**What is Node js?**

It’s a javascript runtime built on chrome’s V8 javascript engine.

Ekhane duita jinish important 1. Js engine 2. Javascript runtime

**What is Runtime?**

Node js ekta runtime. There is difference between programming language and runtime. Javascript ekta programming language.

Js keno eshechilo? Browser e kichu dynamic functionality anar jonno. Html website er structure dey. Css ektake stylize kore. Ar js ekta programming language ja website ke dynamic functionality dey.

Breandan Eich ke Js bananor jonno shurute 1) language ta ke design korte hoyechilo. 2) emon kichu ekta banate hoyechilo ja oi language ta ke run ba execute korte pare.

Orthat take emon kichu korte hoyechilo jate browser js code nite pare ebong sheta ke run korte pare.

Language design korte giye take kichu syntax define korte hoyechilo e.g. function myFunc(){

console.log(“Hello world”)

}

Eta ekta js function er syntax.

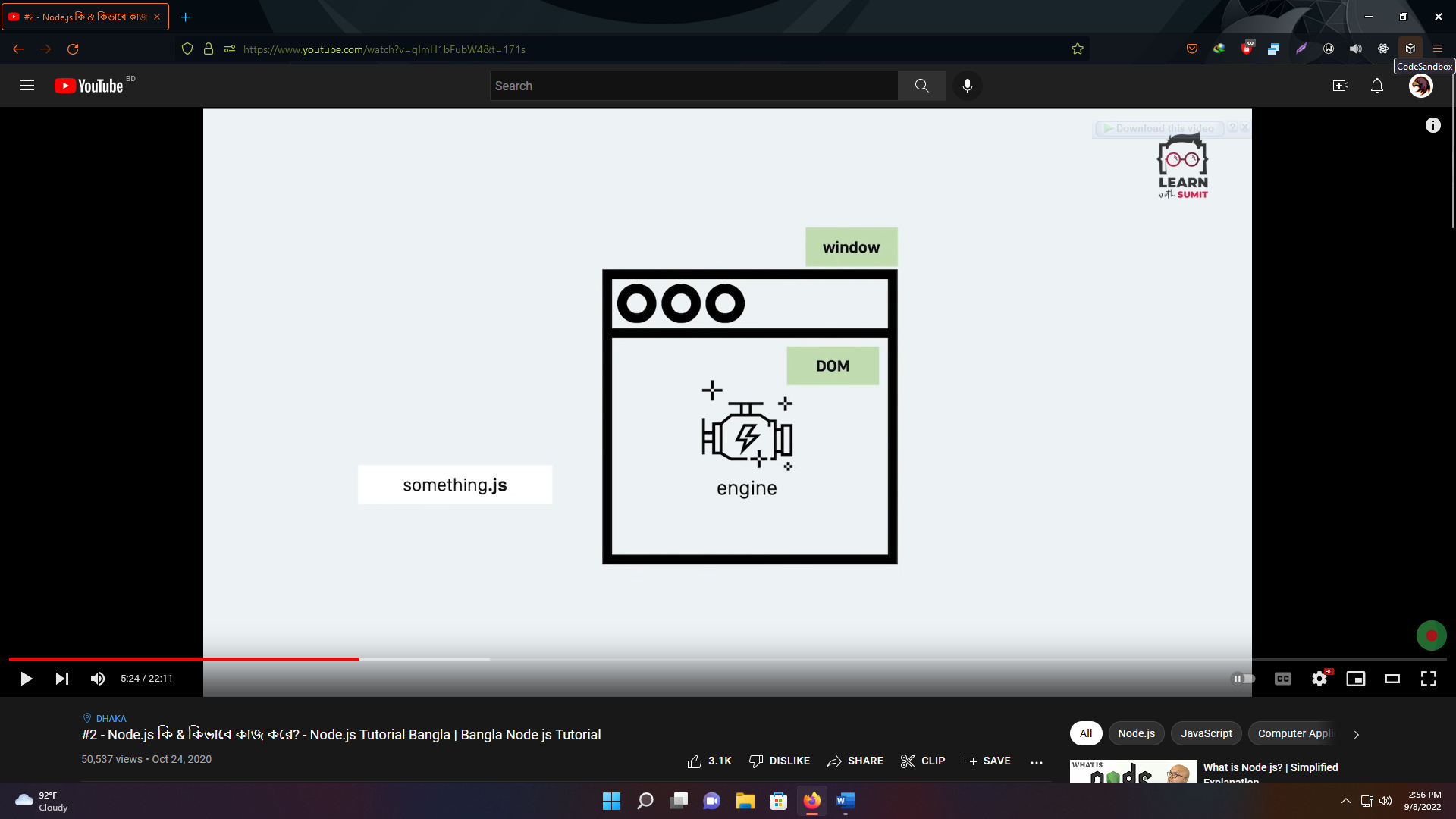
Ekhon kivabe ei progamming language ke run korbo?

