# ***THE BOX MODEL***

In HTML every element is considered a box.

However, there's more to that box than just the content that the element wraps

around.

Let's take a quick look at a simple diagram.

Play video starting at ::16 and follow transcript0:16

Besides their [INAUDIBLE] content, each box consists of padding,

border as well as margins.

The box model refers to the components that make up an HTML box

as well as the rules that govern how these box components affect the layout

as well as how width and height of the box are calculated.

Let's jump into the code and take a look at some examples.

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

Okay, so here I am in subline text and

I'm looking at the file called box-model-before.html.

And it's located in the examples Lecture19 folder.

Let's quickly examine the HTML structure of this document.

It's actually pretty simple.

It just has one h1 here.

Play video starting at ::55 and follow transcript0:55

Has a div that has an ID box and inside of that div there is another

div with an ID content and has some Loren ipsum, basically some dummy text in it.

In addition to that I colored the box itself with a background color blue and

the content which is the div right here.

This content.

I colored it with this color which is basically light green.

And just so we have some more interesting colors in the background

I colored the body to be gray.

Play video starting at :1:27 and follow transcript1:27

Okay, so here is the representation of our page in the browser.

And since div is a block level element,

it tries to fill in the entire parent in terms of width.

So it's trying to fill up the entire width here in the browser.

But there's something strange going on if you take a look.

There's some space that's gone in here right before Lorem and

here at the end of our content beginning and there's spacing going on.

So let's take a look at the chrome developer tools real quick and

figure out what that space is about.

Play video starting at :1:57 and follow transcript1:57

Okay so we'll go ahead and choose the box, that's our box.

And we'll take a look as to what's going on.

There doesn't seem to be any spacing that's set on the box itself.

Well how about the body itself in the body tag?

Go and here we go.

See we'd realized that the body tag here has a margin all around of eight pixels.

Now where is that eight pixel coming from?

Or if we take a look here, it's telling you where it's coming from.

It's coming from the user agent style.

Style sheet.

Which means it's the browser itself.

It's the default browser styles.

So that's actually pretty common that the browsers do stuff like that.

And what we're going to do is in order for

it not to affect what we're doing we're going to reset it.

And it actually resets on the web out there,

plenty of resets on the web how to reset default browser setting.

So we'll go ahead and do margin 0 and

we'll also do padding 0 so we won't be affected by any of that.

And when we do that and we save and refresh, you see that now our content is

flush with the actual borders of the browser window.

Okay, so moving on.

So we have our content.

And you can see the context is green.

Now we're not seeing the background of the box the content is actually sitting in

because we would have seen something blue in the background and always see green.

And that is because the inner box, the inner content div

is covering up the outer one completely because they are the same size.

They both try to fill up their parent which is in this case the body tag.

Both try to fill it up all the way.

And since this one basically sit it on top or inside of this one,

it's covering up the blue background.

Let's go ahead and set some padding on our box.

Let's do padding, and we'll do 10px top.

10px right, 10px bottom, and 10px left.

Play video starting at :3:48 and follow transcript3:48

Now we set that and by the way, there's obviously a shortcut for this.

You could actually remove all this 10px right here and

that would mean exactly the same thing and I let you actually look up as to

all the different shortcuts that you could take, but I'll leave this here for now.

So point is, it's always goes top, right,

bottom, left and let's go ahead and save that and refresh the browser.

And voila, now we've seen the padding.

And what the padding is doing is basically giving some padding around our content and

our content being this div tag right here with the content text, is the one that's

being kind of squeezed all around and the padding is showing up all around and

now you can see the blue because that's the background color of our box.

Okay so next, let's give our box some border.

Let's go ahead and say border, and

we'll give that border that's going to be 3px thick.

Play video starting at :4:46 and follow transcript4:46

And it's going to be solid as opposed to dashed or something like that.

And there's other options here you're welcome to look up.

And we want to give it the color black.

Okay. When we save that, and

when we refresh the browser, you could see that the border showed up.

And in fact, let's make it a little bit thicker.

Let's make it a five pixels border.

And let's make it like that.

Okay and then last but not least, let's give it some margins as well.

Let's go to margin and let's go ahead and use the shortcut notation which is

giving you 40 pixels which means I'm saying I want 40 pixels all around.

So top, right, bottom, and left.

So if I go ahead and refresh that.

You could see now there's 40 pixels on every side.

