ITCSS

Hello, Serbia!

Managing CSS Projects with ITCSS

DaFED – Novi Sad, November 2014

#itcss

Harry Roberts

Consultant Front-end **Architect**.

Products, long-running projects, large teams, **big codebases**.

@csswizardry

What is ITCSS?

Inverted Triangle architecture for **CSS**.

A sane, scalable, **managed architecture**.

A **school-of-thought**, not a library.

A **meta framework**; a framework for frameworks.

Incredibly **simple**.

Problems with CSS at scale

CSS’ fault vs. our fault

CSS’ fault

The cascade and **inheritance**.

Very **loose**.

Highly dependent on **source order**.

Not very **expressive**.

Lots of **gotchas**.

Specificity.

Our fault

Lack of **documentation**.

Lack of structure, **quality assurance**.

Mixture of **abilities**.

Lack of **knowledge** (about CSS or **the project itself**).

Different styles, preferences, **ways of working**.

Not looking to see/being aware of **what exists already**.

Adding new styles to the **end of stylesheets**.

Inheritance, the cascade, and source order

Each piece of CSS needs a knowledge of what came before it and what might come after it – a.k.a. **dependencies**.

About me – CSS Wizardry
csswizardry.com/about/

The CSS is written in **Sass**, and built on top of **inuitcss**. inuitcss is a free, open-source, Sass-based, OOCSS framework with a strong focus on performance, scalability, and a small footprint. This is then built into **ITCSS**, a proprietary CSS architecture of mine which is designed to aid the growth and maintainability of CSS projects.

Elements Network Sources Timeline Profiles Resources Audits Console
Styles Computed Event Listeners DOM Breakpoints Properties

```
<!DOCTYPE html>
▼<html lang="en" class="page  page--about">
  ▶<head>…</head>
  ▼<body data-twttr-rendered="true">
    ▶<header class="page-head">…</header>
    ▼<div class="wrapper">
      ▼<div class="layout">
        ▼<section class="layout__item  desk-three-fifths" data-ui-component="Main
        content">
            <h1>About me</h1>
          ▶<p>…</p>
          ▶<figure>…</figure>
          ▶<p>…</p>
          ▶<p>…</p>
          ▶<blockquote class="pull-quote">…</blockquote>
          ▶<p>…</p>
          ▶<p>…</p>
          ▶<p>…</p>
            <hr>
            <h2 id="colophon">Colophon</h2>
          ▶<p>…</p>
          ▶<p>…</p>
          ▶<p>…</p>
          ▶<p>…</p>
          ▶<p>…</p>
          ▶<p>…</p>
            <hr>
            <h2 id="contact">Contact</h2>
          ▶<p>…</p>
          ▶<p>…</p>
          ▶<p>…</p>
          ▶<dl>…</dl>
          </section>
        ▶<section class="layout__item  desk-three-tenths  desk-push-one-tenth"
        data-ui-component="Sub content">…</section>
        </div>
      </div>
    ▶<section class="[ band  band--tint ]  page-foot">…</section>
    ▶<footer class="page-micro">…</footer>
    ▶<script>…</script>
    ▶<script>…</script>
    </body>
</html>
```

```
h1, h2, h3, h4, h5, h6, ul, ol, dl, blockquote, p, address, table, fieldset, figure, pre, .page-head, .promo__block {    csswizardry.min.css:5
   ~~margin-bottom: 24px;~~
   margin-bottom: 1.5rem;
}
body, h1, h2, h3, h4, h5, h6, p, blockquote, pre, dl, dd, ol, ul, form, fieldset, legend, figure, table, th, td, caption, hr {    csswizardry.min.css:5
   margin:▶0;
   padding:▶0;
}
*, *:before, *:after {    csswizardry.min.css:5
   ~~-webkit-box-sizing: inherit;~~
   ~~-moz-box-sizing: inherit;~~
   box-sizing: inherit;
}
p {    user agent stylesheet
   display: block;
   -webkit-margin-before: 1em;
   -webkit-margin-after: 1em;
   -webkit-margin-start: 0px;
   -webkit-margin-end: 0px;
}
Pseudo ::before element
*, *:before, *:after {    csswizardry.min.css:5
   ~~-webkit-box-sizing: inherit;~~
   ~~-moz-box-sizing: inherit;~~
   box-sizing: inherit;
}
Pseudo ::after element
*, *:before, *:after {    csswizardry.min.css:5
   ~~-webkit-box-sizing: inherit;~~
   ~~-moz-box-sizing: inherit;~~
   box-sizing: inherit;
}
Inherited from div.layout
.layout {    csswizardry.min.css:7
   list-style:▶none;
   margin:▶0;
   padding:▶0;
   margin-left: -24px;
}
Inherited from html.page.page--about
html {    csswizardry.min.css:5
   font-family: "Open Sans", sans-serif;
   font-weight: 300;
}
html {    csswizardry.min.css:5
   font-size: 1em;
   line-height: 1.5;
   background-color: □#f9f9f9;
   color: ■#444;
   overflow-y: scroll;
   min-height: 100%;
⚠  ~~-webkit-text-size-adjust: 100%;~~
   ~~-ms-text-size-adjust: 100%;~~
   ~~-moz-osx-font-smoothing: grayscale;~~
```

html body div.wrapper div.layout section.layout__item.desk-three-fifths p
Find in Styles

CSS is **one giant dependency tree**.

We need a way to **manage this dependency** at a very low level.

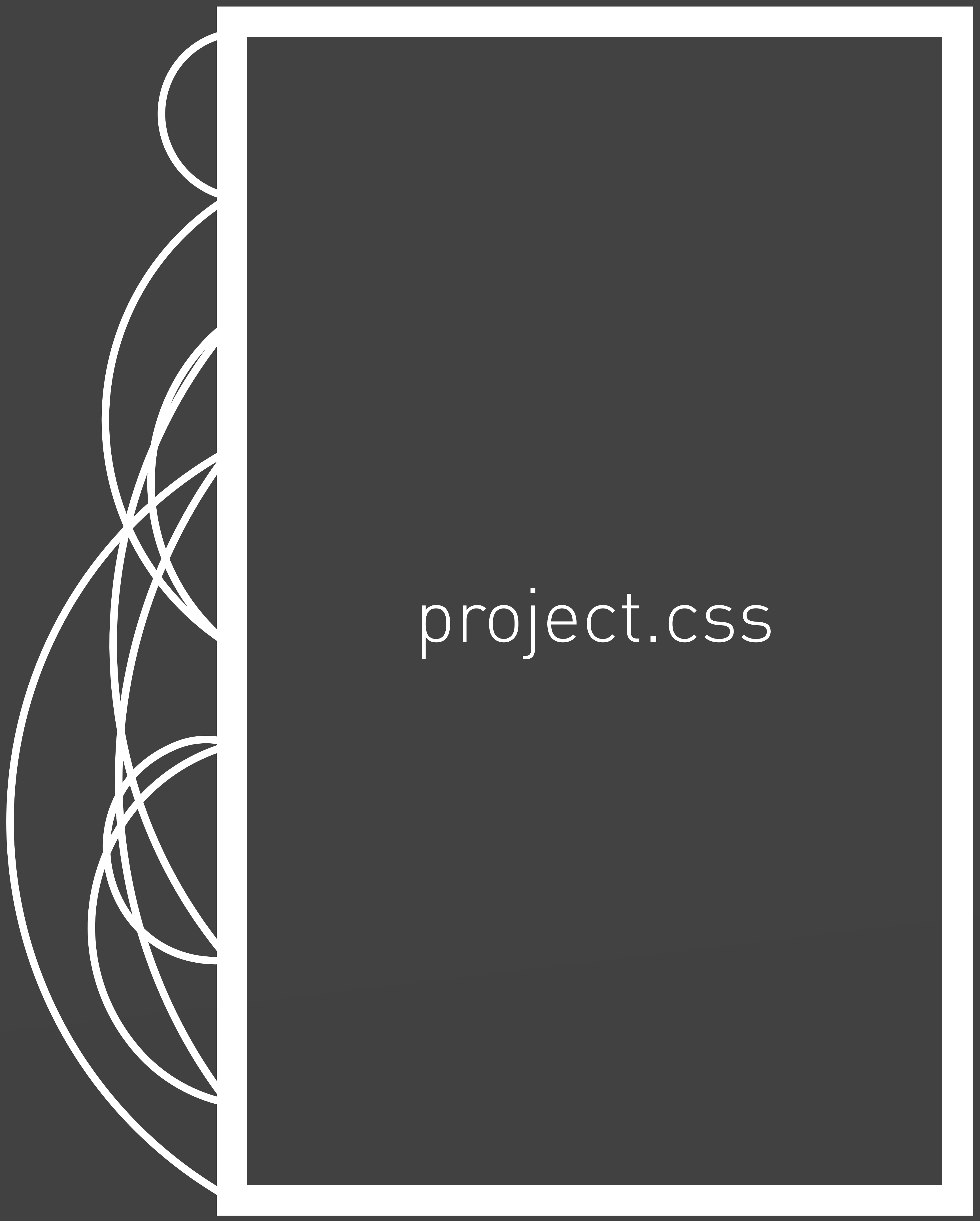
Ways of ordering stylesheets

Mirror the web page – old school!

Thematic chunks – typography, forms, buttons, etc.

Just stick it on **the end of the stylesheet**.

project.css



project.css

Undoing CSS: Writing more CSS in order to undo other CSS.

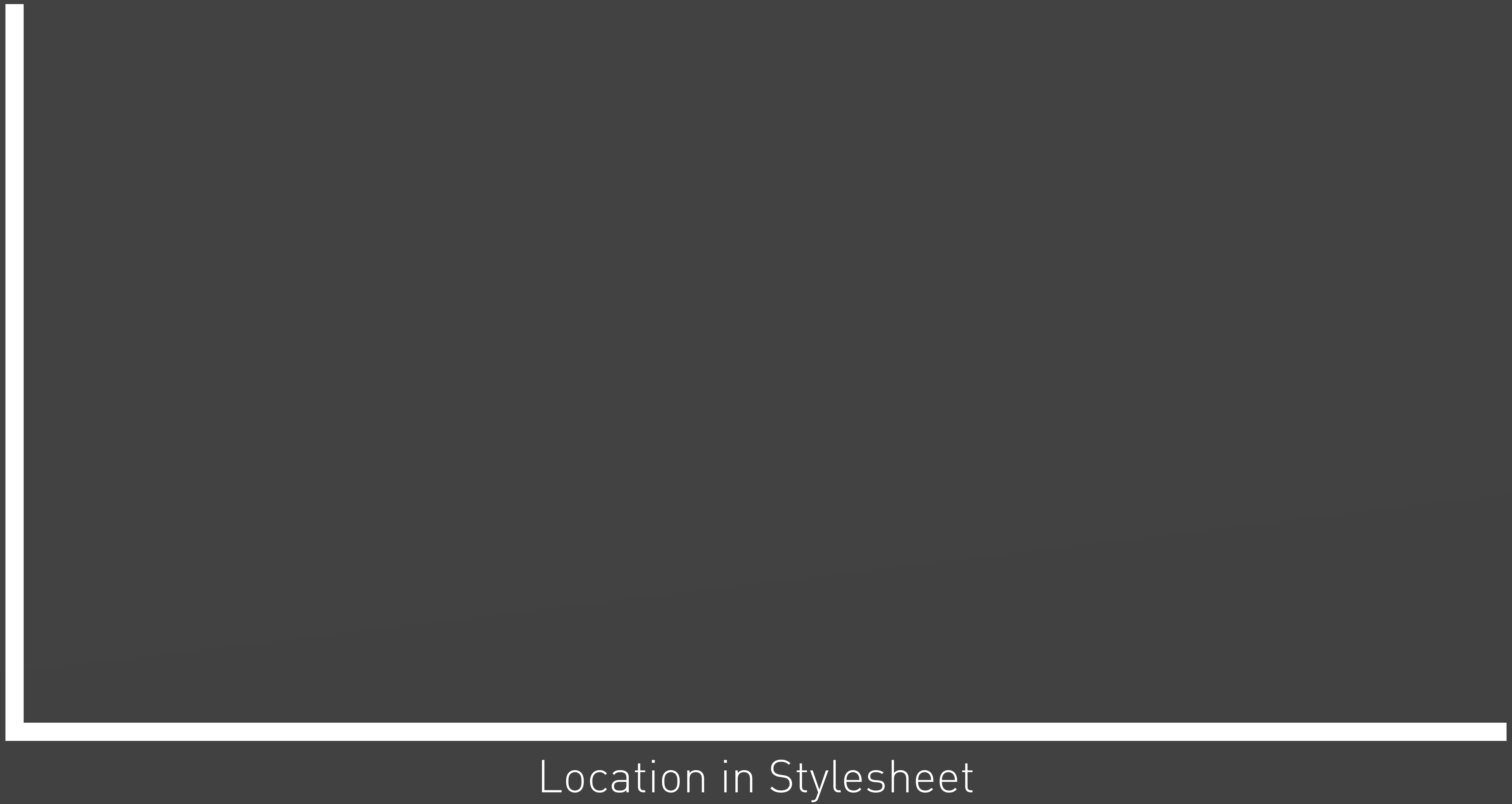
Poor source order coupled with inherited/inheriting styles can lead to **a lot of waste** and/or redundancy.

