

# Modular css

Scott Vandehey — @spaceninja — Devsigner 2016



FridayFrontEnd.com


Front-end development links **tweeted** daily, **emailed** weekly.

@fridayfrontend

Assembled with love by **Scott Vandehey**

CSS at Scale  
is Difficult



A man with short brown hair and a beard, wearing a red sweater, stands behind a podium. He is gesturing with his hands while speaking. The podium is dark with a pattern of small circles and features a large white '22C' and a yellow circle with the word 'tob'. A laptop on the podium has several stickers, including GitHub, Docker, and a yellow circle with 'tob'. The background is dark with a screen showing blue abstract shapes.

“Replace ‘can you build this?’ with  
‘can you **maintain** this without  
losing your minds?’”

—Nicolas Gallagher, @necolas



# Difficult to Understand

```
<div class="box profile pro-user">  
  <img class="avatar image" />  
  <p class="bio">...</p>  
</div>
```

# Difficult to Reuse

- You want to re-use a style from another page, but it's written in a way that only works on that page
- You don't want to break the original, so you duplicate the code
- Now you have two problems

# Difficult to Maintain

- You change the markup and the whole thing breaks
- You want to change a style on one page and it breaks on another
- You try to override the other page, but get caught in a specificity war





**Thomas Fuchs**

@thomasfuchs

Two CSS properties walk into a bar.

A table in a bar across town collapses.

# Modularity

What does that even mean?

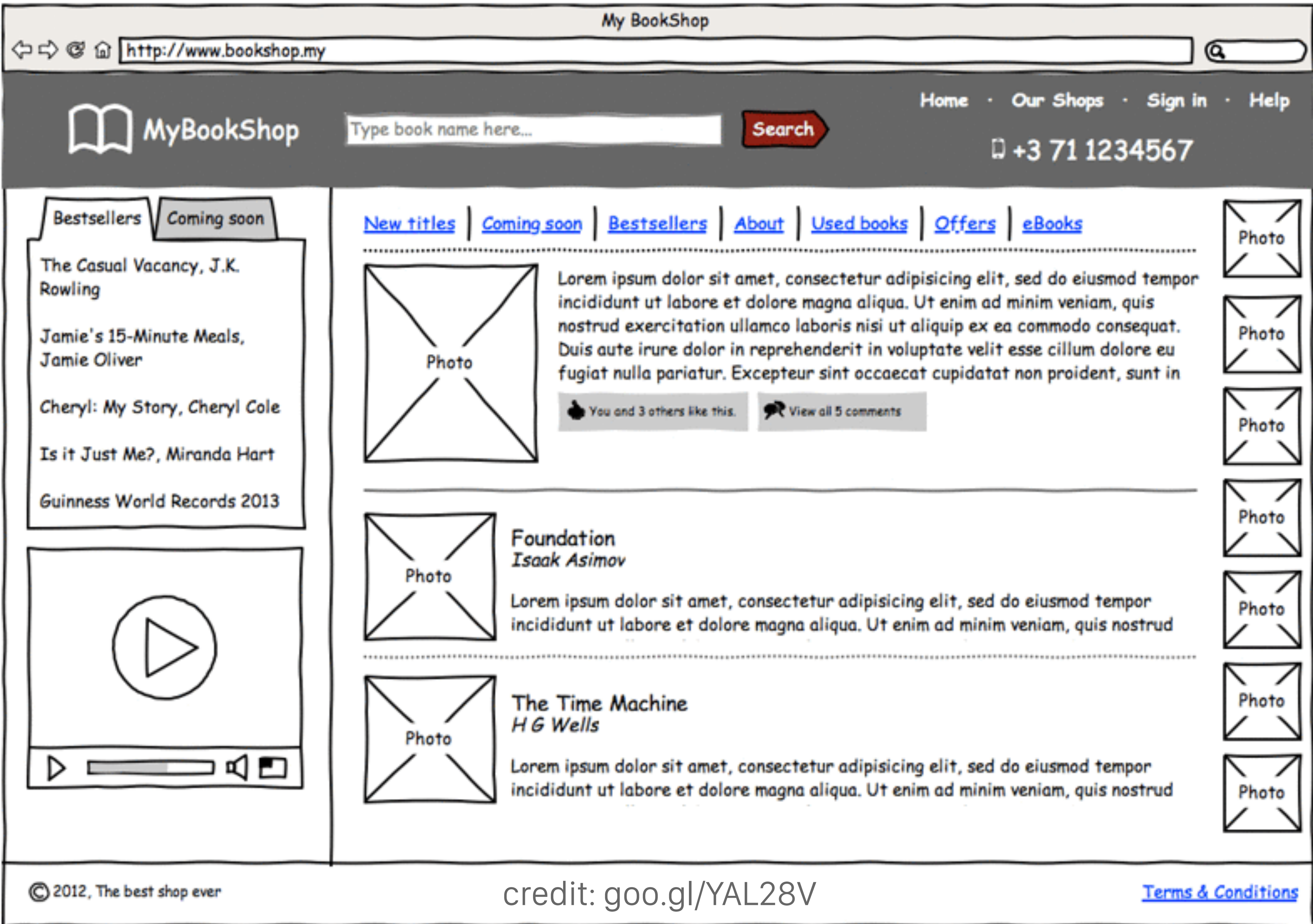




"Code which adheres to the **separation of concerns** can be much more confidently modified, edited, extended, and maintained because we know how far its responsibilities reach. We know that modifying layout, for example, will only ever modify layout—nothing else."

