

Arrays

Arrays

Arrays are lists of ordered, stored data. They can hold items that are of any data type. Arrays are created by using square brackets, with individual elements separated by commas.

Index

Array elements are arranged by *index* values, starting at `0` as the first element index. Elements can be accessed by their index using the array name, and the index surrounded by square brackets.

Property `.length`

The `.length` property of a JavaScript array indicates the number of elements the array contains.

```
// An array containing numbers
const numberArray = [0, 1, 2, 3];

// An array containing different data types
const mixedArray = [1, 'chicken', false];
```

```
// Accessing an array element
const myArray = [100, 200, 300];

console.log(myArray[0]); // 100
console.log(myArray[1]); // 200
console.log(myArray[2]); // 300
```

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

numbers.length // 4
```

Method `.push()`

The `.push()` method of JavaScript arrays can be used to add one or more elements to the end of an array. `.push()` mutates the original array and returns the new length of the array.

Method `.pop()`

The `.pop()` method removes the last element from an array and returns that element.

Mutable

JavaScript arrays are *mutable*, meaning that the values they contain can be changed. Even if they are declared using `const`, the contents can be manipulated by reassigning internal values or using methods like `.push()` and `.pop()`.

```
// Adding a single element:  
const cart = ['apple', 'orange'];  
cart.push('pear');
```

```
// Adding multiple elements:  
const numbers = [1, 2];  
numbers.push(3, 4, 5);
```

```
const ingredients = ['eggs', 'flour', 'chocolate'];  
  
const poppedIngredient = ingredients.pop(); //  
'chocolate'  
console.log(ingredients); // ['eggs', 'flour']
```

```
const names = ['Alice', 'Bob'];  
  
names.push('Carl');  
// ['Alice', 'Bob', 'Carl']
```