

# Make my Node all Enterprisey

Alternate Title:

“What do I do once I’ve built this thing”



# Vanity Plate

- `var name = "Rob Sullivan";`
- `var blog = "datachomp.com";`
- `var twitter = "@datachomp";`
- `var recruiters = "Linkedin.com/TheRobSullivan";`





# Enterprise?

## What does that even mean?

- “Enterprise” is basically a pretentious term used to mean a certain level of application resiliency ... Or a term used to just jack up the price of something.



# Node Package Manager

- Like Ruby Gems, NPM is a library of awesome
- Database Drivers
- CSS engines
- Templates
- Node Express
- The list goes on and on and on

<https://github.com/joyent/node/wiki/modules>



# Hosting: Not My Problem

- The rise of Platform As A Service(PaaS) and NodeJS mean you can just pay someone else to deal with it.
- Even Microsoft Azure is trying to get into the PaaS Node discussion.

<https://github.com/joyent/node/wiki/Node-Hosting>



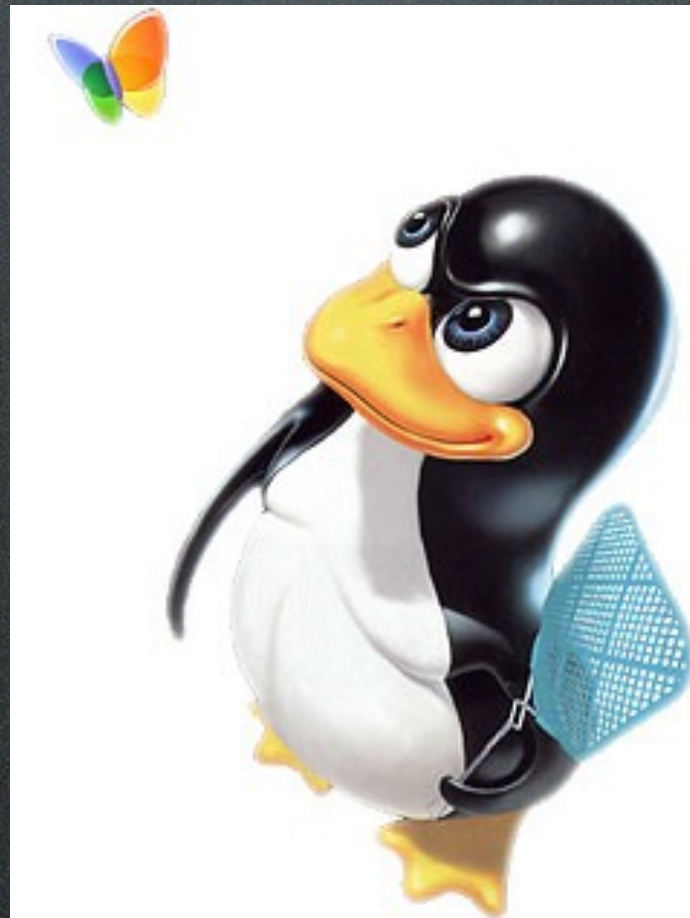
# Cloud just isn't for me

- Compliance
- Latency
- Flexiblity (control freak)
- Existing Hardware/infrastructure
- yada yada yada



# Where can I run this?

- MS has been working pretty hard to get Node to work on its platform
- Works on Windows...Thrives on \*nix



<http://www.okclugnuts.org/>



# Nginx

- Node can serve static files, but Nginx can serve them better

```
location / {  
    proxy_intercept_errors on;  
    proxy_pass http://127.0.0.1:8000; #this is our reverse proxy!  
    proxy_redirect off;  
}
```

```
location /css/ {  
    log_not_found off; # here we take over serving static content  
    access_log off;  
}
```

```
error_page 404 /404.html;  
location = /404.html {  
    root /path/to/error/pages; # we can do some error handling here as well  
}
```



# Daemonizing

- Upstart or Forever from the NPM

Bad

```
rob@server:~$ node server.js
```

```
start on startup
stop on shutdown
```