Specificity

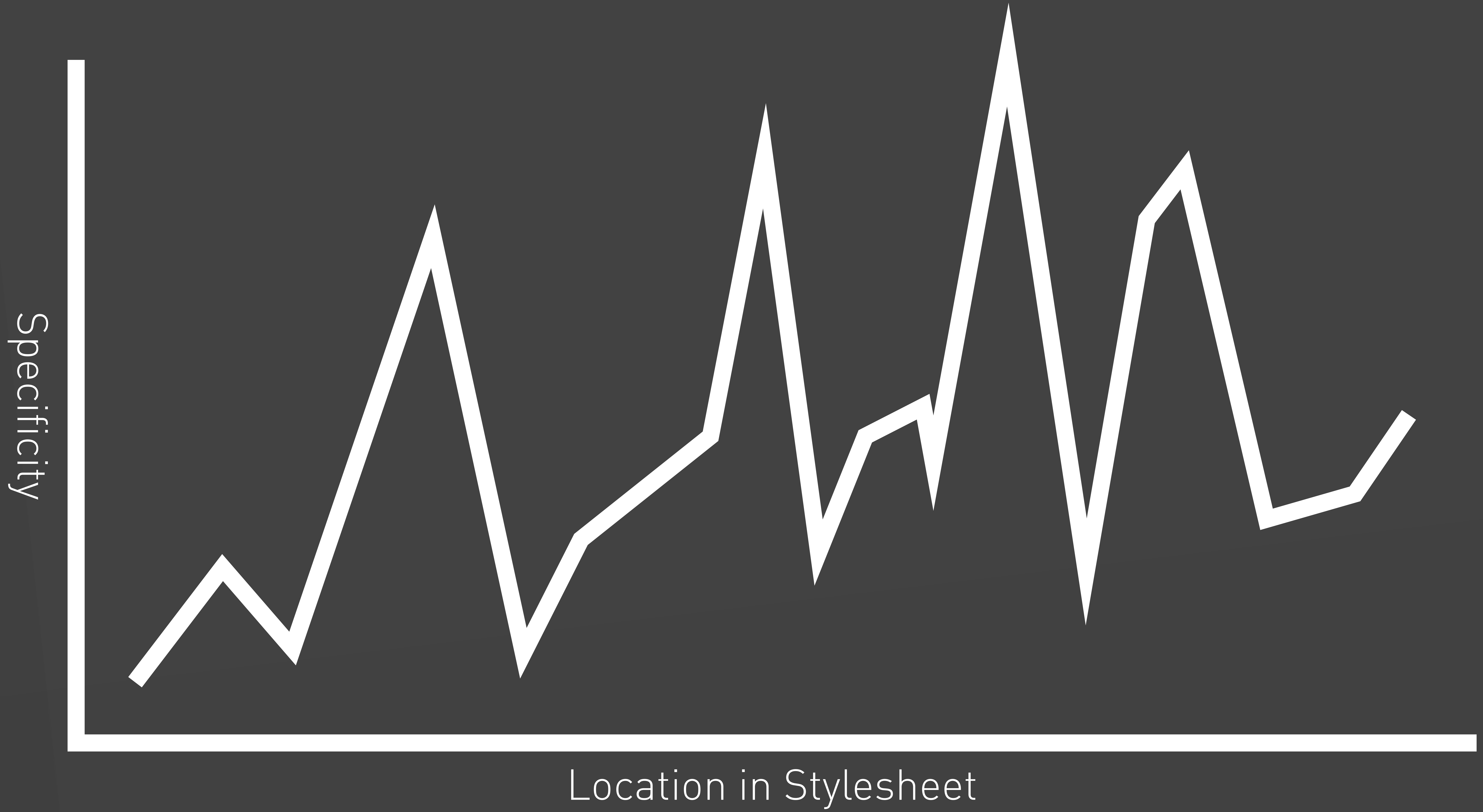
The Specificity **Wars**

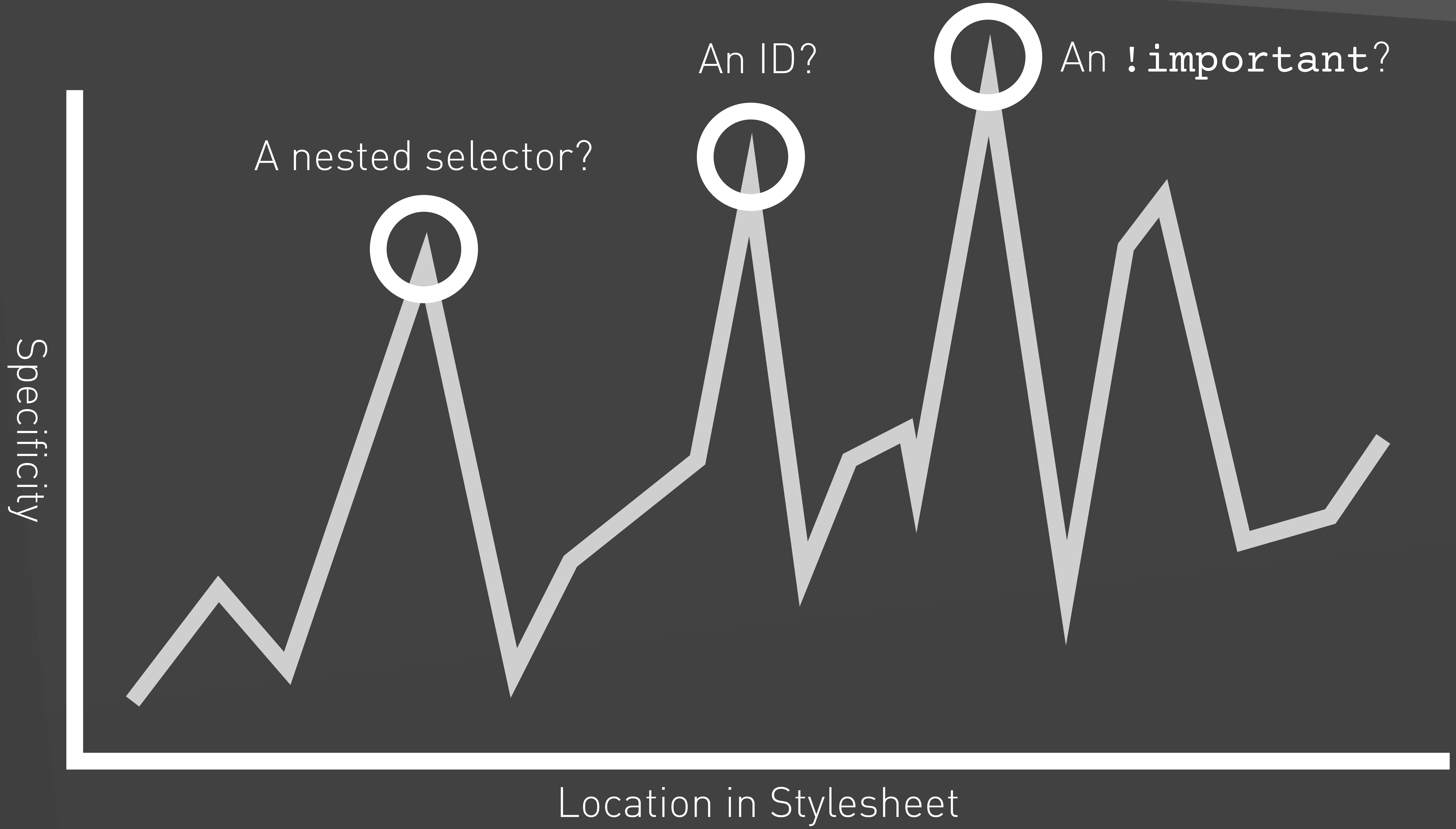
It doesn’t matter how well-considered your **source order** is; how well you’re **utilising the cascade**; what **naming conventions** you use; **specificity can undo everything**.

