Node.js Development

"Server Side JavaScript Environment"

Agenda

- Introduction to Node.js
- History of Node.js
- Introduction to io.js
- What is Node.js Foundation?
- V8 JavaScript Engine
- Why Server-side JavaScript?
- Node.js Architecture

Agenda (Contd.)

- JavaScript Event Loop
- Node.js vs. others Server Side Frameworks
- Node.js Application Area
- Who Use Node.js
- Advantages of Node.js
- Limitations of Node.js
- Node.js Application Deployment Server

Introduction to Node.js

Introduction to Node.js

- Server side JavaScript environment for developing web app like as ASP.NET, JSP, Php etc.
- Open-source and cross-platform framework.
- Based on Google's V8 JavaScript Engine.
- Used to build fast & scalable network applications as well as data-intensive real-time web applications.

History of Node.js

History of Node.js

- Developed by Ryan Dahl at Joyent, Inc. in May 27, 2009.
- First, released in 2009 supporting only Linux.
- In 2011, windows version was released.
- All versions of Node.js are starting from 0.1.0 releases to 0.1.x, 0.2.x, …..0.10.x,0.11.x and 0.12.x.
- In Sep 14, 2015, Node.js 4.0 was announced, a combined release of Node.js(0.12.x) & io.js(3.x) into single code base.

Introduction to io.js

Introduction to io.js

- In December 2014, Fedor Indutny started io.js, a fork of Node.js because of internal conflict over Joyent's governance.
- Created to accelerate the development and predicted releases of code under an "open governance model".
- All versions of io.js are starting from 1.0 releases to 1.x, 2.x and 3.x.

What is Node.js foundation?

What is Node.js foundation?

- It is an independent foundation to take care of development and releases of Node.js.
- It has developers from IBM, Microsoft, PayPal, Joyent, Fidelity, SAP and other companies.
- In Sep 14, 2015, it announced the combined release of Node.js(0.12.x) and io.js(3.x) into a single code base known as Node.js version 4.0.

V8 JavaScript Engine

V8 JavaScript Engine

- An open source JavaScript engine developed by Google in 2008 to be used in Chrome browser.
- Written in C++ language and implements ES5 and ES6.
- Uses just-in-time compilation (JIT) to execute JavaScript.
- Compiles JavaScript to native machine code (IA-32, x86-64, ARM, or MIPS ISAs) before execution.

V8 JavaScript Engine (Contd.)

- It can run standalone or can be embedded with any C++ application.
- Now, used by many open source projects like Node.js and MongoDB to execute JavaScript on server side.

Why Server-side JavaScript?

Why Server-side JavaScript?

- Unified language for both front-end and back-end .
- Increase programmer productivity.
- Code reusability.
- Exchange of data using JSON.
- JavaScript with V8 engine performs faster than Php, Ruby, Python, JSP and ASP.NET.

JavaScript for Desktop Apps

JavaScript for Desktop Apps

- JS is used for Web & Desktop Apps development
- **Electron** is a Framework for Building cross platform desktop apps with JS, HTML, and CSS
- A combination of Chromium and Node.js
- Compatible with **Mac**, **Windows**, and **Linux**
- An open source project maintained by **GitHub**

Desktop Apps built on Electron

Desktop Apps built on Electron

Atom	Slack	Visual Studio Code	Ionic Creator	Ionic Lab	Nuclide	Moeditor
MdNote	WordPress.com	Wagon	PhoneGap	Cocos Creator	Hive	Socialcast
Yeoman	Postman	Rocket.Chat	Caprine	Whatsie	Nylas N1	GitBook

