NODE.JS

CS498RK SUMMER 2017

UNIVERSITY OF ILLINOIS @ URBANA-CHAMPAIGN

ASYNC REVIEW

CODEPEN

Any questions on MP3?

What is Node.js?

NODE.JS

JavaScript runtime built on Chrome's V8 JavaScript engine

Uses an event-driven, non-blocking I/O model

HISTORY

Created by Ryan Dahl in 2009

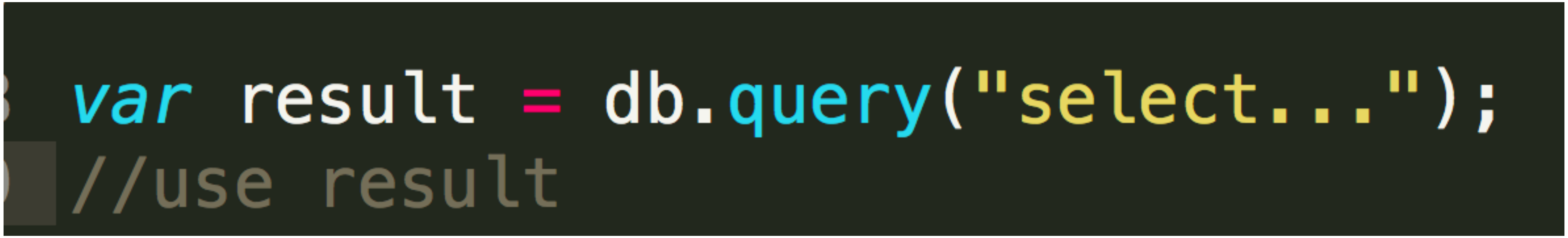
Written in C, C++, and Javascript

HISTORY

Dahl was inspired by an upload progress bar on Flickr (which could take several minutes)

Blocking

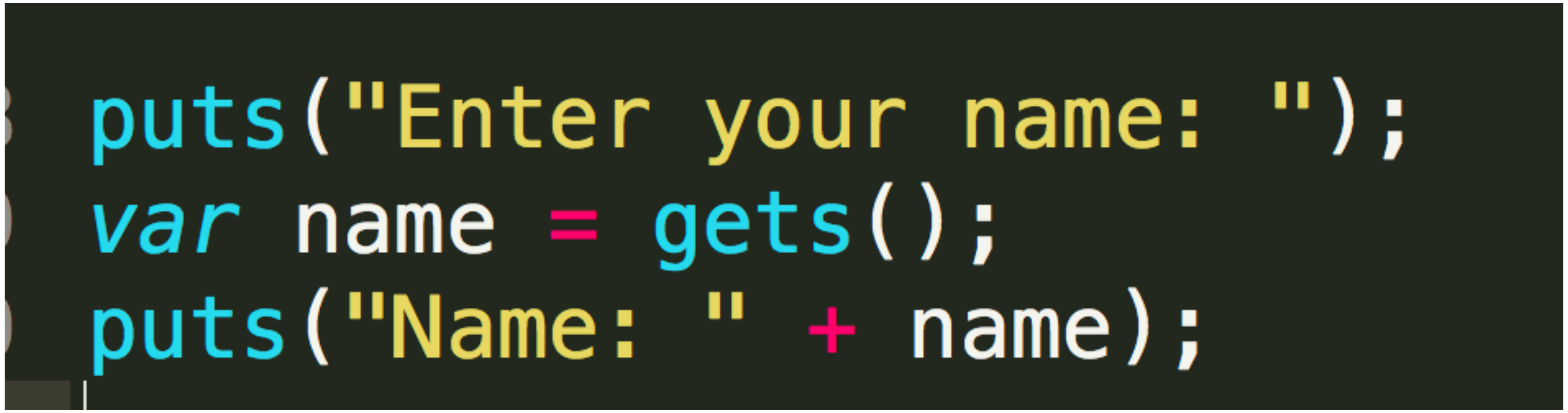
DATABASE CALL EXAMPLE



```
var result = db.query("select...");
//use result
```

REQUEST-RESPONSE

People are used to sending a request and getting a response immediately



```
puts("Enter your name: ");
var name = gets();
puts("Name: " + name);
```

APACHE WEB SERVERS

Dahl disliked limitations of Apache web servers

Used blocking

Multiple threads for concurrent connections (Lots of memory)

HISTORY

Instead, Dahl wanted a single thread with non-blocking I/O

I/O should be “give a callback and move on”

NODE.JS

Node is asynchronous and non-blocking

Single-threaded

Perfect for real-time applications (web apps?)

