

CSS Building Blocks

Selectors

R. Scott Granneman & Jans Carton

© 2005 R. Scott Granneman
Last updated 2023-09-28

You are free to use this work, with certain restrictions.
For full licensing information, please see the last slide/page.

Notes & URLs for this presentation can be found...

- » underneath the link to this slide show on granneman.com
- » at granneman.com/downloads/web-dev//CSS-Selectors.txt

Scott Granneman



[Writing](#) [Presentations](#) [Teaching](#) [Web Dev](#) [Tech](#) [Personal](#)

[Site Map](#) [Search](#)

Scott Granneman

contributed
in St. Louis &
plans, devel
[presentation](#)

- Interviews
- Ladue Chapel
- Exploring HTML5 & CSS
- Advanced CSS
- HTML to CSS to Responsive Web Design
- Responsive Web Design for Designers
- Web Design
- Archived Presentations
- Creative Commons License

Key S

Pages you're

Personal

- [Publications](#)
- [Summer Vacation 2007: The Plains](#)

Teaching

- [Web Development Key Links](#)
- Wash U: [Website Design & Development Syllabus](#)
- Webster U (Comp Sci): [Intro to Web Programming Syllabus](#)

in a Web development company in St. Louis, Missouri. Scott has written six books, for SecurityFocus and *Linux Magazine*. He is an Adjunct Professor at Washington University in technology, security, web development, & the Internet. As a Principal of [WebSanity](#), he states using an open source, UNIX-based [content management system](#). He has given speaking to people.

To fully comprehend these slides, you should first read & understand CSS Overview

- » Slides: files.granneman.com/presentations/webdev/CSS-Overview.pdf
- » Notes: files.granneman.com/presentations/webdev/CSS-Overview.txt

Basics

A *CSS selector* declares which DOM objects should have particular styles applied to them

The browser's rendering engine...

- » looks through the CSS & HTML
- » matches selectors to the appropriate DOM objects
- » applies the CSS style to the rendered DOM objects

HTML

```
<h1 align="center"></h1>
```

Entire thing is an *element*

`h1` is an *tag name**

`align` is an *attribute*

`center` is a *value*

`align="center"` is an *attribute-value pair*

CSS

```
h1 {text-align: center;}
```

Entire thing is a *rule set*

`h1` is a *selector*

`text-align` is a *property*

`center` is a *value*

`text-align: center` is a *declaration*

Everything inside `{ & }` is a *declaration block*

* But most people call it an *element*

How Many?

Selectors Level 3

W3C Recommendation

September 29, 2011

www.w3.org/TR/css3-selectors/

Pattern	Meaning	Described in section	First defined in CSS level
*	any element	Universal selector	2
E	an element of type E	Type selector	1
E[foo]	an E element with a "foo" attribute	Attribute selectors	2
E[foo="bar"]	an E element whose "foo" attribute value is exactly equal to "bar"	Attribute selectors	2
E[foo~="bar"]	an E element whose "foo" attribute value is a list of whitespace-separated values, one of which is exactly equal to "bar"	Attribute selectors	2
E[foo^="bar"]	an E element whose "foo" attribute value begins exactly with the string "bar"	Attribute selectors	3
E[foo\$="bar"]	an E element whose "foo" attribute value ends exactly with the string "bar"	Attribute selectors	3
E[foo*="bar"]	an E element whose "foo" attribute value contains the substring "bar"	Attribute selectors	3
E[foo="en"]	an E element whose "foo" attribute has a hyphen-separated list of values beginning (from the left) with "en"	Attribute selectors	2
E:root	an E element, root of the document	Structural pseudo-classes	3
E:nth-child(n)	an E element, the n-th child of its parent	Structural pseudo-classes	3
E:nth-last-child(n)	an E element, the n-th child of its parent, counting from the last one	Structural pseudo-classes	3
E:nth-of-type(n)	an E element, the n-th sibling of its type	Structural pseudo-classes	3
E:nth-last-of-type(n)	an E element, the n-th sibling of its type, counting from the last one	Structural pseudo-classes	3
E:first-child	an E element, first child of its parent	Structural pseudo-classes	2
E:last-child	an E element, last child of its parent	Structural pseudo-classes	3
E:first-of-type	an E element, first sibling of its type	Structural pseudo-classes	3
E:last-of-type	an E element, last sibling of its type	Structural pseudo-classes	3
E:only-child	an E element, only child of its parent	Structural pseudo-classes	3
E:only-of-type	an E element, only sibling of its type	Structural pseudo-classes	3
E:empty	an E element that has no children (including text nodes)	Structural pseudo-classes	3
E:link	an E element being the source anchor of a hyperlink of which the target is not yet visited (:link) or already visited (:visited)	The link pseudo-classes	1
E:visited			
E:active	an E element during certain user actions	The user action pseudo-classes	1 and 2
E:hover			
E:focus			
E:target	an E element being the target of the referring URI	The target pseudo-class	3
E:lang(fr)	an element of type E in language "fr" (the document language specifies how language is determined)	The :lang() pseudo-class	2
E:enabled	a user interface element E which is enabled or disabled	The UI element states pseudo-classes	3
E:disabled			
E:checked	a user interface element E which is checked (for instance a radio-button or checkbox)	The UI element states pseudo-classes	3
E::first-line	the first formatted line of an E element	The ::first-line pseudo-element	1
E::first-letter	the first formatted letter of an E element	The ::first-letter pseudo-element	1
E::before	generated content before an E element	The ::before pseudo-element	2
E::after	generated content after an E element	The ::after pseudo-element	2
E.warning	an E element whose class is "warning" (the document language specifies how class is determined).	Class selectors	1
E#myid	an E element with ID equal to "myid".	ID selectors	1
E:not(s)	an E element that does not match simple selector s	Negation pseudo-class	3
E F	an F element descendant of an E element	Descendant combinator	1
E > F	an F element child of an E element	Child combinator	2
E + F	an F element immediately preceded by an E element	Adjacent sibling combinator	2
E ~ F	an F element preceded by an E element	General sibling combinator	3

Selectors Level 4

W3C Working Draft

November 21, 2018

www.w3.org/TR/selectors4/

Pattern	Represents	Section	Level
*	any element	Universal selector	2
E	an element of type E	Type (tag name) selector	1
E:not(<i>s1</i> , <i>s2</i>)	an E element that does not match either compound selector <i>s1</i> or compound selector <i>s2</i>	Negation pseudo-class	3/4
E:matches(<i>s1</i> , <i>s2</i>)	an E element that matches compound selector <i>s1</i> and/or compound selector <i>s2</i>	Matches-any pseudo-class	4
E.warning	an E element belonging to the class warning (the document language specifies how class is determined).	Class selectors	1
E#myid	an E element with ID equal to myid.	ID selectors	1
E[foo]	an E element with a foo attribute	Attribute selectors	2
E[foo="bar"]	an E element whose foo attribute value is exactly equal to bar	Attribute selectors	2
E[foo="bar" i]	an E element whose foo attribute value is exactly equal to any (ASCII-range) case-permutation of bar	Attribute selectors: Case-sensitivity	4
E[foo~="bar"]	an E element whose foo attribute value is a list of whitespace-separated values, one of which is exactly equal to bar	Attribute selectors	2
E[foo^="bar"]	an E element whose foo attribute value begins exactly with the string "bar"	Attribute selectors	3
E[foo\$="bar"]	an E element whose foo attribute value ends exactly with the string bar	Attribute selectors	3
E[foo*="bar"]	an E element whose foo attribute value contains the substring bar	Attribute selectors	3
E[foo ="en"]	an E element whose foo attribute value is a hyphen-separated list of values beginning with en	Attribute selectors	2
E:dir(ltr)	an element of type E in with left-to-right directionality (the document language specifies how directionality is determined)	The :dir() pseudo-class	4
E:lang(zh, *-hant)	an element of type E tagged as being either in Chinese (any dialect or writing system) or otherwise written with traditional Chinese characters	The :lang() pseudo-class	2/4
E:any-link	an E element being the source anchor of a hyperlink	The hyperlink pseudo-class	4
E:link	an E element being the source anchor of a hyperlink of which the target is not yet visited	The link history pseudo-classes	1
E:visited	an E element being the source anchor of a hyperlink of which the target is already visited	The link history pseudo-classes	1
E:local-link	an E element being the source anchor of a hyperlink of which the target is the current document	The local link pseudo-class	4
E:local-link(0)	an E element being the source anchor of a hyperlink of which the target is within the current domain	The local link pseudo-class	4
E:target	an E element being the target of the referring URL	The target pseudo-class	3
E:scope	an E element being a designated contextual reference element	The scope pseudo-class	4
E:current	an E element that is currently presented in a time-dimensional canvas	Time-dimensional Pseudo-classes	4
E:current(<i>s</i>)	an E element that is the deepest :current element that matches selector <i>s</i>	Time-dimensional Pseudo-classes	4
E:past	an E element that is in the past in a time-dimensional canvas	Time-dimensional Pseudo-classes	4
E:future	an E element that is in the future in a time-dimensional canvas	Time-dimensional Pseudo-classes	4
E:active	an E element that is in an activated state	The user action pseudo-classes	1
E:hover	an E element that is under the cursor, or that has a descendant under the cursor	The user action pseudo-classes	2

E:focus	an E element that has user input focus	The user action pseudo-classes	2
E:enabled E:disabled	a user interface element E that is enabled or disabled, respectively	The :enabled and :disabled pseudo-classes	3
E:checked	a user interface element E that is checked/selected (for instance a radio-button or checkbox)	The selected-option pseudo-class	3
E:indeterminate	a user interface element E that is in an indeterminate state (neither checked nor unchecked)	The indeterminate-value pseudo-class	4
E:default	a user interface element E that	The default option pseudo-class :default	3-UI/4
E:in-range E:out-of-range	a user interface element E that	The validity pseudo-classes	3-UI/4
E:required E:optional	a user interface element E that	The optionality pseudo-classes	3-UI/4
E:read-only E:read-write	a user interface element E that	The mutability pseudo-classes	3-UI/4
E:root	an E element, root of the document	Structural pseudo-classes	3
E:empty	an E element that has no children (not even text nodes)	Structural pseudo-classes	3
E:first-child	an E element, first child of its parent	Structural pseudo-classes	2
E:nth-child(<i>n</i>)	an E element, the <i>n</i> -th child of its parent	Structural pseudo-classes	3
E:last-child	an E element, last child of its parent	Structural pseudo-classes	3
E:nth-last-child(<i>n</i>)	an E element, the <i>n</i> -th child of its parent, counting from the last one	Structural pseudo-classes	3
E:only-child	an E element, only child of its parent	Structural pseudo-classes	3
E:first-of-type	an E element, first sibling of its type	Structural pseudo-classes	3
E:nth-of-type(<i>n</i>)	an E element, the <i>n</i> -th sibling of its type	Structural pseudo-classes	3
E:last-of-type	an E element, last sibling of its type	Structural pseudo-classes	3
E:nth-last-of-type(<i>n</i>)	an E element, the <i>n</i> -th sibling of its type, counting from the last one	Structural pseudo-classes	3
E:only-of-type	an E element, only sibling of its type	Structural pseudo-classes	3
E:nth-match(<i>n</i> of <i>selector</i>)	an E element, the <i>n</i> -th sibling matching <i>selector</i>	Structural pseudo-classes	4
E:nth-last-match(<i>n</i> of <i>selector</i>)	an E element, the <i>n</i> -th sibling matching <i>selector</i> , counting from the last one	Structural pseudo-classes	4
E:column(<i>selector</i>)	an E element that represents a cell in a grid/table belonging to a column represented by an element that matches <i>selector</i>	Grid-Structural pseudo-classes	4
E:nth-column(<i>n</i>)	an E element that represents a cell belonging to the <i>n</i> th column in a grid/table	Grid-Structural pseudo-classes	4
E:nth-last-column(<i>n</i>)	an E element that represents a cell belonging to the <i>n</i> th column in a grid/table, counting from the last one	Grid-Structural pseudo-classes	4
E F	an F element descendant of an E element	Descendant combinator	1
E > F	an F element child of an E element	Child combinator	2
E + F	an F element immediately preceded by an E element	Next-sibling combinator	2
E ~ F	an F element preceded by an E element	Following-sibling combinator	3
E /foo/ F	an F element ID-referenced by an E element's foo attribute	Reference combinator	4
E! > F	an E element parent of an F element	Determining the subject of a selector + Child combinator	4

CSS 1: 10 different selectors

CSS 2: 13

CSS 3: 21

CSS 4: 34 (as of July 2020)

78 in total

CSS 1

E, `.class`, `#ID`, E F, pseudo classes (`:link`, `:visited`, & `:active`), pseudo-elements (`::first-line` & `::first-letter`), lists (`.classA`, `.classB`, `.classC`)

CSS 2

`*, E[foo], E[foo="bar"], E[foo~="bar"], E[foo|="en"], :first-child, :lang(fr), :active), :hover, :focus, ::before, ::after, E > F, E + F`

3

`E[foo^="bar"]`, `E[foo$="bar"]`, `E[foo*="bar"]`,
`:root`, `:nth-child(n)`, `:nth-last-child(n)`, `:nth-`
`of-type(n)`, `:nth-last-of-type(n)`, `:last-child`,
`:first-of-type`, `:last-of-type`, `:only-child`,
`:only-of-type`, `:empty`, `:target`, `:enabled`,
`:disabled`, `:checked`, `:indeterminate`,
`:not(s)`, `E ~ F`

4

```
[attribute='value' i], [attribute='value' s],  
:blank, :nth-child(), :dir(ltr), :any-link,  
:lang(en-*), :local-link, :is(s1, s2, ...),  
:where(s1, s2, ...), :read-only, :read-write,  
:not(s1, s2, ...), :required, :optional,  
:placeholder-shown, :indeterminate, :valid,  
:invalid, :user-invalid, :has(), :scope, :in-range,  
:out-of-range, :nth-col(n), :nth-last-col(n),  
:current, :past, :future, :default, :focus-within,  
:focus-visible, :target-within, E || F
```

Why so many? Why so specific?

