Mehdi Bouchachi

THE COMPLETE FRONT-END DEVELOPMENT

SECTION

JAVASCRIPT REVEIW

LECTURE

SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??

@Mehdibouchachi

SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??

BEFORE WE START

👉 In JavaScript, some logical operators, such as **the && and the || operator**, have a feature called short circuiting.

👉 **Short circuiting** in logical operators, means that, in certain conditions, the operator will **immediately return** the first value and **not even look** at the second value.

👉 And this probably **sounds confusing** so, of course, let's write some code here.

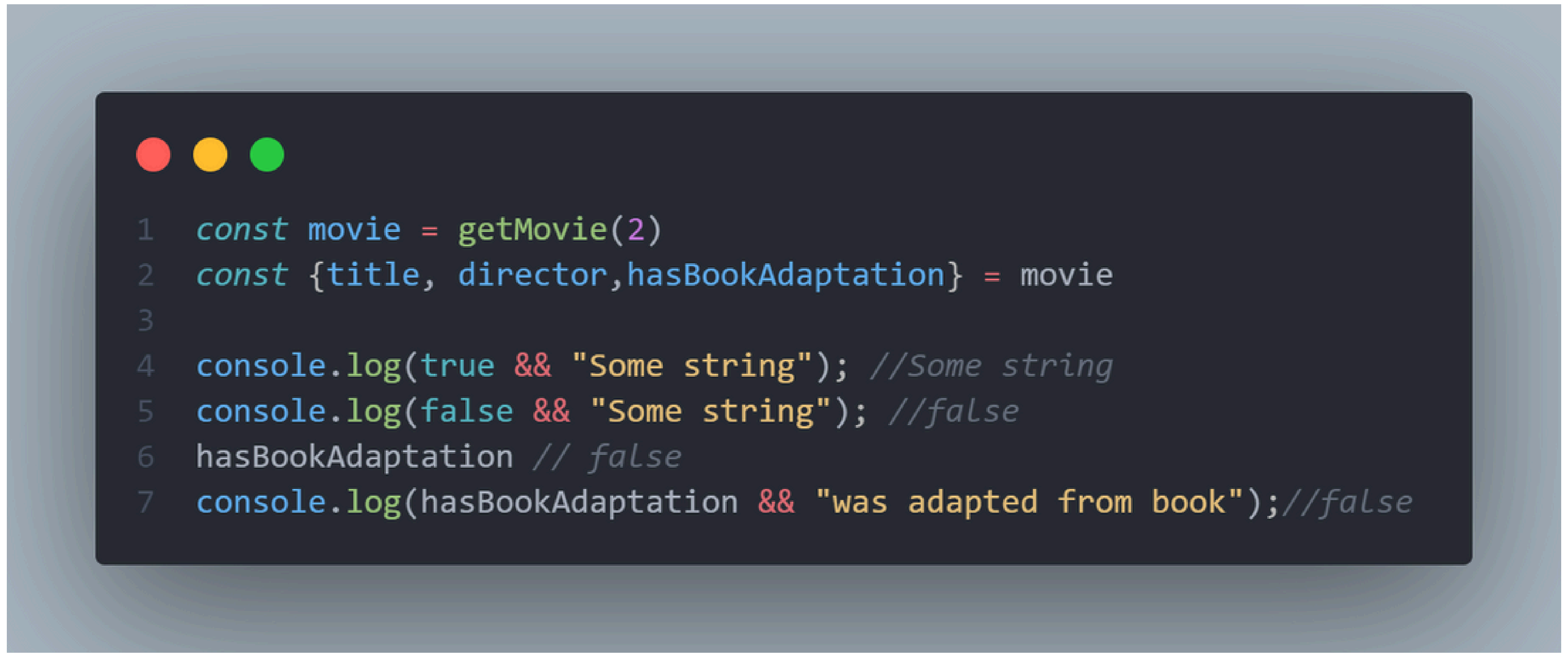
SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??

THE && OPERATOR

The and operator short circuits work when the first operate is false.

- 👉 So when **the first value is** false and then will **immediately return** that first value.

- 👉 So when **the first value is** true the end operator will automatically return the second operant.