NODE.JS

Highly Scalable

No buffering, output data in chunks

Very fast

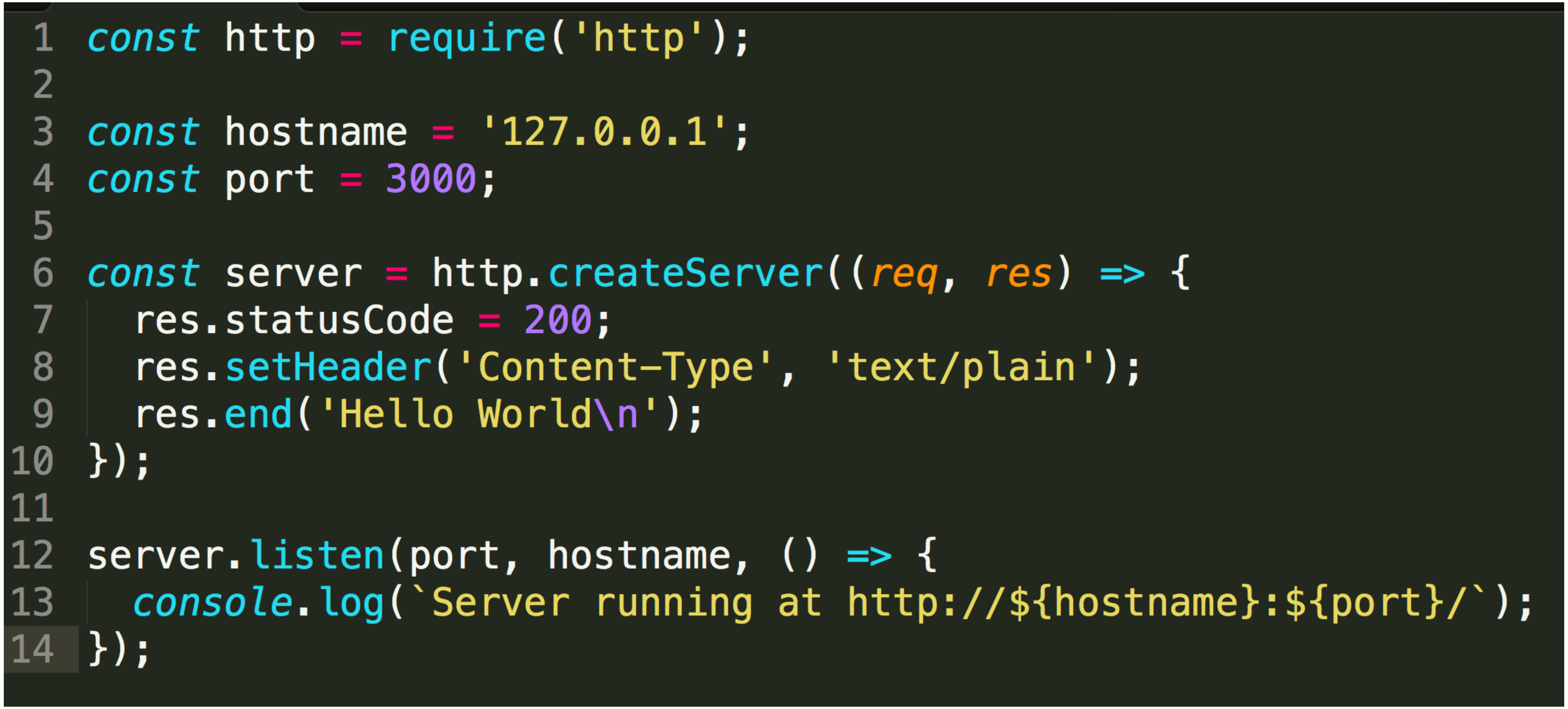
V8

Google's open source high-performance JavaScript engine

Compiles and executes source code and handles memory allocation

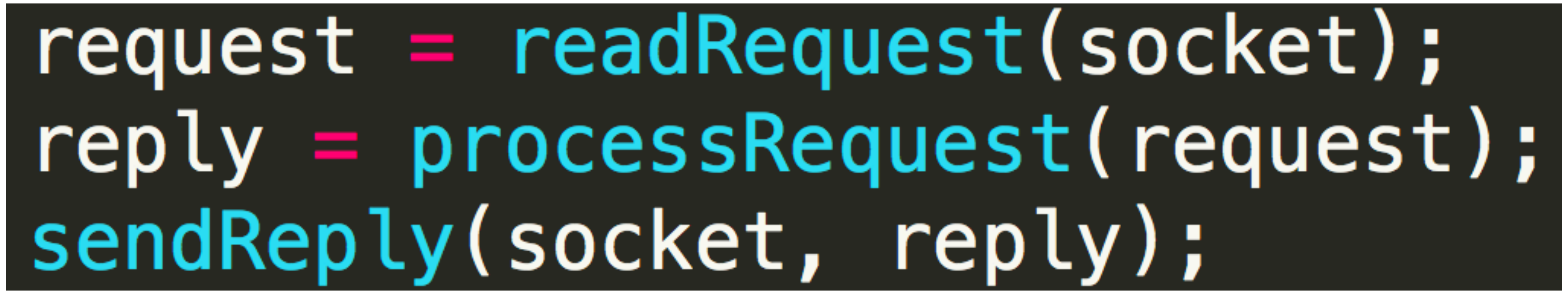
Garbage collects unnecessary objects (makes V8 really fast)