Er jonno creator ra javascript engine create korlen. Ei engine ekta js file or js code ney ebong execute kore. Orthat js file je je instruction gula programmer likhe dise shegula she programmatically run kore. Js creator ra ei engine ta ke browser er jonno banaisilo jate browser jodi kono js file pay tahole ei engine er shahajje code ta ke execute korte pare.

In a nutshell, language or syntax holo js file e ja ja ache shegula. Orthat oi file er contents. Ebong runtime holo jeta oi contentgulake niye execute kore dey.

Ei runtime eka eka kaj kortese na. take browser er context e kaj korte hoitese. Je js programme ta likhse tini browser e load howa web page take target kore code likhse. So programmer er html, body, div, img etc DOM element er access proyojon. Browser er khetre ei context tai holo DOM. Ei DOM ke amra ekta huge js object hishebe imagine korte pari jekhane pura web page er shob element ache. Jokhon amar kache DOM er access thakbe ami shetake and consequently pura web page ta kei manipulate korte parbo.

So lets suppose, browser er context e amar ekta js engine ache jeta browser er vitor boshe ache. Ebong etar DOM er access ache. Eta ke ami ekta js file pass korle amar js code engine er maddhome DOM context (window object) er access peye jacche.



Ekhon bivinno browser ei engine ta ke bivinno vabe implement korse.

Firefox ---> spidermonkey

Google chrome ---> v8

IE ---> chakra

Ei enginegular moddhe shobe shobeche valo engine chrome er v8 engine. V8 engine chrome browser er vitore eki vabe DOM ebong window object context e kaj kore.

Ei jayga theke Node js creator Ryan Dahl vaba shuru krolo. Tini vablen v8 engine ke ber kore ene onno ekta context e use korle khali DOM context e ar pore thakte hocche na. ami onno kono jaygay sheta ke niye amar moto context create kore she onujayi implement korte parbo.

V8 engine c++ diye banano open source platform. Ryan v8 er codebase take ber kore ene nijer c++ code sheta te dhukiye fellen ebong nijer codebase jukto kore ekta open source project banay fellen. Ebong etar naam dilen node js. Shei codebase e tini operating system er permission nilen jaate file system, network, memory … erokom system resources er access nite paren ja browser e kora jeto na. ekehtre context ta holo pura operating system jar shobkichur access node js er kache ache. Orthat ekta server e jokhon node js run kore tokhon oi server er motamuti shob kichuri access she pay.

So node js is not a programming language. It’s a c++ programme which we can install in any machine and run. Since it works in a operating system so it can be run in a server too. Cz server is nothing but a remote computer which has a operating system.

Node js is a javascript runtime powered by chrome’s v8 engine which can run in a web server.

