



## Enterprise/Cloud-ready Node.js

Michael Dawson  
IBM Community Lead for Node.js

Jesse Gorzinski  
jgorzins@us.ibm.com  
Powerpoint Stealer



## Atwood's Law: 2007



“Any application that can be written  
in JavaScript, will eventually be  
written in JavaScript.”

—Jeff Atwood, Cofounder of StackOverflow

## Agenda

---



- Why Node.js ?
- Node.js deep dive (maybe knee-deep)
- Positioning versus Java™
- IBM involvement
- Demo (?)
- A “Happy” Ending



## Why Node.js – What is it?

---



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

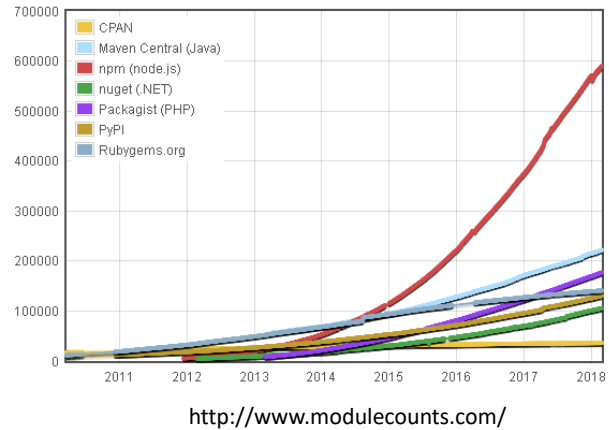


## Why Node.js ? – Ecosystem



- There is a module for that
  - 700K modules
  - #1 on module counts
- #1 on Github (#projects)

### Module Counts



© 2018 IBM Corporation

6

## Why Node.js ? – Ecosystem



- Most used runtime in IBM Cloud



### Infrastructure

#### Containers

Get started by creating a Kubernetes cluster, or manage your Docker images in the registry.



#### Containers in Kubernetes Clusters

Deploy secure, highly available apps in a

IBM

#### Cloud Foundry Apps

Deploy your app without managing underlying infrastructure.



#### SDK for Node.js™

Develop, deploy, and scale server-side

IBM

© 2018 IBM Corporation

7

## Why Node.js ? – Productivity

---



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

## Why Node.js ? – Productivity

---



- **Faster development less code**
- **PayPal** - <https://www.paypal-engineering.com/2013/11/22/node-is-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,"

# Who's Using in Production?



© 2018 IBM Corporation



## Knee-Deep Dive



© 2018 IBM Corporation

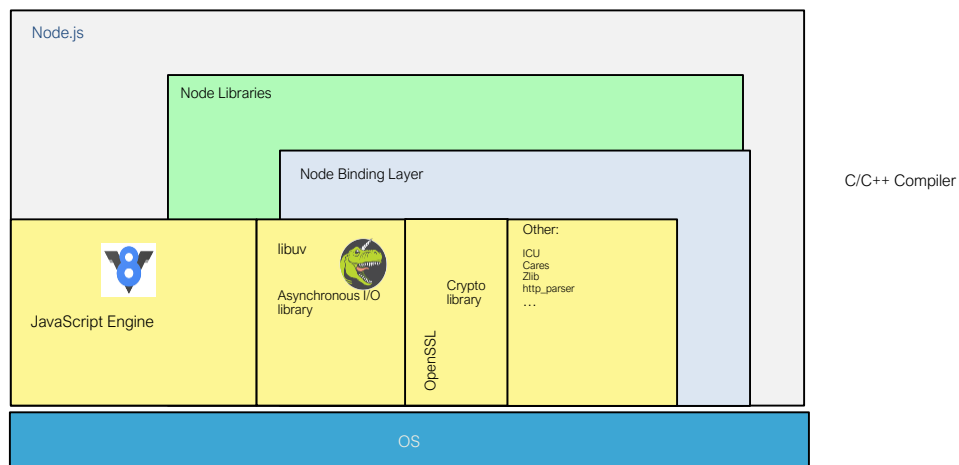
## Node.js – Deep Dive – Key Characteristics



- **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



- Event Based

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

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

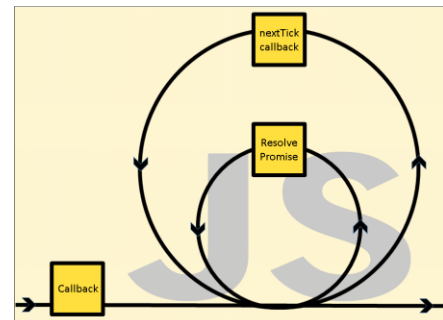
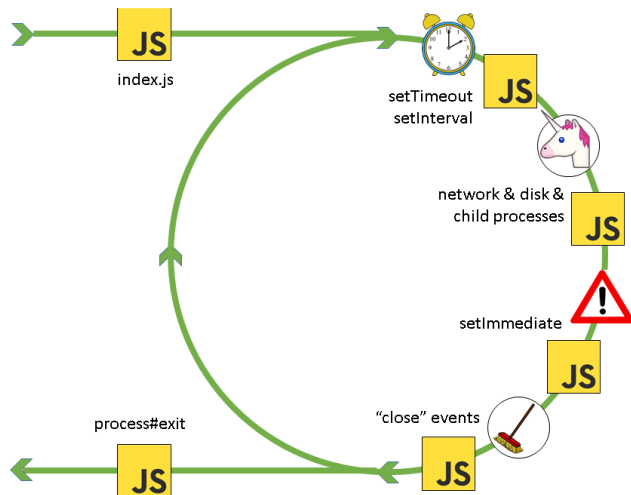
server.on('request', function(request, response) {
    response.writeHead(200, {"Content-Type": "text/plain"});
    response.write("Hello World!\n");
    response.end();
});

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) {});
```

13

© 2018 IBM Corporation

# Node.js – Deep Dive – Event Loop



19

© 2018 IBM Corporation

## 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



