**Combination selectors**

## Question

What is the difference between Adjacent and Sibling combinators?



Adjacent combinators will only target the first element after the specified elements while Sibling combinators target all elements of the same type that follow the specified element.



Sibling combinators will only target the first element after the specified elements while Adjacent combinators target all elements of the same type that follow the specified element.

Correct

That's right. They both affect elements that follow a specified element. Adjacent combinators will only target the first element and Sibling combinators target all elements of the same type after the specified element.

Sometimes you want to apply the same styling to different elements or

to different groups of elements.

CSS makes it possible to combine more than one selector so

that you can apply rules to elements based on their relationship with one another.

You do this with combination selectors, and there are four main types.

Namely, descendant selectors, child selectors, general sibling selectors and

adjacent sibling selectors.

In this video you will learn about combination selectors and

how you can use them to target more specific elements.

Let's start with descendant selectors.

Descendant selectors are useful if you need to select HTML elements that

are contained within another selector.

The easiest way to remember descendant selectors is to think of a family.

You pick a person in the family and then select all descendants, such as children,

grandchildren and great grandchildren, who have, say, brown hair.

For example, you can have an HTML file with a div tag, and

its ID attributes set to blog, inside the div tag are a few heading tags.

In this example,

all h1 elements are descendants of the element with the ID blog.

So to make all the h1 descendants blue, you can add the ID selector hash

#blog followed by the h1 selector with a space in between and

then the properties in curly brackets.

Child selectors on the other hand are more specific than descendant selectors.

To go back to the family example, with child selectors you select a person and

then select only their immediate children who have red hair.

Let's update the previous selector to a child selector by adding a close angle

bracket in between the two selectors.

The effect is that now only the first h1 element turns blue.

This is because it is the only h1 element

that is a child of the element with the blog id attribute.

Next, let's focus on general sibling selectors,

which apply rules to all the selectors of the same type that follow the first type.

To use the family analogy again, general sibling selectors allow you to select

a person and then select all younger brothers with black hair.

You can update the selector with a tilde in between the two selectors to make it

a general sibling selector.

The selector now targets all paragraph elements that are siblings of h1 element,

the result is the last two paragraph elements turn blue.

Notice that the first paragraph element didn't turn blue,

this is because the general sibling selector only select subsequent

elements that match the selector.

In comparison to sibling selectors, you can use adjacent sibling selectors to

target only the first or adjacent element that follows another element.

Once again, if you think about a family, this selector selects a person and

then selects the next younger brothers.

Finally, let's replace the tilde with a plus sign to update this selector to

an adjacent sibling selector.

The selector now selects all paragraph elements that are adjacent siblings

of the h1 element.

The result is that only the middle paragraph element turns blue.

This is because the adjacent sibling selector only

selects the first subsequent element if it matches the selector.

Now that you are more familiar with combination selectors,

let's explore a few practical examples.

I will now demonstrate two combination selectors,

the general sibling selector and the adjacent sibling selector.

Let's start with the general sibling selector.

Remember, the general sibling selectors select all the matching elements of

the second type which follow the first type.

The html file has a paragraph tag with the words, Basic Info About Little Lemon,

and that is followed by a div tag with the word, About.

After the div tag, there is an ordered list with three list items.

This structure repeats below.

So there is another paragraph tag, a div tag, and

then an ordered list with two list items.

Now, let's inspect the CSS rules.

First, I set basic rules for the html body.

This is so

that the special effect that I will apply with the sibling selector will stand out.

Below the styling for the body is the general sibling selector.

I set the div tag as the parent and the un ordered list as the sibling.

Remember, for the sibling selector, you always use a tilde.

This time I specify a contrasting background color, white.

Then notice that I added the box shadow effect.

As the name indicates,

it adds a box with a shadow around the element that you target.

The box shadow property consists of the horizontal offset,

set to one pixel, the vertical offset, set to one pixel, and then the blur radius,

which I set to three pixels.

Thereafter, you need to specify the color of the shadow,

which in this case is set to gray.

So what elements will be styled?

Remember, sibling selectors apply rules to all the selectors of the same type that

follow the first type.

Let's go to the html again to identify what will be styled.

The first div tag is the reference.

So the rules will target the un ordered list that follows it.

And again, the second div tag will also serve as a reference and

the rules will affect the un ordered list that follows it.

Let's inspect the output.

The background is orange and there are two box shadows,

each with un ordered list inside it.

Notice that the subheadings that are inside div tags, About and Employee or

Owner, just have the body styling.

The sibling selector only targets the items that follow the div tags.

By now you should realize that you can use the sibling selector to

efficiently apply the same styling to several elements with one rule.

If there is another instance where an un ordered list follows a div tag on this

page, it will also have a box shadow.

But say the page has other un ordered lists that do not follow a div tag,

the styling will not be applied to those elements.

Next, I will demonstrate how to use the adjacent selector.

The html file contains an image tag and then three paragraph tags.

Thereafter, there is another image tag followed by a div tag and

then a paragraph tag.

Lastly, there is another image tag followed by a paragraph tag.

Notice that the first paragraph after the top and

bottom images are descriptions of the images above it.

So I want to style these paragraphs so that it looks like captions.

In other words, I want to style all paragraphs that follow an image but

not affect any other paragraphs or other elements like a div tag.

And I can do this with the adjacent selector.

Because remember,, the difference between an adjacent selector and

a sibling selector, is that the adjacent selector only targets the first element

following the reference element.

Whereas the sibling selector targets all the elements of the same

type after the reference element.

Let's inspect the output and CSS code.

As explained, the first paragraph that follows the image is styled differently

than the next two paragraphs.

Notice that I set up the styling in the adjacent rule so

that it will appear as a caption.

The text is smaller and aligned in the center.

Again, with one rule I applied styling to two captions.

If you have a site with many images and captions,

the adjacent selector will be very helpful and save you a lot of time.

In this video, you learned about combination selectors and how to use them.

As you become more advanced in styling your Web pages, these CSS selectors will

surely be helpful in targeting specific items and save you a lot of time.