VIDEO TRANSCRIPT

Working with jQuery plugins

Hello, and welcome to the final lesson inside the jQuery module where I would like to give you a walkthrough on how to make use of jQuery plugins when working with Divi. There is so many plugins created for jQuery, which we can find and use within our websites and to make it work, we would usually follow the same logic - install the plugin JavaScript file, add some CSS, and then make sure our HTML structure matches the plugin requirements. Finally use the plugins documentation to trigger its functionality. In this video I will walk you through how to create pop-ups and sliders using popular jQuery plugins.

Downloading and Installing jQuery Plugins

On the jQuery plugin registry website you'll find a very long list of different plugins you might want to use and one of these plugins here is called 'Magnific Popup' and that script is actually used within the Divi theme. Divi is using it to display images in lightbox and to trigger galleries inside pop-ups. So if we would want to use that plugin we do not need to install any additional scripts. It is already being loaded with Divi so it makes perfect sense to use it, for example, to add custom pop-up functionality. So if we go to that plugin page, we have a link to the plugin homepage which is also where we can find the documentation. So right here, we'll see that in order to use that script that plugin we would need to make sure that its CSS is loaded. And also the main jQuery file has to be loaded and the plugin core JS file as well. So all these files are already loaded when you install Divi, so that part is done.

VIDEO TRANSCRIPT

And now we have some instructions on how we can initialize that popup. So anytime you want to install a certain plugin on your website, you have to read the documentation, and usually it will give you step-by-step directions on what to do to make it work.

Using the jQuery Plugin

And I know I don't want to use that pop for images, I want to display a section like additional content, like contact information or signup form or something like that - so that would be an inline type content. So here inside the that part of the documentation it says right here: I need a HTML element that will be the content of my popup and the class mfp-hide is required to hide the pop-up from the page. So that's the div they say I would need to use. So instead of using a code module and adding a div there, my sections in Divi are divs, right? So all I need is to make sure that I use the correct ID and I use that mfp-hide CSS class to hide that content. Okay, so now here on my sample Divi page, I can create a section, I would just use some background color so that we know that this is the section, and let's add a simple text module inside: "I'm in a popup!". Okay, so, let's make this big. That's the section I want to show when I click on a button, okay? So back here on the documentation page, they are using a test-popup ID as an example, and we can change that and define our own, but I will just use it the same just to follow the instructions here, okay? So my section needs an ID - test-popup and also that mfp-hide CSS class, but what this is going to do is it will hide my section. I can show you this right now, okay? So as soon as I'm adding that class, the CSS in Divi is already hiding my popup, and that would be a moment where you would like to use that et-fb CSS class on the body, I mentioned in a previous lesson. So if I add that mfp-hide, I can still use that section ID to make sure that it is visible inside

VIDEO TRANSCRIPT

the Visual Builder. Okay? I want this to be hidden on the front-end, but I want to be able to still see it to edit the content of the popup while I'm in the Visual Builder. So, below that first section, let's add a new standard section, and I will add a code module here, just to add my CSS, okay? So inside the style tag, and as you know, you can add CSS in many different places, but I'll be using that code module to add CSS, and JavaScript as well so just so everything is one place. And now my Visual Builder when it's enabled, the body has a class of et-fb so body with a class et-fb and then ID test-popup. So the ID of my section I wanted this to be displayed: block, okay? And that should make that, let's try with important. Okay. There we have it. Now, it does show and it will only, that CSS will only affect that section when the Visual Builder is enabled, so that's exactly what we want, okay? So that's why I added it here so that we can see our popup. And now inside our standard section below, will add a button that will be the trigger because back here on the documentation page we see that we have to add a button that will open the popup. This, I missed that step, but it's to style my test-popup section, so I don't need to do that. I am doing that using Divi options, right? So that's why I don't include that CSS, it's just to style that popup content, I can style it using Divi settings. And now we need a trigger. So it says add a button that will open the popup. The source must match the CSS ID of an element test-popup in our case, you see? The documentation is very clear, so it's very easy to follow. So that would be a trigger I could use and sure I can just copy that code and paste it in the code module, but it's a simple link so I know I can just as well use a button module. And my button URL needs to be set to #test-popup and I need that open-popup-link CSS class. So let's add that to my button, so - link: #test-popup and my CSS class for that element, let's use the same thing: open-popup-link. Okay? So what's next?

