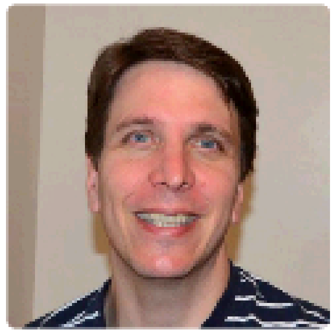
Enterprise, Cloud-Ready Node.js

Michael Dawson
IBM Community Lead for Node.js

Agenda Key: 41ah

About Michael Dawson
IBM Community Lead for Node.js

- Active Node.js community member
  - Collaborator
  - Node.js Technical Steering Committee TSC Chair
  - Community Committee member
  - Working group(s) member/leadership



- Twitter: @mhdawson1
- GitHub: @mhdawson
- Linkedin: https://www.linkedin.com/in/michael-dawson-6051282

Agenda

- Why Node.js ?
- Node.js deep dive
- Positioning versus Java™
- Node.js community
- IBM involvement

Why Node.js ?

- What is it ?
- Ecosystem
- Productivity
- Performance

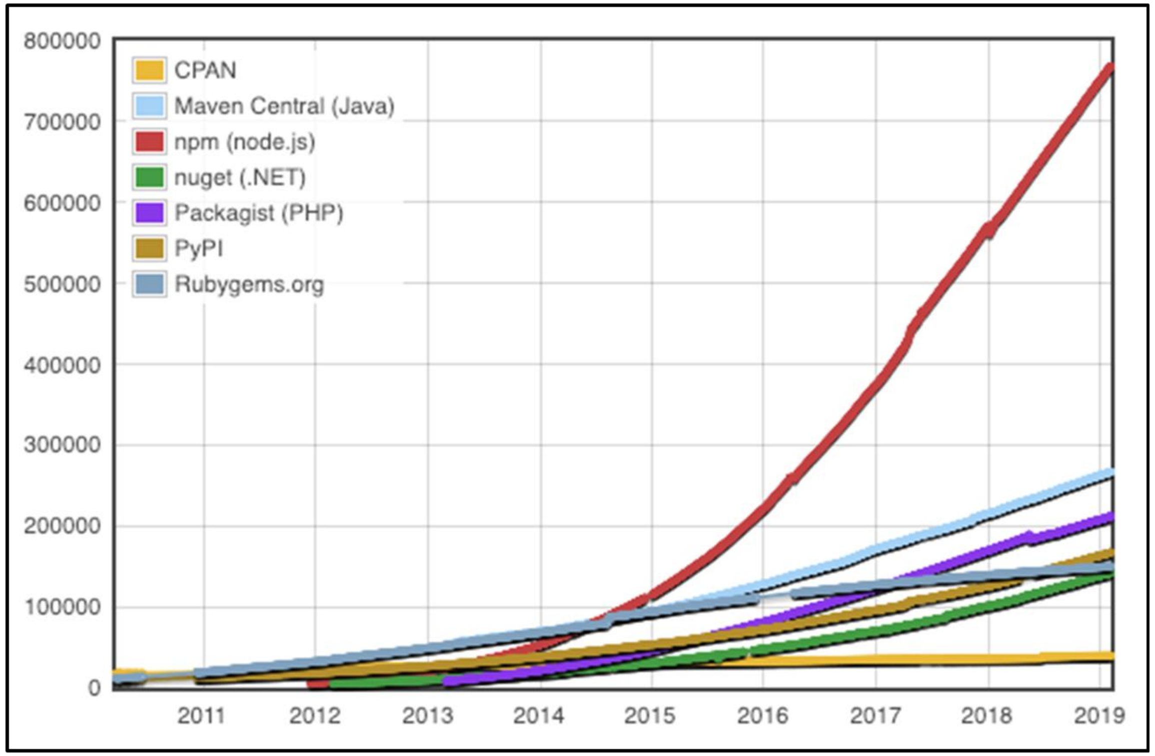
Why Node.js – What is it?

- JavaScript != Java
- Node.js = **Server-side** JavaScript
  - Event-oriented
  - Non-blocking
  - Asynchronous

Why Node.js ? – Ecosystem

- There is a module for that
  - 700K modules +
  - #1 on module counts

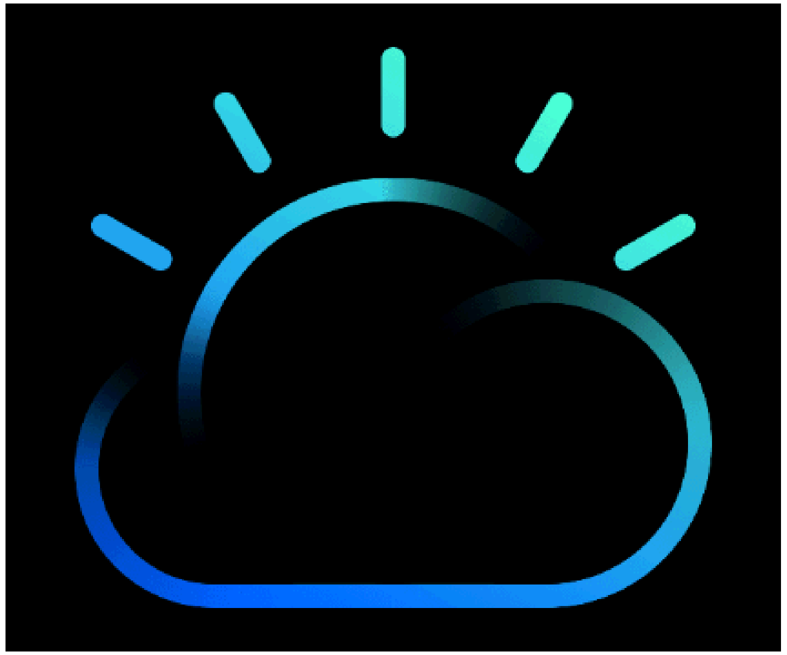
- #1 on Github (#projects)

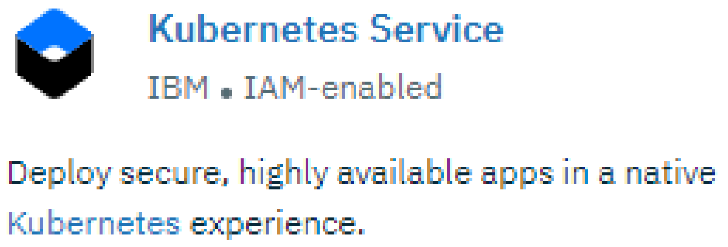


http://www.modulecounts.com/

Why Node.js ? – Ecosystem

- Most used runtime in IBM Cloud (and others)

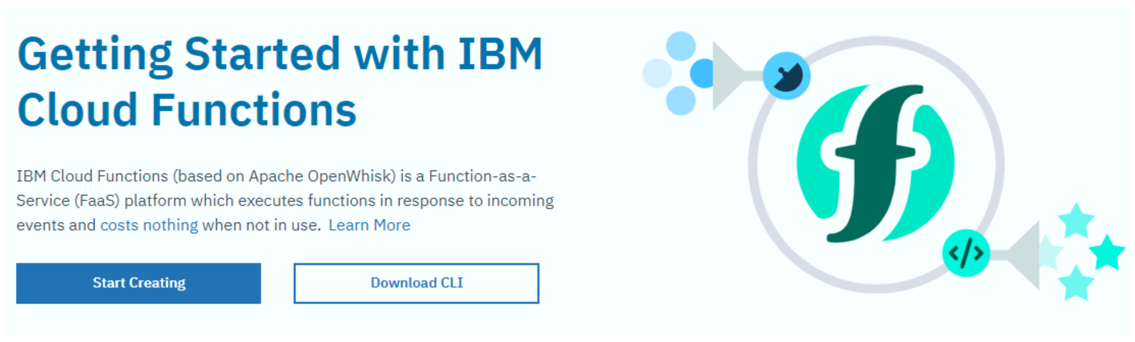




Kubernetes Service

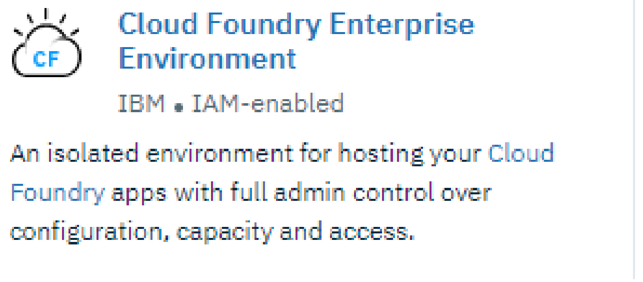
IBM • IAM-enabled

Deploy secure, highly available apps in a native Kubernetes experience.



Getting Started with IBM Cloud Functions

IBM Cloud Functions (based on Apache OpenWhisk) is a Function-as-a-Service (FaaS) platform which executes functions in response to incoming events and costs nothing when not in use. Learn More



Cloud Foundry Enterprise Environment

IBM • IAM-enabled

An isolated environment for hosting your Cloud Foundry apps with full admin control over configuration, capacity and access.

Why Node.js ? – Productivity

- **Faster** development **less code**

- **PayPal - https://www.paypal-engineering.com/2013/11/22/node-js-at-paypal/**
  - **Took 1/2 time with less people**
  - **33% fewer lines of code**
  - **40% fewer files**

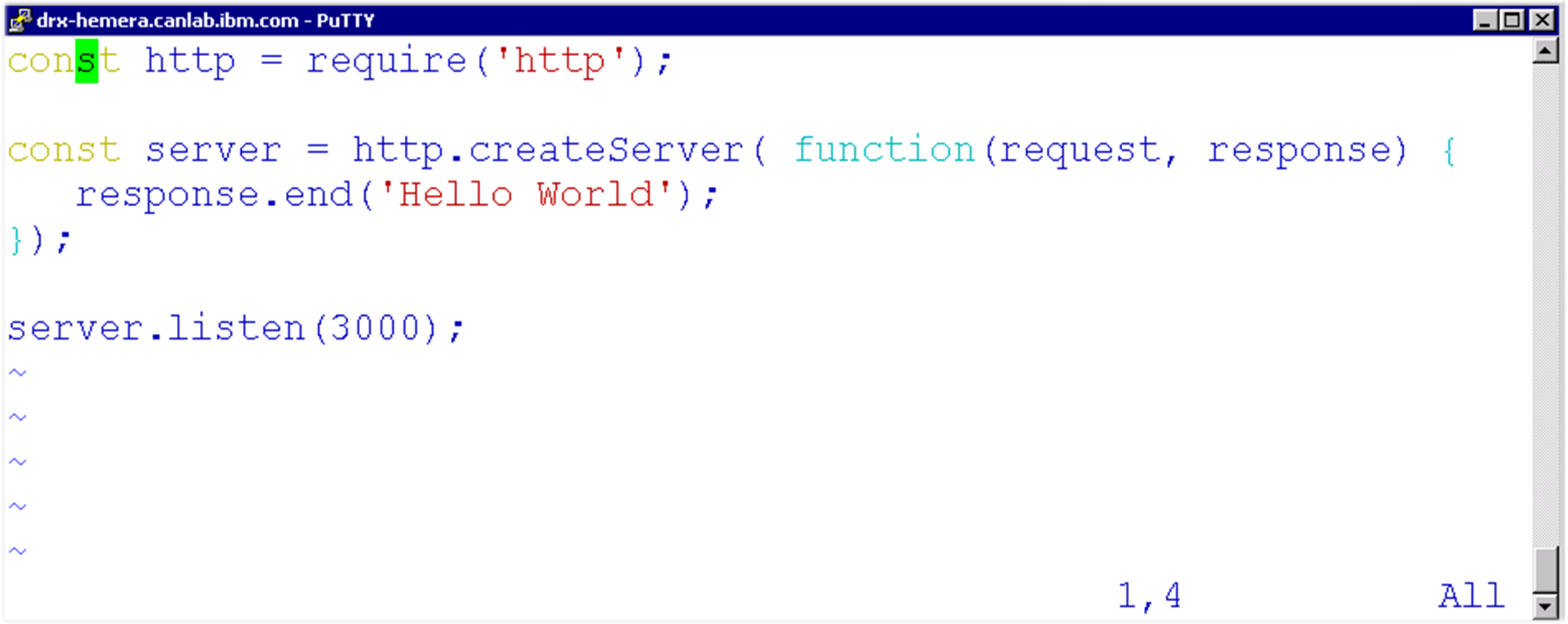
- **NextFlix - http://www.infoworld.com/article/2610110/javascript/paypal-and-netflix-cozy-up-to-node-js.html**

