CSS Pseudo-Classes

Introduction

Pseudo-class syntax

Anchor pseudo-classes

    Active pseudo-class

Hovering over elements

First-child element

First-of-type selector

Nth-child

Nth-last-child

Final thoughts

Resources

Action Items

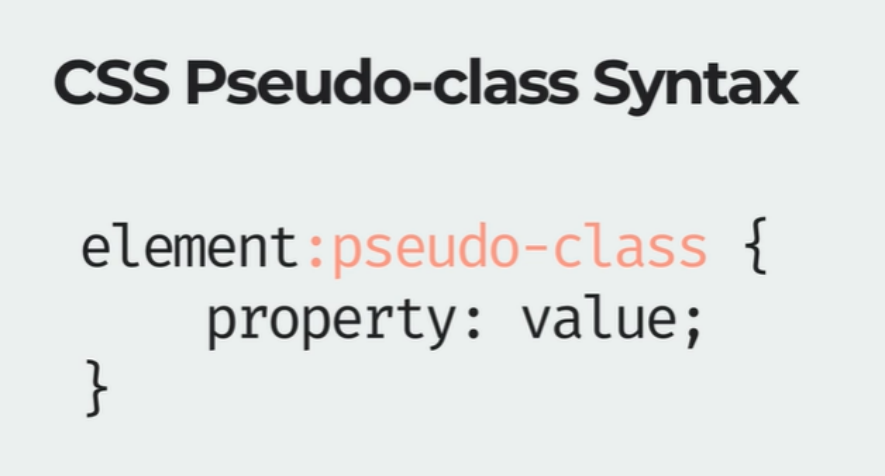
Introduction

Welcome to the lesson about the CSS pseudo-classes. In the previous lesson, we looked in detail into creating CSS selectors, and that included using a CSS class. But we also have an additional way to select things by using what is called a pseudo-class.

So pseudo-classes in CSS are a type of selector that selects elements that are in a specific state. For example, they are the first element of their type or they are being hovered over by the mouse pointer. And there are over 30 different pseudo-classes.

Pseudo-class syntax

But I am going to explain just a few which I think are the most popular and useful, and the syntax is that they use a colon. So you have your selector, colon, pseudo-class, and then the rest of the ruleset with the declaration and a CSS property and value.



Let me show you an example.

Anchor pseudo-classes

First, we'll look at anchor pseudo-classes, which are pseudo-classes used with links. So as you can see here, I've changed our structure a bit, and we now have links to four different pages in the footer.

```
        <h1>Hello!</h1>

        <p class="subtitle">Welcome to my website</p>

        <ul>
            <li>List item</li>
            <li>List item</li>
            <li>List item</li>
            <li>List item with children
                <ul>
                    <li>sub-item</li>
                    <li>sub-item</li>
                    <li>sub-item</li>
                </ul>
            </li>
            <li>List item</li>
            <li>List item</li>
        </ul>
    </div>

    <footer class="content">
        <h2>My links</h2>
        <ul>
            <li><a href="index.html">Page 1</a> - link 1</li>
            <li><a href="page2.html">Page 2</a> - link 2</li>
            <li><a href="page3.html">Page 3</a> - link 3</li>
            <li><a href="page4.html">Page 4</a> - link 4</li>
        </ul>
    </footer>

</div>

</body>

</html>
```

Hello!

Welcome to my website

- List item
- List item
- List item
- List item with children
  - sub-item
  - sub-item
  - sub-item
- List item
- List item

My links

- Page 1 - link 1
- Page 2 - link 2
- Page 3 - link 3
- Page 4 - link 4

But if we go into the Style CSS file, you will see that I've included four different pseudo-classes for the "a" elements.

