Node Program

Introduction

np

Node.js version: 5.1

Last updated: Feb 2016

Before We Start...

You'll need:

- Node.js and npm
- Code editor
- Command line
- Internet connection
- Slides & sample code

You may also want/need a local data store!

- MongoDB
- MySQL
- Postgresql

Introductions

Instructor: Azat Mardan

- Work: Capital One, Storify, FDIC, NIH, DocuSign
- Books: ReactQuickly, Full Stack JavaScript, Practical Node.js, Pro Express.js, Mongoose Course

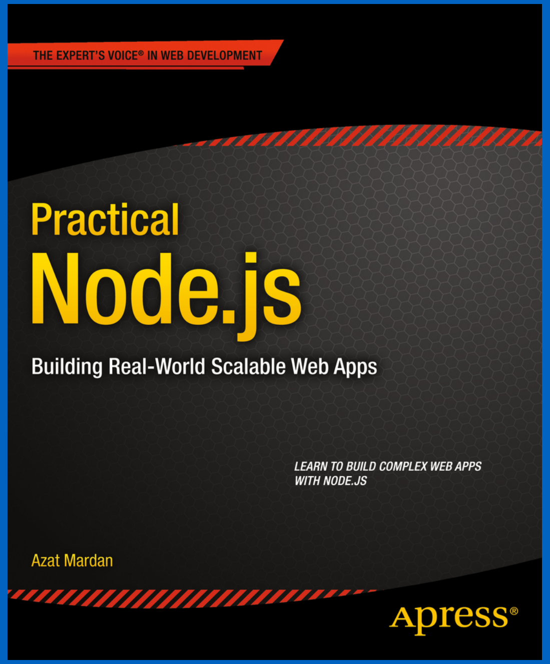
THE EXPERT'S VOICE® IN WEB DEVELOPMENT

Pro Express.js

MASTER EXPRESS.JS FOR YOUR DEVELOPMENT

Azat Mardan

Apress®



THE EXPERT'S VOICE® IN WEB DEVELOPMENT

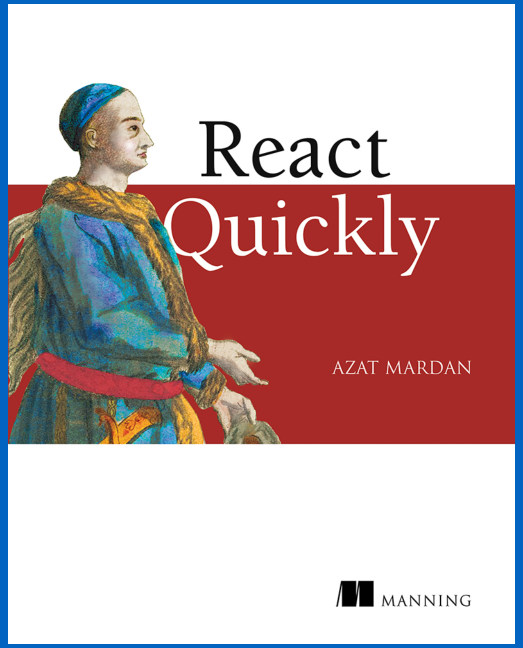