"We're used to working in JavaScript all day long. Having Node just makes it feel like a very natural extension of our work environment,"

Why Node.js ? – Productivity

- Reuse of “isomorphic” code components
- Availability of JavaScript talent
- Developer satisfaction

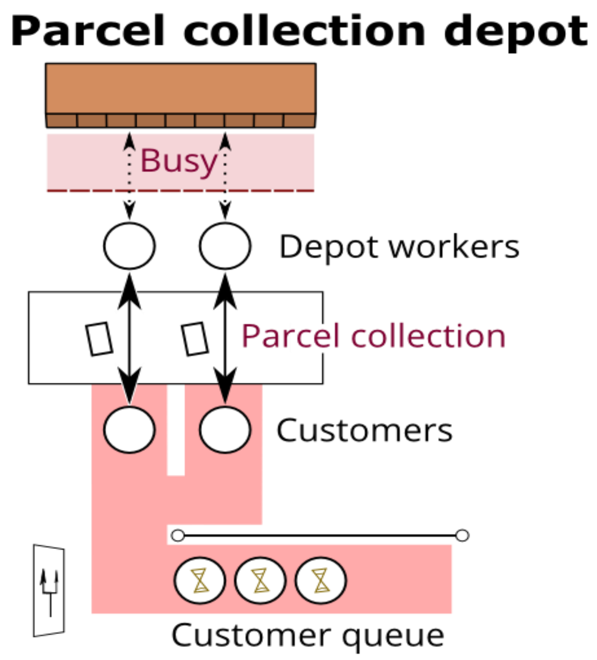
Why Node.js ? – Productivity

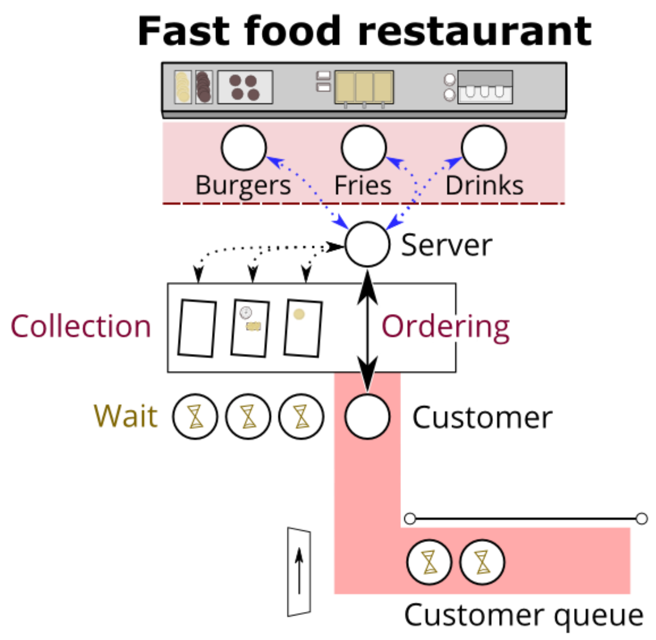


```
const http = require('http');

const server = http.createServer( function(request, response) {
   response.end('Hello World');
});

server.listen(3000);
~
~
~
~
~
                                                    1,4            All
```

Why Node.js ? - Performance

Event based: perfect fit for asynchronous non-blocking I/0





Why Node.js ? - Performance

- Thousands of concurrent connections

- PayPal - https://www.paypal-engineering.com/2013/11/22/node-js-at-paypal/
    - **Double** number of requests/sec
    - Response times 35% **lower**

- Groupon – http://www.nearform.com/nodecrunch/node-js-becoming-go-technology-enterprise/
    - Reduced page load times by 50%

Enterprises Seem to Agree

Node.js 2017 User Survey (http://tinyurl.com/ycma6xjs)

Node.js – Deep Dive

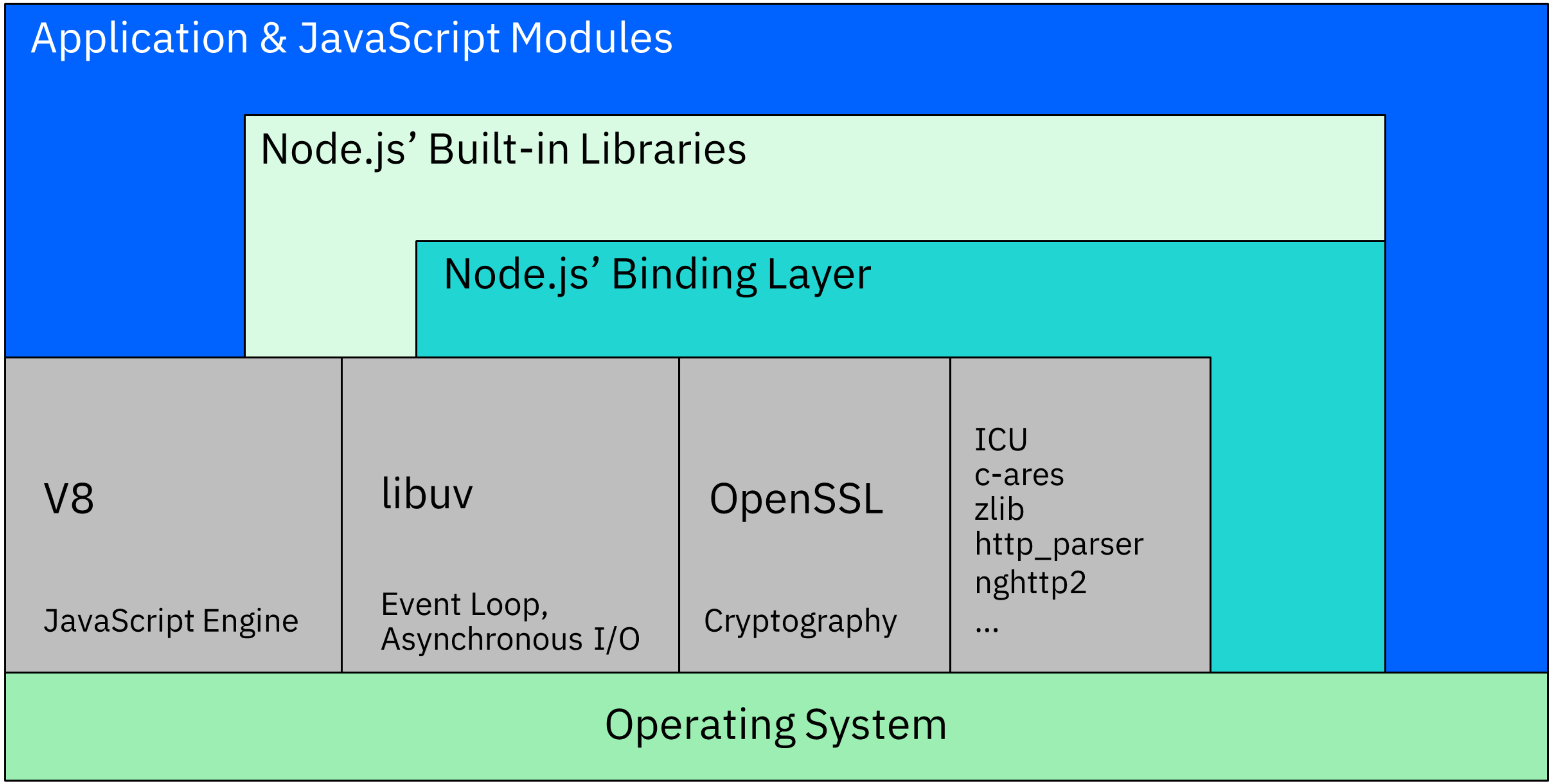
- Key characteristics
- Components
- Programing model
- Event loop
- Native code
- Common use cases

Node.js – Deep Dive – Key Characteristics

- Small (IBM i RPM)
  - Download **20 Mb**
- Fast startup
  - 60 ms
- Small footprint
  - 18 MB