—Harry Roberts, @csswizardry









MyBookShop

Type book name here...

Search

[Home](#) · [Our Shops](#) · [Sign in](#) · [Help](#)

+3 71 1234567

Bestsellers

Coming soon

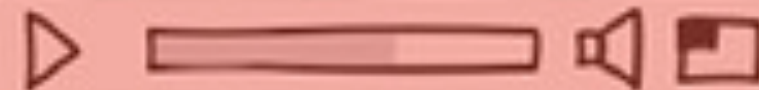
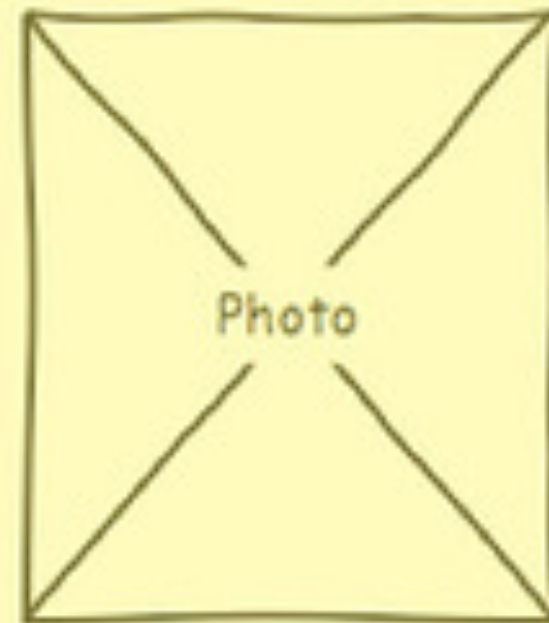
The Casual Vacancy, J.K. Rowling

Jamie's 15-Minute Meals, Jamie Oliver

Cheryl: My Story, Cheryl Cole

Is it Just Me?, Miranda Hart

Guinness World Records 2013

[New titles](#)[Coming soon](#)[Bestsellers](#)[About](#)[Used books](#)[Offers](#)[eBooks](#)

Photo

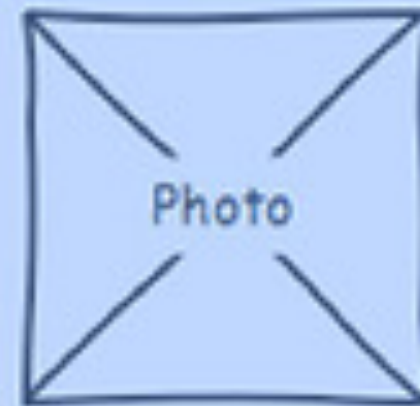
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in



You and 3 others like this.



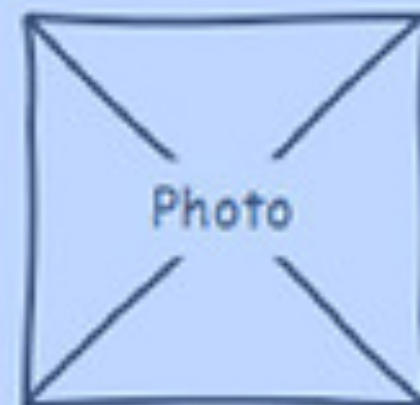
View all 5 comments



Photo

Foundation  
Isaak Asimov

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud



Photo

The Time Machine  
H G Wells

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud

Photo

Photo

Photo

Photo

Photo

Photo

Photo



# Remind you of anything?



credit: Alan Chia, [goo.gl/KwJ17v](https://goo.gl/KwJ17v)





credit: [goo.gl/NOIjdw](https://goo.gl/NOIjdw)





# Media Block

## Smaller Subheading

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna

# Media Block Right

## Smaller Subheading

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna



"The Media Object Saves Hundreds of Lines of Code," [goo.gl/3bi2fS](https://goo.gl/3bi2fS)





Nicole Sullivan  
[View My Profile](#)

## News Feed

Messages (6)

Events (2)

Photos

Friends (17)

Applications

Games

Groups (4)

TripAdvisor – Cities I've Visited™ →

Upcoming →

[More](#)[Chat with Friends](#)[Go Online](#)

## News Feed

Top News · Most Recent 34

Requests

[See All](#)

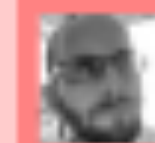
What's on your mind?