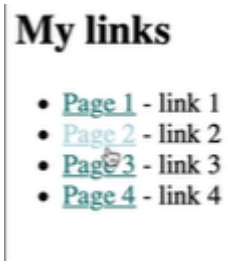
```
/* unvisited link */
a:link {
  color: teal;
}

/* visited link */
a:visited {
  color: red;
}

/* mouse over link */
a:hover {
  color: lightblue;
}

/* selected link */
a:active {
  color: purple;
}
```

/*unvisited link */

```
a:link {color: teal;}

/*visited link */

a:visited {color: red;}

/*mouse over link */

a:hover {color: lightblue;}

/*selected link */

a:active {color: purple;}
```

So we have a:link, and link is a pseudo-class that targets links that were not visited. So if in your browser history the browser doesn't recognize the URL that's provided inside the "href" attribute in a link then it will use the indicated color - so that's our teal color.

And “hover”, you probably already know that, the hover pseudo-class selects elements that we are hovering over with the mouse.



The “visited” pseudo-class selects links whose URLs are inside the browser history (as opposed to the “link” pseudo-class). When the browser recognizes that you've already visited that URL - by default, they have this purple color. When you, for example, Google something, in Google page results you would sometimes see that some of the websites are using that default purple color that's your visited link color.

Active pseudo-class

And finally, we have the “active” pseudo-class, and this one may be confusing. You may think that it's for your active menu items - where you are on a certain page that using

the active pseudo-class would somehow select that link corresponding to that page - but that's not the case.

That “active” pseudo-class is for when you click when the link is in an active state, meaning when you are doing the click - that's the “active” pseudo-class.

Hovering over elements

And what's interesting about hover - or maybe not interesting but important to note - is that the pseudo-class doesn't need to be the last portion of your selector.

So for example, if I would like to change the color of my H2 when I hover over the footer selection. I remember that this footer section had a class “content”, so if I hover over the content I can change its background, for example.

```
.content:hover {background: lightblue;}
```

So we can see that it works. You see, when I hover over my content section, the background changes, but I can also target elements inside. I'll use that selector as a parent container and target my H2 inside and also change its color - #fff for white.

```
.content:hover h2 {color:#fff;}
```

And now if I hover over my section, my H2 color changes. It's very useful to know that - and in Divi, I use it quite a lot when I want to hover over a column, for example, and I want to change the blurb color inside or a button inside.

So hovering over the parent container, you can target the descendant element.

First-child element

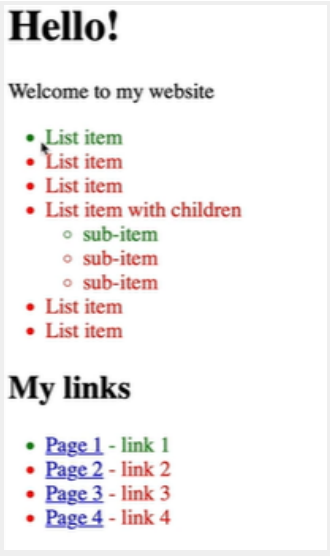
Another type of pseudo-class is the first-child element. So if we wanted to target our list item and say we want the color to be red.

```
li {color: red;}
```

Okay, so that makes all the text inside our list red except for links, because browsers have default colors for links. So that doesn't apply to links, but the regular text will be affected.

But we can use the “first-child” pseudo-class and say we want the first child to be green.

```
li:first-child {color: green;}
```



And that includes the first element here. But also if we go to see our page source, that first sub-item would also be considered a first child because it's basically the first child inside the parent.

First-of-type selector

Now, there is a similar selector, first of type, and that one works a bit different.

So let's say I want to target the "p" paragraph, that's also the first child.

`p:first-child {color: red;}`

And this will not affect my "p", because if we look at our page source, the "p" is not the first child of the parent container (a div which classes “content” and “sticky”).

The first child is H1, so this pseudo-class won;t work to target the p element. But we can use the “first-of-type” pseudo-class, which selects the first element of its type, first of type.

