State of Diagnostic Tools

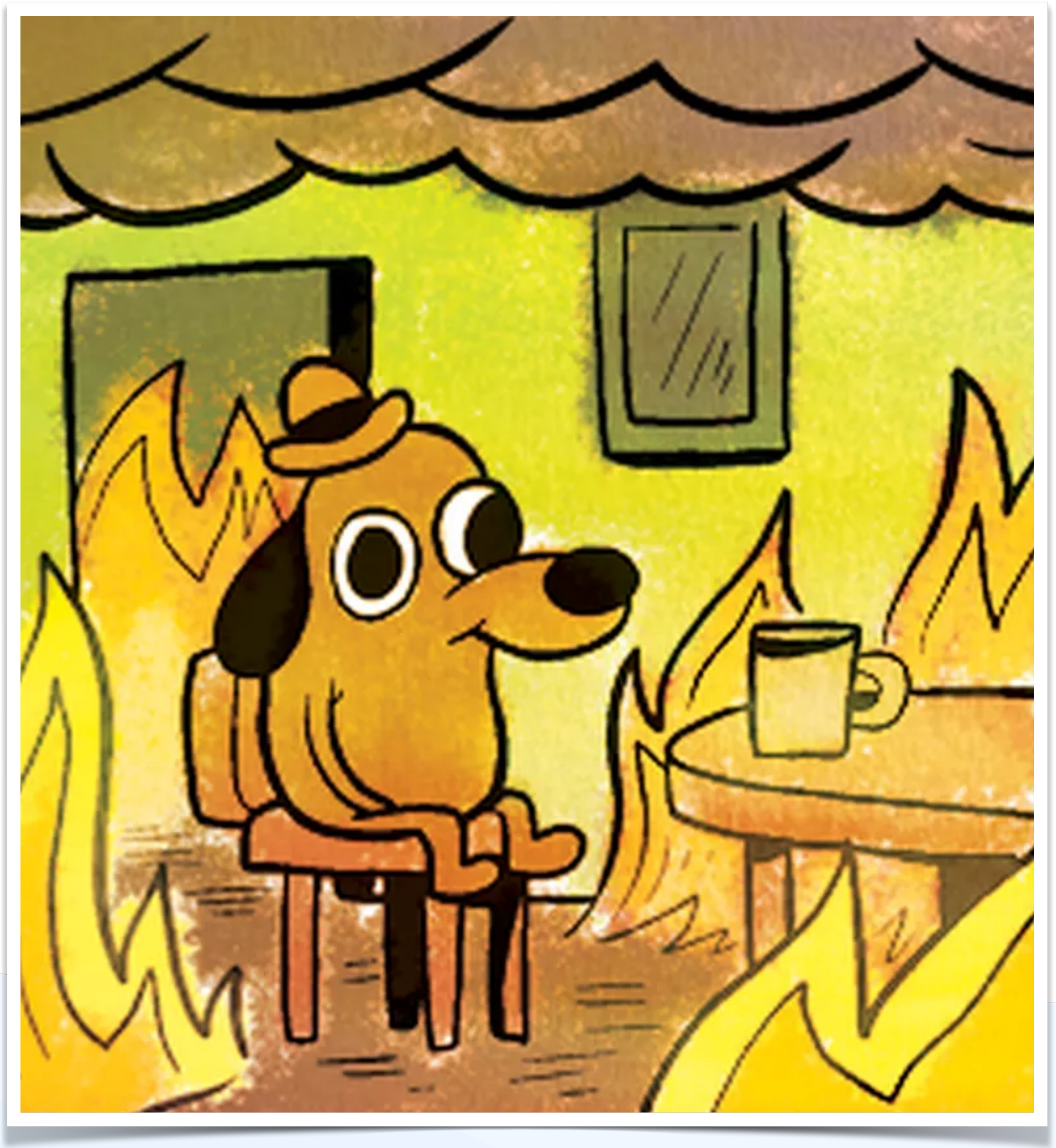
for production environment in the Node.js ecosystem