DotNetTricks

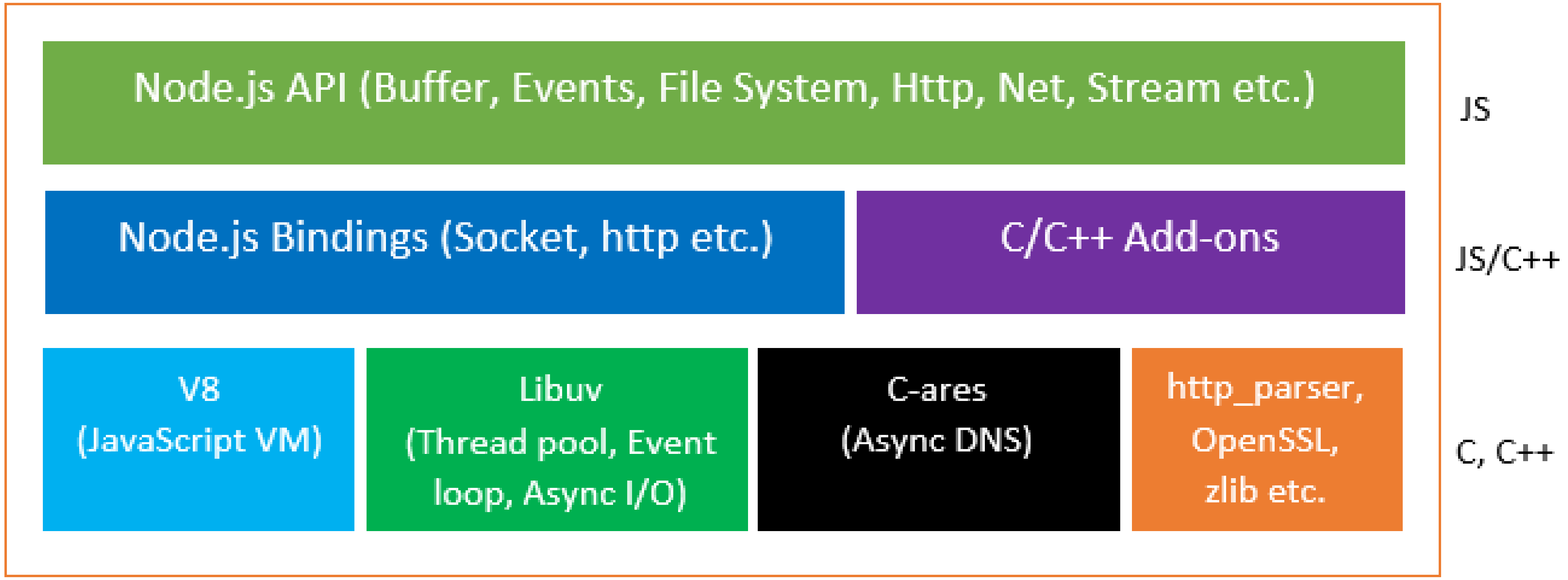
node JS

Node.js Architecture

Node.js Architecture

- Node.js has mainly two types of components – core components and node.js API (modules).
- The core components are written in C and C++, and node.js API are written in JavaScript

Node.js Architecture (Contd.)


Node.js API (Buffer, Events, File System, Http, Net, Stream etc.)
JS
Node.js Bindings (Socket, http etc.)
C/C++ Add-ons
JS/C++
V8 (JavaScript VM)
Libuv (Thread pool, Event loop, Async I/O)
C-ares (Async DNS)
http_parser, OpenSSL, zlib etc.
C, C++

Node.js 4.2.4 Architecture

Node.js Architecture (Contd.)

- **Node.js API –** These are written in JavaScript and directly exposed to outer world to interact with Node.js internal components.
- **Node.js Binding** – These are Core API, which bind the JavaScript with C / C++ libraries.
- **C/C++ Add-ons** – You can also develop your Node.js Add-ons using C/C++ to work with Node.js.

Node.js Architecture (Contd.)

