Express from zero to hero

Nicola Del Gobbo, Developer, Packly S.R.L.

Google Developer Group

#GDG Campobasso

express

open source

LINUX DAY ITALIA

Me

Developer

packly

JS JS C C++

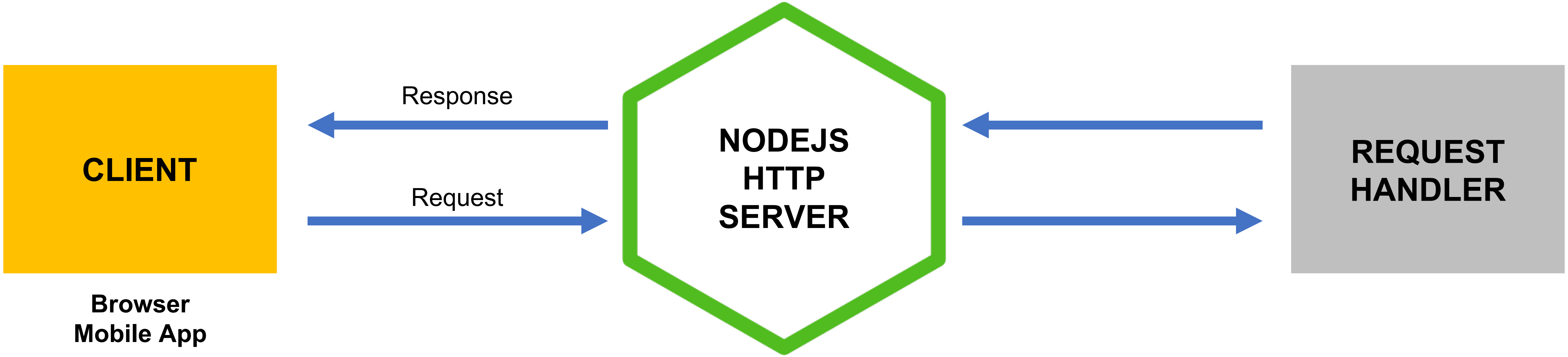
Contribute to

open source

N-API WG

EVANGELISM

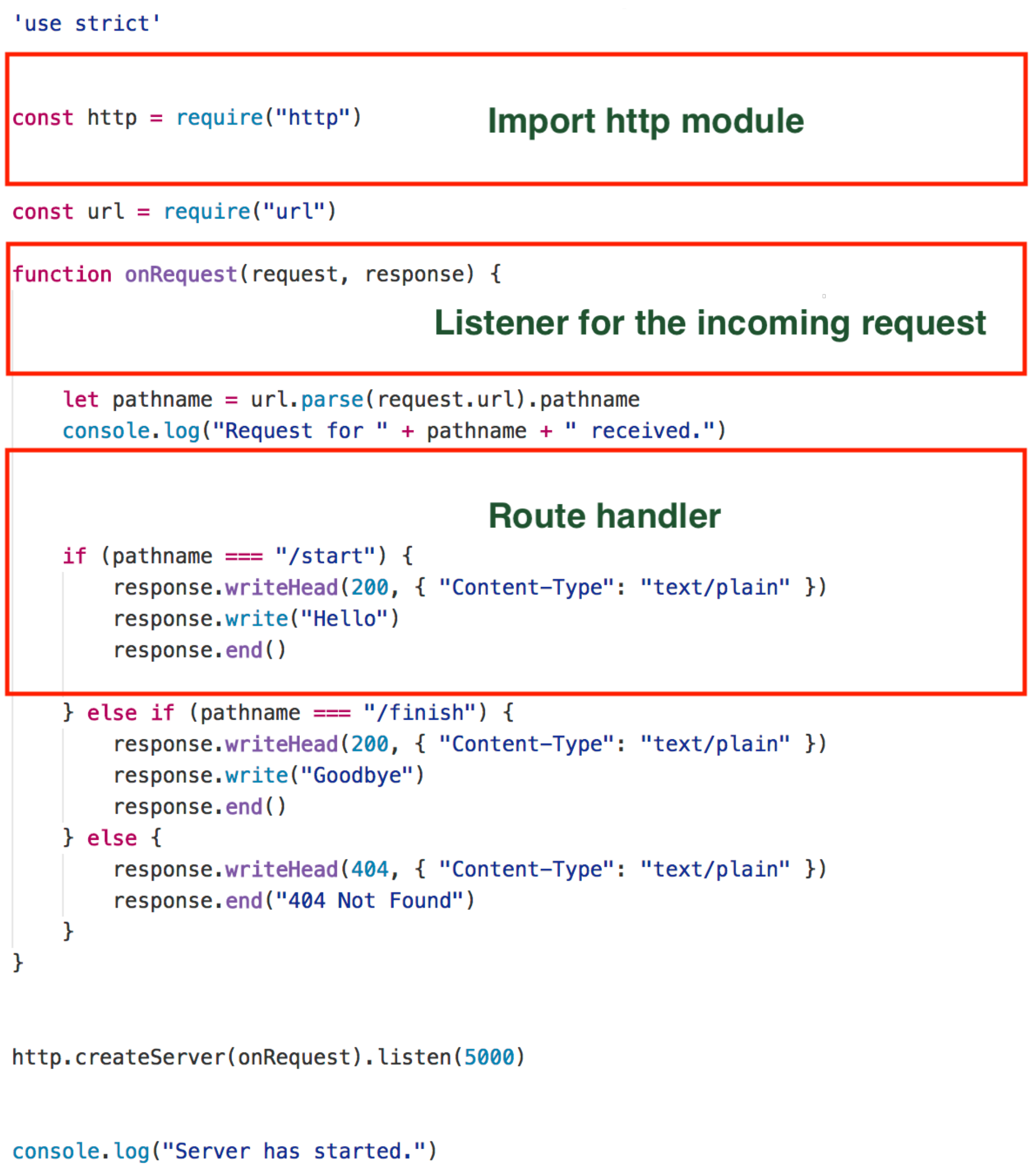
WEB APPLICATION



http://your-web-application

About 40k req/s

Hard to maintain



```
'use strict'

const http = require("http")
```

Import http module

```
const url = require("url")

function onRequest(request, response) {
```

Listener for the incoming request

```
    let pathname = url.parse(request.url).pathname
    console.log("Request for " + pathname + " received.")
```

Route handler

```
    if (pathname === "/start") {
        response.writeHead(200, { "Content-Type": "text/plain" })
        response.write("Hello")
        response.end()
    } else if (pathname === "/finish") {
        response.writeHead(200, { "Content-Type": "text/plain" })
        response.write("Goodbye")
        response.end()
    } else {
        response.writeHead(404, { "Content-Type": "text/plain" })
        response.end("404 Not Found")
    }
}


http.createServer(onRequest).listen(5000)


console.log("Server has started.")
```

