
COMP 431/531: Web Development

Lecture 16: Backend - Express

Mack Joyner (mjoyner@rice.edu)

<https://www.clear.rice.edu/comp431>



Back End Development

PART II
Web Servers
Backend
Architecture
Testing
Web Hosting
Databases



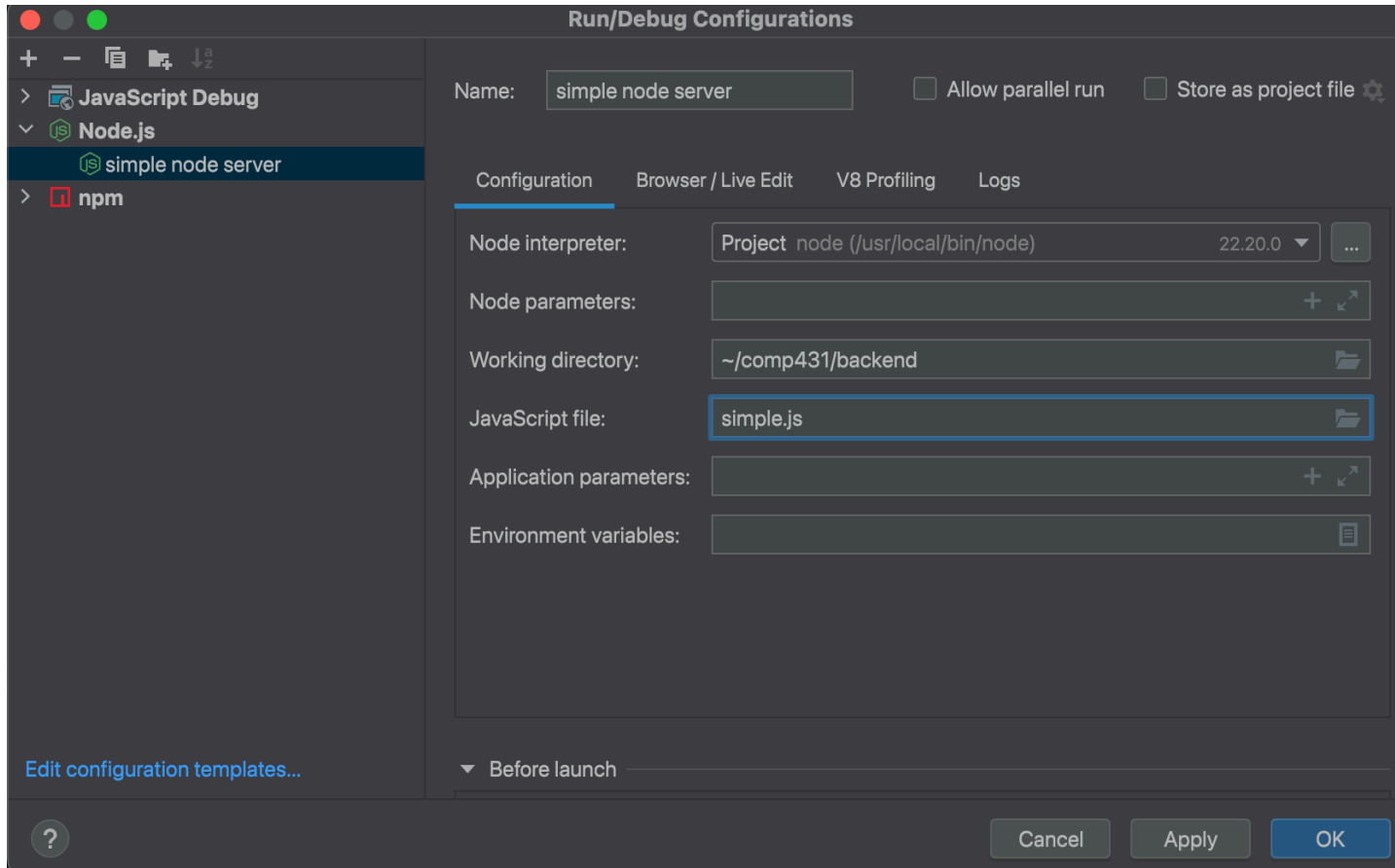
Announcements & Reminders

HW #5 (Front-end) due Thu. Nov. 6th at 11:59pm

classroom hw5 repo: <https://classroom.github.com/a/hlFk8cgA>



Debugging Node.js



<https://www.jetbrains.com/help/idea/running-and-debugging-node-js.html>

<https://plugins.jetbrains.com/plugin/6098-nodejs>



Talend API Tester

The screenshot displays the Talend API Tester interface. At the top, the 'Requests' tab is active. The method is set to 'GET' and the URL is 'http://localhost:3000/articles'. A 'Send' button is visible on the right. Below the URL bar, there are sections for 'QUERY PARAMETERS', 'HEADERS', and 'BODY'. The 'HEADERS' section has a '+ Add header' button and a link to 'Add authorization'. The 'BODY' section shows a message: 'XHR does not allow payloads for GET request.' Below this, the 'Response' section is expanded, showing a '200 OK' status. The response headers are listed on the left, including 'X-Powered-By: Express', 'Content-Type: application/json; charset=utf-8', 'Content-Length: 124 bytes', 'ETag: W/"7c-V303w57HRd1LM0R39NQbnp+QjyU"', 'Date: Wed, 04 Oct 2023 22:00:51 GMT', 'Connection: keep-alive', and 'Keep-Alive: timeout=5'. The response body is shown on the right, formatted as JSON, containing an array of two objects: one with 'id: 0', 'author: "Mack"', and 'body: "Post 1"', and another with 'id: 1', 'author: "Jack"', and 'body: "Post 2"'. A 'Cache Detected - Elapsed Time: 23ms' message is visible in the top right of the response section.

GET | http://localhost:3000/articles | Send

length: 30 byte(s)

QUERY PARAMETERS

HEADERS | Form | BODY

+ Add header | Add authorization

XHR does not allow payloads for GET request.

Response | Cache Detected - Elapsed Time: 23ms

200 OK

HEADERS | pretty | BODY | pretty

X-Powered-B... Express
Content-Typ... application/json; charset=utf-8
Content-Len... 124 bytes
ETag: W/"7c-V303w57HRd1LM0R39NQbnp+QjyU"
Date: Wed, 04 Oct 2023 22:00:51 GMT
Connection: keep-alive
Keep-Alive: timeout=5

COMPLETE REQUEST HEADERS

```
[
  {
    id: 0,
    author: "Mack",
    body: "Post 1"
  },
  {
    id: 1,
    author: "Jack",
    body: "Post 2"
  }
]
```

Add as a Chrome extension from Chrome web store



Talend API Tester

METHOD: POST SCHEME :// HOST [":" PORT] [PATH ["?" QUERY]]

http://localhost:3000/article length: 29 byte(s) Send

QUERY PARAMETERS

HEADERS ¹ Form

☒ Content-Type : application/json x

+ Add header Add authorization trash

BODY ¹ Text

```
1 {
2   "author": "Mack",
3   "body": "Here is a new post"
4 }
```

Text JSON XML HTML Format body ☒ Enable body evaluation length: 54 bytes

200 OK

HEADERS ¹ pretty

X-Powered-By: Express
Content-Type: application/json; charset=utf-8
Content-Length: 177 bytes
ETag: W/"b1-lu1PoDCSL7sutIFxWvTCS6KH9E"
Date: Wed, 04 Oct 2023 22:05:14 GMT
Connection: keep-alive
Keep-Alive: timeout=5

COMPLETE REQUEST HEADERS

BODY ¹ pretty

```
[
  {
    id: 0,
    author: "Mack",
    body: "Post 1"
  },
  {
    id: 1,
    author: "Jack",
    body: "Post 2"
  },
  {
    id: 2,
    author: "Zack",
    body: "Post 3"
  },
  {
    id: 3,
    author: "Mack",
    body: "Here is a new post"
  }
]
```



Talend API Tester Demo



Preprocess Request Body

- Must preprocess req body
 - chunk data
- Would be better to have someone do that for us
 - can use middleware

```
1  var http = require('http');
2
3  var host = '127.0.0.1';
4  var port = 3333;
5
6  http.createServer(preprocess).listen(port, host);
7  console.log('Server running at http://' + host + ':' + port);
8
9  function preprocess(req, res) {
10     var body = '';
11     req.on('data', function(chunk) {
12         body += chunk
13     });
14     req.on('end', function() {
15         req.body = body;
16         server(req, res)
17     })
18 }
```



Middleware

- Middleware is anything you put between the server and the final application/framework
- Middleware is compliant
 - accept a request/response and pass it along
- Can modify the request/response
 - check for authentication
 - add/strip headers
 - format or transform content



Express



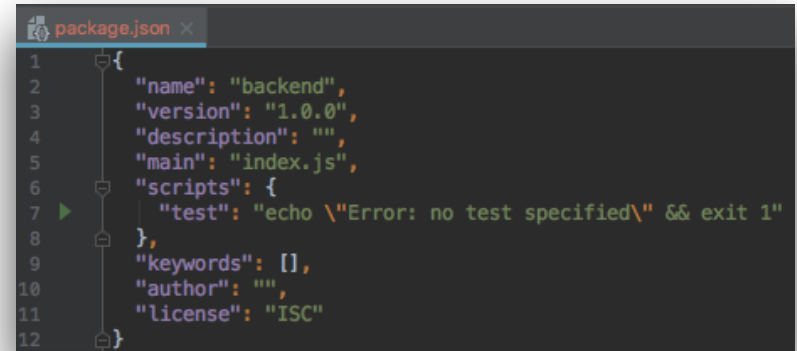
- Express is part of the MEAN or MERN stack
 - [Mongo-Express-Angular-Node](#)
 - [Mongo-Express-React-Node](#)
- Express adds to Node a number of helpful libraries
 - [minimalist approach](#)
 - [middleware](#)

Express
Fast, unopinionated,
minimalist web framework for
[Node.js](#)



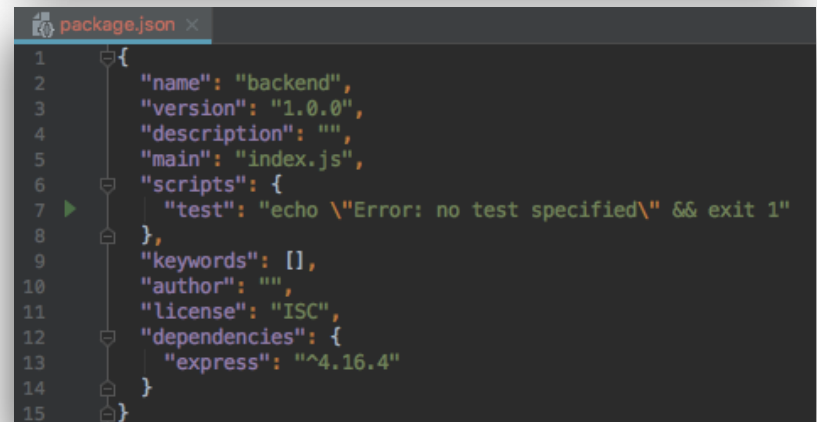
Installing Express

```
>> mkdir backend  
>> cd backend  
>> npm init -y (creates package.json)  
>> npm install express --save
```