Practical Node.js

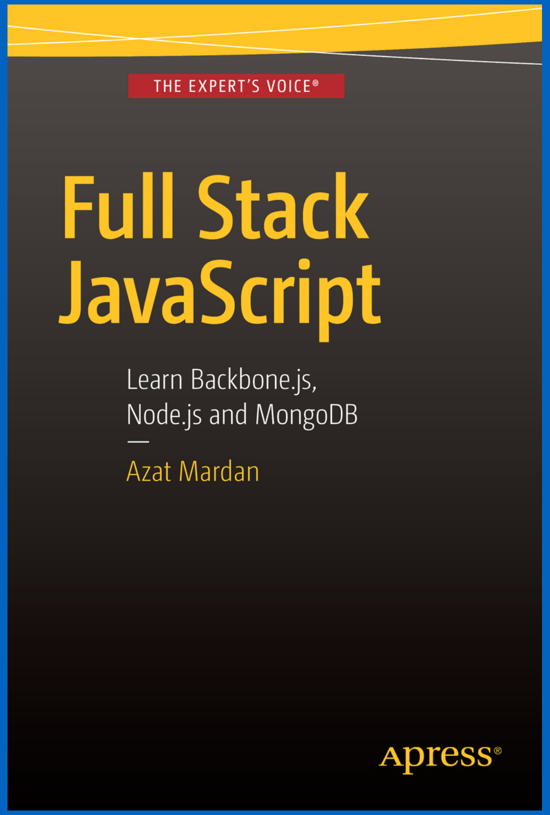
Building Real-World Scalable Web Apps

LEARN TO BUILD COMPLEX WEB APPS WITH NODE.JS

Azat Mardan

Apress®


React
Quickly
AZAT MARDAN
MANNING


THE EXPERT'S VOICE®
Full Stack
JavaScript
Learn Backbone.js,
Node.js and MongoDB
Azat Mardan
apress®

Introduction

Why Server-Side JavaScript?

Node was originally born out of this problem — how can you handle two things at the same time

— Ryan Dahl, The Creator of Node.js

Why Server-Side JavaScript?

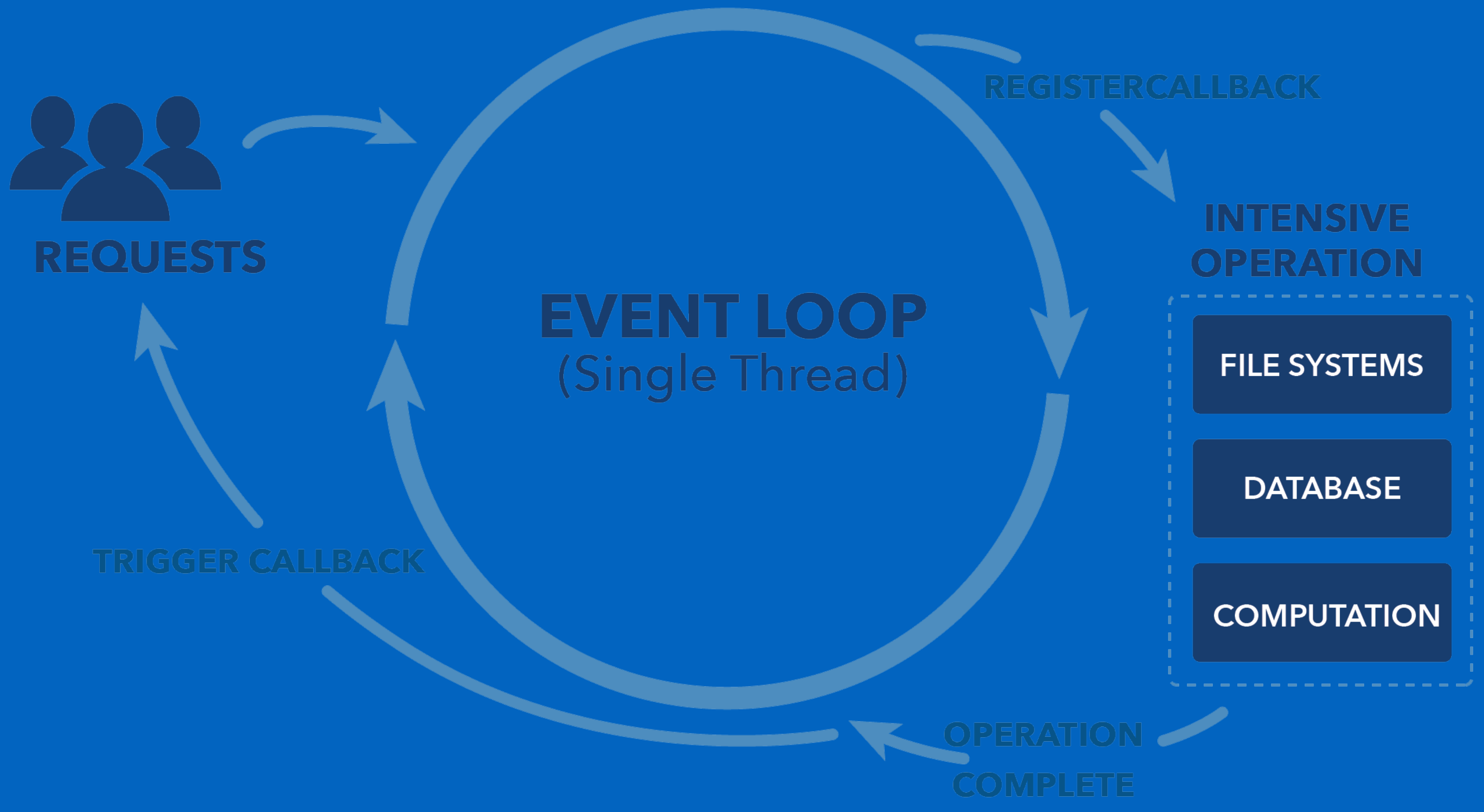
- Non-blocking I/O: performant
- Fast: browser arms race (V8)
- **One language across the stack**
- Expressive: don't waste time on setup
- Solid standard (ECMA)

