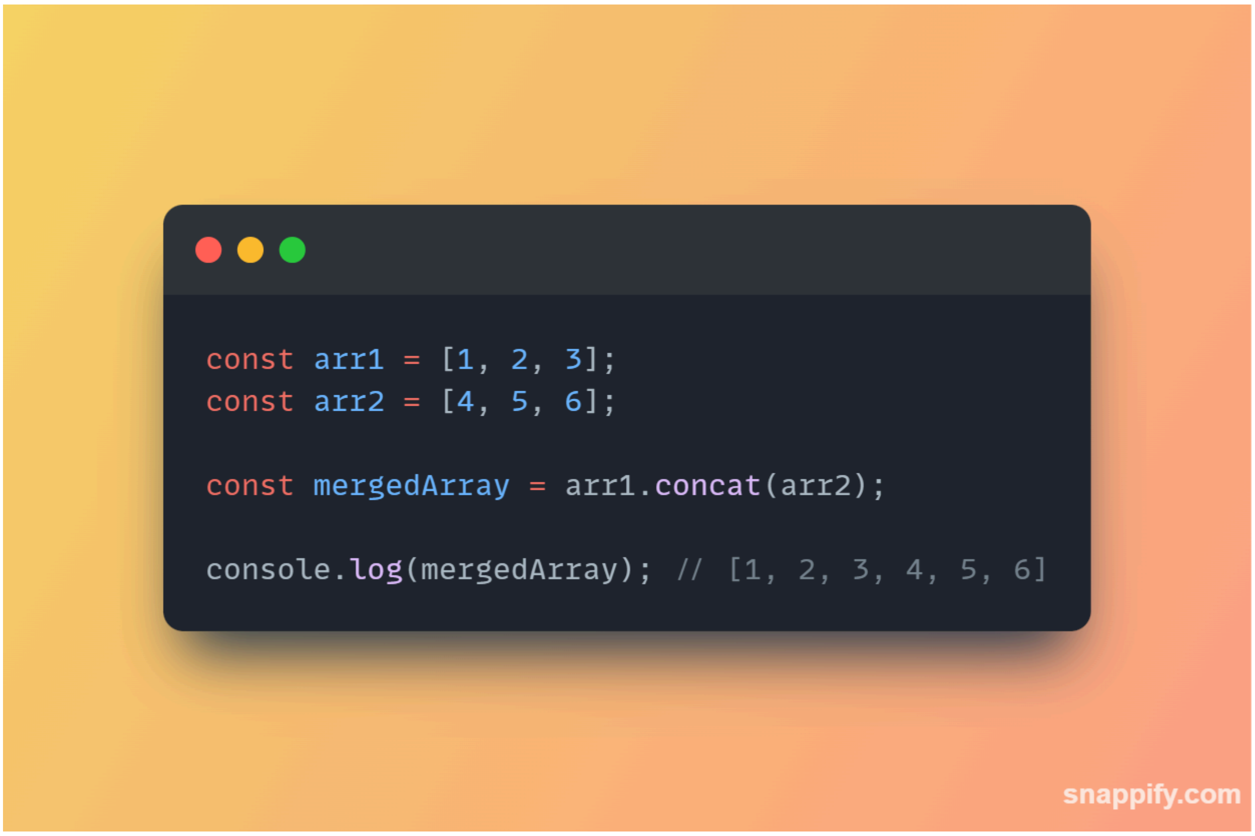
1️⃣concat() - Merge two arrays

The concat() method is used to join two or more arrays without modifying the original arrays.



