Middleware

IIC3585 - Express

Funciones de Middleware:

Definición

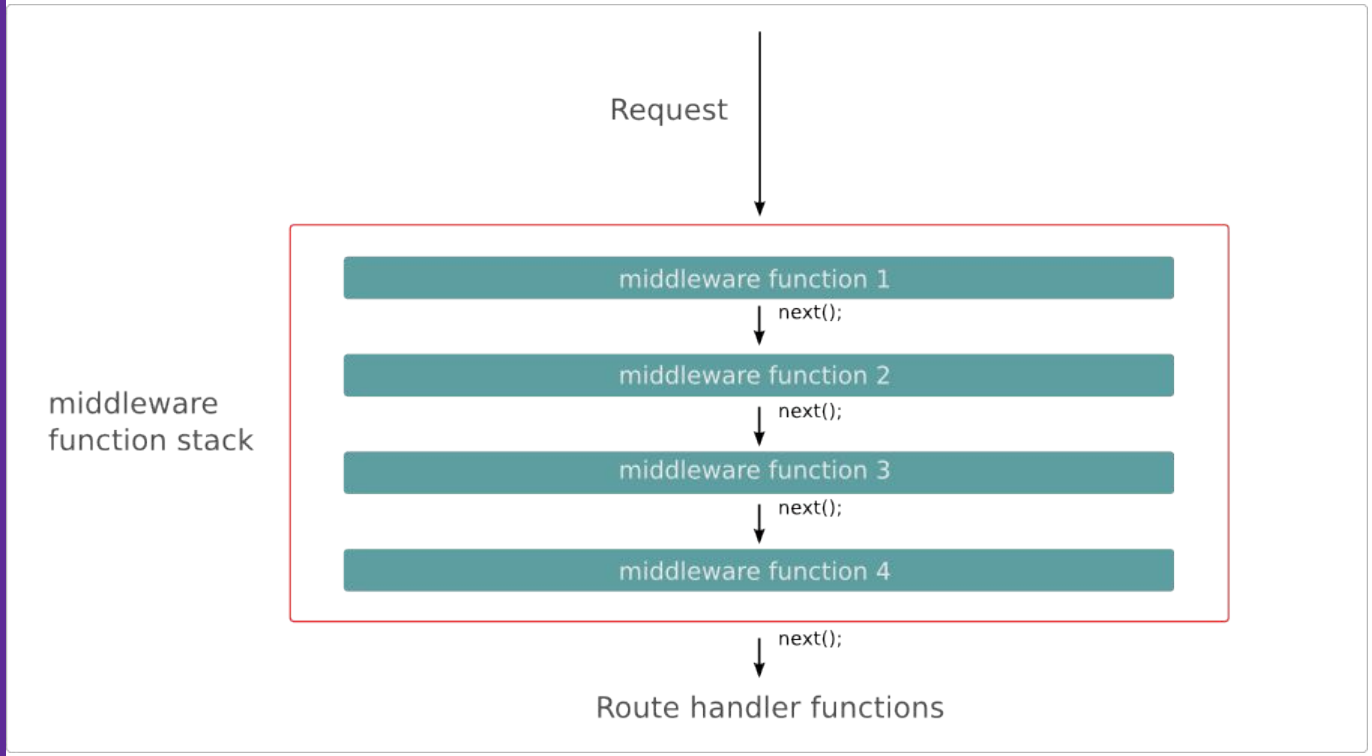
“Las funciones de ***middleware*** son funciones que tienen acceso al objeto de solicitud (req), al objeto de respuesta(res) y a la siguiente función de middleware en el ciclo de solicitud/respuestas de la aplicación”

- Documentación de Express

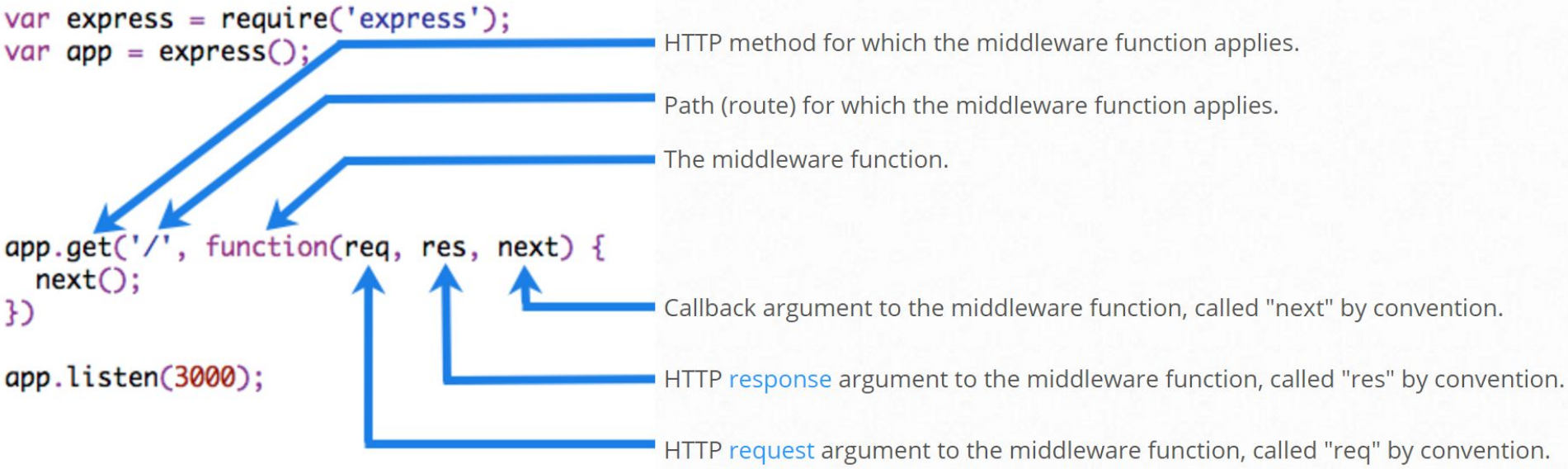
Funciones de Middleware:

