CoGrammar

Welcome to this session:

Task Walkthrough - Functions, Scope and Closure

The session will start shortly...

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

Safeguarding & Welfare

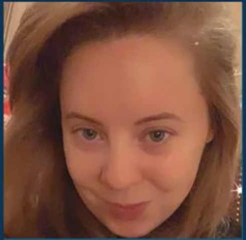
We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

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

Rafiq Manan



Charlotte Witcher

Nurhaan Snyman

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 multiple functions** in JavaScript to handle user input and generate dynamic content.
- ❖ **Use functions** to process user responses and generate an interactive narrative.
- ❖ **Understand and apply closures** by creating a secret-keeping mechanism in a game.
- ❖ **Explain how closures work** to protect and control access to stored values in JavaScript.

Lecture Overview

→ Presentation of the Task
→ Functions
→ Scope
→ Nested Functions
→ Task Walkthrough

CoGrammar

Functions Task

Ever dreamed of going on a grand adventure where you get to decide what happens next? Well, now you can be the hero of your own interactive story! 🎉 In this task, you'll build a story generator that takes your choices and creates a thrilling, personalised journey. Whether you're braving the depths of a forest or sneaking into a mysterious castle, your decisions shape the entire story.

Create functions that ask for your character's name, the path they take, and the action they choose, and watch the magic unfold as your very own adventure story comes to life! 🌲🏰

Write multiple functions that ask the user for input and generate a unique, personalised story based on their responses.

Closures Task

Imagine you're locked in a mysterious escape room, and the only way out is by cracking the secret code! ⏳ In this task, you'll build a thrilling escape room game using JavaScript closures. The secret code to unlock the door is hidden inside your program, and only your smart guesses can set you free!

Your mission? Write code to store the secret passcode, and let the player guess the combination. Will they escape the room or stay trapped forever? The game is in your hands – let the challenge begin! 🔐🗝️

The passcode is hidden using a closure; create a function which returns a function which checks the player's guesses one by one.

What is the purpose of a function in JavaScript?

A. To store multiple values.
B. To group code that can be reused and executed when called.
C. To create a loop.
D. To declare a variable.

How do you pass data into a function in JavaScript?

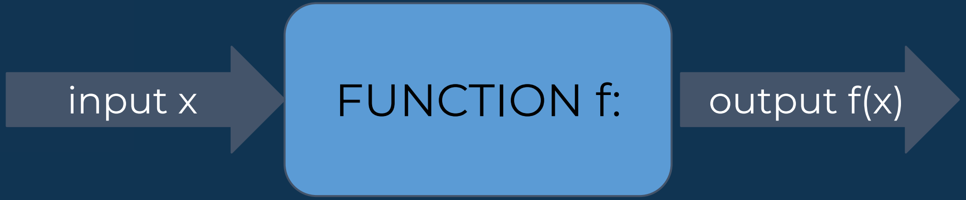
A. Using loops.
B. Using parameters.
C. Using arrays.
D. Using if statements.

CoGrammar

Functions

A block of organised, reusable code that accomplishes a specific task.

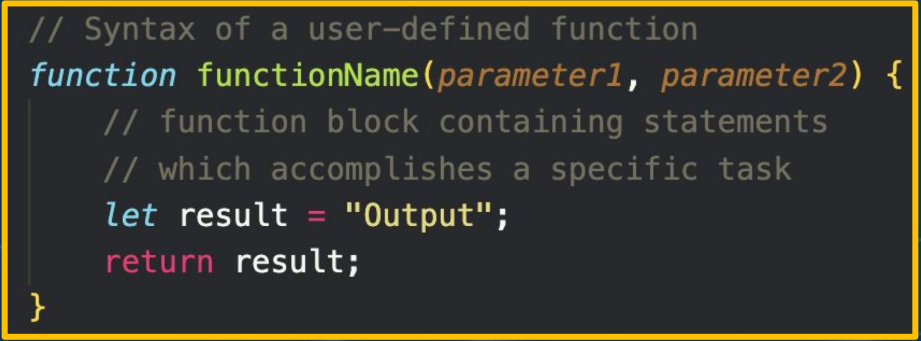
- A function can be **called repeatedly** throughout your code.
- Functions can either be **user-defined** or **built-in**.
- This helps us **minimise repeating lines of code** unnecessarily.
- The main benefits of using functions are:
  - It improves code **modularity, management** and **maintenance**.
  - It makes our code more **readable**.
  - It **reduces potential errors**.