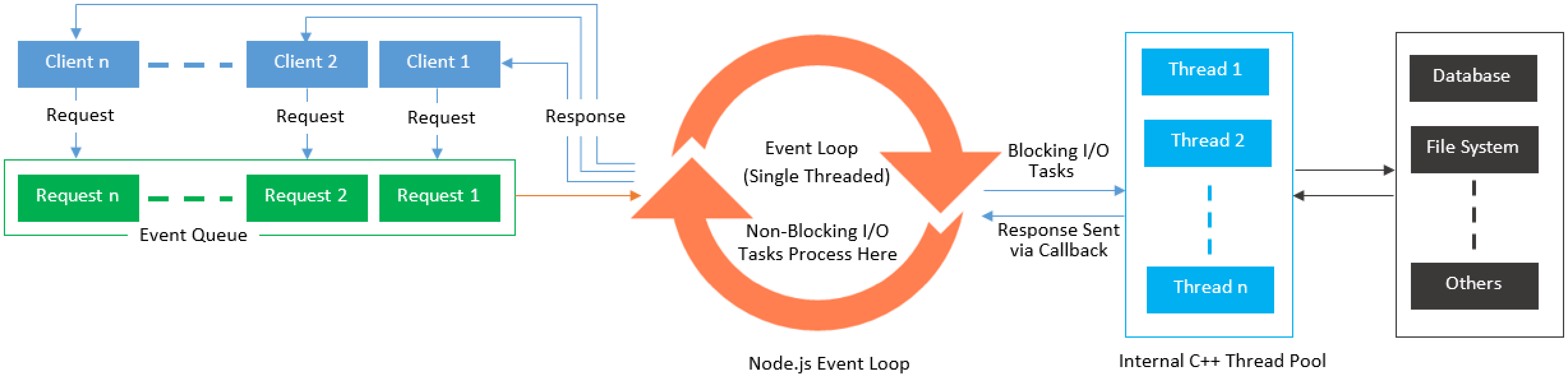
- **V8 –** JavaScript VM which compile the JavaScript code into native machine code instead of interpretation. It is the fastest JIT (Just-In-Time) compiler for JavaScript.
- **Libuv –** A multi-platform support C++ library which is responsible for handling thread pool, event loop and async I/O operations in Node.js.
- **C-ares –** C library for handling async DNS request, name resolves and multiple DNS queries in parallel.

Node.js Architecture (Contd.)

- **http_parser –** C library for parsing HTTP request and response.
- **OpenSSL –** C library for the implementation of Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols. It also provides all the necessary cryptography methods like hash, cipher etc.
- **Zlib –** C library for data compression and decompression.

Event Loop

Event Loop



Client n
Client 2
Client 1
Request
Request
Request
Response
Request n
Request 2
Request 1
Event Queue
Event Loop
(Single Threaded)
Non-Blocking I/O
Tasks Process Here
Node.js Event Loop
Blocking I/O
Tasks
Response Sent
via Callback
Thread 1
Thread 2
Thread n
Internal C++ Thread Pool
Database
File System
Others

Event Loop (Contd.)

- Clients send theirs request to Node.js Server.
- Node.js Server receives those requests and places them into a processing Queue that is known as “Event Queue”.
- Node.js uses JavaScript Event Loop to process each client request.
- Event loop is an indefinite loop to receive requests and process them.

Event Loop (Contd.)

- Event Loop continuously checks for client's requests placed in Event Queue. If requests are available, then it process them one by one.
- If the client request does not require any blocking IO operations, then it process everything, prepare the response and send it back to the client.

Event Loop (Contd.)

