



Лекция 5

Массивы

Типы данных

1. Прimitives

- a. Числа (Numbers);
- b. Строки (Strings);
- c. Логические (Boolean)
- d. Символ (Symbol);
- e. null;
- f. undefined.

2. Сложные

- a. Объекты (Objects);
- b. Массивы (Arrays);
- c. Функции (Functions).

Массив (Array) --

разновидность объекта, которая предназначена для хранения пронумерованных значений и предлагает дополнительные методы для удобного манипулирования такой коллекцией.



Inspect



Conso



Debugg



Style Edit



Filter output

```
>> typeof [1, 2, 3]
```

```
← "object"
```



```
1  /*
2      Метод Array.isArray() возвращает true, если
3      объект является массивом и false, если он
4      массивом не является.
5  */
6
7  const arr = [1, 2, 3, 4, 5];
8
9  Array.isArray(arr); // true
10
```

Значения в массивах
нумеруются с 0

Как создать массив?

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js x
1  /*
2     Первый способ создать массив --
3     через функцию-конструктор
4  */
5
6  const arr = new Array(1, 2, 3, 4);
7  console.log(arr) // [1, 2, 3, 4]
8
9  const secondArr = new Array(4);
10 console.log(secondArr) // [<4 empty slots>]
11
```

Line 11, Column 1 UTF-8 Tab Size: 4 JavaScript

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js x
1  /*
2      Второй способ создать массив --
3      через квадратные скобки
4  */
5
6  const arr = [1, 2, 3, 4];
7  console.log(arr) // [1, 2, 3, 4]
8
9  const secondArr = [4];
10 console.log(secondArr) // [4]
11
```

Line 10, Column 30 UTF-8 Tab Size: 4 JavaScript

Методы **Array.from()** и **Array.of()** преобразуют полученные элементы в массив.

Различие их в том, что **Array.of()** принимает неограниченное число аргументов и превращает их все в массив, а **Array.from()** принимает callback в качестве второго аргумента.

ES6

```
~/Desktop/common.js - Sublime Text (UNREGISTERED)
common.js x
1  const first = Array.of(1, 2, 'string');
2  console.log(first); // [1, 2, "string"]
3
4  const second = Array.from('abc');
5  console.log(first); // ["a", "b", "c"]
6
```

Line 6, Column 1 UTF-8 Tab Size: 4 JavaScript

Как добавить штуки в массив?

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js
1  /*
2      Можно напрямую указать номер элемента
3      в массиве
4  */
5
6  const arr = [];
7  arr[1] = 'String';
8  arr[3] = true;
9
10 console.log(arr);
11 // [ <empty>, "String", <empty>, true ]
12
```

Line 11, Column 40 UTF-8 Tab Size: 4 JavaScript

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js x
1  /*
2      А можно воспользоваться специальными
3      методами -- push и unshift
4  */
5
6  const arr = [3];
7  // Push добавляет элемент в конец
8  arr.push(10); // [3, 10]
9
10 // Unshift добавляет элемент в начало
11 arr.unshift(1, 2); // [1, 2, 3, 10]
12
```

Line 12, Column 1 UTF-8 Tab Size: 4 JavaScript

Размер важен

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js x
1  /*
2     Свойство length возвращает длину.
3     Длина -- это не количество элементов,
4     а индекс последнего элемента + 1
5  */
6
7  const arr = [1, 2, 3, 4, 5];
8  console.log(arr.length); // 5
9
10 const arr = [];
11 arr[1000] = 'Some value';
12
13 console.log(arr.length); // 1001
14
```

Line 14, Column 1 UTF-8 Tab Size: 4 JavaScript

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js x
1  /*
2     Свойство length позволяет
3     укорачивать массив
4  */
5
6  const arr = [1, 2, 3, 4, 5];
7  arr.length = 2;
8
9  console.log(arr); // [1, 2]
10
```

Line 10, Column 1 UTF-8 Tab Size: 4 JavaScript

Удоли(

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js x
1  /*
2      Тут работает оператор delete, однако
3      при его использовании, на месте
4      элемента остается пустое место
5  */
6
7  const arr = [1, 2, 3];
8  delete arr[0];
9  console.log(arr); // [<1 empty slot>, 2, 3];
10
```

Line 10, Column 1 UTF-8 Tab Size: 4 JavaScript

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js x
1  /*
2     Лучше всего использовать методы pop
3     и shift
4  */
5
6  const arr = [1, 2, 3];
7
8  // Pop удаляет последний элемент
9  arr.pop(); // 3
10
11 // Shift удаляет первый элемент
12 arr.shift() // 1
13
```

Line 13, Column 1 UTF-8 Tab Size: 4 JavaScript

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js x
1  /*
2     Для удаления значения, отличного от
3     первого или последнего, можно вызывать
4     метод splice
5  */
6
7  const arr = [1, 2, 3, 4, 5];
8
9  // array.splice(start, deleteCount)
10
11 arr.splice(2, 1);
12 console.log(arr); // [1, 2, 4, 5]
13
```

Line 13, Column 1 UTF-8 Tab Size: 4 JavaScript

```
~/Desktop/common.js - Sublime Text (UNREGISTERED)
common.js x
1  /*
2     Дополнительные аргументы в splice
3     замещают удаленные элементы
4  */
5
6  const arr = [1, 2, 3, 4, 5];
7
8  arr.splice(2, 2, 'new', 1);
9  console.log(arr); // [1, 2, "new", 1, 5]
10
```

Line 10, Column 1 UTF-8 Tab Size: 4 JavaScript

Самостоятельная работа

Напишите программу “Список покупок”. Программа будет состоять из функции/метода **addItem(thing)**, который добавляет продукты в массив **items**.

Функция/метод добавляет в массив только строчные элементы, есть проверка на пустую строку.

Ну это уже перебор

