EXPRESS.JS 201

TRAJECTORY

- Handling URL params
- **app.use** and **next**
- Serving static resources

URL PARAMETERS: **REQ.PARAMS**

- Gives us the ability to have ***wildcard* parameters** in our URLS
- Accessed as a property on the request (req) object
- The name(s) of the route parameter(s) is specified in the path as their respective keys

REQ.PARAMS

GET /USERS/1

REQ.PARAMS

```
// GET /users/:id
app.get('/users/:id', (req, res, next) => {
  // `req.params` is an object
  // This params object contains any URL parameters we state in this route's path (e.g. `id`)
  const userId = req.params.id // Access its value just as any object property/key
  // We can then find the user associated with that `id` and send that user as the response
})
```

APP.USE AND NEXT

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

const app = express()

app.get('/', (req, res, next) => {
  res.send(`<h1>Welcome to the Homepage</h1>`)
})

app.get('/users', (req, res, next) => {
  res.send(`<h1>All of our users!</h1>`)
})

app.listen(3000, () => {
  console.log('Listening on http://localhost:3000/')
})
```

YOUR EXPRESS APP

```
app.get('/')
```

```
app.get('/users')
```

GET /users HTTP /1.1

"Are you a handler for GET /users"?

HTTP/1.1 200 OK
Date: Wed, 03 Jan 2018 17:29:52 GMT
Connection: keep-alive
Transfer-Encoding: chunked

<h1>All of our users!</h1>

```
app.get('*', (req, res, next) => {
  console.log(req.method, req.url)
})

app.get('/', (req, res, next) => {
  res.send(`<h1>Welcome to the Homepage</h1>`)
})

app.get('/users', (req, res, next) => {
  res.send(`<h1>All of our users!</h1>`)
})
```

YOUR EXPRESS APP

```
app.get('*')
```

```
app.get('/')
```

```
app.get('/users')
```

GET /users HTTP /1.1

"Are you a handler for GET /users"?

"I'm a handler for everything! I'll log something to the console and…"

"……"

```
app.get('*', (req, res, next) => {
  console.log(req.method, req.url)
  next()
})

app.get('/', (req, res, next) => {
  res.send(`<h1>Welcome to the Homepage</h1>`)
})

app.get('/users', (req, res, next) => {
  res.send(`<h1>All of our users!</h1>`)
})
```

YOUR EXPRESS APP

```
app.get('*')
```

```
app.get('/')
```

```
app.get('/users')
```

GET /users HTTP /1.1

"Are you a handler for GET /users?"

"I'm a handler for everything! I'll log something to the console and…"

…now I'm ready for you to go the next matching route

HTTP/1.1 200 OK
Date: Wed, 03 Jan 2018 17:29:52 GMT
Connection: keep-alive
Transfer-Encoding: chunked

<h1>All of our users!</h1>

```
app.use '*', (req, res, next) => {
  console.log(req.method, req.url)
  next()
})

app.get('/', (req, res, next) => {
  res.send(`<h1>Welcome to the Homepage</h1>`)
})