Because they are all needed, at one time or another

You don't need to memorize all of them — just be aware of them

1. Simple selectors

- » Universal
- » Type
- » Class
- » ID
- » Pseudo-classes
- » Pseudo-elements
- » Attribute

2. Compound selectors

3. Complex selectors (using combinators)

- » Descendant
- » Child
- » Adjacent sibling
- » General sibling

4. Selector list

Simple Selectors

Selector Notation	Meaning	Introduced
*	Universal selector	CSS 2
element	Type selector	CSS 1
.class	Class	CSS 1
#id	ID	CSS 1

All covered extensively in *CSS Overview*

Pseudo-Classes

`:foo`

indicates a *pseudo-class*, which is used to style an element *based on its current state*

Examples:

- » `:hover` (applies only when hovering over an element)
- » `:first-child`
- » `:enabled`

CSS 1, 3, 4

Location
Pseudo-Class Selectors

Hyperlink: `:any-link`

Link history: `:link & :visited`

Target: `:target`

Reference element: `:scope`

`:any-link`

Hyperlink pseudo-class that *matches any element with the `href` attribute* (`<a>`, `<area>`, or `<link>`)

Basically, `:any-link` works exactly like the `[href]` attribute selector (covered later)

CSS 1

`:link`

`:visited`

Link history pseudo-classes match *unvisited* & *visited* links

By default, `:link` is blue & underlined, while `:visited` is purple & underlined

Let's review...

URL fragment

Name preceded by #, specifying internal location in the current page targeted using the `id` attribute

```
<a href="#chapter-1">Chapter 1</a>
```

...

```
<h2 id="chapter-1">1: Moorings</h2>
```

CSS 3

:target

Identifies the *destination (target) of a link* that points to a specific portion of a document *via a page fragment identifier*

Identifies the destination that is targeted, *not* the link to the target

Only works *after* the link is clicked & the element is targeted, when the URL changes from `https://hpl.com/stories.html` to `https://hpl.com/stories.html#Dagon`

```
HTML Tidy x
1 <h3 id="toc">Table of Contents</h3>
2 <ul>
3 <li><a href="#the-tomb">The Tomb</a></li>
4 <li><a href="#dagon">Dagon</a></li>
5 <li><a href="#beyond-the-walls">Beyond the Walls of Sleep</a></li>
6 </ul>
7
8 <h3 id="the-tomb">The Tomb <a href="#toc">[TOC]</a></h3>
9 <p>In relating the circumstances which have led to my confinement...</p>
10
11 <h3 id="dagon">Dagon <a href="#toc">[TOC]</a></h3>
12 <p>I am writing this under an appreciable mental strain...</p>
13
```

```
CSS Tidy View Compiled x
```

```
JS Tidy x
```

Table of Contents

- [The Tomb](#)
- [Dagon](#)
- [Beyond the Walls of Sleep](#)

The Tomb [\[TOC\]](#)

In relating the circumstances which have led to my confinement...

Dagon [\[TOC\]](#)

I am writing this under an appreciable mental strain...

Beyond the Walls of Sleep [\[TOC\]](#)

I have frequently wondered if the majority of mankind...

```
HTML Tidy x
3 <li><a href="#the-tomb">The
  Tomb</a></li>
4 <li><a href="#dagon">Dagon</a>
  </li>
5 <li><a href="#beyond-the-
  walls">Beyond the Walls of
  Sleep</a></li>
6 </ul>
7
8 <h3 id="the-tomb">The Tomb <a
  href="#toc">[TOC]</a></h3>
9 <p>In relating the circumstances
  which have led to my confinement...
  </p>
```

```
CSS Tidy View Compiled x
1 h3:target {
2   background-color: black;
3   color: white;
4 }
5
6 h3:target a {
7   color: white;
8 }
```

```
JS Tidy x
```

Table of Contents

- [The Tomb](#)
- [Dagon](#)
- [Beyond the Walls of Sleep](#)

The Tomb [\[TOC\]](#)

In relating the circumstances which have led to my confinement...

Dagon [\[TOC\]](#)

I am writing this under an appreciable mental strain...

Beyond the Walls of Sleep [\[TOC\]](#)

I have frequently wondered if the majority of mankind...

Note CSS changes



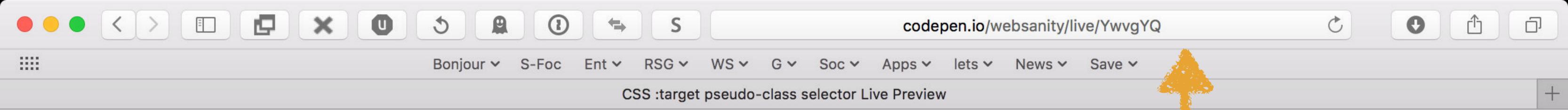


Table of Contents

- [The Tomb](#)
- [Dagon](#)
- [Beyond the Walls of Sleep](#)

When I click on this link...

No page fragment

The Tomb [\[TOC\]](#)

In relating the circumstances which have led to my confinement...

Dagon [\[TOC\]](#) ← ...this ID/URL fragment is targeted

I am writing this under an appreciable mental strain...

Beyond the Walls of Sleep [\[TOC\]](#)

I have frequently wondered if the majority of mankind...

Table of Contents

- [The Tomb](#)
- [Dagon](#)
- [Beyond the Walls of Sleep](#)


Page fragment

The Tomb [\[TOC\]](#)

In relating the circumstances which have led to my confinement...

Dagon [\[TOC\]](#)

I am writing this under an appreciable mental strain...

Beyond the Walls of Sleep [\[TOC\]](#)

I have frequently wondered if the majority of mankind...

CSS 4

`:scope`

Reference element pseudo-class that *matches any element that is a `:scope` element*, an element that is in a specific region of the document tree that a developer chooses

For now, `:scope` matches `:root`, which is the same as `<html>` (wider scoping may come to CSS in the future!)

Lots more useful with JavaScript where you can define scoped areas of a webpage

					ios		
<code>:any-link</code>	—	79	50	9	9	65	Y
<code>:link</code>	6	12	2	3.1	3.2	4	2.1
<code>:visited</code>	6	12	2	3.1	3.2	4	4.4
<code>:target</code>	9	12	2	3.1	3.2	4	2.1
<code>:scope</code>	—	79	32	7	7	27	Y

CSS 1, 2, 4

*User Action
Pseudo-Class Selectors*

Pointer hover: `:hover`

Input focus: `:focus`

Activation: `:active`

Focus container: `:focus-within`

`:hover`

Pointer hover pseudo class matches an element that has a *pointer over it*, so looks like 🖱️

Used with:

- » Dropdown menus
- » Image galleries
- » Anything!

On touch screens, `:hover` doesn't work*

* See compatibility chart for complications

:focus

Input focus pseudo class matches when element is *currently selected to receive input*

- » Mouse: object selected (`<input>` or `<textarea>`)
- » Keyboard: tab to object (usually indicated via `dotted` or solid outline)

Can only focus on interactive elements (how would you focus on `<p>`?)

`:active`

Activation pseudo class matches when element is *currently being activated* by the user

- » Mouse: time between mouse press & release
- » Keyboard: hold down Enter

Often used on `<a>` & `<button>`

PRO TIP

`:active` is overridden by any subsequent link-related pseudo-class (`:link`, `:hover`, or `:visited`)

To style links the way you probably want them, put link-related rules in this order:

```
:link  
:visited  
:hover  
:active
```

```
HTML
1 <p>
2   Click in this area & then press tab to see the
   <code>:focus</code> state. Then
   click & hold to see the
   <code>:active</code> state.
3 </p>
4
5 <button>
6   Click me and I'll invert!
7 </button>
```

```
CSS
1 button {
2   font-size: 24px;
3   border: 5px solid black;
4   padding: 10px 20px;
5 }
6 button:focus,
7 button:hover {
8   border-color: blue;
9   outline: none;
10 }
11 button:active {
12   color: white;
13   background-color: black;
14 }
```

```
JS
```

Click in this area & then press tab to see the `:focus` state. Then click & hold to see the `:active` state.

Click me and I'll invert!

```
HTML
1 <p>
2   Click in this area &amp; then
   press tab to see the
   <code>:focus</code> state. Then
   click &amp; hold to see the
   <code>:active</code> state.
3 </p>
4
5 <button>
6   Click me and I'll invert!
7 </button>
```

```
CSS
1 button {
2   font-size: 24px;
3   border: 5px solid black;
4   padding: 10px 20px;
5 }
6 button:focus,
7 button:hover {
8   border-color: blue;
9   outline: none;
10 }
11 button:active {
12   color: white;
13   background-color: black;
14 }
```

```
JS
```

Click in this area & then press tab to see the `:focus` state. Then click & hold to see the `:active` state.



```
HTML
1 <p>
2   Click in this area &amp; then
   press tab to see the
   <code>:focus</code> state. Then
   click &amp; hold to see the
   <code>:active</code> state.
3 </p>
4
5 <button>
6   Click me and I'll invert!
7 </button>
```

```
CSS
1 button {
2   font-size: 24px;
3   border: 5px solid black;
4   padding: 10px 20px;
5 }
6 button:focus,
7 button:hover {
8   border-color: blue;
9   outline: none;
10 }
11 button:active {
12   color: white;
13   background-color: black;
14 }
```

```
JS
```

Click in this area & then press tab to see the `:focus` state. Then click & hold to see the `:active` state.



`:focus-within`

Focus container pseudo-class matches *elements that contain a focused element*

HTML

```
1 <form>
2   <fieldset>
3     <legend>You Choose</legend>
4     <label for="old_one">Old
   One</label>
5     <input id="old_one"
   type="text">
6   </fieldset>
7 </form>
```

CSS (SCSS)

```
1 fieldset:focus-within {
2   background: yellow;
3 }
4
5 input:focus {
6   border: 3px solid blue;
7 }
8
9
```

JS

You Choose

Old One

Now tab to the text field or click in it...

```
HTML
1 <form>
2   <fieldset>
3     <legend>You Choose</legend>
4     <label for="old_one">Old
5     One</label>
6     <input id="old_one"
7     type="text">
```

```
CSS (SCSS)
1 *:focus-within {
2   border: 1px dashed red;
3 }
4
5 fieldset:focus-within {
6   background: yellow;
7 }
8
9 input:focus {
10  border: 3px solid blue;
11 }
```

```
JS
```



Notice the border around everything that could be selected by `:focus-within`

Notice the entire fieldset changes!

							
<code>:hover</code>	7*	12*	2	3.1	7.1.2†	4	Y
<code>:active</code>	6	12	2	3.1	3.2	4	2.1
<code>:focus</code>	8	12	2	3.1	3.2	4	2.1
<code>:focus-within</code>	—	79	52	10.1	10.3	60	Y

* Hovering over an element & then scrolling up or down without moving the pointer will leave it in `:hover` state until the pointer is moved; still a problem as of Edge 83 in July 2020

† As of iOS Safari 7.1.2, tapping a clickable element that has hidden sub-elements that are revealed on `:hover` causes it to enter `:hover` state, & it will remain in `:hover` until a 2nd click activates it; Safari 13 changes this to briefly show the `:hover` state while the click action is being resolved

*Tree-Structural
Pseudo-Class Selectors*

:root

:empty

CSS 3

`:root`

Matches the element that is the *root of the document*

In HTML, this is always `<html>`

Why does `:root` exist?

1

Remember, CSS can also be used with SVG & XML!

2

`:root` is equivalent to `<html>` in CSS, except that `:root` has a higher specificity

`:empty`

Matches elements that have *no children* (*children* means elements *or* text)

`:empty` matches `<td></td>` so you could apply styles to an empty table cell

It also matches empty elements like ``, `<input>`, & `<hr>`

				iOS		
<code>:root</code>	9	2	3.1	3.2	4	Y
<code>:empty</code>	9	2	3.1	3.2	4	2.1

*Tree-Structural
Child-Indexed
Pseudo-Class Selectors*

:first-child

:last-child

:only-child

:nth-child(n)

:nth-last-child(n)

`:first-child`

Matches the element that is *the 1st child of a parent*

`p:first-child` means a first `<p>` of all parent elements,
not the first child of `<p>`

Read it as “all `<p>`’s that are the first child of a parent”

`:last-child`

Matches the element that is the *last child of a parent*

`p:last-child` means a last `<p>` of all parent elements,
not the last child of `<p>`

Read it as “all `<p>`’s that are the last child of a parent”

`:only-child`

Matches the element that is the *only child of a parent*

`p:only-child` means a `<p>` that is the only child of its parent elements, *not* the only child of `<p>`

Read it as “all `<p>`’s that are the only child of a parent”

CSS 3

`:nth-child(n)`

Rendering engine *counts all children* of an element & then matches based on the selector

`n` can be:

- » a keyword (`odd` or `even`)
- » a number (like `1`)
- » a simple formula (like `2n` or `2n+1`)

`tr:nth-child(odd)`

Matches the *odd rows* of a table

`tr:nth-child(even)`

Matches the *even rows* of a table

`tr:nth-child(1)`

Matches the *1st row* of a table, same as the `:first-child` selector

Also same as `tr:nth-child(0n+1)`

`n` starts with `0` & increments up by `1`

If `n=0`, then $0 \times 0 + 1 = 1$, the 1st row

If `n=1`, then $0 \times 1 + 1 = 1$, the 1st row

If `n=2`, then $0 \times 2 + 1 = 1$, the 1st row

& so on

`tr:nth-child(2n+1)`

Matches the *odd rows* of a table

If $n=0$, then $2 \times 0 + 1 = 1$, the 1st row

If $n=1$, then $2 \times 1 + 1 = 3$, the 3rd row

If $n=2$, then $2 \times 2 + 1 = 5$, the 5th row

`tr:nth-child(2n)`

Matches the *even rows* of a table

If $n=0$, then $2 \times 0 = 0$, which is no row

If $n=1$, then $2 \times 1 = 2$, the 2nd row

If $n=2$, then $2 \times 2 = 4$, the 4th row

`tr:nth-child(-n+3)`

Matches the *1st 3 rows* of a table

If $n=0$, then $0+3=3$, the 3rd row

If $n=1$, then $-1+3=2$, the 2nd row

If $n=2$, then $-2+3=1$, the 1st row

If $n=3$, then $-3+3=0$, which is no row

Remember, the rendering engine counts *all children* of an element

All children get assigned a number based on order

The selector has to match the element (if specified) & the number based on order

HTML



```
1 <div>
2   <p>Lorem ipsum</p>
3   <p>Lorem ipsum</p>
4   <aside>Lorem ipsum</aside>
5   <p>Lorem ipsum</p>
6   <p>Lorem ipsum</p>
7 </div>
```

Lorem ipsum

Lorem ipsum

Lorem ipsum

Lorem ipsum

Lorem ipsum

CSS



```
1 p:nth-child(2n+1) {
2   color: red;
3   font-weight: bold;
4 }
5
6 /* Uninteresting stuff below here */
7
8 html {
9   font-size: 24px;
10  font-family: "Iowan Old Style",
    Georgia, serif;
11 }
```