Advantages of Node.js

- Non-blocking I/O
- Super fast (V8)
- Vibrant ecosystem (npm)
- Ability to re-use code on browser and server
- Ability to use front-end devs for back-end and vice versa

Non-blocking I/O

It's kind of a big deal



Disadvantages of Node.js

- Devs have to think in async and functional+prototypal
- Frameworks and tools are not as mature as in Ruby, Java, Python (yet)
- JavaScript "quirks" (mostly fixed in ES6!)

Node Gotcha

Don't use Node.js for CPU-intensive tasks. Hand them over to other workers.

Downsides of JavaScript (Not only Node)

- Callback Hell
- Prototypal inheritance

JavaScript is Optional in Node.js

It's **possible** to use other languages for Node.js that compile into JavaScript, e.g., CoffeeScript, TypeScript, and ClosureScript.

Nodies are not just Silicon Valley hipsters !

NODE IS DEPLOYED BY BIG BRANDS

Big brands are using Node to power their business

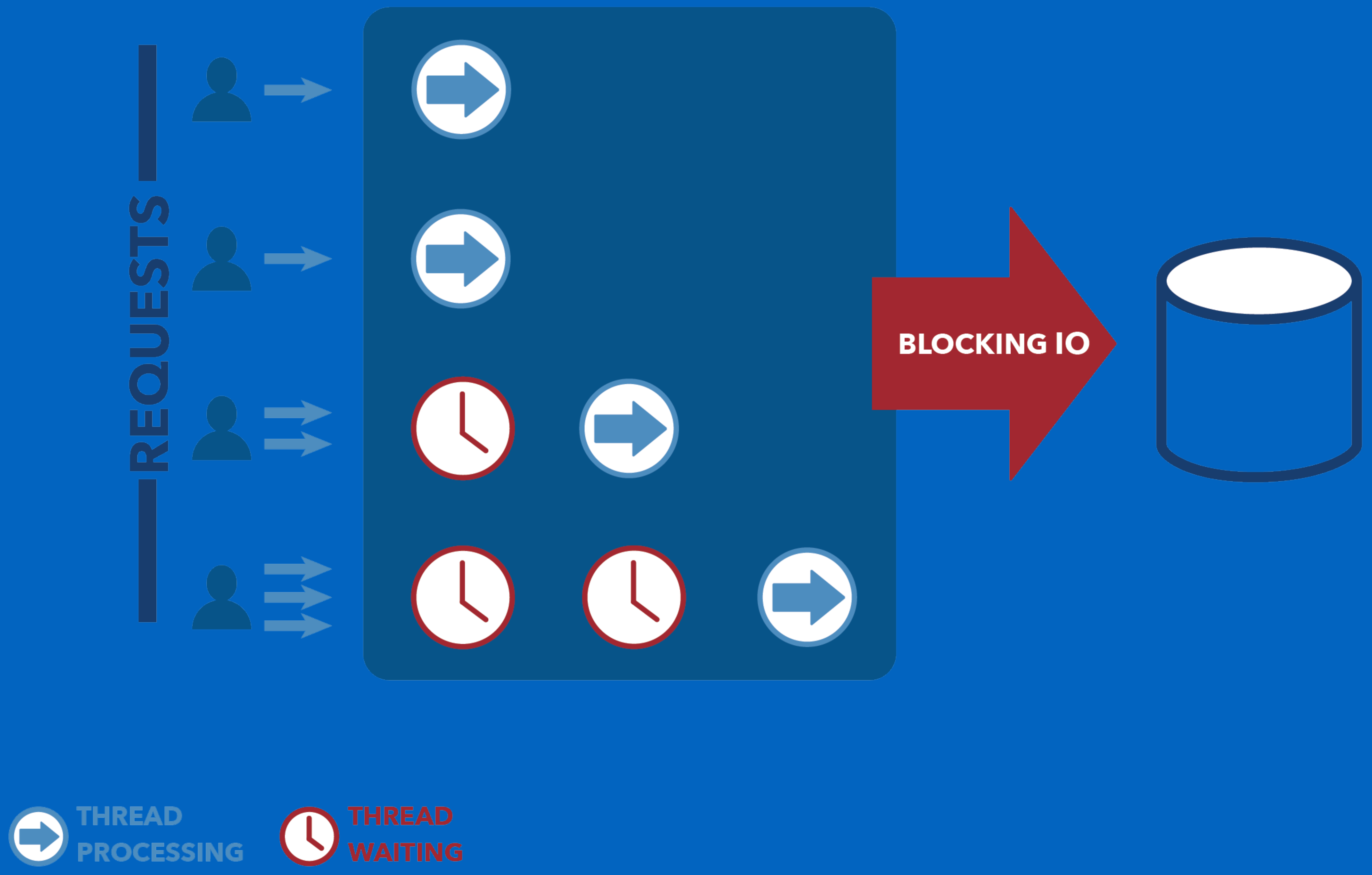
Manufacturing	Financial	eCommerce	Media	Technology
BMW	ADP	amazon.com	beatsMUSIC	salesforce.com
GE	citigroup	BEST BUY	CONDÉ NAST	Apple
General Motors	Goldman Sachs	ebay	DOWJONES	box
Johnson Controls	PayPal	TARGET	The New York Times	intel
SIEMENS	WELLS FARGO	Zappos.com	SONY	YAHOO!

