**NODE JS**

**Introduction**

* Node JS is a runtime environment which allow JavaScript to execute outside the browser
* With the help of node JS we can create networking web application
* Node JS mainly used for backend purpose
* Node JS is developed by Ryan dehl in 2009
* Node JS is open source, cross platform runtime environment
* Node JS is asynchronous in nature

**Feature**

1. Single Tread

* Node JS follow single thread model with event looping

1. Cross platform

* It means that software or application can run on multiple os without change major code

1. Very fast

* Build in google chrome’s V8 engine so its library is very fast in code execution

1. Huge package ecosystem

* Node JS include node package manager to manager that access to millions of reusable packages for free

1. Easy to understand
2. Asynchronous in nature

**Advantages of Node JS**

1. Cost effective with fullstack JS
2. Helps to building cross-platform application
3. High performance for real-applications
4. Easy to learn and quick to adapt
5. Reduce loading time

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

* REPL stand for read, evaluate, print, loop
* It is mainly used for testing purpose
* To start REPL simply running node on Command prompt

**Node Module**

**NPM(node package manager)**

* NPM stand for node package manager
* NPM used for manager the third party open source packages
* It is command line interface app automatically come with node JS

**Modules**

* Set of function is called as module
* Reusable block of code is called as module
* Module is same like a library in JavaScript
* To include the module we use require() method or function with named of module ways
* There are 3 type of module

1. Core module
2. Build-in module: fs, http, path, os, dns, net, url etc……..
3. Third-party module

**Global Object**

* It is an object which are accessible in application from anywhere without importing
* No need to import it before use
* Following are different global object

1. \_\_dirname
2. \_\_filename
3. Console
4. Process
5. Buffer
6. Settimeout

1] \_\_dirname:

* return the name of directory where we currently contain code
* example

console.log("Directory name is ---------->  ",\_\_dirname)

2] \_\_filename:

* specifies filename of code being executed
* example

console.log("file name:  ",\_\_filename);

3]console:

* used to show the message to user on console
* it has different method

1. log()
2. error()
3. warn()
4. time()
5. timeend()
6. table()

// A]console.log()

console.log("Hello how are You");

// B] console.error()

console.error("Error mssage")

// C] console.warn()

console.warn("console.warn object")

// D] console.table

console.table([["Akshay", "Sandip", "Sam", "Suraj", "Dhiraj"], [1, 2, 3, 4, 5]])

// E] timeEnd

console.timeEnd("8")

**Process**

* It is global object used to provide the information about the currently running process
* Example

// console.log(process)

console.log(process.pid);

console.log(process.version);

console.log(process.ppid);

**buffer**

* buffer is class used to deal with binary data
* it is temporary raw chunk of data

let buffer = Buffer.from("Akshay")

console.log(buffer);

**stream**

* stream is flow of data
* stream is sequence of data that is being move from one point to another over time
* following are the type of stream

1. readable stream
2. writeable stream
3. transform stream
4. duplex stream

1] Readable Stream:

* used to read data use the Readable Stream
* example

let fs = require("fs")

let readablestream  = fs.createReadStream("./output.txt","utf-8")

// ADD EVENT

readablestream.on('data',chank=>{       // IT IS AN EVENT LISTENER THAT LISTENS FOR DATA EVENT

    console.log(chank);

})

console.log(fs);

2] writeable Stream

* Used to write operation
* example

let writestream = fs.createWriteStream("./write.txt","utf-8")

readablestream.on('data',chunk=>{

    console.log("successfully run chunk");

    writestream.write(chunk,err=>{

        if (err) throw err

        console.log("successfully Written");

        console.log(chunk);

    })

})

let str = "i am learning node js"

writestream.write(str,err=>{

    if (err) throw err

    console.log("data written succesfully..........");

})

Build-In module

1] FS module

* Fs stand for file system
* Fs module is used to work with file system on your computer
* Two ways to work with FS module
  1. Asynchronous
  2. Synchronous

|  |  |
| --- | --- |
| Asynchronous | Synchronous |
| Task are not executed one by one | Task are executed one by one |
| If any task take time to execute then execute next task instead of wait for execute | Wait for execute task which take time to execute |
| Used for fetching data from server, reading file etc | Used for perform simple operation |
| Faster for I/O operation | Slower for I/O operation |

