

Hoisting in JavaScript



What is Hoisting in JS?

In JavaScript, hoisting is a behavior where variable and function declarations are moved to the top of their containing scope during the compilation phase. This means that you can use a variable or a function before it's actually declared in your code. However, it's important to understand that only the declarations are hoisted, not the initializations or assignments.

Hoisting with Function Declarations:

```
sayHello(); // Output: "Hello, world!"  
function sayHello() {  
    console.log("Hello, world!");  
}
```

```
function sayHello() { // Declaration is hoisted  
    console.log("Hello, world!");  
}  
sayHello(); // Output: "Hello, world!"
```

Hoisting with Variable Declarations:

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

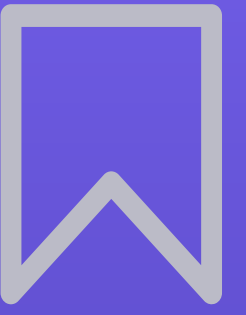
```
var x; // Declaration is hoisted  
console.log(x); // Output: undefined  
x = 10; // Initialization  
console.log(x); // Output: 10
```


Variable Hoisting vs. Function Hoisting:

```
console.log(y); // Output: undefined  
var y = 5;
```

```
sayHi(); // Output: "Hi!"  
function sayHi() {  
    console.log("Hi!");  
}
```

It's essential to be aware of hoisting in JavaScript to avoid unexpected behavior in your code. To write clean and maintainable code, it's recommended to declare your variables and functions at the top of their respective scopes and avoid relying on hoisting to access them before declaration.



[mujtaba.codes_](#)



linkedin.com/in/mujtaba-shafique/

**LIKE AND SAVE IT FOR
LATER**