Save for later

10 Destructuring Tips

Master Object & Array Manipulation

JS

Alamin
@CodePapa360

1. Unpack Object Properties

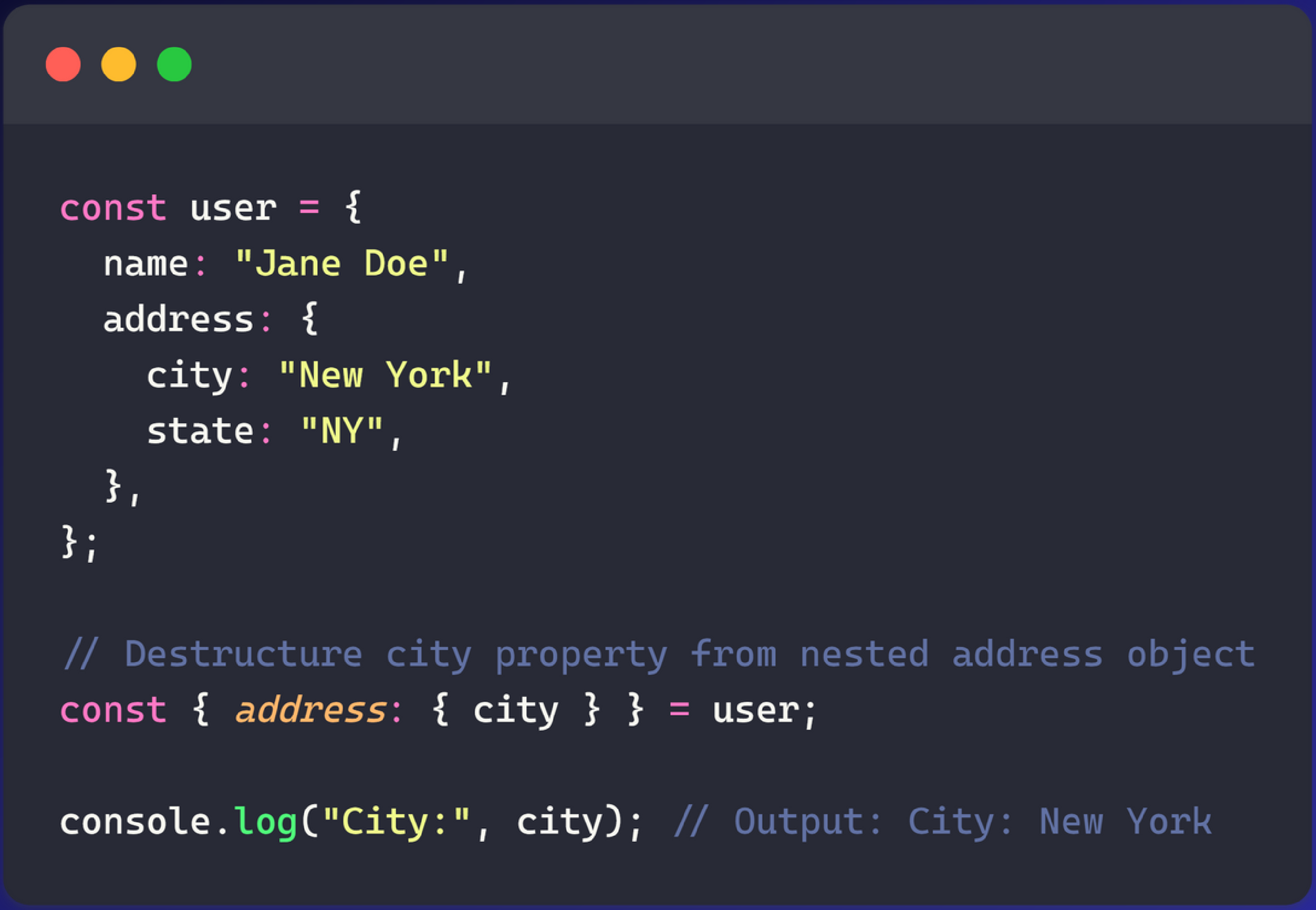
Extract specific properties from objects into variables

```javascript
const person = {
  name: "John Doe",
  age: 30,
  occupation: "Software Developer",
};

// Destructure name and age properties
const { name, age } = person;

console.log("Name:", name); // Output: Name: John Doe
console.log("Age:", age);    // Output: Age: 30
```

Alamin
@CodePapa360

2. Destructure Nested Objects

Access properties within nested objects efficiently



```javascript
const user = {
  name: "Jane Doe",
  address: {
    city: "New York",
    state: "NY",
  },
};

// Destructure city property from nested address object
const { address: { city } } = user;

console.log("City:", city); // Output: City: New York
```

Alamin
@CodePapa360

3. Assign Default Values

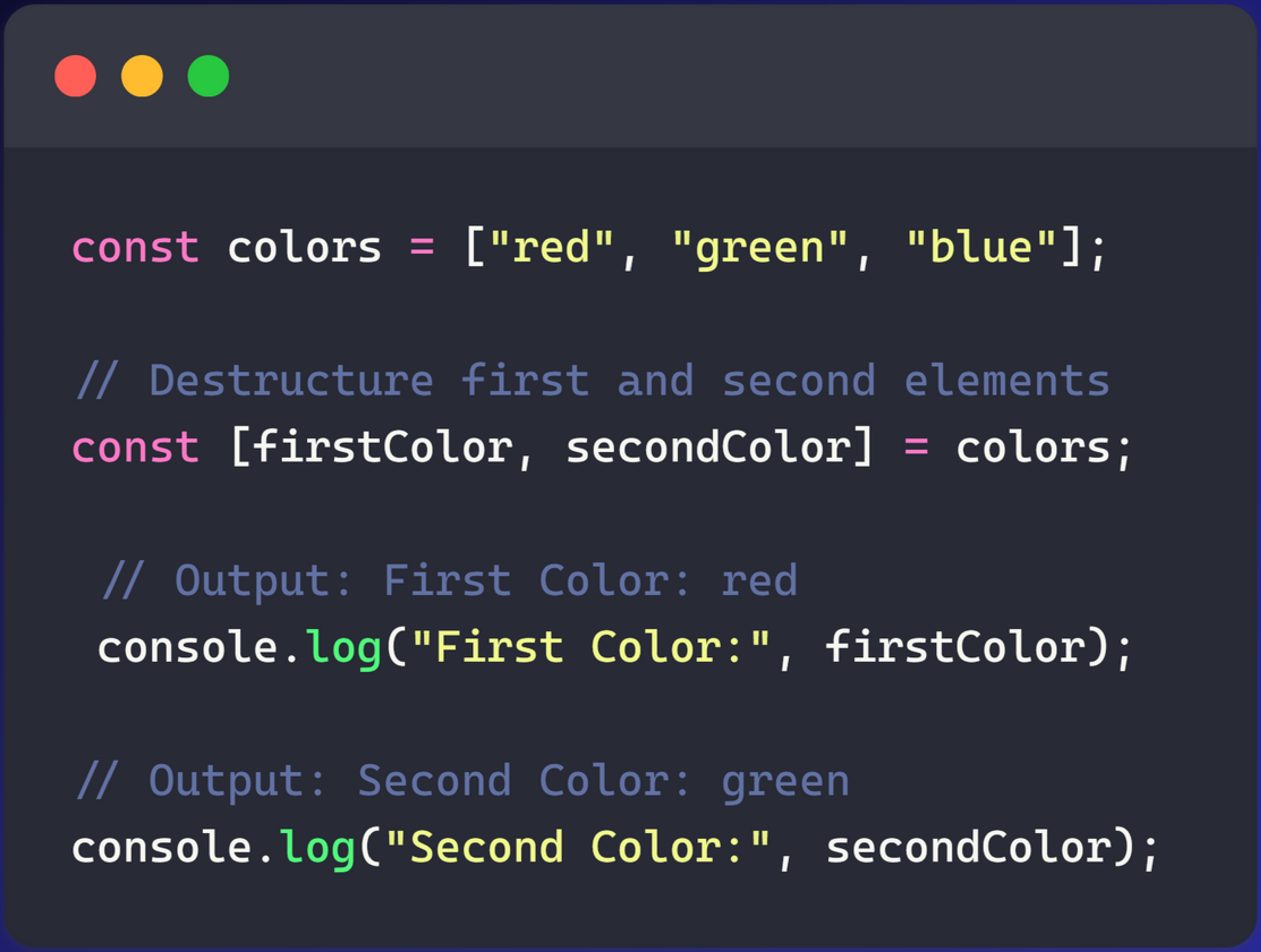
Set defaults for missing properties to avoid undefined errors

```javascript
const profile = {
  name: "Alice",
};

// Destructure age with a default value of 25
const { age = 25 } = profile;

// Output: Age: 25 (assuming no 'age' property in profile)
console.log("Age:", age);
```