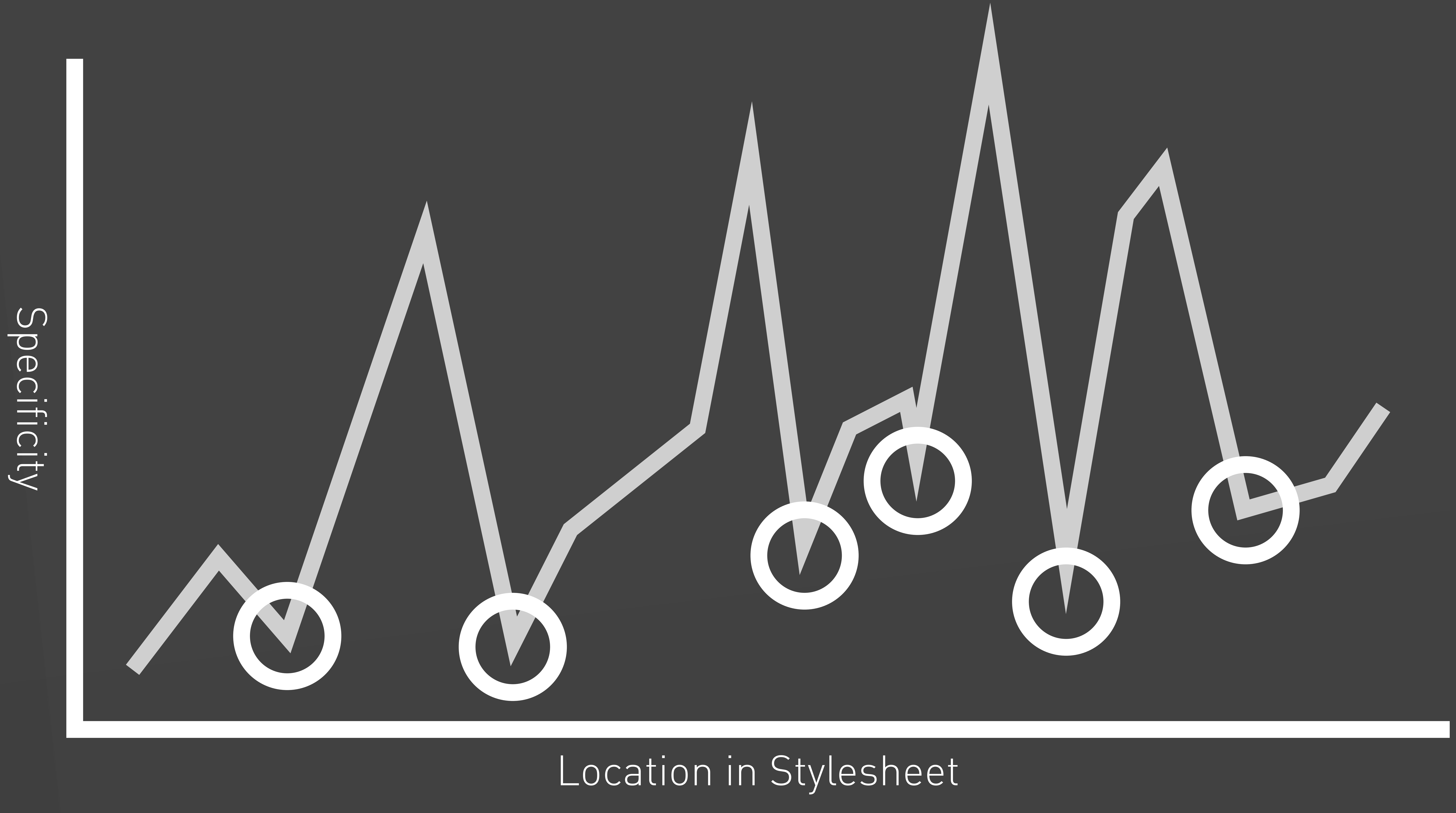
The Specificity Graph


Location in Stylesheet

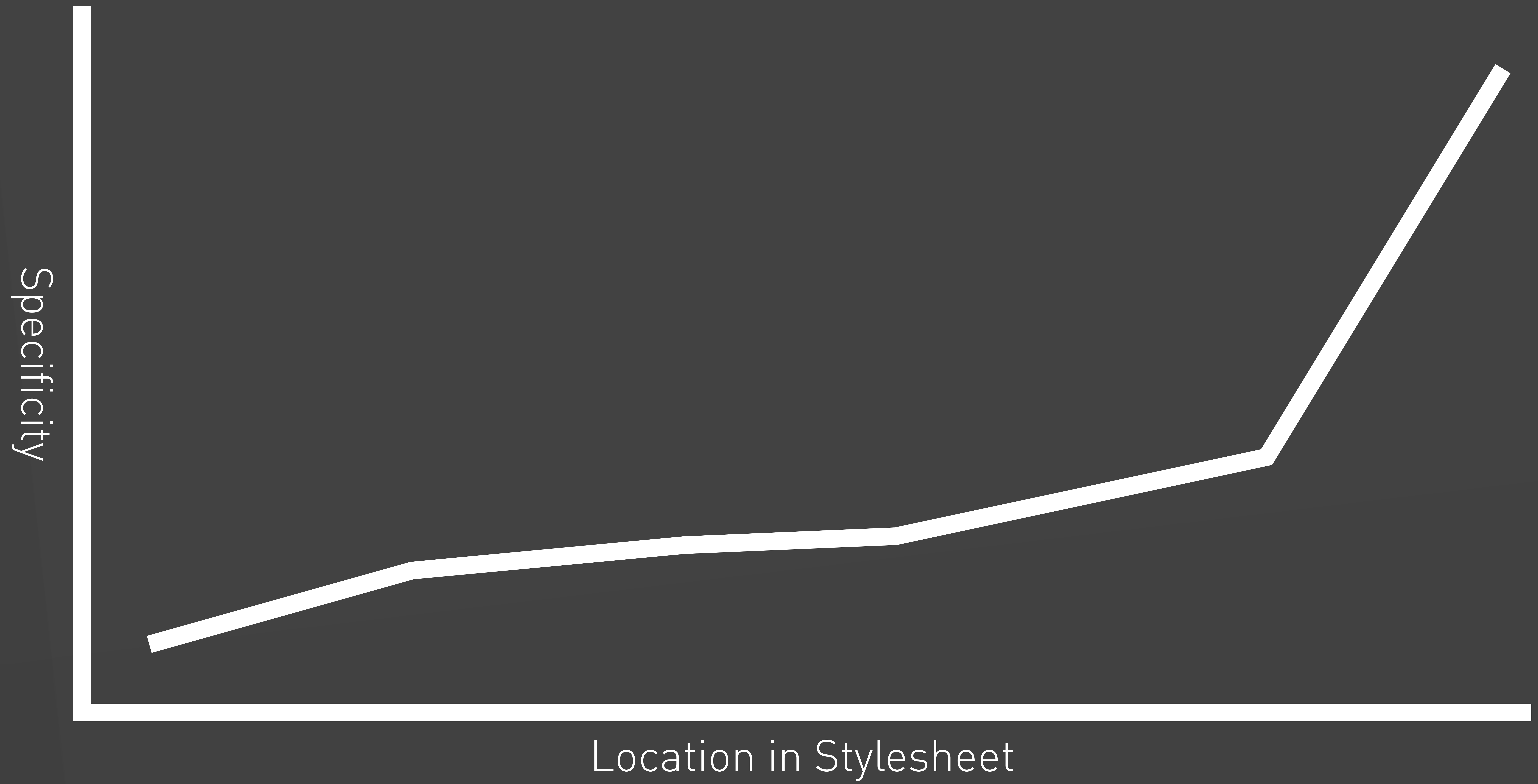
Specificity
Location in Stylesheet


Specificity
Location in Stylesheet

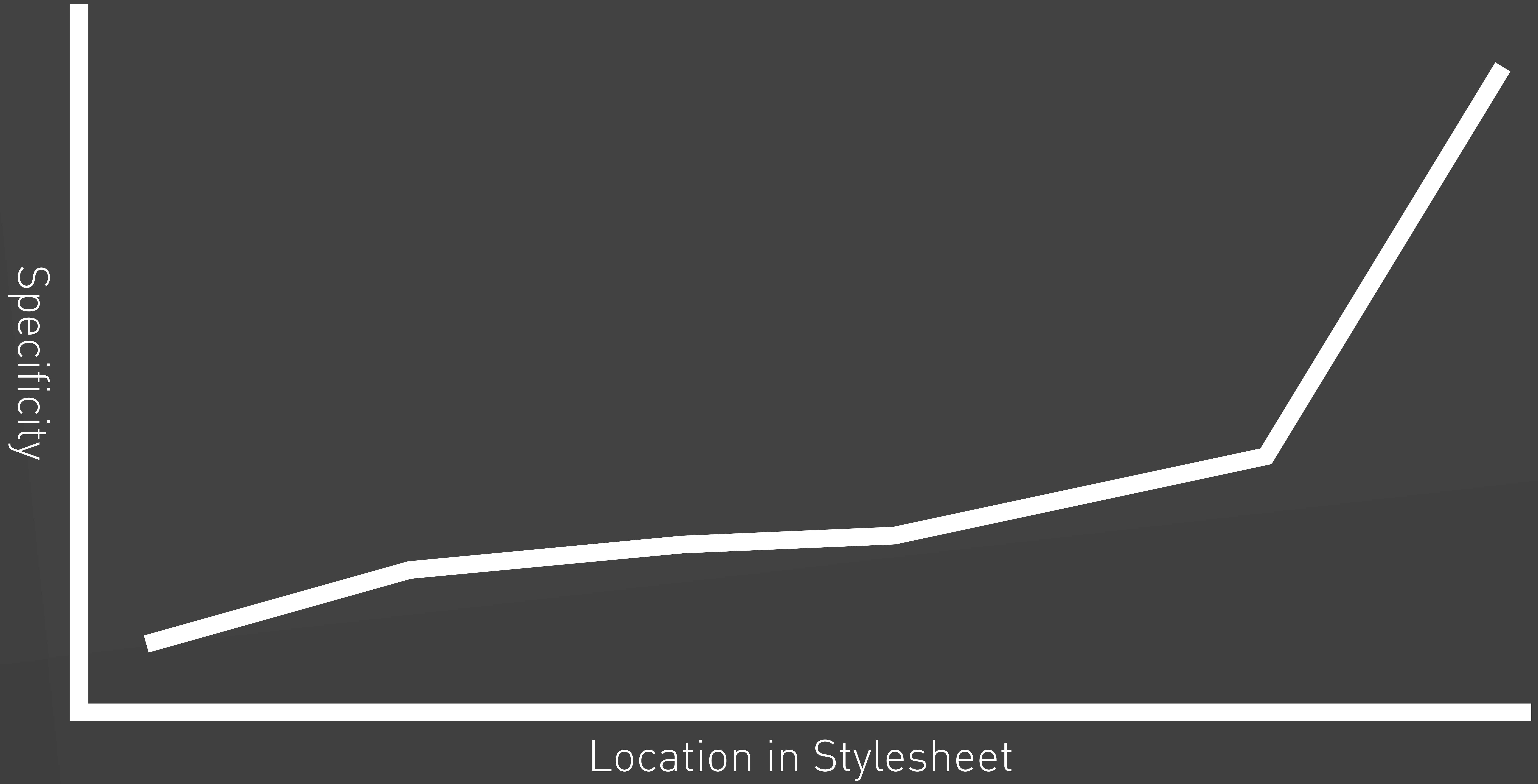

An !important?
An ID?
A nested selector?
Specificity
Location in Stylesheet

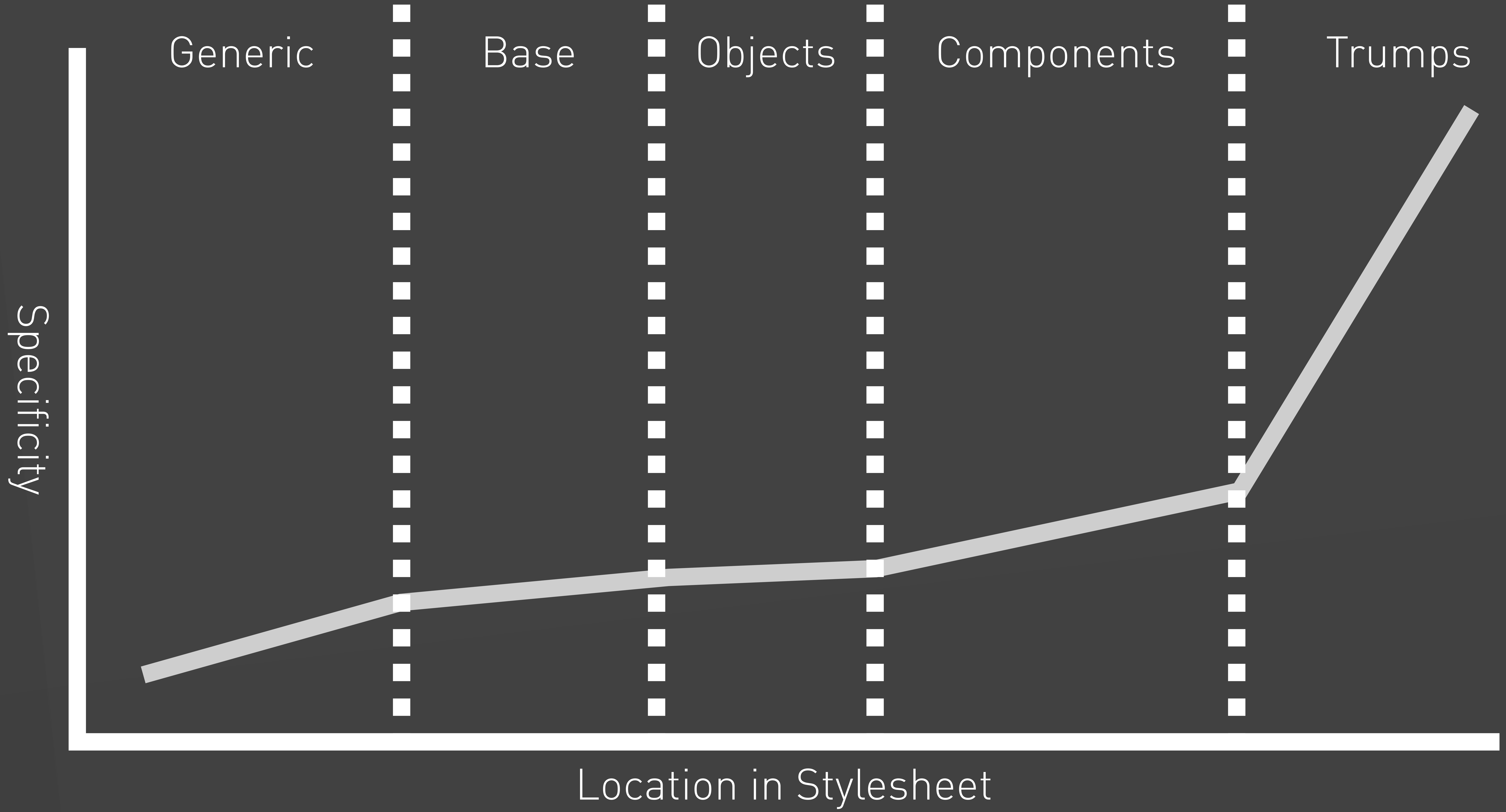

Specificity
Location in Stylesheet

How do we solve this?


Specificity
Location in Stylesheet

tl;dr: Write CSS in **specificity order**.


Specificity
Location in Stylesheet


Generic
Base
Objects
Components
Trumps
Specificity
Location in Stylesheet

These sections form **the basis of ITCSS**.

In short...

