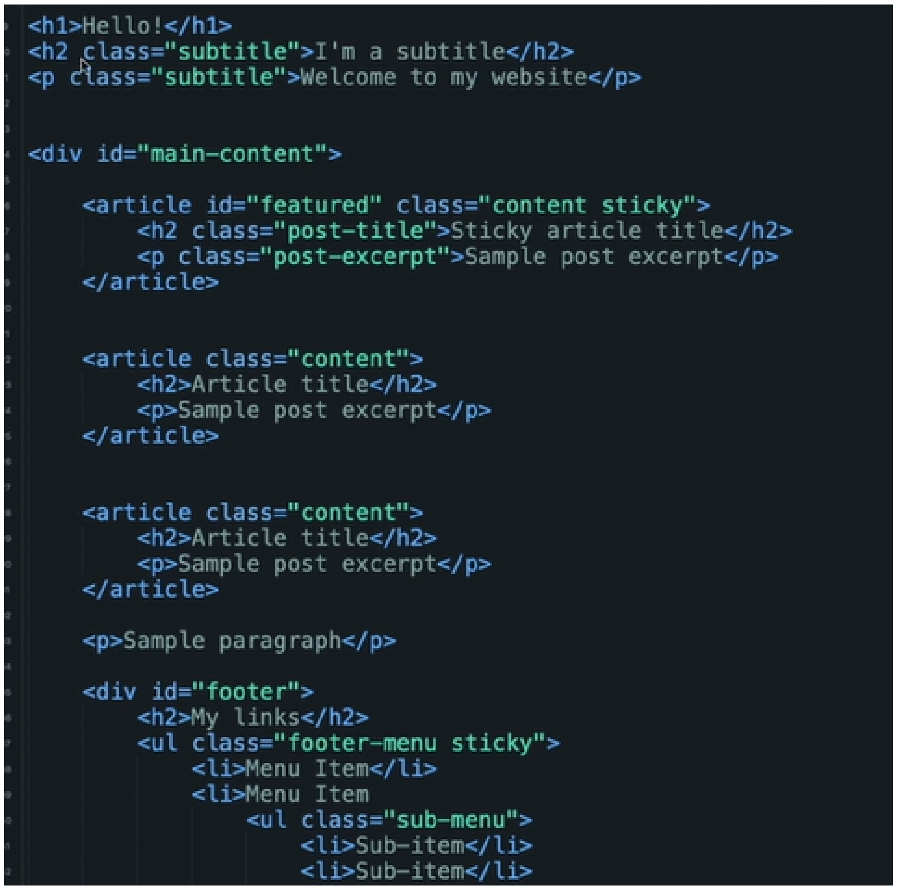
CSS Selectors

Introduction

Welcome to the lesson about the CSS selectors. As we learned in the CSS syntax lesson, a selector is the part of the CSS ruleset that specifies our target. That includes elements, CSS classes, IDs. And in this lesson, I would like to explain all the different, simplest and most popular ways to construct CSS selectors and target website elements and also how it applies to the Divi theme structure.

Sample HTML structure

But before we focus on Divi, let's look at the simple HTML structure first.



```
<h1>Hello!</h1>
<h2 class="subtitle">I'm a subtitle</h2>
<p class="subtitle">Welcome to my website</p>


<div id="main-content">

    <article id="featured" class="content sticky">
        <h2 class="post-title">Sticky article title</h2>
        <p class="post-excerpt">Sample post excerpt</p>
    </article>


    <article class="content">
        <h2>Article title</h2>
        <p>Sample post excerpt</p>
    </article>


    <article class="content">
        <h2>Article title</h2>
        <p>Sample post excerpt</p>
    </article>

    <p>Sample paragraph</p>

    <div id="footer">
        <h2>My links</h2>
        <ul class="footer-menu sticky">
            <li>Menu Item</li>
            <li>Menu Item
                <ul class="sub-menu">
                    <li>Sub-item</li>
                    <li>Sub-item</li>
```

