

 免費電子書

學習

Node.js

Free unaffiliated eBook created from
Stack Overflow contributors.

#node.js

.....	1
1: Node.js	2
.....	2
.....	2
Examples.....	5
Hello World HTTP.....	6
Hello World.....	7
Node.js.....	7
.....	8
.....	8
NodeJS.....	8
.....	8
.....	8
Hello World.....	9
TLS.....	10
.....	10
.....	10
TLS	10
TLS	11
REPLHello World.....	12
.....	13
.....	13
HTTPS Web.....	16
1.....	16
2.....	16
3.....	17
2: ArduinonodeJs	18
.....	18
Examples.....	18
Node JsserialportArduino.....	18
js.....	18

Arduino.....	19
.....	19
3: async.js.....	21
.....	21
Examples.....	21
.....	21
async.parallel().....	22
.....	22
.....	22
async.series().....	23
.....	23
async.times.....	24
async.each.....	25
async.series.....	25
4: CLI.....	26
.....	26
Examples.....	26
.....	26
5: ExpressJSRoute-Controller-Service.....	30
Examples.....	30
- - -.....	30
- - -.....	30
user.model.js.....	30
user.routes.js.....	30
user.controllers.js.....	31
user.services.js.....	31
6: HTTP.....	32
Examples.....	32
http.....	32
http.....	33
7: Koa Framework v2.....	34

Examples.....	34
Hello World.....	34
.....	34
8: Lodash.....	35
.....	35
Examples.....	35
.....	35
9: metalsmith.....	36
Examples.....	36
.....	36
10: Mongodb.....	37
.....	37
.....	37
Examples.....	37
MongoDB.....	37
MongoClientConnect().....	38
.....	38
insertOne().....	38
.....	39
find().....	39
.....	39
updateOne().....	39
.....	40
deleteOne().....	40
.....	40
deleteMany().....	41
.....	41
promises.....	41
11: MSSQL.....	42
.....	42
.....	42
Examples.....	42

SQL mssql npm.....	42
12: Mysql.....	44
Examples.....	44
.....	44
13: MySQL.....	45
.....	45
Examples.....	45
.....	45
.....	45
.....	45
.....	45
.....	45
MySQL.....	46
.....	47
.....	47
.....	47
.....	47
14: Node.js / Express.jsMongoDB.....	49
.....	49
.....	49
Examples.....	49
MongoDB.....	49
Mongoose.....	49
Mongo.....	50
15: Node.js STDINSTDOUT.....	51
.....	51
Examples.....	51
.....	51
16: Node.js v6.....	52
.....	52
Examples.....	52
.....	52
.....	52

.....	52
.....	52
“this”.....	52
17: node.jsWindows.....	55
.....	55
Examples.....	55
activedirectory.....	55
.....	55
.....	55
18: Node.JS.....	56
Examples.....	56
NodeJS.....	56
IntelliJ / Webstorm.....	56
Linux.....	57
19: Node.JSMongoDB.....	58
.....	58
Examples.....	58
.....	58
.....	58
.....	59
.....	59
.....	60
.....	60
.....	60
UpdateOne.....	60
UpdateMany.....	61
ReplaceOne.....	61
.....	61
20: Node.js.....	63
Examples.....	63
.....	63
.....	63

IO	63
.....	64
maxSockets.....	64
.....	64
.....	64
Socket Pooling	64
.....	65
gzip.....	65
21: Node.js	66
Examples.....	66
Node.js -	66
Node.js -	66
22: Node.js	68
Examples.....	68
HTTP.....	68
.....	68
23: Node.jsCORS	70
Examples.....	70
express.jsCORS.....	70
24: Node.JSES6	71
.....	71
Examples.....	71
Node ES6Babel.....	71
NodeJSJS es6.....	72
.....	72
25: Node.jsOracle	75
Examples.....	75
Oracle DB.....	75
.....	75
.....	76

26: Node.js	78
Examples	78
Node.js	78
27: Node.js	79
.....	79
Examples	79
.....	79
.....	79
.....	79
28: nodejs	81
Examples	81
.....	81
29: NodeJS	82
Examples	82
.....	82
30: NodeJS	83
Examples	83
Web	83
.....	83
.....	83
.....	83
Commander.js	83
Vorp.js	84
31: Nodejs	85
.....	85
Examples	85
.....	85
2009	85
2010	85
2011	85
2012	85

2013	85
2014	86
2015	86
Q1	86
Q2	86
Q3	86
Q4	86
2016	87
Q1	87
Q2	87
Q3	87
Q4	87
32: NodeJSRedis	88
.....	88
Examples	88
.....	88
.....	88
node_redis	90
33: NodeJs	91
.....	91
.....	91
Examples	91
Express Web	91
34: NPM	96
.....	96
.....	96
.....	97
Examples	98
.....	98
.....	98
NPM	98

.....	98
.....	99
NPM	100
.....	100
.....	101
.....	101
.....	102
.....	102
.....	103
.....	103
.....	103
npm	104
.....	104
.....	104
.....	105
.....	105
.....	105
.....	105
.....	105
35: nvm -	106
.....	106
Examples	106
NVM	106
NVM	106
Node	106
.....	106
Mac OSXnvm	107
.....	107
NVM	107
.....	107
shell	108

36: N-API	109
.....	109
Examples.....	109
N-API.....	109
37: OAuth 2.0	111
Examples.....	111
RedisOAuth 2 - grant_typepassword.....	111
.....	118
38: passport.js	119
.....	119
Examples.....	119
passport.jsLocalStrategy.....	119
39: PostgreSQL	120
Examples.....	120
PostgreSQL.....	120
.....	120
40: Restful API	121
Examples.....	121
.....	121
41: Sequelize.js	123
Examples.....	123
.....	123
.....	123
1. sequelize.definemodelNameattributes[options]	124
2. sequelize.import	124
42: Socket.io	126
Examples.....	126
“”.....	126
43: TCP	127
Examples.....	127
TCP.....	127

TCP	127
44:	129
Examples	129
multer	129
.....	130
.....	130
.....	130
45:	132
.....	132
Examples	132
.....	132
Eventloop	132
HTTP	132
HTTP	132
HTTP	132
46:	134
.....	134
Examples	134
HTTP	134
.....	135
.....	135
.....	136
47: Browserfiy⁴⁹⁹	137
Examples	137
- file.js	137
.....	137
Browserfy	137
.....	138
.....	138
48: Express.JSajax	139
Examples	139

AJAX.....	139
49: ExpressWeb.....	141
.....	141
.....	141
.....	141
Examples.....	141
.....	141
.....	142
.....	143
.....	143
.....	144
.....	145
.....	145
EJS.....	145
ExpressJSJSON API.....	146
.....	146
.....	147
Django.....	147
.....	147
.....	148
.....	149
Hookreqres.....	151
POST.....	151
cookiecookie.....	152
Express.....	152
Express.....	153
.....	153
.....	153
50: IISNodeIISNode.js Web.....	155
.....	155
/.....	155
.....	155

Examples.....	155
.....	155
.....	155
ExpressHello World.....	155
.....	155
server.js - Express Application.....	155
Web.config.....	156
.....	156
IISNode.....	156
URL.....	156
IIS.....	157
Socket.ioIISNode.....	158
51: Node.jsAPI.....	159
Examples.....	159
Expressapi.....	159
POST apiExpress.....	159
52: Streams.....	161
.....	161
Examples.....	161
StreamsTextFile.....	161
.....	161
/.....	162
Streams.....	163
53:.....	165
.....	165
.....	165
Examples.....	165
.....	165
shell.....	165
.....	166
54: Node.JSWebSocket.....	168

Examples.....	168
WebSocket.....	168
WebSocket.....	168
WebSocketWebSocket.....	168
WebSocket.....	168
55:	170
Examples.....	170
.....	170
56:	171
Examples.....	171
PM2.....	171
.....	171
Forever.....	172
nohup.....	173
.....	173
57: Node.js	174
Examples.....	174
CSRF.....	174
Node.jsSSL / TLS.....	175
HTTPS.....	175
HTTPS.....	176
1.....	176
2.....	176
Secure express.js 3.....	177
58:	178
Examples.....	178
- SIGTERM.....	178
59:	179
.....	179
Examples.....	179
.....	179
hello-world.js.....	179

.....	181
.....	181
.....	182
node_modules.....	182
.....	182
60: PromiseError-FirstNode.js.....	184
.....	184
Examples.....	184
Bluebird.....	184
61:	187
Examples.....	187
require.....	187
62:	188
Examples.....	188
.....	188
63: Node.js.....	189
Examples.....	189
Mac OSXNode.js.....	189
WindowsNode.js.....	189
64:	190
.....	190
Examples.....	190
GruntJs.....	190
gruntplugins.....	191
65:	192
Examples.....	192
.....	192
.....	192
Promise.....	192
/.....	192
66:	193
Examples.....	193

.....	193
.....	193
setTimeoutpromisified.....	194
67: node.js.....	195
Examples.....	195
node.js.....	195
ES6.....	195
ES6.....	196
68: Node.js.....	197
Examples.....	197
NODE_ENV = "".....	197
.....	197
.....	197
.....	198
PM2.....	198
PM2.....	198
.....	199
Forever.....	199
/devqastaging.....	199
.....	200
69: RESTCRUD API.....	202
Examples.....	202
Express 3+CRUDREST API.....	202
70:	203
.....	203
.....	203
Examples.....	203
.....	203
.....	203
71:	205
Examples.....	205
.....	205

.....	205
.....	205
72: Node.js	206
Examples.....	206
UbuntuNode.js.....	206
apt	206
nodesourceLTS 6.x	206
WindowsNode.js.....	206
nvm.....	206
APTSourceNode.js.....	207
MacNode.js.....	208
.....	208
MacPorts	208
MacOS X Installer.....	209
.....	209
Raspberry PINode.js.....	209
.....	209
CentosRHELFedoraNode.js.....	210
nNode.js.....	210
73: -	212
Examples.....	212
/ w ExpressjQueryJade.....	212
74: node.js	214
.....	214
Examples.....	214
Node.jssystemd.....	214
75:	216
Examples.....	216
fs.....	216
fluent-ffmpeg.....	217
76: angular.jsNode.jsexpress.js	218

.....	218
Examples.....	218
.....	218
.....	218
.....	218
PugExpress.....	219
AngularJS.....	219
77: Node.jsECMAScript 2015ES6.....	221
Examples.....	221
const / let.....	221
.....	221
.....	221
.....	222
.....	222
ES6.....	222
78:	224
Examples.....	224
Node.....	224
79:	228
.....	228
.....	228
Examples.....	228
.....	228
.....	229
80: HTML.....	230
.....	230
Examples.....	230
HTML.....	230
.....	230
server.js.....	230
81: MongoDBMongoose.....	231

Examples.....	231
.....	231
.....	231
.....	232
.....	232
82: I / O.....	234
.....	234
Examples.....	234
writeFilewriteFileSync.....	234
.....	234
.....	234
.....	235
.....	235
readdirreaddirSync.....	235
.....	236
.....	236
.....	236
unlinkunlinkSync.....	236
.....	237
.....	237
.....	237
.....	238
.....	238
.....	239
.....	239
.....	239
.....	239
.....	239
.....	240
.....	240
app.js.....	240
.....	240

app.js	240
83:	242
Examples.....	242
Nunjucks.....	242
84: Node.js	244
Examples.....	244
PM2.....	244
85: - REST	246
.....	246
Examples.....	246
Web.....	246
86:	248
Examples.....	248
.....	248
process.argv.....	248
/devqastaging.....	248
“”.....	249
87:	251
Examples.....	251
Node.Js.....	251
.....	252
.....	252
.....	253
.....	253
88: /	254
.....	254
Examples.....	254
Try-Catch.....	254
PromisesAsync / Await.....	254
.....	255
.....	256

89:	257
.....	257
.....	257
Examples	257
.....	257
JavaScript	257
.....	257
.....	257
Node.js	258
.....	259
.....	259
.....	259
.....	260
.....	260
.....	260
.....	260
.....	261
90: Web	262
Examples	262
GCMWebGoogle Cloud Messaging System	262
91: package.json	264
.....	264
Examples	264
.....	264
.....	264
devDependencies	264
.....	265
.....	265
.....	265
.....	266
package.json	266

92: ReadLine	270
.....	270
Examples	270
.....	270
CLI	270
93: jscsv	272
.....	272
Examples	272
FSCSV	272
94: JS	273
.....	273
Examples	273
i18njs app	273
95:	274
.....	274
Examples	274
.....	274
CORS	275
96:	276
.....	276
Examples	276
.....	276
.....	276
.....	276
MacPorts	276
PATH	276
.....	276
.....	276
.....	276
Linux	276
Debian / Ubuntu	276

CentOS / Fedora / RHEL.....	277
.....	277
.....	277
.....	277
.....	277
Shell.....	277
.....	277
NPM.....	277
.....	277
.....	278
Yarn.....	278
97:	280
.....	280
.....	280
Examples.....	280
.....	280
.....	281
98:	282
Examples.....	282
nodemon.....	282
nodemon	282
nodemon	282
nodemon	282
Browsersync.....	282
.....	282
.....	282
Windows.....	283
.....	283
.....	283
Grunt.js.....	283
Gulp.js.....	283

API	283
99:	285
.....	285
Examples.....	285
.....	285
.....	285
console.log.....	285
console.error.....	285
console.timeconsole.timeEnd.....	285
.....	285
.....	285
.....	286
.....	286
.....	286
100:	287
.....	287
Examples.....	287
.....	287
101:	291
Examples.....	291
nodebackPromises.....	291
.....	291
.....	291
Promise.using.....	291
.....	292
102: Node.jsPOST	293
.....	293
Examples.....	293
node.jsPOST.....	293
103:	294
.....	294
.....	

.....	294
Examples.....	294
.....	294
NPM.....	295
104:	296
Examples.....	296
.....	296
.....	296
105: Node.js	297
Examples.....	297
node.js.....	297
.....	297
.....	297
.....	298
.....	298
106:	300
.....	300
Examples.....	300
.....	300
.....	300
Facebook.....	302
-	303
Google Passport.....	303
107:	306
Examples.....	306
MongooseMongoDB.....	306
MongooseExpress.jsMongoDB.....	306
.....	306
.....	306
.....	307

MongooseExpress.jsMongoDB.....	308
.....	308
.....	308
.....	309
MongooseExpress.js\$ textMongoDB.....	310
.....	310
.....	310
.....	311
.....	312
Mongoose.....	313
promisesmongodb.....	313
.....	313
.....	313
.....	315
108: Mongodb.....	316
.....	316
.....	316
Examples.....	316
Node.JSmongoDB.....	316
mongoDBNode.JS.....	316
109:	317
Examples.....	317
.....	317
.....	317
110:	319
.....	319
.....	319
Examples.....	319
.....	319
111:	322
.....	322

.....	322
Examples.....	322
nodejsMVCAPI.....	322
.....	324

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [node-js](#)

It is an unofficial and free Node.js ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Node.js.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: Node.js

Node.js | OGoogleV8 JavaScript。 JavaScript。 。 Node.jsJavaScriptWindowsLinuxNode.js.....

v8.2.1	
V8.2.0	2017719
V8.1.4	2017711
v8.1.3	2017629
V8.1.2	2017615
v8.1.1	2017613
V8.1.0	201768
V8.0.0	2017530
V7.10.0	201752
V7.9.0	2017411
v7.8.0	2017329
v7.7.4	2017321
v7.7.3	2017314
v7.7.2	201738
v7.7.1	201732
v7.7.0	2017228
v7.6.0	2017221
V7.5.0	2017131
V7.4.0	201714
V7.3.0	20161220
v7.2.1	2016126
V7.2.0	20161122
V7.1.0	2016118

7.0.0	20161025
v6.11.0	201766
v6.10.3	201752
v6.10.2	201744
v6.10.1	2017321
v6.10.0	2017221
v6.9.5	2017131
v6.9.4	201715
v6.9.3	201715
v6.9.2	2016126
v6.9.1	20161019
v6.9.0	20161018
v6.8.1	○
v6.8.0	
v6.7.0	2016927
V6.6.0	2016914
V6.5.0	2016826
V6.4.0	2016812
v6.3.1	2016721
v6.3.0	201676
V6.2.2	2016616
V6.2.1	201662
V6.2.0	2016517
V6.1.0	201655
V6.0.0	2016426
v5.12.0	2016623

v5.11.1	201655
v5.11.0	2016421
v5.10.1	201645
V5.10	201641
V5.9	2016316
V5.8	201639
V5.7	2016223
V5.6	201629
V5.5	2016121
V5.4	201616
V5.3	20151215
V5.2	2015129
V5.1	○
V5.0	20151029
V4.4	201638
V4.3	201629
V4.2	20151012
V4.1	2015917
V4.0	2015-09-08
io.js v3.3	201592
io.js v3.2	2015825
io.js v3.1	2015819
io.js v3.0	201584
io.js v2.5	2015728
io.js v2.4	2015717
io.js v2.3	2015613

io.js v2.2	201561
io.js v2.1	2015524
io.js v2.0	201554
io.js v1.8	2015421
io.js v1.7	2015417
io.js v1.6	2015320
io.js v1.5	201536
io.js v1.4	2015227
io.js v1.3	
io.js v1.2	2015211
io.js v1.1	201523
io.js v1.0	2015114
v0.12	201629
v0.11	2013328
v0.10	2013311
V0.9	2012-07-20
V0.8	2012-06-22
V0.7	2012-01-17
V0.6	2011-11-04
V0.5	2011-08-26
V0.4	2011-08-26
V0.3	2011-08-26
V0.2	2011-08-26
V0.1	2011-08-26

Examples

Hello World HTTP

Node.js.

1337HTTPHello, World! 1337.

httpNode.js Node.js httphttp.createServer()HTTP. JavaScript.

```
const http = require('http'); // Loads the http module

http.createServer((request, response) => {

  // 1. Tell the browser everything is OK (Status code 200), and the data is in plain text
  response.writeHead(200, {
    'Content-Type': 'text/plain'
  });

  // 2. Write the announced text to the body of the page
  response.write('Hello, World!\n');

  // 3. Tell the server that all of the response headers and body have been sent
  response.end();

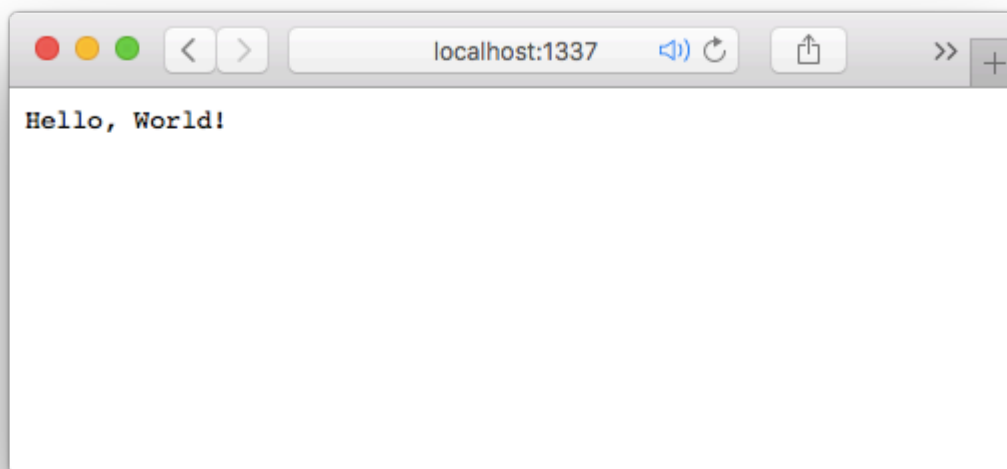
}).listen(1337); // 4. Tells the server what port to be on
```

◦ hello.js

```
node hello.js
```

URL <http://localhost:1337><http://127.0.0.1:1337>.

“HelloWorld”.



Hello World

Node.js. Hello.

Unix

1. . .
2. `chmod 700 FILE_NAME`
3. `./APP_NAME David`

Windows1 `node APP_NAME David`

```
#!/usr/bin/env node

'use strict';

/*
  The command line arguments are stored in the `process.argv` array,
  which has the following structure:
  [0] The path of the executable that started the Node.js process
  [1] The path to this application
  [2-n] the command line arguments

  Example: [ '/bin/node', '/path/to/yourscript', 'arg1', 'arg2', ... ]
  src: https://nodejs.org/api/process.html#process_process_argv
*/

// Store the first argument as username.
var username = process.argv[2];

// Check if the username hasn't been provided.
if (!username) {

  // Extract the filename
  var appName = process.argv[1].split(require('path').sep).pop();

  // Give the user an example on how to use the app.
  console.error('Missing argument! Example: %s YOUR_NAME', appName);

  // Exit the app (success: 0, error: 1).
  // An error will stop the execution chain. For example:
  // ./app.js && ls      -> won't execute ls
  // ./app.js David && ls -> will execute ls
  process.exit(1);
}

// Print the message to the console.
console.log('Hello %s!', username);
```

Node.js.

Node.js.

Windows/.

Mac ◦ `brew install node` **Homebrew** `brew install node` ◦ **HomebrewMacintosh** **Homebrew** ◦

Linux ◦

Node.js `node app.js` `nodejs app.js` `app.js` ◦ **Node.js** ◦

UNIX ◦ **shebang** `#!/usr/bin/env node` ◦ `chmod` ◦ ◦

Node.js `PORT` ◦ `process.env.PORT` ◦

```
http.createServer(function(request, response) {
  // your server code
}).listen(process.env.PORT);
```

```
http.createServer(function(request, response) {
  // your server code
}).listen(process.env.PORT || 3000);
```

3000 ◦

NodeJS

◦ **npm**

```
npm install -g node-inspector
```

```
node-debug app.js
```

Github <https://github.com/node-inspector/node-inspector>

node.js

```
node debug your-script.js
```

```
debugger;
```

node.js 8

```
node --inspect-brk your-script.js
```

`about://inspect` **Google Chrome** `about://inspect` **NodeChrome DevTools** ◦

Express `3000` **HTTP** “HelloWorld” ◦ **ExpressWeb** **HTTP API** ◦

myApp ◦ myApp **JavaScript** hello.js hello.js ◦ npm install --save express **express** ◦ ◦

```
// Import the top-level function of express
const express = require('express');

// Creates an Express application using the top-level function
const app = express();

// Define port number as 3000
const port = 3000;

// Routes HTTP GET requests to the specified path "/" with the specified callback function
app.get('/', function(request, response) {
  response.send('Hello, World!');
});

// Make the app listen on port 3000
app.listen(port, function() {
  console.log('Server listening on http://localhost:' + port);
});
```

```
node hello.js
```

http://localhost:3000 http://127.0.0.1:3000 ◦

Express **Web Apps With Express**

Hello World

HTTP “” “” ◦

if (request.url === 'some/path/here') ◦

```
const http = require('http');

function index (request, response) {
  response.writeHead(200);
  response.end('Hello, World!');
}

http.createServer(function (request, response) {

  if (request.url === '/') {
    return index(request, response);
  }

  response.writeHead(404);
  response.end(http.STATUS_CODES[404]);

}).listen(1337);
```

“” ◦

```
var routes = {
```

```

    '/': function index (request, response) {
      response.writeHead(200);
      response.end('Hello, World!');
    },
    '/foo': function foo (request, response) {
      response.writeHead(200);
      response.end('You are now viewing "foo"');
    }
  }
}

```

2

```

http.createServer(function (request, response) {

  if (request.url in routes) {
    return routes[request.url](request, response);
  }

  response.writeHead(404);
  response.end(http.STATUS_CODES[404]);

}).listen(1337);

```

◦ 404◦

- HTTP Server API◦

TLS

TCPTCP◦

◦ ◦ ◦ ◦ ◦ ◦

openssl genrsa -out private-key.pem 1024

CSR◦ ◦ ◦ ◦

openssl req -new -key private-key.pem -out csr.pem

◦

openssl x509 -req -in csr.pem -signkey private-key.pem -out public-cert.pem

NodeJS◦ ◦

◦ NodeJSrejectUnauthorizedfalse◦ true◦

TLS

```

'use strict';

var tls = require('tls');
var fs = require('fs');

const PORT = 1337;
const HOST = '127.0.0.1'

var options = {
  key: fs.readFileSync('private-key.pem'),
  cert: fs.readFileSync('public-cert.pem')
};

var server = tls.createServer(options, function(socket) {

  // Send a friendly message
  socket.write("I am the server sending you a message.");

  // Print the data that we received
  socket.on('data', function(data) {

    console.log('Received: %s [it is %d bytes long]',
      data.toString().replace(/\n/gm, ""),
      data.length);

  });

  // Let us know when the transmission is over
  socket.on('end', function() {

    console.log('EOT (End Of Transmission)');

  });

});

// Start listening on a specific port and address
server.listen(PORT, HOST, function() {

  console.log("I'm listening at %s, on port %s", HOST, PORT);

});

// When an error occurs, show it.
server.on('error', function(error) {

  console.error(error);

  // Close the connection after the error occurred.
  server.destroy();

});

```

TLS

```

'use strict';

var tls = require('tls');

```

```

var fs = require('fs');

const PORT = 1337;
const HOST = '127.0.0.1'

// Pass the certs to the server and let it know to process even unauthorized certs.
var options = {
  key: fs.readFileSync('private-key.pem'),
  cert: fs.readFileSync('public-cert.pem'),
  rejectUnauthorized: false
};

var client = tls.connect(PORT, HOST, options, function() {

  // Check if the authorization worked
  if (client.authorized) {
    console.log("Connection authorized by a Certificate Authority.");
  } else {
    console.log("Connection not authorized: " + client.authorizationError)
  }

  // Send a friendly message
  client.write("I am the client sending you a message.");

});

client.on("data", function(data) {

  console.log('Received: %s [it is %d bytes long]',
    data.toString().replace(/\n/gm, ""),
    data.length);

  // Close the connection after receiving the message
  client.end();

});

client.on('close', function() {

  console.log("Connection closed");

});

// When an error occurs, show it.
client.on('error', function(error) {

  console.error(error);

  // Close the connection after the error occurred.
  client.destroy();

});

```

REPLHello World

Node.jsREPLRead-Eval-Print-Loop“*Node shell*”。

node ◦


```
$ node
>
```

Node shell>“Hello World”

```
$ node
> "Hello World!"
'Hello World!'
```

Node.js Javascript Chrome V8 C++ Javascript。 Node。

Node34

```
[ 'assert',
  'buffer',
  'c/c++_addons',
  'child_process',
  'cluster',
  'console',
  'crypto',
  'deprecated_apis',
  'dns',
  'domain',
  'events',
  'fs',
  'http',
  'https',
  'module',
  'net',
  'os',
  'path',
  'punycode',
  'querystring',
  'readline',
  'repl',
  'stream',
  'string_decoder',
  'timers',
  'tls_(ssl)',
  'tracing',
  'tty',
  'dgram',
  'url',
  'util',
  'v8',
  'vm',
  'zlib' ]
```

Node API <https://nodejs.org/api/all.html> JSON <https://nodejs.org/api/all.json>。

assert。

ECMAScript 2015 ES6 [TypedArray](#) JavaScript。 [Buffer](#) Node.js API TCP。

ES6 [TypedArray](#) [Buffer](#) Node.js [Uint8Array](#) API。

C / C ++ _

Node.js Addons C++ `require()` Node.js Node.js Node.js JavaScript C / C ++

child_process

`child_process` `popen3`

Node.js Node.js

`console` `Web` `JavaScript`

`crypto` `OpenSSL` `HMAC`

deprecated_apis

Node.js `APIa` `APIb` `APIc` `API`

DNS

`dns`

1. `dns.lookup()`
2. `DNS` `dns` `dns.lookup()`

`API`

Node.js `API` `Function`

FS

`FS` `OPOSIX` `require('fs')`

HTTP

Node.js `HTTP`

HTTPS

`HTTP` `STLS` / `SSL` `HTTP` Node.js

Node.js Node.js

`net` `require('net')` `require('net')`

`os`

`path`

Punycode

Node.js `punycode` ◦

`querystring` `URL` ◦

ReadLine

`readline` `process.stdin` ◦

REPL

`repl` `Read-Eval-Print-Loop` `REPL` ◦

`Node.js` `stream` `API` ◦

`Node.js` `HTTP` `process.stdout` ◦

string_decoder

`string_decoder` `API` `UTF-8` `UTF-16` `Buffer` ◦

`timer` `API` `require('timers')` `API` ◦

`Node.js` `Web` `API` `API` `Node.js` ◦

tls_SSL

`tls` `OpenSSL` `TLSSSL` ◦

`V8` ◦

`Node.js` `--trace-events-enabled` ◦

TTY

`tty` `tty.ReadStream` `tty.WriteStream` ◦ ◦

DGRAM

`dgram` `UDP` ◦

`url` `URL` ◦

UTIL

`util` `Node.js` `API` ◦ ◦

V8

`v8` `Node.js` `V8` `API` ◦

`API` ◦

VM

vmV8API JavaScript

vm

zlib

zlibGzipDeflate / Inflate

HTTPS Web

node.jsHTTPHTTPSWeb

1

1. mkdir conf

2. cd conf

3. ca.cnf

```
wget https://raw.githubusercontent.com/anders94/https-authorized-clients/master/keys/ca.cnf
```

4. openssl req -new -x509 -days 9999 -config ca.cnf -keyout ca-key.pem -out ca-cert.pem

5. ca-key.pemca-cert.pem

```
openssl genrsa -out key.pem 4096
```

6. server.cnf

```
wget https://raw.githubusercontent.com/anders94/https-authorized-clients/master/keys/server.cnf
```

7. openssl req -new -config server.cnf -key key.pem -out csr.pem

8. openssl x509 -req -extfile server.cnf -days 999 -passin "pass:password" -in csr.pem -CA ca-cert.pem -CAkey ca-key.pem -CAcreateserial -out cert.pem

2

1. sudo cp ca-crt.pem /usr/local/share/ca-certificates/ca-crt.pem

2.

CA

```
sudo update-ca-certificates
```

3

server.js°

Node.jsHTTPS

```
var https = require('https');
var fs = require('fs');

var httpsOptions = {
  key: fs.readFileSync('path/to/server-key.pem'),
  cert: fs.readFileSync('path/to/server-crt.pem')
};

var app = function (req, res) {
  res.writeHead(200);
  res.end("hello world\n");
}

https.createServer(httpsOptions, app).listen(4433);
```

http

```
var http = require('http');
var https = require('https');
var fs = require('fs');

var httpsOptions = {
  key: fs.readFileSync('path/to/server-key.pem'),
  cert: fs.readFileSync('path/to/server-crt.pem')
};

var app = function (req, res) {
  res.writeHead(200);
  res.end("hello world\n");
}

http.createServer(app).listen(8888);
https.createServer(httpsOptions, app).listen(4433);
```

1. server.js

```
cd /path/to
```

2. server.js

```
node server.js
```

Node.js <https://riptutorial.com/zh-TW/node-js/topic/340/node-js>

2: Arduino nodeJs

Node.Js Arduino Uno.

Examples

Node Js serialport Arduino

js

Node.js serialport Arduino.

```
npm install express --save
npm install serialport --save
```

app.js

```
const express = require('express');
const app = express();
var SerialPort = require("serialport");

var port = 3000;

var arduinoCOMPort = "COM3";

var arduinoSerialPort = new SerialPort(arduinoCOMPort, {
  baudrate: 9600
});

arduinoSerialPort.on('open', function() {
  console.log('Serial Port ' + arduinoCOMPort + ' is opened.');
```

```
});
```

```
app.get('/', function (req, res) {
```

```
  return res.send('Working');
```

```
})
```

```
app.get('/:action', function (req, res) {
```

```
  var action = req.params.action || req.param('action');
```

```
  if(action == 'led'){
    arduinoSerialPort.write("w");
    return res.send('Led light is on!');
```

```
  }
```

```
  if(action == 'off') {
    arduinoSerialPort.write("t");
    return res.send("Led light is off!");
```

```
  }
```

```
  return res.send('Action: ' + action);
```

```
});  
  
app.listen(port, function () {  
  console.log('Example app listening on port http://0.0.0.0:' + port + '!');  
});
```

```
node app.js
```

Arduino

```
// the setup function runs once when you press reset or power the board  
void setup() {  
  // initialize digital pin LED_BUILTIN as an output.  
  
  Serial.begin(9600); // Begin listening on port 9600 for serial  
  
  pinMode(LED_BUILTIN, OUTPUT);  
  
  digitalWrite(LED_BUILTIN, LOW);  
}  
  
// the loop function runs over and over again forever  
void loop() {  
  
  if(Serial.available() > 0) // Read from serial port  
  {  
    char ReaderFromNode; // Store current character  
    ReaderFromNode = (char) Serial.read();  
    convertToState(ReaderFromNode); // Convert character to state  
  }  
  delay(1000);  
}  
  
void convertToState(char chr) {  
  if(chr=='o'){  
    digitalWrite(LED_BUILTIN, HIGH);  
    delay(100);  
  }  
  if(chr=='f'){  
    digitalWrite(LED_BUILTIN, LOW);  
    delay(100);  
  }  
}
```

1. arduino.
- 2.

js express serverled.

LED

```
http://0.0.0.0:3000/led
```

LED

`http://0.0.0.0:3000/off`

ArduinonodeJs <https://riptutorial.com/zh-TW/node-js/topic/10509/arduinonodejs>

3: async.js

-
- [arg1 [...]]
- ◦ null
- callbacknullmyResult;
- **errresultseq...**
- callbacknullmyResultmyCustomArgument;
- ◦ ◦
- ERR;

Examples

[async.paralleltasksafterTasksCallback](#) ◦

◦

```
function shortTimeFunction(callback) {
  setTimeout(function() {
    callback(null, 'resultOfShortTime');
  }, 200);
}

function mediumTimeFunction(callback) {
  setTimeout(function() {
    callback(null, 'resultOfMediumTime');
  }, 500);
}

function longTimeFunction(callback) {
  setTimeout(function() {
    callback(null, 'resultOfLongTime');
  }, 1000);
}

async.parallel([
  shortTimeFunction,
  mediumTimeFunction,
  longTimeFunction
],
function(err, results) {
  if (err) {
    return console.error(err);
  }

  console.log(results);
});
```

["resultOfShortTime", "resultOfMediumTime", "resultOfLongTime"]。

async.parallel()

tasks。

。

```
async.parallel({
  short: shortTimeFunction,
  medium: mediumTimeFunction,
  long: longTimeFunction
},
function(err, results) {
  if (err) {
    return console.error(err);
  }

  console.log(results);
});
```

{short: "resultOfShortTime", medium: "resultOfMediumTime", long: "resultOfLongTime"}。

。 。 。

```
async.parallel({
  short: function shortTimeFunction(callback) {
    setTimeout(function() {
      callback(null, 'resultOfShortTime1', 'resultOfShortTime2');
    }, 200);
  },
  medium: function mediumTimeFunction(callback) {
    setTimeout(function() {
      callback(null, 'resultOfMediumTime1', 'resultOfMeiumTime2');
    }, 500);
  }
},
function(err, results) {
  if (err) {
    return console.error(err);
  }

  console.log(results);
});
```

```
{
  short: ["resultOfShortTime1", "resultOfShortTime2"],
  medium: ["resultOfMediumTime1", "resultOfMediumTime2"]
}
```

。

async.seriesTasksAfterTasksCallback。

“”。

```
function shortTimeFunction(callback) {
  setTimeout(function() {
    callback(null, 'resultOfShortTime');
  }, 200);
}

function mediumTimeFunction(callback) {
  setTimeout(function() {
    callback(null, 'resultOfMediumTime');
  }, 500);
}

function longTimeFunction(callback) {
  setTimeout(function() {
    callback(null, 'resultOfLongTime');
  }, 1000);
}

async.series([
  mediumTimeFunction,
  shortTimeFunction,
  longTimeFunction
],
function(err, results) {
  if (err) {
    return console.error(err);
  }

  console.log(results);
});
```

["resultOfMediumTime", "resultOfShortTime", "resultOfLongTime"]。

async.series()

tasks。

。

```
async.series({
  short: shortTimeFunction,
  medium: mediumTimeFunction,
  long: longTimeFunction
},
function(err, results) {
  if (err) {
    return console.error(err);
  }

  console.log(results);
});
```

{short: "resultOfShortTime", medium: "resultOfMediumTime", long: "resultOfLongTime"}。

[async.waterfalltasksafterTasksCallback](#) ◦ [async.series](#) ◦

“”◦

```
function getUserRequest(callback) {
  // We simulate the request with a timeout
  setTimeout(function() {
    var userResult = {
      name : 'Aamu'
    };

    callback(null, userResult);
  }, 500);
}

function getUserFriendsRequest(user, callback) {
  // Another request simulate with a timeout
  setTimeout(function() {
    var friendsResult = [];

    if (user.name === "Aamu"){
      friendsResult = [{
        name : 'Alice'
      }, {
        name: 'Bob'
      }];
    }

    callback(null, friendsResult);
  }, 500);
}

async.waterfall([
  getUserRequest,
  getUserFriendsRequest
],
function(err, results) {
  if (err) {
    return console.error(err);
  }

  console.log(JSON.stringify(results));
});
```

resultsfriendsResult ◦

async.times

node.jsfor◦ for◦ **asycn.times**

```
function recursiveAction(n, callback)
{
  //do whatever want to do repeatedly
  callback(err, result);
}
async.times(5, function(n, next) {
  recursiveAction(n, function(err, result) {
```

```

        next(err, result);
    });
}, function(err, results) {
    // we should now have 5 result
});

```

- **async.timesSeries**

async.each

async.each ◦ ◦ ◦

```

function createUser(userName, callback)
{
    //create user in db
    callback(null)//or error based on creation
}

var arrayOfData = ['Ritu', 'Sid', 'Tom'];
async.each(arrayOfData, function(eachUserName, callback) {

    // Perform operation on each user.
    console.log('Creating user '+eachUserName);
    //Returning callback is must. Else it wont get the final callback, even if we miss to
    return one callback
    createUser(eachUserName, callback);

}, function(err) {
    //If any of the user creation failed may throw error.
    if( err ) {
        // One of the iterations produced an error.
        // All processing will now stop.
        console.log('unable to create user');
    } else {
        console.log('All user created successfully');
    }
});

```

async.eachSeries

async.series

/ async.series◦ /

```

var async = require('async');
async.series([functioncallback{console.log'First Execute ..'; callbacknull
'userData';}functioncallback{console.log'Second Execute ..'; callbacknull
'userData';}],functionerrresult{console.logresult;});

```

```

//
First Execute .. Second Execute .. ['userData"userData'] //

```

[async.js](https://riptutorial.com/zh-TW/node-js/topic/3972/async-js) <https://riptutorial.com/zh-TW/node-js/topic/3972/async-js>

4: CLI

- `node [options] [v8 options] [script.js | -e"script"] []`

Examples

```
-v, --version
```

v0.1.3。

```
-h, --help
```

v0.1.3。 ◦

```
-e, --eval "script"
```

v0.5.2JavaScript。 REPL。

```
-p, --print "script"
```

v0.6.4-e。

```
-c, --check
```

v5.0.0。

```
-i, --interactive
```

v0.7.7stdinREPL。

```
-r, --require module
```

v1.6.0。

require。 ◦

```
--no-deprecation
```

v0.8.0。

```
--trace-deprecation
```

v0.8.0。

```
--throw-deprecation
```

v0.11.14。

```
--no-warnings
```

v6.0.0。

```
--trace-warnings
```

v6.0.0。

```
--trace-sync-io
```

v2.1.0 / O。

```
--zero-fill-buffers
```

v6.0.0 BufferSlowBuffer。

```
--preserve-symlinks
```

v6.3.0。

Node.js Node.js“” 。 。 moduleA moduleB

```
{appDir}
├─ app
│  ├─ index.js
│  └─ node_modules
│     ├─ moduleA -> {appDir}/moduleA
│     └─ moduleB
│        ├─ index.js
│        └─ package.json
└─ moduleA
   ├─ index.js
   └─ package.json
```

--preserve-symlinks Node.js。

--preserve-symlinks。 Node.js。

```
--track-heap-objects
```

v2.4.0。

```
--prof-process
```

v6.0.0v8--profProcess v8 profiler。

```
--v8-options
```

v0.1.3v8。

v8 - _。

- stack-trace-limit--stack_trace_limit。

```
--tls-cipher-list=list
```

v4.0.0TLS。 Node.js。

```
--enable-fips
```

v6.0.0FIPS。 ./configure --openssl-fipsNode.js

```
--force-fips
```

v6.0.0FIPS。 。 --enable-fips

```
--icu-data-dir=file
```

v0.11.15ICU。 NODE_ICU_DATA

```
Environment Variables
```

```
NODE_DEBUG=module[,...]
```

v0.1.32" - 。

```
NODE_PATH=path[:...]
```

v0.1.32" - 。

Windows';' - 。

```
NODE_DISABLE_COLORS=1
```

v0.3.01REPL。

```
NODE_ICU_DATA=file
```

v0.11.15 ICUIntl。 small-icu。


```
NODE_REPL_HISTORY=file
```

v5.0.0REPL。 / .node_repl_history。 “”REPL。

CLI <https://riptutorial.com/zh-TW/node-js/topic/6013/cli>

5: ExpressJSRoute-Controller-Service

Examples

- - -

```
|—models
|   |—user.model.js
|—routes
|   |—user.route.js
|—services
|   |—user.service.js
|—controllers
|   |—user.controller.js
```

o

-

- API

- o

-

o o

- - -

user.model.js

```
var mongoose = require('mongoose')

const UserSchema = new mongoose.Schema({
  name: String
})

const User = mongoose.model('User', UserSchema)

module.exports = User;
```

user.routes.js

```
var express = require('express');
var router = express.Router();

var UserController = require('../controllers/user.controller')
```

```
router.get('/', UserController.getUsers)

module.exports = router;
```

user.controllers.js

```
var UserService = require('../services/user.service')

exports.getUsers = async function (req, res, next) {
  // Validate request parameters, queries using express-validator

  var page = req.params.page ? req.params.page : 1;
  var limit = req.params.limit ? req.params.limit : 10;
  try {
    var users = await UserService.getUsers({}, page, limit)
    return res.status(200).json({ status: 200, data: users, message: "Succesfully Users Retrieved" });
  } catch (e) {
    return res.status(400).json({ status: 400, message: e.message });
  }
}
```

user.services.js

```
var User = require('../models/user.model')

exports.getUsers = async function (query, page, limit) {

  try {
    var users = await User.find(query)
    return users;
  } catch (e) {
    // Log Errors
    throw Error('Error while Paginating Users')
  }
}
```

