

NODE JS = Java Script

sequence ← asynchronous  
nahi  
follow  
hati

sequential  
ek ke baad ek

```

1 2 3
| | |
|_|_|
delay ← setTimeout

- console.log (1) ;
ms ns - setTimeout(function a() { console.log (2) }, 2000);
ns ns - console.log (3);






```

output

1 2 3 X  
 (1 3 2) ✓

↓  
 Jab utna  
 time nikal  
 Jayen  
 to kya  
 Karein

↑  
 time  
 interval

a	b	c	d	e
				

python  
9 sec


java  
3 sec

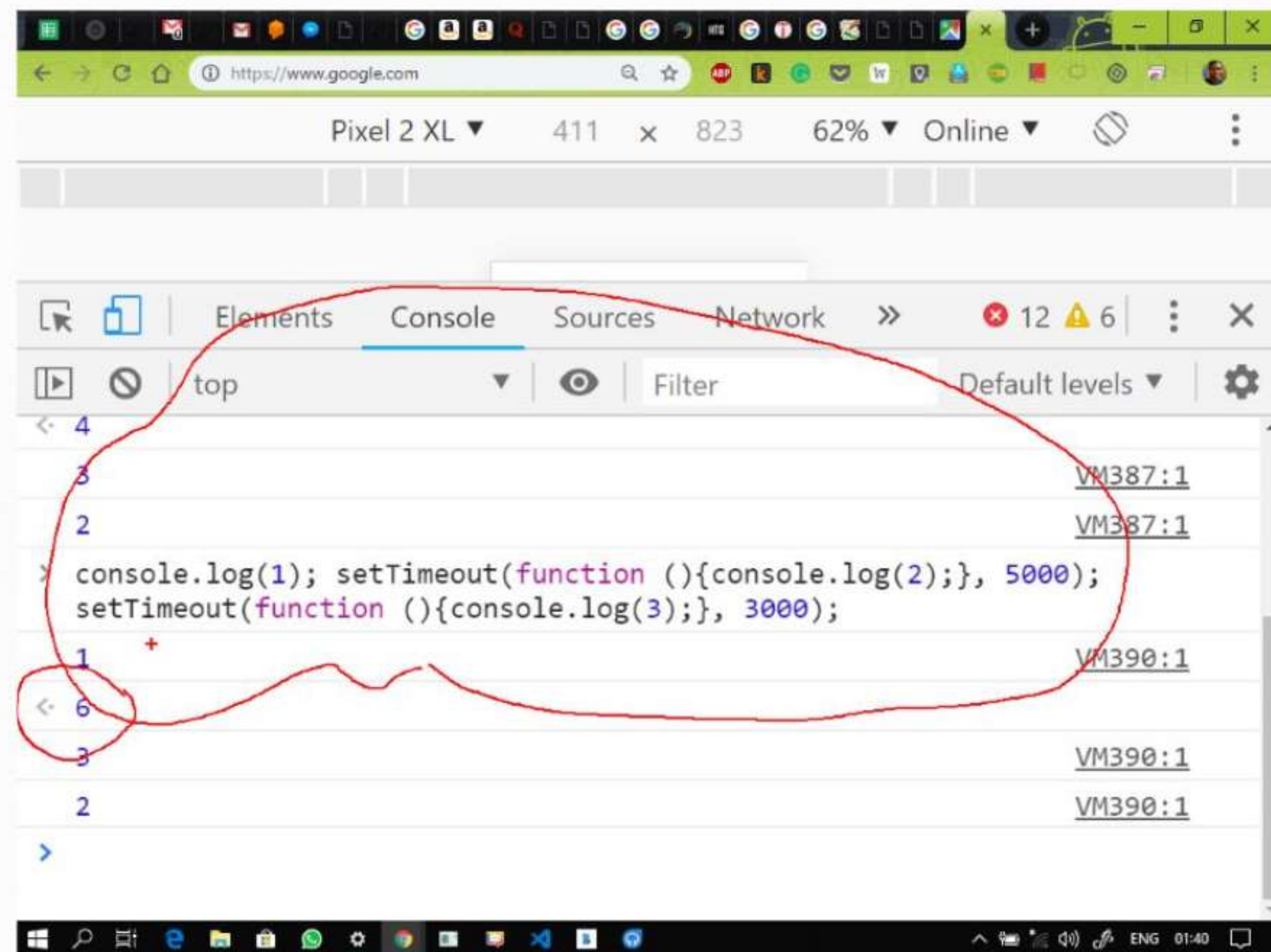
```
c.log(0);  
setTO (fail {c.log(1);}, 1000);  
setTO (fail {c.log(2);}, 800);
```

0	t=0
2	t=800 ms
<b>1</b>	t= <b>1000</b> ms



delay  $\rightarrow$  kuch instructions time  
Leti hai execution hone meh.

