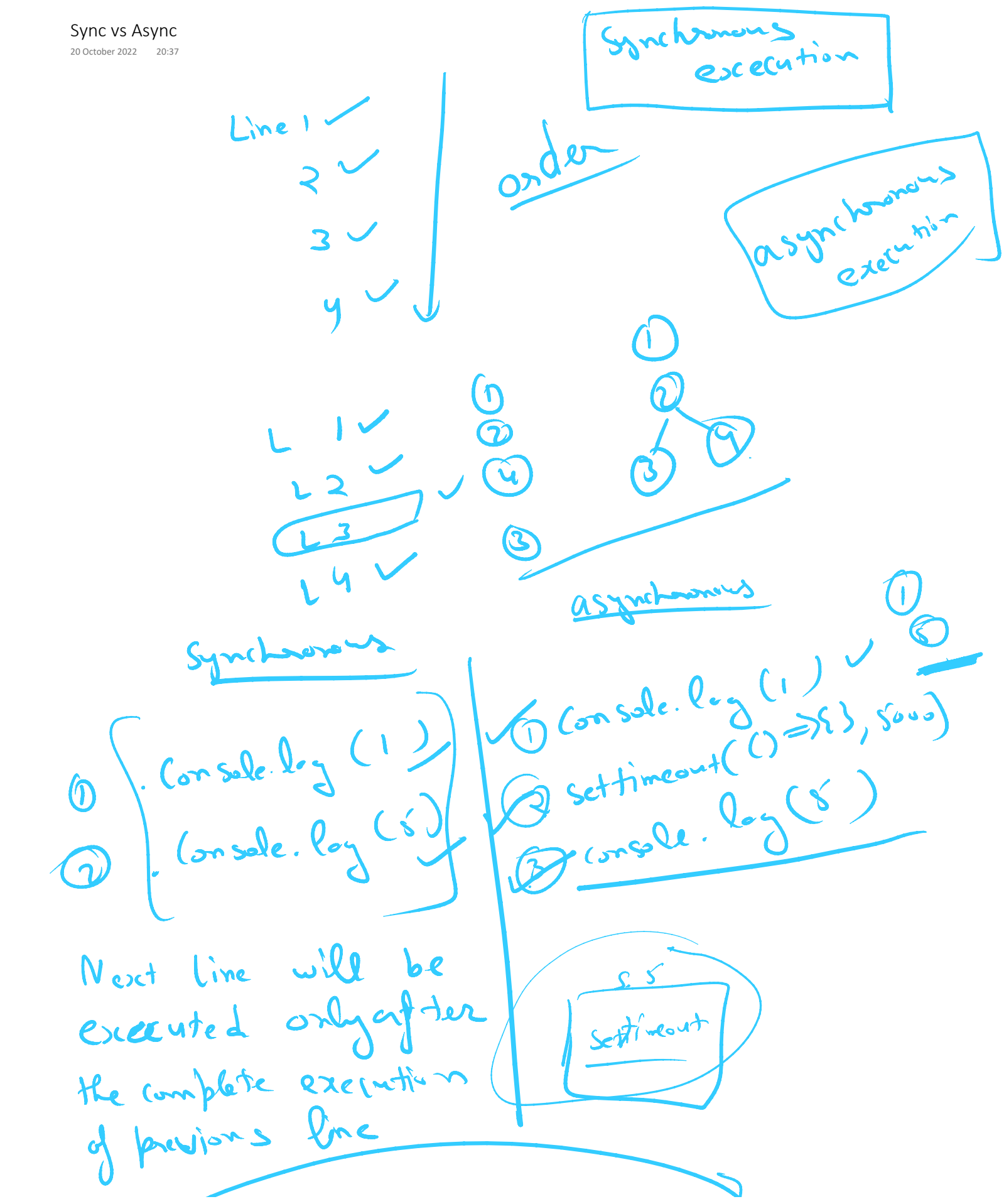
Sync vs Async

20 October 2022 20:37



of previous ...

```
function Print ( ) {
    setTimeout()
    console.log ("hello")
}
```

```
console.log (1)
Print ( )
console.log (5)
```

1

hello

5

```
console.log (1)
console.log ("hello")
console.log (5)
```

unordered way

```
function a ( ) {
    console.log (a)
    b ( )
}
```

```
function b ( ) {
    setTimeout (() => {
        console.log (b)
    }, 5000)
}
```

main

parallely

"c"

"d"

"a"