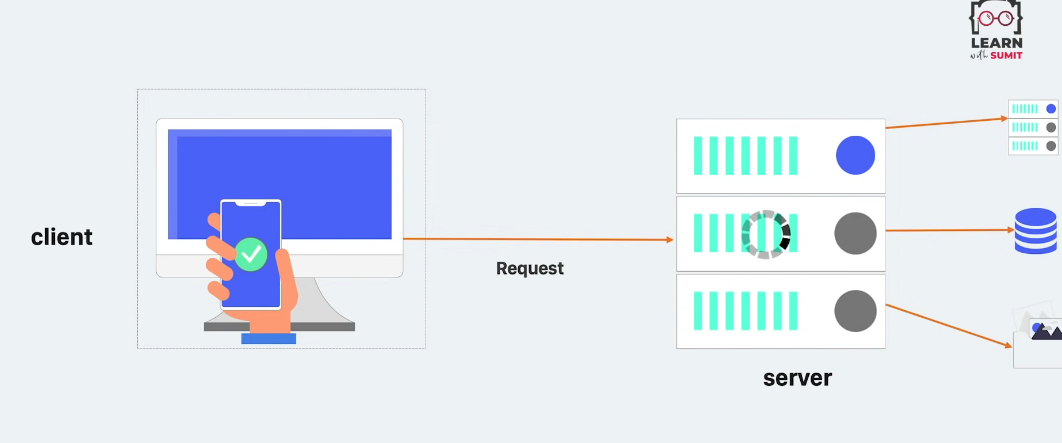
**How does a web server work ?**

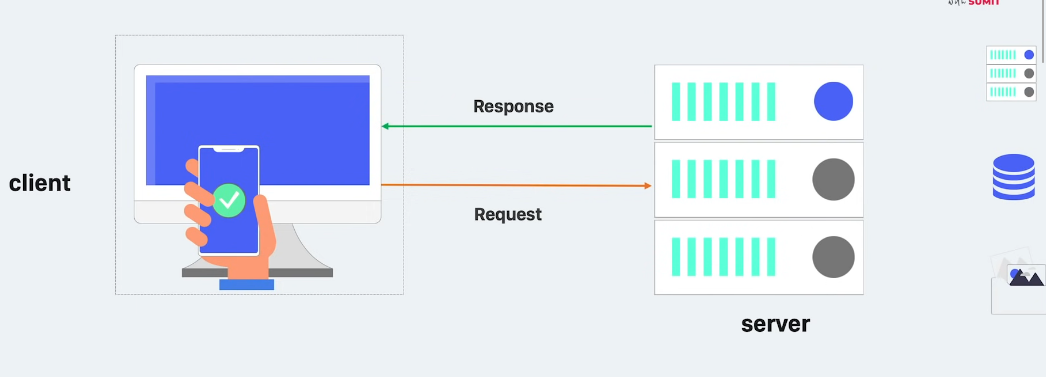
Web server e amader moto programmerder banano web application thake jeta dure thaka kono client access korte pare. Shei client hote pare kono manusher browser ba kono mobile app etc.

Jokhon client web browser tokhon she server theke html expect korbe. Jodio single page application er khetre html ta client er end e rekhe shudhu api er maddhome data request o kora jay. But sheta special case. Mainly ekta browser server theke html expect kore. Othoba SPA er khetre data expect kore.

Kintu amader client mobile app hole tahole pura html server theke expect kori na. amra just data expect kori jeta server theke api request kore niye ashte pari.

So in nutshell, client jai hok amra server theke html or data expect kori. Sheta server ke amra proyojon moto request kori. ebong shei server abar arekta server ba database ba file system ke proyojon moto request kore finally ekta certain time pore response dey.

