# :has()

The functional `:has()` CSS pseudo-class represents an element if any of the relative selectors that are passed as an argument match at least one element when anchored against this element. This pseudo-class presents a way of selecting a parent element or a previous sibling element with respect to a reference element by taking a relative selector list as an argument.

CSS

```css
/* Selects an h1 heading with a
paragraph element that immediately follows
the h1 and applies the style to h1 */
h1:has(+ p) {
  margin-bottom: 0;
}
```

The `:has()` pseudo-class takes on the specificity of the most specific selector in its arguments the same way as `:is()` and `:not()` do.

## Syntax

CSS

```css
:has(<relative-selector-list>) {
  /* ... */
}
```

If the `:has()` pseudo-class itself is not supported in a browser, the entire selector block will fail unless `:has()` is in a forgiving selector list, such as in `:is()` and `:where()`).

The `:has()` pseudo-class cannot be nested within another `:has()`. This is because many pseudo-elements exist conditionally based on the styling of their ancestors and allowing these to be queried by `:has()` can introduce cyclic querying.

Pseudo-elements are also not valid selectors within `:has()` and pseudo-elements are not valid anchors for `:has()`.

## Examples

### With the sibling combinator

The `:has()` style declaration in the following example adjusts the spacing after `<h1>` headings if they are immediately followed by an `<h2>` heading.

#### HTML

HTML Play

```html
<section>
  <article>
    <h1>Morning Times</h1>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
      tempor incididunt ut labore et dolore magna aliqua.
    </p>
  </article>
  <article>
    <h1>Morning Times</h1>
    <h2>Delivering you news every morning</h2>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
```

```
      tempor incididunt ut labore et dolore magna aliqua.
    </p>
  </article>
</section>
```

### CSS

CSS Play

```
h1,
h2 {
  margin: 0 0 1rem 0;
}

h1:has(+ h2) {
  margin: 0 0 0.25rem 0;
}
```

### Result

Play


This example shows two similar texts side-by-side for comparison – the left one with an `H1` heading followed by a paragraph and the right one with an `H1` heading followed by an `H2` heading and then a paragraph. In the example on the right, `:has()` helps to select the `H1` element that is immediately followed by an `H2` element (indicated by the adjacent sibling combinator `+`) and the CSS rule reduces the spacing after such an `H1` element. Without the `:has()` pseudo-class, you cannot use CSS selectors to select a preceding sibling of a different type or a parent element.

## With the :is() pseudo-class

This example builds on the previous example to show how to select multiple elements with `:has()`.

### HTML

HTML Play

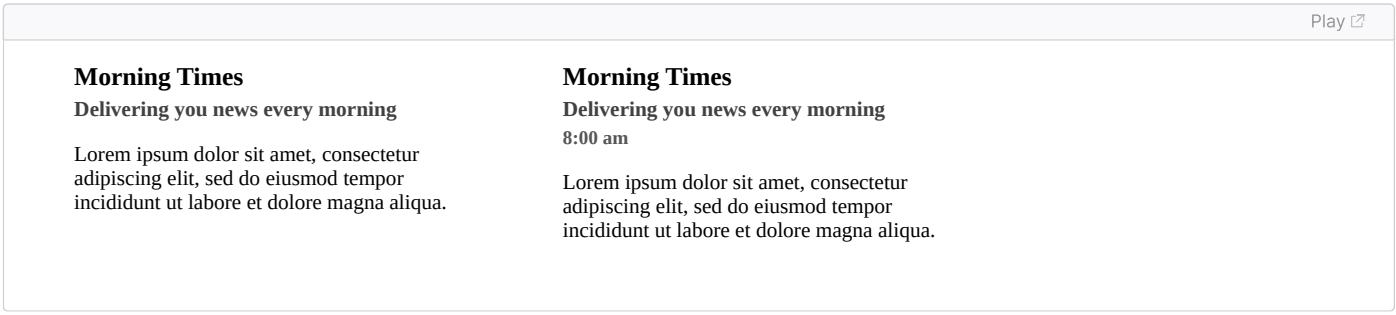
```
<section>
  <article>
    <h1>Morning Times</h1>
    <h2>Delivering you news every morning</h2>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
      tempor incididunt ut labore et dolore magna aliqua.
    </p>
  </article>
  <article>
    <h1>Morning Times</h1>
    <h2>Delivering you news every morning</h2>
    <h3>8:00 am</h3>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
      tempor incididunt ut labore et dolore magna aliqua.
    </p>
  </article>
</section>
```

### CSS

```
h1,
h2,
h3 {
  margin: 0 0 1rem 0;
}

:is(h1, h2, h3):has(+ :is(h2, h3, h4)) {
  margin: 0 0 0.25rem 0;
}
```

### Result




Here, the first `:is()` pseudo-class is used to select any of the heading elements in the list. The second `:is()` pseudo-class is used to pass a list of adjacent sibling selectors as an argument to `:has()`. The `:has()` pseudo-class helps to select any `H1`, `H2`, or `H3` element that is immediately followed by (indicated by `+`) an `H2`, `H3`, or `H4` element and the CSS rule reduces the spacing after such `H1`, `H2`, or `H3` elements.

This selector could have also been written as:

```
:is(h1, h2, h3):has(+ h2, + h3, + h4) {
  margin: 0 0 0.25rem 0;
}
```

### Logical operations

The `:has()` relational selector can be used to check if one of the multiple features is true or if all the features are true.

By using comma-separated values inside the `:has()` relational selector, you are checking to see if any of the parameters exist. `x:has(a, b)` will style `x` if descendant `a` OR `b` exists.

By chaining together multiple `:has()` relational selectors together, you are checking to see if all of the parameters exist. `x:has(a):has(b)` will style `x` if descendant `a` AND `b` exist.

```
body:has(video, audio) {
  /* styles to apply if the content contains audio OR video */
}
body:has(video):has(audio) {
  /* styles to apply if the content contains both audio AND video */
}
```

## Analogy between :has() and regular expressions

Interestingly, we can relate some CSS `:has()` constructs with the lookahead assertion in regular expressions because they both allow you to select elements (or strings in regular expressions) based on a condition without actually selecting the condition matching the element (or string) itself.

### Positive lookahead (?=pattern)

In the regular expression `abc(?=xyz)`, the string `abc` is matched only if it is immediately followed by the string `xyz`. As it is a lookahead operation, the `xyz` is not included in the match.

The analogous construct in CSS would be `.abc:has(+ .xyz)`: it selects the element `.abc` only if there is an adjacent sibling `.xyz`. The part `:has(+ .xyz)` acts as a lookahead operation because the element `.abc` is selected and not the element `.xyz`.

### Negative lookahead (?!pattern)

Similarly, for the negative lookahead case, in the regular expression `abc(?!xyz)`, the string `abc` is matched only if it is *not* followed by `xyz`. The analogous CSS construct `.abc:has(+ :not(.xyz))` doesn't select the element `.abc` if the next element is `.xyz`.

## Specifications

| Specification |
| --- |
| Selectors Level 4<br># relational |

## Browser compatibility

Report problems with this compatibility data on GitHub

| | Chrome | Edge | Firefox | Opera | Safari | Chrome Android | Firefox for Android | Opera Android |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| `:has()` | ✓ Chrome 105 | ✓ Edge 105 | ✗ Firefox 103 | ✓ Opera 91 | ✓ Safari 15.4 | ✓ Chrome 105 Android | ✗ Firefox No for Android | ✓ Opera 7 Android |

*Tip: you can click/tap on a cell for more information.*

mdn web docs

## See also

- `:is()`, `:where()`, `:not()`
- CSS selectors and combinators
- CSS selector structure
- Selector list
- CSS selector module
- Locating DOM elements using selectors

This page was last modified on Jul 11, 2023 by MDN contributors.