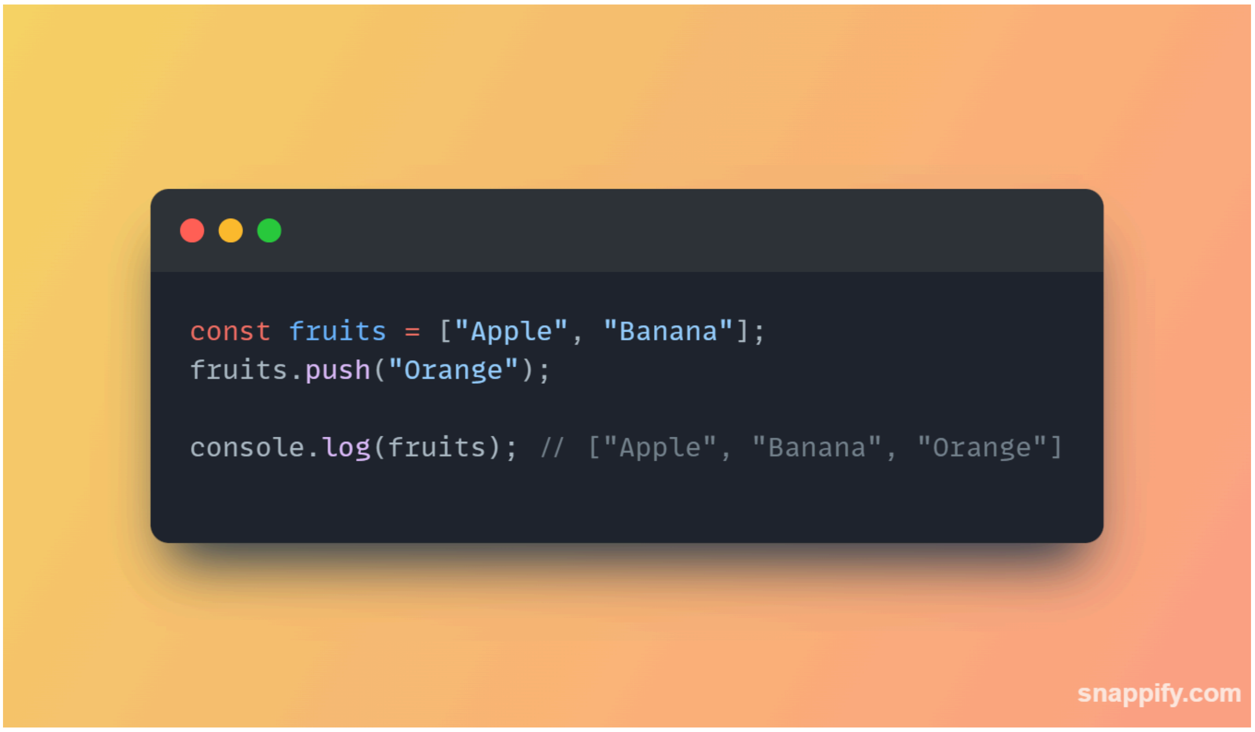
```
const arr1 = [1, 2, 3];
const arr2 = [4, 5, 6];

const mergedArray = arr1.concat(arr2);

console.log(mergedArray); // [1, 2, 3, 4, 5, 6]
```

2️⃣ push() - Add an element at the end

The push() method adds one or more elements to the **end** of an array and returns the new length.

