Running Node.js™ in Production

OR

Y U NO GIVE ME 1.0?!™

@indexzero
www.github.com/indexzero

Problem

How do you stabilize something that is by definition unstable?

Listen for Errors

But don't be too optimistic

```
//
// Do suppress errors
//
process.on('uncaughtException', function (err) {
  //
  // Log it!
  //
  console.dir(err);
});
```

Listen for Errors

But don't be too optimistic

```
//
// Do suppress errors
//
process.on('uncaughtException', function (err) {
  //
  // Log it!
  //
  console.dir(err);

  //
  // Make sure you still exit.
  //
  process.exit(1);
});
```

Listen for Errors

Seriously, listen for errors

```
var http = require('http');

var req = http.request({
  host: 'www.google.com',
  path: '/',
  port: 80,
  method: 'POST'
}, function (response) {
  //
  // Do stuff with the response here
  //
});
```

Listen for Errors

Seriously, listen for errors

```
node.js:134
        throw e; // process.nextTick error, or 'error' event on first
tick
        ^
Error: Uncaught, unspecified 'error' event.
    at EventEmitter.emit (events.js:47:15)
    at Object.<anonymous> (/Users/you/y-u-no-listen-for-errors.js:5:9)
    at Module._compile (module.js:404:26)
    at Object..js (module.js:410:10)
    at Module.load (module.js:336:31)
    at Function._load (module.js:297:12)
    at Array.<anonymous> (module.js:423:10)
    at EventEmitter._tickCallback (node.js:126:26)
```

Listen for Errors

Seriously, listen for errors

```
var http = require('http');

var req = http.request({
  host: 'www.google.com',
  path: '/',
  port: 80,
  method: 'POST'
}, function (response) {
  //
  // Do stuff with the response here
  //
});

req.on('error', function (err) {
  //
  // Decide how to handle this error, it's
  // safe to keep your process running
  //
});
```

But why?
https://github.com/joyent/node/blob/master/lib/http.js#L1204

Watch your Events

Leaking listeners like woah

```
(node) warning: possible EventEmitter memory leak detected. 11
listeners added. Use emitter.setMaxListeners() to increase limit.
Trace:
   at Pool.<anonymous> (events.js:101:17)
   at Object.proxyRequest (~/http-proxy/lib/node-http-proxy.js:185:7)
   at Server.<anonymous> (/Users/some-user/myapp.js:14:9)
   at Server.emit (events.js:45:17)
   at HTTPParser.onIncoming (http.js:1078:12)
   at HTTPParser.onHeadersComplete (http.js:87:31)
   at Socket.ondata (http.js:977:22)
   at Socket._onReadable (net.js:654:27)
   at IOWatcher.onReadable [as callback] (net.js:156:10)
```

FYI: I fixed this in 0.4.x; now at 0.5.x. Proxying FTW

Watch your Events

Event cardinality is important

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

function doSomethingThenTellMe () {
  var emitter = new events.EventEmitter();

  setTimeout(function () {
    emitter.emit('done');
  }, 2000);

  return emitter;
}

var doingIt = doSomethingThenTellMe();

//
// Why are you using `.on()`? You only expect this event once.
//
doingIt.on('done', function () {
  console.log("Ok, it's done");
});
```

Watch your Events

.once is your friend

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

function doSomethingThenTellMe () {
  var emitter = new events.EventEmitter();

  setTimeout(function () {
    emitter.emit('done');
  }, 2000);

  return emitter;
}

var doingIt = doSomethingThenTellMe();

//
// Just use `.once()` instead
//
doingIt.once('done', function () {
  console.log("Ok, it's done. And it won't leak by mistake");
});
```

Use a Process Monitor

Monit, Upstart, Forever, etc.

```
$ cd /path/to/your/app
$ forever start bin/yourapp
info: Running action: start
info: Forever processing file: examples/server.js
$ forever list
info: Running action: list
info: Forever processes running
  [0] bin/yourapp [77041, 77040] /Users/You/.forever/20dL.log 0:0:0:1.788
```

Disclaimer. I'm biased.
https://github.com/indexzero/forever

Stabilize your dependencies

Packages from the ground up

Gotta find ‘em all

Avoiding situations like this

```
node.js:134
        throw e; // process.nextTick error, or 'error' event on
first tick
        ^
Error: Cannot find module 'forgot-to-install-this'
    at Function._resolveFilename (module.js:320:11)
    at Function._load (module.js:266:25)
    at require (module.js:348:19)
    at Object.<anonymous> (/Users/you/your-app/bin/server:8:14)
    at Module._compile (module.js:404:26)
    at Object..js (module.js:410:10)
    at Module.load (module.js:336:31)
    at Function._load (module.js:297:12)
    at Array.<anonymous> (module.js:423:10)
    at EventEmitter._tickCallback (node.js:126:26)
```

