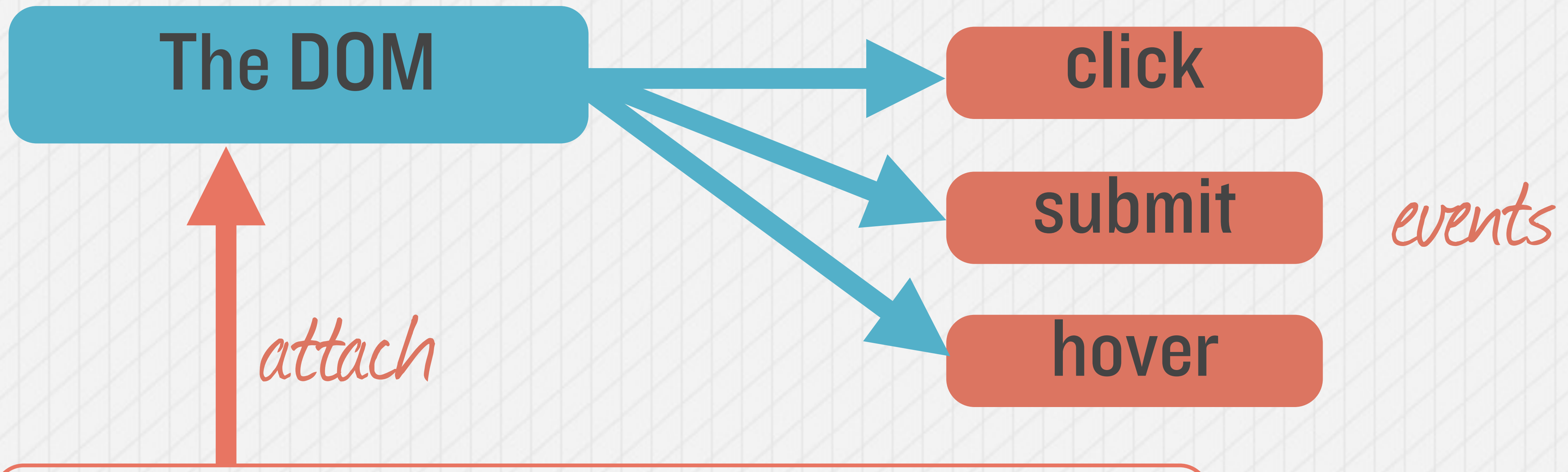
EVENTS

- LEVEL TWO -

EVENTS IN THE DOM

The DOM triggers Events

you can listen for those events

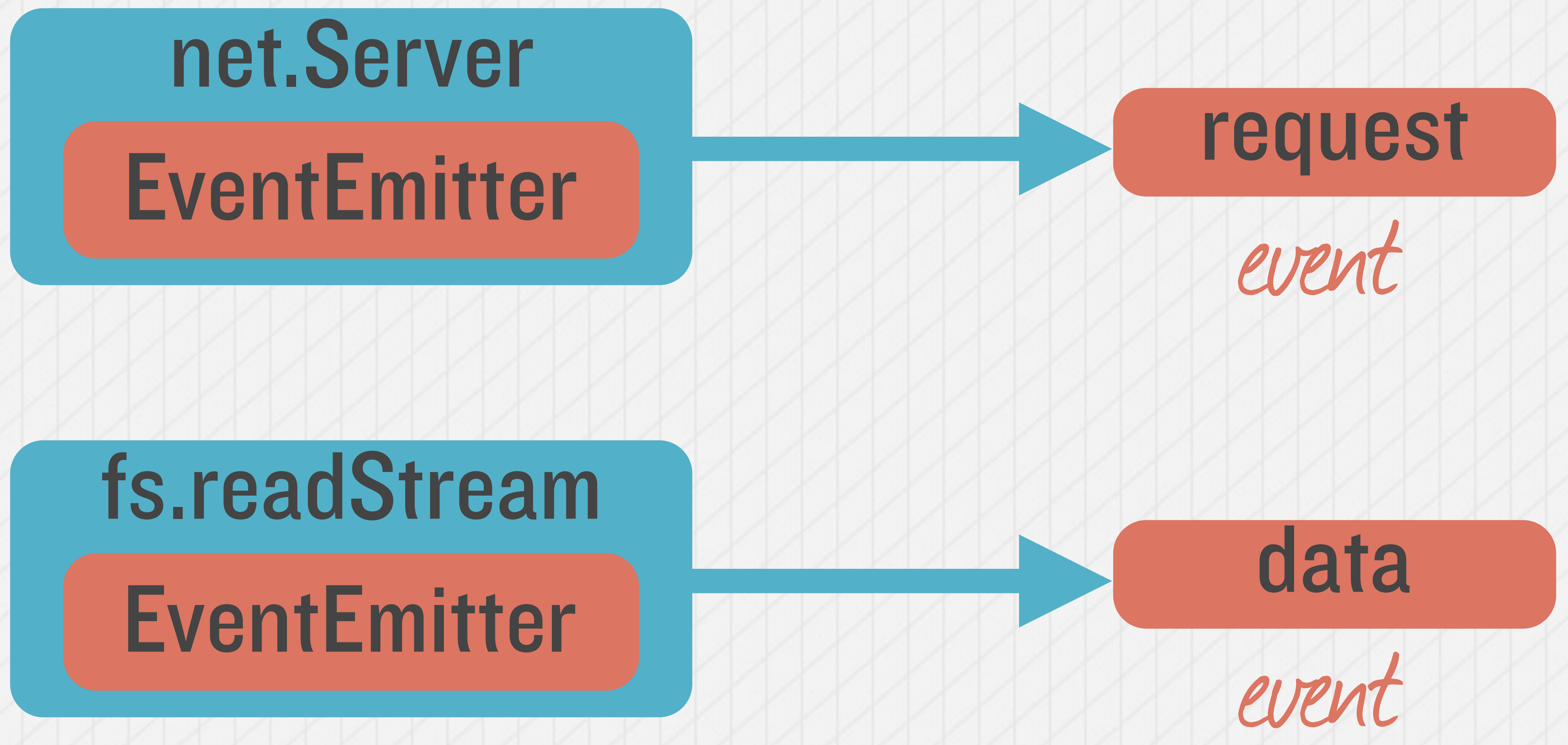


```
$("p").on("click", function(){ ... });
```

When 'click' event is triggered

EVENTS IN NODE

Many objects in Node emit events



CUSTOM EVENT EMITTERS

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

events

error warn info

```
var logger = new EventEmitter();
```

```
logger.on('error', function(message){
  console.log('ERR: ' + message);
});
```

