Why did we used endCount() clearInterval(countDown) in code ?

***Endcount***()

This function updates the heading (h1) to display "Time out".

Without this, the UI would still show 00:00 instead of a clear message indicating that the countdown is over.

**clearInterval(countDown)**

Stops the interval called countdown to prevent unnecessary execution.

It is a built-in JavaScript function, and it takes the setInterval() function’s identifier (ID) as an argument.

setInterval() returns a unique interval ID(number) when it is created.

This ID is stored in a variable (countDown in our case).

clearInterval(intervalID) stops the interval when called.

Is there some reason we stored setinterval function in a variable countdown instead of using it directly ?

We must store the reference (ID) of setInterval() in a variable if we want to stop it later using clearInterval().

If we use directly without variable there's no way to call clearInterval() because we don’t have the interval's ID.

Do we not need a function keyword before setinterval always , or we omit it as it’s kind of a function expression in this scenerio?

Function keyword is needed only when we declare a function.

setInterval is a built-in function, here we are calling it , like any other function (console.log(), alert(),func() , foo() etc.)

If we still add this keyword before it we will get error

setInterval is already a built-in function, and you are trying to redefine it as a function.

Why we had to wrap this part in setinterval method

if (timeSecond == 0 || timeSecond < 1) {

endCount()

clearInterval(countDown)

}

Will we had gotten error if it would have been out of this method ?

This condition must be inside setInterval because:

It needs to be checked repeatedly – Every second, we need to check if timeSecond has reached 0.

If it were outside, it would only run once – If we placed this check outside setInterval, it would execute only once when the script runs, not after each countdown update.

Explain this part of code with depth ?

function displayTime(second) {

const min = Math.floor(second / 60);

const sec = Math.floor(second % 60);

timeH.innerHTML = `

${min < 10 ? "0" : ""}${min}:${sec < 10 ? "0" : ""}${sec}

`

}

This function takes one argument (second), which represents the total number of seconds left in the countdown.

When we enter any number it’s 5 in our case let’s suppose if it is 90

**Minutes Calculation:**

second / 60 gives the total number of minutes (e.g., 90 / 60 = 1.5).

Math.floor(...) ensures we only take the whole number part (e.g., 1.5 → 1).

**Seconds Calculation:**

second % 60 gives the remaining seconds after converting into minutes.

Example: 90 % 60 = 30, so 90 seconds = 1 minute, 30 seconds.

What does it mean by ( min < 10 ? "0" : "" ) ?

${min < 10 ? "0" : ""}${min}

If min is less than 10, it adds a leading zero.

Example:

9 → "09"

12 → "12" (no extra zero needed)

${sec < 10 ? "0" : ""}${sec}

If sec is less than 10, it adds a leading zero.

Example:

5 → "05"

23 → "23" (no extra zero needed)