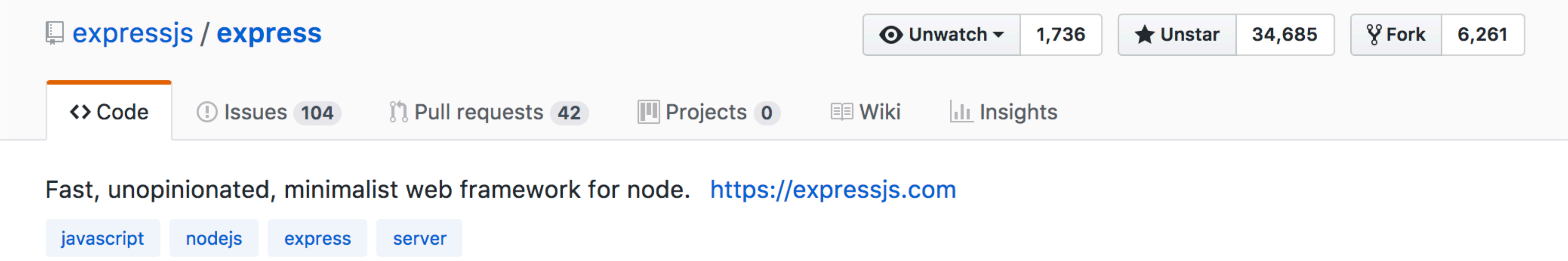
FRAMEWORKS

hapi

fastify

express

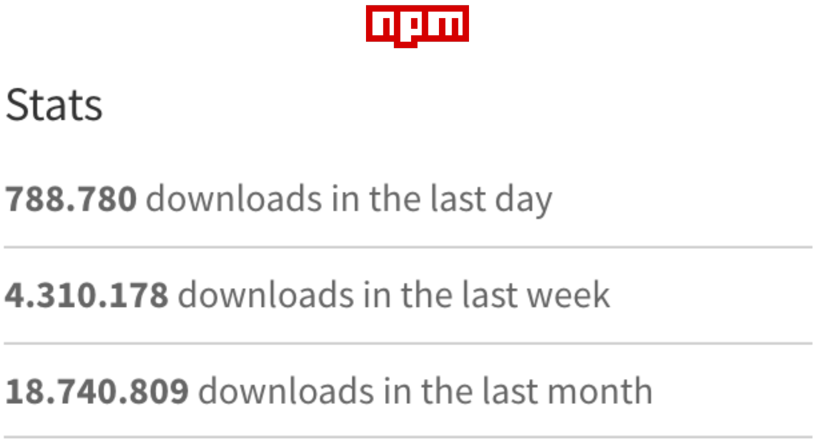
EXPRESS



expressjs / express

Fast, unopinionated, minimalist web framework for node. https://expressjs.com

javascript nodejs express server



Stats

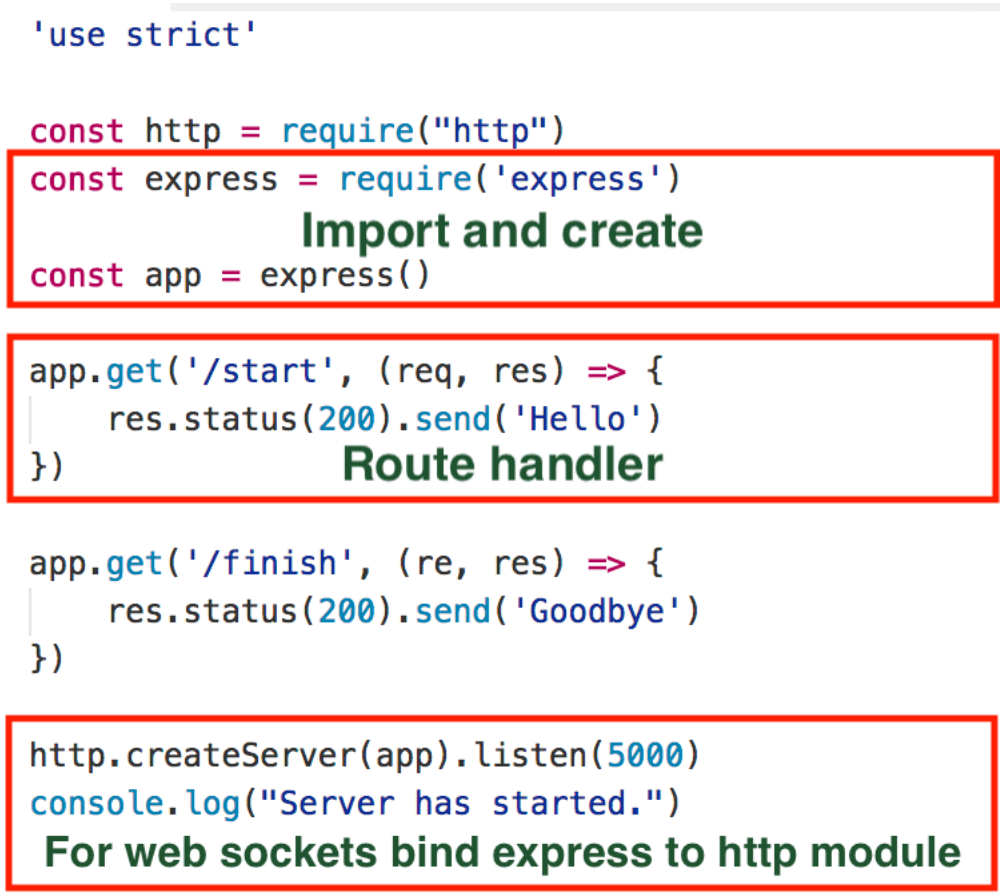
788.780 downloads in the last day

4.310.178 downloads in the last week

18.740.809 downloads in the last month

EXPRESS

- Minimalist
- Unopinioneted
- Fast (about **21k** req/sec)
- Simple (**do one thing well** philosophy from Unix world)
- Wrapper of **http** core module



```
'use strict'

const http = require("http")
const express = require('express')

const app = express()
```

Import and create

```
app.get('/start', (req, res) => {
    res.status(200).send('Hello')
})
```

Route handler

```
app.get('/finish', (re, res) => {
    res.status(200).send('Goodbye')
})

http.createServer(app).listen(5000)
console.log("Server has started.")
```

