Block vs. Inline Elements

[The screen shows a blank background with text saying "Block and Inline Elements".]

Female: Block and inline elements. Remember, every element is inside its own invisible box. This box will either be a block or an inline element. Block level elements by default take up 100% of the space that they have, stretching from right to left as far as they can.

[The screen now shows an image of block elements. The window has rectangles that are vertical to each other, one on top of the other.]

This means they will not allow other elements to share the line that they are on. So they appear to start on a new line in the browser window. These include these types of elements.

[Text appears on the screen with a list of elements. The list contains "div", "header", "main", "footer", "nav", "p", "ul", "li", "section", "article", "form", and "h1 - h6".]

Inline elements by default, will allow other elements to share the same line as long as there's room, then they will only wrap to the next line, once it's full.

[The text and image of block elements is replaced by an image example of inline elements. The rectangles this time aren't as long and are side by side as long as there is room. If there is no room they go to the next line.]

They will only take up as much width as necessary. These include these type of elements.

[Text appears next to the image that contains a list of elements. The list contains "a", "img", "span", "button", and "label".]

The h1 element in all the paragraphs here are block level elements.

[The text and image is removed and replaced with an article. The title is "Ways to Stay Warm Outdoors". Below the title are four paragraphs that give tips about how to stay warm. The first and third paragraph have a background color of gray.]

If I place a border around them, you can see that they take up all the space they can.

[A red border encircles each individual element. The paragraphs with a gray background color have their backgrounds completely contained in the boxes.]

However, if we place some images side-by-side, as long as there's room, they will share the horizontal space available.

[The image of the website changes to one that is lower down on the website. Only the last two paragraphs are shown. Below the paragraphs there are four images. The images are a blanket, a fire that is on a wheely table, hand warmers, and a parent sledding with their kid. The images are all on the same line.]

You can change the defaults of inline and block elements with the display property. If I wanted to make a block element into an inline element, I would target it in CSS and use the display in my declaration.

[The image is removed and replaced with three images. The first one has a bulleted list that has four steps on how to stay warm. The list is vertical. The second image shows CSS for an element that has the id "warm_list" and is an <li> element. Inside the curly braces is the code "display: inline;". The third image shows the list again, but this time it is all on the same line.]

If I wanted to make an inline element into a block level element, I would use the display block declaration.

[The images are removed and replaced with two images. The first image is CSS that selects elements that are in <main> and are an <img>. Inside the curly braces is the code "display: block" The second image is the four images of staying warm in winter that were on the website. This time they are all on their own line.]

There's also one more display property value, inline-block. It's kind of a mix between the two. It would cause a block level element to flow like an inline element, but it will retain other features of a block level element, like allowing you to set widths and heights on the element and top and bottom margins would work more predictably. The declaration you would use is display inline-block. Let's demonstrate the difference here. Here we have three spans, one inline, one inline-block, and one block.

[A new image of a website appears on the screen. On the left side is HTML and CSS that was used to make the website. The title of the website is "The display Property". Beneath the title are examples of inline, inline-block, and block displays. All the different displays have the same width, height, border, and background color. The inline example has the words "Aliquam" and "venenatis" highlighted and it contains a border. The height and width of the words follow the height and width of the other words in the paragraph. They are the same line as the rest of the paragraph. The inline block example has the same words in the same part of the sentence highlighted and bordered. Except, the width and height are what was set in the CSS. They are on the same line as the rest of the paragraph. The block display example has the same words highlighted and bordered. Except, the words have the width and height that was set in the CSS and they are each in their own line.]

Notice the difference between the three. Even though the widths and heights are all set the same between the three, the in-line width, heights, and margins, padding and bottom are not always what is expected. Inline block is sometimes a better choice for a simple way to align a few items side-by-side. Because of these different display values. Centering elements on our page works differently with different elements. Two common ways to center elements are text-align center, margin 0, auto. Text-align center will center the contents of the box. So if I put text-align center on the H1 or the p, then the content will get centered inside that invisible box or container.

[Three new images replace the previous one. The first shows the website titled "Ways to Stay Warm Outdoors". None of the text is centered. The second shows CSS that selects both the <p> and the <h1> elements. Inside the curly braces is "text-align: center;". The third image shows the same website, except all the text is centered inside their respective boxes.]

Margin 0 auto will center the box itself. The first value refers to the top and bottom margin. In our case, we'll just leave them 0. The second value refers to the right and left. It will automatically give an equal gap on each side of the box. With margin 0 auto, if the width of the element takes the whole space, you won't see the centering. There needs to be a width set, so there is space on each side of that side to automatically give an equal measurement on each side. Or in other words, the box needs a width. Otherwise, we'll take up the whole width of the page. So if I give the h1 and p a width, the contents are still centered inside the box with the text-align center. But I could use margin 0 auto to center the box itself.

[The images are replaced with two images. The first image shows the CSS that used "text-align: center", but now it also contains "width: 800px;", "margin: 0 auto;", and "border: 2px solid red;". The second image shows the website with its new changes. The text is still centered within its respective box, but the boxes are 800px and centered on the screen.]

If I tried text-align center with an image which is an inline element, it doesn't appear to center.

[The images are replaced with two images. The first is an image of a blanket. The blanket is on the left side of the window. The second image is of the image's CSS. The CSS adds a border and uses "text-align: center;"]

That's because its border is tight around the image. The content is the image, and it's already centered inside its type border. But we can change the image to display block, give it a width, then we can center it with margin 0 auto.

[The images change. The blanket is now in the center of the window. The CSS now contains "width: 300x;",

“display: block;”, and “margin: 0 auto” in addition to the previous code that was there.]

Or we could place the image inside of a block level element and then use text align center to center the contents of the element, which would include that child element.

[The images change again and a new image is added. The blanket is still centered in the window, but there is a new border that goes along the whole width of the window. The new image has text that is using HTML. The text shows the image element is inside a <div> element with the id of “img_container”. The third image shows the CSS for the div. It selects the div using “#img_container”. The code inside the curly braces is “text-align: center;”.]

[End of video.]