- $\Rightarrow$  requests  $\rightarrow$  
- $\Rightarrow$  recursion ~~~~~~~~~
- $\Rightarrow$  database
- $\Rightarrow$  file storage



requests

a = call-server (   
 console.log (a);   
 )

⇒ undefined

↓

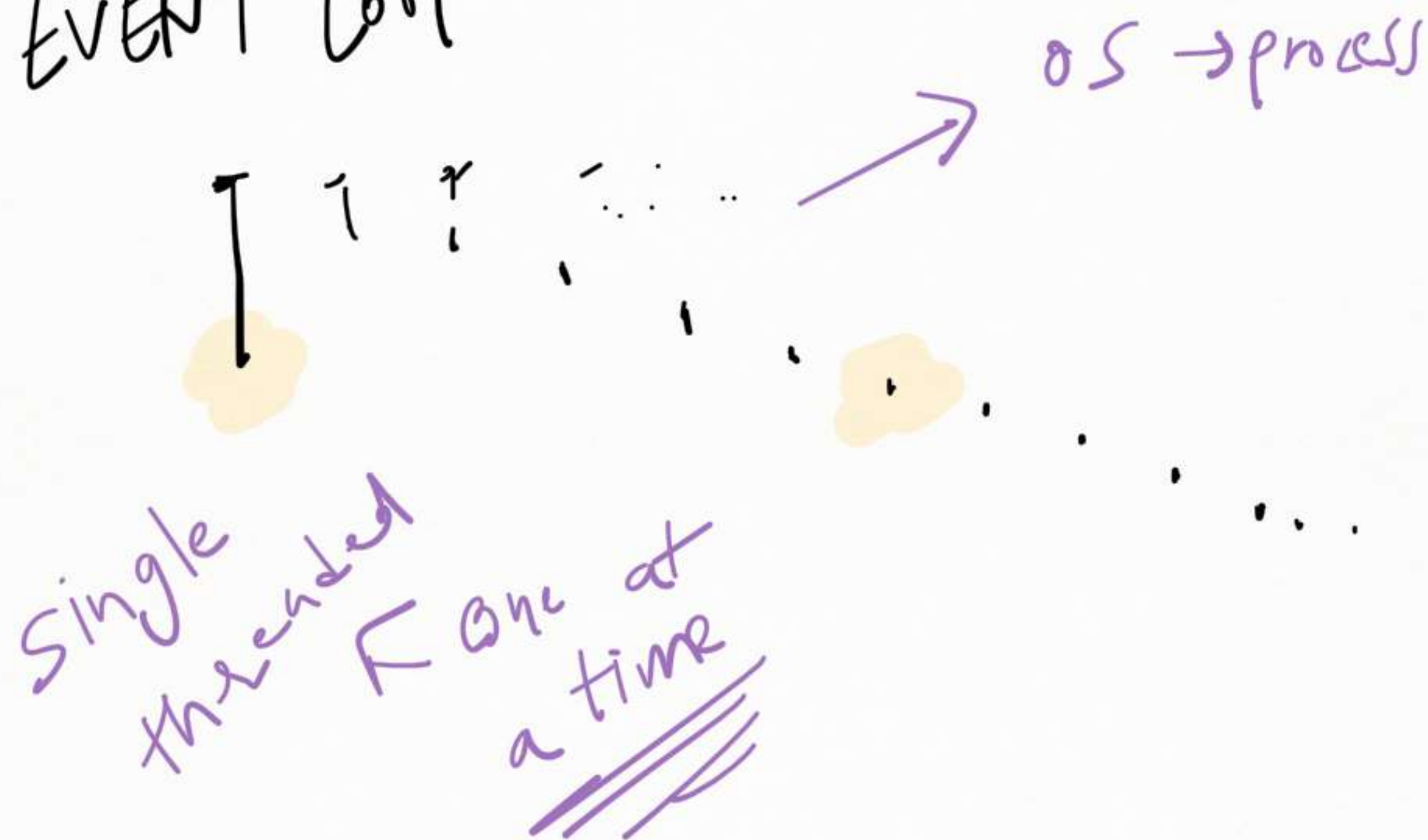
a = open ("1.txt", "r")  
print a

JS

↓ undefined



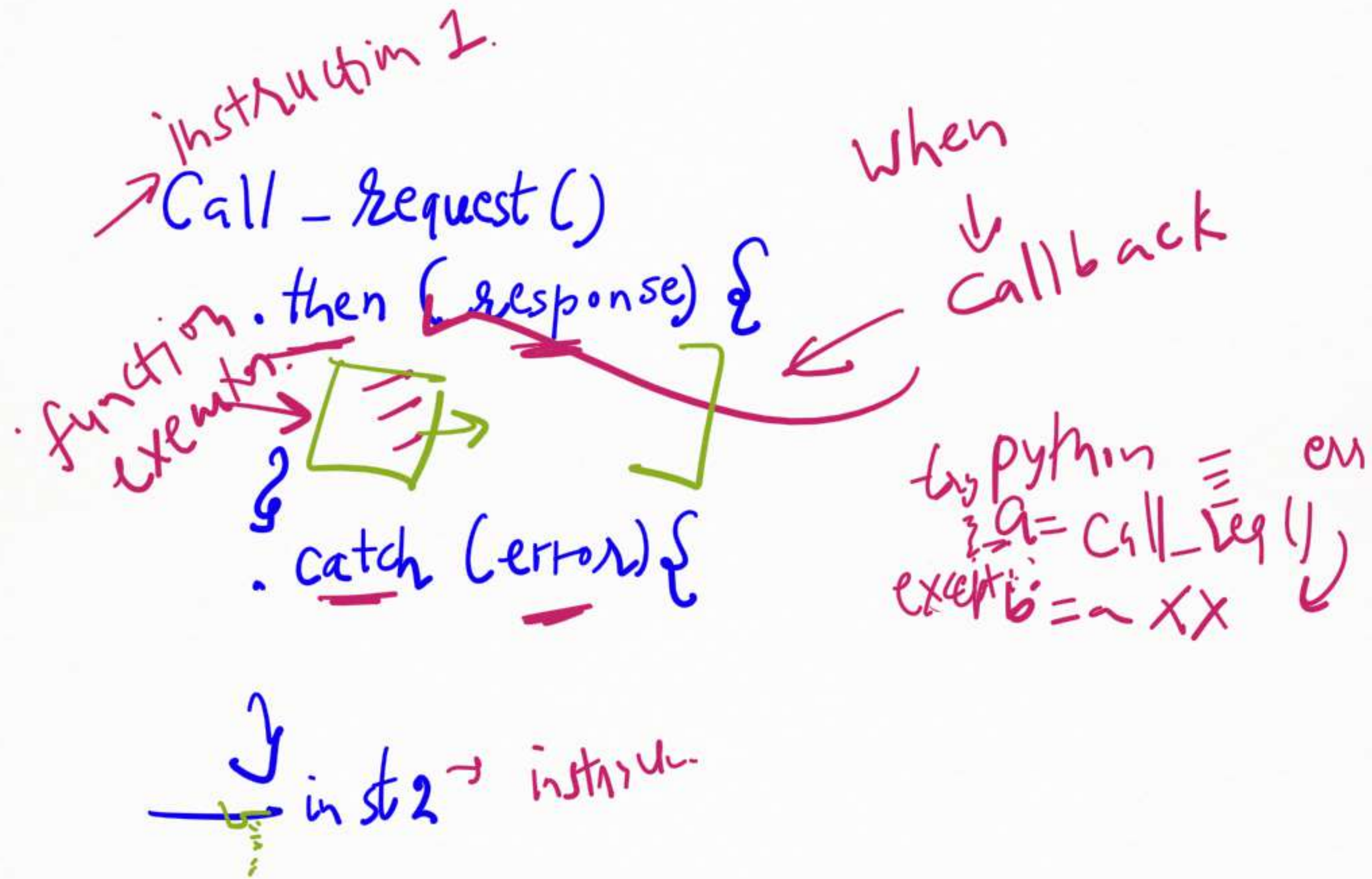
EVENT loop





~~Call~~ ~~function~~ ~~method~~ ~~handler~~ ~~procedures~~ ~~callback~~ ~~round~~