CoGrammar

Functions

- ❖ To declare a function in JavaScript, we use the **function** keyword.
- ❖ We have to provide a **name** for our function (using variable naming conventions), a list of **parameters** (placeholders for function inputs) in brackets and the **body** of the function in curly brackets
- ❖ We also need to add a **return statement** for functions that return a value. This is not necessary for all functions e.g. functions that modify a state.

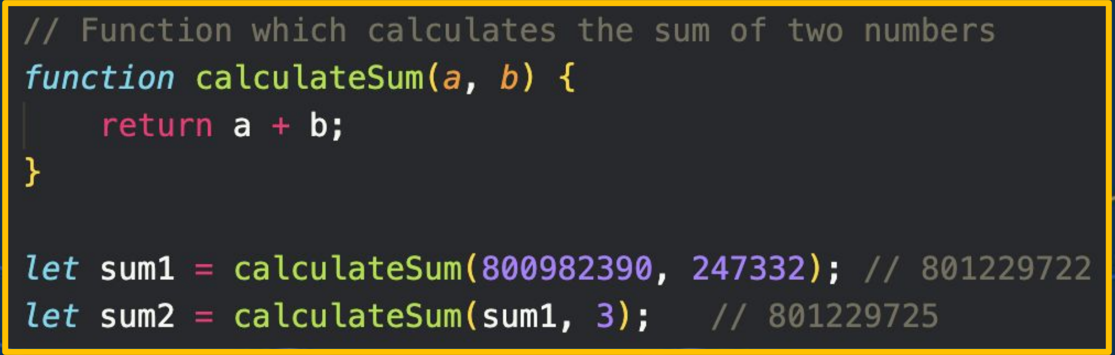


```javascript
// Syntax of a user-defined function
function functionName(parameter1, parameter2) {
    // function block containing statements
    // which accomplishes a specific task
    let result = "Output";
    return result;
}
```

CoGrammar

Functions

- After defining a function, we **call or invoke** it to use it in our code.
- We call a function with its name followed by a list of **arguments** enclosed in brackets, if required by the functions.
- **Arguments** are the input values provided to the function and take the place of the **parameters** defined in the function in the same **same position**.



```
// Function which calculates the sum of two numbers
function calculateSum(a, b) {
    return a + b;
}

let sum1 = calculateSum(800982390, 247332); // 801229722
let sum2 = calculateSum(sum1, 3);    // 801229725
```

CoGrammar

Scope

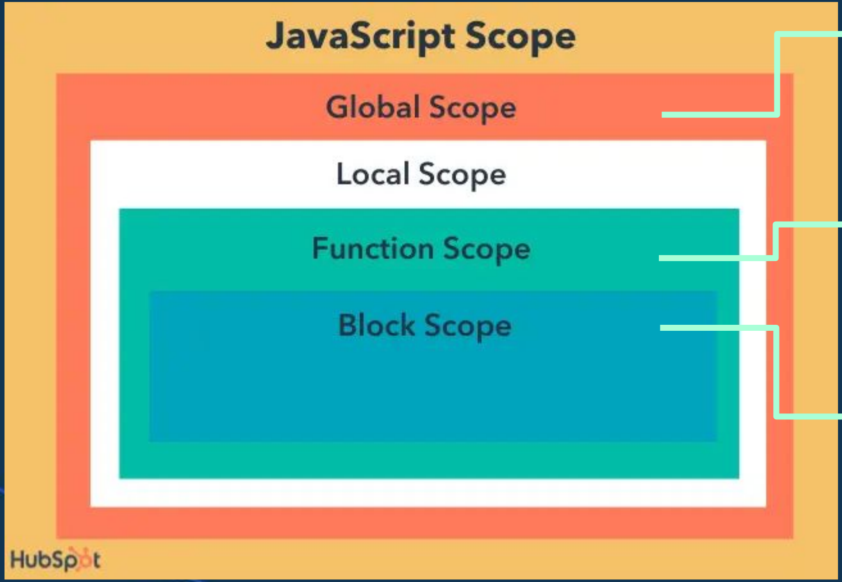
The area of visibility and accessibility of a variable in a program.

