JavaScript Timers & Recursion

Example Script #1 - The Full Script

```
<p><a href="#" id="stop">Stop the script</a></p>
<div id="thediv">Default Content</div>

<script>
    var theDiv = document.getElementById("thediv");
    var content ="<p>adding another paragraph</p>";

    var timer = setInterval( function(){ theDiv.innerHTML += content; }, 1500 );

    document.getElementById("stop").addEventListener("click", function(){ clearInterval(timer); });
</script>
```

Try running this example script and see what you get.

Two Functions for Timers in JavaScript

There are two functions for timers in JavaScript. There is setInterval() and setTimeout().

setTimeout(function-to-run, amount-of-time).

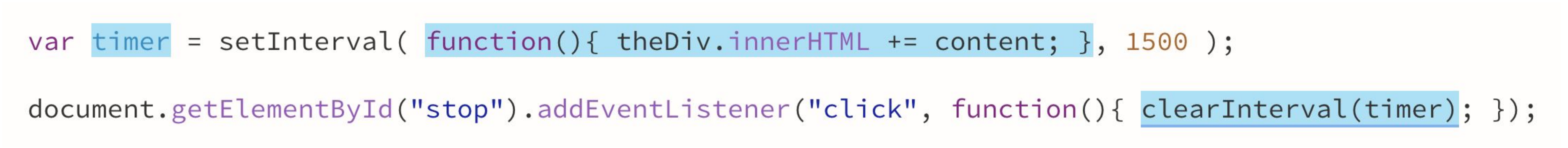
setInterval(function-to-run, amount-of-time).

Examples:

setTimeout(wakeUp, 8 hours);
setInterval(eatMeal, 4 hours);

Here, setTimeout would be used like an alarm that runs once, whereas setInterval runs every 4 hours.

Example #1 - Using the setInterval Function



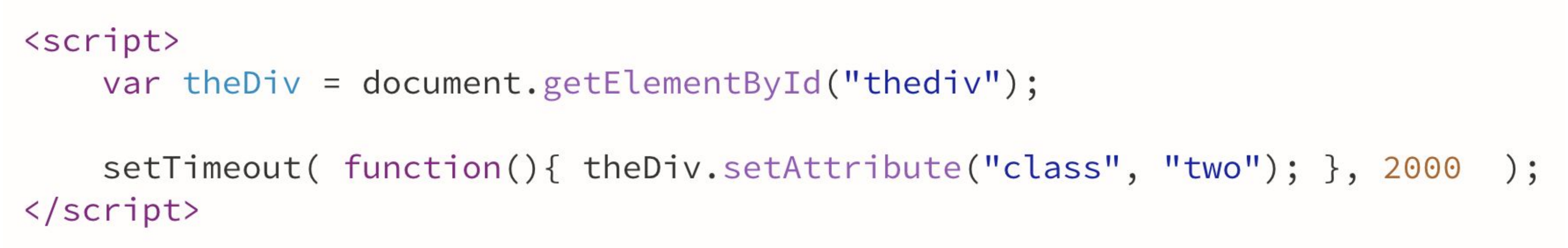
```
var timer = setInterval( function(){ theDiv.innerHTML += content; }, 1500 );

document.getElementById("stop").addEventListener("click", function(){ clearInterval(timer); });
```

The anonymous callback function takes theDiv and adds content to it every 1500 milliseconds, or every 1.5 seconds.

The setInterval function is assigned to a variable. This is called a function expression.

In the click handler, clearInterval is used to stop the timer.

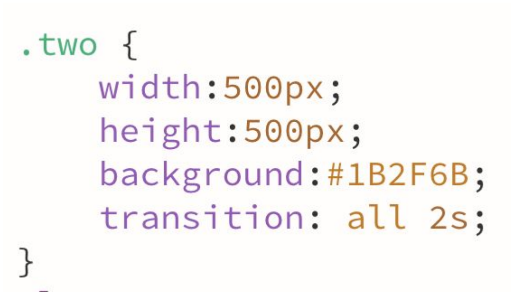
Example 2:



```
<script>
    var theDiv = document.getElementById("thediv");

    setTimeout( function(){ theDiv.setAttribute("class", "two"); }, 2000  );
</script>
```

In example 2, the function runs just one time, after two seconds of waiting.

It is important to note that the animation you see is happening in the CSS



```
.two {
    width:500px;
    height:500px;
    background:#1B2F6B;
    transition: all 2s;
}
```

Example #3 - Recursion

Notice in this script, setTimeout is used, and it runs only once. There is no loop, yet the script continues.

This is because of recursion, which happens when a program calls itself.

In both the if and else statements, whichever one runs, after two seconds, classRotator() is called again.

```
<script>
    var theDiv = document.getElementById("thediv");
    var currentClass = "one";

    function classRotator()
    {
        if( currentClass == "one" )
        {
            setTimeout( function(){
                theDiv.className="two";
                currentClass = "two";
                classRotator(); }, 2000 );
        }
        else
        {
            setTimeout( function(){
                theDiv.className="one";
                currentClass = "one";
                classRotator(); }, 2000  );
        }
    }

    classRotator();

</script>
```

Summary

The setInterval and setTimeout functions are methods of the DOM window object.

Later in the course, you will get deeper into the event model in JavaScript and see how it’s important to understand that these methods tie into the browser, and not the JavaScript engine itself.