Set 24: Timers

Skill 24.01: Explain the need for timers

Skill 24.01 Concepts

Programmers use timing events to delay the execution of code or to repeat code at a specified interval. There are two native functions in the JavaScript library used to accomplish these tasks: *setTimeout()* and *setInterval()*.

The *setTimeout()* function is used to delay the execution of a function by a specified amount of time.

The *setInterval()* function is used to specify the time interval for which a function should be repeated.

Both *setTimeout()* and *setInterval()* allow us to make our applications more interesting by controlling the timing of our functions.

Skill 24.02: Apply the *setTimeout()* function

Skill 24.02 Concepts

You use *setTimeout()* to delay the execution of a function by a specified amount of time. There are two parameters that you pass to *setTimeout()*: the function you want to call, and the amount of time in milliseconds. (There is 1000 milliseconds(ms) in 1 second. Ex: 5000 ms = 5 seconds.) *setTimeout()* will execute one time after the specified time has elapsed. Below is an example,

```
var timer;
delayTimer();

function delayTimer() {
  setTimeout(delayedFunction, 3000);
}

function delayedFunction() {
  alert("Three seconds have elapsed.");
}
```

Skill 24.02 Exercise 1

Skill 24.03: Apply the *setInterval()* function

Skill 24.03 Concepts

You use setInterval() to specify a function to repeat with a time delay between executions. Again, two parameters are required for setInterval(): the function you want to call, and the amount of time in milliseconds. setInterval() will continue to execute until it is cleared.

```
var timer2;
repeatEverySecond();

function repeatEverySecond() {
  timer2 = setInterval(sendMessage, 1000);
}

function sendMessage() {
  var d = new Date();
  document.body.innerHTML = d.toLocaleTimeString();
}
```

Skill 24.03 Exercise 1

Skill 24.04: Stop a timer event

Skill 24.04 Concepts

There are two corresponding native functions to stop the above timing events: *clearTimeout()* and *clearInterval()*.

You may have noticed that each timer function is saved to a variable. When the set function runs it is assigned a number which is saved to this variable. This generated number is unique for each instance of a timer. This assigned number is also how timers are identified to be stopped. For this reason, you should always set your timer to a variable.

To stop a timer, call the corresponding clear function and pass it the timer ID variable that matches the timer you wish to stop. The syntax for *clearInterval()* and *clearTimeout()* are the same and are illustrated below,

```
var timeoutID;
delayTimer();

function delayTimer() {
  timeoutID = setTimeout(delayedFunction, 3000);
}

function delayedFunction() {
  alert("Three seconds have elapsed.");
  clearAlert();
}
```

```
function clearAlert() {
  clearTimeout(timeoutID);
}
```

Skill 24.04 Exercise