

CSS: THE SPECIFICITY WARS

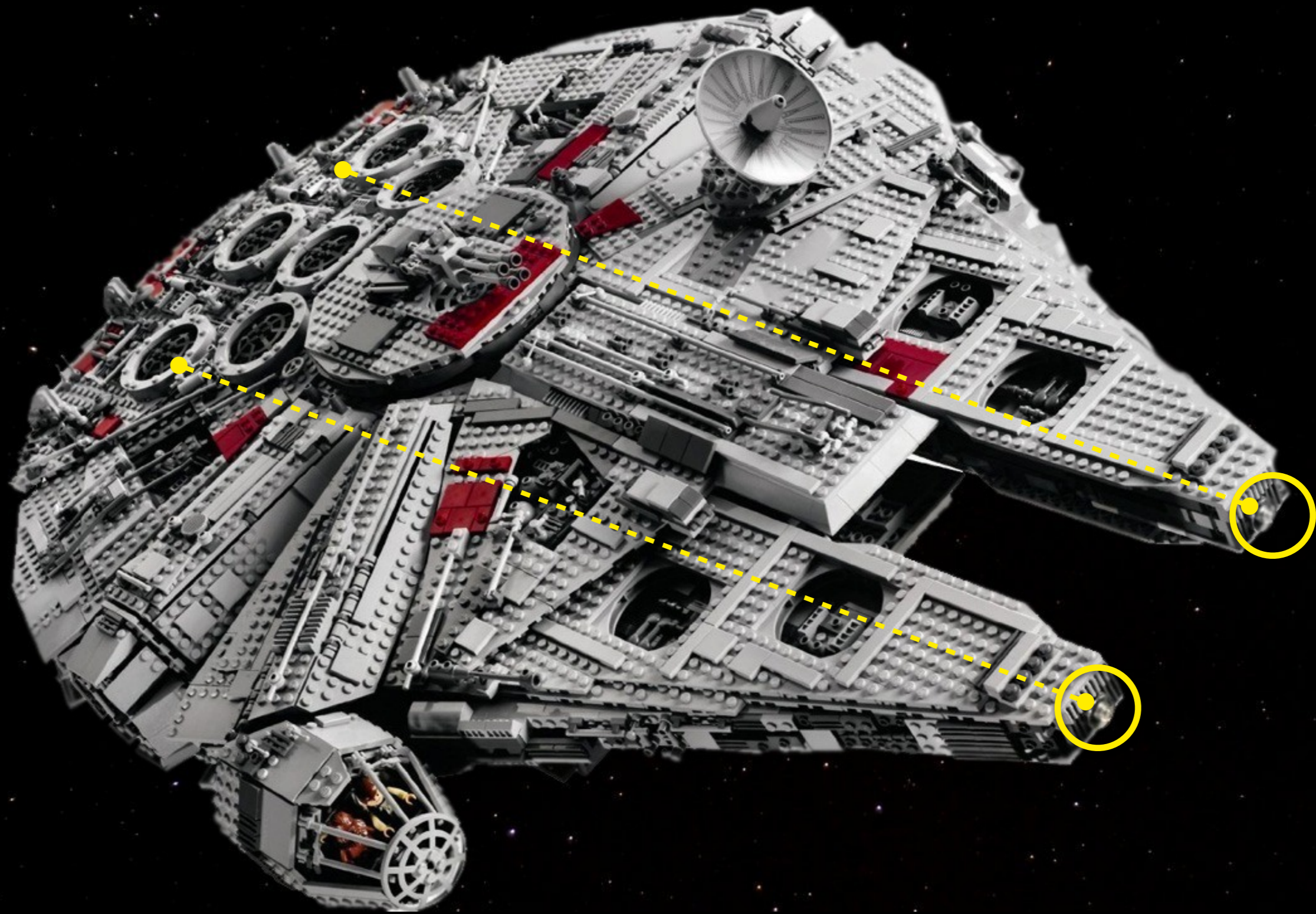
Benjamin Hong

Front End Developer

@bencodezen

“Two CSS properties walk into a bar.
A bar stool in a completely different
bar falls over.”

