

### Module

Module module has exports property e.g module.exports = {}  
 e.ex ports = fn() =>{} module.  
 ex ports = {hello, move}

Require returns module.exports

config Storing configuration settings:  
**default, dev & prod**  
 `confi g.g et( con fiP rop er ty)

Node wraps you code with function expression & invokes it, this function expression has these argument: exports, module, \_\_dirname & \_\_filename. Now you have access to these argument within your code. You can use these argument e.g module.ex ports = hello() to expose ur module

### Express

install npm i express  
 app = require('express')

web server app.get('/', (req, res) => {  
 res.send('hello')  
 })  
 app.li ste n(3000)

route params req.params

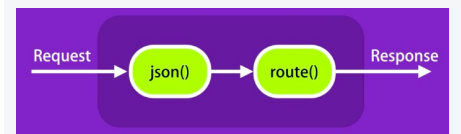
query params req.query

### Express (cont)

app.use(body-parser.json) parse body of req into json:-  
 app.us e(e xpr ess.js on())  
 serves static files:-  
 app.us e(expre ss.stati c(path .join(\_\_dirname, 'public')))

route handle is a middleware.  
 request-processing-pipeline: Req -

### Request processing pipeline



express.j son() will pass request to route handler and route will terminate request by sending response.

### Built - in modules (mostly used)

path to handle paths  
 path = require('path')  
 path.j oin (\_\_dir name, 'public')

os provide operating system info  
 os = require('os')  
 os.fre emem()  
 os.upt ime()  
 os.use rInfo()

http make Node to act as webserver  
 server = require('http')  
 server.cr eat eSe rve r(req, res)=>{}  
 server.li ste n(3000)

process Node process info - global object  
 port = proces s.e nv.PORT || 3000

### Built - in modules (mostly used) (cont)

events to handle events  
 EventE mitter = requir e('EventE mitter = new EventE mit te emitter.on(eventName, liste emitte r.e mit (ev ent Nam

file to handle file system  
 system fs = requir e('fs')  
 fs.rea dfi le( path, callk fs.cop yFi le( sou rce.tx lback)

set environment vars: export PORT = 3000

### ewwee

require rrrrrrrrrrrrrrrrrrrrrrrrr

### www

w w

### Debugging

dbDebugger = require('debug')  
 startD ebu gger= requir e(' deb u :st art')

Two names are will create:- app:db & app:start. You can replace consol e.l og with dbDeb ugger or startD eb u gger.

Define console output:-  
 export DEBUG= app :start or  
 export DEBUG= app:db or  
 export DEBUG= app :st art ,app:db or  
 export DEBUG= app:\*

### NPM

<code>npm init</code>	Create package.json file alt: <code>npm init --yes</code>
<code>npm i</code>	Install npm package alt: <code>npm i --save-dev</code> <code>npm i --save</code>
<code>npm list</code>	lookup dependencies & version alt: <code>npm list --depth=0</code>
<code>npm outdated</code>	check 4 outdated npm package
<code>npm update</code>	update npm package - only updates minor & patches
<code>npm un</code>	uninstall npm package
<code>npm login</code>	login in npm registry
<code>npm adduser</code>	create user on npm registry

C

By **digotetso**  
[cheatography.com/digotetso/](https://cheatography.com/digotetso/)

Not published yet.  
Last updated 11th April, 2018.  
Page 2 of 2.

Sponsored by **Readable.com**  
Measure your website readability!  
<https://readable.com>