For web sockets bind express to http module

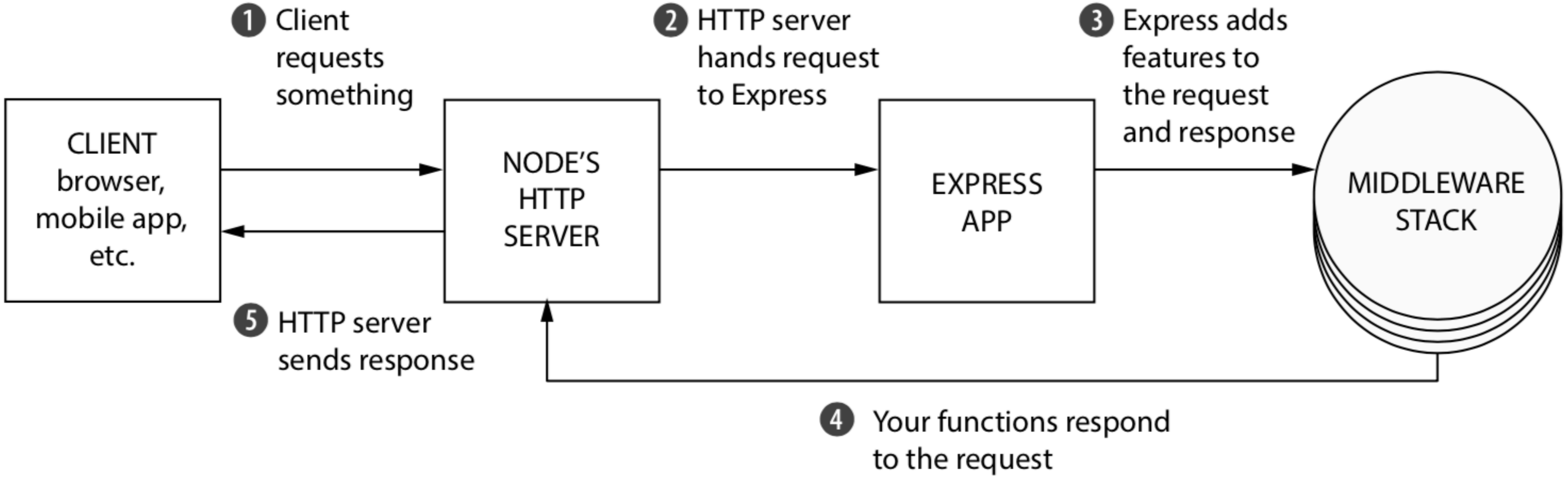
EXPRESS

- Middleware
- Error handler
- Router
- Views / template engine

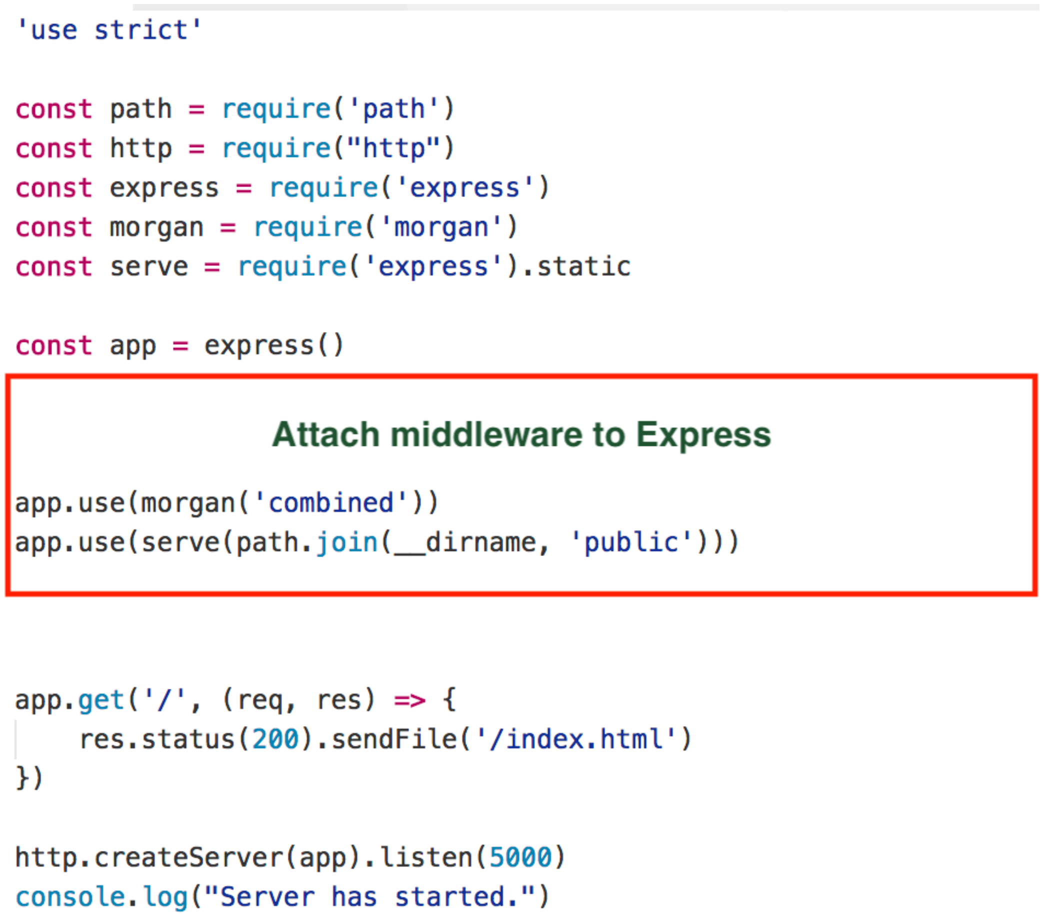
MIDDLEWARES

It’s always a question to manipulate the **Request** and **Response** object

MIDDLEWARES


1 Client requests something
2 HTTP server hands request to Express
3 Express adds features to the request and response
CLIENT browser, mobile app, etc.
NODE'S HTTP SERVER
EXPRESS APP
MIDDLEWARE STACK
5 HTTP server sends response
4 Your functions respond to the request