- Thomas Fuchs



CSS

CRAZY SPECIFIC STYLES

“It doesn’t matter how well-considered your source order is; how well you’re utilizing the cascade; what naming conventions you use; **specificity can undo everything.**”

- Harry Roberts (@csswizardry)

WHAT'S SPECIFICITY?

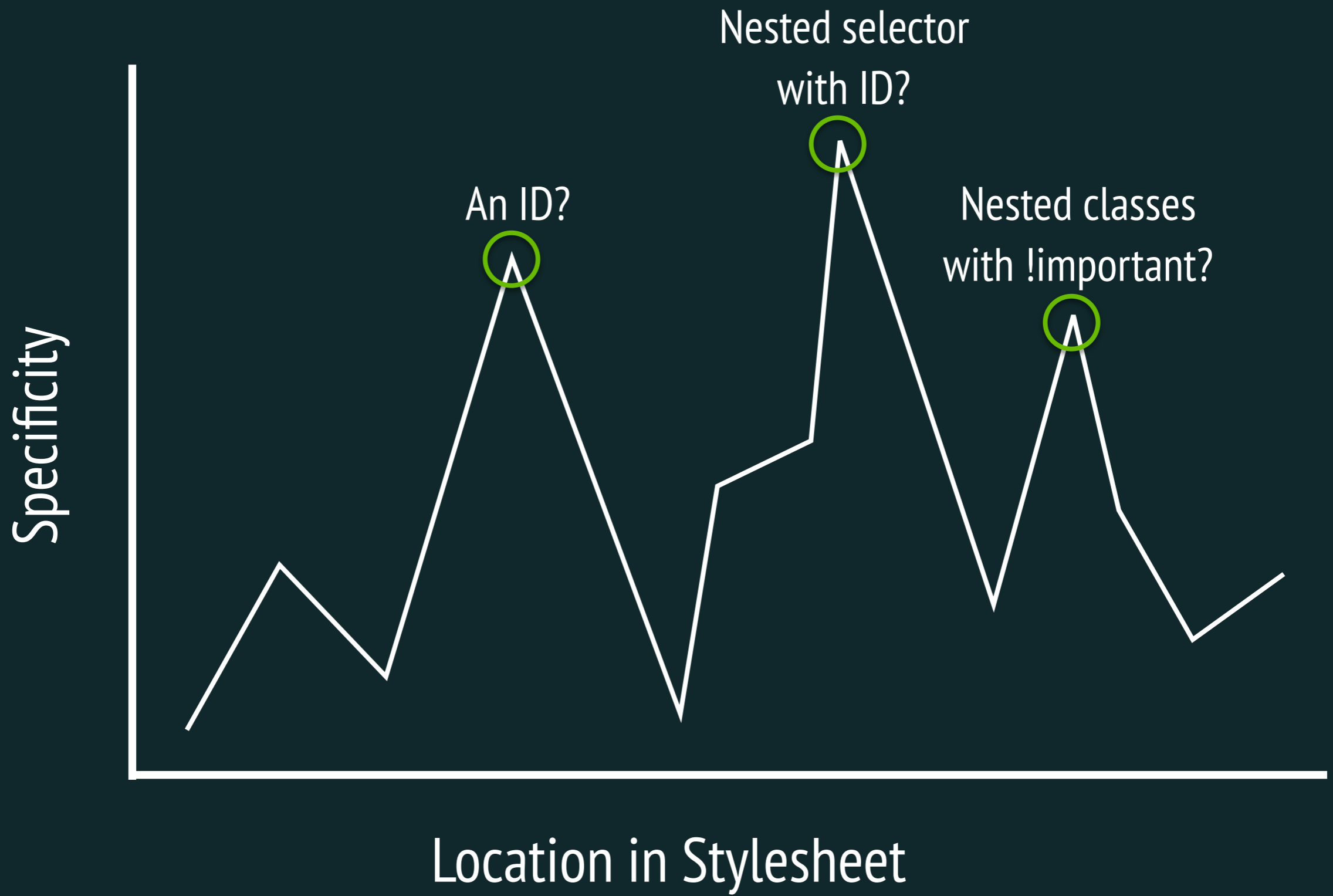


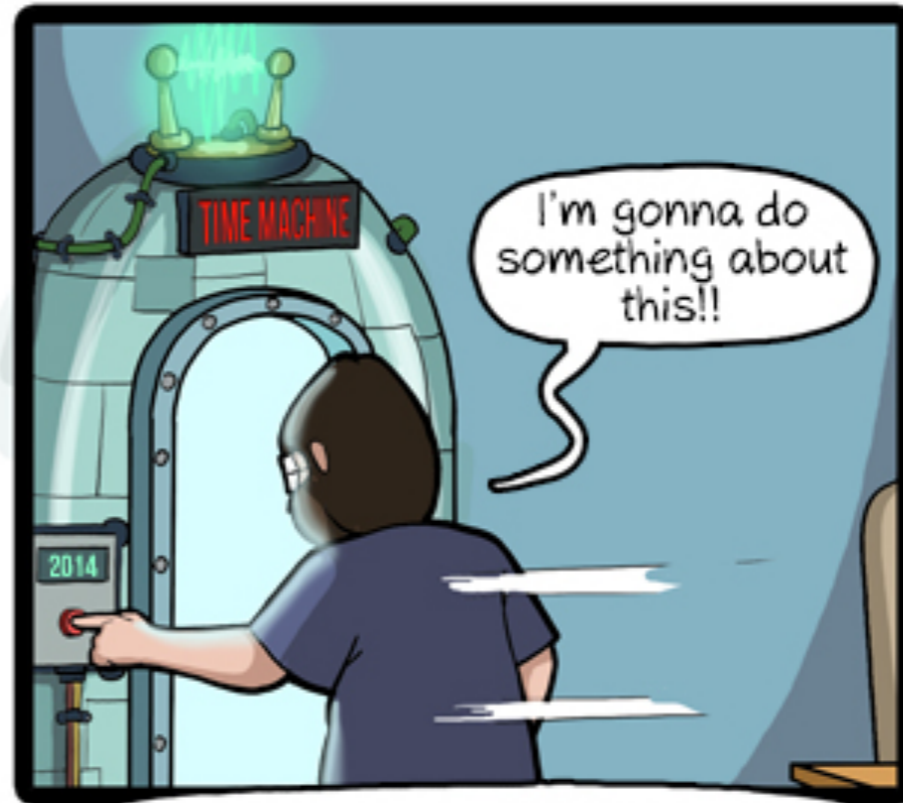
main.css

3,000 - 5,000
lines of code

```
#main-menu ul ul {position: absolute;left: 100%;top: 0;}.search-bo
display: none;list-style: none;box-shadow: 0 -3px rgba(0, 0, 0, 0.
0.1) inset; -ms-box-shadow: 0 -3px rgba(0, 0, 0, 0.1) inset;}.s
button {transition: all 0.25s ease-in-out;-ms-transition: all 0.25
100%;margin: 0;padding: 7px 35px 7px 13px;outline: none;border: no
white;border-color: #555;}#footer .s:focus {border-color: transpar
outline: none;}.s:focus+button {color: #555;}.s+button i {margin-r
wrap {padding-left: 5%;padding-right: 5%;}#page-main {padding-top
margin-bottom: 80px;position: relative;}.post-main, .template-cont
, 0, 0.085) inset; -moz-box-shadow: 0 -1px rgba(0, 0, 0, 0.085)
.085) inset;}.post-main.page-main {margin-bottom: 80px;}.page-titl
bottom: 80px;}.page-404 {text-align: center;}.page-404-title {font
{text-align: center;position: relative;}.entry-title {margin-bott
