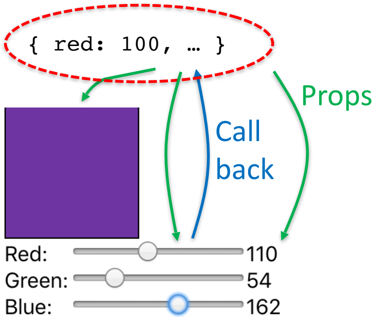
CS 312 Software Development

More React

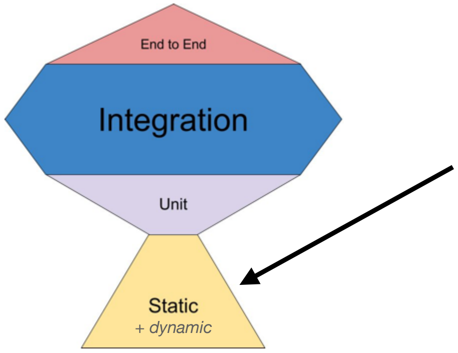
Philosophy of React

reminder

- There is a single source of truth (the state)
- Render the UI as it should appear for any given state of the application
- Update the state as a result of user actions
- Repeat (i.e., re-render the UI with the new state)



Reminder: The test hierarchy



- Typescript
- Flow
- Linter
- PropTypes

actually this is dynamic

Kent C Dodds "Write tests. Not too many. Mostly integration."

PropTypes

```
function LabeledSlider({ label, value, setValue }) {
  return (
    <div>
      <div className="color-label">{label}</div>
      <input type="range"
        min="0"
        max="255"
        value={value}
        onChange={(event)=>{setValue(parseInt(event.target.value,10))}}/>
      <span>{value}</span>
      </div>
  );
}

LabeledSlider.propTypes = {
  label: PropTypes.string.isRequired,
  value: PropTypes.number.isRequired,
  setValue: PropTypes.func.isRequired
}
```

NEXT.JS
marketing slide

- Zero config
  - bundles Webpack and Babel
  - scripts for building and exporting sites
- File-system routing
  - files in the pages directory become routes
- Built in development server with hot reloading
- Server-side page pre-rendering
  - build (SSG) and request time (SSR)
- Support for multiple styling solutions
  - e.g., CSS modules
- API Routes
  - add service APIs without writing an entire custom server

from https://nextjs.org/

NEXT.JS workflow

- Clone an existing repository *or* `npx create-next-app my-app` to create a new application
- Install dependancies with `npm install` (*core dependancies will be installed if you created a fresh app*)
- Run the development server with `npm run dev`
  - website can be viewed at localhost:3000
  - changes will appear automatically when you save files
- Build site for production with `npm run build`
- Run the site in production mode with `npm start`
- If installed, `npm test` to test and `npm run lint` to lint

Next folder structure
Default from create-next-app

```
my-app/
   README.md
   node_modules/
   package-lock.json
   package.json
   public/
      favicon.ico
   pages/
      _app.js
      api/
      index.js
   styles/
      globals.css
      Home.module.css
```

Next folder structure
Modified version

```
my-app/
   README.md
   node_modules/
   package-lock.json
   package.json
   public/
      favicon.ico
   src/
      components/
      pages/
         _app.js
         api/
         index.js
      styles/
         globals.css
         Home.module.css
```

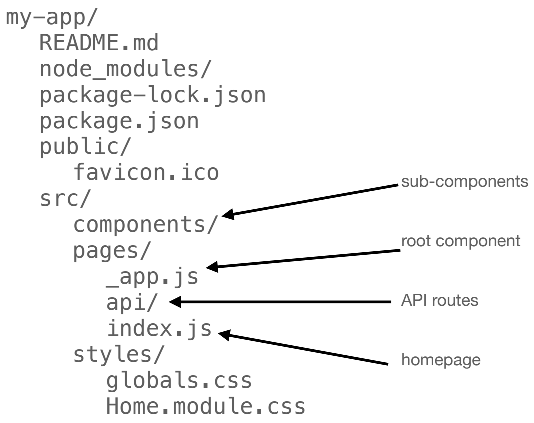
README.md, node_modules/, package-lock.json, package.json: normal NPM package infrastructure

public/, favicon.ico: static assets (like images)

components/ through Home.module.css: all JS and CSS files

Next folder structure

Modified version



```
my-app/
   README.md
   node_modules/
   package-lock.json
   package.json
   public/
      favicon.ico
   src/
      components/
      pages/
         _app.js
         api/
         index.js
      styles/
         globals.css
         Home.module.css
```

Styling components

- Static CSS files
  - global.css
- import CSS files like code
  - import “./styles.css”
- CSS modules
  - import style from “./ColorPicker.module.css”
- CSS-in-JS