```

1. $ mkdir expressjs_app && cd expressjs_app
2. $ npm install express
3. express@4.12.0 node_modules/express
4. |— utils-merge@1.0.0
5. |— methods@1.1.1
6. |— fresh@0.2.4
7. |— merge-descriptors@0.0.2
8. |— cookie-signature@1.0.6
9. |— escape-html@1.0.1
10. |— range-parser@1.0.2
11. |— cookie@0.1.2
12. |— finalhandler@0.3.3
13. |— vary@1.0.0
14. |— content-type@1.0.1
15. |— parseurl@1.3.0
16. |— content-disposition@0.5.0
17. |— serve-static@1.9.1
18. |— path-to-regexp@0.1.3
19. |— depd@1.0.0
20. |— on-finished@2.2.0 (ee-first@1.1.0)
21. |— qs@2.3.3
22. |— debug@2.1.1 (ms@0.6.2)
23. |— proxy-addr@1.0.6 (forwarded@0.1.0, ipaddr.js@0.1.8)
24. |— etag@1.5.1 (crc@3.2.1)

```



## Node.js – Deep Dive – NPM

*holds meta-data about application*



```
$ npm init
```

Creates file `package.json`

```
{
  "name": "expressjs_app",
  "version": "0.0.0",
  "description": "",
  "main": "app.js",
  "dependencies": {
    "express": "^4.12.0"
  },
  "devDependencies": {},
  "author": "Aaron Bartell",
  "license": "ISC"
}
```

Installs these modules when  
`npm install` is run.

[docs.npmjs.com/cli/init](https://docs.npmjs.com/cli/init) - package.json creation  
[docs.npmjs.com/files/package.json](https://docs.npmjs.com/files/package.json) - Docs  
[browsenpm.org/package.json](https://browsenpm.org/package.json) - Easier docs

© 2018 IBM Corporation

## Connecting to Db2 / RPG



- Most important task for developing Node.js applications on the IBM i is connecting to Db2 and/or RPG
- All 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

© 2018 IBM Corporation

## idb-pconnector example



```
const {Connection} = require('idb-pconnector');

async function execExample() {
  try {
    let statement = new Connection().connect().getStatement();

    let result = await statement.exec('SELECT * FROM MYSCHEMA.TABLE');

    console.log(`Select results: \n${JSON.stringify(result)}`);
  } catch(error) {
    console.error(`Error was: \n${error.stack}`);
  }
}

