**Nodejs Server and Core Modules**

Nijer banano modules ebong npm er millions of modules charao core nodejs er moddhe besh kichu important built in module ache. Ei module etotai powerful je npm theke external kichu use na kore only core modules diyei ekta bishal enterprize banaya fela possible.

Amra jokhon js niye browser e kaj kori tokhon amra window, DOM egula niye kaj kori. Kintu node js kono machine ba server e run kore. So tar world ta different. Shekhane browser nai. Shekhane ache operating system, file system, network etc. browser e amra ei jinishgular access petam na. kintu nodejs e pabo. Ei system related jinishgular access paoar jonnoi nodejs er core e kichu built in module diye deya hoise. Amra nodejs.org er docs e giye egula dekhbo. Abar w3schools er site e giyeo dekhte pari core modules er list ta.

At first amra path module ta dekhbo. Amra kon path e achi ba kon directory te achi, absolute path naki relative path e achi etc ei path module diye jana jay.

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

*const* myPath =

  "E:/Web Developement Practice/W3schools and Youtube/Node js/Sumit Nodejs/Nodejs Practice/ind.js";

console.*log*(path.*basename*(myPath));

myPath e forward slash kore nite hobe.

Terminal o/p ----

Ind.js

Orthaat basename ta hocche file er name.

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

*const* myPath =

  "E:/Web Developement Practice/W3schools and Youtube/Node js/Sumit Nodejs/Nodejs Practice/ind.js";

console.*log*(path.*dirname*(myPath));

terminal o/p ----

E:/Web Developement Practice/W3schools and Youtube/Node js/Sumit Nodejs/Nodejs Practice

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

*const* myPath =

  "E:/Web Developement Practice/W3schools and Youtube/Node js/Sumit Nodejs/Nodejs Practice/ind.js";

console.*log*(path.*extname*(myPath));

terminal o/p ----

.js

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

*const* myPath =

  "E:/Web Developement Practice/W3schools and Youtube/Node js/Sumit Nodejs/Nodejs Practice/ind.js";

console.*log*(path.*parse*(myPath));

terminal o/p ----

{

root: 'E:/',

dir: 'E:/Web Developement Practice/W3schools and Youtube/Node js/Sumit Nodejs/Nodejs Practice',

base: 'ind.js',

ext: '.js',

name: 'ind'

}

Ekhon amra OS module niye khela korbo….

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

console.*log*(os.*platform*());

t.o.p ----

win32; which is operating system platform

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

console.*log*(os.*homedir*());

t.o.p ---

C:\Users\UseR; which is the home directory of the operating system.

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

console.*log*(os.*freemem*());

t.o.p ---

2756493312; the amount of free system memory in bytes as an integer.

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

console.*log*(os.*cpus*());

an array of objects containing information about each logical CPU core.

[

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 31243046,

nice: 0,

sys: 24129625,

idle: 776081140,

irq: 6231859

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 12323468,

nice: 0,

sys: 6148468,

idle: 812982093,

irq: 153468

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 70263109,

nice: 0,

sys: 12423265,

idle: 748767484,

irq: 286312

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 26823609,

nice: 0,

sys: 8705578,

idle: 795924828,

irq: 257015

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 43708250,

nice: 0,

sys: 13027312,

idle: 774718390,

irq: 395828

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 10502421,

nice: 0,

sys: 5464625,

idle: 815486921,

irq: 210671

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 39256140,

nice: 0,

sys: 10258453,

idle: 781939390,

irq: 327109

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 12544843,

nice: 0,

sys: 4670234,

idle: 814238875,

irq: 137500

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 36289140,

nice: 0,

sys: 9525406,

idle: 785639359,

irq: 490187

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 12444656,

nice: 0,

sys: 4697812,

idle: 814311468,

irq: 139046

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 27252765,

nice: 0,

sys: 9777906,

idle: 794423312,

irq: 304890

}

},

{

model: '11th Gen Intel(R) Core(TM) i5-11400 @ 2.60GHz',

speed: 2592,

times: {

user: 17690546,

nice: 0,

sys: 5913109,

idle: 807850312,

irq: 130250

}

}

]

----------------------------------------------------------------------------

Ekhon amra file system module niye alochona korbo. Fs module ta file system niye kaj kore. File system niye basically amra ki ki kaj kori? File read ba write kori, rename kori, delete kori etc.