app.get('/users', (req, res, next) => {
  res.send(`<h1>All of our users!</h1>`)
})
```

APP.GET

- Only for **GET** requests
- Exact match of verb and URL

APP.USE

- Handles all verbs
- Fuzzy match of URL (we'll explore this more later)

EXPRESS MIDDLEWARE

A function that handles responding to requests.

Of the form: **function(req, res, next){…}**

EXPRESS MIDDLEWARE

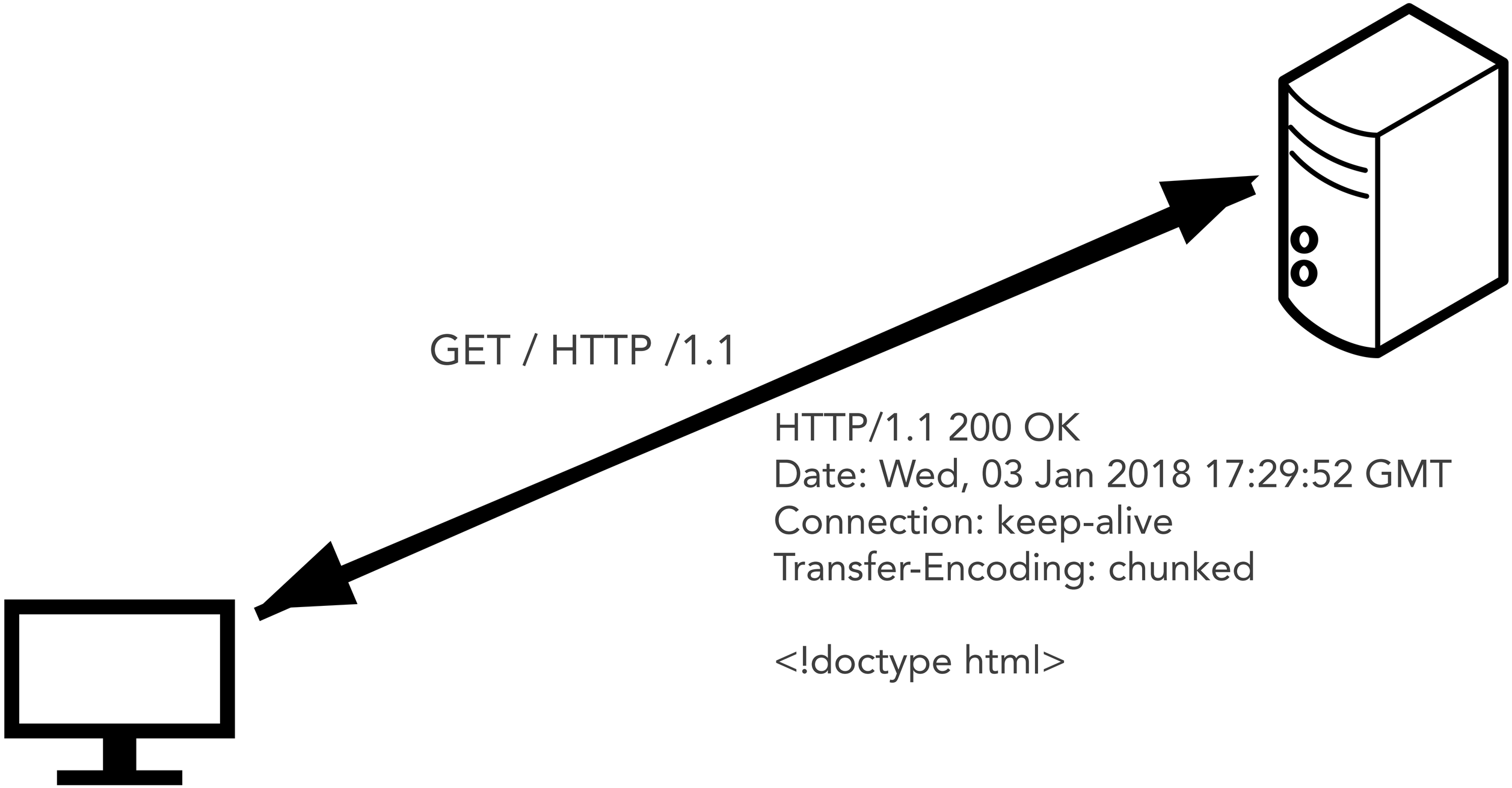
```
app.use((req, res, next) => {
  console.log(req.method, req.url)
  next()
})
```

MORGAN
LOGGING MIDDLEWARE MADE EASY

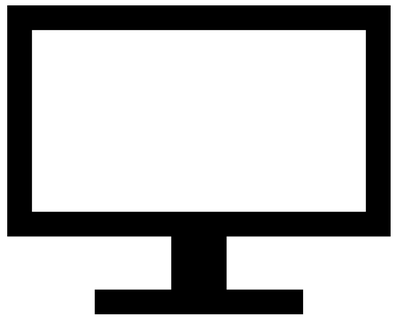
```
const morgan = require('morgan')