listen for error event

```
logger.emit('error', 'Spilled Milk');
```

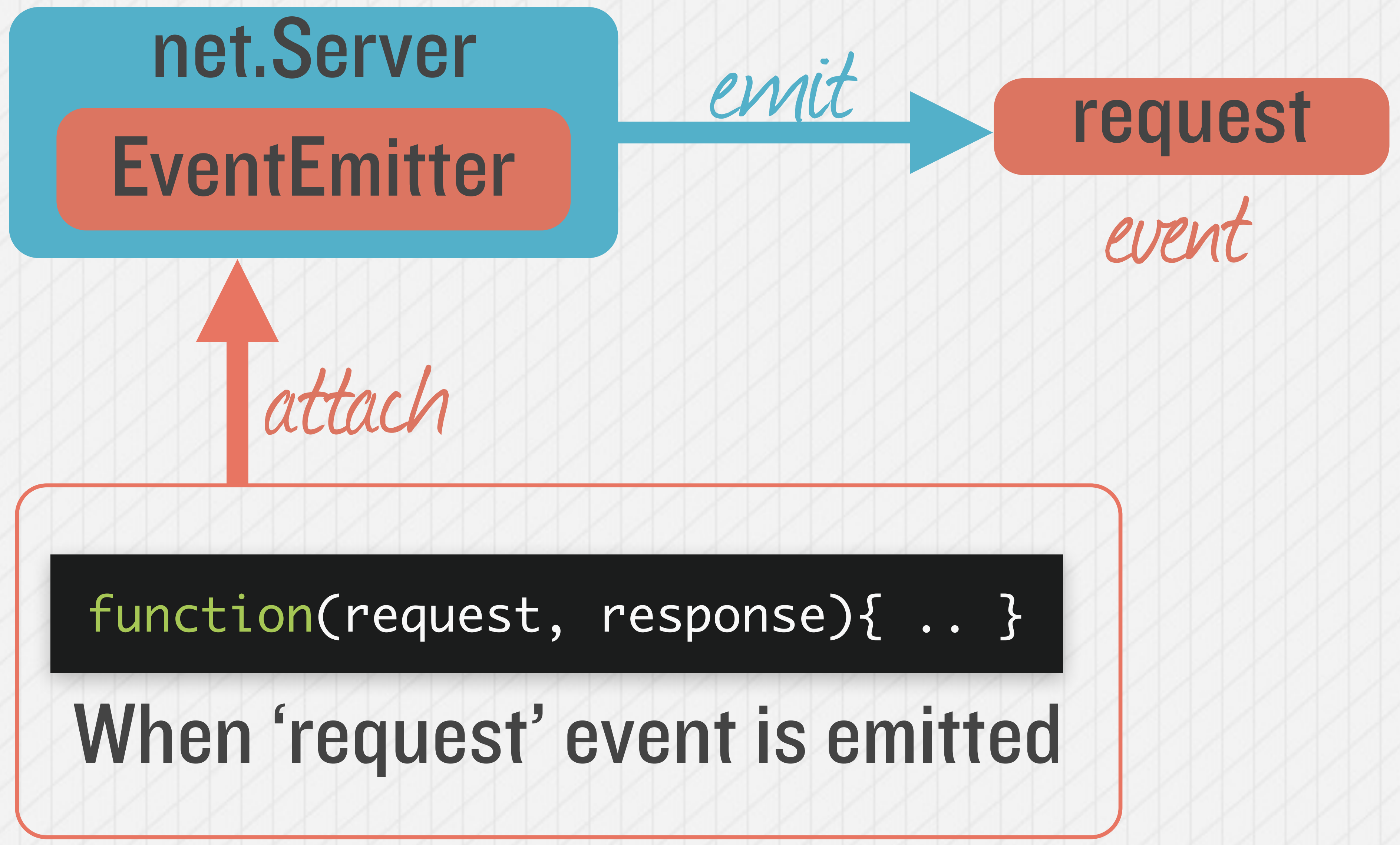
```
--> ERR: Spilled Milk
```

```
logger.emit('error', 'Eggs Cracked');
```

```
--> ERR: Eggs Cracked
```

