CSS Selectors

Learning Objectives

- style elements based on their state, e.g. when hovering over them or setting focus on them
- understanding more complex CSS selectors like `pseudo classes` and `pseudo elements`
- combining selectors into complex rulesets

Id Selectors

Selects one element based on its **id**. Id selectors aren't reusable like class selectors and they are hard to overwrite.

```
#title {
  color: blue;
}
```

> ! We recommend not using Id selectors but class selectors instead.

Attribute Selectors

You can style elements depending on whether they have certain HTML attributes. This can be achieved with attribute selectors written inside `[...]` square brackets. Here are some examples:

- any element with the attribute hidden

```
[hidden] {
  ...;
}
```

- all links which open a new tab:

```
[target="_blank"] {
  ...;
}
```

- all elements with the class `card` and the attribute `role="list"`

```
.card[role="list"] {
  ...;
}
```

> 💡 There are many more of these attribute selectors which can be looked up in the MDN web docs.

Pseudo Classes

Sometimes we want to style an HTML element differently when it is in a specific state. This can be accomplished by **pseudo classes**. These are added to a selector and start with a **:** colon. Here are some examples of states that can be styled with pseudo classes:

- hovered elements

```
h2:hover {
  ...;
}
```

- active elements like a pressed button

```
button:active {
  ...;
}
```

- links that have been visited

```
a:visited {
  ...;
}
```

- form input that has received focus.

```
input:focus {
  ...;
}
```

- elements which are the first child in another element

```
li:first-child {
  ...;
}
```

- elements which are the nth child in another element. **n** is the argument that you can replace for example with a number or the words **even** and **odd**.

```
li:nth-child(n) {
  ...;
}
```

> 💡 There are many more of these pseudo classes which can be looked up in the MDN web docs.

Pseudo Elements

In comparison to **pseudo classes**, **pseudo elements** let you style a specific part of the selected elements like the first line of a paragraph, the first-letter, the selection etc. Similar to pseudo classes they are written with **::** double colons directly after the original selector.

- this selects the first line of paragraphs

```
p::first-line {
  ...;
}
```

- this creates pseudo elements as the first child of the selected elements

```
a::before {
  content: "🌍"; // property needed, can be empty
}
```

- this creates pseudo elements as the last child of the selected elements

```
a::after {
  content: "📎"; // property needed, can be empty
}
```

> 💡 In the MDN dev docs you can find even more pseudo elements.

Combinators

Sometimes it is more efficient to combine multiple selectors instead of defining yet another CSS class. You can chain together multiple selectors to form rather complex CSS selectors which apply only in specific cases. In the example above we already combined three selectors: `a`, `:hover` and `::after`. But there are also other ways to combine selectors:

- `(space)` : a specific element somewhere inside another specific element (regardless of nesting level)
  - `h2 span`: any span inside an h2
  - `.card button`: all buttons inside an element with the class "card"
- `>` : targeting a direct descendant of another element

    - `h2>span`: all spans which are direct children of an h2
    - `.card>button`: all buttons which are direct children of an element with the class "card"
- `~` : any later sibling element after another element
    - `h2~span`: any span which is a later sibling of an h2
- `+` : the direct sibling element after another element
    - `.card+button`: a button coming directly after an element with the class "card"

Many combinators can be chained. Can you figure out which element would be styled by the following selector?

```
body section > ul[role="list"] > li::before {
  ...;
}
```

Resources

- MDN web docs: Attribute-selectors
- MDN web docs: Pseudo-classes
- MDN web docs: Pseudo-elements