HELLO WORLD EXAMPLE



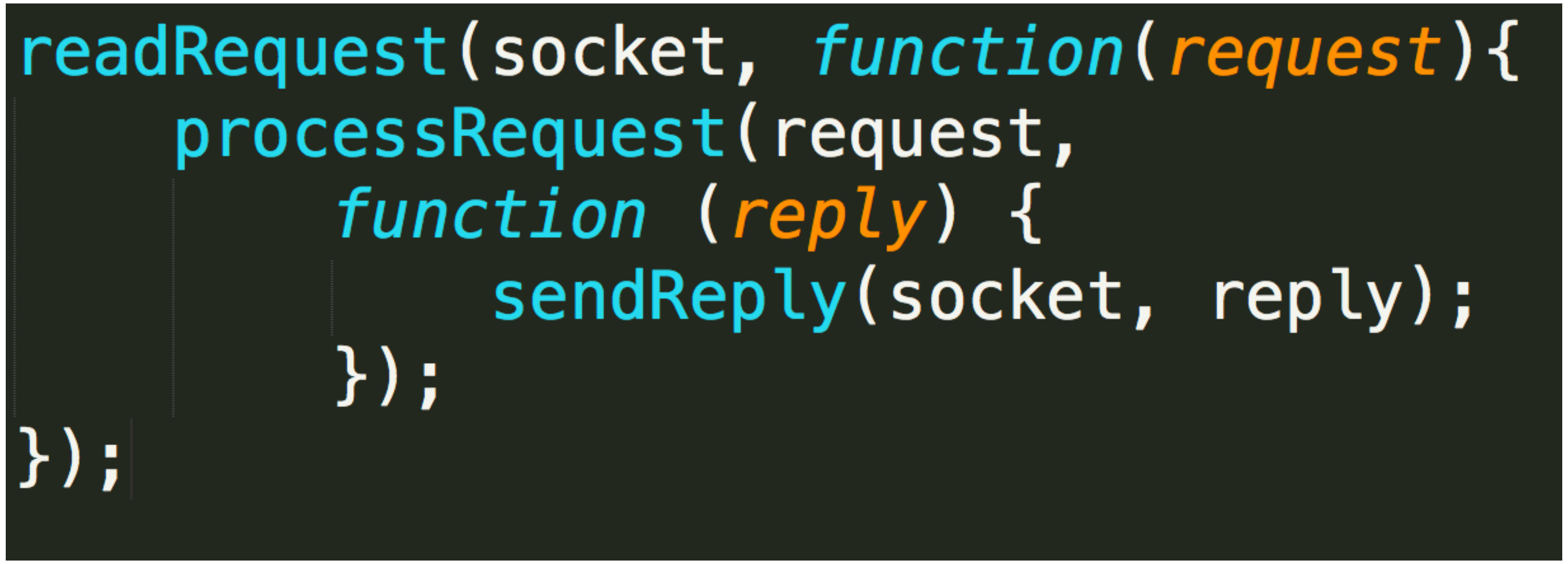
```javascript
const http = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

THREADS VS EVENTS



