

, המלל מתייחס לשורה 9 והלאה2 דוגמאות

In the provided code snippet, the reason we can change the `name` property of the `obj` object to "Uri" is because the `obj` object is defined using `const`, which means the reference to the object itself cannot be reassigned, but the properties of the object can still be modified.

However, in the commented-out code `// name: "Daniela"`, it attempts to modify the object literal itself, which is not allowed with `const`. This will result in a syntax error because the object literal is a constant value, and its properties cannot be changed or reassigned once it is defined.

Here's a breakdown of what's happening:

1. The `obj` object is declared as a constant using `const obj = { name: "ori", age: 45 }`. This means that `obj` cannot be reassigned to a new object reference, but its properties (`name` and `age`) can be modified.

2. The first `console.log(obj)` outputs `{ age: 45, name: 'ori' }`, which is the initial state of the `obj` object.

3. The lines `obj.name = "Uri"` and `obj.age = 50` modify the `name` and `age` properties of the `obj` object, respectively. This is allowed because we're not reassigning the entire object, but rather changing its properties.

4. The second `console.log(obj)` outputs `{ age: 50, name: 'Uri' }`, reflecting the updated values of `name` and `age`.

5. The commented-out code `// name: "Daniela"` attempts to modify the object literal directly, which is not allowed with `const`. If you were to uncomment this line, it would result in a syntax error because you cannot reassign or modify a constant object literal.

In summary, the difference lies in the fact that you can modify the properties of an object declared with `const`, but you cannot reassign or modify the object literal itself. The commented-out code `// name: "Daniela"` attempts to modify the object literal, which is not allowed with `const`.