Writing maintainable and extensible CSS

Mato Žgajner, 2014

Complex projects …

… and puny CSS

Issues

- repetition
- high specificity
- lack of structure

Solutions

Reusability

- relative units
- variables & calculations
- **OOCSS**

Structure

- **SMACSS**
- breakpoint mixins
- auto-prefixing
- other Compass/Bourbon/...

magic

Relative units

px

```
.button {
    font-size: 20px;
    height: 40px;
    line-height: 20px;
    padding: 10px 20px;
}

.button.desktop {
    font-size: 10px;
    height: 20px;
    line-height: 10px;
    padding: 5px 10px;
}
```

em

```
.button {
    font-size: 20px;
    height: 2em;
    line-height: 1em;
    padding: 0.5em 1em;
}

.button.desktop {
    font-size: 10px;
}
```

Variables - basic reuse

Sass

```
$default-font-size: 24px;

.button {
    font-size: $default-font-size;
}

p {
    font-size: $default-font-size;
}
```

CSS

```
:root { var-hot-pink: #ec008c; }

.button {
    background-color: var(hot-pink);
}

a {
    color: var(hot-pink);
}
```

Variables - calculations

Sass

```
$hot-pink: #ec008c;

.button {
	background-color: $hot-pink;
}

.button:hover {
	background-color: darken($hot-pink, 20%);
}
```

Sass

```
$default-font-size: 14px;

h1 {
  font-size: $default-font-size * 2;
}
```

Sass + Bourbon

```
h1 {
  font-size: golden-ratio($default-font-size, 1);
}
```

Object Oriented CSS

Nicole Sullivan, 2009

- separate structure and skin
- separate container and content

Separate structure and skin

before

```
.box {
        width: 400px;
        overflow: hidden;
        border: solid 1px #ccc;
        background: linear-gradient(#ccc, #222);
        box-shadow: rgba(0, 0, 0, .5) 2px 2px 5px;
}

.widget {
        width: 500px;
        min-height: 200px;
        overflow: auto;
        border: solid 1px #ccc;
        background: linear-gradient(#ccc, #222);
        box-shadow: rgba(0, 0, 0, .5) 2px 2px 5px;
}
```

after

```
.box {
        width: 400px;
        overflow: hidden;
}

.widget {
        width: 500px;
        min-height: 200px;
        overflow: auto;
}

.skin {
        border: solid 1px #ccc;
        background: linear-gradient(#ccc, #222);
        box-shadow: rgba(0, 0, 0, .5) 2px 2px 5px;
}
```

Separate container and content

bad

```
footer .button {
	width: 200px;
	height: 50px;
	padding: 10px;
	border: solid 1px #ccc;
	background: linear-gradient(#ccc, #222);
	box-shadow: rgba(0, 0, 0, .5) 2px 2px 5px;
}
```

good

```
.button {
	width: 200px;
	height: 50px;
	padding: 10px;
	border: solid 1px #ccc;
	background: linear-gradient(#ccc, #222);
	box-shadow: rgba(0, 0, 0, .5) 2px 2px 5px;
}
```

Result - building blocks

- a bunch of modules
- just stick classes to elements
- use preprocessors for clean markup

```
%blue {
        background-color: light-blue;
        color: dark-blue;
}

%funny {
        font-family: 'Comic Sans MS', cursive;
        font-weight: bold;
}

.button {
        @extend %blue;
        @extend %funny;
}
```

SMACSS

Jonathan Snook, 2011

Structuring of styling into 4(5) groups:

- Base
- Layout
- Module
- State
- Theme

Base

- reset or normalize
- plain element styling
- no classes - no prefix

Layout

- major page structure
- footer, header, sidebar etc.
- prefix with l-* or layout-*

Modules

- majority of all styling
- no default prefix
- prefix submodules with *parent*-*

```
.button {
	font-size: 20px;
	height: 2em;
	line-height: 1em;
	padding: 0.5em 1em;
}

.button-info {
	background-color: blue;
}

.button-warning {
	background-color: red;
}
```

States

- applied with JS
- prefix with is-*

```
.tab {
    background-color: purple;
    color: white;
}

.is-tab-active {
    background-color: white;
    color: black;
}
```

Summary

- IMHO _the_ book on advanced CSS
- full of other advice:
  - selector performance
  - specificity
  - prototyping
  - ...

Responsive mixins

Sass

```
@mixin respond-to($point) {
    @if $point == "desktop" {
        @media (min-width: 1200px) {
            @content;
        }
    }
}

.some-class {
    font-size: 16px;
    @include respond-to("desktop") {
       font-size: 14px;
    }
}
```

CSS result

```
.some-class {
       font-size: 16px;
}

@media only screen and (max-width: 1200px) {
       .some-class {
              font-size: 14px;
       }
}
```

Other preprocessor magic

- Sprites
- Grids
- Typographic scales

- Compass
- Bourbon
- Susy
- ...

Koniec

Stuff that was mentioned:

- OOCSS
  https://github.com/stubbornella/oocss/wiki
  http://www.smashingmagazine.com/2011/12/12/an-introduction-to-object-oriented-css-oocss/
- SMACSS
  http://smacss.com/
- Sass
  http://sass-lang.com/
- Compass
  http://compass-style.org/
- Bourbon
  http://bourbon.io/

Get this presentation on Twitter:

@matozgajner