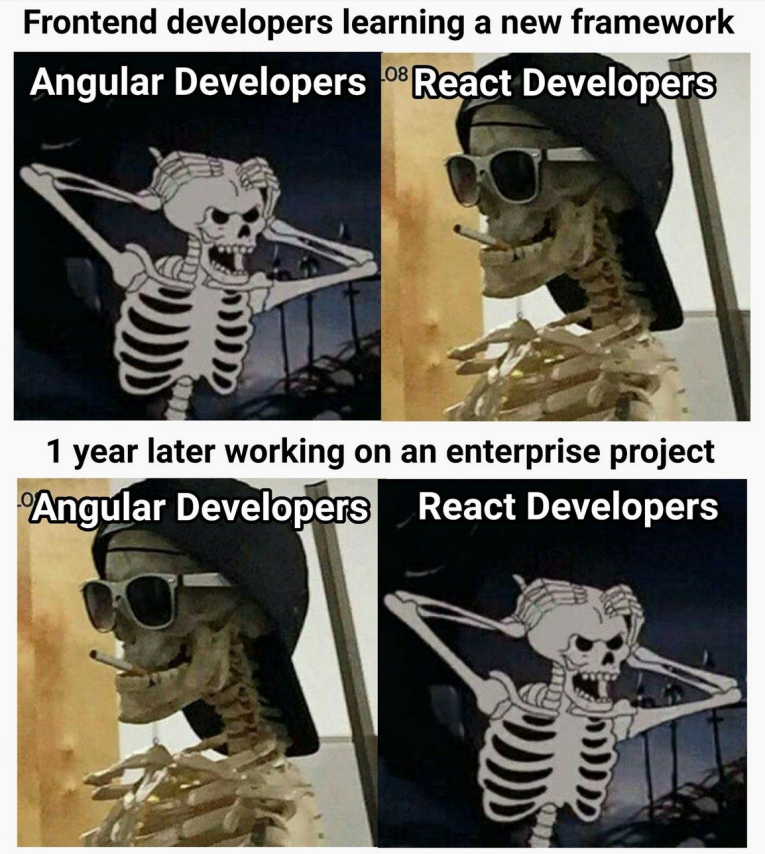
React Router

IMY 220 • Lecture 5

React Router

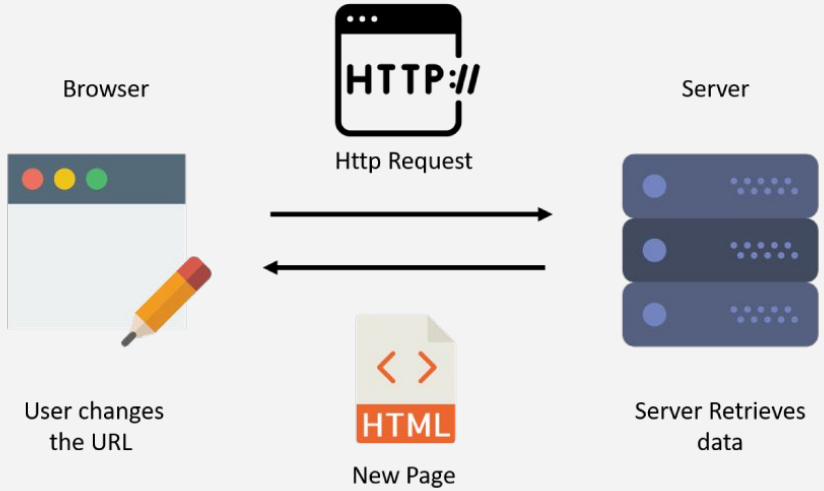
1. How routes and requests work
2. Manually setting up routes in Express and React
3. React Router - What is it?
4. Basic React Router Demo (Old method)
5. Alternate React Router Syntax (Newer method - v6+)
6. Dynamic Routes
7. Next Steps



How Routes and Requests Work

Recap from COS216

1. Client navigates to URL
    a. “https://blahblah.com/**home**”
    b. “.../**home**” is the route
2. Client makes a request to server
    a. “Fetch me the **home**.html page”
3. Server finds the file and sends it back to Client
    a. “Here is **home**.html”
4. Otherwise 404 not found :(