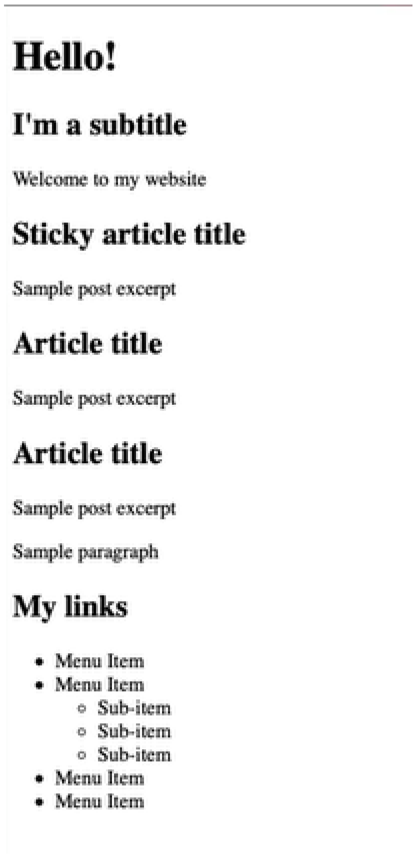
```
<html>
<head>
<h1>Hello!</h1>
</head>

<body>
<h2 class="subtitle">I'm a subtitle</h2>
<p class="subtitle">Welcome to my website</p>

<div id="main-content">
<article id="featured" class="content sticky">
<h2 class="post-title">Sticky article title</h2>
```

```
<p class=”post-excerpt>Sample post excerpt</p>
</article>

<article class=”content”>
<h2>Article title</h2>
<p>Sample post excerpt</p>
</article>

<article class=”content”>
<h2>Article title</h2>
<p>Sample post excerpt</p>
</article>

<p> Sample paragraph</p>

<div id=”footer”>
<h2>My links</h2>
<ul class=”footer-menu sticky”>
<li>Menu item</li>
<li>Menu item
<ul class=”sub-menu”>
<li>Sub-item</li>
<li>Sub-item</li>
<li>Sub-item</li>
</ul>
<li>Menu item</li>
<li>Menu item</li>
</ul>
</div>
</div>
</body>
</html>
```

We have a heading level one: h1, another heading with a class "subtitle", a paragraph with a class "subtitle", a container div with the ID "main-content", and an article that has a heading and a paragraph inside. There are three articles here and the paragraph. Next, a div with the ID "footer" with another h2 heading, an unordered list, which has two different CSS classes and some list items, and a second level, unordered list with a class "sub-menu" with additional list items - and that's it.

This is a simple structure. And now I would like to show you how we can select all of these different elements with CSS. I will create a style tag: `<style> </style>` in the head under the title. And now I can write my CSS and we will see the styling changes when we add some CSS.

This is how the browser sees all these elements when there's no CSS applied.



```
Hello!

I'm a subtitle

Welcome to my website

Sticky article title

Sample post excerpt

Article title

Sample post excerpt

Article title

Sample post excerpt

Sample paragraph

My links

  • Menu Item
  • Menu Item
      ◦ Sub-item
      ◦ Sub-item
      ◦ Sub-item
  • Menu Item
  • Menu Item
```

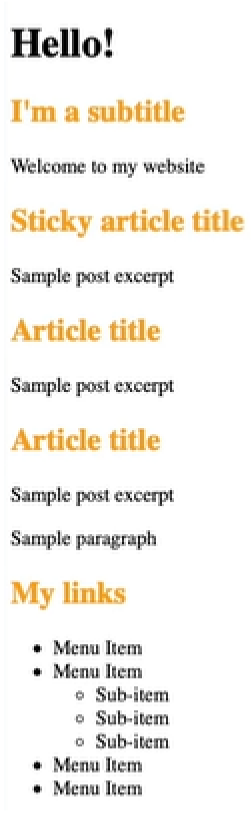
That's the default look for all that HTML structure.

Selecting HTML tags

So we can select the elements themselves. Let's see what happens if I select h2 and then specify the property and value. For example, I'll add the color orange that will target all h2's throughout the page.

```
<style> h2 {color: orange;} </style>
```

So I have an h2 at the top and then h2s inside the articles, and in the footer div as well - and now all these h2s are orange.



Hello!

I'm a subtitle

Welcome to my website

Sticky article title

Sample post excerpt

Article title

Sample post excerpt

Article title

Sample post excerpt

Sample paragraph

My links

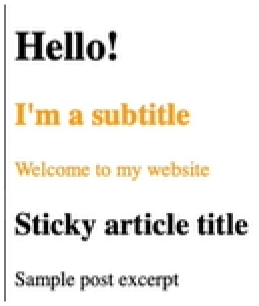
- Menu Item
- Menu Item
  - Sub-item
  - Sub-item
  - Sub-item
- Menu Item
- Menu Item

So this is probably not a very common way to target things, because usually you want to be more specific when targeting elements. That's when we can use a CSS class.

Selecting CSS classes

That first h2 has a class "subtitle". To target a CSS class, we are using a dot. The ruleset `h2.subtitle {color: orange;}` means we're targeting the h2 with a class `.subtitle`. And now, my CSS ruleset only applies to h2s which have that CSS class. And I have only one, so only this one will change the color to orange.

