**JavaScript Tutorial: setInterval & setTimeOut | Web Development Tutorials #58**

In this tutorial, we are going to learn about **setInterval**and **setTimeOut**functions and how they are useful in JavaScripts. Make a new file as *tut58.html* and add the instant boilerplate to get the basic HTML template. Then give the title as **JavaScript SetTimeout and ClearTimeOut**under the <title> tag**.**

Now write the HTML code to get started with the work-

<div class="container">

Time now is <span id="time"></span>

</div>

Copy

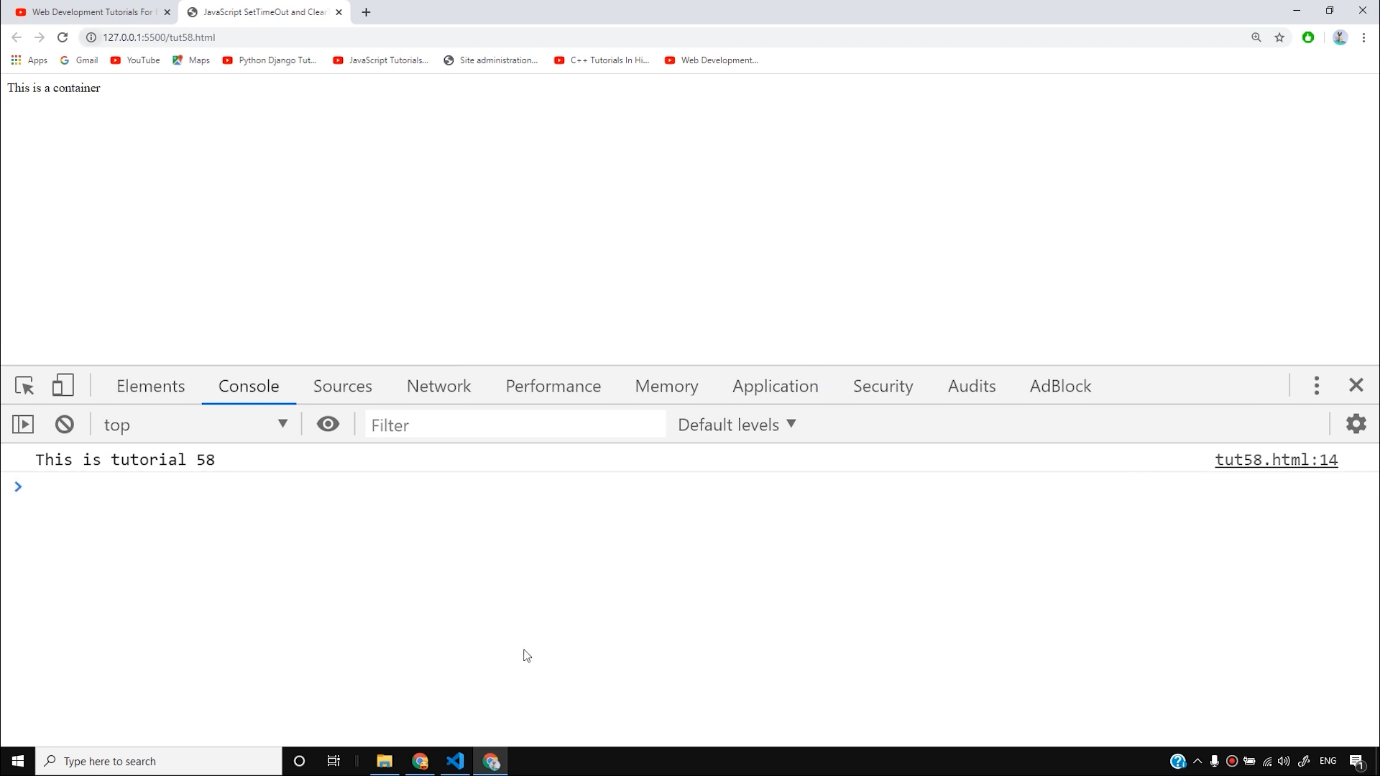
Now we are going to write the Javascript under the <script> tag. We can check if the JavaScript is working or not by writing a very simple code as-

<script>

console.log("This is tutorial 58");

Copy

If we get the following output then everything is fine.



