node.JS

Async Evented I/O and Server-side Javascript

AGENDA

- Who am I?
- The Web is Changing
- Why Node.js?
- What is Node.js?
- Node.js is growing exponentially
- Blocking and Non-Blocking (Oh my!)
- Apache vs. Nginx
- Digging Deeper
- Lets build a Web Service
- Questions?

WHO AM I?

- Lifelong Hacker / Programmer / Engineer
- Founder of Nodejitsu
- Node.js veteran (since 0.1.98)
- All Node.js -- All the time

THE WEB IS CHANGING

- The demand for real-time web applications is increasing dramatically.

- Web applications are becoming more distributed through service oriented architecture and availability of cloud computing.

- Rich, highly responsive user interfaces are becoming the standard, not the exception.

WHY NODE.JS?

- Lightning-fast server-side Javascript environment built on top of Google's V8.

- Innovative programming model that enables high concurrency with little effort from the developer.

- Dual-sided Javascript programming lowers the barrier to entry for novice developers and increased code-reuse from the back-end to the front-end.

WHAT IS NODE.JS

- Node.js is server-side Javascript
- Node.js is asynchronous event driven I/O
- Node.js is an active community of contributors and developers working at the speed of thought

NODE.JS IS GROWING EXPONENTIALLY

- Posts to the nodejs and mailing lists are up ~400%
- Activity in the nodejs IRC room is up ~800%
- Javascript is now the most popular language on GitHub

BLOCKING AND NON-BLOCKING (OH MY!)

Traditional I/O is broken

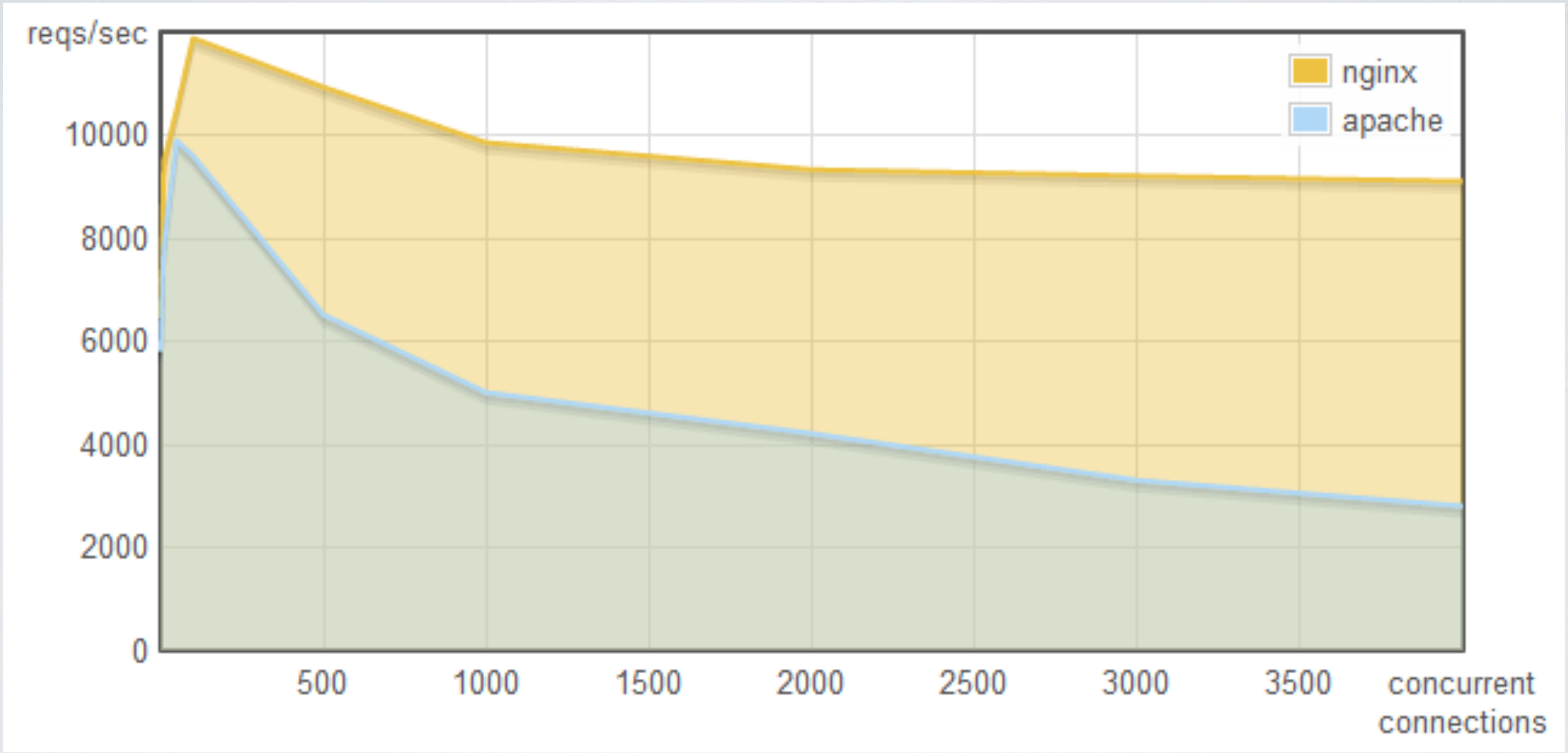
```
// blocking I/O + threads
var urls = db.query("select * from urls"); // wait

urls.each(function (url) {
  var page = http.get(url); // wait
  save(page); // wait
});
```