Now, we do not need to specify an element itself with a class. We can target just the class. Like this: `.subtitle {color: orange;}` And this way, we are targeting all elements with that particular class. I have an h2 with that class. And also the paragraph below also uses that class. That's why we are targeting both of these.


Hello!

I'm a subtitle

Welcome to my website

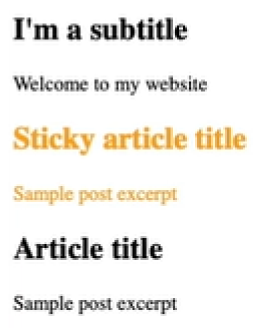
Sticky article title

Sample post excerpt

So these are classes.

Selecting CSS ID

Now, we can also target IDs. For example, the first article has the ID "featured". To target the ID, instead of a dot, we are using a hashtag symbol. `#featured {color: orange;}` This selector targets the element with the ID "featured", and because the color CSS property is inherited - everything inside that element will use that same value. That's why the h2 and p are also orange.


I'm a subtitle

Welcome to my website

Sticky article title

Sample post excerpt

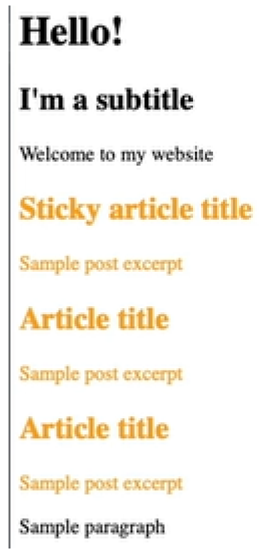
Article title

Sample post excerpt

So everything inside that ID will use that CSS property.

Combining CSS attributes

We have IDs, classes, the elements themselves, and we can combine all this together. So for example, if I would like to target the "content" class: `.content {color: orange;}` So that targets all these articles which have the class "content". The first article has two classes: "content" and "sticky". And the next two have only the "content" class.


Hello!

I'm a subtitle

Welcome to my website

Sticky article title

Sample post excerpt

Article title

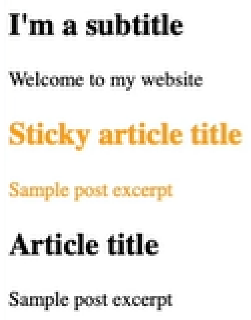
Sample post excerpt

Article title

Sample post excerpt

Sample paragraph

So if I want to target the first one, I could use the following selector: `.content.sticky`. Notice that there's no space here. So that selector applies to one element with these two classes.



I'm a subtitle

Welcome to my website

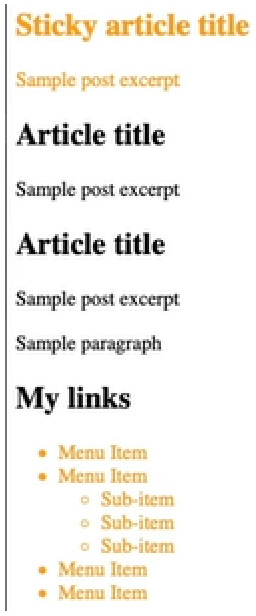
Sticky article title

Sample post excerpt

Article title

Sample post excerpt

If I used only a single class .sticky, we can see that it doesn't just select that article, but further down, our list in the footer also uses that “sticky” class.



Sticky article title

Sample post excerpt

Article title

Sample post excerpt

Article title

Sample post excerpt

Sample paragraph

My links

- Menu Item
- Menu Item
  - Sub-item
  - Sub-item
  - Sub-item
- Menu Item
- Menu Item

That's why we need to be careful when choosing our selector to make sure that it applies to the element that we do want to target.

So to reiterate, if I wanted to target that first article, I could use the selector .content.sticky, I could even add the ID, which is not needed in this case, because I'm already selecting this with these two classes. But if I suspected that maybe there were other elements with these two classes, which I did not want to target, then adding the "featured" ID to the selector - so #featured.content.sticky - would still select the same thing. But now I could be sure that the ruleset would not apply to any other elements because the ID is a unique attribute, right? It should only appear once on a page.

This selector may be a bit too complex for this simple example, but it's just so you know that you can combine multiple attributes together without any spaces here, and the selector will apply to one element. Because when we add spaces inside our selectors, the result will be completely different.