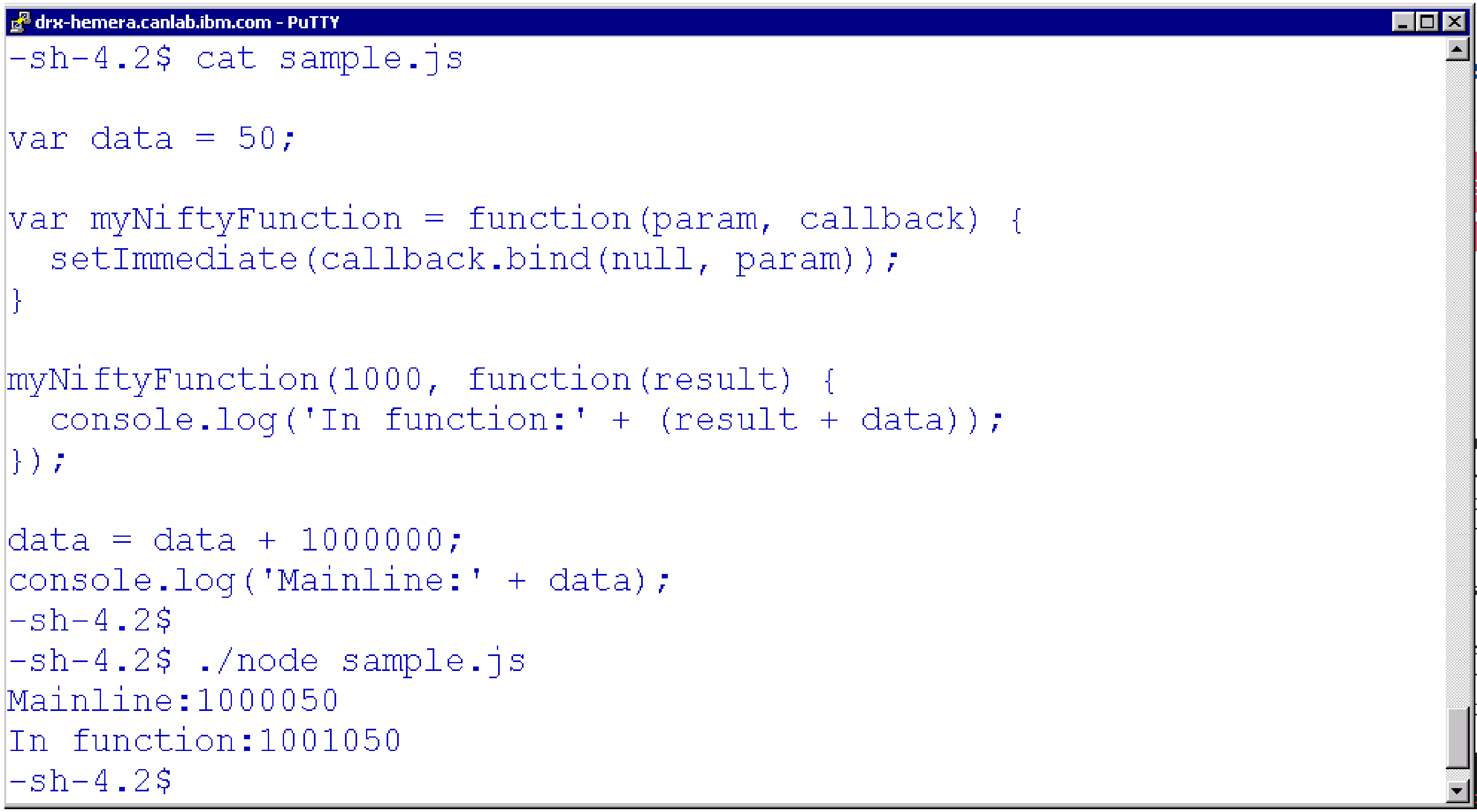
https://benchmarking.nodejs.org/

Node.js – Deep Dive - Components



Node.js – Deep Dive - Programming Model

- Dynamic
- Functional
- Asynchronous
- Event Based



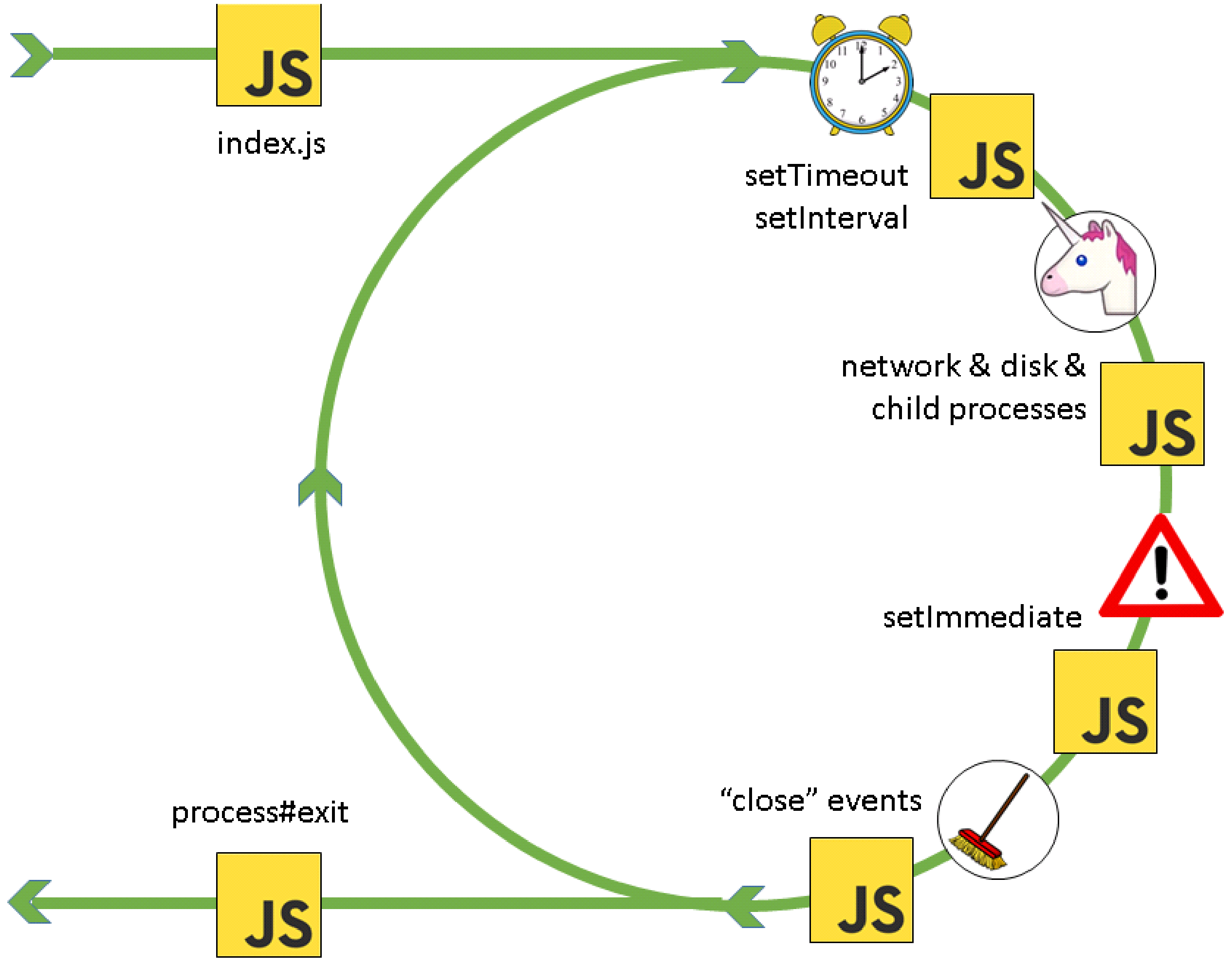
```
-sh-4.2$ cat sample.js

var data = 50;

var myNiftyFunction = function(param, callback) {
  setImmediate(callback.bind(null, param));
}

myNiftyFunction(1000, function(result) {
  console.log('In function:' + (result + data));
});

data = data + 1000000;
console.log('Mainline:' + data);
-sh-4.2$
-sh-4.2$ ./node sample.js
Mainline:1000050
In function:1001050
-sh-4.2$
```

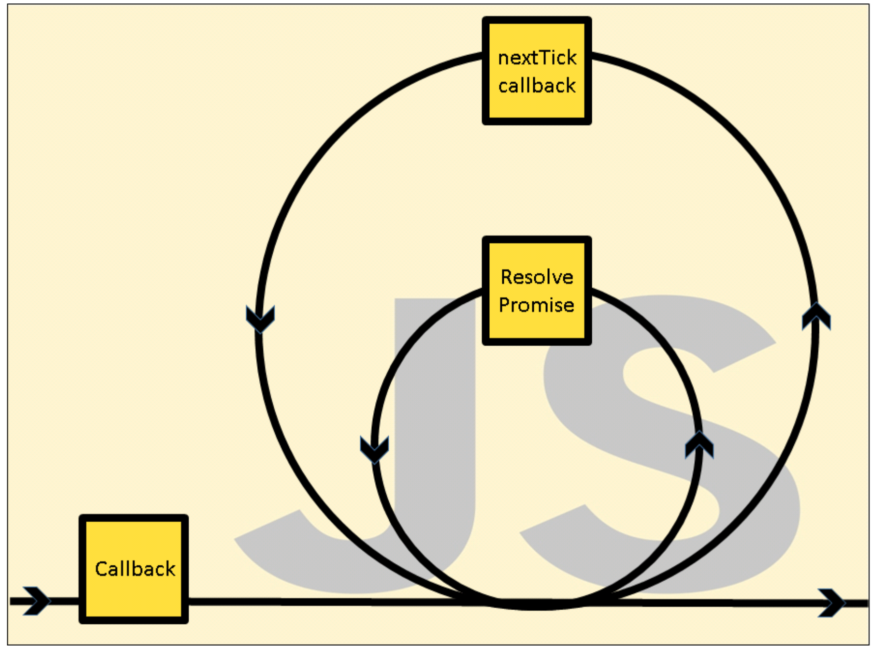
Node.js – Deep Dive - Programming Model

- Event Based

```
var http = require('http');

var server = http.createServer();
server.listen(8080);

server.on('request', function(request, response) {
               response.writeHead(200, {"Content-Type": "text/plain"});
               response.write("Hello World!\n");
               response.end();
});

server.on('connection', function(socket) {});
server.on('close', function() {});
server.on('connect', function(socket) {});
server.on('upgrade', function(request, socket, head) {});
server.on('clientError', function(exception, socket) {});
```

Node.js – Deep Dive – Event Loop





Node.js – Deep Dive – Native Code

N-API

```
#include <node_api.h>
#include <assert.h>

napi_value Method(napi_env env, napi_callback_info info) {
  napi_status status;
  napi_value world;
  status = napi_create_string_utf8(env, "world", 5, &world);
  assert(status == napi_ok);
  return world;
}

#define DECLARE_NAPI_METHOD(name, func) \
  { name, 0, func, 0, 0, 0, napi_default, 0 }

napi_value Init(napi_env env, napi_value exports) {
  napi_status status;
  napi_property_descriptor desc = DECLARE_NAPI_METHOD("hello", Method);
  status = napi_define_properties(env, exports, 1, &desc);
  assert(status == napi_ok);
  return exports;
}

NAPI_MODULE(NODE_GYP_MODULE_NAME, Init)
```

node-addon-api

```
#include <napi.h>

Napi::String Method(const Napi::CallbackInfo& info) {
  Napi::Env env = info.Env();
  return Napi::String::New(env, "world");
}

Napi::Object Init(Napi::Env env, Napi::Object exports) {
  exports.Set(Napi::String::New(env, "hello"),
              Napi::Function::New(env, Method));
  return exports;
}

NODE_API_MODULE(hello, Init)
```

https://github.com/nodejs/node-addon-examples

Node.js – Deep Dive – Native Code

```
var addon = require('bindings')('hello');
console.log(addon.hello()); // 'world'
```

https://github.com/nodejs/node-addon-examples

Node.js – Deep Dive – NPM

- 700,000+ modules!!

- Two types of installs:
  - Global: use for command-line utilities
  - Local (default): use for application dependencies

- Fully encapsulates:
  - Dependency list within package.json file
  - Dependencies themselves within node_modules/ directory

- Advantages:
  - Each application can operate independently
  - No global settings (extensions directory, classpaths, etc) to maintain
  - Portable

Node.js – Deep Dive – NPM

```
$ mkdir expressjs_app && cd expressjs_app
$ npm install express
express@4.12.0 node_modules/express
├── utils-merge@1.0.0
├── methods@1.1.1
├── fresh@0.2.4
├── merge-descriptors@0.0.2
├── cookie-signature@1.0.6
├── escape-html@1.0.1
├── range-parser@1.0.2
├── cookie@0.1.2
├── finalhandler@0.3.3
├── vary@1.0.0
├── content-type@1.0.1
├── parseurl@1.3.0
├── content-disposition@0.5.0
├── serve-static@1.9.1
├── path-to-regexp@0.1.3
├── depd@1.0.0
├── on-finished@2.2.0 (ee-first@1.1.0)
├── qs@2.3.3
├── debug@2.1.1 (ms@0.6.2)
├── proxy-addr@1.0.6 (forwarded@0.1.0, ipaddr.js@0.1.8)
├── etag@1.5.1 (crc@3.2.1)
├── send@0.12.1 (destroy@1.0.3, ms@0.7.0, mime@1.3.4)
├── type-is@1.6.0 (media-typer@0.3.0, mime-types@2.0.9)
└── accepts@1.2.4 (negotiator@0.5.1, mime-types@2.0.9)
```

holds meta-data about application

Node.js – Deep Dive – NPM

```
$ npm init
```

Creates file `package.json`

```
{
  "name": "expressjs_app",
  "version": "0.0.0",
  "description": "",
  "main": "app.js",
  "dependencies": {
    "express": "^4.12.0"
  },
  "devDependencies": {},
  "author": "Michael Dawson",
  "license": "MIT"
}
```

Installs these modules when `npm install` is run.