There are times when we want a particular function should run after a specific interval of time or it should run repeatedly after certain intervals. In that case, we can use setTime and setInterval functions respectively.

The **setTimeout()**method calls a function or evaluates an expression after a specified number of milliseconds. This function is only executed once. Use the **clearTime()**method to prevent the function running.

If we write as follows-

function greet(){

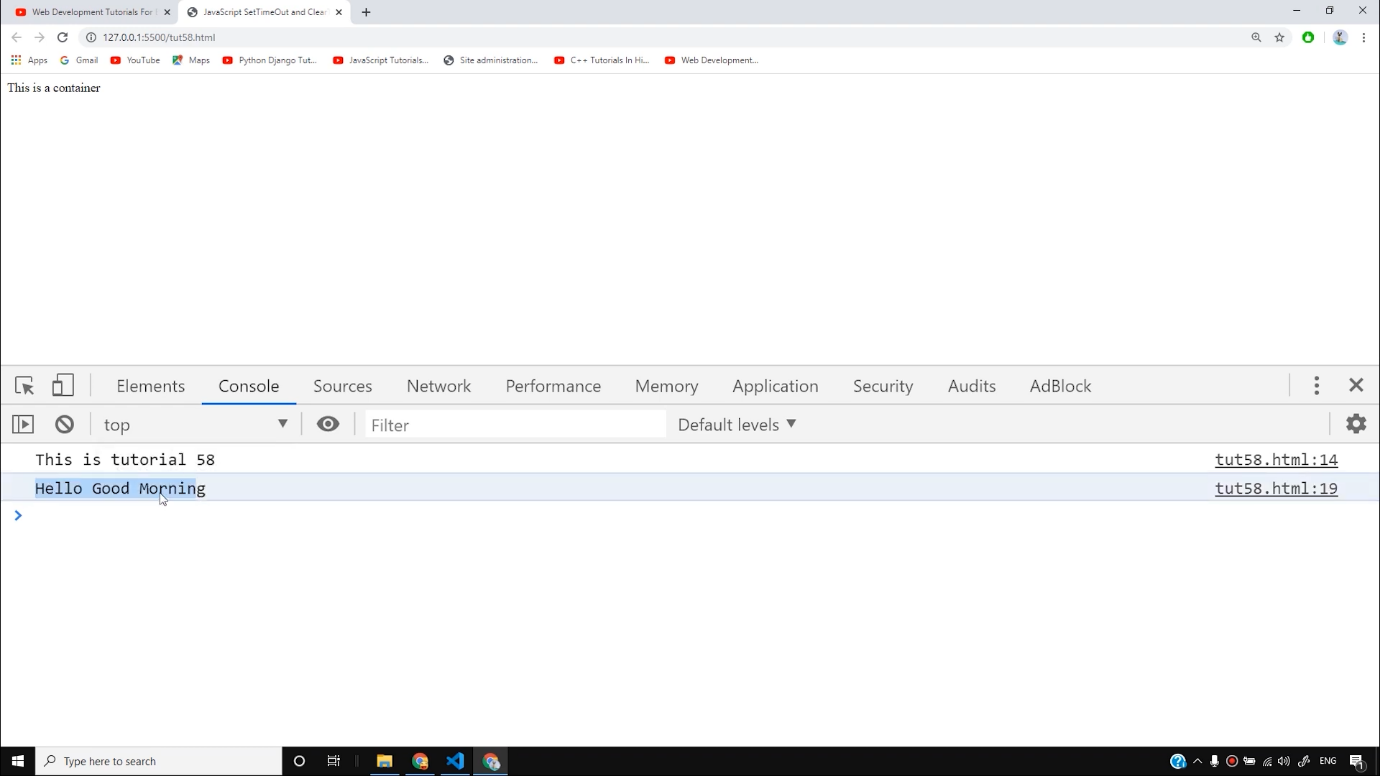
console.log("Hello Good Morning ");

}

setTimeout(greet, 5000);

Copy

If we run the above code then the message will be displayed as follows after 5000ms (5sec).