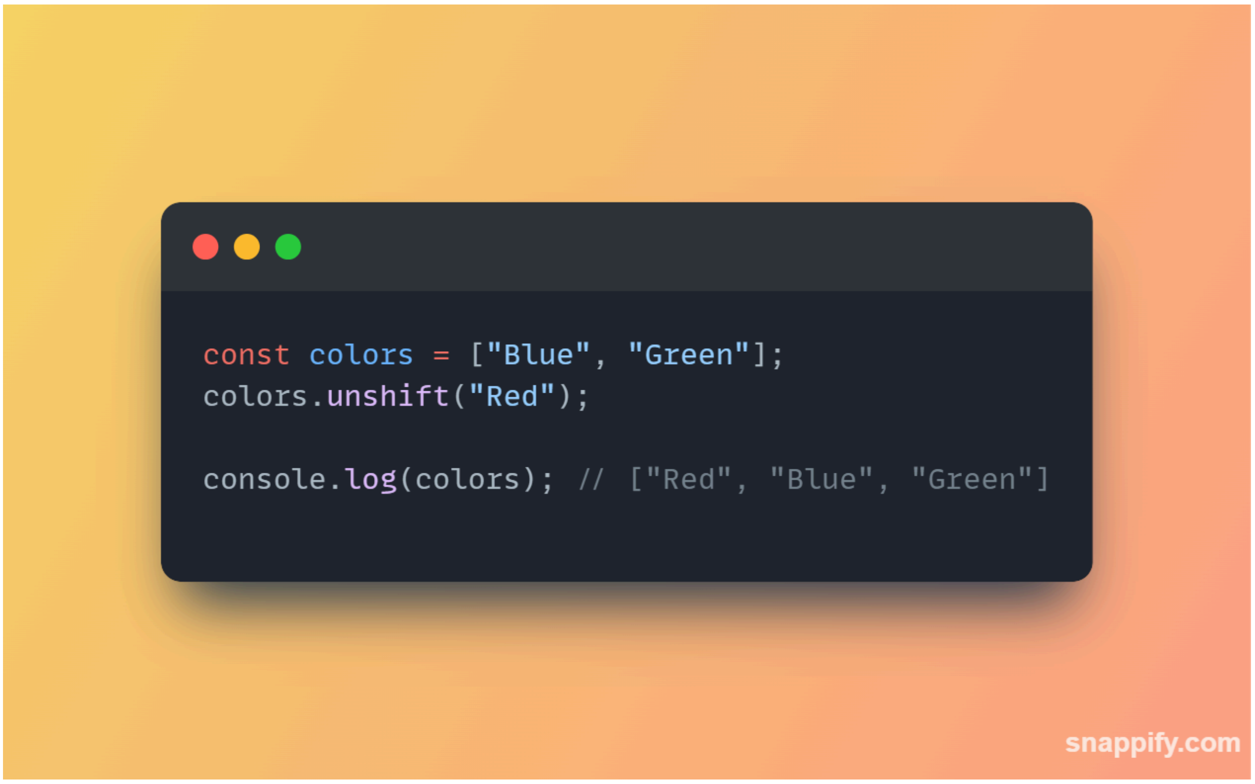


```
const fruits = ["Apple", "Banana"];
fruits.push("Orange");

console.log(fruits); // ["Apple", "Banana", "Orange"]
```

3️⃣ `unshift()` - Add an element at the beginning

The `unshift()` method adds elements to the **beginning** of an array.