Good

```
script
```

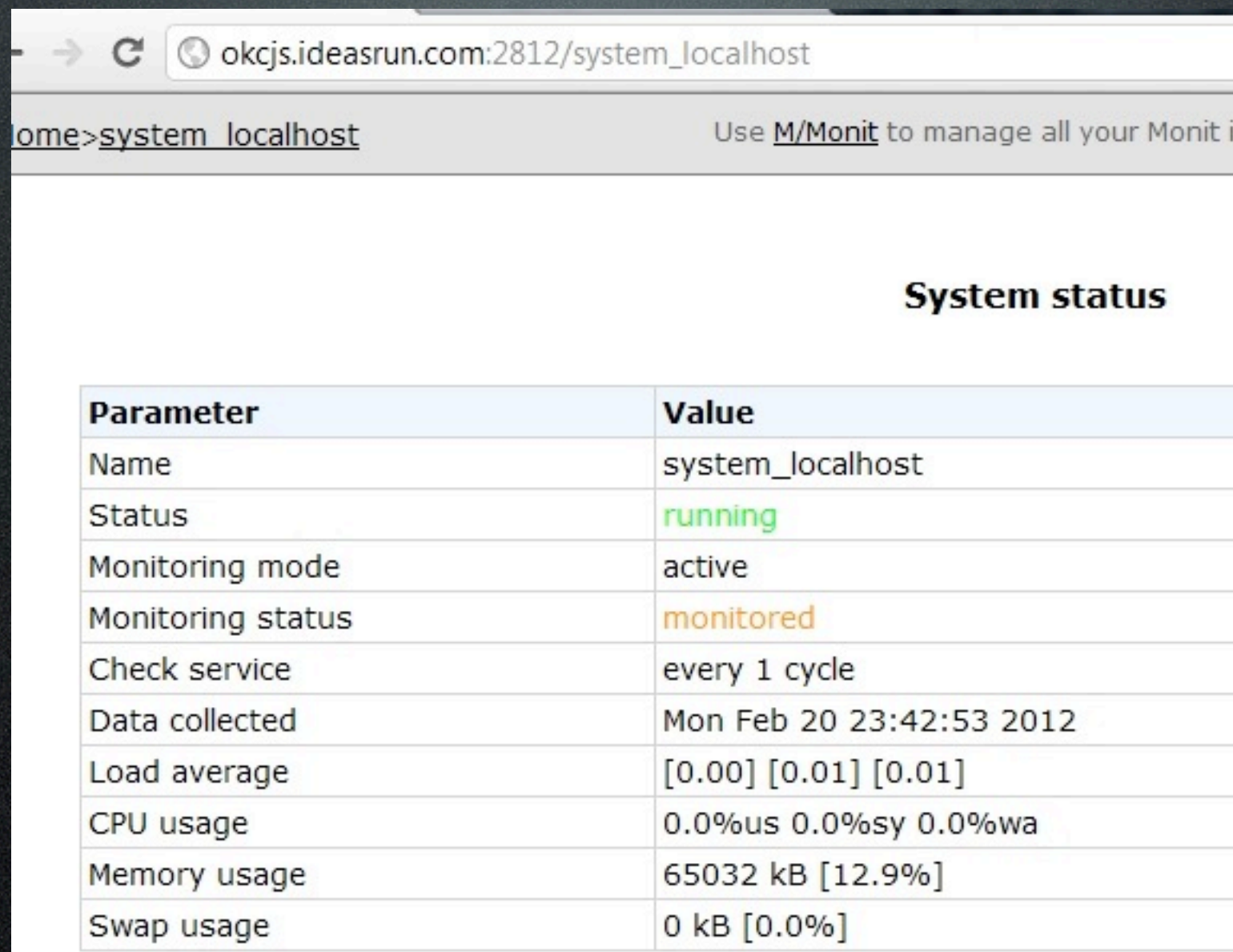
```
export HOME="/root"
```

```
exec /usr/local/bin/node /home/jsdemo/sites/chat_app/
current/server.js >> /var/log/nodechatapp.log
end script
```



# Dude, Where's my service?

- Control/Monitor Services with Monit



The screenshot shows a web browser window with the address bar displaying 'okcjs.ideasrun.com:2812/system\_localhost'. The page content includes a breadcrumb 'home > system\_localhost' and a link 'Use M/Monit to manage all your Monit in'. The main heading is 'System status'. Below it is a table with two columns: 'Parameter' and 'Value'.

Parameter	Value
Name	system_localhost
Status	running
Monitoring mode	active
Monitoring status	monitored
Check service	every 1 cycle
Data collected	Mon Feb 20 23:42:53 2012
Load average	[0.00] [0.01] [0.01]
CPU usage	0.0%us 0.0%sy 0.0%wa
Memory usage	65032 kB [12.9%]
Swap usage	0 kB [0.0%]



# Just spoon feed me

- Maybe TOP and a bunch of other neck beard commands aren't for you
- Throw on a New Relic agent and get back to coding





# Deployments

- Capistrano is sick
- Lots of packages in the NPM to help you out

## Capify .

```
default_run_options[:pty] = true

set :application, 'chat_app'
set :repository,  "https://DataChomp@github.com/okcjs/feb2012_node_talk_chat_app.git"
set :domain, 'okcjs.ideasrun.com'

set :user, "jsdemo" # The server's user for deploys

set :deploy_via, :remote_cache
set :scm, :git
set :branch, "master"
set :scm_verbose, true
set :use_sudo, false

set :deploy_to, "~/sites/#{application}"

role :web, domain # Your HTTP server, Apache/etc
role :app, domain # This may be the same as your `Web` server
```

## Cap Deploy FTW!



# Just for Kicks

- Lets try and run our app into the ground





# Give me the codes!!!

- [https://github.com/okcjs/feb2012\\_node\\_talk\\_chat\\_app](https://github.com/okcjs/feb2012_node_talk_chat_app)
- Thank you again to Principal Tech
- See you next month!