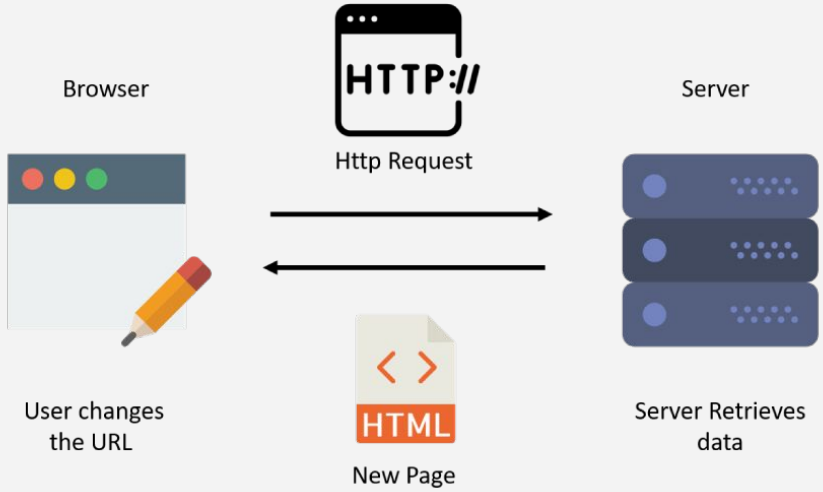


How Routes and Requests Work

This happens for **every single file your website needs** (.html, .css, .js)

> <link **rel="`style.css`"** type="text/stylesheet"/>
>
> <script **src="`index.js`"** type="text/javascript"></script>
>
> These are **both requests to the server for `style.css` and `index.js`** files

Every time your user navigates to a new page by clicking a **link**, **a new set of files** (.html, .css, .js) must be requested, fetched and returned.



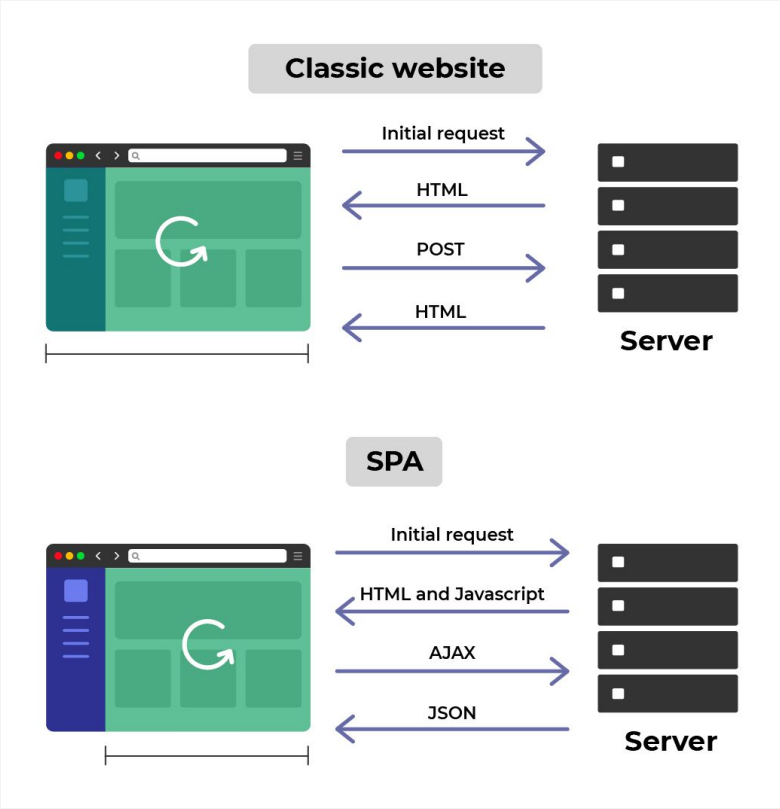
How Routes and Requests Work

Apps with Multiple HTML pages are called **Multi-Page Applications (MPAs)**

- There is a new **HTML file for every page**
- Every time a user clicks on a link (e.g., in the nav), a new HTML page should be fetched.

React Apps are **Single-Page Applications (SPAs)**

- They only use **one HTML file**, and instead swap out components and re-render the DOM
- *So when a user clicks on a link for a "new page"… what happens?*



Manually Setting up Routes in Express and React

Not much support for complex routing out-of-the-box.

