Home → References → CSS →

CSS Selectors

Every **CSS selector**, taken from the latest **CSS3** standard.

Basic selectors

`element`

Type selector. Matches an element.

Example:

```
p { color: red }
/* matches paragraphs */
```

`.class`

Class selector. Matches the value of a `class` attribute.

Example:

```
.warning { color: red }
/* matches elements containing class="warning" */
```

`#id`

ID selector. Matches the value of an `id` attribute.

Example:

```
#warning { color: red }
/* matches elements containing id="warning" */
```

`*`

Universal selector. Matches everything.

Example:

```
* { color: red }
/* matches everything */
```

Attribute selectors

[attribute]

Matches elements **containing a given attribute**.

Example:

```
a[href] { color: red }
/* matches a elements with an href attribute */
```

[attribute="x"]

Matches elements **containing a given attribute with a given value**.

Example:

```
a[href="/sitemap/"] { color: red }
/* matches a elements with the attribute and value
href="/sitemap/" */
```

[attribute~="x"]

Matches elements containing a given attribute with a value that contains a sub-value within a **space-separated list**.

Example:

```
abbr[title~="Style"] { color: red }
/* matches abbr elements with a title that contains 'Style'
```

[attribute|="x"]

Matches elements containing a given attribute with a value that contains a sub-value within a **hyphen-separated list**.

Example:

```
html[lang|="en"] { color: red }
/* matches html elements with a lang attribute that
contains 'en' (such as in lang="en-gb") */
```

[attribute^="x"]

Matches elements containing a given attribute with a value that **starts** with something.

Example:

```
a[href^="http://"] { color: red }
/* matches a elements with an href attribute, the value of
which begins with 'http://' */
```

[attribute$="x"]

Matches elements containing a given attribute with a value that **ends** with something.

Example:

```
a[href$=".com"] { color: red }
/* matches a elements with an href attribute, the value of
which ends with '.com' */
```

```
[attribute*="x"]
```

Matches elements containing a given attribute with a value that **contains** something.

Example:

```
a[href*="htmldog"] { color: red }
/* matches a elements with an href attribute, the value of
which contains 'htmldog' */
```

Pseudo-classes

```
:link
```

Matches a **link that has not been visited**.

Example:

```
a:link { color: blue }
```

:visited

Matches a **link thas has been visited**.

Example:

```
a:visited { color: purple }
```

:active

Matches an element that is being **activated**, such as a link being clicked on.

Example:

```
a:active { color: red }
```

:hover

Matches an element whose box is being **hovered over** by a cursor.

Example:

```
a:hover { text-decoration: none }
```

`:focus`

Matches an element that has **focus**, such as one that has been tabbed to.

Example:

```
a:focus { border: 1px solid yellow }
```

`:target`

Matches an element that has been **linked to** (via `<a href="#x"`..., for example).

Example:

```
h2:target { color: red }
/* matches a second-level heading that has been linked to
*/
```

`:lang()`

Matches an element of a given **language**.

Example:

```
p:lang(fr) { color: red }
/* matches paragraphs that are declared, or otherwise
considered, as French */
```

`:first-child`

Matches the **first child** of an element.

Example:

```
p:first-child { color: red }
/* matches the first child, if it is a paragraph, of an element */
```

`:last-child`

Matches the **last child** of an element.

Example:

```
div p:last-child { color: blue }
/* matches the last child, if it is a paragraph, of an element */
```

`:first-of-type`

Matches the **first sibling of its type** in an element.

Example:

```
li:first-of-type { color: red }
/* matches the first instance of a list item inside an element */
```

`:last-of-type`

Matches the **last sibling of its type** in an element.

Example:

```
li:last-of-type { color: blue }
/* matches the last instance of a list item inside an
element */
```

`:nth-child()`

Matches an element that is the **ordinal number child** of its parent.

Example:

```
p:nth-child(3) { color: red }
/* matches the third child, if it is a paragrpah, of an
element */
```

`:nth-last-child()`

Matches an element that is the **ordinal number child, in reverse order**, of its parent.

Example:

```
p:nth-last-child(2) { color: blue }
/* matches the next-to-last child, if it is a paragraph, of
an element */
```

`:nth-of-type()`

Matches an element that is the **ordinal number sibling of its type**.

Example:

```
li:nth-of-type(5) { color: red }
/* matches the fifth instance of a list item inside an
element */
```

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

Matches an element that is the o**rdinal number sibling, in reverse order, of its type**.

