In Node.js, global objects are accessible anywhere in the application without requiring import or require. These objects are part of the Node.js runtime environment and provide essential functionality. Here are some key global objects in Node.js:

### **1. global**

* The global object is the global scope in Node.js, similar to window in browsers. It holds globally accessible values, objects, and functions.
* Example: global.console.log("Hello") or just console.log("Hello") as console is globally available.

### **2. process**

* Provides information about and control over the Node.js process, including methods and properties for environment variables, memory usage, and process events.
* Example: process.env.NODE\_ENV to access environment variables.

### **3. \_\_dirname and \_\_filename**

* \_\_dirname gives the directory path of the currently executing file.
* \_\_filename gives the absolute path of the current file.
* Example: console.log(\_\_dirname); prints the directory path.

### **4. Buffer**

* Buffer is a global object in Node.js for handling binary data directly. It allows reading and manipulating binary data.
* Example: const buf = Buffer.from("hello");

### **5. setTimeout, clearTimeout, setInterval, and clearInterval**

* Functions for managing timed operations. setTimeout and setInterval schedule operations, while clearTimeout and clearInterval stop them.
* Example: setTimeout(() => console.log("Delayed"), 1000);

### **6. require and module.exports**

* require is used to include modules, while module.exports and exports allow modules to expose functionality.
* Example: const fs = require('fs');

### **7. console**

* Provides methods like console.log, console.error, and console.warn for outputting information to the terminal.
* Example: console.log("Logging info");

### **8. setImmediate and clearImmediate**

* setImmediate is similar to setTimeout, but it executes after I/O events. clearImmediate cancels setImmediate.
* Example: setImmediate(() => console.log("Immediate execution"));

These objects offer essential tools for Node.js development, simplifying tasks like file handling, process control, and timed operations.