

CSS Selectors

CSS Selectors target elements based on html tags, attributes (including class and ID), and hierarchy in markup.

For HTML element diagrams:

- Stacked & left aligned elements are without a specified relationship
- Inline elements are siblings.
- Stacked & centered elements represent nesting structures.

Stacked (No relationship)	Siblings (Same Level)	Nested (Hierarchy)
<div>div</div> <div>section</div> <div>p</div>	<div>a</div> <div>div</div> <div>p</div>	<div>div</div> <div>p</div> <div>a</div> <div>img</div>
<div>These three elements could be in any order & in any hierarchy.</div>	<pre><a> <div></div> <p></p></pre>	<pre><div> <p></p> <a> </div></pre>

Basic

UNIVERSAL

Select all elements with an asterisk. This selector excludes pseudo elements, so you must specifically target **::before** & **::after**

```
* {
  property: value;
}
```

div

a

p

TAG

Select all elements with the same html tag by using the tag name alone (no special characters).

```
div {
  property: value;
}
```

div

a

div

CLASS

Select all elements with the same class by adding a period to the front of the class name. Multiple elements can have the same class.

```
.class {
  property: value;
}
```

p.class

p

a.class

ID

Select an element using it's ID by adding a hashtag to the front of the class name. IDs should be unique to an individual element.

```
#id {
  property: value;
}
```

div#id

a

p

Advanced

DESCENDANT

Select elements based on parent container. This example targets all paragraphs that are inside a div.

```
div p {
  property: value;
}
```

div

p

a

section

img

p

p

DIRECT CHILD

Select elements that are direct children of the parent (therefore excluding elements nested within the children)

```
div > p {
  property: value;
}
```

div

p

a

section

img

p

p

ADJACENT SIBLING

Select an element based on the element directly above in the html. This example targets any paragraph which immediately follows a div.

```
div + p {
  property: value;
}
```

div

p

p

MULTIPLE

Select multiple elements to apply the same styles. Separate selectors with a comma.

```
a, p.class {
  property: value;
}
```

a

p.class

a.class

WITH

Select elements which have a specific tag name and attribute by joining selectors. This example targets all divs with a class of **.class**

```
div.class {
  property: value;
}
```

div

div.class

p.class

NOT

Select all elements which do not match the selector inside **:not()**. This example targets all paragraphs without a class of **.class**

```
p:not(.class) {
  property: value;
}
```

p

p.class

a.class

HAS

Select all elements with a specific attribute.

```
div[title] {
  property: value;
}
```

div

div title="info"

div title="more info"

HAS (EXACT)

Select all elements with a specific attribute and value.

```
div[title="info"] {
  property: value;
}
```

div

div title="info"

div title="more info"

FIRST OF TYPE

Select elements which are the first of its type inside a container.

```
p:first-of-type {
  property: value;
}
```

div

a

p

img

p

LAST OF TYPE

Select elements which are the last of its type inside a container.

```
p:last-of-type {
  property: value;
}
```

div

a

p

img

p

ONLY CHILD

Select elements which are the only child inside a container.

```
a:only-child {
  property: value;
}
```

div

a

a

div

a

NTH CHILD

Select elements which are the nth child inside a container.

```
a:nth-child(3) {
  property: value;
}
```

div

a

a

a

a

Select elements which are odd OR even based on position inside a container.

```
a:nth-child(odd) {
  property: value;
}
```

div

a

a

a

a

Select elements which are the nth child inside a container based on a specified pattern.

```
a:nth-child(2n) {
  property: value;
}
```

div

a

a

a

a

NTH LAST CHILD

Select elements which are the nth child from the end, inside a container.

```
a:nth-last-child(2) {
  property: value;
}
```

div

a

a

a

a

REQUIRED

Style form inputs which have a **required** attribute.

```
input:required {
  property: value;
}
```

input*

States

HOVER

Style the mouse hover state of interactive elements.

```
button:hover {
  property: value;
}
```

button

TAB FOCUS

Style the tab focus state of interactive elements.

```
button:focus {
  property: value;
}
```

button

CHECKED

Style form inputs which are checked.

```
input:checked {
  property: value;
}
```

✓

Pseudo Elements

Pseudo elements are created in CSS. They are not in the flow of content, so will sit either on top of or behind other elements on the page. Pseudo elements can be created by attaching **::before** or **::after** to the selector for a parent element which exists in your HTML.

- The anchoring element must have **relative positioning**
- The pseudo element must have **absolute positioning, content**, and a distance relative to the boundary of the parent container, using either **top, right, bottom, and/or left** with a specified value

BEFORE	AFTER
<div>Creates an empty element before the children of the parent element.</div>	<div>Creates an empty element after the children of the parent element.</div>

<pre>div { position: relative; } div:before { position: absolute; content: ''; top: 0; left: 0; }</pre>	<div>div</div> <div>before</div> <div>p</div>
<pre>div { position: relative; } div:after { position: absolute; content: ''; top: 0; left: 0; }</pre>	<div>div</div> <div>p</div> <div>after</div>