JS



CSS 3

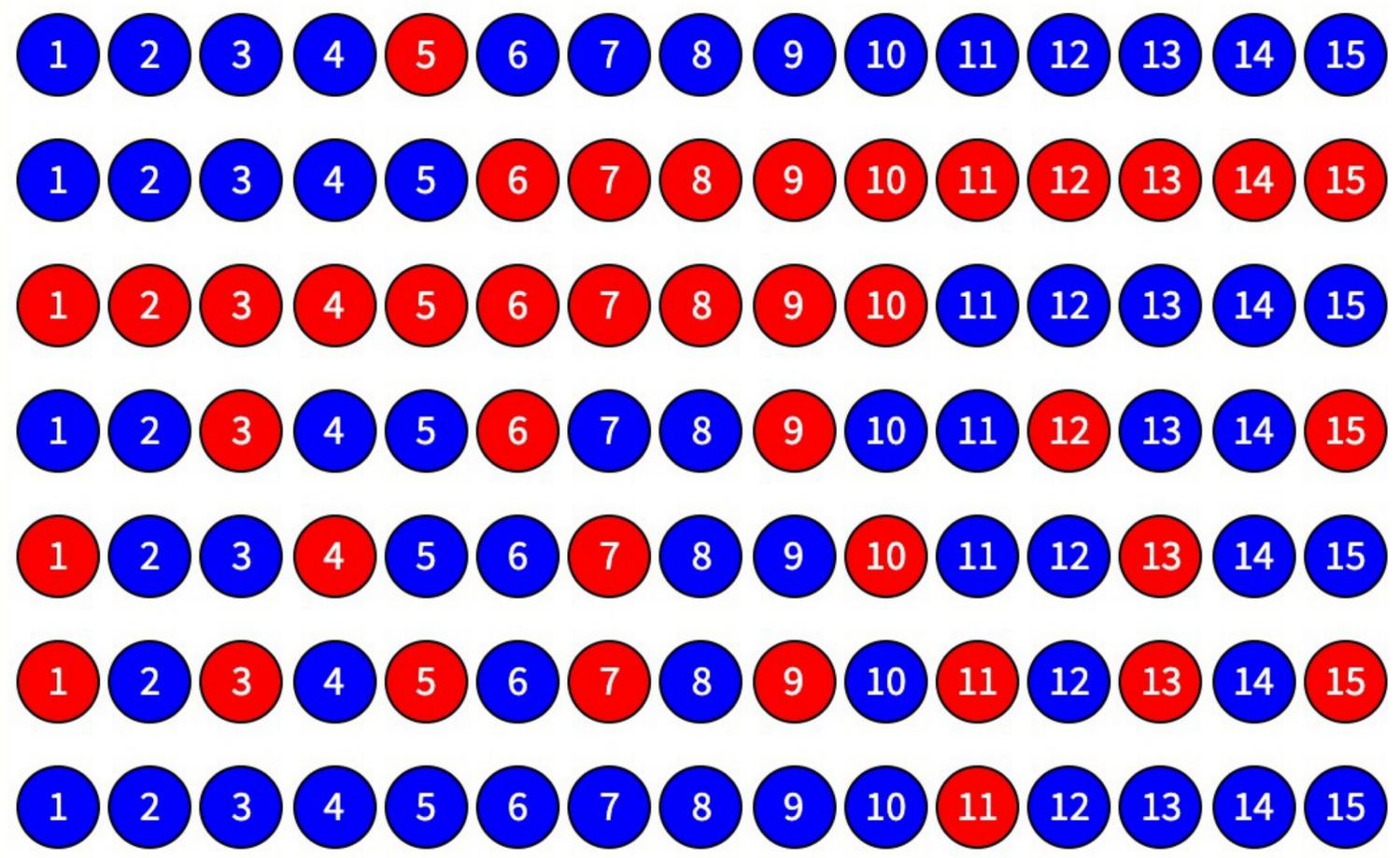
`:nth-last-child(n)`

Rendering engine counts all children of an element (children are counted starting with 1) & then matches based on the selector, *starting at the last element & working backwards*

`n` can be:

- » a keyword (`odd` or `even`)
- » a number (like `1`)
- » a simple formula (like `2n` or `2n+1`)

```
HTML
CSS
1 .only-fifth :nth-child(5) {
2   background-color: red;
3 }
4 .after-fifth > :nth-child(n+6) {
5   background-color: red;
6 }
7 .first-ten > :nth-child(-n+10) {
8   background-color: red;
9 }
10 .every-third > :nth-child(3n) {
11   background-color: red;
12 }
13 .first-then-every-third > :nth-child(3n+1) {
14   background-color: red;
15 }
16 .odd > :nth-child(odd) {
17   background-color: red;
18 }
19 .fifth-from-end :nth-last-child(5) {
20   background-color: red;
21 }
22
23
24
25
JS
```



				iOS		
<code>:first-child</code>	7	12	3.1	4	4	Y
<code>:last-child</code>	9	1	3.2	3.2	1	2.1
<code>:only-child</code>	9	1.5	3.1	3.1	2	2.1
<code>:nth-child(n)</code>	9	3.5	3.1	3.1	1	2.1
<code>:nth-last-child(n)</code>	9	3.5	3.2	3.2	4	2.1

*Tree-Structural
Typed Child-Indexed
Pseudo-Class Selectors*

`:nth-of-type(n)`

`:nth-last-of-type(n)`

`:first-of-type`

`:last-of-type`

`:only-of-type`

`:nth-of-type(n)`

Similar to `:nth-child(n)`, but it *matches a specific element*, no matter where it is inside the parent

```
HTML
1 <h1>Colors of Horror</h1>
2 <p>Cthulhu is green</p>
3 <hr>
4 <p>Hastur is yellow</p>
5 <hr>
6 <p>Cthulhu is green</p>
7
```

```
CSS
1 p:nth-of-type(2n+1) {
2   color: lime;
3 }
4
5 p:nth-of-type(2n) {
6   color: yellow;
7 }
8
```

```
JS
```

Colors of Horror

Cthulhu is green

Hastur is yellow

Cthulhu is green

				ios		
<code>:nth-of-type(n)</code>	9	3.5	3.1	3.1	1	2.1
<code>:nth-last-of-type(n)</code>	9	3.5	3.2	3.2	4	2.1
<code>:first-of-type</code>	9	3.5	3.2	3.2	1	2.1
<code>:last-of-type</code>	9	3.5	3.2	3.2	4	2.1
<code>:only-of-type</code>	9	3.5	3.2	3.2	1	2.1

Logical Combinations
Pseudo-Class Selectors

Matches-Any: `:is()`

Specificity-adjustment: `:where()`

Negation (Matches-None): `:not()`

Relational: `:has()`

```
:is(s1, s2, s3)
```

`()` contains a list of selectors, so `:is()` matches any element that matches any one of the selectors in the list

Very useful for writing lots of selectors in a compact way, without having to write out all the combinations manually as separate selectors (yes, this is similar to Sass nesting)

```
HTML
CSS
1 a {
2   color: dodgerblue;
3   text-decoration: none;
4 }
5
6 /* a:hover, a:focus, a:active */
7 a:is(:hover,:focus,:active) {
8   color: orange;
9   text-decoration: underline;
10 }
11
12 /* ul li, ol li */
13 :is(ul,ol) li {
14   margin-bottom: 1em;
15 }
16
17
JS
```

- Outside the ordered universe is that amorphous blight of nethermost confusion which blasphemes and bubbles at the center of all infinity—the boundless daemon sultan **Azathoth**, whose name no lips dare speak aloud, and who gnaws hungrily in inconceivable, unlighted chambers beyond time and space amidst the muffled, maddening beating of vile drums and the thin monotonous whine of accursed flutes.
- Don't fail to see **Nyarlahotep** if he comes to Providence. He is horrible—horrible beyond anything you can imagine—but wonderful. He haunts one for hours afterwards. I am still shuddering at what he showed.
- Iä! **Shub-Niggurath**! Iä! Shub-Niggurath! The Black Goat of the Woods with a Thousand Young!

pseudo classes `:visited` and `:link` are not supported
inside `:is()` — Safari



SIDE NOTE

`:is()` was formerly known as `:matches()` & `:any()`

:where()

What's the difference between `:is()` & `:where()`?

`:where()` always has 0 specificity

`:is()` takes on the specificity of the most specific selector in its arguments; e.g., `:is(p, .foo, .bar.baz)` would take 2, 0 as its specificity because `.bar.baz` is the most specific

`:not(s)`

Negation (Matches-None) pseudo-class that matches any element *not represented by* `s`

`s` is any selector that does *not* contain another negation selector or any pseudo-elements

HTML

```
1 <h3>Great Old Ones</h3>
2
3 <p class="great-old-one">Cthulhu</p>
4
5 <p>Charles Dexter Ward</p>
6
7 <p class="great-old-one">Tsathoggua</p>
8
9 <p class="great-old-one">Cthugha</p>
10
11 <p class="great-old-one">Ghatanothoa</p>
```

CSS

```
1 p:not(.great-old-one) {
2   color: red;
3 }
4
5 /* Uninteresting stuff below here */
6
7 html {
8   font-size: 24px;
9   font-family: "Iowan Old Style",
10  Georgia, serif;
11 }
```

JS

Great Old Ones

Cthulhu

Charles Dexter Ward

Tsathoggua

Cthugha

Ghatanothoa

This is an academic, not
a real-world example!

```
HTML
1 <h2>The Call of Cthulhu <small>By
  H.P. Lovecraft</small></h2>
2 <p>
3   Three of the Emma's men,
  including Capt. Collins and First
  Mate Green, were killed; and the
  remaining eight under Second Mate
  Johansen proceeded to navigate the
  captured yacht, going ahead in
  their original direction to see if
  any reason for their ordering back
  had existed.
4 </p>
5 <p>
6   The next day, it appears, they
  raised and landed on a small
  island, although none is known to
  exist in that part of the ocean;
  and six of the men somehow died
  ashore, though Johansen is queerly
  reticent about this part of his
  story, and speaks only of their
```

The Call of Cthulhu

By H.P. Lovecraft

Three of the Emma's men, including Capt. Collins and First Mate Green, were killed; and the remaining eight under Second Mate Johansen proceeded to navigate the captured yacht, going ahead in their original direction to see if any reason for their ordering back had existed.

The next day, it appears, they raised and landed on a small island, although none is known to exist in that part of the ocean; and six of the men somehow died ashore, though Johansen is queerly reticent about this part of his story, and speaks only of their falling into a rock chasm.

Later, it seems, he and one companion boarded the yacht and tried to manage her, but were beaten about by the storm of April 2nd.

From that time till his rescue on the 12th the man

```
HTML
1 <h2>The Call of Cthulhu <small>By
  H.P. Lovecraft</small></h2>
2 <p>
3   Three of the Emma's men,
  including Capt. Collins and First
  Mate Green, were killed; and the
CSS
1 p:not(:first-of-type) {
2   text-indent: 2em;
3 }
4
5 /*
6 Negation pseudo-class allows one
  selector instead of two.
7
8 p {
9   text-indent: 2em;
10 }
11
12 p:first-of-type {
13   text-indent: 0;
14 }
15 */
JS
```

The Call of Cthulhu

By H.P. Lovecraft

Three of the Emma's men, including Capt. Collins and First Mate Green, were killed; and the remaining eight under Second Mate Johansen proceeded to navigate the captured yacht, going ahead in their original direction to see if any reason for their ordering back had existed.

The next day, it appears, they raised and landed on a small island, although none is known to exist in that part of the ocean; and six of the men somehow died ashore, though Johansen is queerly reticent about this part of his story, and speaks only of their falling into a rock chasm.

Later, it seems, he and one companion boarded the yacht and tried to manage her, but were beaten about by the storm of April 2nd.

From that time till his rescue on the 12th the man

```
HTML [Settings] [Dropdown]
1 <menu>
2   <li><a href="#😊" >Blog</a></li>
3   <li><a href="#😊" >About</a></li>
4   <li><a href="#😊" >Videos</a></li>
5   <li><a href="#😊" >Music</a></li>
6   <li><a href="#😊" >Merch</a></li>
7 </menu>
```

```
* CSS [Settings] [Dropdown]
1 menu {
2   display: flex;
3 }
4
5 menu li {
6   display: block;
7 }
8
9 menu li:not(:last-child)::after {
10  content: "|";
11  margin-left: .5em;
12  margin-right: .5em;
13  color: orange;
14 }
```

```
JS [Settings] [Dropdown]
```

[Blog](#) | [About](#) | [Videos](#) | [Music](#) | [Merch](#)



SIDE NOTE

Keep in mind that as of 2021-07-17, Safari does not support pseudo classes `:visited` and `:link` inside `:not()`

CSS Selectors Level 4 (still in Working Draft)
allows `:not()` to accept a list of selectors

Instead of `:not(a):not(.b):not([c])` (which is valid!)
you can instead use `:not(a, .b, [c])`

Currently support is bad, however

:has()



SIDE NOTE

Before `:has()` syntax was adopted, the previous syntax was `main a < img`, but there a big problem: the key selector — `a` — wasn't on the right like every other key selector was

The new syntax — `main a:has(> img)` — solved the problem by conforming to the syntax of all other key selectors, as now the whole thing is the `a` with a pseudo-class as part of it



ios



<code>:is()</code>	—	88	78	14	14	88	88
<code>:where()</code>	—	88	78	14	14	88	88
<code>:not()</code>	9	12	3.5	3.2	3.2	4	2.1
<code>:not(s,s)</code>	—	—	—	9	9.2	—	—
<code>:has()</code>	—	—	—	—	—	—	—

As of 2021-06-17

CSS 4 introduces new time-dimensional pseudo-classes

- » `:current`
- » `:past`
- » `:future`

Not currently supported, but they will be covered as they develop (if ever)

CSS 4 introduces new grid-structural pseudo-classes

- » `:column(selector)`
- » ~~`:nth-column(n)`~~ `:nth-col(n)`
- » ~~`:nth-last-column(n)`~~ `:nth-last-col(n)`

Not currently supported, but they will be covered as they develop (if ever)

Pseudo-Elements

`:foo`

indicates a *pseudo-class*, which is used to style an element *based on its current state*

`::foo`

indicates a *pseudo-element*, which is used to style a *specific part of an element*, e.g., `::first-line`

```
h2::before {
  content: "Chapter 1: ";
}
/* Uninteresting stuff below here */
JS Options
```

Chapter 1: A Result and a

h2::before 191.22 x 49

Console Assets Comments 1.0x Last saved 13 DAYS ago Delete Add to Collection Fork Export

Elements Console Sources Network Performance Memory Application Security Lighthouse CSS Overview