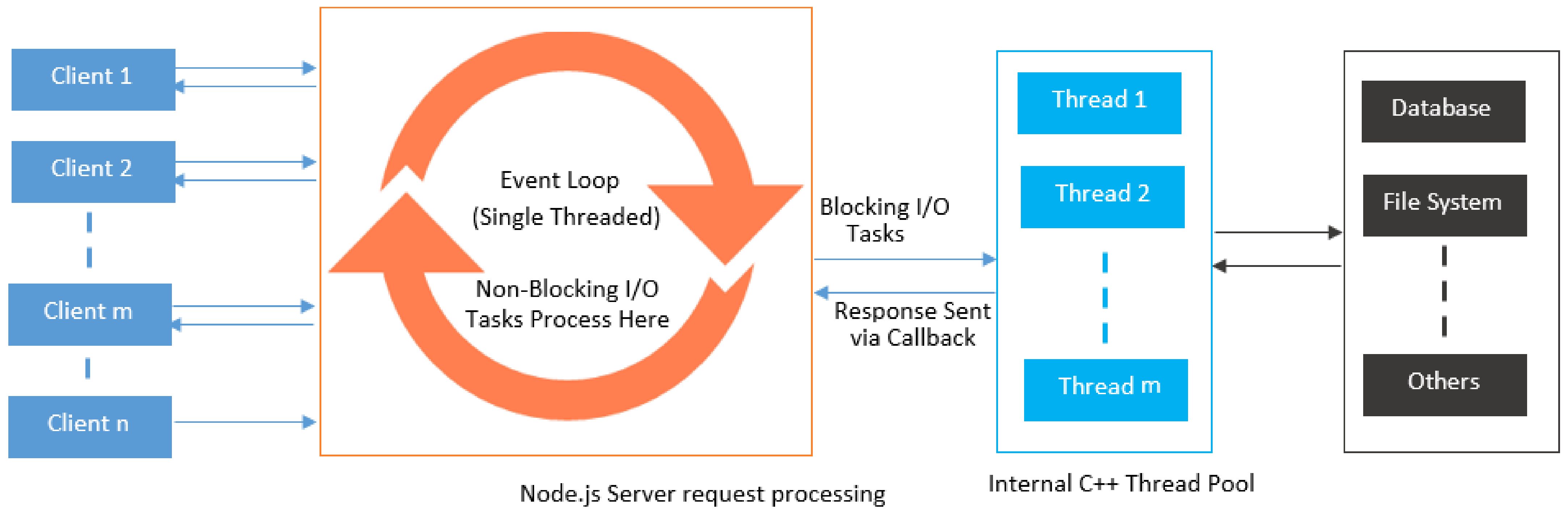
- If the client request requires some blocking IO operations like interacting with database, file system, external services then it uses C++ thread pool to process these operations.

NodeJS vs. others Server Side Frameworks

Node.js Processing

- Node.js is different from existing server-side frameworks because it is based on asynchronous events via JavaScript callback functionality and uses the JavaScript as a programming language.
- Moreover, everything inside Node.js runs in single thread.