Example:

```
li:nth-of-type(5) { color: red }
/* matches the next-to-last instance of a list item inside
an element */
```

`:only-child`

Matches an element if it is the **only child** of its parent.

Example:

```
article p:only-child { color: red }
/* matches a paragraph if it is the only child of an
```

`:only-of-type`

Matches an element if it is the **only sibling of its type**.

Example:

```
article aside:only-of-type { color: blue }
/* matches an aside element if it is the only aside element
in an article element */
```

`:empty`

Matches an element with **no children, or content**.

Example:

```
td:empty { border-color: red }
/* matches table data cells with nothing in 'em */
```

`:root`

Matches the **root element** of a document. This will be the `html` element in HTML.

Example:

```
:root { background: yellow }
```

`:enabled`

Matches **form control elements that are not disabled**.

Example:

```
input:enabled { border-color: lime }
/* matches input elements that are not disabled */
```

`:disabled`

Matches **form control elements that are disabled**.

Example:

```
input:enabled { border-color: red }
/* matches input elements that are disabled */
```

`:checked`

Matches a radio or checkbox type **input element that is checked**.

Example:

```
input:checked { outline: 3px solid yellow }
/* matches checked input elements */
```

`:not()`

Negotiation pseudo-class. Matches an element that does not match a selector.

Example:

```
p:not(:first-child) { color: orange }
/* matches paragraphs that are not first children */
```

Pseudo-elements

`::first-line`

Matches the **first textual line** in an element.

Example:

```
p::first-line { font-weight: bold }
/* matches the first line in a paragraph */
```

`::first-letter`

Matches the **first letter** in an element.

Example:

```
p::first-letter { font-size: 2em }
/* matches the first letter in a paragraph */
```

`::before`

Used with the `content` property to generate content **before** the initial content of an element.

Example:

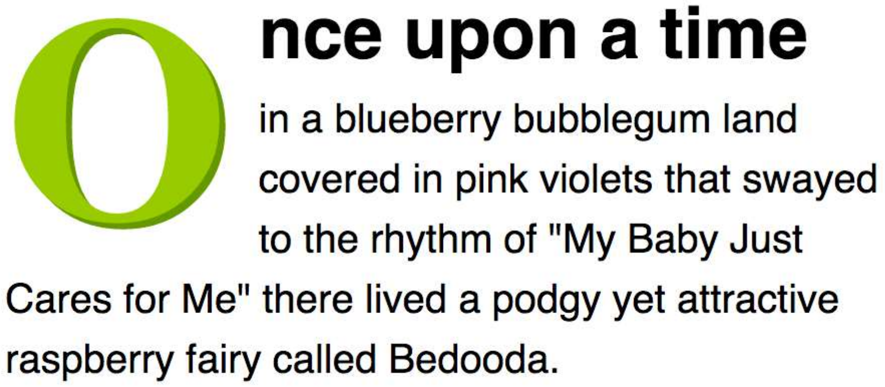
```
h1::before { content: "*" }
/* places an asterisk at the start of a top-level heading
*/
```

`::after`

Used with the `content` property to generate content **after** the initial content of an element.

Example:

```
h1::after { content: "+" }
/* places a plus-sign at the end of a top-level heading */
```



Once upon a time in a blueberry bubblegum land covered in pink violets that swayed to the rhythm of "My Baby Just Cares for Me" there lived a podgy yet attractive raspberry fairy called Bedooda.

Targeting the first letter, and first line, using pseudo-elements.

Pseudo-elements can also be defined using a single, rather than double, colon (`:first-line` as opposed to `::first-line`, for example). While this is old-school, it is also, for that very fact, more widely supported by browsers.

Combinators

`selector selector`

Descendant combinator. Matches elements that are descendants of another element.

Example:

```
/* matches paragraphs inside elements containing
class="warning" */
```

```
selector > selector
```

Child combinator. Matches elements that are children of another element.

Example:

```
.warning > p { color: red }
/* matches paragraphs that are children of elements
containing class="warning" */
```

```
selector + selector
```

Adjacent sibling combinator. Matches elements that immediately follow another element.

Example:

```
h1 + * { color: red }
/* matches the first element to follow a top-level heading
*/
```

```
selector ~ selector
```

General sibling combinator. Matches elements that follow another element.

Example:

```
h2 ~ p { color: red }
/* matches every paragraph that follows a second-level
heading */
```

!

Beyond the basic selectors, some others may cause incompatibility issues, especially with older browsers. As always, test thoroughly.

Related pages

CSS Properties

Web design and development by **HTML Dog, Edinburgh**.