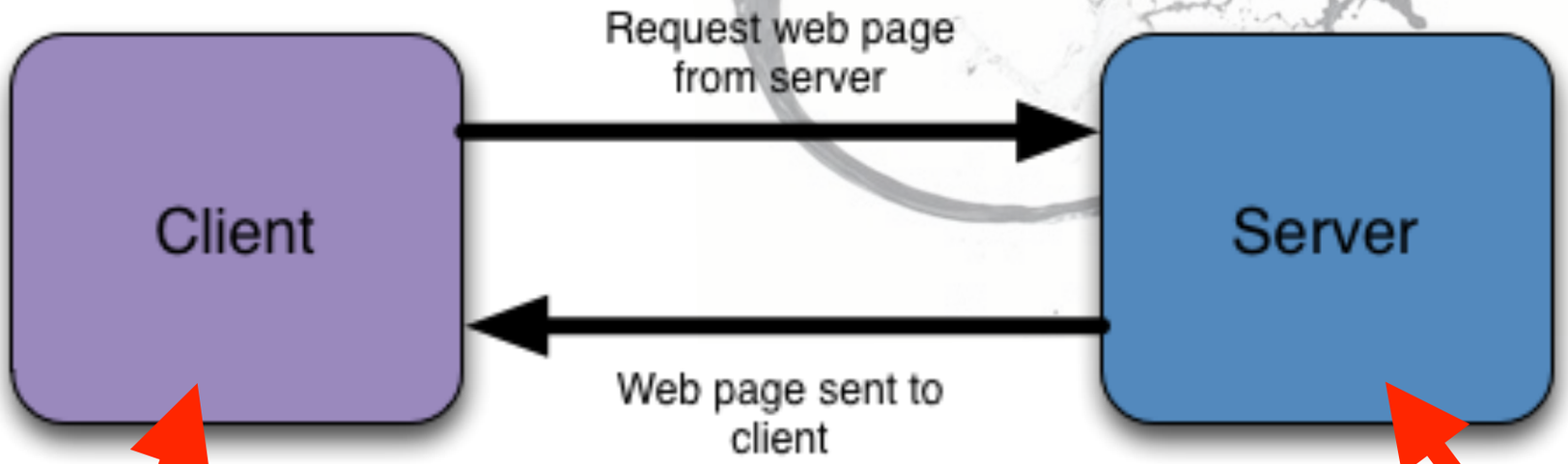
Node.js

What the heck is Node?

Ryan Dahl “A bunch of Sugar on Top of V8”... focus is on doing networking correctly.

Node is a set of bindings to the google V8 JS engine.
Allows you to script programs that do IO in js
focused on hi performance

Node.js is an open-source, cross-platform runtime environment for developing server-side Web applications. Although Node.js is not a JavaScript framework, many of its basic modules are written in JavaScript, and developers can write new modules in JavaScript.



HTML
CSS
jQuery
Angular.js

Node.js


```
1
2  // Load the http module to create an http server.
3  var http = require('http');
4
5  // Configure our HTTP server to respond with Hello World to all requests.
6  var server = http.createServer(function (request, response) {
7      response.writeHead(200, {
8          "Content-Type": "text/plain"
9      });
10     response.end("Hello World\n");
11 });
12
13 // Listen on port 8000, IP defaults to 127.0.0.1
14 server.listen(8000);
15
16 // Put a friendly message on the terminal
17 console.log("Server running at http://127.0.0.1:8000/");
```