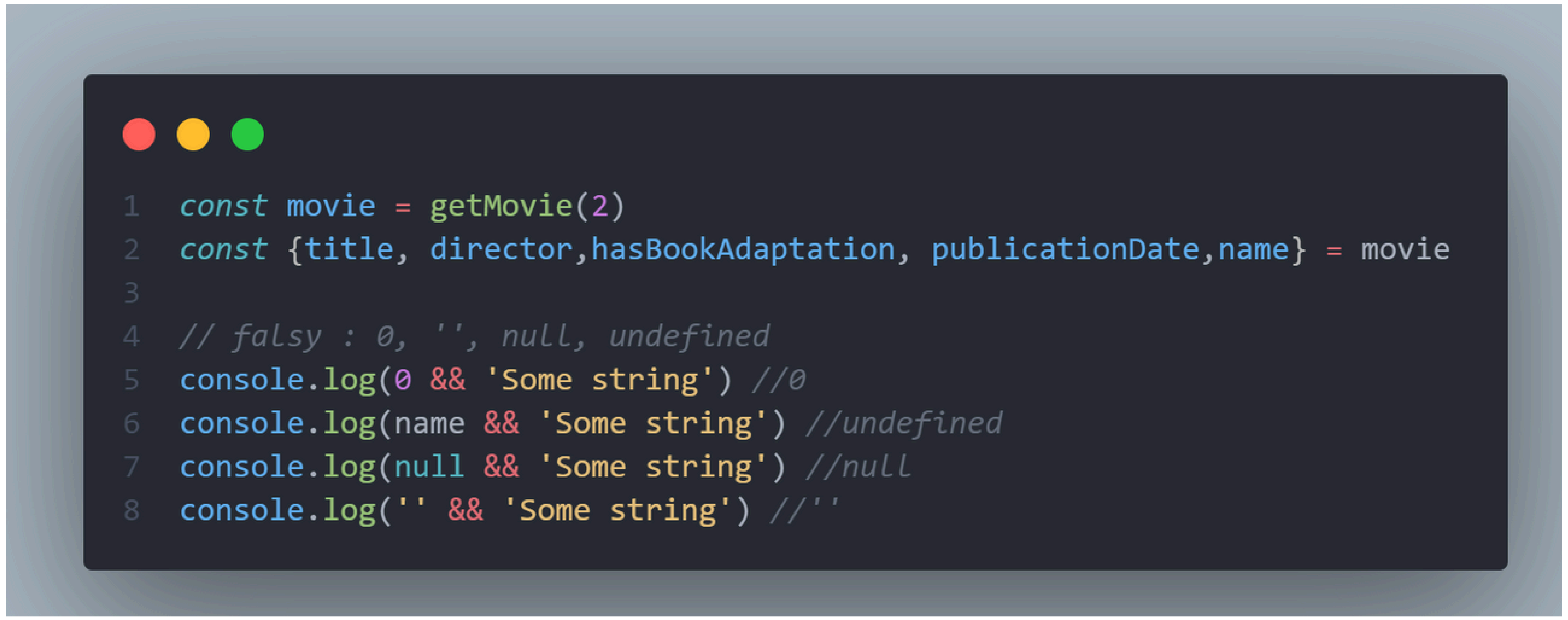


```javascript
const movie = getMovie(2)
const {title, director,hasBookAdaptation} = movie

console.log(true && "Some string"); //Some string
console.log(false && "Some string"); //false
hasBookAdaptation // false
console.log(hasBookAdaptation && "was adapted from book");//false
```

SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??

👉 this also works with so-called **truthy and falsy values.**

👉 The falsy values is 0 , ‘ ’ , null , undefined.



```
const movie = getMovie(2)
const {title, director,hasBookAdaptation, publicationDate,name} = movie

// falsy : 0, '', null, undefined
console.log(0 && 'Some string') //0
console.log(name && 'Some string') //undefined
console.log(null && 'Some string') //null
console.log('' && 'Some string') //''
```