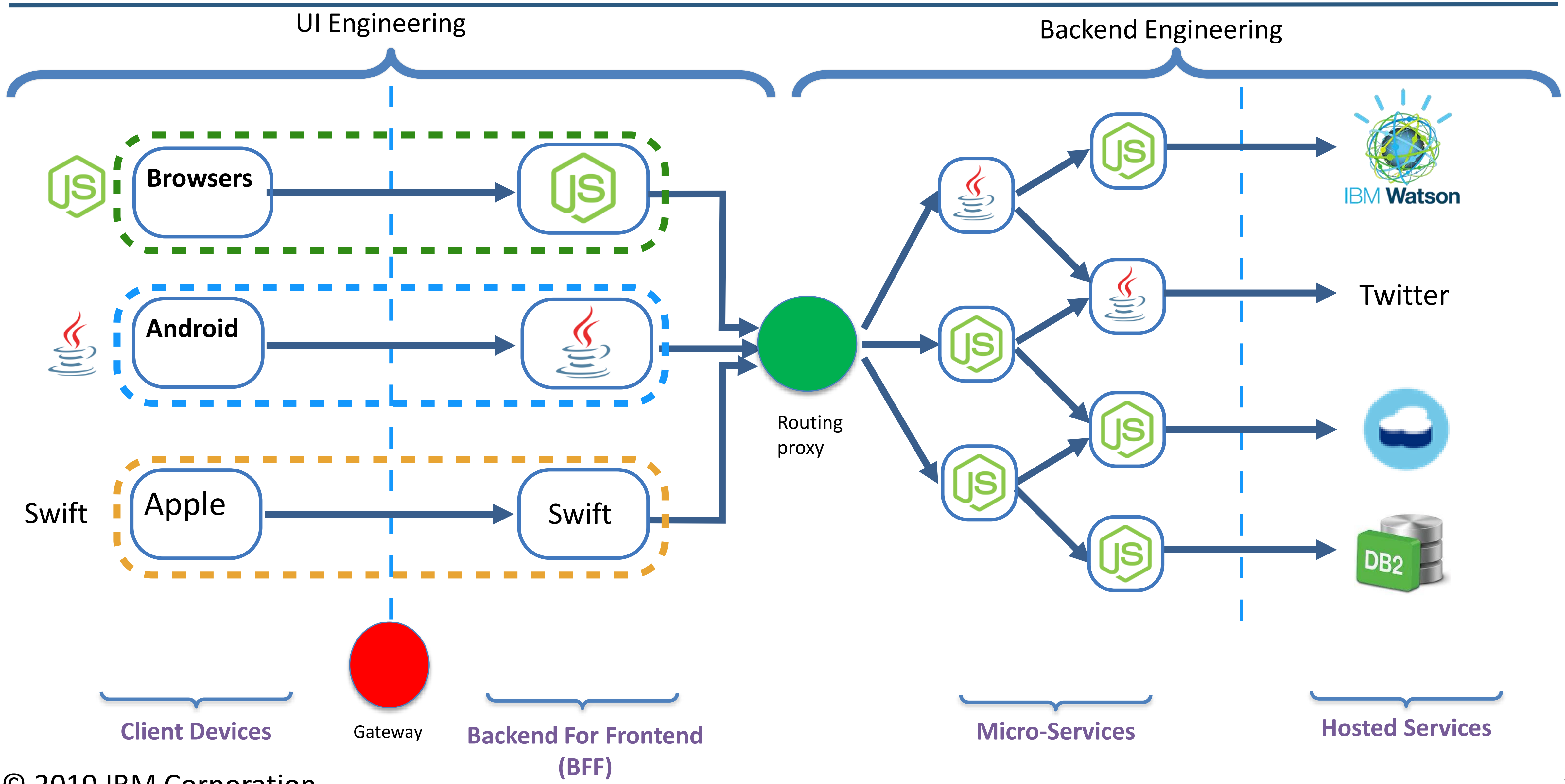
docs.npmjs.com/cli/init - package.json creation
docs.npmjs.com/files/package.json - Docs
browsenpm.org/package.json - Easier docs

Node.js – Deep Dive – Use Cases

https://github.com/nodejs/benchmarking/blob/master/docs/use_cases.md

- Back-end API services
- Service oriented architectures (SOA)
- Microservice-based applications
- Generating/serving dynamic web page content
- SPA applications with bidirectional communication over WebSockets and/or HTTP/2
- Agents and data collectors
- Small scripts

Node.js With Java



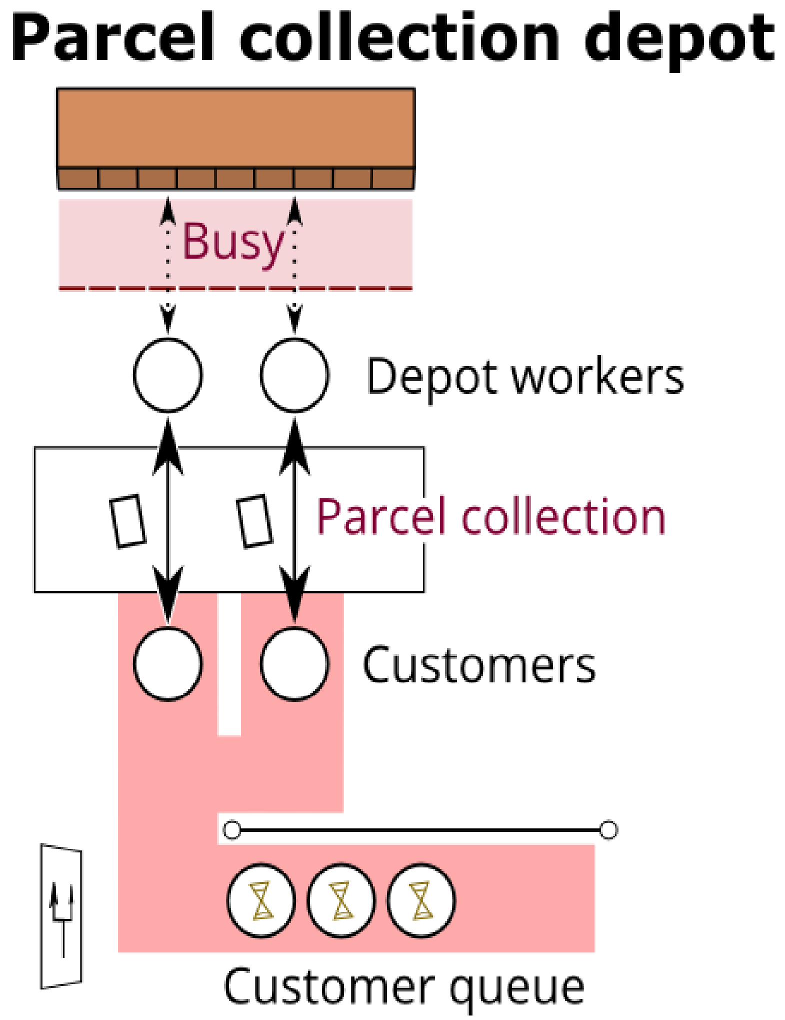
Node.js With Java

- Strengths and weaknesses
- Choosing the right language
- Hybrid applications

Node.js With Java – Scaling with Java

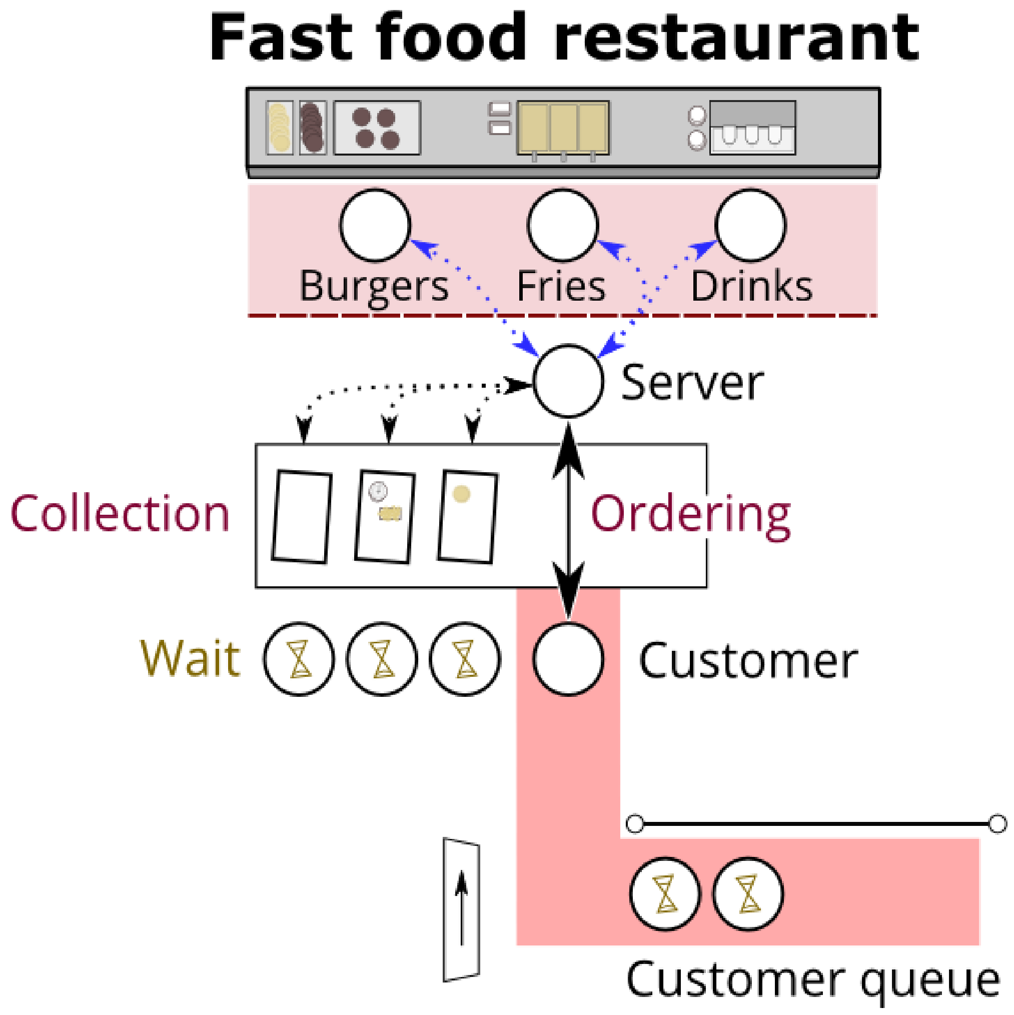
▪

- One thread (or process) per connection
  - Each thread waits on a response
  - Scalability determined by number of threads
- Each thread:
  - Consumes memory
  - Is relatively idle
- Concurrency determined by number of depot workers