Mary Marchini
Lead Software Engineer @ **Sthima**
Node.js Core Collaborator

@mmarkini
oss@mmarchini.me
mmarchini.me
mmarchini

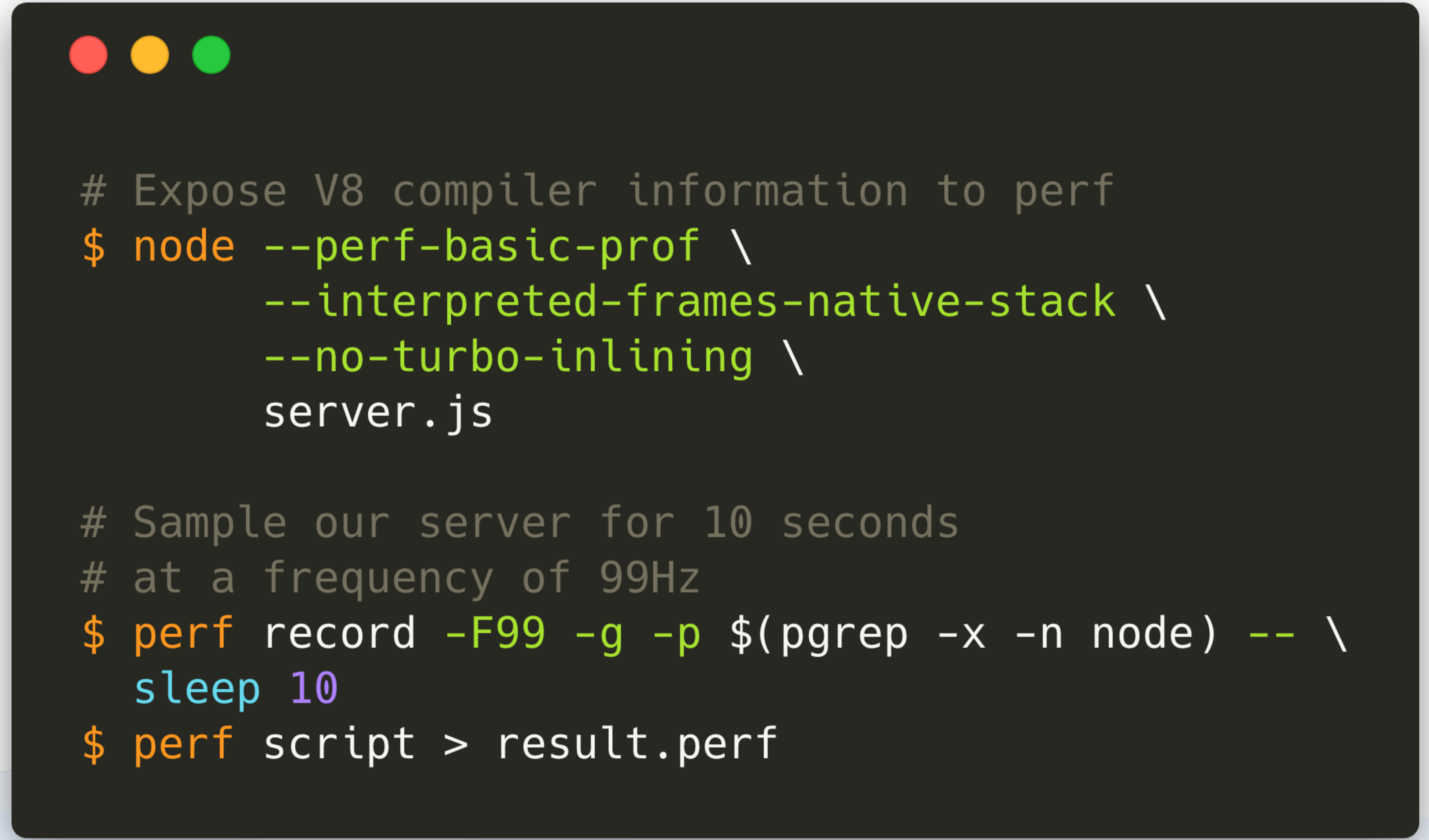
Production should look like this

But sometimes it feels like this



