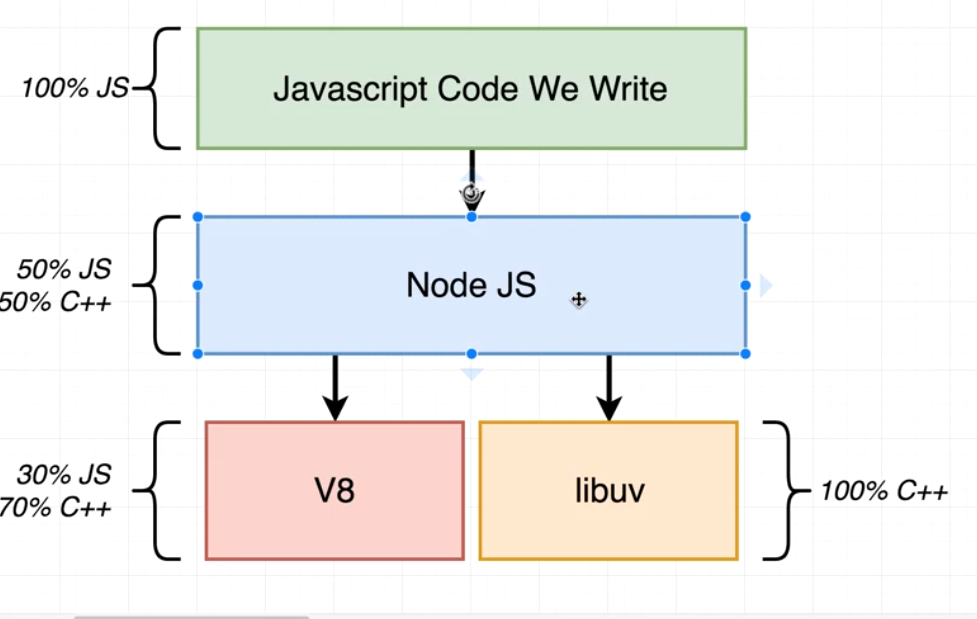
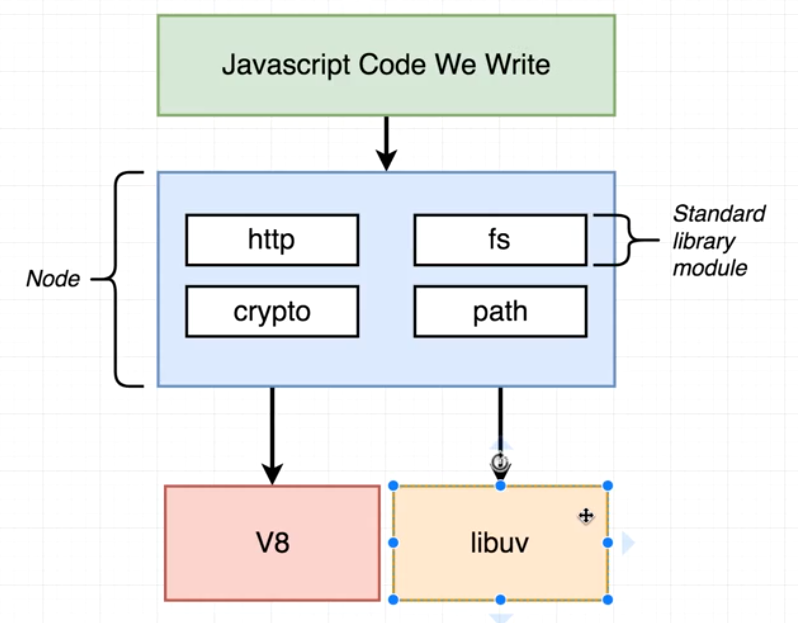
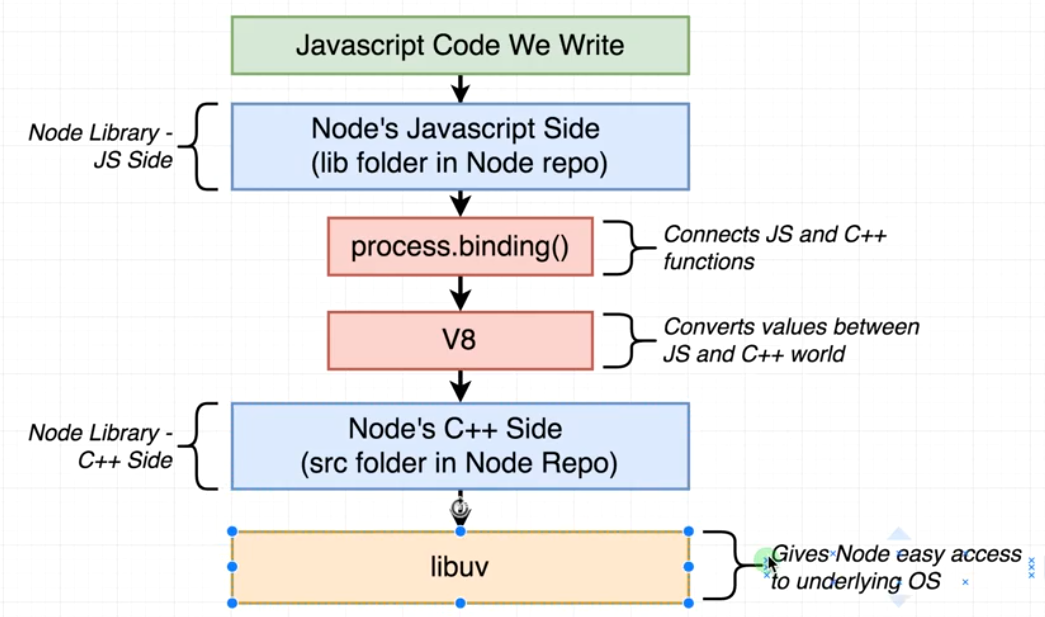
Internal of Nodes:



1. The v8 project is an open source js engine created by google. The purpose of this project is to able to execute js code outside of the browser and that’s
2. The libUV is open source c++ project that gives access to the OS, underlying file system. It gives access to networking and it also handles some aspects of concurrency as well.
3. Whats the use of nodejs then?
4. Bzox v8 is 70% c++ code and libuv is 100%c++ code. But we as a JS developer don’t want to write c++ code. We want to write js code. Sothats the purpose of Nodejs. Node gives us the nice interface to use to relate our JS side of our app to the actual c++ that’s running on our computer to actually interpret and execute.
5. 
6. Nodejs provide us with series of wrappers and a very unified inconsistent API for us to use inside our projects.
7. The v8 project is used to translate the nodejs values that you and I place inside of our different programs like a Boolean, or a function or an object and translate it into their c++ equivalent.
8. So when we write js code we require node modules or node libraries they are depending upon some JS definition which eventually kind of maps up to actual c++ implementation. And on c++ side theres a lot of interoperability b/w v8 project as well as UV project.
9. 

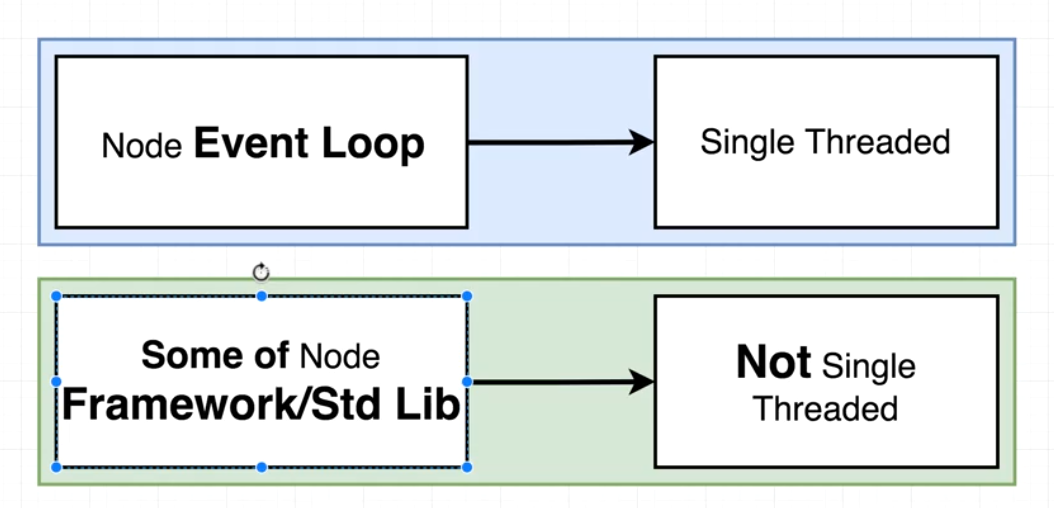
Threads:

