VIDEO TRANSCRIPT

Mobile Menu Customization

Hello and welcome to the lesson where I'd like to give you a few tips on how to target and style the Divi mobile menu. I will first edit the default Divi mobile navigation. And next, I'll apply the same styling to the menu module. The mobile menu is probably one of the weakest points of the Divi theme with very few customization options, but there is a lot we can achieve with CSS and that is what I would like to show you in this video.

Changing Menu Icons

So the first thing we can fix here is this boring hamburger icon and if we select that element, we can see that it is a before pseudo-element and it uses the et_modules font family, so we would only need to change the content and here on the Elegant Themes font overview, I can see that the X, the closing icon is using the / 4, d. Remember the first three symbols is the backslash and then we disregard the semicolon, so 4d. Now, if I change that to 4d, that will change it to X but I don't want it to always be this icon, only when it closes. So when is the menu being opened? And when does it close? We can see here in the HTML preview, when I click on it, the class of the parent div container is changing to open it and now it's closed, okay? So we can use that as the parent to target our elements inside. So if I only want to target that before the pseudo-element when it's opened, I can define a new role and add.
a parent container with a class, open it, and then I have my mobile menu bar before so this before on that span and content: backslash 4d. Simple

as that. Let's maybe let's try to remove that Divi looking border. This is the border on the main UL element. As you can see the ET mobile menu uses border top. We can simply add our new role ET mobile menu, border-top: none. Okay, that looks nicer, I think.

Width Example

Now let's try making this full width so right now, the width is already set to 100% but that is only as wide as the parent container, if we want this to stretch across the entire screen, we can use the viewport width value. So width 100 viewport width but as you can see that puts it off center because the left is 0. Its position absolute in relation to its parent container, the left is 0. The parent container is actually that div with a class container, which has a set width to 80%. So if we know that menu width is actually 80% of the screen, now we can move it to the left. By negative 10 viewport width, which basically means 10% of the screen. So right now it is nice and full width. Now we can change the padding maybe, we can change the background color here to target each of the links, we can target the lis and the <a >element. So all the a's, have a bottom-border, this light grade border, we can remove that - none. We can also remove the background on the parent elements. If we have sub items, so, this element, this link inside here targets a li with a menu item that has children class, and then the first link inside. So that selector here and it makes the font-weight: bold, and changes the background color. So we can overwrite it with our CSS that removes that background. So background: none. And then font-weight I think it's 500, the default is 600. So it's some somewhere between regular and bold. If you want the regular font size on all links, we can target simply them like this with that et_mobile_menu li a and then font-weight 400, which is regular. So here as well, like that, that makes it simpler. We can always

click the inspector stylesheet that is being created every time we add a new role here, so this is the CSS we have so far.

Customizing CTA

Now, let's try customizing that call to action link. I added the CSS class to that element. So that li has a cta-link additional CSS class that I've added here in inside my appearance menus, I have CSS classes cta-link. Remember to show that classes, we have to click on the screen options here and select this check box. So we can target that CSS class and maybe try to style that link to look as a button - a call to action. So li with a class cta-link a, let's say background lightsalmon, color white, display: inline-block so that it doesn't take up the full space. And maybe we can align it. Add some margin so that the button itself, it is nice that it uses padding, but let's align it with these items. So the padding it uses now, we can check, that's on the <a> element. So the padding is in percentages I think, let's see. It's 5%, so we can add margin five percent margin, left, five percent and that sort of aligns it together with these links, you can also add a bit of margin to the top. Let's try maybe with pixel value, something like that. Okay, and we can style the text. Make it uppercase, change the font weight, and add some letter spacing, and maybe border-radius. Okay, that's a nice-looking button. You see, it wasn't at all, difficult to target that element and style it.

