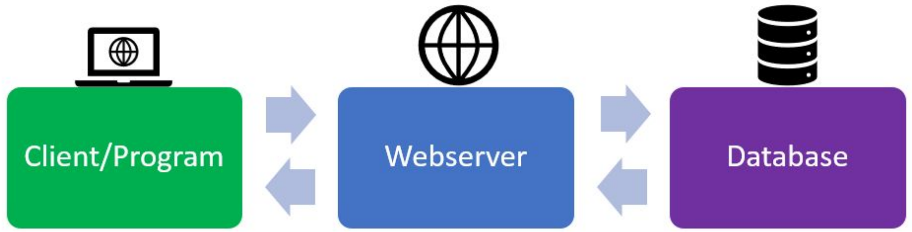
- Distinguish between synchronous and asynchronous operations in JavaScript.
- Utilize Promises and the async/await syntax to manage asynchronous code effectively.
- Discuss the basics of working with APIs using JavaScript.

Lecture Overview

→ An introduction to APIs
→ Asynchronous VS Synchronous
→ Promises VS Async/Await

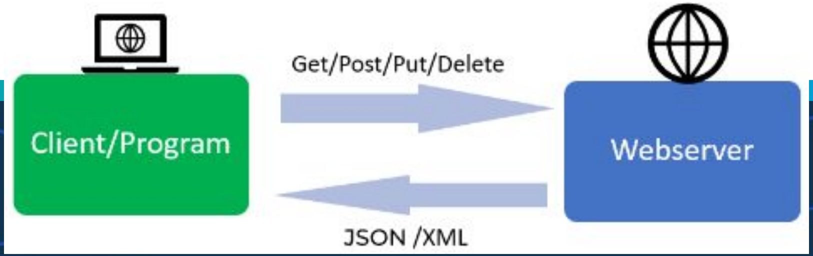
AN INTRODUCTION TO APIs

- ❖ **API** stands for **A**pplication **P**rogramming **I**nterface.
- ❖ An **“application”** refers to any software that interacts with other software or external services.
- ❖ An **“interface”** is the point where these interactions occur, allowing different programs to communicate with each other.



AN INTRODUCTION TO APIs

- ❖ Requests are used to send your information to the server, asking it to perform a set of tasks.
- ❖ Examples of requests of CRUD (Create, Read, Update, and Delete) operations:
  - ➢ **GET** - Request or **read** data that **exists** within the database.
  - ➢ **POST** - We may want to **create** new (add new)items to the database.
  - ➢ **PUT** - This method allows you to **update existing data** on the server, such as changing the image of an item or updating the entire item's details.
  - ➢ **PATCH** - Unlike PUT, which updates the entire resource, PATCH is used to make **partial updates** to **existing data**.
  - ➢ **DELETE** - As the name suggests, this **deletes** an item from a resource.



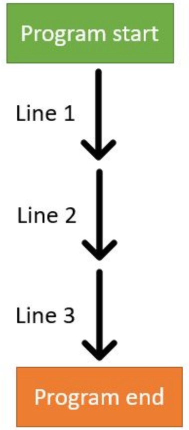
CoGrammar

AN INTRODUCTION TO APIs

- ❖ While APIs can be used for many different tasks, there are some limitations, especially when it comes to access and security.
- ❖ Many websites use API keys to control who can use their API.
- ❖ An API key is a special code, like a unique password, that you include in your requests to the API.
- ❖ It helps identify you to the API and checks what you're allowed to do. Using an API key only allows you to perform actions that the key allows, and the permissions for the key will be set by the API.
- ❖ Therefore, it's important to always read through the API documentation to get a better understanding of what you are allowed or not allowed to do.

ASYNCHRONOUS VS SYNCHRONOUS

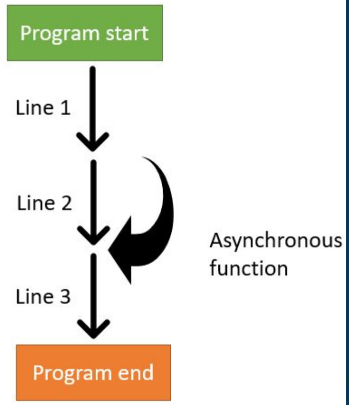
- ❖ Synchronous processing:
  - ➢ Code runs in order, line by line.
  - ➢ Each line of code waits for the one before it to finish before running.
  - ➢ The line below the current line of code won't run until the current line of code is finished running.
  - ➢ This approach is safer because the flow of control moves predictably through the code.
  - ➢ Each line runs one after the other from start to finish.



CoGrammar

ASYNCHRONOUS VS SYNCHRONOUS

❖ Asynchronous processing:
  - ➢ Asynchronous processing is slightly more complex as it involves multiple sets of code being run at the same time.
  - ➢ If this process is not used properly, it can lead to more errors than synchronous programming because different parts of the code might run at the same time and also depend on each other, causing issues.
  - ➢ The general rule of thumb is to only run an asynchronous function if no other code is dependent on it.
  - ➢ Promises are asynchronous and run separately from your normal code.



CoGrammar

Let's take a break

CoGrammar

WHAT IS A PROMISE IN JAVASCRIPT?

- ❖ A promise is an object representing the eventual completion (or failure) of an asynchronous operation and its resulting value.
- ❖ Key Characteristics:
  - ➢ **Pending**: The promise is in the initial state, before the asynchronous operation has finished.
  - ➢ **Resolved** (Fulfilled): The promise has successfully completed, and the operation returned a result (or value).
  - ➢ **Rejected**: The promise has failed, and the operation returned an error or failure reason.

CoGrammar

WHAT IS A PROMISE IN JAVASCRIPT?

