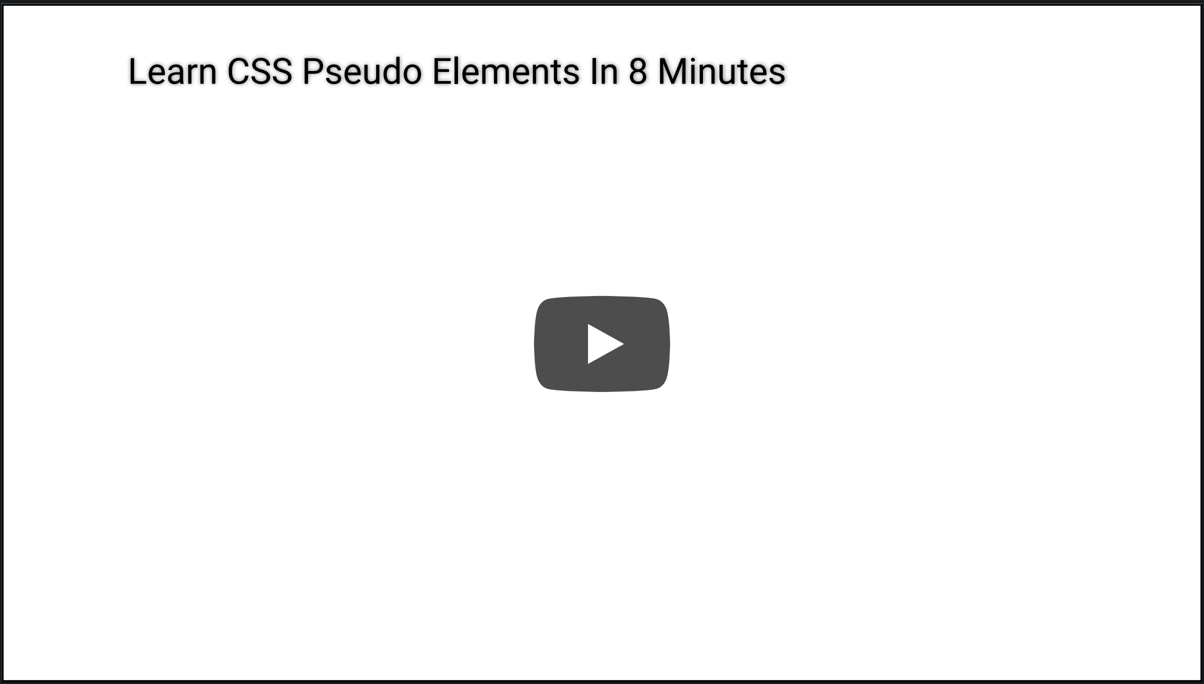
CSS Pseudo Elements Ultimate Guide

December 27, 2021

CSS

You can technically create any CSS design without the use of pseudo elements, but doing so is much harder and leaves you with messy code. Pseudo elements are amazing since they allow you to add/modify the HTML code of your site from CSS which means your HTML can stay clean and the CSS can add the extra content it needs. This may seem really confusing but it will all make sense once we talk about the first type of pseudo element.

If you prefer to learn visually, check out the video version of this article.



What Are Pseudo Elements?

I briefly mentioned that pseudo elements are ways for you to add/modify your HTML from CSS, but it is actually a bit more than that. There are really two styles of pseudo elements.

The first and by far most popular type are pseudo elements that allow you to add content to your HTML from CSS. There are only two of these pseudo elements (before and after) and we will be spending most of the time in this article on these pseudo elements.

The other style of pseudo elements allow you to treat specific sections of your HTML as if they were their own element. Good examples of this are the first-letter and first-line pseudo element that allow you to style the first letter or line of text as if it was its own element. Another example is the selection pseudo element that allows you to style the text that is highlighted. Try highlighting any text in this paragraph and you will see this selector in effect. Below is the CSS I used for this effect.

```css
.class::selection {
  background-color: red;
  color: white;
}
```

You will notice that the selection pseudo element has two colons before it. This is how you define every single pseudo element. You put two colons at the start of the pseudo element and then put the pseudo element itself.