Jeffrey Zeldman The Big Web Show is coming! <http://bit.ly/9Yi5Fs>

 9 hours ago via Twitter · [Comment](#) · [Like](#)

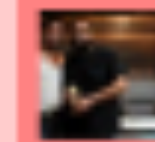
6 people like this.



Alec Pollak Finally! I've been wondering when you were going to leap into making amazing video content...  
9 hours ago



Jeffrey Zeldman Aw shucks. Thanks, Alec.  
8 hours ago



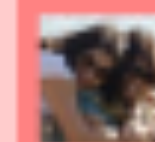
Rami Daud awesome! Can't wait!  
4 hours ago



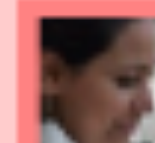
Neha Heera I'm hungry :'(

11 hours ago · [Comment](#) · [Like](#)

View all 4 comments



Neha Heera i had my health assessment – so was fasting for 12 hours :(  
10 hours ago



Richa Priyanka oh.hope thts done now :)  
10 hours ago



Abigail Ipredictariot Ksn watching the peace of a Lion waiting to devour its prey.

17 hours ago · [Comment](#) · [Like](#)

17 friend requests

2 event invitations

4 group invitations

9 other requests

Suggestions

[See All](#)

Fisherman's Friend Tony Bray  
30 mutual friends  
 Add as friend



Dave Raggett  
Catch up on Facebook.  
 Send him a message

Sponsored

[Create an Ad](#)

Feel The Energy Boost



The energy drink for your feet. Take a shot and discover the better way to train. Reebok ZigTech. More train, less pain.

Dwayne Robinson is a fan of Reebok Men.

Become a Fan

Events

[See All](#)

David Donovan's birthday Today

Lalit Kumar's birthday Today

Axel Corjon's birthday Thursday

Oli Studholme's birthday Thursday

Thomas Vergnaud's birthday Friday

Chat (Offline)

Pokes



# oocss

Object Oriented CSS

# OOCSS

- Created by Nicole Sullivan in 2009 based on her work at Yahoo
- Key concept: Objects are **reusable patterns** whose visual appearance is not determined by context





“a CSS ‘object’ is a repeating visual **pattern**, that can be abstracted into an independent snippet of HTML, CSS, and possibly JavaScript. That object can then be **reused** throughout a site.”

—Nicole Sullivan, @stubbornella

credit: John Morrison, [goo.gl/AZBz7y](https://goo.gl/AZBz7y)



# OOCSS: Context

- An object should **look the same** no matter where you put it
- Objects should not be styled based on their context



# OOCSS: Skins

- Abstract the **structure of an object** from the **skin that is being applied**
- Create reusable classes for common visual styles like drop shadows

# OOCSS: Use Classes

- Use classes to name your objects and their components so markup can change without impacting style
- eg, `.site-nav` *not* `header ul`

# OOCSS: No IDs

- They mess up specificity because they are too strong
- They are unique identifiers, which means **components built with them are not reusable** on the same page




# BEM

Block, Element, Modifier

# BEM

- Created in 2009 by Russian internet company Yandex who faced similar problems to Yahoo using CSS at scale
- Key concept: **Blocks** (objects) are made of smaller **elements** and can be **modified** (skinned)



A woman with short, wavy brown hair is shown in profile, facing right. She is wearing a light-colored, textured cardigan over a white top. She appears to be speaking at a podium, with a microphone visible in the foreground. The background is a plain, light-colored wall.

"BEM is a way to **modularize** development of web pages. By breaking your web interface into components... you can have your interface divided into **independent parts**, each one with its own development cycle."

—Varya Stepanova, @varya\_en



# BEM: Blocks

- Logically & functionally independent components
- **Nestable:** Blocks can be nested inside other blocks
- **Repeatable:** An interface can contain multiple instances of the same block



# BEM: Elements

- A constituent part of a block that can't be used outside of it
- For example, a menu item is not used outside the context of a menu block



# BEM: Modifiers

- Defines the appearance and behavior of a block or an element
- For instance, the appearance of the menu block may change depending on a modifier that is used on it

.minifig





.minifig

.minifig\_\_head

.minifig\_\_headgear

.minifig\_\_backpack

.minifig\_\_torso

.minifig\_\_legs



`.minifig--red`

`.minifig__head`

`.minifig__headgear`

`.minifig__backpack`

`.minifig__torso`

`.minifig__legs`





.minifig--yellow-new

.minifig\_\_head

.minifig\_\_headgear

.minifig\_\_backpack

.minifig\_\_torso

.minifig\_\_legs



`.minifig--batman`

`.minifig__head`

`.minifig__headgear`

`.minifig__backpack`

`.minifig__torso`

`.minifig__legs`





# BEM: Naming

`.block-name__elem-name--mod-name`

- Names are written in lower case
- Words within names are separated by hyphens (-)
- Elements are delimited by double underscores (\_\_)
- Modifiers are delimited by double hyphens (--)

# BEM: Example

```
<a class="btn btn--big btn--orange" href="#">  
  <span class="btn__price">$9.99</span>  
  <span class="btn__text">Subscribe</span>  
</a>
```



# BEM: No Nested CSS

- Nested selectors increase specificity, making code reuse more difficult.
- Really only appropriate for styling elements based on the state of a block or its modifier.