1. **Definición**
2. Estructura
3. Usos
4. Tipos
5. MW vs Rutas

Definición



Estructura básica



Funciones de Middleware:

1. Definición

2. Estructura

3. Usos

4. Tipos

5. MW vs Rutas

Estructura - Argumentos

- Request object: objeto que representa una HTTP Request; tiene propiedades asociadas al string de la query.
- Response object: objeto que representa una HTTP Response.
- Next: parámetro que define la función de callback. Se usa para pasar el control a la siguiente función de middleware.

Funciones de Middleware:

1. Definición

2. Estructura

3. Usos

4. Tipos

5. MW vs Rutas

Estructura - Vía de acceso de montaje

```
/*#########################################################################
    Middleware function with no mount path
*/
app.use( (req, res, next) => {
  console.log('Time:', Date.now());
  next();
});

/*  Middleware functions with a mount path.
*/
app.use('/user/:id',
 (req, res, next) => {
  console.log('Request Type:', req.method);
  next();
},
 (req, res, next) => {
  res.send('USER');
  if (req.params.id == 0) {
    app.get('/viewdirectory', require('./mymiddleware.js'));
  }
  else {
    next();
  }
});
```

Funciones de Middleware:

1. Definición
2. **Estructura**
3. Usos
4. Tipos
5. MW vs Rutas

Estructura - Serie de funciones

```
65  /*#####################################################################
66     Series of MW functions
67  */
68  app.use('/series', (req, res, next) => {
69    console.log('Request URL1:', req.originalUrl);
70    next();
71  }, (req, res, next) => {
72    console.log('Request Type1:', req.method);
73    next();
74  });
75
76  /*  Same as above put separated
77  */
78  app.use('/notseries', (req, res, next) => {
79    console.log('Request URL2:', req.originalUrl);
80    next();
81  });
82
83  app.use('/notseries', (req, res, next) => {
84    console.log('Request URL2:', req.originalUrl);
85    next();
86  });
```

Funciones de Middleware:

1. Definición

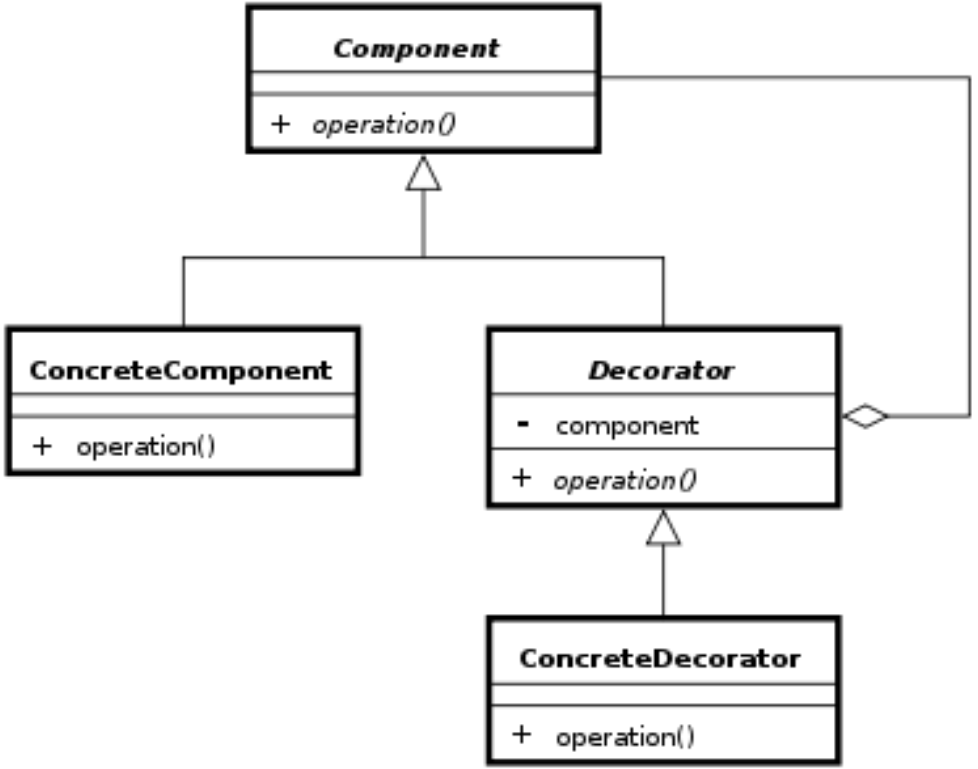
2. Estructura

3. Usos

4. Tipos

5. MW vs Rutas

Estructura - Patrón Decorador



Funciones de Middleware:

Usos

- Ejecutar cualquier código.
- Realizar cambios en la solicitud y los objetos de respuesta.
- Finalizar el ciclo de solicitud/respuestas.
- Invocar la siguiente función de middleware en la pila.