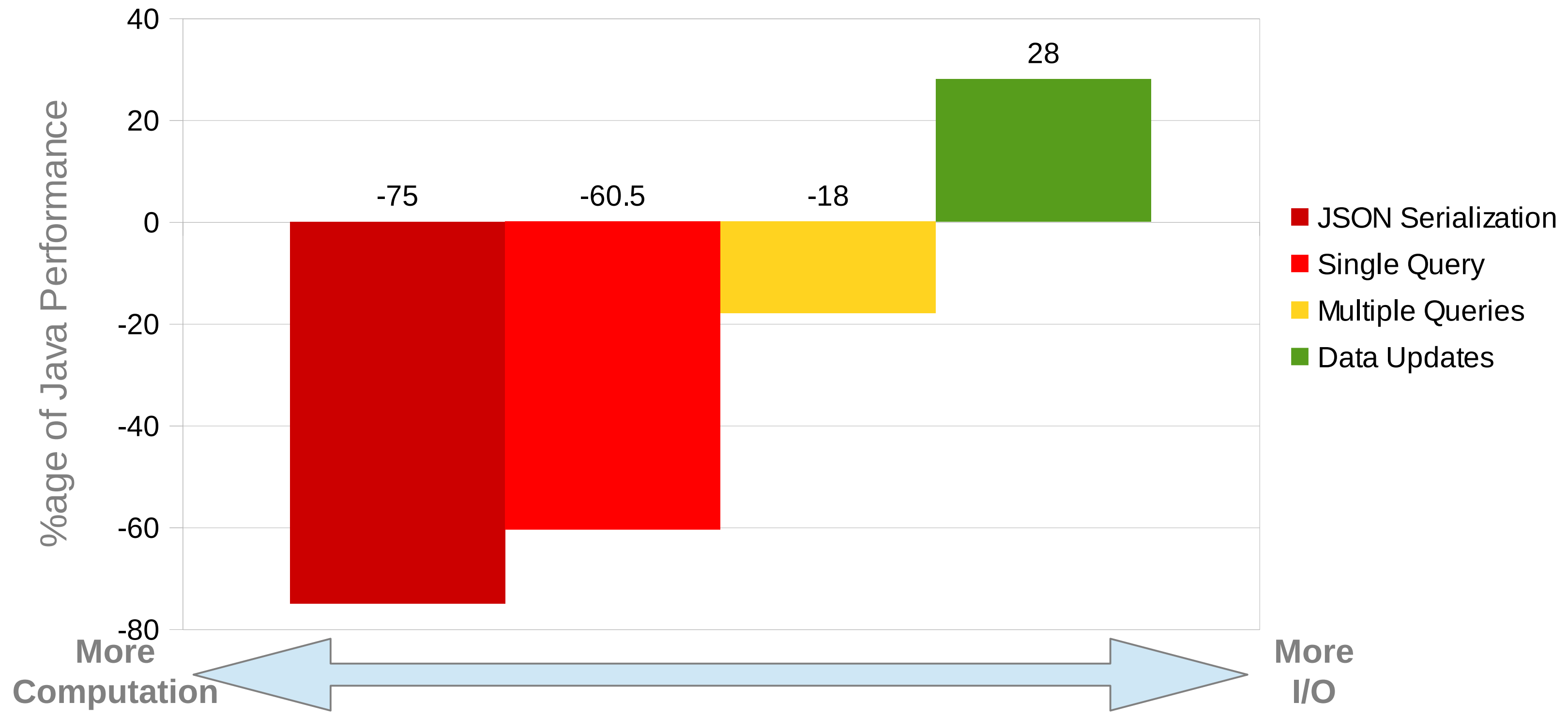


Node.js versus Java – Scaling with Node.js

- One thread multiplexes for multiple requests
  - No waiting for a response
  - Handles return from I/O when notified
- Scalability determined by:
  - CPU Usage
  - "Back end" responsiveness
- Concurrency determined by how fast the food server can work



Node.js With Java– Tradeoffs



Node.js With Java – Choosing the Right Language

- Higher performance for I/O
- Easier async programming
- Fullstack/isomorphic development

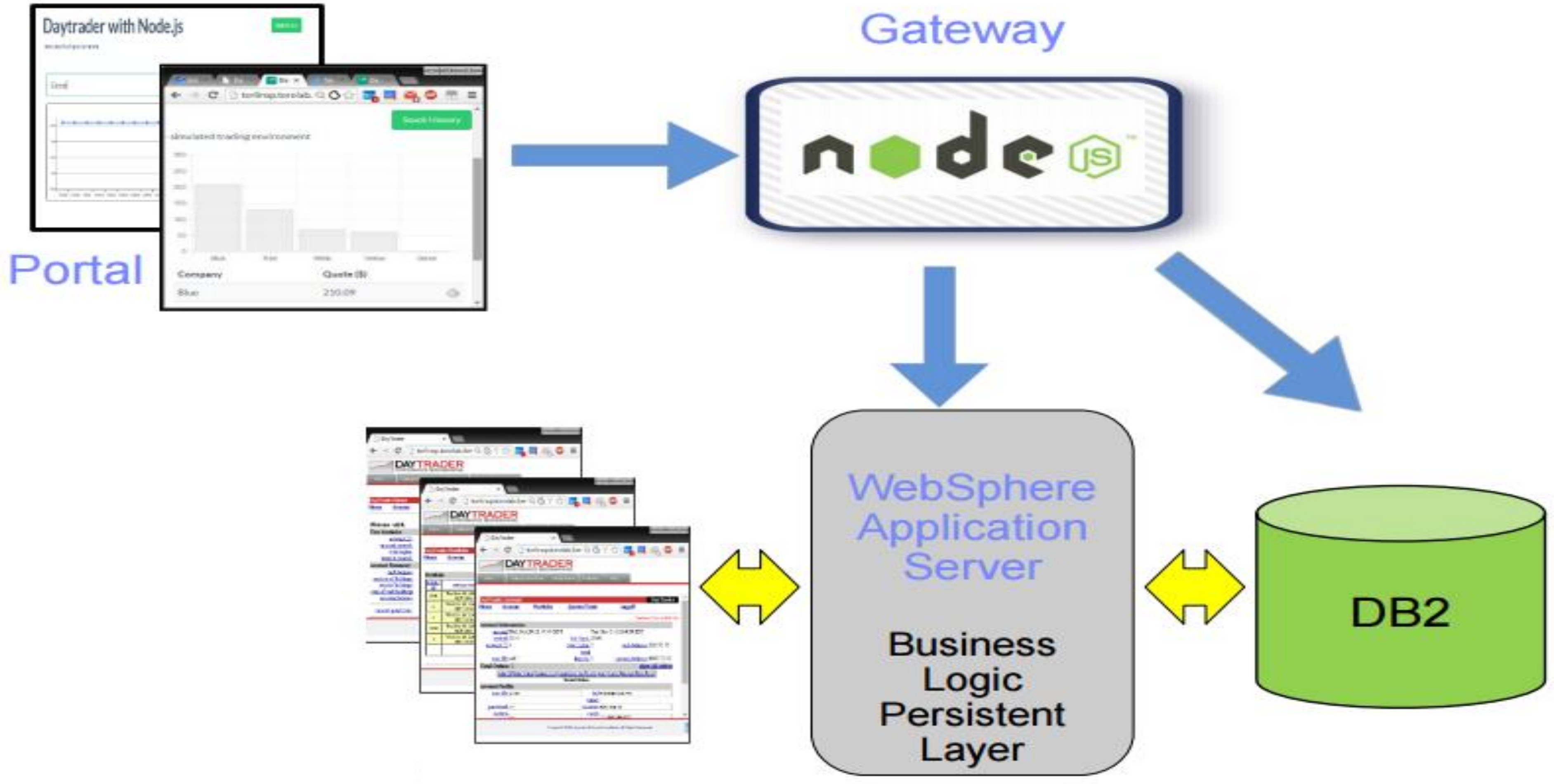
Node.js versus Java – Choosing the Right Language

- Higher processing performance
- Type safety for calculations
- Rich processing frameworks

Node.js With Java– Choosing the Right Language

- Highly performant, scalable rich web applications
- Highly performant, reliable transaction processing
- Self-contained micro-service components

Node.js With Java– Hybrid applications



Node.js Community

- History
- IBM’s Contribution
- Foundation

Node.js Community - History

- **2009 – written by Ryan Dhal (10 Year anniversary this year at JSConfEU !)**

  Jan 2010 - npm

  Sep 2010 – Joyent sponsors Node.js

  June 2011 – Windows support

  2012 – 2014 – Hand over to Isaac Schlueter, then Timothy J. Fontaine

  December 2014 – io.js fork

  June 2015 – Node.js Foundation (https://foundation.nodejs.org)

  Oct 2015 – Node.js 4.x unites io.js/node.js 0.12.x lines

  Oct 2016, Oct 2017,Oct 2018 – Node.js 6.x, 8.x, 10.x

  Oct 2018, Intent to merge Node.js and JS foundation

  March 2019 – OpenJS Foundation announced (https://openjsf.org/)

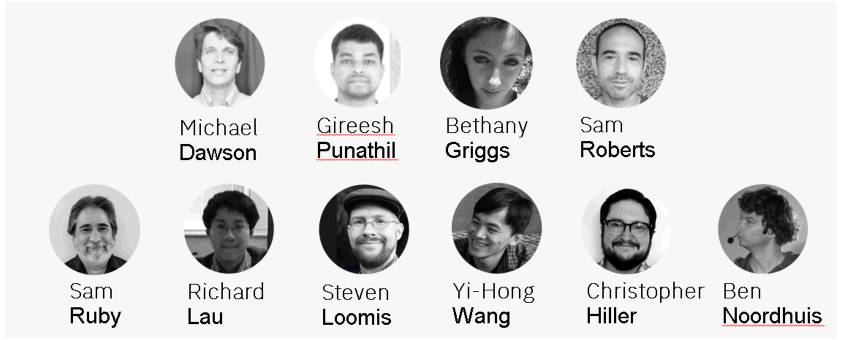
IBM’s Involvement

- Deep expertise at V8
- Developed ports to IBM Platforms
- Contribution back to official V8 repositories: https://github.com/v8/v8
  - **PPC**: V8 4.3 and later have full functional PPC implementation
  - **s390**: V8 5.1 and later have full functional implementatio1n
  - ~10-15 commits per week to V8 to maintain PPC/zlinux port
- Internal port for z/OS and IBM I
  - Working through cycle to contribute to community

IBM's Involvement

- Key Contribution to mending fork
- Platinum Sponsors
  - Founding member of Node.js Foundation
    - Todd Moore is Chair of Board Node.js Board
- Involved in day-to-day Leadership
  - 2 TSC members, 2 Community Committee members
  - 10 Core Collaborators
  - Active in many/most working groups
  - Significant code commits

Core Collaborators



IBM's Involvement – OpenJS foundation

- Part of bootstrap team - https://github.com/openjs-foundation/bootstrap
  - Helped to Shape OpenJS baseline governance

- Volunteering for CPC

IBM Involvement - Address the Challenge for Every Existing Enterprise:

How to make the old work with the new?