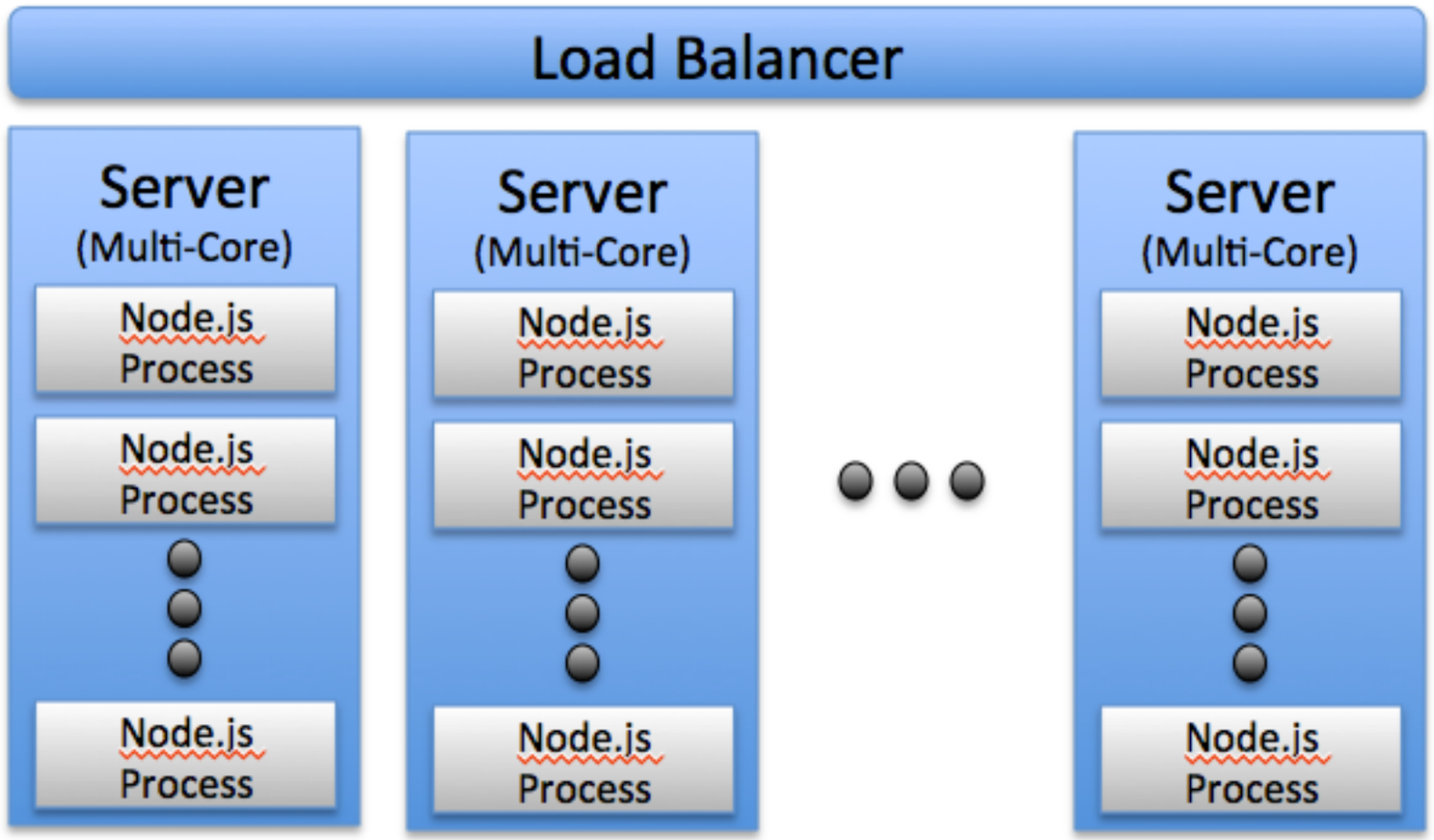
simple-server.js

```
var express = require('express');
var app = express();
var port = 8000;

app.get('/', function (req, res) {
  res.send('Hello World!');
});

app.get('/a', function (req, res) {
  res.send('Hello a class!');
});

app.get('/b', function (req, res) {
  res.send('Hello another class!');
});

app.listen(port, function () {
  console.log('Example app listening on port ' + port);
});
```

Node.js is a software platform that is used to build scalable network (especially server-side) applications. Node.js utilizes JavaScript as its scripting language, and achieves high throughput via non-blocking I/O and a single-threaded event loop.

Node.js is a software platform that is used to build **scalable** network (especially server-side) applications. Node.js utilizes JavaScript as its scripting language, and achieves high throughput via **non-blocking I/O and a single-threaded event loop**.

scalable network (especially server-side) applications.



non-blocking I/O and a single-threaded event loop.

You sit at a table and the server grabs your drink order.

The server goes back to the bar and passes your order to a bartender.

While the bartender is working on your drink, the server moves on to grab another table's drink order.

The server goes back to the bar and passes along the other table's order.

Before the server brings back your drinks, you order some food.