- ❖ A promise is an object representing the eventual completion (or failure) of an asynchronous operation and its resulting value.
- ❖ Purpose:
  - ➢ Promises allow you to manage asynchronous operations more effectively, making it easier to handle multiple steps of asynchronous code without "callback hell."
- ❖ Chaining:
  - ➢ Promises support chaining through *.then()* for handling successful resolutions and *.catch()* for handling errors. This makes it easier to handle sequences of asynchronous tasks.

WHAT IS A PROMISE IN JAVASCRIPT?

- A promise is an object representing the eventual completion (or failure) of an asynchronous operation and its resulting value.
- Benefits:
  - **Avoids “callback hell”**: Promises make asynchronous code more readable and maintainable.
  - **Error handling**: Promises make error handling more structured and avoid nested error handling inside callbacks.
- General syntax on how to use a promise:
  - doSomethingThatReturnsAPromise().then(successCallback, failureCallback);

CoGrammar

WHAT IS A PROMISE IN JAVASCRIPT?

- ❖ Example of a function the returns a promise:
  - ➢ fetch(“https://api.example.com/data”)

    .then(response => response.json())

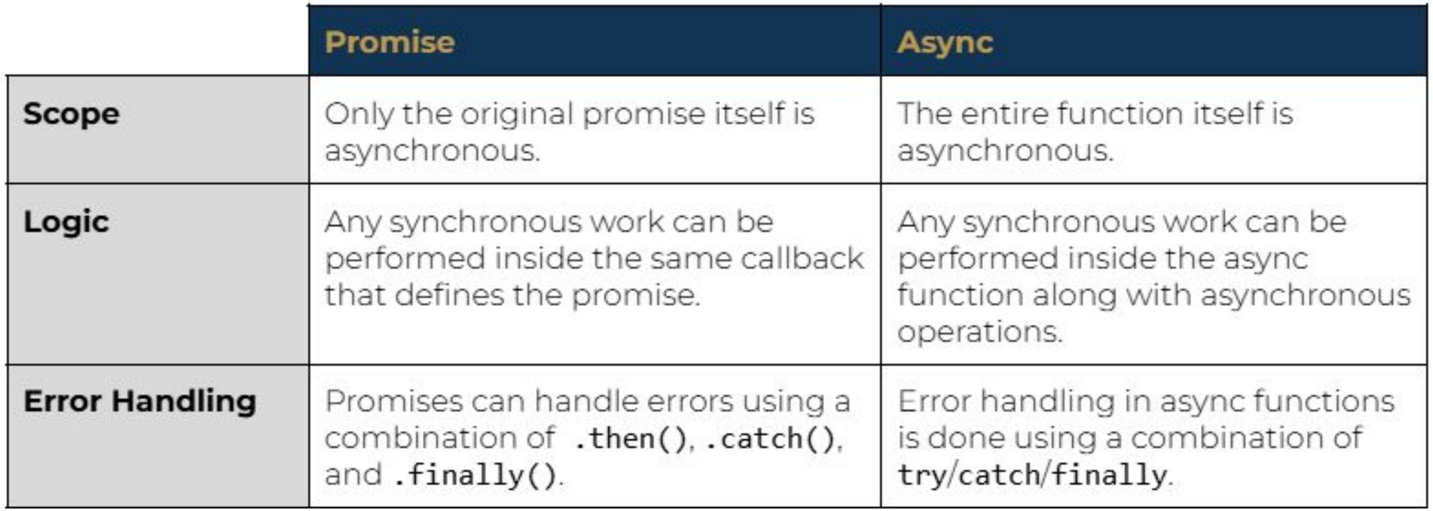
    .then(data => console.log(data))

    .catch(error => console.error('Error:', error));

CoGrammar

PROMISES VS ASYNC/AWAIT

❖ The key differences between the two concepts.



	Promise	Async
Scope	Only the original promise itself is asynchronous.	The entire function itself is asynchronous.
Logic	Any synchronous work can be performed inside the same callback that defines the promise.	Any synchronous work can be performed inside the async function along with asynchronous operations.
Error Handling	Promises can handle errors using a combination of `.then()`, `.catch()`, and `.finally()`.	Error handling in async functions is done using a combination of `try/catch/finally`.

CoGrammar

PROMISES VS ASYNC/AWAIT

- When to use a promise?
  - Promises are a good option should you want to quickly but also concisely grab results from a promise.
  - Promises are a good choice to avoid writing multiple wrapper functions inside an async when a simple **.then** will suffice.

CoGrammar

PROMISES VS ASYNC/AWAIT

- ❖ When to use async?
  - ➢ You’ll typically make use of asynchronous functions when you are using complex code that needs to run separately from the rest of the code.

CoGrammar

PROMISES VS ASYNC/AWAIT

- Using await
  - The await keyword basically tells an async function to wait for the other function’s process to complete before running the rest of the code.
  - This is perfect if you require a set of content to continue the process.

CoGrammar

What is the main purpose of Await?

A. To delay execution for debugging
B. To pause execution until a Promise resolves
C. To create a new Promise
D. To handle errors in async code

CoGrammar

What is the purpose of a Promise in JavaScript?

A. To store values asynchronously.
B. To handle asynchronous operations and manage their outcomes.
C. To execute synchronous code in parallel.
D. To handle errors in code execution.

CoGrammar

What is meant by "Callback Hell" in JavaScript?

A. When multiple callbacks are nested inside one another, making the code difficult to read and maintain.
B. A situation where callbacks are executed too early and cause issues with timing.
C. A scenario where callbacks throw errors continuously.
D. When callbacks are written without proper function declarations.

CoGrammar

Questions and Answers

CoGrammar

Thank you for attending

CoGrammar

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education