**Node.js - EventEmitter**

Node.js uses events module to create and handle custom events. The EventEmitter class can be used to create and handle custom events module.

The syntax to Import the events module are given below:

**Syntax:**

const EventEmitter = require('events');

All EventEmitters emit the event newListener when new listeners are added and removeListener when existing listeners are removed

Listening events: Before emits any event, it must register functions(callbacks) to listen to the events.

**Syntax:**

eventEmitter.addListener(event, listener)

eventEmitter.on(event, listener)

eventEmitter.on(event, listener) and eventEmitter.addListener(event, listener) are pretty much similar. It adds the listener at the end of the listener’s array for the specified event. Multiple calls to the same event and listener will add the listener multiple times and correspondingly fire multiple times. Both functions return emitter, so calls can be chained.

**USING ANGULAR**

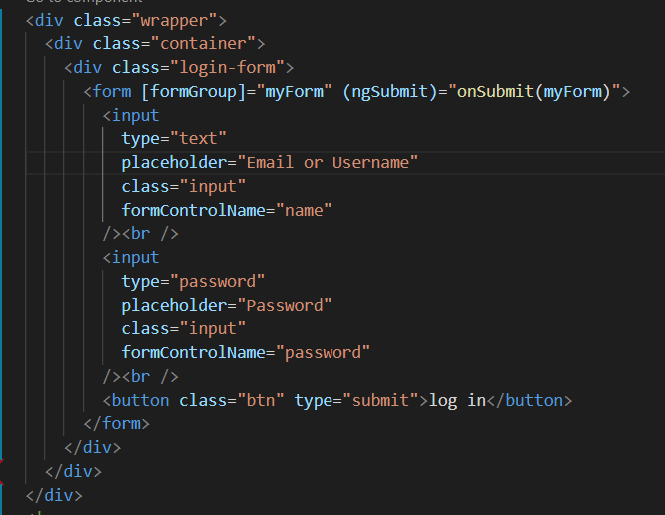
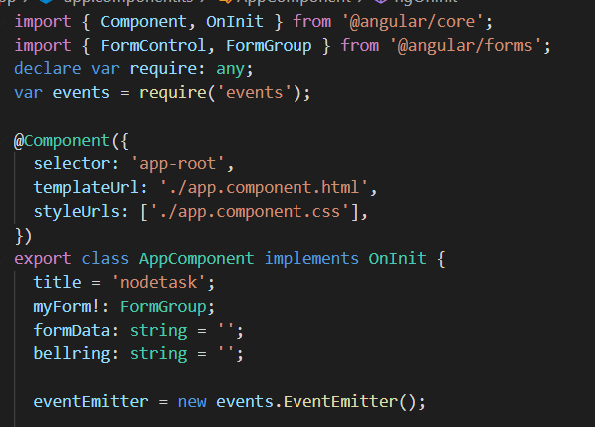
**addListener(event, listener) :** Adds a listener at the end of the listeners array for the specified event. No checks are made to see if the listener has already been added. Multiple calls passing the same combination of event and listener will result in the listener being added multiple times. Returns emitter, so calls can be chained.

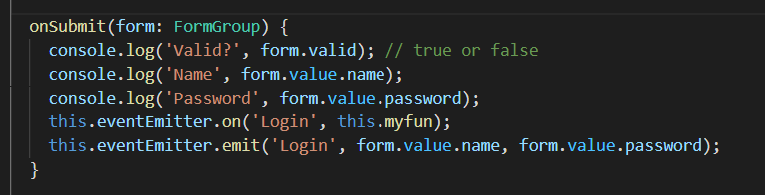
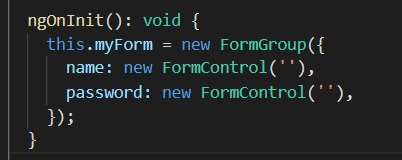
1:importing events

2:creating one emmiter object em (eventmitter is one class it is present in evnts module used to create and handle custom events)