Alamin
@CodePapa360

4. Destructure Arrays

Extract elements from arrays into individual variables

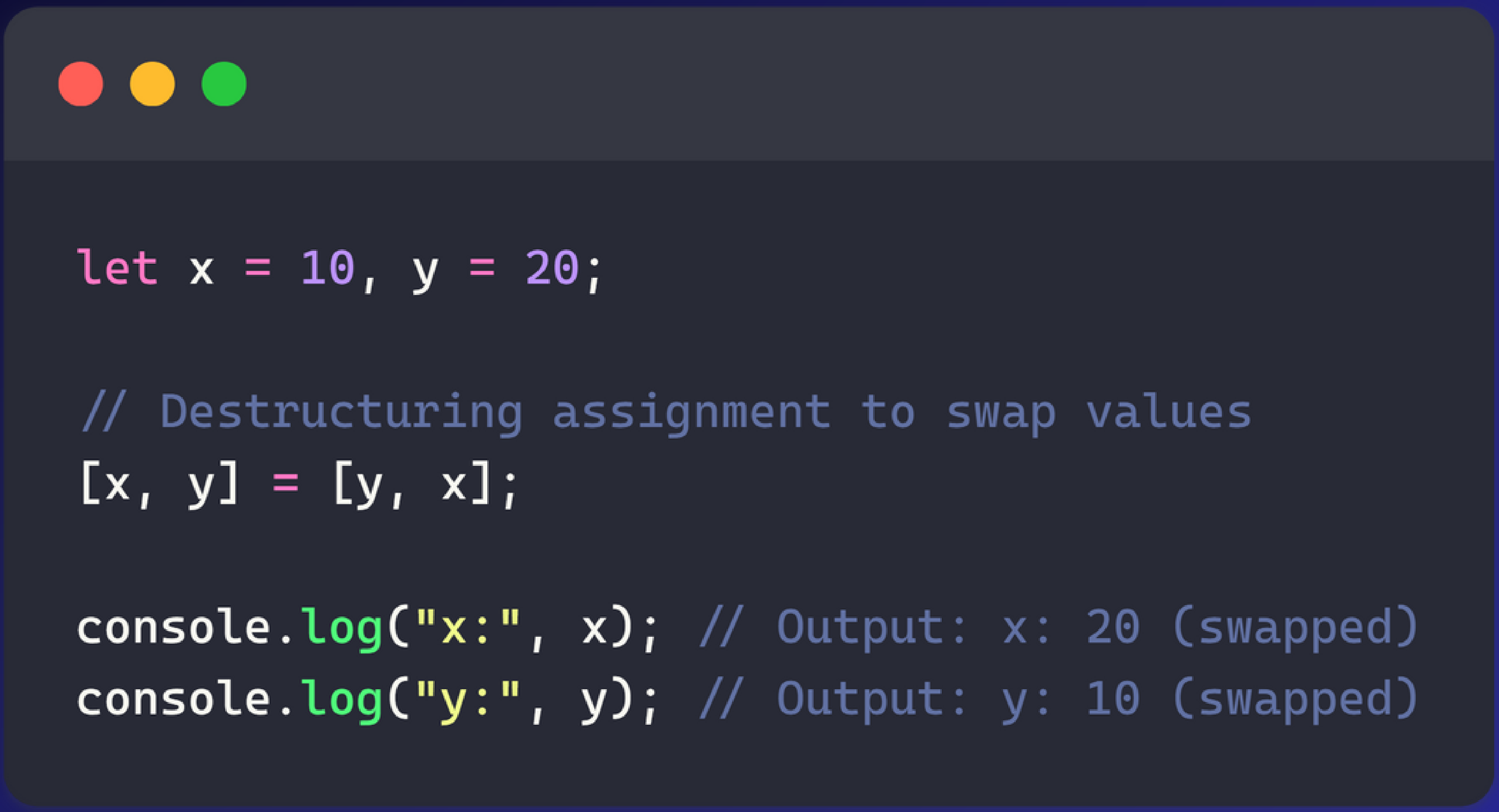


```javascript
const colors = ["red", "green", "blue"];

// Destructure first and second elements
const [firstColor, secondColor] = colors;

 // Output: First Color: red
 console.log("First Color:", firstColor);

// Output: Second Color: green
console.log("Second Color:", secondColor);
```

Alamin
@CodePapa360

5. Swap Variables with Destructuring

Achieve variable swapping in a concise and readable way



```javascript
let x = 10, y = 20;

// Destructuring assignment to swap values
[x, y] = [y, x];

console.log("x:", x); // Output: x: 20 (swapped)
console.log("y:", y); // Output: y: 10 (swapped)
```