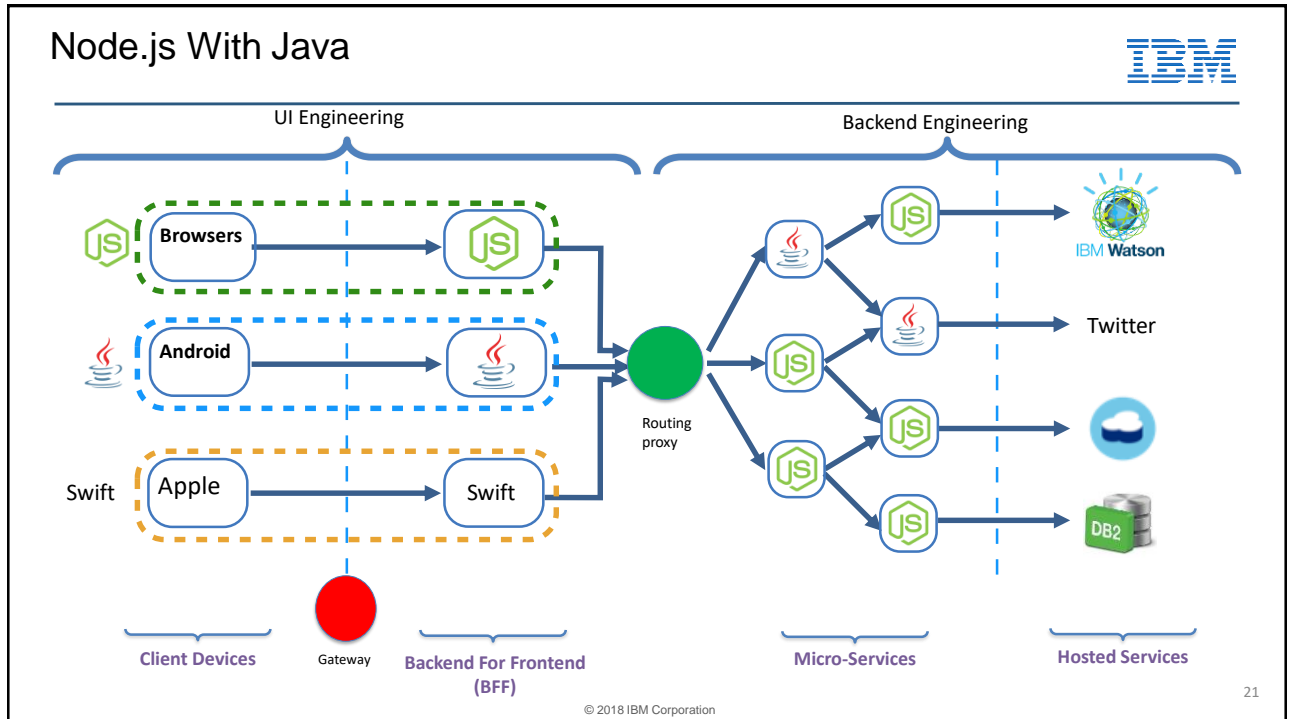
execExample();
```

19



Node.js and Java





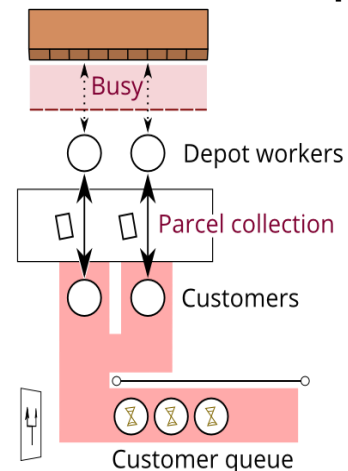
21

## 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

### Parcel collection depot



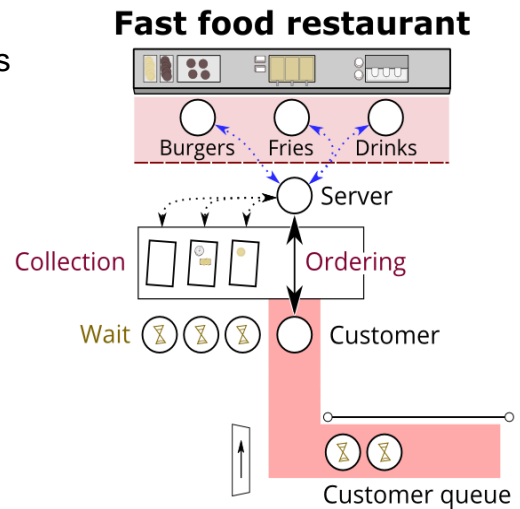
22

© 2018 IBM Corporation

## Node.js versus Java – Scaling with Node.js



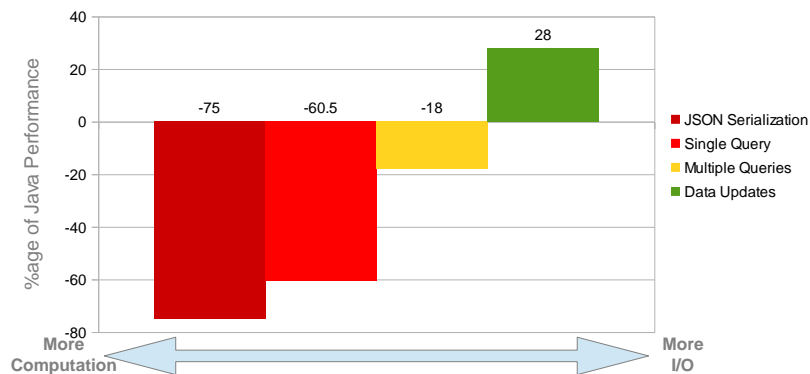
- One thread multiplexes for multiple requests
  - No waiting for a response
  - Handles return from I/O when notified
- Scalability determined by:
  - CPU Usage
  - “Back end” responsiveness
- Concurrency determined by how fast the food server can work



© 2018 IBM Corporation

23

## Node.js With Java– Tradeoffs



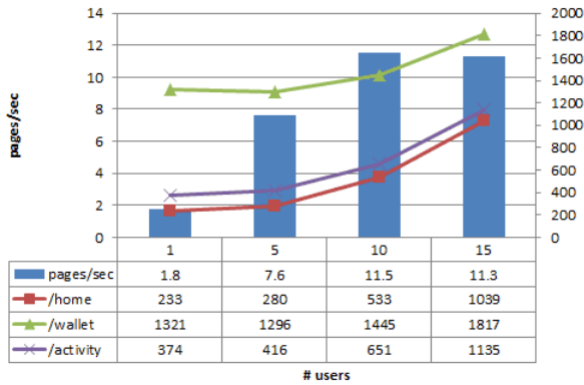
© 2018 IBM Corporation

26

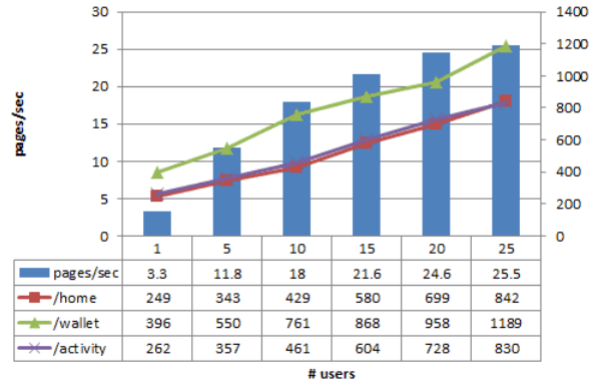
## Why Node.js ? Performance



### Java application



### Node.js application



<https://www.paypal-engineering.com/2013/11/22/node-js-at-paypal/>

© 2018 IBM Corporation

25

## 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%**

© 2018 IBM Corporation

13

## Node.js With Java – Choosing the Right Language



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

© 2018 IBM Corporation

27

## Node.js versus Java – Choosing the Right Language



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

© 2018 IBM Corporation

28

## Node.js With Java– Choosing the Right Language



+

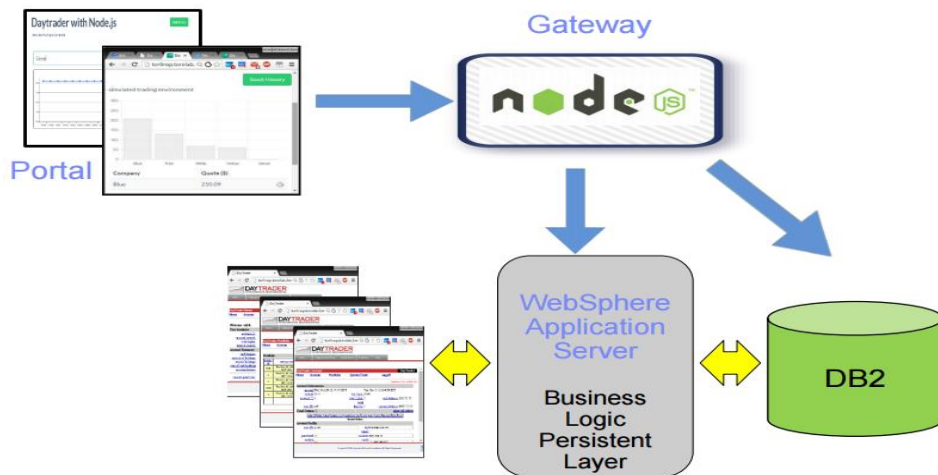


- Highly performant, scalable rich web applications
