Cheatography

JavaScript Array API Cheat Sheet

by PavelChe via cheatography.com/31265/cs/11818/

Методы массивов (простые)	
arr.indexOf(arg);	Находит индекс первого найденного элемента в массиве
arr.lastIndexOf-(arg);	Находит индекс последнего найденного элемента в массиве
Примечание .index и lastIndex	Если не нашло элемент - возвращает -1
arr[arr.length - 1];	Найти последний эл-т массива
arr[arr.length] = el;	Добавить эл-т в конец массива
delete arr[3];	Удаляет элемент, но оставляет дырку (undefined)
x = arr.splice(3, 1);	Вырезаем часть массива где первый аргумент - это индекс начала, а второй - показывает кол-во элементов (вправо), которые мы хотим вырезать (исходный массив будет изменен)
x = arr.splice(-1, 1);	Вырезаем часть массива где первый аргумент - это индекс начала (с конца), а второй - показывает кол-во элементов (вправо), которые мы хотим вырезать (исходный массив будет изменен)
x = arr.splice(a, b, args);	Если передать еще аргументы (args), то они будут вставлены на место вырезанных.



By **PavelChe**

cheatography.com/pavelche/

Published 14th May, 2017.

Last updated 14th May, 2017.

Sponsored by **Readable.com**

Measure your website readability!

https://readable.com

Методы массивов (простые) (cont)

x = arr.slice(3, 4);	Копирует часть массива, не изменяя исходный. Первый аргумент начало фрагмента (включительно), второй аргумент - конец фрагмента (не включительно)
x = arr.slice(-3, -1);	Копирует часть массива, не изменяя исходный. Первый аргумент начало фрагмента (включительно с конца), второй аргумент - конец фрагмента (не включительно с конца)
x = arr.slice(3);	Копирует все после эл-та под индексом 3 (включительно)
arr.push(-args);	Добавить эл-ты в конец массива
arr.pop();	Вырезает последний эл-т массива
arr.unshift('h','y');	Добавить эл-ты в нач массива
arr.shift();	Вырезает первый эл-т массива
arr.reverse()	Перевернуть массив

arr.length - длина массива (св-во)



By **PavelChe**
cheatography.com/pavelche/

Published 14th May, 2017.
Last updated 14th May, 2017.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Сортировка .sort(callback);

```
var array = [1, 3, 2, 15, 8, 7];
array.s or t(f unc tion(a, b) {
                                if (a > b) {
    return 1;
                                } else if (a < b) {
    return -1;
                                }
});
                                                                                //
Результат: [1, 2, 3, 7, 8, 15];
```

a и b - сравниваемые числа
(Изменяет исходный массив)

Перебор .forEach(callback)

```
var arr = ['Яблоко', 'Апельсин', 'Груша'];
arr.fo rEa ch( fun cti on( item, i, arr) {
                                      con sol e.log( i + ' : ' + item + ' : ' + arr);
                                   });
                                                                                //
Результат: Выведет поочередно эл-ты в
                                                                                -
  кон соль;
```

Производится перебор массива
item - элемент, i - индекс (счетчик), arr - перебираемый массив
(Не изменяет исходный массив)

Фильтр .filter(callback)

```
var arr = [1, -1, 2, -2, 3];
var positi veArr = arr.fi lte r(f unc tio n(num) {
return num > 0; // вернет только положи тельные числа
});
                                                                                //
Результат: positi veArr = [1, 2, 3];
```

Если return возвращает true - то item включен в новый массив
item - элемент, i - индекс (счетчик), arr - перебираемый массив
(не изменяет исходный массив)

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Трансформация .map(callback);

```
var arr = [5, 1, 12, 75, 22];
var arr2 = arr.ma p(f unc tio n(item) {
return item * item;
});
                                                                    //
Результат: [25, 1, 144, 5625, 484]
                                                                    -
  (Во звели в квадрат)
```

Трансформирует массив в новый исходя из переданного в return
item - элемент, i - индекс (счетчик), arr - перебираемый массив
(не изменяет исходный массив)

.compare(callback) и .some(callback)

```
var arr = [1, -1, 3, -2, 2];
var result = a.ever y(f unction (item, i, arr) {
return item > 0;
});
var result2 = a.some (fu nction (item, i, arr) {
return item > 0;
});
```

.every() - в result вернет false. Если return вернет true для всех эл-тов массива, то в переменную вернется true
.some() - в result2 вернет true. Если return вернет true хотя-бы для одного эл-та массива, то в переменную вернется true
item - элемент, i - индекс (счетчик), arr - перебираемый массив

.reduce(callback, num)

```
var arr = [1, 2, 3, 4, 5];
var result = arr.re duc e(f unc tio n(p rev iou sValue, curren tItem, i, arr) {
    return previo usValue + curren tItem;
}, 0);
                                                    // Второй аргумент (число) - значение previo usValue для
                                                          первой итерации (по умолч = 0);
                                                    // Результат: result = 15;
```

Применяет функцию callback по очереди к каждому элементу массива слева-направо, сохраняя при этом промежуточный результат.
previousValue – промежуточный результат, currentItem – текущий эл-т, i - индекс (счетчик), arr - перебираемый массив.
.reduceRight(callback, num) - перебирает справа-налево

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com