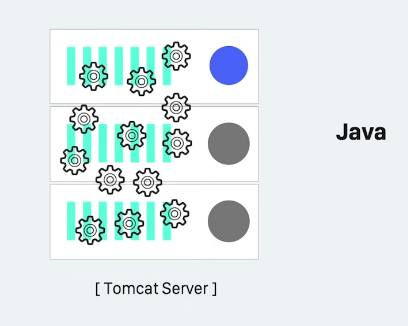




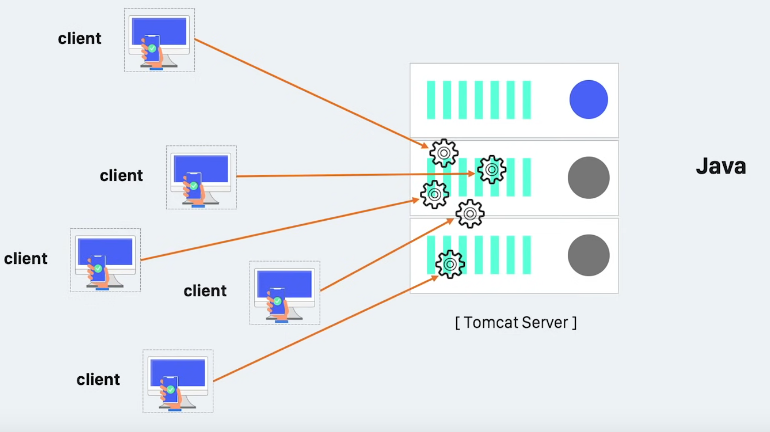
Ekhon amar request ta jodi simple hoy like 2+2 koto hoy sheta jante chailam tahole server quickly response diye dibe 4. Eta ke bole CPU intensive task. Mane eta server er cpu i korte pare. Etar jonno kothao jemon file system, database ba onno sever e request korte hoy na. kintu request ta jodi emon hoy jetar jonno server ke onno server e request korte hobe ba database theke data ante hobe ba file system theke data read korte hobe tahole shetake bole i/o intensive task. Suppose ekhetre 3 sec por response ashlo. Ei beparta ektu tricky. Cz ekhane ekta client ke response dite 3 sec laglo. Ekhon jodi 5 jon clients request kore ek shathe? Tahole ki 5th person ke 15 sec wait korte hobe. Ekhetre kivabe request handled hobe?

Server to onek rokom hote pare. For e.g. java te tomcat server use hoy. Ei tomcat server multiple request handle korte pare.

Now notun ekta concept jante hobe which is threads. Thread bolte amra process dhore nite pari. Ekta tomcat server e multiple thread thake.



Jokhoni ekta client request kore tokhoni tomcat server shei request ke handle korar jonno ekta thread assign kore dey. Jodi second request ashe tahole second arekta thread assign hoy. Orthat jotogula request ashbe totogula thread allocated hobe jegula seperately kaj korbe.



So shobgula request parallelly process hobe ebong ekjon client er data pete 3 sec lagle 5 jon client ero 3 sec lagbe. Tomcat server e 200 ta thread thake. So 201st client ke wait korte hobe beshi shomoy.

Ekhon node js e ferot ashi. Amra ekhane javascript run kortesi. Js is a single threaded language. Ryan Dahl node js create korar shomoy multiple thread use korar option rakhte chan nai. Now how would it handle multiple requests for data?

