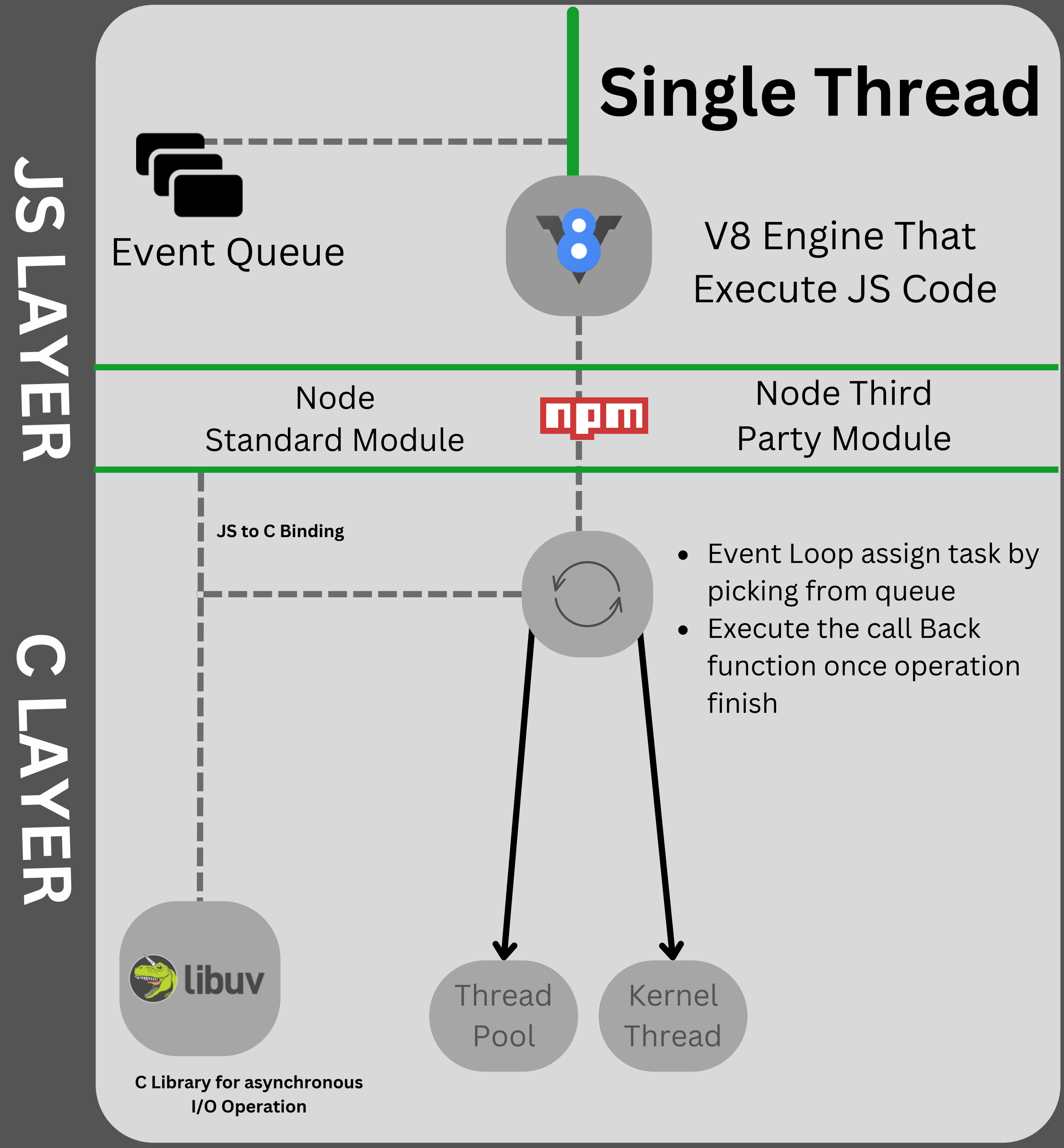
How Node JS works

- Node JS is a single thread language.
- This mean node JS will run on only single thread.
- If server hits 20k hit at same time, the PHP will create a unique process for each which will result in huge memory consumption.
- But Node handle it in single thread and internally managed with help of thread pool and Operating system on execution. So it handle concurrent request more efficiently than traditional.