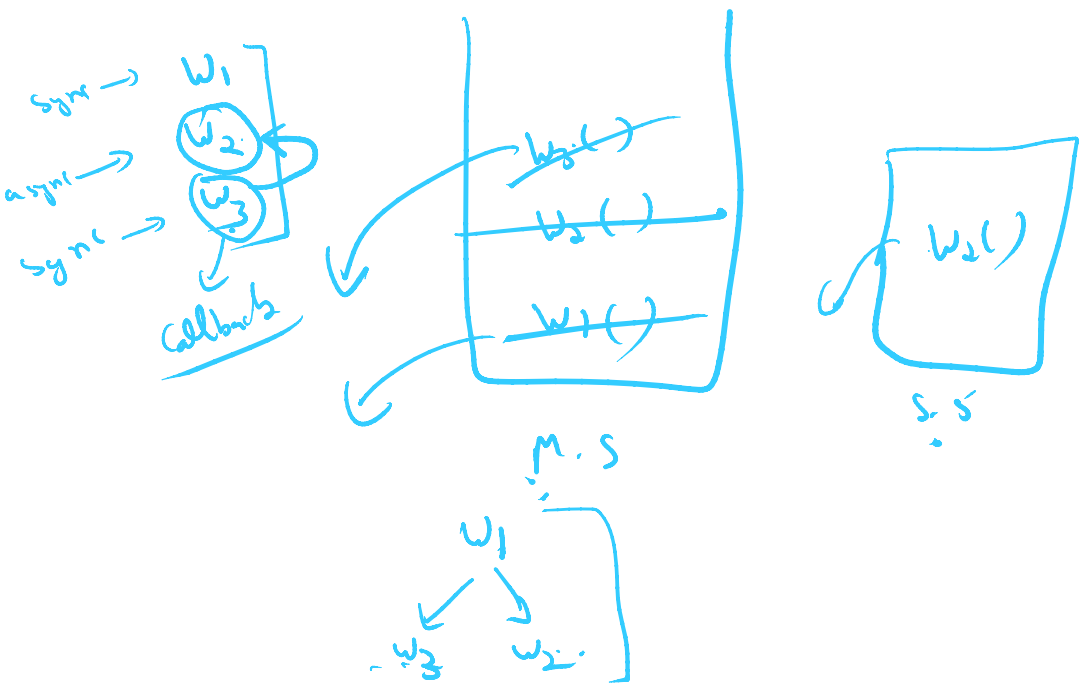

console.log(c)
console.log(d)
a()
b()
"a"
setTimeout
setTimeout
S.S
"c"
"d"
"b"

Callbacks

20 October 2022 20:49

async

It will not get executed in the main stack but will be called in the main stack only



With every asynchronous work, you can attach another work to be done after the completion of asynchronous.

Amazon

to perform order

time

1) Abi to perform order
   ↳ is async

2) Success page

---

function w1 () {

async function w2 () {

w2()

function w3 () {

}

}

w1()

W2 (W3)

The function which get passed as an argument to another function and get called in the end, is known as

Why

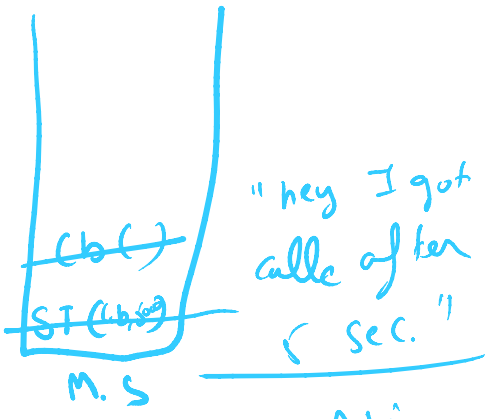
async function executed
need

To get async function exec... in a synchronous we need

call backs

1) setTimeout (callback, 5000)

2) function callback ( ) {
   console.log("hey I got called after 5 sec.")
   }



1) It will ask from C.B Queue, do you have any function
2) It will ask from M.S are you free?

Event loop

true

---

1) async function with its callback get called inside Main Stack
2) But as it was an async function it will be executed inside Web Apis
3) After the complete execution of async function, callback will be sent to callback Queue

3) callback will be sent to Callback Queue

4) There is one event loop asking from Main Stack are you free? and from Callback Queue, do you have any function to execute?