ExpressJSRoute-Controller-Service <https://riptutorial.com/zh-TW/node-js/topic/10785/expressjs-route-controller-service>

6: HTTP

Examples

http

HTTP。

http_server.js

```
var http = require('http');

var httpPort = 80;

http.createServer(handler).listen(httpPort, start_callback);

function handler(req, res) {

    var clientIP = req.connection.remoteAddress;
    var connectUsing = req.connection.encrypted ? 'SSL' : 'HTTP';
    console.log('Request received: ' + connectUsing + ' ' + req.method + ' ' + req.url);
    console.log('Client IP: ' + clientIP);

    res.writeHead(200, "OK", {'Content-Type': 'text/plain'});
    res.write("OK");
    res.end();
    return;
}

function start_callback(){
    console.log('Start HTTP on port ' + httpPort)
}
```

http_server.js

```
node http_server.js
```

```
> Start HTTP on port 80
```

```
http://127.0.0.1:80
```

Linux

```
curl 127.0.0.1:80
```

```
ok
```

```
> Request received: HTTP GET /
> Client IP: ::ffff:127.0.0.1
```

http

http

http_client.js

```
var http = require('http');

var options = {
  hostname: '127.0.0.1',
  port: 80,
  path: '/',
  method: 'GET'
};

var req = http.request(options, function(res) {
  console.log('STATUS: ' + res.statusCode);
  console.log('HEADERS: ' + JSON.stringify(res.headers));
  res.setEncoding('utf8');
  res.on('data', function (chunk) {
    console.log('Response: ' + chunk);
  });
  res.on('end', function (chunk) {
    console.log('Response ENDED');
  });
});

req.on('error', function(e) {
  console.log('problem with request: ' + e.message);
});

req.end();
```

http_client.js

```
node http_client.js
```

```
> STATUS: 200
> HEADERS: {"content-type":"text/plain","date":"Thu, 21 Jul 2016 11:27:17 GMT","connection":"close","transfer-encoding":"chunked"}
> Response: OK
> Response ENDED
```

http

HTTP <https://riptutorial.com/zh-TW/node-js/topic/2973/http>

7: Koa Framework v2

Examples

Hello World

```
const Koa = require('koa')

const app = new Koa()

app.use(async ctx => {
  ctx.body = 'Hello World'
})

app.listen(8080)
```

```
app.use(async (ctx, next) => {
  try {
    await next() // attempt to invoke the next middleware downstream
  } catch (err) {
    handleError(err, ctx) // define your own error handling function
  }
})
```

Koa Framework v2 <https://riptutorial.com/zh-TW/node-js/topic/6730/koa-framework-v2>

8: Lodash

LodashJavaScript。

Examples

lodash。

```
let lodash = require('lodash');

var countries = [
  {"key": "DE", "name": "Deutschland", "active": false},
  {"key": "ZA", "name": "South Africa", "active": true}
];

var filteredByFunction = lodash.filter(countries, function (country) {
  return country.key === "DE";
});
// => [{"key": "DE", "name": "Deutschland"}];

var filteredByObjectProperties = lodash.filter(countries, { "key": "DE" });
// => [{"key": "DE", "name": "Deutschland"}];

var filteredByProperties = lodash.filter(countries, ["key", "ZA"]);
// => [{"key": "ZA", "name": "South Africa"}];

var filteredByProperty = lodash.filter(countries, "active");
// => [{"key": "ZA", "name": "South Africa"}];
```

Lodash <https://riptutorial.com/zh-TW/node-js/topic/9161/lodash>

9: metalsmith

Examples

nodenpm package.json

```
npm install --save-dev metalsmith metalsmith-in-place handlebars
```

build.js

```
var metalsmith = require('metalsmith');
var handlebars = require('handlebars');
var inPlace = require('metalsmith-in-place');

Metalsmith(__dirname)
  .use(inPlace('handlebars'))
  .build(function(err) {
    if (err) throw err;
    console.log('Build finished!');
  });
```

src srcindex.html

```
---
title: My awesome blog
---
<h1>{{ title }}</h1>
```

node build.js src buildindex.html

```
<h1>My awesome blog</h1>
```

metalsmith <https://riptutorial.com/zh-TW/node-js/topic/6111/metalsmith>

10: Mongodb

- `Db.collection.insertOne` *document options wtimeout serializeFunctions forceServerObjectId bypassDocumentValidation callback*
- `Db.collection.insertMany` *[documents] options wtimeout serializeFunctions forceServerObjectId bypassDocumentValidation callback*
- `Db.collection.find`
- `Db.collection.updateOne` *filter update options upsert wtimeout bypassDocumentValidation callback*
- `Db.collection.updateMany` *filter update options upsert wtimeout callback*
- `Db.collection.deleteOne` *filter options upsert wtimeout callback*
- `Db.collection.deleteMany` *filter options upsert wtimeout callback*

	javascript
	<i>null</i>
w ^	
wtimeout	◦ <i>null</i>
↑	<i>false</i>
UPSERT	<i>false</i>
	<i>/false</i>
serializeFunctions	<i>false</i>
forceServerObjectId	<i>_id false</i>
bypassDocumentValidation	MongoDB 3.2 <i>false</i>

Examples

MongoDB

MongoDB".

```
const MongoClient = require('mongodb').MongoClient;
```

```

var url = 'mongodb://localhost:27017/test';

MongoClient.connect(url, function(err, db) { // MongoClient method 'connect'
  if (err) throw new Error(err);
  console.log("Connected!");
  db.close(); // Don't forget to close the connection when you are done
});

```

MongoClient.connect()

`MongoClient.connect url options callback`

url	ip / hostname
options	<i>null</i>
callback	

callback

- err - errerr
- db object - MongoDB

“myFirstDocument”² greetingsfarewell

```

const MongoClient = require('mongodb').MongoClient;

const url = 'mongodb://localhost:27017/test';

MongoClient.connect(url, function (err, db) {
  if (err) throw new Error(err);
  db.collection('myCollection').insertOne({ // Insert method 'insertOne'
    "myFirstDocument": {
      "greetings": "Hellu",
      "farewell": "Bye"
    }
  }, function (err, result) {
    if (err) throw new Error(err);
    console.log("Inserted a document into the myCollection collection!");
    db.close(); // Don't forget to close the connection when you are done
  });
});

```

insertOne()

`db.collection collection.insertOne document options callback`

collection		
document		

options	<i>null</i>
callback	

callback

- err - errerr
- result **object** -

'myCollection'.

```
const MongoClient = require('mongodb').MongoClient;

const url = 'mongodb://localhost:27017/test';

MongoClient.connect(url, function (err, db) {
  if (err) throw new Error(err);
  var cursor = db.collection('myCollection').find(); // Read method 'find'
  cursor.each(function (err, doc) {
    if (err) throw new Error(err);
    if (doc != null) {
      console.log(doc); // Print all documents
    } else {
      db.close(); // Don't forget to close the connection when you are done
    }
  });
});
```

find()

db.collection *collection* .find

collection

```
{ greetings: 'Hellu' }{ greetings: 'Whut?' }
```

```
const MongoClient = require('mongodb').MongoClient;

const url = 'mongodb://localhost:27017/test';

MongoClient.connect(url, function (err, db) {
  if (err) throw new Error(err);
  db.collection('myCollection').updateOne({ // Update method 'updateOne'
    greetings: "Hellu" },
    { $set: { greetings: "Whut?" } },
    function (err, result) {
      if (err) throw new Error(err);
      db.close(); // Don't forget to close the connection when you are done
    });
});
```

updateOne()

db.collection .updateOne ◦

filter		
update		
options		<i>null</i>
callback		

callback

- err -
- db **object** - MongoDB

```
{ greetings: 'Whut?' }{ greetings: 'Whut?' }
```

```
const MongoClient = require('mongodb').MongoClient;

const url = 'mongodb://localhost:27017/test';

MongoClient.connect(url, function (err, db) {
  if (err) throw new Error(err);
  db.collection('myCollection').deleteOne(// Delete method 'deleteOne'
    { greetings: "Whut?" },
    function (err, result) {
      if (err) throw new Error(err);
      db.close(); // Don't forget to close the connection when you are done
    });
});
```

`deleteOne()`

db.collection *collection* .deleteOne *filter options callback*

filter		
options		<i>null</i>
callback		

callback

- err -
- db **object** - MongoDB

“””

```
const MongoClient = require('mongodb').MongoClient;

const url = 'mongodb://localhost:27017/test';
```

```

MongoClient.connect(url, function (err, db) {
  if (err) throw new Error(err);
  db.collection('myCollection').deleteMany(// MongoDB delete method 'deleteMany'
    { farewell: "okay" }, // Delete ALL documents with the property 'farewell: okay'
    function (err, result) {
      if (err) throw new Error(err);
      db.close(); // Don't forget to close the connection when you are done
    });
});

```

`deleteMany()`

`db.collection collection .deleteMany filter options callback`

filter		
options		<i>null</i>
callback		

callback

- err -
- db object - MongoDB

```

MongoDB.connect('mongodb://localhost:27017/databaseName', function(error, database) {
  if(error) return console.log(error);
  const collection = database.collection('collectionName');
  collection.insert({key: 'value'}, function(error, result) {
    console.log(error, result);
  });
});

```

promises

```

const MongoDB = require('mongodb');

MongoDB.connect('mongodb://localhost:27017/databaseName')
  .then(function(database) {
    const collection = database.collection('collectionName');
    return collection.insert({key: 'value'});
  })
  .then(function(result) {
    console.log(result);
  });

```

Mongodb <https://riptutorial.com/zh-TW/node-js/topic/5002/mongodb>

11: MSSQL

nodejsnpmAPI。 mssqlmssqlnodejsSQL。

mssql。 ◦

◦

Examples

SQL。 mssql npm

sql server。

1/。 *npm init*package.json。

```
mkdir mySqlApp
//folder created
cd mwSqlApp
//change to newly created directory
npm init
//answer all the question ..
npm install
//This will complete quickly since we have not added any packages to our app.
```

2App.js sql db。

```
sudo gedit App.js
//This will create App.js file , you can use your fav. text editor :)
npm install --save mssql
//This will install the mssql package to you app
```

3mssql。

```
console.log("Hello world, This is an app to connect to sql server.");
var config = {
  "user": "myusername", //default is sa
  "password": "yourStrong(!)Password",
  "server": "localhost", // for local machine
  "database": "staging", // name of database
  "options": {
    "encrypt": true
  }
}

sql.connect(config, err => {
  if(err){
    throw err ;
  }
  console.log("Connection Successful !");
});
```

```
new sql.Request().query('select 1 as number', (err, result) => {
  //handle err
  console.dir(result)
  // This example uses callbacks strategy for getting results.
})

});

sql.on('error', err => {
  // ... error handler
  console.log("Sql database connection error " ,err);
})
```

4sql server.

```
node App.js
// Output :
// Hello world, This is an app to connect to sql server.
// Connection Successful !
// 1
```

promisesasyncmssql

-
- /

MSSQL <https://riptutorial.com/zh-TW/node-js/topic/9884/mssql>

12: Mysql

Examples

-
- ◦
- -

```
var pool = mysql.createPool({
  connectionLimit : 10,
  host            : 'example.org',
  user           : 'bobby',
  password       : 'pass'
});

pool.getConnection(function(err, connection){
  if(err){
    return cb(err);
  }
  connection.changeUser({database : "firm1"});
  connection.query("SELECT * from history", function(err, data){
    connection.release();
    cb(err, data);
  });
});
```

-

```
{
  connectionLimit : 10,
  host            : 'example.org',
  user           : 'bobby',
  password       : 'pass'
}
```

-

```
connection.changeUser({database : "firm1"});
```

Mysql <https://riptutorial.com/zh-TW/node-js/topic/6353/mysql>

13: MySQL

MYSQLNode.js。 nodejsmysql。

Examples

SQL。 aminadav

```
var username = 'aminadav';
var querystring = 'SELECT name, email from users where name = ?';
connection.query(querystring, [username], function(err, rows, fields) {
  if (err) throw err;
  if (rows.length) {
    rows.forEach(function(row) {
      console.log(row.name, 'email address is', row.email);
    });
  } else {
    console.log('There were no results.');
```

。

MySQL。 10220。 10。

```
var pool = mysql.createPool({
  connectionLimit : 10,
  host             : 'example.org',
  user             : 'bobby',
  password         : 'pass',
  database         : 'schema'
});

for(var i=0;i<10;i++){
  pool.query('SELECT ` as example', function(err, rows, fields) {
    if (err) throw err;
    console.log(rows[0].example); //Show 1
  });
}
```

10。

pool。 。 MySQL。

。

。 。

。 -

```

var pool = mysql.createPool({
  connectionLimit : 10,
  host            : 'example.org',
  user            : 'bobby',
  password        : 'pass'
});

pool.getConnection(function(err, connection){
  if(err){
    return cb(err);
  }
  connection.changeUser({database : "firm1"});
  connection.query("SELECT * from history", function(err, data){
    connection.release();
    cb(err, data);
  });
});

```

-

```

{
  connectionLimit : 10,
  host            : 'example.org',
  user            : 'bobby',
  password        : 'pass'
}

```

-

```

connection.changeUser({database : "firm1"});

```

MySQL

MySQL [mysql](#)。 Node.js [MySQL](#)。

```

npm install --save mysql

```

[mysql](#)。

```

const mysql = require('mysql');
const connection = mysql.createConnection({
  host      : 'localhost',
  user      : 'me',
  password  : 'secret',
  database  : 'database_schema'
});

connection.connect();

// Execute some query statements
// I.e. SELECT * FROM FOO

connection.end();

```

connection

-
- error rows◦ ◦ ◦

```
connection.query('SELECT name,email from users', function(err, rows, fields) {
  if (err) throw err;

  console.log('There are:', rows.length, ' users');
  console.log('First user name is:',rows[0].name)
});
```

MySQL

```
SELECT 1;
SELECT 2;
```

pool.query◦

pool.getConnection

```
pool.getConnection(function (err, conn) {
  if (err) return callback(err);

  conn.query('SELECT 1 AS seq', function (err, rows) {
    if (err) throw err;

    conn.query('SELECT 2 AS seq', function (err, rows) {
      if (err) throw err;

      conn.release();
      callback();
    });
  });
});
```

releaseMySQL

MySQL MySQL ◦

err

```
var q = mysql.query('SELECT `name` FROM `pokedex` WHERE `id` = ?', [ 25 ], function (err,
result) {
  if (err) {
    // Table 'test.pokedex' doesn't exist
    err.query = q.sql; // SELECT `name` FROM `pokedex` WHERE `id` = 25
    callback(err);
  }
  else {
    callback(null, result);
  }
});
```

```
// db.js
```

```
const mysql = require('mysql');

const pool = mysql.createPool({
  connectionLimit : 10,
  host             : 'example.org',
  user             : 'bob',
  password         : 'secret',
  database         : 'my_db'
});

module.export = {
  getConnection: (callback) => {
    return pool.getConnection(callback);
  }
}
```

```
// app.js

const db = require('./db');

db.getConnection((err, conn) => {
  conn.query('SELECT something from sometable', (error, results, fields) => {
    // get the results
    conn.release();
  });
});
```

MySQL <https://riptutorial.com/zh-TW/node-js/topic/1406/mysql>

14: Node.js / Express.js MongoDB

MEAN MongoDB NoSQL Express Mongo Mongoose

<http://mongoosejs.com/docs/guide.html>

Examples

MongoDB

```
npm install --save mongodb
npm install --save mongoose //A simple wrapper for ease of development
```

index.js server.js

```
const express = require('express');
const mongodb = require('mongodb');
const mongoose = require('mongoose');
const mongoConnectionString = 'http://localhost/database name';

mongoose.connect(mongoConnectionString, (err) => {
  if (err) {
    console.log('Could not connect to the database');
  }
});
```

Mongoose

```
const Schema = mongoose.Schema;
const ObjectId = Schema.Types.ObjectId;

const Article = new Schema({
  title: {
    type: String,
    unique: true,
    required: [true, 'Article must have title']
  },
  author: {
    type: ObjectId,
    ref: 'User'
  }
});

module.exports = mongoose.model('Article', Article);
```

◦ MongoDB Mongoose JSON BSON ◦ ◦

new Schema ◦ JSON ◦ ◦ ◦

◦ ◦

ObjectIds。 ""。 ObjectId。 。

API。

Mongo

GET。 ./db/models/Article.js 。

```
const express = require('express');
const Articles = require('./db/models/Article');

module.exports = function (app) {
  const routes = express.Router();

  routes.get('/articles', (req, res) => {
    Articles.find().limit(5).lean().exec((err, doc) => {
      if (doc.length > 0) {
        res.send({ data: doc });
      } else {
        res.send({ success: false, message: 'No documents retrieved' });
      }
    });
  });

  app.use('/api', routes);
};
```

HTTP。

1. 。
2. BSON。 。
3. findfindOne doc.length0.find
4. 。
5. 。

```
const app = express();
require('./path/to/this/file')(app) //
```

[Node.js / Express.jsMongoDB https://riptutorial.com/zh-TW/node-js/topic/9020/node-js---express-jsmongodb](https://riptutorial.com/zh-TW/node-js/topic/9020/node-js---express-jsmongodb)

15: Node.js STDINSTDOUT

node.js。

Node.js。requireNode.js。

Examples

process.stdin。

process.stdoutstdoutWritable。

```
process.stdin.resume()
console.log('Enter the data to be displayed ');
process.stdin.on('data', function(data) { process.stdout.write(data) })
```

Node.js STDINSTDOUT <https://riptutorial.com/zh-TW/node-js/topic/8961/node-js-stdinstdout->

16: Node.js v6

6LTS。 ES6。 ◦

Examples

```
function addTwo(a, b = 2) {  
    return a + b;  
}  
  
addTwo(3) // Returns the result 5
```

◦

```
function argumentLength(...args) {  
    return args.length;  
}  
  
argumentLength(5) // returns 1  
argumentLength(5, 3) //returns 2  
argumentLength(5, 3, 6) //returns 3
```

...◦ ◦

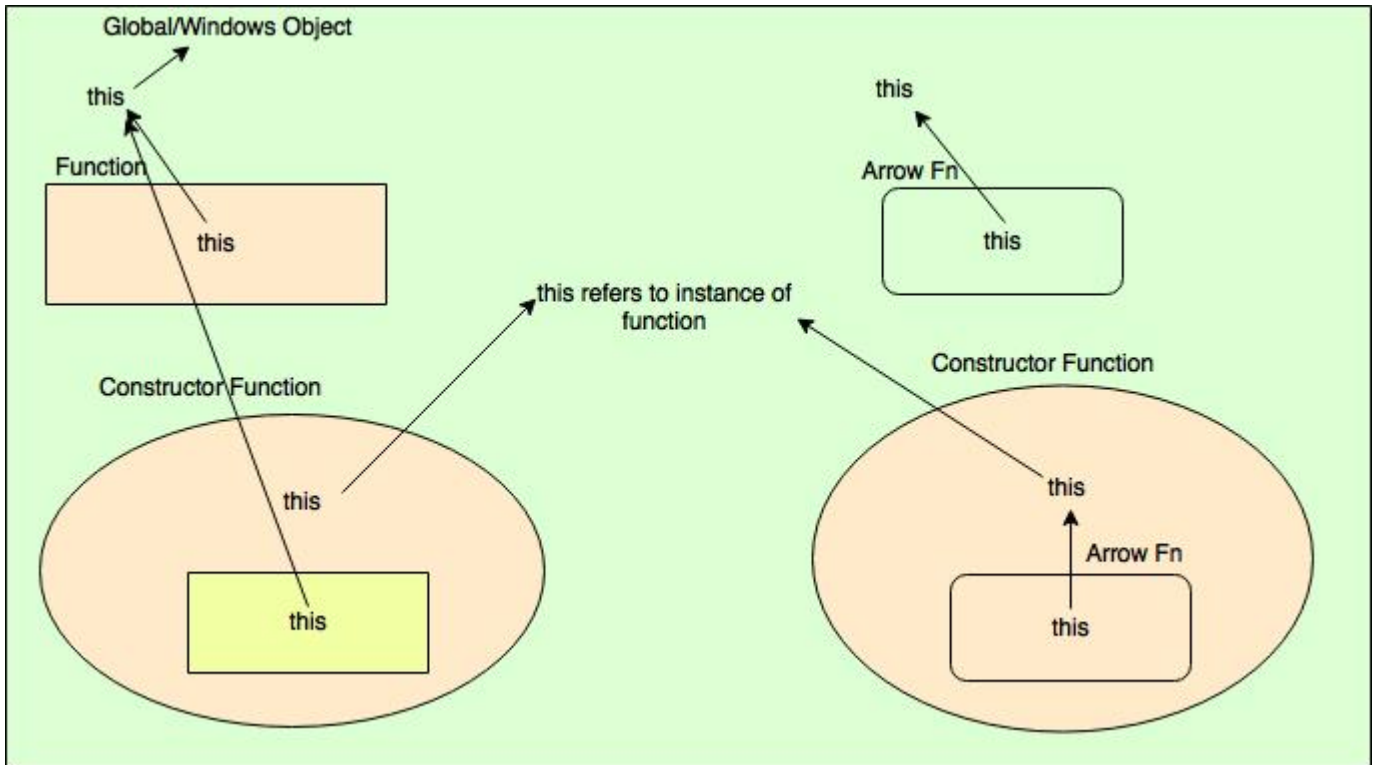
```
function myFunction(x, y, z) { }  
var args = [0, 1, 2];  
myFunction(...args);
```

◦ ...

ECMAScript 6◦

```
// traditional way of declaring and defining function  
var sum = function(a,b)  
{  
    return a+b;  
}  
  
// Arrow Function  
let sum = (a, b)=> a+b;  
  
//Function definition using multiple lines  
let checkIfEven = (a) => {  
    if( a % 2 == 0 )  
        return true;  
    else  
        return false;  
}
```

“this”



```

var normalFn = function(){
  console.log(this) // refers to global/window object.
}

var arrowFn = () => console.log(this); // refers to window or global object as function is
defined in scope of global/window object

var service = {
  constructorFn : function(){
    console.log(this); // refers to service as service object used to call method.

    var nestedFn = function(){
      console.log(this); // refers window or global object because no instance object
was used to call this method.
    }
    nestedFn();
  },
  arrowFn : function(){
    console.log(this); // refers to service as service object was used to call method.
    let fn = () => console.log(this); // refers to service object as arrow function
defined in function which is called using instance object.
    fn();
  }
}

// calling defined functions
constructorFn();
arrowFn();

```

```
service.constructorFn();  
service.arrowFn();
```

◦

/◦

◦ ◦

service◦

- *Node.Jswindows*◦

Node.js v6 <https://riptutorial.com/zh-TW/node-js/topic/8593/node-js-v6>

17: node.jsWindows

Active Directory APIS[activedirectory2adldap](#) ◦

Examples

activedirectory

[GitHub](#)[NPM](#) ◦

```
npm install --save activedirectory
```

```
// Initialize
var ActiveDirectory = require('activedirectory');
var config = {
  url: 'ldap://dc.domain.com',
  baseDN: 'dc=domain,dc=com'
};
var ad = new ActiveDirectory(config);
var username = 'john.smith@domain.com';
var password = 'password';
// Authenticate
ad.authenticate(username, password, function(err, auth) {
  if (err) {
    console.log('ERROR: '+JSON.stringify(err));
    return;
  }
  if (auth) {
    console.log('Authenticated!');
  }
  else {
    console.log('Authentication failed!');
  }
});
```

[node.jsWindows](#) <https://riptutorial.com/zh-TW/node-js/topic/10612/node-jswindows>

18: Node.JS

Examples

NodeJS

--debug ◦ --debug=<port>--debug=<port> ◦

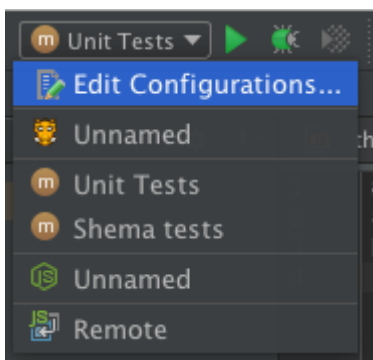
```
Debugger listening on port <port>
```

◦

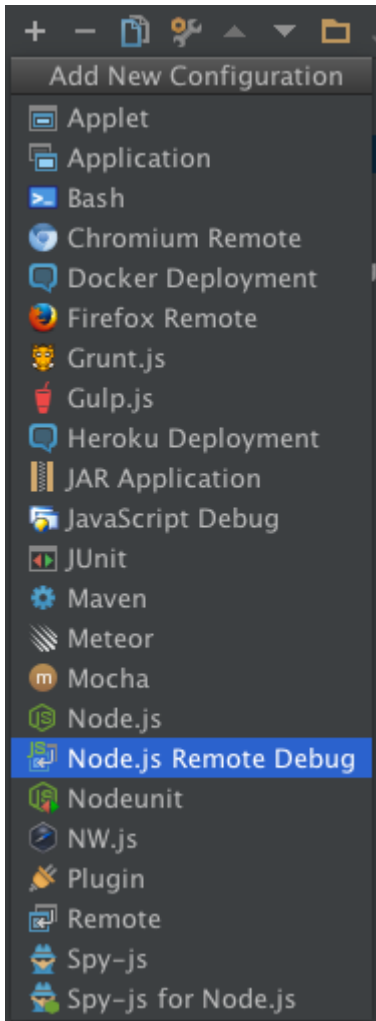
IDE ◦

IntelliJ / Webstorm

1. NodeJS
- 2.



3. + > **Node.js**



4.

Name: Share Single instance only

Host:

Port:

o

Linux

Linux

```
socat TCP-LISTEN:9958,fork TCP:127.0.0.1:5858 &
```

9958。

[Node.JS https://riptutorial.com/zh-TW/node-js/topic/6335/node-js](https://riptutorial.com/zh-TW/node-js/topic/6335/node-js)

19: Node.JSMongoDB。

mongo dbnodejsCRUD。

◦

◦ ◦

mongoose

Examples

mongomongoose。

Mongoosetootmongoose

```
npm install mongoose
```

◦

```
var mongoose = require('mongoose');

//connect to the test database running on default mongod port of localhost
mongoose.connect('mongodb://localhost/test');

//Connecting with custom credentials
mongoose.connect('mongodb://USER:PASSWORD@HOST:PORT/DATABASE');

//Using Pool Size to define the number of connections opening
//Also you can use a call back function for error handling
mongoose.connect('mongodb://localhost:27017/consumers',
  {server: { poolSize: 50 }},
  function(err) {
    if(err) {
      console.log('error in this')
      console.log(err);
      // Do whatever to handle the error
    } else {
      console.log('Connected to the database');
    }
  });
```

MongooseSchema。 ◦

```
var mongoose = require('mongoose');

var Schema = mongoose.Schema;

var AutoSchema = new Schema({
```

```

    name : String,
    countOf: Number,
  });
  // defining the document structure

  // by default the collection created in the db would be the first parameter we use (or the
  plural of it)
  module.exports = mongoose.model('Auto', AutoSchema);

  // we can over write it and define the collection name by specifying that in the third
  parameters.
  module.exports = mongoose.model('Auto', AutoSchema, 'collectionName');

  // We can also define methods in the models.
  AutoSchema.methods.speak = function () {
    var greeting = this.name
      ? "Hello this is " + this.name+ " and I have counts of "+ this.countOf
      : "I don't have a name";
    console.log(greeting);
  }
  mongoose.model('Auto', AutoSchema, 'collectionName');

```

mongoose.model◦

◦

```

var Auto = require('models/auto')
var autoObj = new Auto({
  name: "NewName",
  countOf: 10
});

```

```

autoObj.save(function(err, insertedAuto) {
  if (err) return console.error(err);
  insertedAuto.speak();
  // output: Hello this is NewName and I have counts of 10
});

```

◦ ◦

```

var Auto = require('models/auto')
Auto.find({}, function (err, autos) {
  if (err) return console.error(err);
  // will return a json array of all the documents in the collection
  console.log(autos);
})

```

```

Auto.find({countOf: {$gte: 5}}, function (err, autos) {
  if (err) return console.error(err);
  // will return a json array of all the documents in the collection whose count is
  greater than 5
  console.log(autos);
})

```

```
Auto.find({}, {name:1}, function (err, autos) {
  if (err) return console.error(err);
  // will return a json array of name field of all the documents in the collection
  console.log(autos);
})
```

◦

```
Auto.findOne({name:"newName"}, function (err, auto) {
  if (err) return console.error(err);
  //will return the first object of the document whose name is "newName"
  console.log(auto);
})
```

ID◦

```
Auto.findById(123, function (err, auto) {
  if (err) return console.error(err);
  //will return the first json object of the document whose id is 123
  console.log(auto);
})
```

-
- updateOne
- updateMany
- replaceOne

update

```
db.lights.update(
  { room: "Bedroom" },
  { status: "On" }
)
```

'lights'_{room}**Bedroom** ◦ status **On** WriteResult

```
{ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 }
```

UpdateOne

UpdateOne

```
db.countries.update(
  { country: "Sweden" },
  { capital: "Stockholm" }
)
```


"/ country ◦ capital **Stockholm WriteResult**

```
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

UpdateMany

UpdateMany

```
db.food.updateMany(  
  { sold: { $lt: 10 } },  
  { $set: { sold: 55 } }  
)
```

sold **55** sold **10** *"" ◦ WriteResult

```
{ "acknowledged" : true, "matchedCount" : a, "modifiedCount" : b }
```

a =

b =

ReplaceOne

countries3

```
{ "_id" : 1, "country" : "Sweden" }  
{ "_id" : 2, "country" : "Norway" }  
{ "_id" : 3, "country" : "Spain" }
```

```
{ country: "Spain" } { country: "Finland" }
```

```
db.countries.replaceOne(  
  { country: "Spain" },  
  { country: "Finland" }  
)
```

```
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

```
{ "_id" : 1, "country" : "Sweden" }  
{ "_id" : 2, "country" : "Norway" }  
{ "_id" : 3, "country" : "Finland" }
```

mongoose ◦

```
Auto.remove({_id:123}, function(err, result){
```

```
if (err) return console.error(err);
console.log(result); // this will specify the mongo default delete result.
});
```

Node.JSMongoDB。 <https://riptutorial.com/zh-TW/node-js/topic/7505/node-jsmongodb->

20: Node.js

Examples

```
let loop = (i, max) => {
  while (i < max) i++
  return i
}

// This operation will block Node.js
// Because, it's CPU-bound
// You should be careful about this kind of code
loop(0, 1e+12)
```

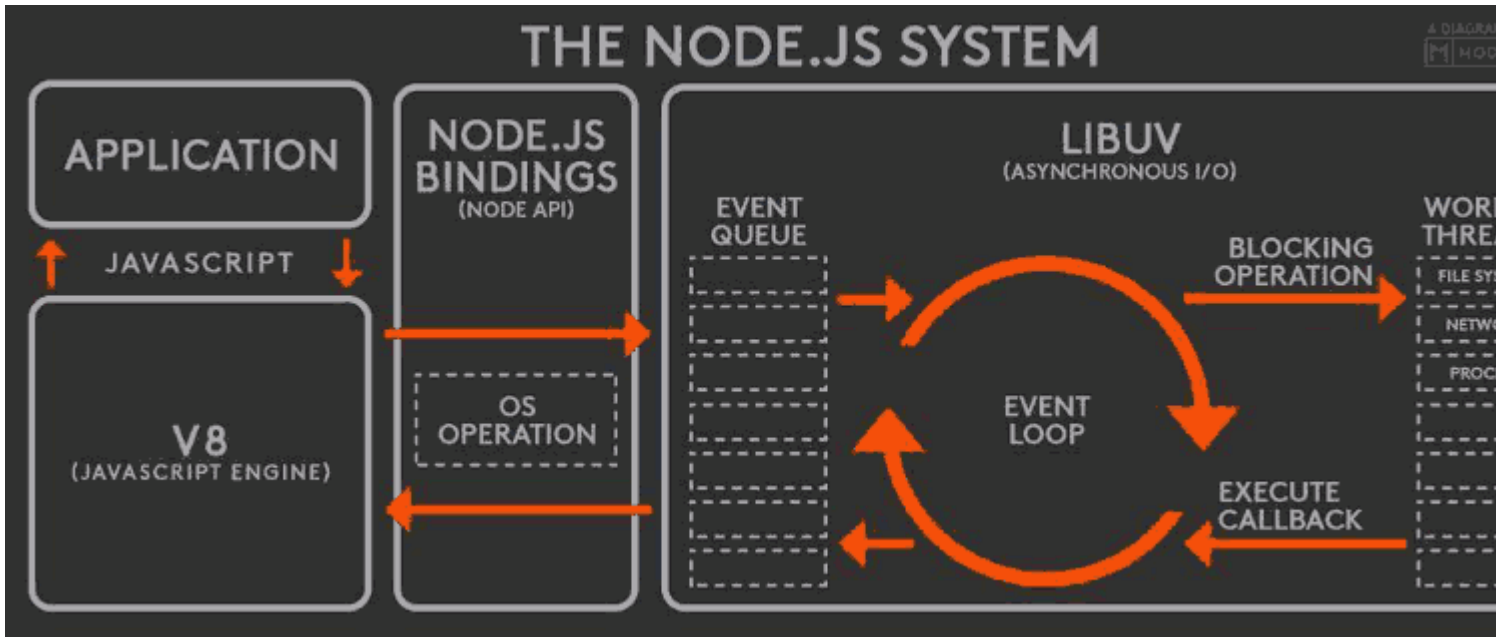
IO

```
let i = 0

const step = max => {
  while (i < max) i++
  console.log('i = %d', i)
}

const tick = max => process.nextTick(step, max)

// this will postpone tick run step's while-loop to event loop cycles
// any other IO-bound operation (like filesystem reading) can take place
// in parallel
tick(1e+6)
tick(1e+7)
console.log('this will output before all of tick operations. i = %d', i)
console.log('because tick operations will be postponed')
tick(1e+8)
```



Event Loop CPU I/O.

Node.js libuv.

- CPU Node.js CPU.

Node.js I/O.

maxSockets

```
require('http').globalAgent.maxSockets = 25

// You can change 25 to Infinity or to a different value by experimenting
```

Node.js maxSockets = Infinity v0.12.0. Node v0.12.0 maxSockets = 5 v0.11.0. 5.

http API " " . .

```
const http = require('http')
const myGloriousAgent = new http.Agent({ keepAlive: true })
myGloriousAgent.maxSockets = Infinity

http.request({ ..., agent: myGloriousAgent }, ...)
```

Socket Pooling

```
const http = require('http')
const options = {.....}

options.agent = false

const request = http.request(options)
```

- [https API](#)
- [AWS50Infinity](#) ◦

gzip

```
const http = require('http')
const fs = require('fs')
const zlib = require('zlib')

http.createServer((request, response) => {
  const stream = fs.createReadStream('index.html')
  const acceptsEncoding = request.headers['accept-encoding']

  let encoder = {
    hasEncoder : false,
    contentEncoding: {},
    createEncoder : () => throw 'There is no encoder'
  }

  if (!acceptsEncoding) {
    acceptsEncoding = ''
  }

  if (acceptsEncoding.match(/\bdeflate\b/)) {
    encoder = {
      hasEncoder : true,
      contentEncoding: { 'content-encoding': 'deflate' },
      createEncoder : zlib.createDeflate
    }
  } else if (acceptsEncoding.match(/\bgzip\b/)) {
    encoder = {
      hasEncoder : true,
      contentEncoding: { 'content-encoding': 'gzip' },
      createEncoder : zlib.createGzip
    }
  }

  response.writeHead(200, encoder.contentEncoding)

  if (encoder.hasEncoder) {
    stream = stream.pipe(encoder.createEncoder())
  }

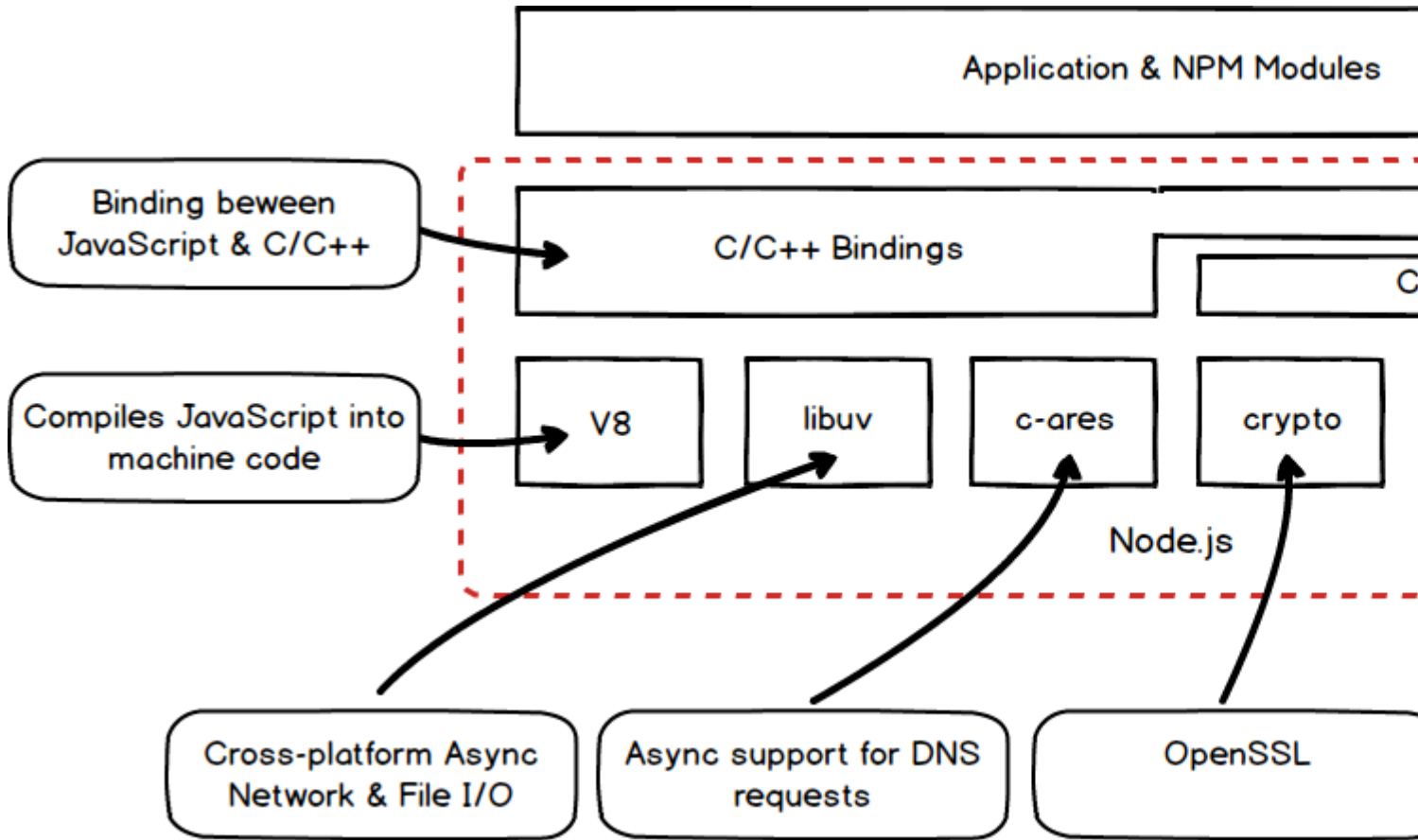
  stream.pipe(response)
}).listen(1337)
```

Node.js <https://riptutorial.com/zh-TW/node-js/topic/9410/node-js>

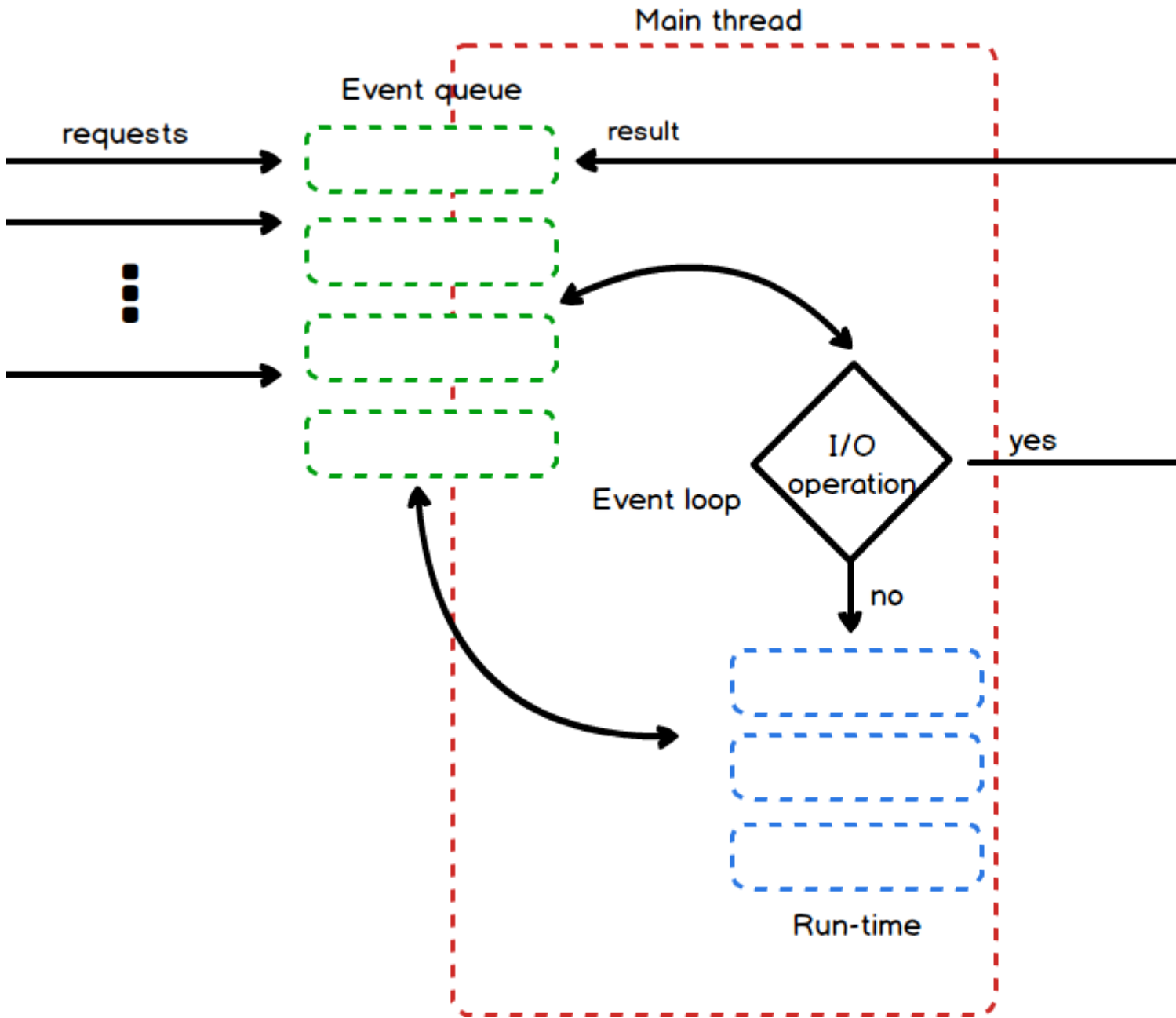
21: Node.js

Examples

Node.js -



Node.js -



Node.js <https://riptutorial.com/zh-TW/node-js/topic/5892/node-js>

22: Node.js

Examples

HTTP

```
const http = require('http');

console.log('Starting server...');
var config = {
  port: 80,
  contentType: 'application/json; charset=utf-8'
};
// JSON-API server on port 80

var server = http.createServer();
server.listen(config.port);
server.on('error', (err) => {
  if (err.code === 'EADDRINUSE') console.error('Port ' + config.port + ' is already in use');
  else console.error(err.message);
});
server.on('request', (request, res) => {
  var remoteAddress = request.headers['x-forwarded-for'] ||
  request.connection.remoteAddress; // Client address
  console.log(remoteAddress + ' ' + request.method + ' ' + request.url);

  var out = {};
  // Here you can change output according to `request.url`
  out.test = request.url;
  res.writeHead(200, {
    'Content-Type': config.contentType
  });
  res.end(JSON.stringify(out));
});
server.on('listening', () => {
  c.info('Server is available: http://localhost:' + config.port);
});
```

```
const process = require('process');
const rl = require('readline').createInterface(process.stdin, process.stdout);

rl.pause();
console.log('Something long is happening here...');

var cliConfig = {
  promptPrefix: ' > '
}

/*
  Commands recognition
  BEGIN
*/
var commands = {
  eval: function(arg) { // Try typing in console: eval 2 * 10 ^ 3 + 2 ^ 4
    arg = arg.join(' ');
```



```

        try { console.log(eval(arg)); }
        catch (e) { console.log(e); }
    },
    exit: function(arg) {
        process.exit();
    }
};
rl.on('line', (str) => {
    rl.pause();
    var arg = str.trim().match(/[^\s]+|"(?:[^\s\\]|\\.)*"/g); // Applying regular expression
    for removing all spaces except for what between double quotes:
    http://stackoverflow.com/a/14540319/2396907
    if (arg) {
        for (let n in arg) {
            arg[n] = arg[n].replace(/^\s|\s$/g, '');
        }
        var commandName = arg[0];
        var command = commands[commandName];
        if (command) {
            arg.shift();
            command(arg);
        }
        else console.log('Command "' + commandName + '" doesn\'t exist');
    }
    rl.prompt();
});
/*
    END OF
    Commands recognition
*/

rl.setPrompt(cliConfig.promptPrefix);
rl.prompt();

```

Node.js <https://riptutorial.com/zh-TW/node-js/topic/7703/node-js>

23: Node.js CORS

Examples

express.js CORS

node.js API CORS.