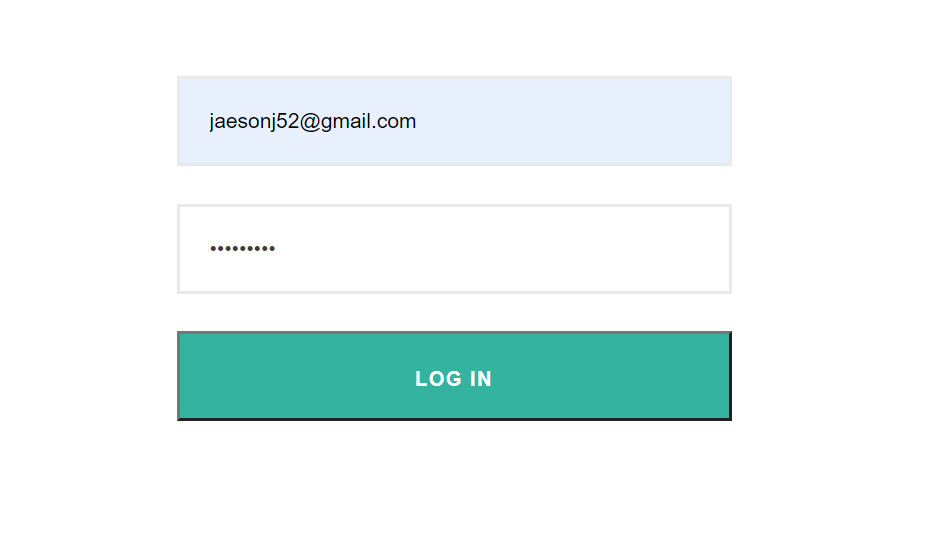
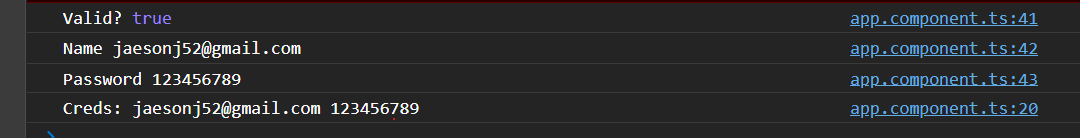
3:on method is needs event name ,it can handle and the callback function to call when the event is raised.

4:emit function (it will invoke the event), the first parameter is the event name as a string and remaining are arguments.

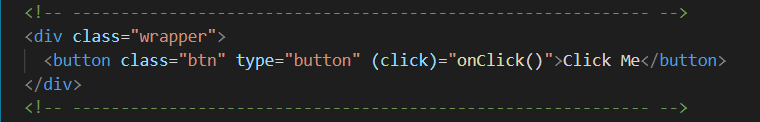
 

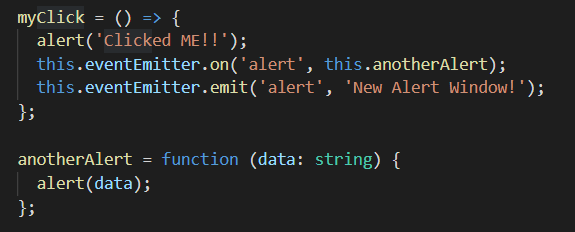
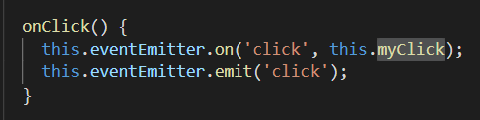
 

