Sri Lanka Institute of Information Technology

APPLICATION FRAMEWORKS

NODEJS

LECTURE 04

Faculty of Computing
Department of Computer Science and Software Engineering
Module Code: SE3040

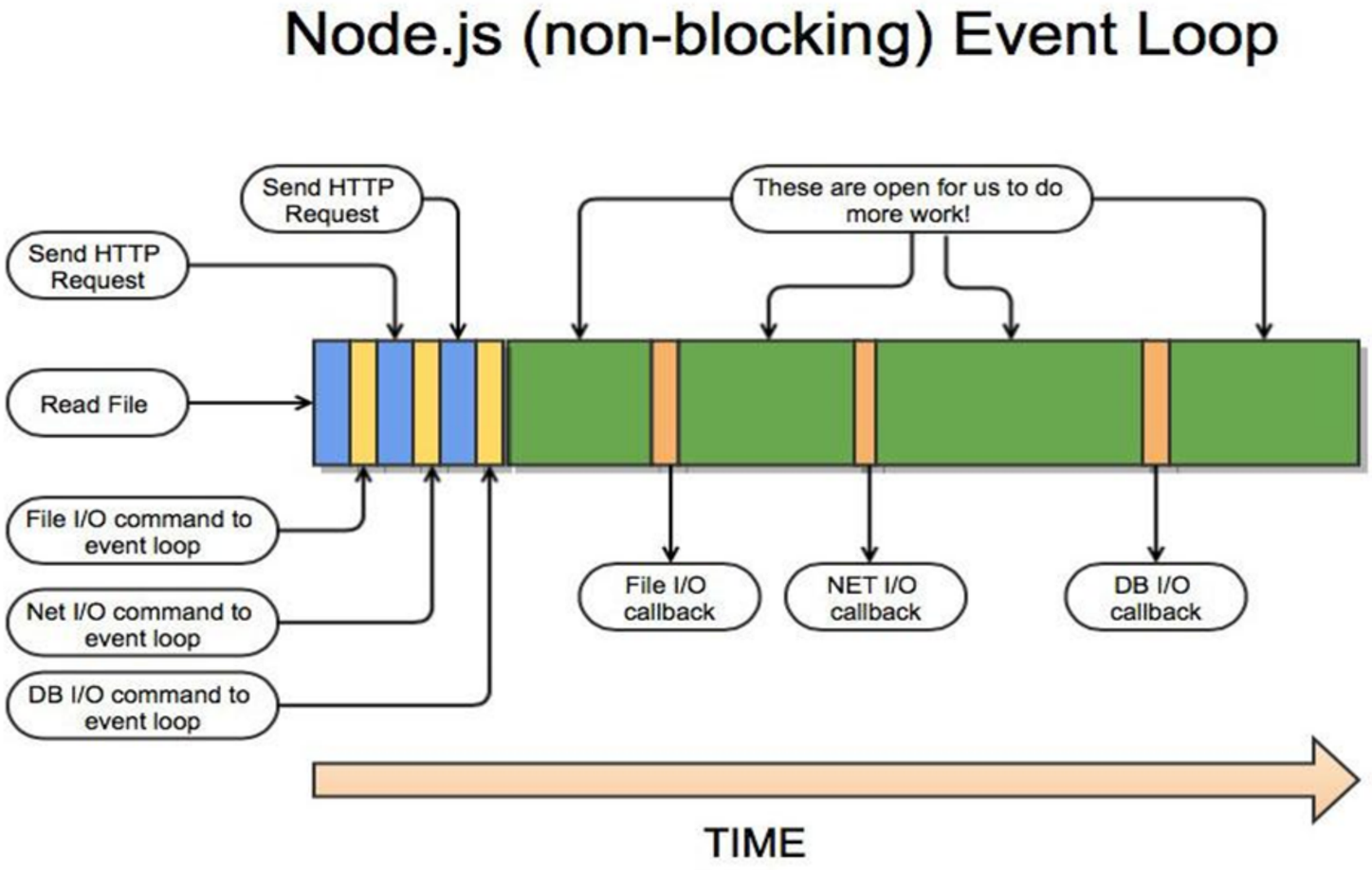
Agenda

1. NodeJS
2. Event loop
3. Use cases
4. Advantages and disadvantages
5. Package manager
6. Require

NODEJS

- Developed by Ryan Dahl.
- Created with the aim of creating real-time websites with push capabilities (websockets).
- NodeJS is an open source, cross platform runtime environment for server-side and networking applications.
- Build on V8 engine, Chrome’s JavaScript engine.
- Uses event-driven, non-blocking I/O model which makes NodeJS lightweight and efficient.
- Ideal for data-intensive real-time applications that run across distributed devices.
- NodeJS comes with several JavaScript libraries that help basic programming.
- NodeJS eco-system ‘npm’ is the largest in the world for open source libraries.

EVENT LOOP


Node.js (non-blocking) Event Loop
Send HTTP Request
Send HTTP Request
These are open for us to do more work!
Read File
File I/O command to event loop
Net I/O command to event loop
DB I/O command to event loop
File I/O callback
NET I/O callback
DB I/O callback
TIME

USE CASES

- Not the best platform for CPU intensive heavy computational applications.
- Ideal for building fast and scalable network applications.
- NodeJS is capable of handling a huge number of simultaneous connections with high throughput.
- For each connection NodeJS does not spawn new Thread causing max out of memory instead handle all in single thread using non-blocking I/O model.
- NodeJS has achieved over 1 Million concurrent connections.
- Bubbling errors up to NodeJS core event loop will cause crashing the entire program.

USE CASES

- I/O bound applications.
- Data streaming applications.
- Data intensive real-time applications.
- JSON APIs based applications.
- Single page applications

ADVANTAGES

- Ability to use single programming language from one end of the application to the other end.
- NodeJS applications are easy to scale both horizontally and vertically.
- Delivers improved performance since V8 engine compile the JS code into machine code directly.
- Performance increased via caching modules into memory after the first use.
- Easily extensible.
- Support for common tools like unit testing.
- Well build ‘npm’ package manager and it’s large number of reusable modules.

DISADVANTAGES

- Even though there are number of libraries available, the actual number of robust libraries is comparatively low.
- Not suitable for computationally intensive tasks.
- Asynchronous programming model is complex than synchronous model.

NODE PACKAGE MANAGER

- Reusable NodeJS components easily available through online repository.
- Build in version and dependency management.
- Build in scripting mechanism.
- Global installations will be available throughout the system while local installations will only be available for that particular application.
- By default all the dependencies will get installed to ‘node_modules’ directory.
- 'package.json’ contains all information related to the NodeJS application. The file be placed in the root directory of the application.
- ‘package.json’ will contain name, version, author, repository, required node version, scripts and dependencies etc.

NODE PACKAGE MANAGER… (CNT)

To denote the compatible version numbers npm has mechanism for defining them;

- Less than or equal ‘<=’, greater than or equal ‘>=’.
- Approximately equivalent to ‘~’.
- Compatible with ‘^’.
- Any ‘*’.
- Any ‘1.2.x’.
- Latest ‘latest’

There are two types of dependencies ‘devDependencies’ (development time dependencies) and ‘dependencies’ (application runtime dependencies).

NODE REQUIRE

- NodeJS follows commonJS pattern when loading modules.
- Require modules get loaded synchronously and will be cached after the first use.
- If the file does not start with ./, ../, or / module is not a core module NodeJS will look the dependency on the node_modules directory.

THAT'S ALL FOLKS !

ANY QUESTIONS ?