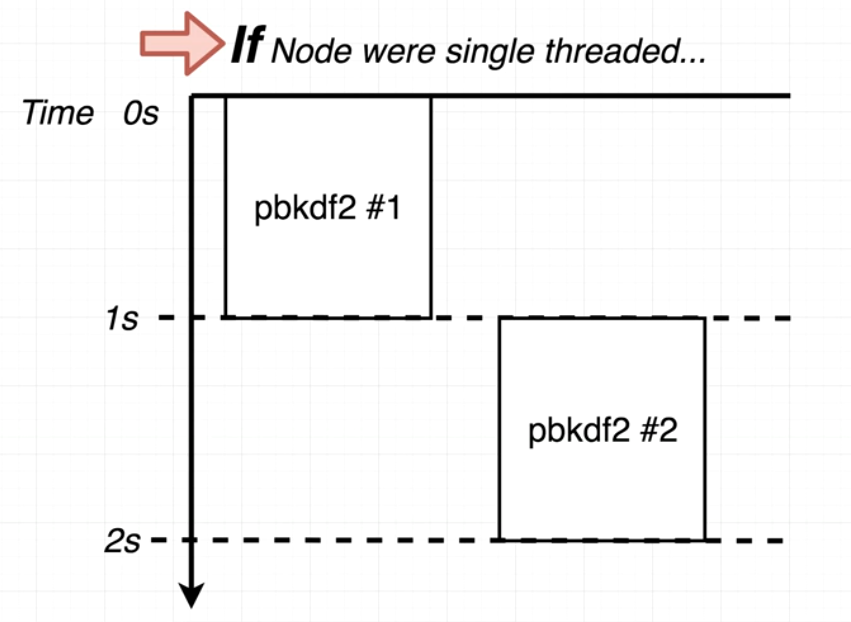
1. Threads are unit of instruction that are waiting to be executed by the CPU deciding which order to execute these threads in is referred to as scheduling.
2. Scheduling is controlled by the OS. 2 ways of improving the rate at which we process
   1. To add more CPU cores to our machine
   2. To allow our OS schedular to detect big pauses in processing time due to expensive input and output operations.

Event-Loop:

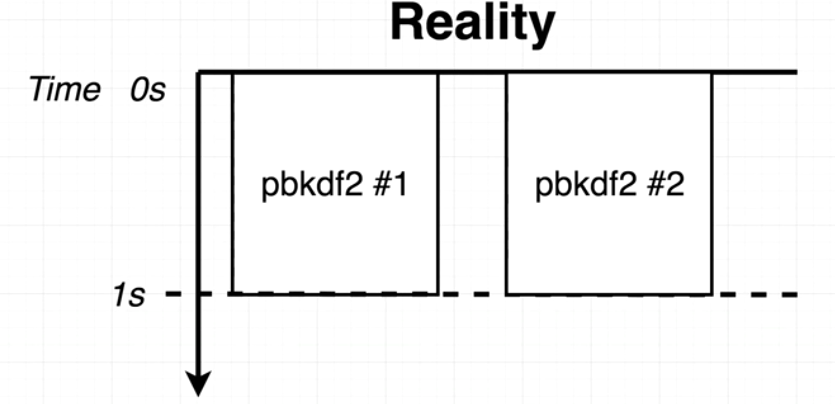
1. A node program is run inside a thread. Different nodejs program required diffreretn number of threads. Inside thread there is 1 event-loop. EL is as being a control structure that decides what our one thread should be doing at any given point of time. Every program has only 1 EL.
2. Now c code to have better understanding.

IS NODE SINGLE THREADED:

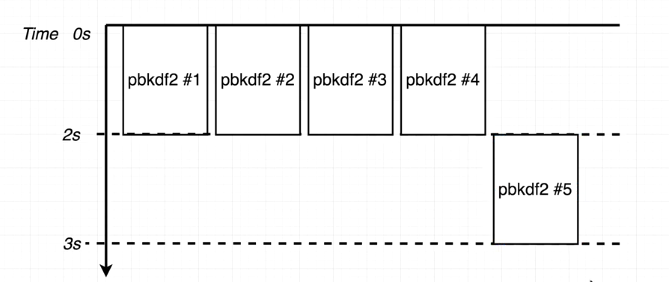




If nIf node was single threaded, then func#2 would have started after func#1 gets finished but the readlity is..