app.use(morgan('dev'))
// 'dev' - Concise output colored by response status for development
use. The :status token will be colored green for success codes, red
for server error codes, yellow for client error codes, cyan for
redirection codes, and uncolored for information codes.
```

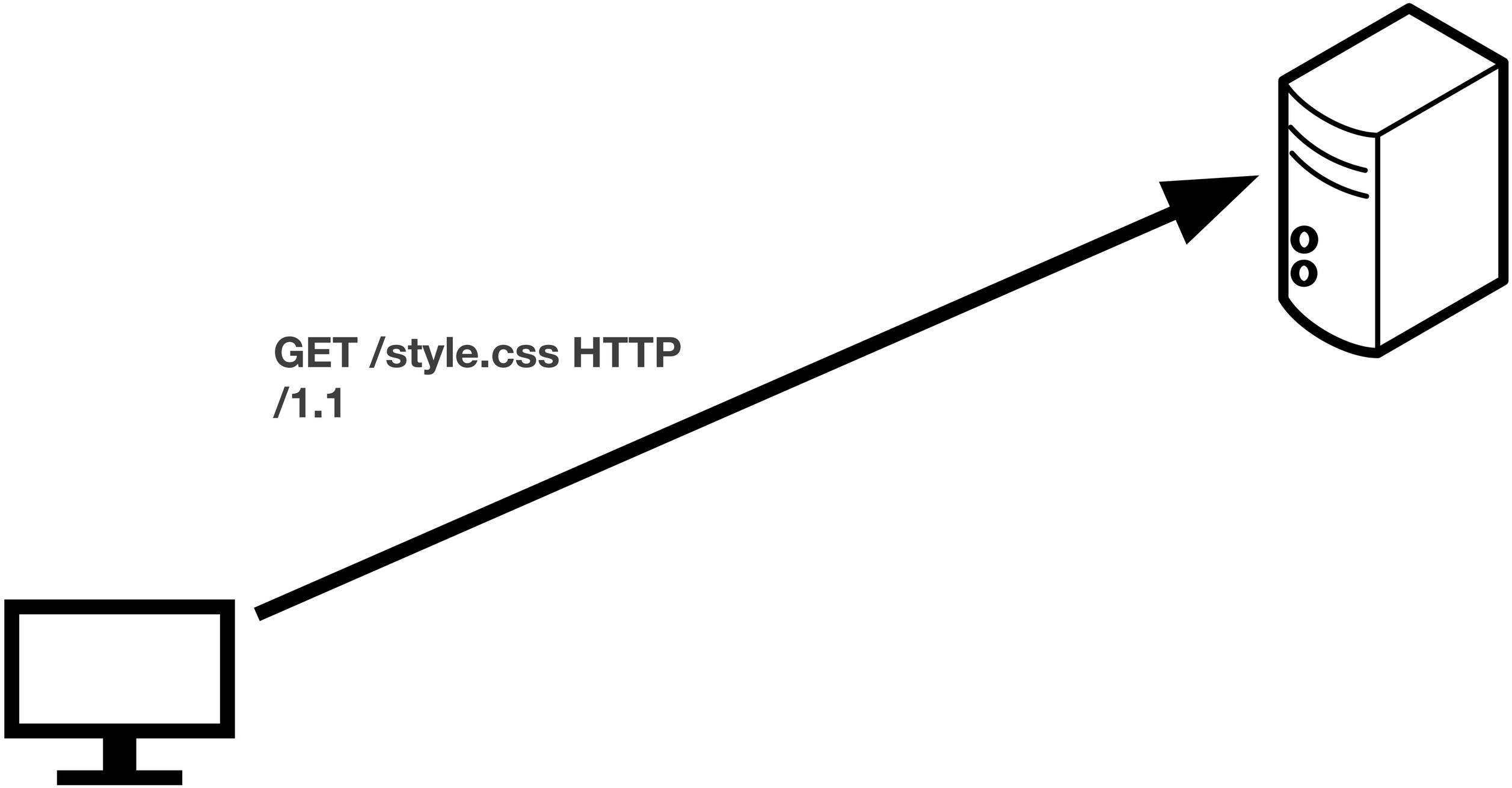
EXPRESS.STATIC

