CSS Selectors Cheatsheet

Element selectors

Element -- selects all `h2` elements on the page

```
h2 {
    foo: bar;
}
```

Group -- selects all `h1`, `h2` and `h3` elements on the page

```
h1, h2, h3 {
    foo: bar;
}
```

Class and ID selectors

Class -- selects all elements with class attribute containing`foo` or only `p` elements with that class

```
.foo {
    bar: fum;
}
p.foo {
    bar: fum;
}
```

ID -- selects the element with 'baz' id attribute value

```
#foo {
    bar: fum;
}
```

Contextual selectors

Descendant -- selects all `p` elements within the infinite-level hierarchy of element`#foo` descendants

```
#foo p {
    bar: fum;
}
```

Adjacent sibling -- selects the sibling element `p` that is immediately next to `h2` element

```
h2 + p {
    foo: bar;
}
```

Child -- selects all `p` elements that are immediate children of `#foo` element

```
#foo > p {
    bar: fum;
}
```

General sibling -- selects all elements `p` that are siblings to the `h2` element

```
h2 ~ p {
    foo: bar;
}
```

Pseudo-class selectors

Pseudo-class selectors for link and user states

Unvisited link -- applies to link elements that have not been visited

```
a:link {
    foo: bar;
}
```

Visited link -- applies to link elements that have been visited

```
a:visited {
    foo: bar;
}
```

Focus state -- applies to selected `.foo` element that is ready for input

```
.foo:focus {
    bar: fum;
}
```

Hover state -- applies when mouse pointer is over the `.foo` element

```
.foo:hover {
    bar: fum;
}
```

Active state -- applies when `.foo` element is in process of being clicked

```
.foo:active {
    bar: fum;
}
```

Pseudo-class selectors that apply to siblings

First child -- selects the specified `.foo` element when it is the first child of its parent

```
.foo:first-child {
    bar: fum;
}
```

Last child -- selects the specified `.foo` element when it is the last child of its parent

```
.foo:last-child {
    bar: fum;
}
```

Only child -- selects the specified `.foo` element when it is the only child of its parent

```
.foo:only-child {
    bar: fum;
}
```

First of type -- selects the `h2` element when it is the first element of its type within its parent element

```
h2:first-of-type {
    foo: bar;
}
```

Last of type -- selects the `h2` element when it is the last element of its type within its parent element

```
h2:last-of-type {
    foo: bar;
}
```

Only of type -- selects the `h2` element when it is the only element of its type within its parent element

```
h2:only-of-type {
    foo: bar;
}
```

Nth child -- selects the `n` th `.foo` child element

```
.foo:nth-child(n) {
    bar: fum;
}
```

Nth last child -- selects the `n` th `.foo` child element counting backwards

```
.foo:nth-last-child(n) {
    bar: fum;
}
```

Nth of type -- selects the `n` th `h2` child element of its type

```
h2:nth-of-type(n) {
    foo: bar;
}
```

Nth last of type -- selects the `n` th `h2` child element of its type counting backwards

```
h2:nth-last-of-type(n) {
    foo: bar;
}
```

Useful `n` values:

- `odd` or `2n+1` -- every odd child or element
- `even` or `2n` -- every even child or element
- `n` -- every nth child or element
- `3n` -- every third child or element (3, 6, 9, ...)
- `3n+1` -- every third child or element starting with `1` (1, 4, 7, ...)
- `n+6` -- all but first five children or elements (6, 7, 8, ...)
- `-n+5` -- only first five children or elements (1, 2, ..., 5)

Pseudo-element selectors

First letter -- selects the first letter of the specified `.foo` element, commonly used with `:first-child` to target first paragraph

```
.foo::first-letter {
    bar: fum;
}
```

First line -- selects the first line of the specified `.foo` element, commonly used with `:first-child` to target first paragraph

```
.foo::first-line {
    bar: fum;
}
```

Before -- adds generated content before the `.foo` element when used with `content` property

```
.foo::before {
    bar: fum;
    content: 'baz';
}
```

After -- adds generated content after the `.foo` element when used with `content` property

```
.foo::after {
    bar: fum;
    content: 'baz';
}
```

Attribute selectors

Present -- selects `.foo` elements with `bar` attribute present, regardless of its value

```
.foo[bar] {
    fum: baz;
}
```

Exact -- selects `.foo` elements where the `bar` attribute has the exact value of `fum`

```
.foo[bar="fum"] {
    baz: qux;
}
```

Whitespace separated -- selects `.foo` elements with `bar` attribute values contain specified partial value of `fum` (whitespace separated)

```
.foo[bar~="fum"] {
    baz: qux;
}
```

Hyphen separated -- selects `.foo` elements with `bar` attribute values contain specified partial value of `fum` immediately followed by hyphen (`-`) character

```
.foo[bar|="fum"] {
    baz: qux;
}
```

Begins with -- selects `.foo` elements where the `bar` attribute begins with `fum`

```
.foo[bar^="fum"] {
    baz: qux;
}
```

Ends with -- selects `.foo` elements where the `bar` attribute ends with `fum`

```
.foo[bar$="fum"] {
    baz: qux;
}
```

Containts -- selects `.foo` elements where the `bar` attribute contains string `fum` followed and preceded by any number of other characters

```
.foo[bar*="fum"] {
    baz: qux;
}
```

Misc selectors

Not -- selects `.foo` elements that are NOT `.bar` elements

```
.foo:not(.bar) {
    fum: baz;
}
```

Root -- selects the highest level parent element in the DOM

```
:root {
    foo: bar;
}
```

Empty -- selects `.foo` elements that have no children or whitespace inside

```
.foo:empty {
    bar: fum;
}
```

In-range and **Out-of-range** -- selects `.foo` elements that have values in or out of range

```css
.foo:in-range {
    bar: fum;
}
.foo:out-of-range {
    bar: fum;
}
```