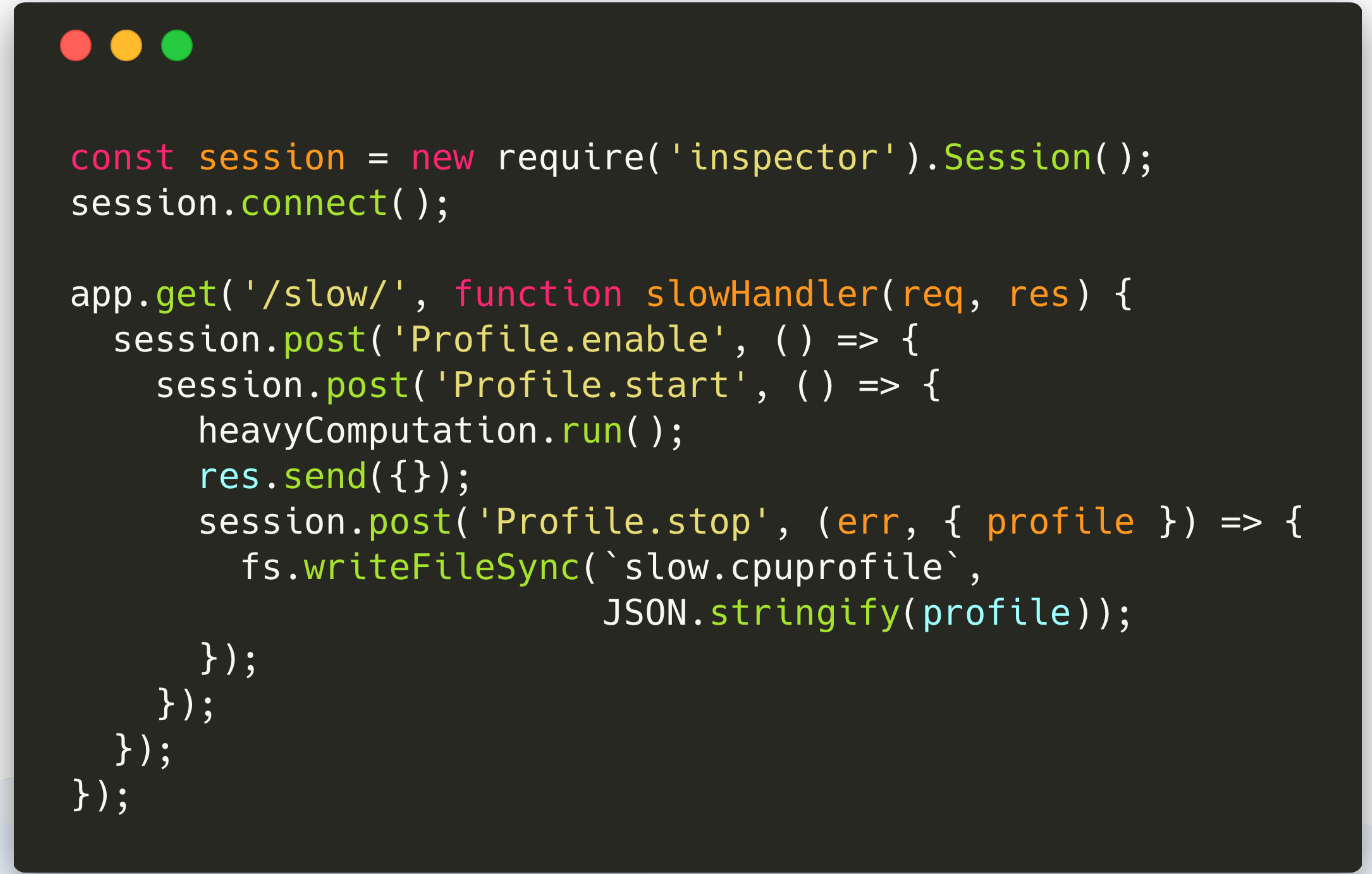
- Linux *perf*
- V8 CpuProfiler
- V8 SamplingHeapProfiler
- llnode
- node-report
- Structured Logs

- Linux *perf*
- V8 CpuProfiler
- V8 SamplingHeapProfiler
- llnode
- node-report
- Structured Logs