And, well, you could see it on the bottom, per say, and

you can't really necessarily see it here.

But there's 40 pixels on every side.

Okay. Well, up until now,

we've sort of let the content and some of our margins, and so on dictate

how big the actual box is or what about if we actually set this width our self.

Let's set the width to be let's say 300 pixels and we save and

refresh the box became smaller but let's go ahead and examine that box.

Let's click on the inside Chrome Developer tools, now let's take a look.

So it's showing us that our box is in fact 300 pixels wide and

since we didn't restrain our height, it went ahead and wrapped around the text.

And basically the height will adjust unless we restrain it.

A manually by specifying the height,

it will adjust to however much it needs to fit.

So if we make it a little bit small, let's make it let's say 100 pixels,

we'll refresh you'll see the height will become a little bit bigger, the height

will become 72, because now we squeezed it enough that extra lines got added.

So now the content width is 100 but the height is larger.

Let's go ahead and put it back to 300 and refresh.

And now if you look at this breakdown

we come to our first pretty interesting point.

But first of all let's go through all the components.

Here this one is the pure content right here that I'm mousing over right here.

And then if I go up a little bit, it'll highlight for

me in the browser the padding that's going on there.

And if I highlight, if I go out a little bit more, I'll go straight up to

the border, right, and if you can't really see the margins if you roll over to

the margin side voila now you see what the margins are gone and even though to

the right side showing you a little bit more than really margin is going on.

But the 40 pixels all around still stands.

Now what's interesting to find is well how big is this

box exactly as we going to start putting more and more boxes right.

We going to start making layouts of different HTML components and

to get our UI just right, we need to find out how big is each box.

Well, we sort of specified that we want the box to be 300 pixels wide.

The problem is that it's not really 300 pixels wide.

Let's take a look.

If we take a look here it says 300 pixels wide, but we're forgetting there's

a border around it or the padding around it, and the padding is 10 pixels on one

side And ten pixels on the other side but that's not all, there's also a border

around it as well that is 5 pixels wide and 5 pixels wide on this side.

So really, it's 300 plus 10 plus 10 that's 20, 320, plus 5 plus 5 that's 330.

So if you look at the actual visible border of this box, this box,

this distance is not actually 300.

This distance is 330.

And the interesting part is, or kind of annoying part,

is that, depending on what border and padding you set, this width will change.

So let's take a look.

Let's say if I make the border something really crazy like 20 pixels.

So go ahead and refresh.

Now the border is 20 pixels so now my box end all the way there.

Play video starting at :8:36 and follow transcript8:36

And if I make the padding even bigger,

let's say I make the padding this one is the right so I make the padding 30 and

on the left I make the padding 30 as well, and I take a look at the box.

Take a look at how it's going to whoa, it became bigger again.

So even though we said I want the width of the box to be 300,

it turns out that we aren't really setting the width of the box,

we're actually setting the width of the content.

And that's this piece right here in the green.

Well it turns out for layouts, and just in general sanity this is kind of annoying.

You really want to change that and CSS3 actually did change that,

it turns out that by default, the box sizing is actually a property of

every HTML element and by default it's actually set to content box,

which means when you specify it's height and width,

you're specifying the height and width of the content box, not the entire thing.

Well CSS 3 came out with a new value for that property,

and that's called border-box.

Let's go ahead and set that right here.

We'll say box-sizing, and we'll say border-box.

And when we refresh let's take a look, look at that, it became smaller.

So now, if you take a look at the breakdown,

you could see that even though it was specified the width would be 300.

That width is guaranteed to be the width of this box right here

from the edge of this border.

To the edge of this border.

So if you take a look here, 200 plus 30 plus 30,

that's 260, plus 20 plus 20, that's another 40, that's 300.

And if we change it back to content-box,

which is the default, it will jump back to its original size.

But the truth of the matter is, all the modern frameworks like Bootstrap and

others use box, border box as its sizing mode,

and as you plan and work with your layouts this is the same choice to make.

So, make sure you always stay with a box sizing of border box.

[SOUND] Okay, so in the previous segment we spoke

about the fact that you should always try to use

box sizing border box as your box sizing model.

Now, of course, you don't want to specify box sizing border box on every

element in your entire HTML, so you definitely want to specify it once and

then have it be inherited everywhere.

So let's go ahead and try to do that.

Let's remove this border box from here.

And body seems to be like a natural place to put it since

body is the top level element and it should inherit everywhere else.