Alamin
@CodePapa360

6. Use Rest Operator (...)

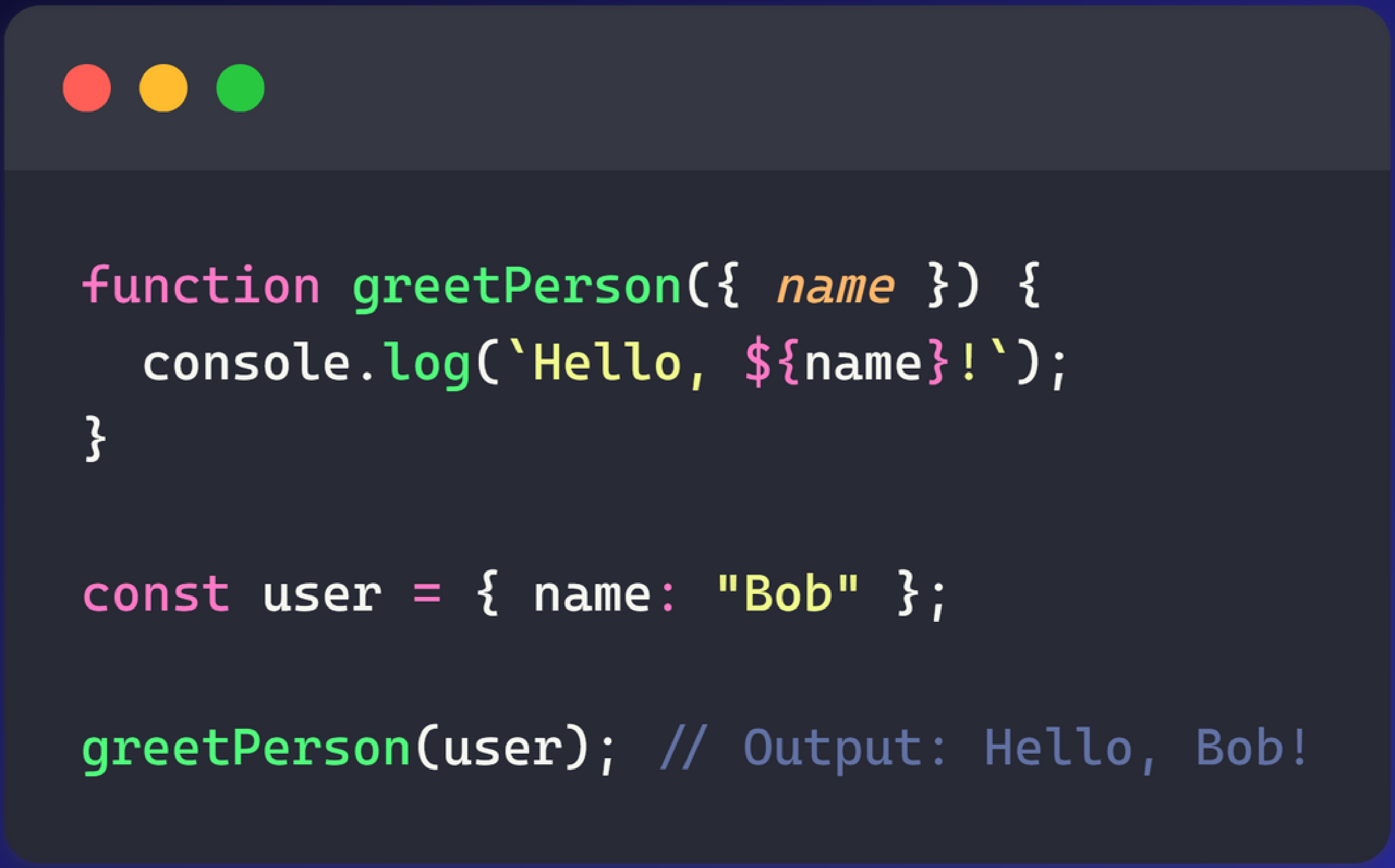
Collect remaining elements of an array into a new variable

```javascript
const numbers = [1, 2, 3, 4, 5];

// Destructure first element and rest into 'otherNumbers'
const [firstNumber, ...otherNumbers] = numbers;

 // Output: First Number: 1
 console.log("First Number:", firstNumber);

 // Output: Other Numbers: [2, 3, 4, 5]
 console.log("Other Numbers:", otherNumbers);
```

