ENTERPRISE CSS

Presented By

Tony MILLER

Who Are You?

- Senior software engineer at McKesson
- Specialize in front-end web development
- Advocate of UX process and practice
- Developed my first commercial website in 1997
- Wrote my first software application in 1978
    - It was a D&D encounter generator written in BASIC on a TRS-80

Enterprise CSS?

Is:

- Complex
- Large dev team
- Rapidly evolving
- Multi-sourced

Is not (necessarily):

- Large in scale or scope

THE GOALS

Enterprise Goals

Required to be:

1. **E**ffective
2. **M**aintainable
3. **R**eusable
4. **E**xtensible
5. **S**calable
6. **T**estable

May not be:

- Performant
- Efficient to create
- Elegant

THE CHALLENGES

Challenges

- Mutation and decay over time
- Varying level of team member skill
- Frameworks
- Best practices

THE PRINCIPLES

Principles

- Maximize readability
  - Class names are for developers, not machines (except for Microformats)
  - Class names should express role & relationships, not content semantics and/or visual characteristics
- Minimize selector specificity
- Maximize modularity

CSS Quick Review

Cascade

1. Origin + Importance:

    USER important > AUTHOR important > AUTHOR normal > USER normal > AGENT normal

2. Specificity (X,X,X,X):

    (inline, id, class/attribute/pseudo-class, element/pseudo-element)

3. Source Order

Inheritance

http://www.w3.org/TR/CSS21/propidx.html

Maximize Modularity

- **Separation of Concerns (SoC)**
  - Decompose purpose into distinct bundles that overlap as little as possible
- **Single Responsibility Principle (SRP)**
  - Bundles should focus on one (or close to one) purpose only
- **Composition over Inheritance**
  - In an instance, prefer combining bundles over inheriting purpose from a parent instance

Maximize Modularity (cont.)

- Patterns (Abstraction)
  - DRY
- Encapsulation
  - Decouple CSS from HTML
  - Focused selector intent
    - **Good**: Classes
    - **Not so good**: Child/Sibling selectors
    - **Bad**: Descendant selectors

THE TECHNIQUES

Techniques

- OOCSS
- BEM
- SMACSS
- CSS Global Namespacing
- General Style

Object-Oriented CSS (OOCSS)

Coined by Nicole Sullivan (@stubbornella)

A CSS “object” is a repeating pattern that can be abstracted into an independent, reusable snippet of HTML, CSS and possibly JavaScript.

Principles of OOCSS

1. Separate structure and skin
2. Separate container and content

BEM (Block Element Modifier)

BEM is a “technology for creating web applications” created by Yandex. I use a variant of the naming part only, authored by Nicolas Gallagher (@necolas).

- Block is a higher level abstraction that corresponds to an OOCSS object or a SMACSS module
- Elements are component that helps form the block
- Modifier corresponds to a state or variant of the block

SMACSS

Scalable and Modular Architecture for CSS (SMACSS), created by Jonathan Snook (@snookca). Focused on categorizing and organizing CSS to streamline maintenance and modification process.

Categories are:

1. **Base** - Defaults, usually elements only
2. **Layout** - Positioning and sectioning (e.g. grids, sidebar, etc)
3. **Module** - Equivalent of OOCSS objects and BEM blocks
4. **State** - Styles that govern changes of state (e.g. active, open, etc.)
5. **Theme** - Visual overrides that implement site design scheme.

CSS Global Namespacing

"Global namespacing in CSS is a dirty act of compromise."

- **Me**, every damn day

Problems:

- Naming collisions break your site
- Frameworks and/or injecting outside CSS make collisions likely

Solution: Prepend a global prefix to **all** your classes

Downsides are:

- Uglier, less scannable class names
- Pain in the ass without using a pre-processor
- Feels dirty

General Style

- Use Comments
  - Table of Contents (TOC)
  - Global values
- Multiline
- Use shorthand notation only to change **all** values from default
- Align related declarations
- Related selectors on one line, unrelated on new line
- Use whitespace

THE CREDITS

Sources and Resources

https://github.com/stubbornella/oocss/wiki

https://bem.info/

https://smacss.com/

http://cssguidelin.es/

http://nicolasgallagher.com/about-html-semantics-front-end-architecture/

https://developer.mozilla.org/en-US/docs/Web/CSS

http://specificity.keegan.st/

THANK YOU