We need…

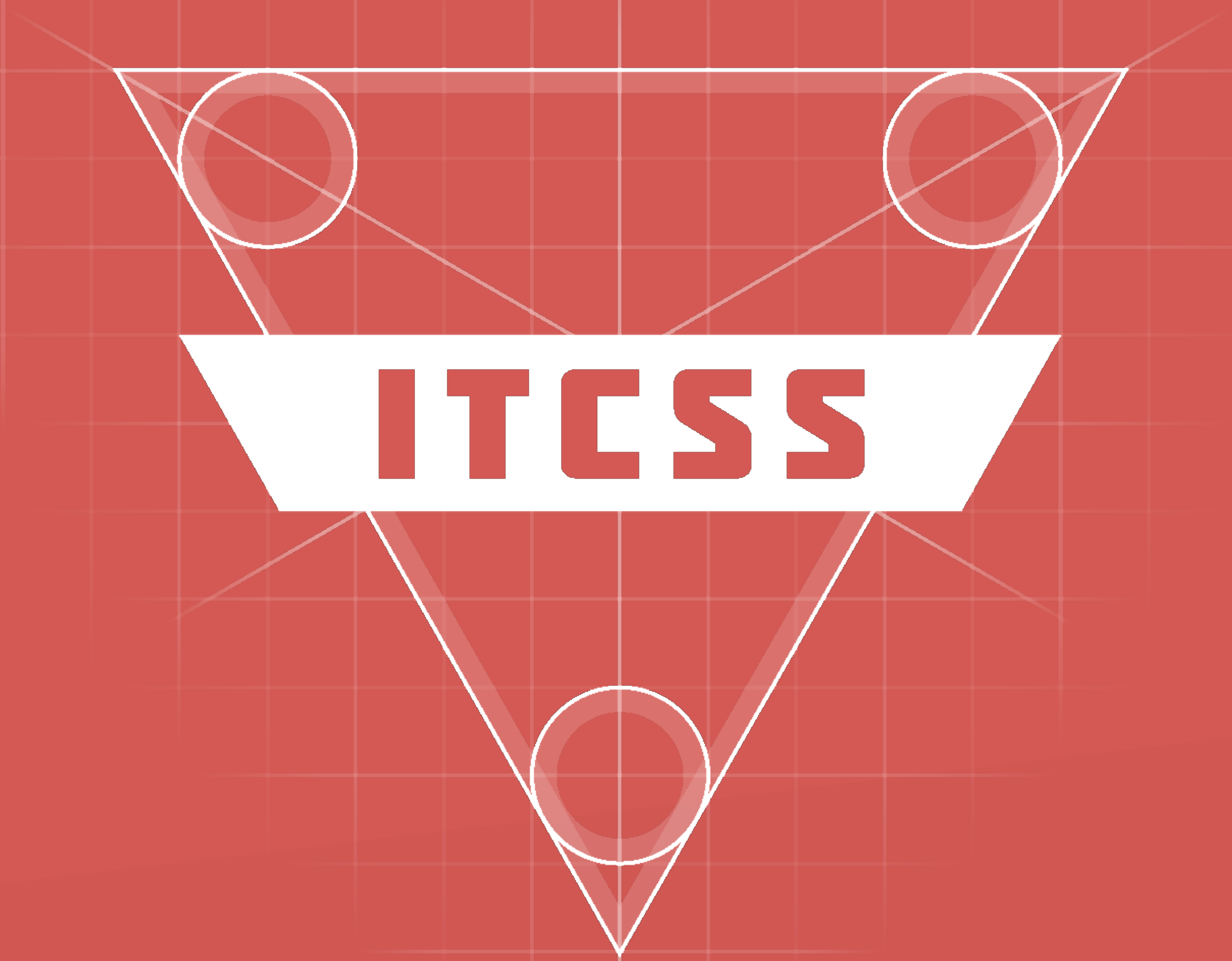
A **sane environment** that is accessible to lots of people.

To **tame and manage** source order and the cascade.

To create **a place for everything** to live (new and old).

To **reduce** waste and redundancy.

To **end the Specificity Wars**.

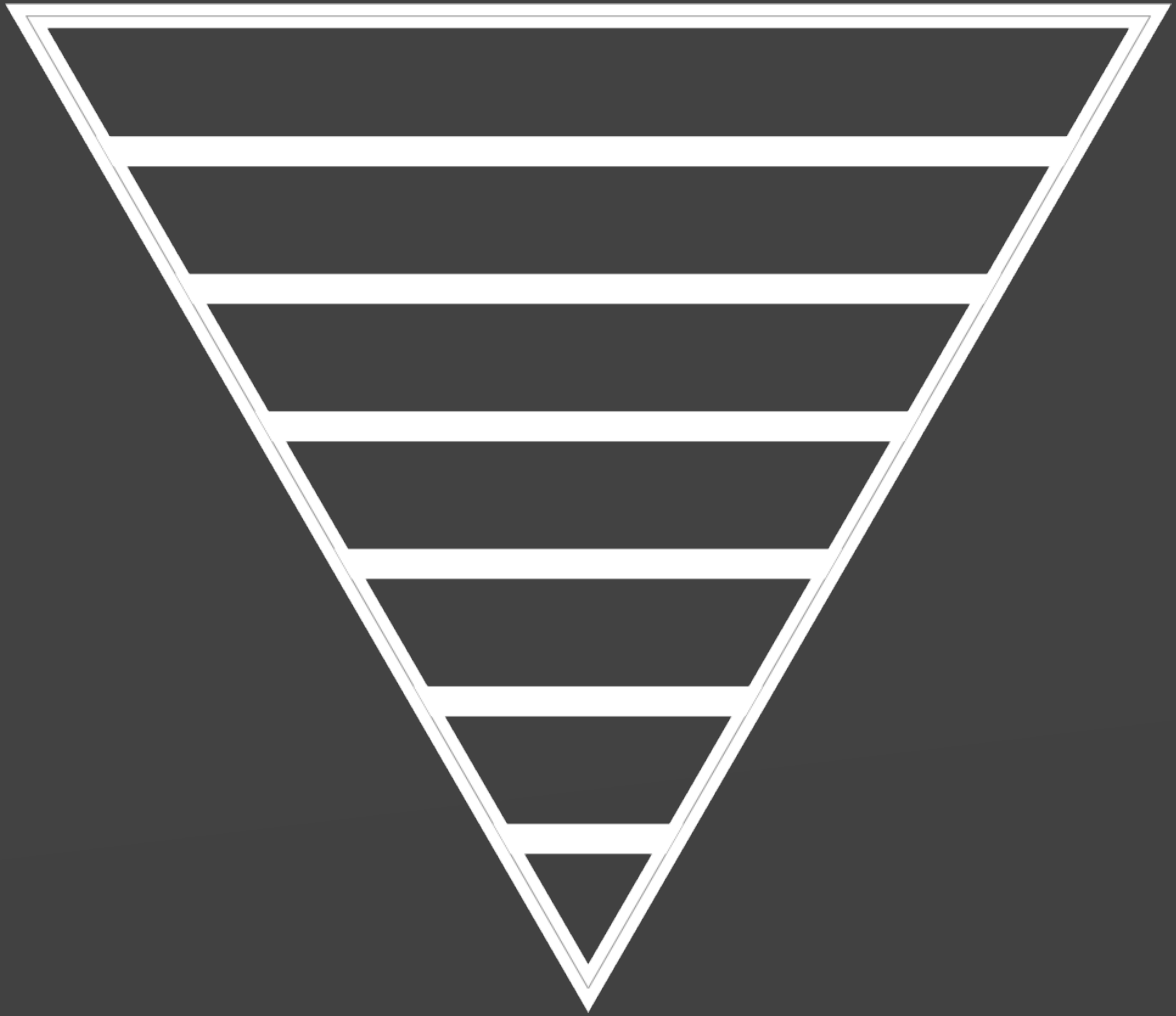

ITCSS

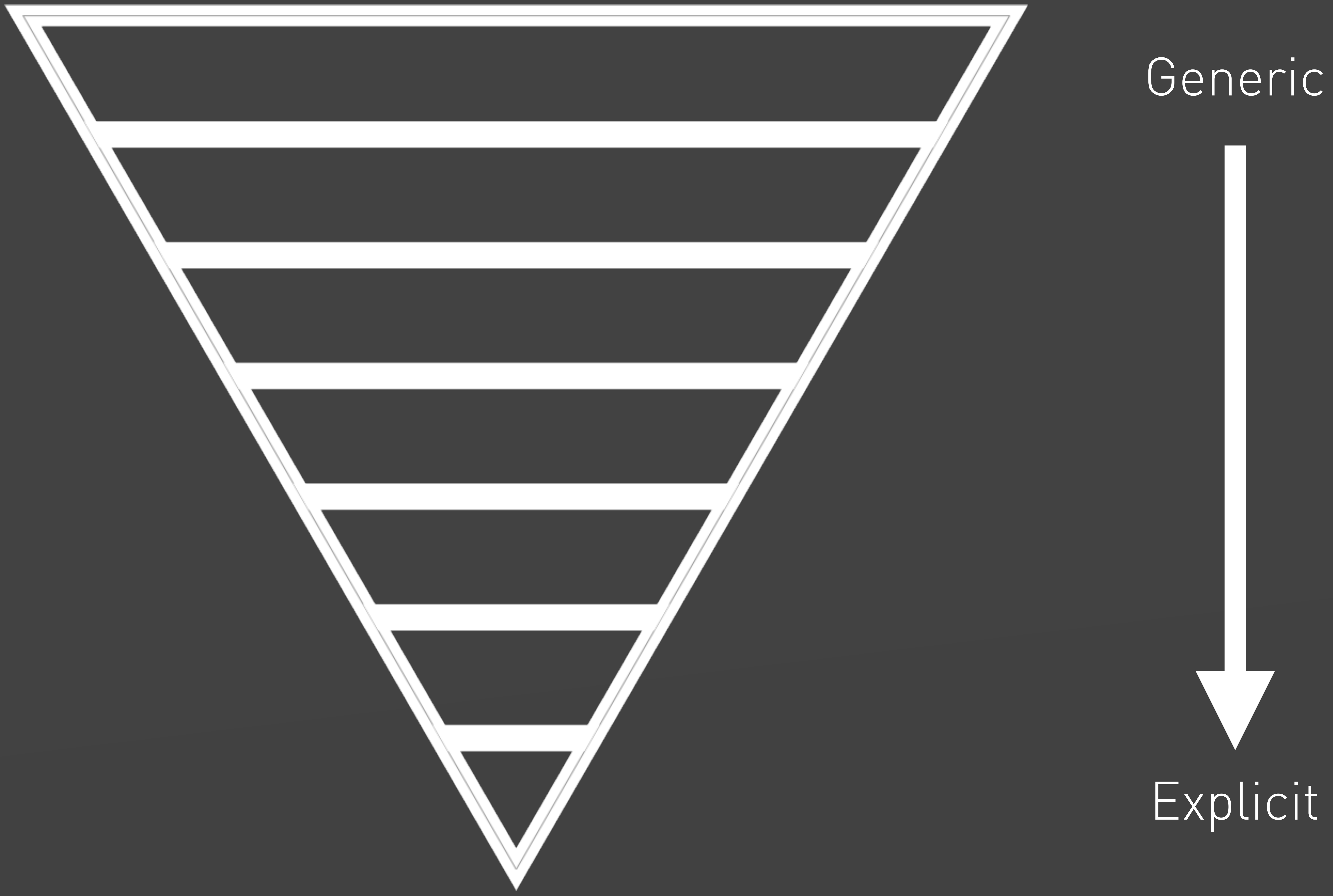
ITCSS: Inverted Triangle CSS

A lot of methodologies try and avoid or **ignore** CSS’ ‘features’…

…ITCSS makes them work to **our advantage**.