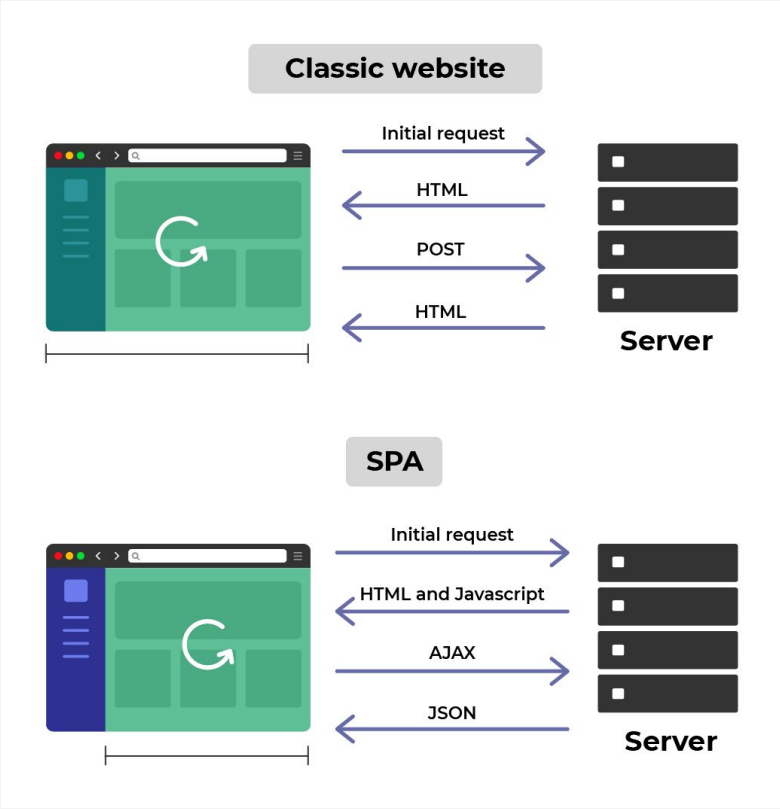
We would need to:

- Specify the page that needs to be sent (index.html) and the **JS components** that need to be rendered to the page, along with any **CSS** needed by those components.

This can naturally get very **complicated**.

It is also **slow**.

- HTML link (<a>) clicked = browser **reloads** the page = all of the React components re-render unnecessarily.
- Constant (unnecessary) requests to server.
- Components **lose their current state**, so pages cannot be **backtracked** (i.e., you can't go back to where you were).



Manually Setting up Routes in Express and React

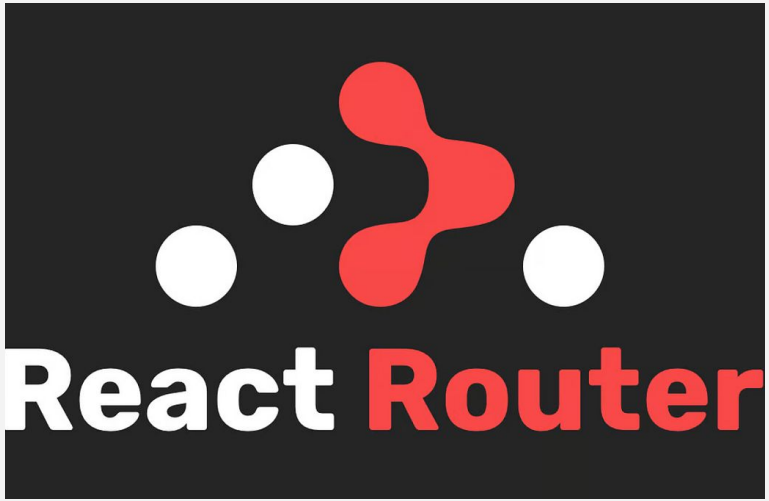
- **Backend**: Trying to set them up in express (backend / server-side) on it’s own requires a lot of extra Packages and Server-side Rendering.
- **Frontend & Backend**: Client-side Rendering & Server calls manually is also very complicated.

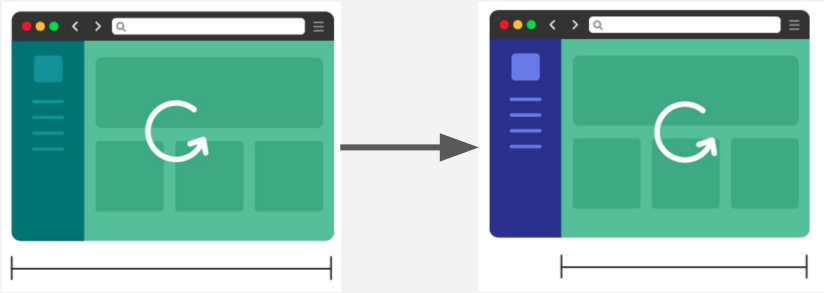
So… What is the Solution?

React Router (react-router-dom)

- A **React Package** that handles the client-side routing for you.

There are other React frameworks that also solve this issue (e.g., *NextJs*). For our purposes we are going to stick to React Router because it's easier to implement.





Why Bother?