MIDDLEWARES

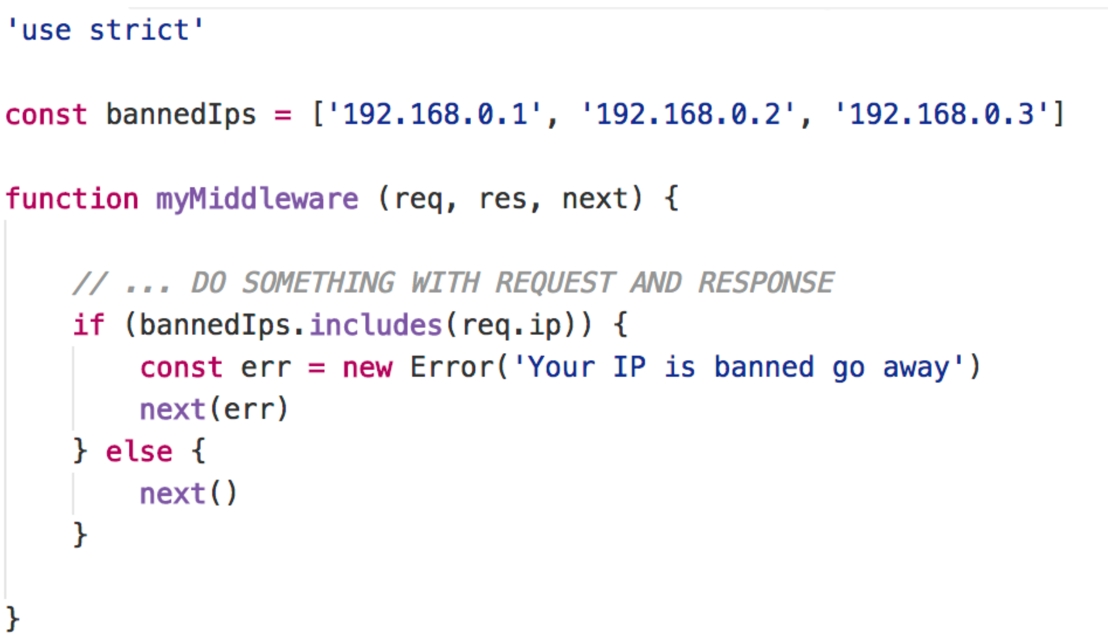


```
'use strict'

const path = require('path')
const http = require("http")
const express = require('express')
const morgan = require('morgan')
const serve = require('express').static

const app = express()
```

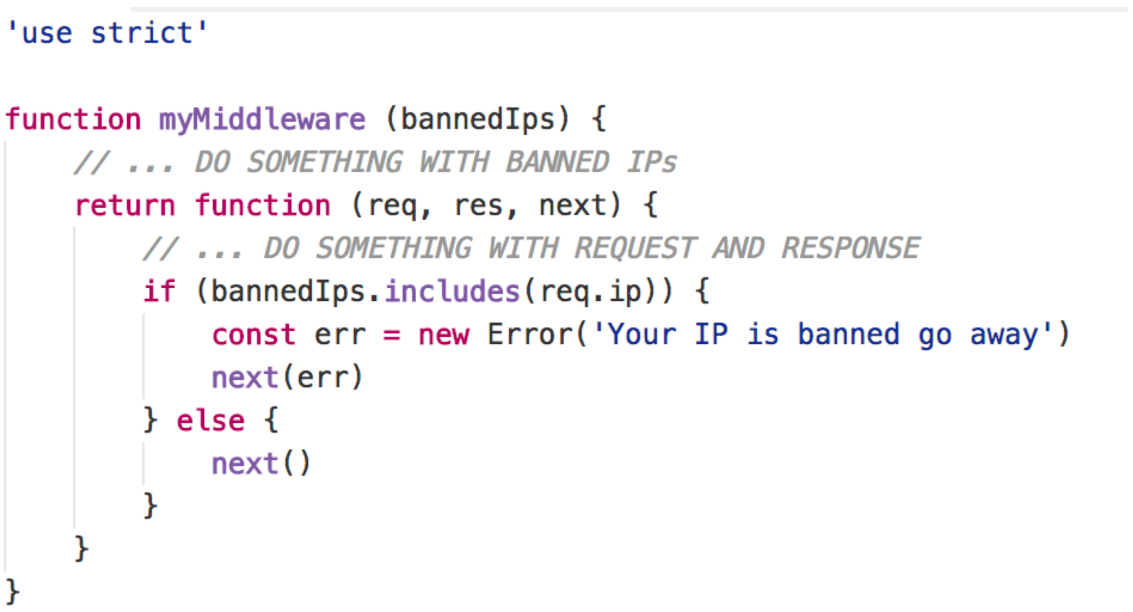
Attach middleware to Express

```
app.use(morgan('combined'))
app.use(serve(path.join(__dirname, 'public')))


app.get('/', (req, res) => {
    res.status(200).sendFile('/index.html')
})

http.createServer(app).listen(5000)
console.log("Server has started.")
```