last-of-type {margin-bottom: 0;}.entry-content > ul {list-style-ty
block;position: relative;line-height: 1;background: #ff6;}.entry-m
32px;position: relative;}.entry-meta a {position: relative;}.entr
opacity: 0
ease-in-ou
entry-meta
in-out;-we
0;opacity
entry-cate
entry-tags
uppercase;
color: tra
{display:
100%;left:
all 0.4s
margin-top
cover {position: absolute;top: 0;left: 0;width: 100%;height: 100%;
: cover;-webkit-background-size: cover;-ms-background-size: cover;
;font-family: NovecentowideDemiBold, sans-serif;font-size: 54px;po
bold;text-transform: uppercase;position: relative;display: inline-
block;}.post-quote > a:hover {opacity: 0.8;}.post-link {background
position: relative;z-index: 10;overflow: hidden;}.entry-link:befor
);z-index: -10;transition: all 0.3s ease-in-out;-ms-transition: al
link:hover::before {top: 0;}.link-icon {margin: 0;padding: 0 15px
-transition: all 0.2s ease-in-out;-moz-transition: all 0.2s ease-i
audio.audio-style1 {padding: 50px 8%;background-repeat: no-repeat;
;height: 40px !important;display: block;}.post-audio .mejs-contain
none;position: static;display: block;}.mejs-controls .mejs-time sp
px;background: url(./images/play.png);background-repeat: no-repeat
mejs-pause { background-image: url(./images/pause.png); }.post-aud
```