Alamin
@CodePapa360

7. Destructure Function Arguments

Extract function arguments into variables for cleaner code

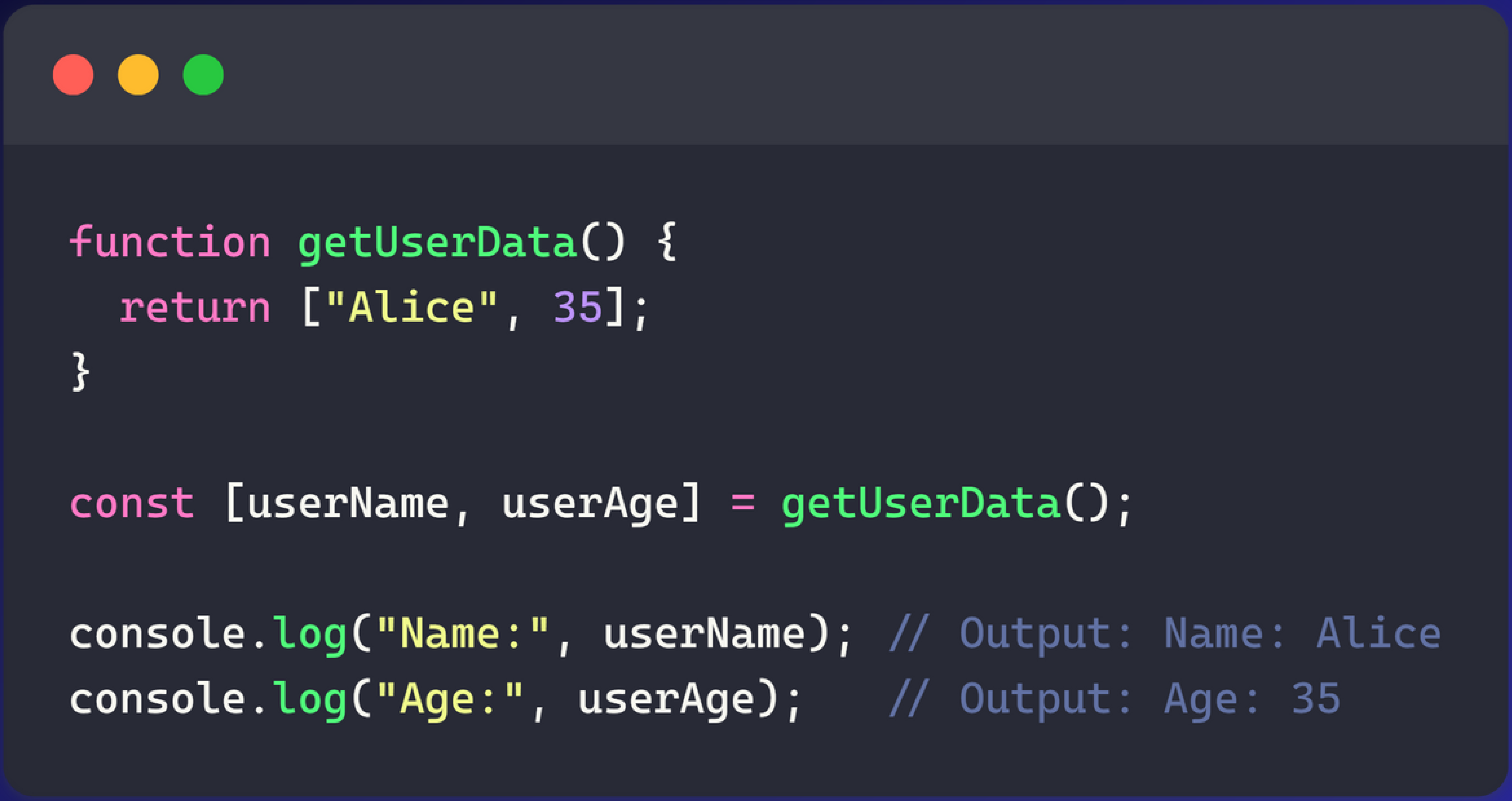


```
function greetPerson({ name }) {
  console.log(`Hello, ${name}!`);
}

const user = { name: "Bob" };

greetPerson(user); // Output: Hello, Bob!
```

Alamin
@CodePapa360

8. Destructure Return Values

Unpack values returned from functions into separate variables



```javascript
function getUserData() {
  return ["Alice", 35];
}

const [userName, userAge] = getUserData();

console.log("Name:", userName); // Output: Name: Alice
console.log("Age:", userAge);   // Output: Age: 35
```

