

Common confusion on basic-express

The assignments practice with common problems

- Let's review some

The public/ directory

I deliberately made you link to the CSS

- Add link to generated HTML
- Get the path in the `href` correct

Key lesson:

- server-side vs client-side paths
 - DIFFERENT

Server-side paths are easy

Paths where the server-side code loads it

- like in `require()`
- all relative to the server-side code

Client-side paths aren't loaded by server!

If `chat-web.js` generates HTML to load CSS

- this load does not happen IN chat-web.js
 - The HTML is just a string of text
- HTML is sent in response to browser
- browser decides to request the css
- browser can't access server code
- server.js says it will look in `public/`
 - or match a dynamic route

public/ will never appear in your urls

- Not in `href`
- Not in `src`
- Only in server code
 - only when dynamically using static files

Route Matching

Server is always responding to a request for a path

- express looks for a matching route
 - In order
 - Stopping once a matching route doesn't send it on to the next route

When we request /

- Server looks in `public/` for `public/index.html`
- Because `express.static()` route is first

Test it:

- Create a `public/index.html`
- See it instead of dynamic `/` route
- Move `app.use(express.static(...))`
 - to just before `app.listen(...)`
 - after `app.get('/', ...)`
- Restart server
- Dynamic `/` route now shows

HTML not "read" on server

- Load `/` in browser
 - look in `DevTools->Network`
- You see `GET /`
- You see `GET /chat.css` (or whatever)

Server sent HTML response

- No CSS file
- Just a reference to the CSS file
- BROWSER decides to request the CSS file
- Server doesn't even know CSS file request is related

Common Best Practices errors

- Indentation communicates
 - It must be done/not done for a reason!
- Names are important
 - They communicate
 - But only if you use them well
- Separation of Concerns matters
 - Code Quality
 - Makes changes easier!