**Node.js**

**Introduction**

- Until Node.js introduced , javascript was only a client-side language and it was executed only on Browsers

- Node.js is a javascript Runtime Environment built on **Chrome’s V8 Javascript engine** , which lets Javascript to run on server-side as well .

- V8 is Google’s open source high-performance JS engine written in C++ , used in chrome and node.js

- V8 is the thing that takes out JS and executes on Chrome or It provides the run-time environment for JS

- Runtime Environment means , the environment in which a program or application is executed . Ex:- For JS back then browsers were only the runtime , but now local system too has become runtime for JS .

- Node.js is designed to built scalable network applications

- It is free and open source server environment

- It can run on multiple OS

- It allows us to run JS on the server or outside of the browser .

**Why use Node.js**

- By Node.js , we can use JS in the entire stack

- Many famous companies use it as their backend

- It comes with lots of useful built-in modules

- It is very fast, lightweight and efficient

**REPL(Read-Evaluate-Print-Loop)**

- REPL is an interactive shell that processes Node.js expressions and it is activated in terminal on writing ‘node’ command

- In REPL underscore or ‘**\_**’ holds the previous variable

- On hitting tab bar twice in REPL all the global variables/modules will be printed and these variables are always available to use and need not to be imported in a node.js file

**Built-in Modules**

- Basically , module is blocks of encapsulated codes written by other coders and can be used by any user simply by Importing them in our file , in order to achieve any functionality .

- In node.js , Modules are the blocks of encapsulated code that communicates with an external application on the basis of their functionality .

- Modules can be a single file or can be a collection of multiple files , these files consist methods , classes and other relevant stuffs .

**Types of modules in Node.js**

1 . Core modules / built-in modules

- These are the built-in modules that are the part of the platform and comes with Node.js installation .

- These modules can be loaded into the program by using th require () method .

ex: - http - creates an HTTP server in Node.js

fs - used to handle file-system

process - provides information and control about the current Node.js process .

os - provides information about the operating system. etc

2 . Local modules/User-defined modules

- Unlike built-in and external modules , local modules are creates locally by the user in the Node.js application . One can create his own local module . Ex:- Let’s create our own calc.js module

Filename: calc.js

exports.add = function (x, y) {

return x + y; };

exports.sub = function (x, y) {

return x - y; };

exports.mult = function (x, y) {

return x \* y; };

exports.div = function (x, y) {

return x / y; };

3 . Third-party Modules

- Third-party modules are modules that are available online and one can access it using the Node Package Manager(NPM)

- These modules can be installed in the project folder of globally

ex:- mongoose(npm install mongoose) , express(npm install express) , angular(npm install -g@angular/cli) and react

**const fs = require("fs");**

- It returns an object

- require(“module\_name”) is an **include-module** used to include or load JS-modules .

- require() module reads a JS-file , executes the file , and then proceeds to return the **exports object**.

- **fs** module stands for **file-system** module which allows you to work with the file system on our computer .

**const text = fs.readFileSync(“text\_file\_name” , “encoding”) -** This method is an inbuilt application programming interface of **fs** module which is used to read the file and return its content .

- utf-8(Unicode Transformation Format) is preferred encoding for e-mail and web-pages

**text = text.replace("fun","run");** -This method replace run with fun , the text file

**fs.writeFileSync("text\_new.txt",text);** - This will write the content of text into text\_new.text file , and here we don’t need to store the statement in a variable