CSS selectors cheatsheet

> . , * + ~ : [ ] ( ) { }

by **nana jeon**

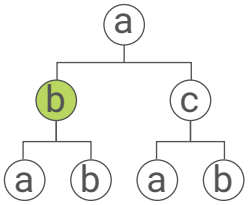
Feb 2019

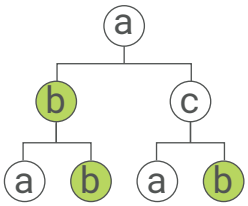
CSS selectors cheatsheet

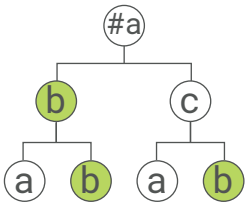
Selector name	HTML	CSS	Infographic

Type Selector	`<a/>`
	`<b/>`
	`<c/>`
		`a {}`	

ID Selector	`<..id="a"/>`
	`<..id="b"/>`
	`<..id="c"/>`
		`#a {}`	
			* **ID** is not recommended as a selector

Child Selector	`<a>`
	`<b>`
	`<a/><b/>`
	`</b>`
	`<c>`
	`<a/><b/>`
	`</c>`
	`</a>`
		`a > b {}`	

Descendant Selector	`<a>`
	`<b>`
	`<a/><b/>`
	`</b>`
	`<c>`
	`<a/><b/>`
	`</c>`
	`</a>`
		`a b {}`	

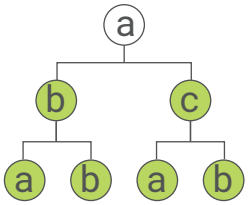
Combine Descendant & ID Selectors	`<a id="a">`
	`<b>`
	`<a/><b/>`
	`</b>`
	`<c>`
	`<a/><b/>`
	`</c>`
	`</a>`
		`#a b {}`	

Class Selector	`<.. class="a"/>`
	`<.. class="b"/>`
	`<.. class="c"/>`
		`.a {}`	

Combine the Class Selector	`<a class="x"/>`
	`<b class="x"/>`
	`<c class="x"/>`
		`b.x {}`	

Comma Combinator Selector	`<a/>`
	`<b/>`
	`<c/>`
	`<d/>`
		`a, c {}`	

Universal Selector	`<a/>`
	`<b/>`
	`<c/>`
	`<d/>`
		`* {}`	

Combine Universal Selector	`<a>`
	`<b>`
	`<a/><b/>`
	`</b>`
	`<c>`
	`<a/><b/>`
	`</c>`
	`</a>`
		`a * {}`	

<table>
<tr><td>Adjacent Sibling
Selector</td><td><a/>

</td><td>a + b {}</td><td>  </td></tr>
<tr><td>General Sibling
Selector</td><td><a/>

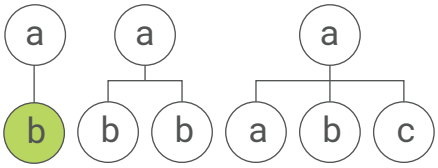
<c/>
</td><td>a ~ b {}</td><td>  </td></tr>
<tr><td rowspan="2">First Child Pseudo
Selector</td><td>

</td><td rowspan="2">b:first-child {}</td><td>  </td></tr>
<tr><td><a/>

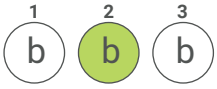
</td><td>first-child
 
*first-child is `<b>a</b>` element, not `<b>b</b>` element.
So there is nothing to be selected.</td></tr>
<tr><td>Only Child Pseudo
Selector</td><td><a>

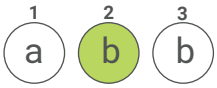
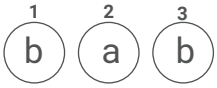
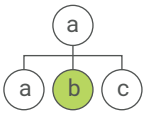
<a>

