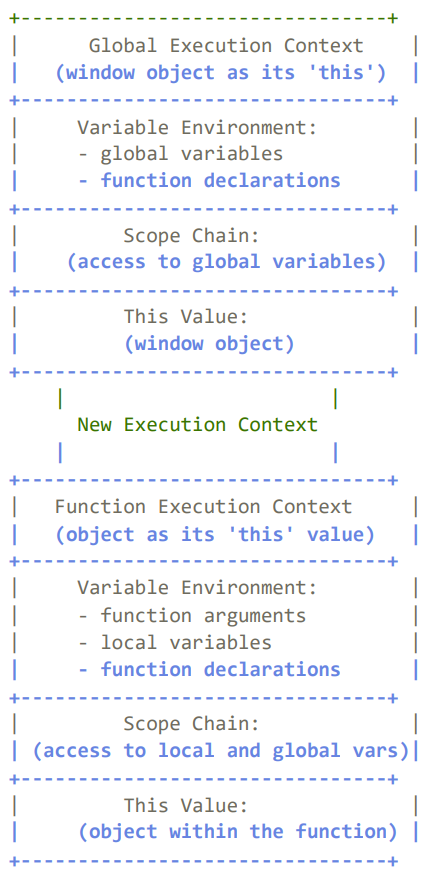
**3. Explain what is execution context in detail with diagram.**

An execution context can be thought of as an environment in which JavaScript code is evaluated and executed. Every time a function is called or code is executed in the global scope, a new execution context is created. Each execution context has its own scope and context-related information.

Let's break down the components of an execution context:

1. Variable Environment:
   * Contains variables and function declarations.
   * Includes function arguments, locally declared variables, and function declarations.
   * Created when an execution context is set up.
   * Variables are initially assigned the value **undefined**.
2. Scope Chain:
   * A list of variables and functions that an execution context has access to.
   * Determines the accessibility of variables.
   * Created when a function is defined and is based on the order of nested functions.
   * Allows accessing variables from parent or outer execution contexts.
3. This Value:
   * Refers to the object that the current function/method belongs to.
   * Provides context-specific reference for the keyword **this**.
   * The value of **this** is determined dynamically when a function is called.

In the diagram, we have a global execution context representing the code executed in the global scope. It has its own variable environment containing global variables and function declarations. The scope chain allows access to the global variables.

When a function is called, a new execution context is created for that function. It has its own variable environment, scope chain, and **this** value. The variable environment includes function arguments, local variables, and function declarations. The scope chain allows access to local and global variables. The **this** value represents the object that the function belongs to.

This process continues as functions are called within functions, creating new execution contexts and building the scope chain accordingly.

Understanding execution contexts helps you grasp how JavaScript manages the flow of code execution, scoping, and variable accessibility.