◦

express server.js

```
// Create express server
const app = express();

app.use((req, res, next) => {
  res.header('Access-Control-Allow-Origin', '*');

  // authorized headers for preflight requests
  // https://developer.mozilla.org/en-US/docs/Glossary/preflight_request
  res.header('Access-Control-Allow-Headers', 'Origin, X-Requested-With, Content-Type,
Accept');
  next();

  app.options('*', (req, res) => {
    // allowed XHR methods
    res.header('Access-Control-Allow-Methods', 'GET, PATCH, PUT, POST, DELETE, OPTIONS');
    res.send();
  });
});
```

◦ Apache Nginx CORS.

node.js CORS.

NODE_ENV

```
const app = express();

if (process.env.NODE_ENV === 'development') {
  // CORS settings
}
```

Node.js CORS <https://riptutorial.com/zh-TW/node-js/topic/9272/node-jscors>

24: Node.JSES6

ES6ECMAScript 6ES2015JavaScript ◦

NodeES6<https://nodejs.org/en/docs/es6/>

Examples

Node ES6Babel

ES6◦ <http://node.green/ES6>

NodeJS v6◦ NodeJS v6ES6◦ ◦

ES6◦ JavaScriptBabel

BabelES6's'tage-0'`import thing from 'thing'var thing = require('thing')` import thing from 'thing'
'stage-0'`import Babel◦ reactVuecommonJS0◦`

```
mkdir my-es6-app
cd my-es6-app
npm init
```

babel ES6stage-0

```
npm install --save-dev babel-preset-es2015 babel-preset-stage-2 babel-cli babel-register
```

server.jsHTTP◦

```
import http from 'http'

http.createServer((req, res) => {
  res.writeHead(200, {'Content-Type': 'text/plain'})
  res.end('Hello World\n')
}).listen(3000, '127.0.0.1')

console.log('Server running at http://127.0.0.1:3000/')
```

```
import http from 'http'import http from 'http'0◦
```

node server.js ◦

.babelrc

```
{
  "presets": ["es2015", "stage-2"],
  "plugins": []
}
```

```
node src/index.js --exec babel-node
```

- **package.json**◦

```
"scripts": {
  "start": "node dist/index.js",
  "dev": "babel-node src/index.js",
  "build": "babel src -d dist",
  "postinstall": "npm run build"
},
```

```
npm install dist npm start◦
```

```
npm run dev babel◦
```

```
nodemon npm install nodemon --save-dev◦
```

```
babelNodeJS◦ package.json“dev”nodemon
```

```
"dev": "nodemon src/index.js --exec babel-node",
```

NodeJSJS es6

JS es6es2015JSOOP◦

1. <http://es6-features.org>es6 - NodeJS
2. <http://node.green>
3. -

JS es6hello world

```
'use strict'

class Program
{
  constructor()
  {
    this.message = 'hello es6 :)';
  }

  print()
  {
    setTimeout(() =>
    {
      console.log(this.message);

      this.print();

    }, Math.random() * 1000);
  }
}
```

```
new Program().print();
```

◦
..

```
'use strict'
```

js es6◦ strictMDN - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Strict_mode

```
class Program
```

-class - es6js... function

```
function MyClass() // class definition
{
}

var myClassObject = new MyClass(); // generating a new object with a type of MyClass
```

OOP.....

```
constructor()
{
  this.message = 'hello es6 :)';
}
```

- - " -

```
print()
{
  setTimeout(() => // this is an 'arrow' function
  {
    console.log(this.message);

    this.print(); // here we call the 'print' method from the class template itself (a
recursion in this particular case)

  }, Math.random() * 1000);
}
```

print - -

.....

```
new Program().print();
```

```
var prog = new Program(); // define a new object of type 'Program'
```

```
prog.print(); // use the program to print itself
```

JS es6 - JS..

Node.JSES6 <https://riptutorial.com/zh-TW/node-js/topic/5934/node-jses6>

25: Node.js Oracle

Examples

Oracle DB

ORACLE [oracledb](#) Node.js Oracle.

```
npm install oracledb
```

ORACLE.

```
const oracledb = require('oracledb');

oracledb.getConnection(
  {
    user      : "oli",
    password  : "password",
    connectString : "ORACLE_DEV_DB_TNS_NAME"
  },
  connExecute
);
```

connectString“ORACLE_DEV_DB_TNA_NAME”tnsnames.orgoracle.

oracle [instant client installation guide](#).

connExecute-Function. ◦ console.log.

```
function connExecute(err, connection)
{
  if (err) {
    console.error(err.message);
    return;
  }
  sql = "select 'test' as c1, 'oracle' as c2 from dual";
  connection.execute(sql, {}, { outFormat: oracledb.OBJECT }, // or oracledb.ARRAY
    function(err, result)
    {
      if (err) {
        console.error(err.message);
        connRelease(connection);
        return;
      }
      console.log(result.metaData);
      console.log(result.rows);
      connRelease(connection);
    });
}
```

◦

```
function connRelease(connection)
{
  connection.close(
    function(err) {
      if (err) {
        console.error(err.message);
      }
    });
}
```

```
[ { name: 'C1' }, { name: 'C2' } ]
[ { C1: 'test', C2: 'oracle' } ]
```

```
[ { name: 'C1' }, { name: 'C2' } ]
[ [ 'test', 'oracle' ] ]
```

ORACLE-DB

```
const oracle = require('./oracle.js');

const sql = "select 'test' as c1, 'oracle' as c2 from dual";
oracle.queryObject(sql, {}, {})
  .then(function(result) {
    console.log(result.rows[0]['C2']);
  })
  .catch(function(err) {
    next(err);
  });
```

oracle.js

```
'use strict';
const oracledb = require('oracledb');

const oracleDbRelease = function(conn) {
  conn.release(function (err) {
    if (err)
      console.log(err.message);
  });
};

function queryArray(sql, bindParams, options) {
  options.isAutoCommit = false; // we only do SELECTs

  return new Promise(function(resolve, reject) {
    oracledb.getConnection(
      {
        user          : "oli",
        password      : "password",
        connectString : "ORACLE_DEV_DB_TNA_NAME"
      })
    .then(function(connection) {
      //console.log("sql log: " + sql + " params " + bindParams);
      connection.execute(sql, bindParams, options)
        .then(function(results) {
          resolve(results);
        });
    });
  });
}
```



```

        process.nextTick(function() {
            oracleDbRelease(connection);
        });
    })
    .catch(function(err) {
        reject(err);

        process.nextTick(function() {
            oracleDbRelease(connection);
        });
    });
    .catch(function(err) {
        reject(err);
    });
});
}

function queryObject(sql, bindParams, options) {
    options['outFormat'] = oracledb.OBJECT; // default is oracledb.ARRAY
    return queryArray(sql, bindParams, options);
}

module.exports = queryArray;
module.exports.queryArray = queryArray;
module.exports.queryObject = queryObject;

```

oraclequeryArrayqueryObject。

Node.jsOracle <https://riptutorial.com/zh-TW/node-js/topic/8248/node-jsoracle>

26: Node.js

Examples

Node.js

-

```
a - "Small is beautiful"
b - "Make each program do one thing well."
```

Reactor Pattern [node.js](#) ◦ ◦

Node.js / O - libuv -

EventEmitter/

```
var events = require('events');
var EventEmitter = new events.EventEmitter();

var ringBell = function ringBell()
{
  console.log('tring tring tring');
}
eventEmitter.on('doorOpen', ringBell);

eventEmitter.emit('doorOpen');
```

[Node.js](https://riptutorial.com/zh-TW/node-js/topic/6274/node-js) <https://riptutorial.com/zh-TW/node-js/topic/6274/node-js>

27: Node.js

ErrorNode.js

◦

Examples

message message ◦ message **Error** ◦ message **object** **Error**.toString() message ◦

```
var err = new Error("The error message");
console.log(err.message); //prints: The error message
console.log(err);
//output
//Error: The error message
//    at ...
```

◦ ◦ ◦ ◦

```
console.log(err);
console.log(err.stack);
```

◦

```
throw new Error("Some error occurred");
```

```
var err = new Error("Some error occurred");
throw err;
```

```
throw "Some error occurred";
```

Error

throw

```
var a = 5;
var err = new Error("Some error message");
throw err; //this will print the error stack and node server will stop
a++; //this line will never be executed
console.log(a); //and this one also
```

```
var a = 5;
var err = new Error("Some error message");
console.log(err); //this will print the error stack
a++;
console.log(a); //this line will be executed and will print 6
```

.....

try ... catch

```
try {
  var a = 1;
  b++; //this will cause an error because b is undefined
  console.log(b); //this line will not be executed
} catch (error) {
  console.log(error); //here we handle the error caused in the try block
}
```

try b++catch

```
try {
  var a = 1;
  b++;
  console.log(b);
} catch (error) {
  error.message = "b variable is undefined, so the undefined can't be incremented"
  throw error;
}
```

error.message

trycatch

```
try {
  var a = 1;
  throw new Error("Some error message");
  console.log(a); //this line will not be executed;
} catch (error) {
  console.log(error); //will be the above thrown error
}
```

Node.js <https://riptutorial.com/zh-TW/node-js/topic/8590/node-js>

28: nodejs

Examples

◦

auto

```
var async = require('async');

async.auto({
  get_data: function(callback) {
    console.log('in get_data');
    // async code to get some data
    callback(null, 'data', 'converted to array');
  },
  make_folder: function(callback) {
    console.log('in make_folder');
    // async code to create a directory to store a file in
    // this is run at the same time as getting the data
    callback(null, 'folder');
  },
  write_file: ['get_data', 'make_folder', function(results, callback) {
    console.log('in write_file', JSON.stringify(results));
    // once there is some data and the directory exists,
    // write the data to a file in the directory
    callback(null, 'filename');
  }],
  email_link: ['write_file', function(results, callback) {
    console.log('in email_link', JSON.stringify(results));
    // once the file is written let's email a link to it...
    // results.write_file contains the filename returned by write_file.
    callback(null, {'file':results.write_file, 'email':'user@example.com'});
  }],
}, function(err, results) {
  console.log('err = ', err);
  console.log('results = ', results);
});
```

get_data make_folder write_fileemail_link◦ Async callbacknull◦

nodejs <https://riptutorial.com/zh-TW/node-js/topic/8287/nodejs>

29: NodeJS

Examples

helloworld.js

```
console.log("Hello World");
```

Node.js

```
node helloworld.js
```

NodeJS <https://riptutorial.com/zh-TW/node-js/topic/7693/nodejs>

30: NodeJS

Examples

Web

```
var express = require('express');
var app = express();

app.get('/', function (req, res) {
  res.send('Hello World!');
});

app.listen(3000, function () {
  console.log('Example app listening on port 3000!');
});
```

```
var koa = require('koa');
var app = koa();

app.use(function *(next) {
  var start = new Date;
  yield next;
  var ms = new Date - start;
  console.log('%s %s - %s', this.method, this.url, ms);
});

app.use(function *(){
  this.body = 'Hello World';
});

app.listen(3000);
```

Commander.js

```
var program = require('commander');

program
  .version('0.0.1')

program
  .command('hi')
  .description('initialize project configuration')
  .action(function () {
    console.log('Hi my Friend!!!');
  });

program
  .command('bye [name]')
  .description('initialize project configuration')
  .action(function (name) {
    console.log('Bye ' + name + '. It was good to see you!');
  });
```

```
program
  .command('*')
  .action(function(env) {
    console.log('Enter a Valid command');
    terminate(true);
  });

program.parse(process.argv);
```

Vorpal.js

```
const vorpal = require('vorpal')();

vorpal
  .command('foo', 'Outputs "bar".')
  .action(function(args, callback) {
    this.log('bar');
    callback();
  });

vorpal
  .delimiter('myapp$')
  .show();
```

NodeJS <https://riptutorial.com/zh-TW/node-js/topic/6042/nodejs>

31: Nodejs

Node.js。

Examples

2009

- 33 “”
- 101 [npm](#)
- 118 [Ryan DahlNode.jsNode.jsJSConf 2009](#)

2010

- [ExpressNode.js Web](#)
- [Socket.io](#)
- 428 [HerokuNode.js](#)
- 728 [Ryan DahlNode.jsGoogle Tech Talk](#)
- 820 [Node.js 0.2.0](#)

2011

- 331Node.js
- 51 [npm 1.0](#)
- 51 [Ryan DahlRedditAMA](#)
- 710 [Node.js](#) ◦
 - Node.js。
- 816 [LinkedInNode.js](#)
 - LinkedIn。
- 105 [Ryan DahlNode.js](#)
- 125 [Node.js](#)
 - [Curtis ChambersNode.js](#)。

2012

- 130 [Node.jsRyan DahlNode](#)
- 625 [Node.js v0.8.0 \[\]](#)
- 1220Node.jsHapi

2013

- 430 [MEAN StackMongoDBExpressJSAngularJSNode.js](#)
- 517 [eBayNode.js](#)
- 1115 [PayPalNode.jsKraken](#)
- 1122 [Node.js](#)
 - [Wal-MartEran HammerNode.js](#)
- 1219 [Koa - Node.jsWeb](#)

2014

- 115 [TJ FontaineNode](#)
- 1023 [Node.js](#)
 - [JoyentNode.jsNode.jsNode.js](#)
- 1119 [Flame GraphsNode.js - Netflix](#)
- 1128 [IO.js - V8 JavascriptEvented I / O.](#)

2015

Q1

- 114 [IO.js 1.0.0](#)
- 10Febrary [JoyentNode.js](#)
 - [JoyentIBMPayPalSAPLinuxNode.js](#)
- 27Febrary [IO.jsNode.js](#)

Q2

- 414 [npm Private Modules](#)
- 528 [TJ FontaineJoyent](#)
- 513 [Node.jsio.jsNode Foundation](#)

Q3

- 82 [Trace - Node.js](#)
 - [Trace](#)
- 813 [4.01.0](#)

Q4

- 1012 [Node v4.2.0](#)

- 128 [ApigeeRisingStackNode.js](#)
- 1289 [Node Interactive](#)
 - [Node.jsNode.js](#)

2016

Q1

- 210 [Express](#)
- 323
- 329 [Google Cloud PlatformNode.js](#)

Q2

- 426 [npm210,000](#)

Q3

- 718 [CJ Silverionpm](#)
- 81 [TraceNode.js](#)
- 915 [Node Interactive](#)

Q4

- 1011
- 1018 [Node.js 6LTS](#)

1. “Node.js”[]◦ [<https://blog.risingstack.com/history-of-node-js>]

[Nodejs https://riptutorial.com/zh-TW/node-js/topic/8653/nodejs](https://riptutorial.com/zh-TW/node-js/topic/8653/nodejs)

32: NodeJSRedis

node_redis RedisNode.js Pub / Sub

Examples

node_redis Node.jsRedis npm

```
npm install redis
```

node_redis app.js Node.jsRedis

app.js

```
var redis = require('redis');  
client = redis.createClient(); //creates a new client
```

redis.createClient(127.0.0.16379) /

```
var client = redis.createClient(port, host);
```

◦ ◦

```
client.on('connect', function() {  
    console.log('connected');  
});
```

app.js

```
var redis = require('redis');  
var client = redis.createClient();  
  
client.on('connect', function() {  
    console.log('connected');  
});
```

node app Redis

Node.jsRedisRedis

Redis

```
client.set('framework', 'AngularJS');
```

```
client.set(['framework', 'AngularJS']);
```

AngularJS ◦ ◦ args client.set() ◦

```
client.set('framework', 'AngularJS', function(err, reply) {
  console.log(reply);
});
```

err ◦

```
client.get('framework', function(err, reply) {
  console.log(reply);
});
```

client.get() **Redis** ◦ ◦ ◦

◦ **Redis** ◦ hmset()

```
client.hmset('frameworks', 'javascript', 'AngularJS', 'css', 'Bootstrap', 'node', 'Express');

client.hgetall('frameworks', function(err, object) {
  console.log(object);
});
```

Redis ◦ hmset() ◦ ◦ hgetall() ◦ ◦

Redis ◦ ◦ **Redis**

```
client.hmset('frameworks', {
  'javascript': 'AngularJS',
  'css': 'Bootstrap',
  'node': 'Express'
});
```

◦

/◦ client.hmset() client.HMSET() ◦

Redis ◦

```
client.rpush(['frameworks', 'angularjs', 'backbone'], function(err, reply) {
  console.log(reply); //prints 2
});
```

frameworks ◦ ◦ args rpush ◦ ◦ lpush() rpush() ◦

lrange()

```
client.lrange('frameworks', 0, -1, function(err, reply) {
  console.log(reply); // ['angularjs', 'backbone']
});
```

-1 lrange() ◦ ◦

◦ ◦ ◦

```
client.sadd(['tags', 'angularjs', 'backbonejs', 'emberjs'], function(err, reply) {
  console.log(reply); // 3
});
```

sadd() ◦ ◦ smembers()

```
client.smembers('tags', function(err, reply) {
  console.log(reply);
});
```

◦ ◦

Redis ◦ RedishyperLogLog ◦ Redis ◦ Redisnode_redis ◦

node_redis ◦

◦ exists()

```
client.exists('key', function(err, reply) {
  if (reply === 1) {
    console.log('exists');
  } else {
    console.log('doesn\'t exist');
  }
});
```

◦ **del**

```
client.del('frameworks', function(err, reply) {
  console.log(reply);
});
```

```
client.set('key1', 'vall');
client.expire('key1', 30);
```

key130 ◦

Redis ◦ incr()

```
client.set('key1', 10, function() {
  client.incr('key1', function(err, reply) {
    console.log(reply); // 11
  });
});
```

incr() **1**.incrby() ◦ decr()decrby() ◦

NodeJSRedis <https://riptutorial.com/zh-TW/node-js/topic/7107/nodejsredis>

33: NodeJs

jsExploring the ExpressExpress Web。

Express Router。

Examples

Express Web

Express Web

Express。。

Express Server。 NPM。

1. Projectpackage.json。 **package.json** {"name" "expressRouter" "version" "0.0.1" "scripts" {"start" "node Server.js"} "dependencies" {"express" "^ 4.12.3"}}

2. *npm install* express。 node_modules。

3. Express Web Server。 Projectserver.js。 **server.js**

```
var express = require("express"); var app = express;
```

```
//Router
```

```
var router = express.Router;
```

```
//。
```

```
router.get("/", function(req, res) {  
  res.json({"message" : "Hello World"});
```

```
};
```

```
app.use("/ API");
```

```
//
```

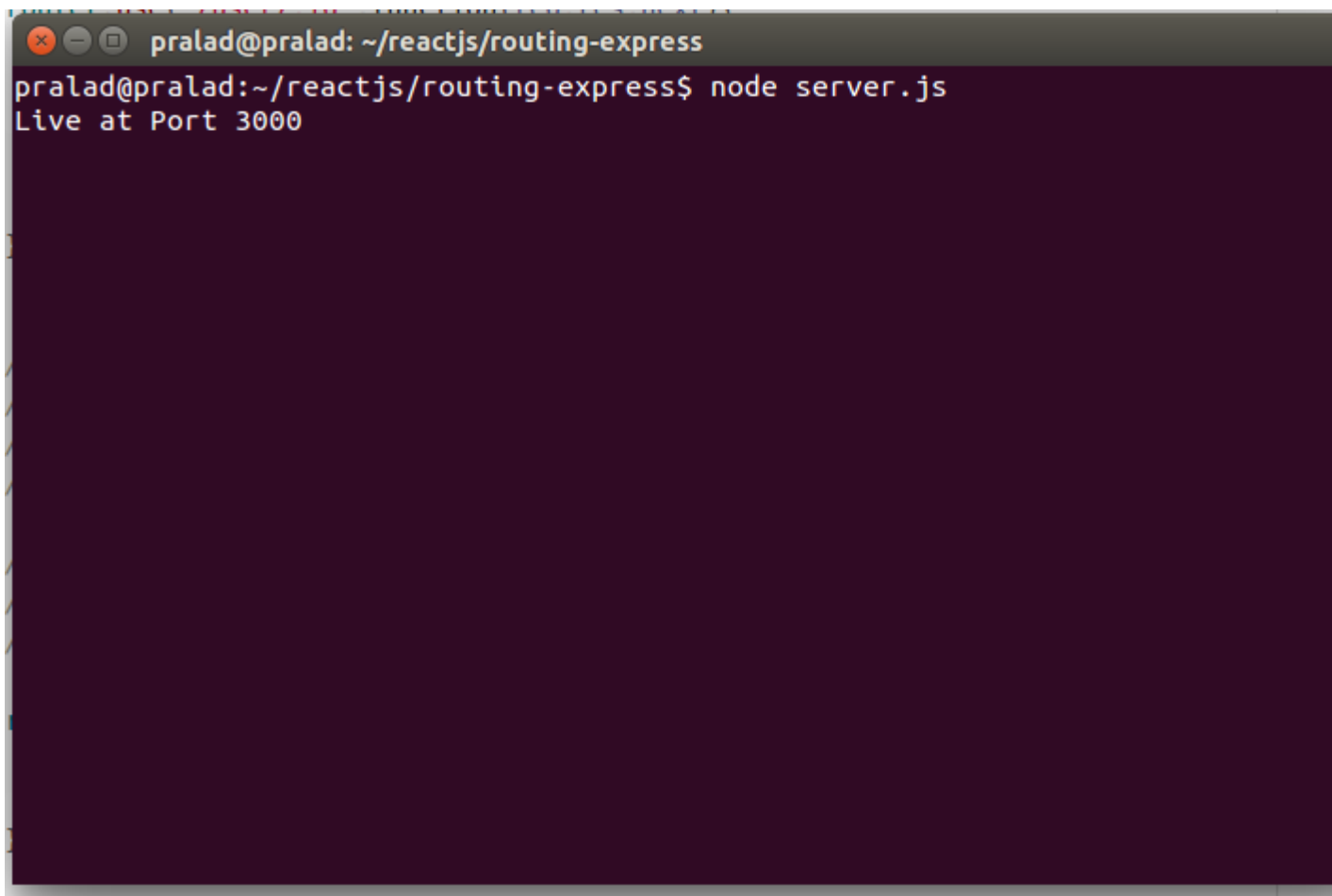
```
app.listen3000function{console.log"Live at Port 3000";};
```

For more detail on setting node server you can see [\[here\]](#)[1].

4.。

```
node server.js
```

Server.

A terminal window with a dark purple background. The title bar reads "pralad@pralad: ~/reactjs/routing-express". The terminal content shows the command "node server.js" being executed, followed by the output "Live at Port 3000".

```
pralad@pralad: ~/reactjs/routing-express
pralad@pralad:~/reactjs/routing-express$ node server.js
Live at Port 3000
```

o

5. [HTTP//3000 / API /](http://localhost:3000/api/)

A browser window showing the response of an API call. The address bar contains "localhost:3000/api/". The page content displays a JSON object: {"message": "Hello World"}.

```
{ "message": "Hello World" }
```

o

Express。

GETPOST。

server.js

```
var express = require("express");
var app = express();

//Creating Router() object

var router = express.Router();

// Router middleware, mentioned it before defining routes.

router.use(function(req,res,next) {
  console.log("/") + req.method);
  next();
});

// Provide all routes here, this is for Home page.

router.get("/",function(req,res){
  res.json({"message" : "Hello World"});
});

app.use("/api",router);

app.listen(3000,function(){
  console.log("Live at Port 3000");
});
```

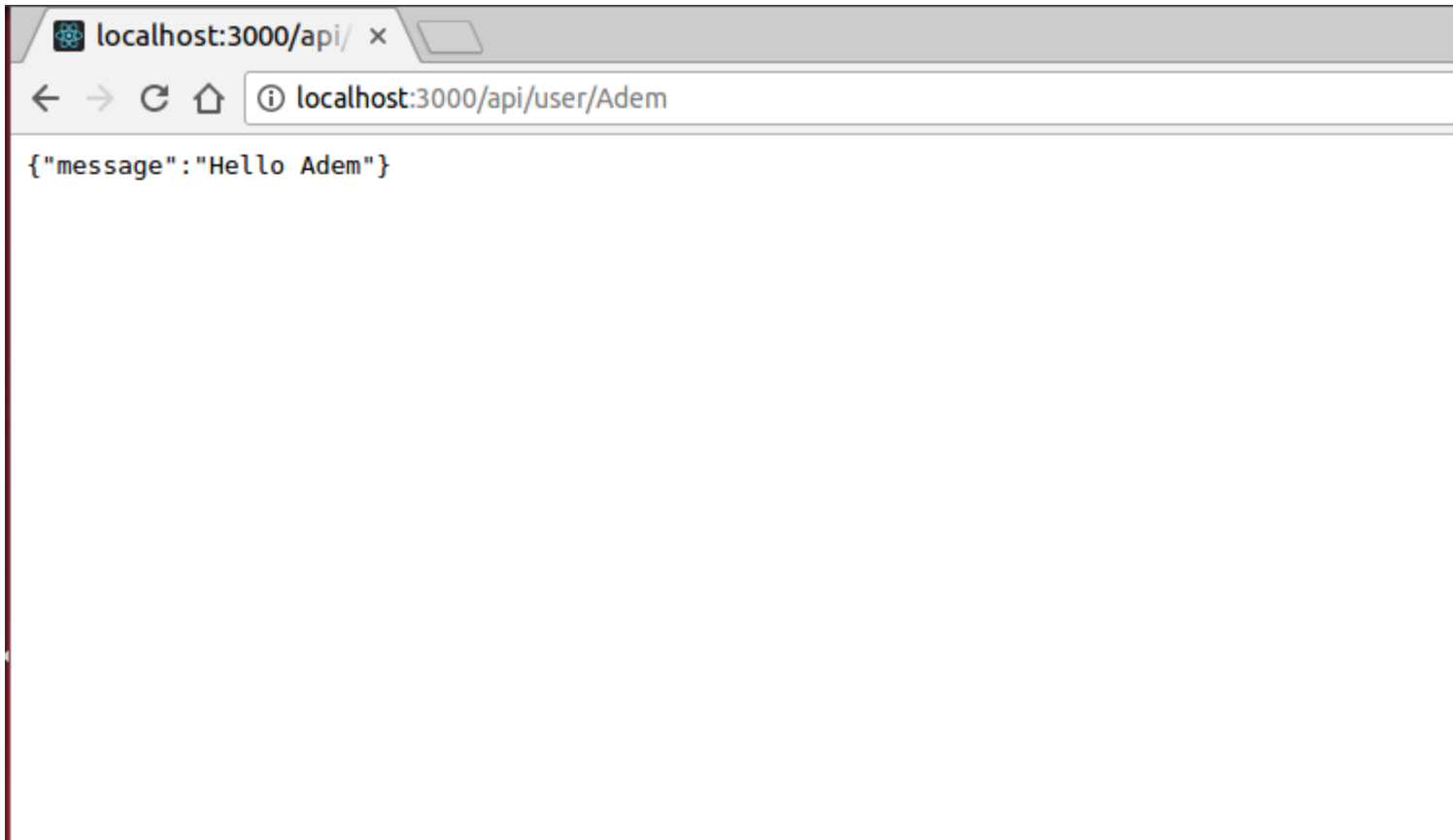
```
http://localhost:3000/api/
```

```
pralad@pralad: ~/reactjs/routing-express
pralad@pralad:~/reactjs/routing-express$ node server.js
Live at Port 3000
/GET
```

url <http://example.com/api/:name/> . name . server.js

```
router.get("/user/:id", function(req, res) {
  res.json({"message" : "Hello "+req.params.id});
});
```

[<http://localhost3000/api/user/Adem>] [4]



◦
[NodeJs https://riptutorial.com/zh-TW/node-js/topic/9846/nodejs](https://riptutorial.com/zh-TW/node-js/topic/9846/nodejs)

34: NPM

npmsearch.nodejs.orgnode.js/◦ Node.jsNode.js◦

- npm <command><command>

-

-

- apihelp

-

-

-

- C

-

-

-

- DDP

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

- 6.

-

-

-

-

-

- LN

-

- LS

-

-

-

-

-

-

-

- [R

- RB

-

-

-

- o
- o R M
- o
- o
- o
- o SE
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o TST
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o
- o V
- o
- o
- o
- o

	npm publish --access=public
	npm bin -g
	npm edit connect
	npm help init
	npm init
	npm install
	npm link
	npm prune
	npm publish ./
	npm restart
	npm start
	npm start
	npm update
	npm version

Examples

npm jQueryAngularJSGulp.js node_modulespackage.json

Npmnpm

NPM

NPM

NPMNode.js npm -v npm version npm version

Node.jsNPM NPMNode.js

```
npm install npm@latest -g
```

```
npm install <package-name>
# or
npm i <package-name>...

# e.g. to install lodash and express
npm install lodash express
```

package.json npm install npm i

```
npm install <name>@<version>

# e.g. to install version 4.11.1 of the package lodash
npm install lodash@4.11.1
```

```
npm install <name>@<version range>

# e.g. to install a version which matches "version >= 4.10.1" and "version < 4.11.1"
# of the package lodash
npm install lodash@">=4.10.1 <4.11.1"
```

```
npm install <name>@latest
```

npmjs.com npm npm

```
# packages distributed as a tarball
npm install <tarball file>
npm install <tarball url>
```

```
# packages available locally
npm install <local path>

# packages available as a git repository
npm install <git remote url>

# packages available on GitHub
npm install <username>/<repository>

# packages available as gist (need a package.json)
npm install gist:<gist-id>

# packages from a specific repository
npm install --registry=http://myreg.mycompany.com <package name>

# packages from a related group of packages
# See npm scope
npm install @<scope>/<name>(@<version>)

# Scoping is useful for separating private packages hosted on private registry from
# public ones by setting registry for specific scope
npm config set @mycompany:registry http://myreg.mycompany.com
npm install @mycompany/<package name>
```

node_modules◦ require()◦

package.json--save -S package.json◦ npmpackage.json◦

```
npm install --save <name> # Install dependencies
# or
npm install -S <name> # shortcut version --save
# or
npm i -S <name>
```

```
npm install --save-dev <name> # Install dependencies for development purposes
# or
npm install -D <name> # shortcut version --save-dev
# or
npm i -D <name>
```

◦ ◦ npm/usr/local/bin/<name> ◦

```
npm install --global <name>
# or
npm install -g <name>
# or
npm i -g <name>

# e.g. to install the grunt command line tool
npm install -g grunt-cli
```

```
npm list
npm list <name>
```

name。

npm sudo npm install -g ...。 npm Nodenvm。

Grunt。 devDependenciespackage.json devDependencies。 --save-dev -D。

```
npm install --save-dev <name> // Install development dependencies which is not included in
production
# or
npm install -D <name>
```

package.jsondevDependencies。

/node.js

```
npm install
# or
npm i
```

npmpackage.json。

NPM

Internetnpm install。 npm

```
npm config set
```

。 URL;npm。

```
$ npm config set proxy http://<username>:<password>@<proxy-server-url>:<port>
$ npm config set https-proxy http://<username>:<password>@<proxy-server-url>:<port>
```

username password port。 npm install npm i -g。

```
# Set the repository for the scope "myscope"
npm config set @myscope:registry http://registry.corporation.com

# Login at a repository and associate it with the scope "myscope"
npm adduser --registry=http://registry.corporation.com --scope=@myscope

# Install a package "mylib" from the scope "myscope"
npm install @myscope/mylib
```

@myscope“myscope”npm publish。

.npmrc

```
@myscope:registry=http://registry.corporation.com
```



```
//registry.corporation.com/:_authToken=xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx
```

Clfe

```
npm uninstall <package name>
```

npmuninstall

```
npm remove <package name>  
npm rm <package name>  
npm r <package name>
```

```
npm unlink <package name>  
npm un <package name>
```

package.json--save -S

```
npm uninstall --save <package name>  
npm uninstall -S <package name>
```

--save-dev-D

```
npm uninstall --save-dev <package name>  
npm uninstall -D <package name>
```

--save-optional -O

```
npm uninstall --save-optional <package name>  
npm uninstall -O <package name>
```

--global -g

```
npm uninstall -g <package name>
```

◦ npm ◦

1.2.3

1. npm version patch => 1.2.4
2. npm version minor => 1.3.0
3. npm version major => 2.0.0

npm version 3.1.4 => 3.1.4

npmnpmpackage.json“v”Git

git tag v3.1.4

BowernpmGit◦

```
git push origin master package.json
```

```
git push origin v3.1.4
```

```
git push origin master --tags
```

Node.js package.json

```
npm init
```

Git ◦

package.json

```
npm init --yes  
# or  
npm init -y
```

npm package.json package.json

1. private **true** ◦
2. license **"UNLICENSED"** ◦

package.json

```
npm install --save <package>
```

◦ **if** --save-dev devDependencies ◦

package.json ◦ **package.json** JSON

```
[...]  
"description": "No description",  
"repository": {  
  "private": true  
},  
[...]
```

◦ **npmjs** ◦

npm

```
npm login
```

```
npm adduser
```

```
npm config ls
```

```
npm publish
```

-
- npm◦

```
{
  name: "package-name",
  version: "1.0.4"
}
```

package.json

```
{
  "name": "your-package",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "author": "",
  "license": "ISC",
  "dependencies": {},
  "devDependencies": {},
  "scripts": {
    "echo": "echo hello!"
  }
}
```

echo npm run echo ◦ npm run <script name>echo above◦ npm preinstall ◦ npm ◦

npm◦

```
"scripts": {
  "test": "mocha tests",
  "start": "pm2 start index.js"
}
```

scripts mocha◦ PATH npm◦

```
"scripts": {
  "very-complex-command": "npm run chain-1 && npm run chain-2",
  "chain-1": "webpack",
  "chain-2": "node app.js"
}
```

npm prune

dev --production

npm prune --production

npm list

ls lallist. lall◦

◦

```
npm list --json
```

- **json** - json
- **long** -
- **parseable** -
- **global** -
- **depth** -
- **dev / development** - devDependencies
- **prod / production** -

◦

```
npm home <package name>
```

npm

npmNode.js◦

Windows

```
npm install -g npm@latest
```

```
npm outdated
```

```
npm update <package name>
```

package.json

package.json

```
npm update <package name> --save
```

npm ◦ semver◦

node_modules

```
npm shrinkwrap
```

package.jsonnpm-shrinkwrap.json **dependencies**◦

npm install -g“”◦ ◦

```
npm install -g gulp-cli
```

gulp

◦

```
npm install -g/usr/bin ◦ sudo npm installsudo◦
```

```
~/.npmrcnpm◦ prefix npm prefix◦
```

```
prefix=~/.npm-global-modules
```

```
npm install -g◦ npm install --prefix ~/.npm-global-modules◦ -g◦
```

```
export PATH=$PATH:~/.npm-global-modules/bin
```

```
npm install -g gulp-cligulp ◦
```

```
npm install -g package.json◦ node_modules/.bin◦ npm install -g◦ node_modules/.bin  
node_modules/.bin◦
```

◦ **NPM**npm link ◦ npm link◦ ◦

```
NAME  
    npm-link - Symlink a package folder  
  
SYNOPSIS  
    npm link (in package dir)  
    npm link [<@scope>/]<pkg>[@<version>]  
  
    alias: npm ln
```

◦

1. **CD** cd ../my-dep
2. npm link
3. **CD**
4. npm link my-depnpm link @namespace/my-dep

1. **CD** cd eslint-watch
2. npm link
- 3.
4. esw --quiet

◦ npm uninstall (-g) <pkg>npm link◦

NPM <https://riptutorial.com/zh-TW/node-js/topic/482/npm>

35: nvm -

URLNode Version Manager。 ◦ nvm GitHubnvmURL。

Examples

NVM

curl

```
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.31.3/install.sh | bash
```

wget

```
wget -qO- https://raw.githubusercontent.com/creationix/nvm/v0.31.3/install.sh | bash
```

NVM

nvm

```
command -v nvm
```

'nvm'。

Node

```
nvm ls-remote
```

```
nvm install <version>
```

```
nvm install 0.10.13
```

NVM

```
nvm ls
```

nvm ls

```
$ nvm ls
  v4.3.0
  v5.5.0
```

v5.5.0

```
nvm use v5.5.0
```

Mac OSXnvm

gitcurlwgetNode Version Manager. **Mac OSX**.

```
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.31.3/install.sh | bash
```

wget

```
wget -qO- https://raw.githubusercontent.com/creationix/nvm/v0.31.3/install.sh | bash
```

NVM

nvmnvm ◦ **nvmcommand not found.bash_profile** ◦ touch ~/.bash_profile◦

nvmcommand

- nano .bashrc ◦

```
export NVM_DIR="/Users/johndoe/.nvm"[ - "$NVM_DIR/nvm.sh" ] && "$NVM_DIR/nvm.sh"
```

- **.bashrc**
- .bashrcCTRL + O - - CTRL + X
- nano .bash_profileBash
- Bash
- BashCTRL + O - - CTRL + X
- nano .bashrc.**bashrc**
-

```
source~/.nvm/nvm.sh
```

- CTRL + O - - CTRL + X
- nvm

```
nvm alias <name> <version>
```

unalias

```
nvm unalias <name>
```

- default◦

```
nvm alias default 5.0.1
```

/5.0.1◦

```
nvm alias # lists all aliases created on nvm
```

shell

```
nvm ls  
v4.5.0  
v6.7.0
```

```
nvm run 4.5.0 --version or nvm exec 4.5.0 node --version  
Running node v4.5.0 (npm v2.15.9)  
v4.5.0
```

```
nvm run 6.7.0 --version or nvm exec 6.7.0 node --version  
Running node v6.7.0 (npm v3.10.3)  
v6.7.0
```

```
nvm run default --version or nvm exec default node --version  
Running node v6.7.0 (npm v3.10.3)  
v6.7.0
```

LTS

```
nvm install --lts
```

```
nvm use v4.5.0 or nvm use stable ( alias )
```

nvm - <https://riptutorial.com/zh-TW/node-js/topic/2823/nvm---->

36: N-API

N-API NodeJS。 N-API。

Examples

N-API

hellohello。 helloprintfHello world1373javascript。

```
#include <node_api.h>
#include <stdio.h>

napi_value say_hello(napi_env env, napi_callback_info info)
{
    napi_value retval;

    printf("Hello world\n");

    napi_create_number(env, 1373, &retval);

    return retval;
}

void init(napi_env env, napi_value exports, napi_value module, void* priv)
{
    napi_status status;
    napi_property_descriptor desc = {
        /*
         * String describing the key for the property, encoded as UTF8.
         */
        .utf8name = "hello",
        /*
         * Set this to make the property descriptor object's value property
         * to be a JavaScript function represented by method.
         * If this is passed in, set value, getter and setter to NULL (since these members
won't be used).
         */
        .method = say_hello,
        /*
         * A function to call when a get access of the property is performed.
         * If this is passed in, set value and method to NULL (since these members won't be
used).
         * The given function is called implicitly by the runtime when the property is
accessed
         * from JavaScript code (or if a get on the property is performed using a N-API call).
         */
        .getter = NULL,
        /*
         * A function to call when a set access of the property is performed.
         * If this is passed in, set value and method to NULL (since these members won't be
used).
         * The given function is called implicitly by the runtime when the property is set
         * from JavaScript code (or if a set on the property is performed using a N-API call).
         */
    }
```

```

        .setter = NULL,
    /*
        * The value that's retrieved by a get access of the property if the property is a
    data property.
        * If this is passed in, set getter, setter, method and data to NULL (since these
    members won't be used).
        */
        .value = NULL,
    /*
        * The attributes associated with the particular property. See
    napi_property_attributes.
        */
        .attributes = napi_default,
    /*
        * The callback data passed into method, getter and setter if this function is
    invoked.
        */
        .data = NULL
    };
    /*
        * This method allows the efficient definition of multiple properties on a given object.
        */
    status = napi_define_properties(env, exports, 1, &desc);

    if (status != napi_ok)
        return;
}

NAPI_MODULE(hello, init)