Space symbol in a selector

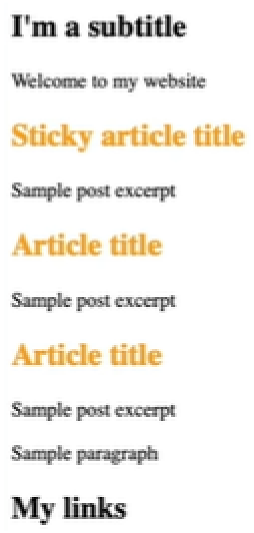
Let me show you an example. Say I want to select the h2s within a standard article:

```
<article class=”content”>

<h2>Article title</h2>
<p>Sample post excerpt</p>
</article>
```

The h2s here don't have any attributes. They don't have any CSS class or ID. And if I define just h2 as the selector, it will select all h2 elements on the page, which I don't want.

That's why I will use the parent container CSS class to target elements inside that parent. And we are using the space symbol to indicate that parent-child relationship. So the parent is the article with a class “content”: `.content h2` - this selector means that we are selecting h2s, but only if the parent container has the class “content”.

That doesn't need to be the closest parent: any parent container with that class will do, and all the h2s inside will use our orange color.


I'm a subtitle

Welcome to my website

Sticky article title

Sample post excerpt

Article title

Sample post excerpt

Article title

Sample post excerpt

Sample paragraph

My links

And we can specify multiple parents, and it has to work along with the page structure. So if I define ID "main-content" as the parent container, and then it has to have a class content, and inside that an h2 element, so the selector looks like this: `#main-content .content h2`.

And this selector will work for all articles with the class "content" within the div element with the ID "main-content".

But if I move one article outside of my div with the ID "main-content", the h2 in that article will no longer be selected because that selector only selects all h2s within all containers with a class "content", which are inside the "main-content" div container.

So we can specify multiple parents, so the full nested structure of our element, which is pretty useful to know when working with Divi elements.

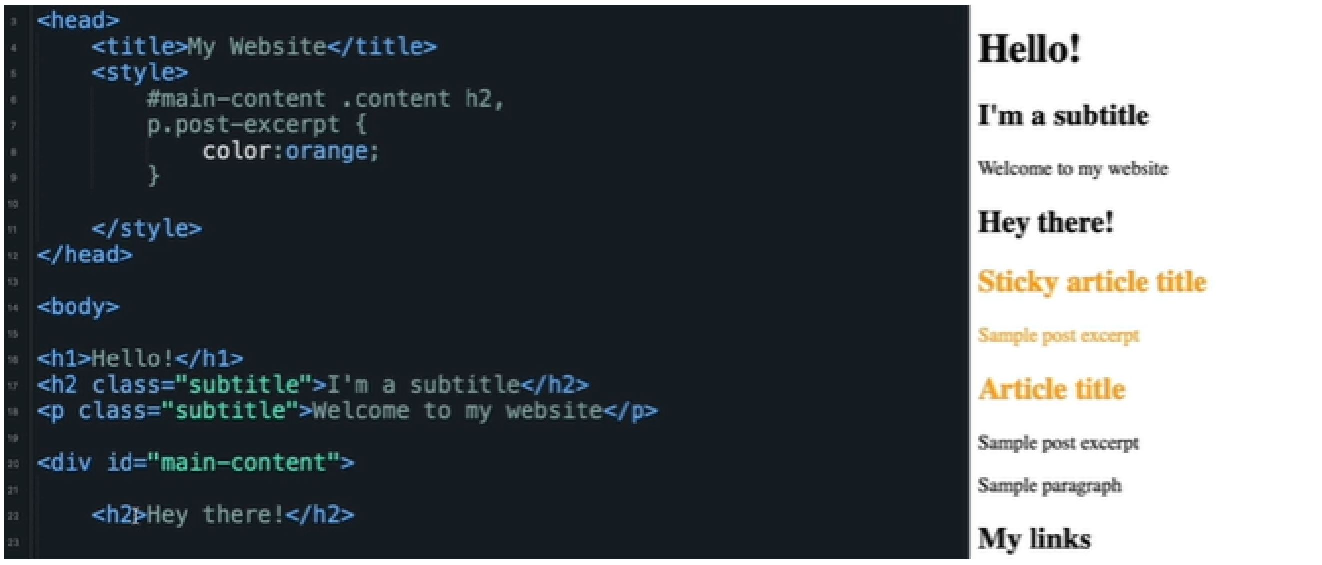
Selecting multiple targets with commas

Another useful character we are using inside our selectors is a comma. Adding commas inside selectors allows us to add a second target to the selector. So, for example, if I want to target all the h2s within these parents from the previous example and also my paragraph with a class "post-excerpt", the selector will look like this: `#main-content .content h2, p.post-excerpt`.