Gotta find ‘em all

Lets find some dependencies

```
$ require-analyzer
info:  require-analyzer starting in /Users/Charlie/Nodejitsu/require-analyzer
warn:  No dependencies found
info:  Analyzing dependencies...
info:  Done analyzing raw dependencies
info:  Retrieved packages from npm
info:  Additional dependencies found
data:  {
data:    findit: '= 0.0.3',
data:    npm: '= 0.3.18'
data:  }
info:  Updating /Users/Charlie/Nodejitsu/require-analyzer/package.json
info:  require-analyzer updated package.json dependencies
$ npm install .
```

https://github.com/nodejitsu/require-analyzer

Gotta find ‘em all

Examining a real package.json

http://github.com/indexzero/winston/blob/master/package.json

```
{
  "name": "winston",
  "description": "A multi-transport async logging library for Node.js",
  "version": "0.2.7",
  "author": "Charlie Robbins <charlie.robbins@gmail.com>",
  "contributors": [
    { "name": "Matthew Bergman", "email": "mzbphoto@gmail.com" }
  ],
  "repository": {
    "type": "git",
    "url": "http://github.com/indexzero/winston.git"
  },
  "keywords": ["logging", "sysadmin", "tools"],
  "dependencies": {
    "colors": ">= 0.3.0",
    "eyes": ">= 0.1.6",
    "loggly": ">= 0.1.4",
    "vows": ">= 0.5.2"
  },
  "main": "./lib/winston",
  "scripts": { "test": "vows test/*-test.js --spec" },
  "engines": { "node": ">= 0.3.0" }
}
```

I WAS DOING IT WRONG™

Thursday, May 5, 2011

Gotta find ‘em all

Examining a real package.json

http://github.com/indexzero/winston/blob/master/package.json

```
{
  "name": "winston",
  "description": "A multi-transport async logging library for Node.js",
  "version": "0.2.7",
  "author": "Charlie Robbins <charlie.robbins@gmail.com>",
  "contributors": [
    { "name": "Matthew Bergman", "email": "mzbphoto@gmail.com" }
  ],
  "repository": {
    "type": "git",
    "url": "http://github.com/indexzero/winston.git"
  },
  "keywords": ["logging", "sysadmin", "tools"],
  "dependencies": {
    "colors": "0.x.x",
    "eyes": "0.1.x",
    "loggly": "0.1.x",
    "vows": "0.5.x"
  },
  "main": "./lib/winston",
  "scripts": { "test": "vows test/*-test.js --spec" },
  "engines": { "node": ">= 0.3.0" }
}
```