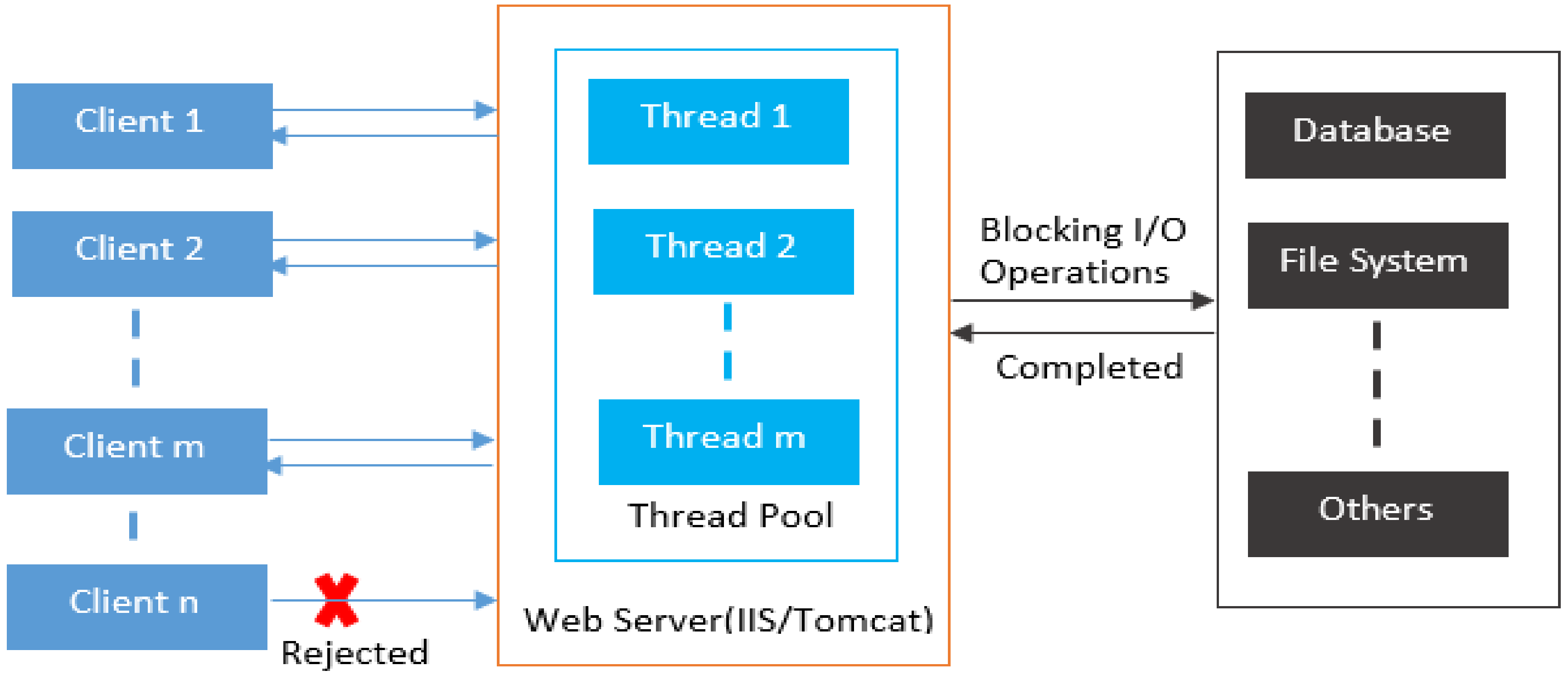
Node.js Processing (Contd.)



Server Side Framework Processing

- While existing server-side framework like ASP.NET, JSP and Php etc. are based on multiple threads web server (IIS/Tomcat).
- In multiple threads system, there is a limit of maximum number of threads, beyond which the throughput decreases.

Server Side Framework Processing (Contd.)



Multi-Threaded Web Server request processing

Issues with Multi-threaded systems

Issues with Multi-threaded systems

- Under heavy load a multi-threaded web server consumes a large amount of memory.
- Most of the time threads wait for some I/O operations to be finish.
- Context-switching and scheduling increases drastically with large number of threads.

Node.js Application Area

Node.js Application Area

- E-Commerce Web Applications
- Social Media Applications
- Real-time Services
- Real-time data Applications like Multiplayer Games, Stock Trading, Chat App etc.
- Data Streaming Applications
- Network Applications
- HTTP Web Server

Node.js Application Area (Contd.)

- High Concurrency Applications
- File Uploading Tools

Who Use NodeJS

Who Use NodeJS

Google

Go Daddy®

Microsoft

YAHOO!

SAP®

LinkedIn

IBM

Walmart

PayPal

DotNetTricks

node JS™

Advantages of Node.js

Advantages of Node.js

- Open Source
- JavaScript as Programming Language
- Scalable - Horizontal Scaling and Vertical Scaling
- Better Performance
- Caching Support
- Lightweight and Extensible
- REST API Support
- Server Development