Funciones de Middleware:

1. Definición
2. Estructura
3. Usos
4. **Tipos**
5. MW vs Rutas

Tipos

- Application-level middleware: enlazado a una instancia de objeto de aplicación (funciones app.use() y app.METHOD())
- Router-level middleware: enlazado a una instancia de express.Router()
- Error-handling middleware

Funciones de Middleware:

1. Definición

2. Estructura

3. Usos

4. Tipos

5. MW vs Rutas

MW vs Routing

Direccionamiento hace referencia a la definición de puntos finales de aplicación (URI) y cómo responden a las solicitudes de cliente.

Middleware define un stack de funciones que debe seguirse entre la llegada del Request y el envío de Response.

Manejo de errores:

1. **Argumentos**

2. Definición

3. Demo

Errores - Argumentos

- Tiene 4 argumentos
- Error, Request, Response, Next
- Nuevo argumento: err o Error. Objeto de tipo Error.

```
11
12  app.use(methodOverride());
13  app.use((err, req, res, next) => {
14    // logic
15  });
```

Manejo de errores:

1. Argumentos

2. Definición

3. Demo

Errores - Argumentos

- La función de manejo de error se define debajo de otras funciones de MW o de routing.
- Se pueden definir varias funciones de manejo de errores y pasar el objeto de error con next(err);
- Si se pasa cualquier valor a next(), excepto ‘route’, express lo interpreta como solicitud con error y se saltara a las funciones de manejo de error.

```
11
12  app.use(methodOverride());
13  app.use((err, req, res, next) => {
14    // logic
15  });
```

Manejo de errores:

1. Argumentos

2. Definición

3. Demo

Errores - Argumentos

```
2  //Handles 500 status
3  function clientErrorHandler(err, req, res, next) {
4    if (req.xhr) {
5      res.status(500).send({ error: 'Something failed!' });
6    } else {
7      next(err);
8    }
9  }
10
11 //The “catch-all” errorHandler:
12 function errorHandler(err, req, res, next) {
13   res.status(500);
14   res.render('error', { error: err });
15 }
```

Manejo de errores:

1. Argumentos

2. Definición

3. Demo

Demo

- Manejo de errores sobre los request HTTP

Detalles
del Middleware

1. Orden de las funciones

2. Omisión de next()

Detalles - Orden del middleware

- next() llamado desde un middleware invoca a el proximo middleware o route.
- next() llamado desde una route invoca siempre a la próxima route, saltando middlewares que están entremedio.
- Middleware definido entre routes será ignorado.

Detalles
del Middleware

1. Orden de las funciones

2. Omisión de next()

Detalles - Omisión de next()

- Si no se llama a next() en un método que no termina el ciclo de respuesta a el request, el request queda pendiente e inconcluso

- Lo anterior puede provocar que parezca que la pagina no responde o no funciona.

Third-party Middleware:

1. Hay muchas cosas prehechas

2. Diferentes MW se pueden unir

3. Este modelo da flexibilidad para adaptarse al caso particular

Third-party Middleware

- Body-parser: ayuda a manejar fácilmente la porción del cuerpo de una Request.
- Cookie-session: middleware de manejo de sesiones de cookies simples.
- Cookie-parser: ayuda a parsear el header de las cookies
- Express-session: manejo de sesiones en el servidor
- Helmet: mejora la seguridad de la aplicación con headers HTTP

Para una lista más detallada:
http://expressjs.com/es/resources/middleware.html

Referencias

http://expressjs.com/en/guide/using-middleware.html
http://expressjs.com/en/guide/error-handling.html
http://expressjs.com/en/guide/writing-middleware.html
http://javascript.tutorialhorizon.com/2014/09/19/understanding-expressjs-middleware-with-a-visual-example/
https://www.safaribooksonline.com/blog/2014/03/10/express-js-middleware-demystified/
https://alexperry.io/javascript/2015/08/06/what-is-express-middleware.html
http://www.hacksparrow.com/how-to-write-midddleware-for-connect-express-js.html (2012, algunas cosas estan obsoletas)
http://stackoverflow.com/questions/7337572/what-does-middleware-and-app-use-actually-mean-in-expressjs
http://stackoverflow.com/questions/7337572/what-does-middleware-and-app-use-actually-mean-in-expressjs
http://expressjs.com/es/resources/middleware.html