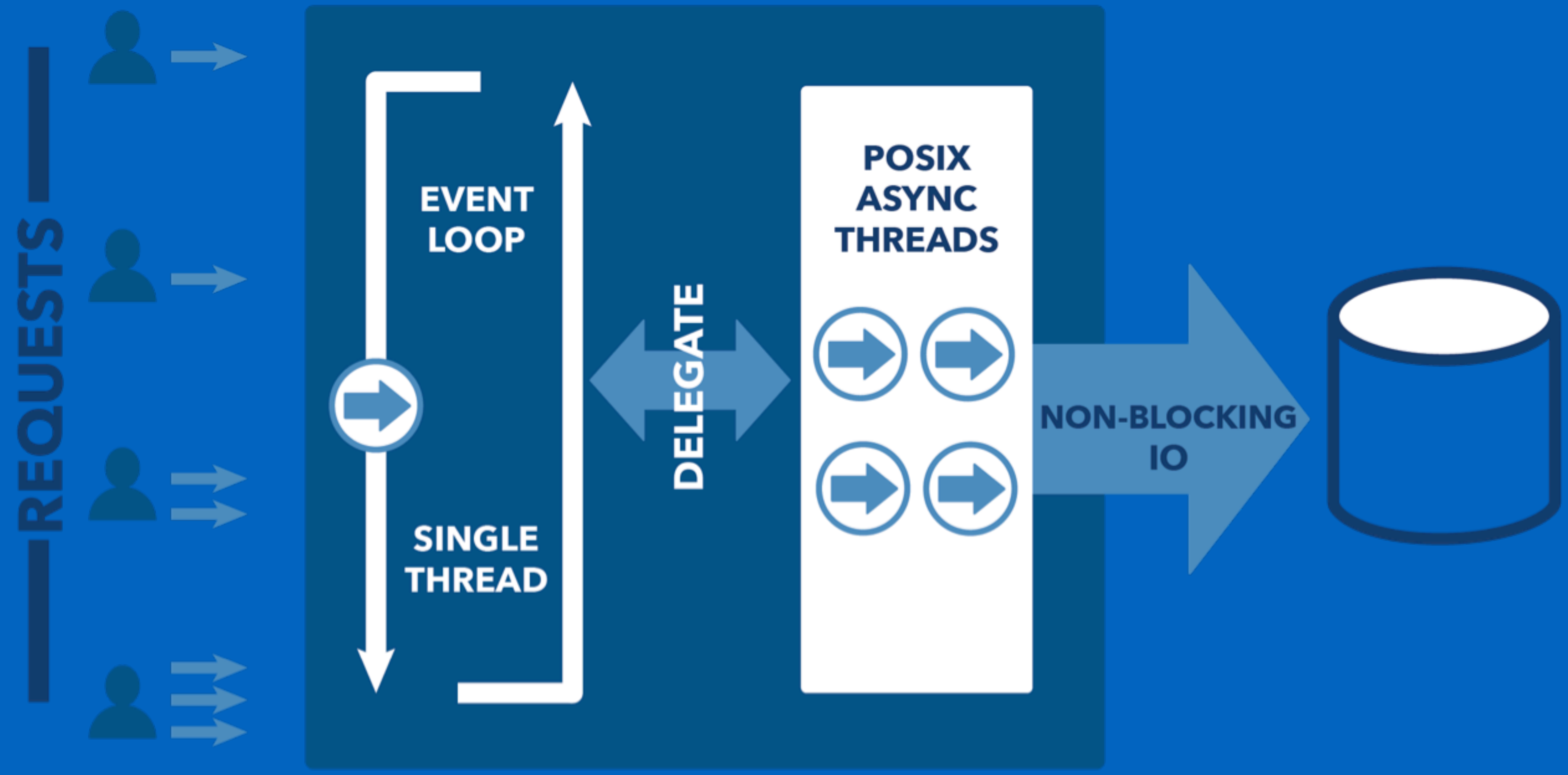
Node is Single-Threaded

Node.js is single-threaded by design to make asynchronous processing simpler. Multi-threading can be very complex: racing condition, deadlocks, priority inversions...

It turned out for web-based application, single-threaded asynchronous event-loop based non-blocking I/O is very performant!

MULTI THREADED SERVER




REQUESTS
EVENT LOOP
SINGLE THREAD
DELEGATE
POSIX ASYNC THREADS
NON-BLOCKING IO

The Coffee Bean
ORDER HERE
ORDER HERE
PICK-UP HERE
ICE BLENDED
NOW HIRING

STICKY FINGERS

Scaling Node Vertically

To scale Node vertically, you can take advantage of multiple CPUs cores or compute units (multi-threading) with clustering (e.g., StrongLoop's PM).

The idea is to have multiple processes from the same code base to listen on the same port for requests.

Integration

- noSQL
- SQL
- OAuth 1.0/2.0
- REST
- SOAP

Databases

- mySQL
- Postgresql
- Oracle
- MS SQL
- MongoDB
- Cassandra

Node + Client MVC Architecture

Single-Page Applications a.k.a. BYOC: REST API in Node + SPA

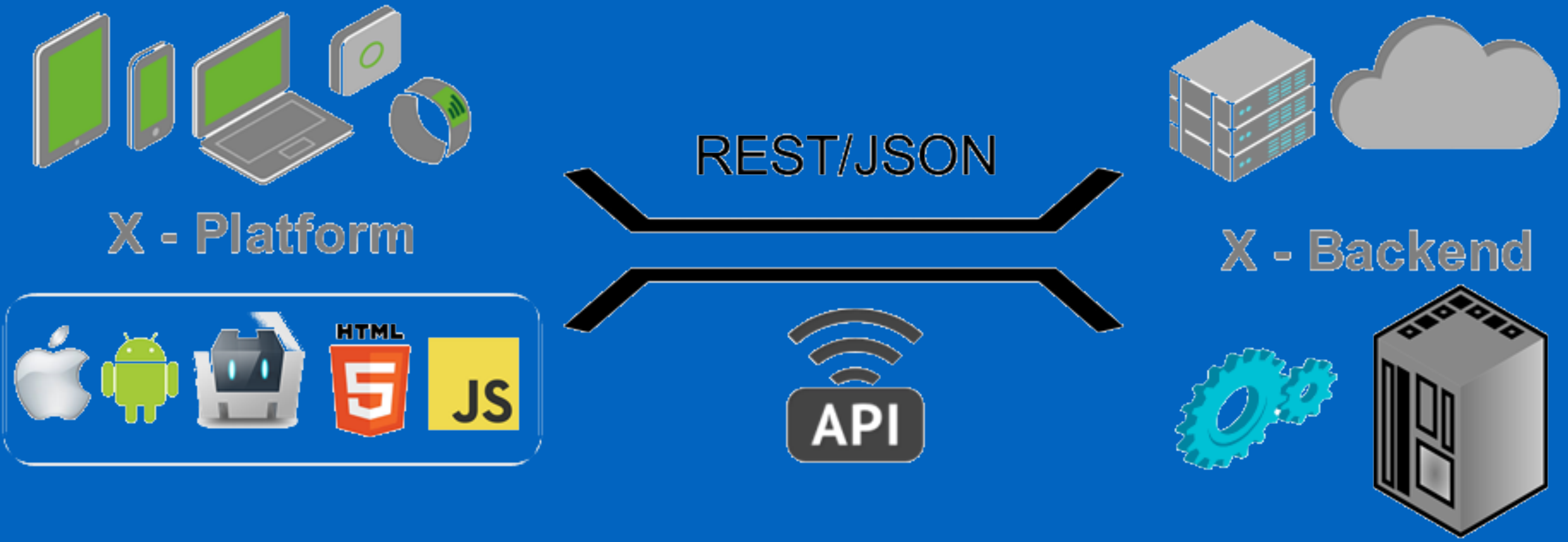
- Backbone
- Angular (e.g., M.E.A.N)
- Ember
- React
- MV*

Server-side Rendering

- Jade
- Handlebars
- EJS
- Hogan

Many more: http://garann.github.io/template-chooser

Node for SOA / REST



So what is ECMAScript?

