Day29(promises , .then , .catch , async , await , DOM)

Thursday, 20 April 2023 7:37 PM

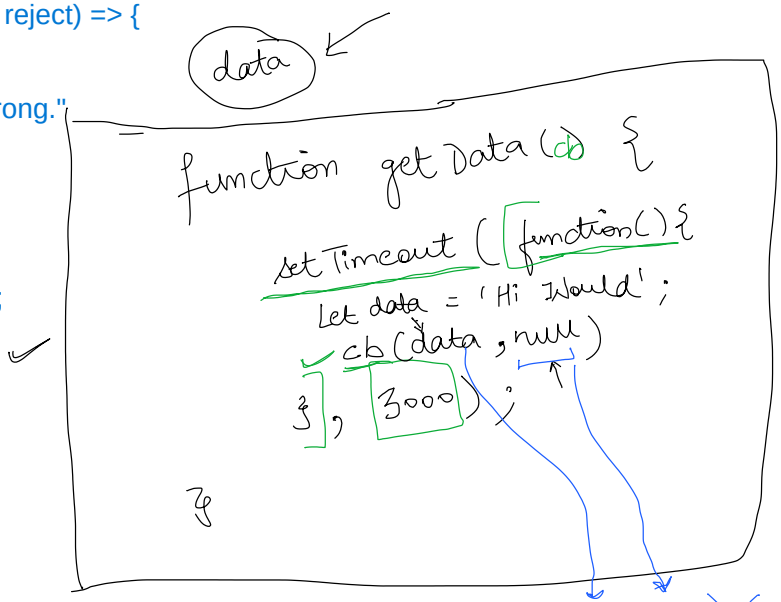
Promise

→ tackle problem of Cb hell.

```
javascriptCopy code
let p = new Promise((resolve, reject) => {

 setTimeout(() => {
let error = "Something went wrong."
 reject(error);
 }, 1000);
 });

 p.catch((error) => {
 console.error(error.message);
 "Something went wrong!"
 });
```



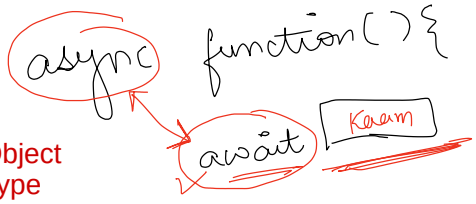
Promise accept a callback function which immediately invoked on constructor calling.

```
getData(function(data, err) {
if(err) { clg(err); }
else { clg(data); }
});
```

Dender proto of Promise Object is Object
Promise.prototype.__proto__ == Object.prototype
true

Async and await are used together. Await is placed before a promise, indicating that the program should wait for the promise to settle (resolve or reject) before proceeding. Matlab tab tak promise handle nahi hoga tab tk code the flow aage nahi badega.

Read the Samarth bhaiya JS Notes.



Dender proto of promise is Promise Object
promise.__proto__ == Promise.prototype
=> true

Awaits means => tab tk mera kaam khatam na ho jaye tab tk tum ruke raho.

await → Kuch intezaar

JS end

DOM

Promise OBJECT RETURNS :

Promise[[Prototype]]:

Promisecatch: ƒ catch()

constructor: ƒ Promise()

finally: ƒ finally()

then: ƒ then()

Symbol(Symbol.toStringTag): "Promise"

[[Prototype]]: Object => Points to Object prototype

[[PromiseState]]: "fulfilled"

[[PromiseResult]]: "Promise executed succuessfully"

Like the above output we get promise object

Promise object to consist of three things:

1. Promise prototype: from this we can access then, catch, finally, promise constructor etc...

2. Promise state => pending, fulfill, reject

3. Promise result => Any error message or successful code output.