VIDEO TRANSCRIPT

Initializing the Script

Initialize script. So that's the jQuery code, that will initialize the script for us. And by looking at it, I know that I need to wrap it inside that document ready function inside my customs scripts. So, let's copy that. And now depending on if this popup I'm using is global. Do I want to display it on every page or is it just on one page? That's the decision I need to make about where I want to place my custom script. Because if that popup is a part of the footer, for example, and I want to display it everywhere. I could insert that script within my child theme or within a functionality plugin that will make sure that it's loaded across each page of the website. But if I'm using that popup only on the contact page, for example, where I'm displaying some additional information or contact form in the popup. So I don't need to load that script on different pages, right? And in this case I could use a code module. So I will do just that as an example. So back here in my code module where I added some of these initial styles I need to wrap my snippet of code within a script tag. If that would go to my scripts JS file, it doesn't need a script tag, but if it's going directly into my HTML structure, using the code module, I need to tell the browser that this is JavaScript, right? This is a script code. So, this is what I copied but what's just the inner part of our jQuery, right? We need the main document ready function first here. So let me paste that here. Now, we can remove that comment so that we can see it better, okay? So I added my document ready and now I need to close this, right? Okay, so basically just the wrapper of my main jQuery function so I'm telling the browser that I do want to use jQuery now, okay? And that's the jQuery that's inside. That's what I'm using to trigger the plugin. Hopefully that's clear. Let's save that, let's save that page, and let's preview that on the front-end.

VIDEO TRANSCRIPT

Testing and Customizing the Plugin

So as you can see that CSS, that was showing the section in the Builder doesn't work here, which is what we want. And now, if I click it, here's my popup, right? Just simple as that. And if I go through that documentation, I can see that there are different options I can use, like, making it close on content click or background click or where's the button? Is it inside or what classes it uses - all these different settings we could use to display that popup in a bit different way, maybe if we want it, okay? So back here, that's where, let me show you, we have only specified type inline and midClick: true. I could add the removal delay or different basically all these different settings I could choose to use or not. And then place them inside that trigger function, basically that's all it takes to make use of that plugin. Hopefully that's helpful. And now let's see how we can install something, which is not loaded in Divi yet. So another plugin I would like to show you is called slick, responsive touch carousel. I do think that you should avoid using sliders, users do not slide through sliders, especially if you would like to include some important content on slide number three, no one will see it, okay? But there are certain use cases when using a carousel like something like this. Can work really well and might be very useful when you have a bunch of testimonials you want to show. Showing only a few at a time allows people to scroll through, if they want to read more, that is a great idea. And a solution like this which is fully responsive, makes it a great mobile experience, where you can drag your testimonials to the site to read more, you don't need to scroll through each one if you're not interested in reading it.

Installing a Plugin in Divi

VIDEO TRANSCRIPT

Okay, so how do we install a plugin like this in Divi? We always have to look at the plugin demo page and plugin documentation. So, here we have a usage section. And first, we need to set up our HTML markup. We have a div and inside we have more divs with content. So each div is a content that holds the content of each part of the carousel. So I have a similar structure here where I've included several testimonial modules inside a row. So my wrapper div is a column, right? So I have a column that has all these divs inside, so that's done right? Now it says: "move the slick folder into your project". We could go ahead and go to their GitHub page, download the zip file and install all these files within our child theme, for example, but we can also use CDN - content delivery network. So basically load the scripts and styles the plugin needs from an outside server. So we could either include the slick CSS slick-theme CSS, and then jQuery and slick, min.js - all these files enqueue them within our style sheet or we can copy the CDN class links. So this would link to the slick CSS file, slick-theme CSS. And here, we have a slick.min.js so I can basically just copy that. Okay? And if I only want to use that plugin on one page where I have my testimonial section, I will create a new section here with a code module inside and I can basically include all these links, okay? So we have a link to a stylesheet: slick.CSS file and another, with a slick-theme.CSS. I could basically copy that CSS from, if I go to that page, this is all the CSS that it includes. So I could just copy that CSS and put it right there in my stylesheet. I do not need to load an additional file. So whatever method you use to include that CSS it's fine as long as it's being loaded on the page, okay? So we have two CSS files that have to be loaded and also the plugin file itself, the slick.min.js so I will also use that CDN link to load that file on my page. But we could download that and enqueue it just the same way as we are in queueing custom JavaScript within the functionality plugin or inside

a child theme as I showed you earlier, okay? So any method is fine as long as it is being loaded on your page and that plugin script has to be loaded after the main jQuery file. That's also very important. So, just putting it like this inside the code module, will make sure that it is loaded after jQuery, jQuery is loaded in the head, so anything inside your body can already use jQuery. Okay? So what next?