1] Synchronous way

**Method**

**1] readFileSync()**

* Used to read file
* Syntax

Fs.readFileSync(path,character\_encoder)

const fs = require("fs")

let readdata = fs.readFileSync("write.txt")

console.log(readdata.toString())

console.log("Data read succesfully.......");

**2] writeFileSync()**

* Used to write file in synchronous way
* Syntax

Fs.writeFileSync(path,text)

const fs = require("fs")

let p = "i clrear my intervirew and now i want to learn Nodejs"

fs.writeFileSync("output.txt",p)

console.log("write data sucessfully...........");

**3] mkdirSync()**

* Use to create directory
* Syntax

Fs.mkdirSync(foldername)

const fs = require("fs")

if (!fs.existsSync("Qspider")) {

    let dir = fs.mkdirSync("Qspider")

    console.log(dir);

    console.log("folder Created succefully");

} else {

    console.log("folder Already exits...........");

}

**4] unlinkSync()**

* Used to delete file
* Syntax

Fs.unlinkSync(path)

const fs = require("fs")

fs.unlinkSync("write.txt")

console.log("deleted sucessfuly........");

5] rmdirSync()

* Used to remove the directory
* Syntax

Fs.rmdirSync(path)

const fs = require("fs")

fs.rmdirSync("Qspider")

console.log("deleted succesfuly..");

6] renameSync()

Used to rename file

Syntax

Fs.renameSync(oldname,newname)

const fs = require("fs")

fs.renameSync("output.txt","new.txt")

7] appendFileSync()

* Used to append the data in existing file
* Syntax

Fs.appendFileSync(path,data)

const fs = require("fs")

fs.appendFileSync("new.txt","util not i learn basic function of FS module")

console.log("data append succefully...");

**Asynchronous way**

* To overcome synchronous, we have to use asynchronous way

**1] readFile()**

* Used to read file
* Syntax

Fs.readFile(path,code-uncoder,callback function)

fs.readFile("write.txt",(err,data)=>{

    if(err) throw err

    console.log("Fetch data is : ",data.toString());

    console.log("data fetch sucessfully");

})

console.log("operation done");

**2] writeFile()**

* Used to write file in Asynchronous way
* Syntax

Fs.writeFileSync(path,data,callback\_function)

fs.writeFile("output.txt","hello hi how are you",(e)=>{

    if(e) throw e

    console.log("data write sucessfully........");

})

console.log("done........");

**3] mkdir()**

* Use to create directory
* Syntax

Fs.mkdir(foldername,callback\_func)

try {

    fs.mkdir("qspider1", (err) => {

        if (err) {

            console.log(err);

        }

        console.log("Directory created...");

    })

} catch (error) {

    console.error("folder Already exits...........", error);

}

**4] unlink()**

* Used to delete file
* Syntax

Fs.unlink(path,callback\_func)

fs.unlink("./qspider/abc.txt",(err)=>{

   if(err)

    throw err

   console.log("file deleted succesfully........");

})

console.log("done!!");

5] rmdir()

* Used to remove the directory
* Syntax

Fs.rmdir(path,callback\_func)

fs.rmdir("qspider1",(err)=>{

    if (err){

        console.log("Error occure..")

        return

    }

    console.log("deleted............");

})

console.log("done.........");

6] rename()

Used to rename file

Syntax

Fs.rename(oldname,newname,callback\_func)

fs.rename("new.txt", "new1.txt", (e) => {

    if (e) throw e

    console.log("rename done.........");

})

7] appendFile()

* Used to append the data in existing file
* Syntax

Fs.appendFile(path,data,callback\_func)

fs.appendFile("new1.txt","hello hi how are youlearn appendfile function using asynchronuous",(e)=>{

    if(e)

    {

        console.log("error............");

    }

    console.log("Data will be added.......");

})

console.log("operation done......");

Http module

* http stands for hypertext transfer protocol
* http module is used to create server and handle the http request and response
* http module is built in module in node JS allow to create http server that listen to server port & gives response to client
* method

1] createServer():-

with the help of createServer() method we can create the http server