Express: A Jump-Start

•••

Naveen Pete

JS Meetup #4 (Saturday, July 23, 2016)

Agenda

- Introduction to Express
- Installing Express
- Middleware
- Serving Static Files
- Routing
- Template Engines
- Express Generator
- Q & A

Introduction to Express

- Node.js web application framework
- Minimal
- Flexible
- Fast
- Extensible

Installing Express

1. Create an application directory
2. Make it your working directory
3. Create package.json
4. Install Express and save it in the dependencies list

```
$ mkdir myapp
$ cd myapp
$ npm init
$ npm install express --save
```

Middleware

A function that has access to:

1. HTTP request object (req)
2. HTTP response object (res)
3. The next middleware function (next)

It can perform the following tasks:

- Execute any code
- Make changes to the request and the response objects
- End the request-response cycle
- Call the next middleware in the stack

To load the middleware function, call **`app.use()`**, specifying the middleware function

Note: The order of loading middleware is important

Serving Static Files

- Static files:
  - Image
  - CSS
  - JS

- Use built-in middleware function:
  - **`express.static()`**

Examples:

```
app.use(express.static(__dirname + ‘/public’));
app.use(‘/static’, express.static(__dirname + ‘/public’));
```

Routing

Definition of application end points and how they respond to client requests

```
app.METHOD(PATH, HANDLER)
```

Where:

- app - an instance of express
- METHOD - an HTTP request method (lowercase)
- PATH - a path on the server
- HANDLER - the function executed when the route is matched

Note: Each route can have one or more handler functions

Routing

Route Methods

- Derived from one of the HTTP methods (GET, POST, PUT, DELETE, etc.)
- Is attached to an instance of the ‘express’ class

```
// GET method route
app.get('/', function (req, res) {
  res.send('GET request to the homepage');
});

// POST method route
app.post('/', function (req, res) {
  res.send('POST request to the homepage');
});
```

Routing

Route Paths

- Define the ‘endpoints’ at which requests can be made
- Can be strings, string patterns, or regular expressions

Endpoint

- A combination of a URI (path) and a HTTP request method (GET, POST, ...)

Route Handlers

- Callback function - one or multiple
- Can be a function, an array of functions, or combinations of both

Routing

Response Methods

- Send a response to the client
- Terminate the request - response cycle
- If not called, the client request will be left hanging

Method	Description
res.send()	Send a response of various types
res.json()	Send a JSON response
res.render()	Render a view template
res.end()	End the response process

Routing

express.Router

- Is used to create modular, mountable route handlers
- Instance is a complete middleware and routing system
- Often called as a “mini-app”

Template Engines

A template engine:

- Enables you to use static template files in your application
- Replaces variables in a template file with actual values at runtime
- Transforms the template into an HTML file sent to the client

Popular template engines: Pug, Mustache, EJS

```
// set the following application setting properties
app.set(‘views’, ‘./views’);
app.set(‘view engine’, ‘pug’);

// render the view file
res.render('index', { title: 'Hey', message: 'Hello there!'});
```

Express Generator

- An application generator tool
- Used to quickly create an Express application skeleton

Setting up

1. Install express-generator
2. Create an express app
3. Install dependencies
4. Run the app

```
$ npm install express-generator -g
$ express myapp
$ cd myapp
$ npm install
$ npm start
```

Q & A

Thank you!