3 Adding Text to our New Div

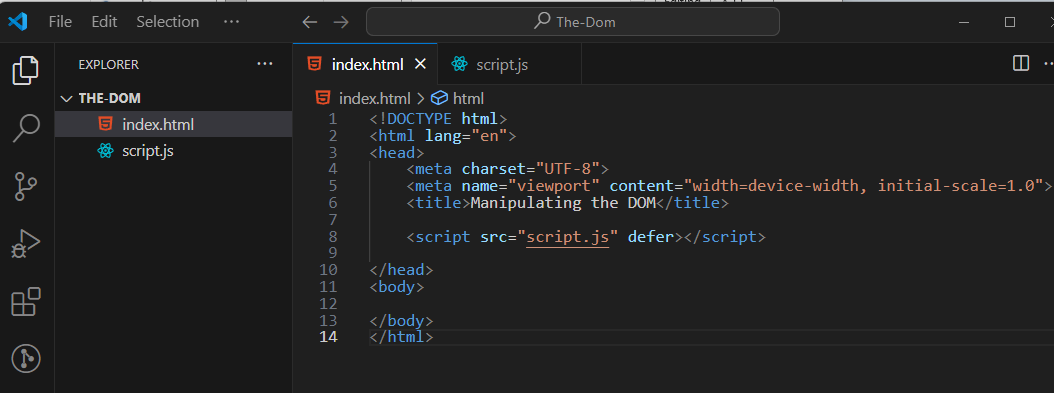
<https://www.youtube.com/watch?v=y17RuWkWdn8>

**Our Starting Point**

We can start (again) here with the index page that we created in last week’s tutorial entitled:

*index.html*

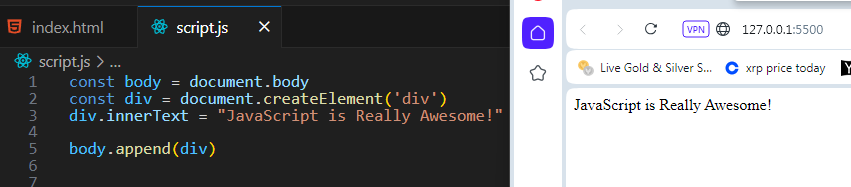
And if you do start with that, you should already have the script file created, and attached to the index file.



# InnerText

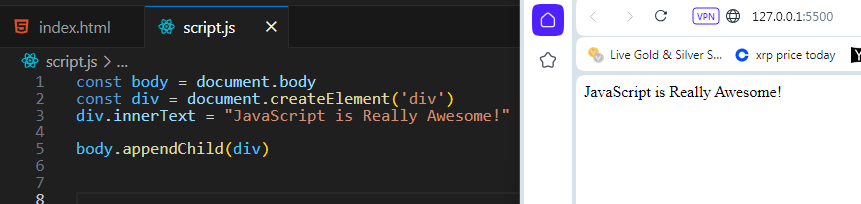
We have a couple of ways in which we can insert text into our web page. One way is to use innerText. Remember that we do have an empty div that was appended to the body. Now we want to attach our inner text to that empty div. Attach it using the dot operator.

This is not a method here, it is an assignment. The equal sign is used to attach the text to the div.InnerText.



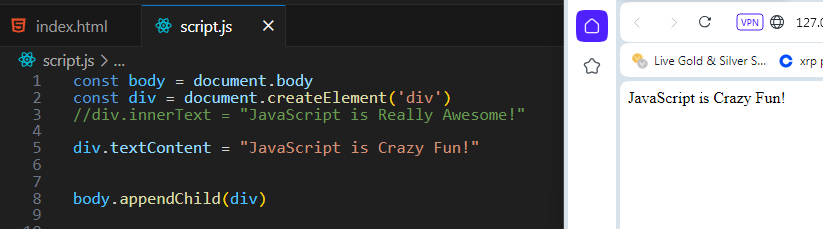
# appendChild

Remember when I said that **appendChild** would work with a **node**? Well, a **div** is considered a node in JavaScript. Yes, maybe a node might be a strange sounding word, but that is what a **div** is; and guess what? Now that we do have our text inside of a div, we can switch out append with appendChild and it will work just fine. That is because now our text is a child of the div.



# textContent

Another way in which we can have our text show up on the web page is to use textContent. It works the same way as **innerText**.



# The Difference Between innerText and textContent

If you just take a look at these two ways of adding text and what it looks like on a web page, you would think that both of them are exactly the same thing. But you would be wrong because under-the-hood, they are actually different.

Using console.log in your browser

The use of textContent will print out all of the text content, with all of the spacing and indentation intact inside of the div. And this text would print out in the console even if you set the display of that text to have a display of **none**, which means you set it to be invisible.

On the other hand, if you were to use innerText, and then looked at it in the console of the browser, you would find that setting the text to display **none**, would actually have meaning and that text would be invisible.