- Highly performant, reliable transaction processing
- Self-contained micro-service components

© 2018 IBM Corporation

29

## Node.js With Java– Hybrid applications



© 2018 IBM Corporation

30



## IBM involvement



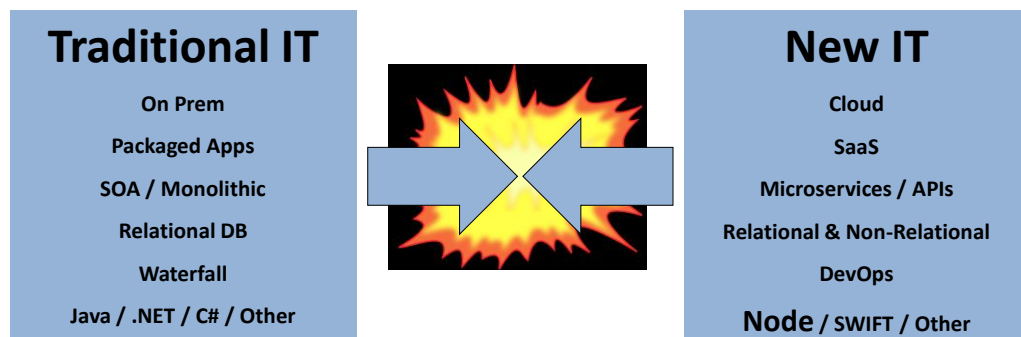
© 2018 IBM Corporation

31

## The Challenge for Every Existing Enterprise:



How to make the old work with the new?



© 2018 IBM Corporation

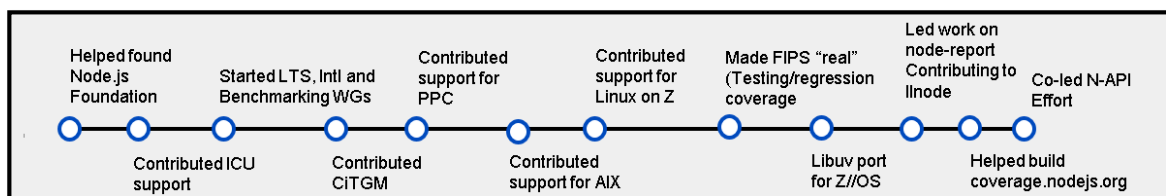


## IBM Node.js Strategy



- Enterprise Ready Runtime
- Production Enablement
- Production Support

## Enterprise Ready Runtime



## N-API

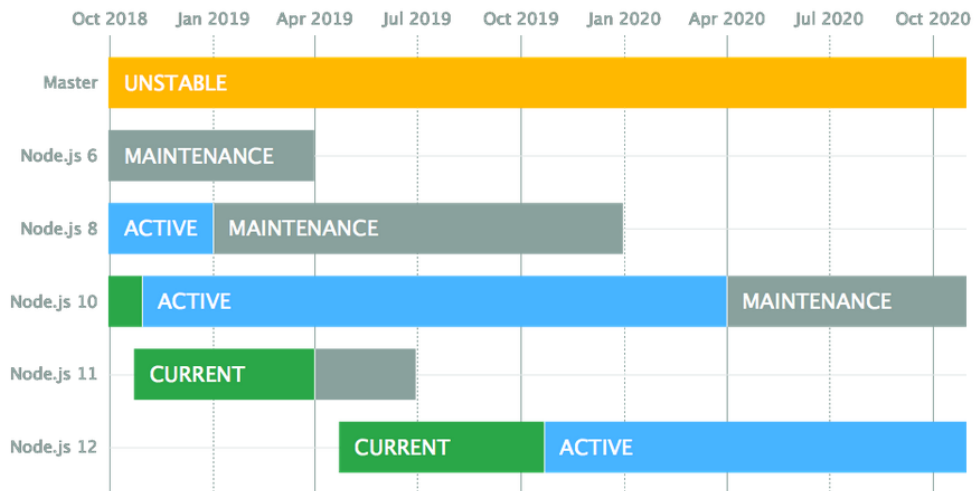


- Before N-API
  - Native modules coded to V8 API's
  - Modules needed to be recompiled for each new version of V8
  - Source changes sometimes also needed
  - Code less portable
  - Required non-trivial currency cost for module owners
- After N-API
  - Native modules coded to N-API
  - Need to be only built once for each platform
  - No code changes or recompiles needed to work with future versions of Node.js

© 2018 IBM Corporation

35

## Stable and Predictable Releases - Schedule for 2018



<https://github.com/nodejs/Release>

© 2018 IBM Corporation

36

# Node.js IBM – Tooling - NodeReport



## NodeReport example - heap out of memory error

### NodeReport content:

- Event summary
- Node.js and OS versions
- JavaScript stack trace
- Native stack trace
- Heap and GC statistics
- Resource usage
- libuv handle summary
- Environment variables
- OS ulimit settings

<https://github.com/nodejs/nodereport>

```

NodeReport
Event: Allocation failed - JavaScript heap out of memory, location: "CALL_AND_RETRY_LAST"
Filename: NodeReport.20160815.125548.97.001.txt
Dump event time: 2016/08/15 12:55:48
Module load time: 2016/08/15 12:46:43

Node.js version: v6.3.0
(v8: 5.0.71.52, libuv: 1.9.1, ssl: 1.2.8, ares: 1.10.1-DEV)
OS version: Linux 3.15.0-33-generic #38-14.04.1-Ubuntu SMP Fri Nov 6 18:17:28 UTC 2015
Machine: 19Sp79815v x86_64
Process ID: 97

----- JavaScript Stack Trace -----
/home/vcap/app/app.js:57:15
Layer.handle [as handle_request] (/home/vcap/app/node_modules/express/lib/router/layer.js:95:5)
next (/home/vcap/app/node_modules/express/lib/router/route.js:111:13)
Route.dispatch (/home/vcap/app/node_modules/express/lib/router/route.js:112:3)
Layer.handle [as handle_request] (/home/vcap/app/node_modules/express/lib/router/layer.js:95:5)
/home/vcap/app/node_modules/express/lib/router/index.js:127:22
....

----- Native Stack Trace -----
0: [po=0x918a8] v8::Utils::ReportApiFailure(char const*, char const*) [node]
1: [po=0x920a2] v8::internal::V8::FatalProcessOutOfMemory(char const*, bool) [node]
2: [po=0x19e88] v8::internal::Factory::NewUninitializedFixedArray(int) [node]
3: [po=0x0b047] [node]
4: [po=0x102f3] v8::internal::Runtime_GrowArrayElements(int, v8::internal::Object**, v8::internal::Isolate*) [
5: [po=0x2751e3706338]

----- Javascript Heap and Garbage Collector -----
Heap space name: new_space
Memory size: 1,048,576 bytes, committed memory: 52,416 bytes
Capacity: 1,051,936 bytes, used: 35,776 bytes, available: 996,160 bytes
Heap space name: old_space
Memory size: 51,298,304 bytes, committed memory: 50,673,800 bytes
Capacity: 50,117,880 bytes, used: 49,317,408 bytes, available: 800,472 bytes
  
```

© 2018 IBM Corporation

37

# 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.



© 2018 IBM Corporation

38

## IBM Node.js Community Leadership



### Participation in Technical Steering Committee



Michael  
Dawson

## IBM Node.js Community Leadership



### 9 Core Collaborators



Michel  
Dawson



Ben  
Noordhuis



Gireesh  
Punathil



Bethany  
Griggs



Yi-Hong  
Wang



Sam  
Roberts



Steven  
Loomis



Richard  
Lau



Ryan  
Graham

## Node.js IBM – V8 Community Involvement



- Deep expertise at V8
- Developed ports to IBM Platforms
- Contribution back to official V8 repositories:
  - **PPC:** V8 4.3 and later have full functional PPC implementation
  - **s390:** V8 5.1 and later have full functional implementation
  - ~10-15 commits per week to V8 to maintain PPC/zlinux port
- Internal port for z/OS and IBM i

© 2018 IBM Corporation

41

## Freedom of Platform Choice



### • Community Binaries

- Linux on Z
- Linux on P
- AIX



LTS		Windows Installer	Mac/Unix Installer	Source Code
Current	Linux Binaries			
Windows Installer (Linux)	22 Jan	14 Jan		
Windows Binaries (Linux)	22 Jan	14 Jan		
Mac OS X Binaries (Linux)	14 Jan	14 Jan		
Linux Binaries (Linux)	14 Jan	14 Jan		
Source Code	14 Jan	14 Jan		

Additional Platforms		22 Jan	14 Jan	14 Jan
AIX Binaries (Linux)				
S/390 Binaries (Linux)				
Doc/Kit Image				
Linux on Power Systems				
Linux on System z				
AIX on Power Systems				

### • IBM Binaries

- IBM i
- z/OS



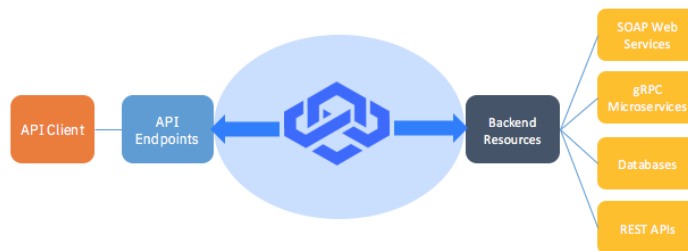
© 2018 IBM Corporation

42

## 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)



© 2018 IBM Corporation

43

## Production Support - IBM Support for Runtimes



- Years of experience
- Foundation -Community binaries
- Advanced – Key Modules from the Ecosystem (Express.js & Loopback)

<https://www.ibm.com/uk-en/marketplace/support-for-runtimes/faq>

© 2018 IBM Corporation

44

## IBM TSS Support for IBM i

---



- Git
  - Jenkins
  - rsync
  - Node.js
  - Apache Tomcat
  - WordPress
  - Python
- 
- For more resources, see my blog post:  
<http://ibmsystemsmag.com/blogs/open-your-i/december-2018/a-game-changer-for-open-source-support/>

© 2018 IBM Corporation

## Node.js foundation

---

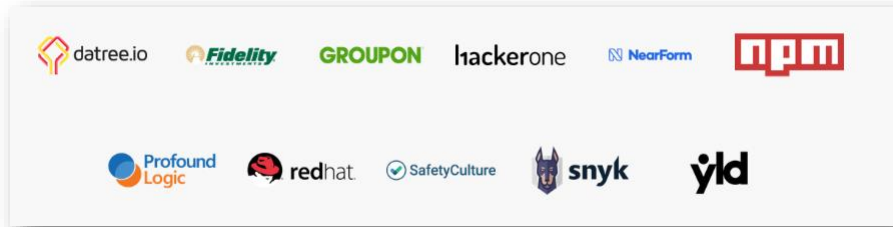


- <https://foundation.nodejs.org/>
- The Node.js Foundation's mission is to enable widespread adoption and help accelerate development of Node.js and other related modules through an open governance model that encourages participation, technical contribution, and a framework for long term stewardship by an ecosystem invested in Node.js' success.

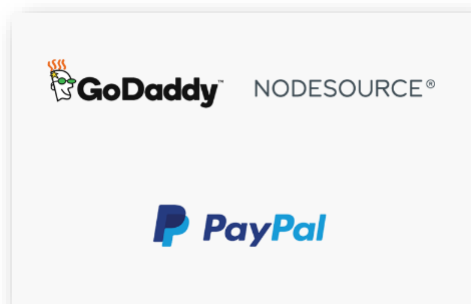
© 2018 IBM Corporation

46

## Node.js foundation – Silver Members



## Node.js foundation – Gold Members





## Node.js foundation – Platinum Members



© 2018 IBM Corporation

49

## Debugging Node.js in a browser



The screenshot shows the Chrome DevTools interface with the Sources panel open to a file named `hi.js`. The code is as follows:

```

1 (function (exports, require, module, __filename, __dirname)
2 http = require('http');
3 http.createServer(function (req, res) {
4   res.writeHead(200, {'Content-Type': 'text/plain'});
5   res.end('Hello World\n');
6 }).listen(8080);
7 console.log('Server running on port 8080.');
```

The right-hand pane shows the Call Stack with the following entries:

- (anonymous function) hi.js:2
- Module\_compile module.js:570
- Module\_extension module.js:579
- s.js

More debugging options: [bit.ly/rs-debug-nodejs](https://bit.ly/rs-debug-nodejs)

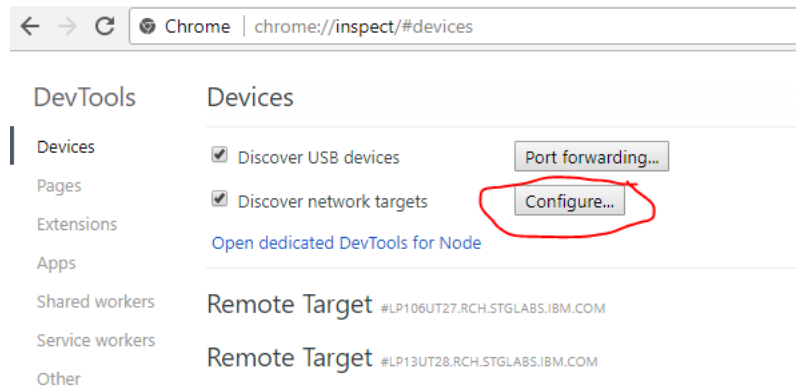
© 2018 IBM Corporation

## Debugging Node.js in a browser



Visit `chrome://inspect` in chrome

Configure your hostname and port as a "network target" (port 9229 is default port)



© 2018 IBM Corporation

## Debugging Node.js in a browser



- `$ node --inspect=0.0.0.0 hi.js`
- Debugger listening on port 9229.

Start node with `--inspect`

NOTE: IP Address '0.0.0.0' is important! Port will default to 9229 if not specified

© 2018 IBM Corporation

## Debugging Node.js in a browser



VOILA!! You will now see the remote target and can launch debug!

chrome://inspect/#devices

DevTools

- Devices
- Pages
- Extensions
- Apps
- Shared workers
- Service workers
- Other

Devices

- Discover USB devices Port forwarding...
- Discover network targets Configure...

[Open dedicated DevTools for Node](#)

Remote Target #LP106UT27.RCH.STGLABS.IBM.COM

Remote Target #LP13UT28.RCH.STGLABS.IBM.COM

**Target**

- app.js file:///home/AMUSSE/hello\_world/app.js inspect

© 2018 IBM Corporation

chrome-devtools://devtools/bundled/inspector.html?v8only=true&remoteBase=https://chrome-devtools-frontend.appspot.com/serve\_file/@2e6edcfee630baa3775f37cb11796b1603a64360/...

Console Sources Memory Profiler

Network Filesystem

async\_hooks.js app.js x \_http\_outgoing.js