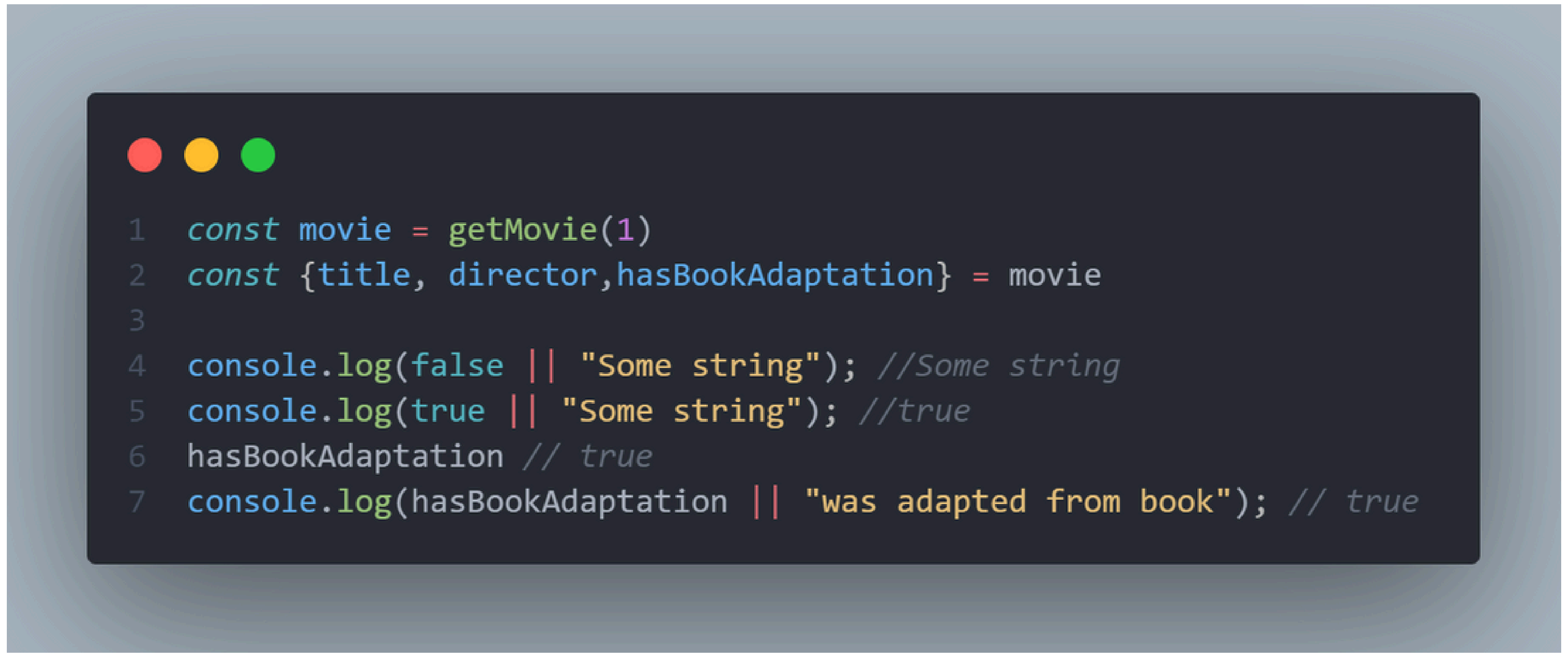
SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??

THE || OPERATOR

The or operator short circuits work when the first operate is true.

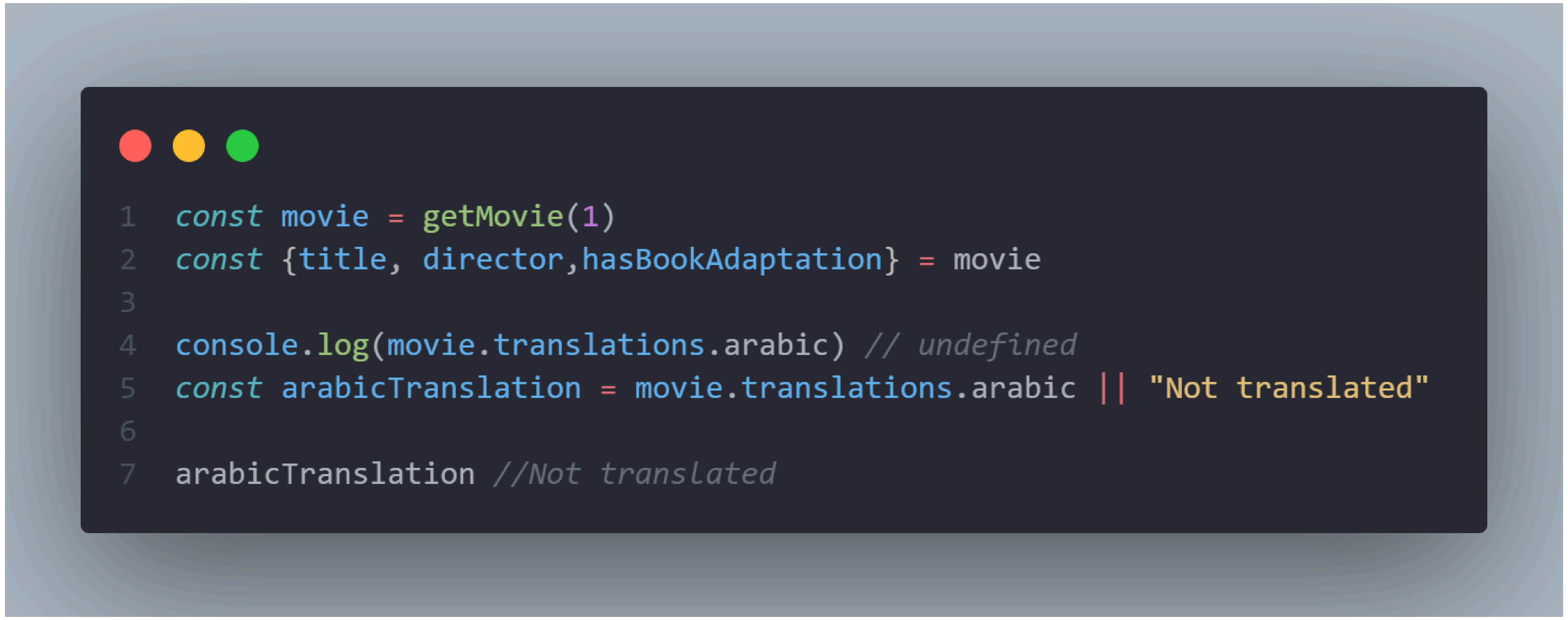
👉 So when **the first value is** true and then will immediately return that first value.

👉 So when the first value is false the end operator will automatically return the second operant.



```
const movie = getMovie(1)
const {title, director,hasBookAdaptation} = movie

console.log(false || "Some string"); //Some string
console.log(true || "Some string"); //true
hasBookAdaptation // true
console.log(hasBookAdaptation || "was adapted from book"); // true
```

SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??



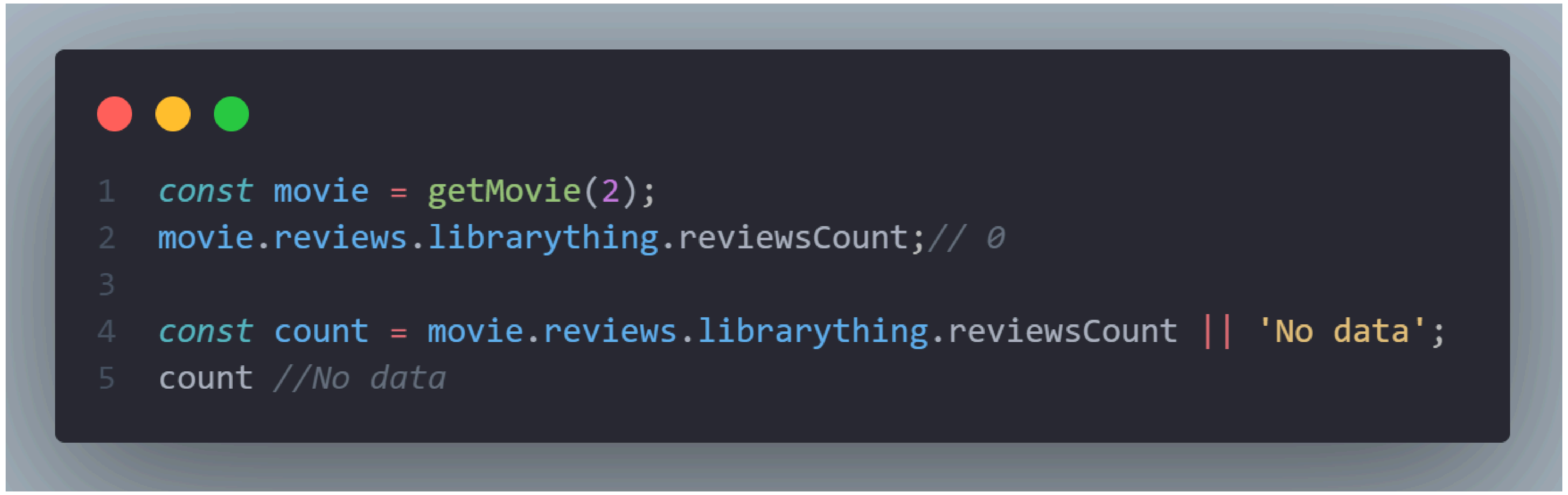
```javascript
const movie = getMovie(1)
const {title, director,hasBookAdaptation} = movie

console.log(movie.translations.arabic) // undefined
const arabicTranslation = movie.translations.arabic || "Not translated"

arabicTranslation //Not translated
```

SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??

PROBLEM!

👉 this can also go wrong because this works for **all the falsy values** such as zero or empty string as well.

👉 Sometimes can have some consequences.



```
const movie = getMovie(2);
movie.reviews.librarything.reviewsCount;// 0

const count = movie.reviews.librarything.reviewsCount || 'No data';
count //No data
```

SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??

SOLUTION

👉 JavaScript has recently added a **new logical operator** which is called the nullish coalescing operator.

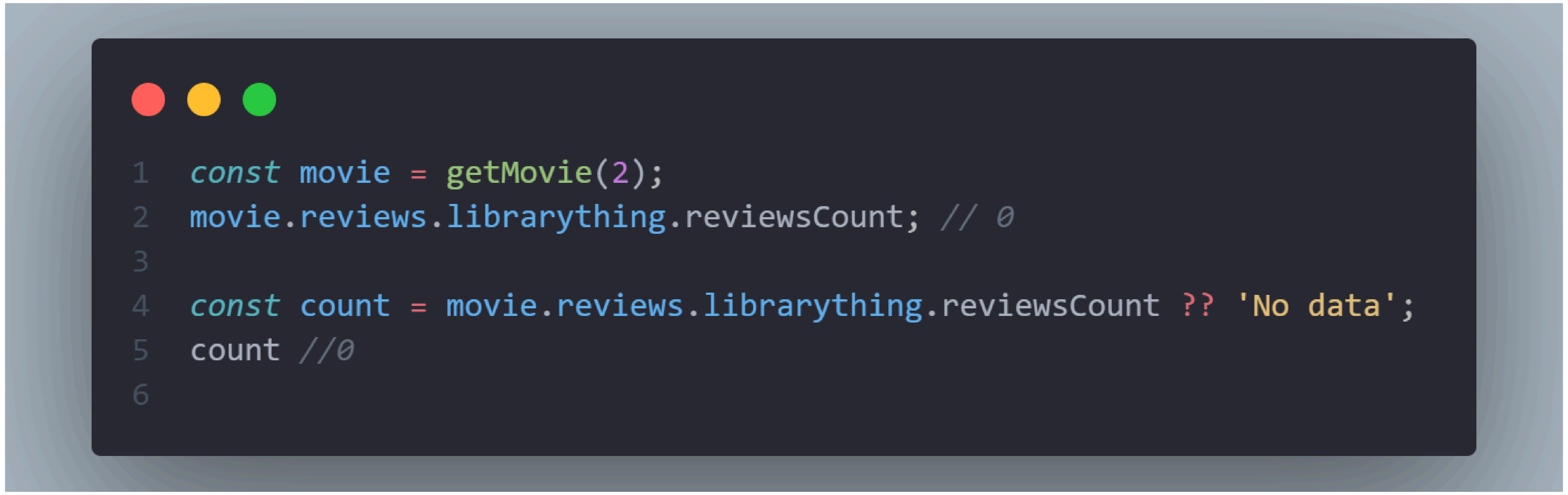
👉 it works very similarly to **the or operator.**

SHORT-CIRCUITING AND LOGICAL OPERATORS: &&, ||, ??

THE ?? OPERATOR

The nullish coalescing operator short circuits work same as the or operator.

👉 The falsy values is **null** , **undefined**.



```javascript
const movie = getMovie(2);
movie.reviews.librarything.reviewsCount; // 0

const count = movie.reviews.librarything.reviewsCount ?? 'No data';
count //0

```

SEE YOU SOON...