# JavaScript Homework #5

# Callbacks and timers

Timers created using setTimeout() or setInterval() can be used to achieve interesting and useful functionality. Despite the common misconception that timers in JavaScript are actually threads, JavaScript runs in a single-threaded environment. Timers, then, simply schedule code execution to happen at some point in the future. When a timer is trigged, its associated callback function will be executed.

Timers work with JavaScript event execution by inserting the callback funciton when a particular amount of time has passed. Note that adding the function to the queue doesn’t mean it’s executed immediately; it simply means that it will be executed as soon as possible after the event (timer expires) is trigged. Setting a timer for execution in 150 milliseconds doesn’t mean that the code will be executed in 150 milliseconds; it means that the code will be added to the queue in 150 milliseconds. If nothing else is in the queue at that point in time, the timer code will be executed, giving the appearance that the code executed exactly when specified. At other times, the code may take significantly longer to execute.

Write a function that sets a specific timer (with SetTimeout) to insert a new message into your html. Then, the function is called whenever a button in your html is clicked.

Timers created using setInterval() ensure regular injection of timer code into the queue. As such, write another function to use setInterval() instead and repeat the same as above, except this time a user only needs to click once.