So when using commas inside our selectors, we can specify multiple elements. They will use the same CSS properties, but we can target multiple targets.

Selecting direct descendants with right caret symbol

And another character I would like you to know is the right caret. Let me add a new h2 inside the “main-content” div, so I’m adding `<h2>Hey there!</h2>`.



```
<head>
    <title>My Website</title>
    <style>
        #main-content .content h2,
        p.post-excerpt {
            color:orange;
        }

    </style>
</head>

<body>

<h1>Hello!</h1>
<h2 class="subtitle">I'm a subtitle</h2>
<p class="subtitle">Welcome to my website</p>

<div id="main-content">

    <h2>Hey there!</h2>
```

Hello!

I'm a subtitle

Welcome to my website

Hey there!

Sticky article title

Sample post excerpt

Article title

Sample post excerpt

Sample paragraph

My links

Let's say I want to select that h2 and only that. It doesn't have any specific attribute, so I can look at the parent container. And I know its ID is “main-content”. So to target that h2, I could use a selector like this: `#main-content h2`. Right. But it also targets all these other h2s, which are further down the page source, inside the same parent container.

And I don't want that. I only want to select this one. So there are actually a few different ways we could approach this, but the character I want to show you is the right caret. And it means direct descendant.

`#main-content > h2`

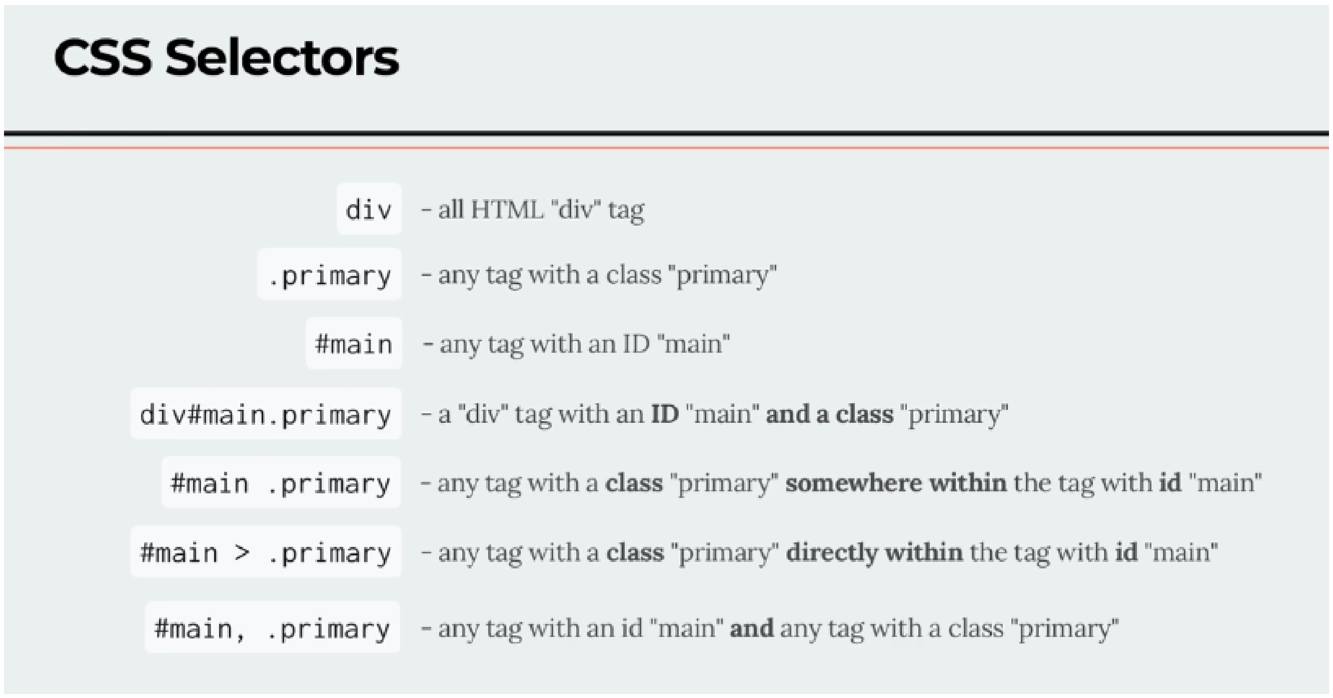
If we are using that, it doesn't need spaces here, so you could remove them: `#main-content>h2`, but it's just easier to read with the spaces.

So this means that the h2 we're targeting is the child of the "main-content" div, but even more, it's the immediate child.

