

COMP4021  
Internet Computing

# Using Timers

David Rossiter & Gibson Lam

# Using Timers

- You can use these JavaScript functions for controlling the exact timing of running some code:
  - `setTimeout()`
  - `clearTimeout()`

# setTimeout()

- For example, you can use `setTimeout()` to show an alert message 5 seconds later:

```
setTimeout(function() {  
    alert("Ring! Ring!");  
}, 5000);
```

*5000 milliseconds = 5 seconds*

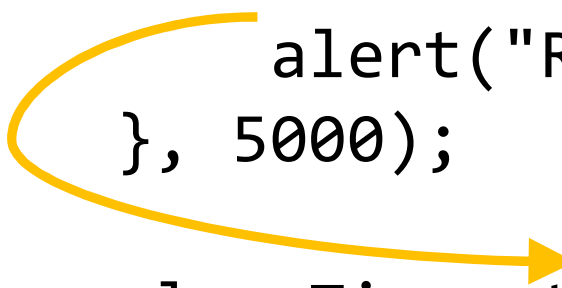
- This function will be executed 5 seconds later

# Cancelling a Timeout

- You can easily cancel a timeout using `clearTimeout()`
- To do that, when you set a timeout, you need to remember the timeout id, like this:

```
var timerid = setTimeout(function() {  
    alert("Ring! Ring!");  
}, 5000);
```

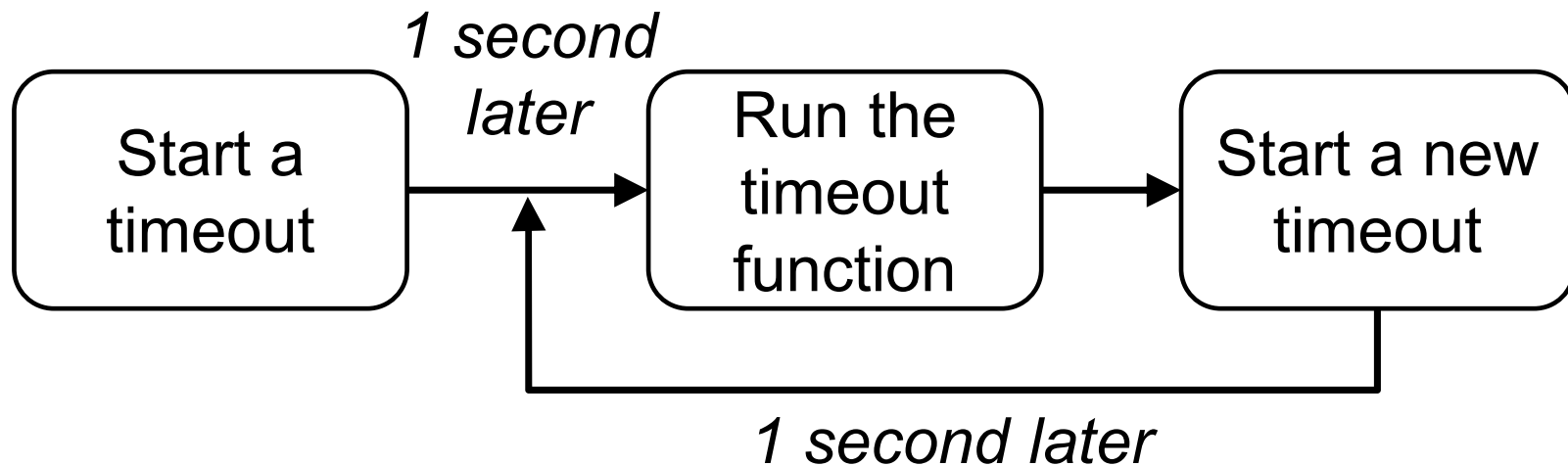
when you create a timer using `setTimeout`, it will return a `timerid`.



```
clearTimeout(timerid); // now nothing will happen
```

# Making a Repeating Timer

- You can easily make a repeating timer using `setTimeout()`
- You just keep making new timeouts when a previous one fires



# Example of a Countdown Timer

- Here is an example that shows a countdown of 5 seconds

```
var timeRemaining = 5;
```

```
function countDown() {  
    timeRemaining = timeRemaining - 1;  
    document.getElementById("time")  
        .textContent = timeRemaining;  
    if (timeRemaining > 0)  
        setTimeout(countDown, 1000);  
}
```

```
setTimeout(countDown, 1000);
```

*Update the display of time in  
an HTML element called 'time'*

`<h1 id="time">5</h1>`

*Start the next timeout when  
the time is greater than zero*

# Running the Code