JS LAYER
C LAYER
Single Thread
Event Queue
V8 Engine That Execute JS Code
Node Standard Module
npm
Node Third Party Module
JS to C Binding
Event Loop assign task by picking from queue
Execute the call Back function once operation finish
libuv
C Library for asynchronous I/O Operation
Thread Pool
Kernel Thread

Asynchronous

Node JS is a asynchronous programming. What really means?

Example:
Consider the below pseudocode

- Line 1
- Line 2
- Line 3
- Read a file of 50 MB.
- Line 5
- Line 6.

In the above example after finishing up to line 3(consider them as non blocking code) then it comes to reading a file. It is considered as blocking code. Why the because file has to read from some place(i.e. hard disk or remote source) which will take time to finish.

These are all some examples of blocking operation
- Reading a file.
- Calling the remote source
- Connection to db.

It all take sometime to finish which will depend on various factors (file size, bandwidth in case of remote connection)

So node JS did not wait for that. It go to next line for execution, after triggering that.
But in other programs like PHP the line will wait until it finish the execution.

What is the advantage?
Instead of waiting for output, meantime it utilize resource for other work.

But wait then how I know when it is was finished and I want to perform some activities after file reading.

Let's introduce our another Hero at next page!!

Event Loop

What it will do?
As name suggest , it will run the event in loop until certain point, until there is no event to listen.

It will check for tasks in event queue and categorize it.
Is it Blocking or Non Blocking task?

- If it is non blocking it execute and return back result.
- If it is blocking it assign the task to thread from thread pool or kernel

Once the operation finishes, it will receive the message, Hey file read was done. The event loop is achieved using **LIBUV** in node JS. It was Asynchronous I/O library in C.

Also some core packages like file operations , network related operations are actually executing in C. Node JS connect to C using binding.

So here also power of C comes - mother of all programming language.

Event loop pick thread from thread pool(Collection of Thread). It assigns task to that thread. (It is actually multithreading here since it is now at C layer not at JS layer). Some operation handled by OS level kernel thread.

I don't want to dive in these thread more deeper because then it goes like ocean.

Call Back Function:

And for each blocking operation, we will attach callback function which actually tells what to do once after finish executing the code. The Event Loop will automatically called that function once the code execution done.

Example:

```
read_file("node js.pdf" , Call back function()
{
   download_file("nodejs.pdf");      -- Will execute after read file is done
})
```

NPM

Node package manager

- It Comes built in with node JS installation
- It is like a central registry or repository where all node package exists developed by many developers around the world

Package is a collection of module that do certain operation
Example:

1. For reading file - File
2. For sending mail - Mail

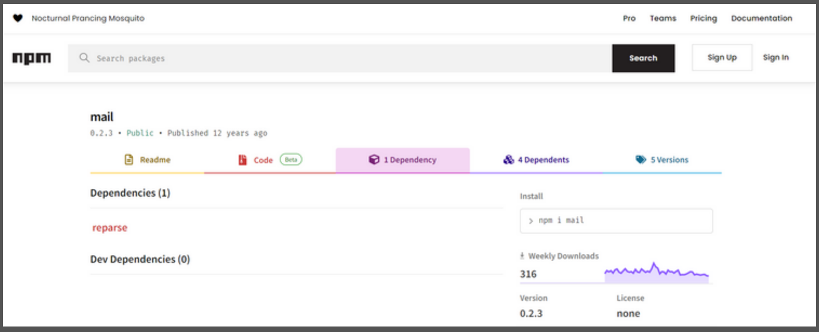
We can use it on our project by installing it.

You can also create your own package and deploy to NPM. Cool!!

- Some Packages are comes with node installation by default, that is called standard package. **Example: HTTP**
- Some we have to install it based on our requirement, that is called third party package

One package may be dependent on another packages also . So without that it cannot work.

Example:
https://www.npmjs.com/



mail
0.2.3 • Public • Published 12 years ago

Dependencies (1)

reparse

Dev Dependencies (0)

Install

```
> npm i mail
```

Weekly Downloads
316

Version
0.2.3

License
none

When Node JS Fail

Since Node JS is a single threaded, it is not recommended for things that require heavy computation and calculation.

Consider the scenario.

There is an application which do video editing on internet. It will require heavy computation

For this application if we go with Node JS what happens?

- Since node is single threaded, if the application accessed concurrently the event queue becomes full and also it not released immediately, since each concurrent access has doing heavy computation which will result in application fail for further access.
- So further request cannot be accepted, since it is single threaded.
- For this type of application Node JS is not recommended approach