Alamin
@CodePapa360

9. Destructure in Loops

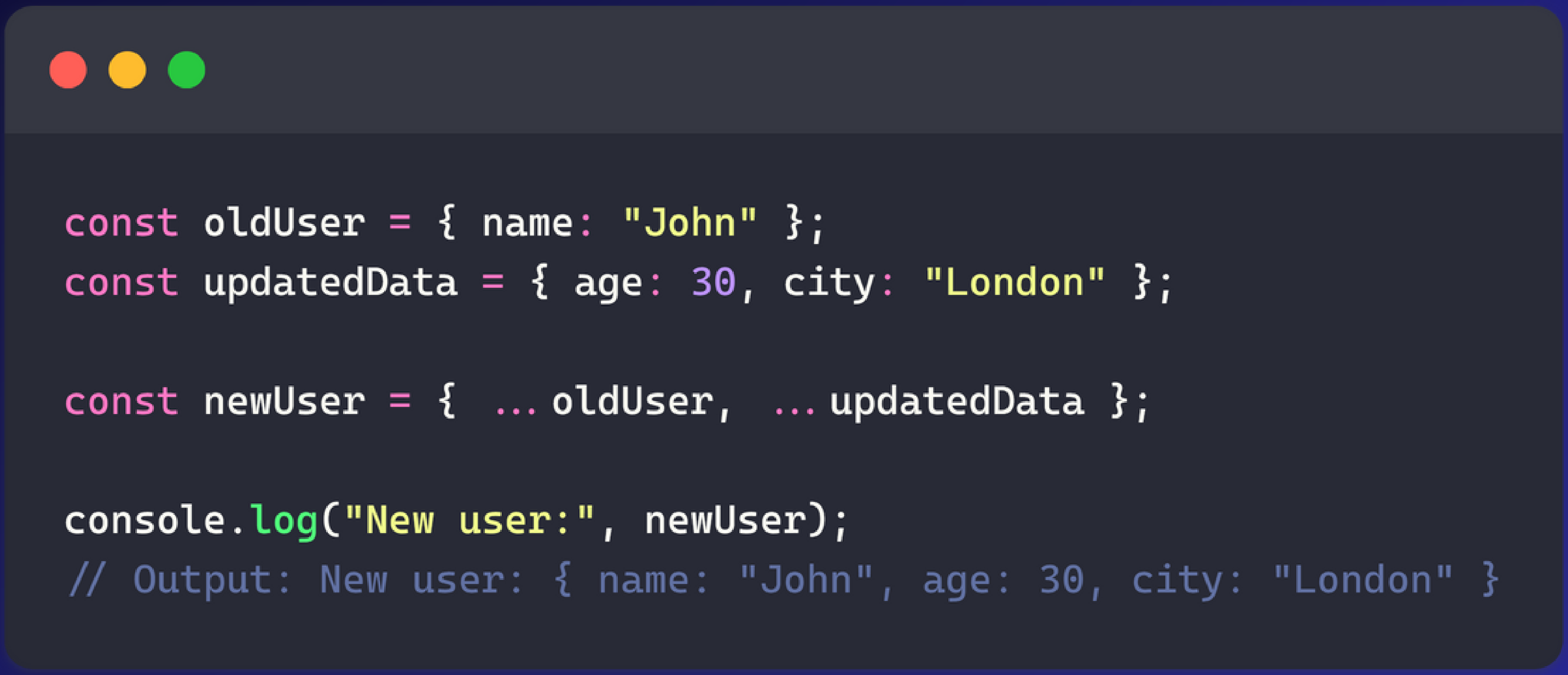
Improve readability and efficiency when iterating over objects or arrays

```javascript
const employees = [
  { name: "John", age: 30 },
  { name: "Jane", age: 28 },
];

for ({ name } of employees) {
  console.log(name); // Outputs: John, Jane
}
```

Alamin
@CodePapa360

10. Combine with Spread Operator

Combine destructuring with spread operator for complex object manipulation



```
const oldUser = { name: "John" };
const updatedData = { age: 30, city: "London" };

const newUser = { ...oldUser, ...updatedData };

console.log("New user:", newUser);
// Output: New user: { name: "John", age: 30, city: "London" }
```

Alamin
@CodePapa360

Did you find it useful?

Alamin | Front-end Developer

@CodePapa360

Follow for more

Like | Comment | Repost