

The Node.js logo, featuring the word "node" in a stylized white font with a green hexagon in place of the dot of the second 'o', followed by a green hexagon containing the letters "JS" in white, and a small "TM" trademark symbol.

# Introduction

Wednesday, August 31, 2011

James Wang  
Referentia Systems Incorporated  
Hi-Capacity Honolulu Makerspace

***"I/O needs to be done differently"***

***"Never block or wait."***

# Fundamentals

- Evented and Non-Blocking IO
- Similar to EventMachine (Ruby) or Twisted (Python)
- V8 Server-side Javascript
- Every line executes fast - non-blocking IO.
  - "The less you block, the more you do."

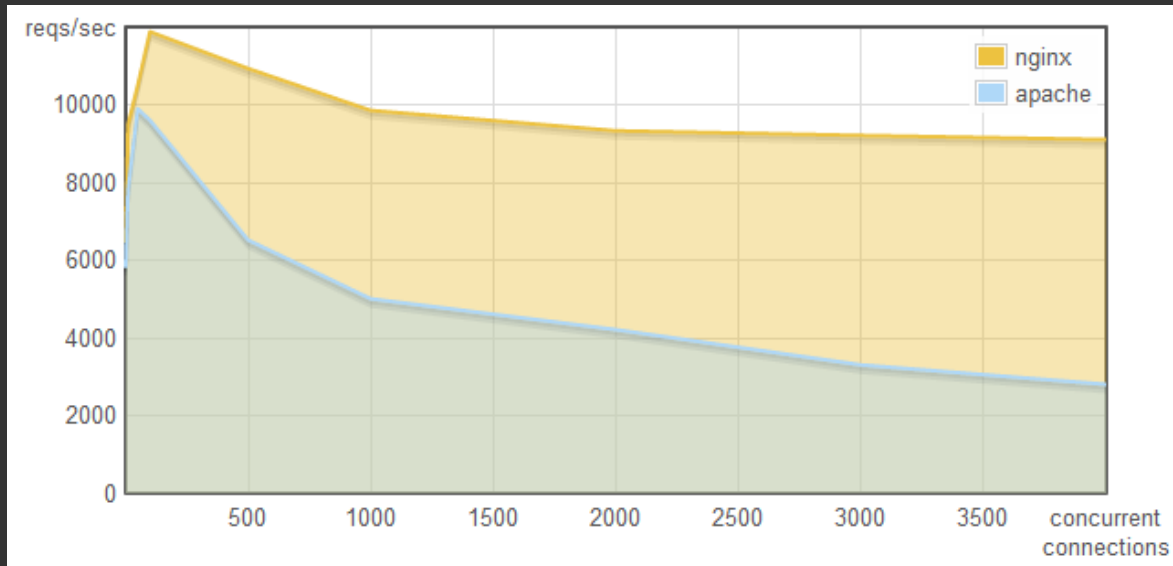
# Installation

- Package managers - apt-get, yum, ports, homebrew
- Compile - configure, make, make install
- Node Package Manager (NPM)
  - Node module repository
  - Handy for dependencies, global utilities, very much like rubygems.