Editing Animation

Now let's try changing the default animation from the slide in to slide from the side. Okay, that's a bit more complicated, but let me explain how we would approach this. We need to see how that menu behaves now, the ul, that's the element that's being shown and hidden, changes the display block and then it's display: none when it's not visible. So,

VIDEO TRANSCRIPT

that's controlled by some JavaScript and to be able to slide it from the side, we have to make sure that it is visible always, just outside of the viewport. That's why we need to add the display: block important because only using important will overwrite the inline styling that's added with JavaScript. So using block important makes it always visible, even right now when it's supposed to be closed, it's visible. So now when it's always visible, we also want to make it fixed so that its position is relative to the viewport and not to the parent container, so position: fixed. And now, let's move it to the right side out of the viewport, so, right negative 100%. And we have to change the left to auto. Now let me show you, right now it uses the right zero and that's right negative 100%. So basically it's here on the side and now if we use the opened class, so opened, and then et_mobile_menu, right: 0, ok? It does show it hidden but not in a way we want. And let's see what else is being animated here. So you can see that when I click on that icon, we can see some properties being changed here in line on this element, so that's height and padding. So if we define the fixed padding here, padding 5% important, and height. Let's actually make it full screen because it will be fixed, so 100 vh and let's go with important, so that it doesn't animate the height. Okay, that's very good, right now the only thing is the transition because it goes from right, but we don't see it because there's no transition, so transition, all .3 second easy-in-out. Let's see. Lovely. But now we've set the height to 100 viewport height but it's not at the top. So let's also do top 0, okay?And that makes it shown above the closing icon. So, we also need a negative z-index, just to show it. That's great that we can increase the padding at the top here. So instead of five percent, let's do: 80 pixels, 5% 5%. So 80 pixels from the top and then 5% on each of the sides. Lovely. We can change the width so it does is not full width. Let's try with maybe 50 viewport width. Simple as that.

Okay, we could make it slide in a bit longer. I hope you see that simply being able to target each of these elements and apply CSS we can change the appearance in any way we like. Now, if I actually wanted to use the CSS I wrote, I would wrap all of these declarations inside a media query. So @media, max-width 980 pixels. So that's the default breakpoint and I need a closing bracket here. That's the default breakpoint for the tablet and phone. And I would copy that CSS and place inside my theme options.

Customizing The Menu Module

Now let's do a similar thing with the menu module. Okay, so I'm in the theme builder and I will create a new global header. And add just a regular section and a menu module. So, for the standard menu module, let's try with this one. We have a bit more customization options within the Divi Builder. In the design settings, we can change the border color, so drop-down menu, line color, we can make it transparent here, we can change the link color, active link color, and background color of the entire element. So there are certain things we can style here. But that is basically it. We don't have any position control over that menu. So let me add a CSS class here - dsa-menu. And now, let's save that. And save the changes here and preview the page here. Okay, so it will work very similarly to what we've already had for the default navigation. We will just need to change our selectors. So let's try to add the same CSS that we used for the default navigation. Okay, so let's find our et_mobile_menu. So the et_mobile_menu is the same selector we are using, okay? Here you see our CSS that we've added. And some of the values are being overwritten. So, the right doesn't work. Let's see what overrides it here on the side. We can see right - Oh, that's our own opened class, okay, but the top is also being overwritten. And

coming from this part. You see et_pb_fullwidth_menu, et_mobile_menu and the regular menu, et_mobile_menu has a top CSS of 100%. So we would need to match the specificity of that selector. So, here in our Chrome stylesheet, we can add the et_ pb_menu., and now that top value does work. We would need to change the background of the main menu container for mobile if we wanted to use that slide-in style. But as you can see, everything else works the same way, the same selectors applied to that menu, as well. If we want to be more specific and only target that particular menu module, we would need to use our own CSS class. So dsa_menu and add it here that would be inside that element. And here as a parent of et_mobile_menu, also here as a parent, as a parent to that link and also here, like that. So here, as well as a parent container. We can see that in a page structure, so our menu module et_pb_module with the dsa_menu class is a parent container of the et_mobile_nav_menu that has the mobile_nav with the opened class. So our main class is the main parent for all these elements, our older lis and the mobile menu itself. So here this has to be the parent selector if we only want to target that one particular menu module. And let's see if we can change the background color of that module. Okay, and now let's add my CSS to my page instead of the page inspector. And as you can see, it will look differently in the Visual Builder because we would need to make our selectors more specific, their IDs, that targets these elements here in the preview, but we do not need to worry about it. As long as the front end of the site looks correctly, which it does, we don't need to worry about the theme builder preview.

So hopefully that is a helpful overview of how you can target your mobile menu and style it anyway you like with CSS.