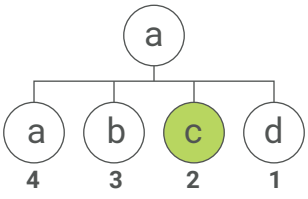
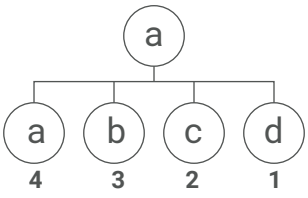
<a>
<a/><c/>
</td><td>b:only-child {}

a :only-child {}</td><td>  </td></tr>
<tr><td rowspan="2">Last Child Pseudo
Selector</td><td>
</td><td rowspan="2">b:last-child {}</td><td>  </td></tr>
<tr><td>

<c/></td><td>last-child
 
*last-child is `<b>c</b>` element, not `<b>b</b>` element.
So there is nothing to be selected.</td></tr>
<tr><td rowspan="4">Nth Child Pseudo
Selector</td><td>

</td><td rowspan="3">b:nth-child(2) {}</td><td>  </td></tr>
<tr><td><a/>

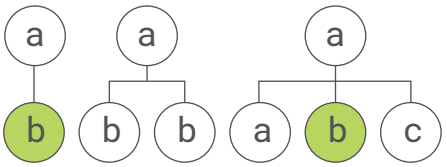
</td><td>  </td></tr>
<tr><td>
<a/>
</td><td> 
* nth-child(2) is `<b>a</b>` element, not `<b>b</b>` element.
So there is nothing to be selected.</td></tr>
<tr><td><a>
<a/><c/>
</td><td>a :nth-child(2) {}</td><td>  </td></tr>
<tr><td rowspan="5">Nth Last Child
Selector</td><td rowspan="5"><a>
<a/>

<c/>
<d/>
</td><td>a :nth-last-child(2) {}</td><td rowspan="2">  </td></tr>
<tr><td>c:nth-last-child(2) {}</td></tr>
<tr><td>a:nth-last-child(2) {}</td><td rowspan="3">  </td></tr>
<tr><td>b:nth-last-child(2) {}</td></tr>
<tr><td>d:nth-last-child(2) {}</td></tr>
</table>

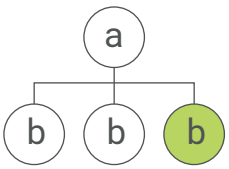
First of Type Selector	`<a/>`
	`<b/>`
	`<a/>`
	`<b/>`
		`b:first-of-type {}`	

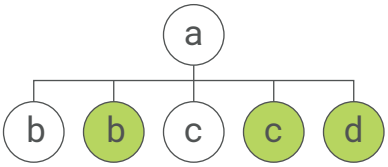
Nth of Type Selector	`<a/>`
	`<b/>`
	`<a/>`
	`<b/>`
	`<a/>`
	`<b/>`
	`<a/>`
		`a:nth-of-type(2) {}`	

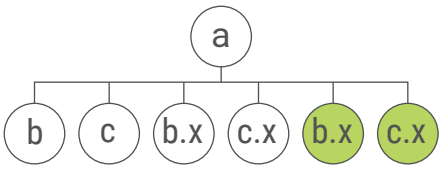
		`a:nth-of-type(even) {}`	
		`a:nth-of-type(odd) {}`	
		`a:nth-of-type(2n+1) {}`	 * `n` is an every positive integer or zero value.

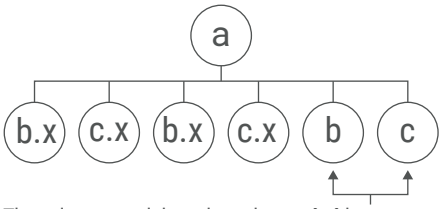
Only of Type Selector	`<a>`
	`<b/>`
	`</a>`
	`<a>`
	`<b/><b/>`
	`</a>`
	`<a>`
	`<a/><b/><c/>`
	`</a>`
		`b:only-of-type {}`	

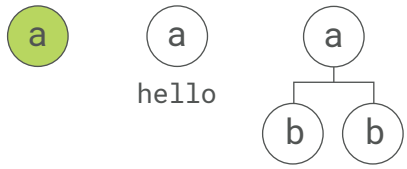
Last of Type Selector	`<a/>`
	`<b/>`
	`<b/>`
		`b:last-of-type {}`	

