











[SOUND] Media queries allow you to group styles together and target

them to devices based on some criteria. For example, you can target

a device by its width, its height, or orientation like landscape or portrait. Of course one of the most obvious

differences between viewing a website on a desktop browser and

your cell phone is the screen size. Remember that using CSS you have the power

to produce very different web page layouts from the same HTML. Remember CSSgarden dot com? So, it only makes sense then, that the

most common way to adjust the styling and layout of your page is to

provide different styles for different screen sizes

of the user's devices. This is why it's crucial to learn

at least the basics of how to utilize media queries. Without them responsive design, which

we'll start talking about fairly soon, wouldn't be possible. Let's go ahead and

explore the basic syntax of media queries. A media query starts with

a keyword @media, and then it is followed by a media feature,

and followed by curly braces. Within those curly braces

you have your styles, it's basically like a style

sheet within a style sheet. Each media feature resolves

to either true or false. You can have more than one media features

combine together using logical operators. If the media features resolve to true,

the style within the curly braces apply. When writing media queries,

make sure you close the curly braces for the media query as a whole and

then separately each styling rule. There are quite a bit of these

media features that are available. So you can have max-width, you can have

min-width, you can have height which is not listed here, you can even target

orientation of your device, portrait or landscape, you can target only screens

as opposed to targeting only print. Again, if any of these things evaluate

to true, the styles contained within the curly braces of the media

query will be in effect. As I mentioned before, even though there

are quite a bit of these media features that are available the most common ones

are the max-width and the min-width. And again, the fact that they're the most common

really is connected to the fact that the most common way of targeting different

devices is by the width of the device. In addition as I said before, the media features can be

combined using logical operators. One of the most common logical

operators is the and operator. As an example, here's a code

snippet that targets a width range. Here you see we're targeting any

device that falls within the range of its width being anywhere from

768 pixels until 991 pixels. If the devices has a width

that is smaller than that or larger than that the styles applied

within this media query will not apply. Another way to combine media features is

to place comma in between them which will basically translate into being

equivalent to an OR operator. In this code snippet we're

showing that we're targeting any device whose width is no

larger than 767 pixels or any device that is at

least 992 pixels in width. Now, practically speaking,

when you approach responsive design and responsive layouts, the most common

logical operator is the and operator. So if you know that operator,

you're in pretty good shape. Now before we jump into a code example I'd

like to show you a very common approach how you structure media queries

within your style sheet. What usually happens is you have

a few of these media queries, but you almost always start

with some base styles. Base styles will apply across

the board no matter what screen size you actually are viewing the website on. Then you go ahead and

start targeting a particular screen size by either changing some of

the properties of the base styles or adding something else to them or

maybe taking something away. Now an important point to point out, and especially when you're using

widths of screen sizes of the devices is that you have to be very careful

not to overlap range boundaries. So in this example you will notice that

the first query has the minimum width of 1200 pixels and the second query

is coming up it's a range again. And it comes up,

up to max width of 1199 pixels, if I were to say 1200 pixels it

will mean there always inclusive. It will mean that both sets

of styles will apply and most probably things will be pretty

messy and very hard to maintain. So it's important when you structure these

media queries that you have very clean, separate boundaries. In part two of this lecture, we're

going to jump into the code editor and see these concepts in action.

[MUSIC] Okay, so I'm in sublime text and I'm looking at the file code

media queries before.html. And it's located in

the examples lecture 23 folder. Let's take a quick look at

the structure of our html. And it's a very simple one. Again, just a H1 just to

tell us where we're holding. You know we have in our HTML

really is two paragraphs. First one is ID'd with p1, and

the second one is ID'd with p2. Let's take a look at the styles that

we have for these paragraphs so far. As you can see, we're following our own

advice, and we have some base styling that we're going to apply to these paragraphs

before we even get to the media queries. We're not going to pay

too much attention to H1. That's just really for our web page

to look a little bit more decent. We'll pay attention to

the paragraph tags however. So here we're targeting our paragraph and

basically just giving it a border of 1 pixel solid black and

giving it a bottom margin of 15 pixels. And then we're distinguishing each

paragraph by using the ID selector. By giving it a particular color in

the first paragraph is going to be fairly large. It's going to be 300 pixels by 300 pixels. And the second one is going to

be a different color and it's going to be 50 pixels by 50 pixels. Let's take a look as to what

this looks like in the browser. So here are our two paragraphs. One is fairly large, another one's small. Let's take a look at what happens if

we make the browser width smaller and smaller and smaller. And as you can see,

really nothing happens at all, right? We could go on and nothing is going on. So let's take a look at what we

could do with our media queries. So here I have a couple of sections

that I specifically designated for our media queries. The first one is going to try

to target large devices only. So what are large devices? Well, large devices in my vocabulary

will be anything that is at least 1,200 pixels wide. So let's go ahead and

write media query for that. And the min width, right, min width is

going to be 1200, and don't forget pixels, and we'll open the curly brace and close

the curly brace and we're ready to target our paragraphs just for devices that

are at least as large as 1200 pixels wide. So we'll go ahead and do our first

paragraph and what we'll do is we'll change it's width to let's say,

a windows width to let's 80%. So what that will mean is paragraph

number one at width 1200 pixels or wider will take 80% of our screen. As for paragraph number two, we'll