Before/After Pseudo Elements

In order to add content to your HTML you need to use the before and after pseudo elements. These pseudo elements allow you to add one child elem as the first and/or last child element of whatever element you are selecting in your CSS.

```
.class-name::before {
  /* This is the first child of .class-name */
}

.class-name::after {
  /* This is the last child of .class-name */
}
```

In the above code we are adding two new elements to the page. The before element is the first child and the after element is the last child. You will notice if this is all you do, though, nothing actually changes in your HTML. This is because the before/after pseudo elements require the `content` property to be set to a value in order to show up on the page. This content property defines what is put inside the new child element.

```
<div class="class-name">
  <span>First Child</span>
  <span>Second Child</span>
</div>
```

For the next example assume we start with the above HTML and then apply the below CSS.

```css
.class-name::before {
  content: "New First Child";
}

.class-name::after {
  content: "";
}
```

```html
<div class="class-name">
  <span>New First Child</span>
  <span>First Child</span>
  <span>Second Child</span>
  <span></span>
</div>
```

You can now see that even though our original HTML only had 2 children our actual HTML that the browser sees has 4 children. Two of the children come from the HTML and the other 2 come from the CSS. If you inspect your page using the browser dev tools it will probably look something like the below example. I just wrote out what the actual HTML would look like in terms of how the browser actually renders the content.

```html
<div class="class-name">
  ::before
  <span>First Child</span>
  <span>Second Child</span>
  ::after
</div>
```

You will also notice in our CSS that the content property for the after elem
is an empty string. This is very common to do and just means we don't wa

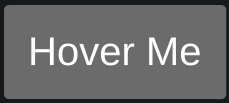
Privacy - Terms

show any content in that element since we will instead do our own styling. This is useful when you want to create shapes with CSS like the triangle that shows up at the bottom of a tooltip.

Why Use Before/After Pseudo Elements

It may seem like these elements are useless, but they are really useful when you want to add specific content to specific HTML elements without repeating that content in your HTML.

For example, you can create the below button with a tooltip by using a single HTML element and having the CSS take care of all the tooltip code.



```html
<button data-tooltip="Hovered">Hover Me</button>
```

```css
[data-tooltip] {
  position: relative;
}

[data-tooltip]::before {
  content: attr(data-tooltip);
  position: absolute;
  left: 50%;
  bottom: calc(100% + .25rem);
  transform: translateX(-50%);
  background-color: blue;
  padding: .25rem .5rem;
}
```

If you want to learn more about the `attr` function used in this example check out my data-attributes CSS article.

If I didn't use pseudo elements in the above example then I would need to create a separate HTML element in my HTML that is just for the tooltip and that can get really messy and is prone to errors. This is why pseudo elements are much better.

Other Pseudo Elements

As I mentioned at the beginning of this article there are 2 styles of pseudo elements and while the before/after elements are the most common pseudo elements you will use there are still other useful elements you can use.

`::first-letter`

The first-letter pseudo element selects the first letter inside a p tag. You can actually see it in action at the start of this paragraph. This is great for adding special styling to your first letter like some books do for the first letter in a chapter. One thing to note is that only some CSS properties can be used when styling the first-letter. For example you cannot change things like the display or position properties, but you can change things like padding or color.

```css
.class-name::first-letter {
  font-size: 2em;
  color: red;
  font-weight: bold;
}
```

`::first-line`

Similar to the first-letter pseudo element, the first-line pseudo element lets you change the style of just the first line of content in any block level element. This pseudo element is even more restricted on which CSS properties you can use. You cannot use things like margin or padding, but you can change pretty much anything to do with color or font.

```css
.class-name::first-line {
  color: red;
}
```

`::selection`

We have already covered this pseudo element, but essentially it lets you style the highlighted content on a page. Again you can only use certain CSS properties. Pretty much only properties that modify the color or text-decoration are allowed. You can highlight this paragraph for an example.

```css
.class::selection {
  background-color: red;
  color: white;
}
```

Conclusion

Pseudo elements are great for cleaning up your HTML while still having advanced CSS designs. The before and after pseudo elements will be the
you use the most, but there are many other pseudo elements, even beyon

the ones covered in this article, that will make doing specific things much easier.