Local Skills - HelloWorldpart3_TimerEvent

In part 3 of *Local Skills - Hello World* we'll use timed events to trigger a change in Misty's chest LED every second. Timed events allow us to specify an amount of time to pass before an event occurs and the callback is triggered. Also, we'll introduce global variables and demonstrate how they persist across new threads.

HelloWorldpart3_TimerEvent.json

```
{
      "Name": "HelloWorldpart3_TimerEvent",
      "UniqueId": "8a380289-2939-4e81-94d9-86d511b7a8ce",
      "Description": "Local 'Hello, World!' tutorial series, part 3.",
      "StartupRules": [ "Manual", "Robot" ],
      "Language": "javascript",
      "BroadcastMode": "verbose",
      "TimeoutInSeconds": 300,
      "CleanupOnCancel": false,
      "WriteToLog": false
}
```

HelloWorldpart3_TimerEvent.js

When registering for a timed event use the `RegisterTimerEvent()` method. Pass in the name of the event we're creating, the amount of time (in ms) we want Misty to wait before triggering the callback function, and set the `keepAlive` parameter to `true` in order to have the event trigger the callback automatically every second until it is unregistered.

```
misty.RegisterTimerEvent("TimerEvent", 1000, true);
```

Define a global variable to track the amount of callbacks that have been triggered. In order for the data to persist across new threads created by callbacks, prefix the name with an underscore. Initialize the variable as "1" to represent the first time we are calling the event.
Note: You don't have to incude a type when creating global variables.

```
_count++;
```

In the callback function (automatically named `_TimerEvent`) we'll generate three random values between 0 and 255 and pass them in to `ChangeLED` to trigger a change in Misty's chest LED:

```
function _TimerEvent() {
   let value1 = Math.floor(Math.random() * (256));
   let value2 = Math.floor(Math.random() * (256));
   let value3 = Math.floor(Math.random() * (256));
   misty.ChangeLED(value1, value2, value3);
}
```

Increment `_count` by one to keep track of the amount of times we are changing the LED. Then, check if it is greater than or equal to 6. If so, turn the LED off and unregister the timer event, ending the skill.

```
if (_count >= 6) {
   misty.ChangeLED(0, 0, 0); // off
   misty.UnregisterEvent("TimerEvent");
   misty.Debug("ending skill helloworld part3");
}
```

Using timed events we have told Misty to change her chest LED to a random color in one-second intervals. We've demonstrated how we can use global variables prefixed with an underscore to have data persist across threads that are created in our program as callbacks are triggered. This is a simple example of two powerful tools that you have at your disposal when writing local skills for Misty. See the skill file below for reference.

```
// Debug message to indicate the skill has started
misty.Debug("starting skill helloworld part3");

// Register for the timer event, specifying the duration of the timer
misty.RegisterTimerEvent("TimerEvent", 1000, true);

// Global variable to track the amount of callbacks triggered
_count = 1;

// Callback specified for Timer event
function _TimerEvent() {
      // Specify random RGB values and send command to change LED
    let value1 = Math.floor(Math.random() * (256));
      let value2 = Math.floor(Math.random() * (256));
      let value3 = Math.floor(Math.random() * (256));
    misty.ChangeLED(value1, value2, value3);
```

```
	// Increment count by 1
	_count++;

	// Check if count is greater than or equal to 6
	if (_count >= 6) {
		// If so, turn off LED, unregister for the timer event and
		// signal end of skill
		misty.ChangeLED(0, 0, 0); // off
		misty.UnregisterEvent("TimerEvent");

		misty.Debug("ending skill helloworld part3");
	}
}
```