Instead of building beautiful products, you're **wasting time and energy** fixing someone's poorly written code.

WHY NOW?

Why is this so
• important all of a sudden?

- Everything is “agile”
- Websites have evolved
- Explosion of interest in coding

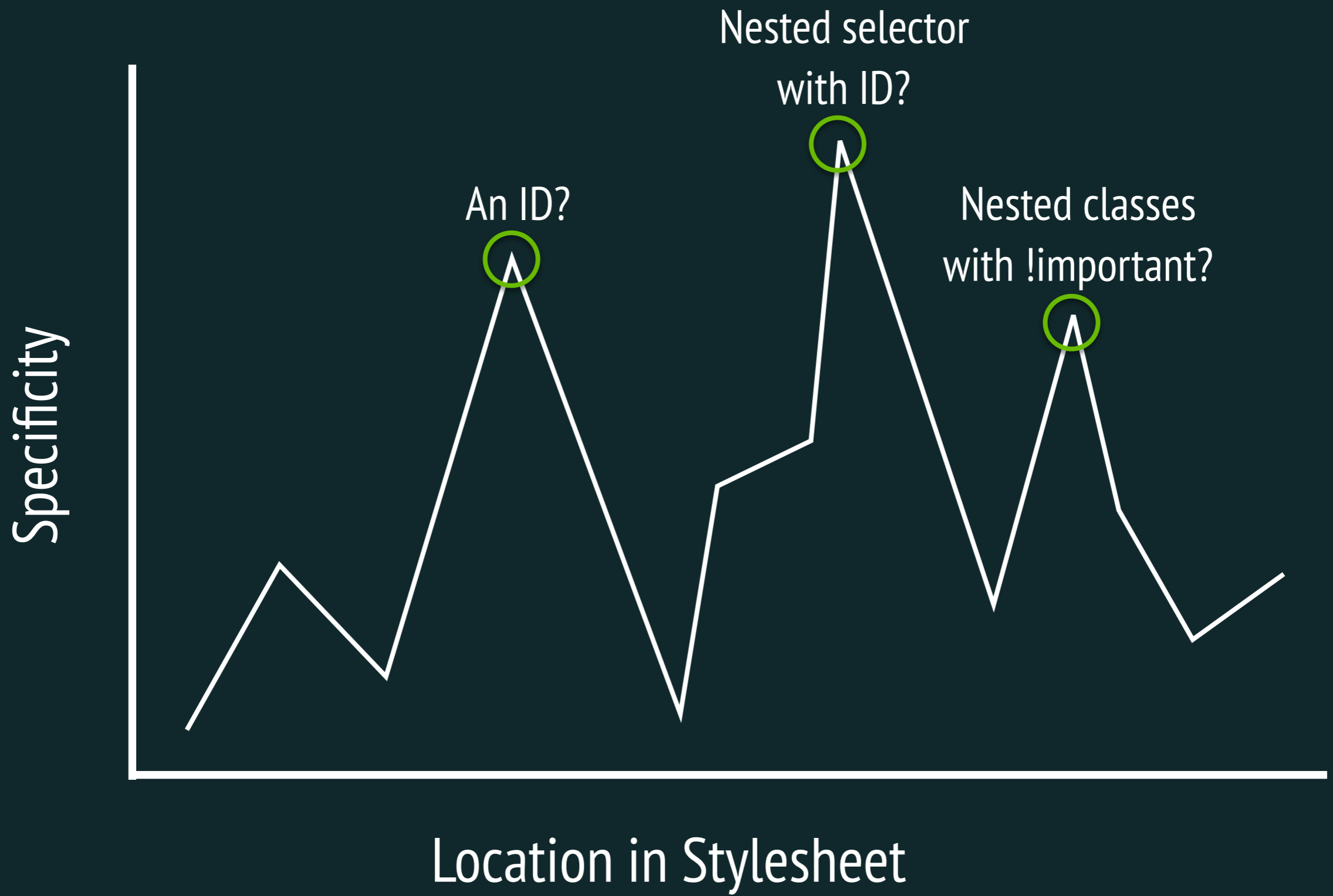
**ALL CSS IS NOT CREATED
EQUALLY**

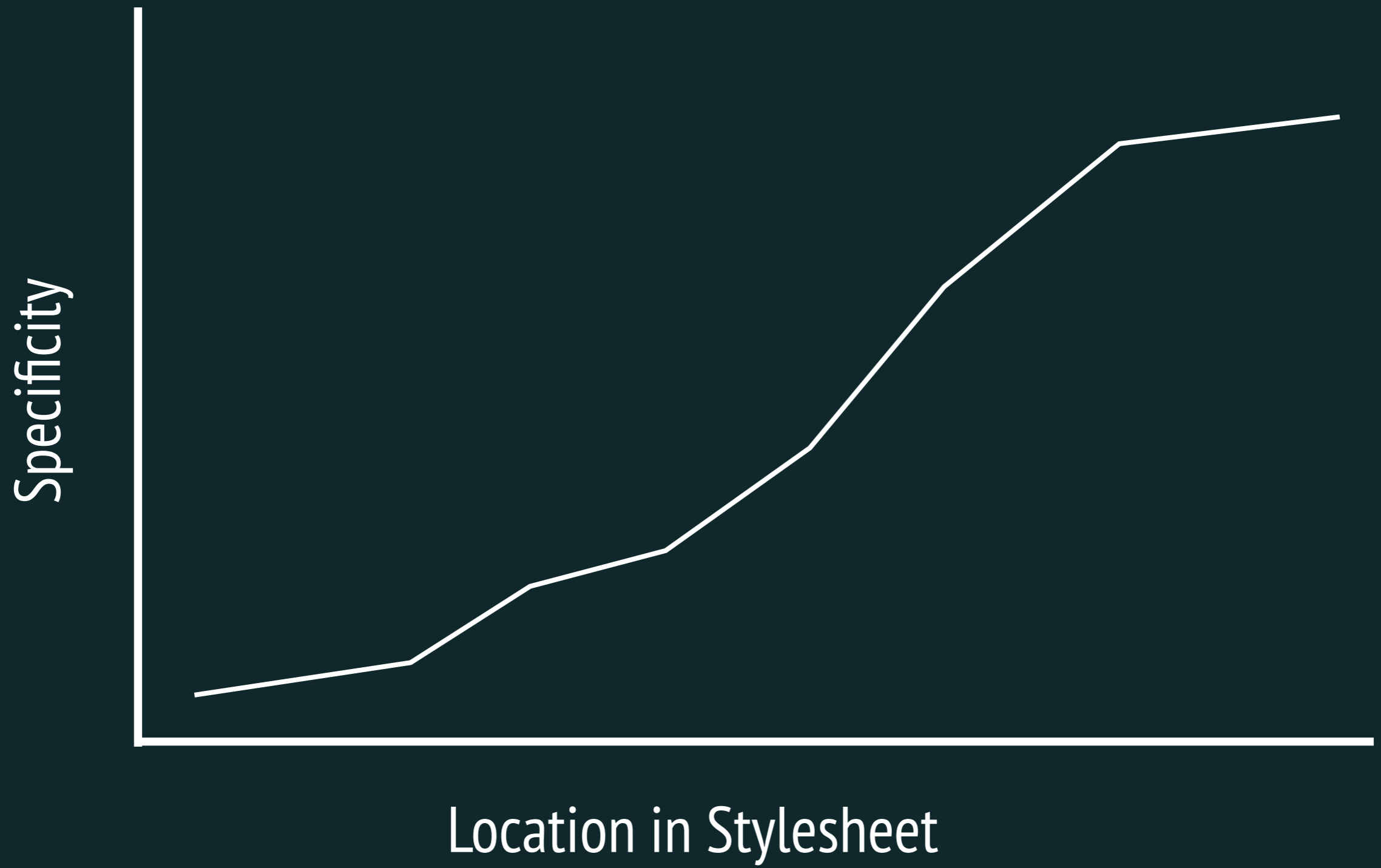
```
p {  
  color: red;  
  font-size: 1em;  
  line-height: 1.5;  
}
```

```
html body p {  
  color: red;  
  font-size: 1em;  
  line-height: 1.5;  
}
```