Traditional IT

On Prem

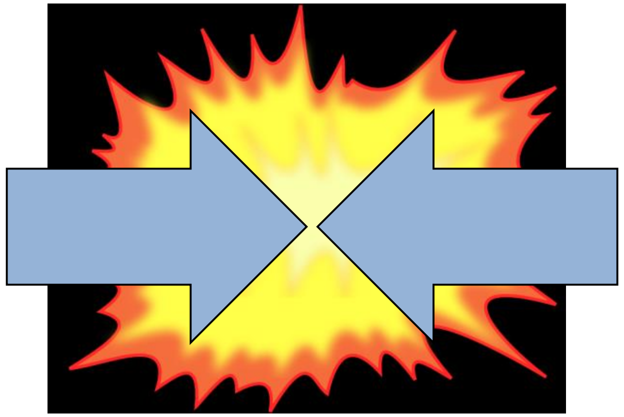
Packaged Apps

SOA / Monolithic

Relational DB

Waterfall

Java / .NET / C# / Other



New IT

Cloud

SaaS

Microservices / APIs

Relational & Non-Relational

DevOps

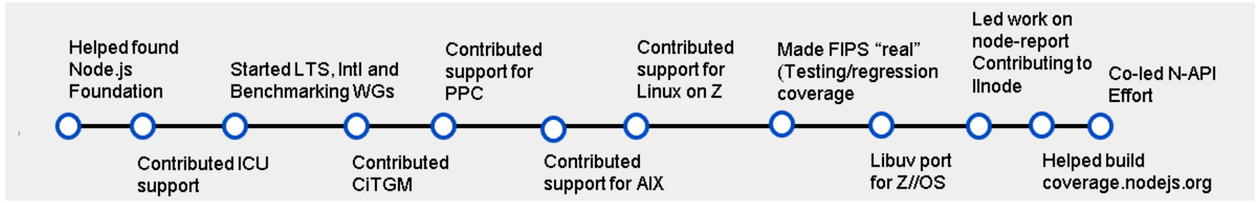
Node / SWIFT / Other

IBM Node.js Strategy

- Enterprise Ready Runtime
- Production Enablement
- Simplify Module Consumption
- Production Support

Enterprise Ready Runtime

- Embrace and Improve Community Runtime
  - Engage and lead
  - Develop expertise and influence
  - Platinum member of Node.js Foundation
- Enterprise Focus
  - Stable and Predictable releases
  - Platform support
  - Security
  - Diagnostics
  - Performance
  - Code quality and safety net
  - Key Features

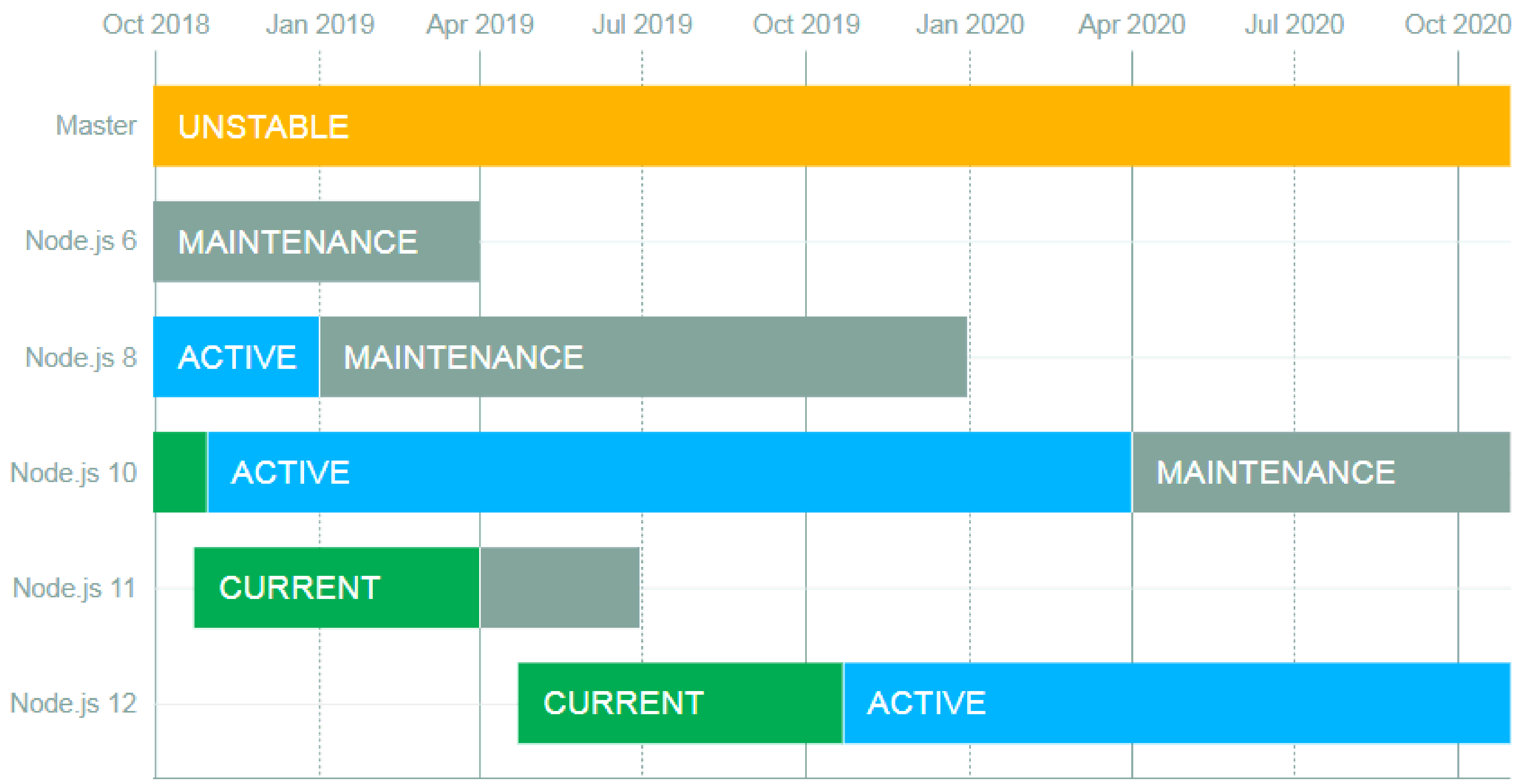

Helped found Node.js Foundation
Contributed ICU support
Started LTS, Intl and Benchmarking WGs
Contributed CiTGM
Contributed support for PPC
Contributed support for AIX
Contributed support for Linux on Z
Made FIPS "real" (Testing/regression coverage
Libuv port for Z//OS
Led work on node-report Contributing to llnode
Helped build coverage.nodejs.org
Co-led N-API Effort

Stable and Predictable Releases

- Bleeding Edge
  - Canary
  - Nightlies
- Current
  - Every 6 months
  - Even releases promoted to LTS
- LTS
  - Every 12 months
  - 30 Months support (18 active, 12 maintenance)

Stable and Predictable Releases - Schedule for 2018

https://github.com/nodejs/Release



N-API

- N-API is a **stable API layer for native modules**, which provides ABI **compatibility guarantees** across different Node.js versions & flavors.
  https://nodejs.org/dist/latest/docs/api/n-api.html
- N-API enables native modules to **just work** across Node.js versions without recompilations!
- A handy-dandy C++ API maintained by the Node.js organization is also available:
  https://github.com/nodejs/node-addon-api

```
#include <node_api.h>

napi_value RunCallback(napi_env env,
                       const napi_callback_info info) {
  napi_status status;
  size_t argc = 1;

  napi_value args[1];
  status = napi_get_cb_info(env, info, &argc, args,
                            nullptr, nullptr);
  napi_value cb = args[0];

  napi_value argv[1];
  status = napi_create_string_utf8(env, "hello world",
                                   NAPI_AUTO_LENGTH, argv);

  napi_value global;
  status = napi_get_global(env, &global);

  napi_value result;
  status = napi_call_function(env, global, cb, 1,
                              argv, &result);
  return nullptr;
}
```

Diagnostic Reports

- Released in **Node.js v11.8.0**
- Usable **via flag only** `--experimental-report`
  - `--diagnostic-report-directory=directory`
  - `--diagnostic-report-filename=filename`
  - `--diagnostic-report-on-fatalerror`
  - `--diagnostic-report-on-signal`
  - `--diagnostic-report-signal=signal`
  - `--diagnostic-report-uncaught-exception`
  - `--diagnostic-report-verbose`
- **JSON** output; see example at https://nodejs.org/docs/latest/api/report.html

```
// automatic trigger
process.report.setDiagnosticReportOptions({
  events: ['exception', 'fatalerror', 'signal'],
  signal: 'SIGUSR2',
  filename: 'myreport.json',
  path: '/home/nodeuser',
  verbose: true
});

// manual trigger
try {
  process.chdir('/non-existent-path');
} catch (err) {
  process.report.triggerReport(err);
}

// custom handling
const report = process.report.getReport(
  new Error('custom error')
);
console.log(report); // JSON string
```

Production Enablement

- First Class Cloud Deployment Options
- Freedom of Platform Choice
- Leverage existing Data assets

- Tools to Accelerate Development/Deployment

First-Class Cloud Support

- **IBM Private Cloud**
  - Enterprise Kubernetes platform
  - Get the speed of public, control of private
  - Fast, flexible, enterprise-grade

- **IBM Public Cloud**
  - Kubernetes (IKS)
  - PaaS Cloud Foundry
  - FaaS Cloud Functions (Apache OpenWhisk)
  - Broad Catalog of Services

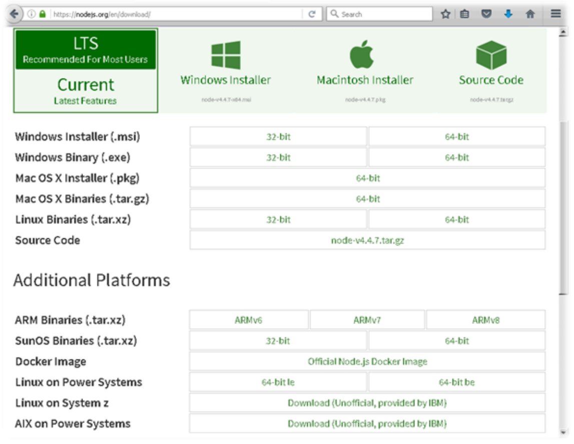
IBM Multicloud Manager

- Enterprise-grade multicloud

  management solution for Kubernetes

Freedom of Platform Choice

- Community Binaries
  - Linux on Z
  - Linux on P
  - AIX



https://nodejs.org/en/download/

LTS
Recommended For Most Users

Current
Latest Features

Windows Installer
node-v4.4.7-x64.msi

Macintosh Installer
node-v4.4.7.pkg

Source Code
node-v4.4.7.tar.gz

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.exe)	32-bit	64-bit
Mac OS X Installer (.pkg)	64-bit	
Mac OS X Binaries (.tar.gz)	64-bit	
Linux Binaries (.tar.xz)	32-bit	64-bit
Source Code	node-v4.4.7.tar.gz	

Additional Platforms

ARM Binaries (.tar.xz)	ARMv6	ARMv7	ARMv8
SunOS Binaries (.tar.xz)	32-bit	64-bit	
Docker Image	Official Node.js Docker Image		
Linux on Power Systems	64-bit le	64-bit be	
Linux on System z	Download (Unofficial, provided by IBM)		
AIX on Power Systems	Download (Unofficial, provided by IBM)		

OpenPOWER LNX ONE AIX IBM

- IBM Binaries
  - IBM i
  - z/OS

for Business

Leveraging Existing Data Assets (IBM i)

- Connecting to Db2 / RPG – packages available on NPM

- For RPG, CL, QSH, Db2, etc, use itoolkit

- Some options for Db2:

1. ibm_db
   - LUW license needed
2. idb-connector
   - Direct Access (traditional)
3. idb-pconnector
   - Direct Access (Promises-based)
4. node-odbc
   - Uses an ODBC driver

Leveraging Existing Data Assets (z/OS)

- 68% of the world’s production workloads and associated data is hosted in z/OS environments

- Enable Collocation with Data hosted on z/OS
  - Up to 2.5x better throughput,
  - 60% faster response time to DB2 on z/OS*

Enhance Node.js ecosystem to access z/OS middleware and assets

CICS, Db2, VSAM, etc.

