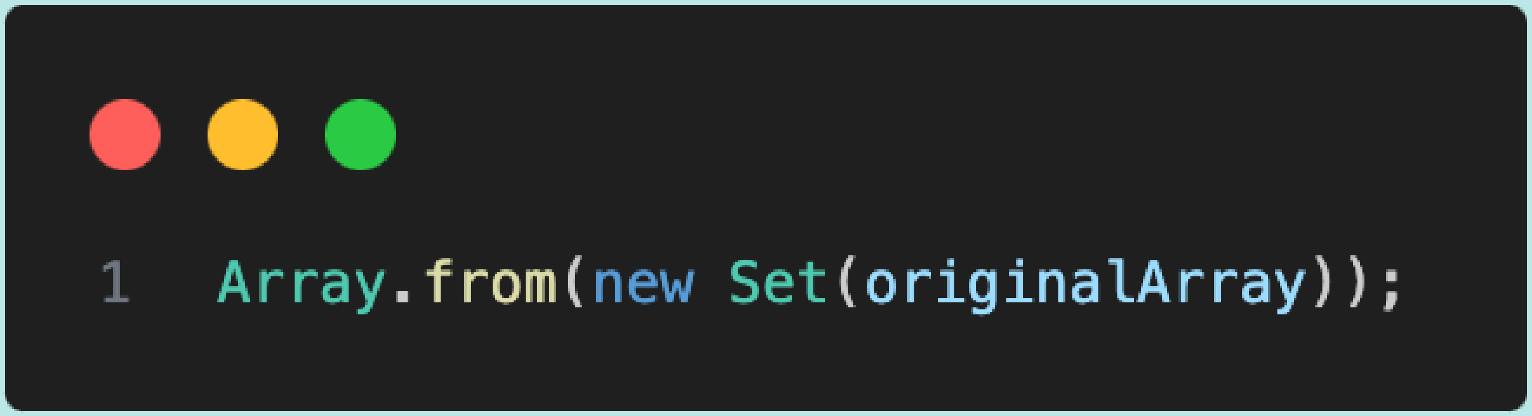
Remove Duplicates From ARRAY

Omkesh B. kendre

omken.dev

JavaScript

Using Set

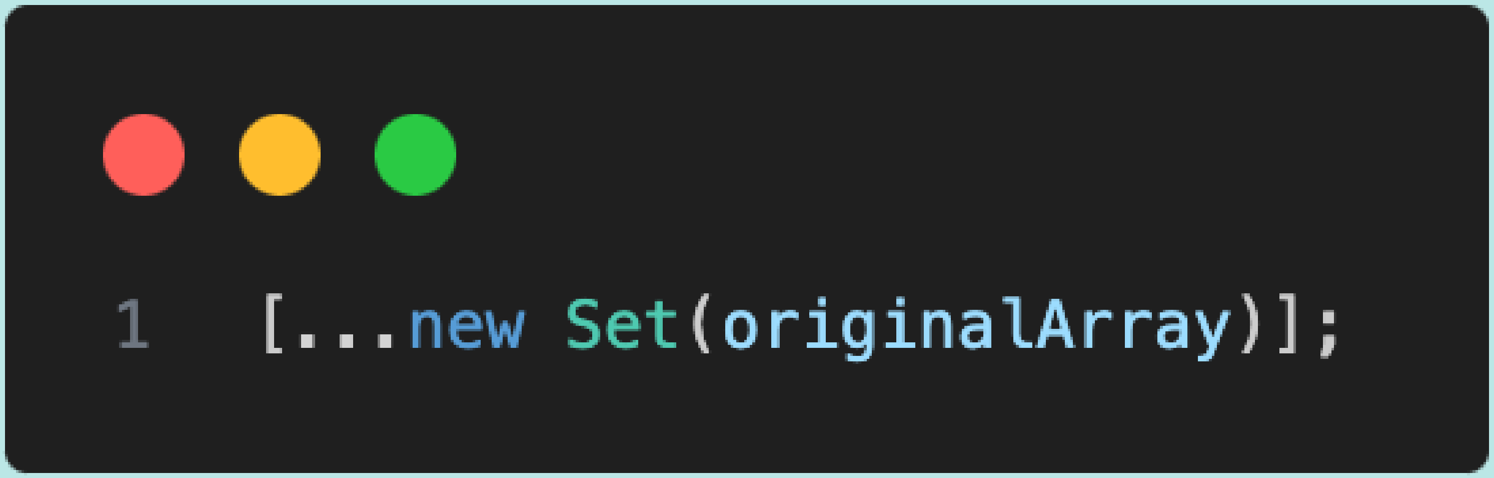


```javascript
Array.from(new Set(originalArray));
```

- **Efficiency:** Utilizes the Set data structure, ensuring uniqueness without the need for manual checks, resulting in a concise and efficient solution.
- **Conciseness:** Provides a succinct one-liner using the Set constructor and Array.from, making the code easy to read and understand.

JavaScript

Using Spread Operator and Set

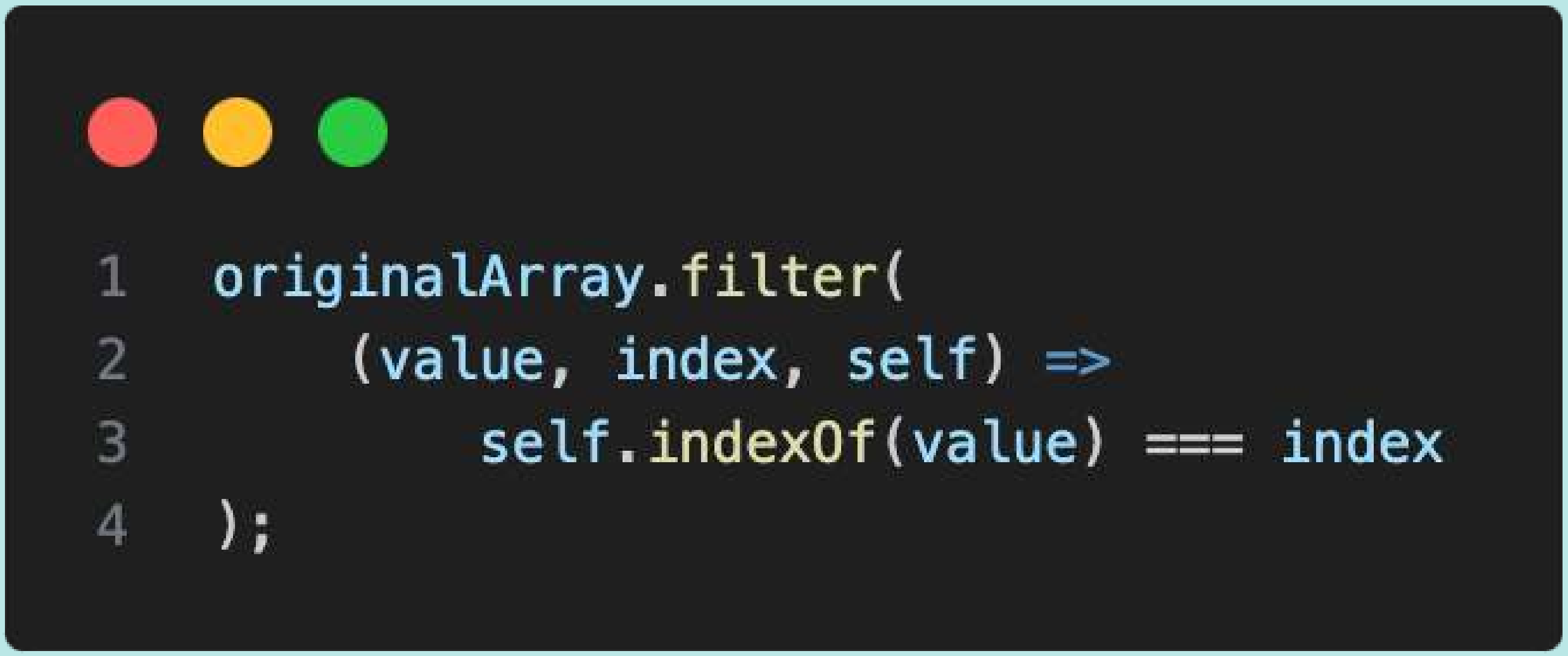


```javascript
[...new Set(originalArray)];
```

- **Compact Syntax:** Leverages the spread operator to create a new array from the set, providing a clean and concise syntax.
- **Readability:** Offers a straightforward and readable approach, making it an elegant solution for removing duplicates.

JavaScript

Using Filter

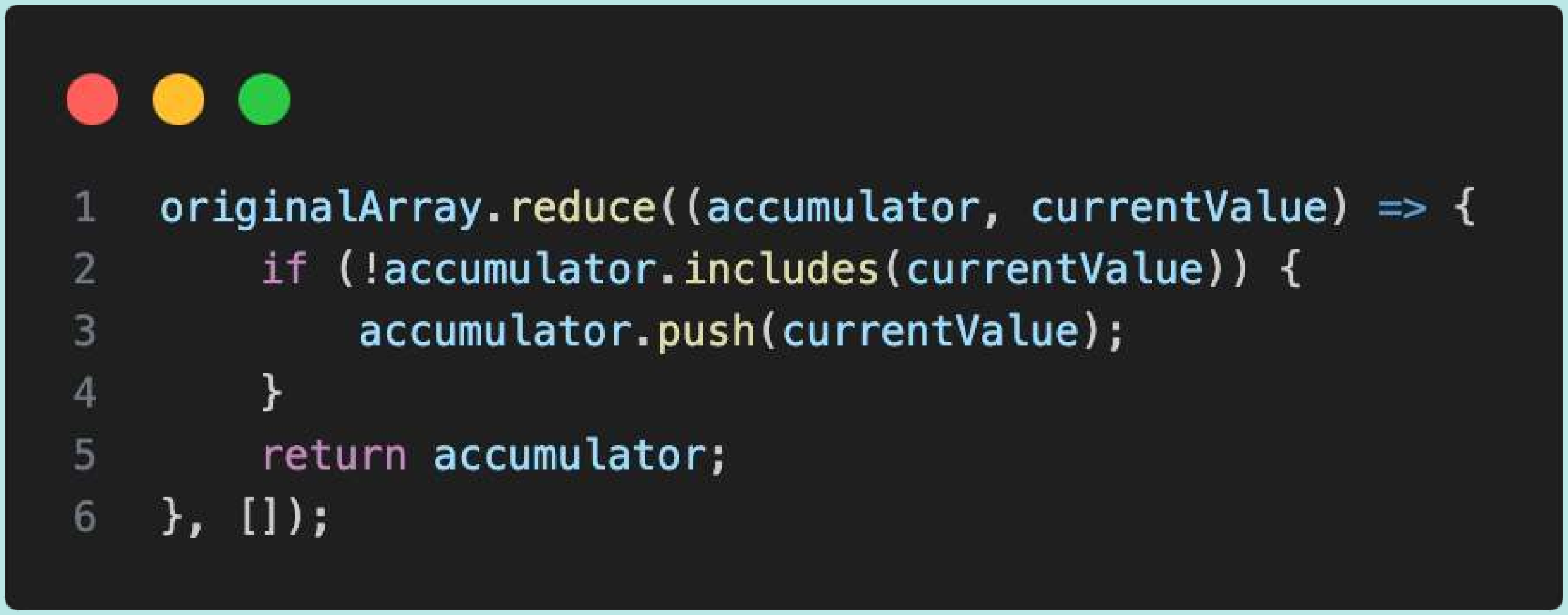


```
originalArray.filter(
    (value, index, self) =>
        self.indexOf(value) === index
);
```

- **Clarity:** Utilizes the filter method with an inline condition to create a new array containing only unique elements, enhancing code clarity.
- **Flexibility:** Allows for customization by adjusting the condition inside the filter function, making it adaptable to various scenarios.

JavaScript

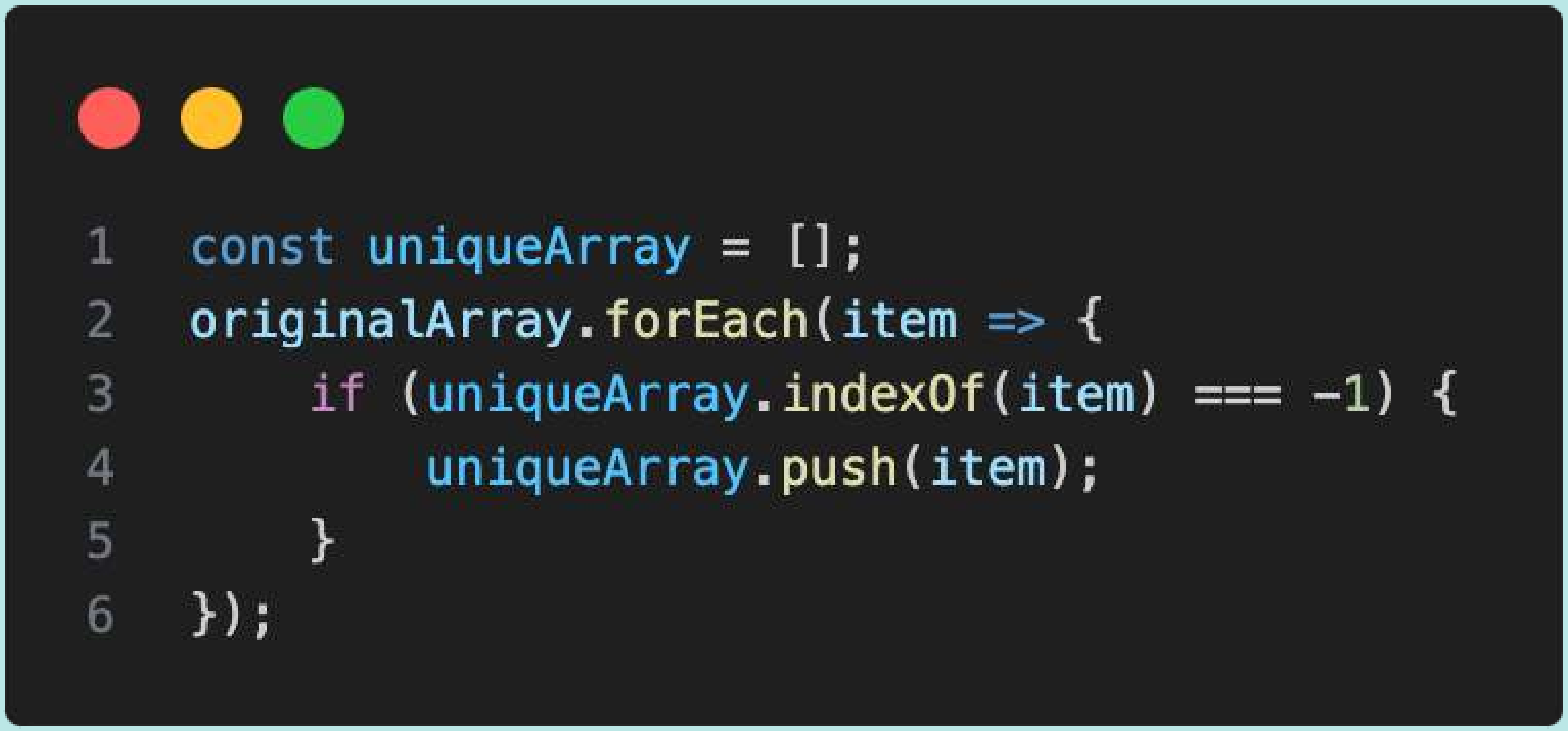
Using Reduce


```
originalArray.reduce((accumulator, currentValue) => {
    if (!accumulator.includes(currentValue)) {
        accumulator.push(currentValue);
    }
    return accumulator;
}, []);
```

- **Controlled Accumulation:** Uses the reduce method for controlled accumulation, ensuring duplicates are not added to the resulting array.
- **Versatility:** Provides a foundation for more complex logic within the reduce function, offering flexibility for additional processing.

JavaScript

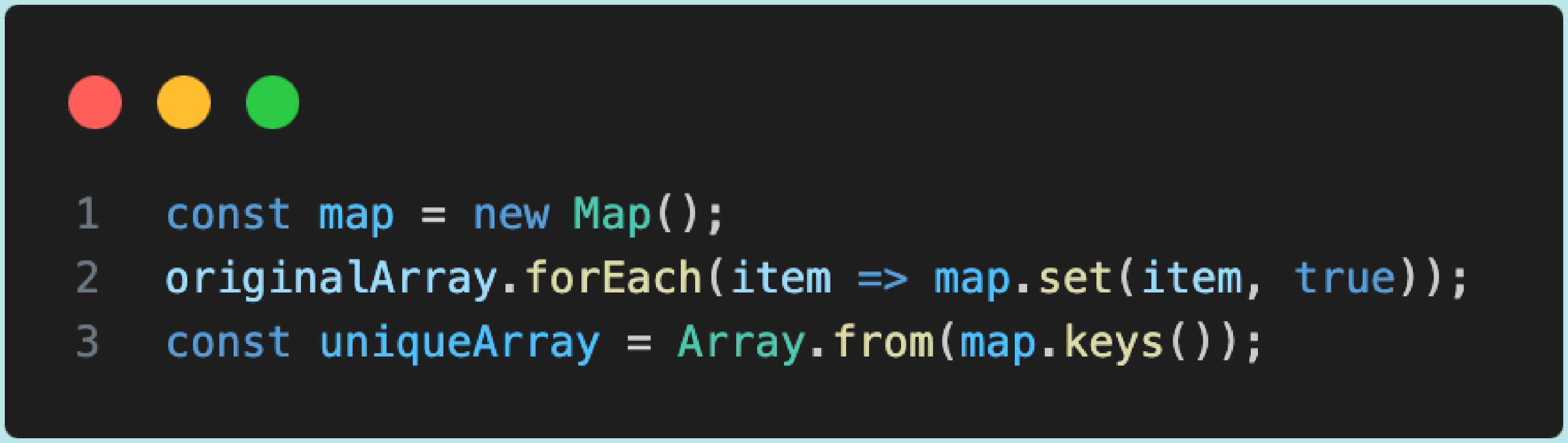
Using forEach and indexOf


```
const uniqueArray = [];
originalArray.forEach(item => {
    if (uniqueArray.indexOf(item) === -1) {
        uniqueArray.push(item);
    }
});
```

- **Elementary Approach:** Implements a basic loop using forEach and indexOf for a straightforward way to filter out duplicates.
- **Intuitiveness:** Offers an intuitive method suitable for smaller arrays and scenarios where simplicity is prioritized.

JavaScript

Using Map


```javascript
const map = new Map();
originalArray.forEach(item => map.set(item, true));
const uniqueArray = Array.from(map.keys());
```

- **Key-Value Pairs:** Leverages the unique key feature of a **Map** to store distinct elements, taking advantage of the Map's inherent uniqueness.
- **Conversion to Array:** Transforms the map keys into an array, providing a neat way to obtain the array of unique elements.

THANK YOU

FOR YOUR SUPPORT

Omkesh B. Kendre

omken.dev

DO you Like it

Save it or loos it