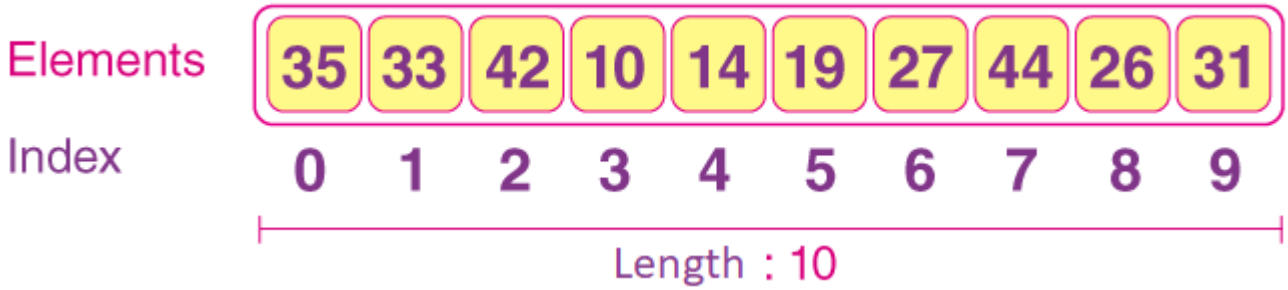
7.6 JS Arrays

pulis pangkalawakan - answered by Ralph Emerson Prado



Parts of an array

- An **array** is a special variable, which can hold more than one value:

  ```
  const cars = ["Saab", "Volvo", "BMW"];
  ```

- If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

  ```
  let car1 = "Saab";
  let car2 = "Volvo";
  let car3 = "BMW";
  ```

  - However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300?
  - The solution is an array.

- An array can hold many values under a single name, and you can access the values by referring to an index number.

Keyword(s)	Description	Sample output	Console output

`.length`	• Returns the number of elements in an array or characters in a string.	`const fruits = ["apple", "banana", "orange", "strawberry"];`
		`const num_fruits = fruits.length;`
		`console.log("There are " + num_fruits + " fruits in the basket.");`
			`There are 4 fruits in the basket.`

`.sort()`	• Sorts the elements of an array in ascending order based on their string representations.	`const fruits = ["apple", "banana", "orange", "strawberry"];`
		`fruits.sort();`
		`console.log(fruits);`
			`[ "apple", "banana", "orange", "strawberry" ]`

`.reverse()`	• Reverses the order of elements in an array.	`const fruits = ["apple", "banana", "orange", "strawberry"];`
		`fruits.reverse();`
		`console.log(fruits);`
			`[ "strawberry", "orange", "banana", "apple" ]`

`.push()`	• Adds one or more elements to the end of an array and returns the new length of the array.	`const fruits = ["apple", "banana", "orange"];`
		`fruits.push("strawberry");`
		`fruits.unshift("grape");`
		`const last_fruit = fruits.pop();`
		`const first_fruit = fruits.shift();`
			`Fruits after manipulation:`
			`[ "apple", "banana", "orange"]`
			`Last fruit removed: strawberry`
			`First fruit removed: grape`

`.unshift()`	• Adds one or more elements to the beginning of an array and returns the new length of the array.		

`.pop()`	• Removes and returns the last element from an array.	`console.log("Fruits after manipulation:");` `console.log(fruits);` `console.log("Last fruit removed:", last_fruit);` `console.log("First fruit removed:", first_fruit);`	
`.shift()`	• Removes and returns the first element from an array.		

Creating an array

- Using an array literal is the easiest way to create a JavaScript Array.

  ```
  const array_name = [item1, item2, ...];
  ```

- It is a common practice to declare arrays with the `const` keyword.

  ```
  const cars = ["Saab", "Volvo", "BMW"];
  ```

- You can also create an array, and then provide the elements:

  ```
  const cars = [];
  cars[0]= "Saab";
  cars[1]= "Volvo";
  cars[2]= "BMW";
  ```

- Alternatively, the following example also creates an Array, and assigns values to it:

  ```
  const cars = new Array("Saab", "Volvo", "BMW");
  ```

Getting the array length

- The `length` property of an array returns the length of an array (the number of array elements).

  ```
  const fruits = ["Banana", "Orange", "Apple", "Mango"];
  let length = fruits.length;
  ```

- The `length` property also returns the length of a string:

  ```
  let txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  let length = txt.length;
  ```

Accessing array elements

- You access an array element by referring to the **index number**.

```
const cars = ["Saab", "Volvo", "BMW"];
let car = cars[0];
```

- To access the first element of an array, you would access the first element of the array at index 0.

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
let fruit = fruits[0];
```

- To access the last element of an array, you would access the last element of the array at the last position.

```
const fruits = ["Banana", "Orange", "Apple", "Mango"];
let fruit = fruits[fruits.length - 1];
```

Accessing the full array

- The full array can be accessed by referring to the array name:

```
const cars = ["Saab", "Volvo", "BMW"];
alert(cars);
console.log(cars);
```

- The elements can also be cycled through using loops.

```
const cars = ["Saab", "Volvo", "BMW"];
for (let i = 0; i < cars.length; i++) {
  alert(cars);
  console.log(cars);
}
```

Array properties and methods

- The real strength of JavaScript arrays are the built-in array properties and methods:

```
cars.length // Returns the number of elements
cars.sort() // Sorts the array
cars.reverse() // Reverses the array
```

Updating an array element

- This statement changes the value of the first element in cars:

```
cars[0] = "Opel";
```

- For example:

```
const cars = ["Saab", "Volvo", "BMW"];
cars[0] = "Opel";
```

Additional Material

- **Learn more**
  - W3Schools
- **Recommended watch**