Styles Computed Event Listeners DOM Breakpoints

```

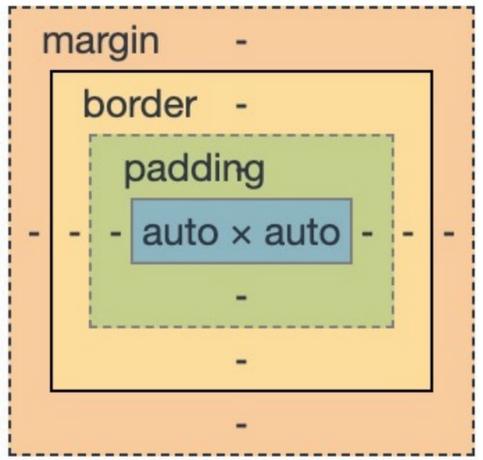
#document
  <!DOCTYPE html>
  <html lang="en" class stopthemadness-user-select="true">
    <head>...</head>
    <body>
      <h2>
        ::before == $0
        "A Result and a Prologue"
      </h2>
    
```

margin -
border -
padding -
auto x auto

Filter Show all

content display "Chapter 1: " inline

Note how the Inspector handles the pseudo-element in Brave



To inspect pseudo-elements fully in Chromium-based browsers, click on the cog in the upper right of the Inspector to open Inspector Preferences

Scroll down to the Elements section

✓ Show User Agent Shadow DOM



In CSS 2, `:` was used for both pseudo-classes & pseudo-elements

CSS 3 introduced `::` for pseudo-elements while leaving `:` for pseudo-classes

Most rendering engines support both, but you should stick to `::` for your pseudo-elements

::first-line
Pseudo-Element
Selector

`::first-line`

Selects *1st line* of an element

You are limited as to the properties you can use — see MDN for the complete list

HTML

```
1 <p>
2   When age fell upon the world, and wonder went out of
   the minds of men; when grey cities reared to smoky
   skies tall towers grim and ugly, in whose shadow
   none might dream of the sun or of Spring's flowering
   meads; when learning stripped the Earth of her
   mantle of beauty and poets asang no more of twisted
   phantoms seen with bleared and inward looking eyes;
   when these things had come to pass, and childish
   hopes had gone forever, there was a man who traveled
   out of life on a quest into spaces whither the
   worlds dreams had fled.
3 </p>
```

CSS

```
1 p::first-line {
2   text-transform: uppercase;
3   font-family: Creepster, cursive;
4   font-size: 1.1em;
5 }
6
7 /* Uninteresting stuff below */
8
9 html {
10  font-size: 24px;
11  font-family: "Iowan Old Style", Georgia, serif;
12 }
```

JS

WHEN AGE FELL UPON THE WORLD, AND WONDER WENT OUT
of the minds of men; when grey cities reared to
smoky skies tall towers grim and ugly, in whose
shadow none might dream of the sun or of
Spring's flowering meads; when learning stripped
the Earth of her mantle of beauty and poets asang
no more of twisted phantoms seen with bleared
and inward looking eyes; when these things had
come to pass, and childish hopes had gone forever,
there was a man who traveled out of life on a
quest into spaces whither the worlds dreams had
fled.

HTML

```
1 <p>
2   When age fell upon the world, and
   wonder went out of the minds of men;
   when grey cities reared to smoky skies
   tall towers grim and ugly, in whose
   shadow none might dream of the sun or
   of Spring's flowering meads; when
   learning stripped the Earth of her
   mantle of beauty and poets asang no
   more of twisted phantoms seen with
   bleared and inward looking eyes; when
   these things had come to pass, and
   childish hopes had gone forever, there
   was a man who traveled out of life on
   a quest into spaces whither the worlds
```

CSS

```
1 p::first-line {
2   text-transform: uppercase;
3   font-family: Creepster, cursive;
4   font-size: 1.1em;
5 }
6
7 /* Uninteresting stuff below */
8
9 html {
10  font-size: 24px;
11  font-family: "Iowan Old Style",
   Georgia, serif;
```

JS

WHEN AGE FELL UPON THE WORLD, AND WONDER WENT OUT OF THE MINDS OF
men; when grey cities reared to smoky skies tall towers grim
and ugly, in whose shadow none might dream of the sun or of
Spring's flowering meads; when learning stripped the Earth of
her mantle of beauty and poets asang no more of twisted
phantoms seen with bleared and inward looking eyes; when
these things had come to pass, and childish hopes had gone
forever, there was a man who traveled out of life on a quest into
spaces whither the worlds dreams had fled.

::first-letter

Pseudo-Element

Selector

```
::first-letter
```

Selects *1st letter* of the first line of text

Often used to create a “drop cap” effect

⚙ HTML



```
1 <p>
2   In my tortured ears there sounds
   unceasingly a nightmare whirring and
   flapping, and a faint distant baying as
   of some gigantic hound. It is not dream
   —it is not, I fear, even madness—for
   too much has already happened to give
   me these merciful doubts.
3 </p>
```

⚙ CSS



```
1 /* Create a drop cap */
2
3 p::first-letter {
4   text-transform: uppercase;
5   font-family: Creepster, cursive;
6   font-size: 4em;
7   padding: 10px 10px 0 0;
8   float: left;
9 }
10
11 /* Uninteresting stuff below */
12
```

⚙ JS



In my tortured ears there sounds unceasingly a nightmare whirring and flapping, and a faint distant baying as of some gigantic hound. It is not dream—it is not, I fear, even madness—for too much has already happened to give me these merciful doubts.

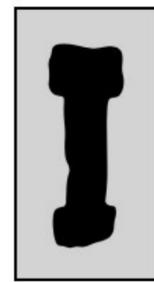
HTML

```
1 <p>
2   In my tortured ears there sounds unceasingly a
   nightmare whirring and flapping, and a faint distant
   baying as of some gigantic hound. It is
   not dream—it is not, I fear, even madness—for too
   much has already happened to give me these
   merciful doubts.
3 </p>
```

CSS

```
1 /* Create a drop cap */
2
3 p::first-letter {
4   text-transform: uppercase;
5   font-family: Creepster, cursive;
6   font-size: 96px;
7   line-height: 1;
8   padding: 10px 10px 0 0;
9   float: left;
10 /* Uncomment below for boxed drop cap;
11    note later padding wins over earlier padding */
12 border: 1px solid black;
13 padding: 0 10px 0 10px;
14 margin: 10px 10px 10px 0;
15 background-color: lightgray;
16 }
```

JS



In my tortured ears there sounds unceasingly a nightmare whirring and flapping, and a faint distant baying as of some gigantic hound. It is not dream—it is not, I fear, even madness—for too much has already happened to give me these merciful doubts.

<string> Data Type

Data types specify which kinds of values are allowed for CSS properties

`<string>`

Represents a quoted string of Unicode characters

Quotation marks can be either `"foobar"` or `'foobar'`

Used with:

- » data URIs
- » `::before`
- » `::after`

				iOS		
<code><string></code>	<5	1	1	Y	1	Y

::before

*Pseudo-Element
Selector*

`::before`

Selects element & *inserts content before it*

Content can be text, images, or *counters*

Used to be `:before`, & most rendering engines support both

Must include the `content` property!

HTML

```
1 <h2>A Result and a Prologue</h2>
```

CSS

```
1 h2::before {  
2   content: "Chapter 1: ";  
3 }  
4  
5 /* Uninteresting stuff below here */  
6  
7 html {  
8   font-size: 24px;  
9   font-family: "Iowan Old Style",  
10  Georgia, serif;  
11 }
```

JS

Chapter 1: A Result and a Prologue

We'll improve this in *Tables & Lists*

HTML

```
1 <menu>
2   <li><a href="X">Search</a></li>
3   <li><a href="X">Log In</a></li>
4   <li><a href="X">Shopping
   Cart</a></li>
5 </menu>
```

CSS (SCSS)

```
1 // All but the first <li> get a
   pipe
2
3 menu li:not(:first-of-type)::before
4 {
5   content: "|";
6 }
7
8
9
10
11
```

JS

Search | Log In | Shopping Cart

Values for `content`:

- » `string`: `<string>`
- » `image`: `<url>` or `<gradient>`
- » `alt text`: `url("<url>") / "<string>"`
- » `counter`: `<counter>()`
- » `attribute value`: `attr()`
- » `keywords`: `open-quote`, `close-quote`, `no-open-quote`, `no-close-quote`
- » `several values together`: `open-quote`
`chapter_counter`

`open-quote`
`close-quote`

Replaced by the appropriate string from the `quotes` property

Default depends on the language value currently set on the selected elements

```
HTML
4
5 <blockquote>
6   <p>
7     What I heard in my youth about the
      shunned house was merely that people
      died there in alarmingly great numbers.
      That, I was told, was why the original
      owners had moved out some twenty years
      after building the place. It was
      plainly unhealthy, perhaps because of
      the dampness and fungous growth in the
      cellar, the general sickish smell, the
      draughts of the hallways, or the
      quality of the well and pump water.
8   </p>
9 </blockquote>
10
11
```

```
CSS (SCSS) Compiled
1 <blockquote p::before {
2   content: open-quote;
3   color: #333;
4   font-size: 3em;
5   line-height: 0;
6   vertical-align: -0.4em;
7 }
```

```
JS
```

From H.P. Lovecraft's "The Shunned House":

“ What I heard in my youth about the shunned house was merely that people died there in alarmingly great numbers. That, I was told, was why the original owners had moved out some twenty years after building the place. It was plainly unhealthy, perhaps because of the dampness and fungous growth in the cellar, the general sickish smell, the draughts of the hallways, or the quality of the well and pump water.

no-open-quote
no-close-quote

Doesn't insert quotation marks; instead, increments or decrements the level of nesting for quotes

```
HTML 36 unsaved changes x
5 </blockquote>
6 <p>
7   As he copied the formula he
  finally chose, Dr. Armitage looked
  involuntarily over his shoulder at
  the open pages; the left-hand one
  of which, in the Latin version,
  contained such monstrous threats to
  the peace and sanity of the world.
8 </p>
9 <blockquote>
10 <p>
11   “Nor is it to be thought,”
  ran the text as Armitage mentally
  translated it, “that man is either
  the oldest or the last of earth’s
  masters, or that the common bulk of
  life and substance walks alone. The
  Old Ones were, the Old Ones are,
  and the Old Ones shall be. Not in
  the spaces we know, but
  <i>between</i> them, They walk
  serene and primal, undimensioned
  and to us unseen.
12 </p>
13 </blockquote>
CSS (SCSS)
JS
```

From H.P. Lovecraft’s “The Dunwich Horror”:

“As he copied the formula he finally chose, Dr. Armitage looked involuntarily over his shoulder at the open pages; the left-hand one of which, in the Latin version, contained such monstrous threats to the peace and sanity of the world.

“Nor is it to be thought,” ran the text as Armitage mentally translated it, “that man is either the oldest or the last of earth’s masters, or that the common bulk of life and substance walks alone. The Old Ones were, the Old Ones are, and the Old Ones shall be. Not in the spaces we know, but *between* them, They walk serene and primal, undimensioned and to us unseen.

HTML

```
5 <blockquote>
6   <p>
7     As he copied the formula he
      finally chose, Dr. Armitage looked
      involuntarily over his shoulder at
      the open pages; the left-hand one
      of which, in the Latin version,
      contained such monstrous threats to
      the peace and sanity of the world.
8   </p>
9 </blockquote>
10 <p>
11   “Nor is it to be thought,”
      ran the text as Armitage mentally
      translated it, “that man is either
```

CSS (SCSS) Compiled

```
1 blockquote > p::before {
2   content: open-quote;
3   color: #333;
4   font-size: 3em;
5   line-height: 0;
6   vertical-align: -0.4em;
7 }
8 blockquote blockquote p::before {
9   content: no-open-quote;
10 }
```

JS

From H.P. Lovecraft’s “The Dunwich Horror”:

“ As he copied the formula he finally chose, Dr. Armitage looked involuntarily over his shoulder at the open pages; the left-hand one of which, in the Latin version, contained such monstrous threats to the peace and sanity of the world.

“Nor is it to be thought,” ran the text as Armitage mentally translated it, “that man is either the oldest or the last of earth’s masters, or that the common bulk of life and substance walks alone. The Old Ones were, the Old Ones are, and the Old Ones shall be. Not in the spaces we know, but *between* them, They walk serene and primal, undimensioned and to us unseen.

::after

*Pseudo-Element
Selector*

`::after`

Selects element & *inserts content after it*

Content can be text, images, or *counters*

Used to be `:after`, & most rendering engines support both

Must include the `content` property!

HTML



```
href="http://miskatonic-  
expedition.com">join our expedition  
to the Antarctic</a>!
```

```
5 </p>
```

```
7 <h3>With ::after</h3>
```

```
9 <p>
```

```
10 Sign up today to <a  
href="http://miskatonic-  
expedition.com">join our expedition  
to the Antarctic</a>!
```

```
11 </p>
```

CSS



```
1 a::after {content:" (" attr(href) ")"}  
2 /* Insert an open parentheses, then the  
value of the href attribute, & then a  
close parentheses */
```

```
4 /* Uninteresting stuff below here */
```

```
6 html {  
7   font-size: 24px;  
8   font-family: "Iowan Old Style",  
Georgia, serif;
```

```
9 }
```

JS



Without `::after`

Sign up today to [join our expedition to the Antarctic!](http://miskatonic-expedition.com)

With `::after`

Sign up today to [join our expedition to the Antarctic
\(http://miskatonic-expedition.com\)!](http://miskatonic-expedition.com)

What about images?



Cascading Style Sheets

From Wikipedia, the free encyclopedia

"CSS" redirects here. For other uses, see [CSS \(disambiguation\)](#).

For the use of CSS on Wikipedia, see [Help:Cascading style sheets](#).

Cascading Style Sheets (CSS) is a [style sheet language](#) used for describing the [look and formatting](#) of a document written in a [markup language](#). While most often used to style [web pages](#) and user interfaces written in [HTML](#) and [XHTML](#), the language can be applied to any kind of [XML](#) document, including [plain XML](#), [SVG](#) and [XUL](#). CSS is a cornerstone specification of [the web](#) and almost all web pages use CSS style sheets to describe their presentation.

CSS is designed primarily to enable [the separation of document content from document presentation](#), including elements such as the [layout](#), [colors](#), and [fonts](#).^[1] This separation can improve content [accessibility](#), provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for [tableless web design](#)). It [obviates](#) those portions of markup that would specify presentation by instead providing that information in a separate file. For each relevant [HTML element](#) (identified by tags), it provides a list of formatting instructions. For example, it might say (in CSS syntax), "All heading 1 elements should be [bold](#)." Therefore, no formatting markup such as bold tags (``) is needed within the content; what is needed is simply semantic markup saying, "this text is a level 1 heading."

CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or [screen reader](#)) and on [Braille-based](#), tactile devices. It can also be used to allow the web page to display differently depending on the screen size or device on which it is being viewed. While the author of a document typically links that document to a CSS file, readers can use a different style sheet, perhaps one on their own computer, to override the one

Cascading Style Sheets (CSS)

Filename extension	.css
Internet media type	text/css
Developed by	Håkon Wium Lie · Bert Bos · World Wide Web Consortium
Initial release	December 17, 1996; 17 years ago
Type of format	Style sheet language
Standards	Level 1 (Recommendation) Level 2 (ditto) Level 2 Revision 1 (ditto)

Cascading Style Sheets

- [Style sheet](#)
- [CSS Zen Garden](#)
- [The Zen of CSS Design](#)
- [CSSTidy](#)
- [Internet Explorer box model bug](#)

Comparisons

- [Main page](#)
- [Contents](#)
- [Featured content](#)
- [Current events](#)
- [Random article](#)
- [Donate to Wikipedia](#)
- [Wikimedia Shop](#)

Interaction

- [Help](#)
- [About Wikipedia](#)
- [Community portal](#)
- [Recent changes](#)
- [Contact page](#)

Tools

- [What links here](#)
- [Related changes](#)
- [Upload file](#)
- [Special pages](#)
- [Permanent link](#)
- [Page information](#)
- [Wikidata item](#)
- [Cite this page](#)

Print/export

- [Create a book](#)
- [Download as PDF](#)

See also [edit]

- [Acid3](#)
- [Comparison of layout engines \(CSS\)](#)
- [Comparison of stylesheet languages](#)
- [CSS Zen Garden](#)
- [CSSTidy](#)
- [List of stylesheet languages](#)

- [Minification](#)
- [Progressive enhancement](#)
- [Responsive web design](#)
- [Validator](#)
- [X resources](#)

References [edit]

- ↑ "[What is CSS?](#)" [↗]. World Wide Web Consortium. Retrieved December 2010.
- ↑ "[W3C CSS validation service](#)" [↗].
- ↑ "[W3C CSS2.1 specification for pseudo-elements and pseudo-classes](#)" [↗]. World Wide Web Consortium. 7 June 2011. Retrieved 30 April 2012.
- ↑ see [the complete definition of selectors at the W3C Web site](#) [↗].
- ↑ "[Selectors Level 3](#)" [↗]. W3.org. Retrieved 2014-05-30.
- ↑ "[W3C CSS2.1 specification for rule sets, declaration blocks, and selectors](#)" [↗]. World Wide Web Consortium. 7 June 2011. Retrieved 2009-06-20.
- ↑ "[Full property table](#)" [↗]. W3.org. Retrieved 2014-05-30.
- ↑ animation-name (2014-04-30). "[CSS Legal Color Values](#)" [↗]. W3schools.com. Retrieved 2014-05-30.
- ↑ "[HTML 5. A vocabulary and associated APIs for HTML and XHTML](#)" [↗]. World Wide Web Consortium. Retrieved 28 June 2014.
- ↑ ^{*a*} ^{*b*} Meyer, Eric A. (2006). *Cascading Style Sheets: The Definitive Guide* [↗] (3rd Edition ed.). O'Reilly Media, Inc. ISBN 0-596-52733-0.
- ↑ [Assigning property values, Cascading, and Inheritance](#) [↗]
- ↑ Bos, Håkon Wium Lie, Bert (1999). *Cascading style sheets : designing for the Web* (2nd ed. ed.). Harlow, Essex, England: Addison-Wesley. ISBN 0-201-59625-3.
- ↑ Isakowitz, Tomas; Edward A. Stohr, and P. Balasubramanian "BMM: A
- ↑ Nitot, Tristan (18 March 2002). "[Incorrect MIME Type for CSS Files](#)" [↗]. *Mozilla Developer Center*. Mozilla. Retrieved 20 June 2010.
- ↑ Bos, / Håkon Wium Lie, Bert (1997). *Cascading style sheets : designing for the Web* (1st print. ed.). Harlow, England ; Reading, MA.: Addison Wesley Longman. ISBN 0-201-41998-X.
- ↑ [W3C: Cascading Style Sheets, level 1](#) [↗] CSS 1 specification
- ↑ [W3C: Cascading Style Sheets level 1 specification](#) [↗] CSS level 1 specification
- ↑ [W3C: Cascading Style Sheets, level 2](#) [↗] CSS 2 specification (1998 recommendation)
- ↑ [W3C: Cascading Style Sheets, level 2 revision 1](#) [↗] CSS 2.1 specification (W3C Proposed Recommendation)
- ↑ [W3C: Cascading Style Sheets Standard Boasts Unprecedented Interoperability](#) [↗]
- ↑ ^{*a*} ^{*b*} Bos, Bert (18 February 2011). "[Descriptions of all CSS specifications](#)" [↗]. World Wide Web Consortium. Retrieved 3 March 2011.
- ↑ Bos, Bert (26 February 2011). "[CSS current work](#)" [↗]. World Wide Web Consortium. Retrieved 3 March 2011.
- ↑ [Etemad, Erika](#) (12 December 2010). "[Cascading Style Sheets \(CSS\) Snapshot 2010](#)" [↗]. World Wide Web Consortium. Retrieved 3 March 2011.
- ↑ "[All CSS specifications](#)" [↗]. W3.org. 2014-05-22. Retrieved 2014-05-30.
- ↑ Atkins Jr, Tab. "[A Word About CSS4](#)" [↗]. Retrieved 18 October 2012.

HTML



```
1 <p>
2   Sign up today to <a href="http://miskatonic-
   expedition.com">join our expedition to the
   Antarctic</a>. Flights available on <a
   href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>!
3 </p>
```

CSS



```
1 a.flight::after {content: url(https://s3-us-west-
2 .amazonaws.com/s.cdpn.io/122116/airplane-icon.png) }
3
4 /* Insert an icon */
5
6 /* Uninteresting stuff below here */
7
8 html {
9   font-size: 24px;
10  font-family: "Iowan Old Style", Georgia, serif;
11 }
```

JS



Sign up today to [join our expedition to the Antarctic](http://miskatonic-expedition.com). Flights available on [Cthulhu Airlines](http://cthulhu-airlines.com) !

Instead of images you can use Unicode

Unicode Character Search

Unicode Character Search

Query:

include Han codepoints?

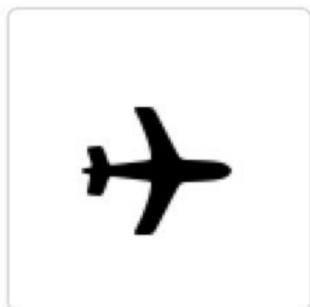
Preview:

No

HTML Entities

[A-Z index](#) | [Search options](#)

Unicode Character 'AIRPLANE' (U+2708)



[Browser Test Page](#)
[Outline \(as SVG file\)](#)
[Fonts that support U+2708](#)

Unicode Data

Name AIRPLANE

Block [Dingbats](#)

Category [Symbol, Other \[So\]](#)

Combine 0

BIDI Other Neutrals [ON]

Mirror N

Index entries AIRPLANE

Version [Unicode 1.1.0 \(June, 1993\)](#)



HTML

```
1 <p>
2   Sign up today to <a href="http://miskatonic-
   expedition.com">join our expedition to the
   Antarctic</a>. Flights available on <a
   href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>!
3 </p>
```

CSS

```
1 a.flight::after {content:" \2708"}
2 /* Insert a Unicode character for "airplane" */
3
4 /* Uninteresting stuff below here */
5
6 html {
7   font-size: 24px;
8   font-family: "Iowan Old Style", Georgia,
   serif;
9 }
```

JS

Sign up today to [join our expedition to the Antarctic](http://miskatonic-expedition.com).
Flights available on [Cthulhu Airlines ✈](http://cthulhu-airlines.com)!

To insert Unicode in your CSS, look up the UTF-8 code point value (2708 for ✈) & enter that in with `\` (an *escape*) in front of it

				iOS		
<code>:before</code>	8	1	1.3	3.2	4	2.1
<code>::before</code>	9	1.5	3.1	5.1	4	2.1
<code>:after</code>	8	1	1.3	3.2	4	2.1
<code>::after</code>	9	1.5	3.1	5.1	4	2.1

Increasingly, `::before` & `::after` are used to *create boxes* that are used for layout

Bootstrap, for instance, uses this commonly

However, there are some limitations...



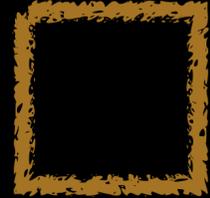
Inline box created by `::before`

Block box created by `<p>`

Inline box created by `::after`



Boxes that cannot be created with CSS



Inline box created by `::before`

Block box created by `<p>`

Inline box created by `::after`



You can combine pseudo-elements with the user action pseudo-classes

However, order matters!

`p::first-line:hover` only matches if the 1st line is hovered

`p:hover::first-line` matches the 1st line if any part of the `<p>` is hovered

Attributes

Attribute selectors allow you to select a particular element based on attribute conditions

Existence	<code>element[attribute]</code>	CSS 2
Equality	<code>element[attribute="value"]</code>	CSS 2
Space	<code>element[attribute~="value"]</code>	CSS 2
Hyphen	<code>element[attribute ="value"]</code>	CSS 2
Prefix	<code>element[attribute^="value"]</code>	CSS 3
Substring	<code>element[attribute*="value"]</code>	CSS 3
Suffix	<code>element[attribute\$="value"]</code>	CSS 3

```
<h1 align="center"> ... </h1>
```

`h1` is an *tag name* (but most people call it an *element*)

`align` is an *attribute*

`center` is a *value*

`align="center"` is an *attribute-value pair*

```
[title]
```

Simple attribute selector: Does element *foo* have attribute `title`?

It doesn't matter what the value is, just that it has the attribute

HTML



```
1 <p>
2   <a href="http://cthulhu.com">Cthulhu</a>
   invites you to join the <a
   href="http://www.miskatonic-expedition.com"
   title="Led by Pabodie, Lake, Atwood &
   Dyer">Miskatonic Expedition</a> to the
   Antarctic! Flights available on <span
   lang="pt">Gol Transportes Aéreos</span> & <a
   href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>.
3 </p>
```

CSS



```
1 a[title] {
2   font-weight: bold;
3 }
4
5 /* Uninteresting stuff below here */
6
7 html {
8   font-size: 24px;
9   font-family: "Iowan Old Style", Georgia,
   serif;
10 }
```

JS



[Cthulhu](#) invites you to join the [Miskatonic Expedition](#) to the Antarctic! Flights available on Gol Transportes Aéreos & [Cthulhu Airlines](#).

```
[title="Cthulhu"]
```

Attribute value selectors: Does element foo have the attribute `title` with the exact attribute value `Cthulhu` & *only* `Cthulhu`?

Exact value, so would not match `Cthulhu!`

HTML

```
1 <p>
2   <a href="http://cthulhu.com">Cthulhu</a>
   invites you to join the <a
   href="http://www.miskatonic-expedition.com"
   title="Led by Pabodie, Lake, Atwood &
   Dyer">Miskatonic Expedition</a> to the
   Antarctic! Flights available on <span
   lang="pt">Gol Transportes Aéreos</span> & <a
   href="http://cthulhu-airlines.com">Cthulhu
   Airlines</a>.
3 </p>
```

CSS

```
1 a[href="http://cthulhu-airlines.com"]::after {
2   content: url(https://s3-us-west-
   2.amazonaws.com/s.cdpn.io/122116/airplane-icon-
   24.png);
3   margin-left: .2em;
4 }
5
```

JS

[Cthulhu](#) invites you to join the [Miskatonic Expedition](#) to the Antarctic! Flights available on Gol Transportes Aéreos & [Cthulhu Airlines](#) ✈️.

HTML



```
1 <p>
2   <a href="http://cthulhu.com">Cthulhu</a>
   invites you to join the <a
   href="http://www.miskatonic-expedition.com"
   title="Led by Pabodie, Lake, Atwood &
   Dyer">Miskatonic Expedition</a> to the
   Antarctic! Flights available on <span
   lang="pt">Gol Transportes Aéreos</span> & <a
   href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>.
3 </p>
```

[Cthulhu](#) invites you to join the [Miskatonic Expedition](#) to the Antarctic! Flights available on Gol Transportes Aéreos & [Cthulhu Airlines](#)✈️.

What's wrong with this selector?

CSS



```
1 a[class="flight"]::after {
2   content: url(https://s3-us-west-
3   2.amazonaws.com/s.cdpn.io/122116/airplane-
4   icon-24.png)
5 }
6
7 /* Uninteresting stuff below here */
8
9 html {
10  font-size: 24px;
11  font-family: "Iowan Old Style", Georgia,
12  serif;
```

JS



Also useful when you want to override inline styles injected by a JavaScript library, like this:

```
<h1 style="color:blue">
```

Grrr... 

HTML

```
1 <h1 style="color:blue">This text is blue</h1>
```

This text is blue

CSS

```
1 h1[style="color:blue"] {  
2   color: green !important;  
3 }
```

```
[title~="R'lyeh"]
```

One-of-many (space-separated) attribute value selectors: Does element *foo* have the attribute `title` with a specific attribute value `R'lyeh` among a list of space separated values?

Space-separated, so `R'lyeh` matches `Cthulhu R'lyeh` & `R'lyeh Cthulhu`, but not `Cthulhu/R'lyeh`

HTML

```
1 <p>
2   <a href="http://cthulhu.com">Cthulhu</a>
   invites you to join the <a
   href="http://www.miskatonic-expedition.com"
   title="Led by Pabodie, Lake, Atwood &
   Dyer">Miskatonic Expedition</a> to the
   Antarctic! Flights available on <span
   lang="pt">Gol Transportes Aéreos</span> & <a
   href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>.
3 </p>
```

CSS

```
1 a[title~="Dyer"] {
2   cursor: url(https://s3-us-west-
   2.amazonaws.com/s.cdpn.io/122116/old-bearded-
   man.png), auto;
3 }
4 /* Note that Pabodie & Lake won't work because
   of the commas! */
5
6 /* Uninteresting stuff below here */
7
8 html {
9   font-size: 24px;
```

JS

[Cthulhu](#) invites you to join the [Miskatonic Expedition](#) to the Antarctic! Flights available on [Gol Transportes Aéreos](#) & [Cthulhu Airlines](#).



e, Lake, Atwood & Dyer

Jans messaged me one night at 11:34 PM:

I just used this selector...

```
[class]:not([class~="m-1v1"])
```

What is it selecting?

```
[ src |="headshot" ]
```

Hyphen-separated attribute value selectors: Does element *foo* have the attribute `src` with a specific attribute value `headshot` among a list of hyphen separated values?