```

N-API <https://riptutorial.com/zh-TW/node-js/topic/10539/n-api>

37: OAuth 2.0

Examples

RedisOAuth 2 - grant_type=password

redisrest apioauth2

redisLinuxWindowsredis manager。

node.jsredis。

- **app.js**

```
var express = require('express'),
    bodyParser = require('body-parser'),
    oauthserver = require('oauth2-server'); // Would be: 'oauth2-server'

var app = express();

app.use(bodyParser.urlencoded({ extended: true }));

app.use(bodyParser.json());

app.oauth = oauthserver({
  model: require('./routes/Oauth2/model'),
  grants: ['password', 'refresh_token'],
  debug: true
});

// Handle token grant requests
app.all('/oauth/token', app.oauth.grant());

app.get('/secret', app.oauth.authorise(), function (req, res) {
  // Will require a valid access_token
  res.send('Secret area');
});

app.get('/public', function (req, res) {
  // Does not require an access_token
  res.send('Public area');
});

// Error handling
app.use(app.oauth.errorHandler());

app.listen(3000);
```

- **routes / Oauth2 / model.jsOauth2**

```
var model = module.exports,
```

```

    util = require('util'),
    redis = require('redis');

var db = redis.createClient();

var keys = {
  token: 'tokens:%s',
  client: 'clients:%s',
  refreshToken: 'refresh_tokens:%s',
  grantTypes: 'clients:%s:grant_types',
  user: 'users:%s'
};

model.getAccessToken = function (bearerToken, callback) {
  db.hgetAll(util.format(keys.token, bearerToken), function (err, token) {
    if (err) return callback(err);

    if (!token) return callback();

    callback(null, {
      accessToken: token.accessToken,
      clientId: token.clientId,
      expires: token.expires ? new Date(token.expires) : null,
      userId: token.userId
    });
  });
};

model.getClient = function (clientId, clientSecret, callback) {
  db.hgetAll(util.format(keys.client, clientId), function (err, client) {
    if (err) return callback(err);

    if (!client || client.clientSecret !== clientSecret) return callback();

    callback(null, {
      clientId: client.clientId,
      clientSecret: client.clientSecret
    });
  });
};

model.getRefreshToken = function (bearerToken, callback) {
  db.hgetAll(util.format(keys.refreshToken, bearerToken), function (err, token) {
    if (err) return callback(err);

    if (!token) return callback();

    callback(null, {
      refreshToken: token.accessToken,
      clientId: token.clientId,
      expires: token.expires ? new Date(token.expires) : null,
      userId: token.userId
    });
  });
};

model.grantTypeAllowed = function (clientId, grantType, callback) {
  db.sismember(util.format(keys.grantTypes, clientId), grantType, callback);
};

model.saveAccessToken = function (accessToken, clientId, expires, user, callback) {

```

```

db.hmset(util.format(keys.token, accessToken), {
  accessToken: accessToken,
  clientId: clientId,
  expires: expires ? expires.toISOString() : null,
  userId: user.id
}, callback);
};

model.saveRefreshToken = function (refreshToken, clientId, expires, user, callback) {
  db.hmset(util.format(keys.refreshToken, refreshToken), {
    refreshToken: refreshToken,
    clientId: clientId,
    expires: expires ? expires.toISOString() : null,
    userId: user.id
  }, callback);
};

model.getUser = function (username, password, callback) {
  db.hgetall(util.format(keys.user, username), function (err, user) {
    if (err) return callback(err);

    if (!user || password !== user.password) return callback();

    callback(null, {
      id: username
    });
  });
};
};

```

redis

```

#!/usr/bin/env node

var db = require('redis').createClient();

db.multi()
  .hmset('users:username', {
    id: 'username',
    username: 'username',
    password: 'password'
  })
  .hmset('clients:client', {
    clientId: 'client',
    clientSecret: 'secret'
  })
  //clientId + clientSecret to base 64 will generate Y2xpZW50OnNlY3JldA==
  .sadd('clients:client:grant_types', [
    'password',
    'refresh_token'
  ])
  .exec(function (errs) {
    if (errs) {
      console.error(errs[0].message);
      return process.exit(1);
    }

    console.log('Client and user added successfully');
    process.exit();
  });

```

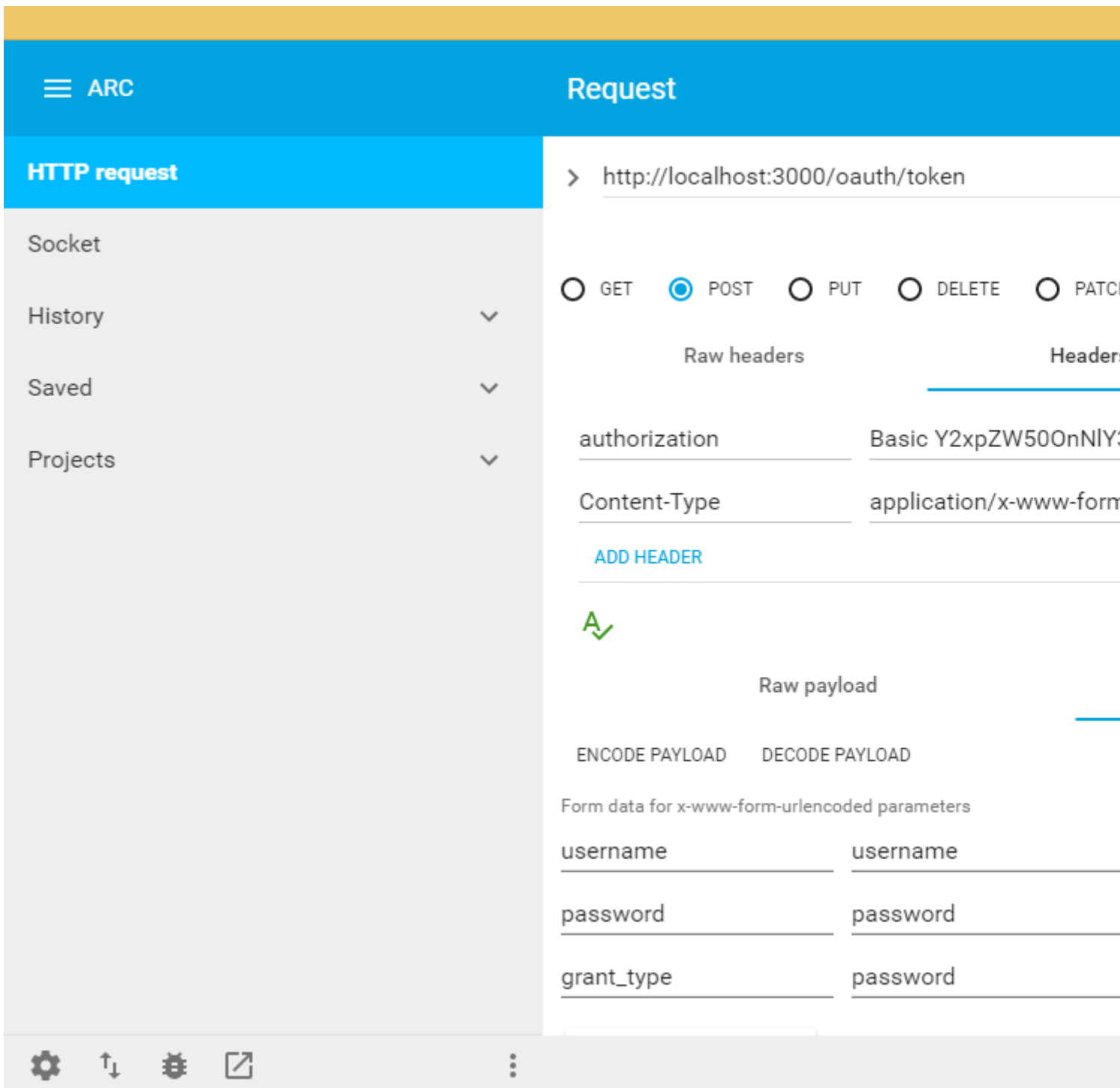
redis

The screenshot shows the Redis Desktop Manager interface. On the left, a tree view displays the database structure under 'local'. The selected key is 'users:username' in the 'db0' database. The right pane shows the details for this key, which is a HASH. A table displays the key-value pairs:

row	key	value
1	id	username
2	username	username
3	password	password

Below the table, the 'Key' and 'Value' sizes are both shown as 0 bytes. The bottom status bar includes buttons for 'Import / Export', 'Connect to Redis Server', and 'System log'. The Windows taskbar at the bottom shows various application icons.

api



1. redis

- clientId + secretIdbase64

2. username

grant_typepasswordredisredis

```
{
  "access_token": "1d3fe602da12a086ecb2b996fd7b7ae874120c4f",
  "token_type": "bearer", // Will be used to access api + access+token e.g. bearer
  1d3fe602da12a086ecb2b996fd7b7ae874120c4f
  "expires_in": 3600,
```

```
"refresh_token": "b6ad56e5c9aba63c85d7e21b1514680bbf711450"
}
```

api

The screenshot shows the ARC API client interface. On the left is a sidebar with navigation options: HTTP request (selected), Socket, History, Saved, and Projects. The main area is titled 'Request' and shows a GET request to 'http://localhost:3000/secret'. The request headers are displayed as follows:

Raw headers	Header
authorization	Bearer 1d3fe602da12a0
Content-Type	application/x-www-form

Below the headers, there is a green checkmark icon and a status bar indicating '200 OK' with a response time of '12.00 ms'. At the bottom, there are icons for copy, save, toggle visibility, and expand, along with the text 'Secret area'.

apiapi

The screenshot shows an HTTP client interface with a sidebar on the left containing 'ARC', 'HTTP request', 'Socket', 'History', 'Saved', and 'Projects'. The main area displays a GET request to 'http://localhost:3000/secret'. The headers section shows 'authorization: Bearer 1d3fe602da12a0...' and 'Content-Type: application/x-www-form...'. Below the headers, a green checkmark icon is visible. The response status is '401 Unauthorized' with a duration of '9.00 ms'. The raw response body is shown as a JSON object: { "code": 401, "error": "invalid_token", "error_description": "The access token provided is invalid" }.

redisaccess_tokenrefresh_tokenrefresh_token grant_typeoauth / tokenBasic clientId
 clientsecret64refresh_tokenaccess_token.

ARC Request

HTTP request

- Socket
- History
- Saved
- Projects

http://localhost:3000/oauth/token

GET POST PUT DELETE PATCH

Raw headers Header

authorization	Basic Y2xpZW50OnNIY
Content-Type	application/x-www-form

ADD HEADER

Raw payload

ENCODE PAYLOAD DECODE PAYLOAD

Form data for x-www-form-urlencoded parameters

refresh_token	b6ad56e5c9aba63c85d7
grant_type	refresh_token

ADD ANOTHER PARAMETER

⚙️ ↕️ 🐛 📄 ⋮

OAuth 2.0 <https://riptutorial.com/zh-TW/node-js/topic/9566/oauth-2-0>

38: passport.js

Passport。 ◦ Passport300Facebook / Google。 ◦

Examples

passport.jsLocalStrategy

```
var passport = require('passport');
var LocalStrategy = require('passport-local').Strategy;

passport.serializeUser(function(user, done) { //In serialize user you decide what to store in
the session. Here I'm storing the user id only.
  done(null, user.id);
});

passport.deserializeUser(function(id, done) { //Here you retrieve all the info of the user
from the session storage using the user id stored in the session earlier using serialize user.
  db.findById(id, function(err, user) {
    done(err, user);
  });
});

passport.use(new LocalStrategy(function(username, password, done) {
  db.findOne({'username':username},function(err,student){
    if(err)return done(err,{message:message});//wrong roll_number or password;
    var pass_retrieved = student.pass_word;
    bcrypt.compare(password, pass_retrieved, function(err3, correct) {
      if(err3){
        message = [{"msg": "Incorrect Password!"}];
        return done(null,false,{message:message}); // wrong password
      }
      if(correct){
        return done(null,student);
      }
    });
  });
}));

app.use(session({ secret: 'super secret' })); //to make passport remember the user on other
pages too.(Read about session store. I used express-sessions.)
app.use(passport.initialize());
app.use(passport.session());

app.post('/',passport.authenticate('local',{successRedirect:'/users' failureRedirect: '/'}),
function(req,res,next){
});
```

[passport.js](https://riptutorial.com/zh-TW/node-js/topic/8812/passport-js) <https://riptutorial.com/zh-TW/node-js/topic/8812/passport-js>

39: PostgreSQL

Examples

PostgreSQL

PostgreSQL npm◦

npm

```
npm install pg --save
```

PostgreSQL◦

Database_Name = studentsHost = localhostDB_User = postgres

```
var pg = require("pg")
var connectionString = "pg://postgres:postgres@localhost:5432/students";
var client = new pg.Client(connectionString);
client.connect();
```

◦

```
var queryString = "SELECT name, age FROM students " ;
var query = client.query(queryString);

query.on("row", (row, result)=> {
  result.addRow(row);
});

query.on("end", function (result) {
  //LOGIC
});
```

PostgreSQL <https://riptutorial.com/zh-TW/node-js/topic/7706/postgresql>

40: Restful API

Examples

```
Router.route('/')
  .get((req, res) => {
    Request.find((err, r) => {
      if(err){
        console.log(err)
      } else {
        res.json(r)
      }
    })
  })
  .post((req, res) => {
    const request = new Request({
      type: req.body.type,
      info: req.body.info
    });
    request.info.user = req.user._id;
    console.log("ABOUT TO SAVE REQUEST", request);
    request.save((err, r) => {
      if (err) {
        res.json({ message: 'there was an error saving your r' });
      } else {
        res.json(r);
      }
    });
  });
});
```

```
Router.route('/')
  .get((req, res) => {
    Request.find((err, r) => {
      if(err){
        console.log(err)
      } else {
        return next(err)
      }
    })
  })
  .post((req, res) => {
    const request = new Request({
      type: req.body.type,
      info: req.body.info
    });
    request.info.user = req.user._id;
    console.log("ABOUT TO SAVE REQUEST", request);
    request.save((err, r) => {
      if (err) {
        return next(err)
      } else {
        res.json(r);
      }
    });
  });
});
```

Restful API <https://riptutorial.com/zh-TW/node-js/topic/6490/restful-api->

41: Sequelize.js

Examples

Node.js `npm` `npmsequelize.js`

```
npm install --save sequelize
```

Node.js `mysql`

MySQL `Mariadb`

```
npm install --save mysql
```

PostgreSQL

```
npm install --save pg pg-hstore
```

SQLite

```
npm install --save sqlite
```

MSSQL

```
npm install --save tedious
```

Sequelize `ES5`

ES5

```
var Sequelize = require('sequelize');  
var sequelize = new Sequelize('database', 'username', 'password');
```

ES6 stage-0 Babel

```
import Sequelize from 'sequelize';  
const sequelize = new Sequelize('database', 'username', 'password');
```

`sequelize`

```
var db = new Sequelize('database', 'username', 'password');
```

```
var database = new Sequelize('database', 'username', 'password');
```

◦ API <http://docs.sequelizejs.com/en/v3/api/sequelize/>

sequelize;sequelize.define(...)sequelize.import(...) ◦ sequelize◦

1. sequelize.definemodelNameattributes [options]

◦

```
/* Initialize Sequelize */
const config = {
  username: "database username",
  password: "database password",
  database: "database name",
  host: "database's host URL",
  dialect: "mysql" // Other options are postgres, sqlite, mariadb and mssql.
}
var Sequelize = require("sequelize");
var sequelize = new Sequelize(config);

/* Define Models */
sequelize.define("MyModel", {
  name: Sequelize.STRING,
  comment: Sequelize.TEXT,
  date: {
    type: Sequelize.DATE,
    allowNull: false
  }
});
```

[docssequelize.com](https://docs.sequelize.com) ◦

2. sequelize.import

import◦ Sequelizeimport

```
/* Initialize Sequelize */
// Check previous code snippet for initialization

/* Define Models */
sequelize.import("../models/my_model.js"); // The path could be relative or absolute
```

```
module.exports = function(sequelize, DataTypes) {
  return sequelize.define("MyModel", {
    name: DataTypes.STRING,
    comment: DataTypes.TEXT,
    date: {
      type: DataTypes.DATE,
      allowNull: false
    }
  });
};
```


import **Sequelize** [GitHub](#) ◦

Sequelize.js <https://riptutorial.com/zh-TW/node-js/topic/7705/sequelize-js>

42: Socket.io

Examples

“”

```
npm install express
npm install socket.io
```

Node.js

```
const express = require('express');
const app = express();
const server = app.listen(3000, console.log("Socket.io Hello World server started!"));
const io = require('socket.io')(server);

io.on('connection', (socket) => {
  //console.log("Client connected!");
  socket.on('message-from-client-to-server', (msg) => {
    console.log(msg);
  })
  socket.emit('message-from-server-to-client', 'Hello World!');
});
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Hello World with Socket.io</title>
  </head>
  <body>
    <script src="https://cdn.socket.io/socket.io-1.4.5.js"></script>
    <script>
      var socket = io("http://localhost:3000");
      socket.on("message-from-server-to-client", function(msg) {
        document.getElementById('message').innerHTML = msg;
      });
      socket.emit('message-from-client-to-server', 'Hello World!');
    </script>
    <p>Socket.io Hello World client started!</p>
    <p id="message"></p>
  </body>
</html>
```

Socket.io <https://riptutorial.com/zh-TW/node-js/topic/4261/socket-io>

43: TCP

Examples

TCP

```
// Include Nodejs' net module.
const Net = require('net');
// The port on which the server is listening.
const port = 8080;

// Use net.createServer() in your code. This is just for illustration purpose.
// Create a new TCP server.
const server = new Net.Server();
// The server listens to a socket for a client to make a connection request.
// Think of a socket as an end point.
server.listen(port, function() {
  console.log(`Server listening for connection requests on socket localhost:${port}`.);
});

// When a client requests a connection with the server, the server creates a new
// socket dedicated to that client.
server.on('connection', function(socket) {
  console.log('A new connection has been established.');
```

```
  // Now that a TCP connection has been established, the server can send data to
  // the client by writing to its socket.
  socket.write('Hello, client.');
```

```
  // The server can also receive data from the client by reading from its socket.
  socket.on('data', function(chunk) {
    console.log(`Data received from client: ${chunk.toString}`.);
  });

  // When the client requests to end the TCP connection with the server, the server
  // ends the connection.
  socket.on('end', function() {
    console.log('Closing connection with the client');
  });

  // Don't forget to catch error, for your own sake.
  socket.on('error', function(err) {
    console.log(`Error: ${err}`.);
  });
});
```

TCP

```
// Include Nodejs' net module.
const Net = require('net');
// The port number and hostname of the server.
const port = 8080;
const host = 'localhost';

// Create a new TCP client.
```

```
const client = new Net.Socket();
// Send a connection request to the server.
client.connect({ port: port, host: host }, function() {
  // If there is no error, the server has accepted the request and created a new
  // socket dedicated to us.
  console.log('TCP connection established with the server.');
```



```
  // The client can now send data to the server by writing to its socket.
  client.write('Hello, server.');
```



```
});

// The client can also receive data from the server by reading from its socket.
client.on('data', function(chunk) {
  console.log(`Data received from the server: ${chunk.toString()}.`);

  // Request an end to the connection after the data has been received.
  client.end();
});

client.on('end', function() {
  console.log('Requested an end to the TCP connection');
});
```

TCP <https://riptutorial.com/zh-TW/node-js/topic/6545/tcp>

44:

Examples

multer

- uploads ◦
- `multer` `npm i -S multer`

server.js

```
var express = require("express");
var multer = require('multer');
var app = express();
var fs = require('fs');

app.get('/', function(req, res) {
    res.sendFile(__dirname + "/index.html");
});

var storage = multer.diskStorage({
    destination: function (req, file, callback) {
        fs.mkdir('./uploads', function(err) {
            if(err) {
                console.log(err.stack)
            } else {
                callback(null, './uploads');
            }
        })
    },
    filename: function (req, file, callback) {
        callback(null, file.fieldname + '-' + Date.now());
    }
});

app.post('/api/file', function(req, res) {
    var upload = multer({ storage : storage }).single('userFile');
    upload(req, res, function(err) {
        if(err) {
            return res.end("Error uploading file.");
        }
        res.end("File is uploaded");
    });
});

app.listen(3000, function() {
    console.log("Working on port 3000");
});
```

index.html

```
<form id = "uploadForm"
  enctype = "multipart/form-data"
  action = "/api/file"
```

```
    method    = "post"
  >
  <input type="file" name="userFile" />
  <input type="submit" value="Upload File" name="submit">
</form>
```

Node.js

server.js

```
var path = require('path');
```

```
callback(null, file.fieldname + '-' + Date.now());
```

```
callback(null, file.fieldname + '-' + Date.now() + path.extname(file.originalname));
```

-
- - `var upload = multer({ storage : storage}).single('userFile');` **fileFilter**

```
var upload = multer({
  storage: storage,
  fileFilter: function (req, file, callback) {
    var ext = path.extname(file.originalname);
    if(ext !== '.png' && ext !== '.jpg' && ext !== '.gif' && ext !== '.jpeg') {
      return callback(new Error('Only images are allowed'))
    }
    callback(null, true)
  }
}).single('userFile');
```

png jpg gifjpeg

```
npm i formidable@latest
```

8080

```
var formidable = require('formidable'),
    http = require('http'),
    util = require('util');

http.createServer(function(req, res) {
  if (req.url === '/upload' && req.method.toLowerCase() === 'post') {
    // parse a file upload
    var form = new formidable.IncomingForm();

    form.parse(req, function(err, fields, files) {
      if (err)
        do-smth; // process error
    });
  }
});
```

```
// Copy file from temporary place
// var fs = require('fs');
// fs.rename(file.path, <targetPath>, function (err) { ... });

// Send result on client
res.writeHead(200, {'content-type': 'text/plain'});
res.write('received upload:\n\n');
res.end(util.inspect({fields: fields, files: files}));
});

return;
}

// show a file upload form
res.writeHead(200, {'content-type': 'text/html'});
res.end(
  '<form action="/upload" enctype="multipart/form-data" method="post">'+
  '<input type="text" name="title"><br>'+
  '<input type="file" name="upload" multiple="multiple"><br>'+
  '<input type="submit" value="Upload">'+
  '</form>'
);
}).listen(8080);
```

<https://riptutorial.com/zh-TW/node-js/topic/4080/>

45:

EventloopGUI。

Examples

。

Eventloop

```
while true:
    wait for something to happen
    react to whatever happened
```

HTTP

```
while true:
    socket = wait for the next TCP connection
    read the HTTP request headers from (socket)
    file_contents = fetch the requested file from disk
    write the HTTP response headers to (socket)
    write the (file_contents) to (socket)
    close(socket)
```

HTTP。 。 HTTP。

。

HTTP

```
function handle_connection(socket):
    read the HTTP request headers from (socket)
    file_contents = fetch the requested file from disk
    write the HTTP response headers to (socket)
    write the (file_contents) to (socket)
    close(socket)
while true:
    socket = wait for the next TCP connection
    spawn a new thread doing handle_connection(socket)
```

HTTP。 。 Apache。

。 。 。

HTTP

```
while true:
    event = wait for the next event to happen
    if (event.type == NEW_TCP_CONNECTION):
        conn = new Connection
        conn.socket = event.socket
        start reading HTTP request headers from (conn.socket) with userdata = (conn)
    else if (event.type == FINISHED_READING_FROM_SOCKET):
        conn = event.userdata
        start fetching the requested file from disk with userdata = (conn)
    else if (event.type == FINISHED_READING_FROM_DISK):
        conn = event.userdata
        conn.file_contents = the data we fetched from disk
        conn.current_state = "writing headers"
        start writing the HTTP response headers to (conn.socket) with userdata = (conn)
    else if (event.type == FINISHED_WRITING_TO_SOCKET):
        conn = event.userdata
        if (conn.current_state == "writing headers"):
            conn.current_state = "writing file contents"
            start writing (conn.file_contents) to (conn.socket) with userdata = (conn)
        else if (conn.current_state == "writing file contents"):
            close(conn.socket)
```

◦ ◦ ◦ ◦

Linuxpollpoll“”◦ recvsend“”◦

[1]◦ “” []◦ [https //www.quora.com/How-does-an-event-loop-work](https://www.quora.com/How-does-an-event-loop-work)

<https://riptutorial.com/zh-TW/node-js/topic/8652/>

46:

```
emit()
```

```
myDog.on('bark', (howLoud, howLong, howIntense) => {
  // handle the event
})
myDog.emit('bark', 'loudly', '5 seconds long', 'fiercely')
```

```
myDog.on('urinate', () => console.log('My first thought was "Oh-no"'))
myDog.on('urinate', () => console.log('My second thought was "Not my lawn :)"))
myDog.emit('urinate')
// The console.logs will happen in the right order because they were registered in that order.
```

```
prependListener()
```

```
myDog.prependListener('urinate', () => console.log('This happens before my first and second
thoughts, even though it was registered after them'))
```

```
once on prependOnceListener prependListener ◦ ◦
```

```
myDog.removeAllListeners()
```

Examples

HTTP

HTTP_{server.js}

```
const EventEmitter = require('events')
const serverEvents = new EventEmitter()

// Set up an HTTP server
const http = require('http')
const httpServer = http.createServer((request, response) => {
  // Handler the request...
  // Then emit an event about what happened
  serverEvents.emit('request', request.method, request.url)
});

// Expose the event emitter
module.exports = serverEvents
```

supervisor.js

```
const server = require('./server.js')
// Since the server exported an event emitter, we can listen to it for changes:
server.on('request', (method, url) => {
  console.log(`Got a request: ${method} ${url}`)
})
```

request◦

Nodepub-sub ◦ ◦

```
// Require events to start using them
const EventEmitter = require('events').EventEmitter;
// Dogs have events to publish, or emit
class Dog extends EventEmitter {};
class Food {};

let myDog = new Dog();

// When myDog is chewing, run the following function
myDog.on('chew', (item) => {
  if (item instanceof Food) {
    console.log('Good dog');
  } else {
    console.log(`Time to buy another ${item}`);
  }
});

myDog.emit('chew', 'shoe'); // Will result in console.log('Time to buy another shoe')
const bacon = new Food();
myDog.emit('chew', bacon); // Will result in console.log('Good dog')
```

dogpublisher / EventEmitter/◦

```
myDog.on('bark', () => {
  console.log('WHO'S AT THE DOOR?');
  // Panic
});
```

```
myDog.on('chew', takeADeepBreathe);
myDog.on('chew', calmDown);
// Undo the previous line with the next one:
myDog.removeListener('chew', calmDown);
```

```
myDog.once('chew', pet);
```

◦

EventEmitter.eventNames◦

```
const EventEmitter = require("events");
class MyEmitter extends EventEmitter{}

var emitter = new MyEmitter();

emitter
.on("message", function(){ //listen for message event
  console.log("a message was emitted!");
})
.on("message", function(){ //listen for message event
  console.log("this is not the right message");
```

```

})
.on("data", function(){ //listen for data event
  console.log("a data just occurred!!");
});

console.log(emitter.eventNames()); //=> ["message","data"]
emitter.removeAllListeners("data");//=> removeAllListeners to data event
console.log(emitter.eventNames()); //=> ["message"]

```

RunKit

Emitter.listenerCounteventName

```

const EventEmitter = require("events");
class MyEmitter extends EventEmitter{}
var emitter = new MyEmitter();

emitter
.on("data", ()=>{ // add listener for data event
  console.log("data event emitter");
});

console.log(emitter.listenerCount("data")) // => 1
console.log(emitter.listenerCount("message")) // => 0

emitter.on("message", function mListener(){ //add listener for message event
  console.log("message event emitted");
});
console.log(emitter.listenerCount("data")) // => 1
console.log(emitter.listenerCount("message")) // => 1

emitter.once("data", (stuff)=>{ //add another listener for data event
  console.log(`Tell me my ${stuff}`);
})

console.log(emitter.listenerCount("data")) // => 2
console.log(emitter.listenerCount("message"))// => 1

```

<https://riptutorial.com/zh-TW/node-js/topic/1623/>

47: Browserify“”

Examples

- file.js

file.js。

JavaScriptNodeJSURL。

```
const querystring = require('querystring');
var ref = querystring.parse("foo=bar&abc=xyz&abc=123");
```

URL。

```
const querystring = require('querystring');
```

.parseURL。 URLstr。

'foo=bar&abc=xyz&abc=123'

```
{ foo: 'bar', abc: ['xyz', '123'] }
```

requireNode.js。

Browserify

BrowserifyrequireNode。 。

1. npm。

npm install -g **browserify**

2. file.jsnpm

npm install **querystring**

。

3. browserify file.jsbundle.js

browserify file.js -o bundle.js

Browserifyfor *require*

4.

html

```
<script src="bundle.js"></script>
```

.jsfile.jsbundle.js ◦ ◦

file.js ◦ **bundle.js**

file.js ◦ **bundle.js****bundle.js****file.js** ◦

Browserfy"" <https://riptutorial.com/zh-TW/node-js/topic/7123/browserfy-->

48: Express.JSajax

Examples

AJAX

```
app.jsvar app = express.app()var app = express.app()
```

```
app.post(function(req, res, next){
  next();
});
```

index.js

```
router.get('/ajax', function(req, res){
  res.render('ajax', {title: 'An Ajax Example', quote: "AJAX is great!"});
});
router.post('/ajax', function(req, res){
  res.render('ajax', {title: 'An Ajax Example', quote: req.body.quote});
});
```

```
/viewsajax.jade / ajax.pugajax.ejs
```

Jade / PugJS

```
extends layout
script(src="http://code.jquery.com/jquery-3.1.0.min.js")
script(src="/magic.js")
h1 Quote: !{quote}
form(method="post" id="changeQuote")
  input(type='text', placeholder='Set quote of the day', name='quote')
  input(type="submit", value="Save")
```

EJS

```
<script src="http://code.jquery.com/jquery-3.1.0.min.js"></script>
<script src="/magic.js"></script>
<h1>Quote: <%=quote%> </h1>
<form method="post" id="changeQuote">
  <input type="text" placeholder="Set quote of the day" name="quote"/>
  <input type="submit" value="Save">
</form>
```

```
/publicmagic.js
```

```
$(document).ready(function(){
  $("form#changeQuote").on('submit', function(e){
    e.preventDefault();
    var data = $('input[name=quote]').val();
    $.ajax({
      type: 'post',
```

```
    url: '/ajax',
    data: data,
    dataType: 'text'
  })
  .done(function(data) {
    $('h1').html(data.quote);
  });
});
```

Express.JSajax <https://riptutorial.com/zh-TW/node-js/topic/6738/express-jsajax>

49: ExpressWeb

ExpressNode.js WebWeb。

Expressexpressjs.com 。 [GitHub](https://github.com) 。

- app.getpath [middleware]callback [callback ...]
- app.putpath [middleware]callback [callback ...]
- app.post[[...]
- app ['delete'][[...]
- app.usepath [middleware]callback [callback ...]
- app.use

path	URL。
middleware	◦ callback◦ ◦
callback	path◦ callback(request, response, next) request response◦next◦
request	HTTP。
response	◦
next	◦ ◦

Examples

shellnpm install express --save**npm**Express

app.js**Express**app.getapp.get /ping

```
const express = require('express');  
  
const app = express();  
  
app.get('/ping', (request, response) => {  
  response.send('pong');  
});  
  
app.listen(8080, 'localhost');
```

shell

```
> node app.js
```

localhost8080◦ app.listenhostname|Pllocalhost. 0◦

shell“pong”

```
> curl http://localhost:8080/ping
pong
```

WebURL [http// localhost8080 / ping](http://localhost8080/ping)

```
const express = require('express');
const app = express();
```

```
app.get('/someUri', function (req, res, next) {})
```

HTTP。 HTTP

```
// GET www.domain.com/myPath
app.get('/myPath', function (req, res, next) {})

// POST www.domain.com/myPath
app.post('/myPath', function (req, res, next) {})

// PUT www.domain.com/myPath
app.put('/myPath', function (req, res, next) {})

// DELETE www.domain.com/myPath
app.delete('/myPath', function (req, res, next) {})
```

。 HTTP

```
app.all('/myPath', function (req, res, next) {})
```

```
app.use('/myPath', function (req, res, next) {})
```

```
app.use('*', function (req, res, next) {})
```

```
// * wildcard will route for all paths
```

```
app.route('/myPath')
  .get(function (req, res, next) {})
  .post(function (req, res, next) {})
  .put(function (req, res, next) {})
```

HTTP。 reqresnext。

```
// GET www.domain.com/myPath
app.get('/myPath', myFunction, function (req, res, next) {})
```

```
// other.js
exports.doSomething = function(req, res, next) { /* do some stuff */};
```

```
const other = require('./other.js');
app.get('/someUri', myFunction, other.doSomething);
```

◦

URL_{req}◦ **/settings/:user_id/settings/32135?field=name**

```
// get the full path
req.originalUrl // => /settings/32135?field=name

// get the user_id param
req.params.user_id // => 32135

// get the query value of the field
req.query.field // => 'name'
```

```
req.get('Content-Type')
// "text/plain"
```

◦ ◦

```
var app = require('express')();
var bodyParser = require('body-parser');

app.use(bodyParser.json()); // for parsing application/json
app.use(bodyParser.urlencoded({ extended: true })); // for parsing application/x-www-form-urlencoded
```

```
PUT /settings/32135
{
  "name": "Peter"
}
```

```
req.body.name
// "Peter"
```

cookie[cookie-parser](#)

```
req.cookies.name
```

Web

```
// greet.js
const express = require('express');

module.exports = function(options = {}) { // Router factory
  const router = express.Router();

  router.get('/greet', (req, res, next) => {
    res.end(options.greeting);
  });
};
```

```
    return router;
};
```

```
// app.js
const express = require('express');
const greetMiddleware = require('./greet.js');

express()
  .use('/api/v1/', greetMiddleware({ greeting: 'Hello world' }))
  .listen(8080);
```

◦

http://<hostname>:8080/api/v1/greet Hello world

◦

```
// greet.js
const express = require('express');

module.exports = function(options = {}) { // Router factory
  const router = express.Router();
  // Get controller
  const {service} = options;

  router.get('/greet', (req, res, next) => {
    res.end(
      service.createGreeting(req.query.name || 'Stranger')
    );
  });

  return router;
};
```

```
// app.js
const express = require('express');
const greetMiddleware = require('./greet.js');

class GreetingService {
  constructor(greeting = 'Hello') {
    this.greeting = greeting;
  }

  createGreeting(name) {
    return `${this.greeting}, ${name}!`;
  }
}

express()
  .use('/api/v1/service1', greetMiddleware({
    service: new GreetingService('Hello'),
  }))
  .use('/api/v1/service2', greetMiddleware({
    service: new GreetingService('Hi'),
  }));
```

```
.listen(8080);
```

http://<hostname>:8080/api/v1/service1/greet?name=World Hello, World

http://<hostname>:8080/api/v1/service2/greet?name=WorldHi, World ◦

Jade ◦ 201512Jade_pug ◦

```
const express = require('express'); //Imports the express module
const app = express(); //Creates an instance of the express module

const PORT = 3000; //Randomly chosen port

app.set('view engine','jade'); //Sets jade as the View Engine / Template Engine
app.set('views','src/views'); //Sets the directory where all the views (.jade files) are
stored.

//Creates a Root Route
app.get('/',function(req, res){
  res.render('index'); //renders the index.jade file into html and returns as a response.
  The render function optionally takes the data to pass to the view.
});

//Starts the Express server with a callback
app.listen(PORT, function(err) {
  if (!err) {
    console.log('Server is running at port', PORT);
  } else {
    console.log(JSON.stringify(err));
  }
});
```

Handlebars hbs ejs ◦ npm install ◦ Handlebars hbs Jade jade EJS ejs ◦

EJS

EJSHTML ◦

EJS“ <% "" %> ”<%=var_name%>

```
<h1><%= title %></h1>
<ul>
<% for(var i=0; i<supplies.length; i++) { %>
  <li>
    <a href='supplies/<%= supplies[i] %>'>
      <%= supplies[i] %>
    </a>
  </li>
<% } %>
```

HTMLEJSforli forEJSfor

<%=

```
Message:<br>
<input type="text" value="<%= message %>" name="message" required>
```

EJS◦ `res.render('index', {message: message});res.render('index', {message: message});` **index.ejs**
ejs◦

EJSif while**javascript**◦

ExpressJSJSON API

```
var express = require('express');
var cors = require('cors'); // Use cors module for enable Cross-origin resource sharing

var app = express();
app.use(cors()); // for all routes

var port = process.env.PORT || 8080;

app.get('/', function(req, res) {
  var info = {
    'string_value': 'StackOverflow',
    'number_value': 8476
  }
  res.json(info);

  // or
  /* res.send(JSON.stringify({
    string_value: 'StackOverflow',
    number_value: 8476
  })); */

  //you can add a status code to the json response
  /* res.status(200).json(info) */
})

app.listen(port, function() {
  console.log('Node.js listening on port ' + port)
})
```

<http://localhost:8080/>

```
{
  string_value: "StackOverflow",
  number_value: 8476
}
```

ExpressWeb◦

index.htmlscript.js◦

“public”◦

```
project root
├─ server.js
```

```
├─ package.json
├─ public
│  └─ index.html
│  └─ script.js
```

Express

```
const express = require('express');
const app = express();

app.use(express.static('public'));
```

index.htmlscript.js“public”URL/public/ ◦ **express**◦

```
app.use('/static', express.static('public'));
```

/static/◦

```
app.use(express.static('public'));
app.use(express.static('images'));
app.use(express.static('files'));
```

Express◦ ◦

Django

Express◦ [express-reverse](#)

```
npm install express-reverse
```

```
var app = require('express')();
require('express-reverse')(app);
```

```
app.get('test', '/hello', function(req, res) {
  res.end('hello');
});
```

route **Express**◦ app

```
require('./middlewares/routing')(app);
```

```
module.exports = (app) => {
  app.get('test', '/hello', function(req, res) {
    res.end('hello');
  });
};
```

◦

Express/views””””views

/error.pug

```
html
  body
    h1= message
    h2= error.status
    p= error.stack
```

◦ (err, req, res, next)(err, req, res, next)

app.js

```
// catch 404 and forward to error handler
app.use(function(req, res, next) {
  var err = new Error('Not Found');
  err.status = 404;

  //pass error to the next matching route.
  next(err);
});

// handle error, print stacktrace
app.use(function(err, req, res, next) {
  res.status(err.status || 500);

  res.render('error', {
    message: err.message,
    error: err
  });
});
```

◦

Expressnext◦ next() express◦ next(err)◦ next('route')◦ ◦

/api/foo/api/bar◦

```
app.get('/api', function(req, res, next) {
  // Both /api/foo and /api/bar will run this
  lookupMember(function(err, member) {
    if (err) return next(err);
    req.member = member;
    next();
  });
});

app.get('/api/foo', function(req, res, next) {
  // Only /api/foo will run this
  doSomethingWithMember(req.member);
});

app.get('/api/bar', function(req, res, next) {
  // Only /api/bar will run this
  doSomethingDifferentWithMember(req.member);
});
```



```
});
```

function(err, req, res, next)function(err, req, res, next) ◦ app.get('/foo', function(err, req, res, next) ◦

```
app.get('/foo', function(req, res, next) {
  doSomethingAsync(function(err, data) {
    if (err) return next(err);
    renderPage(data);
  });
});

// In the case that doSomethingAsync return an error, this special
// error handler middleware will be called with the error as the
// first parameter.
app.use(function(err, req, res, next) {
  renderErrorPage(err);
});
```

◦ ◦

```
app.get('/bananas', function(req, res, next) {
  getMember(function(err, member) {
    if (err) return next(err);
    // If there's no member, don't try to look
    // up data. Just go render the page now.
    if (!member) return next('route');
    // Otherwise, call the next middleware and fetch
    // the member's data.
    req.member = member;
    next();
  });
}, function(req, res, next) {
  getMemberData(req.member, function(err, data) {
    if (err) return next(err);
    // If this member has no data, don't bother
    // parsing it. Just go render the page now.
    if (!data) return next('route');
    // Otherwise, call the next middleware and parse
    // the member's data. THEN render the page.
    req.member.data = data;
    next();
  });
}, function(req, res, next) {
  req.member.parsedData = parseMemberData(req.member.data);
  next();
});

app.get('/bananas', function(req, res, next) {
  renderBananas(req.member);
});
```

◦

```
app.get('/path/:id(\\d+)', function (req, res, next) { // please note: "next" is passed
  if (req.params.id == 0) // validate param
    return next(new Error('Id is 0')); // go to first Error handler, see below
```

```

// Catch error on sync operation
var data;
try {
  data = JSON.parse('/file.json');
} catch (err) {
  return next(err);
}

// If some critical error then stop application
if (!data)
  throw new Error('SmtH wrong');

// If you need send extra info to Error handler
// then send custom error (see Appendix B)
if (smth)
  next(new MyError('smth wrong', arg1, arg2))

// Finish request by res.render or res.end
res.status(200).end('OK');
});

// Be sure: order of app.use have matter
// Error handler
app.use(function(err, req, res, next) {
  if (smth-check, e.g. req.url != 'POST')
    return next(err); // go-to Error handler 2.

  console.log(req.url, err.message);

  if (req.xhr) // if req via ajax then send json else render error-page
    res.json(err);
  else
    res.render('error.html', {error: err.message});
});

// Error handler 2
app.use(function(err, req, res, next) {
  // do smth here e.g. check that error is MyError
  if (err instanceof MyError) {
    console.log(err.message, err.arg1, err.arg2);
  }
  ...
  res.end();
});

```

A.

```

// "In Express, 404 responses are not the result of an error,
// so the error-handler middleware will not capture them."
// You can change it.
app.use(function(req, res, next) {
  next(new Error(404));
});

```

B.

```

// How to define custom error
var util = require('util');

```

```

...
function MyError(message, arg1, arg2) {
  this.message = message;
  this.arg1 = arg1;
  this.arg2 = arg2;
  Error.captureStackTrace(this, MyError);
}
util.inherits(MyError, Error);
MyError.prototype.name = 'MyError';

