Execution Context in JavaScript

Overview

An execution context is an abstract concept describing the environment within which JavaScript code is executed. It contains information about the code currently being executed and its scope.

Components of Execution Context

1. Variable Object (VO): Contains function arguments, local variables, and function declarations.
2. Scope Chain: Contains the current variable object as well as the variable objects of all its parent execution contexts.
3. this: A reference to the object that is the current context of the code.

Types of Execution Contexts

1. Global Execution Context:
   - Created when the JavaScript file is first executed.
   - There is only one global execution context.
   - Sets up the global scope and global object (window in browsers).

2. Function Execution Context:
   - Created each time a function is called.
   - Has its own variable object and scope chain.

3. Eval Execution Context:
   - Created when code is executed inside an `eval` function.

Example of Global Execution Context

```
var a = 10;

function foo() {
  var b = 20;
  console.log(a + b);
}

foo();
```

- Global context: `{ a: 10, foo: <function> }`

Example of Function Execution Context

```
var a = 10;

function foo() {
  var b = 20;
  function bar() {
    var c = 30;
    console.log(a + b + c);
  }
  bar();
}

foo();
```

- `foo` context: `{ b: 20, bar: <function> }`
- `bar` context: `{ c: 30 }`

Detailed Execution Context Example

```
var a = 10;

function outer() {
  var b = 20;

  function inner() {
    var c = 30;
    console.log(a, b, c); // 10, 20, 30
  }

  inner();
}

outer();
```

- Global Execution Context:
  - VO: `{ a: 10, outer: <function> }`
  - Scope Chain: `[Global VO]`
  - `this`: `window`

- Outer Function Execution Context:
  - VO: `{ b: 20, inner: <function> }`
  - Scope Chain: `[Outer VO, Global VO]`
  - `this`: `window`

- Inner Function Execution Context:
  - VO: `{ c: 30 }`
  - Scope Chain: `[Inner VO, Outer VO, Global VO]`
  - `this`: `window`

Lifecycle of Execution Context

1. Creation Phase:
   - Global Execution Context: Creates the global object and `this`.
   - Function Context: Creates arguments object, variable object, and `this`.

2. Execution Phase:
   - Executes the code line-by-line.
   - Resolves variables and functions using the scope chain.