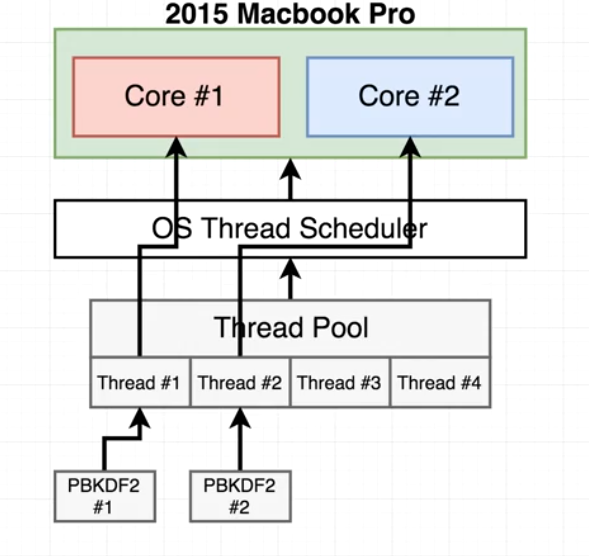


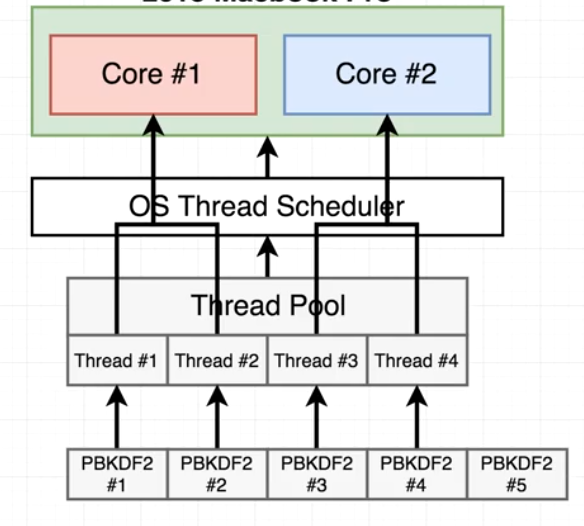
Suppose we 5 func instead of 2, then 1st 4 gets executed within 1s and the 5th gets executed by taking a pause and has time greater than 1s.

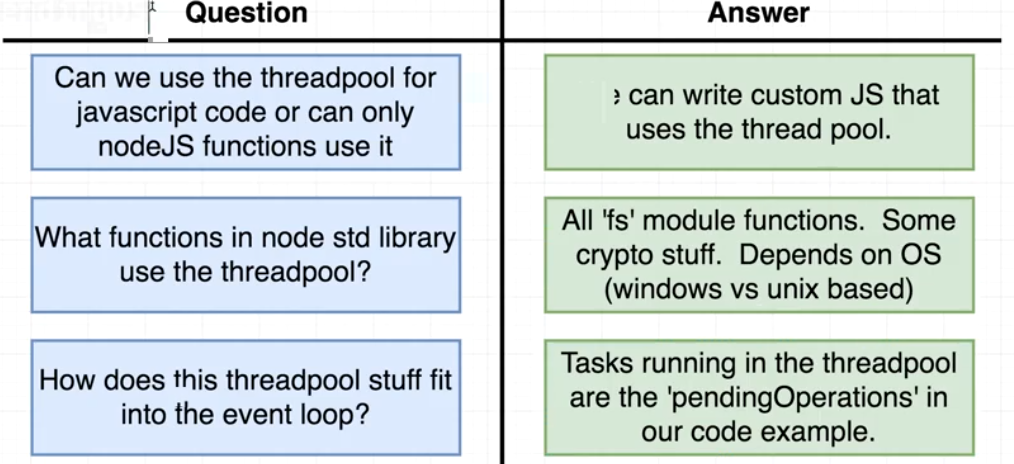


For us it’s 0-1s for the 1st 4 operation and and greater>1 for the 5th operation.

1. When there are 2 hashing functions, each function get assigned to 2 different threads and that thread gets connected to different cores(supposing that our CPU is dual core.) So both took more or less exactly one second for each core to finish the hashing function call.



1. When there are 2 hashing functions, each function get assigned to 2 different threads and that thread gets connected to different cores(supposing that our CPU is dual core.) So both took more or less exactly one second for each core to finish the hashing function call.
2. 
3. The 1st 4func took nearly around double time to complete and then the 5th func came to picture and it took 1s



Uptill now we have discussed the threadpool that is automatically created by the libuv any time we start up our node program that explains the presence of this pending operations[] inside our event loop.

NOTE: WHAT IS CALLBACK FUNCTION

1. Suppose in a restaurant we have one server, one chef and one customer arrives. Now server will take order of customer and report it to chef. Here can now arise 2 scenarios
   * 1. The server could wait for the chef to make the order and then serve it to the customer.
     2. The other scenarios is the server could take order of other customer and as soon as the1st order is made by chef he serves the order to the 1st customer

So which is feasible ofcourse the 2nd one, this is what callback means.



So in short a callback func is another func that can be passed as an argument to another function and it is usally invoked after some kind of event, in our case it is going to be 5 second.

Now lets start with PendingOStasks[].

1. Firstly deriving the benchmark as done previously with the (hash func).
2. Next we are going to derive some understanding out of that benchmark.
3. 
4. So remember by default the thread pool has 4 threads which means only 4 task can be processed at a time and the 5th task was taking a 1s extra to execute.
5. But in this all the six task all completed simultaneously.(But in my machine only 3 task were completed simultaneouly).
6. Becoz, libUV sees that we are attempting to make a HTTP request. Neither libUV nor node code has any code to handle all of this super low level operations. Instead libuv delegates the request making to the underlying OS.
7. So its actally our OS that does the HTTP request , libUV is used to issue request and then its just wait for the OS syste to emit a signal that some response come back to the request.