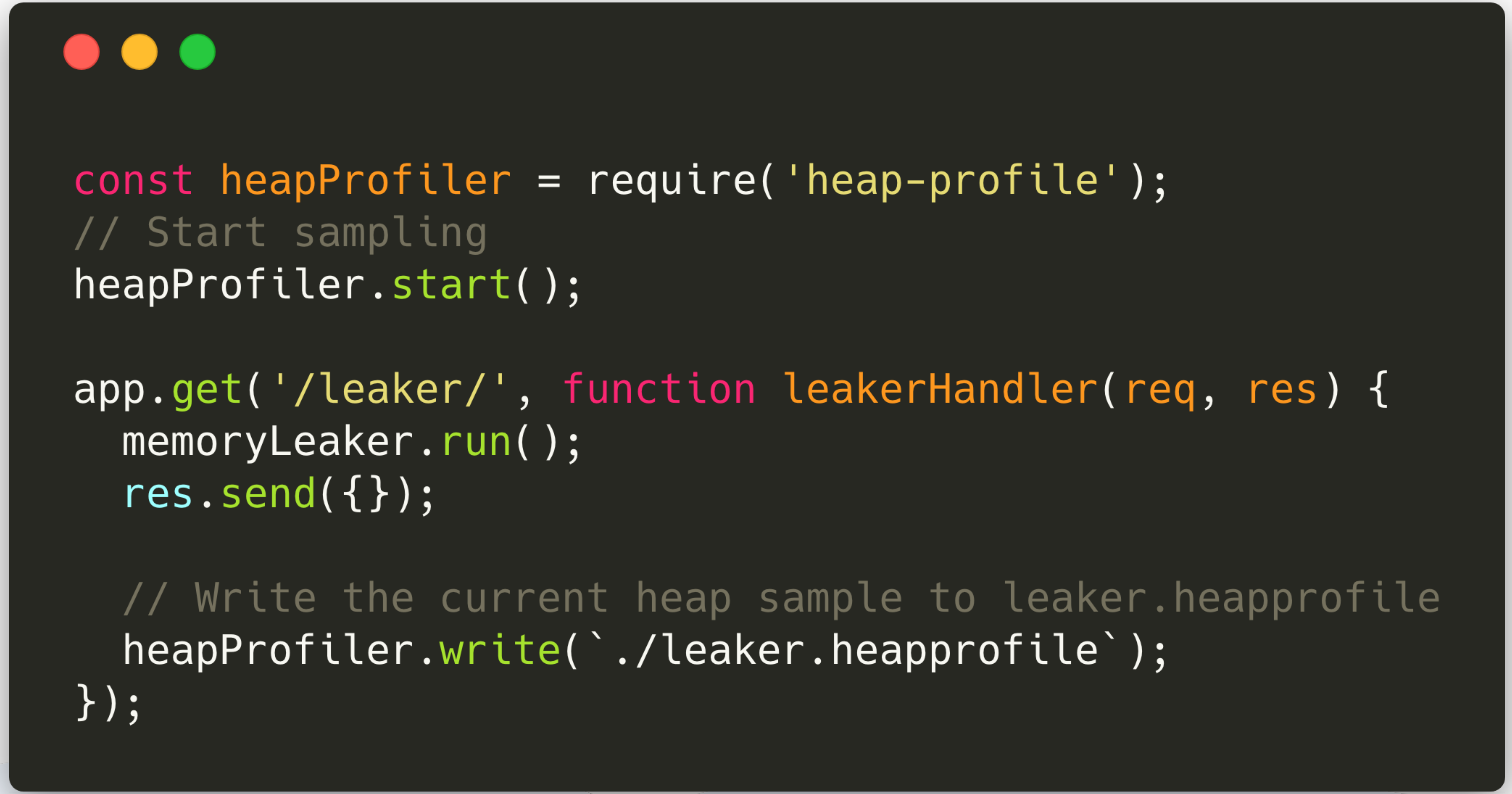
```
# Expose V8 compiler information to perf
$ node --perf-basic-prof \
       --interpreted-frames-native-stack \
       --no-turbo-inlining \
       server.js

# Sample our server for 10 seconds
# at a frequency of 99Hz
$ perf record -F99 -g -p $(pgrep -x -n node) -- \
  sleep 10
$ perf script > result.perf
```

- Linux *perf*
- V8 CpuProfiler
- V8 SamplingHeapProfiler
- llnode
- node-report
- Structured Logs