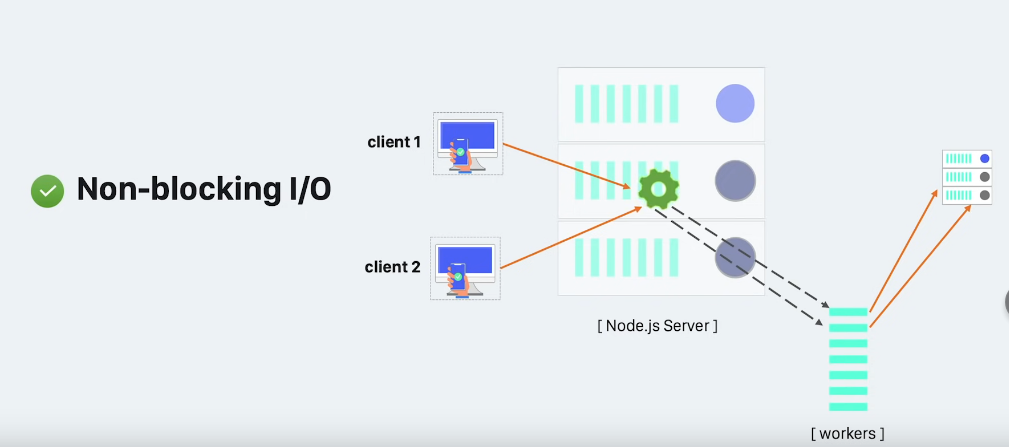
Node js er khetreo simple cpu intensive task korte problem hobe na. kintu i/o intensive task korar shomoy ekadhik request following way te handle kore.

Node js says, ami ekta single threaded application. Ami only request accept korbo. Amar onno kaj korar dorkar nai. Baki kaj korbe amar assistant ra. So node js er single thread request accept korbe thiki but onno server theke response er jonno wait korbe na.

Node js er duita concept ache….

1. Asynchronous
2. Non blocking i/o

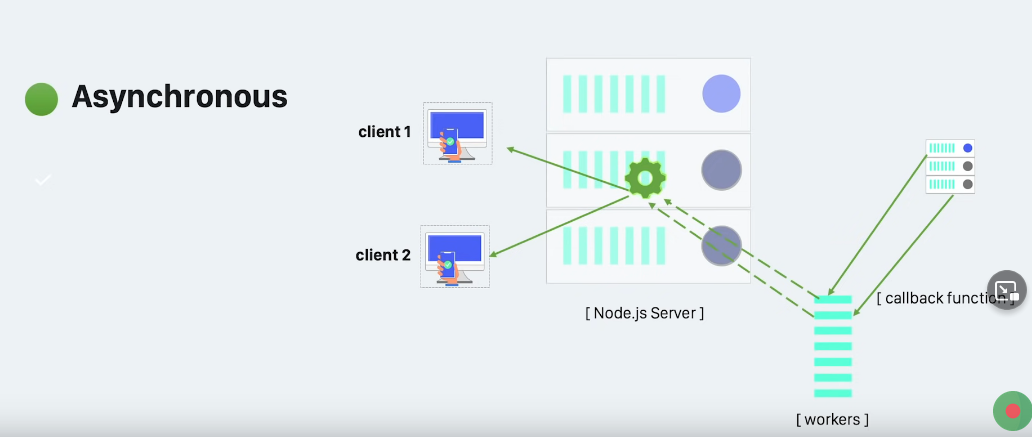
Non blocking i/o mane holo single thread request accept kore arekjonke request pathay dibe. Current thread oi request niye busy hoye boshe thakbe na. suppose amader kichu assistant worker ache jara ei server jonno kaj kortese. Node js would just say, “Hey worker ! I’ve just accepted a request. Take it and make request to fetch data from another server”. Worker will say, “Ok boss! I’m doing it right now.” In the mean time another request comes up. Node js er single thread ekhono open. So second client er request o she accept korbe. Ebong second worker ke bolbe “go fetch data for me from server.”



Node js evabe non blocking i/o concept use kore. Ekhane she simply ekta request er jonno wait kore thakbe na. mane blocked thakbe na.

Ekhon kono client eshe jodi bole amar ekta 1 theke 10000000 porjonoto number er sum lagbe. It is a huge cpu intensive task. Eta onno kauke dite parbe na. thread thekei etar response dite hobe. Ei complex kajta korte giye thread busy ba blocked hoye jabe. Ekarone bola hoy node js is not suitable for large CPU intensive task. Eta i/o intensive kajer jonno valo. Orthat jokhon ami external system (database, file system, network etc) er shathe interact korte chaibo.