**OUTPUT:**

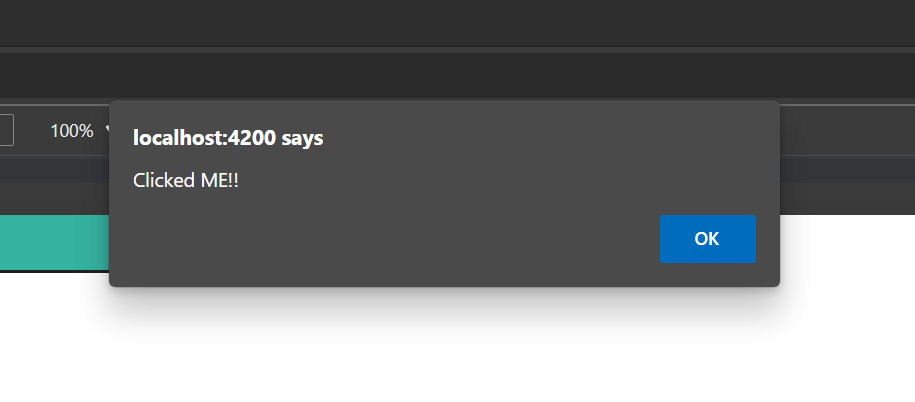
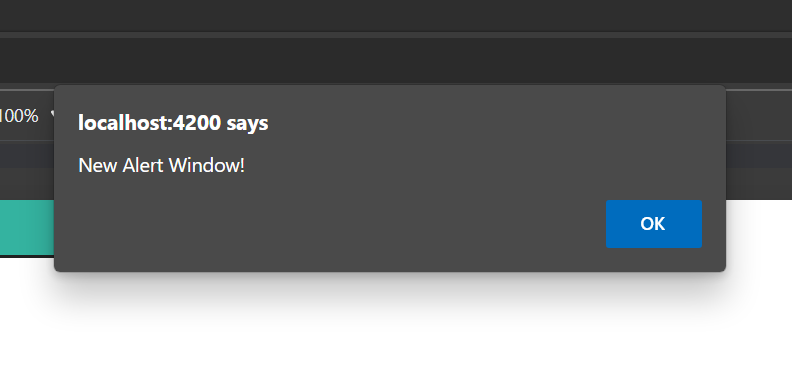
**Chain of Events and Data Passing:**

