NODE JS

1st Feb 2024

- It is a JS runtime.

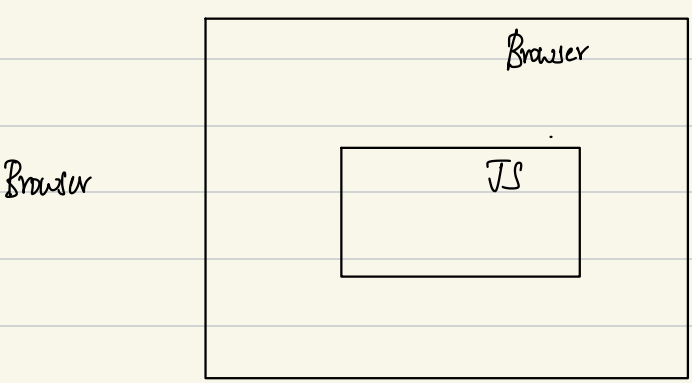
Runtime :

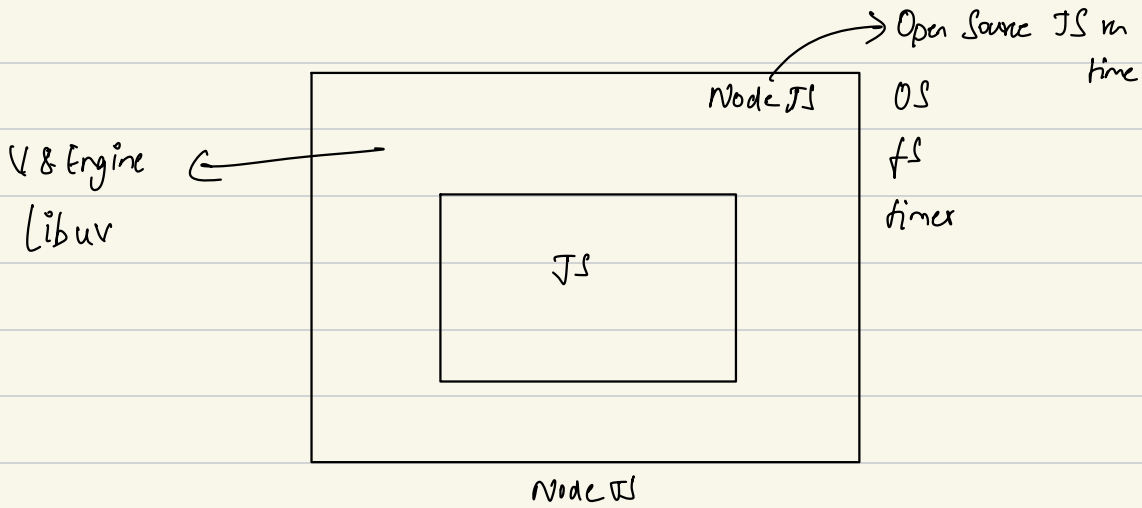
- Enivorment which provide necessary infrastructure for code to execute.

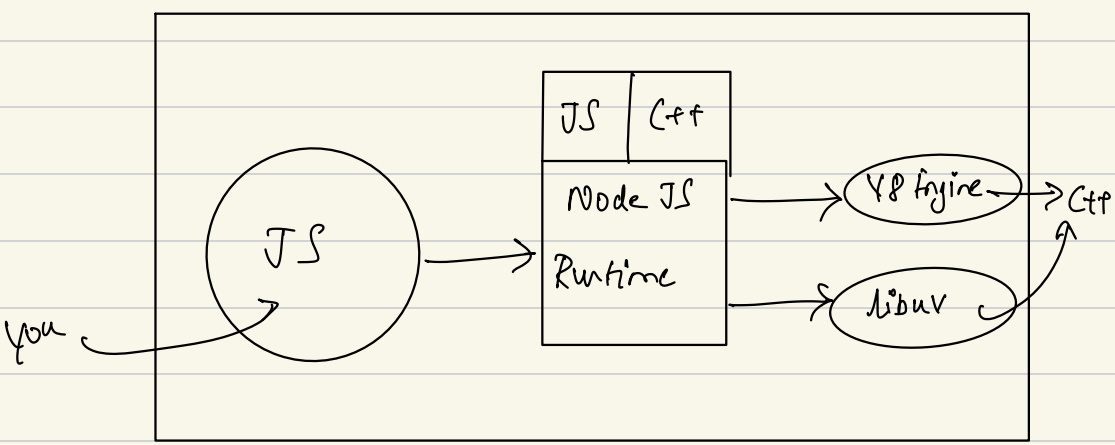
- Browser is also a runtime.