MIDDLEWARES

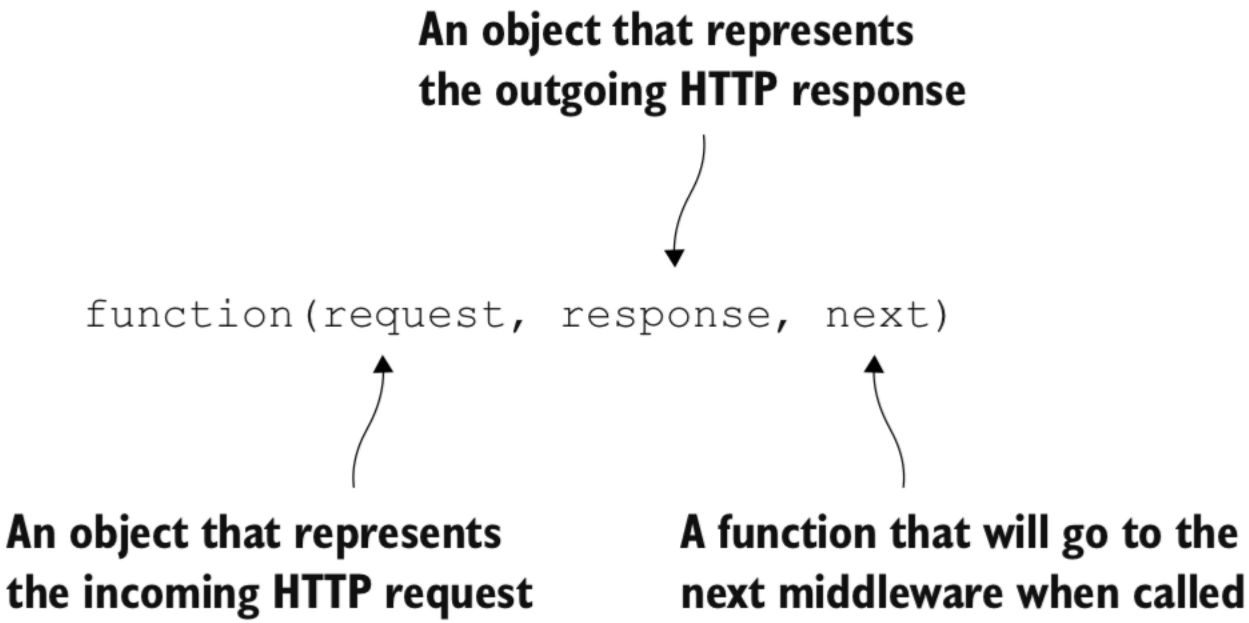


```
'use strict'

const bannedIps = ['192.168.0.1', '192.168.0.2', '192.168.0.3']

function myMiddleware (req, res, next) {

    // ... DO SOMETHING WITH REQUEST AND RESPONSE
    if (bannedIps.includes(req.ip)) {
        const err = new Error('Your IP is banned go away')
        next(err)
    } else {
        next()
    }

}
```



```
'use strict'

function myMiddleware (bannedIps) {
    // ... DO SOMETHING WITH BANNED IPs
    return function (req, res, next) {
        // ... DO SOMETHING WITH REQUEST AND RESPONSE
        if (bannedIps.includes(req.ip)) {
            const err = new Error('Your IP is banned go away')
            next(err)
        } else {
            next()
        }
    }
}
```

MIDDLEWARES



An object that represents the outgoing HTTP response

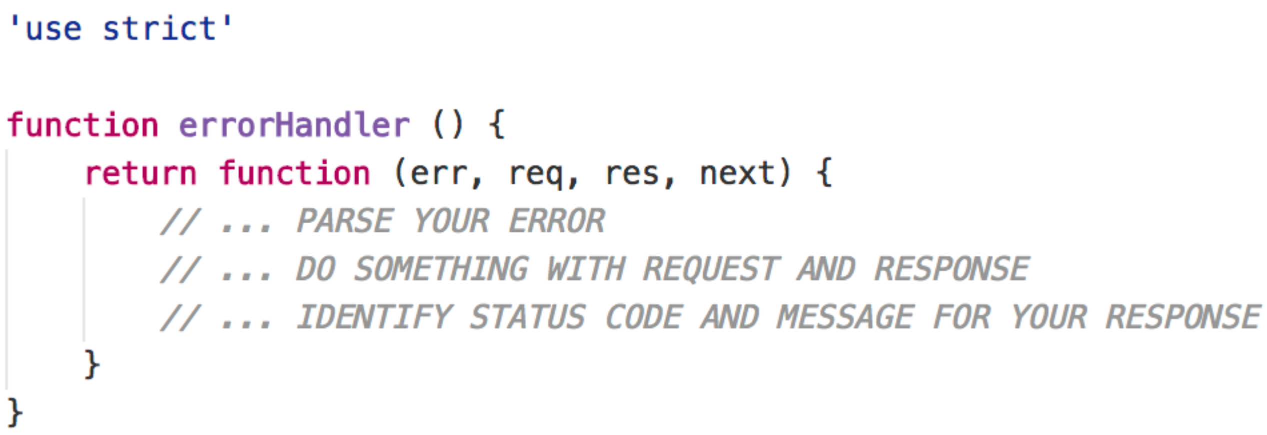
```
function(request, response, next)
```

