headacheFree.css

Ben Schmidt
@b3nschmidt / @10PoundGorilla

10 POUND gorilla

- Web Development
- Internet Marketing
- SEO & Adwords
- Social Media
- Email Marketing
- Graphic Design

Working World-Wide with DNN Since 2003

CASSIDI
founder

DUSTIN
director

BOB
manager

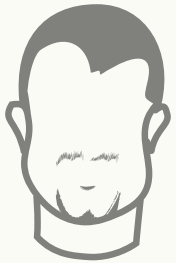
LEE
designer/dev

BEN
designer/dev

MARK
technical manager

JONATHAN
sales

KEVIN
dev



AARON
software dev

CSS makes things look neat

BUT

it’s far from looking neat

Why Should You Care

It Makes Life Easier

- Makes for easier maintenance
- Allows people of varying skillsets to work on the same project while maininting the same level of quality throughout
- Productivity goes up, Profanity goes down

TL,DR

TB,DL
(To Boring, Didn't Listen)

Components exist and look right even if their parent elements are removed or changed.

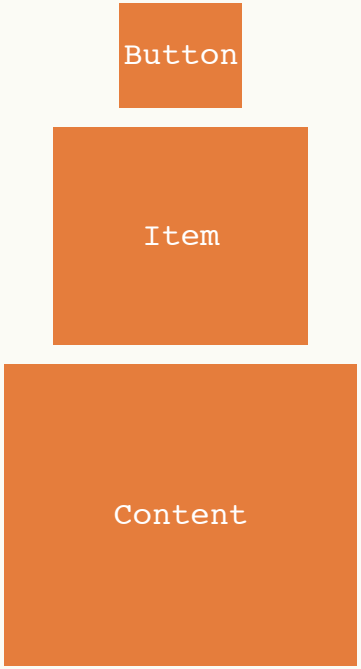
```
.content {
     background: gray;
     padding: 10px;
}

.content__item {
     border: 1px solid black;
     padding: 10px;
}



.btn {
     background: blue;
     border-radius: 5px;
     color: white;
     display: inline-block;
     padding: 5px 15px;
}
```

NOT THIS



Components here are largely dependent upon parent elements

```
.content {
     background: gray;
     padding: 10px;
}

     .content div {
          border: 1px solid black;
          padding: 10px;
     }

     .content a {
          background: blue;
          border-radius: 5px;
          color: white;
          display: inline-block;
          padding: 5px 15px;
     }
```

headacheFree.css Goals

01 Keep code readable

02 Keep code maintainable

03 Keep code adaptable & scalable

The Easy Stuff

Syntax & Formatting

Line-Length

Keep line-length in CCS to a maximum width of

80 Characters

- Helps when viewing multiple files side by side
- Easier Readability
- Better legibilty for git sites and terminal

Writing Rulesets

Follow the Universal Standard

NO

```
.shape {
     background:blue;
     display:block;
     font-size:20px;
     padding:20px; }
```

```
.shape, .shape__rectangle, .box {
     background:blue;
     display:block;
     font-size:20px;
     padding:20px; }
```

Writing Rulesets

Follow the Universal Standard

YES

```
.shape {
     background: blue;
     display: block;
     font-size: 20px;
     padding: 20px;
}
```

```
.shape, .shape__rectangle,
.box {
     background: blue;
     display: block;
     font-size: 20px;
     padding: 20px;
}
```

Writing Rulesets

Use Multi-Line CSS

NO

```
.shape { background: blue; display: block; font-size: 20px }
```

Writing Rulesets

Use Multi-Line CSS

YES

```
.shape {
     background: blue;
     display: block;
     font-size: 20px;
}
```

```
/*-------------------------------------------
     #EXCEPTIONS: Single Line Declarations
-------------------------------------------*/

.shape--green { background: green; }
```

Spaces or Indents

The Great Debate

Use Smart Indenting

We indent HTML, So why Not CSS? (Not Nesting)

```
<div class="box">
     <div class="box__image">
          <div class="box__text">
     </div>
</div>
```

```
.box { }

     .box__image { }

          .box__text { }
```

One empty line between related rulesets

Indent classes in relation to their parent

Easy to see where the elements live

No need to reference HTML

What About Nesting?