make it that it's width is 150 pixels. Which is a little bit larger than it is. Actually, three times as large, right? Because it was 50 pixels before. And, we'll actually change it's

height to 150 pixels as well. So, it will be a perfect square. Okay, so let's save our file and

switch to our browser, and let's reload and let's take a look. Since our browser now is

definitely wider than 1200 pixels, you see that both of the paragraphs

actually were affected. Now, how do I know that my browser

is wider than 1200 pixels? Well, if I press Cmd+Option+I, or on PCs Ctrl+Option,

I will bring up my Chrome developer tools. And now if I start dragging the side of

the browser you'll see in the top right corner that it's showing

you the actual size. And you see something else interesting, you see at 1214 pixels as we go smaller

and smaller when it goes to pass 1200 pixels, you see that everything jumps

to it's original base sizes, right? And if I make it a little wider again,

it would jump again to this styles that the media query width meet

with 1200 pixels specified. Let's go back and make this larger again. Let switch back to our code and

write one more media query but this time for medium size devices. So what are medium sized devices? Well medium sized devices is

probably something around 992 pixels and 1199 pixels. Now where am I pulling out these numbers? Well I'll tell you the secret,

I'm cheating a little bit. I'm pulling them out out of

the standard ranges, width ranges, that the Twitter Bootstrap framework defines

for its different media device sizes. We'll speak about

Twitter Bootstrap fairly soon. But for now,

our media query should be minimum width of 992px and remember that and,

and max-width Of 1099 pixels. And remember we're trying make sure

we don't overlap with the previous media query otherwise we'll basically

going to end up applying both style again. So we're going to open up the curly brace

and we'll target again Paragraph 1, and we'll go ahead and make it smaller. We'll make its width still in percent,

but at least we'll make it small. We'll make it 50%. And paragraph number 2,

we'll make it's width to 100 pixels. So again, a little bit smaller than for

a larger screen size. And the height also,

we'll make it 100 pixels. So now you can see that our base

size makes our first one 300 by 300, second 50 by 50. If the screen width falls into range

of 992 pixels and 1199 pixels. We're going to increase our

size of the first one and we'll make it 50% of the screen. And paragraph number two will become

100 by 100 instead of 50 by 50. And if we stretch our browser

all the way to 1200 pixels and above, the first paragraph will

become 80% of the screen and the second paragraph will

become 150 pixels large. And obviously not of this makes any sense

in terms of a nice layout this is really just an example to try it up. So let's switch the browser and

see how that worked out. Let's make this a little bit smaller so

we could see all of them. But still see the screen sizes. And go ahead and refresh it. So we're now at above 1200. As we go a little bit slower to 1200 when we get to a little bit less that 1200

we switch to our second media query which makes this be 50% of

the screen as opposed to 80. We could go back. We could see the 80% of the screen. That's the 80% of the screen. And go a little bit down and if we below

1200 this is 50% of the screen and this is 100 by 100 pixels and

if we go even below 992 pixels. As we get closer and

we switch from 992 to 991, you can see that we're back to our

base style then because we have no media query that is defined for

that particular screen size. I'd like to show you one more thing that

the Chrome developer tools has here. You'll see this cell-phone looking icon. If we click that we'll

see a whole different view of our page that

we're able to play with. And it tells us here that we might

need to reload so we'll go ahead and reload that right now to

get rid of that warning. But you see here I can select any device I

want and even select between portrait and landscape mode and see what my website

looks like on that particular device. And I could also go ahead and

change the device size right here just by dragging the border of this thing. And one more thing is if you take a look, it actually shows us in our own

CSS that we loaded into this page, what are the different ranges in terms

of the media queries that exist. And you can see here we have

one that's from 992 to 1199. And the other from 1200 and on. And if we go ahead and start selecting some devices let's see

what does it look like on the Apple iPad. It looks just like that

in the portrait mode. But if I switch,

it'll be in landscape mode. It's telling us again to refresh. Go ahead and do that. And you can see that

we can actually go and see what happens at these break points. And the points between one media query and

another one, it's called break points. So we're now going towards

our second break point, and when we get there you see that our

first box basically made itself 80% and I can't remember what the size of that box

anymore is, but you see the difference. And if we jump back,

now this one is applying and we make it even smaller than that,

that breakpoint, 992 pixels. At 991 pixels,

the break points will change. And now we are back to our base styling. So Chrome development tools are incredibly

useful when you're trying to design and test and basically see and debug

a little bit your responsive designs. So in summary, we'll look at

a basic syntax of a media query. And remember it's @media with some

media feature in parenthesis. Or you could combine media features

with some logical operators, or here we have media

feature in parenthesis. Followed by some logical operator, and in responsive layouts it's usually the and

operator. And then followed another media feature,

also again in parenthesis. Remember not to overlap breakpoints. We talked about how if we structure our

media queries with different widths of different devices, that the breakpoints

of these widths should not over lap, even by one pixel. Usually the way you approach this is

you provide some base styling and then you change or add to them in

each of the media queries, and usually structuring your assess in

this way, makes it much cleaner and also much easier to

maintain down the line. Next we're going to speak

about responsive layouts.