lagbhag  
same  
hai

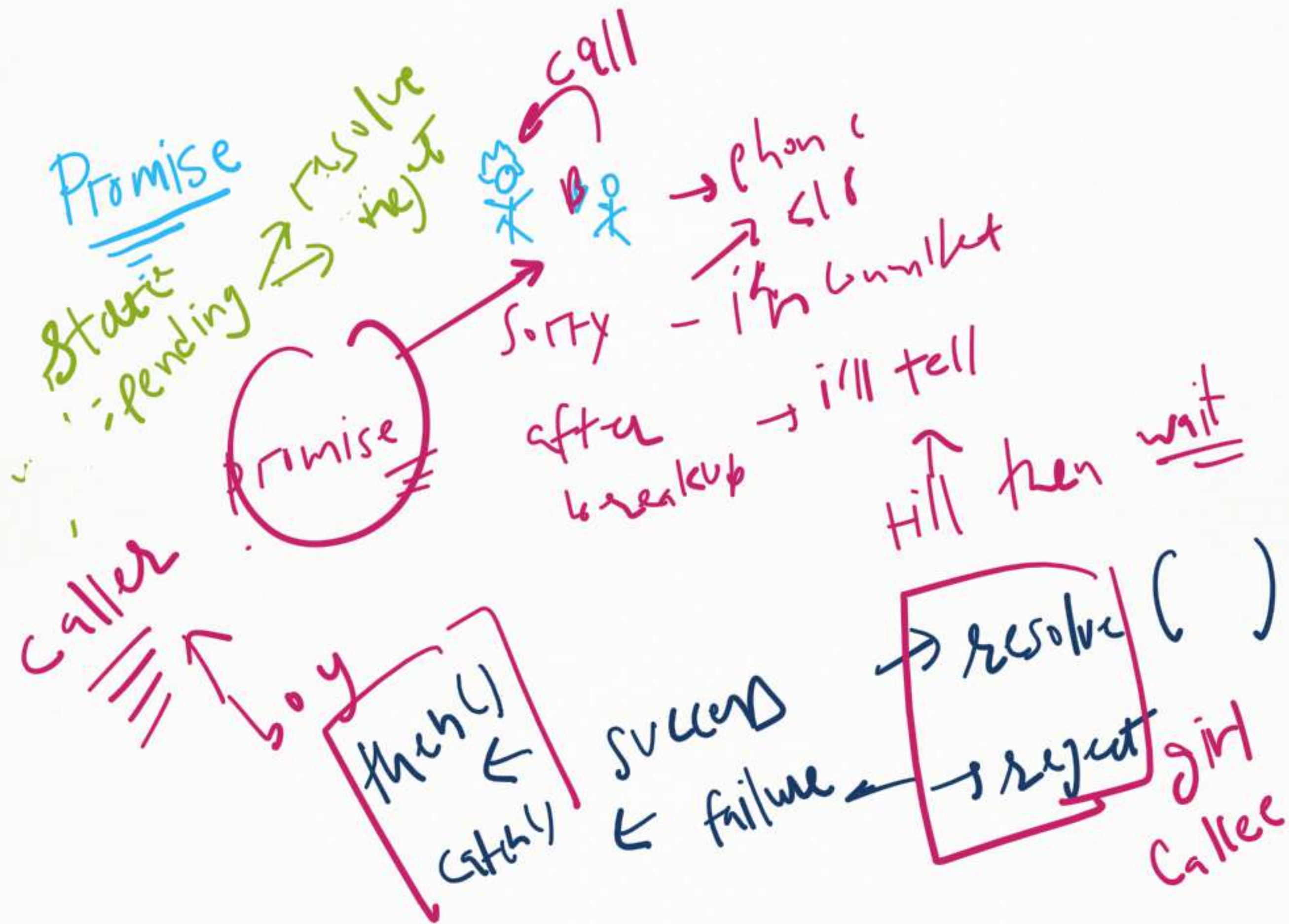


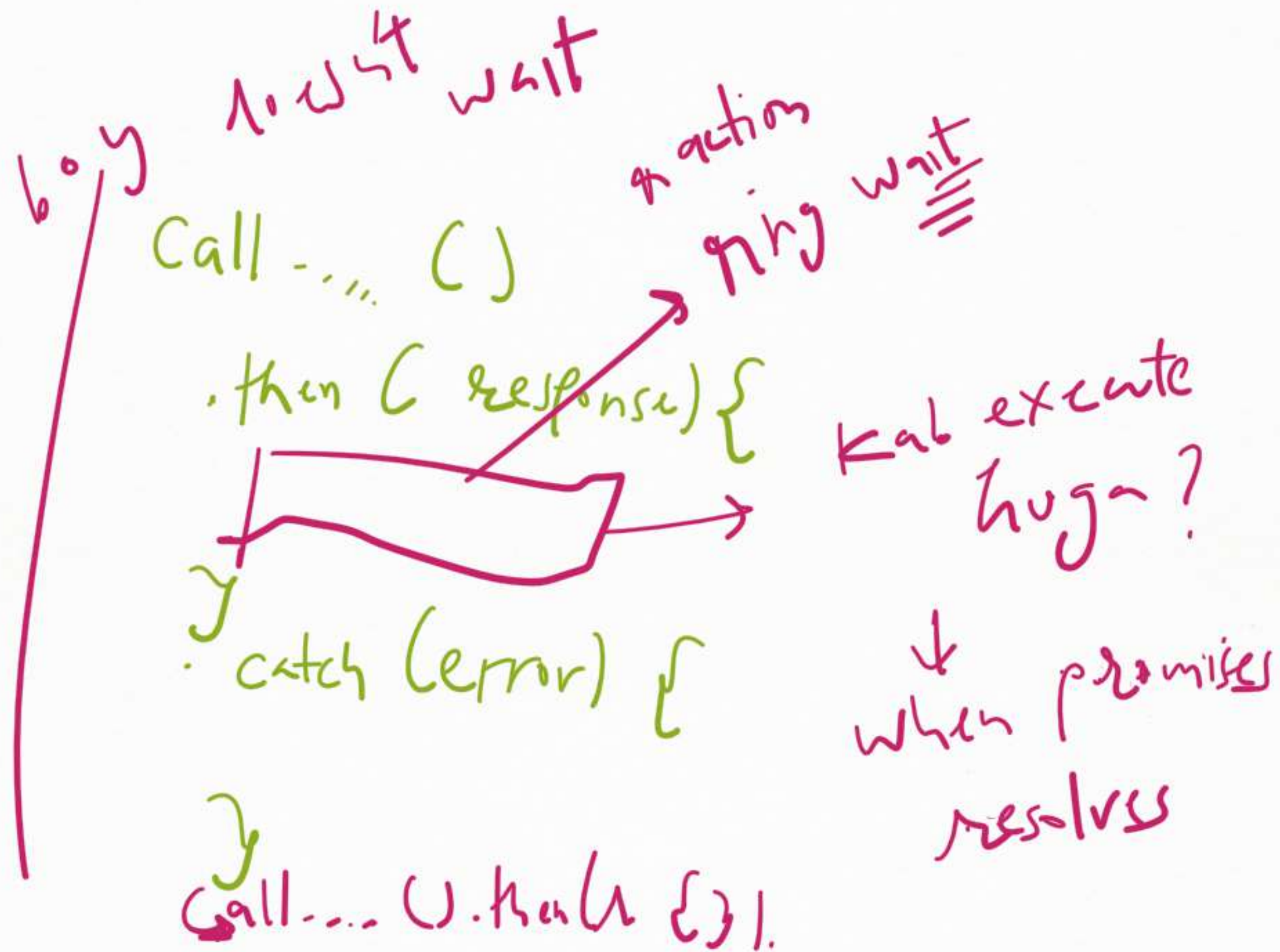


call 2  
then  
catch  
success  
failure / part of life  
part of communication  
don't call backs

```
graph TD; A[call 2] --> B[then]; B --> C[catch]; C --> D[success]; C --> E[failure / part of life]; F[don't call backs] --> A; G[part of communication] --> C;
```









girl

check → anam patra(  
if no:  
reject ("Kuch aur reason")  
if yes:  
resolve ("kuch uti") →

if no:  
reject ("friendzone/  
bro/family/  
no feelings / Duplicate found / Missing etc")  
↳ heart.



python

a = call\_api1()

dependency  
↓

b = call\_api2(a)

c = call\_api3(b)

print c

same  
code

JS

call\_api1()

.then(response) {

call\_api2(response)

.then(response2) {

call\_api3(response2)

.then(res) {

console.log(res);

callback  
→ call  
→ call  
→ call  
...