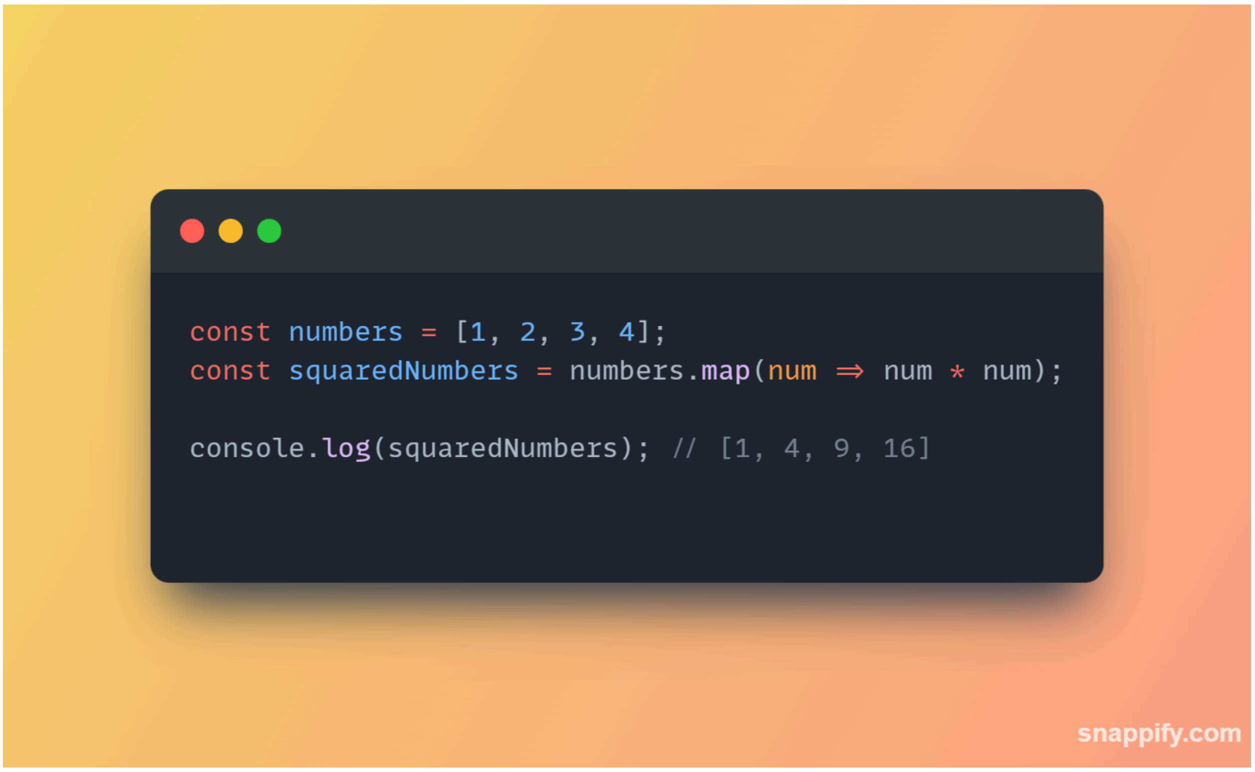


```
const colors = ["Blue", "Green"];
colors.unshift("Red");

console.log(colors); // ["Red", "Blue", "Green"]
```

5️⃣ `map()` - Transform each element in an array

The `map()` method creates a new array by applying a function to each element.



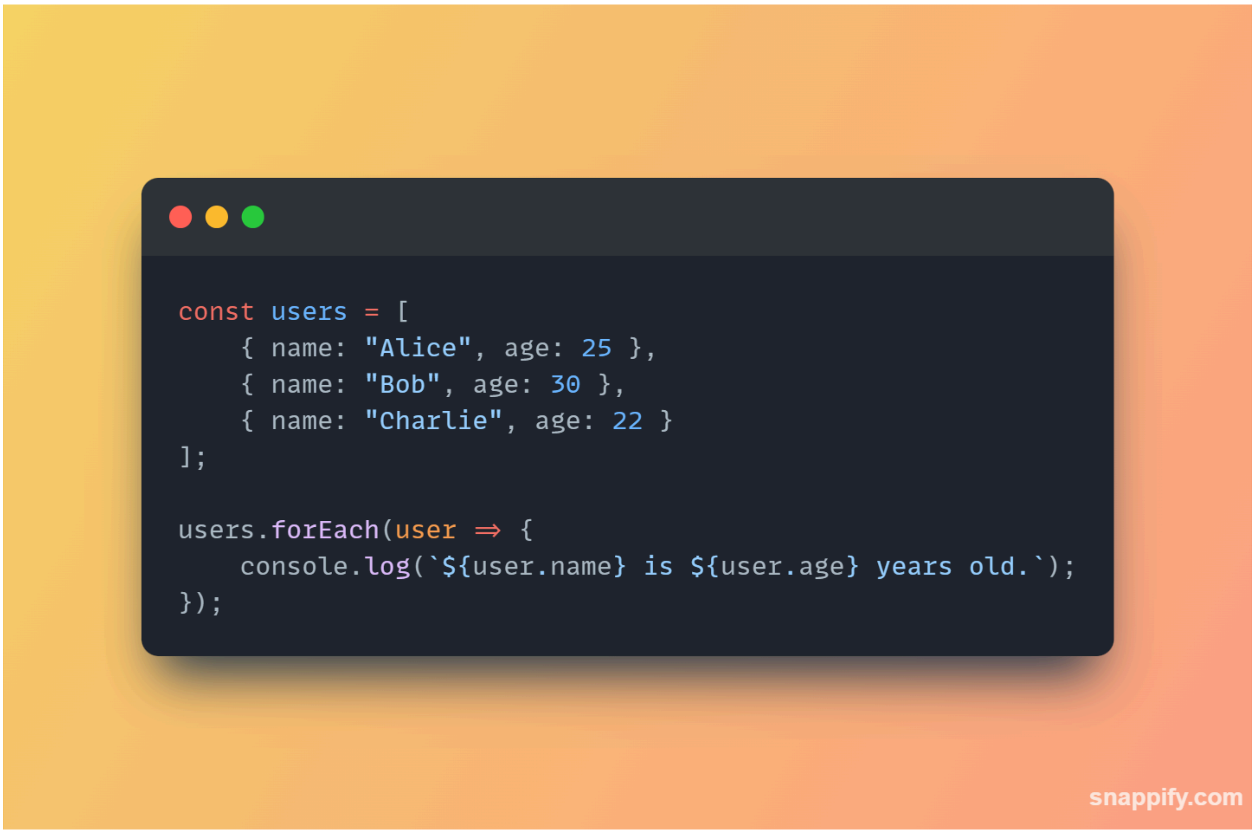
```javascript
const numbers = [1, 2, 3, 4];
const squaredNumbers = numbers.map(num => num * num);

console.log(squaredNumbers); // [1, 4, 9, 16]
```