Limitations of Node.js

Limitations of Node.js

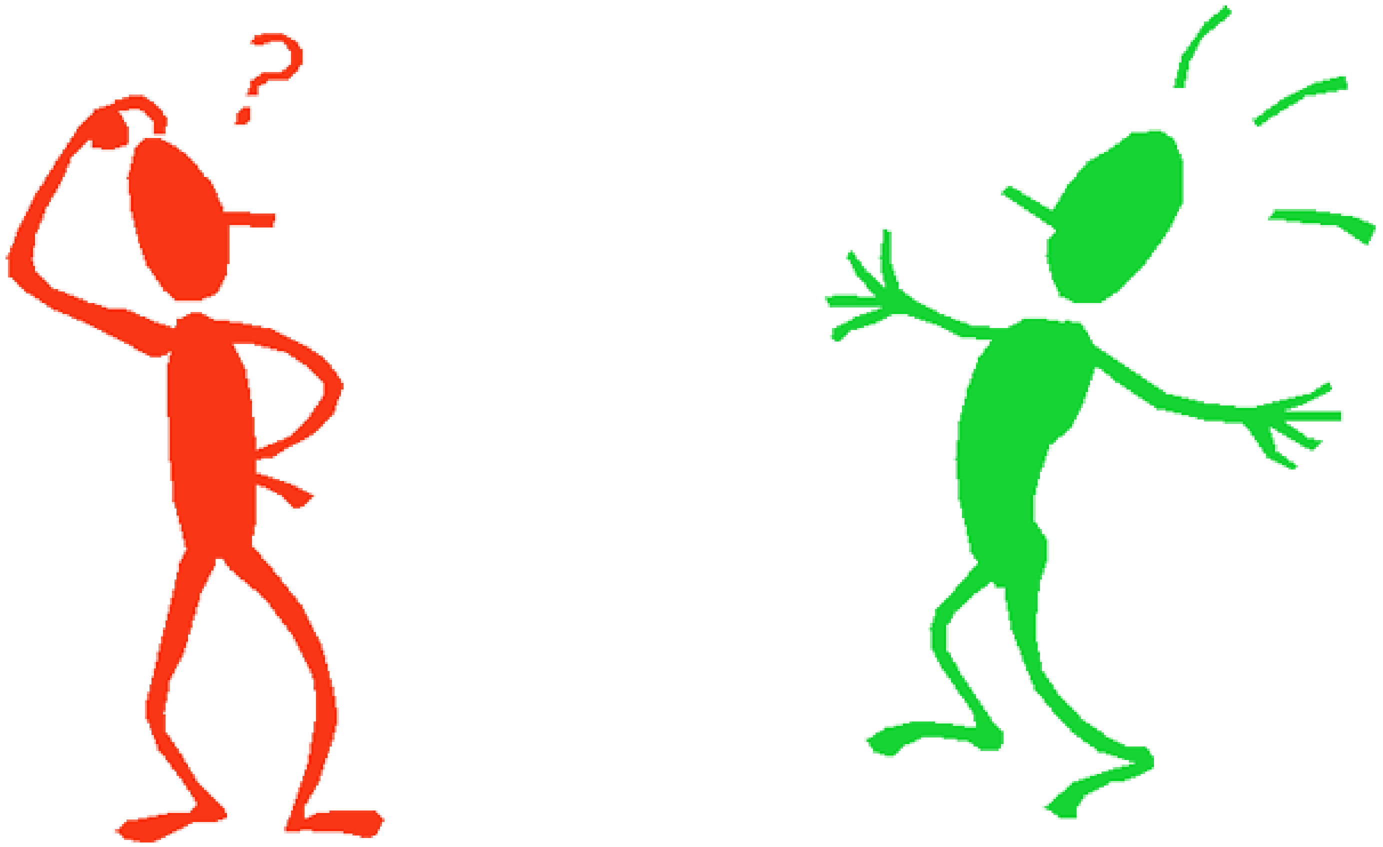
- Not good for multi-threaded programming.
- Not good for executing long running task
- Not good for executing synchronous and CPU intensive tasks.

Node.js Application Deployment Server

Node.js Application Deployment Server

- Node.js app cannot be deployed on your existing hosts like shared web hosting etc.
- You can use VPS or dedicated servers to install node and run your application.
- The easiest way to deploy your node application is to use a scalable service like Heroku.

Q&A



It's the beginning...