* Based on performance test on Linux on Z Z

Tools to Accelerate Development/Deployment

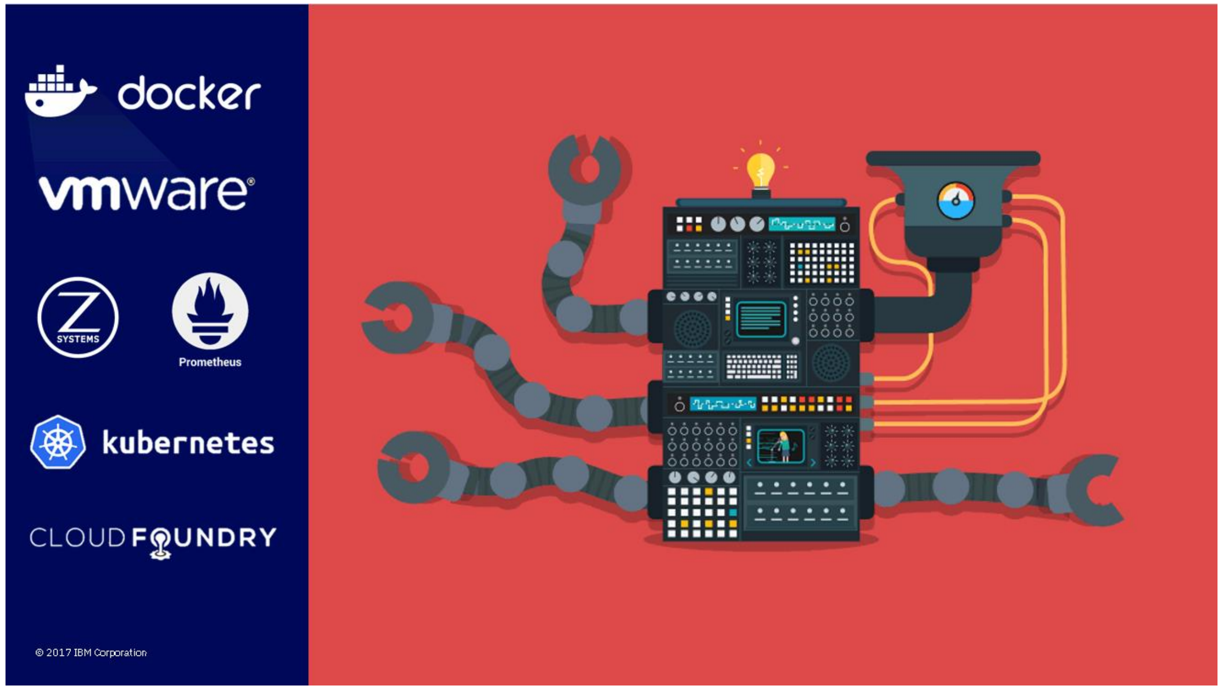
- NodeServer
- IBM Cloud Application Service
- MicroClimate
- AppMetrics
- Loopback
- Documentation/guidance

Create
Deploy
Monitor

NodeServer – open source generators

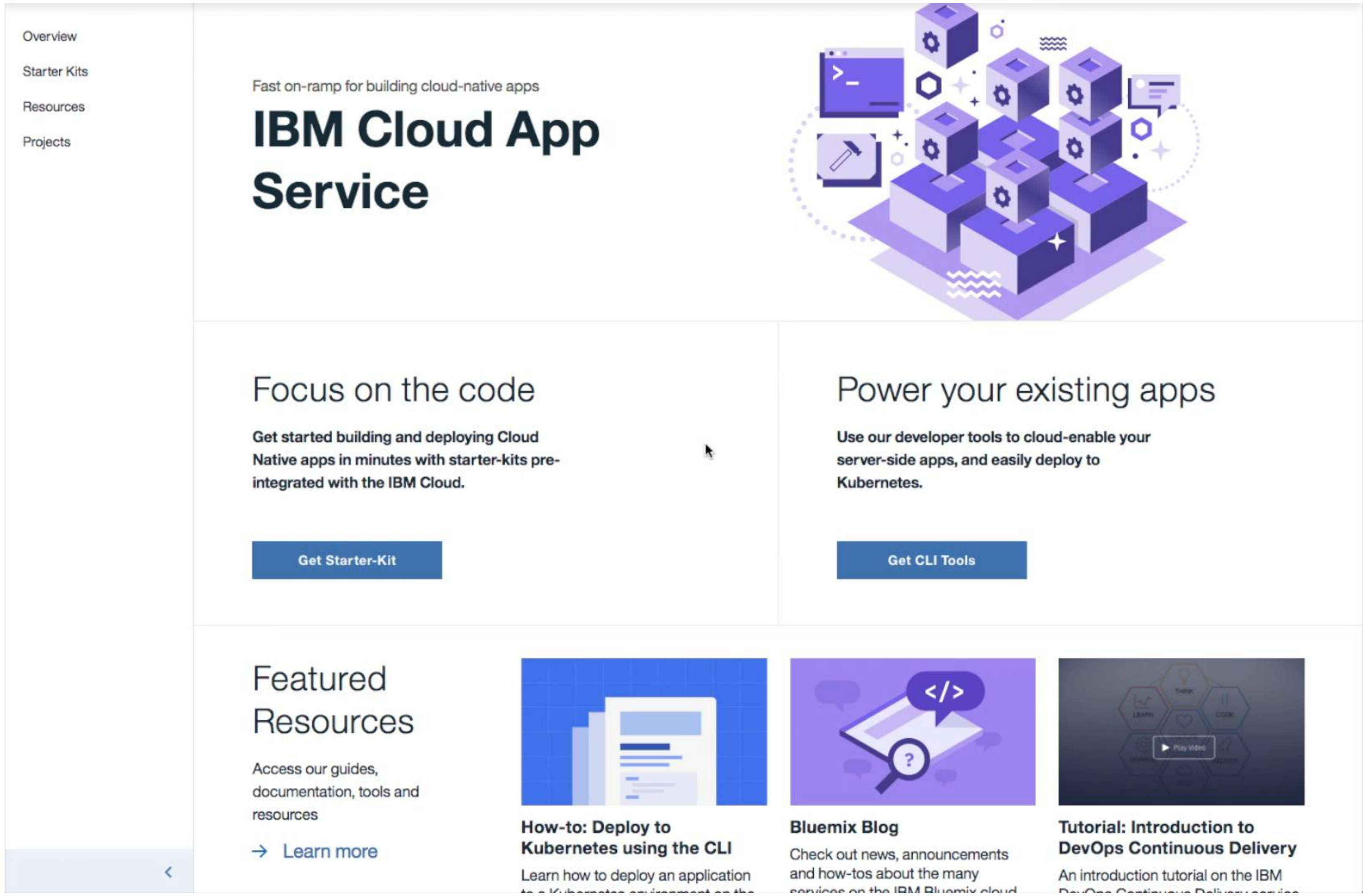
- Create Projects pre-wired for monitoring

- Deploy to
  - Docker
  - Kubernetes
  - Cloud Foundry
  - Dev-ops pipeline



https://www.npmjs.com/package/generator-nodeserver

IBM Cloud Application Service



Fast on-ramp for building cloud-native apps

IBM Cloud App Service

Focus on the code

Get started building and deploying Cloud Native apps in minutes with starter-kits pre-integrated with the IBM Cloud.

Power your existing apps

Use our developer tools to cloud-enable your server-side apps, and easily deploy to Kubernetes.

Featured Resources

Access our guides, documentation, tools and resources

How-to: Deploy to Kubernetes using the CLI

Learn how to deploy an application

Bluemix Blog

Check out news, announcements and how-tos about the many

Tutorial: Introduction to DevOps Continuous Delivery

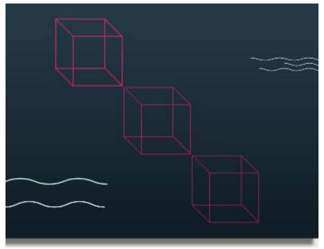
An introduction tutorial on the IBM

© 2019 IBM Corporation

MicroClimate - IBM Developer Experience

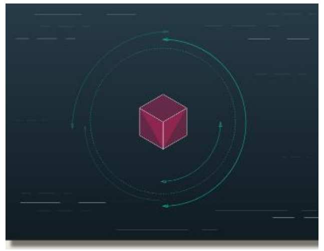
Microclimate is an end to end development platform for the creation of cloud native applications and microservices. You can create, edit, build, test and deploy your applications via Continuous Delivery pipelines then run and manage them with IBM Cloud Private

https://microclimate-dev2ops.github.io/

1


Containerized Development

Start to from scratch using lightweight containers that are easily reproducible to match your production environment locally or on IBM Cloud Private

2


Rapid Iteration

Lightning fast round-tripping through edit, build, and run allows real-time performance insights, regardless of what development phase you're in, with an integrated IDE or use your editor of choice with Language Server Protocols

3

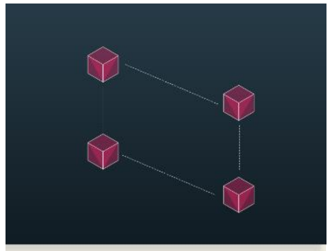
Intelligent Feedback

Best practices and immediate feedback to help improve your application through your IDE

4

Diagnostic Services

Add capability at development time to improve problem determination in production through application metrics.

5


Integrated DevOps Pipeline

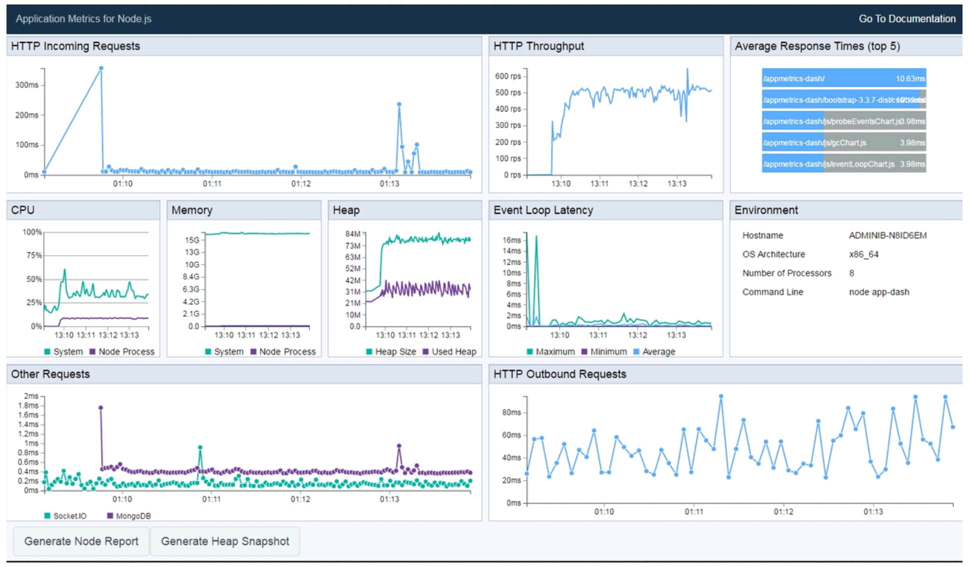
Get into production fast with a preconfigured DevOps pipeline that can be tailored to your needs

AppMetrics - open-source Node.js monitoring

What is it?
An open source module created by IBM for collecting application metrics to diagnose issues while developing your application. Metrics range from HTTP requests, event loop, memory usage, CPU usage, MongoDB connects, and more.