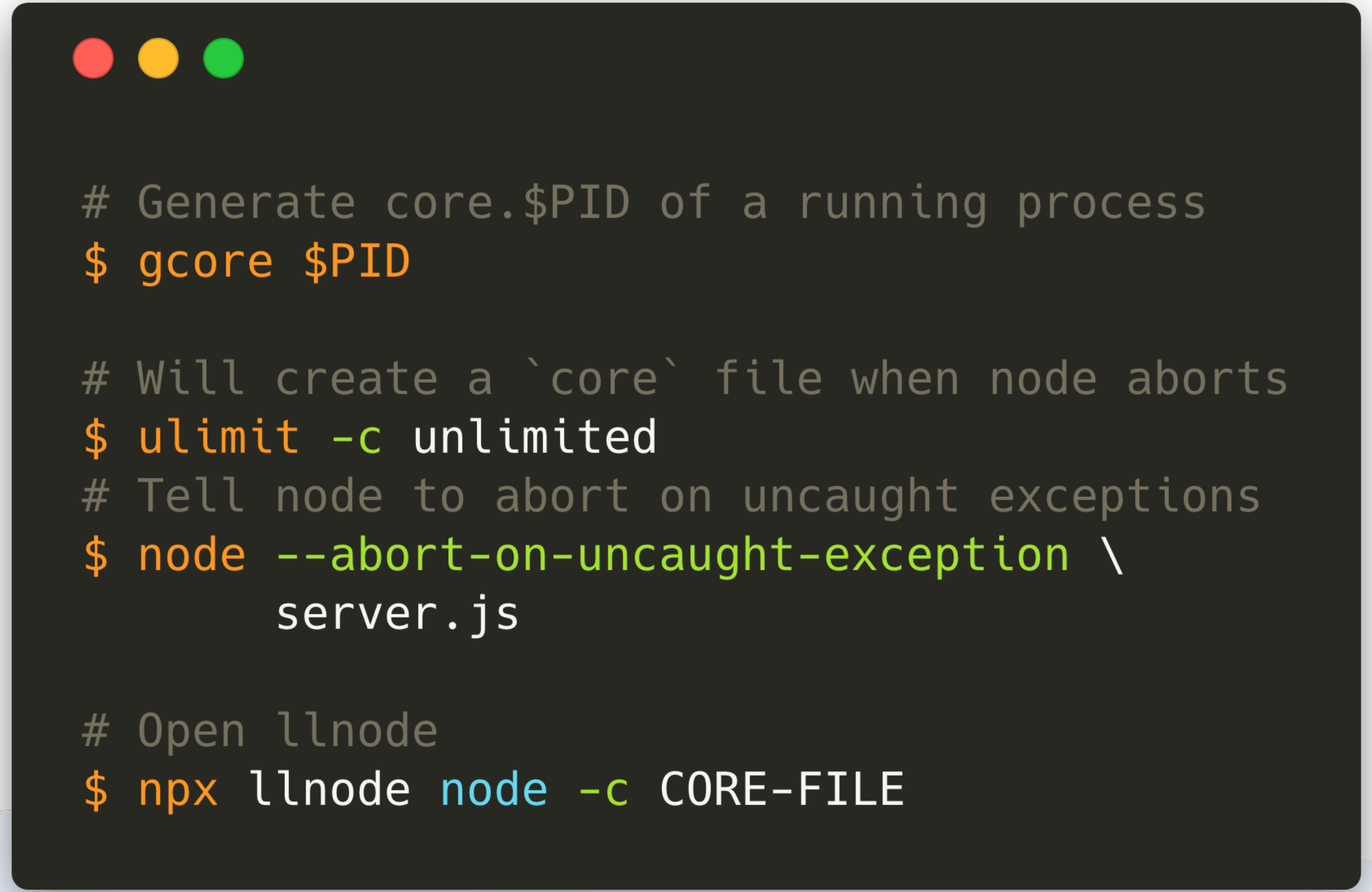
```javascript
const session = new require('inspector').Session();
session.connect();

app.get('/slow/', function slowHandler(req, res) {
  session.post('Profile.enable', () => {
    session.post('Profile.start', () => {
      heavyComputation.run();
      res.send({});
      session.post('Profile.stop', (err, { profile }) => {
        fs.writeFileSync(`slow.cpuprofile`,
                         JSON.stringify(profile));
      });
    });
  });
});
```

- Linux *perf*
- V8 CpuProfiler
- V8 SamplingHeapProfiler
- llnode
- node-report
- Structured Logs



