In this video, we explore the concept of scope in JavaScript and learn about two types of scope: local scope and block scope.

* Local scope refers to variables that are only accessible within the function where they are declared.
* Block scope, introduced in ES6, states that variables declared within a block of code (enclosed in curly braces) are only accessible within that block.
* Variables declared with the var keyword have function scope, meaning they are accessible throughout the entire function.
* The recommended way to declare variables in ES6 is to use the let or const keywords, which have block scope.
* The main difference between var, let, and const is that let and const are more strict: they cannot be used before they are declared, and const variables cannot be reassigned.
* A pro tip is to use let if the value of a variable might change in the future, and use const if the value will never change.

Var let const

Lenient strict

Here is a list of resources that may be helpful as you continue your learning journey.

[MDN Functions Guide](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Functions)

[MDN Glossary: Recursion](https://developer.mozilla.org/en-US/docs/Glossary/Recursion)

[MDN Glossary: Scope](https://developer.mozilla.org/en-US/docs/Glossary/Scope)

[Functional Programming in JavaScript](https://www.toptal.com/javascript/functional-programming-javascript)

[MDN: First-class functions](https://developer.mozilla.org/en-US/docs/Glossary/First-class_Function)

Object oriented programming

Certainly! In programming, there are different ways or styles to write code, and these are called programming paradigms. One popular paradigm is called Object-Oriented Programming (OOP). OOP is all about organizing our programs using objects, which group related data and functionality together.

In contrast, another paradigm called Functional Programming separates data from the functions that work on that data.

To understand this better, let's take an example. Suppose we want to write code to calculate the total cost of buying a pair of shoes, including tax.

In functional programming, we would have separate variables for the shoes price, tax amount, and total price. We would also have a function that calculates the total price by multiplying the shoes price and tax amount.

In OOP, we would create an object called "purchase1" and store all the data related to that purchase, like the shoes price, tax amount, and a method (a function inside an object) called "totalPrice" that calculates the total price. We can then access the totalPrice method on the purchase1 object to get the result.

The advantage of using OOP is that we can create multiple objects, like "purchase2", and access the totalPrice method on each object. We can also make the methods in different objects identical by using the "this" keyword, which refers to the current object. This allows us to reuse the same method code for different objects.

To make the code even more efficient, we can create a template for objects using something called a class. This allows us to create new objects based on the template, reducing code repetition and making our programs more efficient.

I hope this explanation helps! Let me know if you have any further questions.

03:03 AM

Summary of objected oriented programming

In this video, we learn about the programming paradigms, specifically the object-oriented programming (OOP) paradigm. OOP focuses on organizing programs using objects to group related data and functionality. We compare it to the functional programming approach, where data and functions are kept separate.

To illustrate the difference, we use an example of calculating the total cost of buying a pair of shoes. In functional programming, we separate the data and the function that performs the calculation. In OOP, we create an object that contains the data, functions, and output statements. We can access the data and methods of the object using dot notation.

We also explore the use of the "this" keyword, which refers to the current object. By using "this", we can make methods identical across different objects, reducing code duplication. However, this approach still requires us to repeat the method for each new object, which can be inefficient.

To address this, we can create templates for objects using something called a class. This allows us to efficiently build new objects with shared properties and methods.

03:03 AM