If we were to put this h2 somewhere further down the page source, for example, its direct parent would be something else, not the div ID "main-content". So with the right caret symbol, we can specify the direct parent and that would only select the immediate descendants. And that's the only h2 that's immediately inside that div.

Selectors overview

So a quick recap: we can select elements just by using their HTML tag names. We have the dot symbol for selecting classes, hashtags for IDs. We can combine all these together to select a specific element, a specific tag. We can use the space symbol to define the parent containers, and we can use the right caret to target only direct descendants inside a parent container. And we can use commas to define multiple targets inside one selector.



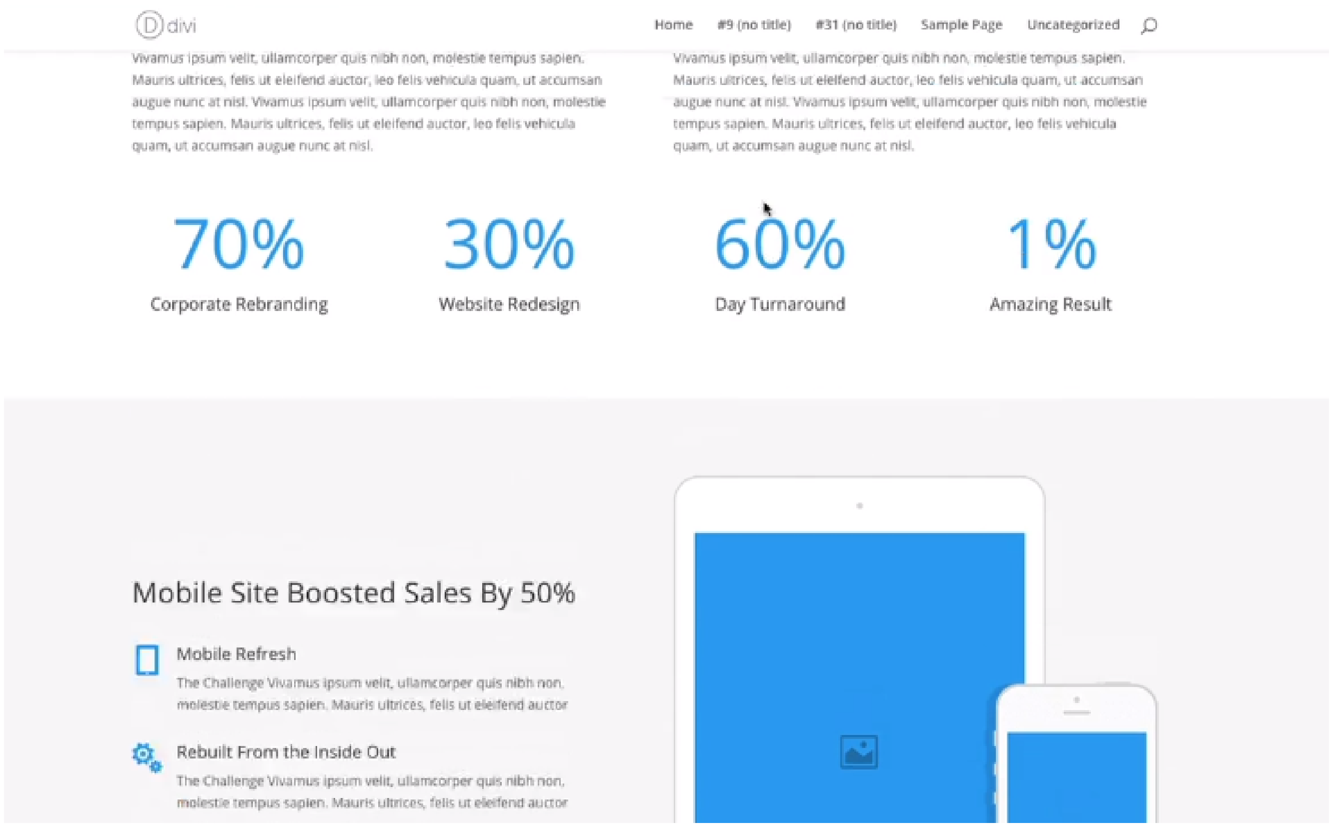
CSS Selectors

`div`	- all HTML "div" tag
`.primary`	- any tag with a class "primary"
`#main`	- any tag with an ID "main"
`div#main.primary`	- a "div" tag with an **ID** "main" **and a class** "primary"
`#main .primary`	- any tag with a **class** "primary" **somewhere within** the tag with **id** "main"
`#main > .primary`	- any tag with a **class** "primary" **directly within** the tag with **id** "main"
`#main, .primary`	- any tag with an id "main" **and** any tag with a class "primary"

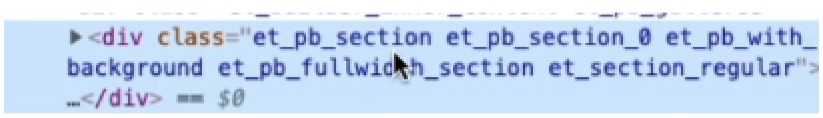
Now let's go ahead and look how we can target the HTML generated by the Divi theme.