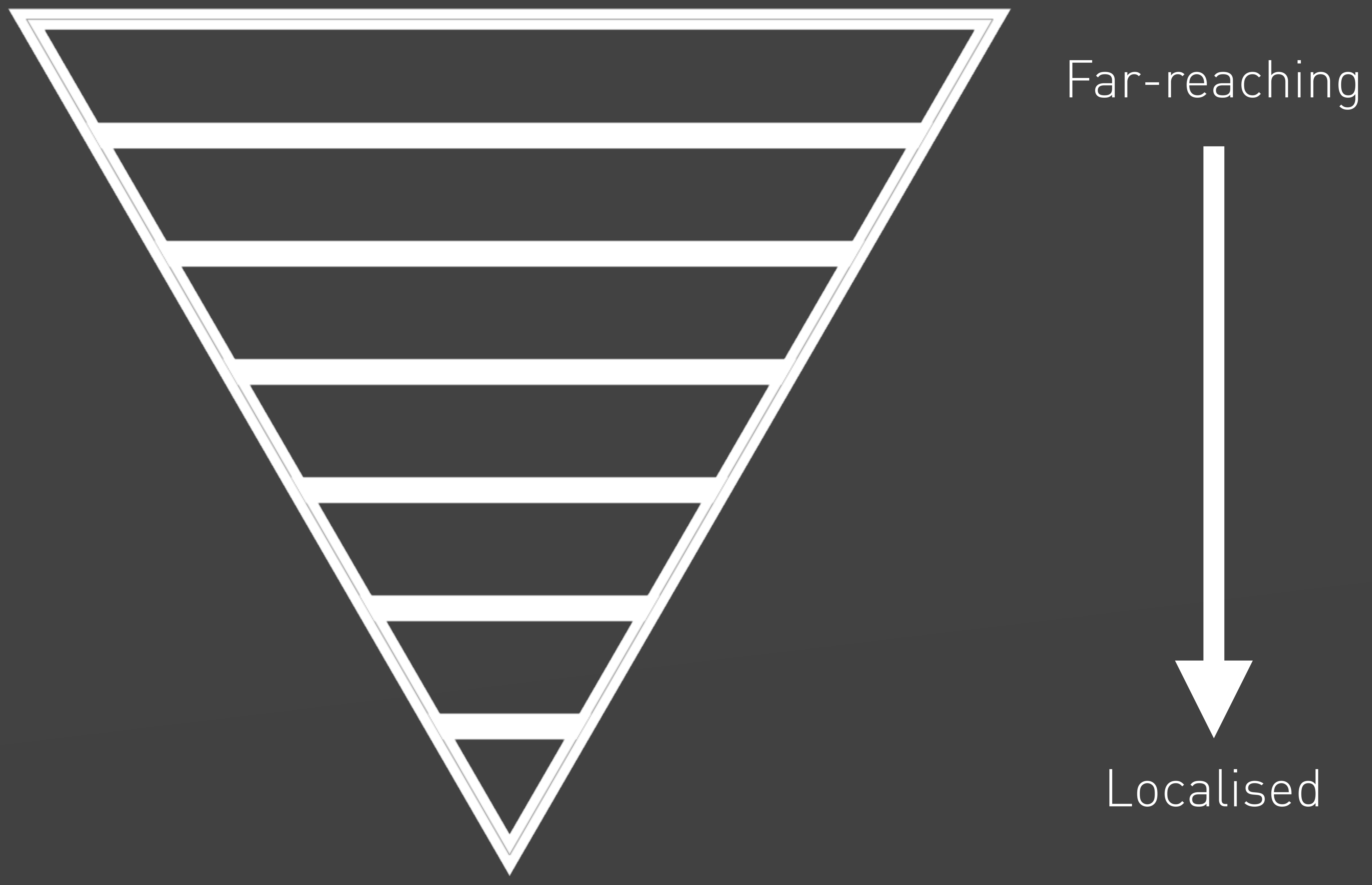
ITCSS is a **sane**, **scalable**, **managed** architecture for CSS.



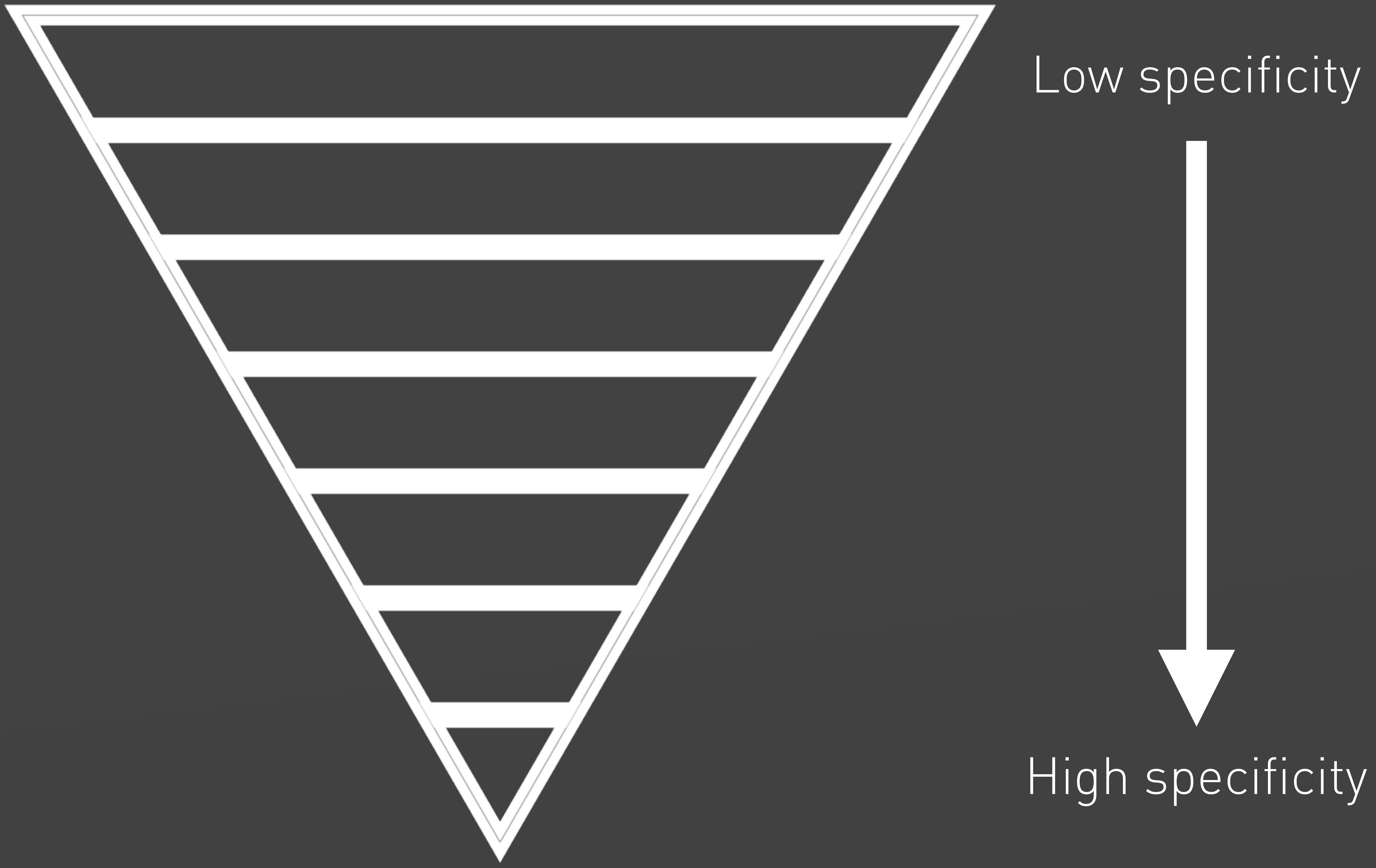


Generic

Explicit

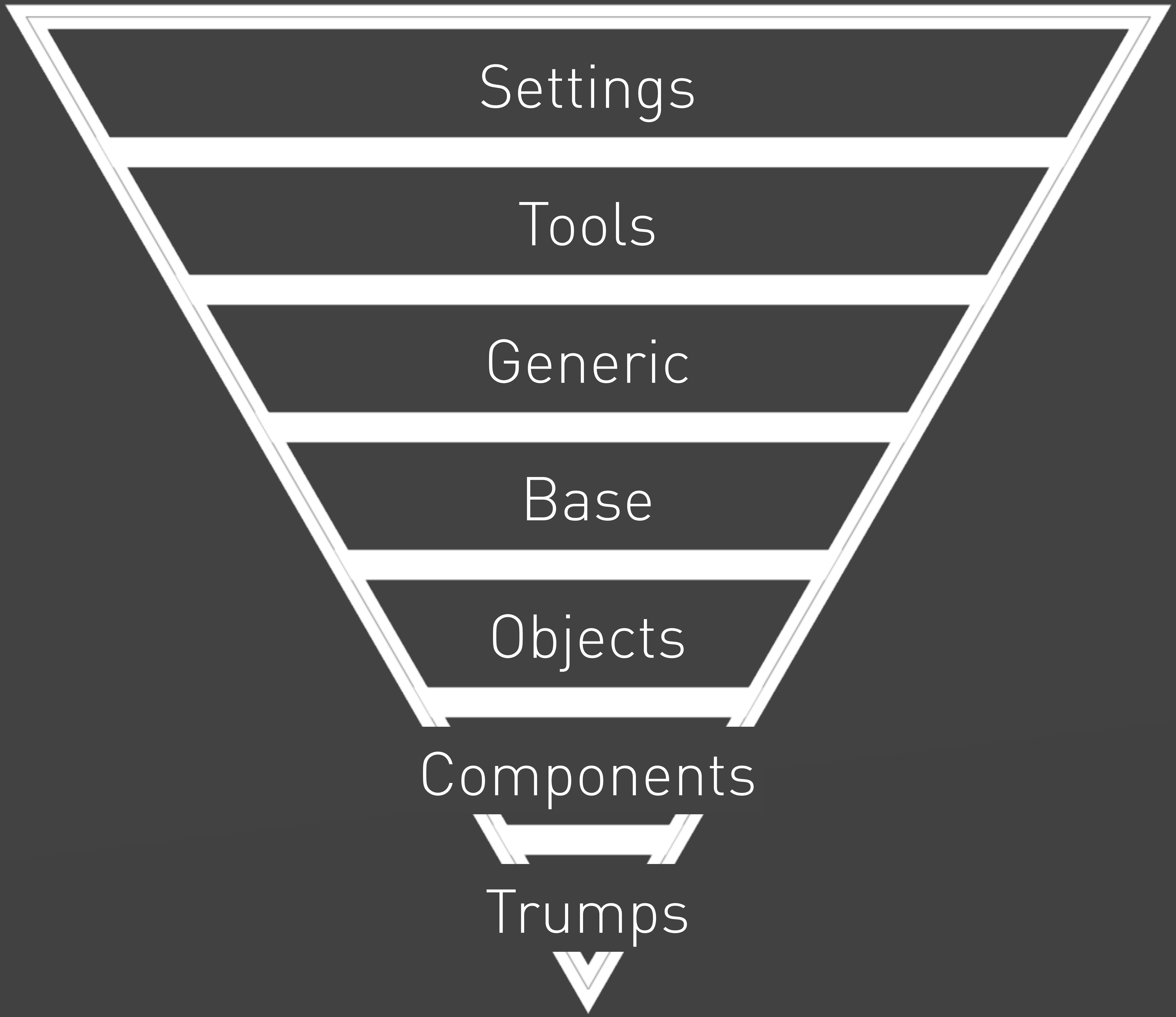


Far-reaching
Localised



Low specificity

High specificity



Settings

Tools

Generic

Base

Objects

Components

Trumps

Default layers

Settings: Global variables, config switches.

Tools: Default mixins and functions.

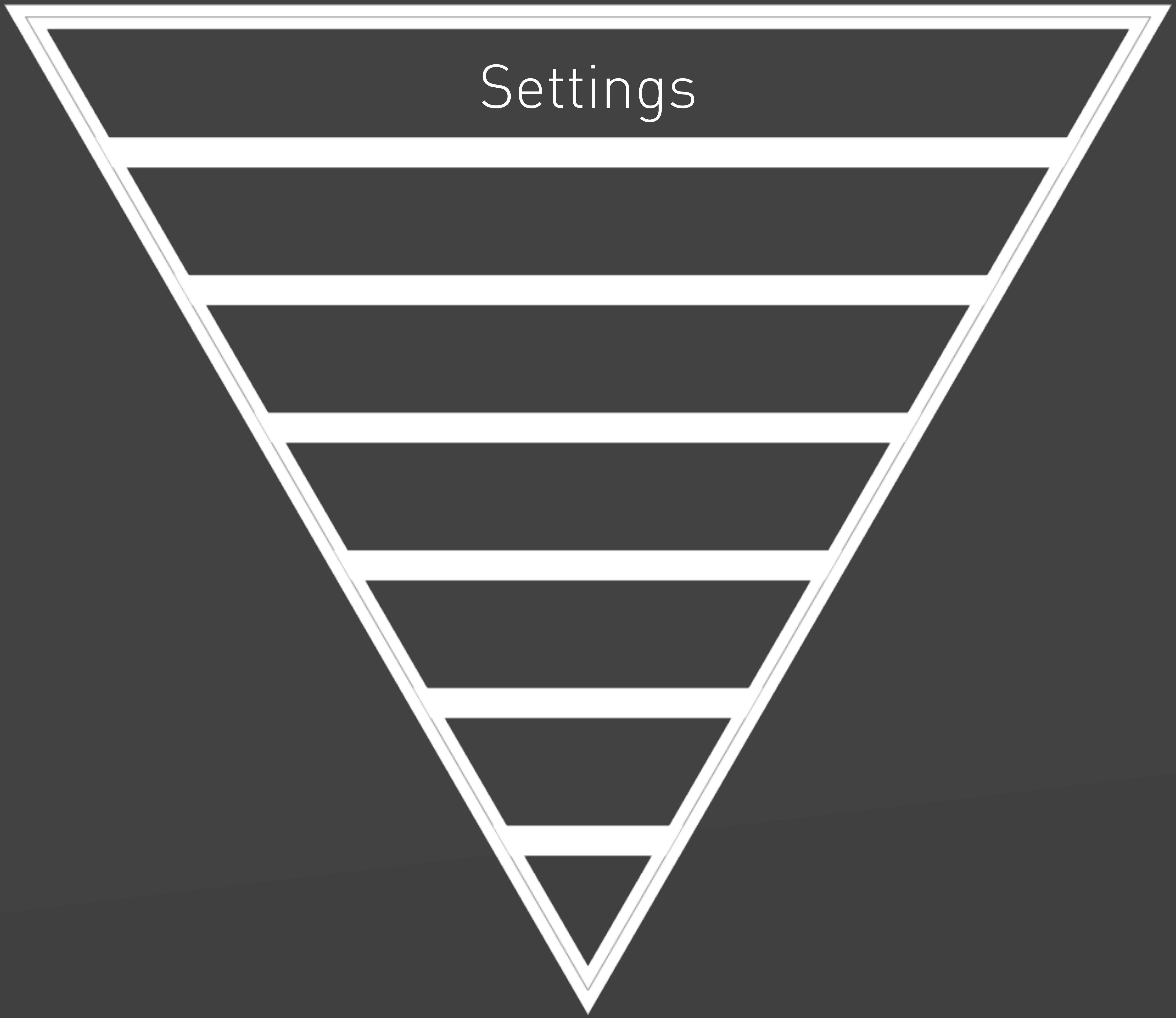
Generic: Ground-zero styles (Normalize.css, resets, box-sizing).