Prothome amra ekta file write korbo. Notun ekta file create kore tate kichu likhe save korbo.

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

fs.*writeFileSync*("myFile.txt", "Hello World");

erpor terminal e node ind korlam. Tar fole myFile.txt name ekta file create holo. Ebong tar content holo ‘Hello World’.

Ekhon code ta ke ektu modify korbo.

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

fs.*writeFileSync*("myFile.txt", "Hello World");

fs.*writeFileSync*("myFile.txt", "I eat rice.");

er fole myFile.txt file er ‘Hello World’ lekhata replaced hoye ‘I eat rice.’ Hoye gese. Ekhon ami jodi replace na kore append korte ba add korte chai tahole following way te kajta korbo.

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

fs.*writeFileSync*("myFile.txt", "Hello World! ");

fs.*appendFileSync*("myFile.txt", "I am a web developer.");

now myFile.txt er content holo “Hello World! I am a web developer.” Now file read kora dekhbo.

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

fs.*writeFileSync*("myFile.txt", "Hello World! ");

fs.*appendFileSync*("myFile.txt", "I am a web developer.");

*const* data = fs.*readFileSync*("myFile.txt");

console.*log*(data);

ekhon node ind korle nicher jinish pai.

<Buffer 48 65 6c 6c 6f 20 57 6f 72 6c 64 21 20 49 20 61 6d 20 61 20 77 65 62 20 64 65 76 65 6c 6f 70 65 72 2e>

Nodejs er ekta data type holo ei buffer. myFile.txt e je data ta ache sheta txt format e ache. Shetari binary form holo buffer. Ekhon console.log(data.toString()) korle ashol text format e data chole ashbe.

“Hello World! I am a web developer.”

Etokkhon amra sync jukto functiongula use korlam.

Nodejs asynchronous non blocking way te by default kaj kore.

Amra je file system theke data read korlam sheta ekta i/o operation. Ei i/o operation er kaj ta main thread er korar kotha na. ei kaj ta tar koranor kotha file system ke diye. Ei jinish ta OS er khetre hobe na orthat OS module e hobe na. shekhane main thread kei kaj ta korte hobe. Kintu file system, database theke data read kora ba http theke data ana ----- ei kajgula onno karo korar kotha. Ekhon syncjukto read or write function gula use korle synchronous way te kaj hobe. Main thread ta blocked thakbe. Orhtat default asynchronous feature ta used hocche na. but eta always recommended je amra asynchronous way te kaj korbo orthat syncjukto functiongula use na kore only readFile or writeFile functiongula use korbo.

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

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

  console.*log*(data.*toString*());

});

t.o.p ----

Hello World! I am a web developer.

Ekhane amra asynchronous way te kaj ta korlam. Asynchronous kaj dekha matroi sheta i/o ke diye deya hobe. Amra jani asynchronous kajer jonno callback function lage. Jeta amra function er 2nd parameter hishebe pass kore disi. Nodejs shokol module er khetrei callback function lekhar shomoy parameter duita jinish thake. 1) error 2) data

Always error ta age thake. Ei duitar moddhe jekono ekta null hoy.

Ekhon jinishta je asynchronous way te hoise sheta bojhar jonno nicher kajta korlam.

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

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

  console.*log*(data.*toString*());

});

console.*log*("I am learning.");

t.o.p ---

I am learning.

Hello World! I am a web developer.

Porerta age chole ashche. Orthat async way te kaj hoise. Event loop blocked hoye thaktese na.

Ekhon amra nodejs er event module niye kaj korbo. Ei event module ta nodejs eco system er one of the important building blocks. Node js ke bole ekta non blocking event driven run time. Shob kichui she event er upor base kore kore. Event mane kono ekta action ba ghotona ghoteche. Event docs e EventEmitter name ekta class dekhtesi. Emit ortho ber kora. Event emit kora mane event raise kora. Orthat ekta event hoise. Sheta boltesi. Amra nodejs e nijer banano event raise korte pari. For e.g. school e period shesh hole pion ghonta bajay. Ekhane period shesh howa ekta event. Ar pion ghonta bajaya ekta event raise kore. Nodejs e event hole amra jante pari. Ebong event hole ki korte hobe sheta amra thik kore dite pari. Ekhon example shohokare dekhbo.