Server passes your food order to the kitchen.

our drinks are ready now, so the server picks them up and brings them back to your table.

ne other table's drinks are ready, so the server picks them up and takes them to the other table.

nally your food is ready, so server picks it up and brings it back to your table.

asically every interaction with the server follows the same pattern. First, you der something. Then, the server goes on to process your order and return it to you when it's ready.

Demo

Let’s write a basic node application…
Step 1 & Hello.java

Applications written in Node run on a single-thread. This means that they can only do one thing at a time.

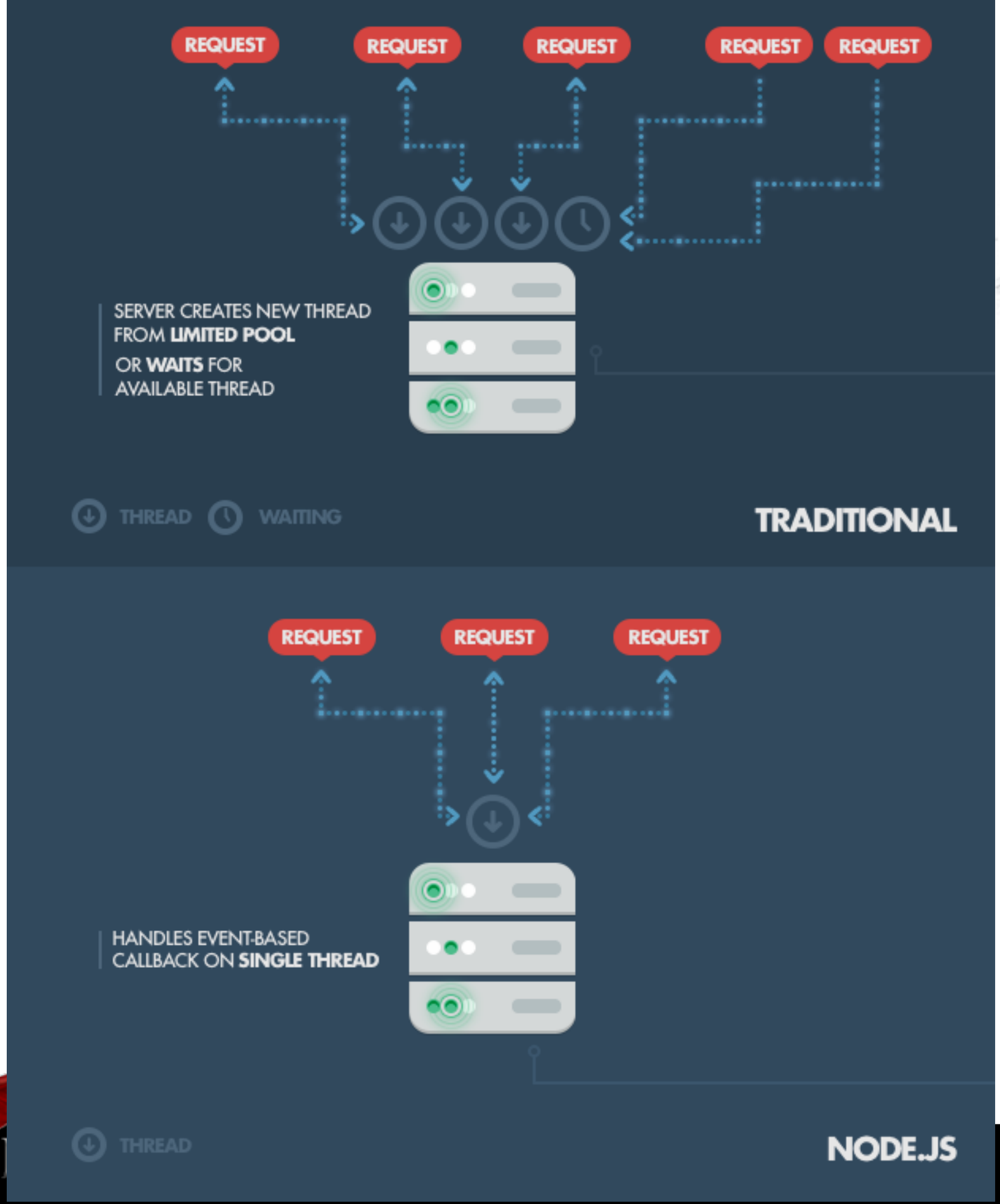
step1.js

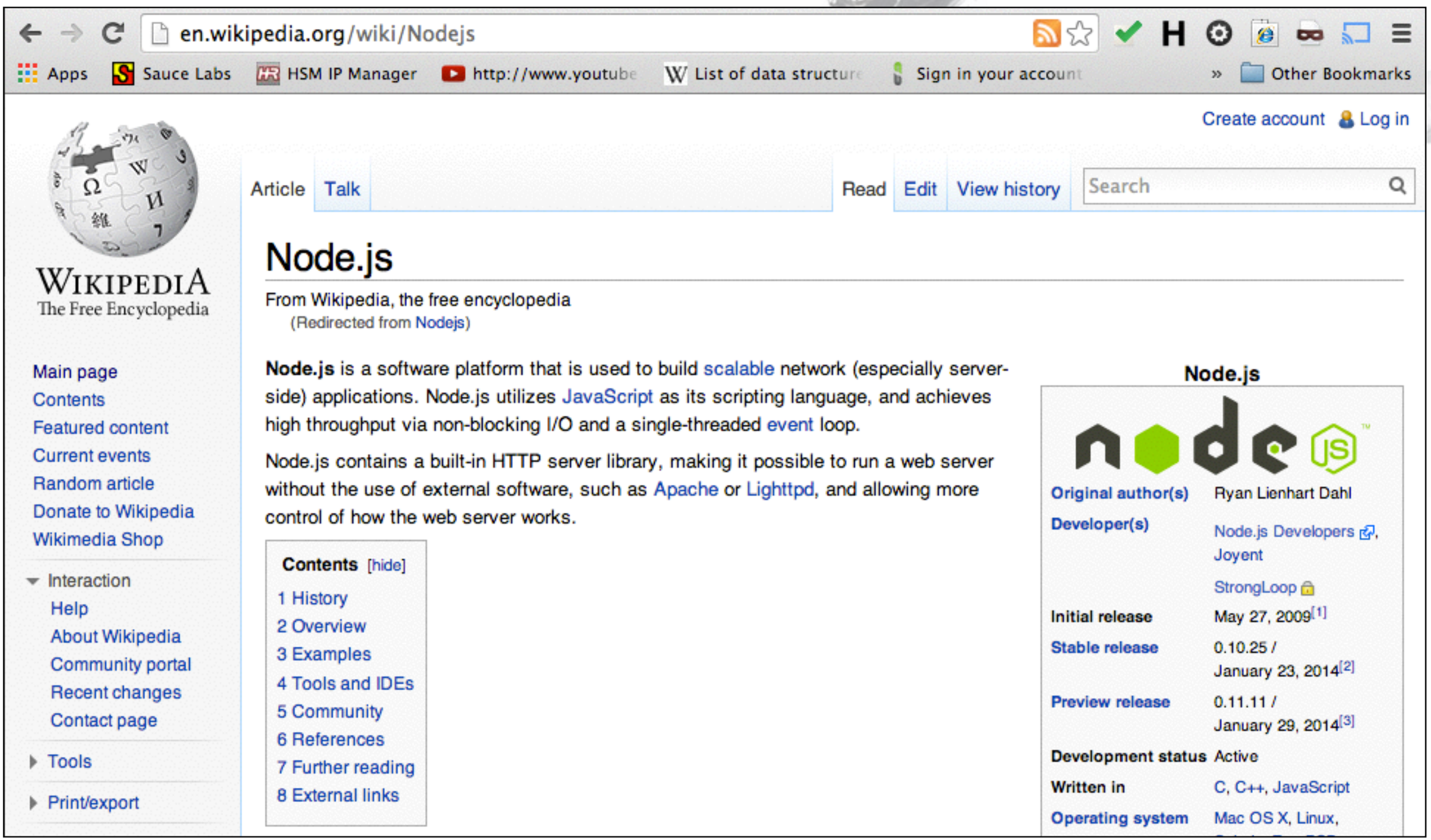
```
1  //node is nonblocking so it's going to read right through this
2  //event and keep going. It doesn't block and wait for the response
3  //before continuing.
4  setTimeout(function(){
5      console.log("world");
6  },2000);
7  consdle.log("hello");
8
9
```