So if I place the box size and border box here and

then I refresh, this should stay the same.

It didn't work.

Now, why it didn't that work?

It jumped back to the content box.

Well, the reason that didn't work is because

box sizing is one of those CSS properties that is not inherited.

You can't set it on the parent element and

then expect that the child elements will inherit that property.

So how do we get around that?

Well, the way we get around it is by learning about one more selector.

And that selector is the star selector.

Let's go back up here and we'll go ahead and say star.

And then we'll cut and paste that border box property right here.

Now what is the star?

First let's refresh to see if it worked.

And sure enough, now it's working.

What star does, star selector says is, go ahead and

select every element there is and apply these particular CSS properties to them.

The difference between star and placing some property in a parent element for

example is that star says I am not asking you to inherit anything,

I am saying select every element.

Which means it's as if it took this property and this value and

then it went to the HTML and it applied it to every element it saw, it applied

that particular property and value, which in this case is box-sizing: border-box.

So that's how properties like box-sizing are applied across your entire page.

Now I told you that border-box is a property that's new in CSS3.

Whenever you hear something like that an alarm bell should go off.

Does every browser that I deal with support border-box?

Well, we know just the resource to check that out.

And we'll go to the website called www.caniuse.com and

we'll type in border box.

Border Box is a CSS3 box sizing property, and if we take a look, excellent!

Every single browser is green.

So, every single browser fully supports this property value.

Okay, so next, let's close this up and

let's try to advance our margin understanding a little bit better.

Play video starting at :2:45 and follow transcript2:45

Okay, so let's take a look at this diagram.

We have two boxes next to each other, in other words two elements, and

the gray part is the margins.

Well, margins that are left to right are cumulative.

So if I have a margin, right margin of 40 pixels on the left box and

a left margin of 50 pixels on the right box.

The cumulative margin will be 90 pixels.

40 plus 50.

And that's pretty simple and this is exactly what you would expect.

But what happens if you have one element on top of the other element.

Which let's say these are two divs and each one is specifying a margin.

So one is specifying margin of 30, margin top for the bottom.

For the bottom element margin top is 30, and for

the top element margin bottom is 20.

Would you expect this to be 50 pixels, the cumulative?

Well, the answer is it won't be, the answer is the margins collapse, and

the larger margin wins.

So the actual distance between the border of one element and

the border of another element will be, in this case,

30 pixels, meaning the larger margin will win and the other one will just collapse.

Let's take a look at an example inside our code editor.

Play video starting at :3:58 and follow transcript3:58

Okay so going back to the same file Box Model before that HTML,

we can take a look and see that we have a h1 here and

we defined a margin-bottom of 30 pixels on h1.

Let's go ahead and go to our box and recture we'll move the margin altogether.

And refresh, will take a look what it is without the margin.

So, you could see there's some margin going on here and

that margin is coming from this h1 here margin bottom.

Let's go ahead and define a margin top so,

when you need a margin bottom is on h1 here, when you need margin top on our box.

Let's define a margin top of 20 pixels and

when we do that when we refresh absolutely nothing will happen.

And the reason that is is because the box is defining a 20 pixel margin.

Yet, the h1 ia already defining a 30 pixels margin.

So when they come together and touch,

they collapse into whatever the larger margin is.

So in this case, that's 30.

However if I made the margin to be 40 and

let's say even 50 pixels now you'll see that the box will move down because now

the box has a larger margin than the one specified on h1.

So there is now a bigger distance between the box model h1 and

the actual box and it's actually 50 pixels.

We can also notice, by the way, that the box model,

the words box model are not flush at the top with the corner of the browser.

Now why is that?

Let's go and investigate.

Let's go and inspect the element.

Let's take a look, it's got a margin of 21.4 something.

And where is this margin coming from?

Well, it's coming from, again, the user agent style sheet.

And the reason it's coming here is because of some browser specific prefixes that we

actually haven't talked about that are defining the margin before and

margin after.

We're not going to delve too much into this, but the bottom line it means that

before the element and after the element they'll define some margin.

Now the reason our margin on the body that was specified margin:

0 is not working here is very simple.

Body says anything that is below me and

inside of me is going to have the margin inherited.

But that's only if you don't override it specifically for that element.

As you see here, the user agent stylesheet has a selector h1

that is specifically overriding the margin, which therefore, wins.

This is a bit of a side point but how do we override this?

