Exercise: Using Multiple Router Files

11 min

Generally, we will keep each router in its own file, and require them in the main application. This allows us to keep our code clean and our files short.

To do this with monstersRouter, we would create a new file **monsters.js** and move all code related to /monsters

Preview: Docs Loading link description

[requests]

into it.

```
// monsters.js
const express = require('express');
const monstersRouter = express.Router();

const monsters = {
  '1': {
    name: 'godzilla',
    age: 250000000
  },
  '2': {
    Name: 'manticore',
    age: 21
  }
}

monstersRouter.get('/:id', (req, res, next) => {
  const monster = monsters[req.params.id];
  if (monster) {
    res.send(monster);
  } else {
    res.status(404).send();
  }
});

module.exports = monstersRouter;
```

This code contains all the monsters specific code. In a more full-fledged API, this file would contain multiple routes. To use this router in another file, we use module.exports so that other files can access monstersRouter. The only other new line of code required is that Express must be required in each file, since we'll need to create a router with express.Router().

Our **main.js** file could then be refactored to import the monstersRouter:

```
// main.js
const express = require('express');
const app = express();
const monstersRouter = require('./monsters.js');

app.use('/monsters', monstersRouter);
```

In this example, the monstersRouter is required in **main.js** from **monsters.js** and used exactly as it was before.

Instructions

1. Checkpoint 1 Passed

1.

Let's start to refactor our /expressions routes to **expressions.js**.

Open the **expressions.js** file. Create an expressionsRouter instance of Express.Router.

2. Checkpoint 2 Passed

2.

Export expressionsRouter from that file with module.exports. Remove your old expressionsRouter from **app.js**. require your expressionsRouter from **expressions.js** into **app .js** and make sure it is mounted at /expressions.

3. Checkpoint 3 Passed

3.

Move your GET /expressions handler from **app.js** to **expressions.js**. You'll have to also move the expressions array to that file. Finally, make sure that you move the seedElements(expressions) line into **expressions.js**.

app.js

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

const app = express();

const expressionsRouter = require('./expressions.js');


const { getElementById, getIndexById, updateElement,

  seedElements, createElement } = require('./utils');
```

```
const PORT = process.env.PORT || 4001;
// Use static server to serve the Express Yourself Website
app.use(express.static('public'));
app.use('/expressions', expressionsRouter);

const animals = [];
seedElements(animals, 'animals');

// Get a single expression
app.get('/expressions/:id', (req, res, next) => {
  const foundExpression = getElementById(req.params.id, expressions);
  if (foundExpression) {
    res.send(foundExpression);
  } else {
    res.status(404).send();
  }
});

// Update an expression
app.put('/expressions/:id', (req, res, next) => {
  const expressionIndex = getIndexById(req.params.id, expressions);
  if (expressionIndex !== -1) {
    updateElement(req.params.id, req.query, expressions);
    res.send(expressions[expressionIndex]);
  } else {
    res.status(404).send();
  }
});
```

```
// Create an expression
app.post('/expressions', (req, res, next) => {
  const receivedExpression = createElement('expressions', req.query);
  if (receivedExpression) {
    expressions.push(receivedExpression);
    res.status(201).send(receivedExpression);
  } else {
    res.status(400).send();
  }
});

// Delete an expression
app.delete('/expressions/:id', (req, res, next) => {
  const expressionIndex = getIndexById(req.params.id, expressions);
  if (expressionIndex !== -1) {
    expressions.splice(expressionIndex, 1);
    res.status(204).send();
  } else {
    res.status(404).send();
  }
});

// Get all animals
app.get('/animals', (req, res, next) => {
  res.send(animals);
});

// Get a single animal
app.get('/animals/:id', (req, res, next) => {
  const animal = getElementById(req.params.id, animals);
  if (animal) {
    res.send(animal);
  } else {
    res.status(404).send();
  }
});

// Create an animal
app.post('/animals', (req, res, next) => {
  const receivedAnimal = createElement('animals', req.query);
  if (receivedAnimal) {
    animals.push(receivedAnimal);
    res.status(201).send(receivedAnimal);
  } else {
    res.status(400).send();
  }
});

// Update an animal
app.put('/animals/:id', (req, res, next) => {
  const animalIndex = getIndexById(req.params.id, animals);
  if (animalIndex !== -1) {
    updateElement(req.params.id, req.query, animals);
    res.send(animals[animalIndex]);
  } else {
    res.status(404).send();
  }
});

// Delete a single animal
```

```
app.delete('/animals/:id', (req, res, next) => {
  const animalIndex = getIndexById(req.params.id, animals);
  if (animalIndex !== -1) {
    animals.splice(animalIndex, 1);
    res.status(204).send();
  } else {
    res.status(404).send();
  }
});

app.listen(PORT, () => {
  console.log(`Server is listening on ${PORT}`);
});
```

expressions.js

```
const express = require('express');
const { seedElements } = require('./utils');
const expressionsRouter = express.Router();

const expressions = [];
seedElements(expressions, 'expressions');

// Get all expressions
expressionsRouter.get('/', (req, res, next) => {
  res.send(expressions);
});

module.exports = expressionsRouter;
```