Base: Unclassed HTML elements (type selectors).

Objects: Cosmetic-free design patterns.

Components: Designed components, chunks of UI.

Trumps: Helpers and overrides.

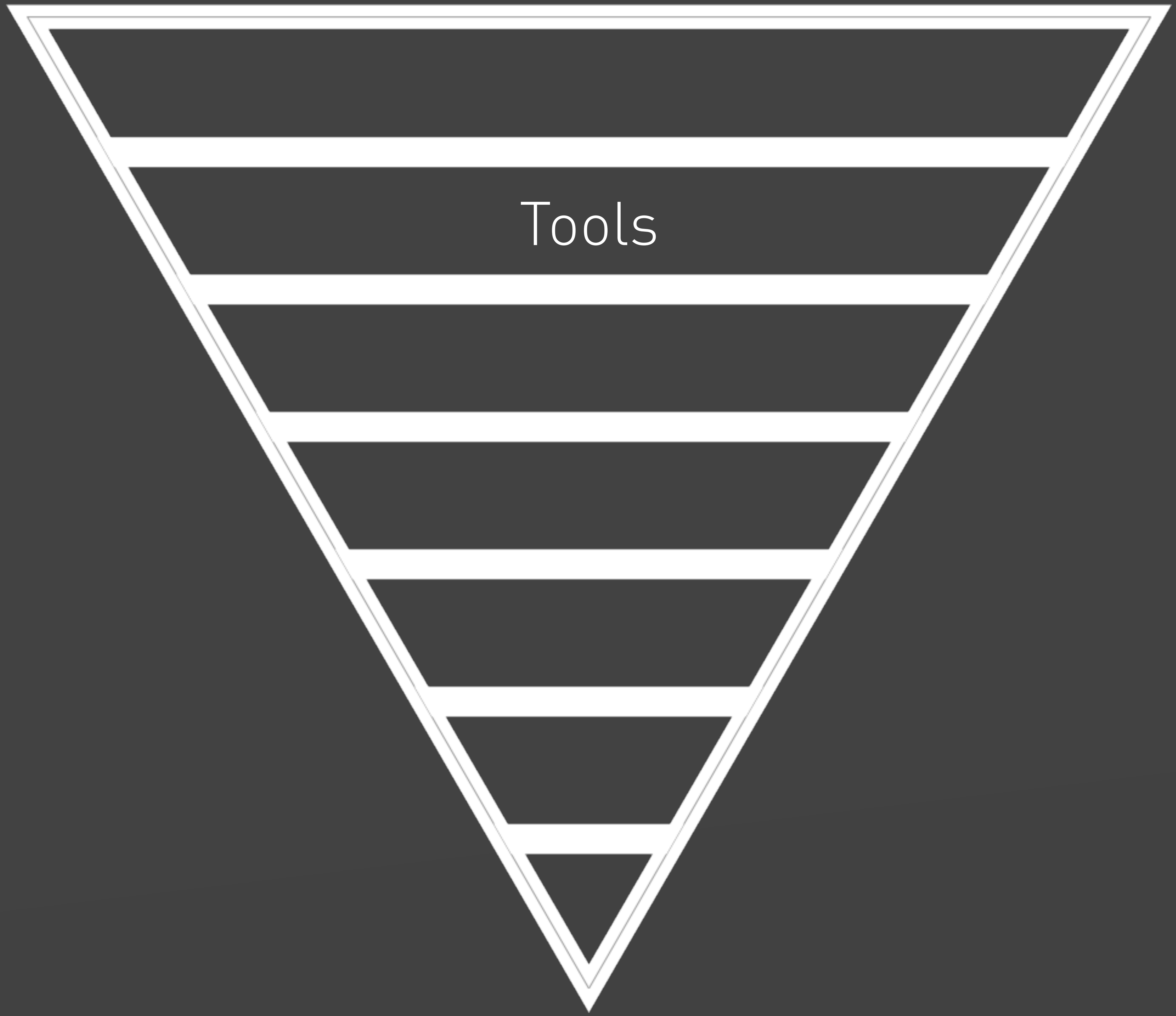


Globally-available settings.

Config switches.

Brand colours, etc.

```
$color-ui:      #BADA55;
$spacing-unit:  10px;
```

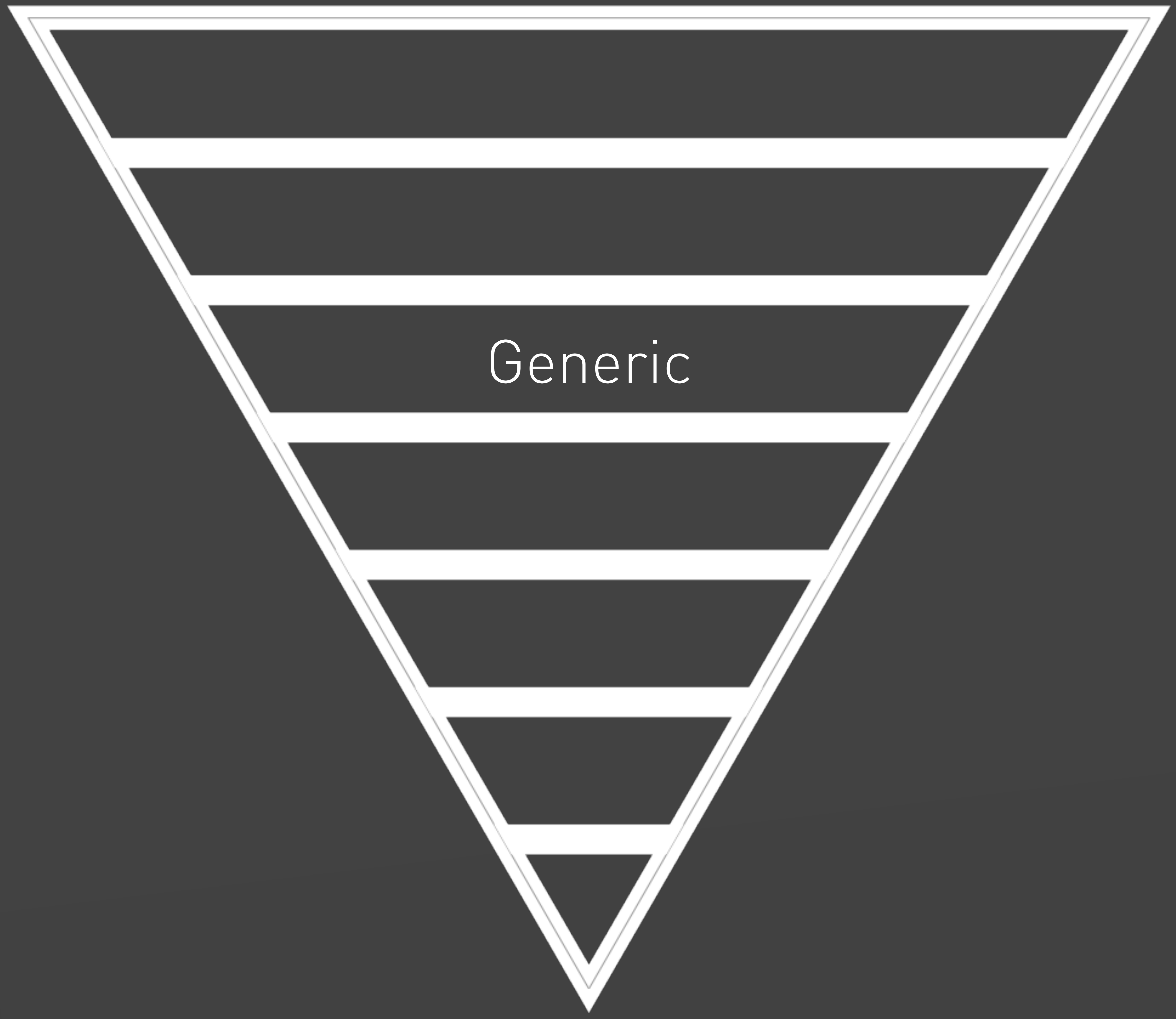


Globally-available tools.

Public mixins.

Helper functions.

```
@mixin font-brand() {
    font-family: "UI Font", sans-serif;
    font-weight: 400;
}
```

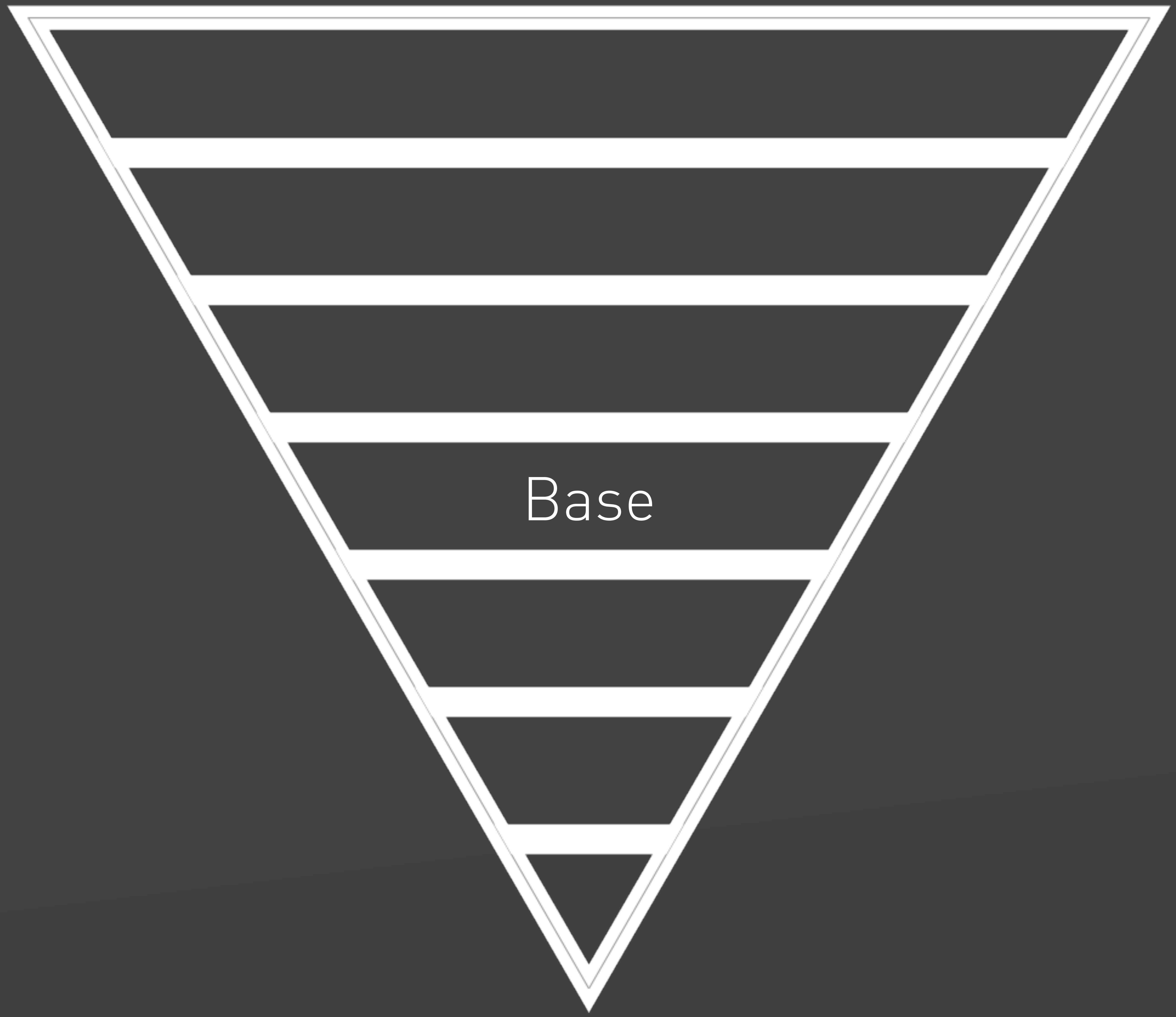


Ground zero styles.

