***Relative and Absolute Element Positioning***

[SOUND] Previously we spoke

about one type of alternative element positioning and that's floating elements.

I say alternative because it is It is alternative to the normal document flow.

This lecture, we're going to speak about a couple more positioning schemes, and

that's relative and absolute.

These positioning schemes allow you to specify precise offsets to move

the target element to the different part of the page origin.

Play video starting at ::31 and follow transcript0:31

First let's talk about static positioning.

Static positioning is basically a different way of saying normal

document flow.

It is actually a default setting for all elements, except html.

And if you try to apply positioning offsets on elements

who's position property is set to static, the offsets are just ignored.

Play video starting at ::51 and follow transcript0:51

A different type of positioning scheme is called relative positioning.

When you apply position relative on an element.

The element is positioned relative to its position in the normal document flow.

In other words, if you were to apply offsets on that element, there will be

offset, from the original normal document flow position of that element.

So the positioning CSS offset properties are top, bottom, left, and right.

So when you position element as relative you're basically creating like an anchor

for the offsets.

So the elements top, bottom, left,

and right edges become the boundaries from which you offset the element.

The important thing to know about relative position is that the element

that is set to relative positioning is not taken out of normal document flow.

In fact even if it's moved by using the offsets, its original spot is preserved.

So as far as the rest of the HTML elements and

the rest of the HTML page is concerned, that element is still sitting in its

original spot even though visually its off somewhere else.

So let's give an example.

So here's the paragraph.

And it's basically represented by this orange box.

As soon as we set position: relative on the element,

its edges become almost like an anchor for future offsets.

Now when we apply the offsets, the element moves

relatively to its original position in the normal document flow.

So in this case it's moving top 50 pixels and left 50 pixels.

Now these values might be a little bit confusing since we're saying left 50

pixels but yet we're moving the element to the right.

So the way you could think about these offsets is really not top or

left but more like from the top and from left.

So if you take from left 50 pixels, you move 50 pixels to the right, and

if you take from top 50 pixels, you're really removing 50 pixels from the top.

Also note that nothing really changed about the document.

And other than this element moving.

The original space for that element still remains and the originally laid out

elements around that element still remain exactly the same,

because they think the element is still sitting in its original spot.

You can also use negative values for these offsets and

which one you use, bottom right or top left.

It's really more of a convenience for you.

Depending on the use case you're trying to code to.

So in this example we achieved the same result by using negative

bottom right values as we did by using positive top and left values.

Play video starting at :3:23 and follow transcript3:23

Now the idea of absolute positioning is that all offsets, top, bottom, left,

right, are all relative to the position of the nearest ancestor

which has positioning set on it other than static.

In other words, some parent, grandparent, on and on and

on, ancestor has to have its positioning set other than static, and

then the absolute positioning will actually start working.

By default,

HTML element is the only element that has non-static positioning set on it.

And it's actually set to relative.

Unlike relative positioning, the element is taken out

out of its normal document flow if it's positioning a set to absolute.

Play video starting at :4:2 and follow transcript4:02

Let's say we have this grey box and it's some kind of a container element and

it's position is set to relative.

Then we have this box number one, that's the one we're going to target and

some kind of box number two, some other element that's box number two.

So if we set the position absolute on number one, what's going to happen is,

it's going to be first of all, taken out of its normal document flow.

And it's going to remain in the place where it was without any other offsets.

So as you can see in this illustration, element two moved up

underneath element one since element one is no longer part of the document flow.

Element two doesn't think it's even there.

Play video starting at :4:41 and follow transcript4:41

Now if we do some offsets, and in this case we're doing bottom right offsets,

we're going to be relative to our container elements,

since that container element's position is set to relative.

And we're going to do from bottom 10 pixels, and

then from right 10 pixels over.

And that's how we end up with this number 1 box being in the bottom right

of our container element.

Play video starting at :5:3 and follow transcript5:03

A really cool feature of this combination is, that if your container element,

is itself offset, everything inside of that container is offset with it.

So you could set our particular layout with a container element.

And then move that container element, or offset it all around the screen,

without worrying about having to offset particular values,

of every element within that container element.

Play video starting at :5:28 and follow transcript5:28

Let's jump into the code editor and take a look at some examples of these concepts.

Play video starting at :5:34 and follow transcript5:34

Okay, so here I am in subline text and I'm looking at a file called positioning

before.html and it's located in the examples lecture 22 folder.

So let's examine real quick the structure of the HTML.

It's super simple.

We just have an H1 just announcing what we are, what we're doing here And

we have a div, which has an id container, and

then four paragraphs, each one having id p1, p2, p3, p4, and that's it.