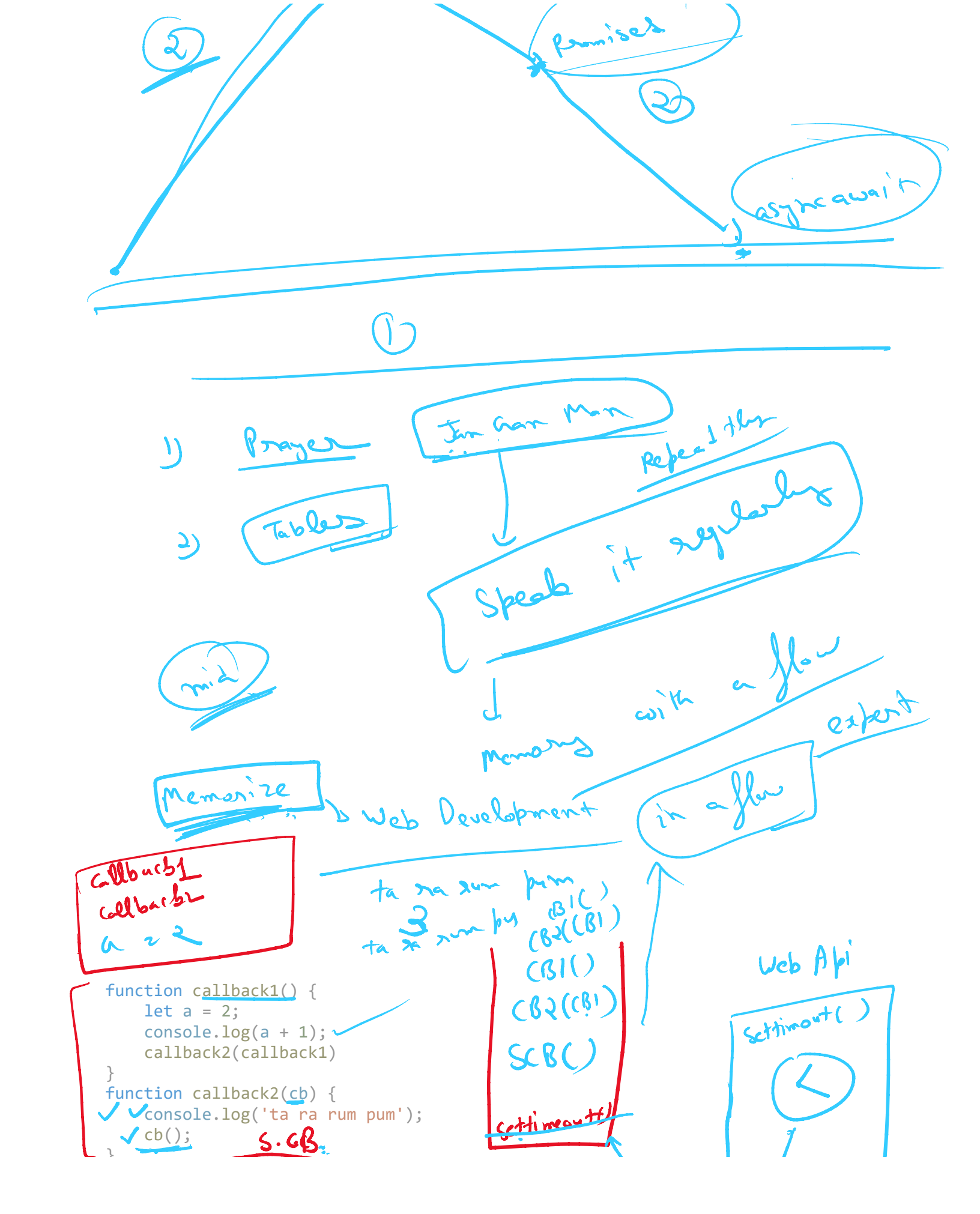
Parallely

5) If both the above conditions are true, Cb will be sent to mainstack to execute.

6) Finally call back will be executed.

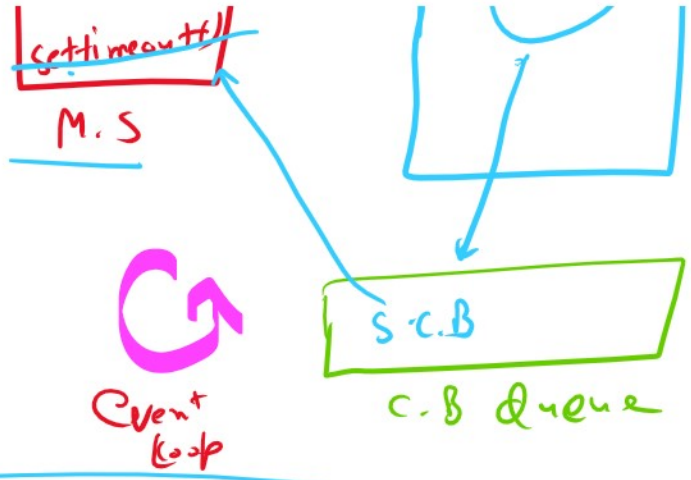
```
function callback1 ( ) {
    let a = 2;
    console.log(a+1);
    callback2(callback1);
}
function callback2 ( cb ) {
    console.log("Ta Ra Rum pum pum");
    cb()
}
setTimeout(() => { callback2(callback1); },
5000
)
```

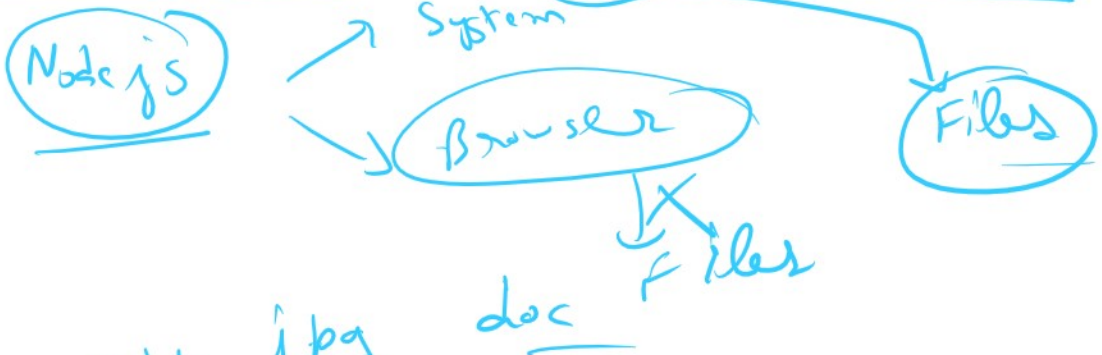
Infinite


2
Promises
3
async await
1
1) Prayer
2) Tables
Im non Man
Repeat this
Speak it regularly
mid
Memorize
Memory
with a flow
expert
Web Development
in a flow
callback1
callback2
a = 2
ta ra rum pum
3
ta * rum pum
CB1()
CB2(CB1)
CB1()
CB2(CB1)
SCB()
settimeout()
Web Api
Settimout()
function callback1() {
let a = 2;
console.log(a + 1);
callback2(callback1)
}
function callback2(cb) {
console.log('ta ra rum pum');
cb();
S.CB

```
    console.log('ta ra rum pum');
    cb();
}
setTimeout(() => {
    callback2(callback1);
}, 5000);
```