Browser :

- Accessing DOM tree
- Access CSS on tree
- XML HTTP Request (Network)
- Timer







folder

lib → JS

src → C++

Internal of setTimeOut :

```
setTimeout(cb, 0);
```

timer.js (lib/timer.js)

- Check the flow of the code in github.

- if you set a timer of 0ms, it is 1ms, check the timeout class code.

How V8 engine works :

Responsibilities of V8 :

- Compilation
- Garbage collection
- Execution
- Memory Mangement

- V8 doesn't contain browser features or OS features.

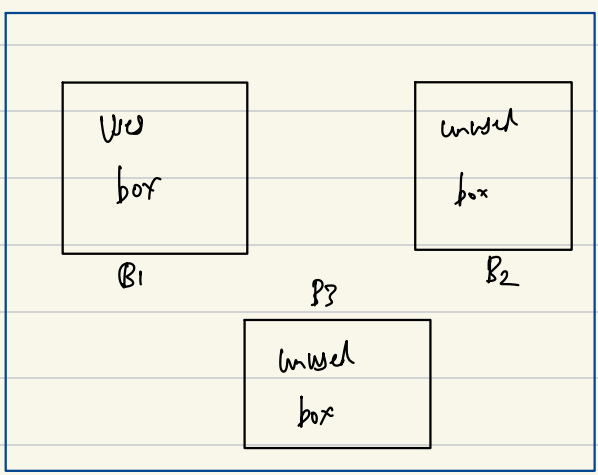
Memory Mangement :

- Call Stack (Single threaded)
- Heap Memory

Garbage Collection :

- Orinoco

Orinoco :



- B2, B3 you remove it as it's unused box

- Orinoco follows color coding system

Black, Grey, White

S-1

Black — Completely checked, Revelant Stuff

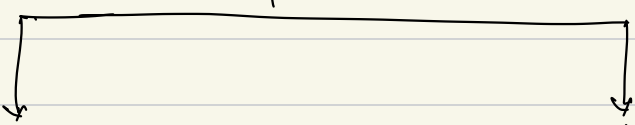
Grey — Have checked, Something might needed

White — Haven't checked, may be needed or not needed

S-2

Black Allocation — Mark as safe

Segerattion of memory



Young generation

- Newly created obj
- Young generation dies early.

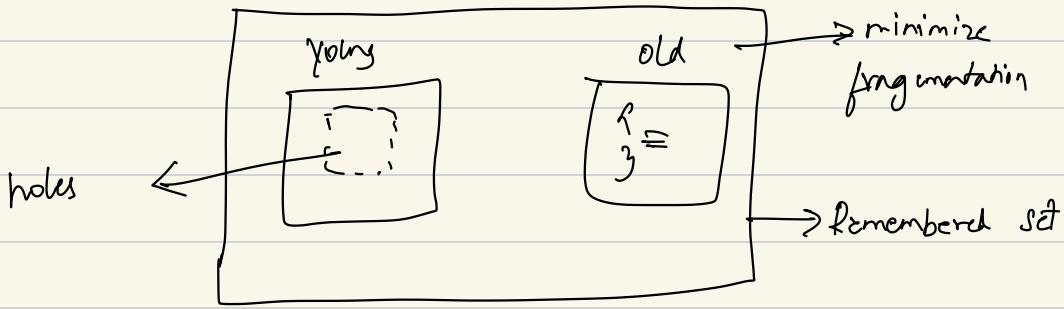
old generation

- Old Obj that may survive multiple garbage collection

Movement :

The young generation which survive from one pass

are moved to old generation.



Compilation : (Check the ppt)

- Ignition
- Turbofan

Feb 06-02-24.

Q:

→ Is nodejs single threaded ?

→ is event loop part of nodeJS or it get it from someone ?