`.forEach()` Method

The `.forEach()` method in JavaScript loops through each element in an array and executes a callback function for each item.

✅ Does not return a new array (unlike `.map()`).
✅ Executes a function on each element.
✅ Simplifies iteration over arrays.

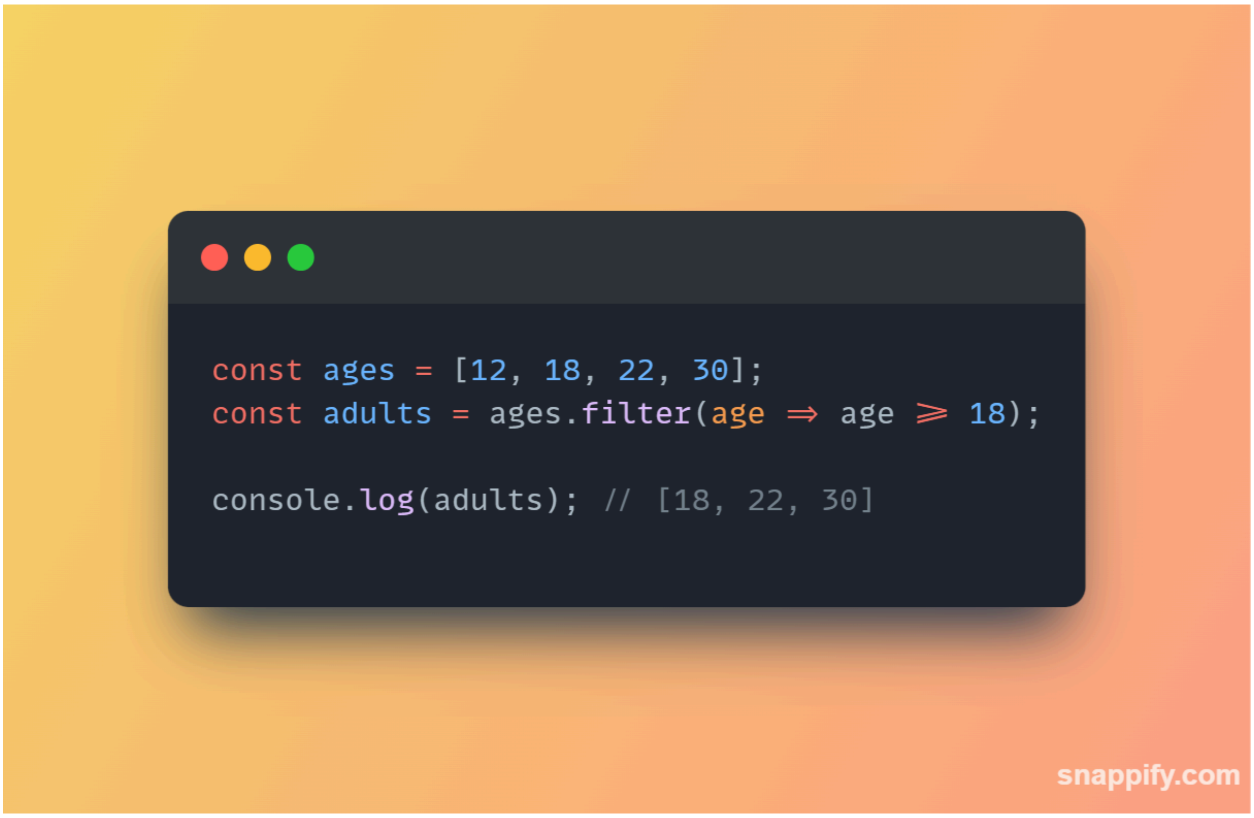


```javascript
const users = [
    { name: "Alice", age: 25 },
    { name: "Bob", age: 30 },
    { name: "Charlie", age: 22 }
];

users.forEach(user ⇒ {
    console.log(`${user.name} is ${user.age} years old.`);
});
```