Low-specificity, far-reaching.

Resets, Normalize.css, etc.

```
* {
    -webkit-box-sizing: border-box;
       -moz-box-sizing: border-box;
            box-sizing: border-box;
}
```

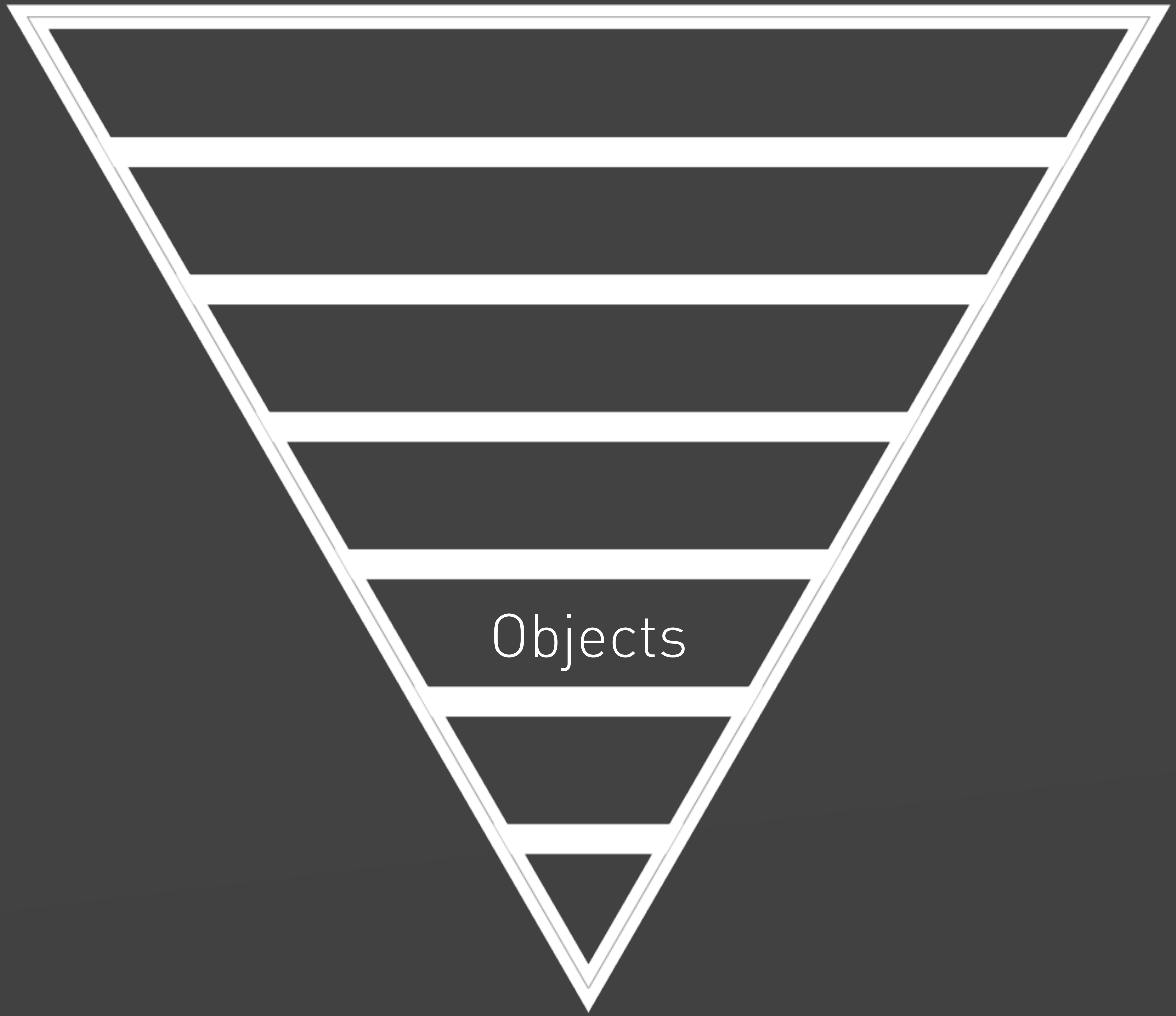


Unclassed HTML elements.

H1–H6, basic links, lists, etc.

Last layer we see type selectors (e.g. **`a {}`**, **`blockquote {}`**).

```
ul {
    list-style: square outside;
}
```



OOCSS.

Design patterns.

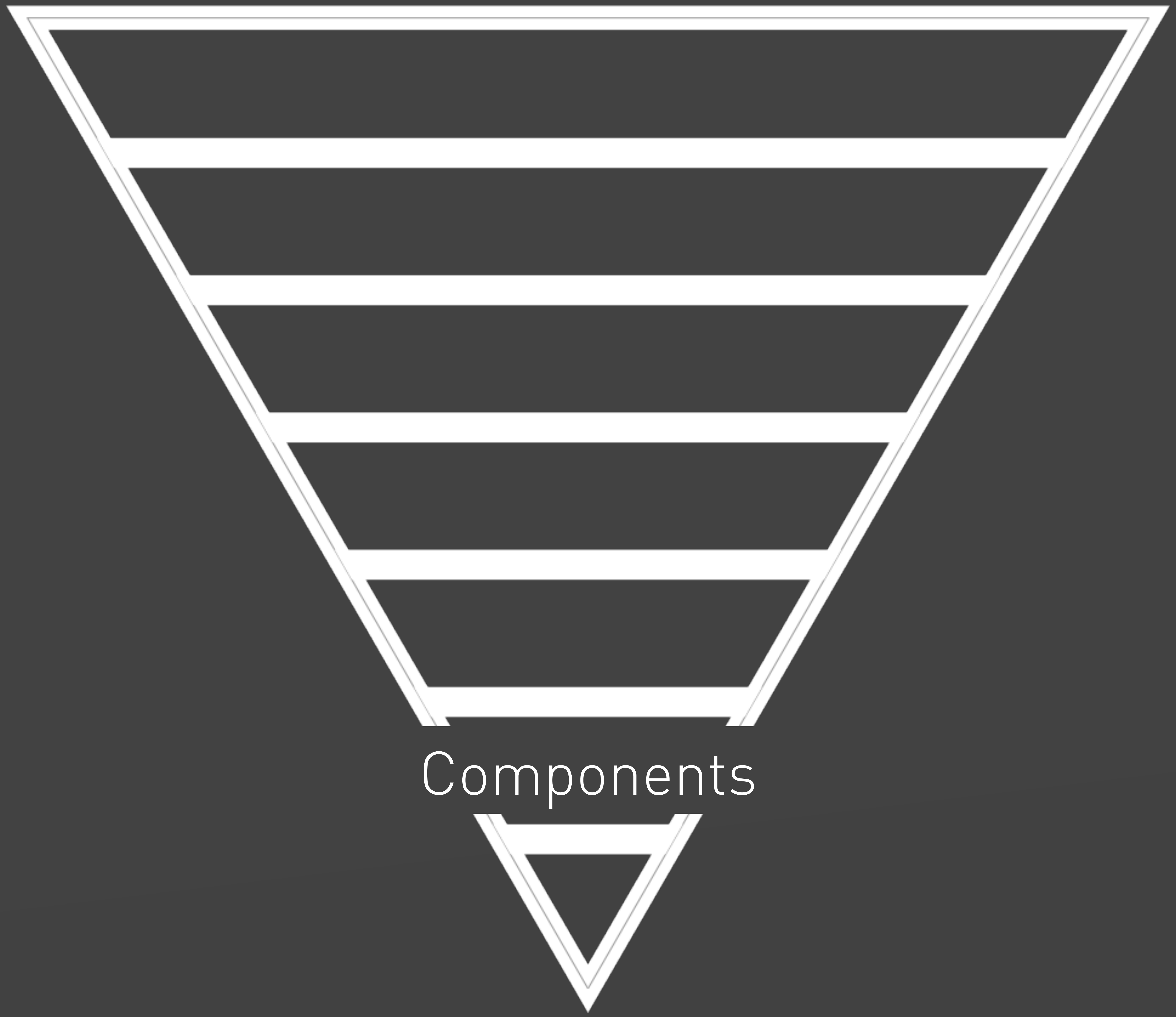
No cosmetics.

Begin using classes exclusively.

Agnostically named (e.g. `.ui-list {}`).

```
.ui-list {
    margin:  0;
    padding: 0;
    list-style: none;
}

    .ui-list__item {
        padding: $spacing-unit;
    }
```



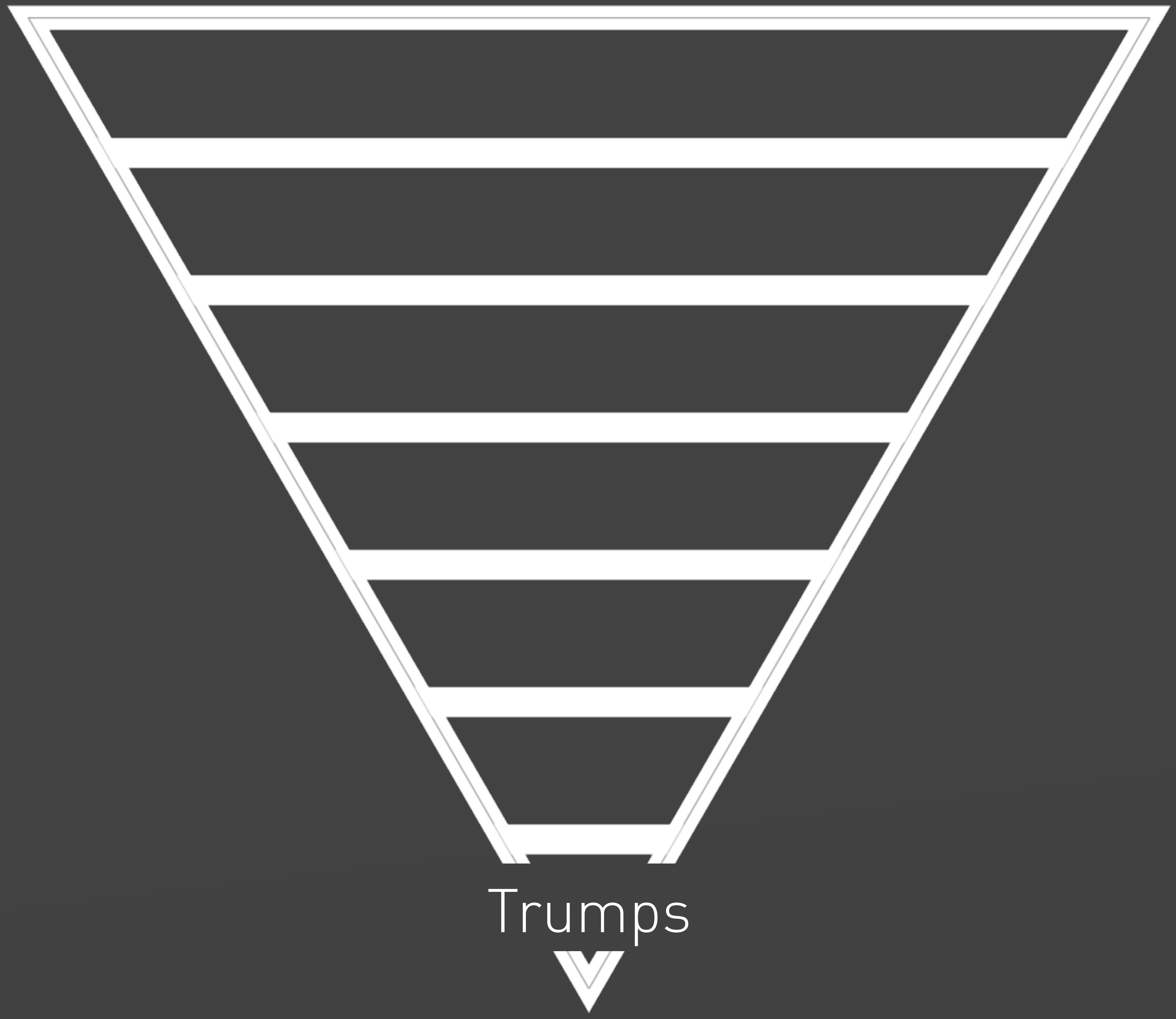
Designed pieces of UI.