```
request = readRequest(socket);
reply = processRequest(request);
sendReply(socket, reply);
```

Thread switching (blocking) and scheduler

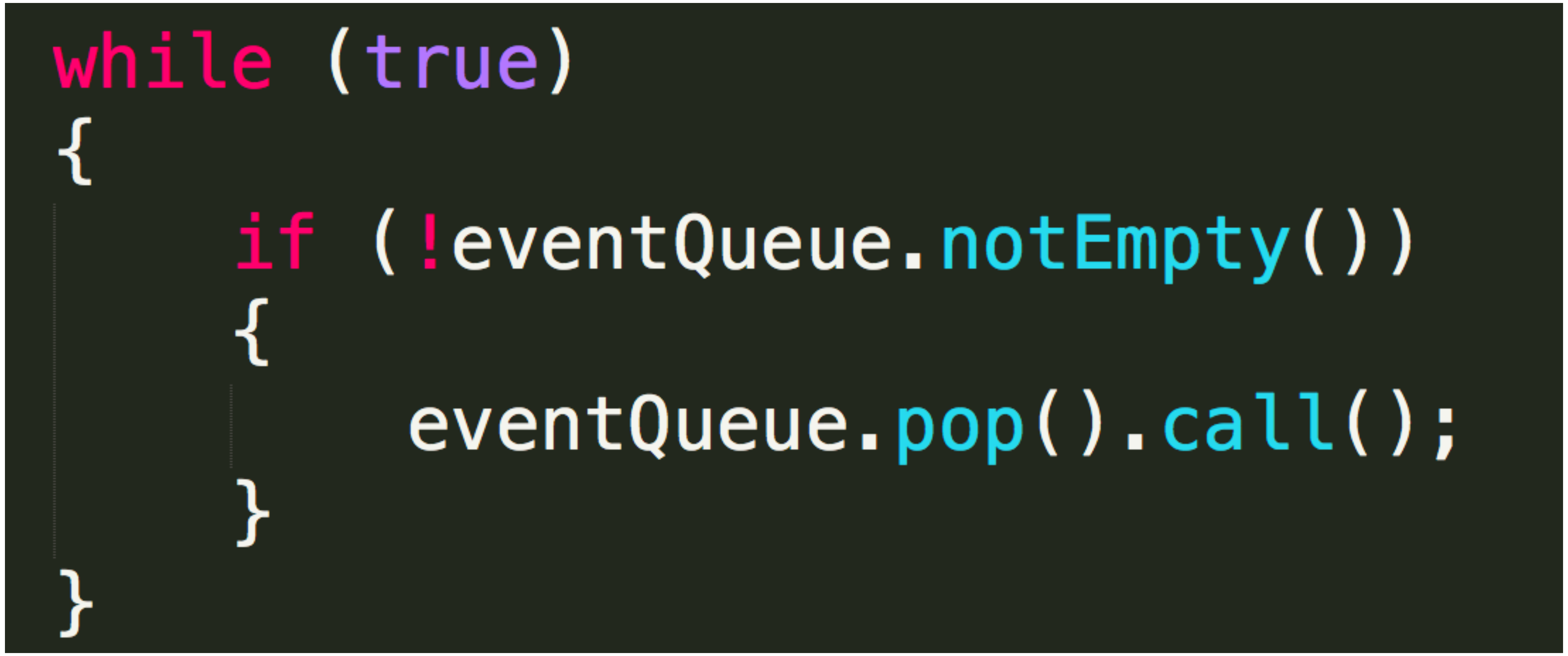


```
readRequest(socket, function(request){
    processRequest(request,
        function (reply) {
            sendReply(socket, reply);
        });
});
```

Event queue processing

EVENT QUEUE

Imagine the inner loop:



```
while (true)
{
    if (!eventQueue.notEmpty())
    {
        eventQueue.pop().call();
    }
}
```

NODE PACKAGE MANAGER (NPM)

Package manager for JavaScript (default for Node)

Used to install and share code

Manage dependencies in your projects

NPM DEMO

Terminal:

```
npm init
npm install —save <pkgname>
```

Script:

```
require(‘<pkgname’)
```

REQUIRE

Import using require():

System module: require(“f”);

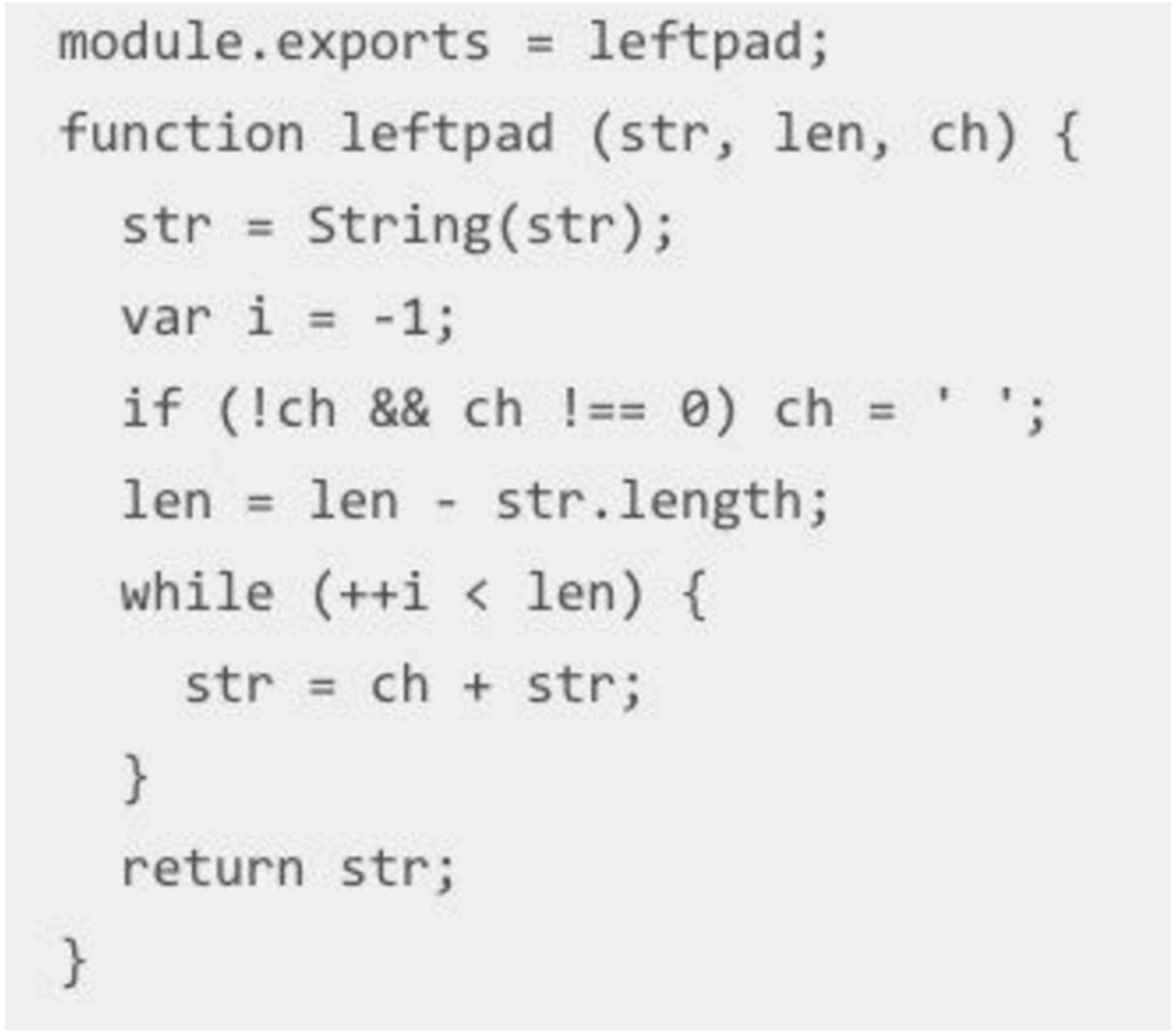
File: require(“./f”);

COMMUNITY SUPPORT

NPM has 485448 modules (as of July 17)

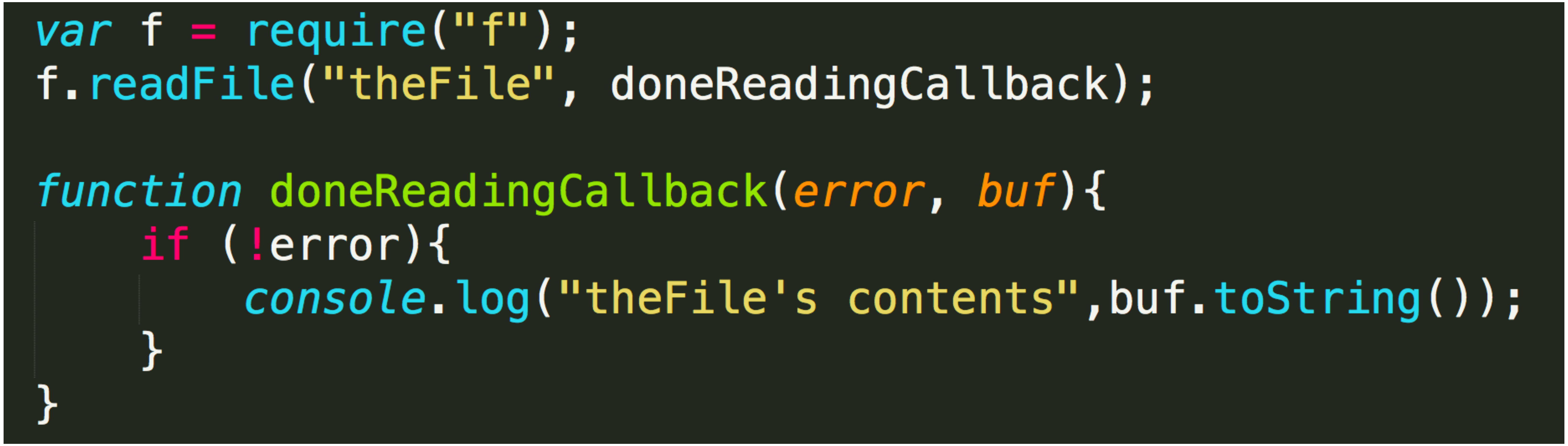