6️⃣ `filter()` - Get specific elements from an array

The `filter()` method returns a new array containing elements that satisfy a condition.

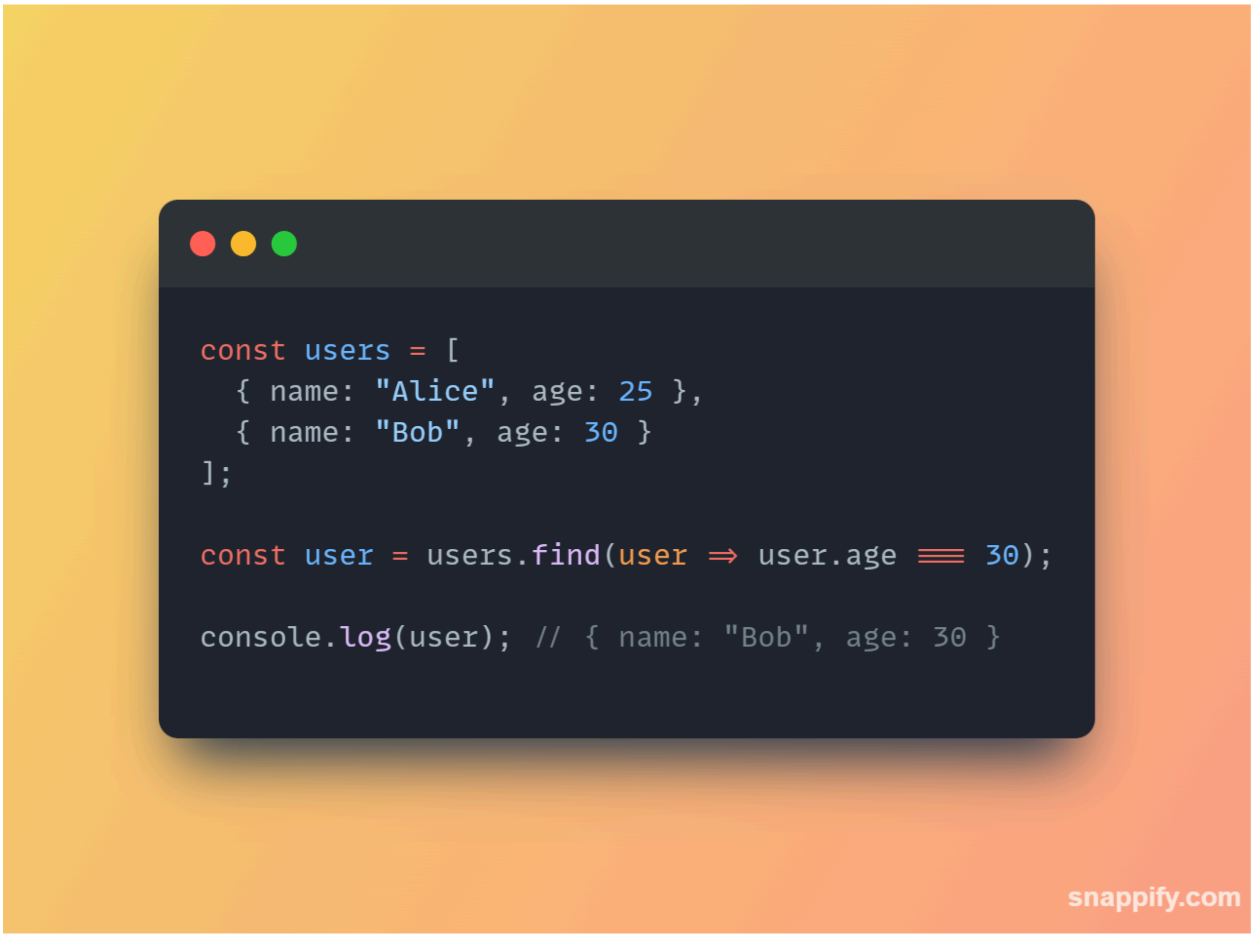


```javascript
const ages = [12, 18, 22, 30];
const adults = ages.filter(age => age >= 18);

console.log(adults); // [18, 22, 30]
```