```
/server
  app.js
/public
  styleA.css
  styleB.css
  clientSideScript.js
```


GET / HTTP /1.1
HTTP/1.1 200 OK
Date: Wed, 03 Jan 2018 17:29:52 GMT
Connection: keep-alive
Transfer-Encoding: chunked
<!doctype html>

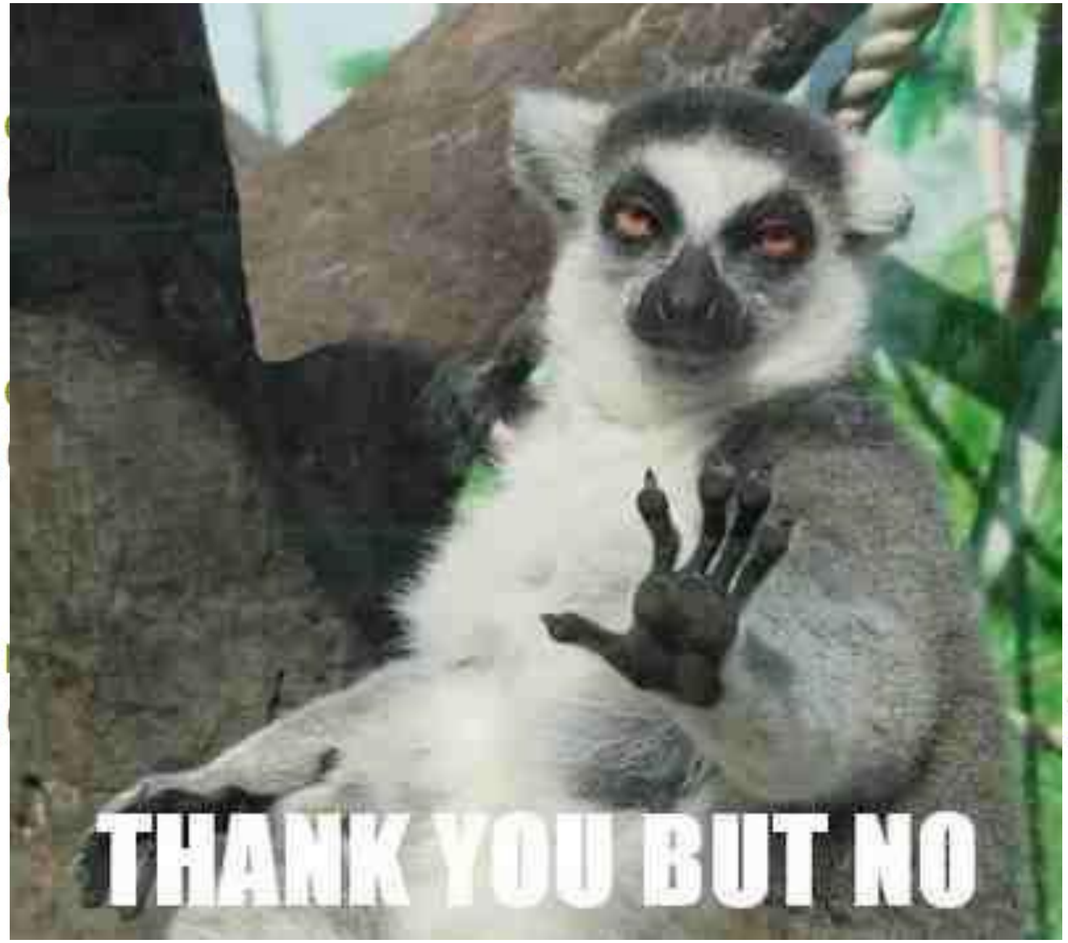
```
<!doctype html>
  <head>
    <link rel="stylesheet" href="/style.css" />
  </head>
  <body>
  </body>
</html>
```




GET /style.css HTTP
/1.1

```
app.get('/style.css', (req, res, next) ⇒ {
  res.sendFile(path.join(__dirname, 'public', 'style.css'))
})
```

```
app.get('/style
  res.sendFile(                                    tyleA.css'))
})

app.get('/style
  res.sendFile(                                    tyleB.css'))
})

app.get('/clie                                      ⇒ {
  res.sendFile(                                    lientSideScript.js'))
})
```



```
const staticMiddleware = express.static(path.join(__dirname, 'public'))
app.use(staticMiddleware)
```