Who?

What?

2011-04-30 02:40:41 -0400

http://google.com

I was a Java guy for 10 years and I've been a Rubyist for the last 5 years. Over the years, I've tried to develop expertise in a particular area of technology that will both pay the bills and make me happy as a programmer while also watching for upcoming changes in the tech world. I often find myself diving into a particular technology just to get my hands dirty and get a feel for its strengths and weaknesses. As my JavaScript skills have always been weak, I've decided to deep dive into
Node.js
to understand what it does well and improve my JavaScript skills at the same time.

For this post, I'm just going to cover the basics; I'll follow up soon with deeper posts.

Overview

JavaScript has an interesting history – it hasn't developed like most other languages; until recently, executing JavaScript meant embedding it in a web page for a browser to execute. A few things happened which radically hastened the rise in JavaScript as an reasonable server-side language:

AJAX and the Browser Wars have resulted in dramatic improvements in Javascript runtime performance and high-quality developer tools.,
, Node.js built Process, File and Network I/O APIs on top of Google's V8 JavaScript engine, allowing command line programs and daemons to be built in JavaScript.,
Node.js adds a friendly command line face to V8 and APIs that are conceptually similar to Ruby's EventMachine library: all I/O is asynchronous and threads are unavailable to user code. Additionally JavaScript is a prototype-based language, not object-oriented. This makes for a programming model that is radically different from what Ruby or Java developers are used to.

Installation

I'm going to assume OSX and I like to install things with
Homebrew
. We'll install node and npm, node's package manager, with these commands:

```
brew update # update Homebrew's formulas to the latest
brew install node # install node
curl http://npmjs.org/install.sh | sudo sh # install npm
```

Once installed, you should be able to run
node --help
and
npm --help
.

A minimal web server using Node.js:

```
var http = require('http');
http.createServer(function (req, res) {
res.writeHead(200, {'Content-Type': 'text/plain'});
res.end('Hello World\n');
}).listen(8124, "127.0.0.1");
```

Copy that code into
hello.js
and run it:

```
node hello.js
```

Now let’s slam it with some requests:

ab -n 10000 -c 50 http://127.0.0.1:8124/

Results:

```
Server Hostname:        127.0.0.1
Server Port:            8124
Document Path:          /
Document Length:        12 bytes
Concurrency Level:      50
Time taken for tests:   1.479 seconds
Complete requests:      10000
Failed requests:        0
Write errors:           0
Total transferred:      760000 bytes
HTML transferred:       120000 bytes
Requests per second:    6760.79 [#/sec] (mean)
Time per request:       7.396 [ms] (mean)
Time per request:       0.148 [ms] (mean, across all concurrent requests)
Transfer rate:          501.78 [Kbytes/sec] received

Connection Times (ms)
min  mean[+/-sd] median   max
Connect:        0    0   0.2      0       3
Processing:     1    7   3.6      7      20
Waiting:        1    7   3.6      7      20
Total:          1    7   3.6      7      22

Percentage of the requests served within a certain time (ms)
50%      7
66%      9
75%     10
80%     11
90%     12
95%     13
98%     15
99%     16
100%     22 (longest request)
```

Not bad. Of course, this is using localhost and a trivial app but at least we know it’s up and running well. In my next post, we’ll explore the Node.js source code itself.