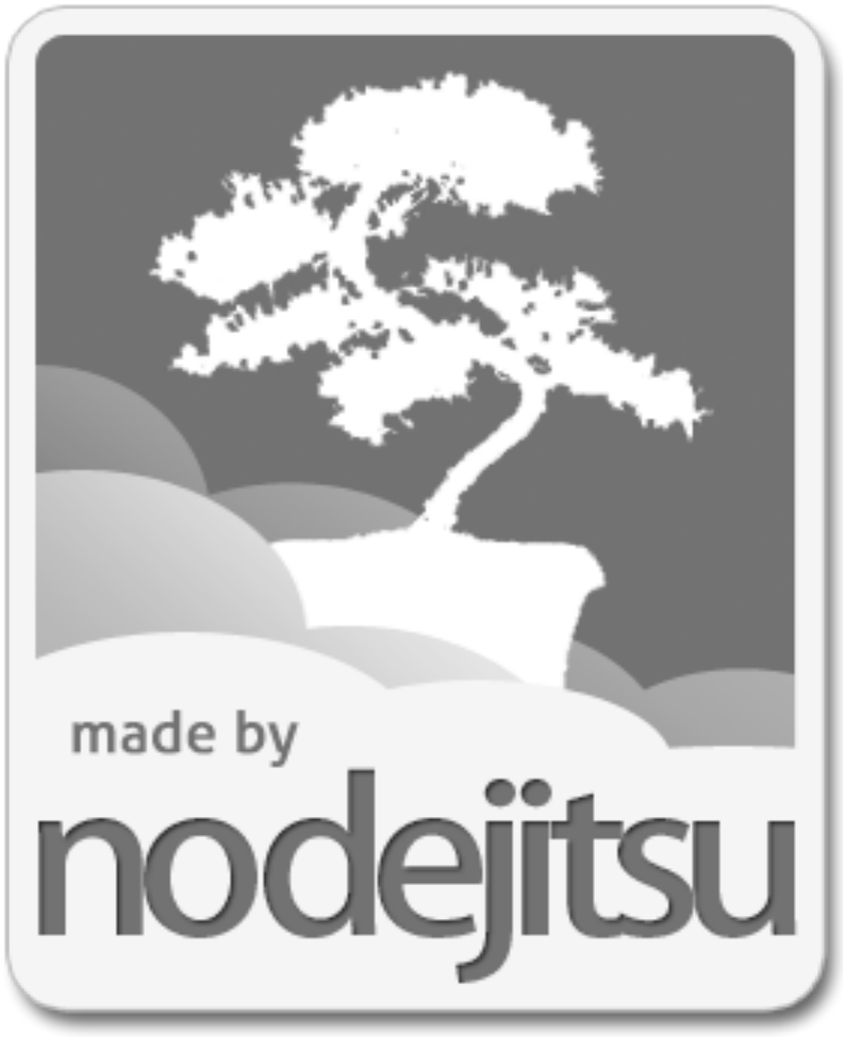
Gotta find ‘em all

Understanding node-semver

```
$ node
> var semver = require('semver');
> semver.satisfies('1.0.0', '0.x.x');
false
> semver.satisfies('0.5.0', '0.x.x');
true
>
```

Gotta find ‘em all

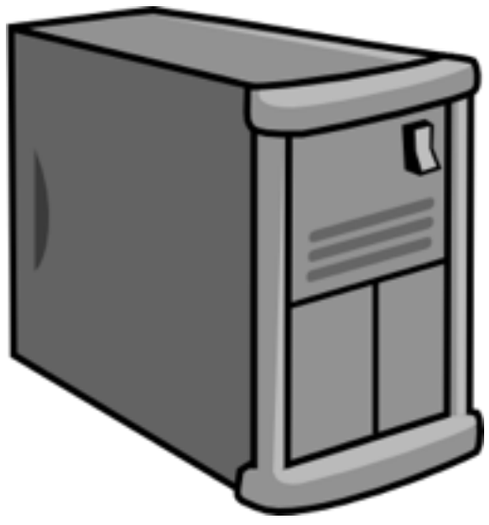
Best tool for the job



Structure your code

I don't really care how. Just have a plan.

```
bin/
  server
config/
  env/
    development.json
    production.json
lib/
  myapp/
    module1/
      index.js
    module2/
      index.js
    module3/
      index.js
  myapp.js
public/
  [static-files]
test/
  module1/
    module1-component1-test.js
    module1-component2-test.js
  [...]
vendor/
  some-git-submodule
  some-other-git-submodule
```



Decide on a Network Architecture

To Proxy or Not to Proxy. That is the question.

Option 1: Shared File Descriptors

https://github.com/LearnBoost/cluster/blob/master/lib/worker.js#L83

```
// stdin
this.stdin = new net.Socket(0, 'unix');
this.stdin.setEncoding('ascii');
this.stdin.on('fd', this.server.listenFD.bind(this.server));
```

https://github.com/LearnBoost/cluster/blob/master/lib/worker.js#L83

```
// spawn worker process
this.proc = spawn(
    node
  , this.master.cmd
  , { customFds: customFds, env: env });

// unix domain socket for ICP + fd passing
this.sock = new net.Socket(fds[1], 'unix');
```

https://github.com/learnboost/cluster

Option 2: Round-Robin Proxy

https://gist.github.com/869781

```
var httpProxy = require('http-proxy');

// Addresses to use in the round robin proxy
var addresses = [{
    host: 'ws1.0.0.0',
    port: 80
  },
  {
    host: 'ws2.0.0.0',
    port: 80
}];

httpProxy.createServer(function (req, res, proxy) {
  //
  // Get the first location off of the 'queue'.
  //
  var target = addresses.shift();

  // Proxy to the specified location
  proxy.proxyRequest(req, res, target);

  //
  // Push the location to the end of the 'queue'.
  //
  addresses.push(target);
});
```

BUT WAIT! THERE'S MORE!™

But wait! There's more!

We did the work so you don't have to™

```
$ bin/haibu-server
```

```
This is Open Source Software available under
the MIT License

© 2010 Nodejitsu Inc.
All Rights Reserved - www.nodejitsu.com
haibu started @ 127.0.0.1 on port 9002 as api-server
```

Y U NO MAKE QUESTIONS?!™