Hello.java

```
1  /*
2  Java can be writen using a NIO (nonblock io) library,
3  but in Java when you pause, it generally blocks (or waits).
4  */
5  public class Hello{
6      public static void main(String[] args) throws Exception{
7          doCall();
8          System.out.println("Hello ");
9      }
10     private static void doCall() throws Exception{
11         Thread.sleep(4000);
12         System.out.println("World ");
13     }
14 }
```


Build a Network Application with Node (page 3 of 3)
REQUESTS, ETC
EVENT LOOP
(single thread)
Register Callback
INTENSIVE
OPERATION
File System
Database
Computation
Trigger Callback
Operation Complete


REQUEST
REQUEST
REQUEST
REQUEST
REQUEST
SERVER CREATES NEW THREAD
FROM LIMITED POOL
OR WAITS FOR
AVAILABLE THREAD
THREAD
WAITING
TRADITIONAL
REQUEST
REQUEST
REQUEST
HANDLES EVENT-BASED
CALLBACK ON SINGLE THREAD
THREAD
NODE.JS



http://en.wikipedia.org/wiki/Nodejs

- http://highscalability.com/blog/2013/12/11/using-nodejs-paypal-doubles-rps-lowers-latency-with-fewer-de.html

Node.js is a lightweight web container that is a strong option for development of micro services and as a server to mobile and single-page web applications. Due to the asynchronous nature of node.js, developers are turning to promise libraries to simplify their application code.

As the use of promises mature within the node.js community, we expect to see more applications developed for node.js.

For those teams that are reluctant to try node.js in production, it is still worthwhile to consider node.js for development tasks like running JavaScript tests outside of the browser or generating static web content from tools like CoffeeScript, SASS, and LESS.

Where Can I find node?

- https://github.com/joyent/node (of course)
- http://nodejs.org/dist/

- Walk thru a super simple chat application:
  - install node (http://nodejs.org/dist/)
  - write a chat client

Upgrade/Install node

Current NodeJs Version – First check current nodejs version on your system using following command. In my case it is v0.10.37.

```
rahul@tecadmin:~$ node -v

v0.10.37
```

Clean Cache Forcefully – Now clean all npm cache from your system forcefully.

```
rahul@tecadmin:~$ sudo npm cache clean -f

npm WARN using --force I sure hope you know what you are doing.
```

Install n Module – After cleaning all cache from your system, now install n modules using npm command.

```
rahul@tecadmin:~$ sudo npm install -g n

/usr/local/bin/n -> /usr/local/lib/node_modules/n/bin/n
n@2.0.2 /usr/local/lib/node_modules/n
```

Install Nodejs – Let's install or update latest nodejs version on your system using n module.

```
rahul@tecadmin:~$ sudo n stable

      install : node-v5.0.0
        mkdir : /usr/local/n/versions/node/5.0.0
        fetch : https://nodejs.org/dist/v5.0.0/node-v5.0.0-linux-
    installed : v5.0.0
```

Setup Binary Link – Now link your node binary with latest nodejs installed binary file using following command.

```
rahul@tecadmin:~$ sudo ln -sf /usr/local/n/versions/node/5.0.0/b
```

Check Nodejs Version – Finally recheck your nodejs version. You will find that it showing latest version on your system.