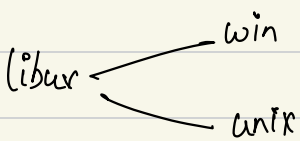
Threading nature of NodeJS

- Node JS is extremely powerly
- NodeJS runs on a single thread - Not completely true.

- Libuv comes into picture.

Libuv :

- It's written in C
- It provide cross platform support
- It is main driving force to make NODE JS support async programing.



- Support of event loop.

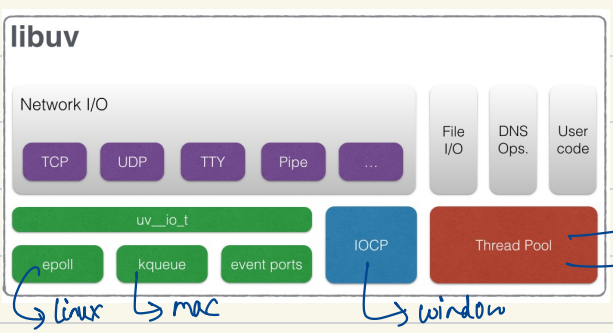
unix ⟶ core.c ⟶ uv_run (whole logic of event loop) # 415

win ⟶ core.c ⟶ uv_run

- TCP connection / TCP socket

- UDP connection
- File I/O (fs package)
- DNS resolution
- Thread pool (most imp feature)
  - ↳ gives power of multithreading
    initial value → 4 (configurable)

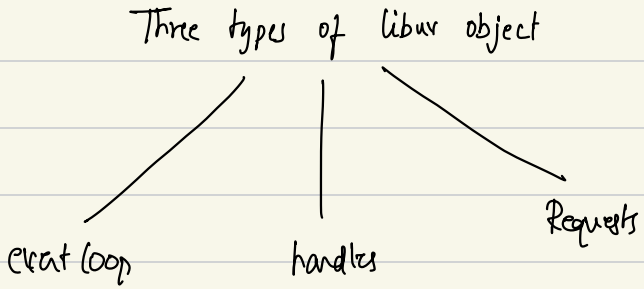
Archetiure of libur:



UV_THREADPOOL_SIZE (config)

JS → single thread

Based on runtime → multithreading (power of runtime)



Handles :

- This are long lived resources. Ex: TCP socket
- Handle are the main objects that keep node js alive.

Request :

- They are short lived resource. Ex: file I/O

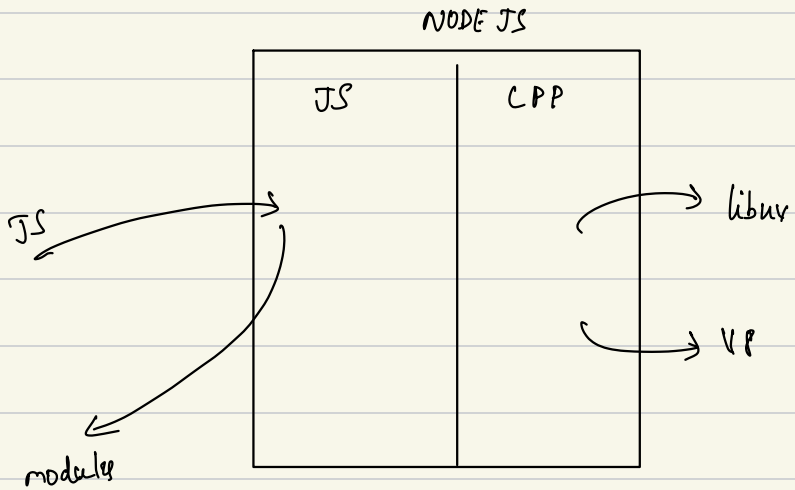
- Libuv enables access to O.S. for node js.

like epoll / k-queue / IOCP

this layer is for event handler

for eg. on click of btn what event occurs.

- NOT every operation of libuv access the thread pool.
  for e.g. file I/O or DNS ⟶ thread pool
  TCP/UDP connection ⟶ you don't use thread pool



↳ JS reusable pieces of task doing one particular task } — file I/O → fs package

- JS layer lives in lib folder
- C++ converted stays in src folder.

lib ⟶ fs.js (file System) ⟶ copyFile (function)

binding: from JS layer it call C++ layer