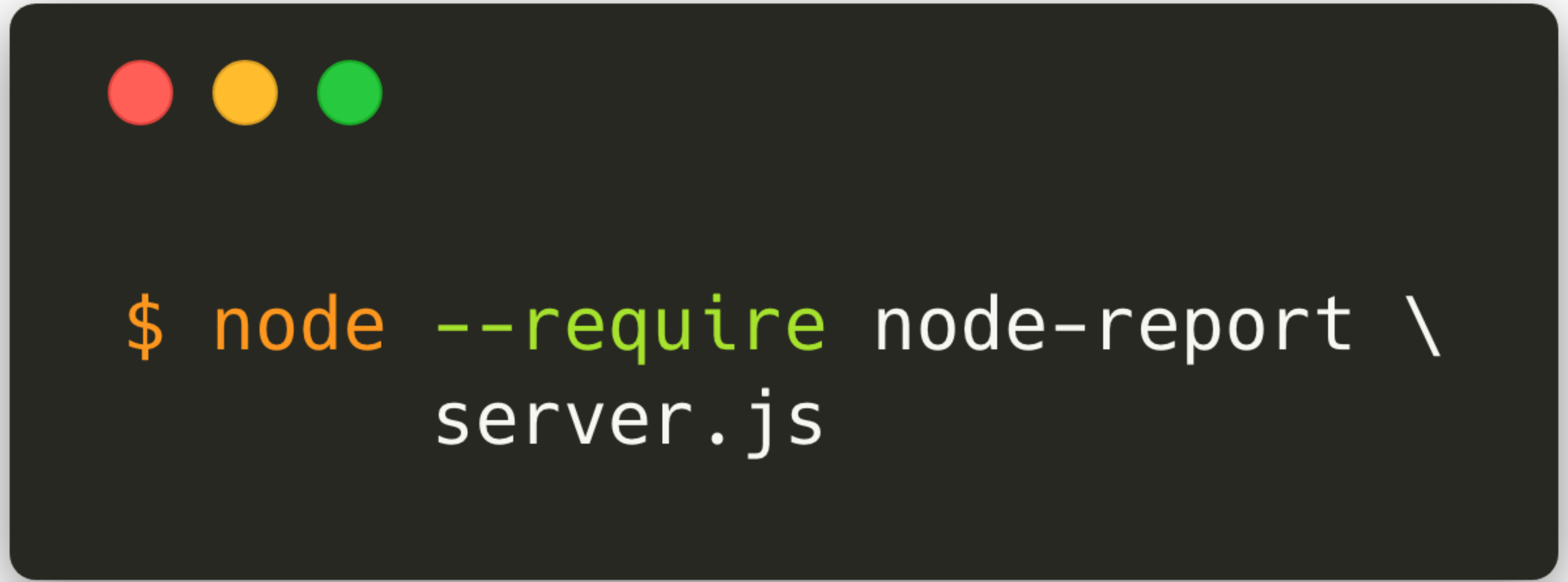
```javascript
const heapProfiler = require('heap-profile');
// Start sampling
heapProfiler.start();

app.get('/leaker/', function leakerHandler(req, res) {
  memoryLeaker.run();
  res.send({});

  // Write the current heap sample to leaker.heapprofile
  heapProfiler.write(`./leaker.heapprofile`);
});
```

- Linux *perf*
- V8 CpuProfiler
- V8 SamplingHeapProfiler
- llnode
- node-report
- Structured Logs