# SMAcss

Scalable & Modular Architecture for CSS  
(pronounced "smacks")



# SMACSS

- Created by Jonathan Snook in 2011. He had experience writing CSS at scale, including Yahoo Mail
- Key concept: Different **categories** of objects need to be handled differently





"At the very core of SMACSS is **categorization**. By categorizing CSS rules, we begin to see patterns and can define better practices around each of these patterns."

—Jonathan Snook, @snookca

credit: elidr, [goo.gl/Te8zQl](https://goo.gl/Te8zQl)



# SMACSS: Categories

1. **Base** rules are default styles for things like links, paragraphs, and headlines
2. **Layout** rules divide the page into sections, hold one or more modules together
3. **Modules** are the reusable, modular parts of a design. Callouts, sidebar sections, product lists, etc.
4. **State** rules describe how modules or layouts look in a particular state. Hidden, expanded, active, etc.

# SMACSS: Naming

- Use prefixes to differentiate between different types of rules:
  - l- for layout rules
  - m- for module rules
  - is- for state rules



OOCSS  
& BEM  
& SMACSS  
= Modular

# Modular CSS

- These methodologies are more alike than different
- Their evolution represents our industry's growing experience with CSS at scale
- We don't have to limit ourselves. Look at what they share and keep the best parts



# Modular Elements

- **Module:** a reusable pattern  
(aka Object, Block)
- **Child Element:** discrete piece of the module that can't stand alone
- **Module Modifier:** alters the visual appearance of a module

# Modular Categories

1. **Base** rules are default styles for HTML elements
2. **Layout** rules control how modules are laid out, but not visual appearance: `.l-centered`
3. **Modules** are visual styles for reusable, self-contained UI components: `.m-profile`
4. **State** rules are added by JavaScript: `.is-hidden`
5. **Helper** rules are small in scope and unconnected to modules: `.h-uppercase`



# Modular Rules

- Don't use IDs
- Don't nest CSS deeper than one level
- Add classes to child elements so you're not tied to specific markup
- Prefix class names so you can tell at a glance what a class does

FAQ



# So many classes!

- Having lots of classes might look ugly, but it doesn't hurt performance
- Carefully scoped classes help others combine your lego blocks in new ways
- Don't be afraid of long class names. They're self-documenting!

# Grandchild classes?

.minifig

.minifig\_\_arm

~~.minifig\_\_arm\_\_hand~~

.minifig\_\_hand



# Module conflicts?

- Modules *shouldn't* overlap much
- You *should* be able to load modules in any order
- Consider an `!important` helper class



**Trek Glowacki**

@trek

Your daily reminders that  
components aren't about reuse,  
they're about isolation.

Reuse is a useful emergent  
property of isolation.



# Flexbox modules?

- It's tricky to make layout modules using flexbox
- If you're not careful, you'll find yourself making modifiers for every single flex option
- `-\(ツ)\-/`

# Preprocessors?

- Modular CSS is more of a philosophy than a framework
- As a result, it works with any preprocessor (or not) that you need



# Bootstrap?

- Bootstrap is a pattern library, not a methodology
- That said, it's built in a modular way:
- `.btn .btn-primary .btn-sm`

# Recap

Modular CSS is dope



# Remember This?

```
<div class="box profile pro-user">  
  <img class="avatar image" />  
  <p class="bio">...</p>  
</div>
```

# Self-Documenting

```
<div class="box profile profile--is-pro-user">  
  <img class="avatar profile__image" />  
  <p class="profile__bio">...</p>  
</div>
```



# Modular Benefits

- Simplifies code and facilitates refactoring
- Self-documenting code
- Reusable code that doesn't influence outside its own scope
- Naturally leads to a pattern library

# Modular Benefits

- Predictable
- Maintainable
- Performant





**Thomas Fuchs**

@thomasfuchs

Two CSS properties walk into a bar.

~~A table in a bar across town collapses.~~

Everything is fine, thanks to modular code and proper namespacing.

# Thanks!

slides: [goo.gl/sp9wRS](https://goo.gl/sp9wRS)

Scott Vandehey — @spaceninja — Devsigner 2016