Still only using classes.

More explicitly named (e.g. `.products-list {}`).

```
.products-list {
    @include font-brand();
    border-top: 1px solid $color-ui;
}

    .products-list__item {
        border-bottom: 1px solid $color-ui;
    }
```



Overrides, helpers, utilities.

Only affect one piece of the DOM at a time.

Usually carry !important.

TOP TRUMPS
I ♥ 1D
ONE DIRECTION
The Journey So Far
COLLECT 5
WHO'S THE FIERCEST?
MONSTER HIGH
TOP TRUMPS PLAY ◆ DISCOVER
GOLDEN CARD INSIDE
Dinosaurs
WORLD FOOTBALL STARS
2013/14
The Top players in the world
WHO'S THE SCARIEST?
DOCTOR WHO
BBC
WHO'S THE FASTEST?
Disney
PLANES
WHO'S THE BRAVEST?
THE HOBBIT
THE DESOLATION OF SMAUG
TOP TRUMPS SPECIALS
moshi monsters
Peppa Pig
5 great games an exciti activi
TOP TRUMPS Activity Pack!
Baby Animals

```
.one-half {
    width: 50% !important;
}
```

You should notice…

Specificity **slowly increases** layer-by-layer.

We affect **smaller and smaller** bits of the DOM at a time.

Progressively adding styles; **never undoing**.

ITCSS…

Manages source order.

Filters **explicitness**.

Tames the cascade.

Sanitises inheritance.

Each layer is **a pass over the DOM**.

How sculptors work

Blast some rock out of a quarry.

Cut it down into a **large block**.

Rough it into a **general** shape.

Begin adding **features**.

Add fine **details**.

Each stage is **more detailed and explicit** than the last one.

Outcomes

Outcomes

Everything has **a place to live**.

People know where to look to find types of rule.

A **sane** source oder.

Reduced waste/redundancy.

Increased **scalability**.

The **Specificity Wars are over**!

Scaling ITCSS

Scaling ITCSS

We can **scale our CSS much more easily**.

But we can also **scale the architecture itself**!

Scaling our CSS

No longer **the end of a stylesheet** to worry about.

Add things **into the relevant layers** (usually the last ones).

Things **never get more complicated**, only bigger.

Everything grows in a **well-rounded** manner.

The Specificity Graph keeps **trending upward**.

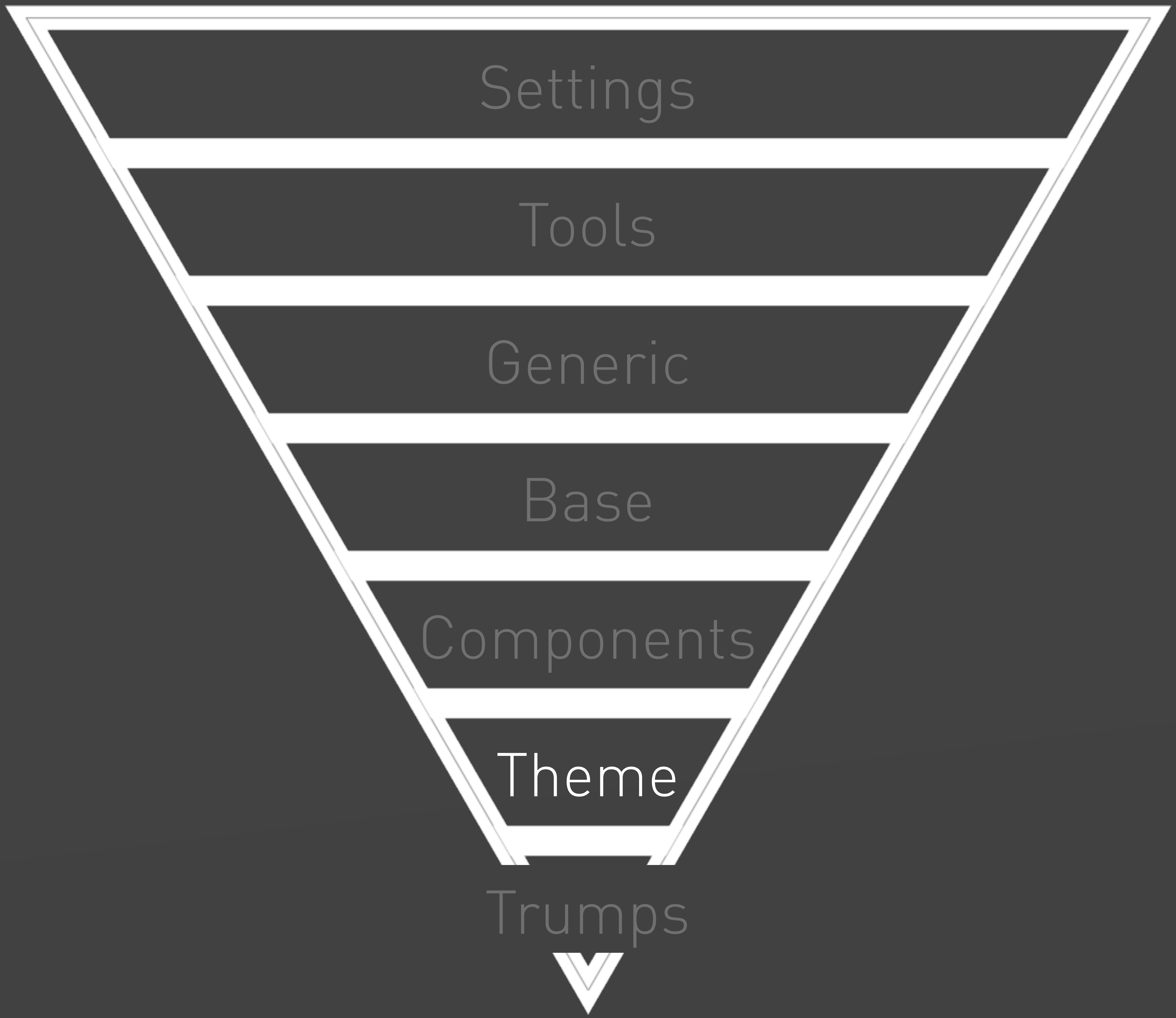
Scaling the architecture

Add or remove layers if, as, and **when you need to**.

Not using a preprocessor? **Remove** the Settings and Tools layers.

Don’t use OOCSS? **Remove** the Objects layer.

Need theming? Add a **Theme layer**.



Settings

Tools

Generic

Base

Components

Theme

Trumps

Booking.com

Booking.com run a lot of **A/B tests**.

How do we **isolate temporary styles**?

Create a **Test layer** (before the Trumps layer).

Adding layers

Add layers **in the correct place**.

Specificity and explicitness of selectors should dictate this.

Honour the Specificity Graph (always trending upward).

On the filesystem?

vim
vim
bash

```
harryroberts:~/Sites/csswizardry.com/css (master)$ ls -l
total 312
-rw-r--r--   1 harryroberts  staff    552 19 Aug 22:25 _base.links.scss
-rw-r--r--   1 harryroberts  staff    166 19 Aug 22:25 _base.page.scss
-rw-r--r--   1 harryroberts  staff    894 19 Aug 22:25 _base.quotes.scss
-rw-r--r--   1 harryroberts  staff   1862 19 Aug 22:25 _base.type.scss
-rw-r--r--   1 harryroberts  staff    141 16 Oct 15:18 _components.ads.scss
-rw-r--r--   1 harryroberts  staff   1169 19 Aug 22:25 _components.annotate.scss
-rw-r--r--   1 harryroberts  staff    506 19 Aug 22:25 _components.bands.scss
-rw-r--r--   1 harryroberts  staff    334 19 Aug 22:25 _components.boxes.scss
-rw-r--r--   1 harryroberts  staff    858 19 Aug 22:25 _components.buttons.scss
-rw-r--r--   1 harryroberts  staff   5027 19 Aug 22:25 _components.carousel.scss
-rw-r--r--   1 harryroberts  staff   3252 22 Aug 17:54 _components.feature-list.scss
-rw-r--r--   1 harryroberts  staff    191 19 Aug 22:25 _components.iconic.scss
-rw-r--r--   1 harryroberts  staff    246 19 Aug 22:25 _components.page-foot.scss
-rw-r--r--   1 harryroberts  staff   3759 22 Aug 17:26 _components.page-head.scss
-rw-r--r--   1 harryroberts  staff    348 19 Aug 22:25 _components.page-micro.scss
-rw-r--r--   1 harryroberts  staff    333 19 Aug 22:25 _components.post.scss
-rw-r--r--   1 harryroberts  staff    485 19 Aug 22:25 _components.promo.scss
-rw-r--r--   1 harryroberts  staff   1290 19 Aug 22:25 _components.pull-quote.scss
-rw-r--r--   1 harryroberts  staff   4486 22 Aug 17:36 _components.site-nav.scss
-rw-r--r--   1 harryroberts  staff   1518 27 Aug 10:56 _components.sprites.scss
-rw-r--r--   1 harryroberts  staff    603 19 Aug 22:25 _objects.wrappers.scss
-rw-r--r--   1 harryroberts  staff   1650 19 Aug 22:25 _settings.colors.scss
-rw-r--r--   1 harryroberts  staff    177 19 Aug 22:25 _settings.global.scss
-rw-r--r--   1 harryroberts  staff    332 19 Aug 22:25 _tools.aliases.scss
-rw-r--r--   1 harryroberts  staff   1187 19 Aug 22:25 _tools.mixins.scss
-rw-r--r--   1 harryroberts  staff    673 19 Aug 22:25 _trumps.show-hide.scss
-rw-r--r--   1 harryroberts  staff   3776 19 Aug 22:25 _trumps.widths-responsive.scss
-rw-r--r--   1 harryroberts  staff    975 19 Aug 22:25 bower.json
drwxr-xr-x  25 harryroberts  staff    850 22 Jul 12:53 bower_components
-rw-r--r--   1 harryroberts  staff  23947 16 Oct 15:18 csswizardry.min.css
-rw-r--r--   1 harryroberts  staff   6044 16 Oct 15:18 csswizardry.scss
-rwxr-xr-x   1 harryroberts  staff    102 15 Oct 11:25 watch
harryroberts:~/Sites/csswizardry.com/css (master)$ 
```

```
@import "bower_components/inuit-defaults/settings.defaults";
@import "settings.global";
@import "settings.colors";
@import "bower_components/inuit-responsive-settings/settings.responsive";

@import "bower_components/inuit-functions/tools.functions";
@import "bower_components/inuit-mixins/tools.mixins";
@import "tools.mixins";
@import "bower_components/inuit-responsive-tools/tools.responsive";
@import "tools.aliases";

@import "bower_components/inuit-normalize/generic.normalize";
@import "bower_components/inuit-reset/generic.reset";
@import "bower_components/inuit-box-sizing/generic.box-sizing";
@import "bower_components/inuit-shared/generic.shared";

@import "bower_components/inuit-page/base.page";
@import "base.page";
@import "bower_components/inuit-headings/base.headings";
@import "base.type";
@import "base.links";
@import "bower_components/inuit-lists/base.lists";
@import "bower_components/inuit-images/base.images";
@import "base.quotes";

@import "objects.wrappers";
@import "bower_components/inuit-buttons/objects.buttons";
@import "bower_components/inuit-layout/objects.layout";
@import "bower_components/inuit-box/objects.box";
@import "bower_components/inuit-media/objects.media";

@import "components.page-head";
@import "components.bands";
@import "components.promo";
@import "components.page-foot";
@import "components.page-micro";
@import "components.site-nav";
@import "components.buttons";
@import "components.carousel";
@import "components.iconic";
@import "components.sprites";
@import "components.feature-list";
@import "components.annotate";
@import "components.ads";

@import "bower_components/inuit-clearfix/trumps.clearfix";
```

Recap

Recap

Write CSS in **specificity order**.

Maintain the **Specificity Graph**.

All rulesets should only ever **add to and inherit from previous ones**.

Order stylesheets **from far-reaching to very localised**.

Add layers as needed, but only **in the right place**.

Thank you – csswz.it/itcss-dafed

Harry Roberts – @csswizardry