7️⃣ `find()` - Find the first match

The `find()` method returns the first element that meets a condition.



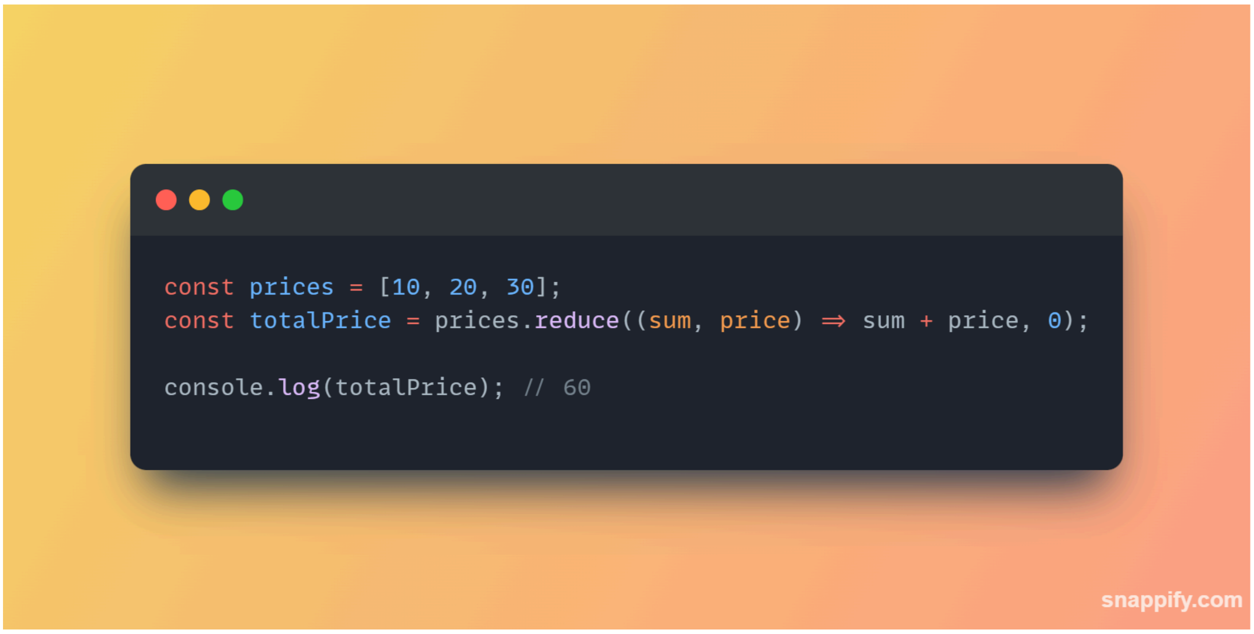
```javascript
const users = [
  { name: "Alice", age: 25 },
  { name: "Bob", age: 30 }
];

const user = users.find(user => user.age === 30);

console.log(user); // { name: "Bob", age: 30 }
```

8 `reduce()` - Accumulate values into a single result

The `reduce()` method executes a function on each element to reduce the array into a single value.