- ❖ The **scope** of a variable determines **where in the code it can be seen**.
- ❖ JavaScript has **function scope**, meaning variables declared **inside a function** are only **accessible within** that function.
- ❖ Variables declared outside of a function, known as **global variables**, can be accessed anywhere (**hoisting** allows for variables to be accessed before their definition).
- ❖ JavaScript has **three types of scope:**

  - ➢ Global Scope
  - ➢ Function Scope
  - ➢ Block Scope

CoGrammar

Scope



Source: HubSpot

Global Scope: variables declared outside all functions or blocks. They can be accessed from any part of the code.

Function Scope: variables declared within a function. They are only accessed within their function body.

Block Scope: variables declared with the **let** or **const** keyword inside a block. They can only be accessed in their block (does not apply to **var** keyword).

CoGrammar

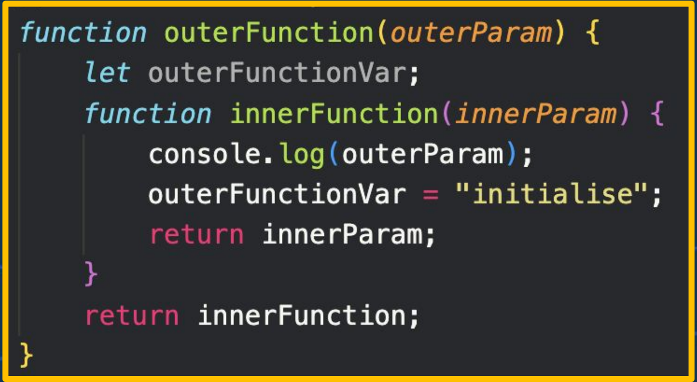
Nested Functions

A function that is defined inside another function.

❖ The **nested function** is referred to as the **inner function** and the **containing function** is known as the **outer function**.

❖ Nested functions can only be called **within the containing function.**

❖ A nested function forms a **closure**, the function has its **own local variables and parameters** and is able to reference and use its containing **function's function variables and parameters**.



```
function outerFunction(outerParam) {
    let outerFunctionVar;
    function innerFunction(innerParam) {
        console.log(outerParam);
        outerFunctionVar = "initialise";
        return innerParam;
    }
    return innerFunction;
}
```

CoGrammar

Functions Task

Ever dreamed of going on a grand adventure where you get to decide what happens next? Well, now you can be the hero of your own interactive story! 🎉 In this task, you'll build a story generator that takes your choices and creates a thrilling, personalised journey. Whether you're braving the depths of a forest or sneaking into a mysterious castle, your decisions shape the entire story.

Create functions that ask for your character's name, the path they take, and the action they choose, and watch the magic unfold as your very own adventure story comes to life! 🌲🏰

Write multiple functions that ask the user for input and generate a unique, personalised story based on their responses.

Closures Task

Imagine you’re locked in a mysterious escape room, and the only way out is by cracking the secret code! ⏳ In this task, you’ll build a thrilling escape room game using JavaScript closures. The secret code to unlock the door is hidden inside your program, and only your smart guesses can set you free!

Your mission? Write code to store the secret passcode, and let the player guess the combination. Will they escape the room or stay trapped forever? The game is in your hands – let the challenge begin! 🔐🗝️

The passcode is hidden using a closure; create a function which returns a function which checks the player’s guesses one by one.

What is a closure in JavaScript?

A. A function inside another function that retains access to the outer function’s variables.
B. A loop that runs indefinitely.
C. A way to store multiple strings.
D. A way to declare variables globally.

CoGrammar

What is the purpose of scope in JavaScript?

A. To declare multiple variables at once.
B. To control where variables are accessible in the code.
C. To create functions.
D. To repeat a block of code.

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