

Single Page Application (SPA)

- A single HTML page w/JS to change contents
 - Can appear to be many "pages" to user
 - All one "page" to browser!
- React is not only way to make SPA
- React does not have to be a SPA
- SPA sometimes desirable
 - When complex JS state
 - No reload JS when no real page navigation
- SPA NOT always desirable!
 - Why not a plain website?
 - Why not a React generated website?

Navigation in a SPA

- Core SPA issue
 - Browser navigation reloads HTML + JS
- Links and Forms?
 - Change state and contents
 - Looks like page change
 - Do NOT actually navigate
 - Keeps base HTML + JS
 - Apply changes in content

Prevent forms and links from navigating

Links and Forms

- `.preventDefault()` on event
 - `onSubmit` (forms) or `onClick` (buttons)
- Update state so render changes HTML
 - Looks like a new page

Simple vs Complex

- Faking page navigation has lots of details
- Libraries are made to handle this **routing**
 - Ex: `react-router`, `@tanstack/router`
- We aren't using any such libraries
 - Not because the libraries aren't good
 - Learning the concepts, not one library
 - You can then learn any library
 - AFTER this course

Simple Example

- We want to show the user the current "page"
- So some variable is used to decide what to render
 - We will have a "page" state

```
const [page, setPage] = useState('Home');  
  
return (  
  <>  
    { (page === 'Home') && <Home/> }  
    { (page === 'About') && <About/> }  
  </>  
);
```

Simple Navigation

- Changing page needs to change page state
 - Pass the setter

```
const [page, setPage] = useState('Home');

return (
  <>
    <NavBar setPage={setPage} />
    { (page === 'Home') && <Home/> }
    { (page === 'About') && <About/> }
  </>
);
```

- Simplified example!
 - Learn the concept, not the specific syntax

Using the Setter

```
function NavBar({ setPage }) {  
  
  function go(event, page) {  
    event.preventDefault();  
    setPage(page);  
  }  
  
  return (  
    <nav>  
      <ul>  
        <li><a href="" onClick={ (e) => go(e, "Home") }>  
          Home  
        </a></li>  
        <li><a href="" onClick={ (e) => go(e, "About") }>  
          About  
        </a></li>  
      </ul>  
    </nav>  
  );  
}
```

Details to Consider

- What to use as href?
 - # vs #something vs a path
 - How does this impact UX?
- What about Back button?
- What about Reloading the page?
 - Or saving/sending the url?
- Can have different ways to call state change
 - Should preventDefault()
 - Way to change state needs to be passable

Core SPA Concepts

- Everything is one HTML file
 - Even when generated from many `.jsx`
 - Remember that's what the browser sees!
- **state** is a vital concept
 - Changed behavior is either CSS or state!
 - Or both
 - Functionality is a lot of state
- No clear line between UI and Functionality