**Container Query**

Previously, if you wanted to style a container according to the width of the container, styling had to be applied by adding or removing classes/styling by calculating the width of the container with JavaScript. A cumbersome task.

**Accent-color**

A CSS property that allows you to apply colors to specific form elements. Currently only supported by the following elements:

* Radio button
* Checkboxes
* Progress
* Range

Previously this type of styling had to be done by building a custom checkbox from scratch and then mimic the functionality of a real checkbox.

We are not able to style all elements inside each form element. Only the border and background color. So, for example in regards of the checkbox, we cannot apply a custom color to the checkmark. The reason, as far as I found out, is to support browser color schemes and ensure an optimal contrast and optimal support for darkmode contrasts. Gradient colors are also no supported.

Even though not being able to do more than apply just one color, this is a big improvement and will make many things a lot easier!

**Scroll-snap-type**

The ability to snap to a specified element when you scroll.

**Aspect-ratio**

By setting the aspect-ratio property we can control the width and height without further calculations. This is especially useful if you have round elements like for example a round close button. Or if you want to fit an image inside a 4/3 container.

**CSS class nesting and chaining**

Until recently CSS class nesting and chaining was only available through a CSS post processors like SASS, LESS and similar. This can now also be achieved with plain CSS. But since this feature was released two months ago so support is still not 100%. Desktop is well supported but mobile support still lacks behind. Powerful feature but can very easily be overdone and lead to hard-to-read styling and very specific CSS selectors, that can be difficult to override later in your project.

**scroll-timeline**

Apply keyframe animations to your scrollbar. Previously this would require a lot of JavaScript but can now be achieved with three lines of CSS:

#container {  
 scroll-timeline: --squareTimeline y; /\* timeline id and scroll direction\*/  
}

#square {  
    animation-name: rotateAnimation; /\* apply keyframes animation \*/  
    animation-timeline: --squareTimeline; /\* bind to container \*/  
}

So this provides us a lot of possibilities to do scroll effects and parallax backgrounds, and I decided to play around with the scroll-timeline CSS property! [TIMELINE PARALLAX DEMO]