S.CB





mp4 webb jpg doc

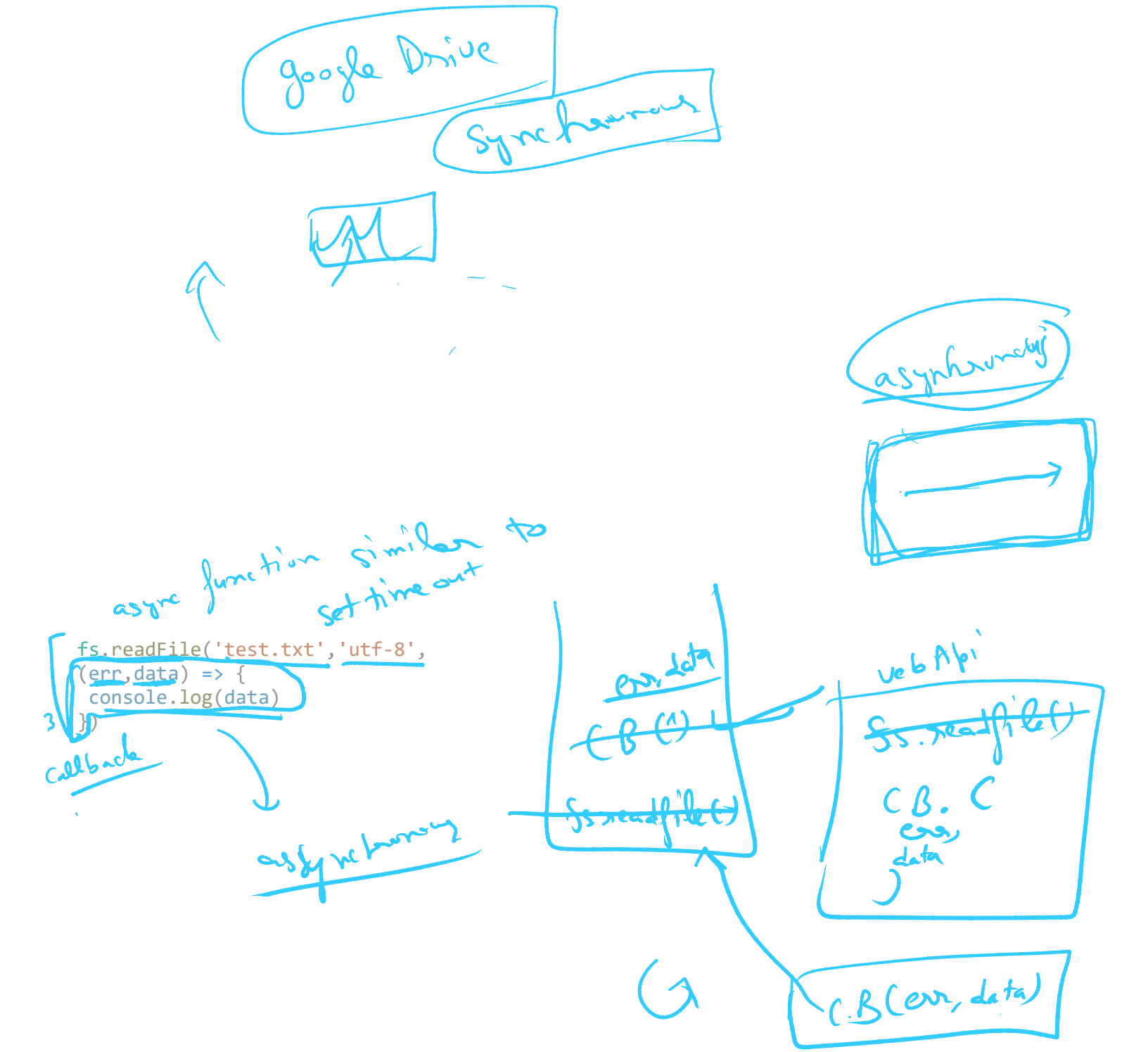
pdf Png

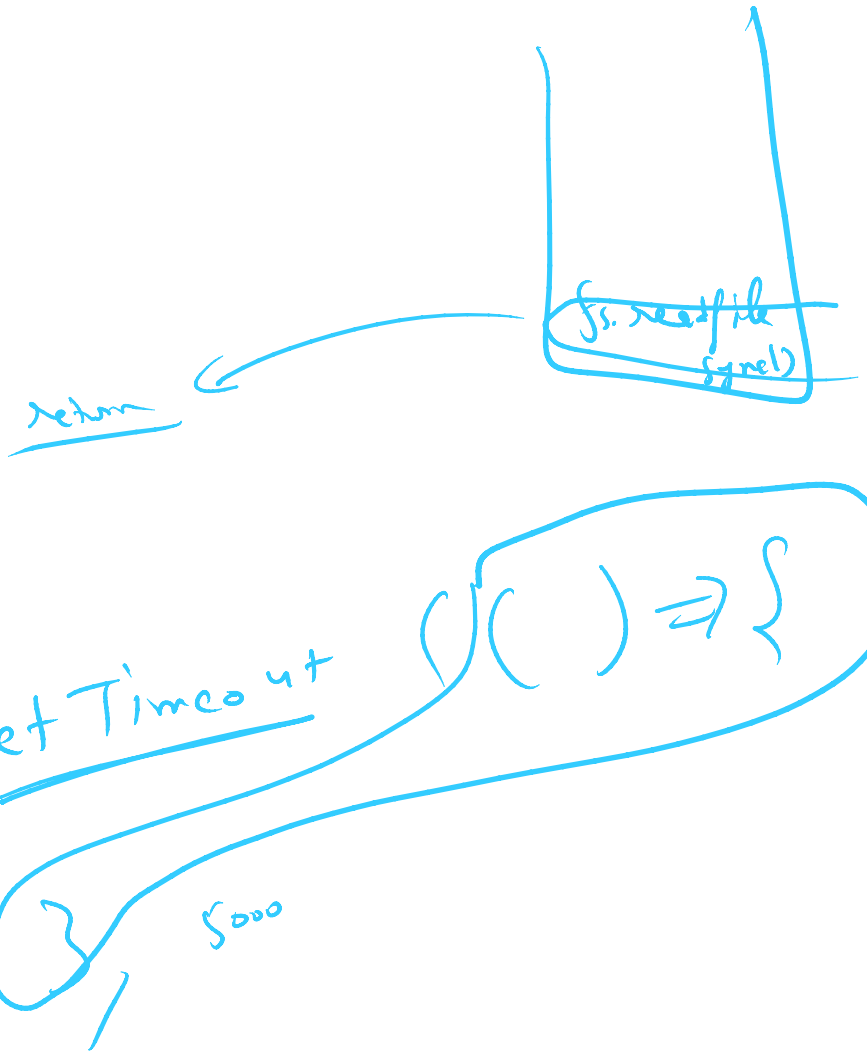
↓

different language

Buffer is also one of the format

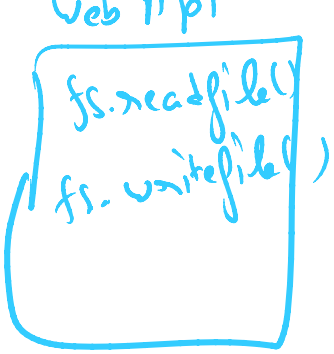
Utf-8


google Drive
Synchronous
asynchronous
async function similar to
setTimeout
fs.readFile('test.txt','utf-8',
(err,data) => {
console.log(data)
})
callback
async behaviour
err, data
CB (1)
fs.readFile()
Web Api
fs.readFile()
CB. C
err,
data
C.B(err, data)



