is digit() method:
checks whether the string

⇒ command prompt

Node JS

→ node -v

→ npm -v

→ npm help ↵

→ npm install -h

→ npm help -search update

⇒ npm init

⇒ npm init --yes

⇒ npm config set init-author-name "Mounika"

⇒ npm set init-license "NIT"

⇒ npm init --yes

default values setup

→ npm config get init-author-name

→ npm get init-license

⇒ delete the default values

⇒ npm config delete init-license

→ npm config delete init-author-name

→ npm init --yes ↵

⇒ npm install moment

- delete -

⇒ npm install moment --save.

→ npm install --save-dev

→ npm uninstall moment --save

→ npm uninstall lodash --save-dev

⇒ install Globally :-

⇒ npm install moment -g

→ npm uninstall moment -g

→ npm → package name.

list of package's:-

npm list → show like's the structure.

→ npm list --depth 1

→ npm list --dept 0 → list of all the package's & No dependencies

→ Global pakage's

npm list --global true --depth 0

Node JS:-

Node JS is an open-source, class-platform runtime enviroment for developi.

Smeltic version in the:-

npm install lodash --save.

list of package's:- *locally*

npm list → It will display the tree like structu

→ litserves depend's on package'

Eg lodash doesn't depend on anything else.

we can also restric the

list --dept 1

↳ It giving the imdeiable dependencies of the lite server

list --dept 0

↳ list of all the package's

for the global package's

→ list --global true --depth 0

→ list of global installed globally

* npm versioning:-

npm install loadash --save

loadash enter by latest version

4 → Major ver

16 → minor vers

1 → patch version

→ ~~first create folder~~ symantic version

** npm install lodash@3.3.0 --save

↳ the package.json file updated to 3.3.0

→*** npm install lodash@4.12. --save.

↳ It shows latest patch version.

→ * npm install lodash@4 --save

↳ It show latest major & patch version.

→* npm install ⇒ is the mostly common command

~ restaict upgrading major & minior version's → only path version update in latest version.

⇒* npm install

⇒ update the package latest version

* ⇒ npm update lodash --save ↑ update to latest version

⇒* update npm update --dev --save

↳ latest version of devdependicy

⇒* npm update ⇒ all the dependencie & the dev dependen

→ we update the package globally all

** npm install np.n@latest -g ↵

⇒ npm list --dept 0

⇒ npm prune ⇒ command to Remove additionally don't require a project

⇒ npm list --dept 0

Shortcuts:

① npm init -y ⇒ it will use package.json file with default values

② npm install ⇒ install the package locally

npm install lodash

③ npm lodash -S ⇒ into lodash dependencies saved into package.json file package save in to

④ npm install (package name) -D ⇒ Dev

npm shortcuts → docs.npmjs.com/misc/config

npm Scripts:-

To write basic programm

app.js

console.log("basic")

Terminal → node app.js

⇒ goto package.json then change to "test" ↳ Replace with "start"

"start": "node app.js"

⇒ command prompt ⇒ npm start →

↳ o/p

Node JS:-

JS uses an event - driven, non-blocking model that makes it lightweight & efficient, Node.js package's ecosystem, npm is the largest ecosystem of open source libraries in the would

⇒ A platform which allows us to run JS on a server
⇒ Read, delet & update files.
⇒ Easily communicate with a db

Why Node.JS so popular?-

→ It uses JS.
→ Very fast
→ Huge - eco system.
→ Huge ecosystem of open source packages (npm)
→ Great for real time services (like chats)

What we will learn

⇒ The inner workings of Node.JS
- V8 engine
- modules
- event Emitter
- The file system

⇒ Creating web servers
- Routing
- Expoess
- Templating

→ make a Node.JS application

⇒ node - v

⇒ basic program

⇒ node app

⇒ chang dy ⇒ cd..

⇒ cd [illegible]

⇒ [illegible] > node app

⇒ The V8 Engine:-

JS Engines:- A JS engine takes JS, & converts it into something it does understand - machine code.

⇒ The node is written in c++

⇒ at the heart of Node.JS is the V8 engine.

⇒ The V8 engine converts our JS into machine code.

Node.js with V8

JS → Node.JS (V8) → machine code.

⇒ The Global object:-

the objects are available in all modules. Some of these objects aren't actually in the global space but in the module scope

⇒ app.js

↳ console.log("test")

clients & servers

every server identified by own ip add.

client —Request→ server
client ←Response— server

Protocols:- A set of communication rules, that two sides agree to use when communicating

client —socket— server
IP IP

TCP:-

← socket —— • • ∘ ∘ ∘
packets

Ports:-

A prgm running on a computer can listen for req's sent to a particular port No.

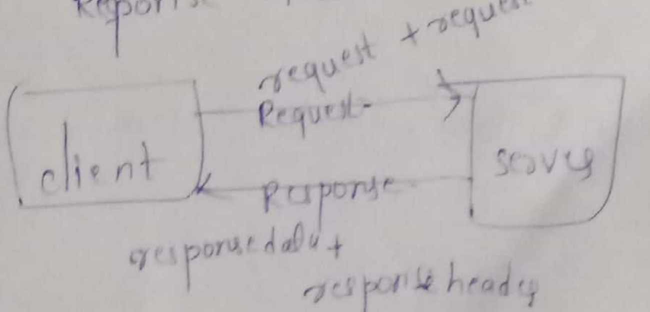
=> Eg: 172.75.76:3000

=> **Creating server**

```
var http = require('http');
var server = http.createServer(function(req,res){
  res.writeHead(200, {'content-Type': text/plain'});
  res.end("Hey data");
});
```

→ server.listen(3000, '127.0.0.1')
console.log('...')

Response Headers



* Response Headers
→ content - Type
- status

Readable streams:-

streams:-

writable streams — allow node js to write data to a stream

Readable " — " " " " read " "

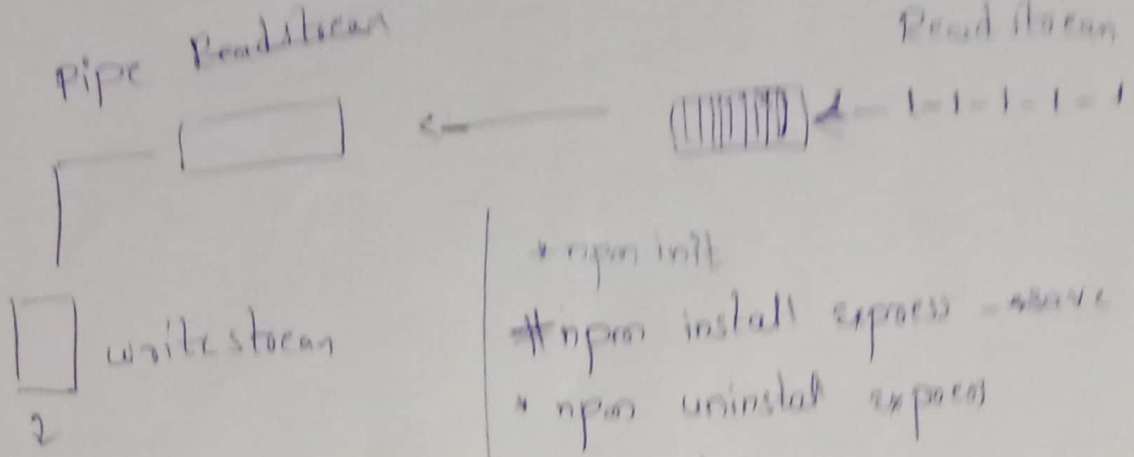
Duplex — can read & write to a stream

```
var http = require('http')
var fs = require('fs');
var myReadStream = fs.createReadStream(__dirname + '/readme.txt', 'utf8');
myReadStream.on('data', function(chunk){
    console.log('new chunk received:');
    console.log(chunk);
});
```

writable streams:-

```
var http = require('http');
var fs = require('fs');
var myReadStream = fs.createReadStream(__dirname +
                   '/readme.txt', 'utf8');
var myWriteStream = fs.createWriteStream(__dirname +
                   '/writeme.txt');
myReadStream.on('data', function(chunk){
    console.log('new chunk');
    myWriteStream.write(chunk);
});
```

Pipe's:



* Serversing HTML pages:

* node package manager:

node install express
npm uninstall express

* npm init
* npm install express --save
* npm uninstal express
* npm instal
-> npm install nodemon -g
-> nodemon filename.js