Node Js

What is Node.js?

* **Node is not a Languages**
* **Node.js is an open source server environment**
* **Node can connect with database but Javascript Not direct**
* **Code & syntex is very similar to javascript but Not exactly same**
* **Node.js is free**
* **Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)**
* **Node.js uses JavaScript on the server**
* **Node.js runs the chrome V8 JavaScript engine, the core of Google Chrome, outside of the browser.**

* **Written in c,c++,javascript**

**==============================================================**

* It is a popular tool for almost any kind of project!
* Most useful for create api and also create website
* A Node.js app is run in a single process, without creating a new thread for every request.

* Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm When Node.js needs to perform an I/O operation, like reading from the network, accessing a database or the filesystem, instead of blocking the thread and wasting CPU cycles waiting, Node.js will resume the operations when the response comes back.

Why do we use Node.js?

* **Node Mostly used for api create (Fast API WORK)**
* **So we can connect the same database with web app, Mobile App**
* **Node easy because who know javascript**
* **Node super fast for APIs**
* **With Node & Javascript , you can become full stack Developer / Mern Stack**

**====================================================================**

Javascript and Node are same?

* **Javascript and Node.js code syntax is same**
* **If you know javascript then you can learn node very easily**
* **But Both are note same**
* **You can not connect javascript with database**
* **Node js run on server side**
* **Javascript run on browser side**

**===================================================================**

**Topic**

* **Node js Introduction**
* **How works**
* **Install and run**
* **Make basic api**
* **Use with express.js**
* **UI with Node**
  + **UI EVENT**
  + **FORMS**
  + **WEB PAGES**

**====================================================================**

Node.js uses asynchronous programming!

A common task for a web server can be to open a file on the server and return the content to the client.

Here is how PHP or ASP handles a file request:

1. Sends the task to the computer's file system.
2. Waits while the file system opens and reads the file.
3. Returns the content to the client.
4. Ready to handle the next request.

Here is how Node.js handles a file request:

1. Sends the task to the computer's file system.
2. Ready to handle the next request.
3. When the file system has opened and read the file, the server returns the content to the client.

Node.js eliminates the waiting, and simply continues with the next request.

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

=================================================

What Can Node.js Do?

* Node.js can generate dynamic page content
* Node.js can create, open, read, write, delete, and close files on the server
* Node.js can collect form data
* Node.js can add, delete, modify data in your database

==============================================

What is a Node.js File?

* Node.js files contain tasks that will be executed on certain events
* A typical event is someone trying to access a port on the server
* Node.js files must be initiated on the server before having any effect
* Node.js files have extension ".js"

=================================================

Install and Setup Node JS

* **Download Node**

The official Node.js website has installation instructions for Node.js: [https://nodejs.org](https://nodejs.org/)

* **Install NPM AND NODE  => node js**
* **Code Editors**

**VS CODE**

**Sublime**

**========================================================================**

First Script with Node

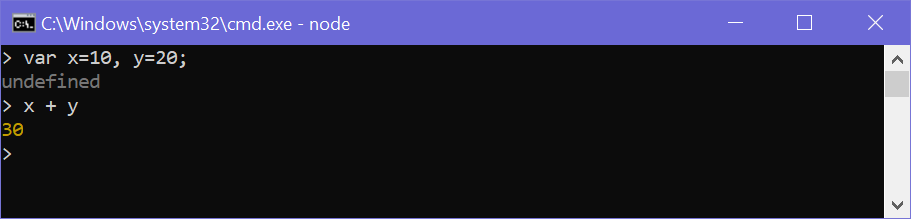
Node.js Console/REPL Terminal

Node.js comes with virtual environment called REPL (aka Node shell). REPL stands for Read-Eval-Print-Loop. It is a quick and easy way to test simple Node.js/JavaScript code.

To launch the REPL (Node shell), open command prompt (in Windows) or terminal (in Mac or UNIX/Linux) and type *node* as shown below. It will change the prompt to > in Windows and MAC.

Run : start => node.js

You can also define variables and perform some operation on them.



You can execute an external JavaScript file by executing the node fileName command. For example, the following runs mynodejs-app.js on the command prompt/terminal and displays the result.

mynodejs-app.js Copy

console.log("Hello World");

Now, you can execute mynodejs-app from command prompt as shown below.



=================================================

Create a Node.js file named "myfirst.js", and add the following code:

Myfirst.js

Code : console.log(‘Hello World’);

Run : node Myfirst

Output : Hello World’

The following table lists important REPL commands.

|  |  |
| --- | --- |
| REPL Command | Description |
| .help | Display help on all the commands |
| tab Keys | Display the list of all commands. |
| Up/Down Keys | See previous commands applied in REPL. |
| .save filename | Save current Node REPL session to a file. |
| .load filename | Load the specified file in the current Node REPL session. |
| ctrl + c | Terminate the current command. |
| ctrl + c (twice) | Exit from the REPL. |
| ctrl + d | Exit from the REPL. |
| .break | Exit from multiline expression. |
| .clear | Exit from multiline expression. |

**Interview Que**

**Que :why its error come in cmd like undefined :**

Ans : Not return anything

**Que :how to different node console & javascript console**

Ans : We write console in cmd its Node build in module & we write in js file console its javascript browser functionality

in node module provide console for debugging

console.log()

console.error()

console.warn()