We can also pass more arguments in this function. One thing to note here is we can never pass the function in setTimeout. Else it returns the output immediately irrespective of what time interval you have set.

To clear the timeout, we can use the function **clearTimeout()**as follows-

console.log(timeOut);

Copy

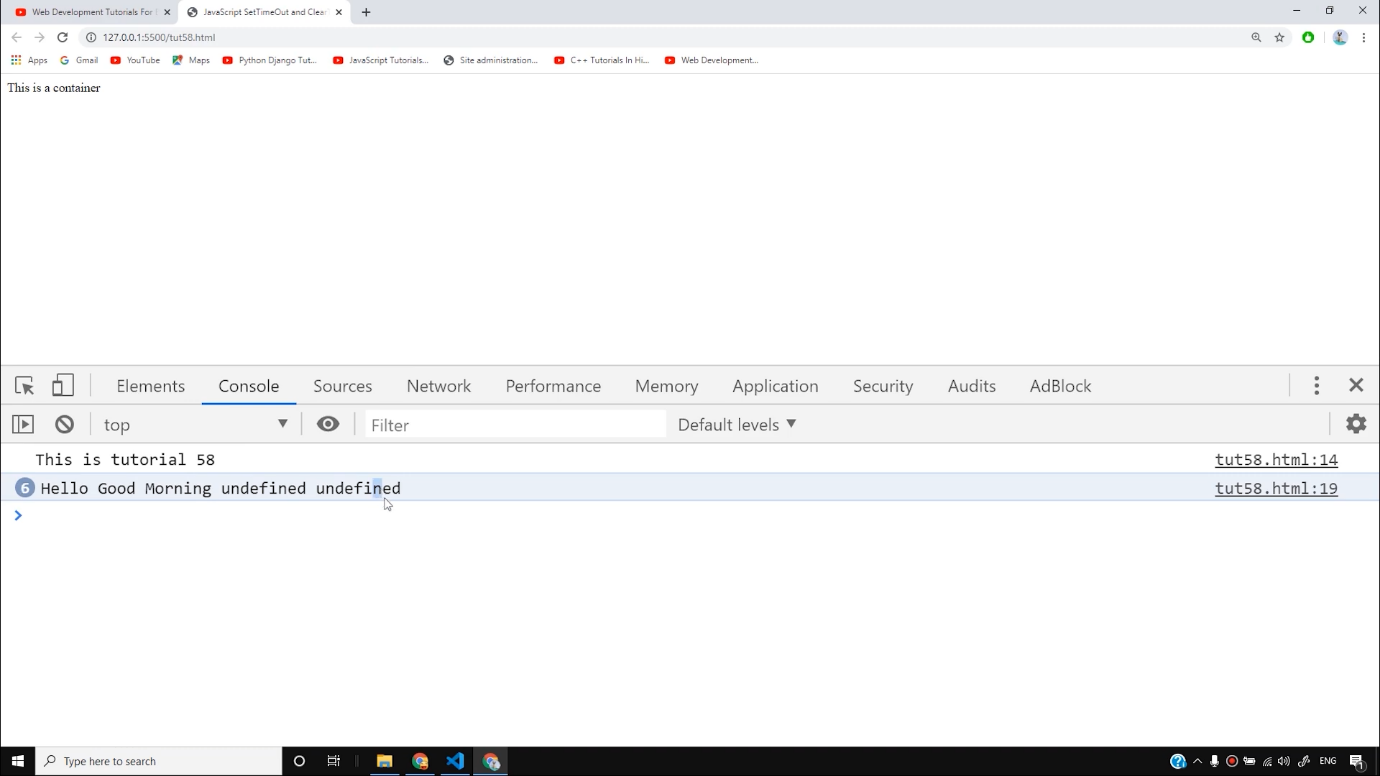
Now let us see how the setInterval function is used. It calls a function or evaluates an expression at specified intervals in milliseconds. It will continue calling the function until **clearInterval()**function is called or the window is closed. Let us understand with an example.