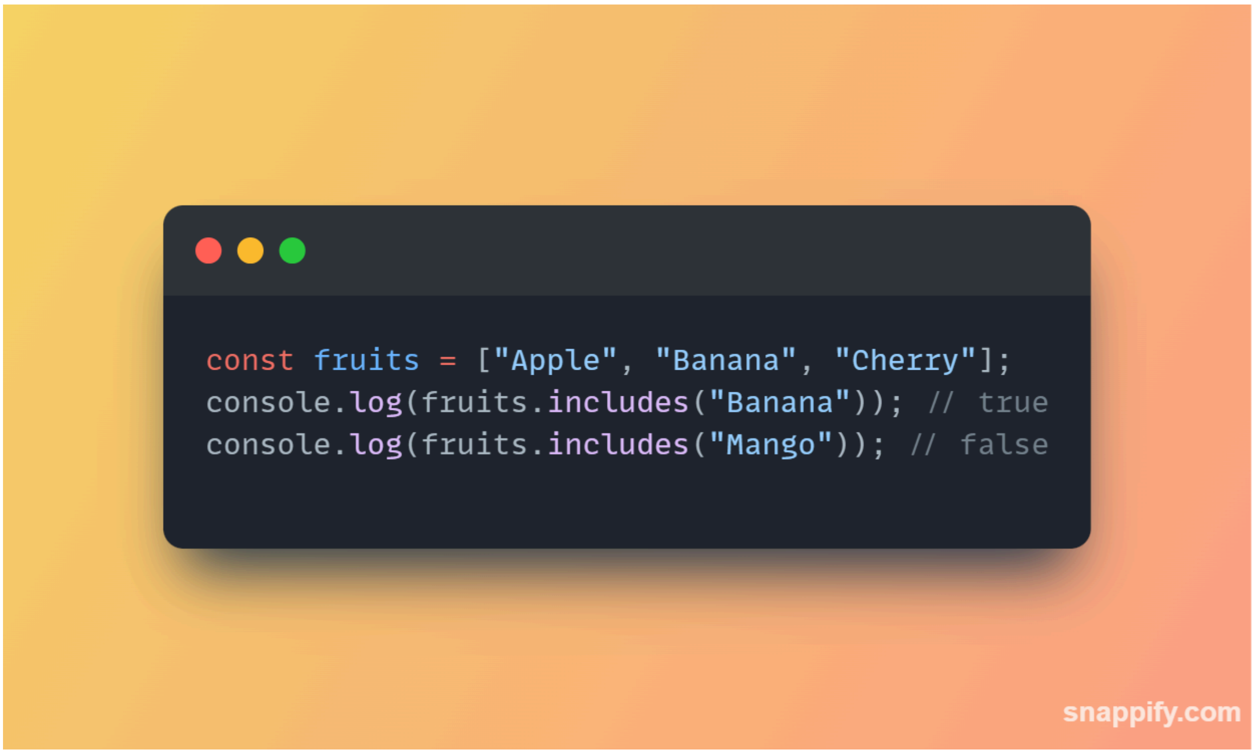


```
const prices = [10, 20, 30];
const totalPrice = prices.reduce((sum, price) => sum + price, 0);

console.log(totalPrice); // 60
```

🔟 `includes()` - Check if an element exists in an array

The `includes()` method returns `true` if an element is found in the array.



```
const fruits = ["Apple", "Banana", "Cherry"];
console.log(fruits.includes("Banana")); // true
console.log(fruits.includes("Mango")); // false
```