1. Access the file
2. Start Reading Data
3. End of Reading Data
4. Return data to CB.

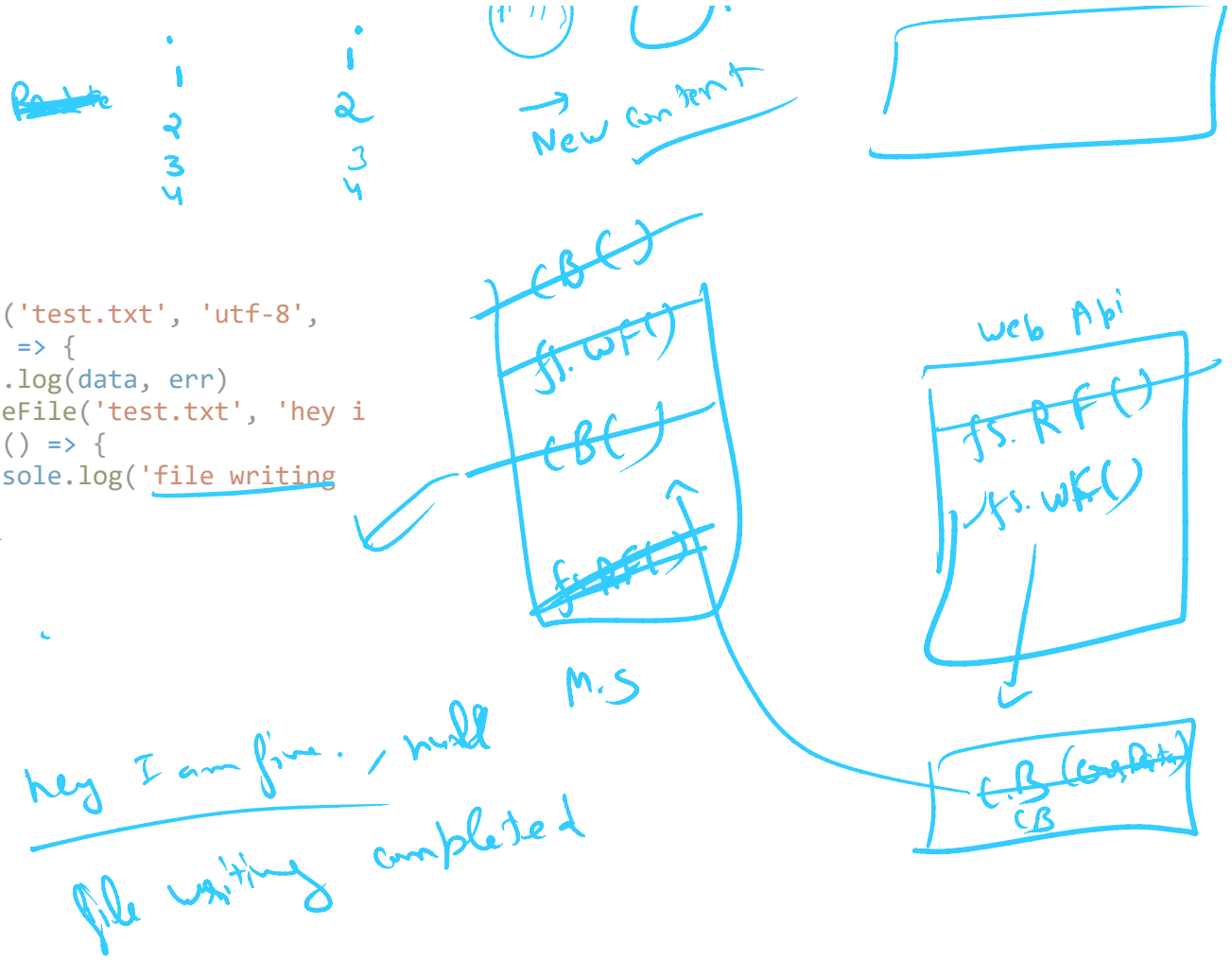
1. Access the file
2. Empty the file
3. Start writing the file
4. End writing the file