If we write as follows-

setInterval(greet, 1000, "Harry", "Good Night");

Copy

After running this function, you will notice that the message is being returned after every single second as follows-



To stop this, we have to use clearInterval() as follows-

clearInterval(intervalId);

Copy

Let us now see another example of the above codes. If we write as follows-

function displayTime(){

time = new Date();

console.log(time);

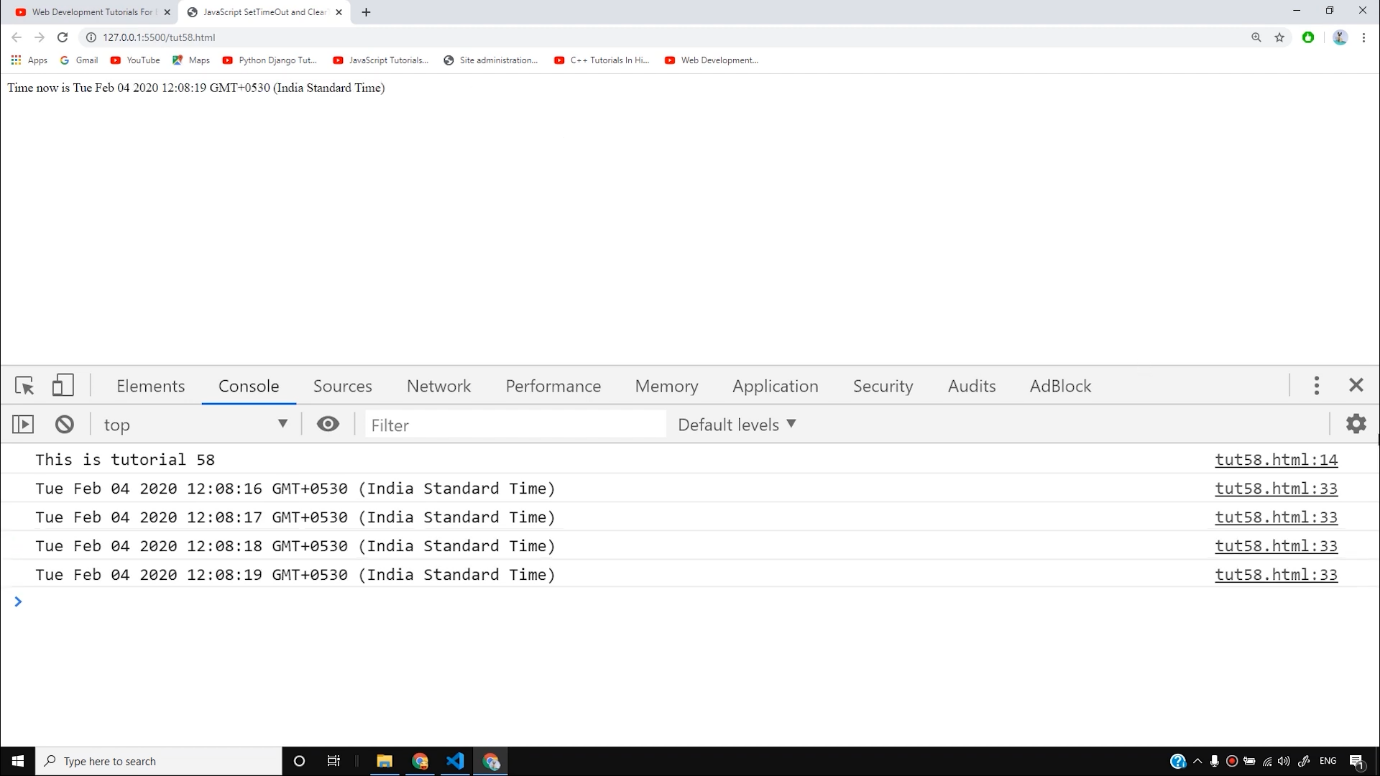
document.getElementById('time').innerHTML = time;

}

setInterval(displayTime, 1000);

Copy

If we run the above code, we will see that after every second the current time is being displayed as follows-



With the help of these functions, we can create many websites or even websites that display a stopwatch. So I hope you must have understood these concepts. In the upcoming videos, we will see some more practical implementation of them.