Selecting Divi Theme elements

Here is a basic Divi layout, and to be able to target different elements with CSS, you need to get familiar with the HTML generated by the Divi Builder, and we will do that by reading the source code in the Chrome Inspector.



Each section uses the “et_pb_section” CSS class, but that's not all. Many times, you will see multiple CSS classes on a single Divi element.



```
▶<div class="et_pb_section et_pb_section_0 et_pb_with_
background et_pb_fullwidth_section et_section_regular">
…</div> == $0
```

The full-width section has an additional CSS class, and depending on some of the settings you set within the Builder, you might see more additional CSS classes.

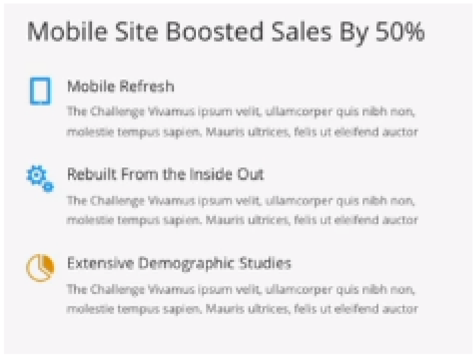
There are also these numbered classes starting from zero. The first section on a page has a CSS class of “et_pb_section_0”. The second section has a class “et_pb_section_1”, and so on, all the way to the bottom of the page. The rows and modules also use these numbered classes, and the numbers for the modules do not reset if the module is inside a new section.

So if we look at the first few Blurbs, for example, we see numbers zero one and two, and then further down, in a new section, there are Blurbs with numbers three and four.

Choosing selectors in Divi

So how do we decide which CSS class to choose when defining our selectors? Choosing the generic module class, for example, et_pb_blurb for the Blurb module, is one of the options. But if I target that class either as a parent or directly that container, it will affect all the Blurbs throughout the entire page. So another option is to use that numbered class, which is unique for each module.

And that is how the Builder does it. Let's see how it looks translated into CSS.



Mobile Site Boosted Sales By 50%

Mobile Refresh

The Challenge Vivamus ipsum velit, ullamcorper quis nibh non, molestie tempus sapien. Mauris ultrices, felis ut eleifend auctor

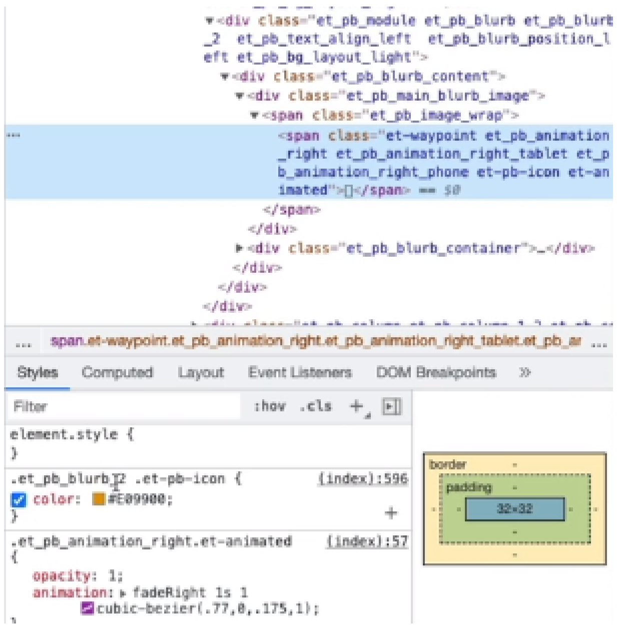
Rebuilt From the Inside Out

The Challenge Vivamus ipsum velit, ullamcorper quis nibh non, molestie tempus sapien. Mauris ultrices, felis ut eleifend auctor

Extensive Demographic Studies

The Challenge Vivamus ipsum velit, ullamcorper quis nibh non, molestie tempus sapien. Mauris ultrices, felis ut eleifend auctor