# IO Latency

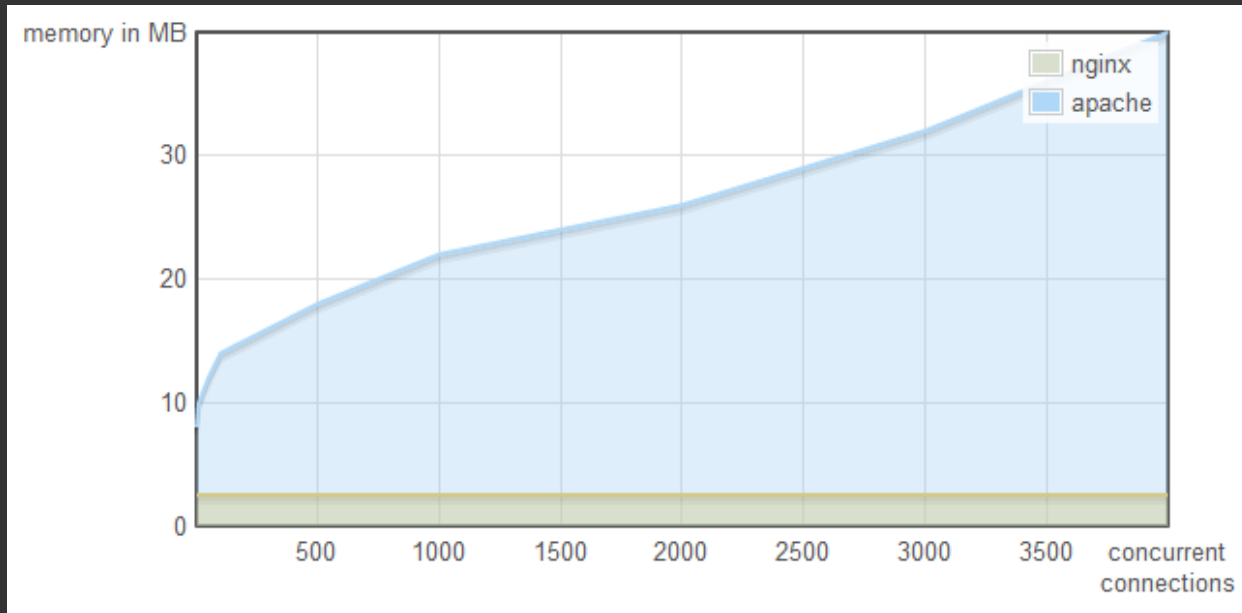
- L1: 3 cycles
- L2: 14 cycles
- Ram: 250 cycles
- Disk: 41,000,000 cycles
- Network: 240,000,000 cycles

# Case Study: Apache vs Nginx



- Nginx is event-based
- Handles more concurrent connections through using the event loop

# Case Study: Apache vs Nginx



- Doesn't need to spawn new processes or threads per request
- Memory usage is very low.

# Why Javascript?

- People who work with web technologies probably already know Javascript.
- Javascript's web application culture of Javascript is already geared towards evented IO.
- Node's goal is to "To provide a purely evented, non-blocking infrastructure to script highly concurrent programs."



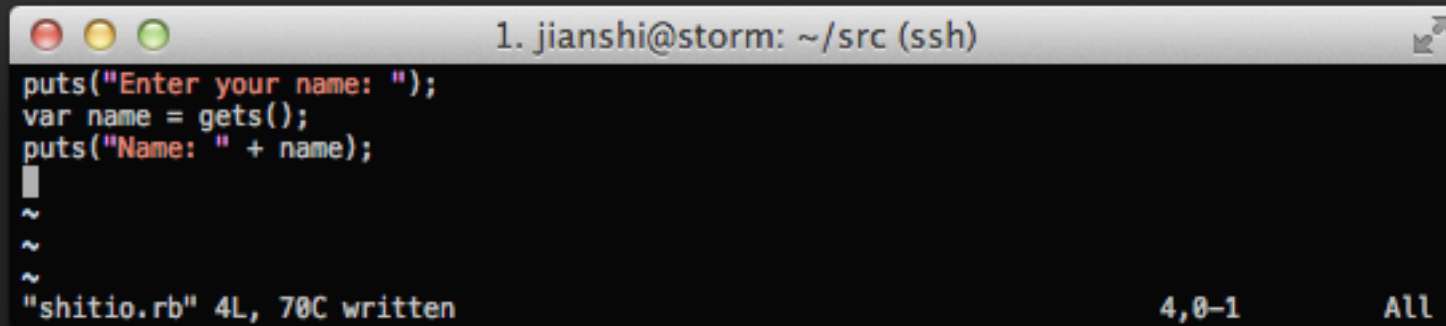
# Popularity

Lots of traction!