ES as a Language Specification

- Browser implementations (like Chrome's V8)
- Node builds on V8 with C++

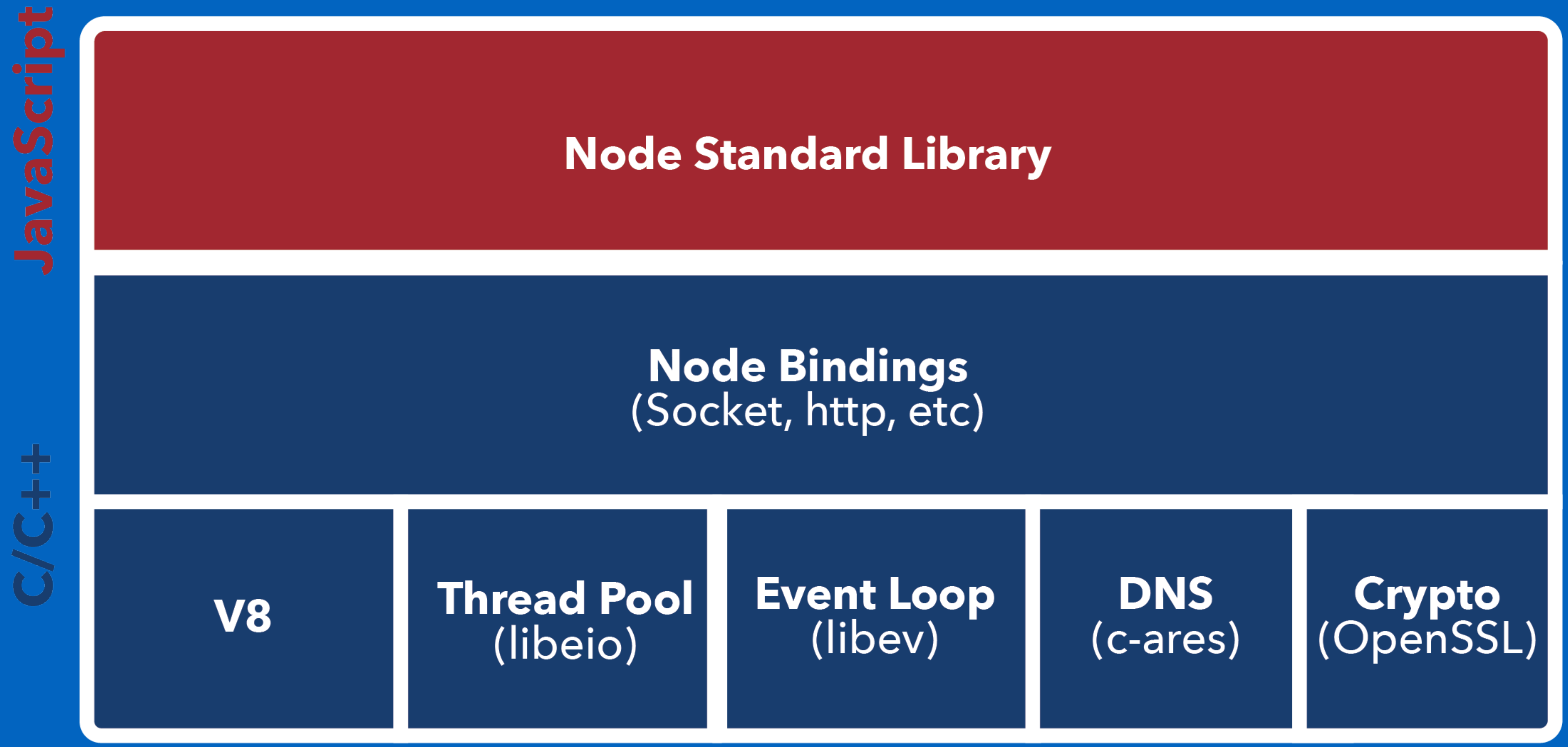
Browser JS != Node

- Modules
- Scopes
- window vs. global and process
- `fs` and other modules

Node Core: V8, libev, and libeio

- Libev: The event loop
- LibEio: Async I/O
- LibUv: Abstraction on libEio, libev, c-ares (for DNS) & iocp (for Windows)

Node Core Architecture



JavaScript

C/C++

Node Standard Library

Node Bindings
(Socket, http, etc)