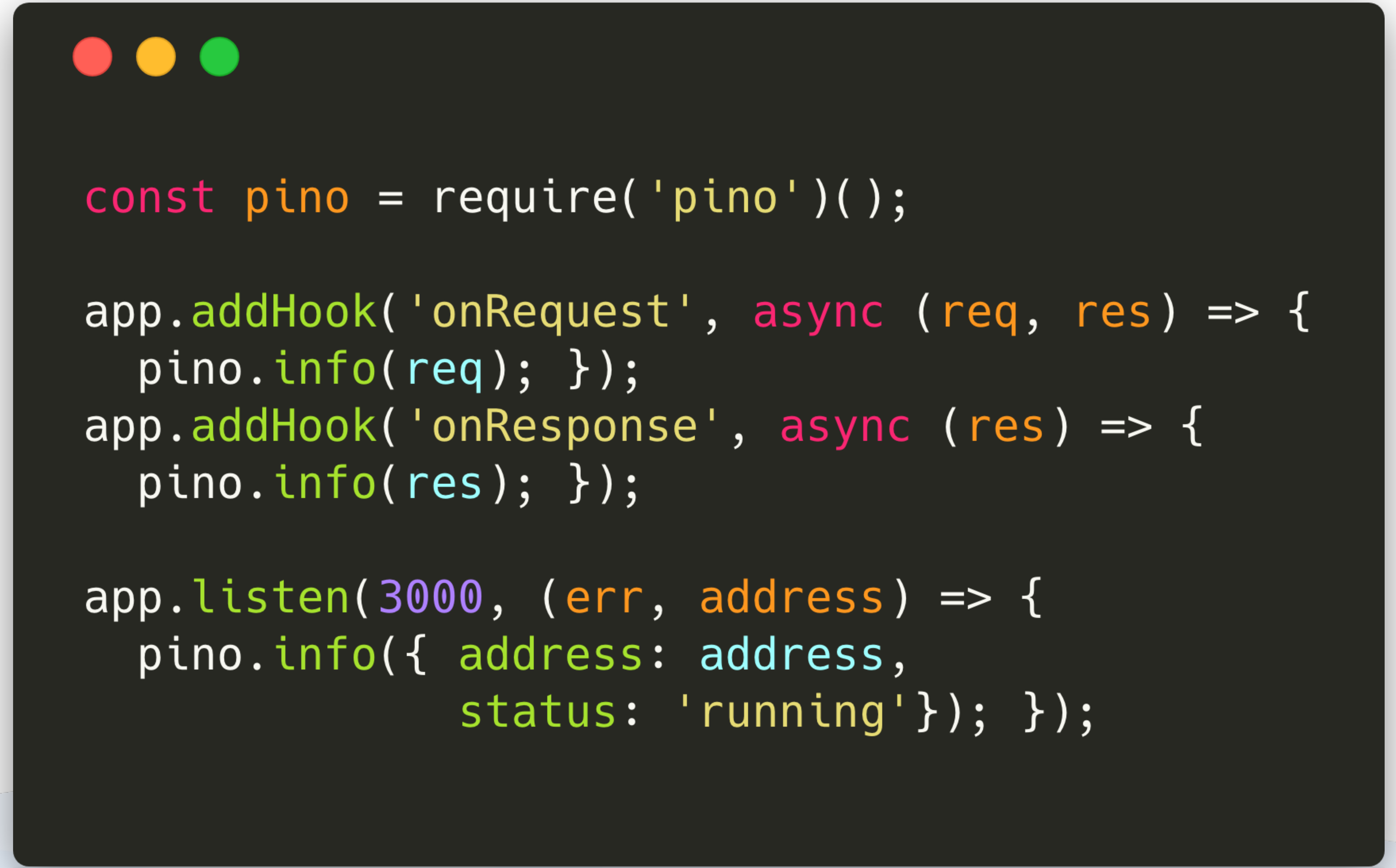
```
# Generate core.$PID of a running process
$ gcore $PID

# Will create a `core` file when node aborts
$ ulimit -c unlimited
# Tell node to abort on uncaught exceptions
$ node --abort-on-uncaught-exception \
       server.js

# Open llnode
$ npx llnode node -c CORE-FILE
```

- Linux *perf*
- V8 CpuProfiler
- V8 SamplingHeapProfiler
- llnode
- node-report
- Structured Logs



```
$ node --require node-report \
       server.js
```

- Linux *perf*
- V8 CpuProfiler
- V8 SamplingHeapProfiler
- llnode
- node-report
- Structured Logs



```
const pino = require('pino')();

app.addHook('onRequest', async (req, res) => {
  pino.info(req); });
app.addHook('onResponse', async (res) => {
  pino.info(res); });

app.listen(3000, (err, address) => {
  pino.info({ address: address,
              status: 'running'}); });
```

Now what?

INSTRUMENT ALL THE THINGS

- Linux *perf*
- V8 CpuProfiler
- V8 SamplingHeapProfiler
- llnode
- node-report
- Structured Logs

https://github.com/mmarchini/nodejs-production-diagnostic-tools

```
$ docker pull mmarchini/nodejs-production-diagnostic-tools:latest
```

@mmarkini

oss@mmarchini.me

https://mmarchini.me/

mmarchini

sthima
www.sthima.com