Why Bother with Routes in the first place?

Why not keep it single page?

So… What is the Solution?

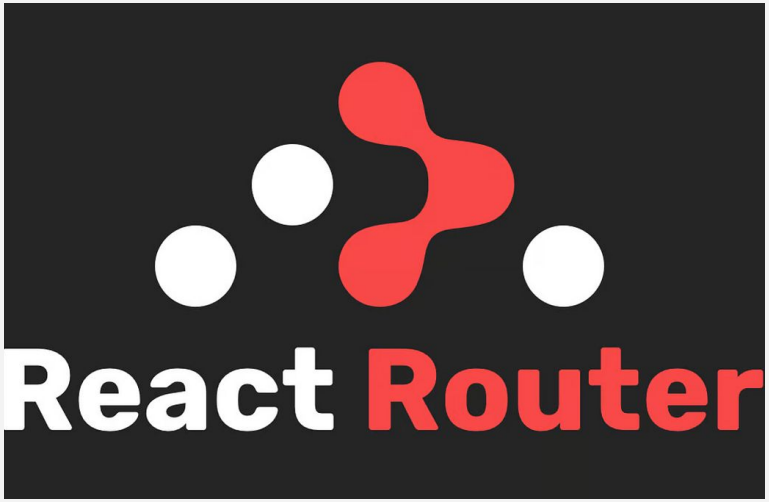
From their docs:

*“Client side routing allows your app to update the URL from a link click **without making another request for another document from the server**.*

[...]

*This enables **faster user experiences** because the browser doesn't need to request an entirely new document or re-evaluate CSS and JavaScript assets for the next page.”*

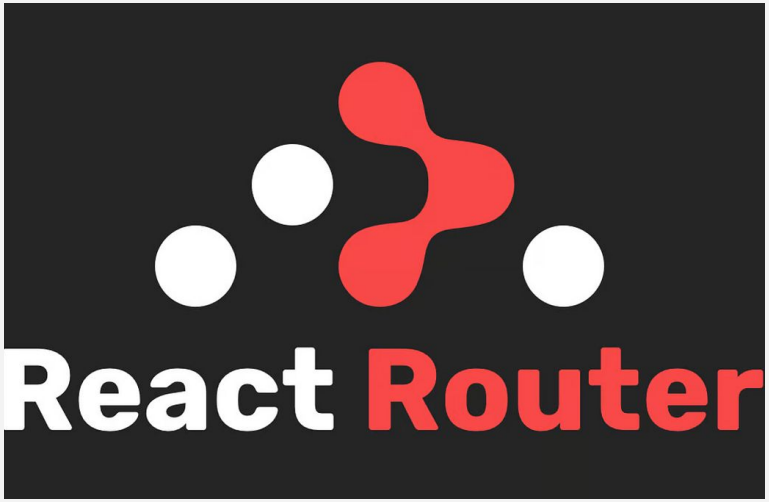
https://reactrouter.com/en/main/start/overview



How React Router Works

1. **Intercepts** server requests for ‘pages’ e.g., “../home”, “../about”, “../contact”, etc.
2. **Renders** the corresponding component (tree) for each route that’s needed e.g., <Home/>, <About/>, <Contact/>, etc.
    a. I say ‘tree’ because these components can have child components.
3. **Updates** the DOM with the relevant components, without needing to re-render all of them (i.e., reload the page)

