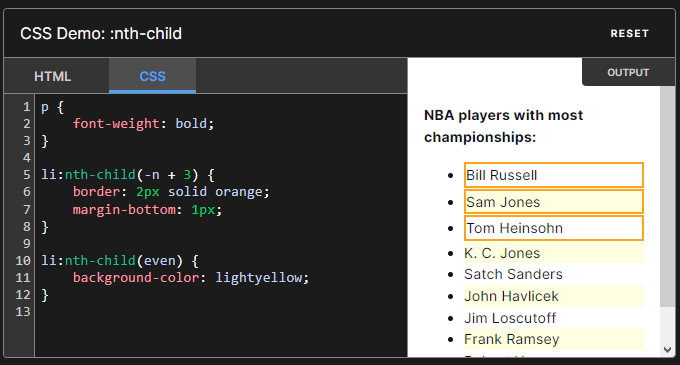
**Pseudo-classes:**

**:nth-child()**



[**Syntax**](https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-child#syntax)

:nth-child() takes a single argument that describes a pattern for matching element indices in a list of siblings. Element indices are 1-based.

:nth-child(<nth> [of <complex-selector-list>]?) {

/\* ... \*/

}

Copy to Clipboard

[**Keyword values**](https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-child#keyword_values)

1. odd
   1. Represents elements whose numeric position in a series of siblings is odd: 1, 3, 5, etc.
2. even
   1. Represents elements whose numeric position in a series of siblings is even: 2, 4, 6, etc.

[**Functional notation**](https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-child#functional_notation)

<An+B>

Represents elements whose numeric position in a series of siblings matches the pattern An+B, for every positive integer or zero value of n, where:

* A is an integer step size,
* B is an integer offset,
* n is all nonnegative integers, starting from 0.

It can be read as the An+B-th element of a list. The A and B must both have [<integer>](https://developer.mozilla.org/en-US/docs/Web/CSS/integer) values.

[**The of <selector> syntax**](https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-child#the_of_selector_syntax)

By passing a selector argument, we can select the **nth** element that matches that selector. For example, the following selector matches the first three list items which have a class="important" set.

:nth-child(-n + 3 of li.important) {

}

Copy to Clipboard

This is different from moving the selector outside of the function, like:

li.important:nth-child(-n + 3) {

}

Copy to Clipboard

This selector selects list items if they are among the first three children and match the selector li.important.

[**Examples**](https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-child#examples)

[**Example selectors**](https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-child#example_selectors)

tr:nth-child(odd) or tr:nth-child(2n+1)

Represents the odd rows of an HTML table: 1, 3, 5, etc.

tr:nth-child(even) or tr:nth-child(2n)

Represents the even rows of an HTML table: 2, 4, 6, etc.

:nth-child(7)

Represents the seventh element.

:nth-child(5n)

Represents elements **5** [=5×1], **10** [=5×2], **15** [=5×3], **etc.** The first one to be returned as a result of the formula is **0** [=5x0], resulting in a no-match, since the elements are indexed from 1, whereas n starts from 0. This may seem weird at first, but it makes more sense when the B part of the formula is >0, like in the next example.

:nth-child(n+7)

Represents the seventh and all following elements: **7** [=0+7], **8** [=1+7], **9** [=2+7], **etc.**

:nth-child(3n+4)

Represents elements **4** [=(3×0)+4], **7** [=(3×1)+4], **10** [=(3×2)+4], **13** [=(3×3)+4], **etc.**

:nth-child(-n+3)

Represents the first three elements. [=-0+3, -1+3, -2+3]

p:nth-child(n)

Represents every <p> element in a group of siblings. This selects the same elements as a simple p selector (although with a higher specificity).

p:nth-child(1) or p:nth-child(0n+1)

Represents every <p> that is the first element in a group of siblings. This is the same as the [:first-child](https://developer.mozilla.org/en-US/docs/Web/CSS/:first-child) selector (and has the same specificity).

p:nth-child(n+8):nth-child(-n+15)

Represents the eighth through the fifteenth <p> elements of a group of

Example

HTML

<h3>

<code>span:nth-child(2n+1)</code>, WITHOUT an <code>&lt;em&gt;</code> among

the child elements.

</h3>

<p>Children 1, 3, 5, and 7 are selected.</p>

<div class="first">

<span>Span 1!</span>

<span>Span 2</span>

<span>Span 3!</span>

<span>Span 4</span>

<span>Span 5!</span>

<span>Span 6</span>

<span>Span 7!</span>

</div>

<br />

<h3>

<code>span:nth-child(2n+1)</code>, WITH an <code>&lt;em&gt;</code> among the

child elements.

</h3>

<p>

Children 1, 5, and 7 are selected.<br />

3 is used in the counting because it is a child, but it isn't selected because

it isn't a <code>&lt;span&gt;</code>.