An object that represents the incoming HTTP request

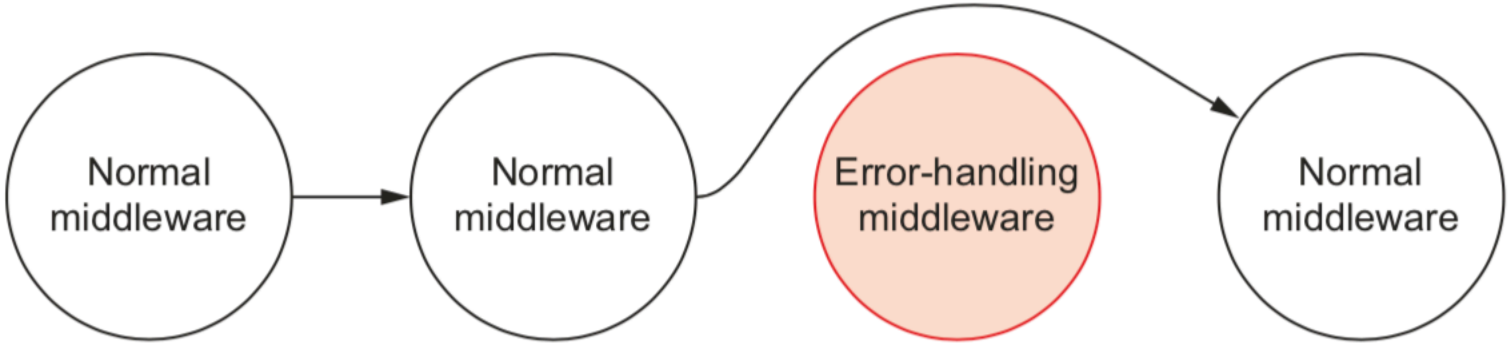
A function that will go to the next middleware when called

http://expressjs.com/en/resources/middleware.html

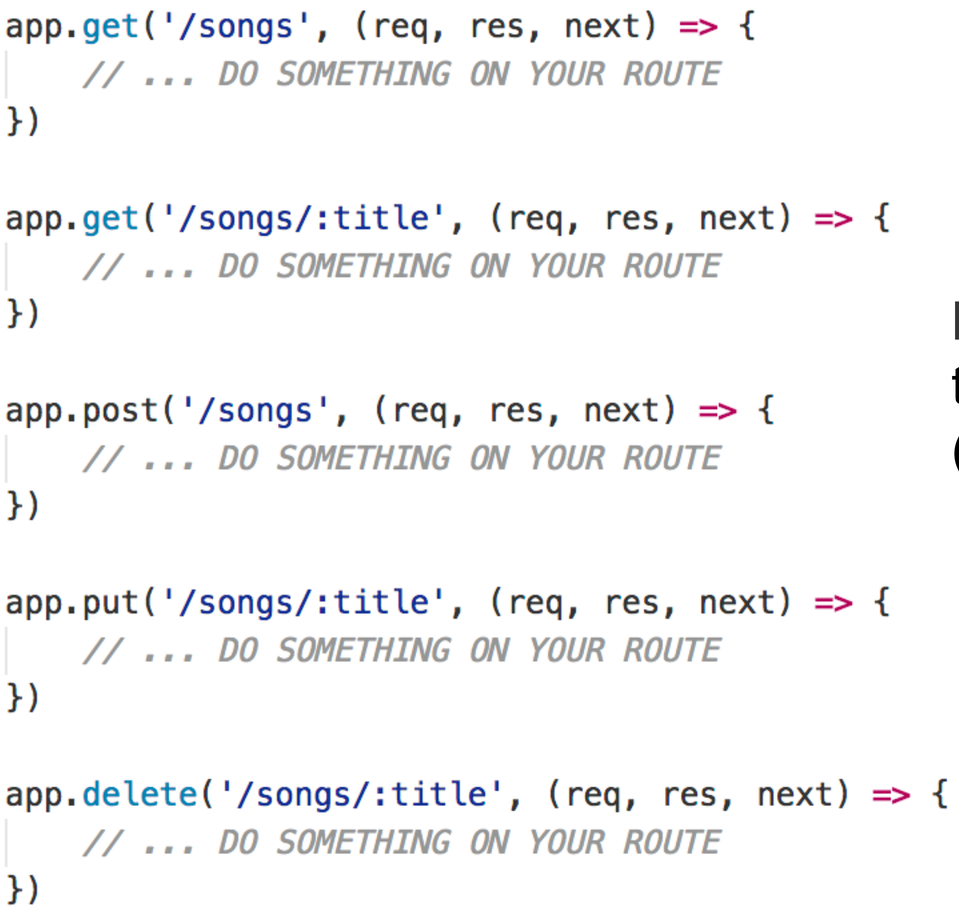
ERROR HANDLER



```
'use strict'

function errorHandler () {
    return function (err, req, res, next) {
        // ... PARSE YOUR ERROR
        // ... DO SOMETHING WITH REQUEST AND RESPONSE
        // ... IDENTIFY STATUS CODE AND MESSAGE FOR YOUR RESPONSE
    }
}
```