Ind.js e const EventEmitter = require(“events”)

Events module er khetre require function ta ekta class dey. Ekaronei EventEmitter shurute capital letter diye lekhsi. Cz amra jani jekono programming language e class er convention hocche capital letter diye variable er name ta lekha. Class kono real life object na. eta holo object er blueprint ba structure.

Ami require(“events”) kore module ta ke niye ashlam. Er fole ekta class pelam. Sheta EventEmitter variable e nilam.

Ebar shei class diye ekta real life emitter object create korbo.

Amader purpose events module diye nijossho ekta event raise kora. Ei ‘emitter’ object diye amra ekhon event raise korte parbo. Mane ekta ghotona ghoteche sheta janan dite parbo. Er jonno emitter.emit() method use korte hobe.

*const* EventEmitter = *require*("events");

*const* emitter = new *EventEmitter*();

emitter.*emit*("bellRing");

ekhane ‘bellRing’ holo event er name. ‘bellRing’ event ta raised holo. but terminal e kono output nai. Cz ami ekta event raise kore disi. Nodejs sheta jane. Kintu event hole ki hobe sheta to bole dei nai.

Event hole ki korte hobe sheta ekhon bole dibo. Orthat event er ekta listener register kore dibo. Listener ‘bellRing’ namok ekta event er jonno wait kore thakbe. Jokhoni oi event ta fire hobe listener tar function ta call kore dibe.

emitter.emit() er motoi arekta method emitter.on() use korbo. Amra browser e vanilla js e erokom addEventListener use korechi. Like mouse click , mouse hover etc korle ki korte hobe sheta addEventListener e function er maddhome bole ditam. Kintu nodejs kaj kortese server e . ekhane browser nai, mouseclick event nai. Ekhetre event bolte bivinno action hobe like database theke data chole ashche ba operating system er memory ekta certain level e neme gese. Tokhon amra shei onujayi action nibo. Orthat amra node js e nijer moto event fire kore rakhte pari. Shei event hoye gele ki korte hobe shetao bole dite pari.

*const* EventEmitter = *require*("events");

*const* emitter = new *EventEmitter*();

*// register a listener for the 'bellRIng' event*

emitter.*on*("bellRing", () *=>* {

  console.*log*("The bell has rung. Let's go !!!");

});

*// raise a event*

emitter.*emit*("bellRing");

t.o.p ----

The bell has rung. Let's go !!!

Ekhon ami chaitesi event ta 2 sec por ghotabo. Tahole just ekta setTimeOut function use korbo.

*const* EventEmitter = *require*("events");

*const* emitter = new *EventEmitter*();

*// register a listener for the 'bellRIng' event*

emitter.*on*("bellRing", () *=>* {

  console.*log*("The bell has rung. Let's go !!!");

});

*// raise a event*

*setTimeout*(() *=>* {

  emitter.*emit*("bellRing");

}, 2000);

Ekhon node ind korar 2 sec por o/p ashlo.

Abar amader event listener emit.on jodi event raiser er por likhi taholeo kaj hobe na.

Amra chaile callback function ke kichu parameter o dite pari.

*const* EventEmitter = *require*("events");

*const* emitter = new *EventEmitter*();

*// register a listener for the 'bellRIng' event*

emitter.*on*("bellRing", (*period*) *=>* {

  console.*log*(`The bell has rung. ${period}!!!`);

});

*// raise a event*

*setTimeout*(() *=>* {

  emitter.*emit*("bellRing", "second period ended");

}, 1000);

t.o.p ----

The bell has rung. second period ended!!!

Multiple parameter pass korte hole object akare pass korte hobe.