```

Hookreqres

`app.use()` **“closefinish”**。

```

app.use(function (req, res, next) {
  function afterResponse() {
    res.removeListener('finish', afterResponse);
    res.removeListener('close', afterResponse);

    // actions after response
  }
  res.on('finish', afterResponse);
  res.on('close', afterResponse);

  // action before request
  // eventually calling `next()`
  next();
});
...
app.use(app.router);

```

。

`app.router`“”。

POST

`app.getExpressgetapp.post`。

POST `body-parser`。 `POST` `PUT` `DELETE`。

`Body-Parserreq.body`

```

var bodyParser = require('body-parser');

const express = require('express');

const app = express();

// Parses the body for POST, PUT, DELETE, etc.
app.use(bodyParser.json());

app.use(bodyParser.urlencoded({ extended: true }));

```

```

app.post('/post-data-here', function(req, res, next){

    console.log(req.body); // req.body contains the parsed body of the request.

});

app.listen(8080, 'localhost');

```

cookiecookie

cookie-parsercookie

```

var express = require('express');
var cookieParser = require('cookie-parser'); // module for parsing cookies
var app = express();
app.use(cookieParser());

app.get('/setcookie', function(req, res){
    // setting cookies
    res.cookie('username', 'john doe', { maxAge: 900000, httpOnly: true });
    return res.send('Cookie has been set');
});

app.get('/getcookie', function(req, res) {
    var username = req.cookies['username'];
    if (username) {
        return res.send(username);
    }

    return res.send('No cookie found');
});

app.listen(3000);

```

Express

Express◦

```

app.use(function(req, res, next){ }); // signature

```

user◦

```

var express = require('express');
var app = express();

//each request will pass through it
app.use(function(req, res, next){
    req.user = 'testuser';
    next(); // it will pass the control to next matching route
});

app.get('/', function(req, res){
    var user = req.user;
    console.log(user); // testuser
    return res.send(user);
});

```

```
});  
  
app.listen(3000);
```

Express

Express ◦

```
var express = require('express');  
var app = express();  
  
//GET /names/john  
app.get('/names/:name', function(req, res, next){  
  if (req.params.name == 'john'){  
    return res.send('Valid Name');  
  } else{  
    next(new Error('Not valid name')); //pass to error handler  
  }  
});  
  
//error handler  
app.use(function(err, req, res, next){  
  console.log(err.stack); // e.g., Not valid name  
  return res.status(500).send('Internal Server Occured');  
});  
  
app.listen(3000);
```

reqres - ◦

resreq◦

cors◦ **CORS**

```
app.use(cors());
```

◦

Expresshello world◦

- '/'
- “/”

“404”◦

```
'use strict';  
  
const port = process.env.PORT || 3000;  
  
var app = require('express')();  
app.listen(port);  
  
app.get('/', (req, res) => res.send('HelloWorld!'));
```

```
app.get('/wiki', (req, res) => res.send('This is wiki page.'));  
app.use((req, res) => res.send('404-PageNotFound'));
```

404Express。

ExpressWeb <https://riptutorial.com/zh-TW/node-js/topic/483/expressweb>

50: IISNodeIISNode.js Web

ExpressView EnginevirtualDirPath

```
`res.render('index', { virtualDirPath: virtualDirPath });`
```

◦ IISNode◦

-
- v4.x
 - IIS 7.x / 8.x.
 - Socket.io v1.3.x

Examples

[IISNodeNode.js Web.NETIIS 7/8◦ Windowsnode.exeIIS◦](#)

[IISNodenode.exe◦ node.exe IIS ◦](#)

[IISNode.jsIISNode◦](#)

1. [Node.jsIIS3264◦](#)
2. [IISNodex86x64 IIS◦](#)
3. [IISMicrosoft URL-Rewrite Module for IIS ◦](#)
 - [Node.js◦](#)
4. [Node.jsWeb.config ◦](#)
5. [iisnode.ymlWeb.config<iisnode>IISNode◦](#)

ExpressHello World

[IISIIS 7/8Node.js Web App◦ /Node.js◦ Node.js 64◦](#)

[IISNode / Node.js Web◦ Web.configIISNode Web App◦](#)

```
- /app_root
- package.json
- server.js
- Web.config
```

server.js - Express Application

```
const express = require('express');
const server = express();

// We need to get the port that IISNode passes into us
// using the PORT environment variable, if it isn't set use a default value
const port = process.env.PORT || 3000;

// Setup a route at the index of our app
server.get('/', (req, res) => {
  return res.status(200).send('Hello World');
});

server.listen(port, () => {
  console.log(`Listening on ${port}`);
});
```

Web.config

Web.config IIS Web.config URL <rewrite><rules> IISNode <handler> ◦ <system.webServer>◦

iisnode.yml Web.config <iisnode><system.webServer> IISNode◦ Web.config iisnode.yml iisnode.yml ◦ ◦

IISNode

IIS server.js Node.js Web App◦ IISNode <handler><handlers>◦

```
<handlers>
  <add name="iisnode" path="server.js" verb="*" modules="iisnode"/>
</handlers>
```

URL

IISNode.js IISNode◦ URL http://<host>/server.js server.js 404◦ IISNode Web URL◦

```
<rewrite>
  <rules>
    <!-- First we consider whether the incoming URL matches a physical file in the /public
    folder -->
    <rule name="StaticContent" patternSyntax="Wildcard">
      <action type="Rewrite" url="public/{R:0}" logRewrittenUrl="true"/>
      <conditions>
        <add input="{REQUEST_FILENAME}" matchType="IsFile" negate="true"/>
      </conditions>
      <match url="*.*/>
    </rule>

    <!-- All other URLs are mapped to the Node.js application entry point -->
```



```

    <rule name="DynamicContent">
      <conditions>
        <add input="{REQUEST_FILENAME}" matchType="IsFile" negate="True"/>
      </conditions>
      <action type="Rewrite" url="server.js"/>
    </rule>
  </rules>
</rewrite>

```

[Web.config](#) [64Node.js](#)

[IISNode.js](#)

IIS

[IISIISNode](#)

[IISNodeIISNode](#) [appSetting](#) [Web.config](#) [<appSettings>](#) [process.env](#)

[<appSettings>](#)

```

<appSettings>
  <add key="virtualDirPath" value="/foo" />
</appSettings>

```

[Node.js](#) [virtualDirPath](#)

```

console.log(process.env.virtualDirPath); // prints /foo

```

[<appSettings>](#)

```

// Access the virtualDirPath appSettings and give it a default value of '/'
// in the event that it doesn't exist or isn't set
var virtualDirPath = process.env.virtualDirPath || '/';

// We also want to make sure that our virtualDirPath
// always starts with a forward slash
if (!virtualDirPath.startsWith('/', 0))
  virtualDirPath = '/' + virtualDirPath;

// Setup a route at the index of our app
server.get(virtualDirPath, (req, res) => {
  return res.status(200).send('Hello World');
});

```

[virtualDirPath](#)

```

// Public Directory
server.use(express.static(path.join(virtualDirPath, 'public')));
// Bower
server.use('/bower_components', express.static(path.join(virtualDirPath,
'bower_components')));

```

```

const express = require('express');
const server = express();

const port = process.env.PORT || 3000;

// Access the virtualDirPath appSettings and give it a default value of '/'
// in the event that it doesn't exist or isn't set
var virtualDirPath = process.env.virtualDirPath || '/';

// We also want to make sure that our virtualDirPath
// always starts with a forward slash
if (!virtualDirPath.startsWith('/', 0))
    virtualDirPath = '/' + virtualDirPath;

// Public Directory
server.use(express.static(path.join(virtualDirPath, 'public')));
// Bower
server.use('/bower_components', express.static(path.join(virtualDirPath,
'bower_components')));

// Setup a route at the index of our app
server.get(virtualDirPath, (req, res) => {
    return res.status(200).send('Hello World');
});

server.listen(port, () => {
    console.log(`Listening on ${port}`);
});

```

Socket.io IIS Node

Socket.io IIS Node / Web.config ◦

Socket.io / socket.io IIS Node IIS IIS Node ◦ IIS Node <handler><handler> ◦

```

<handlers>
  <add name="iisnode-socketio" path="server.js" verb="*" modules="iisnode" />
</handlers>

```

<handlers> URL ◦ /socket.io Socket.io ◦

```

<rule name="SocketIO" patternSyntax="ECMAScript">
  <match url="socket.io.+" />
  <action type="Rewrite" url="server.js" />
</rule>

```

IIS 8 Web.config webSockets ◦ IIS 7 webSocket ◦

<webSocket enabled="false" />

IIS Node IIS Node.js Web <https://riptutorial.com/zh-TW/node-js/topic/6003/iisnodeiisnode-js-web>

51: Node.js API

Examples

Express api

Node.js **api** Express **Web**

GET **api**

```
var express = require('express');
var app = express();

var users = [{
  id: 1,
  name: "John Doe",
  age : 23,
  email: "john@doe.com"
}];

// GET /api/users
app.get('/api/users', function(req, res){
  return res.json(users);    //return response as JSON
});

app.listen('3000', function(){
  console.log('Server listening on port 3000');
});
```

POST api Express

Express **POST api** GET **post** req.body body-parser

```
var express = require('express');
var app = express();
// for parsing the body in POST request
var bodyParser = require('body-parser');

var users = [{
  id: 1,
  name: "John Doe",
  age : 23,
  email: "john@doe.com"
}];

app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());

// GET /api/users
app.get('/api/users', function(req, res){
  return res.json(users);
});
```

```
/* POST /api/users
  {
    "user": {
      "id": 3,
      "name": "Test User",
      "age" : 20,
      "email": "test@test.com"
    }
  }
*/
app.post('/api/users', function (req, res) {
  var user = req.body.user;
  users.push(user);

  return res.send('User has been added successfully');
});

app.listen('3000', function(){
  console.log('Server listening on port 3000');
});
```

Node.jsAPI <https://riptutorial.com/zh-TW/node-js/topic/5991/node-jsapi>

52: Streams



Examples

StreamsTextFile

I/O.

```
var http = require('http');
var fs = require('fs');

var server = http.createServer(function (req, res) {
  fs.readFile(__dirname + '/data.txt', function (err, data) {
    res.end(data);
  });
});
server.listen(8000);
```

data.txt。 data.txt。

。

reqresfs.createReadStreamfs.readFile

```
var http = require('http');
var fs = require('fs');

var server = http.createServer(function (req, res) {
  var stream = fs.createReadStream(__dirname + '/data.txt');
  stream.pipe(res);
});
server.listen(8000);
```

.pipefs.createReadStream'data'end'。 data.txt。

“”。

```
var fs = require('fs')

var readable = fs.createReadStream('file1.txt')
var writable = fs.createWriteStream('file2.txt')

readable.pipe(writable) // returns writable
```

。

```
var zlib = require('zlib')

fs.createReadStream('style.css')
  .pipe(zlib.createGzip()) // The returned object, zlib.Gzip, is a duplex stream.
  .pipe(fs.createWriteStream('style.css.gz'))
```

◦

```
var readable = fs.createReadStream('source.css')
readable.pipe(zlib.createGzip()).pipe(fs.createWriteStream('output.css.gz'))
readable.pipe(fs.createWriteStream('output.css'))
```

“” ◦ ◦

error;

```
var readable = fs.createReadStream('file3.txt')
var writable = fs.createWriteStream('file4.txt')
readable.pipe(writable)
readable.on('error', console.error)
writable.on('error', console.error)
```

/

fs

StreamNodeJs

```
var fs = require("fs");
var stream = require("stream").Writable;

/*
 * Implementing the write function in writable stream class.
 * This is the function which will be used when other stream is piped into this
 * writable stream.
 */
stream.prototype._write = function(chunk, data){
  console.log(data);
}

var customStream = new stream();

fs.createReadStream("aml.js").pipe(customStream);
```

◦ write ◦ NodeJs 4.xx NodeJs 6.x **ES6** ◦ 6.JNodeJ

```
const Writable = require('stream').Writable;

class MyWritable extends Writable {
  constructor(options) {
    super(options);
  }

  _write(chunk, encoding, callback) {
```

```
    console.log(chunk);
  }
}
```

Streams

```
fs.readFile(`${__dirname}/utils.js`, (err, data) => {
  if (err) {
    handleError(err);
  } else {
    console.log(data.toString());
  }
})
```

streams

```
var fileStream = fs.createReadStream(`${__dirname}/file`);
var fileContent = '';
fileStream.on('data', data => {
  fileContent += data.toString();
})

fileStream.on('end', () => {
  console.log(fileContent);
})

fileStream.on('error', err => {
  handleError(err)
})
```

◦

-
-

streams 10

streams10 ◦

streams streams - ◦

streams

gzip◦ url

-
-
-

[] [1]s3◦ ◦

```
var startTime = Date.now()
s3.getObject({Bucket: 'some-bucket', Key: 'tweets.gz'}, (err, data) => {
```

```
// here, the whole file was downloaded

zlib.gunzip(data.Body, (err, data) => {
  // here, the whole file was unzipped

  fs.writeFile(`${__dirname}/tweets.json`, data, err => {
    if (err) console.error(err)

    // here, the whole file was written to disk
    var endTime = Date.now()
    console.log(`${endTime - startTime} milliseconds`) // 1339 milliseconds
  })
})

// 1339 milliseconds
```

streams

```
s3.getObject({Bucket: 'some-bucket', Key: 'tweets.gz'}).createReadStream()
  .pipe(zlib.createGunzip())
  .pipe(fs.createWriteStream(`${__dirname}/tweets.json`));

// 1204 milliseconds
```

- 80KB ◦ 71MB **gzip** 382MBstreams

- 71MB**20925**71MB382MB - ◦
- streams**1343435**

Streams <https://riptutorial.com/zh-TW/node-js/topic/2974/streams>

53:

- `child_process.execcommand` [options] [callback]
- `child_process.execFilefile` [args] [options] [callback]
- `child_process.forkmodulePath` [args] [options]
- `child_process.spawncommand` [args] [options]
- `child_process.execFileSyncfile` [args] [options]
- `child_process.execSynccommand` [options]
- `child_process.spawnSynccommand` [args] [options]

`ChildProcess` ◦ `Node.js` ◦

Examples

`child_process.spawn()` ◦

◦ `ChildProcess` `stdout` `stderr` ◦ `stream.Readable` ◦

`ls -lh /usr` ◦

```
const spawn = require('child_process').spawn;
const ls = spawn('ls', ['-lh', '/usr']);

ls.stdout.on('data', (data) => {
  console.log(`stdout: ${data}`);
});

ls.stderr.on('data', (data) => {
  console.log(`stderr: ${data}`);
});

ls.on('close', (code) => {
  console.log(`child process exited with code ${code}`);
});
```

```
zip -0vr "archive" ./image.png
```

```
spawn('zip', ['-0vr', '"archive"', './image.png']);
```

shell

`shell` `child_process.exec` ◦ `cat *.js file | wc -l`

```
const exec = require('child_process').exec;
exec('cat *.js file | wc -l', (err, stdout, stderr) => {
  if (err) {
    console.error(`exec error: ${err}`);
    return;
  }
});
```

```
console.log(`stdout: ${stdout}`);
console.log(`stderr: ${stderr}`);
});
```

```
child_process.exec(command[, options][, callback]);
```

commandoptionscallback ◦ `exec`

```
{
  encoding: 'utf8',
  timeout: 0,
  maxBuffer: 200*1024,
  killSignal: 'SIGTERM',
  cwd: null,
  env: null
}
```

options `shell` `UNIX/bin/sh` `Windowscmd.exe uidgid` ◦

`(err, stdout, stderr)` ◦ `err` `null` `Error` `err.code` `err.signal` ◦

`stdout` `stderr` ◦ **options** `string` `Buffer` ◦

`exec` `execSync` ◦ `stdout` `ChildProcess` ◦ ◦

```
const execSync = require('child_process').execSync;
const stdout = execSync('cat *.js file | wc -l');
console.log(`stdout: ${stdout}`);
```

`child_process.execFile` ◦ `child_process.exec` **shell** ◦

```
const execFile = require('child_process').execFile;
const child = execFile('node', ['--version'], (err, stdout, stderr) => {
  if (err) {
    throw err;
  }

  console.log(stdout);
});
```

`child_process.exec`

```
child_process.execFile(file[, args][, options][, callback]);
```

`child_process.exec` ◦

```
const execFileSync = require('child_process').execFileSync;
const stdout = execFileSync('node', ['--version']);
console.log(stdout);
```

<https://riptutorial.com/zh-TW/node-js/topic/2726/>

54: Node.JSWebSocket

Examples

WebSocket

WebSocket

```
npm install --save ws
```

package.json

```
"dependencies": {  
  "ws": "*"   
},
```

WebSocket

WS

```
var ws = require('ws');
```

WebSocketWebSocket

WebSocket

```
var WebSocket = require("ws");  
var ws = new WebSocket("ws://host:8080/OptionalPathName");  
// Continue on with your code...
```

```
var WebSocketServer = require("ws").Server;  
var wss = new WebSocketServer({port: 8080, path: "OptionalPathName"});
```

WebSocket

```
var WebSocketServer = require('ws').Server  
, wss = new WebSocketServer({ port: 8080 }); // If you want to add a path as well, use path:  
"PathName"  
  
wss.on('connection', function connection(ws) {  
  ws.on('message', function incoming(message) {  
    console.log('received: %s', message);  
  });  
  
  ws.send('something');  
});
```

55:

Examples

- 1.
- 2.
- 3.

DI.

◦

◦

<https://riptutorial.com/zh-TW/node-js/topic/7681/>

56:

Examples

PM2

PM2nodejs◦ PM2◦

PM2nodejs

```
npm install pm2 -g
```

nodejspm2nodejs

```
pm2 start server.js --name "app1"
```

1. pm2nodejs

```
pm2 list
```

```
[tknew:~/Unitech/pm2] master(+84/-121)+* ± pm2 list
```

PM2 Process listing

App Name	id	mode	PID	status	Restarted	Uptime	memory	err logs
bashscript.sh	6	fork	8278	online	0	10s	1.379 MB	/home/tkne
checker	5	cluster	0	stopped	0	2m	0 B	/home/tkne
interface-api	3	cluster	7526	online	0	3m	15.445 MB	/home/tkne
interface-api	2	cluster	7517	online	0	3m	15.453 MB	/home/tkne
interface-api	1	cluster	7512	online	0	3m	15.449 MB	/home/tkne
interface-api	0	cluster	7507	online	0	43m	15.449 MB	/home/tkne

2. nodejs

```
pm2 stop <instance named>
```

3. nodejs

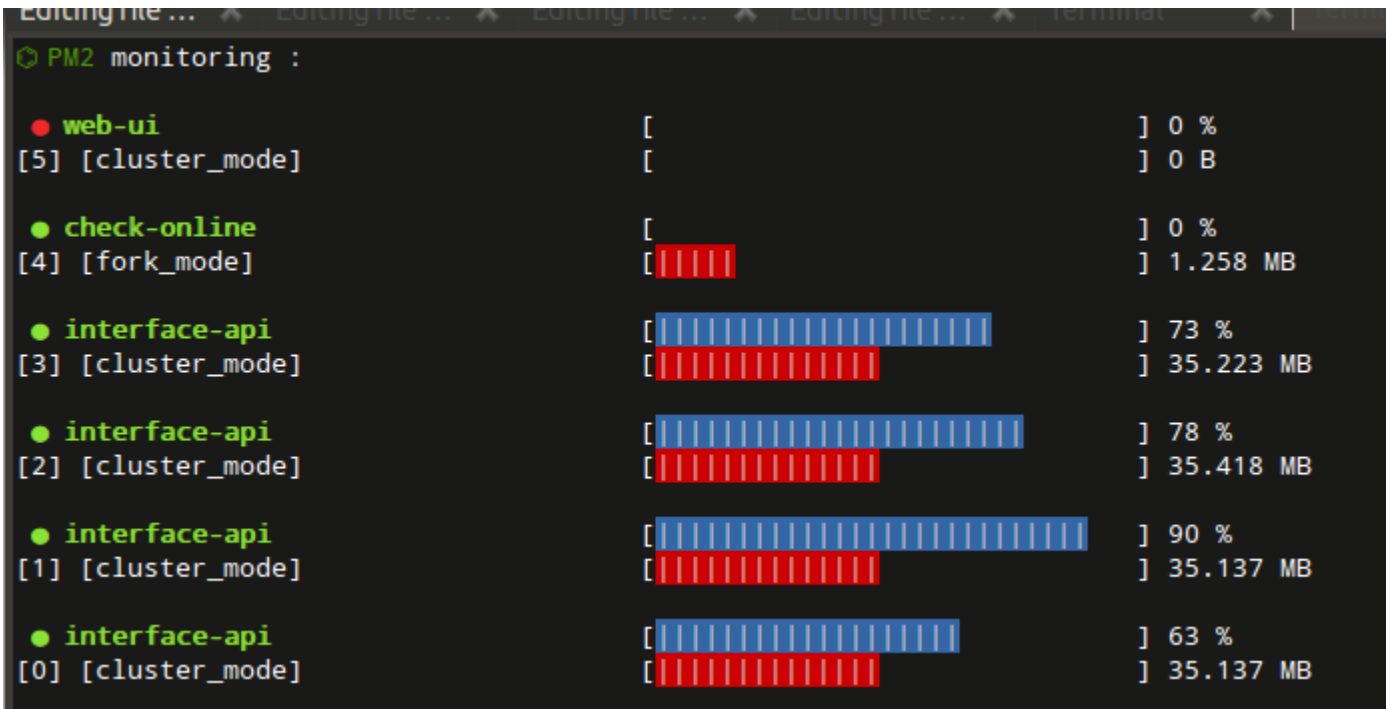
```
pm2 delete <instance name>
```

4. nodejs

```
pm2 restart <instance name>
```

5. nodejs

```
pm2 monit
```



6. pm2

```
pm2 kill
```

7.0

```
pm2 reload <instance name>
```

8. pm2 logs <instance_name>

Forever

```
$ forever start index.js
warn: --minUptime not set. Defaulting to: 1000ms
warn: --spinSleepTime not set. Your script will exit if it does not stay up for at least
1000ms
info: Forever processing file: index.js
```

Forever

```
$ forever list
info: Forever processes running

|data: | index | uid | command | script | forever pid|id | logfile
|uptime |
|-----|-----|-----|-----|-----|-----|-----|-----|
---|-----|
|data: | [0] | f4Kt | /usr/bin/nodejs | src/index.js|2131 |
2146|/root/.forever/f4Kt.log | 0:0:0:11.485 |
```

```
$ forever stop 0
```



```
$ forever stop 2146  
$ forever stop --uid f4Kt  
$ forever stop --pidFile 2131
```

nohup

Linuxnohup

nohup

1. `cd app.js`
 2. `nohup nodejs app.js &`
-
1. `ps -ef|grep nodejs`
 2. `kill -9 <the process number>`

```
npm install forever -g  
cd /node/project/directory
```

```
forever start app.js
```

<https://riptutorial.com/zh-TW/node-js/topic/2820/>

57: Node.js

Examples

CSRF

CSRF/Web。

cookie - 。

csrf。

```
var express = require('express')
var cookieParser = require('cookie-parser') //for cookie parsing
var csrf = require('csrf') //csrf module
var bodyParser = require('body-parser') //for body parsing

// setup route middlewares
var csrfProtection = csrf({ cookie: true })
var parseForm = bodyParser.urlencoded({ extended: false })

// create express app
var app = express()

// parse cookies
app.use(cookieParser())

app.get('/form', csrfProtection, function(req, res) {
  // generate and pass the csrfToken to the view
  res.render('send', { csrfToken: req.csrfToken() })
})

app.post('/process', parseForm, csrfProtection, function(req, res) {
  res.send('data is being processed')
})
```

GET /form **csrf**csrfToken。

csrfToken_csrf。

handlebar

```
<form action="/process" method="POST">
  <input type="hidden" name="_csrf" value="{{csrfToken}}">
  Name: <input type="text" name="name">
  <button type="submit">Submit</button>
</form>
```

jade

```
form(action="/process" method="post")
  input(type="hidden", name="_csrf", value=csrfToken)
```

```
span Name:
input (type="text", name="name", required=true)
br

input (type="submit")
```

ejs

```
<form action="/process" method="POST">
  <input type="hidden" name="_csrf" value="<%= csrfToken %>">
  Name: <input type="text" name="name">
  <button type="submit">Submit</button>
</form>
```

Node.jsSSL / TLS

Node.jsSSL / TLSSSL / TLS。 - SSL / TLS。

Node.jsSSL / TLS。

CAca。 Node.jsca。 1_ca.crt2_ca.crt。 ca。

```
const https = require('https');
const fs = require('fs');

const options = {
  key: fs.readFileSync('privatekey.pem'),
  cert: fs.readFileSync('certificate.pem'),
  ca: [fs.readFileSync('1_ca.crt'), fs.readFileSync('2_ca.crt')]
};

https.createServer(options, (req, res) => {
  res.writeHead(200);
  res.end('hello world\n');
}).listen(8000);
```

HTTPS

Node.jsHTTPS

```
const https = require('https');
const fs = require('fs');

const httpsOptions = {
  key: fs.readFileSync('path/to/server-key.pem'),
  cert: fs.readFileSync('path/to/server-crt.pem')
};

const app = function (req, res) {
  res.writeHead(200);
  res.end("hello world\n");
}

https.createServer(httpsOptions, app).listen(4433);
```

http

```
const http = require('http');
const https = require('https');
const fs = require('fs');

const httpsOptions = {
  key: fs.readFileSync('path/to/server-key.pem'),
  cert: fs.readFileSync('path/to/server-crt.pem')
};

const app = function (req, res) {
  res.writeHead(200);
  res.end("hello world\n");
}

http.createServer(app).listen(8888);
https.createServer(httpsOptions, app).listen(4433);
```

HTTPS

node.jsHTTPHTTPSWeb

1

1. mkdir conf

2. cd conf

3. ca.cnf

```
wget https://raw.githubusercontent.com/anders94/https-authorized-clients/master/keys/ca.cnf
```

4. openssl req -new -x509 -days 9999 -config ca.cnf -keyout ca-key.pem -out ca-cert.pem

5. ca-key.pemca-cert.pem

```
openssl genrsa -out key.pem 4096
```

6. server.cnf

```
wget https://raw.githubusercontent.com/anders94/https-authorized-clients/master/keys/server.cnf
```

7. openssl req -new -config server.cnf -key key.pem -out csr.pem

8. openssl x509 -req -extfile server.cnf -days 999 -passin "pass:password" -in csr.pem -CA ca-cert.pem -CAkey ca-key.pem -CAcreateserial -out cert.pem

2

```
1. sudo cp ca-crt.pem /usr/local/share/ca-certificates/ca-crt.pem
```

2. CA

```
sudo update-ca-certificates
```

Secure express.js 3

express.js3

```
var fs = require('fs');
var http = require('http');
var https = require('https');
var privateKey = fs.readFileSync('sslcert/server.key', 'utf8');
var certificate = fs.readFileSync('sslcert/server.crt', 'utf8');

// Define your key and cert

var credentials = {key: privateKey, cert: certificate};
var express = require('express');
var app = express();

// your express configuration here

var httpServer = http.createServer(app);
var httpsServer = https.createServer(credentials, app);

// Using port 8080 for http and 8443 for https

httpServer.listen(8080);
httpsServer.listen(8443);
```

http / https

1024sudonginxhaproxy。

[Node.js https://riptutorial.com/zh-TW/node-js/topic/3473/node-js](https://riptutorial.com/zh-TW/node-js/topic/3473/node-js)

58:

Examples

- SIGTERM

server.closeprocess.exit ◦

```
var http = require('http');

var server = http.createServer(function (req, res) {
  setTimeout(function () { //simulate a long request
    res.writeHead(200, {'Content-Type': 'text/plain'});
    res.end('Hello World\n');
  }, 4000);
}).listen(9090, function (err) {
  console.log('listening http://localhost:9090/');
  console.log('pid is ' + process.pid);
});

process.on('SIGTERM', function () {
  server.close(function () {
    process.exit(0);
  });
});
```

<https://riptutorial.com/zh-TW/node-js/topic/5996/>

59:

Node.js `require()` ◦ ◦

◦ ◦ ◦

Examples

`require()` `http` ◦ Node.js

```
const http = require('http');
```

`npm install express` ◦ `npm install express`

```
const express = require('express');
```

`lib.js` ◦ `lib.js`

```
const mylib = require('./lib');
```

`.js` ◦ ◦

```
const http = require('http');

// The `http` module has the property `STATUS_CODES`
console.log(http.STATUS_CODES[404]); // outputs 'Not Found'

// Also contains `createServer()`
http.createServer(function(req, res) {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write('<html><body>Module Test</body></html>');
  res.end();
}).listen(80);
```

hello-world.js

Node `module.exports` ◦ ◦

world.js

```
module.exports = function(subject) {
  console.log('Hello ' + subject);
};
```

`exports` ◦

- `hello-venus.js` `module.exports` `module.exports`
- `hello-jupiter.js` `module.exports` `module.exports`

- `hello-mars.js` `exportsmodule.exports`

venus.js

```
function hello(subject) {
  console.log('Venus says Hello ' + subject);
}

module.exports = {
  hello: hello
};
```

jupiter.js

```
module.exports = {
  hello: function(subject) {
    console.log('Jupiter says hello ' + subject);
  },

  bye: function(subject) {
    console.log('Jupiter says goodbye ' + subject);
  }
};
```

mars.js

```
exports.hello = function(subject) {
  console.log('Mars says Hello ' + subject);
};
```

hello

index.js

```
// hello/index.js
module.exports = function(){
  console.log('Hej');
};
```

main.js

```
// hello/main.js
// We can include the other files we've defined by using the `require()` method
var hw = require('./hello-world.js'),
    hm = require('./hello-mars.js'),
    hv = require('./hello-venus.js'),
    hj = require('./hello-jupiter.js'),
    hu = require('./index.js');

// Because we assigned our function to the entire `module.exports` object, we
// can use it directly
hw('World!'); // outputs "Hello World!"

// In this case, we assigned our function to the `hello` property of exports, so we must
// use that here too
```



```
hm.hello('Solar System!'); // outputs "Mars says Hello Solar System!"

// The result of assigning module.exports at once is the same as in hello-world.js
hv.hello('Milky Way!'); // outputs "Venus says Hello Milky Way!"

hj.hello('Universe!'); // outputs "Jupiter says hello Universe!"
hj.bye('Universe!'); // outputs "Jupiter says goodbye Universe!"

hu(); //output 'hej'
```

```
require()° °
```

```
delete°
```

```
var a = require('./a');
```

```
var rpath = require.resolve('./a.js');
delete require.cache[rpath];
```

```
var a = require('./a');
```

```
delete° °
```

```
const auth = module.exports = {}
const config = require('../config')
const request = require('request')

auth.email = function (data, callback) {
  // Authenticate with an email address
}

auth.facebook = function (data, callback) {
  // Authenticate with a Facebook account
}

auth.twitter = function (data, callback) {
  // Authenticate with a Twitter account
}

auth.slack = function (data, callback) {
  // Authenticate with a Slack account
}

auth.stack_overflow = function (data, callback) {
  // Authenticate with a Stack Overflow account
}
```

```
const auth = require('./auth')

module.exports = function (req, res, next) {
  auth.facebook(req.body, function (err, user) {
    if (err) return next(err)

    req.user = user
    next()
  })
}
```

```
  })
}
```

NodeJS。 requireObject。 Noderequire。 。

myModule.js

```
console.log(123) ;
exports.var1 = 4 ;
```

index.js

```
var a=require('./myModule') ; // Output 123
var b=require('./myModule') ; // No output
console.log(a.var1) ; // Output 4
console.log(b.var1) ; // Output 4
a.var2 = 5 ;
console.log(b.var2) ; // Output 5
```

node_modules

require dnode_modules 。

index.js requirefoo

```
index.js
|- node_modules
  |- foo
    |- foo.js
  |- package.json
```

package.json。 package.jsonmain - require('your-module')require('your-module') 。

mainindex.js。 require require('your-module/path/to/file') 。

required node_modules。

```
my-project
|- node_modules
  |- foo // the foo module
  |- ...
|- baz // the baz module
  |- node_modules
  |- bar // the bar module
```

requirefoobarrequire('foo')

/ node_modules。 Node。

npmnpm。

.js。 my_module

function_one.js

```
module.exports = function() {  
  return 1;  
}
```

function_two.js

```
module.exports = function() {  
  return 2;  
}
```

index.js

```
exports.f_one = require('./function_one.js');  
exports.f_two = require('./function_two.js');
```

```
var split_module = require('./my_module');
```

`./requireNode``node_modules`。

package.json

```
{  
  "name": "my_module",  
  "main": "./your_main_entry_point.js"  
}
```

“index”。

<https://riptutorial.com/zh-TW/node-js/topic/547/>

60: PromiseError-FirstNode.js

promises/async / await。 bluebirdasCallbackQnodeify。

Examples

Bluebird

math.js

```
'use strict';

const Promise = require('bluebird');

module.exports = {

  // example of a callback-only method
  callbackSum: function(a, b, callback) {
    if (typeof a !== 'number')
      return callback(new Error('"a" must be a number'));
    if (typeof b !== 'number')
      return callback(new Error('"b" must be a number'));

    return callback(null, a + b);
  },

  // example of a promise-only method
  promiseSum: function(a, b) {
    return new Promise(function(resolve, reject) {
      if (typeof a !== 'number')
        return reject(new Error('"a" must be a number'));
      if (typeof b !== 'number')
        return reject(new Error('"b" must be a number'));
      resolve(a + b);
    });
  },

  // a method that can be used as a promise or with callbacks
  sum: function(a, b, callback) {
    return new Promise(function(resolve, reject) {
      if (typeof a !== 'number')
        return reject(new Error('"a" must be a number'));
      if (typeof b !== 'number')
        return reject(new Error('"b" must be a number'));
      resolve(a + b);
    }).asCallback(callback);
  },

};
```

index.js

```
'use strict';
```

```

const math = require('./math');

// classic callbacks

math.callbackSum(1, 3, function(err, result) {
  if (err)
    console.log('Test 1: ' + err);
  else
    console.log('Test 1: the answer is ' + result);
});

math.callbackSum(1, 'd', function(err, result) {
  if (err)
    console.log('Test 2: ' + err);
  else
    console.log('Test 2: the answer is ' + result);
});

// promises

math.promiseSum(2, 5)
  .then(function(result) {
    console.log('Test 3: the answer is ' + result);
  })
  .catch(function(err) {
    console.log('Test 3: ' + err);
  });

math.promiseSum(1)
  .then(function(result) {
    console.log('Test 4: the answer is ' + result);
  })
  .catch(function(err) {
    console.log('Test 4: ' + err);
  });

// promise/callback method used like a promise

math.sum(8, 2)
  .then(function(result) {
    console.log('Test 5: the answer is ' + result);
  })
  .catch(function(err) {
    console.log('Test 5: ' + err);
  });

// promise/callback method used with callbacks

math.sum(7, 11, function(err, result) {
  if (err)
    console.log('Test 6: ' + err);
  else
    console.log('Test 6: the answer is ' + result);
});

// promise/callback method used like a promise with async/await syntax

```

```
(async () => {  
  
  try {  
    let x = await math.sum(6, 3);  
    console.log('Test 7a: ' + x);  
  
    let y = await math.sum(4, 's');  
    console.log('Test 7b: ' + y);  
  
  } catch(err) {  
    console.log(err.message);  
  }  
  
})();
```

[PromiseError-FirstNode.js](https://riptutorial.com/zh-TW/node-js/topic/9874/promiseerror-firstnode-js) <https://riptutorial.com/zh-TW/node-js/topic/9874/promiseerror-firstnode-js>

61:

Examples

require

```
require.extensions.require()◦
```

XML

```
// Add .xml for require()
require.extensions['.xml'] = (module, filename) => {
  const fs = require('fs')
  const xml2js = require('xml2js')

  module.exports = (callback) => {
    // Read required file.
    fs.readFile(filename, 'utf8', (err, data) => {
      if (err) {
        callback(err)
        return
      }
      // Parse it.
      xml2js.parseString(data, (err, result) => {
        callback(null, result)
      })
    })
  }
}
```

```
hello.xml
```

```
<?xml version="1.0" encoding="UTF-8"?>
<foo>
  <bar>baz</bar>
  <qux />
</foo>
```

```
require()
```

```
require('./hello')((err, xml) {
  if (err)
    throw err;
  console.log(err);
})
```

```
{ foo: { bar: [ 'baz' ], qux: [ '' ] } }◦
```

<https://riptutorial.com/zh-TW/node-js/topic/6645/>

62:

Examples

CassandraDataStax [cassandra-driver](#) ◦ ◦

```
const cassandra = require("cassandra-driver");
const clientOptions = {
  contactPoints: ["host1", "host2"],
  keyspace: "test"
};

const client = new cassandra.Client(clientOptions);

const query = "SELECT hello FROM world WHERE name = ?";
client.execute(query, ["John"], (err, results) => {
  if (err) {
    return console.error(err);
  }

  console.log(results.rows);
});
```

<https://riptutorial.com/zh-TW/node-js/topic/5949/>

63: Node.js.

Examples

Mac OSXNode.js.

Mac2

```
lsbom -f -l -s -pf /var/db/receipts/org.nodejs.pkg.bom | while read f; do sudo rm /usr/local/${f}; done

sudo rm -rf /usr/local/lib/node /usr/local/lib/node_modules /var/db/receipts/org.nodejs.*
```

WindowsNode.js.

WindowsNode.js“”

1. “Add or Remove Programs”。
2. Node.js

Windows 10

3. Node.js.
4. “”。
5. 。

Windows 7-8.1

3. Node.js。

Node.js. <https://riptutorial.com/zh-TW/node-js/topic/2821/node-js->

64:

“grunt”Gruntgrunt-cli。

“”Gruntfile。

“”Grunt。

Examples

GruntJs

GruntJavaScriptlinting。

GruntCLI。

```
npm install -g grunt-cli
```

Gruntpackage.jsonGruntfile。

package.jsonnpmnpm。 gruntGruntdevDependencies。

GruntfileGruntfile.jsGrunt。

Example package.json:

```
{
  "name": "my-project-name",
  "version": "0.1.0",
  "devDependencies": {
    "grunt": "~0.4.5",
    "grunt-contrib-jshint": "~0.10.0",
    "grunt-contrib-nodeunit": "~0.4.1",
    "grunt-contrib-uglify": "~0.5.0"
  }
}
```

gruntfile

```
module.exports = function(grunt) {

  // Project configuration.
  grunt.initConfig({
    pkg: grunt.file.readJSON('package.json'),
    uglify: {
      options: {
        banner: '/*! <%= pkg.name %> <%= grunt.template.today("yyyy-mm-dd") %> */\n'
      },
      build: {
        src: 'src/<%= pkg.name %>.js',
        dest: 'build/<%= pkg.name %>.min.js'
      }
    }
  });
};
```

```
    }  
  }  
});  
  
// Load the plugin that provides the "uglify" task.  
grunt.loadNpmTasks('grunt-contrib-uglify');  
  
// Default task(s).  
grunt.registerTask('default', ['uglify']);  
  
};
```

gruntplugins

gruntplugin◦ jshint◦

```
npm install grunt-contrib-jshint --save-dev
```

--save-devpackage.jsonnpm install ◦

loadNpmTasks**gruntfile**◦

```
grunt.loadNpmTasks('grunt-contrib-jshint');
```

gruntfilejshintgrunt.initConfiggrunt.initConfig ◦

```
grunt.initConfig({  
  jshint: {  
    all: ['Gruntfile.js', 'lib/**/*.js', 'test/**/*.js']  
  }  
});
```

◦

◦

```
grunt jshint
```

jshint◦

```
grunt.registerTask('default', ['jshint']);
```

grunt◦

<https://riptutorial.com/zh-TW/node-js/topic/6059/>

65:

Examples

```
describe('Suite Name', function() {
  describe('#method()', function() {
    it('should run without an error', function() {
      expect([ 1, 2, 3 ].length).to.be.equal(3)
    })
  })
})
```

```
var expect = require("chai").expect;
describe('Suite Name', function() {
  describe('#method()', function() {
    it('should run without an error', function(done) {
      testSomething(err => {
        expect(err).to.not.be.equal(null)
        done()
      })
    })
  })
})
```

Promise

```
describe('Suite Name', function() {
  describe('#method()', function() {
    it('should run without an error', function() {
      return doSomething().then(result => {
        expect(result).to.be.equal('hello world')
      })
    })
  })
})
```

/

```
const { expect } = require('chai')

describe('Suite Name', function() {
  describe('#method()', function() {
    it('should run without an error', async function() {
      const result = await answerToTheUltimateQuestion()
      expect(result).to.be.equal(42)
    })
  })
})
```

<https://riptutorial.com/zh-TW/node-js/topic/6731/>

66:

Examples

```
db.notification.email.find({subject: 'promisify callback'}, (error, result) => {
  if (error) {
    console.log(error);
  }

  // normal code here
});
```

bluebirdpromisifyAll。 bluebirdpromisepromiseAsync

```
let email = bluebird.promisifyAll(db.notification.email);

email.findAsync({subject: 'promisify callback'}).then(result => {

  // normal code here
})
.catch(console.error);
```

promisify

```
let find = bluebird.promisify(db.notification.email.find);

find({locationId: 168}).then(result => {

  // normal code here
});
.catch(console.error);
```

MassiveJS。 context。

```
let find = bluebird.promisify(db.notification.email.find, { context: db.notification.email });

find({locationId: 168}).then(result => {

  // normal code here
});
.catch(console.error);
```

◦ promisify

fs.exists

```
var fs = require('fs');

var existsAsync = function(path) {
  return new Promise(function(resolve, reject) {
    fs.exists(path, function(exists) {
```

```
// exists is a boolean
if (exists) {
  // Resolve successfully
  resolve();
} else {
  // Reject with error
  reject(new Error('path does not exist'));
}
});

// Use as a promise now
existsAsync('/path/to/some/file').then(function() {
  console.log('file exists!');
}).catch(function(err) {
  // file does not exist
  console.error(err);
});
```

setTimeout promisified

```
function wait(ms) {
  return new Promise(function (resolve, reject) {
    setTimeout(resolve, ms)
  })
}
```

<https://riptutorial.com/zh-TW/node-js/topic/2346/>

67: node.js

Examples

node.js

node.js

◦ ◦

◦

printer.js

```
"use strict";

exports.printHelloWorld = function () {
  console.log("Hello World!!!");
}
```

animals.js

```
"use strict";

module.exports = {
  lion: function() {
    console.log("ROAARR!!!");
  }
};
```

app.js

node app.js

```
"use strict";

//require('./path/to/module.js') node which module to load
var printer = require('./printer');
var animals = require('./animals');

printer.printHelloWorld(); //prints "Hello World!!!"
animals.lion(); //prints "ROAARR!!!"
```

ES6

Node.jsV8。 Node.jsJavaScript ECMA-262。

ECMAScript 2015ES6

V8Node.js。 V8 - ◦ ◦ V8。

ES6import

fun.js.....

```
export default function say(what){
  console.log(what);
}

export function sayLoud(whoot) {
  say(whoot.toUpperCase());
}
```

...app.js

```
import say from './fun';
say('Hello Stack Overflow!!'); // Output: Hello Stack Overflow!!
```

say() export default ...

```
import { sayLoud } from './fun';
sayLoud('JS modules are awesome.');
```

import sayLoud sayLoud

```
import * as i from './fun';
i.say('What?'); // Output: What?
i.sayLoud('Whoot!'); // Output: WHOOT!
```

* as i import i fun

./

```
import express from 'express';
```

node_modules

ES6

ES6

```
export function printHelloWorld() {
  console.log("Hello World!!!");
}
```

[node.js https://riptutorial.com/zh-TW/node-js/topic/1173/node-js](https://riptutorial.com/zh-TW/node-js/topic/1173/node-js)

68: Node.js

Examples

`NODE_ENV=""`

`NODE_ENV "production"` ◦

`NODE_ENV`

```
if(process.env.NODE_ENV === 'production') {  
  // We are running in production mode  
} else {  
  // We are running in development mode  
}
```

`NODE_ENV 'production' npm install package.json devDependencies ◦ --production`

```
npm install --production
```

`NODE_ENV`

1 NODE_ENV

Windows

```
set NODE_ENV=production
```

Linux/unix

```
export NODE_ENV=production
```

`NODE_ENV bash NODE_ENV NODE_ENV production ◦`

2 NODE_ENV

```
NODE_ENV=production node app.js
```

`NODE_ENV NODE_ENV ◦`

3 .env

◦ ◦

`.env bash ◦`

bashenv-cmd.env

```
env-cmd .env node app.js
```

4cross-env

o

npmpackage.json

```
"build:deploy": "cross-env NODE_ENV=production webpack"
```

NodeJS o PM2 o PM2

PM2

PM2

```
npm install pm2 -g
```

```
pm2 start app.js -i 0 --name "api" -i 0CPU
```

PM2 pm2PM2configpm2 o

```
PM2_HOME=/etc/.pm2 pm2 start app.js
```

PM2

PM2Node.js PM2

pm2 o

```
npm install -g pm2
```

PM2.node.jsPM2.

```
pm2 start server.js --name "my-app"
```

```
$ pm2 start app.js --name my-app  
[PM2] restartProcessId process id 0
```

App name	id	mode	pid	status	restart	uptime	memory	watching
my-app	0	fork	64029	online	1	0s	17.816 MB	disabled

Use the `pm2 show <id|name>` command to get more details about an app.