V8

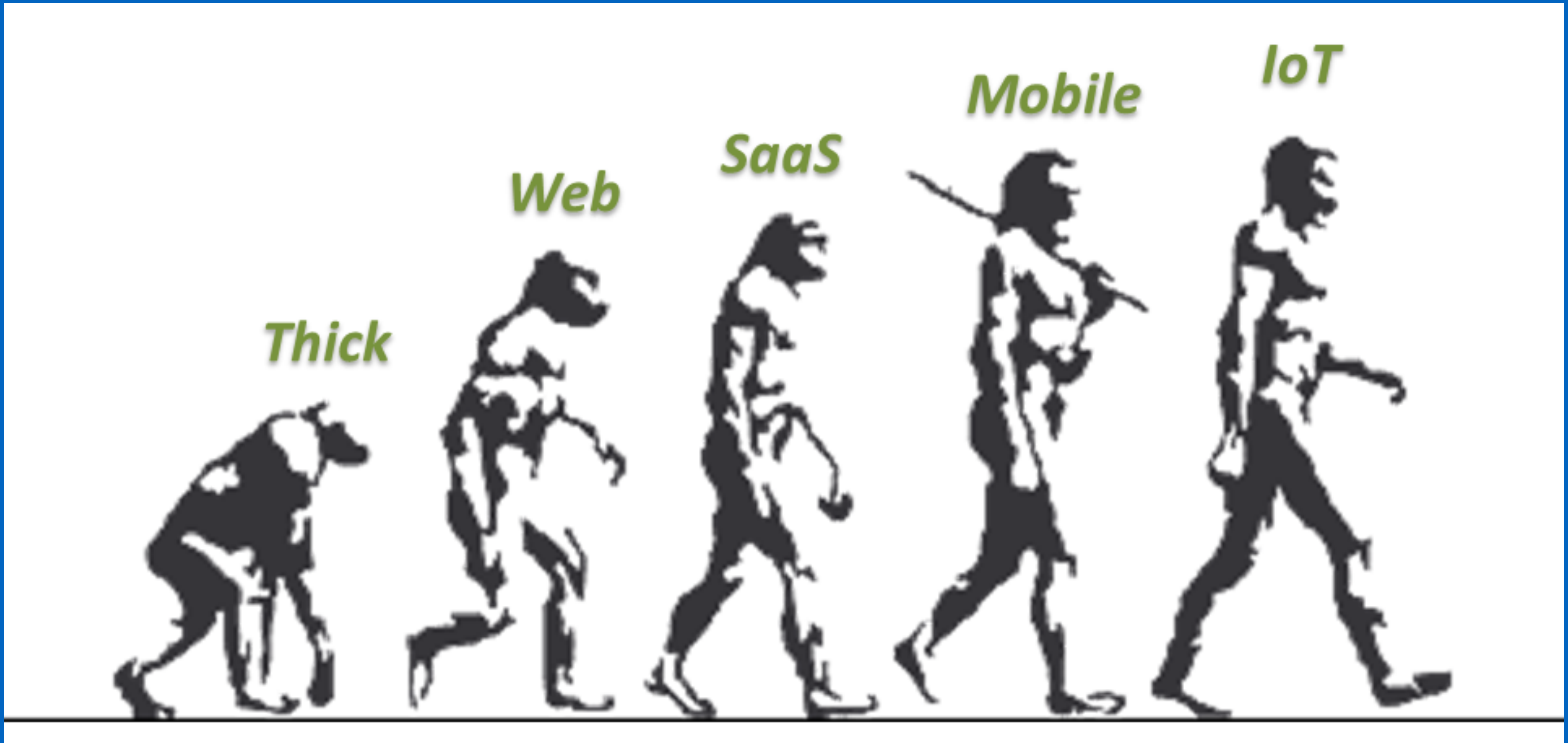
Thread Pool
(libeio)

Event Loop
(libev)

DNS
(c-ares)

Crypto
(OpenSSL)

Patterns Evolve to Serve Market Needs



Framework Categories

- KISS Servers: small core, small modules
- Convention: follow the leader, steep learning curve
- Configuration: open path, manual effort for advanced
- ORM & Isomorphic: model-driven, shared code, steep learning

Framework Examples

- KISS Servers: Node core
- Convention: Restify, Total.js
- Configuration: Express, Hapi, Kraken
- ORM & Isomorphic: LoopBack, Sails, Meteor*

Node Program

Effective Learning

50% workshops +

50% lectures +

50% discussions

(yes, we deliver 150%!)

workshops = coding + collaboration + pair programming + solo programming + discussions + reading + solving problems

Questions and Exercises

Discussion forum

✋ if stuck

👍

No workshop for this lesson. 😁