- Mailing list: 4000 people (For comparison: Ruby on Rails 22000, Tornado: 1300, EventMachine 300)
- IRC: 600 people during peak hours
- Node Knockout #2 just happened this past weekend.
- Many meetups
- Documentation translations: Chinese, Japanese, Russian, Spanish, Persian, etc
- Three books in progress
- ~3600 modules posted on the NPM registry
- Fanboys and Haters Spamming Hacker News daily

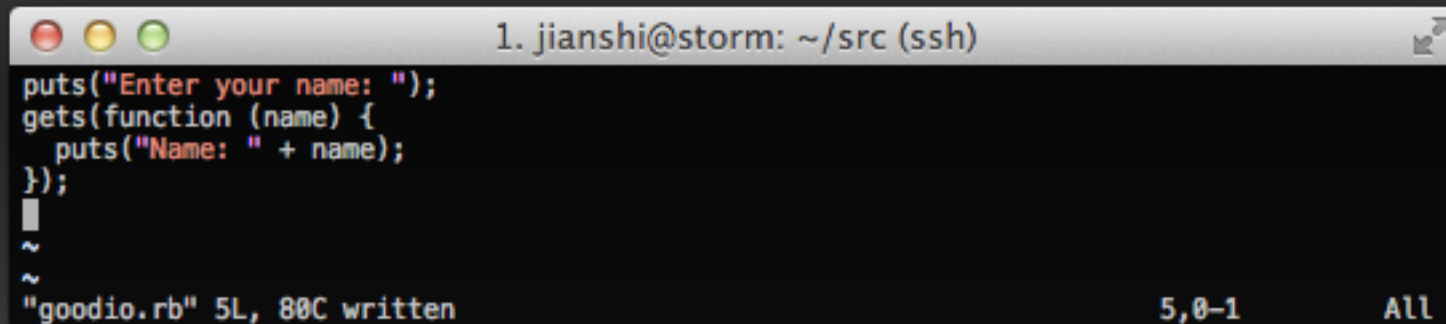
# Biased by Experience

Blocking IO. This is how we learned programming.

A terminal window titled "1. jianshi@storm: ~/src (ssh)" with a dark background. It contains Ruby code for a blocking IO program. The code prompts for a name and prints it. The status bar at the bottom indicates "4,0-1" lines and "All" columns are visible.

```
puts("Enter your name: ");
var name = gets();
puts("Name: " + name);
~
~
~
"shitio.rb" 4L, 70C written      4,0-1      All
```

Non-blocking IO. We're not used to this. Most will find this hard to read.

A terminal window titled "1. jianshi@storm: ~/src (ssh)" with a dark background. It contains Ruby code for a non-blocking IO program using a lambda function. The status bar at the bottom indicates "5,0-1" lines and "All" columns are visible.

```
puts("Enter your name: ");
gets(function (name) {
  puts("Name: " + name);
});
~
~
~
"goodio.rb" 5L, 80C written      5,0-1      All
```

# Event Loop

- Single-threaded - does a single operation at a time.
- Program exits when there's nothing left to do.
- Emitted events run right away.

# Simple Callback via setTimeout and setInterval

- Same functions in the web browser
- Shows asynchronous behavior of code
- Non-blocking IO demonstration

# TCP/IP Chat

- Uses the 'net' module
- Connect via telnet on port 8000
- Type random things to each other anonymously

# Monitoring an Node Application with Supervisor and the FS Module

- Uses the 'fs' module to monitor file modifications.
- Supervisor restarts node application when changed. So you don't have to restart it yourself.
- Simple web application using express.
- Supervisor used to start/restart on change or crash.

# Twitter's Streaming API

- Twitter's Streaming REST API
- Socket IO web sockets to the client side.
- JQuery updates on the client side.

# Follow me!

James Wang

- Email: [james@hicapacity.org](mailto:james@hicapacity.org)
- Twitter: [@synthes1s](https://twitter.com/synthes1s)
- Facebook: <https://www.facebook.com/wangbus>
- Freenode IRC: wangbus at #hicapacity