PM2◦

```
pm2 list
```

```
pm2 stop my-app
```

```
pm2 restart my-app
```

```
pm2 show my-app
```

PM2

```
pm2 delete my-app
```

nodejs◦ ◦

pm2◦

Forever

[forever](#)◦ foreverNode.js◦

forever◦

forever◦

```
$ npm install -g forever
```

```
$ forever start server.js
```

id◦

```
$ forever restart 0
```

oid◦

```
$ forever stop 0
```

restart oid◦ IDid◦

<https://www.npmjs.com/package/forever>

/devqastaging◦

◦ NodeJs◦

- dev.json

```
{
  "PORT": 3000,
  "DB": {
    "host": "localhost",
    "user": "bob",
    "password": "12345"
  }
}
```

- qa.json

```
{
  "PORT": 3001,
  "DB": {
    "host": "where_db_is_hosted",
    "user": "bob",
    "password": "54321"
  }
}
```

```
process.argv.forEach(function (val) {
  var arg = val.split("=");
  if (arg.length > 0) {
    if (arg[0] === 'env') {
      var env = require('./' + arg[1] + '.json');
      exports.prop = env;
    }
  }
});
```

```
node app.js env=dev
```

```
forever start app.js env=dev
```

Node.js。 Node.js。

```
var cluster = require('cluster');

var numCPUs = require('os').cpus().length;

if (cluster.isMaster) {
  // In real life, you'd probably use more than just 2 workers,
  // and perhaps not put the master and worker in the same file.
```

```
//  
// You can also of course get a bit fancier about logging, and  
// implement whatever custom logic you need to prevent DoS  
// attacks and other bad behavior.  
//  
// See the options in the cluster documentation.  
//  
// The important thing is that the master does very little,  
// increasing our resilience to unexpected errors.  
console.log('your server is working on ' + numCPUs + ' cores');  
  
for (var i = 0; i < numCPUs; i++) {  
    cluster.fork();  
}  
  
cluster.on('disconnect', function(worker) {  
    console.error('disconnect!');  
    //clearTimeout(timeout);  
    cluster.fork();  
});  
  
} else {  
    require('./app.js');  
}
```

Node.js <https://riptutorial.com/zh-TW/node-js/topic/2975/node-js>

69: RESTCRUD API

Examples

Express 3+CRUDREST API

```
var express = require("express"),
    bodyParser = require("body-parser"),
    server = express();

//body parser for parsing request body
server.use(bodyParser.json());
server.use(bodyParser.urlencoded({ extended: true }));

//temporary store for `item` in memory
var itemStore = [];

//GET all items
server.get('/item', function (req, res) {
  res.json(itemStore);
});

//GET the item with specified id
server.get('/item/:id', function (req, res) {
  res.json(itemStore[req.params.id]);
});

//POST new item
server.post('/item', function (req, res) {
  itemStore.push(req.body);
  res.json(req.body);
});

//PUT edited item in-place of item with specified id
server.put('/item/:id', function (req, res) {
  itemStore[req.params.id] = req.body;
  res.json(req.body);
});

//DELETE item with specified id
server.delete('/item/:id', function (req, res) {
  itemStore.splice(req.params.id, 1);
  res.json(req.body);
});

//START SERVER
server.listen(3000, function () {
  console.log("Server running");
});
```

RESTCRUD API <https://riptutorial.com/zh-TW/node-js/topic/5850/restcrud-api>

70:

Node.js。 Node。

Node.js。 I/O。 。

。

Examples

cluster。

。 。 RamEvent Loop。

/ API。 。 RedisCookie。

```
// runs in each instance
var cluster = require('cluster');
var numCPUs = require('os').cpus().length;

console.log('I am always called');

if (cluster.isMaster) {
  // runs only once (within the master);
  console.log('I am the master, launching workers!');
  for(var i = 0; i < numCPUs; i++) cluster.fork();
} else {
  // runs in each fork
  console.log('I am a fork!');

  // here one could start, as an example, a web server
}

console.log('I am always called as well');
```

。 forks child_process.forks。

。

../ parent.js

```
var child_process = require('child_process');
console.log('[Parent]', 'inititalize');

var child1 = child_process.fork(__dirname + '/child');
child1.on('message', function(msg) {
  console.log('[Parent]', 'Answer from child: ', msg);
});

// one can send as many messages as one want
```

```
child1.send('Hello'); // Hello to you too :)
child1.send('Hello'); // Hello to you too :)

// one can also have multiple children
var child2 = child_process.fork(__dirname + '/child');
```

../child.js

```
// here would one initialize this child
// this will be executed only once
console.log('[Child]', 'inititalize');

// here one listens for new tasks from the parent
process.on('message', function(messageFromParent) {

    //do some intense work here
    console.log('[Child]', 'Child doing some intense work');

    if(messageFromParent == 'Hello') process.send('Hello to you too :');
    else process.send('what?');

})
```

“”“”“”。

。。

<https://riptutorial.com/zh-TW/node-js/topic/10592/>

71:

Examples

NodeNPM/usr/local/lib/node_modules ◦ shellNPM/usr/local/lib/node_modules/express **sax**

```
$ npm install -g express
```

◦

◦ **Npm**node_modules/home/user/apps/my_appnode_modules /home/user/apps/my_app/node_modules
/home/user/apps/my_app/node_modules

node_module**Node**node_modules◦ **Node**./node_modules/myModule.js

```
var myModule = require('myModule.js');
```

Node./node_modules/myModule.js◦ ◦

.js.js◦

```
var myModule = require('./myModuleDir');
```

Node◦ **Node**◦ package.json◦ package.jsonindex.js**Node**./myModuleDir/index.js

./myModuleDir/index.js ◦

◦

<https://riptutorial.com/zh-TW/node-js/topic/7738/>

72: Node.js.

Examples

UbuntuNode.js.

apt

```
sudo apt-get update
sudo apt-get install nodejs
sudo apt-get install npm
sudo ln -s /usr/bin/nodejs /usr/bin/node

# the node & npm versions in apt are outdated. This is how you can update them:
sudo npm install -g npm
sudo npm install -g n
sudo n stable # (or lts, or a specific version)
```

nodesourceLTS 6.x

```
curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -
apt-get install -y nodejs
```

npmsudoEACCES

```
mkdir ~/.npm-global
echo "export PATH=~/.npm-global/bin:$PATH" >> ~/.profile
source ~/.profile
npm config set prefix '~/.npm-global'
```

WindowsNode.js.

Node.js^o

node.exeWindows .msi npm Node.js^o

Chocolatey^o

```
# choco install nodejs.install
```

choco^o

nvm

`nvmbashNode.js`。

`nvm`

```
$ curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.31.3/install.sh | bash
```

Windows `nvm-windows`。 [GitHub nvm-windows](#)。

`nvm`“`nvm on`”。 `nvm`。

`nvm`。

Node

```
$ nvm install node
```

/Node

```
$ nvm install 6  
$ nvm install 4.2
```

```
$ nvm ls-remote
```

```
$ nvm use 5
```

Node

```
$ nvm alias default 4.2
```

```
$ nvm ls
```

`.nvmrc`。 ◦

```
$ echo "4.2" > .nvmrc  
$ nvm use  
Found '/path/to/project/.nvmrc' with version <4.2>  
Now using node v4.2 (npm v3.7.3)
```

`nvmNode``sudo home`。 `npm i -g http-server`。

APTSourceNode.js

```
sudo apt-get install build-essential  
sudo apt-get install python  
  
[optional]  
sudo apt-get install git
```

```
cd ~
git clone https://github.com/nodejs/node.git
```

LTS Node.js 6.10.2

```
cd ~
wget https://nodejs.org/dist/v6.3.0/node-v6.10.2.tar.gz
tar -xvzf node-v6.10.2.tar.gz
```

```
cd ~/node-v6.10.2
```

```
./configure
make
sudo make install
```

MacNode.js.

[Homebrew](#) Node.js.

brew

```
brew update
```

o

```
brew doctor
```

Node.js

```
brew install node
```

Node.js

```
node -v
```

MacPorts

[Macports](#) node.js.

```
sudo port selfupdate
```

nodejsnpm

```
sudo port install nodejs npm
```

nodeCLI node ◦

```
node -v
```

MacOS X Installer

[Node.js](#) ◦ [Node.js LTS](#) ◦ [Node LTS Macintosh Installer](#) ◦

NodeJS ◦ .pkg ◦

.pkg **ofcourse** ◦ Node.js npm “Installation Type” customize customize ◦ ◦

terminal [wikihow](#) ◦ node --version ◦ **Node**

```
$ node --version  
v7.2.1
```

v7.2.1 **Node.js** command not found: node ◦

Raspberry Pi Node.js.

v6.x

```
curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -
```

apt

```
sudo apt-get install -y nodejs
```

nvm **Node.js** sudo npm install ... ◦ [Fish Shell](#) fish “OS X Linux shell” bash shell ◦ [Oh My Fish](#) omf
Fish shell ◦

Fish shell ◦

nvm

```
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.31.4/install.sh | bash
```

```
curl -L https://github.com/oh-my-fish/oh-my-fish/raw/master/bin/install | fish
```

◦ ◦

Oh My Fish plugin-nvm

Oh My Fish [plugin-nvm](#) Fish shell nvm

```
omf install nvm
```

Node.js.

nvm ◦ Node.js◦

- Node nvm install node
- 6.3.1 nvm install 6.3.1
- nvm ls nvm ls
- 4.3.1 nvm use 4.3.1

Node.js sudo ◦

Centos RHEL Fedora Node.js.

-
- clang clang++ 3.4 ^ gcc gcc++ 4.8 ^
- Python 2.6 2.7
- GNU Make 3.81 ^

Node.js v6.x LTS

```
git clone -b v6.x https://github.com/nodejs/node.git
```

Node.js v7.x

```
git clone -b v7.x https://github.com/nodejs/node.git
```

```
cd node
./configure
make -jX
su -c make install
```

X -

□

```
cd
rm -rf node
```

nNode.js.

n◦

```
curl -L https://git.io/n-install | bash
```

n◦

n latest

n stable

LTS

```
n lts  
n <version>  
n 4.4.7  
.  
n° Enter°
```

Node.js. <https://riptutorial.com/zh-TW/node-js/topic/1294/node-js->

73: -

Examples

/ w ExpressjQueryJade

```
//'client.jade'  
  
//a button is placed down; similar in HTML  
button(type='button', id='send_by_button') Modify data  
  
#modify Lorem ipsum Sender  
  
//loading jQuery; it can be done from an online source as well  
script(src='./js/jquery-2.2.0.min.js')  
  
//AJAX request using jQuery  
script  
  $(function () {  
    $('#send_by_button').click(function (e) {  
      e.preventDefault();  
  
      //test: the text within brackets should appear when clicking on said button  
      //window.alert('You clicked on me. - jQuery');  
  
      //a variable and a JSON initialized in the code  
      var predeclared = "Katamori";  
      var data = {  
        Title: "Name_SenderTest",  
        Nick: predeclared,  
        FirstName: "Zoltan",  
        Surname: "Schmidt"  
      };  
  
      //an AJAX request with given parameters  
      $.ajax({  
        type: 'POST',  
        data: JSON.stringify(data),  
        contentType: 'application/json',  
        url: 'http://localhost:7776/domaintest',  
  
        //on success, received data is used as 'data' function input  
        success: function (data) {  
          window.alert('Request sent; data received.');  
          var jsonstr = JSON.stringify(data);  
          var jsonobj = JSON.parse(jsonstr);  
  
          //if the 'nick' member of the JSON does not equal to the predeclared  
          string (as it was initialized), then the backend script was executed, meaning that  
          communication has been established  
          if(data.Nick != predeclared){  
            document.getElementById("modify").innerHTML = "JSON changed!\n" +  
jsonstr;  
          }  
        }  
      });  
    }  
  });  
}
```



```
        });
    });
});

//'domaintest_route.js'

var express = require('express');
var router = express.Router();

//an Express router listening to GET requests - in this case, it's empty, meaning that nothing
is displayed when you reach 'localhost/domaintest'
router.get('/', function(req, res, next) {
});

//same for POST requests - notice, how the AJAX request above was defined as POST
router.post('/', function(req, res) {
    res.setHeader('Content-Type', 'application/json');

    //content generated here
    var some_json = {
        Title: "Test",
        Item: "Crate"
    };

    var result = JSON.stringify(some_json);

    //content got 'client.jade'
    var sent_data = req.body;
    sent_data.Nick = "ttony33";

    res.send(sent_data);
});

module.exports = router;
```

// <https://gist.github.com/Katamori/5c9850f02e4baf6e9896>

- <https://riptutorial.com/zh-TW/node-js/topic/6222/--->

74: node.js

WebNode ◦ init ◦

Examples

Node.js systemd

systemdLinux init ◦ Node systemd service ◦

◦ Debian /etc/systemd/system/node.service

```
[Unit]
Description=My super nodejs app

[Service]
# set the working directory to have consistent relative paths
WorkingDirectory=/var/www/app

# start the server file (file is relative to WorkingDirectory here)
ExecStart=/usr/bin/node serverCluster.js

# if process crashes, always try to restart
Restart=always

# let 500ms between the crash and the restart
RestartSec=500ms

# send log to syslog here (it doesn't compete with other log config in the app itself)
StandardOutput=syslog
StandardError=syslog

# nodejs process name in syslog
SyslogIdentifier=nodejs

# user and group starting the app
User=www-data
Group=www-data

# set the environment (dev, prod...)
Environment=NODE_ENV=production

[Install]
# start node at multi user system level (= sysVinit runlevel 3)
WantedBy=multi-user.target
```

```
service node start
service node stop
service node restart
```

systemd systemctl enable node ◦

◦

node.js <https://riptutorial.com/zh-TW/node-js/topic/9258/node-js>

75:

Examples

fs

VOD。。

```
var movie = path.resolve('./public/' + req.params.filename);

fs.stat(movie, function (err, stats) {

  var range = req.headers.range;

  if (!range) {

    return res.sendStatus(416);

  }

  //Chunk logic here
  var positions = range.replace(/bytes=/, "").split("-");
  var start = parseInt(positions[0], 10);
  var total = stats.size;
  var end = positions[1] ? parseInt(positions[1], 10) : total - 1;
  var chunksize = (end - start) + 1;

  res.writeHead(206, {

    'Transfer-Encoding': 'chunked',

    "Content-Range": "bytes " + start + "-" + end + "/" + total,

    "Accept-Ranges": "bytes",

    "Content-Length": chunksize,

    "Content-Type": mime.lookup(req.params.filename)

  });

  var stream = fs.createReadStream(movie, { start: start, end: end, autoClose: true
})

  .on('end', function () {

    console.log('Stream Done');

  })

  .on("error", function (err) {

    res.end(err);

  })
})
```

```
.pipe(res, { end: true });  
  
});
```

◦ ◦ ◦

.pipe(node.js)

fluent-ffmpeg

fluent-ffmpeg.mp4.flv

res.contentType'FLV';

```
var pathToMovie = './public/' + req.params.filename;  
  
var proc = ffmpeg(pathToMovie)  
  
  .preset('flashvideo')  
  
  .on('end', function () {  
  
    console.log('Stream Done');  
  
  })  
  
  .on('error', function (err) {  
  
    console.log('an error happened: ' + err.message);  
  
    res.send(err.message);  
  
  })  
  
  .pipe(res, { end: true });
```

<https://riptutorial.com/zh-TW/node-js/topic/6994/>

76: angular.jsNode.jsexpress.js

AngularJS。

Examples

。

```
mkdir our_project
cd our_project
```

。

```
npm install -g express express-generator
```

LinuxMacsudo**nodejsroot**。

```
express
```

```
bin/
public/
routes/
views/
app.js
package.json
```

npm[http// localhost3000](http://localhost3000)AngularJS 。

ExpressNodejs [Express](#)。 **Express** [http// localhost3000 / home](http://localhost3000/)。

```
FILE: routes/index.js
var express = require('express');
var router = express.Router();

/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});

module.exports = router;
```

[http// localhost3000](http://localhost3000/)**titleExpressJSON** 。

viewsindex.jade

```
extends layout
block content
  h1= title
  p Welcome to #{title}
```

Express ◦ **.jade JadePug**◦

PugExpress◦

Pug

```
npm install --save pug
```

Pugpackage.json ◦ **app.js**

```
var app = express();
// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'pug');
```

◦ **npm start**◦

AngularJS

AngularJSJavascript **MVW** Model-View-Whatever**SPA** Simple Page Application[AngularJSv1.6.4](#)

◦

AngularJS**public / javascriptscssjavacript**◦ **app.js**◦ **ng-app.js**javascriptsAngularJS◦ AngularJS**views / layout.pug**

```
doctype html
html (ng-app='first-app')
  head
    title= title
    link (rel='stylesheet', href='/stylesheets/style.css')
  body (ng-controller='indexController')
    block content

    script (type='text-javascript', src='javascripts/angular.min.js')
    script (type='text-javascript', src='javascripts/ng-app.js')
```

AngularJS**ng-app.js**AngularJS**ng-app**AngularJS◦

ng-app.js

```
angular.module('first-app', [])
  .controller('indexController', ['$scope', indexController]);

function indexController($scope) {
  $scope.name = 'sigfried';
}
```

AngularJS [GoogleStackOverflow](#)◦

AngularJSindex.pug

```
extends layout
block content
  div (ng-controller='indexController')
    h1= title
    p Welcome {{name}}
    input (type='text' ng-model='name')
```

AngularJS

```
$scope.name = 'sigfried';
```

{{name}} AngularJS。

npm start[http:// localhost3000](http://localhost:3000)AngularJS。

[angular.js](#)[Node.js](#)[express.js](#) <https://riptutorial.com/zh-TW/node-js/topic/9757/angular-jsnode-js-express-js->

77: Node.js ECMAScript 2015 ES6

Examples

const / let

var const / let

```
{
  var x = 1 // will escape the scope
  let y = 2 // bound to lexical scope
  const z = 3 // bound to lexical scope, constant
}

console.log(x) // 1
console.log(y) // ReferenceError: y is not defined
console.log(z) // ReferenceError: z is not defined
```

RunKit

“this”

```
performSomething(result => {
  this.someVariable = result
})
```

VS

```
performSomething(function(result) {
  this.someVariable = result
}).bind(this)
```

3,57

```
let nums = [3, 5, 7]
let squares = nums.map(function (n) {
  return n * n
})
console.log(squares)
```

RunKit

.map function **arrow** =>

```
let nums = [3, 5, 7]
let squares = nums.map((n) => {
  return n * n
})
console.log(squares)
```

RunKit

◦ return◦

```
let nums = [3, 5, 7]
let squares = nums.map(n => n * n)
console.log(squares)
```

RunKit

```
let [x,y, ...nums] = [0, 1, 2, 3, 4, 5, 6];
console.log(x, y, nums);
```

```
let {a, b, ...props} = {a:1, b:2, c:3, d:{e:4}}
console.log(a, b, props);
```

```
let dog = {name: 'fido', age: 3};
let {name:n, age} = dog;
console.log(n, age);
```

```
/* @flow */
```

```
function product(a: number, b: number){
  return a * b;
}
```

```
const b = 3;
let c = [1,2,3,,{}];
let d = 3;
```

```
import request from 'request';
```

```
request('http://dev.markitondemand.com/MODApis/Api/v2/Quote/json?symbol=AAPL', (err, res,
payload)=>{
  payload = JSON.parse(payload);
  let {LastPrice} = payload;
  console.log(LastPrice);
});
```

ES6

```
class Mammel {
  constructor(legs) {
    this.legs = legs;
  }
  eat() {
    console.log('eating...');
  }
  static count() {
    console.log('static count...');
  }
}
```

```
class Dog extends Mammel {
  constructor(name, legs) {
    super(legs);
  }
}
```

```
    this.name = name;
  }
  sleep(){
    super.eat();
    console.log('sleeping');
  }
}

let d = new Dog('fido', 4);
d.sleep();
d.eat();
console.log('d', d);
```

[Node.jsECMAScript 2015ES6](https://riptutorial.com/zh-TW/node-js/topic/6732/node-jsecmascript-2015-es6) [https://riptutorial.com/zh-TW/node-js/topic/6732/node-jsecmascript-2015-es6-](https://riptutorial.com/zh-TW/node-js/topic/6732/node-jsecmascript-2015-es6)

78:

Examples

Node

Node.

“” jQuery.

1. .
2. ID.
3. ID.

```
project
├── package.json
├── index.html
├── js
│   ├── main.js
│   └── jquery-1.12.0.min.js
└── srv
    ├── app.js
    ├── models
    │   └── task.js
    └── tasks
        └── data-processor.js
```

app.js

```
var express      = require('express');
var app          = express();
var http         = require('http').Server(app);
var mongoose     = require('mongoose');
var bodyParser   = require('body-parser');

var childProcess= require('child_process');

var Task         = require('./models/task');

app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());

app.use(express.static(__dirname + '/../'));

app.get('/', function(request, response){
    response.render('index.html');
});

//route for the request itself
app.post('/long-running-request', function(request, response){
    //create new task item for status tracking
```

```

var t = new Task({ status: 'Starting ...' });

t.save(function(err, task){
  //create new instance of node for running separate task in another thread
  taskProcessor = childProcess.fork('./srv/tasks/data-processor.js');

  //process the messages coming from the task processor
  taskProcessor.on('message', function(msg){
    task.status = msg.status;
    task.save();
  }).bind(this));

  //remove previously opened node instance when we finished
  taskProcessor.on('close', function(msg){
    this.kill();
  });

  //send some params to our separate task
  var params = {
    message: 'Hello from main thread'
  };

  taskProcessor.send(params);
  response.status(200).json(task);
});

//route to check is the request is finished the calculations
app.post('/is-ready', function(request, response){
  Task
    .findById(request.body.id)
    .exec(function(err, task){
      response.status(200).json(task);
    });
});

mongoose.connect('mongodb://localhost/test');
http.listen('1234');

```

task.js

```

var mongoose = require('mongoose');

var taskSchema = mongoose.Schema({
  status: {
    type: String
  }
});

mongoose.model('Task', taskSchema);

module.exports = mongoose.model('Task');

```

processor.js

```

process.on('message', function(msg){
  init = function(){
    processData(msg.message);
  }.bind(this)();

```

```

function processData(message){
  //send status update to the main app
  process.send({ status: 'We have started processing your data.' });

  //long calculations ..
  setTimeout(function(){
    process.send({ status: 'Done!' });

    //notify node, that we are done with this task
    process.disconnect();
  }, 5000);
}
});

process.on('uncaughtException',function(err){
  console.log("Error happened: " + err.message + "\n" + err.stack + ".\n");
  console.log("Gracefully finish the routine.");
});

```

index.html

```

<!DOCTYPE html>
<html>
  <head>
    <script src="./js/jquery-1.12.0.min.js"></script>
    <script src="./js/main.js"></script>
  </head>
  <body>
    <p>Example of processing long-running node requests.</p>
    <button id="go" type="button">Run</button>

    <br />

    <p>Log:</p>
    <textarea id="log" rows="20" cols="50"></textarea>
  </body>
</html>

```

main.js

```

$(document).on('ready', function(){

  $('#go').on('click', function(e){
    //clear log
    $('#log').val('');

    $.post("/long-running-request", {some_params: 'params' })
      .done(function(task){
        $('#log').val( $('#log').val() + '\n' + task.status);

        //function for tracking the status of the task
        function updateStatus(){
          $.post("/is-ready", {id: task._id })
            .done(function(response){
              $('#log').val( $('#log').val() + '\n' + response.status);

              if(response.status != 'Done!'){
                checkTaskTimeout = setTimeout(updateStatus, 500);
              }
            });
        }
      });
  });
});

```

```
        }
      });
    }

    //start checking the task
    var checkTaskTimeout = setTimeout(updateStatus, 100);
  });
});
});
```

package.json

```
{
  "name": "nodeProcessor",
  "dependencies": {
    "body-parser": "^1.15.2",
    "express": "^4.14.0",
    "html": "0.0.10",
    "mongoose": "^4.5.5"
  }
}
```

◦ ◦

<https://riptutorial.com/zh-TW/node-js/topic/6325/>

79:

WebPush.js SoneSignalWeb /。

Javascript。 Notification API Chrome Safari Firefox IE 9+。

Socket.io Express。

/	
node.js /	Node.js
Socket.io	Socket.IO。 。
Push.js	
OneSignal	
	Firebase Google。

Examples

[Push.js](#)。

```
$ npm install push.js --save
```

[CDN](#)

```
<script src="./push.min.js"></script> <!-- CDN link -->
```

。

```
Push.create('Hello World!')
```

[Socket.io](#) 。

```
var app = require('express')();
var server = require('http').Server(app);
var io = require('socket.io')(server);

server.listen(80);

app.get('/', function (req, res) {
  res.sendfile(__dirname + '/index.html');
});

io.on('connection', function (socket) {
```



```
socket.emit('pushNotification', { success: true, msg: 'hello' });
});
```

- Socket.io [CDN](#) *index.html*

```
<script src="../../socket.io.js"></script> <!-- CDN link -->
<script>
  var socket = io.connect('http://localhost');
  socket.on('pushNotification', function (data) {
    console.log(data);
    Push.create("Hello world!", {
      body: data.msg, //this should print "hello"
      icon: '/icon.png',
      timeout: 4000,
      onClick: function () {
        window.focus();
        this.close();
      }
    });
  });
</script>
```

Android [Firebase](#) Push.js

Apple [OneSignal](#)

<https://riptutorial.com/zh-TW/node-js/topic/10892/>

80: HTML

- `response.sendFile(fileName, options, function(err){`

Examples

HTML

Express `index.html / /page1page1.html` ◦

```
project root
|   server.js
|___views
|   index.html
|   page1.html
```

server.js

```
var express = require('express');
var path = require('path');
var app = express();

// deliver index.html if no file is requested
app.get("/", function (request, response) {
  response.sendFile(path.join(__dirname, 'views/index.html'));
});

// deliver page1.html if page1 is requested
app.get('/page1', function(request, response) {
  response.sendFile(path.join(__dirname, 'views', 'page1.html', function(error) {
    if (error) {
      // do something in case of error
      console.log(err);
      response.end(JSON.stringify({error:"page not found"}));
    }
  }));
});

app.listen(8080);
```

`sendFile()` ◦ [HTML Pug](#) [Mustache](#) [EJS](#) ◦

[HTML https://riptutorial.com/zh-TW/node-js/topic/6538/html](https://riptutorial.com/zh-TW/node-js/topic/6538/html)

81: MongoDB Mongoose

Examples

```
mongodb mongod --dbpath data/
```

package.json

```
"dependencies": {  
  "mongoose": "^4.5.5",  
}
```

server.js ECMA 6

```
import mongoose from 'mongoose';  
  
mongoose.connect('mongodb://localhost:27017/stackoverflow-example');  
const db = mongoose.connection;  
db.on('error', console.error.bind(console, 'DB connection error!'));
```

server.js ECMA 5.1

```
var mongoose = require('mongoose');  
  
mongoose.connect('mongodb://localhost:27017/stackoverflow-example');  
var db = mongoose.connection;  
db.on('error', console.error.bind(console, 'DB connection error!'));
```

app / models / user.js ECMA 6

```
import mongoose from 'mongoose';  
  
const userSchema = new mongoose.Schema({  
  name: String,  
  password: String  
});  
  
const User = mongoose.model('User', userSchema);  
  
export default User;
```

app / model / user.js ECMA 5.1

```
var mongoose = require('mongoose');  
  
var userSchema = new mongoose.Schema({  
  name: String,  
  password: String  
});  
  
var User = mongoose.model('User', userSchema);
```

```
module.exports = User
```

ECMA 6

```
const user = new User({
  name: 'Stack',
  password: 'Overflow',
});

user.save((err) => {
  if (err) throw err;

  console.log('User saved!');
});
```

ECMA5.1

```
var user = new User({
  name: 'Stack',
  password: 'Overflow',
});

user.save(function (err) {
  if (err) throw err;

  console.log('User saved!');
});
```

ECMA6

```
User.findOne({
  name: 'stack'
}, (err, user) => {
  if (err) throw err;

  if (!user) {
    console.log('No user was found');
  } else {
    console.log('User was found');
  }
});
```

ECMA5.1

```
User.findOne({
  name: 'stack'
}, function (err, user) {
  if (err) throw err;

  if (!user) {
    console.log('No user was found');
  } else {
    console.log('User was found');
  }
});
```

MongoDB Mongoose <https://riptutorial.com/zh-TW/node-js/topic/6411/-mongodbmongoose->

82: I / O.

Node.js | `fs.readFile` `fs.readFileSync` | Node.js

◦ ◦

◦

Examples

writeFilewriteFileSync

```
var fs = require('fs');

// Save the string "Hello world!" in a file called "hello.txt" in
// the directory "/tmp" using the default encoding (utf8).
// This operation will be completed in background and the callback
// will be called when it is either done or failed.
fs.writeFile('/tmp/hello.txt', 'Hello world!', function(err) {
  // If an error occurred, show it and return
  if(err) return console.error(err);
  // Successfully wrote to the file!
});

// Save binary data to a file called "binary.txt" in the current
// directory. Again, the operation will be completed in background.
var buffer = new Buffer([ 0x48, 0x65, 0x6c, 0x6c, 0x66 ]);
fs.writeFile('binary.txt', buffer, function(err) {
  // If an error occurred, show it and return
  if(err) return console.error(err);
  // Successfully wrote binary contents to the file!
});
```

`fs.writeFileSync` `fs.writeFile` | [node.js](#)

[node.js](#) ◦

```
// Write a string to another file and set the file mode to 0755
try {
  fs.writeFileSync('sync.txt', 'anni', { mode: 0o755 });
} catch(err) {
  // An error occurred
  console.error(err);
}
```

filesystem

```
const fs = require('fs');
```

`/tmp/hello.txt` ◦

```

fs.readFile('/tmp/hello.txt', { encoding: 'utf8' }, (err, content) => {
  // If an error occurred, output it and return
  if(err) return console.error(err);

  // No error occurred, content is a string
  console.log(content);
});

```

binary.txt ◦ 'encoding' - Node.js

```

fs.readFile('binary', (err, binaryContent) => {
  // If an error occurred, output it and return
  if(err) return console.error(err);

  // No error occurred, content is a Buffer, output it in
  // hexadecimal representation.
  console.log(content.toString('hex'));
});

```

◦ __dirname__filename

```

fs.readFile(path.resolve(__dirname, 'someFile'), (err, binaryContent) => {
  //Rest of Function
}

```

readdirreaddirSync

```

const fs = require('fs');

// Read the contents of the directory /usr/local/bin asynchronously.
// The callback will be invoked once the operation has either completed
// or failed.
fs.readdir('/usr/local/bin', (err, files) => {
  // On error, show it and return
  if(err) return console.error(err);

  // files is an array containing the names of all entries
  // in the directory, excluding '.' (the directory itself)
  // and '..' (the parent directory).

  // Display directory entries
  console.log(files.join(' '));
});

```

readdirSync ◦ IO ◦

```

let files;

try {
  files = fs.readdirSync('/var/tmp');
} catch(err) {
  // An error occurred
  console.error(err);
}

```

```

const fs = require('fs');

// Iterate through all items obtained via
// 'yield' statements
// A callback is passed to the generator function because it is required by
// the 'readdir' method
function run(gen) {
  var iter = gen((err, data) => {
    if (err) { iter.throw(err); }

    return iter.next(data);
  });

  iter.next();
}

const dirPath = '/usr/local/bin';

// Execute the generator function
run(function* (resume) {
  // Emit the list of files in the directory from the generator
  var contents = yield fs.readdir(dirPath, resume);
  console.log(contents);
});

```

```
const fs = require('fs');
```

`fs.readFileSync` `fs.readFile` `node.js`

`encodingBuffer`

```

// Read a string from another file synchronously
let content;
try {
  content = fs.readFileSync('sync.txt', { encoding: 'utf8' });
} catch(err) {
  // An error occurred
  console.error(err);
}

```

unlink

```

var fs = require('fs');

fs.unlink('/path/to/file.txt', function(err) {
  if (err) throw err;

  console.log('file deleted');
});

```

*

```
var fs = require('fs');
```



```
fs.unlinkSync('/path/to/file.txt');
console.log('file deleted');
```

*。

fs.readFile() **Stream** ◦ ◦

```
const fs = require('fs');

// Store file data chunks in this array
let chunks = [];
// We can use this variable to store the final data
let fileBuffer;

// Read file into stream.Readable
let fileStream = fs.createReadStream('text.txt');

// An error occurred with the stream
fileStream.once('error', (err) => {
  // Be sure to handle this properly!
  console.error(err);
});

// File is done being read
fileStream.once('end', () => {
  // create the final data Buffer from data chunks;
  fileBuffer = Buffer.concat(chunks);

  // Of course, you can do anything else you need to here, like emit an event!
});

// Data is flushed from fileStream in chunks,
// this callback will be executed for each chunk
fileStream.on('data', (chunk) => {
  chunks.push(chunk); // push data chunk to array

  // We can perform actions on the partial data we have so far!
});
```

fs.access() ◦ fs.access ◦

fs.constants

- fs.constants.F_OK - //
- fs.constants.R_OK -
- fs.constants.W_OK -
- fs.constants.X_OK - **Windows** fs.constants.F_OK

```
var fs = require('fs');
var path = '/path/to/check';

// checks execute permission
fs.access(path, fs.constants.X_OK, (err) => {
  if (err) {
    console.log("%s doesn't exist", path);
  }
});
```

```

    } else {
        console.log('can execute %s', path);
    }
});
// Check if we have read/write permissions
// When specifying multiple permission modes
// each mode is separated by a pipe : `|`
fs.access(path, fs.constants.R_OK | fs.constants.W_OK, (err) => {
    if (err) {
        console.log("%s doesn't exist", path);
    } else {
        console.log('can read/write %s', path);
    }
});

```

fs.access **fs.accessSync** ◦ **fs.accessSync** **try / catch** ◦

```

// Check write permission
try {
    fs.accessSync(path, fs.constants.W_OK);
    console.log('can write %s', path);
}
catch (err) {
    console.log("%s doesn't exist", path);
}

```

Node

1. `fs.stat()`
- 2.

◦ `fs.mkdir()` `fs.mkdirSync()` `EEXIST` ◦ `EPERM` ◦

fs.mkdir()

```

var fs = require('fs');

function mkdir (dirPath, callback) {
    fs.mkdir(dirPath, (err) => {
        callback(err && err.code !== 'EEXIST' ? err : null);
    });
}

mkdir('./existingDir', (err) => {

    if (err)
        return console.error(err.code);

    // Do something with `./existingDir` here

});

```

fs.mkdirSync()

```

function mkdirSync (dirPath) {

```

```

try {
  fs.mkdirSync(dirPath);
} catch(e) {
  if ( e.code !== 'EEXIST' ) throw e;
}
}

mkdirSync('./existing-dir');
// Do something with `./existing-dir` now

```

```

var fs = require('fs');

fs.stat('path/to/file', function(err) {
  if (!err) {
    console.log('file or directory exists');
  }
  else if (err.code === 'ENOENT') {
    console.log('file or directory does not exist');
  }
});

```

try/catch◦

```

var fs = require('fs');

try {
  fs.statSync('path/to/file');
  console.log('file or directory exists');
}
catch (err) {
  if (err.code === 'ENOENT') {
    console.log('file or directory does not exist');
  }
}

```

createReadStream()createWriteStream()◦

```

//Require the file System module
var fs = require('fs');

/*
  Create readable stream to file in current directory (__dirname) named 'node.txt'
  Use utf8 encoding
  Read the data in 16-kilobyte chunks
*/
var readable = fs.createReadStream(__dirname + '/node.txt', { encoding: 'utf8', highWaterMark:
16 * 1024 });

// create writable stream
var writable = fs.createWriteStream(__dirname + '/nodeCopy.txt');

// Write each chunk of data to the writable stream
readable.on('data', function(chunk) {
  writable.write(chunk);
});

```

stream.pipe()

```
// require the file system module
var fs = require('fs');

/*
  Create readable stream to file in current directory named 'node.txt'
  Use utf8 encoding
  Read the data in 16-kilobyte chunks
*/
var readable = fs.createReadStream(__dirname + '/node.txt', { encoding: 'utf8', highWaterMark:
16 * 1024 });

// create writable stream
var writable = fs.createWriteStream(__dirname + '/nodePipe.txt');

// use pipe to copy readable to writable
readable.pipe(writable);
```

◦ emailindex.txtname index.txt **RegExp** replace(/email/gim, 'name')

```
var fs = require('fs');

fs.readFile('index.txt', 'utf-8', function(err, data) {
  if (err) throw err;

  var newValue = data.replace(/email/gim, 'name');

  fs.writeFile('index.txt', newValue, 'utf-8', function(err, data) {
    if (err) throw err;
    console.log('Done!');
  })
})
```

app.js

```
const readline = require('readline');
const fs = require('fs');

var file = 'path.to.file';
var linesCount = 0;
var rl = readline.createInterface({
  input: fs.createReadStream(file),
  output: process.stdout,
  terminal: false
});
rl.on('line', function (line) {
  linesCount++; // on each linebreak, add +1 to 'linesCount'
});
rl.on('close', function () {
  console.log(linesCount); // print the result when the 'close' event is called
});
```

app.js

```
const readline = require('readline');
const fs = require('fs');

var file = 'path.to.file';
var rl = readline.createInterface({
  input: fs.createReadStream(file),
  output: process.stdout,
  terminal: false
});

rl.on('line', function (line) {
  console.log(line) // print the content of the line on each linebreak
});
```

I / O. <https://riptutorial.com/zh-TW/node-js/topic/489/i---o->

83:

Examples

Nunjucks

- [jinja2Twigphp](#)◦

- <http://mozilla.github.io/nunjucks/>

- `npm i nunjucks`

Express◦

app.js

```
var express = require ('express');
var nunjucks = require('nunjucks');

var app = express();
app.use(express.static('/public'));

// Apply nunjucks and add custom filter and function (for example).
var env = nunjucks.configure(['views/'], { // set folders with templates
  autoescape: true,
  express: app
});
env.addFilter('myFilter', function(obj, arg1, arg2) {
  console.log('myFilter', obj, arg1, arg2);
  // Do smth with obj
  return obj;
});
env.addGlobal('myFunc', function(obj, arg1) {
  console.log('myFunc', obj, arg1);
  // Do smth with obj
  return obj;
});

app.get('/', function(req, res){
  res.render('index.html', {title: 'Main page'});
});

app.get('/foo', function(req, res){
  res.locals.smthVar = 'This is Sparta!';
  res.render('foo.html', {title: 'Foo page'});
});

app.listen(3000, function() {
  console.log('Example app listening on port 3000...');
});
```

/views/index.html

```
<html>
```

```
<head>
  <title>Nunjucks example</title>
</head>
<body>
{% block content %}
  {{title}}
{% endblock %}
</body>
</html>
```

/views/foo.html

```
{% extends "index.html" %}

{# This is comment #}
{% block content %}
  <h1>{{title}}</h1>
  {# apply custom function and next build-in and custom filters #}
  {{ myFunc(smithVar) | lower | myFilter(5, 'abc') }}
{% endblock %}
```

<https://riptutorial.com/zh-TW/node-js/topic/5885/>

84: Node.js。

Examples

PM2。

ecosystem.json

```
{
  "name": "app-name",
  "script": "server",
  "exec_mode": "cluster",
  "instances": 0,
  "wait_ready": true
  "listen_timeout": 10000,
  "kill_timeout": 5000,
}
```

```
wait_ready
```

listenprocess.send'ready';

```
listen_timeout
```

。

```
kill_timeout
```

SIGKLL。

server.js

```
const http = require('http');
const express = require('express');

const app = express();
const server = http.Server(app);
const port = 80;

server.listen(port, function() {
  process.send('ready');
});

process.on('SIGINT', function() {
  server.close(function() {
    process.exit(0);
  });
});
```

DB ///。 PM2。 wait_ready: true 。 PM2。 process.send('ready');。

PM2/。

SIGINT° 1.6sSIGKILL° SIGINT°

Node.js° <https://riptutorial.com/zh-TW/node-js/topic/9752/node-js->

85: - REST

- LoopbackREST

Examples

Web

```
/{
  "name": {
    "type": "string"
    "debug": true
    "useQueryString": true
    "limit": 10000
    "type": {
      "application": "application / json"
      "content-type": "application / json"
    }
  }
}
{
  "name": {
    {
      "type": {
        "method": "GET"
        "url": "https://itunes.apple.com/search"
        "type": {
          "term": "{keyword}"
          "country": "{country = IN}"
          "media": "{itemType = music}"
          "limit": "{limit = 10}"
        }
      }
    }
  }
}
{
  "name": {
    {
      "type": {
        "method": "GET"
        "url": "https://itunes.apple.com/lookup"
        "type": {
          "id": "{id}"
        }
      }
    }
  }
}
{
  "name": {
    "findById": [
      "ID"
    ]
  }
}
```

```
    }  
  }  
]  
}  
}
```

- REST <https://riptutorial.com/zh-TW/node-js/topic/9234/---rest>

86:

Examples

process.env

```
{
  TERM: 'xterm-256color',
  SHELL: '/usr/local/bin/bash',
  USER: 'maciej',
  PATH: '~/.bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin',
  PWD: '/Users/maciej',
  EDITOR: 'vim',
  SHLVL: '1',
  HOME: '/Users/maciej',
  LOGNAME: 'maciej',
  _: '/usr/local/bin/node'
}
```

```
process.env.HOME // '/Users/maciej'
```

FOOfoobar

```
process.env.FOO // 'foobar'
```

process.argv

[process.argv](#) ◦ [node JavaScript](#) ◦ ◦

index.js

```
var sum = 0;
for (i = 2; i < process.argv.length; i++) {
  sum += Number(process.argv[i]);
}

console.log(sum);
```

```
node index.js 2 5 6 7
```

20

for for (i = 2; i < process.argv.length; i++) **2**process.argv ['path/to/node.exe',
'path/to/js/file', ...]