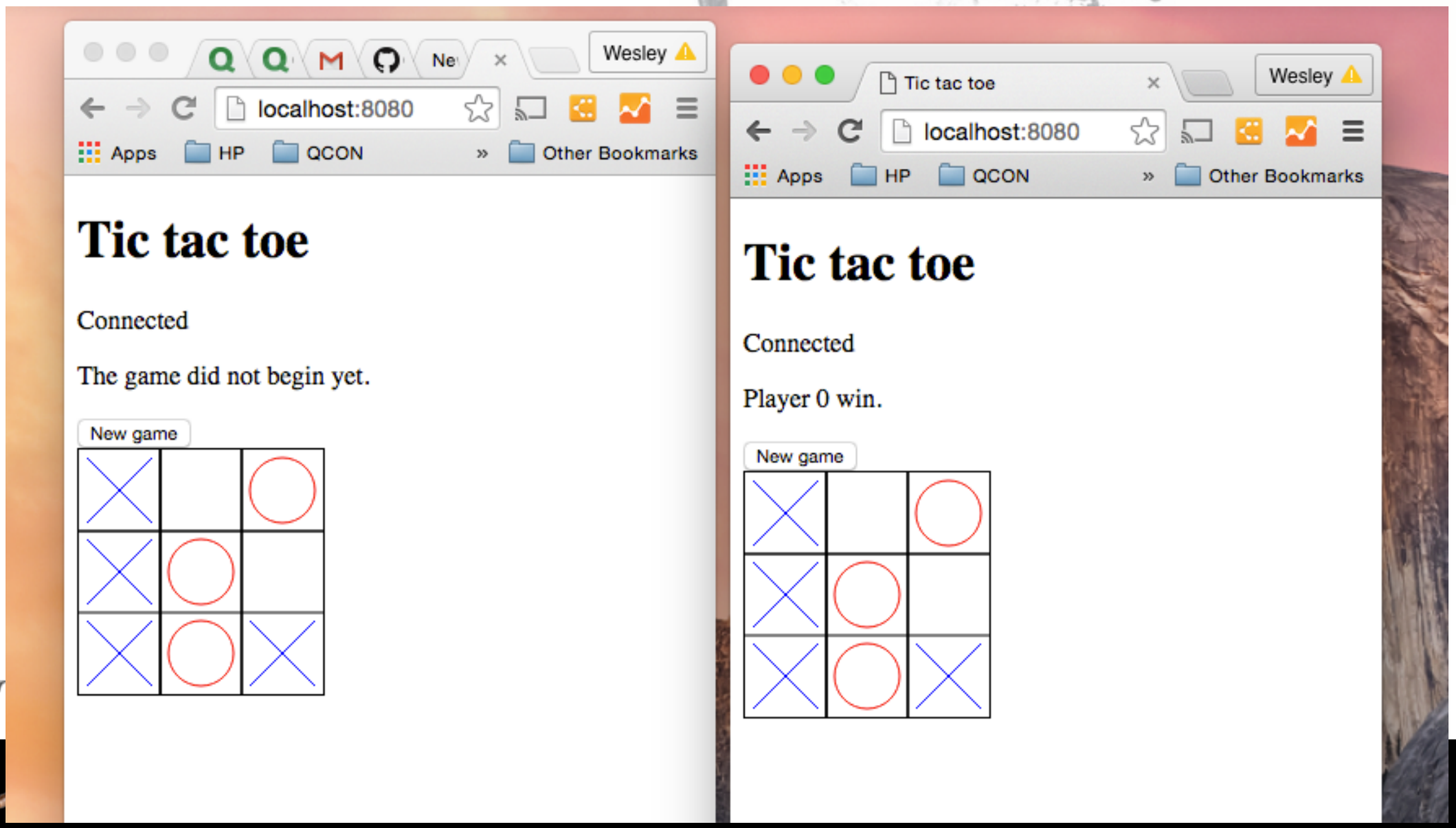
```
rahul@tecadmin:~$ node -v

v5.0.0
```

Demo

netstat-loop.sh × | chat.js × | chat1.js × | tcp-server-c

```
1   var net = require('net');
2   var port = 8000;
3   var sockets = [];
4   var s = net.Server(function(socket){
5       sockets.push(socket);
6
7       socket.on('data',function(d){
8         for(var i=0; i<sockets.length; i++){
9             if(sockets[i] == socket){
10              continue;
11            }
12            var msg = sockets[i].remoteAddress + ": " + d.toString();
13            console.log(msg);
14            sockets[i].write(msg);
15        }
16      });
17      socket.on('end',function(){
18        var i = sockets.indexOf(socket);
19        sockets.splice(i,1);
20      });
21  });
22
23  require('dns').lookup(require('os').hostname(), function (err, add, fam) {
24    console.log('You can connect using telnet: '+add + ' ' + port);
25  })
26
27  s.listen(port);
28
```

- How about something more non trival
  - https://github.com/juchi/tictactoe



"A" Benchmark

Java

Concurrent Requests	Average Response time (ms)	Requests/second
10	23	422
50	119	416
100	243	408
150	363	411