COMMON MISTAKES

- Defaulting to ID for selecting elements
 - `#hero` instead of `.hero`
- Selectors that are more than 3 layers deep
 - `section #hero ul li a.current-page { ... }`
- `!important`
- `a:link`





MODULAR ARCHITECTURE

**ARCHITECTUR
E**

**GSS MODULAR
ARCHITECTURE**

**PRE-
PROCESSOR**

NAMING CONVENTION



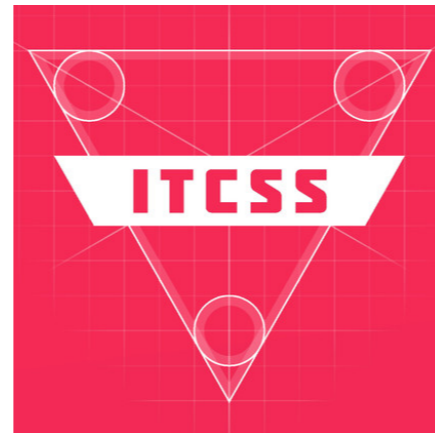


MY WORKFLOW

PRE-PROCESSOR

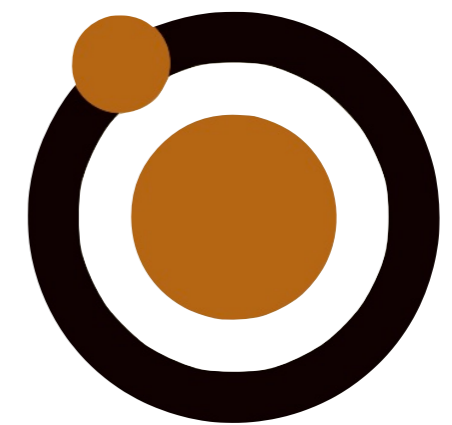
Sass

ARCHITECTURE



NAMING CONVENTION

BEM



ATOMIC DESIGN

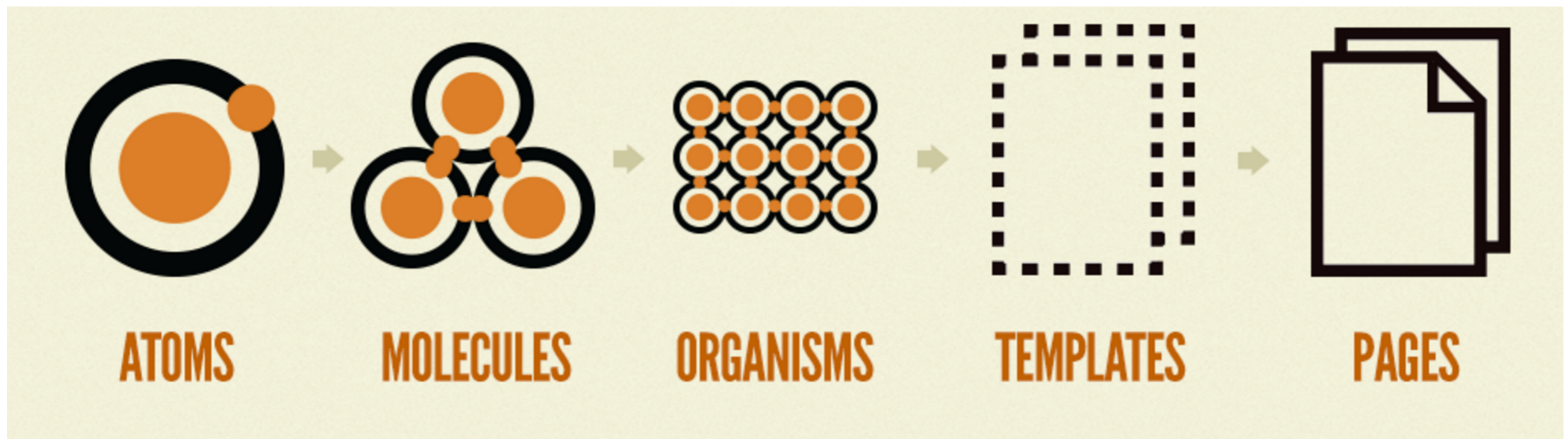
MY ARCHITECTURE

