Advanced Javascript

Day 10- Promises, async await

Promise

A promise is a commitment by someone to do or not do something

Imagine that you’re a top singer, and fans ask day and night for your upcoming song.

Why Promises in Js?

A promise is a commitment by someone to do or not do something

Promises have consequences.

1. They will be fullfilled.
2. They will not be fullfilled.
3. They will be in the process.

Promise by nature, takes time to fulfill or not fulfill.

Basically, we have to WAIT.

In JS programming, What kind of code might make you to wait?

Promises are useful when we have to wait.

Early ways of making a promise in Js.

JavaScript Promise Object

```
var promise = new Promise()
```

Resolve : When promise is fullfilled

Reject : When promise is rejected

```
var mypromise = new Promise(function (resolve,reject) {

resolve('success')
reject('error')

})
```

Resolve is the function that runs when promise is fullfiiled
Reject is the function that runs when promise is not fullfilled

Async Await

It’s just a new and better way of writing promises.

```
async function f() {
  return 1;
}
```

Async functions returns a promise, always.

```
//Resolve

async function hello() {
  return 'world';
}



//Reject
async function foo() {
  throw Error('bar');
}
```

So, we are getting promise using new async keyword.

So, we are getting promise using new async keyword.

How do we get the Promise result?

Earlier, we got succesfull promise using, .then()
and unsuccesfull promise using .catch()

await

```
async function f() {

  let promise = new Promise((resolve, reject) => {
    setTimeout(() => resolve("done!"), 1000)
  });

  let result = await promise; // wait until the promise resolves (*)

  alert(result); // "done!"
}

f();
```

await literally suspends the function execution until the promise settles, and then resumes it with the promise result

try / catch block

```
async function f() {

  try {
    let response = await fetch('http://no-such-url');
  } catch(err) {
    alert(err); // TypeError: failed to fetch
  }
}

f();
```

When we use async/await, we rarely need .then, because await handles the waiting for us. And we can use a regular try..catch instead of .catch. That’s usually (but not always) more convenient.