	`<a>`
	`<b/>`
	`<b/>`
	`<c/>`
	`<c/>`
	`<d/>`
	`</a>`
		`a :last-of-type {}`	

	`<a>`
	`<b/>`
	`<b/>`
	`<c/>`
	`<c/>`
	`<d/>`
	`</a>`
			

	`<a>`
	`<b/>`
	`<c/>`
	`<b class="x"/>`
	`<c class="x"/>`
	`<b class="x"/>`
	`<c class="x"/>`
	`</a>`
		`.x:last-of-type {}`	

	`<a>`
	`<b class="x"/>`
	`<c class="x"/>`
	`<b class="x"/>`
	`<c class="x"/>`
	`<b/>`
	`<c/>`
	`</a>`
			
			* Those items won't be selected as no `.x` is presented

Empty Selector	`<a/>`
	`<a>hello</a>`
	`<a>`
	`<b/><b/>`
	`</a>`
		`a:empty {}`	
			* `empty` indicates no children elements or text.

Negation Pseudo-class Selector	`<a/>`
	`<b/>`
	`<a class="x"/>`
	`<b class="x"/>`
		`a:not(.x) {}`	

	`<a/>`
	`<b/>`
	`<a/>`
	`<a/>`
		`a:not(:last-of-type){}`	

Attribute Selector	`<a/>`
	`<a for="x"/>`
	`<a for="y"/>`
	`<a for="z"/>`
		`[for]`	

	`<a/>`
	`<a for="x"/>`
	`<b for="y"/>`
	`<c for="z"/>`
		`a[for]`	

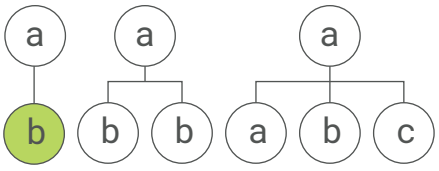
Attribute Value Selector	`<a/>`
	`<a for="x"/>`
	`<a for="y"/>`
	`<a for="z"/>`
		`a[for="x"]`	

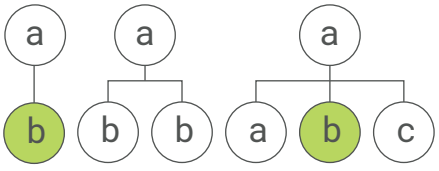
Attribute Starts with Selector	`<a for="x"/>`
	`<a for="xy"/>`
	`<a for="yz"/>`
	`<a for="zx"/>`
		`[for^="x"]`	

Attribute Ends with Selector	`<a for="x"/>`
	`<a for="xy"/>`
	`<a for="yz"/>`
	`<a for="zx"/>`
		`[for$="x"]`	

Attribute Wildcard Selector	`<a for="x"/>`
	`<a for="xy"/>`
	`<a for="yz"/>`
	`<a for="zx"/>`
		`[for*="x"]`	

Only Child Pseudo VS Only of Type

Only Child Pseudo Selector	`<a>`
	`<b/>`
	`</a>`
	
	`<a>`
	`<b/><b/>`
	`</a>`
	
	`<a>`
	`<a/><b/><c/>`
	`</a>`
		`b:only-child {}`	

Only of Type Selector		`b:only-of-type {}`	

* Happy codesign today!!! *

This cheatsheet is desinged for a quick search on CSS selectors. :)
There are so many CSS selectors with unfamiliar symbols, `> . , * + ~ [ ]` etc, so I am often confused with how CSS selectors work.
I wish this infographic helps you find proper CSS selectors. Print this lovely cheatsheet out and stick it on the wall.
And if you need more details on CSS selectors, check out my blog at ***medium.com/@nana8***.

I would love to hear your feedback on how I can make it better. Please leave your comments on my twitter ***@nanacodesign***.
Thank you !!! 감사합니다!!!

Collaborated with Ryan Yu

the author of `<FrontEnd30/>` which is
for anyone who loves front-end and wanted to
improve their front-end skills.

www.FrontEnd30.com