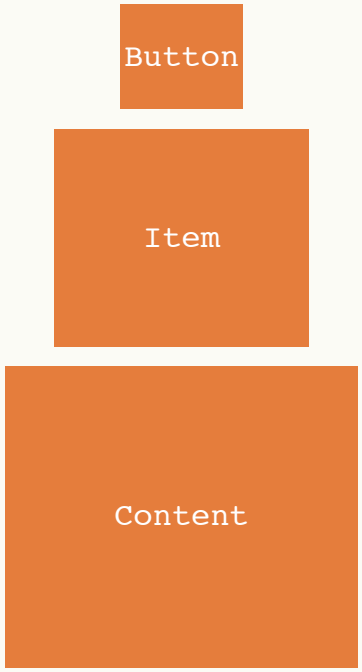
Pre-Processors

```
.content {

      .content__item {
      }


            .btn {
            }

}
```



Smart Commenting

Make a Table of Contents

```
/*

----------------------------------------
#TYPOGRAPHY
----------------------------------------
     Headers......H1-H6
     Styles.......P Tags, Links, Buttons

----------------------------------------
#Site Header
----------------------------------------
     Logo
     Navigation

----------------------------------------
#Banner
----------------------------------------
     Backgrounds......Images, Overlays
     Type.............Styles, Color, Size

/*
```

Smart Commenting

Headers & Sub-Headers

Start Each New Section with a Title

```
/*-------------------------------------
    #Typography
-------------------------------------*/
```

Use Double-Line Comments for Sub-Sections within a Section

```
/* Headers
-------------------------------------*/
```

```
/*-------------------------------------
    #Typography
-------------------------------------*/

/* Headers
-------------------------------------*/

h1 { }

h2 { }

/* Styles
-------------------------------------*/

p { }

a { }



/*-------------------------------------
    #Site Header
-------------------------------------*/

/* Logo
-------------------------------------*/

.site-logo { }
```

White Space

5 empty lines between new sections

One empty line between section titles and sub headers

One empty line between closey related rulesets

Two empty lines between loosely related rulesets

Use More Comments

CSS Is Hard to Distinguish

It's a Bad Story Teller

Are styles inheriting values from elsewhere?

Will changing styles have consequences?

Is a style being used elsewhere?

Why elements have certain sytles?

What components are related to this ruleset?

Why does something have a certain height?

Use More Comments

Do Future You a Favor

If It's Not Obvious, It Needs a Comment

```
.section { color: red; }
```

```
/*
 *This has an overflow to clear a float
*/
.section { overflow: hidden; }
```

Use More Comments

Comments When Working with Multiple Files

```
/*
 * .btn extensions in colors.css
*/
.btn {
     border-radius: 5px;
     color: #fff;
     display: inline-block;
     font-size: 18px;
     padding: 15px;
}
```

```
/*
 * These extend .btn in general.css
*/
.btn--green { background: green; }
.btn--blue { backgorund: blue; }
.btn--red { background: red; }
```

Media Queries & Supports

Organization

```
/*-------------------------------------
     #Main Section
-------------------------------------*/

.main-section { width: 100%; }

@media screen and (min-width: 992px) {

     .main-section {
          float: left;
          width: 50%;
     }
}

/*-------------------------------------
     #Side Section
-------------------------------------*/

.side-section { width: 100%; }
```

```
.blog__photo {
     float: left;
     height: 140px;
     width: 140px;
)

/*
 *Convert image to circle and add circle rag
 *Browser Support: Chrome, XXX, XXX
/*
@supports (shape-outside: circle(50%)) {

     .blog__photo {
          border-radius: 50%;
          shape-outside: circle(50%);
     }

}
```

Diving Deeper

Naming Conventions

“There are only two hard things in Computer Science: cache invalidation and **naming things.**”

Phil Karlton

Choosing Class Names

It's Harder Than You Think

Clear & Vague

.site-navigation → **.primary-navigation**

.footer-links → **.sub-links**

Clear in that they're navigation & links

Vague in that it's not directly clear where they belong

Allows for reusability

Naming Conventions

Good Class Names

Say What It Does

How They Can be Used

What it's Related To

Choosing Class Names

Pay Attention to Longevity & Reusability

```
/*-------------------------------------
     Not Maintainable
-------------------------------------*/

.red { color: red; }

/*-------------------------------------
     Location Dependent
-------------------------------------*/

.header p { color: red; }

/*-------------------------------------
     Too Limiting
-------------------------------------*/

.header-color { color: red; }

/*-------------------------------------
     Reusable / Easily Changeable
-------------------------------------*/

.highlight-color { color: red; }
```