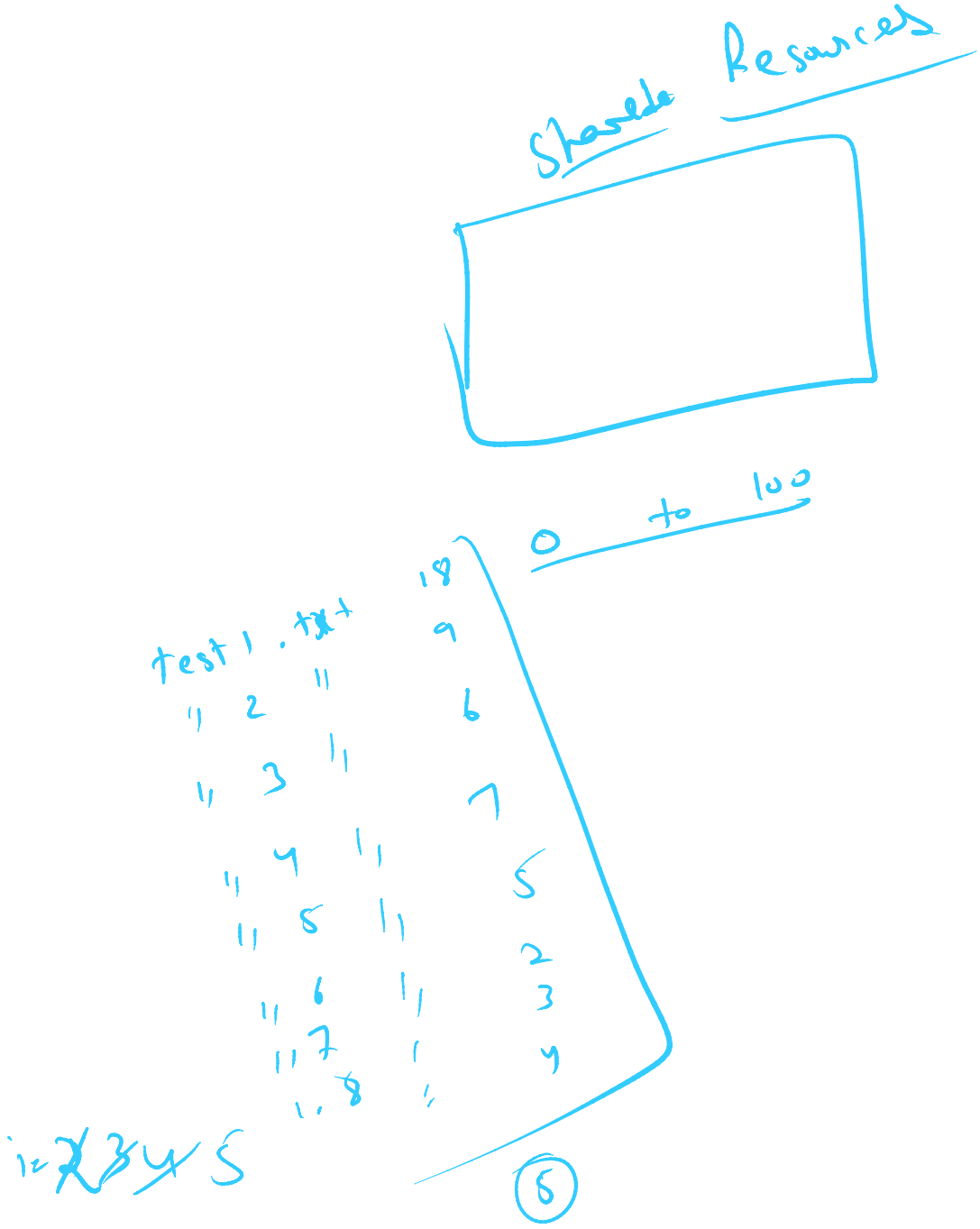



```
fs.readFile('test.txt','utf-8',
(err,data) => {
 console.log(data,err)
})
fs.writeFile('test.txt','hey i
am fine.',() => {
    console.log('file writing
completed')
})
```