Well, now that we know about the star selector, it's easy.

You could just say and move the padding in the margin to the star selector.

And now, what this is basically saying is that I am selecting

every element specifically, and then I'm overriding its margin.

So, if I refresh the page now, you could see now it's flushed together to the top.

We won't necessarily be doing this in our pages left and

right, but at least now you know that if you wanted to reset in,

it's called CSS resetting, if you wanted to reset the CSS browser defaults

you would do this most of the time using the star selector.

PLEASE CHECK THE SCREENSHOTS

[MUSIC]

Okay, one more small topic before we finish this lecture.

We spoke before about box sizing, so specifying the size of the box.

And we talked about box sizing being the border,

box meaning that if I specified some width or some height, it will include,

always include from the edge of the border to the other edge of the border.

And we'll also show that as we change the width of our element,

since the content inside the box shifted, the height of the box shifted as well.

So let's go ahead and add some more content into our box, and

then we'll us the Emmet plugin, E-M-M-E-T, go ahead and look it up.

We'll save it.

We'll refresh it, and now we see it's much bigger than before.

Let's lower this a little bit, so you can see.

So because I added extra content,

you can see that the height of our element is now 162.

But what happens if I constrain our element's height as well?

Something's going to have to give, right, because there's no space left.

So let's adjust a couple of things.

First of all, let's go ahead and make this back to ten, so

the padding will be just ten all around.

And the border is kind of looking unseemly, so we'll make it like five,

five pixels.

And go ahead in the margin, those margins doesn't matter now.

So let's refresh, and now it's a little bit better.

Okay, so let's go ahead and now give it a height that will clearly not be able to

fit all the content of our box, so let's give it a height of, let's say, 50 pixels.

And we'll save and refresh.

As you can see, what's happening now is is that our contents spilled over,

outside of our box.

And if we actually place some element right below it,

let's go ahead and place another div, and and we'll say again, Lorem ipsum here, and

we'll refresh, you could see it's spilling over right over the other content.

So it's basically spilling over out of the box.

I'll just go ahead and remove that for a minute.

Okay, so how do we deal with this?

Well, there's a couple of ways of dealing with it.

And there's just a property to help us do that.

And the property's called overflow.

So we are overflowing, at the moment,

we are overflowing our content outside of the box.

So what should we do with that overflow?

So by default, the overflow is visible.

And this is exactly what's going on.

So if I save this and refresh this again, it's the same thing.

However, there's a couple more options we could do.

Number one is we could just clip it.

So in order to clip it, we could go ahead and type hidden, so if it If it's hidden,

then if we refresh, it'll just go ahead and clip it where the box ends.

However, there's a couple more options we could do.

We could say auto.

Auto means go ahead and put scroll bars wherever they're needed, in order for

us to look at the entire content.

And so let's go ahead and refresh that, and you can see I have a scroll bar here,

and I can now scroll through my entire content right inside that box.

If I wanted to, I can keep the scroll gutters always visible, no matter whether

or not I need them or not, by just saying scroll, as the property value.

If I refresh, now I have scroll both ways even though there is nothing to

scroll here, but I could still scroll up and down.

Lets go ahead and change it back to auto and refresh.

Okay, so now we have the scroll bar just where we need it.

While this is a perfectly acceptable solution to get your content inside

the available space that you have,

I will warn you that users absolutely hate double scrolling.

Having the scroll bar on the side of the browser plus having to scroll inside some

element is something that is not preferred.

However, there are cases where it's perfectly normal and expected to do.

For example, if you're looking through, let's say, terms of service, and

you just really want to pack in your entire UI into one visible chunk,

yet you want them to be able to scroll through some terms of service or

something like that, or some of the content.

A solution for your overflow being auto is certainly an acceptable one.

Okay, so in summary, we spoke about the box model.

And really, the box model is this very essential topic

to understand about CSS and as we go onto CSS layouts.

We talked about the fact that you should prefer to use box-sizing border-box.

And that will keep things consistent for you as you change the border and

the padding properties.

We spoke about the universal selector, the star selector,

and how it can affect every single element in the entire HTML page.

And we also spoke about cumulative and collapsing margins.

Remember, horizontal margins are cumulative, and

for the most part, in most circumstances, the vertical ones collapse.

We also spoke about the cases where the content overflows the size of

the actual box and what do we do about it using the overflow property.

Next we're going to talk about the background property.