I/O WILL COST YOU

- L1: 3 cycles
- L2: 14 cycles
- RAM: 250 cycles
- DISK: 41,000,000 cycles
- NETWORK: 240,000,000 cycles
- **So why do we treat blocking and non-blocking I/O the same way when programming?**

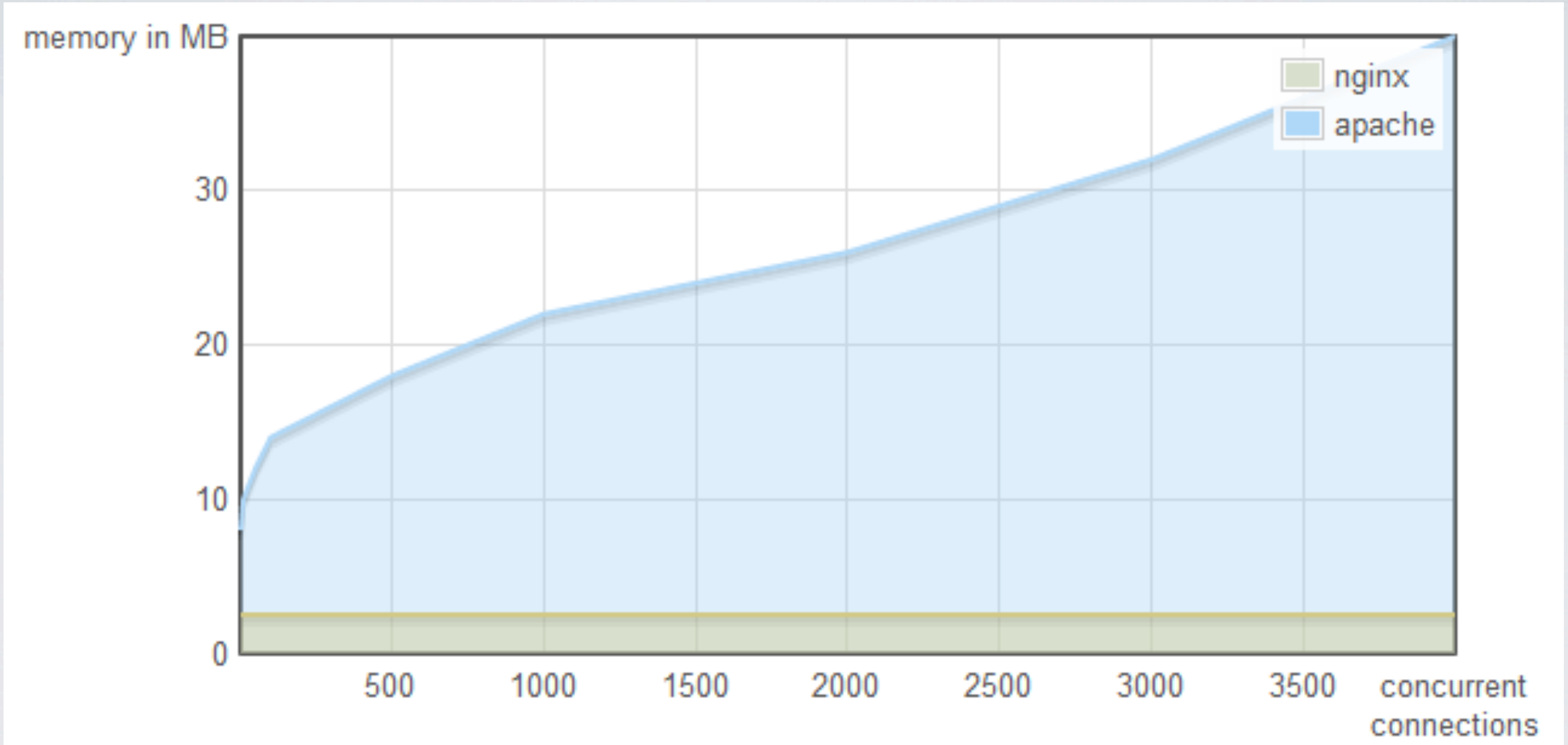
http://nodejs.org/jsconf.pdf

APACHE VS. NGINX



http://blog.webfaction.com/a-little-holiday-present

APACHE VS. NGINX



http://blog.webfaction.com/a-little-holiday-present

APACHE VS. NGINX

The difference?

- Apache uses one-thread per connection
- Nginx doesn't use threads. It uses an **event loop.**
- Node.js is built on-top of an event loop: **eio + lib_ev**

http://nodejs.org/jsconf.pdf

DON'T BLOCK THE PROC

```
// non-blocking I/O + event loop
db.query("select * from urls", function (urls) {
  urls.each(function (url) {
    http.get(url, function (page) {
      save(page);
    });
  });
});
```

DIGGING DEEPER

```
// A simple out-going HTTP request in node.js
var http = require('http');
var google = http.createClient(80, 'www.google.com');
var request = google.request('GET', '/', {
  host: 'www.google.com'
});

request.on('response', function (response) {
  var data = '';
  console.log(response.statusCode); // 200
  response.on('data', function (chunk) {
    data += chunk;
