The above code may seem intimidating at first, but it’s quite straightforward once you break it down.

We start the code by defining a function, currentTime(). Inside the function, we store the current time in a variable called date.

We do that by using the new Date() constructor, this constructor returns the browser’s date along with the time zone.

**Note:** The date object is static, we would have to keep updating it, we do that later in the code.

let date = new Date();

Once this is done, we extract the hours, minutes, and seconds from the variable (date) using the getHours(), getMinutes() and getSeconds() methods.

These methods return the respective values when a date is passed, we store them in different variables.

lastly, we use a variable called "session" to store the AM or PM tag.

let hh = date.getHours();

let mm = date.getMinutes();

let ss = date.getSeconds();

let session = "AM";

**Note**: Here date in date.getHours(), etc, is the variable we used to store the current date earlier.

The date.getHours method returns values between 0-23, and given we are programming a 12 hours clock we use the following if statement to reset 12 to 0.

if(hh == 0){

hh = 12;

}

And, we use another if to subtract hours greater than 12 and to assign the value of the session variable to “PM”.

if(hh > 12){

hh = hh - 12;

session = "PM";

}

To understand the next bit of code, you need to be familiar with two concepts.

Firstly, the getHours(), getMinutes() and getSeconds() methods return values between 0 to 23, 59, 59 respectively. The key point here is that single digit values are returned as # (eg: 7), however, in our clock, these values need to be displayed as ## (eg: 07).

And to achieve this we use ternary operators. This operator returns a value if the condition is true and another value if it is false. I’ve added the syntax below.

(condition ? if true : if false);

Using this operator, we solve the above problem by adding a 0 before the digit that is less than 10.

hh = (hh < 10) ? "0" + hh : hh;

mm = (mm < 10) ? "0" + mm : mm;

ss = (ss < 10) ? "0" + ss : ss;

Next, we create a variable time to store the time in the desired format.

let time = hh + ":" + mm + ":" + ss + " " + session;

To display the time we use the following code.

document.getElementById("clock").innerText = time

Here, document represents the webpage, .getElementbyId() returns the element whose ID has been passed as the parameter in our case “clock”. And the innertext property sets the context of the node to time.

And lastly, the most important part, remember I mentioned that get date() returns a static value. We use the setTimeout() method to update it. I’ve added the syntax for it below.

setTimeout(function, milliseconds, param1, param2, ...)

This method calls or runs a function after a specified number of milliseconds. Note: 1000ms = 1 second.

We use this method to update the code every second, to keep our clock running.

let t = setTimeout(function(){ currentTime() }, 1000);

And all that’s left is to run the function.

currentTime();