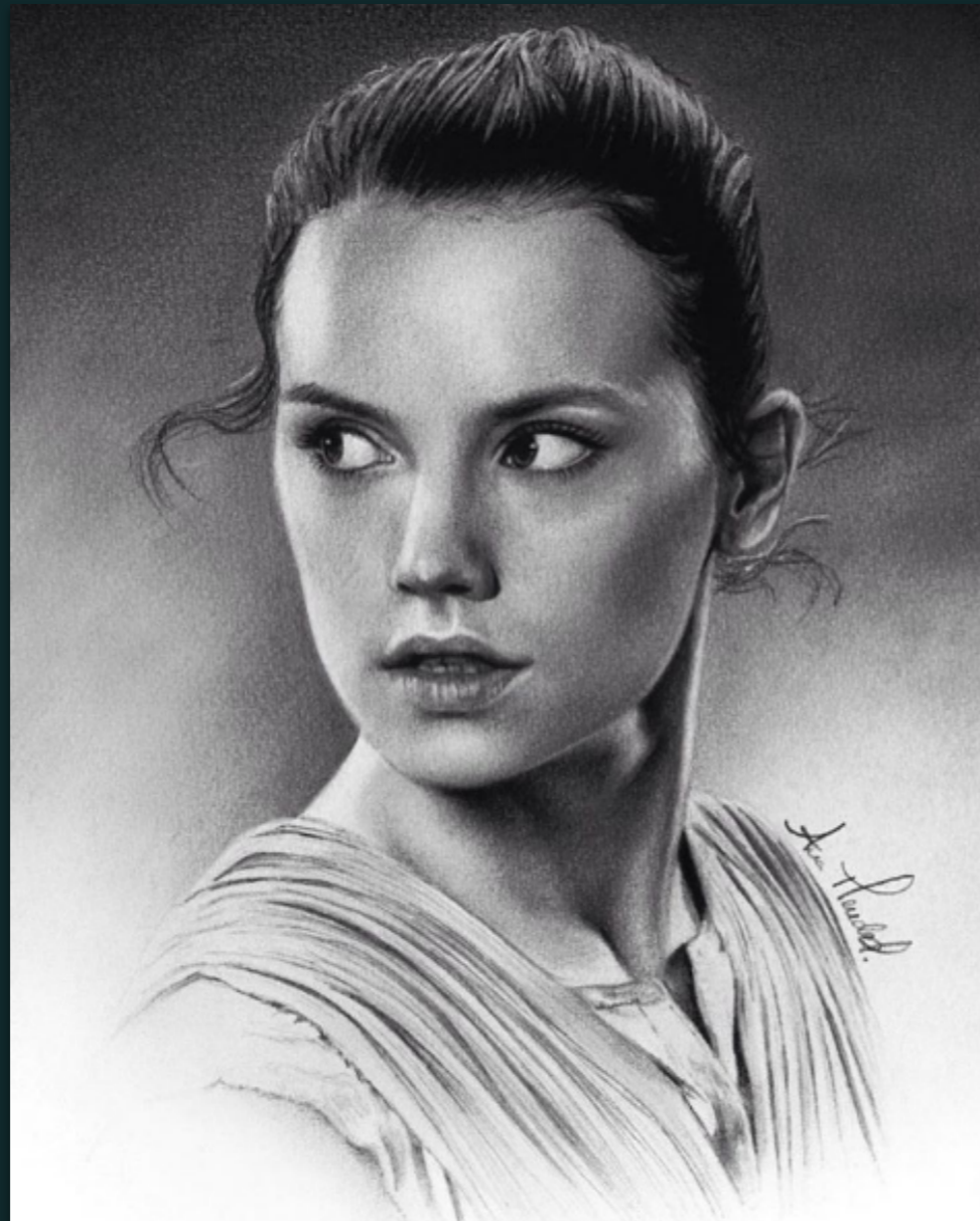
- **main.css**
 1. **settings** - configuration (i.e., colors, fonts, mixins, etc.)
 2. **vendor** - third party styles (i.e., normalize, reset, etc.)
 3. **base** - unclassed HTML elements (i.e., global, html, body, etc.)
 4. **layout** - generic page layout styles (i.e., grid, spacing, etc.)
 5. **components** - UI modules of the site (i.e., buttons, carousels, etc.)
 6. **pages** - custom page layouts (i.e., short marketing campaign)
 7. **overrides** - takes priority over everything else (i.e., 508, etc.)