  });

  response.on('end', function () {
    console.log(data); // Entire webpage from google.com
  });
});

request.end();
```

HELLO NODE

```
// A simple 'hello world' server in node.js
var sys = require('sys'),
    http = require('http');

http.createServer(function (req, res) {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('hello node');
}).listen(8000);

sys.puts('Hello node listening on http://127.0.0.1:8000')
```

Now lets benchmark it:

```
$ node hello-node.js
$ ab -c 200 -n 7000 http://127.0.0.1:8000/
```

http://github.com/indexzero/nodejs-intro

LETS BUILD A WEB SERVICE

- Writing a web service in Node.js is easy
- *RESTful* Bookmarking *API*
- Support for HTTP Basic Auth
- Backed by *CouchDB*
- Code Available: http://github.com/indexzero/nodejs-intro

LETS USE SOME LIBRARIES

- Routing: https://github.com/cloudhead/journey
- CouchDB: https://github.com/cloudhead/cradle
- Logging: https://github.com/indexzero/winston
- Tests: https://github.com/cloudhead/http-console

PAUSE FOR CODE

CITATIONS

This presentation couldn't have been possible without:

http://www.slideshare.net/jacekbecela/introduction-to-nodejs

http://nodejs.org/jsconf.pdf

http://www.yuiblog.com/blog/2010/05/20/video-dahl/

QUESTIONS?