A screenshot of a code editor showing the contents of a newly created `package.json` file. The file is named "package.json" and contains the following JSON structure:

```
{  
  "name": "backend",  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.js",  
  "scripts": {  
    "test": "echo \\\"Error: no test specified\\\" && exit 1"  
  },  
  "keywords": [],  
  "author": "",  
  "license": "ISC"  
}
```



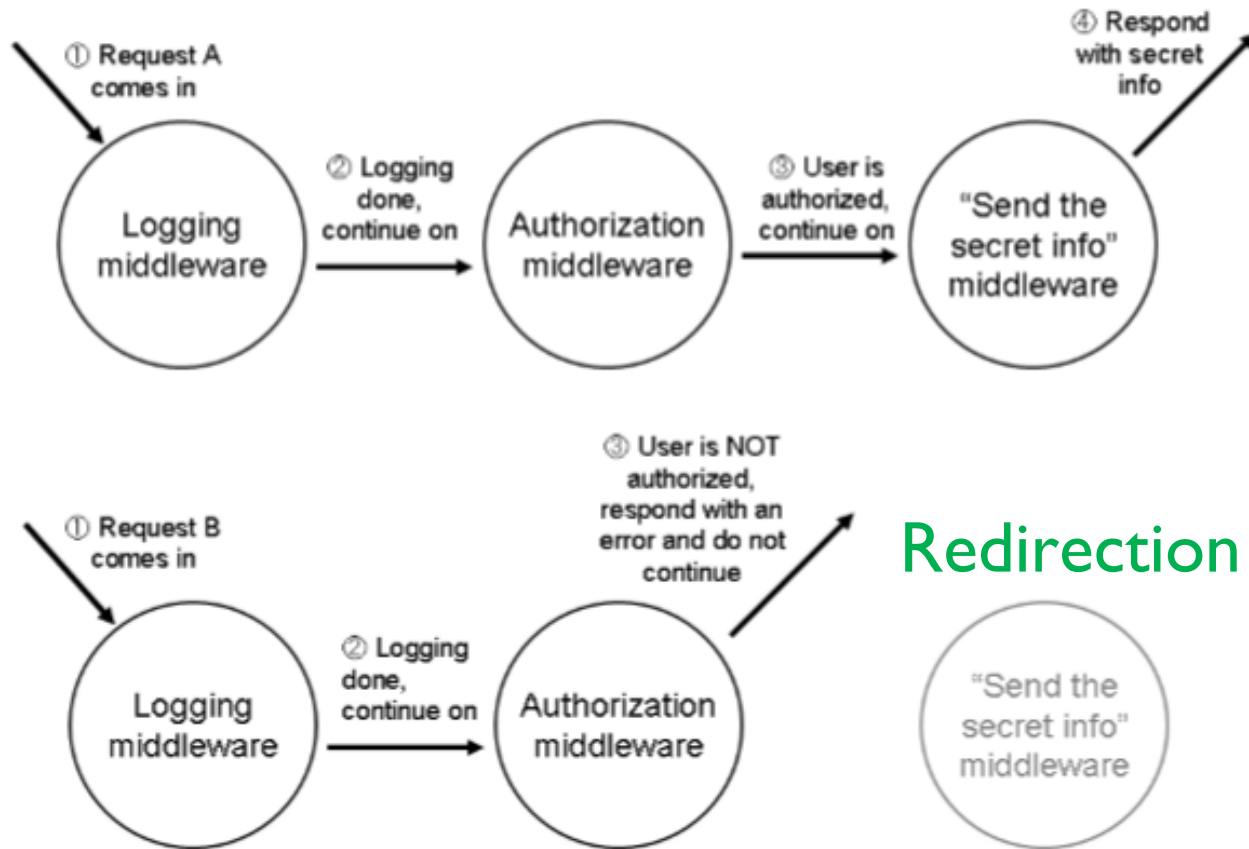
A screenshot of a code editor showing the contents of the `package.json` file after installing Express. The file now includes a `dependencies` section with `express` listed as a dependency.

```
{  
  "name": "backend",  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.js",  
  "scripts": {  
    "test": "echo \\\"Error: no test specified\\\" && exit 1"  
  },  
  "keywords": [],  
  "author": "",  
  "license": "ISC",  
  "dependencies": {  
    "express": "^4.16.4"  
  }  
}
```



