ExpressJS: Middlewares y control de errores

Gabriel Rodríguez Flores

November 16, 2021

ADA ITS
INSTITUTO TECNOLÓGICO SUPERIOR

    - Middleware genérico
  - 404
  - Manejo de errores
  - Morgan + Winston

Contents

1 Teoría

1.1 Middlewares

Un middleware es la unión de *middle* (en el centro) y *ware* (cosas), que en informática es comunmente usado para definir el componente. Así un middleware lo entendemos como un asistente para un fin, algo que está en medio de la peticion y la respuesta.

1.1.1 Uso de un middleware

En express podremos definirlo de la siguiente manera:

```
app.use(function (req, res, next) {
  console.log('LOGGED');
  next();
});
```

1.1.2 Control de errores

Incluir a todos los controladores el `next()` y si llega al final, se envía error.

Los errores 4XX (cliente) se enviará toda la información, no así con los errores 5XX (servidor) que serán enmascarados.

- Controlador

```
function controller(req, res, next) {
  try {
    // TODO
  } catch (err) {
    return next(err);
  }
}
```

- Middleware de errores

```
function errorHandler(err, req, res, next) {
  res.status(500);
  res.send({ error: err });
}
```

- Servidor

```
app.use(controller);
app.use(errorHandler);
```

1.2 Winston

Winston es un sistema de logger que permite definir un formato y particularizar la salida de información, tanto para consola, como para distintos ficheros, permitiendo en estos la escritura cíclica definiendo un número y tamaño controlado.

- Uso
- Uso y organizacion

1.3 Morgan

Morgan es un middleware que intercepta la petición y permite procesarla, con distintos criterios, en un salida, como por ejemplo consola o un logger específico.

- Uso

1.4 Winston + Morgan

- Integración

2 Ejemplos

2.1 Middlewares

2.1.1 Ejemplo control de errores

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

function controller(req, res, next) {
  try {
    throw Error('Hmmmm... there is an error over here ....')
  } catch (err) {
    return next(err);
  }
}
```

```
11
12 function errorHandler(err, req, res, next) {
13   const statusCode = err.statusCode < 500 ? err.statusCode : 500;
14   res.status(statusCode).send({ code: statusCode, message: 'Server
       Error'});
15 }
16
17 app.use(controller);
18 app.use(errorHandler);
19
20 app.listen(3000);
```

2.2 Morgan + Winston

2.2.1 Configuración básica

```
1  const express = require('express');
2  const morgan = require('morgan');
3  const winston = require('winston');
4
5  const logger = winston.createLogger({
6      transports: [
7          new winston.transports.Console({
8              level: 'debug',
9              handleExceptions: true,
10             json: false,
11             colorize: true
12         })
13     ],
14     exitOnError: false
15 });
16
17 logger.stream = {
18   write(message) {
19     logger.info(message);
20   },
21 };
22
23 morgan.format('combined', '(routes) [:method] :status :url');
24 server.use(morgan('combined', {
25   skip(req, res) {
26     return res.statusCode >= 400;
27   },
28   stream: logger.stream,
29 }));
```

3 Ejercicios

- Realizar un logger que imprima a distinto nivel y formato las peticiones con respuesta
  - `2XX` –> info
  - `4XX` –> warn
  - `5XX` –> error
- Realizar un middleware que valide el acceso a una zona restringida para usuarios admin. Para ello, en la petición se enviará en la cabecera el parámetro `password` con el valor `patata`. En caso contrario, o si no define se define el parámetro, no permitirá el acceso.
  - Acceso correcto: Se enviará como respuesta el mensaje `Bienvenid@, disfrute del contenido`, con el código `200 OK`.
  - Acceso incorrecto: Se enviará un objeto de error con el código `401` unauthorized y el mensaje `Acceso restringido, por favor, incluya la palabra secreta en el parámetro 'password'en la cabera de la petición`

4 Entregables

4.1 En clase

- Crear un middleware para el control de errores que devuelva un error 500 con el mensajes `'Server Error'`
- Crear un logger con winston para imprimir a múltiples niveles formateando la salida con colores
  - Formato: [Fecha] nivel: Mensaje
- Crear un servidor que muestre por consola todos los accesos a la API con morgan

4.2 Tarea

- Realizar todos los ejercicios descritos