======================

With File script

Step: 1 Create file **index.js**

**Code file** : console.log(‘Hello world’);

**Run cmd : node index.js**

Node.js Basics

* Node.js supports JavaScript. So, JavaScript syntax on Node.js is similar to the browser's JavaScript syntax.
* Javascript Fundamental for Node.js
* Condition , Loop , Array
* The Import Function and variables from another file

**NodeJS module system**

Modulein Node.js is a simple or complex functionality organized in single or multiple JavaScript files which can be reused throughout the Node.jsapplication Each module in Node.js has its own context, so it cannot interfere with other modules or pollute global scope. Also, each module can be placed in a separate .js file under a separate folder.

1. Core Modules

2. Own Modules or Local Modules

3. NPM Modules or Third Party Module

* 1. **CoreModules**

Core Modules covers minimum functionality of Node.js

These core modules are compiled into its binary distribution and load automatically when Node.js process starts You only need to import the core module first in order to use it in your application

Loading Core Modules

**var module = require('module\_name')**

|  |  |
| --- | --- |
| **Module** | **Description** |
| [assert](https://www.w3schools.com/nodejs/ref_assert.asp) | Provides a set of assertion tests |
| [buffer](https://www.w3schools.com/nodejs/ref_buffer.asp) | To handle binary data |
| child\_process | To run a child process |
| [cluster](https://www.w3schools.com/nodejs/ref_cluster.asp) | To split a single Node process into multiple processes |
| [crypto](https://www.w3schools.com/nodejs/ref_crypto.asp) | To handle OpenSSL cryptographic functions |
| [dgram](https://www.w3schools.com/nodejs/ref_dgram.asp) | Provides implementation of UDP datagram sockets |
| [dns](https://www.w3schools.com/nodejs/ref_dns.asp) | To do DNS lookups and name resolution functions |
| domain | Deprecated. To handle unhandled errors |
| [events](https://www.w3schools.com/nodejs/ref_events.asp) | To handle events |
| [fs](https://www.w3schools.com/nodejs/ref_fs.asp) | To handle the file system |
| [http](https://www.w3schools.com/nodejs/ref_http.asp) | To make Node.js act as an HTTP server |
| [https](https://www.w3schools.com/nodejs/ref_https.asp) | To make Node.js act as an HTTPS server. |
| [net](https://www.w3schools.com/nodejs/ref_net.asp) | To create servers and clients |
| [os](https://www.w3schools.com/nodejs/ref_os.asp) | Provides information about the operation system |
| [path](https://www.w3schools.com/nodejs/ref_path.asp) | To handle file paths |
| punycode | Deprecated. A character encoding scheme |
| [querystring](https://www.w3schools.com/nodejs/ref_querystring.asp) | To handle URL query strings |
| [readline](https://www.w3schools.com/nodejs/ref_readline.asp) | To handle readable streams one line at the time |
| [stream](https://www.w3schools.com/nodejs/ref_stream.asp) | To handle streaming data |
| [string\_decoder](https://www.w3schools.com/nodejs/ref_string_decoder.asp) | To decode buffer objects into strings |
| [timers](https://www.w3schools.com/nodejs/ref_timers.asp) | To execute a function after a given number of milliseconds |
| [tls](https://www.w3schools.com/nodejs/ref_tls.asp) | To implement TLS and SSL protocols |
| tty | Provides classes used by a text terminal |
| [url](https://www.w3schools.com/nodejs/ref_url.asp) | To parse URL strings |
| [util](https://www.w3schools.com/nodejs/ref_util.asp) | To access utility functions |
| v8 | To access information about V8 (the JavaScript engine) |
| [vm](https://www.w3schools.com/nodejs/ref_vm.asp) | To compile JavaScript code in a virtual machine |
| [zlib](https://www.w3schools.com/nodejs/ref_zlib.asp) | To compress or decompress files |

**Exa 1 : http**

**App.js**

var fs=require('fs');

fs.appendFileSync('demo.txt','Hi in am first created file by fs Core module');

**Exa 2 : http**

**var http = require('http');**

**http.createServer((req,resp)=>{**

**resp.writeHead(200, { 'Content-Type': 'text/html' });**

**//resp.write('<h1>Hi i am http module sserver </h1>')**

**resp.write(JSON.stringify({name:"raj",age:31,mobile:"1234567891"}));**

**resp.end();**

**}).listen(5000);**

* 1. **Own Modules or Local Modules**

**Local modules are modules created locally in your Node.js application.**

**These modules include different functionalities of your application in separate**

**files and folders.**

**You can also package it and distribute it via NPM,**

**so that Node.js communitycan use it.**

**We need module.exports to define what couldbe access from other file**

**todatDate.js**

**// this my custome own /local module**

**var todayDate=()=>{**

**var today = new Date();**

**var date = (today.getDate())+'-'+ (today.getMonth()+1) +'-'+(today.getFullYear());**

**return date;**

**}**

**// local module export**

**module.exports = todayDate ;**

**app.js**

// import

var date=require('./todayDate.js');

var result=date();

console.log(result);