Number(process.argv[i]) **process.argv**

/devqastaging ◦

◦ NodeJs◦

◦

- dev.json

```
{
  PORT : 3000,
  DB : {
    host : "localhost",
    user : "bob",
    password : "12345"
  }
}
```

- qa.json

```
{
  PORT : 3001,
  DB : {
    host : "where_db_is_hosted",
    user : "bob",
    password : "54321"
  }
}
```

◦

environment.js

```
process.argv.forEach(function (val, index, array) {
  var arg = val.split("=");
  if (arg.length > 0) {
    if (arg[0] === 'env') {
      var env = require('./' + arg[1] + '.json');
      module.exports = env;
    }
  }
});
```

```
node app.js env=dev
```

```
forever start app.js env=dev
```

```
var env= require("environment.js");
```

“”

•

```
npm install properties-reader --save
```

- **env**

```
mkdir env
```

- **environments.js**

```
process.argv.forEach(function (val, index, array) {  
  var arg = val.split("=");  
  if (arg.length > 0) {  
    if (arg[0] === 'env') {  
      var env = require('./env/' + arg[1] + '.properties');  
      module.exports = env;  
    }  
  }  
});
```

- **development.properties**

```
# Dev properties  
[main]  
# Application port to run the node server  
app.port=8080  
  
[database]  
# Database connection to mysql  
mysql.host=localhost  
mysql.port=2500  
...
```

-

```
var environment = require('./environments');  
var PropertiesReader = require('properties-reader');  
var properties = new PropertiesReader(environment);  
  
var someVal = properties.get('main.app.port');
```

-

```
npm start env=development
```

```
npm start env=production
```

<https://riptutorial.com/zh-TW/node-js/topic/2340/>

87:

Examples

Node.Js

Node.js3/

1. -
2. `callback`
3. `eventEmitter` `emit`

try-catch ◦ `...try / catch` ◦

try-catch ◦ `promises` ◦

```
// ** Example - 1 **
function doSomeSynchronousOperation(req, res) {
  if(req.body.username === ''){
    throw new Error('User Name cannot be empty');
  }
  return true;
}

// calling the method above
try {
  // synchronous code
  doSomeSynchronousOperation(req, res)
} catch(e) {
  //exception handled here
  console.log(e.message);
}

// ** Example - 2 **
function doSomeAsynchronousOperation(req, res, cb) {
  // imitating async operation
  return setTimeout(function(){
    cb(null, []);
  },1000);
}

try {
  // asynchronous code
  doSomeAsynchronousOperation(req, res, function(err, rs){
    throw new Error("async operation exception");
  })
} catch(e) {
  // Exception will not get handled here
  console.log(e.message);
}
// The exception is unhandled and hence will cause application to break
```

Node.js ◦

◦

```

function doSomeAsynchronousOperation(req, res, callback) {
  setTimeout(function(){
    return callback(new Error('User Name cannot be empty'));
  }, 1000);
  return true;
}

doSomeAsynchronousOperation(req, res, function(err, result) {
  if (err) {
    //exception handled here
    console.log(err.message);
  }

  //do some stuff with valid data
});

```

EventEmitter

```

const EventEmitter = require('events');

function doSomeAsynchronousOperation(req, res) {
  let myEvent = new EventEmitter();

  // runs asynchronously
  setTimeout(function(){
    myEvent.emit('error', new Error('User Name cannot be empty'));
  }, 1000);

  return myEvent;
}

// Invoke the function
let event = doSomeAsynchronousOperation(req, res);

event.on('error', function(err) {
  console.log(err);
});

event.on('done', function(result) {
  console.log(result); // true
});

```

Node.js

node.js

-

```

process.on('uncaughtException', function (err) {
  console.log(err);
});

```

- ◦
- db

“uncaughtException”。

- CLI

```
npm install forever -g
```

-

```
forever start app.js
```

。

-

```
process.on('uncaughtException', function (err) {  
  console.log(err);  
  
  // some logging mechanism  
  // ....  
  
  process.exit(1); // terminates process  
});
```

。

。

Promise。

```
const p = new Promise(function (resolve, reject) {  
  reject(new Error('Oops'));  
});  
  
// anything that is `reject`ed inside a promise will be available through catch  
// while a promise is rejected, `.then` will not be called  
p  
  .then(() => {  
    console.log("won't be called");  
  })  
  .catch(e => {  
    console.log(e.message); // output: Oops  
  })  
  // once the error is caught, execution flow resumes  
  .then(() => {  
    console.log('hello!'); // output: hello!  
  });
```

- [eslintlinting](#) catch。

8。

<https://riptutorial.com/zh-TW/node-js/topic/2819/>

88: /

Async / await promise-chaining .then().then().then() ◦

await promise async promise async ◦

node.js 8 Async / await --harmony-async-await 7 ◦

Examples

Try-Catch

async / await try-catch ◦

```
const myFunc = async (req, res) => {
  try {
    const result = await somePromise();
  } catch (err) {
    // handle errors here
  }
};
```

Express promise-mysql

```
router.get('/flags/:id', async (req, res) => {

  try {

    const connection = await pool.createConnection();

    try {
      const sql = `SELECT f.id, f.width, f.height, f.code, f.filename
                    FROM flags f
                    WHERE f.id = ?
                    LIMIT 1`;
      const flags = await connection.query(sql, req.params.id);
      if (flags.length === 0)
        return res.status(404).send({ message: 'flag not found' });

      return res.send({ flags[0] });

    } finally {
      pool.releaseConnection(connection);
    }

  } catch (err) {
    // handle errors here
  }
};
```

Promises Async / Await

```
function myAsyncFunction() {
  return aFunctionThatReturnsAPromise()
    // doSomething is a sync function
    .then(result => doSomething(result))
    .catch(handleError);
}
```

Async / Await

```
async function myAsyncFunction() {
  let result;

  try {
    result = await aFunctionThatReturnsAPromise();
  } catch (error) {
    handleError(error);
  }

  // doSomething is a sync function
  return doSomething(result);
}
```

async **write** return new Promise((resolve, reject) => {...}) ◦

await **then** ◦

GIF

GIF

```
const getTemperature = (callback) => {
  http.get('www.temperature.com/current', (res) => {
    callback(res.data.temperature)
  })
}

const getAirPollution = (callback) => {
  http.get('www.pollution.com/current', (res) => {
    callback(res.data.pollution)
  });
}

getTemperature(function(temp) {
  getAirPollution(function(pollution) {
    console.log(`the temp is ${temp} and the pollution is ${pollution}.`)
    // The temp is 27 and the pollution is 0.5.
  })
})
```

promises ◦

```
const getTemperature = () => {
  return new Promise((resolve, reject) => {
    http.get('www.temperature.com/current', (res) => {
      resolve(res.data.temperature)
    })
  })
}
```

```

    })
  }

  const getAirPollution = () => {
    return new Promise((resolve, reject) => {
      http.get('www.pollution.com/current', (res) => {
        resolve(res.data.pollution)
      })
    })
  }

  getTemperature()
  .then(temp => console.log(`the temp is ${temp}`))
  .then(() => getAirPollution())
  .then(pollution => console.log(`and the pollution is ${pollution}`))
  // the temp is 32
  // and the pollution is 0.5

```

◦ **async / await** ◦

```

const temp = await getTemperature()
const pollution = await getAirPollution()

```

promise_{await}◦

```

try{
  await User.findByIdAndUpdate(user._id, {
    $push: {
      tokens: token
    }
  }).exec()
}catch(e){
  handleError(e)
}

```

[https://riptutorial.com/zh-TW/node-js/topic/6729/-](https://riptutorial.com/zh-TW/node-js/topic/6729/)

89:

Node ◦ ◦

- `doSomething[args]function[argsCB]{/** /};`
- `doSomething[args][argsCB]=> {/** /};`

Examples

JavaScript

JavaScript ◦ JavaScript ◦

◦

◦ ◦

```
// a function that uses a callback named `cb` as a parameter
function getSyncMessage(cb) {
  cb("Hello World!");
}

console.log("Before getSyncMessage call");
// calling a function and sending in a callback function as an argument.
getSyncMessage(function(message) {
  console.log(message);
});
console.log("After getSyncMessage call");
```

```
> Before getSyncMessage call
> Hello World!
> After getSyncMessage call
```

◦ ◦ **6**Before getSyncMessage call◦ **8**getSyncMessagegetSyncMessagecb◦ **3**getSyncMessagecb message
“Hello World”◦ **9**Hello World!◦ [callstack](#) 10411◦

- ◦ ◦
- ◦
- ◦ **8**messagestatement msg jellybean◦ ◦

◦

JavaScriptNode.jsAPI ◦

JavaScript

-
- **setTimeout**
- **setInterval**
- **fetch API**
-

promises.

◦

```
// a function that uses a callback named `cb` as a parameter
function getAsyncMessage(cb) {
  setTimeout(function () { cb("Hello World!") }, 1000);
}

console.log("Before getSyncMessage call");
// calling a function and sending in a callback function as an argument.
getAsyncMessage(function(message) {
  console.log(message);
});
console.log("After getSyncMessage call");
```

```
> Before getSyncMessage call
> After getSyncMessage call
// pauses for 1000 ms with no output
> Hello World!
```

6“getSyncMessage” 8param cbgetAsyncMessage 3setTimeout300 setTimeout 100010004111
setTimeout3getAsyncMessagesmessage“Hello World”9

Node.js

NodeJSerrdata ◦ ◦

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

fs.readFile("./test.txt", "utf8", function(err, data) {
  if(err) {
    // handle the error
  } else {
    // process the file text given with data
  }
});
```

◦

◦ **elseifelse**.

errdata

4.x

```
// this code snippet was on http://expressjs.com/en/4x/api.html
const express = require('express');
const app = express();

// this app.get method takes a url route to watch for and a callback
// to call whenever that route is requested by a user.
app.get('/', function(req, res){
  res.send('hello world');
});

app.listen(3000);
```

◦ **params** reqres◦

JavaScriptJavaScript◦

```
setTimeout(function() {
  console.log("A");
}, 1000);

setTimeout(function() {
  console.log("B");
}, 0);

getDataFromDatabase(function(err, data) {
  console.log("C");
  setTimeout(function() {
    console.log("D");
  }, 1000);
});

console.log("E");
```

EBAD ◦ C◦

setTimeoutgetDataFromDatabase◦ E setTimeout

1. EsetTimeout
2. B0
3. A1000
4. DD1000A ◦
5. C◦ A◦

◦ try catch ◦

```
try {
  setTimeout(function() {
    throw new Error("I'm an uncaught error and will stop the server!");
  }, 100);
}
catch (ex) {
  console.error("This error will not be work in an asynchronous situation: " + ex);
}
```

V0.8

Node.JS。

```
process.on("UncaughtException", function(err, data) {
  if (err) {
    // error handling
  }
});
```

V0.8

。 。 。

```
var domain = require("domain");
var d1 = domain.create();
var d2 = domain.create();

d1.run(function() {
  d2.add(setTimeout(function() {
    throw new Error("error on the timer of domain 2");
  }, 0));
});

d1.on("error", function(err) {
  console.log("error at domain 1: " + err);
});

d2.on("error", function(err) {
  console.log("error at domain 2: " + err);
});
```

。 ES6。

```
const fs = require('fs');
let filename = `${__dirname}/myfile.txt`;

fs.exists(filename, exists => {
  if (exists) {
    fs.stat(filename, (err, stats) => {
      if (err) {
        throw err;
      }
      if (stats.isFile()) {
        fs.readFile(filename, null, (err, data) => {
          if (err) {
            throw err;
          }
          console.log(data);
        });
      }
    });
  }
  else {
    throw new Error("This location contains not a file");
  }
});
```



```

    }
    else {
        throw new Error("404: file not found");
    }
});

```

“”

2。 。 2。 。

asyncnpm。 。

V6.0.0

Promise。 JavaScriptpromises。 then。 ""。 ""。 。 """"。 newpromise。 new Promise(function (resolve, reject) {})。 。

Promise。 resolve。 reject。 """"。 """"。 reject。 resolve。 。

timeoutPromise。 。

```

function timeout (ms) {
    return new Promise(function (resolve, reject) {
        setTimeout(function () {
            resolve("It was resolved!");
        }, ms)
    });
}

timeout(1000).then(function (dataFromPromise) {
    // logs "It was resolved!"
    console.log(dataFromPromise);
})

console.log("waiting...");

```

```

waiting...
// << pauses for one second>>
It was resolved!

```

Promise。 setTimeout。 ms=1000。 setTimeout。 then/Promise。 catch/promise"。 。

'.....'。 setTimeout。 "It was resolved"。 resolve。 then。 。

setTimeout。 promise。 。

JavaScriptPromisespromises。 。

<https://riptutorial.com/zh-TW/node-js/topic/8813/>

90: Web

Examples

GCMWebGoogle Cloud Messaging System

PWA WebNodeJSES6

1. Node-GCM `npm install node-gcm`
2. Socket.io `npm install socket.io`
3. [GoogleGCM](#).
4. GrabeGCMID
5. GrabeGCM.

```
6. 'use strict';

const express = require('express');
const app = express();
const gcm = require('node-gcm');
app.io = require('socket.io')();

// [*] Configuring our GCM Channel.
const sender = new gcm.Sender('Project Secret');
const regTokens = [];
let message = new gcm.Message({
  data: {
    key1: 'msg1'
  }
});

// [*] Configuring our static files.
app.use(express.static('public/'));

// [*] Configuring Routes.
app.get('/', (req, res) => {
  res.sendFile(__dirname + '/public/index.html');
});

// [*] Configuring our Socket Connection.
app.io.on('connection', socket => {
  console.log('we have a new connection ...');
  socket.on('new_user', (reg_id) => {
    // [*] Adding our user notification registration token to our list typically
    // hidden in a secret place.
    if (regTokens.indexOf(reg_id) === -1) {
      regTokens.push(reg_id);

      // [*] Sending our push messages
      sender.send(message, {
        registrationTokens: regTokens
      });
    }
  });
});
```

```
    }, (err, response) => {
      if (err) console.error('err', err);
      else console.log(response);
    });
  }
})
});

module.exports = app
```

PShackSocket.ioExpress。

.json Manifest.json

```
{
  "name": "Application Name",
  "gcm_sender_id": "GCM Project ID"
}
```

ROOT。

PSManifest.json。

1. index.htmlsocket.io。
2. **index.html**
3. socket.io **new_user**index.html **WebAPI**。
4. **node-gcmService Workers`**。

NodeJS。 ..etc。

PS。

Web <https://riptutorial.com/zh-TW/node-js/topic/6333/web>

91: package.json

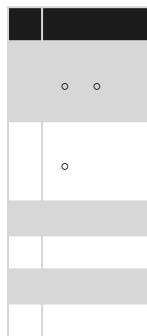
package.json

```
npm init
```

◦

Examples

```
{
  "name": "my-project",
  "version": "0.0.1",
  "description": "This is a project.",
  "author": "Someone <someone@example.com>",
  "contributors": [{
    "name": "Someone Else",
    "email": "else@example.com"
  }],
  "keywords": ["improves", "searching"]
}
```



“dependencies”{“module-name”“0.1.0”}

- 0.1.0◦
- ^0.1.00.2.0 1.0.0
- 0.1.x~0.1.00.1.4 0.2.01.0.0 ◦
- *◦
- **git repository** git repomastertarball◦ #sha #tag#branch
 - **GitHub** user/projectuser/project#v1.0.0
 - **url** git://gitlab.com/user/project.gitgit://gitlab.com/user/project.git#develop
- file:../lib/project

package.jsonnpm install ◦

devDependencies

```
"devDependencies": {
```

```
"module-name": "0.1.0"
}
```

ext. “npm install”dev-dependencies.

o

```
{
  "scripts": {
    "pretest": "scripts/pretest.js",
    "test": "scripts/test.js",
    "posttest": "scripts/posttest.js"
  }
}
```

```
$ npm run-script test
$ npm run test
$ npm test
$ npm t
```

prepublish	o
	o
	o
postinstall	o
	o
postuninstall	o
preversion	o
postversion	o
	npm test
prestopstoppoststop	npm stop
	npm start
prerestartrestartpostrestart	npm restart

```
{
  "scripts": {
```

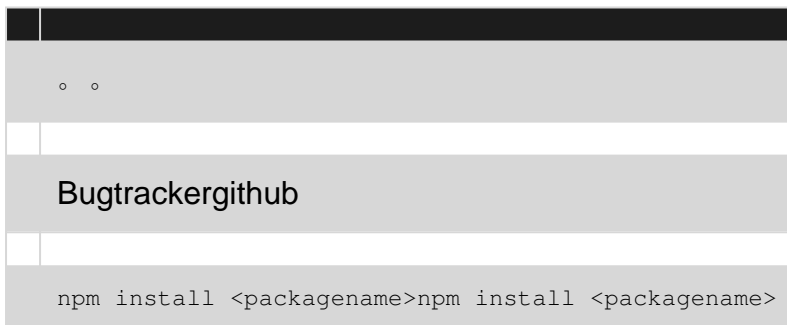
```
"preci": "scripts/preci.js",
"ci": "scripts/ci.js",
"postci": "scripts/postci.js"
}
}
```

```
$ npm run-script ci
$ npm run ci
```

◦

npmrepository bugshomepage

```
{
  "main": "server.js",
  "repository" : {
    "type": "git",
    "url": "git+https://github.com/<accountname>/<repositoryname>.git"
  },
  "bugs": {
    "url": "https://github.com/<accountname>/<repositoryname>/issues"
  },
  "homepage": "https://github.com/<accountname>/<repositoryname>#readme",
  "files": [
    "server.js", // source files
    "README.md", // additional files
    "lib" // folder with all included files
  ]
}
```



package.json

package.jsonnpm installnpm◦

package.jsonnpm init◦

package.json

```
npm init --yes
# or
npm init -y
```

package.json

```
npm install {package name} --save
```

```
npm i -S {package name}
```

NPM-S--save-D--save-dev

;--save-dev--save **devDependencies**.

package.json

```
{
  "name": "module-name",
  "version": "10.3.1",
  "description": "An example module to illustrate the usage of a package.json",
  "author": "Your Name <your.name@example.org>",
  "contributors": [{
    "name": "Foo Bar",
    "email": "foo.bar@example.com"
  }],
  "bin": {
    "module-name": "./bin/module-name"
  },
  "scripts": {
    "test": "vows --spec --isolate",
    "start": "node index.js",
    "predeploy": "echo About to deploy",
    "postdeploy": "echo Deployed",
    "prepublish": "coffee --bare --compile --output lib/foo src/foo/*.coffee"
  },
  "main": "lib/foo.js",
  "repository": {
    "type": "git",
    "url": "https://github.com/username/repo"
  },
  "bugs": {
    "url": "https://github.com/username/issues"
  },
  "keywords": [
    "example"
  ],
  "dependencies": {
    "express": "4.2.x"
  },
  "devDependencies": {
    "assume": "<1.0.0 || >=2.3.1 <2.4.5 || >=2.5.2 <3.0.0"
  },
  "peerDependencies": {
    "moment": ">2.0.0"
  },
  "preferGlobal": true,
  "private": true,
  "publishConfig": {
    "registry": "https://your-private-hosted-npm.registry.domain.com"
  },
  "subdomain": "foobar",
  "analyze": true,
  "license": "MIT",
  "files": [
```

```
    "lib/foo.js"  
  ]  
}
```

name

◦ ◦

1. 214◦
2. ◦
3. ◦

version

Semantic Versioning semver◦ MAJOR.MINOR.PATCH

1. API MAJOR
2. MINOR
3. PATCH

description

◦ ◦

author

◦

bin

◦ ◦

CLI◦

script

npm◦ **npm**◦ npm run {command name}npm run-script {command name}◦

◦ mocha./node-modules/.bin/mocha ◦

main

- require('{module name}')◦
- **HTTP**◦ exports.init = function () {...}◦

keywords

◦ ◦

```
devDependencies
```

◦ `NODE_ENV=production` ◦ `npm install --dev`

```
peerDependencies
```

peerDependencies ◦ `moment-timezone moment require("moment")`

```
preferGlobal
```

`npm install -g {module-name}` ◦ `npm install -g {module-name}` ◦ **CLI** ◦

◦

```
publishConfig
```

publishConfig ◦ `npm` ◦

`publishConfig` ◦ `npm` ◦ `npmURL` ◦

```
files
```

◦ ◦ ◦ ◦ `.npmignore` ◦

package.json <https://riptutorial.com/zh-TW/node-js/topic/1515/package-json>

92: ReadLine

- `const readline = require('readline')`
- `readline.close`
- `readline.pause`
- `readline.prompt[preserveCursor]`
- `readline.question`
- `readline.resume`
- `readline.setPrompt`
- `readline.writedata [key]`
- `readline.clearLinestreamdir`
- `readline.clearScreenDown`
- `readline.createInterface`
- `readline.cursorTostreamxy`
- `readline.emitKeypressEventsstream [interface]`
- `readline.moveCursorstreamdxdy`

Examples

```
const fs = require('fs');
const readline = require('readline');

const rl = readline.createInterface({
  input: fs.createReadStream('text.txt')
});

// Each new line emits an event - every time the stream receives \r, \n, or \r\n
rl.on('line', (line) => {
  console.log(line);
});

rl.on('close', () => {
  console.log('Done reading file');
});
```

CLI

```
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

rl.question('What is your name?', (name) => {
  console.log(`Hello ${name}!`);

  rl.close();
});
```

ReadLine <https://riptutorial.com/zh-TW/node-js/topic/1431/readline>

93: jscsv

CSV ◦ csv ◦ ◦

Examples

FSCSV

fsAPI ◦ fsreadFile data.csv csv ◦

data.csv ◦

```
'use strict'

const fs = require('fs');

fs.readFile('data.csv', 'utf8', function (err, data) {
  var dataArray = data.split(/\r?\n/);
  console.log(dataArray);
});
```

◦

[jscsv](https://riptutorial.com/zh-TW/node-js/topic/9162/jscsv) <https://riptutorial.com/zh-TW/node-js/topic/9162/jscsv>

94: JS

nodejs

Examples

i18njs app

json。 vanilla node.js express restify res req app.use。 app templates __'...'。 web translate it json json。
。 。

+ i18n-node + cookieParser

```
// usual requirements
var express = require('express'),
    i18n = require('i18n'),
    app = module.exports = express();

i18n.configure({
  // setup some locales - other locales default to en silently
  locales: ['en', 'ru', 'de'],

  // sets a custom cookie name to parse locale settings from
  cookie: 'yourcookiename',

  // where to store json files - defaults to './locales'
  directory: __dirname + '/locales'
});

app.configure(function () {
  // you will need to use cookieParser to expose cookies to req.cookies
  app.use(express.cookieParser());

  // i18n init parses req for language headers, cookies, etc.
  app.use(i18n.init);
});

// serving homepage
app.get('/', function (req, res) {
  res.send(res.__('Hello World'));
});

// starting server
if (!module.parent) {
  app.listen(3000);
}
```

JS <https://riptutorial.com/zh-TW/node-js/topic/9594/js>

95:

Node

Express

ALL-IN-ONE UNITY。

/。

CORS。

Examples

```
var http = require('http');
var fs = require('fs');
var path = require('path');

http.createServer(function (request, response) {
  console.log('request ', request.url);

  var filePath = '.' + request.url;
  if (filePath == './')
    filePath = './index.html';

  var extname = String(path.extname(filePath)).toLowerCase();
  var contentType = 'text/html';
  var mimeTypes = {
    '.html': 'text/html',
    '.js': 'text/javascript',
    '.css': 'text/css',
    '.json': 'application/json',
    '.png': 'image/png',
    '.jpg': 'image/jpeg',
    '.gif': 'image/gif',
    '.wav': 'audio/wav',
    '.mp4': 'video/mp4',
    '.woff': 'application/font-woff',
    '.ttf': 'application/font-ttf',
    '.eot': 'application/vnd.ms-fontobject',
    '.otf': 'application/font-otf',
    '.svg': 'application/image/svg+xml'
  };

  contentType = mimeTypes[extname] || 'application/octet-stream';

  fs.readFile(filePath, function(error, content) {
    if (error) {
      if(error.code == 'ENOENT'){
        fs.readFile('./404.html', function(error, content) {
          response.writeHead(200, { 'Content-Type': contentType });
          response.end(content, 'utf-8');
        });
      }
    }
    else {
```

```
        response.writeHead(500);
        response.end('Sorry, check with the site admin for error: '+error.code+' ..\n');
        response.end();
    }
}
else {
    response.writeHead(200, { 'Content-Type': contentType });
    response.end(content, 'utf-8');
}
});

}).listen(8125);
console.log('Server running at http://127.0.0.1:8125/');
```

CORS

```
// Website you wish to allow to connect to
response.setHeader('Access-Control-Allow-Origin', '*');

// Request methods you wish to allow
response.setHeader('Access-Control-Allow-Methods', 'GET, POST, OPTIONS, PUT, PATCH, DELETE');

// Request headers you wish to allow
response.setHeader('Access-Control-Allow-Headers', 'X-Requested-With,content-type');

// Set to true if you need the website to include cookies in the requests sent
// to the API (e.g. in case you use sessions)
response.setHeader('Access-Control-Allow-Credentials', true);
```

<https://riptutorial.com/zh-TW/node-js/topic/5910/>

96:

[YarnNode.jsnpm](#) ◦ [Yarnnpm](#) ◦

Examples

Yarn ◦

```
brew update  
brew install yarn
```

MacPorts

```
sudo port install yarn
```

PATH

shell .profile .bashrc .zshrc

```
export PATH="$PATH:`yarn global bin`"
```

Node.js ◦

[YarnYarn.msi](#) ◦

```
choco install yarn
```

Linux

Debian / Ubuntu

Node.js

```
curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -  
sudo apt-get install -y nodejs
```

YarnPkg

```
curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -
```



```
echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee
/etc/apt/sources.list.d/yarn.list
```

```
sudo apt-get update && sudo apt-get install yarn
```

CentOS / Fedora / RHEL

Node.js.

```
curl --silent --location https://rpm.nodesource.com/setup_6.x | bash -
```

```
sudo wget https://dl.yarnpkg.com/rpm/yarn.repo -O /etc/yum.repos.d/yarn.repo
sudo yum install yarn
```

AUR.

yaourt

```
yaourt -S yarn
```

```
sudo eopkg install yarn
```

shell .profile .bashrc .zshrc

```
export PATH="$PATH:`yarn global bin`"
```

Shell

```
curl -o- -L https://yarnpkg.com/install.sh | bash
```

```
curl -o- -L https://yarnpkg.com/install.sh | bash -s -- --version [version]
```

```
cd /opt
wget https://yarnpkg.com/latest.tar.gz
tar zvxf latest.tar.gz
```

NPM

npm

```
npm install -g yarn
```

Yarn

```
yarn --version
```

```
yarn initpackage.json◦ npm initnpm init◦
```

```
yarn init
```

```
mkdir my-package && cd my-package  
yarn init
```

CLI

```
question name (my-package): my-package  
question version (1.0.0):  
question description: A test package  
question entry point (index.js):  
question repository url:  
question author: StackOverflow Documentation  
question license (MIT):  
success Saved package.json  
□ Done in 27.31s.
```

package.json

```
{  
  "name": "my-package",  
  "version": "1.0.0",  
  "description": "A test package",  
  "main": "index.js",  
  "author": "StackOverflow Documentation",  
  "license": "MIT"  
}
```

◦ yarn add [package-name]

ExpressJS

```
yarn add express
```

package.jsondependencies**ExpressJS**

```
"dependencies": {  
  "express": "^4.15.2"  
}
```

Yarn

Yarnnpm◦ npmYarn◦

```
yarn add package ◦
```

```
yarn add package@version
```

◦

yarn add package@tag ◦

<https://riptutorial.com/zh-TW/node-js/topic/9441/>

97:

- `const cluster = require("cluster")`
- `cluster.fork`
- `cluster.isMaster`
- `cluster.isWorker`
- `cluster.schedulingPolicy`
- `cluster.setupMaster`
- `cluster.settings`
- `cluster.worker //worker`
- `cluster.workers //master`

```
cluster.fork() fork() C
```

Node.js

Examples

cluster.js

```
const cluster = require('cluster');
const http = require('http');
const numCPUs = require('os').cpus().length;

if (cluster.isMaster) {
  // Fork workers.
  for (let i = 0; i < numCPUs; i++) {
    cluster.fork();
  }

  cluster.on('exit', (worker, code, signal) => {
    console.log(`worker ${worker.process.pid} died`);
  });
} else {
  // Workers can share any TCP connection
  // In this case it is an HTTP server
  require('./server.js')();
}
```

server.js

```
const http = require('http');

function startServer() {
  const server = http.createServer((req, res) => {
    res.writeHead(200);
    res.end('Hello Http');
  });

  server.listen(3000);
}
```

```

if(!module.parent) {
  // Start server if file is run directly
  startServer();
} else {
  // Export server, if file is referenced via cluster
  module.exports = startServer;
}

```

Webworker。 CPU。 Node.jsCPUNode.js。 8000。 Round-Robin。

Node.js。 Node.js。

cluster。

worker。

```

const cluster = require('cluster');
const http = require('http');
const numCPUs = require('os').cpus().length; //number of CPUs

if (cluster.isMaster) {
  // Fork workers.
  for (var i = 0; i < numCPUs; i++) {
    cluster.fork(); //creating child process
  }

  //on exit of cluster
  cluster.on('exit', (worker, code, signal) => {
    if (signal) {
      console.log(`worker was killed by signal: ${signal}`);
    } else if (code !== 0) {
      console.log(`worker exited with error code: ${code}`);
    } else {
      console.log('worker success!');
    }
  });
} else {
  // Workers can share any TCP connection
  // In this case it is an HTTP server
  http.createServer((req, res) => {
    res.writeHead(200);
    res.end('hello world\n');
  }).listen(3000);
}

```

<https://riptutorial.com/zh-TW/node-js/topic/2817/>

98:

Examples

nodemon

nodemon ◦

nodemon

```
npm install -g nodemon npm i -g nodemon
```

nodemon

```
npm install --save-dev nodemon npm i -D nodemon
```

nodemon

```
nodemon entry.js nodemon entry
```

```
node entry.js node entry ◦
```

nodemonnpm ◦

package.json

```
"scripts": {  
  "start": "nodemon entry.js -devmode -something 1"  
}
```

```
npm start ◦
```

Browsersync

[Browsersync](#) ◦ [NPM](#) ◦

[Browsersync](#) [Node.js](#) [NPM](#) ◦ [Node.js](#) [SO](#) ◦

[Browsersync](#)

```
$ npm install browser-sync -D
```

node_modulesBrowsersync◦

-g-D◦

Windows

WindowsBrowsersyncVisual StudioBrowsersync◦ Visual Studio

```
$ npm install browser-sync --msvs_version=2013 -D
```

Visual Studio2013◦

JavaScript

```
$ browser-sync start --proxy "myproject.dev" --files "**/*.js"
```

myproject.devWeb◦ Browsersync◦

Browsersync[Grunt.js](#)[Gulp.js](#)◦

Grunt.js

Grunt.js

```
$ npm install grunt-browser-sync -D
```

gruntfile.js

```
grunt.loadNpmTasks('grunt-browser-sync');
```

Gulp.js

Browsersync[CommonJS](#)[Gulp.js](#)◦

```
var browserSync = require('browser-sync').create();
```

[Browsersync API](#)◦

API

Browsersync API [https //browsersync.io/docs/api](https://browsersync.io/docs/api)

<https://riptutorial.com/zh-TW/node-js/topic/1743/>

99:

- console.log[data] [...]
- console.error[data] [...]
- console.time
- console.timeEnd

Examples

JavaScriptnode.js。

console.log console.errorconsole.time ◦ console.info ◦

console.log

stdout ◦

```
console.log('Hello World');
```

```
> console.log('Hello World')
Hello World
```

console.error

stderr ◦

```
console.error('Oh, sorry, there is an error.');
```

```
> console.error("Oh, sorry, error");
Oh, sorry, error
```

console.timeconsole.timeEnd

console.time◦ console.timeEndstdout◦

```
> console.time("label");
undefined
> console.timeEnd("label");
label: 9297.320ms
```

◦ process.stdout.write ◦ console.log◦

◦

```
> process.stdout.write("123");process.stdout.write("456");
123456true
```

```
> console.log("\033[31mThis will be red");  
This will be red
```

	\033[0m
	\033[1m
	\033[4m
	\033[7m

	\033[30m
	\033[31m
	\033[32m
	\033[33m
	\033[34m
	\033[35m
	\033[36m
	\033[37m

	\033[40m
	\033[41m
	\033[42m
	\033[43m
	\033[44m
	\033[45m
	\033[46m
	\033[47m

<https://riptutorial.com/zh-TW/node-js/topic/5935/>

100:

- ps10◦ TIA◦

Examples

node.js/◦ ◦ ◦

index.js - ◦

```
//Import Libraries
var express = require('express'),
    session = require('express-session'),
    mongoose = require('mongoose'),
    request = require('request');

//Import custom modules
var userRoutes = require('./app/routes/userRoutes');
var config = require('./app/config/config');

//Connect to Mongo DB
mongoose.connect(config.getDBString());

//Create a new Express application and Configure it
var app = express();

//Configure Routes
app.use(config.API_PATH, userRoutes());

//Start the server
app.listen(config.PORT);
console.log('Server started at - '+ config.URL+ ":" +config.PORT);
```

config.js - ◦

```
var config = {
  VERSION: 1,
  BUILD: 1,
  URL: 'http://127.0.0.1',
  API_PATH : '/api',
  PORT : process.env.PORT || 8080,
  DB : {
    //MongoDB configuration
    HOST : 'localhost',
    PORT : '27017',
    DATABASE : 'db'
  },
  /*
   * Get DB Connection String for connecting to MongoDB database
   */
  getDBString : function(){
    return 'mongodb://' + this.DB.HOST + ':' + this.DB.PORT + '/' + this.DB.DATABASE;
  },
}
```

```

/*
 * Get the http URL
 */
getHttpUrl : function(){
    return 'http://' + this.URL + ":" + this.PORT;
}

module.exports = config;

```

user.js -

```

var mongoose = require('mongoose');
var Schema = mongoose.Schema;

//Schema for User
var UserSchema = new Schema({
  name: {
    type: String,
    // required: true
  },
  email: {
    type: String
  },
  password: {
    type: String,
    //required: true
  },
  dob: {
    type: Date,
    //required: true
  },
  gender: {
    type: String, // Male/Female
    // required: true
  }
});

//Define the model for User
var User;
if(mongoose.models.User)
  User = mongoose.model('User');
else
  User = mongoose.model('User', UserSchema);

//Export the User Model
module.exports = User;

```

UserController - signUp

```

var User = require('../models/user');
var crypto = require('crypto');

//Controller for User
var UserController = {

  //Create a User
  create: function(req, res){
    var repassword = req.body.repassword;

```



```
module.exports = UserController;
```

userRoutes.js - userController

```
var express = require('express');
var UserController = require('../controllers/userController');

//Routes for User
var UserRoutes = function(app)
{
    var router = express.Router();

    router.route('/users')
        .post(UserController.create);

    return router;
}

module.exports = UserRoutes;
```

node.js

<https://riptutorial.com/zh-TW/node-js/topic/6489/>

101:

Examples

nodebackPromises

```
const Promise = require('bluebird'),
      fs = require('fs')

Promise.promisifyAll(fs)

// now you can use promise based methods on 'fs' with the Async suffix
fs.readFileAsync('file.txt').then(contents => {
  console.log(contents)
}).catch(err => {
  console.error('error reading', err)
})
```

```
Promise.resolve([ 1, 2, 3 ]).map(el => {
  return Promise.resolve(el * el) // return some async operation in real world
})
```

```
Promise.resolve([ 1, 2, 3 ]).filter(el => {
  return Promise.resolve(el % 2 === 0) // return some async operation in real world
}).then(console.log)
```

```
Promise.resolve([ 1, 2, 3 ]).reduce((prev, curr) => {
  return Promise.resolve(prev + curr) // return some async operation in real world
}).then(console.log)
```

```
const promiseReturningFunction = Promise.coroutine(function* (file) {
  const data = yield fs.readFileAsync(file) // this returns a Promise and resolves to the file contents

  return data.toString().toUpperCase()
})

promiseReturningFunction('file.txt').then(console.log)
```

Promise.using

```
function somethingThatReturnsADisposableResource() {
  return getSomeResourceAsync(...).disposer(resource => {
    resource.dispose()
  })
}

Promise.using(somethingThatReturnsADisposableResource(), resource => {
  // use the resource here, the disposer will automatically close it when Promise.using exits
})
```

```
Promise.resolve([1, 2, 3])
  .mapSeries(el => Promise.resolve(el * el)) // in real world, use Promise returning async
function
  .then(console.log)
```

<https://riptutorial.com/zh-TW/node-js/topic/6728/>

102: Node.js POST

Node.js。

Node.js。 API。

POST request。 requestdata。

```
request.on('data', chunk => {
  buffer += chunk;
});
request.on('end', () => {
  // POST request body is now available as `buffer`
});
```

data。

1. data [Buffer](#)
2. buffer。

Examples

node.js POST

```
'use strict';

const http = require('http');

const PORT = 8080;
const server = http.createServer((request, response) => {
  let buffer = '';
  request.on('data', chunk => {
    buffer += chunk;
  });
  request.on('end', () => {
    const responseString = `Received string ${buffer}`;
    console.log(`Responding with: ${responseString}`);
    response.writeHead(200, "Content-Type: text/plain");
    response.end(responseString);
  });
}).listen(PORT, () => {
  console.log(`Listening on ${PORT}`);
});
```

[Node.js POST https://riptutorial.com/zh-TW/node-js/topic/5676/node-jspost](https://riptutorial.com/zh-TW/node-js/topic/5676/node-jspost)

103:

NodeJS require() ◦

NodeJS ◦ ◦ require() require ◦

- `module.exports = {testFunction:testFunction};`
- `var test_file = require('./ testFile.js'); //testFile`
- `test_file.testFunctionour_data; //testFiletestFunction`

require() Java ◦ .export “require” ◦ .export ◦

Examples

RequireNodegetter ◦ analysis.js

```
function analyzeWeather(weather_data) {
  console.log('Weather information for ' + weather_data.time + ': ');
  console.log('Rainfall: ' + weather_data.precip);
  console.log('Temperature: ' + weather_data.temp);
  //More weather_data analysis/printing...
}
```

analyzeWeather(weather_data) ◦ ◦ Node ◦

export ◦

```
module.exports = {
  analyzeWeather: analyzeWeather
}
function analyzeWeather(weather_data) {
  console.log('Weather information for ' + weather_data.time + ': ');
  console.log('Rainfall: ' + weather_data.precip);
  console.log('Temperature: ' + weather_data.temp);
  //More weather_data analysis/printing...
}
```

module.exports ◦ require() ◦

require ◦ var const ◦ analyze.js handleWeather.js

```
const analysis = require('./analysis.js');

weather_data = {
  time: '01/01/2001',
  precip: 0.75,
  temp: 78,
  //More weather data...
};
analysis.analyzeWeather(weather_data);
```

require()analysis.js◦ require◦

NPM

[NPM](#)require◦ getWeather.js[NPM](#)require◦ [NPM](#)git install request◦ getWeather.js

```
var https = require('request');

//Construct your url variable...
https.get(url, function(error, response, body) {
  if (error) {
    console.log(error);
  } else {
    console.log('Response => ' + response);
    console.log('Body => ' + body);
  }
});
```

require **S**request◦ requestget◦ [HTTP GET](#)◦

<https://riptutorial.com/zh-TW/node-js/topic/10742/-->

104:

Examples

```
const options = require("commander");

options
  .option("-v, --verbose", "Be verbose");

options
  .command("convert")
  .alias("c")
  .description("Converts input file to output file")
  .option("-i, --in-file <file_name>", "Input file")
  .option("-o, --out-file <file_name>", "Output file")
  .action(doConvert);

options.parse(process.argv);

if (!options.args.length) options.help();

function doConvert(options){
  //do something with options.inFile and options.outFile
};
```

```
const options = require("commander");

options
  .option("-v, --verbose")
  .parse(process.argv);

if (options.verbose){
  console.log("Let's make some noise!");
}
```

<https://riptutorial.com/zh-TW/node-js/topic/6174/>

105: Node.js

Examples

node.js

Node.js◦

```
node debug filename.js
```

debugDemo.js Node.js

```
'use strict';

function addTwoNumber(a, b){
// function returns the sum of the two numbers
  debugger
  return a + b;
}

var result = addTwoNumber(5, 9);
console.log(result);
```

debugger◦

1.