</p>

<div class="second">

<span>Span!</span>

<span>Span</span>

<em>This is an `em`.</em>

<span>Span</span>

<span>Span!</span>

<span>Span</span>

<span>Span!</span>

<span>Span</span>

</div>

<br />

<h3>

<code>span:nth-of-type(2n+1)</code>, WITH an <code>&lt;em&gt;</code> among the

child elements.

</h3>

<p>

Children 1, 4, 6, and 8 are selected.<br />

3 isn't used in the counting or selected because it is an

<code>&lt;em&gt;</code>, not a <code>&lt;span&gt;</code>, and

<code>nth-of-type</code> only selects children of that type. The

<code>&lt;em&gt;</code> is completely skipped over and ignored.

</p>

<div class="third">

<span>Span!</span>

<span>Span</span>

<em>This is an `em`.</em>

<span>Span!</span>

<span>Span</span>

<span>Span!</span>

<span>Span</span>

<span>Span!</span>

</div>

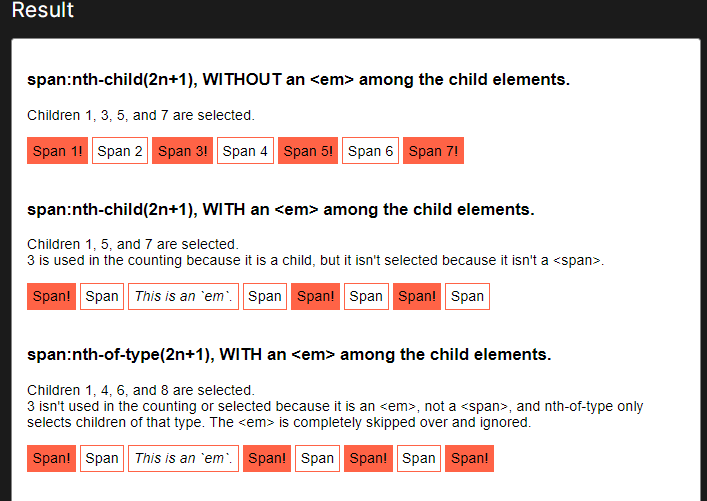
first span:nth-child(2n + 1),

.second span:nth-child(2n + 1),

.third span:nth-of-type(2n + 1) {

background-color: tomato;

}



**Using 'of <selector>'**

In this example there is an unordered list of names, some of them have been marked as noted using class="noted". These have been highlighted with a thick bottom border.

HTML

<ul>

<li class="noted">Diego</li>

<li>Shilpa</li>

<li class="noted">Caterina</li>

<li>Jayla</li>

<li>Tyrone</li>

<li>Ricardo</li>

<li class="noted">Gila</li>

<li>Sienna</li>

<li>Titilayo</li>

<li class="noted">Lexi</li>

<li>Aylin</li>

<li>Leo</li>

<li>Leyla</li>

<li class="noted">Bruce</li>

<li>Aisha</li>

<li>Veronica</li>

<li class="noted">Kyouko</li>

<li>Shireen</li>

<li>Tanya</li>

<li class="noted">Marlene</li>

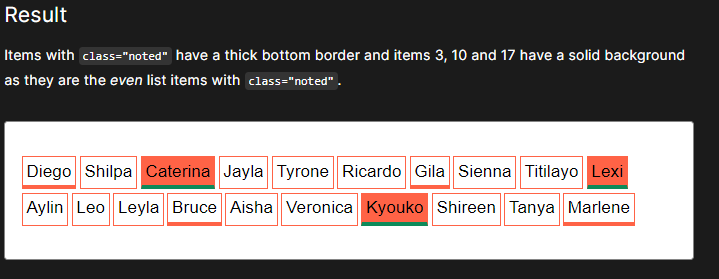
</ul>

li:nth-child(even of .noted) {

background-color: tomato;

border-bottom-color: seagreen;

}



EXAMPLE:

<div class="container">

<div class="my-div" data-text-before=" I am before" data-text-after="I am after">

<div class="real-child">I'm a real child</div>

</div>

</div>

.container {

width: 900px;

margin: 50px auto;

font-family: "Arial", sans-serif;

}

.my-div {

display: flex;

align-items: center;

margin: 0 auto;

width: 600px;

height: 400px;

background: #dbe8ff;

}

.my-div:after,

.my-div:before,

.real-child {

display: block;

padding: 20px;

border-radius: 6px;

width: 100%;

}

.my-div:after {

content: attr(data-text-after);

color: white;

flex: 0 1 auto;

margin: 20px 20px 20px 10px;

background: #00478e;

}

.my-div:before {

content: attr(data-text-before);

flex: 0 2 auto;

margin: 20px 10px 20px 20px;

background: #fff09b;

}

.real-child {

background: #a6ff72;

}