****

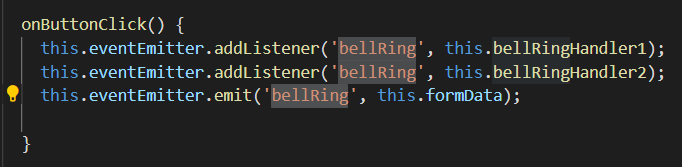
**OUTPUT:**

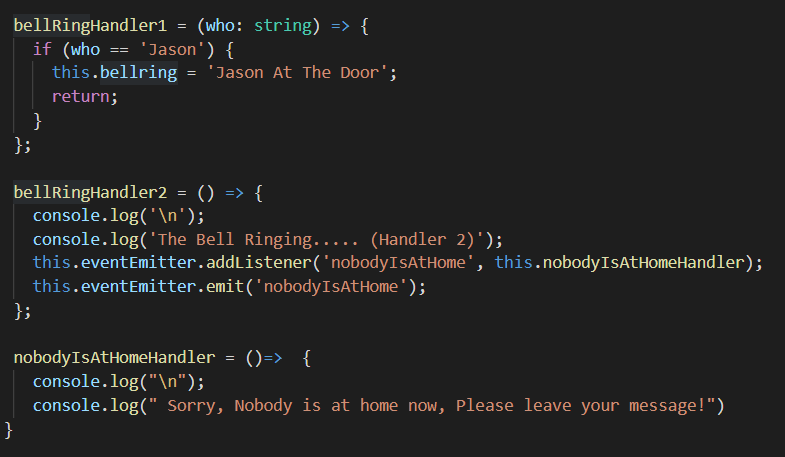


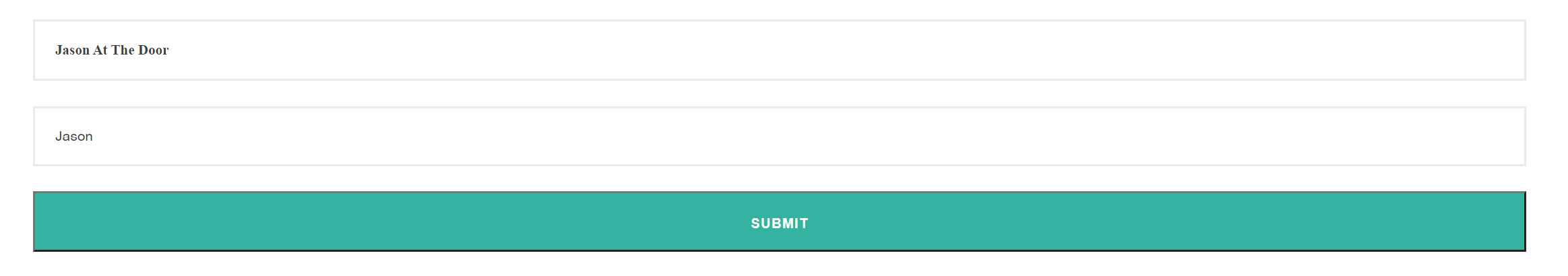
**Multiple Event Listeners:**

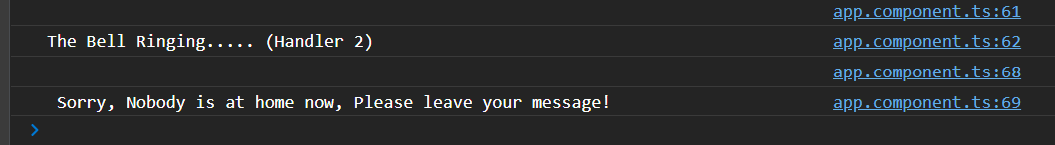






**OUTPUT:**



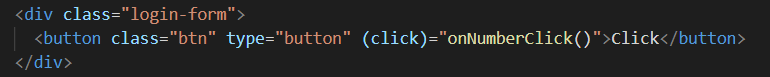


**on(event, listener):**

Adds a listener at the end of the listeners array for the specified event. No checks are made to see if the listener has already been added. Multiple calls passing the same combination of event and listener will result in the listener being added multiple times. Returns emitter, so calls can be chained.

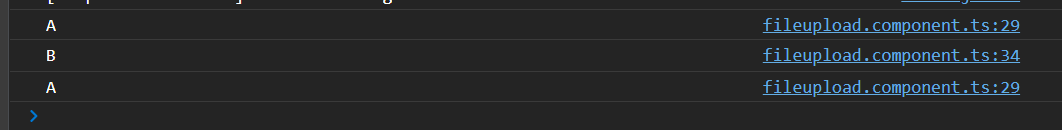
**removeListner(event,listener):**

Removes a listener from the listener array for the specified event. **Caution −** It changes the array indices in the listener array behind the listener. removeListener will remove, at most, one instance of a listener from the listener array. If any single listener has been added multiple times to the listener array for the specified event, then removeListener must be called multiple times to remove each instance. Returns emitter, so calls can be chained.





**OUTPUT:**

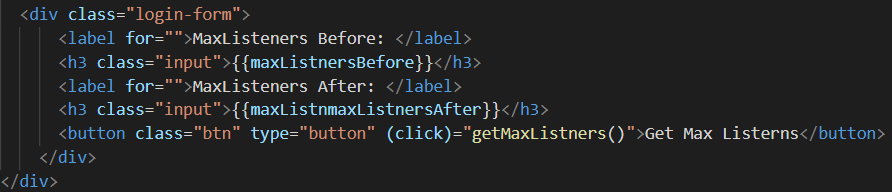


**once(event, listener):**

Adds a one time listener to the event. This listener is invoked only the next time the event is fired, after which it is removed. Returns emitter, so calls can be chained.