Azer Koçulu broke the internet by deleting a package called left-pad

COMMUNITY SUPPORT



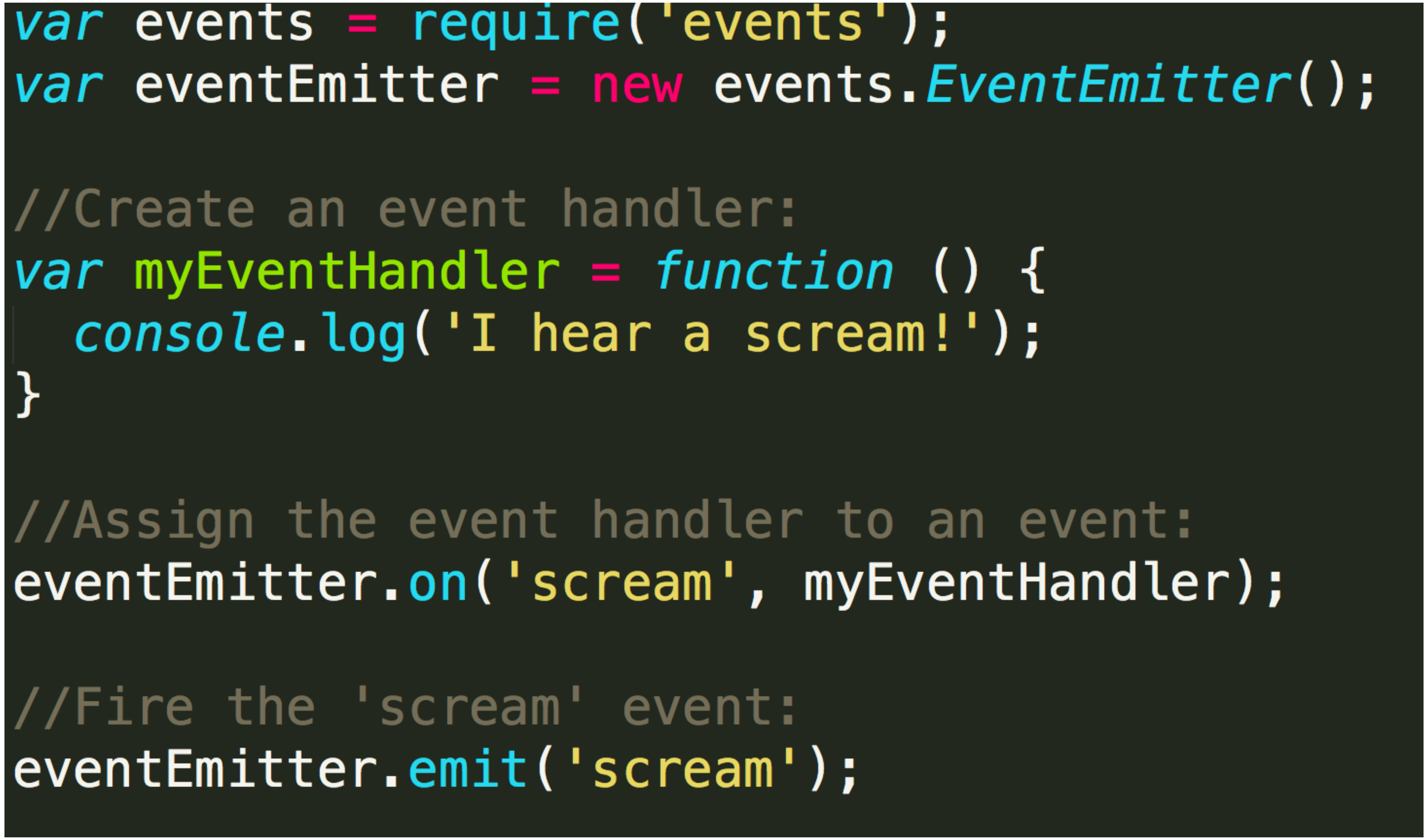
```
module.exports = leftpad;
function leftpad (str, len, ch) {
  str = String(str);
  var i = -1;
  if (!ch && ch !== 0) ch = ' ';
  len = len - str.length;
  while (++i < len) {
    str = ch + str;
  }
  return str;
}
```

READING A FILE



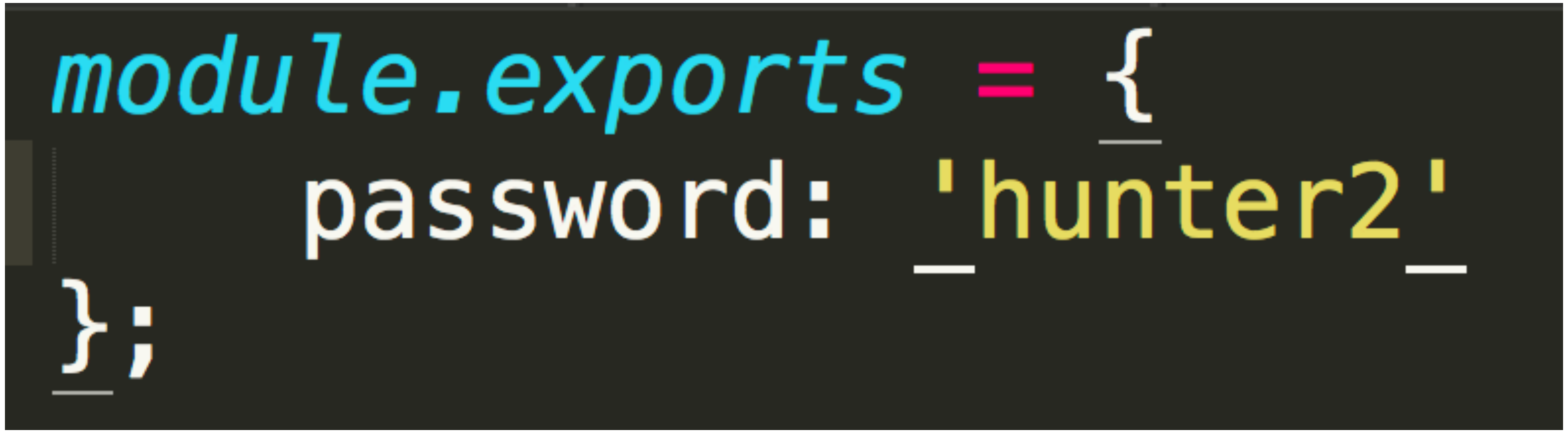
```
var f = require("f");
f.readFile("theFile", doneReadingCallback);

function doneReadingCallback(error, buf){
  if (!error){
    console.log("theFile's contents",buf.toString());
  }
}
```

EVENTS



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

//Create an event handler:
var myEventHandler = function () {
  console.log('I hear a scream!');
}

//Assign the event handler to an event:
eventEmitter.on('scream', myEventHandler);

//Fire the 'scream' event:
eventEmitter.emit('scream');
```

MODULE.EXPORTS EXAMPLE



```
module.exports = {
    password: 'hunter2'
};
```

TCP CHAT SERVER EXAMPLE

Debugging

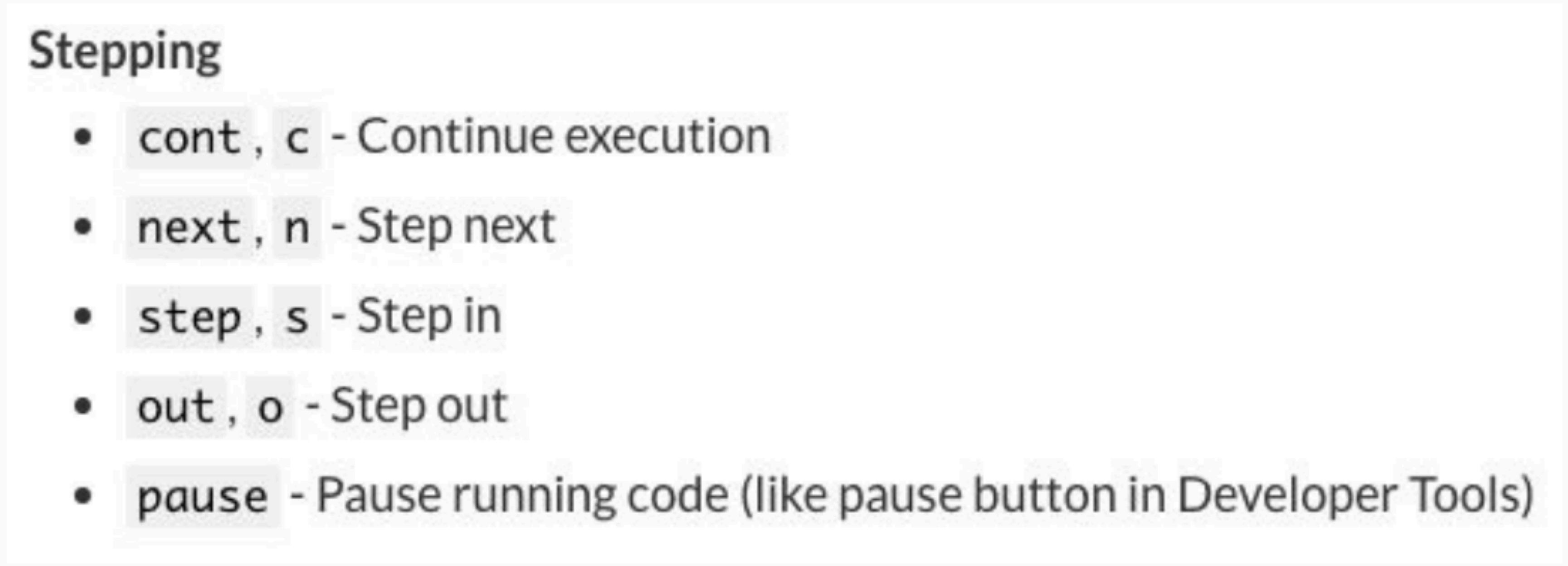
`node debug <filename>`

Set breakpoints by adding

`debugger;`

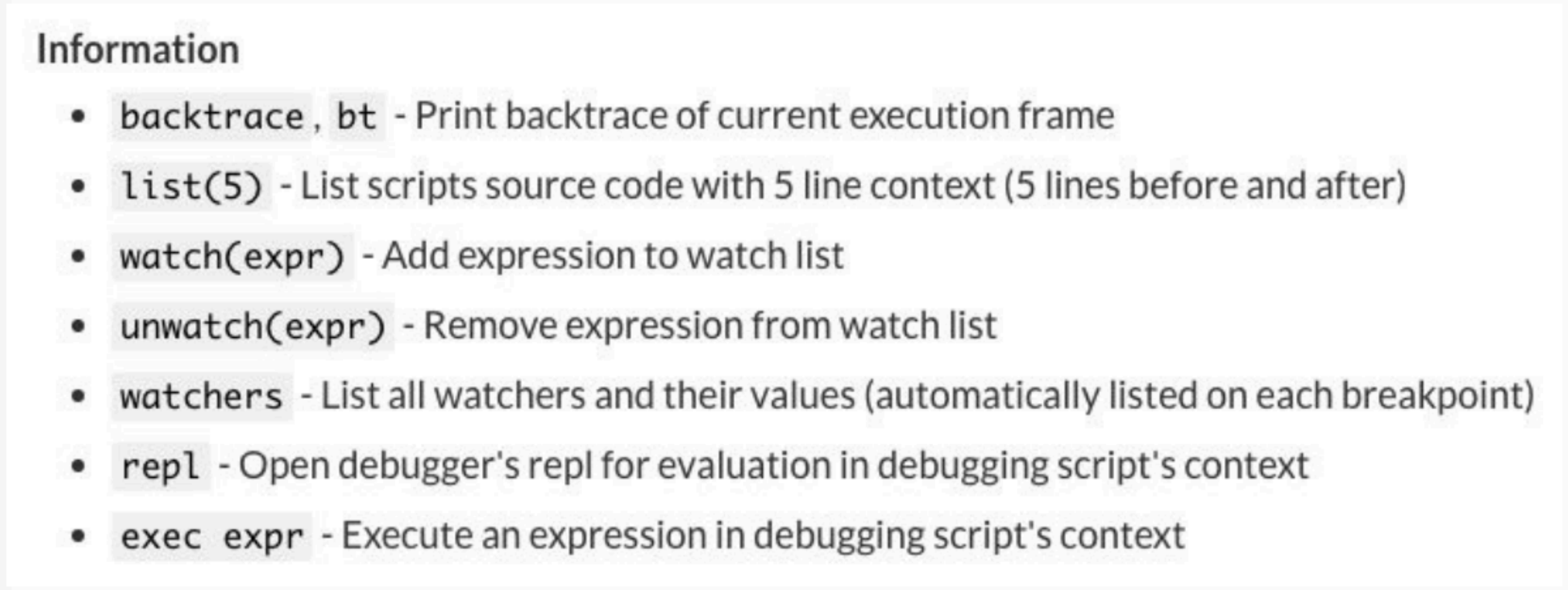
in your script

Also try: node-inspector, WebStorm debugger



Stepping

- `cont`, `c` - Continue execution
- `next`, `n` - Step next
- `step`, `s` - Step in
- `out`, `o` - Step out
- `pause` - Pause running code (like pause button in Developer Tools)



Information

- `backtrace`, `bt` - Print backtrace of current execution frame
- `list(5)` - List scripts source code with 5 line context (5 lines before and after)
- `watch(expr)` - Add expression to watch list
- `unwatch(expr)` - Remove expression from watch list
- `watchers` - List all watchers and their values (automatically listed on each breakpoint)
- `repl` - Open debugger's repl for evaluation in debugging script's context
- `exec expr` - Execute an expression in debugging script's context

express mongoose

EXPRESS AND MONGOOSE

Node modules

Express - Rapidly develop Node apps

Mongoose - Elegant MongoDB object modeling for Node

REFERENCES

https://www.youtube.com/watch?v=ztspvPYybIY

https://web.stanford.edu/class/cs142/lectures/NodeJS.pdf

http://nodeguide.com/

https://www.w3schools.com/nodejs/nodejs_events.asp

https://nodejs.org/api/events.html

https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code/