NAMING CONVENTION

BLOCK ELEMENT MODIFIER

`.hero__button--primary`





angel4mckay

FOLLOW

[Person #1](#), [Person #2](#) and [Person #3](#) like this 3h

Person #3: Nice job!

#starwars #jedi #theforce
#maythe4thbewithyou #art #drawing
#fanart

Add a comment...

Problem: <http://bit.ly/1TKP5FC>

Solution: <http://bit.ly/10eW0aP>

FINAL THOUGHTS

GENERAL RULES

1. Always choose classes over IDs for selectors*
2. Avoid using general element selectors (i.e., h1, h2, p, span, etc.)*
3. If you have more than 3 selectors in a declaration, you're probably doing something wrong*
4. If your BEM class name is more than one level deep, you need to break it up into something small.
5. In the event you have to use `!important` or a very specific selector, add a quick comment so that future developers can make an informed decision.

THINGS TO REMEMBER

1. You won't get it right the first time.
2. "Simpler is better" when writing good CSS
3. Discipline to write beautiful CSS is one of the hardest parts
4. Remember to pick and choose what works best for your team and projects.
5. This philosophy is just as much about improving code quality as it is for keeping your mental sanity

TOOLS INVENTORY

- **Pre-Processors**

- Sass (SCSS) - <http://sass-lang.com/>
- Stylus - <http://stylus-lang.com/>

- **Architecture**

- ITCSS - <http://itcss.io/>
- SMACSS - <https://smacss.com/>

- **Naming Conventions**

- BEM - <http://getbem.com/>
- Atomic Design - <http://atomicdesign.bradfrost.com/>
- OOCSS - <https://www.smashingmagazine.com/2011/12/an-introduction-to-object-oriented-css-oocss/>

ADDITIONAL RESOURCES

- **Managing CSS Projects with ITCSS** by Harry Roberts
<https://speakerdeck.com/dafed/managing-css-projects-with-itcss>
- **Atomic OOBEMITSCSS** by Una Kravetz
<http://www.sitepoint.com/atomic-oobemitscss/>
- **An Introduction to BEM**
<http://getbem.com/introduction/>
- **Installing Sass**
<http://sass-lang.com/install>

Q&A

THANK YOU!

If you have any additional questions, feel free to reach out to me @bencodezen!