`p:first-of-type {color: red;}`

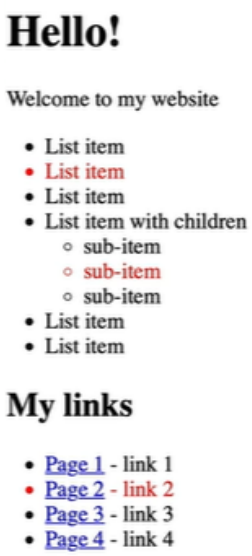
And now it applies to this "p" tag, even though it's not the first child of that div.

Nth-child

Okay, so we have first child, first of type, but we also have nth-child. So let's go back to our li and let's do nth-child and then in the brackets, we can specify the number.

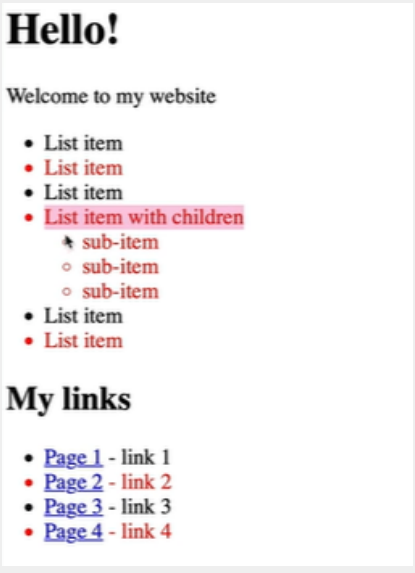
```
li:nth-child(2) {color: red;}
```

So let's say I want to select the second list item. You see that will apply to the sub ul as well, and to the main ul.



I can do every second using the n after the number:

```
li:nth-child(2n) {color: red;}
```

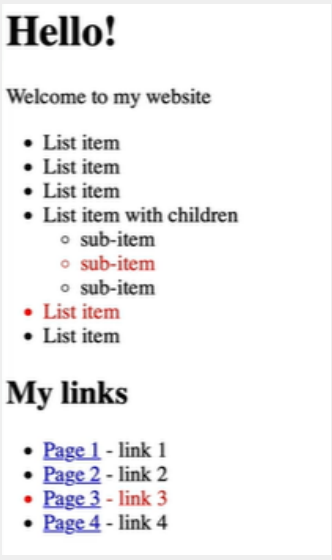


So second, fourth, sixth, and so on. And as I'm selecting a parent, that includes all the sub-item child elements as well.

Nth-last-child

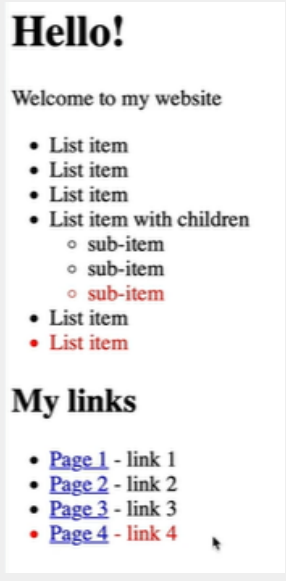
And another similar selector would be nth-last-child. And similarly, we can use a number here and that would count this from the nth. So second to last in this example.

```
li:nth-last-child(2) {color: red;}
```



Or consider this:

`li:last-child {color: red;}`



That would select the last one.

Or last of type, which does the same thing as the above:

`li:last-of-type {color: red;}`

But last-of-type would work on our "p": `p:last-of-type {color: red;}`

That's because it's last and also the first, but last-child will not work on that "p" because "p" is not the last-child. The ul is the last child of the parent container.

Final thoughts

I hope it's clear and helpful. I use these pseudo-classes all the time, so it's important that you kind of understand that you can apply them to any selector, not only at the end, not only to the thing you are targeting, but you can use that on the parent element. And also when you're targeting something inside that parent.

Resources

GET CODING:

:nth Tester

Action Items

- [ ] Choose an element on your website and try to target it without using its class or id. Try using the pseudo-class and order in which the element is displayed on a page. Repeat for as many elements as possible until you get familiar with the pseudo-selectors (try using different ones: nth-child, last-of-type, etc.)