Hyphen-separated, so `headshot` matches `cthulhu-headshot` but not `Cthulhu headshot`

Mostly used with languages (e.g., `en-US`, `en-UK`)

Can also be used with images, which often use hyphens in file names (e.g., `headshot-jones.jpg`, `headshot-smith.jpg`)

HTML



```
1 <p>
2   <a href="http://cthulhu.com">Cthulhu</a>
   invites you to join the <a
   href="http://www.miskatonic-expedition.com"
   title="Led by Pabodie, Lake, Atwood &
   Dyer">Miskatonic Expedition</a> to the
   Antarctic! Flights available on <span
   lang="pt-BR">Gol Transportes Aéreos</span> &
   <a href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>.
3 </p>
```

CSS



```
1  [_lang|"pt"] {
2    font-style: italic;
3  }
4
5  /* Uninteresting stuff below here */
6
7  html {
8    font-size: 24px;
9    font-family: "Iowan Old Style", Georgia,
   serif;
10 }
```

JS



[Cthulhu](#) invites you to join the [Miskatonic Expedition](#) to the Antarctic! Flights available on *Gol Transportes Aéreos* & [Cthulhu Airlines](#).

WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.

OH NO! THE KILLER MUST HAVE FOLLOWED HER ON VACATION!

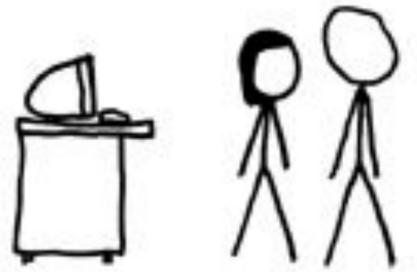


BUT TO FIND THEM WE'D HAVE TO SEARCH THROUGH 200 MB OF EMAILS LOOKING FOR SOMETHING FORMATTED LIKE AN ADDRESS!



IT'S HOPELESS!

EVERYBODY STAND BACK.



I KNOW REGULAR EXPRESSIONS.



CSS 3 introduces 3 new ways to select a particular element based on attribute conditions

If you know *regular expressions (regex)*, you should recognize these

What is *regex*?

Normal find & replace looks for a literal string of characters, & if found, replaces it; e.g.:

Find `cthulhu@lovecraft.com`

Replace with `hpl@lovecraft.com`

 PRO TIP

Regex, on the other hand, matches patterns

Find all misspellings of Jans' name: `[HJY][aeo]n+[sz]`

 PRO TIP

`[HJY][aeo]n+[sz]`

- » `[HJY]`: matches 1 `H`, `J`, or `Y`
- » `[aeo]`: matches 1 `a`, `e`, or `o`
- » `n+`: matches 1 or more `n`s
- » `[sz]`: matches 1 `s` or `z`

Will match Hans, Jenz, Jonnz, Yans...

 PRO TIP

Multiple spellings of the Jewish holiday: Chanuka, Chanukah, Chanukkah, Channukah, Hanukah, Hannukah, Hanukkah, Hanuka, Hanukka, Hanaka, Haneka, Hanika, Khanukkah

Find all spellings: `[CHK]h?ann?[aeiu]kk?ah?`

💡 PRO TIP

`[CHK]h?ann?[aeiu]kk?ah?`

- » `[CHK]`: matches 1 `C`, `H`, or `K`
- » `h?`: matches 0 or 1 `h`
- » `ann?`: look for `a` followed by `n` followed by 0 or 1 `n`
- » `[aeoi]`: matches 1 `a`, `e`, `i`, or `u`
- » `kk?`: matches `k` followed by 0 or 1 `k`
- » `ah?`: matches `a` followed by 0 or 1 `h`

Matches Chanuka, Chanukah, Chanukkah, Channukah,
Hanukah, Hannukah, Hanukkah, Hanuka, Hanukka,
Hanaka, Haneka, Hanika, Khanukkah



PRO TIP

Multiple spellings of the former Libyan leader's name:
Gadaffi, Gadafi, Gadafy, Gaddafi, Gaddafy, Gaddhafi,
Gadhafi, Gathafi, Ghadaffi, Ghadafi, Ghaddafi,
Ghaddafy, Gheddafi, Kadaffi, Kadafi, Kaddafi, Kadhafi,
Kazzafi, Khadaffy, Khadafy, Khaddafi, Qadafi, Qaddafi,
Qadhafi, Qadhhdhafi, Qadthafi, Qathafi, Quathafi,
Qudhafi, Kad'afi

Find all spellings: $(Kh? | Gh? | Qu?) [aeu] (d['dt]? | t |$
 $zz | dh) h? aff? [iy]$

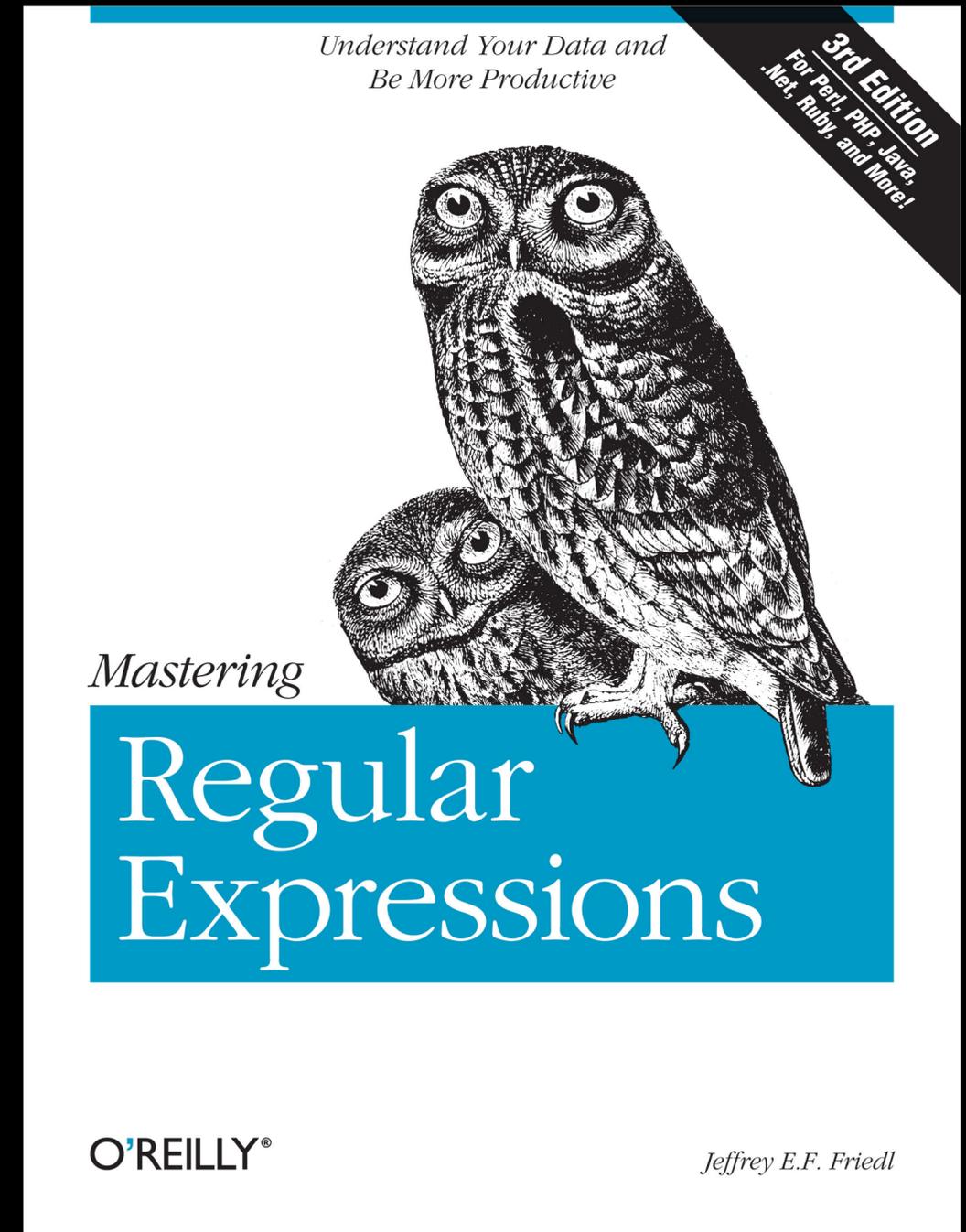
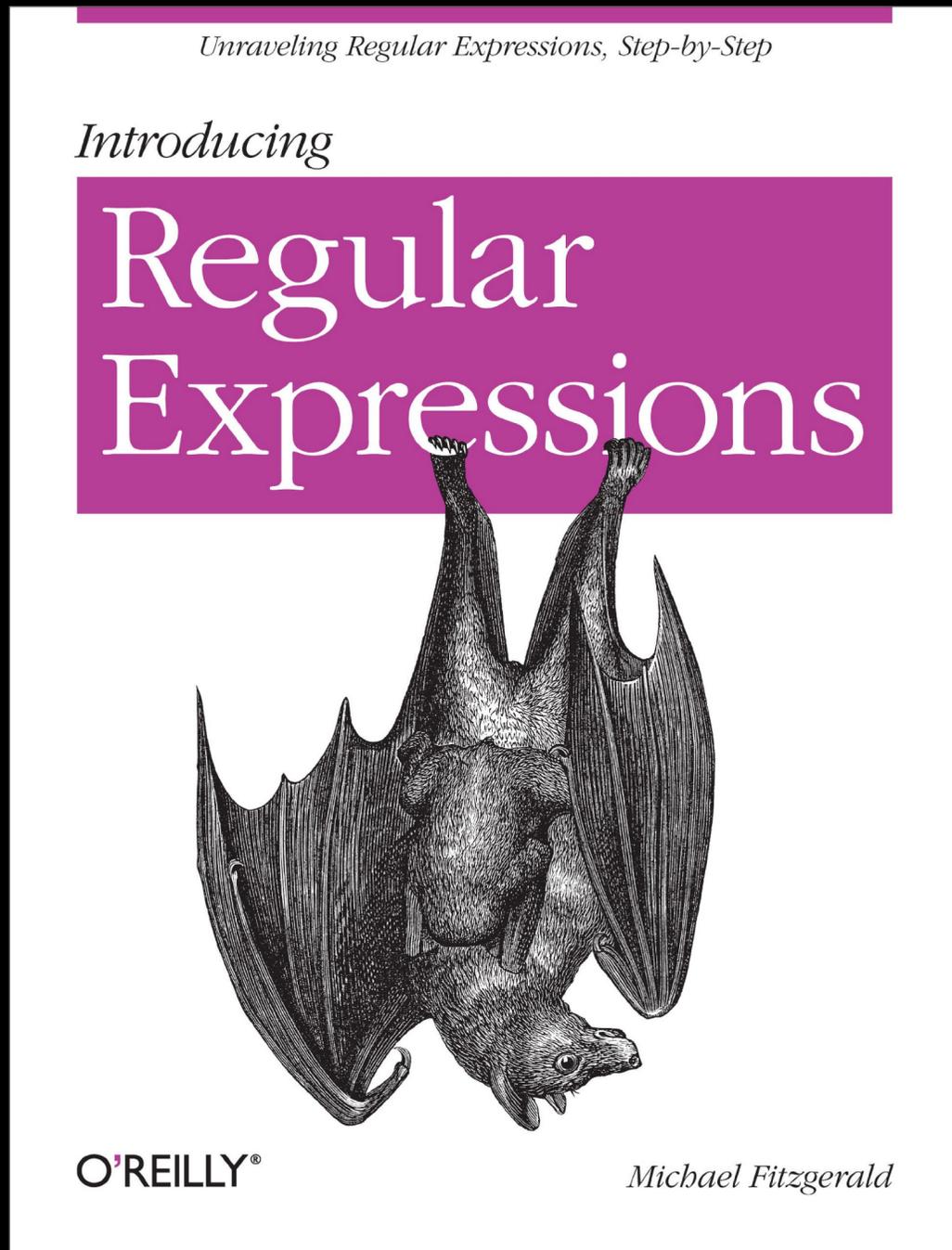
💡 PRO TIP

$(Kh? | Gh? | Qu?) [aeu] (d['dt]? | t | zz | dhd) h? aff? [iy]$

- » $(Kh? | Gh? | Qu?)$: Look for **K** followed by 0 or 1 **h** OR **G** followed by 0 or 1 **h** OR **Q** followed by 0 or 1 **u**
- » $[aeu]$: look for an **a**, **e**, or **u**
- » $(d['dt]? | t | zz | dhd)$: look for a **d** followed by 0 or 1 **'**, **d**, or **t** OR a **t** OR **zz** OR **dhd**
- » $h?$: look for 0 or 1 **h**
- » $aff?$: look for **a** followed by **f** followed by 0 or 1 **f**
- » $[iy]$: look for an **i** or a **y**

💡 PRO TIP

Find all email addresses: `[a-zA-Z0-9_]+(?:\.[A-Za-z0-9!#$%&'*/=?^_`{|}~-]+)*@(?!([a-zA-Z0-9]*\.[a-zA-Z0-9]*\.[a-zA-Z0-9]*\.)+)(?:[A-Za-z0-9](?:[a-zA-Z0-9-]*[A-Za-z0-9])?\.)+[a-zA-Z0-9](?:[a-zA-Z0-9-]*[a-zA-Z0-9])?`



Books to get

RegEx you need to know in order to understand these selectors

- ^ matches start of string or start of line; e.g., `^Young` would match `Young Americans` but not `All the Young Dudes`
- \$ matches end of string or end of line; e.g., `Rock$` would match `Black Country Rock` but not `Rock 'n' Roll Suicide`
- * matches zero or more characters; e.g., `rr*` would match `Lodger & Heroes & Subterraneans`

```
[title^="Cthulhu"]
```

Begins with attribute value selectors: Does element *foo* have the attribute `title` with the specific text `Cthulhu` at the beginning of the `title` value?