Choosing Class Names

Real Life Example

```
/*-------------------------------------------------------
    DNN Summit Background Colors
-------------------------------------------------------*/

/* Backgrounds */
.bg-primary { background: #1E345D; } /* Navy Blue */
.bg-secondary { background: #E57D3C; } /* Orange */
.bg-tertiary { background: #727126; } /* Green */
.bg-quaternary { background: #138A87; } /* Turqouise */
.bg-quinary { background: #EBEBCA; } /* Tan */
```

What Changed?

Two Lines of CSS

0 HTML Classes Changed

Naming HTML Classes

```
<div class="box team-member executive">
     <div class="profile">
          <div class="photo bio">

          </div>
     </div>
</div>
```

How are these classes related?

Can they be used outside of this componenet?

Where do these classes live in css?

```
<div class="box team-member team-member--executive">
     <div class="team-member__profile">
          <div class="photo team-member__bio">

          </div>
     </div>
</div>
```

Easily see what classes relate to each other

See which elements can be used outside of this component

Know what classes rely on parent elements

Life-Pro-Tip

Use Slashes to Group HTML Classes

```
<div class="col-sm-12  pane  /  box  /  team-member  team-member--executive">

</div>
```

Benefits?

Easy to see how classes relate to each other

Easy for other developers to make sense of it

Javascript HTML Classes

```
<div class="btn">

</div>


$(".btn").click(function(){
     $(".content").show();
});
```

No way to know js is tied to this class just by looking at HTML

Any time a ".btn" is clicked, ".content" will show

```
<div class="btn js-btn">

</div>


$(".js-btn").click(function(){
     $(".content").show();
});
```

Clearly visible that js is related to this

Can freely use ".btn" without worrying about related js

Naming Conventions

Hyphen Delimeters

```
.site-nav

.site-logo

.links
```

YES

```
.siteNav

.site_logo

.Links
```

NO

Naming Conventions

BEM Like Syntax

Block	The Sole Root of the Component
Element	A Component Part of the Block
Modifier	A Variant or Extension of the Block

Naming Conventions

BEM Like Syntax

```
/* Block */
.house


/* Element */
.house__window


/* Modifier */
.house--large
```

Blocks are written as single words or with a hyphen delimeter

Elements are delimited with 2 underscores (__)

Modifiers are delimited by 2 hyphens (--)

house = main component | house__window = piece of house | house--large = changes house

Title of Blog Post

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur suscipit dictum neque et accumsan. Fusce porttitor ut tortor vitae tempus. Integer

Title of Blog Post

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur suscipit dictum neque et accumsan. Fusce porttitor ut tortor vitae tempus. Integer

Title of Blog Post

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur suscipit dictum neque et accumsan. Fusce porttitor ut tortor vitae tempus. Integer

```
/* Block */
.blog-item


/* Element */
blog__content


/* Modifier */
.blog-item--featured
```

Selectors & Reusability

Be a Fortune Teller

Selector Intent

Not Where, But What

`header ul { }`	Styles any list in the header What we really meant was to style the site's main navigation Any future lists in the header will inherit these styles
`.primary-nav`	Only styles the intended list in the header Can be re-used No conflict with future lists

Selector Intent

Not Where, But What

`.checkout a`	Call to Action Button Styles a tags inside of .checkout Can't be reused Future a tags inside checkout will always be styled this way

`.btn--checkout`
 	
`"btn btn--checkout"`
	Call to Action Button
	
	Reusing code from exisisting button structure
	
	Reusable
	
	Can add future a tags without having to override existing css

Think Ahead

Don’t Be Lazy

```
.content a {
  position: relative;
  color: #000;
  text-decoration: none;
}

.content a:hover {
  color: #000;
}

.content a:before {
     background: #000;
     bottom: 0;
     content: "";
     height: 2px;
     left: 0;
     position: absolute;
     transition: all 0.3s ease-in-out 0s;
     width: 100%;
}

.content a:hover:before {
  visibility: visible;
  -webkit-transform: scaleX(1);
  transform: scaleX(1);
}
```

I Messed Up

Wanted inline text links to have a cool hover border effect.

