***Node JS is an open-source cross platform runtime environment for executing JavaScript code outside of a browser.***

Node is not a programming language nor a framework.

Node.js was developed by **Ryan Dahl in 2009.**

Node.js runs ***single-threaded, non-blocking, asynchronous programming***, which is very memory efficient.

Node.js uses ***Single threaded event loop architecture*** to handle multiple concurrent clients.

* Node is used to build back-end services (like APIs).
* Node is ideal to build highly scalable, I/O-intensive apps (disk & network intensive), data intensive and real-time backend apps.
* Great for prototyping and agile development
* Superfast and highly scalable
* JavaScript everywhere
* Cleaner and more consistent codebase
* Large ecosystem of open-source libraries

Node should **not** be used for

* CPU-intensive apps (video encoding and image manipulation service)

**Node JS architecture**

Graphical user interface, application

Description automatically generated

C++ program

**How to run the first program**

Step 1- Install node

Step 2- Create a folder and cd to that folder

Step 3- Create a new JS file and run it using **node <filename.js>**

***Node don’t have window or document objects (no global object) as browser JS engine.***

[How Node JS Works? - YouTube](https://www.youtube.com/watch?v=YSyFSnisip0)

**Node is non-blocking asynchronous |** Node applications are **asynchronous** by default

2

A screenshot of a computer

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated with medium confidence

1

The global scope in browser JS file (every variable & function is attached to global scope).

**Problem with global scope:** It is possible to have two files having the same name function which will override the previous function.

To build reliable and maintainable application we should avoid defining var and fun in global scope. Instead, we need modularity.

**Node Module**

In Node every file is called as module and each variable and function is scope to that module (In OOP it terms as private).

They are not available outside of the module. We need to explicitly need to export them and make it public.

Module is not global object. // module != global.module

**Export Module**

module.exports.< ContentNameToExport > = < function >; or

exports.< ContentNameToExport > = < function >;

//export an object of key value pair of { ContentNameToExport : fun }

module.exports = < function >; //export a single function.

**Import Module**

const x = require( < JS File location & name > ) //take one argument

.js extension can be remove from file name and node by default add .js extension to file

x.< contentNameToExport >

Note- To get predefine module (os, http, event, path etc) and method visit node website > document section.

***In the top-level code in a Node module, this is equivalent to module.exports. That's the empty object you see.***

Node.js comes with ***REPL***, an abbreviation for read–eval–print loop. REPL contains three different states:

* a **read** state where it reads the input from a user
* the **eval** state where it evaluates the user’s input
* the **print** state where it prints out the evaluation to the console.
* //node is typed in the console to access REPL
* $ node
* //the > indicates that REPL is running
* // anything written after > will be evaluated
* > console.log("HI")
* // REPL has evaluated the line and has printed out HI
* HI