*Important: the page does **not** reload. The DOM is simply **updated**.*



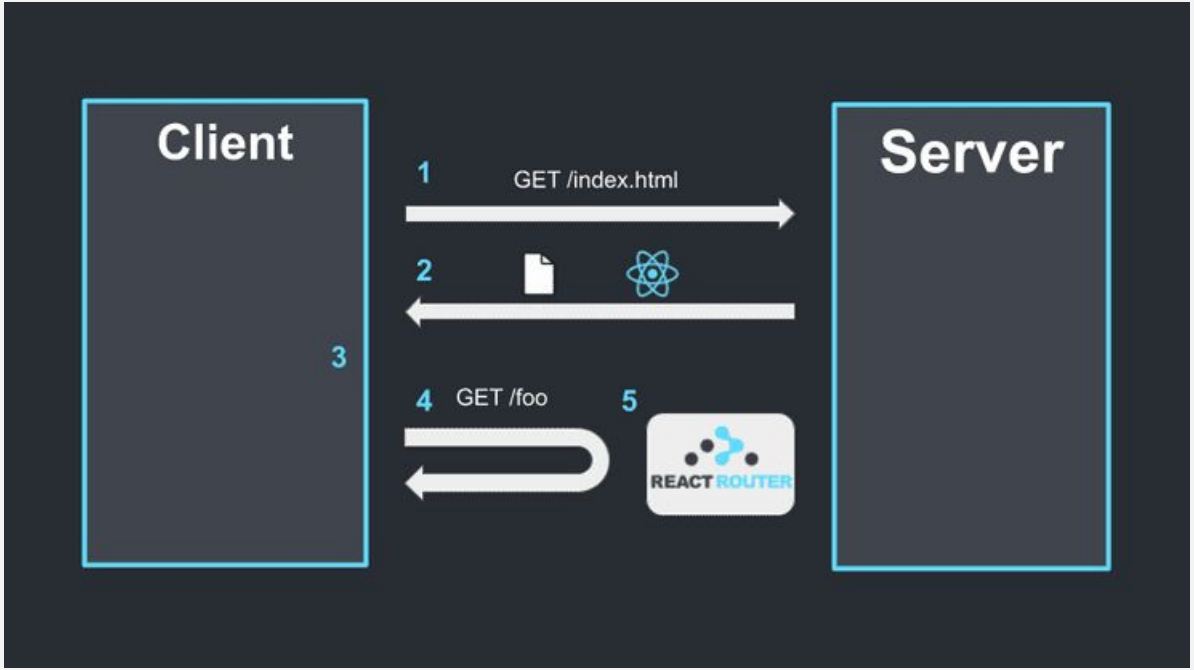
How React Router Works - Example

High-level Example

1. **Initial load** = index.html and entire React App (bundle).
2. **Route to a specific 'page'** e.g., /foo = RR intercepts and loads just the components necessary, leaving others alone.
3. This means the **page isn't reloaded** and components can keep their state.
4. React Router also has a number of other nifty features.

But isn't the initial loading much larger and slower then?

- Yes, and there is a way to fix this (beyond the scope of this lecture)



https://medium.com/data-science-community-srm/reactjs-routing-from-the-very-basics-226971205609

Basic React Router Demo (Old Method)

`<BrowserRouter></BrowserRouter>`

- Demo
  - Setting up the ‘pages’ (components)
  - Installing React Router
  - Setting up the `BrowserRouter` component
  - Setting up the routes (`Routes` and `Route` components)
  - Setting up links (`Link` component)
  - Routing in action

Alternate React Router Syntax (New v6+ Method)

```
const router = createBrowserRouter([...]);
```

- You may see this syntax being used along with a `RouterProvider` component
- It functions practically the same as the method we just used, but is the newer way of doing things (v6.4+) and is recommended by React Router in their documentation because more properties / APIs you can use with it besides `path` and `element`
- Instead of `<Route path="" element=""/>`
- You'll have `{ path: "" , element: "", …etc. }`

Dynamic Routes

Dynamic Routes are routes that are only determined when the request is made.

e.g., “.../products/12”

Sometimes you cannot determine ahead of time what specific route (URL) you will need.

- Fetching a specific product / event / profile / etc. out of hundreds.
- Cannot manually define routes for each one (what if we add or remove some?)
- **So we use dynamic routes.**

Dynamic Routes in React Router

Called ‘Dynamic Segments’

Specified using a colon (:)

- **e.g., ‘/products/`:id`’**
- Everything else is the same as we have done thus far

When a user navigates using a dynamic route, the route parameter can be accessed in the component that was loaded through React Router **useParams() function**

Dynamic Routes in React Router

However,

- **useParams()** in v6+ does **not support class components**, only functional ones.
- The method of getting around this has also been deprecated and scrapped in favour of functional components :/
- **Luckily, I have two workarounds for this (using v6)**
  - One that **only** works with the **new method**
  - One that works with both the **old** and **new methods**
  - Downgrading to v5 will let you use the established workaround method

Dynamic Routes in React Router

- Demo
  - Setting up a dynamic route e.g., “`/:id`”
  - Accessing the route parameter using `useParams()` using two workarounds

https://stackoverflow.com/questions/58548767/react-router-dom-useparams-inside-class-component

Next Steps

React Router has lots of nifty features

- Path matching (e.g., specifying exact routes with `exact` (v6)`/exactly` (v5) keyword)
- Getting if the current page is the active one (v5 and v6)
- Error / 404 pages (v5 and v6)
- Nested Routes (v5 and v6)
- Redirects (v5 and v6)
- Lots more

The things covered in this lecture is enough to get you started.

The docs cover more complex concepts.

React Router Docs Tutorial (functional components):
https://reactrouter.com/en/main/start/tutorial


react