```

1 (function (exports, require, module, __filename, __dirname) { var express = require('express');
2 var app = express();
3 app.get('/', function (req, res) { req = IncomingMessage { readableState: ReadableState, readable: true
4   res.send('Hello World!!');
5 });
6 app.listen(1984, function () {
7   console.log('Example app listening on port 1984!');
8 });
9 });

```

Paused on breakpoint

Watch

Call Stack

- (anonymous) app.js:4
- handle layer.js:95
- next route.js:137
- dispatch route.js:112
- handle layer.js:95
- (anonymous) index.js:281
- process\_params index.js:335
- next index.js:275
- expressInit init.js:40
- handle layer.js:95
- trim\_prefix index.js:317

Line 4, Column 7

Console

Nodejs Main Co... | Filter | Default levels

Example app listening on port 1984! app.js:7

© 2018 IBM Corporation

## Debugging Node.js in VSCode



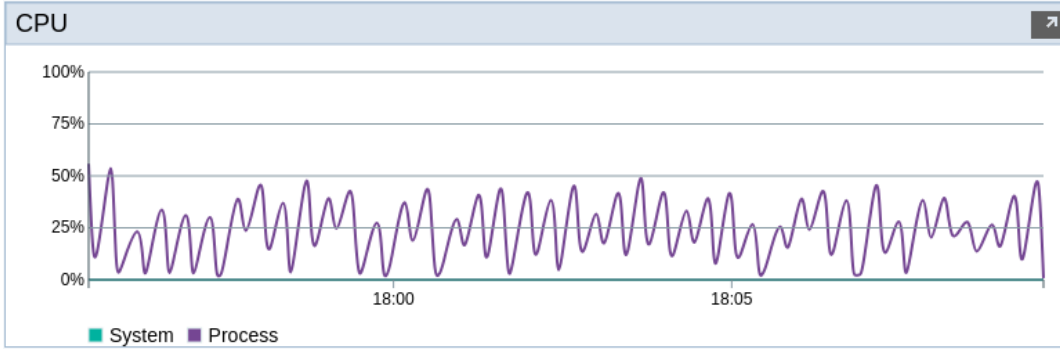
```
file.js - node-inspector
launch.json  file.js
1  function hello(name){
2  var output = `Hello ${name}`;
3  console.log(output);
4  return true;
5  }
6
7  hello('Aaron Bartell');
```

Ln 2, Col 1 Spaces: 2 UTF-8 LF JavaScript ESLint

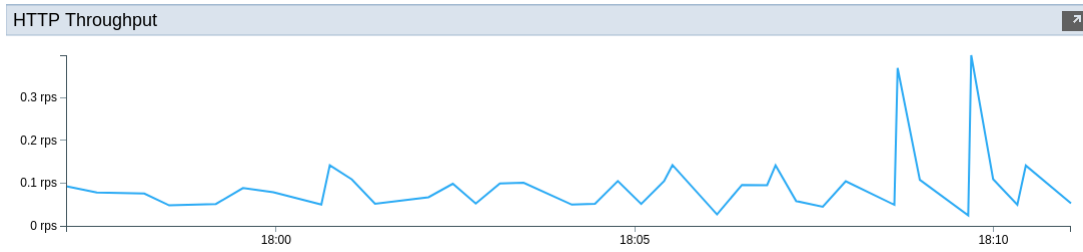


# Demo!

# AppMetrics



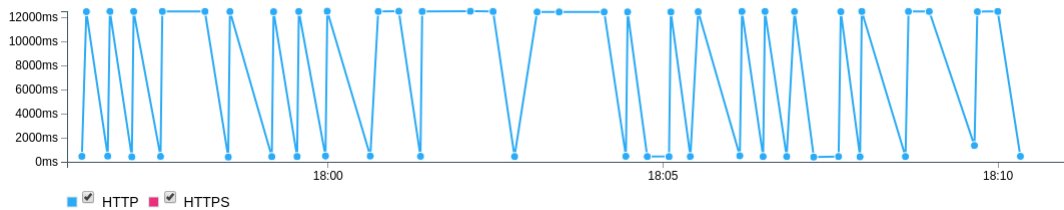
# AppMetrics



# AppMetrics



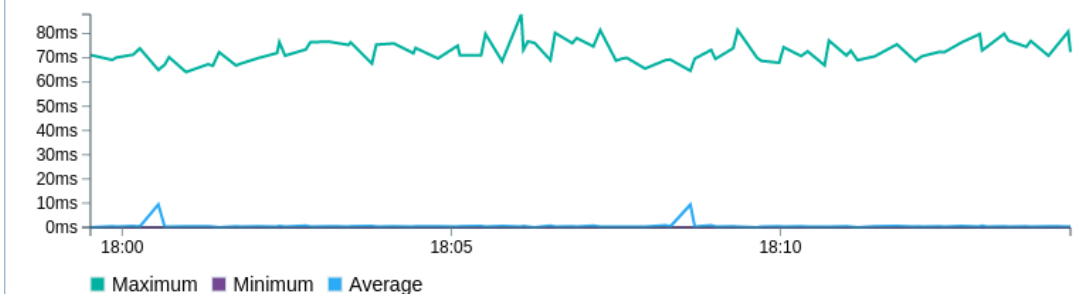
### HTTP Incoming Requests



# AppMetrics



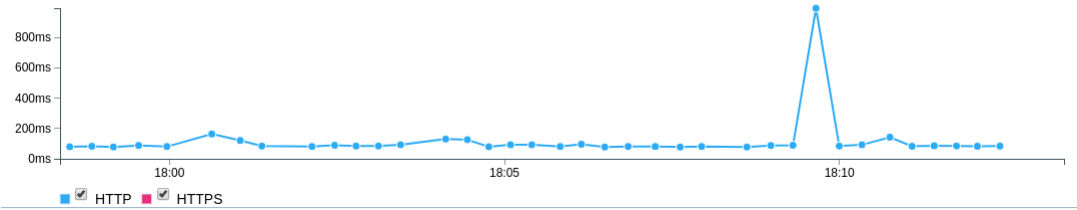
### Loop Times



# AppMetrics



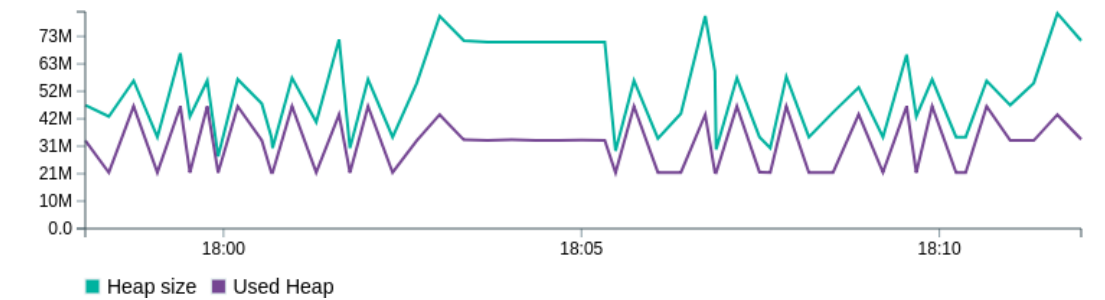
### HTTP Outbound Requests



# AppMetrics



### Heap



## AppMetrics



Flame Graph



Function Details

Self+Children 89.8% (Self: 89.8%)

Call Stack

1. generatePrimes (/OpenSys/home/amusse/repos/express\_books/routes/primes.js:7)
2. router.get (/OpenSys/home/amusse/repos/express\_books/routes/primes.js:31)
3. handle (express/lib/router/layer.js:86)
4. next (express/lib/router/route.js:114)
5. dispatch (express/lib/router/route.js:98)
6. handle (express/lib/router/layer.js:86)
7. <anonymous function> (express/lib/router/index.js:275)
8. process\_params (express/lib/router/index.js:327)
9. next (express/lib/router/index.js:176)
10. handle (express/lib/router/index.js:136)
11. router (express/lib/router/index.js:46)
12. handle (express/lib/router/layer.js:86)
13. trim\_prefix (express/lib/router/index.js:288)
14. <anonymous function> (express/lib/router/index.js:275)
15. process\_params (express/lib/router/index.js:327)
16. next (express/lib/router/index.js:176)
17. <anonymous function> (express/lib/router/index.js:629)
18. next (express/lib/router/index.js:176)
19. handle (express/lib/router/index.js:136)
20. router (express/lib/router/index.js:46)
21. handle (express/lib/router/layer.js:86)
22. trim\_prefix (express/lib/router/index.js:288)
23. <anonymous function> (express/lib/router/index.js:275)
24. process\_params (express/lib/router/index.js:327)
25. next (express/lib/router/index.js:176)
26. <anonymous function> (/OpenSys/home/amusse/repos/express\_books/app.js:77)
27. handle (express/lib/router/layer.js:86)
28. trim\_prefix (express/lib/router/index.js:288)
29. <anonymous function> (express/lib/router/index.js:275)
30. process\_params (express/lib/router/index.js:327)
31. next (express/lib/router/index.js:176)
32. <anonymous function> (connect-flash/lib/Flash.js:18)
33. handle (express/lib/router/layer.js:86)
34. trim\_prefix (express/lib/router/index.js:288)
35. <anonymous function> (express/lib/router/index.js:275)
36. process\_params (express/lib/router/index.js:327)
37. next (express/lib/router/index.js:176)
38. strategy.parse (passport/lib/middleware/authenticate.js:337)
39. SessionStrategy.authenticate (passport/lib/strategies/session.js:44)
40. attempt (passport/lib/middleware/authenticate.js:177)
41. authenticate (passport/lib/middleware/authenticate.js:94)
42. handle (express/lib/router/layer.js:86)

© 2018 IBM Corporation

63

## AppMetrics



Function Details

Self+Children 89.8% (Self: 89.8%)

Call Stack

1. generatePrimes (/OpenSys/home/amusse/repos/express\_books/routes/primes.js:7)
2. router.get (/OpenSys/home/amusse/repos/express\_books/routes/primes.js:31)
3. handle (express/lib/router/layer.js:86)
4. next (express/lib/router/route.js:114)
5. dispatch (express/lib/router/route.js:98)
6. handle (express/lib/router/layer.js:86)
7. <anonymous function> (express/lib/router/index.js:275)
8. process\_params (express/lib/router/index.js:327)
9. next (express/lib/router/index.js:176)
10. handle (express/lib/router/index.js:136)
11. router (express/lib/router/index.js:46)
12. handle (express/lib/router/layer.js:86)
13. trim\_prefix (express/lib/router/index.js:288)
14. <anonymous function> (express/lib/router/index.js:275)
15. process\_params (express/lib/router/index.js:327)
16. next (express/lib/router/index.js:176)
17. <anonymous function> (express/lib/router/index.js:629)
18. next (express/lib/router/index.js:176)
19. handle (express/lib/router/index.js:136)
20. router (express/lib/router/index.js:46)
21. handle (express/lib/router/layer.js:86)
22. trim\_prefix (express/lib/router/index.js:288)
23. <anonymous function> (express/lib/router/index.js:275)
24. process\_params (express/lib/router/index.js:327)
25. next (express/lib/router/index.js:176)

64





---

# A “Happy” Ending



---

# A “Hapi” Ending



## Walmart creates a framework!



- Express.js appeared in 2009.
- Walmart saw Express.js insufficient for very large projects, but saw the huge potential in Node.js.
- Willing to invest millions of dollars in a new framework.

- <https://garage.socialisten.at/2016/12/enterprise-level-backend-framework-from-walmart>



© 2018 IBM Corporation

68

## Black Friday 2013



- Full deployment for all mobile shopping!!
- The hardware?
  - 10 CPU cores
  - 28 GB memory



© 2018 IBM Corporation

69

## Q&A



# The END!

© 2018 IBM Corporation

70

## Special notices



This document was developed for IBM offerings in the United States as of the date of publication. IBM may not make these offerings available in other countries, and the information is subject to change without notice. Consult your local IBM business contact for information on the IBM offerings available in your area.

Information in this document concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquiries, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

The information contained in this document has not been submitted to any formal IBM test and is provided "AS IS" with no warranties or guarantees either expressed or implied.

All examples cited or described in this document are presented as illustrations of the manner in which some IBM products can be used and the results that may be achieved.

Actual environmental costs and performance characteristics will vary depending on individual client configurations and conditions.

IBM Global Financing offerings are provided through IBM Credit Corporation in the United States and other IBM subsidiaries and divisions worldwide to qualified commercial and government clients. Rates are based on a client's credit rating, financing terms, offering type, equipment type and options, and may vary by country. Other restrictions may apply. Rates and offerings are subject to change, extension or withdrawal without notice.

IBM is not responsible for printing errors in this document that result in pricing or information inaccuracies.

All prices shown are IBM's United States suggested list prices and are subject to change without notice; reseller prices may vary.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally-available systems. Some measurements quoted in this document may have been estimated through extrapolation. Users of this document should verify the applicable data for their specific environment.

Revised September 26, 2006

© 2018 IBM Corporation

71

## Special notices (cont.)



IBM, the IBM logo, [www.ibm.com](http://www.ibm.com), AIX, AIX (logo), AIX 5L, AIX 6 (logo), AS/400, BladeCenter, Blue Gene, ClusterProven, Db2, ESCON, i5/OS, i5/OS (logo), IBM Business Partner (logo), IntelliStation, LoadLeveler, Lotus, Lotus Notes, Notes, Operating System/400, OS/400, PartnerLink, PartnerWorld, PowerPC, pSeries, Rational, RISC System/6000, RS/6000, THINK, Tivoli, Tivoli (logo), Tivoli Management Environment, WebSphere, xSeries, z/OS, zSeries, Active Memory, Balanced Warehouse, CacheFlow, Cool Blue, IBM Systems Director VMControl, pureScale, TurboCore, ChipSopper, Cloudscape, Db2 Universal Database, DS4000, DS6000, DS8000, EnergyScale, Enterprise Workload Manager, General Parallel File System, GFS, HACMP, HACMP/6000, HASM, IBM Systems Director Active Energy Manager, iSeries, Micro-Partitioning, POWER, PowerExecutive, PowerVM, PowerVM (logo), PowerHA, Power Architecture, Power Everywhere, Power Family, POWER Hypervisor, Power Systems, Power Systems (logo), Power Systems Software, Power Systems Software (logo), POWER2, POWER3, POWER4, POWER4+, POWER5, POWER5+, POWER6, POWER6+, POWER7, System i, System p, System p5, System Storage, System z, TME 10, Workload Partitions Manager and X-Architecture are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries.

A full list of U.S. trademarks owned by IBM may be found at: <http://www.ibm.com/legal/copytrade.shtml>.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

AltiVec is a trademark of Freescale Semiconductor, Inc.

AMD Opteron is a trademark of Advanced Micro Devices, Inc.

InfiniBand, InfiniBand Trade Association and the InfiniBand design marks are trademarks and/or service marks of the InfiniBand Trade Association.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Java, JavaScript and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries or both.

Microsoft, Windows and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries or both.

NetBench is a registered trademark of Ziff Davis Media in the United States, other countries or both.

SPECint, SPECintp, SPECjbb, SPECweb, SPECjAppServer, SPEC OMP, SPECviewperf, SPECcapc, SPECchp, SPECjvm, SPECmail, SPECimap and SPECcfs are trademarks of the Standard Performance Evaluation Corp (SPEC).

The Power Architecture and Power.org wordmarks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

TPC-C and TPC-H are trademarks of the Transaction Performance Processing Council (TPPC).

UNIX is a registered trademark of The Open Group in the United States, other countries or both.

\*Node.js is an official trademark of Joyent. IBM SDK for Node.js is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

\*TWITTER, TWEET, RETWEET and the Twitter logo are trademarks of Twitter, Inc. or its affiliates.\*

Revised December 2, 2010

Other company, product and service names may be trademarks or service marks of others.

© 2018 IBM Corporation