File

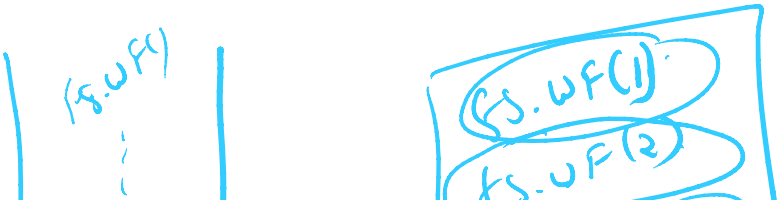
```
fs.readFile('test.txt', 'utf-8',
(err, data) => {
    console.log(data, err)
    fs.writeFile('test.txt', 'hey i
am fine.', () => {
        console.log('file writing
completed')
    })
})
```

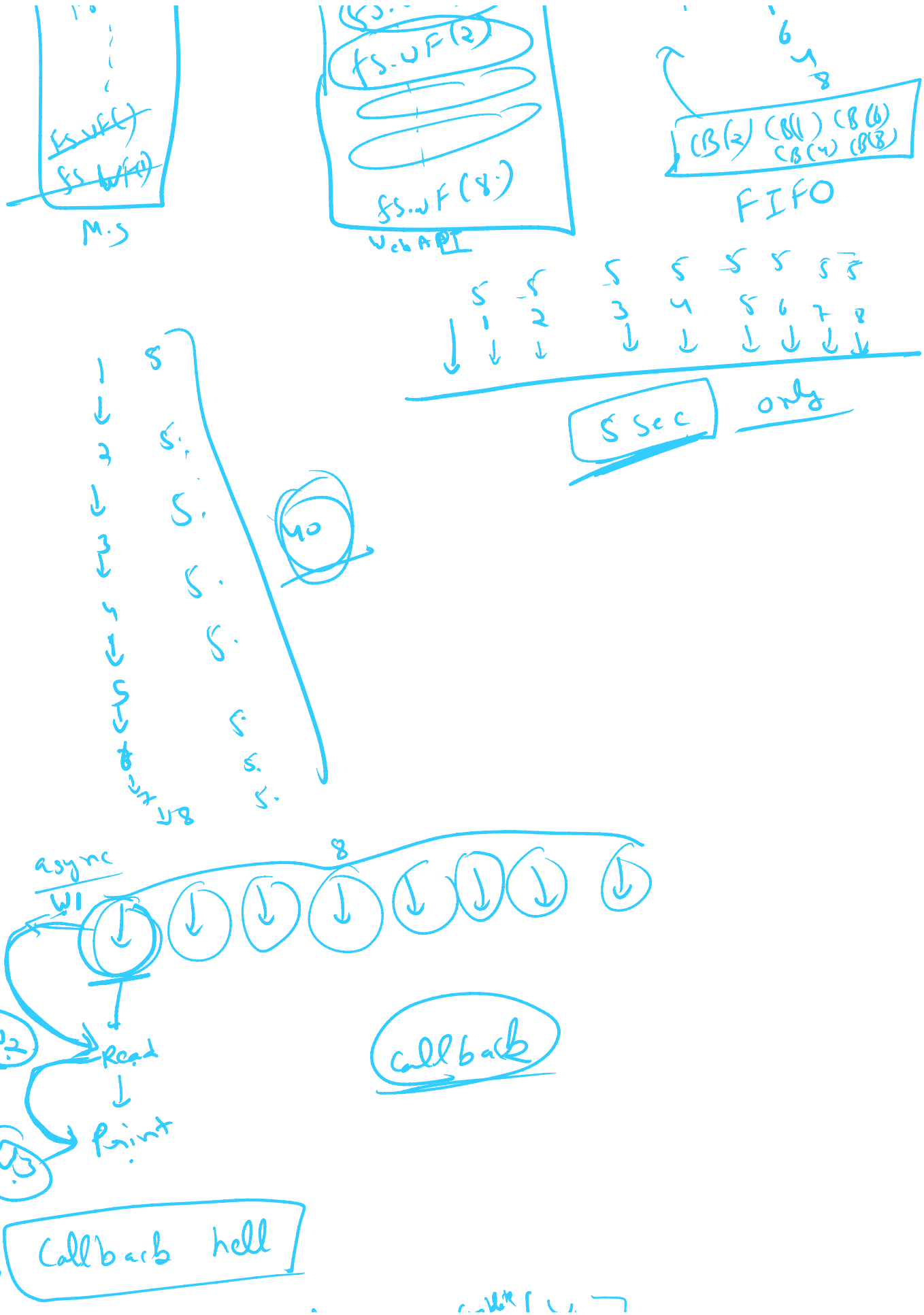


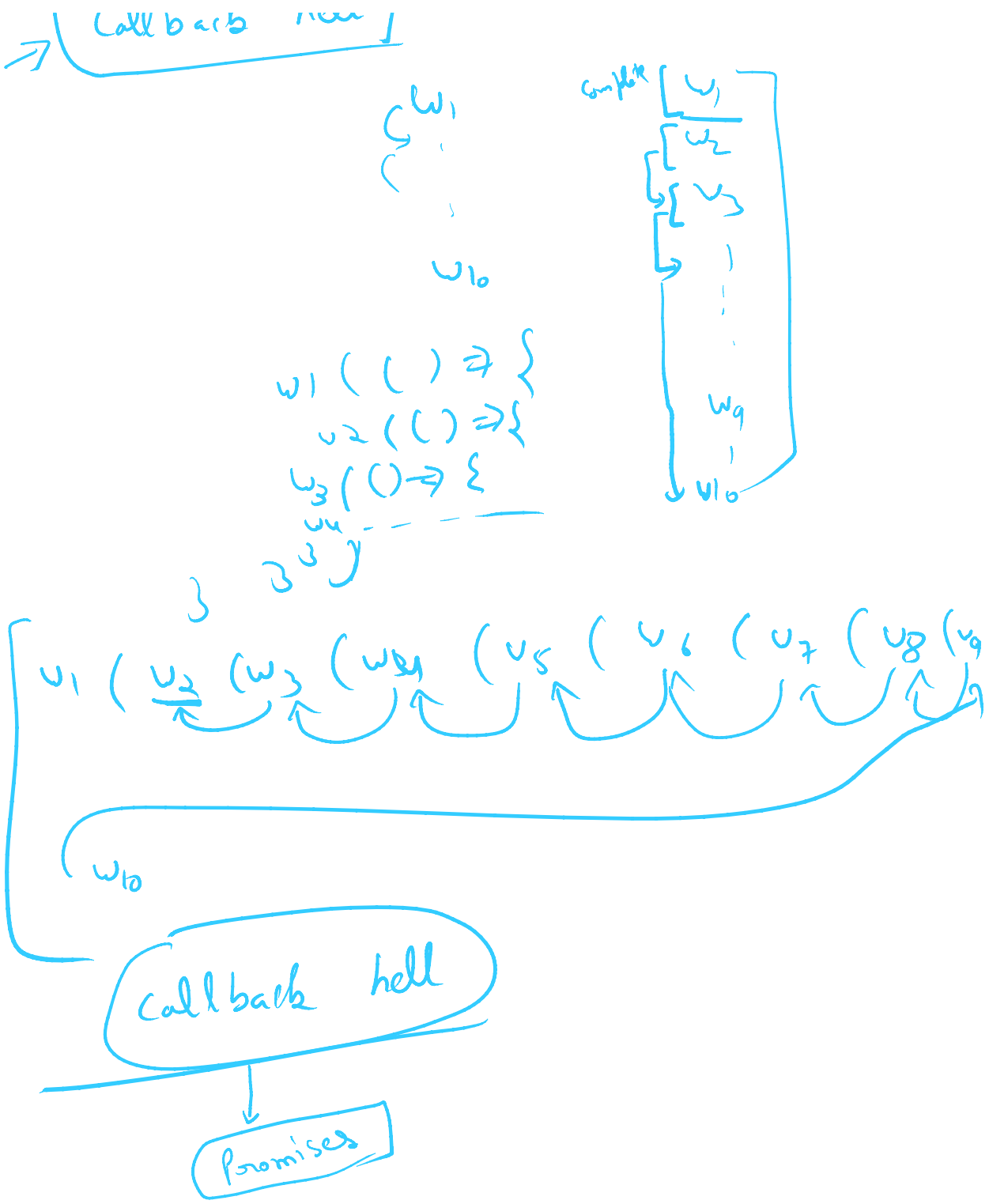



```
for(let i= 1; i<= 8; i++) {
    fs.writeFile(`test
${i}.txt`,JSON.stringify(Math.round(Math.random()*100)), () => {
        console.log(`file ${i} wrote successfully.`)
    })

}
```




fs.uf(2)
fs.uf(4)
M.S
fs.uf(2)
fs.uf(4)
Web API
FIFO
5 Sec
only
40
async
W1
W2
W3
Read
Print
callback
Callback hell


callback hell
Promises

Promises

25 October 2022 20:43