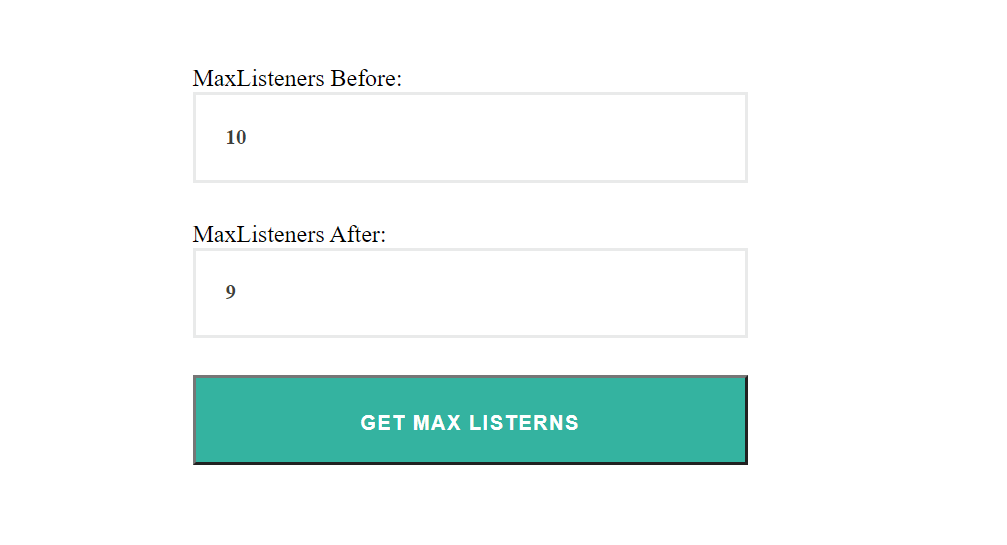
**getMaxListeners():**

By default, EventEmitters will print a warning if more than 10 listeners are added for a particular event. This is a useful default which helps finding memory leaks. Obviously not all Emitters should be limited to 10. This function allows that to be increased. Set to zero for unlimited.





**OUTPUT:**





**setMaxListeners(n):**

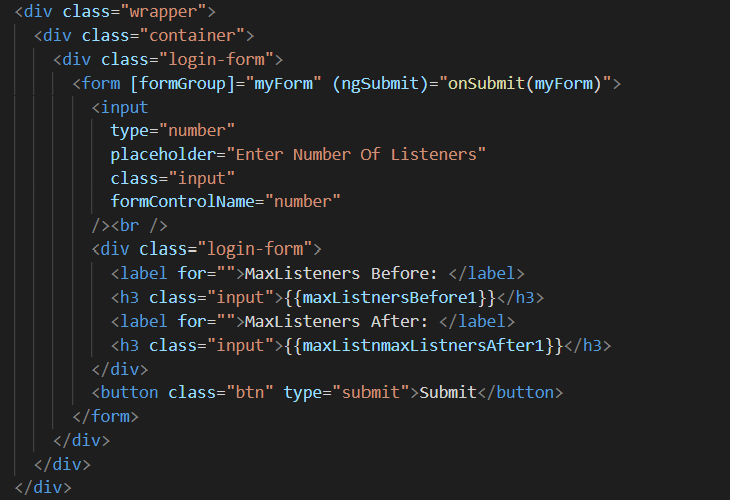
By default, EventEmitters will print a warning if more than 10 listeners are added for a particular event. This is a useful default which helps finding memory leaks. Obviously not all Emitters should be limited to 10. This function allows that to be increased. Set to zero for unlimited.

