

[◀ Back to Week 1](#)[X Lessons](#)[Prev](#)[Next](#)

Exercise (Instructions): Introduction to Express Part 2

Objectives and Outcomes

In this exercise, you will develop a web server that exports a REST API. You will use the Express framework, and the Express router to implement the server. At the end of this exercise, you will be able to:

- Use application routes in the Express framework to support REST API
- Use the Express Router in Express framework to support REST API

Setting up a REST API

- Create a new file named *server-3.js* and add the following code to it:

```

1  var express = require('express');
2  var morgan = require('morgan');
3  var bodyParser = require('body-parser');
4
5  var hostname = 'localhost';
6  var port = 3000;
7
8  var app = express();
9
10 app.use(morgan('dev'));
11 app.use(bodyParser.json());
12
13 app.all('/dishes', function(req,res,next) {
14     res.writeHead(200, { 'Content-Type': 'text/plain' });
15     next();
16 });
17
18 app.get('/dishes', function(req,res,next){
19     res.end('Will send all the dishes to you!');
20 });
21
22 app.post('/dishes', function(req, res, next){
23     res.end('Will add the dish: ' + req.body.name + ' with details: ' + req
24         .body.description);
25 });
26
27 app.delete('/dishes', function(req, res, next){
28     res.end('Deleting all dishes');
29 });
30
31 app.get('/dishes/:dishId', function(req,res,next){
32     res.end('Will send details of the dish: ' + req.params.dishId + ' to
33         you!');
34 });
35
36 app.put('/dishes/:dishId', function(req, res, next){
37     res.write('Updating the dish: ' + req.params.dishId + '\n');
38     res.end('Will update the dish: ' + req.body.name +
39         ' with details: ' + req.body.description);
40 });
41
42 app.delete('/dishes/:dishId', function(req, res, next){
43     res.end('Deleting dish: ' + req.params.dishId);
44 });
45
46 app.use(express.static(__dirname + '/public'));
47
48 app.listen(port, hostname, function(){
49     console.log('Server running at http://${hostname}:${port}/');
50 });

```

- Install body-parser by typing the following at the command prompt:

```
1      npm install body-parser --save
```

- Start the server and interact with it from the browser/postman.

Using Express Router

- Create a new file named *server-4.js* and add the following code to it:

```
1 var express = require('express');
2 var morgan = require('morgan');
3 var bodyParser = require('body-parser');
4
5 var hostname = 'localhost';
6 var port = 3000;
7
8 var app = express();
9
10 app.use(morgan('dev'));
11
12 var dishRouter = express.Router();
13
14 dishRouter.use(bodyParser.json());
15
16 dishRouter.route('/')
17 .all(function(req,res,next) {
18     res.writeHead(200, { 'Content-Type': 'text/plain' });
19     next();
20 })
21
22 .get(function(req,res,next){
23     res.end('Will send all the dishes to you!');
24 })
25
26 .post(function(req, res, next){
27     res.end('Will add the dish: ' + req.body.name + ' with details: ' + req.body
28         .description);
29 })
30
31 .delete(function(req, res, next){
32     res.end('Deleting all dishes');
33 });
34
35 dishRouter.route('/:dishId')
36 .all(function(req,res,next) {
37     res.writeHead(200, { 'Content-Type': 'text/plain' });
38     next();
39 })
40
41 .get(function(req,res,next){
42     res.end('Will send details of the dish: ' + req.params.dishId + ' to
43         you!');
44 })
45
46 .put(function(req, res, next){
47     res.write('Updating the dish: ' + req.params.dishId + '\n');
48     res.end('Will update the dish: ' + req.body.name +
49         ' with details: ' + req.body.description);
50 })
51
52 .delete(function(req, res, next){
53     res.end('Deleting dish: ' + req.params.dishId);
54 });
55
56 app.use('/dishes',dishRouter);
57
58 app.use(express.static(__dirname + '/public'));
59
60 app.listen(port, hostname, function(){
61     console.log(`Server running at http://${hostname}:${port}/`);
62 });
```

- Start the server and interact with it and see the result.

Conclusions

In this exercise, you used the Express framework and Express router to build a server supporting a REST API.

✓ Complete