So for that module, I changed the third icon's color. So here this <span> which is inside that "et_pb_module", "et_pb_blurb_2" class container uses that color.



```
▼<div class="et_pb_module et_pb_blurb et_pb_blurb
_2  et_pb_text_align_left  et_pb_blurb_position_l
eft et_pb_bg_layout_light">
  ▼<div class="et_pb_blurb_content">
    ▼<div class="et_pb_main_blurb_image">
      ▼<span class="et_pb_image_wrap">
          <span class="et-waypoint et_pb_animation
          _right et_pb_animation_right_tablet et_p
          b_animation_right_phone et-pb-icon et-an
          imated"></span> == $0
        </span>
      </div>
    ▶<div class="et_pb_blurb_container">…</div>
    </div>
  </div>
</div>
```

```
span.et-waypoint.et_pb_animation_right.et_pb_animation_right_tablet.et_pb_ar
```

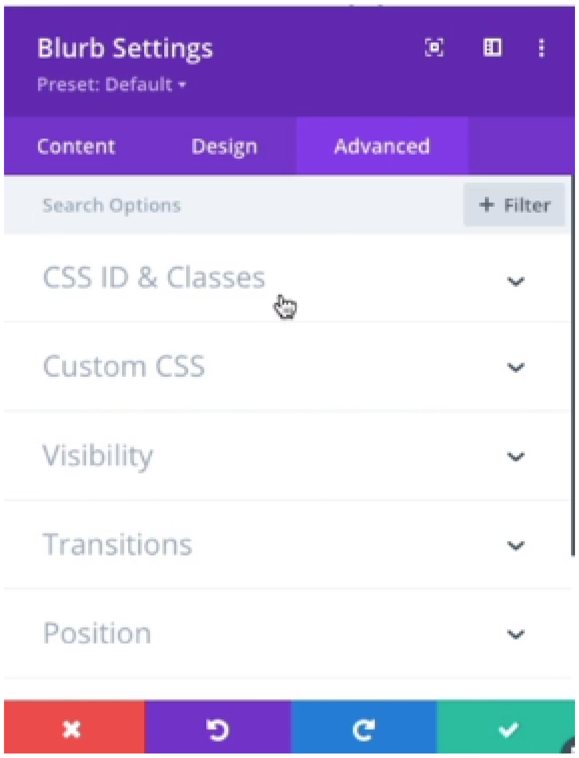
```
element.style {
}
.et_pb_blurb_2 .et-pb-icon {                (index):596
  color: #E09900;
}
.et_pb_animation_right.et-animated          (index):57
{
  opacity: 1;
  animation: fadeRight 1s 1
      cubic-bezier(.77,0,.175,1);
```

So that bit of CSS code: `.et_pb_blurb_2 .et_pb_icon {color: E09900;}` is generated by the Divi Builder for that Blurb number two, which is the third Blurb on that page.

But as soon as I move that Blurb somewhere else and refresh the page... the icon will be using a different selector. So the Divi Builder is smart enough to generate the CSS code that always will work.

But if I wanted to change something about that Blurb with CSS, and I used that numbered class, and then I moved it to a different place - its number would change and the selector wouldn't work correctly.

So it's not the best choice to use these numbered classes. Luckily, we can use the custom CSS option in the Advanced tab of every element, every section, row, or column inside the row. And in every module, we can add our own CSS class and CSS ID, so if you want to change something specific about a single element, you can add your own CSS class.



It will be added to the main container. If I add a custom CSS class “my-class” to a module, it will be visible as another class added to that container in the Inspector. I can even use multiple classes, assign multiple classes to a single element.

Basically, using your own CSS class is the safest way to add your custom code. Sometimes it's better to use an ID, as adding an ID gives your selectors more specificity value, which we'll talk more about soon. But just remember that you can only assign one CSS ID, and multiple CSS classes, and you can target all the elements within your modules using your custom attribute, customs class or ID as the parent container.

And this way, for example, you could add more space below the header inside the Blurb module, which might not be inside the module settings. And the only way to target it is with custom CSS. So using our own selector is the recommended approach.

I hope it will give you some clarity on how to define your selectors in CSS and how to use it within Divi. And in the next lesson, we will look into using pseudo-classes on these selectors.

Resources

UNDERSTAND SELECTORS:

Selectors Explained is a dictionary that basically translates CSS selectors into English

CSS Cheatsheet (available in the Downloads section of this lesson)

Action Items

- ☐ Navigate to the Selectors Explained online tool, paste or write a sample selector and see the explanation
- ☐ Choose a single element on your site and try to define 3 different selectors you could use to target it. Do that for as many elements as possible.
- ☐ Download and print the CSS Cheatsheet if you haven't yet.