```
cont, c - Continue execution
next, n - Step next
step, s - Step in
out, o - Step out
```

2.

```
setBreakpoint(), sb() - Set breakpoint on current line
setBreakpoint(line), sb(line) - Set breakpoint on specific line
```

```
node debug debugDemo.js
```

◦ process.exit()

```

ankuranand:~/workspace/nodejs/nodejsDebugging $ node debug debugDemo.js
< Debugger listening on port 5858
debug> . ok
break in debugDemo.js:3
  1 // A Demo Code Showing the basic capabilities of the nodejs  debugging module
  2
> 3 'use strict';
  4
  5 function addTwoNumber(a, b){
debug> n
break in debugDemo.js:11
  9 }
 10
>11 let result = addTwoNumber(5, 9);
 12 console.log(result);
 13
debug> c
break in debugDemo.js:7
  5 function addTwoNumber(a, b){
  6 // function returns the sum of the two numbers
> 7 debugger
  8   return a + b;
  9 }
debug> c
< 14
debug> process.exit()
ankuranand:~/workspace/nodejs/nodejsDebugging $ █

```

watch(expression) restart◦

repl◦ repl◦ ◦ Ctrl+C◦

v6.3.0

node v8 [node-inspector](#)◦

URL

```
node --inspect server.js
```

```
npm install -g node-inspector
```

node-debug

```
node-debug filename.js
```

Chrome

```
http://localhost:8080/debug?port=5858
```

8080。

0.0.0.0:8080。 EACCES。

```
$node-inspector --web-port=6500
```

```
1 // A Demo Code Showing the basic capabilities of the nodejs debugging module
2
3 'use strict';
4
5 function addTwoNumber(a, b){
6 // function returns the sum of the two numbers
7   return a + b;
8 }
9
10 var result = addTwoNumber(5, 9);
11 console.log(result);
12
```

10:1 JavaScript Spaces: 4

Node.js <https://riptutorial.com/zh-TW/node-js/topic/5900/node-js>

106:

- **NodeJSbcrypt-NodeJS**◦

Examples

passport.initialize() **Passport** ◦ passport.session()◦

passport.serialize()passport.deserializeUser()◦ **Passport**

```
const express = require('express');
const session = require('express-session');
const passport = require('passport');
const cookieParser = require('cookie-parser');
const app = express();

// Required to read cookies
app.use(cookieParser());

passport.serializeUser(function(user, next) {
  // Serialize the user in the session
  next(null, user);
});

passport.deserializeUser(function(user, next) {
  // Use the previously serialized user
  next(null, user);
});

// Configuring express-session middleware
app.use(session({
  secret: 'The cake is a lie',
  resave: true,
  saveUninitialized: true
}));

// Initializing passport
app.use(passport.initialize());
app.use(passport.session());

// Starting express server on port 3000
app.listen(3000);
```

◦

Node.js◦

```
const passport = require('passport');
const LocalStrategy = require('passport-local').Strategy;

// A named strategy is used since two local strategy are used :
// one for the registration and the other to sign-in
passport.use('localSignup', new LocalStrategy({
  // Overriding defaults expected parameters,
```



```

    // which are 'username' and 'password'
    usernameField: 'email',
    passwordField: 'password',
    passReqToCallback: true // allows us to pass back the entire request to the callback
  },
  function(req, email, password, next) {
    // Check in database if user is already registered
    findUserByEmail(email, function(user) {
      // If email already exists, abort registration process and
      // pass 'false' to the callback
      if (user) return next(null, false);
      // Else, we create the user
      else {
        // Password must be hashed !
        let newUser = createUser(email, password);

        newUser.save(function() {
          // Pass the user to the callback
          return next(null, newUser);
        });
      }
    });
  });
});

```

```

const passport = require('passport');
const LocalStrategy = require('passport-local').Strategy;

passport.use('localSignin', new LocalStrategy({
  usernameField : 'email',
  passwordField : 'password',
}),
function(email, password, next) {
  // Find the user
  findUserByEmail(email, function(user) {
    // If user is not found, abort signing in process
    // Custom messages can be provided in the verify callback
    // to give the user more details concerning the failed authentication
    if (!user)
      return next(null, false, {message: 'This e-mail address is not associated with any
account.'});
    // Else, we check if password is valid
    else {
      // If password is not correct, abort signing in process
      if (!isPasswordValid(password)) return next(null, false);
      // Else, pass the user to callback
      else return next(null, user);
    }
  });
});
});

```

```

// ...
app.use(passport.initialize());
app.use(passport.session());

// Sign-in route
// Passport strategies are middlewares
app.post('/login', passport.authenticate('localSignin', {
  successRedirect: '/me',
  failureRedirect: '/login'

```

```

});

// Sign-up route
app.post('/register', passport.authenticate('localSignup', {
  successRedirect: '/',
  failureRedirect: '/signup'
}));

// Call req.logout() to log out
app.get('/logout', function(req, res) {
  req.logout();
  res.redirect('/');
});

app.listen(3000);

```

Facebook

passport-facebookFacebook

```

const passport = require('passport');
const FacebookStrategy = require('passport-facebook').Strategy;

// Strategy is named 'facebook' by default
passport.use({
  clientID: 'yourclientid',
  clientSecret: 'yourclientsecret',
  callbackURL: '/auth/facebook/callback'
},
// Facebook will send a token and user's profile
function(token, refreshToken, profile, next) {
  // Check in database if user is already registered
  findUserByFacebookId(profile.id, function(user) {
    // If user exists, returns his data to callback
    if (user) return next(null, user);
    // Else, we create the user
    else {
      let newUser = createUserFromFacebook(profile, token);

      newUser.save(function() {
        // Pass the user to the callback
        return next(null, newUser);
      });
    }
  });
});

```

```

// ...
app.use(passport.initialize());
app.use(passport.session());

// Authentication route
app.get('/auth/facebook', passport.authenticate('facebook', {
  // Ask Facebook for more permissions
  scope : 'email'
}));

// Called after Facebook has authenticated the user

```

```

app.get('/auth/facebook/callback',
  passport.authenticate('facebook', {
    successRedirect : '/me',
    failureRedirect : '/'
  }));

//...

app.listen(3000);

```

routes / index.js

userSchema

```

router.post('/login', function(req, res, next) {
  if (!req.body.username || !req.body.password) {
    return res.status(400).json({
      message: 'Please fill out all fields'
    });
  }

  passport.authenticate('local', function(err, user, info) {
    if (err) {
      console.log("ERROR : " + err);
      return next(err);
    }

    if(user) {
      console.log("User Exists!")
      //All the data of the user can be accessed by user.x
      res.json({"success" : true});
      return;
    } else {
      res.json({"success" : false});
      console.log("Error" + errorResponse());
      return;
    }
  })(req, res, next);
});

```

Google Passport

npm passport-google-oauth20

passport.js google.js config. app.js

```

var express = require('express');
var session = require('express-session');
var passport = require('./config/passport'); // path where the passport file placed
var app = express();
passport(app);

```

```
//
```

configpassport.js

```
var passport = require ('passport'),
    google = require('./google'),
    User = require('./../model/user'); // User is the mongoose model

module.exports = function(app){
  app.use(passport.initialize());
  app.use(passport.session());
  passport.serializeUser(function(user, done){
    done(null, user);
  });
  passport.deserializeUser(function (user, done) {
    done(null, user);
  });
  google();
};
```

google.js

```
var passport = require('passport'),
    GoogleStrategy = require('passport-google-oauth20').Strategy,
    User = require('./../model/user');
module.exports = function () {
  passport.use(new GoogleStrategy({
    clientID: 'CLIENT ID',
    clientSecret: 'CLIENT SECRET',
    callbackURL: "http://localhost:3000/auth/google/callback"
  },
  function(accessToken, refreshToken, profile, cb) {
    User.findOne({ googleId : profile.id }, function (err, user) {
      if(err){
        return cb(err, false, {message : err});
      }else {
        if (user != '' && user != null) {
          return cb(null, user, {message : "User "});
        } else {
          var username = profile.displayName.split(' ');
          var userData = new User({
            name : profile.displayName,
            username : username[0],
            password : username[0],
            facebookId : '',
            googleId : profile.id,
          });
          // send email to user just in case required to send the newly created
          // credentials to user for future login without using google login
          userData.save(function (err, newuser) {
            if (err) {
              return cb(null, false, {message : err + " !!! Please try again"});
            }else{
              return cb(null, newuser);
            }
          });
        }
      }
    });
  });
};
```

```
    }  
  });  
};
```

DBgoogleIdDB。

<https://riptutorial.com/zh-TW/node-js/topic/7666/>

107:

Examples

MongooseMongoDB

Mongoose

```
npm install mongoose
```

server.js

```
var mongoose = require('mongoose');  
var Schema = mongoose.Schema;
```

```
var schemaName = new Schema({  
  request: String,  
  time: Number  
}, {  
  collection: 'collectionName'  
});
```

```
var Model = mongoose.model('Model', schemaName);  
mongoose.connect('mongodb://localhost:27017/dbName');
```

MongoDBserver.jsnode server.js

mongoose.connectionopen error ◦

```
var db = mongoose.connection;  
db.on('error', console.error.bind(console, 'connection error:'));  
db.once('open', function() {  
  // we're connected!  
});
```

MongooseExpress.jsMongoDB

```
npm install express cors mongoose
```

server.jsExpress.jsMongoDB

```
var express = require('express');  
var cors = require('cors'); // We will use CORS to enable cross origin domain requests.  
var mongoose = require('mongoose');  
var Schema = mongoose.Schema;
```

```

var app = express();

var schemaName = new Schema({
  request: String,
  time: Number
}, {
  collection: 'collectionName'
});

var Model = mongoose.model('Model', schemaName);
mongoose.connect('mongodb://localhost:27017/dbName');

var port = process.env.PORT || 8080;
app.listen(port, function() {
  console.log('Node.js listening on port ' + port);
});

```

Express.js

```

app.get('/save/:query', cors(), function(req, res) {
  var query = req.params.query;

  var savedata = new Model({
    'request': query,
    'time': Math.floor(Date.now() / 1000) // Time of save the data in unix timestamp
format
  }).save(function(err, result) {
    if (err) throw err;

    if(result) {
      res.json(result)
    }
  })
})

```

query HTTP<query> MongoDB

```

var savedata = new Model({
  'request': query,
  //...

```

MongoDB。JSON。

```

//...
}).save(function(err, result) {
  if (err) throw err;

  if(result) {
    res.json(result)
  }
})
//...

```

MongoDB node server.js server.js

URL

```
http://localhost:8080/save/<query>
```

<query>°

```
http://localhost:8080/save/JavaScript%20is%20Awesome
```

JSON

```
{
  __v: 0,
  request: "JavaScript is Awesome",
  time: 1469411348,
  _id: "57957014b93bc8640f2c78c4"
}
```

MongooseExpress.jsMongoDB

```
npm install express cors mongoose
```

server.js **Express.jsMongoDB**

```
var express = require('express');
var cors = require('cors'); // We will use CORS to enable cross origin domain requests.
var mongoose = require('mongoose');
var Schema = mongoose.Schema;

var app = express();

var schemaName = new Schema({
  request: String,
  time: Number
}, {
  collection: 'collectionName'
});

var Model = mongoose.model('Model', schemaName);
mongoose.connect('mongodb://localhost:27017/dbName');

var port = process.env.PORT || 8080;
app.listen(port, function() {
  console.log('Node.js listening on port ' + port);
});
```

Express.js

```
app.get('/find/:query', cors(), function(req, res) {
  var query = req.params.query;
```



```

Model.find({
  'request': query
}, function(err, result) {
  if (err) throw err;
  if (result) {
    res.json(result)
  } else {
    res.send(JSON.stringify({
      error : 'Error'
    }))
  }
})
})

```

```

{
  "_id" : ObjectId("578abe97522ad414b8eeb55a"),
  "request" : "JavaScript is Awesome",
  "time" : 1468710551
}
{
  "_id" : ObjectId("578abe9b522ad414b8eeb55b"),
  "request" : "JavaScript is Awesome",
  "time" : 1468710555
}
{
  "_id" : ObjectId("578abea0522ad414b8eeb55c"),
  "request" : "JavaScript is Awesome",
  "time" : 1468710560
}

```

"request" "JavaScript is Awesome"◦

MongoDB server.js node server.js

URL

```
http://localhost:8080/find/<query>
```

<query>◦

```
http://localhost:8080/find/JavaScript%20is%20Awesome
```

```

[ {
  _id: "578abe97522ad414b8eeb55a",
  request: "JavaScript is Awesome",
  time: 1468710551,
  __v: 0
},
  {
  _id: "578abe9b522ad414b8eeb55b",
  request: "JavaScript is Awesome",
  time: 1468710555,
  __v: 0
},

```

```
{
  _id: "578abea0522ad414b8eeb55c",
  request: "JavaScript is Awesome",
  time: 1468710560,
  __v: 0
}]
```

MongooseExpress.js\$ textMongoDB

```
npm install express cors mongoose
```

server.jsExpress.jsMongoDB

```
var express = require('express');
var cors = require('cors'); // We will use CORS to enable cross origin domain requests.
var mongoose = require('mongoose');
var Schema = mongoose.Schema;

var app = express();

var schemaName = new Schema({
  request: String,
  time: Number
}, {
  collection: 'collectionName'
});

var Model = mongoose.model('Model', schemaName);
mongoose.connect('mongodb://localhost:27017/dbName');

var port = process.env.PORT || 8080;
app.listen(port, function() {
  console.log('Node.js listening on port ' + port);
});
```

Express.js

```
app.get('/find/:query', cors(), function(req, res) {
  var query = req.params.query;

  Model.find({
    'request': query
  }, function(err, result) {
    if (err) throw err;
    if (result) {
      res.json(result)
    } else {
      res.send(JSON.stringify({
        error : 'Error'
      }))
    }
  })
})
```

```

{
  "_id" : ObjectId("578abe97522ad414b8eeb55a"),
  "request" : "JavaScript is Awesome",
  "time" : 1468710551
}
{
  "_id" : ObjectId("578abe9b522ad414b8eeb55b"),
  "request" : "JavaScript is Awesome",
  "time" : 1468710555
}
{
  "_id" : ObjectId("578abea0522ad414b8eeb55c"),
  "request" : "JavaScript is Awesome",
  "time" : 1468710560
}

```

"request" "JavaScript"◦

"request" ◦ server.js

```
schemaName.index({ request: 'text' });
```

```

Model.find({
  'request': query
}, function(err, result) {

```

```

Model.find({
  $text: {
    $search: query
  }
}, function(err, result) {

```

`$text``$search` **MongoDBcollection** `collectionName`◦

URL

```
http://localhost:8080/find/<query>
```

<query>◦

```
http://localhost:8080/find/JavaScript
```

```

[ {
  _id: "578abe97522ad414b8eeb55a",
  request: "JavaScript is Awesome",
  time: 1468710551,
  __v: 0
},
{
  _id: "578abe9b522ad414b8eeb55b",
  request: "JavaScript is Awesome",

```

```

    time: 1468710555,
    __v: 0
  },
  {
    _id: "578abead0522ad414b8eeb55c",
    request: "JavaScript is Awesome",
    time: 1468710560,
    __v: 0
  }
}]

```

◦

MongoDB。 Mongoose。 ◦

```

var strConnection = 'mongodb://localhost:27017/dbName';
var db = mongoose.createConnection(strConnection)

```

```

var Schema = require('mongoose').Schema;
var usersSchema = new Schema({
  username: {
    type: String,
    required: true,
    unique: true
  },
  email: {
    type: String,
    required: true
  },
  password: {
    type: String,
    required: true
  },
  created: {
    type: Date,
    default: Date.now
  }
});

var userModel = db.model('users', usersSchema);
module.exports = userModel;

```

mongoose。 ◦

_ID

_id。 ObjectIdMongoDB。 _id。 ◦

```

var usersSchema = new Schema({
  username: {
    type: String,
    required: true,
    unique: true
  }, {
    _id: false
  }
});

```

__vversionKey

versionKeyMongoose◦ ◦ ◦

```
var usersSchema = new Schema({
  username: {
    type: String,
    required: true,
    unique: true
  }, {
    versionKey: false
  });
```

Mongoose◦

```
usersSchema.index({username: 1 });
usersSchema.index({email: 1 });
```

◦ ◦

```
usersSchema.index({username: 1, email: 1 });
```

mongooseensureIndexensureIndex'index'◦

MongoDBensureIndex3.0.0createIndex◦

autoIndexfalseconfig.autoIndexfalse◦

```
usersSchema.set('autoIndex', false);
```

Mongoose

Mongoosefind()◦

```
doc.find({'some.value':5},function(err,docs){
  //returns array docs
});

doc.findOne({'some.value':5},function(err,doc){
  //returns document doc
});

doc.findById(obj._id,function(err,doc){
  //returns document doc
});
```

promisesmongodb

```
npm install express cors mongoose
```

server.js Express.js MongoDB

```
var express = require('express');
var cors = require('cors'); // We will use CORS to enable cross origin domain requests.
var mongoose = require('mongoose');
var Schema = mongoose.Schema;

var app = express();

var schemaName = new Schema({
  request: String,
  time: Number
}, {
  collection: 'collectionName'
});

var Model = mongoose.model('Model', schemaName);
mongoose.connect('mongodb://localhost:27017/dbName');

var port = process.env.PORT || 8080;
app.listen(port, function() {
  console.log('Node.js listening on port ' + port);
});

app.use(function(err, req, res, next) {
  console.error(err.stack);
  res.status(500).send('Something broke!');
});

app.use(function(req, res, next) {
  res.status(404).send('Sorry cant find that!');
});
```

Express.js

```
app.get('/find/:query', cors(), function(req, res, next) {
  var query = req.params.query;

  Model.find({
    'request': query
  })
  .exec() //remember to add exec, queries have a .then attribute but aren't promises
  .then(function(result) {
    if (result) {
      res.json(result)
    } else {
      next() //pass to 404 handler
    }
  })
  .catch(next) //pass to error handler
});
```

```
{
  "_id" : ObjectId("578abe97522ad414b8eeb55a"),
  "request" : "JavaScript is Awesome",
  "time" : 1468710551
}
```

```
    "_id" : ObjectId("578abe9b522ad414b8eeb55b"),
    "request" : "JavaScript is Awesome",
    "time" : 1468710555
  }
  {
    "_id" : ObjectId("578abea0522ad414b8eeb55c"),
    "request" : "JavaScript is Awesome",
    "time" : 1468710560
  }
}
```

"request" "JavaScript is Awesome"◦

MongoDB server.js node server.js

URL

```
http://localhost:8080/find/<query>
```

<query>◦

```
http://localhost:8080/find/JavaScript%20is%20Awesome
```

```
[{
  _id: "578abe97522ad414b8eeb55a",
  request: "JavaScript is Awesome",
  time: 1468710551,
  __v: 0
},
{
  _id: "578abe9b522ad414b8eeb55b",
  request: "JavaScript is Awesome",
  time: 1468710555,
  __v: 0
},
{
  _id: "578abea0522ad414b8eeb55c",
  request: "JavaScript is Awesome",
  time: 1468710560,
  __v: 0
}]
```

<https://riptutorial.com/zh-TW/node-js/topic/3486/>

108: Mongodb

MongoDB。 MongoDBNoSQLJSON。

<https://www.mongodb.com/>

- MongoClient.connect('mongodb://127.0.0.1:27017 / crud'functionerrdb{// do womething here};

Examples

Node.JSmongoDB

```
MongoClient.connect('mongodb://localhost:27017/myNewDB',function (err,db) {
  if(err)
    console.log("Unable to connect DB. Error: " + err)
  else
    console.log('Connected to DB');

  db.close();
});
```

myNewDB。

mongoDBNode.JS

```
var MongoClient = require('mongodb').MongoClient;

//connection with mongoDB
MongoClient.connect("mongodb://localhost:27017/MyDb", function (err, db) {
  //check the connection
  if(err){
    console.log("connection failed.");
  }else{
    console.log("successfully connected to mongoDB.");
  }
});
```

Mongodb <https://riptutorial.com/zh-TW/node-js/topic/6280/mongodb>

109:

Examples

GitHub◦ npm

```
$ npm install --save async
```

Bower

```
$ bower
```

```
var async = require("async");
async.parallel([
  function(callback) { ... },
  function(callback) { ... }
], function(err, results) {
  // optional callback
});
```

Async.js◦ Async◦

Async◦ ◦

Async◦ ◦ ◦ ◦ NodeAsync◦

<https://github.com/caolan/async>◦ npm

```
$ npm install --save async
```

Bower

```
$ bower
```

Async

```
var fs = require('fs');
var async = require('async');

var myFile = '/tmp/test';

async.waterfall([
  function(callback) {
    fs.readFile(myFile, 'utf8', callback);
  },
  function(txt, callback) {
    txt = txt + '\nAppended something!';
    fs.writeFile(myFile, txt, callback);
  }
], function (err, result) {
  if(err) return console.log(err);
```

```
    console.log('Appended text!');  
  });
```

<https://riptutorial.com/zh-TW/node-js/topic/10045/>

110:

nodejs。 nodejsv8。

chrome devtoolschrome inspector。

bebugdevtools。 devtools。

node-inspector。

Examples

1 npmnode-inspector

```
$ npm install -g node-inspector
```

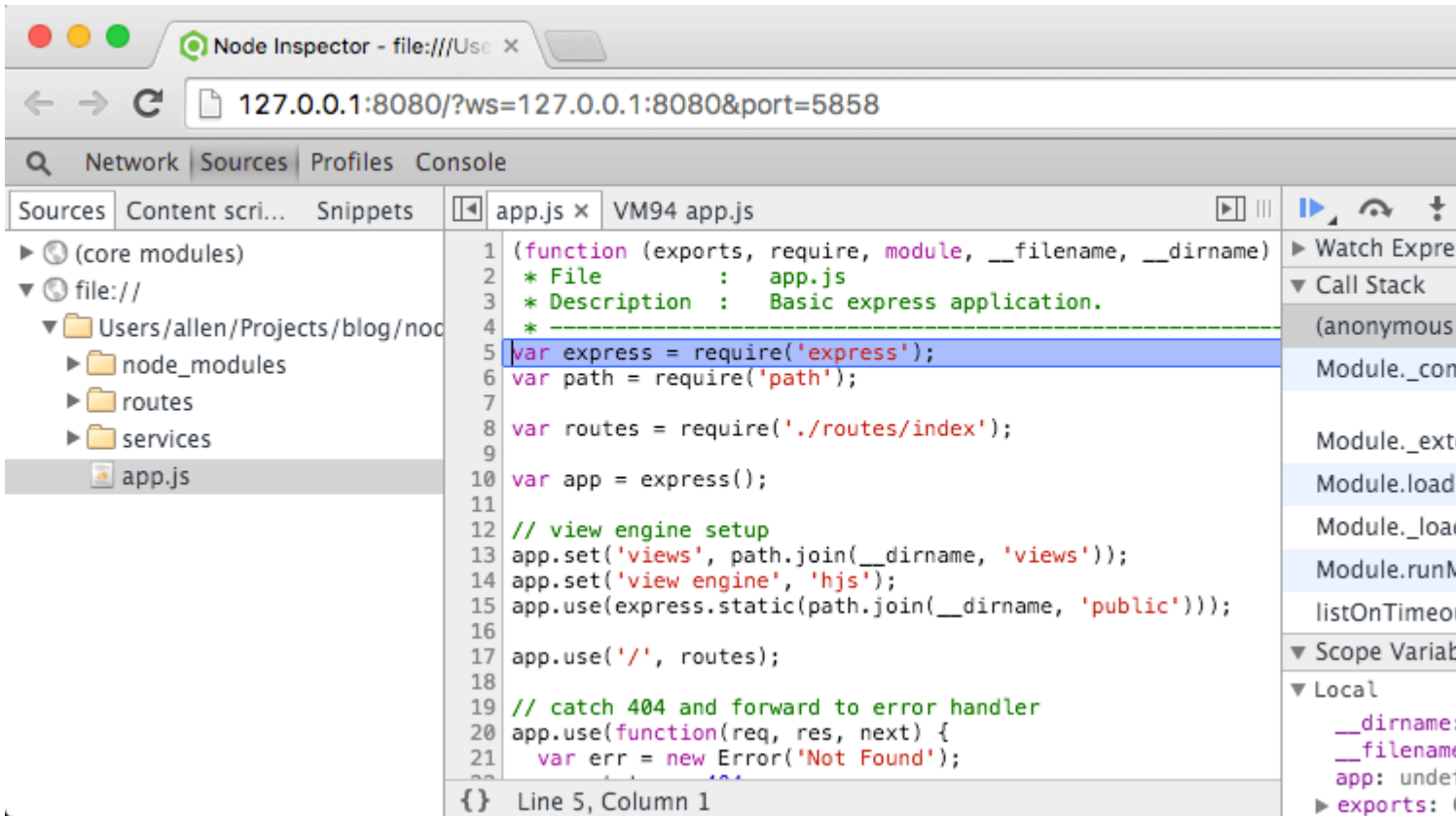
2 node-inspector

```
$ node-inspector
```

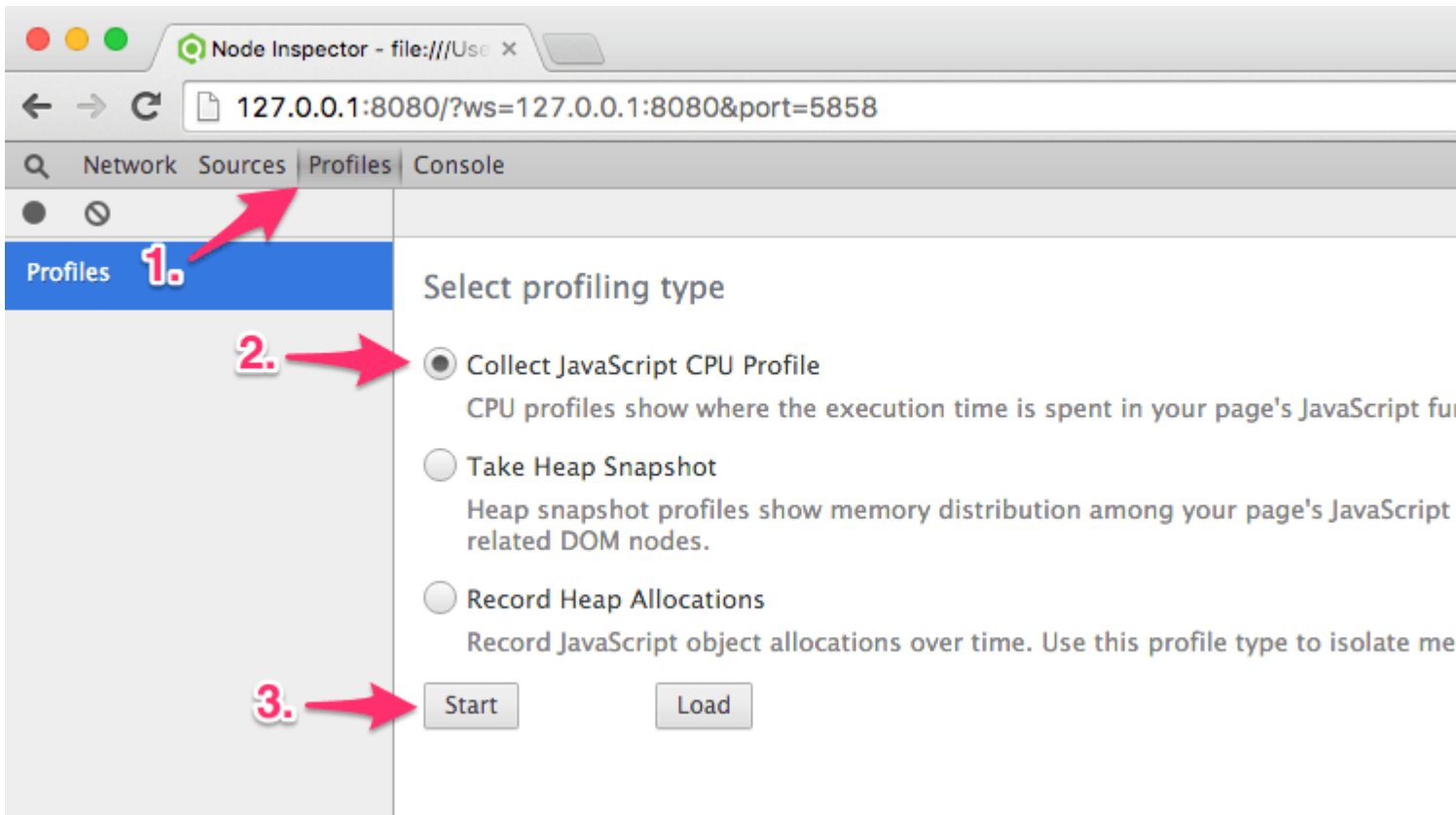
3

```
$ node --debug-brk your/short/node/script.js
```

4 Chrome<http://127.0.0.1:8080/?port=5858>。 nodejschrom-dev。 debug break。



5 . .



6 CPU/.

- CPU

- [Chrome CPU](#)

<https://riptutorial.com/zh-TW/node-js/topic/9347/>

111:

nodejs。 arc'。 MVC。 APICRUD

browserifyvue.js。 mvcpblicmvc。 ◦

Examples

nodejsMVCAPI

-
- config◦

```
|-- Config
  |-- config.json
  |-- appConfig
    |-- pets.config
    |-- payment.config
```

- /。 2webapp◦ src◦
webapp◦ ◦
- App.js / index.jsnodejs◦ ◦ dtoAPI◦

```
|-- server
  |-- dto
    |-- pet.js
    |-- payment.js
  |-- controller
    |-- PetsController.js
    |-- PaymentController.js
  |-- App.js
```

- webapppublicmvc◦ [browserifywebappMVCmvc◦](#)

| - webapp | - public | - mvc

- csssaasHTML◦

```
|-- public
  |-- build // will contianed minified scripts(mvc)
  |-- images
    |-- mouse.jpg
    |-- cat.jpg
  |-- styles
    |-- style.css
  |-- views
    |-- petStore.html
    |-- paymentGateway.html
```

```
|-- header.html
|-- footer.html
|-- index.html
```

- *mvc* *Utils*。 *index.jshell.js*。

```
|-- mvc
  |-- controllers
    |-- Dashborad.js
    |-- Help.js
    |-- Login.js
  |-- utils
  |-- index.js
```

below.AndbrowserifyMVC。 ***express.usesatic'public'*** api。

```
|-- node_modules
|-- src
  |-- server
    |-- controller
    |-- App.js // node app
  |-- webapp
    |-- public
      |-- styles
      |-- images
      |-- index.html
    |-- mvc
      |-- controller
      |-- shell.js // mvc shell
|-- config
|-- Readme.md
|-- .gitignore
|-- package.json
```

<https://riptutorial.com/zh-TW/node-js/topic/9935/>

S. No		Contributors
1	Node.js	4444 , Abdelaziz Mokhnache , Abhishek Jain , Adam , Aeolingamenfel , Alessandro Trinca Tornidor , Aljoscha Meyer , Amila Sampath , Ankit Gomkale , Ankur Anand , arcs , Aule , B Thuy , baranskistad , Bundit J. , Chandra Sekhar , Chezzwizz , Christopher Ronning , Community , Craig Ayre , David Gatti , Djizeus , Florian Hämmerle , Franck Dernoncourt , ganesshkumar , George Aidonidis , Harangue , hexacyanide , Iain Reid , Inanc Gumus , Jason , Jasper , Jeremy Banks , John Slegers , JohnnyCoder , Joshua Kleveter , KolesnichenkoDS , krishgopinath , Léo Martin , Majid , Marek Skiba , Matt Bush , Meinkraft , Michael Irigoyen , Mikhail , Milan Laslop , ndugger , Nick , olegzhermal , Peter Mortensen , RamenChef , Reborn , Rishikesh Chandra , Shabin Hashim , Shiven , Sibeesh Venu , sigfried , SteveLacy , Susanne Oberhauser , thefourtheye , theunexpected1 , Tomás Cañibano , user2314737 , Volodymyr Sichka , xam , zurfyx
2	ArduinonodeJs	sBanda
3	async.js	David Knipe , devnull69 , DrakaSAN , F. Kauder , jerry , Isampaio , Shriganesh Kolhe , Sky , walid
4	CLI	Ze Rubeus
5	ExpressJSRoute-Controller-Service	nomanbinhussein
6	HTTP	Ahmed Metwally
7	Koa Framework v2	David Xu
8	Lodash	M1kstur
9	metalsmith	RamenChef , vsjn3290ckjnaoij2jikndckjb
10	Mongodb	cyanbeam , FabianCook , midnightsyntax
11	MSSQL	damitj07
12	Mysql	KlwntSingh
13	MySQL	Aminadav , Andrés Encarnación , Florian Hämmerle , Ivan Schwarz , jdrydn , JohnnyCoder , Kapil Vats , KlwntSingh , Marek Skiba , Rafael Gadotti Bachovas , RamenChef , Simplans ,

		Sorangwala Abbasali , surjikal
14	Node.js / Express.js MongoDB	William Carron
15	Node.js STDIN STDOUT	Syam Pradeep
16	Node.js v6	creyD , DominicValenciana , KlwntSingh
17	node.jsWindows	CJ Harries
18	Node.JS	Rick , VooVoo
19	Node.JSMongoDB	midnightsyntax , RamenChef , Satyam S
20	Node.js	Florian Hämmerle , Inanc Gumus
21	Node.js	Ivan Hristov
22	Node.js	vintproykt
23	Node.jsCORS	Buzut
24	Node.JSES6	Inanc Gumus , xam , ymz , zurfyx
25	Node.jsOracle	oliolioli
26	Node.js	Ankur Anand , pietrovismara
27	Node.js	Karlen
28	nodejs	Craig Ayre , Veger
29	NodeJS	Niroshan Ranapathi
30	NodeJS	dthree
31	Nodejs	Kelum Senanayake
32	NodeJSRedis	evalsocket
33	NodeJs	parlad neupane
34	NPM	Abhishek Jain , AJS , Amreesh Tyagi , Ankur Anand , Asaf Manassen , Ates Goral , ccnokes , CD. , Cristian Cavalli , David G. , DrakaSAN , Eric Fortin , Everettss , Explosion Pills , Florian Hämmerle , George Bailey , hexacyanide , HungryCoder , Ionică Bizău , James Taylor , João Andrade , John Slegers , Jojodmo , Josh , Kid Binary , Loufylouf , m02ph3u5 , Matt , Matthew Harwood , Mehdi El Fadil , Mikhail , Mindsers , Nick , notgiorgi , num8er ,

		oscar , Pete TNT , Philipp Flenker , Pieter Herroelen , Pyloid , QoP , Quill , Rafal Wiliński , RamenChef , Ratan Kumar , RationalDev , rdegges , refaelos , Rizowski , Shiven , Skanda , Sorangwala Abbasali , still_learning , subbu , the12 , tlo , Un3qual , uzaif , VladNeacsu , Vsevolod Goloviznin , Wasabi Fan , Yerko Palma
35	nvm -	cyanbeam , guleria , John Vincent Jardin , Luis González , pranspach , Shog9 , Tushar Gupta
36	N-API	Parham Alvani
37	OAuth 2.0	tyehia
38	passport.js	Red
39	PostgreSQL	Niroshan Ranapathi
40	Restful API	fresh5447 , nilakantha singh deo
41	Sequelize.js	Fikra , Niroshan Ranapathi , xam
42	Socket.io	Forivin , N.J.Dawson
43	TCP	B Thuy
44		Aikon Mogwai , Iceman , Mikhail , walid
45		Kelum Senanayake
46		DrakaSAN , Duly Kinsky , Florian Hämmerle , jamescostian , MindlessRanger , Mothman
47	Browserfy [™]	Big Dude
48	Express.JSajax	RamenChef , SynapseTech
49	ExpressWeb	Aikon Mogwai , Alex Logan , alexi2 , Andres C. Viesca , Aph , Asaf Manassen , Batsu , bekce , brianmearns , Community , Craig Ayre , Daniel Verem , devnull69 , Everettss , Florian Hämmerle , H. Pauwelyn , Inanc Gumus , jemiloi , Kid Binary , kunerd , Marek Skiba , Mikhail , Mohit Gangrade , Mukesh Sharma , Naeem Shaikh , Niklas , Nivesh , noob , Ojen , Pasha Rumkin , Paul , Rafal Wiliński , Shabin Hashim , SteveLacy , tandrewnichols , Taylor Ackley , themole , tverdohle , Vsevolod Goloviznin , xims , Yerko Palma
50	IISNodeIISNode.js Web	peteb

51	Node.jsAPI	Mukesh Sharma
52	Streams	cyanbeam , Duly Kinsky , efeder , johni , KlwntSingh , Max , Ze Rubeus
53		guleria , hexacyanide , iSkore
54	Node.JSWebSocket	Rowan Harley
55		Niroshan Ranapathi
56		Alex Logan , Bearington , cyanbeam , Himani Agrawal , Mikhail , mscdex , optimus , pietrovismara , RamenChef , Sameer Srivastava , somebody , Taylor Swanson
57	Node.js	akinjide , devnull69 , Florian Hämmerle , John Slegers , Mukesh Sharma , Pauly Garcia , Peter G , pranspach , RamenChef , Simplans
58		RamenChef , Sathish
59		Aminadav , Craig Ayre , cyanbeam , devnull69 , DrakaSAN , Fenton , Florian Hämmerle , hexacyanide , Jason , jdrydn , LoufyLouf , Louis Barranqueiro , m02ph3u5 , Marek Skiba , MrWhiteNerdy , MSB , Pedro Otero , Shabin Hashim , tkone , uzaif
60	PromiseError-First Node.js	Dave
61		signal
62		Vsevolod Goloviznin
63	Node.js.	John Vincent Jardin , RamenChef , snuggles08 , Trevor Clarke
64		Naeem Shaikh , Waterscroll
65		David Xu , Florian Hämmerle , skiilaa
66		Clement JACOB , Michael Buen , Sanketh Katta
67	node.js	AndrewLeonardi , Bharat , commonSenseCode , James Billingham , Oliver , sharif.io , Shog9
68	Node.js	Apidcloud , Brett Jackson , Community , Cristian Boariu , duncanhall , Florian Hämmerle , guleria , haykam , KlwntSingh , Mad Scientist , MatthieuLemoine , Mukesh Sharma , raghu , sjmarshy , tverdohle , tyehia
69	RESTCRUD API	Iceman

70		arcs
71		RamenChef , umesh
72	Node.js.	Alister Norris , Aminadav , Anh Cao , asherbar , Batsu , Buzut , Chance Snow , Chezzwizz , Dmitriy Borisov , Florian Hämmerle , GilZ , guleria , hexacyanide , HungryCoder , Inanc Gumus , Jacek Labuda , John Vincent Jardin , Josh , KahWee Teng , Maciej Rostański , mmhyamin , Naing Lin Aung , NuSkooler , Shabin Hashim , Siddharth Srivastva , Sveratum , tandrewnichols , user2314737 , user6939352 , V1P3R , victorkohl
73	-	Zoltán Schmidt
74	node.js	Buzut
75		Beshoy Hanna
76	angular.js Node.js express.js	sigfried
77	Node.js ECMAScript 2015ES6	David Xu , Florian Hämmerle , Osama Bari
78		Antenka , SteveLacy
79		Mario Rozic
80	HTML	Himani Agrawal , RamenChef , user2314737
81	MongoDB Mongoose	zurfyx
82	I / O.	4444 , Accepted Answer , Aeolingamenfel , Christophe Marois , Craig Ayre , DrakaSAN , Duly Kinsky , Florian Hämmerle , gnerkus , Harshal Bhamare , hexacyanide , jakerella , Julien CROUZET , Louis Barranqueiro , midnightsyntax , Mikhail , peteb , Shiven , still_learning , Tim Jones , Tropic , Vsevolod Goloviznin , Zanon
83		Aikon Mogwai
84	Node.js	gentlejo
85	- REST	Roopesh
86		Chris , Freddie Coleman , KlwntSingh , Louis Barranqueiro , Mikhail , sBanda
87		KlwntSingh , Nivesh , riyadhالنور , sBanda , sjmarshy , topheman

88	/	Cami Rodriguez , Cody G. , cyanbeam , Dave , David Xu , Dom Vinyard , m_callens , Manuel , nomanbinhussein , Toni Villena
89		Ala Eddine JEBALI , cyanbeam , Florian Hämmerle , H. Pauwelyn , John , Marek Skiba , Native Coder , omgimanerd , slowdeath007
90	Web	Housseem Yahiaoui
91	package.json	Ankur Anand , Asaf Manassen , Chance Snow , efeder , Eric Smekens , Florian Hämmerle , Jaylem Chaudhari , Kornel , lauriys , mezzode , OzW , RamenChef , Robbie , Shabin Hashim , Simplans , SteveLacy , Sven 31415 , Tomás Cañibano , user6939352 , V1P3R , victorkohl
92	ReadLine	4444 , Craig Ayre , Florian Hämmerle , peteb
93	jscsv	aisflat439
94	JS	Osama Bari
95		Hasan A Yousef , Taylor Ackley
96		Andrew Brooke , skiilaa
97		Benjamin , Florian Hämmerle , Kid Binary , MayorMonty , Mukesh Sharma , riyadhainur , Vsevolod Goloviznin
98		ch4nd4n , Dean Rather , Jonas S , Joshua Kleveter , Nivesh , Sanketh Katta , zurfyx
99		ScientiaEtVeritas
100		Ajitej Kaushik , RamenChef
101		David Xu
102	Node.jsPOST	Manas Jayanth
103		Philip Cornelius Glover
104		yrtimiD
105	Node.js	4444 , Alister Norris , Ankur Anand , H. Pauwelyn , Matthew Shanley
106		Ankit Rana , Community , Léo Martin , M. A. Cordeiro , Rupali Pemare , shikhar bansal
107		Alex Logan , manuerumx , Mikhail , Naeem Shaikh , Qiong Wu , Simplans , Will

108	Mongodb	FabianCook , Nainesh Raval , Shriganesh Kolhe
109		tyehia
110		damitj07
111		damitj07