ROUTING

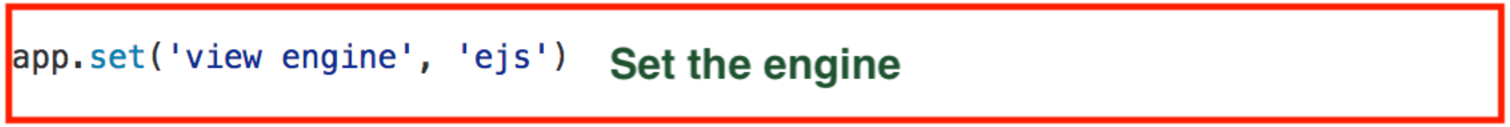


```
app.get('/songs', (req, res, next) => {
    // ... DO SOMETHING ON YOUR ROUTE
})

app.get('/songs/:title', (req, res, next) => {
    // ... DO SOMETHING ON YOUR ROUTE
})

app.post('/songs', (req, res, next) => {
    // ... DO SOMETHING ON YOUR ROUTE
})

app.put('/songs/:title', (req, res, next) => {
    // ... DO SOMETHING ON YOUR ROUTE
})

app.delete('/songs/:title', (req, res, next) => {
    // ... DO SOMETHING ON YOUR ROUTE
})
```

Routing refers to determining how an application responds to a client request to a particular endpoint, which is a URI (or path) and a specific HTTP request method

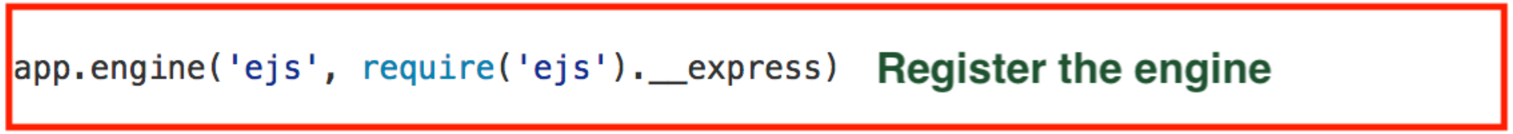
VIEWS / TEMPLATE ENGINE

Pug - Mustache - Dust - Nunjuks - EJS



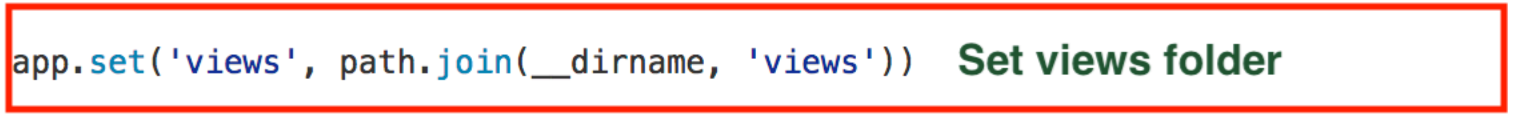
```
app.set('view engine', 'ejs')
```

Set the engine



```
app.engine('ejs', require('ejs').__express)
```

Register the engine



```
app.set('views', path.join(__dirname, 'views'))
```

Set views folder

SECURITY

Helmet helps you secure your Express apps by setting various HTTP headers

npm install helmet

https://github.com/helmetjs/helmet

VALIDATE YOUR INPUT

Ajv

The fastest JSON Schema validator for Node.js and browser

npm install ajv

https://github.com/epoberezkin/ajv

express

EXAMPLES

DOMANDE?

GRAZIE

https://github.com/NickNaso/expressjs-from-zero-to-hero

Nicola Del Gobbo @NickNaso