Lets go back to previous example. Worker 1 arekta server ke je Prothom request korechilo shetar response ferot ashar por ki hobe? Client er kache response to pathano hoy nai. Etar jonno arekta concept ache. Asynchronous or callback. Jokhoni worker response peye jabe tokhoni amra callback function execute korbo. Callback mane holo kaj hoye jabar por ekta notification or event paoar moto bepar. Er jonno node js er architecture ke event based architecture o bola hoye thake. Programmer hishebe amra worker er kaj hoye gele jante pari ebong amader pochondo moto callback function ke call kore diye boli erpor ki korte hobe. Er jonno etake bola hoy asynchronous. orthar por por na ba programatically sequence maintained hocche na. but vitore onnovabe sequence maintain hocche.



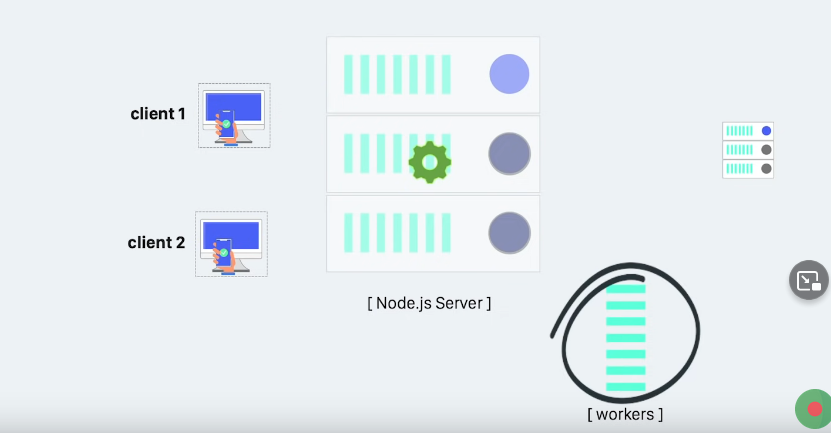
Node js niye bolte gele arekta concept chole ashe sheta holo event loop.



Etokkhon amra dekhlam synchronous and asynchronous both way tei single thread kaj kore jaitese. Kintu shob kichu ke control kortese ke? Single thread er ei event loop i mainly shob kichu ke control kortese i.e. kokhon kake call korte hobe, kar kache ki pathate hobe etc. etake simply ekta infinite while loop hishebe imagine kora jete pare. Continuously new task er jonno wait kortese ebong ektar por ekta task e shothik jaygay pathay ditese. Shetar jonno she ekta task queue maintain kortese ebong task order onujayi thread pool e giye worker thread er maddhome complete hocche. Ei event loop node js ke multiple request handling er jonno perfect banay dise.

Video te dekha jacce, jodi high CPU intensive task (red marked) ashe sheta thread ke block kore rekhe dey ebong shekhetre other requests ke wait korte hote pare. So high CPU intensive task node js e deya recommendable na. dileo choto choto micro task e vag kore deya uchit. Tobe ei major weakness er jonno node js 10.5 version the multi threading introduce kora hoise.

Finally arekta jinish dekhbo. Question ashte pare node js jodi multiple threads support na kore tahole kivabe ei worker gula toiri hocche? Kothay hocche? Ke korche? Tara jodi thread na hoy tara ashole ki?



Well, eta ke implement korar jonno node js arekta concept use kore. Sheta holo libuv. Libuv ekta special type of library jeta node js er jonno specially banano holeo onnano jaygayo use kora hoy. Eta amader non blocking i/o concept ta dey. Eta C language e banano jeta amader system er kernel ke use kore. Kernel mane je programme ta amader pura operating system ke control kore. Ei kernel er multiple thread ache. Node js e amra programmer ra multiple thread use kortesi na. but behind the scene libuv er maddhome node js system kernel ke diye multiple threads implement kortese. So je worker gula niye kotha bollam shegula ashole threads.