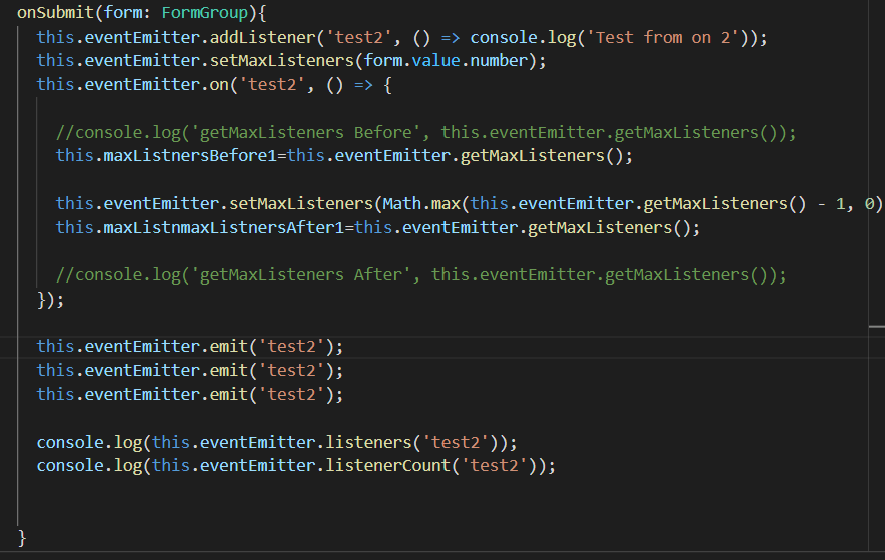
**listeners():**

Returns an array of listeners for the specified event.

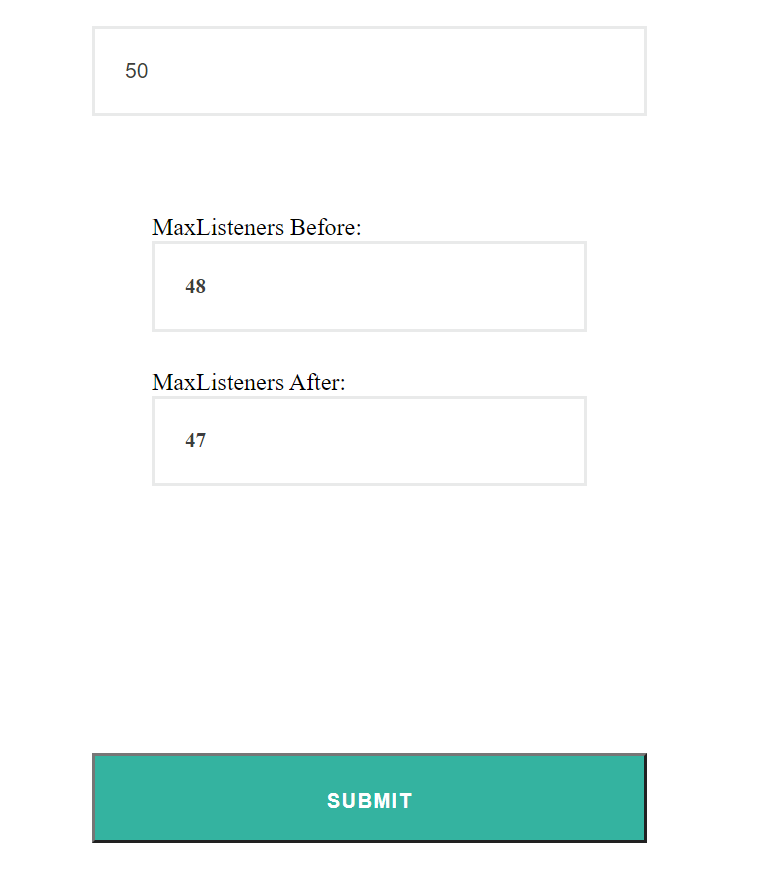
**listenerCount(event):**

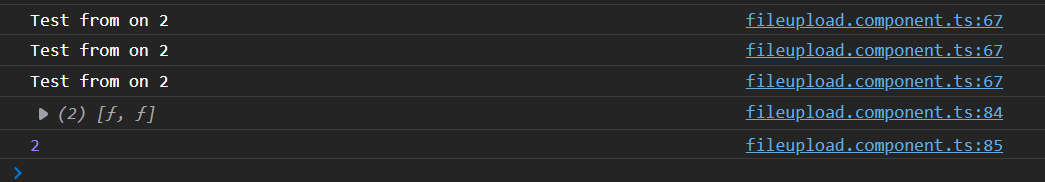
Returns the number of listeners for a given event.





**OUTPUT:**





**USING JAVASCRIPT:**