Node.js

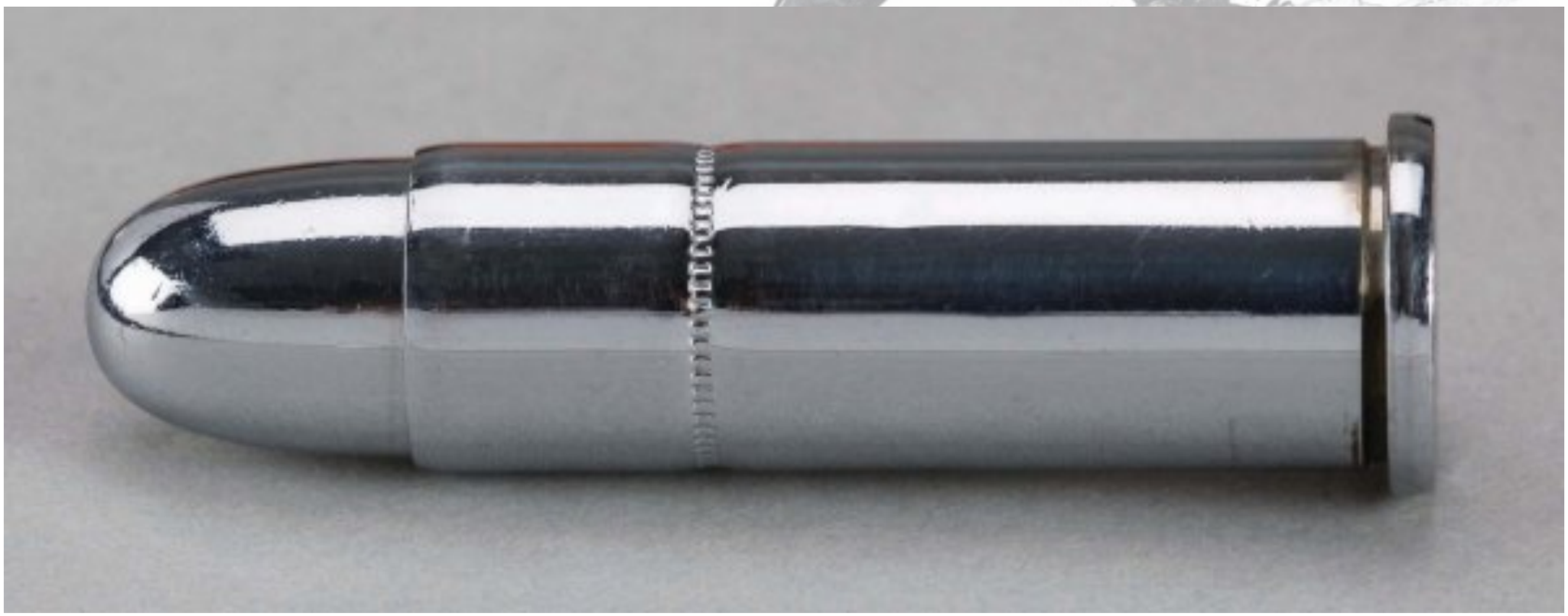
Concurrent Requests	Average Response time (ms)	Requests/second
10	19	509
50	109	453
100	196	507
150	294	506

The Node.js is 20% faster than the Java EE solution for the problem at hand. That amazed me. An interpreted language as fast as a compiled language on a VM in which years of optimization have gone into. Not bad at all!

http://java.dzone.com/articles/performance-comparison-between?page=0,1

- Other Stuff (yes, that's a technical term):
  - Node exposes the full debugger from V8 (see debugger)
  - It's fast... it's really fast.
  - Easy move for Web Developers... that are WEB developers.
  - It's JavaScript
  - Extremely active Community
  - It has massive velocity!
  - Shared NOTHING model... if you want to talk to other instances, you have to open a socket and send it data. This means SCALE.
  - Express.js is the web framework of choice

- Now the Bad:
  - It's a start from scratch. If you don't have a core driver.... you have to write one. ...or the ones that are out there are crap (not my words).
  - Node.js is NOT immature anymore, but is still evolving fast. Careful of the bleeding edge
  - Don't have a stack trace to follow back with as in threaded applications. You have short stacks in node. Can be difficult to see what called the error point.
  - Managing state requires something like gossip or redis



WESLEY REISZ

- Resources:
  - http://www.youtube.com/watch?v=jo_B4LTHi3I
  - http://java.dzone.com/articles/performance-comparison-between
  - https://github.com/wesreisz/nodeSamples