**Code as described/written in the video**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>JavaScript SetTimeOut and ClearTimeout</title>

</head>

<body>

<div class="container">

Time now is <span id="time"></span>

</div>

<script>

console.log("This is tutorial 58");

// setTimeout --> Alllows us to run the function once after the interval of time

// clearTimeout --> Alllows us to run the function repeatedly after the interval of time

function greet(name, byeText){

console.log("Hello Good Morning " + name + " " + byeText);

}

// timeOut = setTimeout(greet, 5000, "Harry", "Take Care");

// console.log(timeOut);

// clearTimeout(timeOut);

// setTimeout(greet(), 12000); --> Wrong as it is calling the function inside setTimeout

// intervalId = setInterval(greet, 1000, "Harry", "Good Night");

// clearInterval(intervalId);

function displayTime(){

time = new Date();

console.log(time);

document.getElementById('time').innerHTML = time;

}

setInterval(displayTime, 1000);

</script>

</body>

</html>

# JavaScript Timing Events

[❮ Previous](https://www.w3schools.com/js/js_popup.asp)[Next ❯](https://www.w3schools.com/js/js_cookies.asp)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | JavaScript can be executed in time-intervals.  This is called timing events. |

## **Timing Events**

The window object allows execution of code at specified time intervals.

These time intervals are called timing events.

The two key methods to use with JavaScript are:

* setTimeout(function, milliseconds)  
  Executes a function, after waiting a specified number of milliseconds.

* setInterval(function, milliseconds)  
  Same as setTimeout(), but repeats the execution of the function continuously.

The setTimeout() and setInterval() are both methods of the HTML DOM Window object.

## **The setTimeout() Method**

window.setTimeout(*function*,*milliseconds*);

The window.setTimeout() method can be written without the window prefix.

The first parameter is a function to be executed.

The second parameter indicates the number of milliseconds before execution.

### **Example**

Click a button. Wait 3 seconds, and the page will alert "Hello":

<button onclick="setTimeout(myFunction, 3000)">Try it</button>  
  
<script>  
function myFunction() {  
  alert('Hello');  
}  
</script>

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_timing1)

ADVERTISEMENT

## **How to Stop the Execution?**

The clearTimeout() method stops the execution of the function specified in setTimeout().

window.clearTimeout(timeoutVariable)

The window.clearTimeout() method can be written without the window prefix.

The clearTimeout() method uses the variable returned from setTimeout():

myVar = setTimeout(*function*,*milliseconds*);  
clearTimeout(myVar);

If the function has not already been executed, you can stop the execution by calling the clearTimeout() method:

### **Example**

Same example as above, but with an added "Stop" button:

<button onclick="myVar = setTimeout(myFunction, 3000)">Try it</button>  
  
<button onclick="clearTimeout(myVar)">Stop it</button>

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_settimeout2)

## **The setInterval() Method**

The setInterval() method repeats a given function at every given time-interval.

window.setInterval(*function*,*milliseconds*);

The window.setInterval() method can be written without the window prefix.

The first parameter is the function to be executed.

The second parameter indicates the length of the time-interval between each execution.

This example executes a function called "myTimer" once every second (like a digital watch).

### **Example**

Display the current time:

setInterval(myTimer, 1000);  
  
function myTimer() {  
  const d = new Date();  
  document.getElementById("demo").innerHTML = d.toLocaleTimeString();  
}

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_setinterval2)

There are 1000 milliseconds in one second.

## **How to Stop the Execution?**

The clearInterval() method stops the executions of the function specified in the setInterval() method.

window.clearInterval(timerVariable)

The window.clearInterval() method can be written without the window prefix.

The clearInterval() method uses the variable returned from setInterval():

let myVar = setInterval(*function*,*milliseconds*);  
clearInterval(myVar);

### **Example**

Same example as above, but we have added a "Stop time" button:

<p id="demo"></p>  
  
<button onclick="clearInterval(myVar)">Stop time</button>  
  
<script>  
let myVar = setInterval(myTimer, 1000);  
function myTimer() {  
  const d = new Date();  
  document.getElementById("demo").innerHTML = d.toLocaleTimeString();  
}  
</script>