JavaScript Callbacks

- A callback is a function passed as an argument to another function
- This technique allows a function to call another function
- A callback function can run after another function has finished

myCalculator(5, 5, myDisplayer);

// Call removeNeg with a callback
const posNumbers = removeNeg(myNumbers, (x) => x >= 0);
In the example above, (x) => x >= 0 is a callback function.

JavaScript Asynchronous

- Functions running in parallel with other functions are called asynchronous
- With asynchronous programming, JavaScript programs can start long-running tasks, and continue running other tasks in parallel.
- Asynchronous programmes are difficult to write and difficult to debug.
- Because of this, most modern asynchronous JavaScript methods don't use callbacks. Instead, in JavaScript, asynchronous programming is solved using Promises instead.

Asynchronous Examples :

- setTimeout(myFunction, 3000);

```
function myFunction() {
  document.getElementById("demo").innerHTML = "I love You !!";
}
```

- setInterval(myFunction, 1000);

```
function myFunction() {
  let d = new Date();
  document.getElementById("demo").innerHTML=
  d.getHours() + ":" +
  d.getMinutes() + ":" +
  d.getSeconds();
}
```

JavaScript Promises

- "Producing code" is code that can take some time
- "Consuming code" is code that must wait for the result
- A Promise is an Object that links Producing code and Consuming code

Promise Object Properties

A JavaScript Promise object can be:

- Pending
- Fulfilled
- Rejected

The Promise object supports two properties: state and result.

- While a Promise object is "pending" (working), the result is undefined.
- When a Promise object is "fulfilled", the result is a value.
- When a Promise object is "rejected", the result is an error object.

Promise How To

Here is how to use a Promise:

```
myPromise.then(
  function(value) { /* code if successful */ },
  function(error) { /* code if some error */ }
);
```

JavaScript Async / Await

- "async and await make promises easier to write"
- async makes a function return a Promise
- await makes a function wait for a Promise

Async Syntax

The keyword async before a function makes the function return a promise: