Express Basics

https://expressjs.com/

- can define option for static files
  - `app.use(express.static(documentRoot))`
- Route == path (mostly)
- Define callbacks for method + routes
  - `app.get(path, callback);`
  - `app.post(path, callback);`
- Callback is passed `req` and `res`
- Read from `req`, write to `res`
- Remember the parts of request/response

Express Middleware

- Middleware can handle or alter request
    - in the "middle" of incoming and next step
- Middleware options:
    - On all requests with `app.use()`
    - On one route as 2nd param in route definition

```
app.get( '/cats', express.urlencoded(), (req, res) => {
```

More on the Request

https://expressjs.com/en/4x/api.html#req

- Query params are in `req.query` (parsed to object)
    - everything is strings (because url)
- Body params require **middleware**
    - Need to know how to parse
    - Parsed result is `req.body`

Request is already sent, read-only

Reading a URL-encoded POST body

- Use the `express.urlencoded()` middleware
- Body fields mapped to `req.body` object

Will complain unless you pass `{ extended: false }`

Dynamic Routes

What if you want data in path itself?

- e.g. `/students/Mengchin` vs `/students/Preetha`?

Define route with `:variableName`

- mapped to object `req.params`
- e.g. route: `app.get('/students/:name', ...)`
- e.g. url: `/students/Preetha`
- e.g. `req.params.name === 'Preetha'`

https://expressjs.com/en/guide/routing.html#route-parameters

Writing Response

https://expressjs.com/en/4x/api.html#res

- Status, Headers, Body
    - Sends *as you go*
- `.status(code)` send status
    - 200 sent if you skip this step
- `.redirect(url)` sends a 302 w/location
- `.send(content)` sends content
    - Content is just string, even if HTML

Static Assets

```
app.use(express.static('./public'));
```

`express.static(...)` returns middleware func

- Checks requests against static file?
- Pass document root to `express.static(...)`
- `'./public'` is "public" subdirectory
- `/cats/mary.html` matches:
    - `(your project)/public/cats/maru.html`
- If it isn't inside `public/`, it can't be viewed
    - Does not show your `server.js`, etc

Using Express (and beyond)

- Match the route (method + route)
- Build the response as a string
    - use request params/body as needed
    - templating outside this course
- Send the response
    - Status, Headers, Body

Finishing Express

- Routes are checked in order of addition
- `app.listen(...)` is passed port and callback
    - Listens on port
    - Calls callback when ready
    - Doesn't respond until `app.listen(...)`
- Enters loop until killed

Other Webservers

- Express is just one webserver framework
- We are doing the basics
    - More abstractions available
    - dynamic vs static
- All webservers handle
    - method
    - path
    - query
    - headers
    - body
- Response is just strings

Summary

- Use middleware
- Static routes with `express.static(...)`
- Handle routes (method+path)
    - callback passed `req` + `res`
- Read inputs
    - query vs params vs body
- Compose response
    - status, headers, body