~~How to Style your App~~

One approach to Style your App

with Chris Green

Angular Sussex Meetup 7/7/21

What this presentation will not include

- Tutorial on CSS
- Which CSS frameworks are the best
- Mixin Libraries/Atomic Design/ABEM
- LESS, Sass, Stylus

What the presentation will include

1. A demonstration of how badly structured SCSS can quickly get out of control
2. A step by step approach to refactoring a template, broken down as:
    - creating a semantic HTML layout
    - naming conventions for your markup classes
    - structuring SCSS for scaleability
3. Finally, how to make your application slicker by adding Angular animations

What will we refactor?

A custom profile page built using these components:

- Async data service
- Tabs component
- Accordion component
- Image component
- Loader component
- Menu component

- https://github.com/richgenres/style-your-app

Commonplace evolution of a template

- Profile page - initial version
- Profile page - revised version
- What’s happened to the CSS already?

Solution - refactor or patch?

- CSS can become unmaintainable if patched over time with poor architecture

- Refactoring gives us an opportunity to implement cleaner CSS architecture that can be maintained better and be more scaleable

Refactoring steps...

1. Sometimes just strip everything away and start again
2. Create a nested tree diagram of your page structure
3. Create a semantic HTML structure from the tree nodes created in (2)

   Layout.component
4. Next step... define a naming convention for your HTML element classnames

Why do we need a naming convention

- Just like coding standards, aim for a naming standard that every developer can follow and name HTML classes the same way

- A naming standard could provide us with a better way to understand the data it is describing

Introducing BEM - A class naming standard

- What is BEM? Block/Element/Modifier

BEM Fast Track

- Blocks are the highest level, ***eg: header***
- A block can have many elements, ***eg: logo***
- Naming convention for the logo is double underscore between block and element, ***eg: header__logo***
- Some elements can have a modifier with a double dash naming convention, ***eg: header__logo—large***
- BEM fully describes the data and the hierarchy
- BEM can be a little verbose

Apply BEM to our refactor

- flattened-BEM.component

Do’s and Dont’s of using BEM

- Avoid nesting BEM Elements
- Structured BEM - Best for > 3 levels deep

Create a working wireframe

wireframe-layout.component

- Clearly see what is going on with the page layout
- Enable you to focus only on the layout markup
- Get the layout markup right before adding any detail styling to the page

Complete your refactor

refactored-profile.component

- Add detail styling to your refactor

Make your app slicker

- Animations can add subtle professional touches to your App
- Don’t add too many animations - Less is more
- Complex page transitions can get annoying very quickly
- A page fade transition is a nice touch to give your app a nice feel

Summary

- Be consistent about structuring your HTML and SCSS across your app
- HTML should describe the data, ***eg: section***
- Follow a naming convention for HTML markup classnames ***eg: BEM***
- Be strict in Pull Requests to maintain these standards
- Remember FE development should be about visual presentation just as much as it is about Angular code patterns - make it visually appealing, but don’t over do it