Explain what is execution context in detail with diagram

In JavaScript, an execution context is an environment in which JavaScript code is executed. It consists of variables, functions, and the scope chain that determines the accessibility of variables and functions during runtime. Understanding execution context is crucial for understanding how JavaScript code is executed.

Components of an Execution Context:

Variable Object (VO): The variable object is an internal JavaScript object that contains all the variables, function declarations, and function arguments defined within the context. It stores the identifier-variable mappings.

Scope Chain: The scope chain is an ordered list of variable objects that are accessible within the current context. It is used to resolve variable names during runtime. The scope chain ensures lexical scoping, allowing inner functions to access variables from their outer function scopes.

'this' Value: The 'this' value refers to the object that is currently executing the code. It is determined by how a function is called and can vary depending on the context of the function invocation.

Creation of Execution Context: When JavaScript code is executed, execution contexts are created in a specific order:

Global Execution Context: The global execution context is created first. It represents the default or global scope. It contains variables and functions that are not inside any function.

Function Execution Context: Whenever a function is invoked, a new function execution context is created. Each function has its own execution context, which is pushed onto the call stack.

Execution Context Diagram: Source: freecodecamp

