COMP1531

🍓 Other - Project - Iteration 3

Lecture 8.1

Author(s): Hayden Smith

(Download as PDF)

In This Lecture

- **Why?** 🤔
  - Let's help you out with getting started on iteration 3
- **What?** 📰
  - Review the spec
  - Standups (timeout review)
  - Storing static files
  - Emails

⚔️ Iteration 3

Let's look at the iteration 3 spec together.

🧍 🧍 Standups

The hint we'll give you running standups is:

```
1 const time = 3;
2 console.log(`Standup start for ${time} seconds!`);
3 function finishStandup() {
4   console.log('Standup is now over');
5 }
6 setTimeout(finishStandup, time * 1000);
```

8.1_timeout.ts

📸 Storing Images

There are a few things we want to review for helping you store profile pictures:

1. Serving images from a folder on a server.
2. Collecting & Storing an image.

📸 Storing Images

1. Serving Images From A Folder On A Server

```
1  import express from 'express';
2
3  const app = express();
4  const port = 3001;
5
6  app.use('/static', express.static('static'));
7
8  app.listen(port, () => {
9    console.log(`Listening on port ${port}`);
10 });
```

8.1_img_serve.ts

Learn more about it here.

📸 Storing Images

2. Collecting & Storing An Image

```
1 import request from 'sync-request';
2 import fs from 'fs';
3
4 const res = request(
5   'GET',
6   'http://www.traveller.com.au/content/dam/images/h/1/p/(
7 );
8 const body = res.getBody();
9 fs.writeFileSync('test.jpg', body, { flag: 'w' });
```

8.1_pull_img.ts

Please note: Generally you'll only be able to pull images that are `http` (as opposed to `https`). Sometimes you can just change the URL and see if it works.

📧 Sending Emails

Sending emails is generally considered one of the more challenging parts of the assignment. There a lot of free email services out there. All you need is something that allows you to send emails over SMTP (which you can do via various NodeJS libraries). NodeJS can send requests to an email server to send emails.

🍅 Express With Exceptions

If we just throw an exception on the server that isn't caught, it will result in an actual `500` error...

```
1  /* eslint-disable no-unreachable */
2  import express from 'express';
3
4  const app = express();
5  const port = 3001;
6
7  app.get('/goerror', (req, res) => {
8    throw new Error('Error, oh no!');
9    res.send('Probably won\'t get this message');
10 });
11
12 app.listen(port, () => {
13   console.log(`Listening on port ${port}`);
14 });
```

8.1_express_error_1.ts

Express With Exceptions

If we just throw an exception on the server that isn't caught, it will result in an actual `500` error...

```
1  /* eslint-disable no-unreachable */
2  import express from 'express';
3
4  const app = express();
5  const port = 3001;
6
7  app.get('/goerror', (req, res) => {
8    throw new Error('Error, oh no!');
9    res.send('Probably won\'t get this message');
10 });
11
12 app.listen(port, () => {
13   console.log(`Listening on port ${port}`);
14 });
```

8.1_express_error_1.ts

This is bad because it means we can't distinguish between a genuine error on the server, and our server just saying "bad input"

Express With Exceptions

However we can modify this slightly...

```
1  /* eslint-disable no-unreachable */
2
3  import express from 'express';
4  import HTTPError from 'http-errors';
5  import errorHandler from 'middleware-http-errors';
6
7  const app = express();
8  const port = 3001;
9
10 app.use(express.json());
11
12 app.get('/goerror', (req, res) => {
13   throw HTTPError(400, 'Error, oh no!');
14   res.send('Probably won\'t get this message');
15 });
16
17 app.get('/hello', (req, res) => {
18   res.json({ msg: 'Hello there!' });
19 });
20
21 app.use(errorHandler());
22
23 app.listen(port, () => {
24   console.log(`Listening on port ${port}`);
25 });
```

8.1_express_error_2.ts

Install two libraries and add some middleware.

Express With Exceptions

However we can modify this slightly...

```
1  /* eslint-disable no-unreachable */
2
3  import express from 'express';
4  import HTTPError from 'http-errors';
5  import errorHandler from 'middleware-http-errors';
6
7  const app = express();
8  const port = 3001;
9
10 app.use(express.json());
11
12 app.get('/goerror', (req, res) => {
13   throw HTTPError(400, 'Error, oh no!');
14   res.send('Probably won\'t get this message');
15 });
16
17 app.get('/hello', (req, res) => {
18   res.json({ msg: 'Hello there!' });
19 });
20
21 app.use(errorHandler());
22
23 app.listen(port, () => {
24   console.log(`Listening on port ${port}`);
25 });
```

8.1_express_error_2.ts

Install two libraries and add some middleware.

🍌 Express With Coverage

Now we need to know how to run coverage with `express` and `jest`...

Whilst `jest` has coverage built in, we need an extra dependency to allow a web server to work with jest. `npm install --save-dev nyc`. We also need to update our `jest.config.js` (which we've done for you).

🍌 Express With Coverage

We will also set up some useful scripts for us!

```
"scripts": {
  "ts-node": "ts-node",
  "jest": "jest",
  "ts-node-coverage": "nyc --reporter=text --reporter=lcov ts-node"
}
```

Now we just run:

- `npm run ts-node-coverage theserver.ts` in terminal 1
- `npm run jest thetests.tests.ts` (or just `npm test`) in terminal 2
- CTRL+C in terminal 1 when coverage has finished running
- Open `coverage/lcov-report/index.html` in the web browser.

🍌 Express With Coverage

```
/* eslint-disable no-unreachable */

import express from 'express';
import HTTPError from 'http-errors';
import errorHandler from 'middleware-http-errors';

const app = express();
const port = 3001;

app.use(express.json());

app.get('/goerror', (req, res) => {
  throw HTTPError(400, 'Error, oh no!');
  res.send('Probably won\'t get this message');
});

app.get('/hello', (req, res) => {
  res.json({ msg: 'Hello there!' });
});

app.use(errorHandler());

app.listen(port, () => {
  console.log(`Listening on port ${port}`);
});
```

8.1_express_error_2.ts

```
import request from 'sync-request';

describe('Test Apple', () => {
  test('If it returns a name string successfully', () => {
    const res = request(
      'GET',
      'http://localhost:3001/hello',
      {}
    );
    const bodyObj = JSON.parse(String(res.getBody()));
    expect(bodyObj.msg).toBe('Hello there!');
  });
});
```

8.1_express_error_2.test.ts

🦻 Feedback

Or go to the form here.