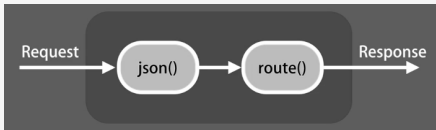


Module	Express (cont)	Built - in modules (mostly used) (cont)
Module module has exports property e.g module.exports = {} e.ex ports = fn() =>{} module. ex ports = {hello, move}	middlewares app.use(body-parser.json()) parse body of req into json:- app.us e(e xpr ess.js on()) serves static files:- app.use(express.static(path .join(__dirname, 'public')))	events to handle events EventE mitter = requir e('event-emitter') emitter = new EventE mit te emitter.on(eventName, liste emitte r.e mit (ev ent Nam
Require returns module.exports	route handle is a middleware. request-processing-pipeline: Req -	file to handle file system fs = requir e('fs') fs.rea dfi le(path, callb fs.cop yFi le(sou rce.tx lback)
config Storing configuration settings: default, dev & prod `confi g.g et(con fiP rop ert y)	Request processing pipeline  <pre> graph LR Request --> json(json()) json --> route(route()) route --> Response </pre> expres s.j son() will pass request to route handler and route will terminate request by sending response.	system set environment vars: export PORT = 3000
Node wraps you code with function expression & invokes it, this function expression has these argument: exports, module, __dirname & __filename.N ow you have access to these argument within your code. You can use these argument e.g module.ex ports = hell o() to expose ur module	Built - in modules (mostly used) path to handle paths path = requir e(' path') path.j oin (__ dir name, 'public')	www
Express	os provide operating system info os = requir e('os') os.fre emem() os.upt ime() os.use rInfo()	Debugging dbDebugger = requir e(' deb ug') startD ebu gger= requir e(' deb u :st art')
install npm i express app = requir e(' exp ress')	http make Node to act as webserver server = requir e(' http') server.cr eat eSe rve r(req, res)= >{} server.li ste n(3000)	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:*
web server app.ge t('/', (req, res) => { res.send('hello') }) app.li ste n(3000)	process Node process info - global object port = proces s.e nv.PORT 3000	
route params req.params		
query params req.query		

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/

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

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
<https://apollopad.com>