



Promises in JavaScript

What is a Promise in JavaScript?

A promise is a special JavaScript object that represents an eventual result of an asynchronous action.

Promise has 2 properties

1. **PromiseState** - Promise can be in one of 3 states pending, fulfilled and rejected
 - a. **Pending**- Neither rejected nor fulfilled, at this state value property of promise is undefined
 - b. **Fulfilled**- Action completed successfully, value property contains the real value
 - c. **Rejected**- Action failed, value property contains the reason of failure
2. **PromiseValue** - If everything goes well, this property will contain the real value of the promise.

Reject state of Promise



Promises in Reject State

```
1 const myPromise = new Promise((resolve, reject) => {  
2   reject("I am rejected")  
3 });
```

Output

```
< ▼ Promise {<rejected>: 'I am rejected'} ⓘ  
  ► [[Prototype]]: Promise  
    [[PromiseState]]: "rejected"  
    [[PromiseResult]]: "I am rejected"
```

Resolved state of Promise



Promises in Resolve State

```
1 const myPromise = new Promise((resolve, reject) => {  
2   resolve("I am resolved")  
3 });
```

Output

```
< ▼ Promise {<fulfilled>: 'I am resolved'} ⓘ  
  ► [[Prototype]]: Promise  
    [[PromiseState]]: "fulfilled"  
    [[PromiseResult]]: "I am resolved"
```

Pending state of Promise



Promises in Pending State

```
1 const myPromise = new Promise((resolve, reject) => {  
2  
3 });
```

Output

```
◀ ▼ Promise {<pending>} ⓘ  
  ▶ [[Prototype]]: Promise  
    [[PromiseState]]: "pending"  
    [[PromiseResult]]: undefined
```

Getting value of Promise

```
const myPromise = new Promise((resolve,reject) => {  
    resolve("I am resolved");  
});  
  
myPromise.then((value)=>{  
    console.log(value);  
})
```

Output


I am resolved

Function to calculate square of a number

```
function calculateSquare(number) {  
  const promise = new Promise((resolve, reject) => {  
    setTimeout(()=>{  
      const result = number*number;  
      resolve(result)  
    },1000);  
  });  
  return promise;  
}
```

We are avoiding callback hell situation while calculating squares for 3 numbers by making use of promises

This is called promise chaining
which is used to avoid callback hell



```
calculateSquare(1)
  .then(valueOne => {
    console.log(valueOne);
    return calculateSquare(2)
  })
  .then(valueTwo => {
    console.log(valueTwo);
    return calculateSquare(3)
  })
  .then(valueThree =>{
    console.log(valueThree);
  })
```

Output will be 1, 4 and 9 after 1 second interval for each

At present `async/await` is used
heavily

It is just a wrapper to restyle
code and make promises easier
to read and use.

Stay tuned for `async/await`...



Follow Rohit Sharma for more
such updates