*const* EventEmitter = *require*("events");

*const* emitter = new *EventEmitter*();

*// register a listener for the 'bellRIng' event*

emitter.*on*("bellRing", (*activity*) *=>* {

  console.*log*(

    `The bell has rung. ${activity.firstSentence}. ${activity.secondSentence}!!!`

  );

});

*// raise a event*

*setTimeout*(() *=>* {

  emitter.*emit*("bellRing", {

    firstSentence: "Last period has ended.",

    secondSentence: "Let's play football",

  });

}, 1000);

t.o.p ---

The bell has rung. Last period has ended.. Let's play football!!!

Ekhon amra arekta separate js file ba module create kore shekhane amader amader event raiser ta ke niye jete chacchi. Notun js file er name dilam school.js.

School.js …

*const* EventEmitter = *require*("events");

*const* emitter = new *EventEmitter*();

*function* *startPeriod*() {

  console.*log*("Class has started.");

*// raise a event when the bell rings*

*setTimeout*(() *=>* {

    emitter.*emit*("bellRing", {

      firstSentence: "Last period has ended.",

      secondSentence: "Let's play football",

    });

  }, 1000);

}

module.exports = startPeriod;

ind.js …

*const* EventEmitter = *require*("events");

*const* emitter = new *EventEmitter*();

*const* startPeriod = *require*("./school");

*// register a listener for the 'bellRIng' event*

emitter.*on*("bellRing", (*activity*) *=>* {

  console.*log*(

    `The bell has rung. ${activity.firstSentence}. ${activity.secondSentence}!!!`

  );

});

*startPeriod*();

node ind t.o.p ---

Class has started.

Ekhane school.js theke event raise hoise. But ind.js event listener er function ta call hoy nai.

School.js EventEmitter ekta class. Shei class diye amra ekta object emitter create korsi. Shei emitter ke diye ekta event raise korsi startPeriod function body er moddhe.

Onnodike ind.js e emitter.on namok event listener register korsi sheta arekta separate emitter object er shahajje. Ei emitter object er shathe school.js er emitter object er kono relation nai. Ei problem solve korte code take ektu modify korbo.

School.js …

*const* EventEmitter = *require*("events");

*class* School extends EventEmitter {

*startPeriod*() {

    console.*log*("Class has started.");

*// raise a event when the bell rings*

*setTimeout*(() *=>* {

      this.*emit*("bellRing", {

        firstSentence: "Last period has ended.",

        secondSentence: "Let's play football",

      });

    }, 1000);

  }

}

module.exports = School;

ind.js …

*const* School = *require*("./school");

*const* school = new *School*();

*// register a listener for the 'bellRIng' event*

school.*on*("bellRing", (*activity*) *=>* {

  console.*log*(

    `The bell has rung. ${activity.firstSentence}. ${activity.secondSentence}!!!`

  );

});

school.*startPeriod*();

node ind t.o.p ---

Class has started.

The bell has rung. Last period has ended.. Let's play football!!!

School.js e School name ekta class niye sheta ke EventEmitter class e extend korlam. Er fole EventEmitter class e shob property and methods School class ta te chole ashlo. School class tar moddhe startPeriod function ta niye giye sheta te dorkari modification korlam. Erpor School class ta kei export kore dilam.

Ei kaj ta korlam jate ami porobortite ekta single object diyei event raise korte pari ebong listener set korte pari.

Erpor ind.js e giye School class ta ke require kore anlam. Shei class diye ekta object school create korlam. Ei school object tai ekhon ager example gular emitter object er kaj korbe. school.startPeriod() kore event ta raise kore dilam. er age school.on(…) kore listener set kore dilam.

Nodejs er moduler system e ekta event driven kaj kore fellam.

Ekhon amra http module niye kaj korbo. http module ta amader shob application ei thakbe. Ei module ta network er shathe communicate korte pare. Nodejs er khetre amar application er moddhe server create korte pari http module er shahajje. Localhost e kaj korar shomoy ami jokhon create server korbo tokhon ami notun ekta server create kori. Tokhon oita accessible hobe.

First e http module ta require kore niye ashbo. http.createServer() kore server create korchi. Sheta ke ‘server’ namok ekta variable e nilam. Ei ‘server’ ekta object. Eta ekta event emitter. Orthat ero on, listen ei jatio jinishgula ache.