This caused any link (AKA Buttons) inside of ".content" to be wonky because of the :before styles.

Think Ahead

Don’t Over Complicate

UX DEV

headacheFree.css

Ben Schmidt

While it might sound like one of the other million css frameworks out there, headacheFree.css isn't so much a framework but a way of thinking. Writing css that's easily manageable, scalable, and headache free is always an end goal that we start out each project with high hopes for but as the site grows and client changes keep rolling in, what started off clean and pretty is now a huge mess, that even you, the one who wrote it, struggles to maintain efficiently. This headacheFree.css methodology will walk you through naming conventions, syntax, organization, and best practices so you can keep your css headache free.

```
<div class="session">

    <!-- Category -->
    <div class="session__category cat-one-bg">
        <div class="session__category--name">UX Dev</div>
    </div>

    <!-- Info -->
    <div class="session__info">
        <h5 class="session-title cat-one-text">headacheFree.css</h5>

        <div class="session__author__img cat-one-border">
            <img src="/Ben-Schmidt.jpg" />
        </div>

    </div>


</div>
```

UX DEV

headacheFree.css

Ben Schmidt

While it might sound like one of the other million css frameworks out there, headacheFree.css isn't so much a framework but a way of thinking. Writing css that's easily manageable, scalable, and headache free is always an end goal that we start out each project with high hopes for but as the site grows and client changes keep rolling in, what started off clean and pretty is now a huge mess, that even you, the one who wrote it, struggles to maintain efficiently. This headacheFree.css methodology will walk you through naming conventions, syntax, organization, and best practices so you can keep your css headache free.

```
<div class="session session--cat-one">

    <!-- Category -->
    <div class="session__category">
        <div class="session__category--name">UX Dev</div>
    </div>

    <!-- Info -->
    <div class="session__info">
        <h5 class="session-title">headacheFree.css</h5>

        <div class="session__author__img">
            <img src="/Ben-Schmidt.jpg" />
        </div>

    </div>


</div>
```

Tips for Reusability

Break Components into 2 Categories

Structure

Layout
Padding
Display

Cosmetics

Colors
Fonts
Weights

Tips for Reusability

Breaking Components into 2 Categories

```
/* Extend this button with ".btn--* skin class */
.btn {
     border: 1px solid;
     display: inline-block;
     padding: 10px;
}

/* Primary Button - Extends ".btn" */
.btn--primary {
     background: blue;
     border-color: blue;
     color: #fff;
}
```

.btn handles the structure of the button

.btn--primary handles the look and feel of the button

One can be modified without affecting the other

Scalability

Focus on One Piece at a Time

```
<div class=”row”>

     <div class=”col-sm-6 / item “>
     </div>

     <div class=”col-sm-6 / item “>
     </div>

</div>


<!-- ....................................................

     Our layout exists on its own and isn’t dependent on
     anything else

.................................................... -->
```

```
<div class="row">

    <div class="col-sm-6 / item ">

        <section class="content">
        ...
        </section>

    </div>

    <div class="col-sm-6 / item ">

        <section class="content">
        ...
        </section>

    </div>

</div>


<!-- ..................................................

    Our content lives inside the layout. Working one piece
    at a time ensures that when we modify layout, only the
    layout is modified — nothing else. They live independtly
    of each other.

.................................................. -->
```

Modifications

Modifications Shouldn't Be Mandatory

Any New Additions & Features Should be Opt-In Only

Make modifications via extensions: .class--ext

Lots of variations can be made without stepping on the original

Allows for small changes; not changes across the board

Adding Modifications

Use Extensions

Wrong

```
.box {
     display: block;
     width: 20px;
}

.content .box { width: 40px; }
```

Right

```
.box {
     display: block;
     width: 20px;
}

.box--large { width: 40px; }
```

Look Clean

=

Feel Clean

=

Headache Free

Resources

Taking it Further

csswizardry.com

cssguidelin.es

maintainablecss.com

THANKS TO OUR SPONSORS

Managed.com
Managed Website Hosting and Support

DNN

10 POUND gorilla

CARTER SOLUTIONS

DNN4Less™
DotNetNuke It's in our ...DNA

GRAVITYWORKS
DESIGN + DEVELOPMENT

nvisionative