Node.js `EventEmitter`

Events: an overview

- Events are information (strings, objects, etc.) that has a name or label with no particular destination
- Senders of this information are called "emitters"
- Receivers of this information are called "listeners"

When are events used?

- File I/O - `open`, `close`
- UI actions - `click`, `scroll`, `keydown`
- Node HTTP servers - `connect`, `continue`
- Uncaught exceptions - `uncaughtException`

Using events: `.on()`

- Lets you listen for an event:
  ```js
  //pull in the native EventEmitter library
  var EventEmitter = require('events');

//create a new emitter
var myEmitter = new EventEmitter();

//on `my-event`, print the incoming message
myEmitter.on('my-event', function(message) {
 console.log(message);
  ```

Using events: `.emit()`

- `.emit()` triggers an event
- May be handled by multiple listeners where a single function call would only be handled by one function
- Events are handled asynchronously

~~~
//when the `my-event` handler is called from the
//previous example, this will print "hello world!"
myEmitter.emit('my-event', 'hello world!');
``
---

# Using events: `.removeListener()`
- Stops listening to an event - we may want to do this when we want to change listeners or when the event is no longer important
- Supplying a reference to the listening function is mandatory
```js
var EventEmitter = require('events');
//let's track telephone rings and pick up on the second
var telephone = new EventEmitter();

var rings = 0;
//let the listener function be declared as a variable
var listener = function() {
    //increment the number of rings
    rings++;
    //on the second ring, we pick up
    if(rings == 2){
        //and remove the listener
        telephone.removeListener('phone-ring', listener);
    }
};
myEmitter.on('phone-ring', listener);
~~~

Reminder: Inheritance is useful

- Any class can inherit from `EventEmitter` to become an event emitter itself!

```js
var EventEmitter = require('events');

//declare a Cat type
function Cat() {
 //keep a reference to this for use in other scopes
 var self = this;
 //declare a method to speak that emits a message
 self.speak = function(){
```