Ekhon server.listen(3000) function ta call korlam. Ekhane 3000 holo port number. Eta convention. Er ortho server 3000 port e listen kortese. Orthat amra jokhon nodejs application create kori tokhon amra port number bole dite pari amader application ei. Ei application e amra chaile ekadhik server o create kore felte pari.

http.js ….

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

*const* server = http.*createServer*();

server.*listen*(3000);

console.*log*("Listening on Port 3000");

node http t.o.p ---

Listening on Port 3000

Dekha jabe terminal e cursor ta pending hoye ache. Er ortho server ta running ba on. Ekhon browser e giye localhost:3000 e gele loading dekhabe. Cz 3000 port e hit korle ki o/p dibe sheta ami ekhono define kore dei nai.

Jokhon createServer kore server.listen dei tokhon event loop on hoye jay ebong ei task ta repeatedly hotei thake. Event loop ke kokhono she chere dey na. event loop ekta while loop er moto. Tar jodi kono task korar thake she sheta korte thake. Ekta certain time por por she event loop ke on rakhe. Jeno server ta running thake.

Jehetu ‘server’ ta ekta event emitter shehetu ami ekhon etar on listener method ta use korte parbo.

Jokhoni kono user amar server e hit korbe tokhoni ‘connection’ namok event ta fire hobe. Tokhon on listener er callback function ta run hobe.

http.js …

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

*const* server = http.*createServer*();

server.*on*("connection", () *=>* {

  console.*log*("New Conncetion Found...");

});

server.*listen*(3000);

console.*log*("Listening on Port 3000");

node http kore browser localhost:3000 e gele t.o.p ---

Listening on Port 3000

New Conncetion Found...

Jokhoni notun keu amar server e request kore tokhoni ‘connection’ event ta emit hoy. Real life e kokhono amra ‘connection’ event listen korbo na.

Server.on use na kore amra onno vabeo listener ta implement korte pari. Tar jonno createSever() function er parameter er moddhe callback function ta diye dibo. Prottekta user jokhon server e request korbe tokhon ei callback function tai run hobe. ‘connecction’ event er belay ja hoto same bepartai ekhane ghotbe. Ekhane callback function ta duita jinish nibe parameter hishebe. 1) request 2) response. Server e jokhon kono kichu request korbo tokhon sheta holo ‘request’. Ar server jokhon kichu pathabe tokhon sheta holo ‘response’.

‘res’ objecter moddhe ami jodi write function ta call kore dei tahole res er moddhe jinishta dhukabe. Finally res.end() kore response ta end korte hobe. Ami chaile res.write() multiple time o likhte pari. ‘req’, ‘res’ ei duita object er onek property and method ache jegula pore dekhbo.

http.js …

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

*const* server = http.*createServer*((*req*, *res*) *=>* {

  res.*write*("Hello guys!");

  res.*write*("How are you?");

  res.*end*();

});

server.*listen*(3000);

console.*log*("Listening on Port 3000");

node http kore localhost:300 e gelam. o/p holo

Hello guys!How are you?

Amra browser e root ba localhost:3000/ ei address e hit kortesi. Amra chaile localhost:3000/about ei address eo hit korte pari. ‘req’ object er url property er upor depend kore amra bivinno address er jonno different output dite pari.

http.js …

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

*const* server = http.*createServer*((*req*, *res*) *=>* {

  if (req.url *===* "/") {

    res.*write*("Hello guys!");

    res.*write*(" How are you?");

    res.*end*();

  } else if (req.url *===* "/about") {

    res.*write*("I am a full stack web developer");

    res.*end*();

  } else {

    res.*write*("Not Found");

    res.*end*();

  }

});

server.*listen*(3000);

console.*log*("Listening on Port 3000");

node http kore browser localhost:3000/about korle ashtese “I am a full stack web developer”. Onnanno outputgulao thik moto ashtese.

Ajke dekhlam http module ta diye kivabe server create korte hoy, kivabe server ta ekta certain port e listen koracchi orthat server ta always running rakhchi. Server ta ke running rakhe actually event loop.