Starts with, so `Cthulhu` matches `Cthulhu R'lyeh`, but not `R'lyeh Cthulhu`

HTML



```
1 <p>
2   <a href="http://cthulhu.com">Cthulhu</a>
   invites you to join the <a
   href="http://www.miskatonic-expedition.com"
   title="Led by Pabodie, Lake, Atwood &
   Dyer">Miskatonic Expedition</a> to the
   Antarctic! Flights available on <span
   lang="pt">Gol Transportes Aéreos</span> & <a
   href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>.
3 </p>
```

CSS



```
1 a[title^="Led"] {
2   font-weight: bold;
3 }
4
5 /* Uninteresting stuff below here */
6
7 html {
8   font-size: 24px;
9   font-family: "Iowan Old Style", Georgia,
   serif;
10 }
```

JS



[Cthulhu](#) invites you to join the [Miskatonic Expedition](#) to the Antarctic! Flights available on Gol Transportes Aéreos & [Cthulhu Airlines](#).

```
[title$="fhtagn"]
```

Ends with attribute value selectors: Does element *foo* have the attribute `title` with the specific text `fhtagn` at the end of the `title` value?

Ends with, so `fhtagn` matches `R'lyeh fhtagn` but not `R'lyeh fhtagn!`

HTML



```
1 <p>
2   <a href="http://cthulhu.com">Cthulhu</a>
   invites you to join the <a
   href="http://www.miskatonic-expedition.com"
   title="Led by Pabodie, Lake, Atwood &
   Dyer">Miskatonic Expedition</a> to the
   Antarctic! Flights available on <span
   lang="pt">Gol Transportes Aéreos</span> & <a
   href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>.
3 </p>
```

CSS



```
1 a[title$="Dyer"] {
2   cursor: url(https://s3-us-west-
   2.amazonaws.com/s.cdpn.io/122116/old-bearded-
   man.png), auto;
3 }
4 /* Note that Pabodie & Lake won't work because
   of the commas! */
5
6 /* Uninteresting stuff below here */
7
8 html {
9   font-size: 24px;
```

JS



[Cthulhu](#) invites you to join the [Miskatonic Expedition](#) to the Antarctic! Flights available on Gol  ie, Lake, Atwood & Dyer)S & [Cthulhu Airlines](#).

```
[title*="cthulhu"]
```

Substring match attribute value selectors: Does element *foo* have the attribute `title` with the specific text `cthulhu` somewhere in the `title` value?

Substring, so `mad` matches `madness` & `armadillo` & `nomad`

HTML



```
1 <p>
2   <a href="http://cthulhu.com">Cthulhu</a>
   invites you to join the <a
   href="http://www.miskatonic-expedition.com"
   title="Led by Pabodie, Lake, Atwood &
   Dyer">Miskatonic Expedition</a> to the
   Antarctic! Flights available on <span
   lang="pt-BR">Gol Transportes Aéreos</span> &
   <a href="http://cthulhu-airlines.com"
   class="flight">Cthulhu Airlines</a>.
3 </p>
```

CSS



```
1 a[href*="miskatonic"] {
2   text-transform: uppercase;
3   font-family: Creepster, cursive;
4 }
5
6 /* Uninteresting stuff below here */
7
8 html {
9   font-size: 24px;
10  font-family: "Iowan Old Style", Georgia,
   serif;
11 }
```

JS



[Cthulhu](#) invites you to join the [MISKATONIC EXPEDITION](#) to the Antarctic! Flights available on Gol Transportes Aéreos & [Cthulhu Airlines](#).

Chain 'em together!

HTML

```
<div style="color:red; margin-left:40px;">  
  Attention!  
</div>
```

CSS

```
div[style*="color:red"][style*="-left:40px"]{  
  margin-right: 40px;  
}
```

By default, all attribute selectors are *case-sensitive*

- » `[title] ≠ `
- » `[title="cthulhu"] ≠ `
- » `[title$="fhtagn"] ≠ `
- » `[title*="cthulhu"] ≠ <a title="Me
Chtulhuizing my hoodie">`

```
[title*="Cthulhu" i]
```

Adding *i* to the end of *any* attribute selector makes the match *case-insensitive*, so *Cthulhu* matches *Cthulhu* & *cthulhu* & *CtHuLhU*

					ios		
<pre>[attr], [attr="value"], [attr~="value"], [attr ="value"]</pre>	7	12	2	3.1	3.2	4	2.1
<pre>[attr^="value"], [attr\$="value"], [attr*="value"]</pre>	7	12	3.5	3.2	3.2	4	2.1
i	—	79	47	9	9.2	49	69

Compound Selectors

A compound selector consists of a chain of simple selectors connected together that describes *multiple conditions* on an element

Not connected by a combinator (which is coming up next)

`table.inventory` matches

```
<table class="inventory">
```

`table.inventory.northwest` matches

```
<table class="inventory northwest">
```

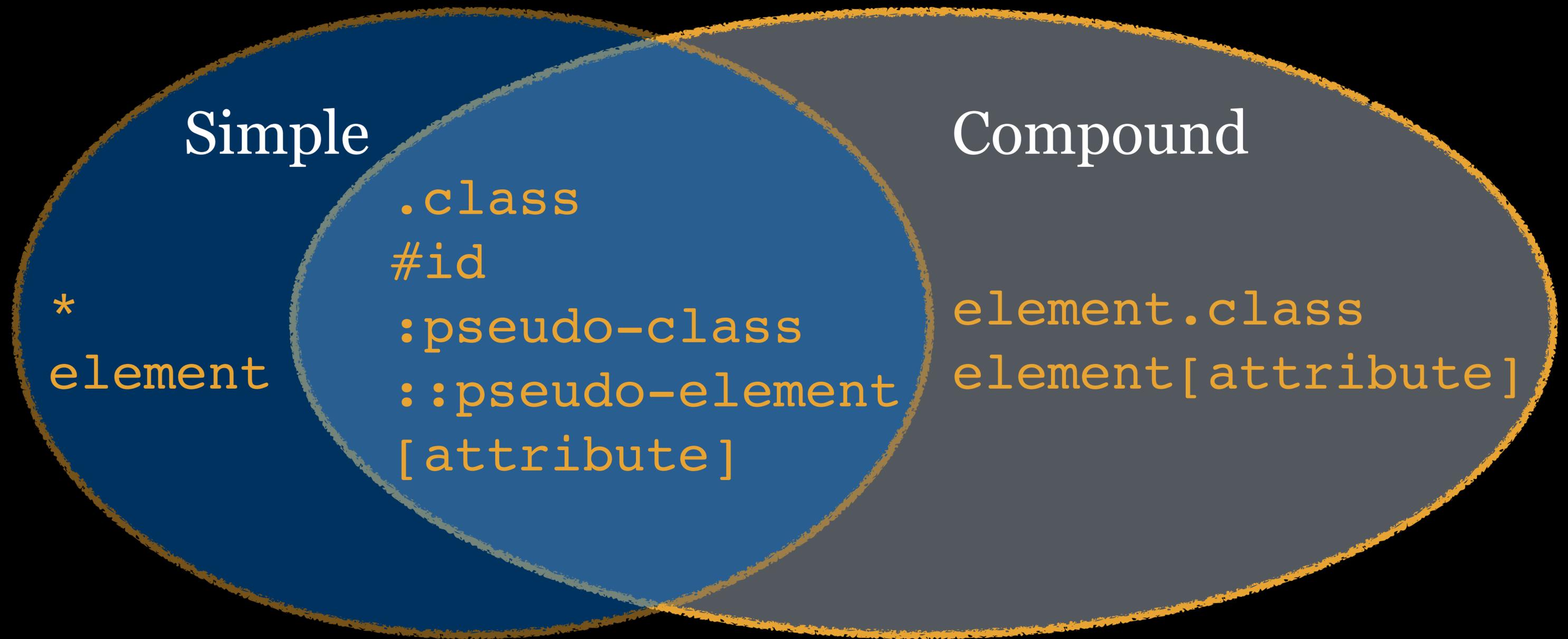
`.cthulhu:first-child` matches the first

```
<element class="cthulhu">
```

In *CSS Overview*, we said that these are all simple selectors

```
*  
element  
.class  
#id  
::pseudo-classes  
::pseudo-elements  
[attribute]
```

This is true, but not the whole story



Technically, most of the simple selectors are also compound because you can put `*` in front of them

Complex Selectors Using Combinators

Combinator	Name	Ex.	Which B is selected?
└ (space)	Descendant	A B	Any descendant of A
>	Child	A > B	Direct children of A
+	Next Sibling	A + B	Next sibling after A
~ (tilde)	Subsequent Sibling	A ~ B	All siblings after A

A complex selector uses a combinator to *express a relationship between selectors*

Descendent Combinator

`selectorA selectorB`

Selects any `selectorB` who has `selectorA` as *an ancestor*

`selectorB` can be a child, grandchild, or later descendant of `selectorA`

Any other `selectorB` is unaffected



```
1 <section>
2   <h2>Cthulhu</h2>
3   <p><span class="alert-indicator">!</span>
  You know they were here long before the
  fabulous epoch of Cthulhu was over, and
  remember all about sunken R'lyeh when it
  was above the waters.</p>
4 </section>
5 <aside>
6   <h2>Nyarlathotep</h2>
7   <p><span class="alert-indicator">!</span>
  And where Nyarlathotep went, rest
  vanished, for the small hours were rent
  with the screams of nightmare.</p>
8 </aside>
```



Cthulhu

! You know they were here long before the fabulous epoch of Cthulhu was over, and remember all about sunken R'lyeh when it was above the waters.

Nyarlathotep

! And where Nyarlathotep went, rest vanished, for the small hours were rent with the screams of nightmare.

```
HTML
CSS
1 /* Create an alert icon using CSS */
2
3 .alert-indicator {
4   display: inline-block;
5   border-radius: 50%;
6   background-color: red;
7   color: white;
8   width: 25px;
9   text-align: center;
10  font: bold 20px/1.25 Verdana;
11 }
12
13 /* Make the alert icon smaller when it's
   in the sidebar */
14
15 aside .alert-indicator {
16   font-size: 12px;
17   width: 15px;
18 }
19
20 /* Beautification (not part of the
   lesson) */
21
22 html {
23   font-size: 24px;
24   font-family: "Iowan Old Style",
   Georgia, serif;
JS
```

Cthulhu

! You know they were here long before the fabulous epoch of Cthulhu was over, and remember all about sunken R'lyeh when it was above the waters.

Nyarlathotep

! And where Nyarlathotep went, rest vanished, for the small hours were rent with the screams of nightmare.

SIDE NOTE

Used to be called *Contextual Selector* before W₃C
renamed it to *Descendant Combinator*

Child Combinator

`selectorA > selectorB`

Selects any `selectorB` who is a *direct child* of `selectorA`, not a grandchild or any other descendants

All siblings who are direct children of `selectorA` are selected

Siblings: 2 or more elements that share a parent

Contrasts with the descendant combinator, which selects both direct children & any descendants, no matter how deep



```
1 <aside>
2   <h2>Nyarlathotep</h2>
3   <p>And where Nyarlathotep went, rest
  vanished, for the small hours were
  rent with the screams of nightmare.
4   </p>
5   <ul>
6     <li>terrible phantasms</li>
7     <li>monstrous guilt</li>
8     <li>hellish moon-glitter</li>
9     <li>mindless gargoyles</li>
10  </ul>
11 </aside>
```

Nyarlathotep

And where Nyarlathotep went, rest vanished, for the small hours were rent with the screams of nightmare.

- terrible phantasms
- monstrous guilt
- hellish moon-glitter
- mindless gargoyles



HTML

CSS

```
1 /* The aside is fixed width with a
   black border */
2
3 aside {
4   width: 20em;
5   border: 1px solid black;
6 }
7
8 /* A reversed, full-width header is
   nice */
9
10 aside > h2 {
11   margin: 0 0 1em 0;
12   padding: .25rem 1rem;
13   background: darkslategray;
14   color: white;
15 }
16
17 /* Other elements just inside the aside
   need some room */
18
19 aside > * {
20   margin-right: 1rem;
21   margin-left: 1rem;
22 }
23
24 /* Beautification (not part of the
```

JS

Nyarlathotep

And where Nyarlathotep went, rest vanished, for the small hours were rent with the screams of nightmare.

- terrible phantasms
- monstrous guilt
- hellish moon-glitter
- mindless gargoyles

Next-Sibling

(aka Adjacent Sibling)

Combinator

`selectorA + selectorB`

Selects any `selectorB` who is an *immediately following (next) sibling* of `selectorA`

2nd element is selected, not both (remember *key selectors?*)

Elements must be listed in the order in which they appear in HTML

HTML

```
1 <h2>
2 Manuscript Found On The Coast Of
  Yucatan
3 </h2>
4 <p>
5 On August 20, 1917, I, Karl Heinrich,
  Graf von Altberg-Ehrenstein,
  Lieutenant-Commander in the Imperial
  German Navy and in charge of the
  submarine U-29, deposit this bottle
  and record in the Atlantic Ocean at a
  point to me unknown but probably about
  N. Latitude 20 degrees, W. Longitude
  35 degrees, where my ship lies
  disabled on the ocean floor. I do so
```

CSS

```
1 h2 {
2   margin-bottom: 0;
3 }
4
5 h2 + p {
6   margin-top: 0;
7 }
8
9 /* Uninteresting stuff below here */
10
11 html {
12   font-size: 24px;
```

JS

Manuscript Found On The Coast Of Yucatan

On August 20, 1917, I, Karl Heinrich, Graf von Altberg-Ehrenstein, Lieutenant-Commander in the Imperial German Navy and in charge of the submarine U-29, deposit this bottle and record in the Atlantic Ocean at a point to me unknown but probably about N. Latitude 20 degrees, W. Longitude 35 degrees, where my ship lies disabled on the ocean floor. I do so because of my desire to set certain unusual facts before the public; a thing I shall not in all probability survive to accomplish in person, since the circumstances surrounding me are as menacing as they are extraordinary, and involve not only the hopeless crippling of the U-29, but the impairment of my iron German will in a manner most disastrous.