And our initial styles are pretty simple as well.

First of all, we reset the margin and

padding as well as set our box sizing to border box.

We also gave the h1, since we reset the margin, we gave h1, the very first one,

a bit of a bottom margin 15 to push down the rest of the contents of the page.

Play video starting at :6:16 and follow transcript6:16

We gave our div container that familiar light bluish color, and

every paragraph tag is 50 pixels by 50 pixels, and it has a border black border

of one pixel, and also has a margin bottom of 15 pixels, every single one of them.

And every single one of them has just a different color just to differentiate

which one we're looking at.

So let's go ahead and take a look at what this looks like in the browser.

Play video starting at :6:40 and follow transcript6:40

So as you can see here's our four paragraph boxes.

They're sitting inside this light bluish container.

So the first thing we're going to do is we're going to work on this

first box right here.

And what we'll do is set it's position to relative.

When we do that, save and refresh, we see nothing really happened.

Well actually, what's going on here is that we've set an anchor so

now all offsets of this element are going to be relative to its

normal document flow position which still remains right here.

Let's go ahead and give it some offsets to simulate it going basically down to

the second row.

We'll give it top 65, so it's from the top, 65 pixels down.

And then from the left, 65 pixels from the left.

And we'll refresh the page.

And here we go so we move this element form here to here and as you can see this

element is not taken out of document flow, because te rest of the elements are still

sitting in their original spots and this space is still completely unoccupied.

Well the truth of the matter, the browser is thinking that it's actually

occupied by this element that we just moved over here.

So, that's the basic run down of irrelative positioning.

And next, we're going to go over an example of absolute positioning.

[SOUND] Let's

play around a little bit with the absolute positioning.

Let's take this third box which is colored cadet blue, even

though to me it looks a little bit green, let's take that and let's try to set

its absolute positioning to move it right back to where the maroon box, used to be.

So let's go to paragraph three and we'll set it's position to absolute and

we'll give it a top of 0 and we'll give it a left of 0 as well.

And when we save it let's see what happens when we refresh.

Play video starting at ::38 and follow transcript0:38

And, Voila, when we refreshed you could see it was taken out of the regular

normal document flow but it's not quite where we wanted it.

We wanted it right here in its original, in the spot where the maroon box used to

be, but yet it's flush, right flush with the browsers edges.

Now why is that happening?

Well let's take a look at our document structure one more time to

figure that out.

This is the box number 3 right here, represented by paragraph with ID p3.

Well, we remember that absolute positioning needs a relative or

an absolute parent or an ancestor.

Well, it goes up the chain and says is this element

something that has it's position set to either absolute or relative.

And the answer is no.

So it keeps going and it goes up to the body element.

And asks if this element have an absolute or a relative position set on it?

And the answer is no again.

So it jumps all the way to the html element

which has its position set to relative by default.

And that's how we end up with the box going all the way up top.

So how do we fix it?

Well, simple, we just need to anchor the ancestor or

the container of our paragraph tag with the position relative.

And any element contained within that container element

will then be positioned relative to its edges.

In our case, since we want this box to be relative to the containing div element,

we need to set the containing div element's position to relative.

Let's go ahead and do that.

We'll say position relative, we'll save it, we'll refresh, and

when we do it should jump right here where the maroon box used to be.

And exactly, so it does.

Play video starting at :2:13 and follow transcript2:13

Last but not least,

let's try to move the entire container element down a few pixels and see if

all the containing elements will move exactly in the same position down with it.

In order to do that let's go ahead and

go to our div container and we'll give it a top value of 60 pixels.

And once we save let's refresh And

the entire container moved down together with all its containing elements and

they're all positioned exactly the same as we positioned them before.

Play video starting at :2:44 and follow transcript2:44

So, in summary, static positioning is default for all elements, except html.

Html actually has it's positioning set to relative.

Play video starting at :2:53 and follow transcript2:53

Relative positioning offsets the element relative to it's normal document

flow position.

And the normal document flow is not affected what so

ever, by moving or offsetting the relative position element.

Play video starting at :3:6 and follow transcript3:06

Absolute positioning is relative to it's closest ancestor which has positioning

set to a non-static value, which is either absolute or relative.

The important thing to remember about absolute positioning is that the element

is taken out out of it's normal document flow.

And the rest of the elements basically behave as if that element doesn't exist.

Play video starting at :3:29 and follow transcript3:29

Also you saw, offsetting the relative container element offsets its contents as

well, which gives you great flexibility to move around a whole set of elements and

not just one.

Play video starting at :3:40 and follow transcript3:40

Next, we're going to talk about media queries.