EVENTS IN NODE

Many objects in Node emit events



```
function(request, response){ .. }
```

When ‘request’ event is emitted

HTTP ECHO SERVER

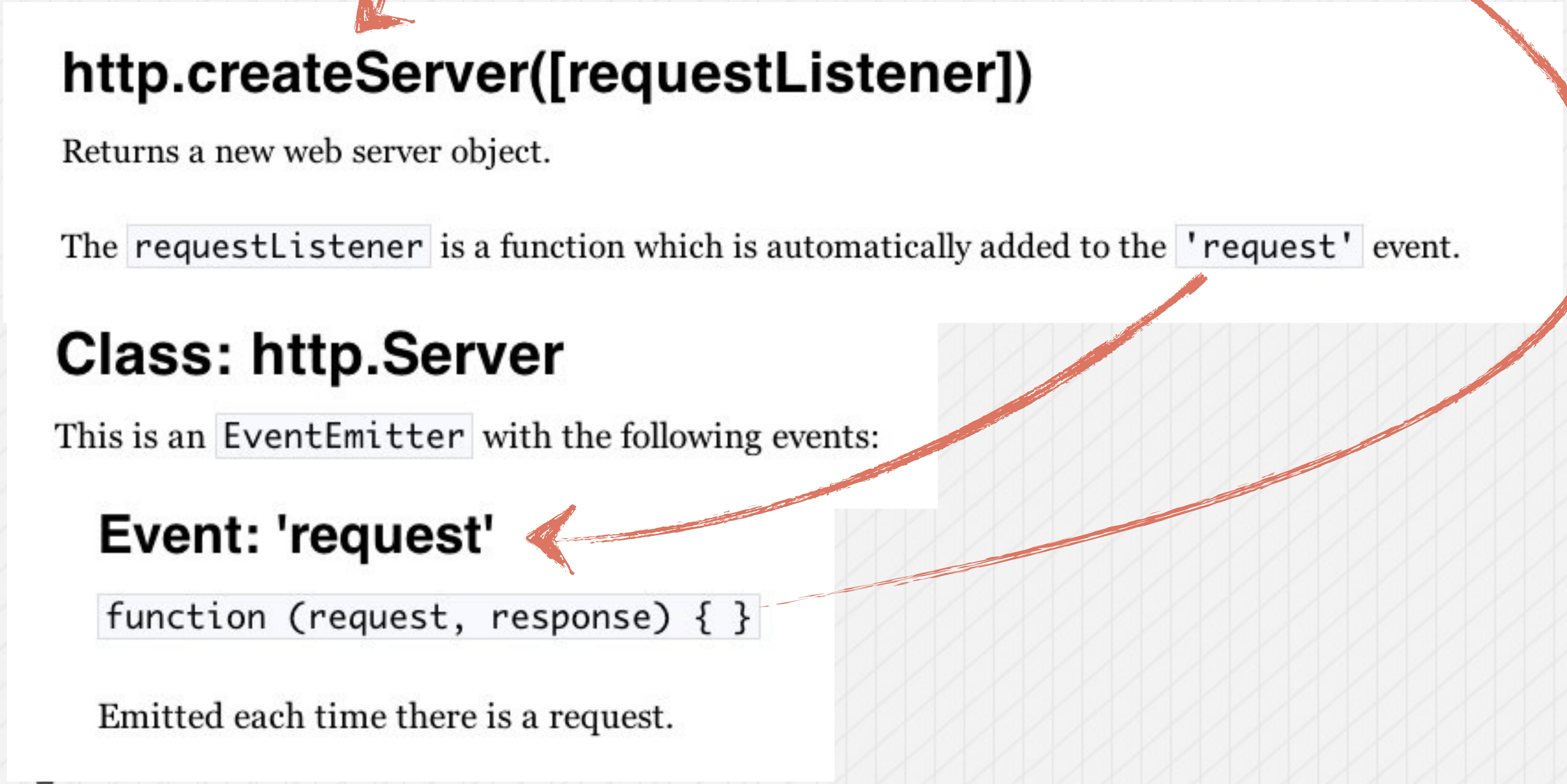
```
http.createServer(function(request, response){ ... });
```

But what is really going on here?

http://nodejs.org/api/

BREAKING IT DOWN

```
http.createServer(function(request, response){ ... });
```



http.createServer([requestListener])

Returns a new web server object.

The `requestListener` is a function which is automatically added to the `'request'` event.

Class: http.Server

This is an `EventEmitter` with the following events:

Event: 'request'

`function (request, response) { }`

Emitted each time there is a request.

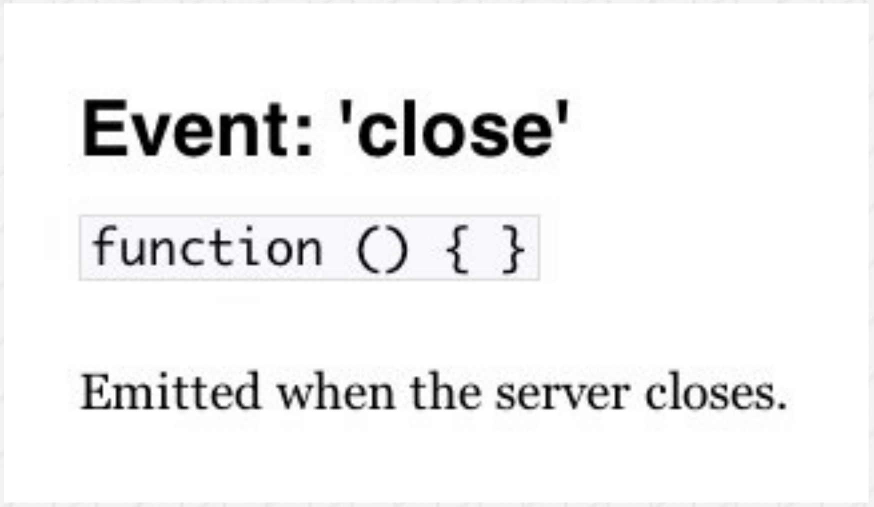
ALTERNATE SYNTAX

```
http.createServer(function(request, response){ ... });
```

Same as

```
var server = http.createServer();
server.on('request', function(request, response){ ... });
```

This is how we add event listeners



Event: 'close'

`function () { }`

Emitted when the server closes.

```
server.on('close', function(){ ... });
```