```
~/Desktop/arrays.js - Sublime Text (UNREGISTERED)
arrays.js
1  /*
2      Для перебора массивов можно
3      использовать цикл for
4  */
5
6  const arr = [23, 15, 14, 9, 256, 61];
7
8  for (let i = 0; i <= arr.length - 1; i++)
9      console.log(`${i} элемент массива --
10                  ${arr[i]}`);
11  }
12
```

Line 12, Column 1

UTF-8 Tab Size: 4 JavaScript

```
0 элемент массива -- 23
1 элемент массива -- 15
2 элемент массива -- 14
3 элемент массива -- 9
4 элемент массива -- 256
5 элемент массива -- 61
```


Перебирающие методы массивов

filter() -- выполняет какое-то действие на каждой итерации и возвращает только те объекты, для которых callback вернул *true*;

map() -- превращает один массив элементов в другой;

forEach() -- циклически обходит массив и выполняет заданное действие на каждой итерации;

reduce() -- сворачивает все элементы массива слева направо в одно значение;

reduceRight() -- сворачивает все элементы массива справа налево в одно значение.

Функция обратного вызова (callback) --

функция, которая передается в качестве аргумента другой функции и выполняется после выполнения основного кода.

```
~/Desktop/common.js - Sublime Text (UNREGISTERED)
common.js x
1  /*
2     Метод Array.prototype.forEach() перебирает все
3     элементы массива и выполняет для каждого callback
4  */
5
6  const arr = [1, 2, 3, 4, 5, 6];
7
8  arr.forEach((element, index, array) => {
9      console.log(element * 2); // 2 4 6 8 10 12
10     // Менее изысканный способ
11     console.log(arr[index] * 2);
12 });
13
14 console.log(arr); // Array(6) [ 1, 2, 3, 4, 5, 6 ]
```

Line 14, Column 51 UTF-8 Tab Size: 4 JavaScript

```
common.js x
1  /*
2     Метод Array.prototype.filter() перебирает все
3     элементы массива и возвращает новый массив
4     значений, для которых callback возвращает true
5  */
6
7  const arr = [1, 2, 3, 4, 5, 6];      номер
8                                     элемент  элемента  массив
9  const newArr = arr.filter((element, index, array) => {
10     if (element % 2 === 0) {
11         return element;
12     }
13 });
14
15 console.log(newArr); // Array(3) [ 2, 4, 6 ]
```

```
~/Desktop/common.js - Sublime Text (UNREGISTERED)
common.js x
1  /*
2     Метод Array.prototype.map() создает новый массив,
3     выполняя callback для каждого элемента базового
4     массива
5  */
6
7  const arr = [1, 2, 3, 4, 5, 6];
8
9  const result = arr.map((element, index, array) => {
10     return element * 2;
11 })
12
13 console.log(arr); // Array(6) [ 1, 2, 3, 4, 5, 6 ]
14 console.log(result); // Array(6) [ 2, 4, 6, 8, 10, 12 ]
15
```

Line 15, Column 1 UTF-8 Tab Size: 4 JavaScript

```
common.js x
1  /*
2     map отличается от filter тем, что map возвращает
3     в новый массив столько значений, сколько было и в
4     исходном массиве
5  */
6  const arr = [1, 2, 3, 4, 5, 6];
7
8  const result = arr.map((element, index, array) => {
9      if (element % 2 === 0) {
10         return element;
11     }
12 })
13
14 console.log(result);
15 // Array(6) [ undefined, 2, undefined, 4, undefined, 6 ]
16
```

~/Desktop/common.js - Sublime Text (UNREGISTERED)

common.js x

```
1  /*
2     Метода Array.prototype.reduce() применяет
3     указанную функцию к каждому элементу, сводя
4     массив к одному значению
5  */
6
7  const arr = [1, 2, 3, 4, 5];
8
9  const result = arr.reduce((prev, cur, i, arr) => {
10     return prev + cur;
11  });
12
13  console.log(result); // 15
14
```

Line 14, Column 1 UTF-8 Tab Size: 4 JavaScript

~/Desktop/common.js - Sublime Text (UNREGISTERED)

common.js x

```
1  /*
2      Метода Array.prototype.reduceRight() делает
3      то же самое, только начинается справа, а не
4      слева.
5  */
6
7  const arr = [1, 2, 3, 4, 5];
8
9  const result = arr.reduceRight((prev, cur) => {
10      return prev + cur;
11  });
12
13  console.log(result); // 15
14
```

Line 14, Column 1

UTF-8 Tab Size: 4 JavaScript

Некоторые другие методы массивов

Array.prototype.concat() -- объединяет несколько массивов;

Array.prototype.indexOf() -- возвращает первый индекс, по которому можно найти элемент в массиве;

Array.prototype.every() -- проверяет, удовлетворяют ли все элементы указанному условию;

Array.prototype.some() -- проверяет, удовлетворяют ли какие-то элементы указанному условию;

Некоторые другие методы массивов

ES6 **Array.prototype.find()** -- возвращает значение первого найденного элемента, который соответствует условию;

Array.prototype.join() -- объединяет все элементы массива в строку, можно указать разделитель в качестве параметра;

Array.prototype.toString() -- преобразует массив в строку;