Initializing Plugin in Divi

Next, we have to initialize your slider in your script file or an inline script tag. Okay? Again, super simple. We know what that means, we have to copy that code and it will initialize our plugin file, okay? So we are still in the same code module and inside the script tag that's the inline method. We need to initialize our script. We could put that function, that code inside the script .js file, that's a file that we may be using for all sorts of different jQuery customizations on our site, okay? And here, at the beginning, we have to use jQuery because it will not know yet that we want to use jQuery. So that needs to be jQuery. And by adding that dollar sign here, inside the function, we will basically map all the dollar signs and tell the browser that we do want to use jQuery, okay? So basically that's what we need. And now, as you can see, it is just an example, it has a setting name and setting value. Let's leave that empty for now. And now the container - that the function is triggered on uses a class of your class, let's copy that. We can obviously define something unique for our use case but just, let's copy that. I will save that code module setting.

Customizing the Plugin in Divi

And now, the class has to be a parent container for our carousel elements. I want each testimonial to be a separate carousel element, so

heir parent, is the column div, okay? So I have to go to the column settings and add my custom class here or basically define a different target. I could target the row class and then column within that class but just to make it easier. We are using a CSS class on the column and now all the divs, all the modules inside that column are each of the should work as separate carousel elements. Okay, so now let's save that. It doesn't work in the Visual Builder, but if we exit the Builder, the basic part works. Okay, it's again, as simple as that. And now we can look through different settings and options that the plugin has or just look through different demo examples that may look the way we want our version to look. So I think this is the responsive style, this is something we could use. So let's copy these settings, again it's basically the different parameters for that function and we have a list of all the options we can use inside our documentation. So if you are wondering, if it can behave this way or that you would need to read through that documentation to see what options are available. But as a start, we can copy this, okay? And now, back to our code module. We can put the same parameters here. I will remove that comment. Okay, so we have dots: true, it doesn't go infinite, speed slides to show for slides to scroll for. Let's see how that looks. OK, let's save that. Let's refresh. And it shows four slides. We have these dots here, we can drag it, but obviously for maybe a bit too much or we would need to disable the image. Let's try with three and let's add some spacing in between. So here are slides to show, three slides to scroll three, okay? And then a responsive breakpoint. Let's make it the same as Divi for tablets, so 980, and then we only want to show two and then for 600 - let's change it to 1, okay? And we don't need that last breakpoint. If we are showing text, it may be easy to read on this screen size. Okay, so let's save that. And now for my testimonial, I want to add some margin. So in the spacing margin, let's

VIDEO TRANSCRIPT

try with 20 pixels on each side and let's extend that to all testimonials throughout that page. That's fine. Okay. And now if I save that you see simple as that. So, basically, now I could check all the different settings if I inspect that, let's see how that looks on. So on smaller screen sizes we have only two, we would need to probably change that breakpoint, but I just want to show you how that would look, okay? So below 600, we have a single testimonial that people can, you know, scroll through on their phones and it just makes it easier to, it just makes your website shorter, basically, you don't need to scroll all the way down to if you have a lot of these, right? So this is a nice use case. And let's say I want to customize it like this: dots here are super tiny so I can check that. It's slick dots li button that uses font size 6. What if it would be 16? Okay, now I can actually see them, so I would just override that CSS and it's basically ready to use, right?So hopefully that gives you a good overview. As I mentioned, always check what is the recommended HTML structure you need then include all the CSS and JavaScript that the plugin needs to work. And then basically define your function to trigger the plugin functionality. It will work in a similar way for all the different jQuery plugins you can think of.

So hopefully that gives you a better understanding of how to incorporate some of that jQuery you'll find online within the Divi.