Why use it?
Monitor and diagnose issues while developing your application. App Metrics then connects with IBM Cloud and API Connect for auto-scaling and more detailed availability monitoring

How to get it?
Github at https://github.com/RuntimeTools/appmetrics. Users can view the dashboard by going to /appmetrics-dash or feeding it into their existing dashboard.

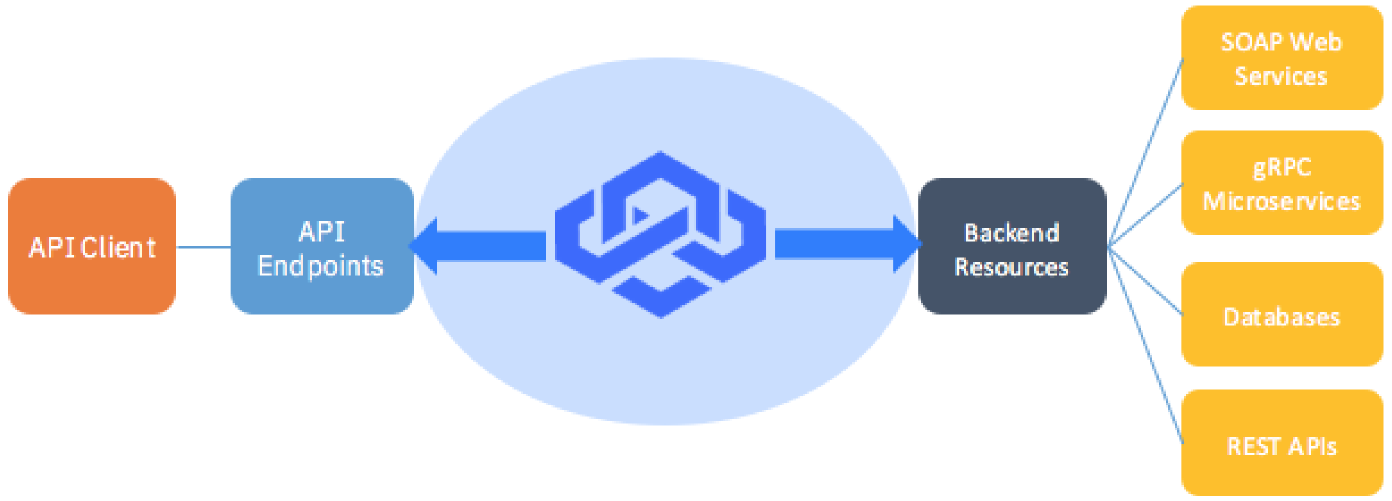


/appmetrics-dash/	10.63ms
/appmetrics-dash/js/probeEventsChart.js	3.98ms
/appmetrics-dash/js/gcChart.js	3.98ms
/appmetrics-dash/js/eventLoopChart.js	3.98ms

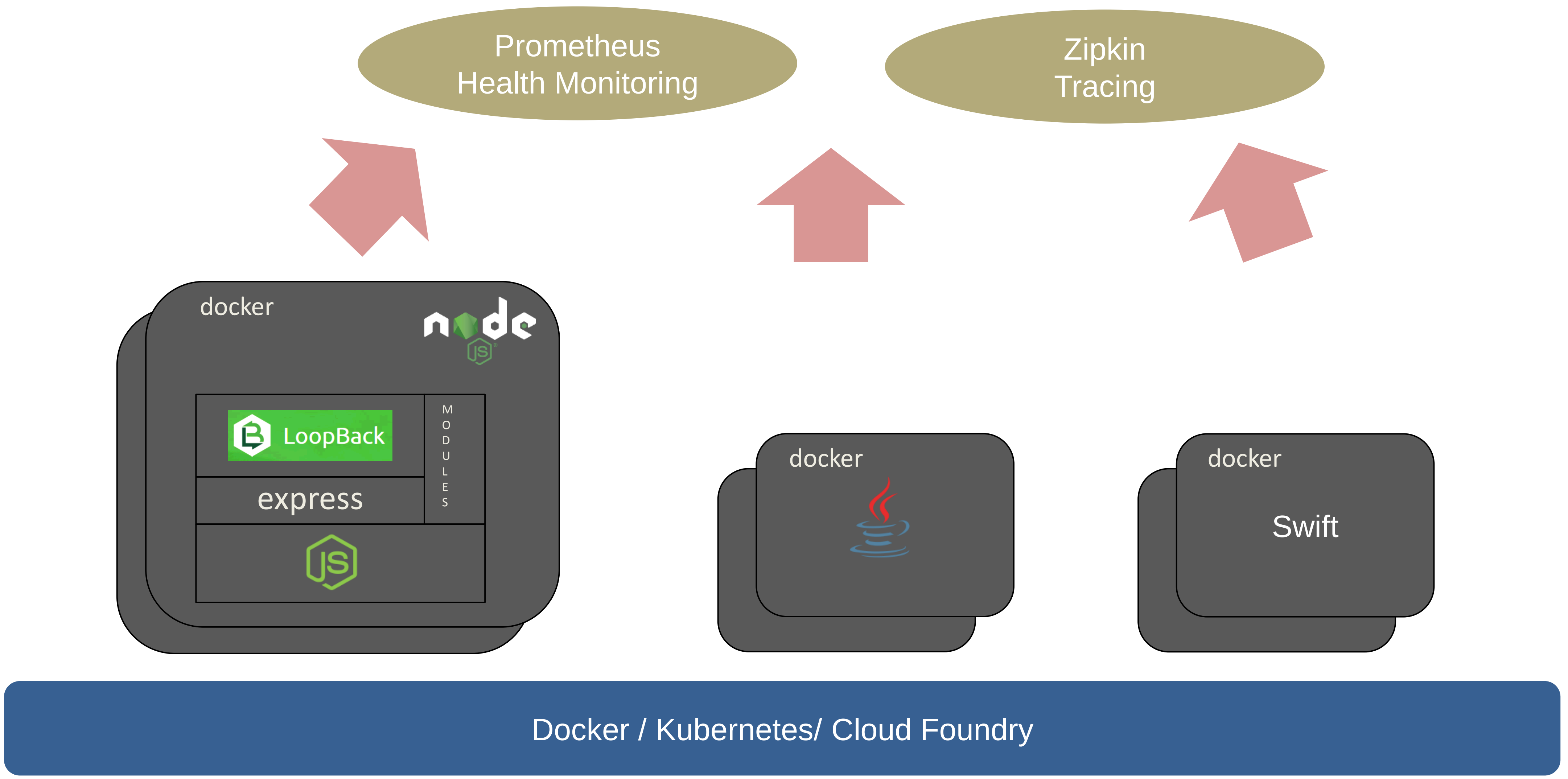
Hostname	ADMINIB-N8ID6EM
OS Architecture	x86_64
Number of Processors	8
Command Line	node app-dash

LoopBack – open-source Node.js framework

- Extends Express to accelerate API creation
- Create APIs quickly as microservices from existing services and databases
- Connects the dots between accepting API requests and interacting with backend
- Built for developers by developers (Reached 10k+ GitHub stars)
- LB3 is for production use. LB4 is under active development
- LB4 brings in support for TypeScript



Tools to Accelerate Development – End Result



Node.js – Package Maintenance

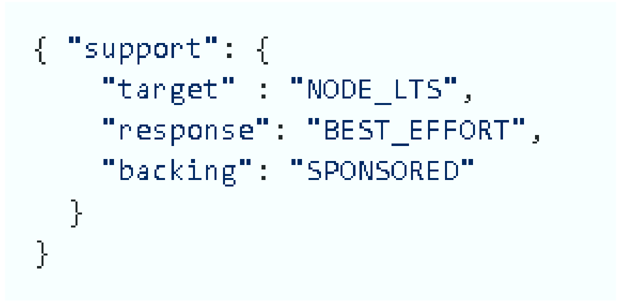
- Call to Action: Accelerating Node.js Growth

  https://medium.com/@nodejs/call-to-action-accelerating-node-js-growth-e4862bee2919

- Challenge
  - Many broadly used packages
    - Small core -> external packages part of core toolbox
    - Hard to keep up with basic maintenance
    - No easy/obvious way for consumers to help

Node.js – Package Maintenance

- Baseline Practice – support level
  - https://github.com/nodejs/package-maintenance/pull/139
    - target
    - response
    - response-paid
    - Backing



```
{ "support": {
    "target" : "NODE_LTS",
    "response": "BEST_EFFORT",
    "backing": "SPONSORED"
  }
}
```

- CI/CD guidelines - https://github.com/nodejs/package-maintenance/pull/146
- Deprecation guidelines - https://github.com/nodejs/package-maintenance/pull/150

Production Support - IBM Support for Runtimes

- Years of experience
- Foundation -Community binaries
- Advanced – Key Modules from the Ecosystem

Q&A

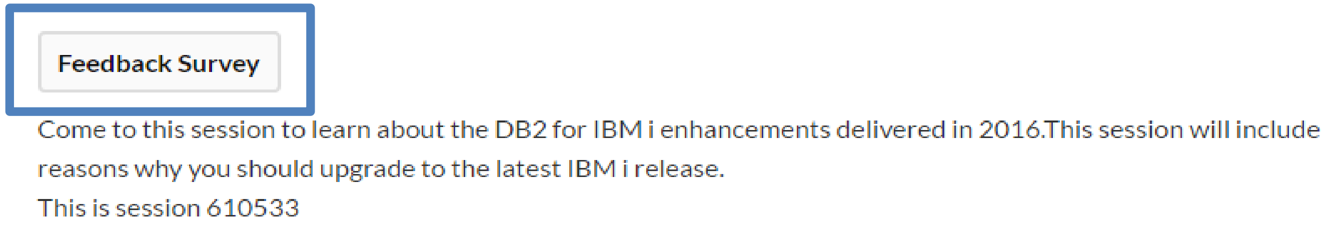
Questions

&

Answers

Don’t Forget Your Session Survey!

Sign in to the Online Session Guide (www.common.org/sessions)
Go to your personal schedule
Click on the session that you attended
Click on the *Feedback Survey* button located above the abstract.



Come to this session to learn about the DB2 for IBM i enhancements delivered in 2016.This session will include reasons why you should upgrade to the latest IBM i release.
This is session 610533

Completing session surveys helps us plan future programming and provides feedback used in speaker awards.
Thank you for your participation.