Middleware

```
app.put('/logout', isLoggedIn, logout)  
function isLoggedIn(req, res, next)
```



Redirection



Routing with Express

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

var app = express()

app.get('/', getIndex)
app.post('/', postIndex)

function getIndex(req, res) {
  res.send('hello world!')
}

function postIndex(req, res) {
  res.send('You POSTed to the homepage')
}
```

```
const port = process.env.PORT || 3000
const server = app.listen(port, () => {
  const addr = server.address()
  console.log(`Server listening at http://${addr.address}:${addr.port}`)
})
```



Preprocess Request Body

- Must preprocess req body
 - chunk data
- Would be better to have someone do that for us
 - can use middleware

```
1  var http = require('http');
2
3  var host = '127.0.0.1';
4  var port = 3333;
5
6  http.createServer(preprocess).listen(port, host);
7  console.log('Server running at http://' + host + ':' + port);
8
9  function preprocess(req, res) {
10     var body = '';
11     req.on('data', function(chunk) {
12         body += chunk
13     });
14     req.on('end', function() {
15         req.body = body;
16         server(req, res)
17     })
18 }
```



Install Middleware (body-parser)

>> npm install body-parser --save

```
2  const express = require('express')
3
4
5  const addArticle = (req, res) => {
6    console.log('Payload received', req.body)
7    res.send(req.body)
8  }
9
10 const hello = (req, res) => res.send({ hello: 'world' })
11
12 const app = express()
13
14 app.post('/article', addArticle)
15 app.get('/', hello)
16
17 const port = process.env.PORT || 3000
18 const server = app.listen(port, () => {
19   const addr = server.address()
20   console.log(`Server listening at http://${addr.address}:${addr.port}`)
21 })
```

Server listening at http://:::3000
Payload received undefined

mjoyner@FVFG21ZAQ6LT ~ % curl -H 'Content-Type: application/json' -d
'{"text":"This is my new article"}' http://127.0.0.1:3000/article



Accepting JSON Payloads

>> npm install body-parser --save

```
2  const express = require('express')
3  const bodyParser = require('body-parser')
4
5  const addArticle = (req, res) => {
6    console.log('Payload received', req.body)
7    res.send(req.body)
8  }
9
10 const hello = (req, res) => res.send({ hello: 'world' })
11
12 const app = express()
13 app.use(bodyParser.json())
14 app.post('/article', addArticle)
15 app.get('/', hello)
16
17 const port = process.env.PORT || 3000
18 const server = app.listen(port, () => {
19   const addr = server.address()
20   console.log(`Server listening at http://${addr.address}:${addr.port}`)
21 })
```

Server listening at http://:::3000
Payload received { text: 'This is my n

mjoyner@FVFG21ZAQ6LT ~ % curl -H 'Content-Type: application/json' -d
'{"text":"This is my new article"}' http://127.0.0.1:3000/article



Express JSON Payloads

- Node uses `require` to load modules
- Express `use` is applied before every endpoint `subsequently` defined
- No longer need `JSON.parse()` for `req.body`
- Passes request, response objects to function



Request Params

- Implement GET /articles/:id, not
 - /articles/1
 - /articles/2
 - /articles/3
- The `:id` is a parameter whose value is determined at runtime
- Get the value from request (e.g. `req.params.id`)

```
app.get('/article/:id', getArticle);
```



Create a New Post

Attempt to create a post with a unique id:

```
let post = { posts.length: req.body };
```

Error: Can't perform field access for object key

```
let post = {}; post[posts.length] = req.body;
```

Use dictionary approach to access post by id using length

