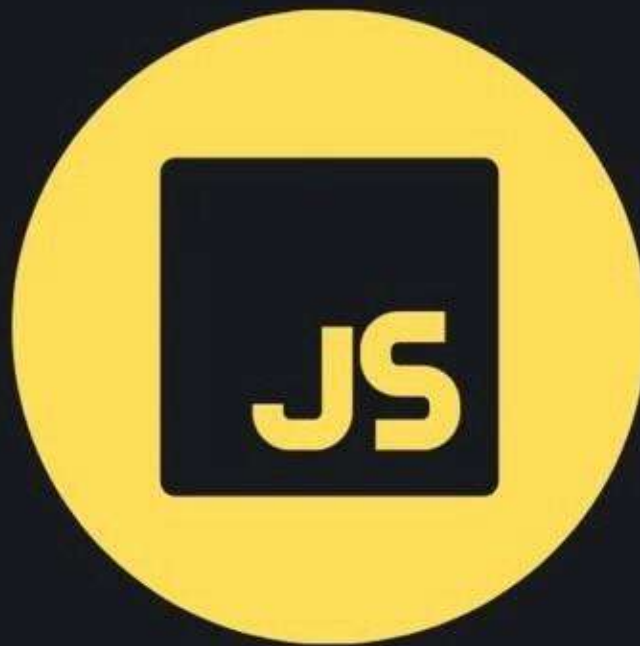


Swipe



Must Know JS Concepts



Closures

Closures allow functions to access variables from their **parent scope** even after the parent function has finished executing.

```
function outer() {  
  let outerVar = "I'm outer!";  
  function inner() {  
    console.log(outerVar);  
  }  
  return inner;  
}  
  
let innerFn = outer();  
innerFn(); // Output: I'm outer!
```

Callback Functions

Callback functions are functions **passed as arguments** to other functions and executed later, typically after an asynchronous operation completes.

```
// Function with callback
function greet(callback) {
  console.log("Hello!");
  callback(); // Invoke the callback function
}

// Callback function
function sayGoodbye() {
  console.log("Goodbye!");
}

// Call function with callback
greet(sayGoodbye);
```

Promises

A JavaScript object representing the eventual completion or failure of an **asynchronous operation**.

Promises simplify asynchronous code and help avoid callback hell

```
const myPromise = new Promise((resolve, reject) => {  
  setTimeout(() => {  
    resolve('Success!'); // Resolve the Promise after a delay  
  }, 1000);  
});  
  
myPromise.then(result => console.log(result)); // Output: Success!
```


Async/Await

A modern JavaScript feature that allows you to write asynchronous code in a **synchronous-like manner**, making it easier to understand and maintain.

```
async function fetchData() {  
  const data = await fetch('https://api.example.com/data');  
  console.log(data);  
}
```

Hoisting

A JavaScript behavior where variable and function declarations are **moved to the top of their containing scope** during compilation, allowing them to be used before they are declared.

```
console.log(myVar); // Output: undefined  
var myVar = 10;
```

```
// This code is interpreted as:
```

```
var myVar;  
console.log(myVar); // Output: undefined  
myVar = 10;
```

Try/Catch

This construct in JavaScript is used for **error handling**, allowing you to manage exceptions gracefully.

```
try {  
    // Code that might throw an error  
    throw new Error('Something went wrong');  
} catch (error) {  
    console.error(error);  
}
```