# some/every

Download Demo Code <../js-array-methods-some-every-demo.zip>

## Goals

- Understand what some and every do
- Write your own version of some and every

## some

- Iterates through an array
- Runs a callback on each value in the array
- If the callback returns true for at least one single value, return true
- Otherwise, return false
- the result of the callback will always be a boolean

### An Example

```
let numbers = [1,2,3];

numbers.some(function(value, index, array){
  return value < 3;
});

// true

let numbers = [1,2,3];

numbers.some(function(value, index, array){
  return value > 10;
});

// false
```

### How Does It Work?

```
function some(array, callback){
  for(let i = 0; i < array.length; i++){
    if(callback(array[i], i, array) === true){
      return true;
    }
  }
  return false;
}
```

- Iterates through an array
- Runs a callback on each value in the array
- If the callback returns true for at least one single value, return true
- Otherwise, return false

### Using Some In A Function

```
function hasAdmin(arr){
  return arr.some(function(value){
    return value.admin
  });
}

hasAdmin([
  {name: "Colt", admin: true},
  {name: "Poppy", admin: false}
]); // true

hasAdmin([{name: "Colt"}, {name: "Poppy", admin: false}]); // false
```

```
function hasQuestionMark(str){
  return str.split('').some(function(value){
    return value === '?';
  });
}

hasQuestionMark('How are you feeling'); // false
hasQuestionMark('How are you feeling now?'); // true
```

### When You Would Use Some

- You need to determine if at least one value in an array exists and you have to determine this by using a callback (not includes/indexOf)
- A simple alternative to using filter and seeing if the array contains at least one element

## every

- Iterates through an array
- Runs a callback on each value in the array
- If the callback returns false for any single value, return false
- Otherwise, return true
- the result of the callback will always be a boolean

### An Example

```
let numbers = [1,2,3];

numbers.every(function(value, index, array){
  return value > 0;
});

// true
let numbers = [1,2,3];

numbers.every(function(value, index, array){
  return value > 2;
});

// false
```

## How Does It Work?

- Iterates through an array
- Runs a callback on each value in the array
- If the callback returns false for any single value, return false
- Otherwise, return true

```
function every(array, callback){
  for(let i = 0; i < array.length; i++){
    if(callback(array[i], i, array) === false){
      return false;
    }
  }
  return true;
}
```

## Using Every In A Function

```
function allVowels(str){
  return str.split('').every(function(value){
    return "aeiou".includes(value)
  });
}

allVowels('awesome') // false
allVowels('aiaieoeoiu') // true
function allIntegers(arr){
  return arr.every(Number.isInteger);
}

allIntegers([1,2,3,4,4,4,4]) // true
allIntegers([5,1,4,3,2.2]) // false
```

## When You Would Use Every

- You need to determine if every value in an array exists and you have to determine this by using a callback
- A simple alternative to using filter and seeing if the filtered array is of the same length as the original array

## Recap

- some iterates through an array and runs a callback on each value,
- if the callback for at least one value returns true, some returns true, otherwise false
- every iterates through an array and runs a callback on each value,
- if the callback at any time returns false, every returns false