On the afternoon of June 18, as reported by wireless to the U-61, bound for Kiel, we torpedoed the British freighter *Victory*, New York to Liverpool, in N. Latitude 45 degrees 16 minutes, W. Longitude 28 degrees 34 minutes; permitting the crew to

Subsequent-Sibling

(aka General Sibling)

Combinator

`selectorA ~ selectorB`

Selects `selectorB` *only if preceded by* `selectorA`, & both `selectorA` & `selectorB` share a common parent

`selectorA` & `selectorB` do *not* have to be adjacent siblings

If the Adjacent Sibling Combinator should really be called the *Next Sibling Combinator*, then the General Sibling Combinator should really be called the *All Following Siblings Combinator*

HTML

```
1 <ol>
2   <li>The Little Glass Bottle</li>
3   <li>The Secret Cave or John Lees
  Adventure</li>
4   <li>The Mystery of the Grave-
  Yard</li>
5   <li>The Mysterious Ship</li>
6   <li>The Beast in the Cave</li>
7   <li>The Alchemist</li>
8   <li class="current">The Tomb</li>
9   <li>Dagon</li>
10  <li>A Reminiscence of Dr. Samuel
  Johnson</li>
11  <li>Sweet Ermengarde</li>
12  <li>Polaris</li>
```

CSS

```
1 /* Using the general sibling combinator
  to show current position in a reading
  list */
2
3 .current {
4   font-weight: bold;
5 }
6
7 .current ~ li {
8   color: silver;
9 }
10
```

JS

1. The Little Glass Bottle
2. The Secret Cave or John Lees Adventure
3. The Mystery of the Grave-Yard
4. The Mysterious Ship
5. The Beast in the Cave
6. The Alchemist
7. **The Tomb**
8. Dagon
9. A Reminiscence of Dr. Samuel Johnson
10. Sweet Ermengarde
11. Polaris
12. Beyond the Wall of Sleep
13. Memory
14. Old Bugs
15. The Transition of Juan Romero
16. The White Ship
17. The Doom That Came to Sarnath
18. The Statement of Randolph Carter
19. The Terrible Old Man

Selector Lists

`selectorA, selectorB, selectorC`

List selectors that have similar declarations for simpler & cleaner CSS & HTML

⚠ Warning! ⚠

If there is an invalid pseudo-element or pseudo-class in the list of selectors, the entire selector list will fail!

```
.cthulhu, .azathoth, .hastur:~nope {  
  background-color: dodgerblue;  
  border: 1px solid darkblue;  
}
```



If a pseudo-element has a `-webkit-` prefix, newer browsers assume it's 👍 & do not invalidate the selector list

This *only* applies to pseudo-elements, *not* pseudo-classes!

“A way to remedy [the invalid selector problem is] to use the `:is()` or `:where()` selectors, which accept a forgiving selector list. This will ignore invalid selectors in the list but accept those which are valid.” —MDN

```
:is(.cthulhu, .azathoth, .hastur:nope) {  
  background-color: dodgerblue;  
  border: 1px solid darkblue;  
}
```



```
:where(.cthulhu, .azathoth, .hastur:nope) {  
  background-color: dodgerblue;  
  border: 1px solid darkblue;  
}
```





							
<code>:is()</code>	—	88	82	—	—	88	88
<code>:where()</code>	—	88	82	—	—	88	88

Support here is *only* for forgiving wrong selectors in a selector list, not for the selector itself;
data good as of 2021-06-16

Jans' 3 Wishes

Parents

There are no selectors to identify parents

As a result, always put your classes & IDs on the outermost element

```
<ul>
  <li><a href="http://foo.com">Foo</a></li>
  <li><a href="http://bar.com">Bar</a></li>
</ul>
```

Where would you put a class for `foo` & `bar` that best gives you most CSS flexibility later?

```
<ul class="social-media">
  <li><a href="http://foo.com">Foo</a></li>
  <li><a href="http://bar.com">Bar</a></li>
</ul>
```

Where would you put a class for `foo` & `bar` that best gives you most CSS flexibility later?



SIDE NOTE

Selectors Level 4 (Working Draft dated November 2018) introduces the Relational pseudo-class `:has()`

`a:has(> img)` would match `<a>` elements that contain `` as a direct child

Previous Siblings

You can select next (*Adjacent*) or any following (*General*) siblings, but you cannot select *previous* siblings



SIDE NOTE

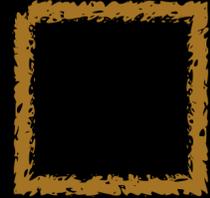
Selectors Level 4 (Working Draft dated November 2018) introduces the Relational pseudo-class `:has()`

`a:has(+img)` would match `<a>` elements that are immediately followed by `` as a sibling

CSS-Generated Wrappers

Remember this diagram when we discussed `::before`
& `::after`?

Boxes that cannot be created with CSS



Inline box created by `::before`

Block box created by `<p>`

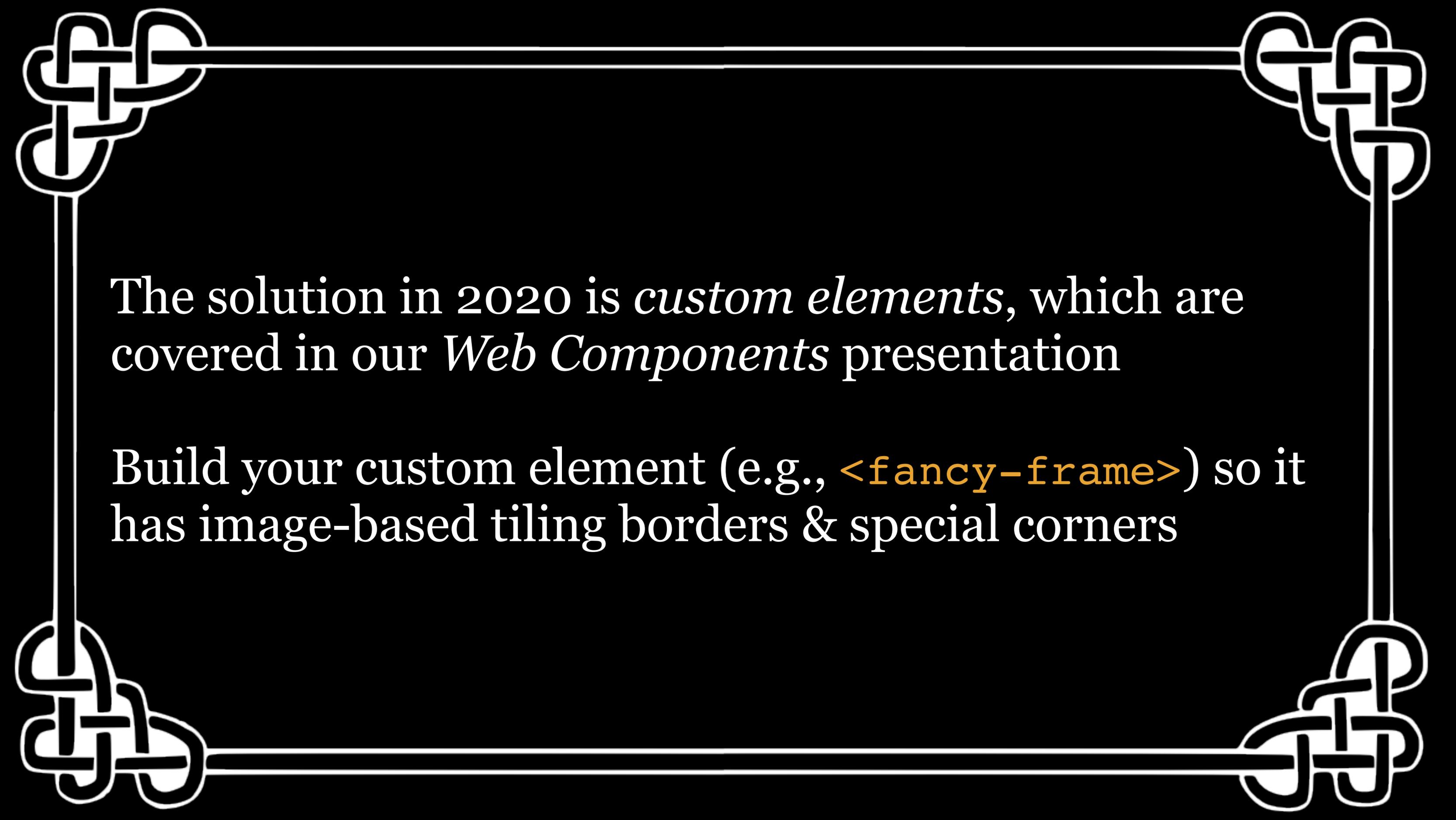
Inline box created by `::after`



`::before` & `::after` create a box inside the selected element, but there's no way to draw a box outside the selected element

`::outside` was around for a while in a spec, but died

Jans would also love an `::inside` as well!



The solution in 2020 is *custom elements*, which are covered in our *Web Components* presentation

Build your custom element (e.g., `<fancy-frame>`) so it has image-based tiling borders & special corners

Thank you!

scott@granneman.com

www.granneman.com

ChainsawOnATireSwing.com

[@scottgranneman](https://www.instagram.com/scottgranneman)

jans@websanity.com

websanity.com

CSS Building Blocks

Selectors

R. Scott Granneman & Jans Carton

© 2005 R. Scott Granneman
Last updated 2023-09-28

You are free to use this work, with certain restrictions.
For full licensing information, please see the last slide/page.

Changelog

2023-09-28 2.4: Actually, `:has()` is not in danger & has wide support!

2021-06-16 2.3: Added slides listing all pseudo-elements; updated new CSS4 combinators; more info re: hazardous state of `:has()`; added screenshot for `:is()`; better screenshot for `:not()`; added slide with `:not([class~="m-1v1"])` in *Attributes*

Changelog

2020-12-25 2.2: Updated screenshots for *User Action Pseudo-Class Selectors*; improved wording for `:not(s)`; updated *Compound Selectors* to be more accurate; added section in *Selector Lists* about invalid selectors causing the entire list to fail; changed section name to *Complex Selectors Using Combinators* & improved combinator chart; added custom elements as solution in *CSS-Generated Wrappers*

Changelog

2020-07-24 2.1: Minor fixes; changed descriptions for pseudo-classes & pseudo-elements; added screenshot for pseudo-elements; Lovecraft-ized explanation of `:target`; fixed `:hover` for Safari/iOS & added note for Edge bug with `:hover` in compatibility chart; updated screenshot for Tree-Structural Child-Indexed Pseudo-Class Selectors; added note to 1st `:not()` example; corrected number & list of CSS 4 selectors in *How Many?*

Changelog

2020-07-14 2.0: (con't. from ↓) better examples/
screenshots for `:not`; added list of values for `content`
under `::before` & added examples for `open-quote` &
`no-open-quote`; added examples of regex in *Attributes*;
updated compatibility charts in *Pseudo-Classes*;
updated slides for attribute selectors; so many changes
I bumped it up to 2.0!

Changelog

2020-07-14 2.0: Updated Jans' 3 Wishes; updated HTML/CSS vocabulary table; provided descriptive names for all pseudo-classes; moved `:target` into *Location* & added `:any-link` & `:scope`; added `:focus-within` to *User Action*; under *Logical Combinations* we now have `:is()`, `:not()`, `:where()`, & `:has()`; moved `:enabled` & `:disabled` into new *Input Control States* section in *Forms*; moved `:checked` into new *Input Value States* in *Forms*; moved `:valid` & `:invalid` into new *Input Value-Checking* section in *Forms*; (con't. ↑)

Changelog

2019-08-23 1.17: Added boxed drop cap in `::first-letter`; updated compatibility chart for case-insensitive attribute selectors; updated regex examples with `rr*`

2018-12-02 1.16: CSS 1 & 2 both support `:hover`, not `:active`; un-bolded regex Bowie songs that don't match since that's clearer

Changelog

2018-11-06 1.15: Updated icons & column order in compatibility table

2018-10-01 1.14: Added Can I Use screenshot for `:has`; added full list of selectors level 4; added slides listing selectors for 1, 2, 3, & 4; better example for `:empty`; clarified `:disabled`; updated chart for `:fullscreen` & `::backdrop`; added chart on case insensitive attribute selectors (`i`); updated theme to Granneman 1.5; minor fixes

Changelog

2018-02-05 1.13: Switched theme to Granneman 1.4; fixed formatting errors; added `:fullscreen` & `::backdrop`

2017-10-22 1.12 (con't ↓): added better example for attribute value selector & made old example a bad example; provided full list of selectors at beginning; `content` property is required for `::before` & `::after`; improved intro to attribute selectors; added `:not` can accept list of selectors

Changelog

2017-10-22 1.12: Added slide explaining why so many selectors; moved pseudo-classes, pseudo-elements, & attributes under Simple; hid simple selectors details covered in CSS Overview; simplified organization of selectors; renamed Combinators to Complex Selectors with Combinators; added note re: requiring CSS Overview; indicated that Child-Indexed & Typed Child-Indexed are both Tree Structural; better example for Typed Child-Indexed; (con't ↑)

Changelog

2016-09-23 1.10: Added another example of `:not()` pseudo-class selector; changed theme to Granneman 1.2; fixed formatting caused by changing theme; added info re: key selectors & speed; changed title from Basic Selectors to Common Selectors; added arrows to IDs as JavaScript hooks; created Combinator table; minor wording changes; reference combinators are dead; added xkcd regex cartoon; Bowie-ized regex Attribute Selectors

Changelog

2016-02-01 1.9: Switched to Georgia Pro theme & fixed resulting issues; added slides on `:focus` & `:active`; removed old & added new & better example for `:target`; updated Selectors Level 4 as still in Editor's Draft; added slide re: combining pseudo-classes & pseudo-elements; added detail re: `::outside`

Changelog

2016-01-20 1.8: Added slide re: regex to explain regex-based attribute selectors; reorganized list of combinators; IDs can be used for JavaScript too; added picture of ID used for ToC; indicated earlier that adjacent sibling should be thought of as next sibling

Changelog

2016-01-10 1.7: Fixed formatting of lists of selectors; got rid of **E** & **F** in selectors & made them clearer; changed **.bigRed** to **.big-red**; clarified source of class & ID names; made number of selectors clearer; redefined selector; added slides on multiple class selectors; added slide re: IDs as URL fragment identifiers; added another example of Child Combinator

Changelog

2015-05-14 1.6: Better explained the adjacent sibling combinator & added note; added CSS 4 selectors that will be discussed in the future; added using layout via `::before` & `::after`; fixed unclear key selector; fixed slide about overcoming inline styles with attribute selector

Changelog

2015-05-02 1.5: Added details about `:empty`; fixed `counters()` screenshot; moved slide explaining combinators; corrected number of selectors in CSS; added slides on CSS 4 selectors

2014-09-27 1.4: Changed “browser” to “rendering engine” in several places

2014-08-12 1.3: Improved Descendant Combinator section

Changelog

2014-08-06 1.2: Added definition of *data type*; clarified Adjacent Sibling & General Sibling combinators; added If Jans Had 3 Wishes section

2014-05-19 1.1.2: Added attribute selectors

2014-05-18 1.1.1: Added “Legal Numbering Using HTML Heading Levels” to `::before`

TODO

Add ex's for `:checked`, `:fullscreen`, & `::backdrop`;

Licensing of this work

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.

To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/>.

You are free to:

- » *Share* — copy and redistribute the material in any medium or format
- » *Adapt* — remix, transform, and build upon the material for any purpose, even commercially

Under the following terms:

Attribution. You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. Give credit to:

Scott Granneman • www.granneman.com • scott@granneman.com

Share Alike. If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions. You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Questions? Email scott@granneman.com