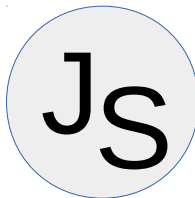


# Объекты и массивы



- Объект
- Свойства объекта
- Объект как массив
- Стандартные свойства и методы объектов
- Массив
- Доступ к элементам массива
- Добавление нового элемента в массив
- Длина массива
- Методы массивов

# Объект



## Конструктор

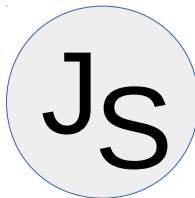
```
let o = new Object( );  
let now = new Date( );  
var new_year_evening = new Date(2017, 11, 31);
```

## Литерал (JSON)

```
let circle = {x:0, y:0, radius:2};
```

```
let person = {  
  name: "Jack Daniel",  
  age: 64,  
  married: true,  
  occupation: "businessman",  
  email: "jack.daniel@google.com"  
};
```

# Свойства объекта



```
table.caption = "Employees";  
table.body = new Object();  
table.body.rows = [  
    {name:"Иванов", salary:1000},  
    {name:"Петров", salary:1700}  
]  
table.body.color = "gray";
```

```
let names = "";  
for(let name in obj){  
    names += name + "\n";  
}  
alert(names);
```

```
delete table.body;  
alert(table.body);
```

# Объект как массив



```
table['caption'] = "Employees";  
table['body'].color = "gray";
```

```
var addr = "";  
for(i = 0; i < 4; i++) {  
    addr += customer["address" + i] + '\n';  
}
```

```
let box = {l:10, w:8, h:3};  
for (dim in box){  
    box[dim] *= 1.2;  
}
```

# Стандартные свойства и методы



## constructor

```
let c = new Complex(3, -4);  
c.constructor == Complex;           //true
```

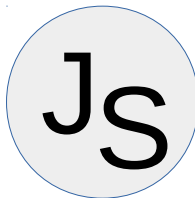
## toString(), toLocaleString(), valueOf()

```
alert('c=' + c);                     // c=5 (valueOf)  
alert('c=' + c.toString());          // c={3,4}
```

## hasOwnProperty()

```
Math.hasOwnProperty("cos");           // true  
c.hasOwnProperty("toString");         // false
```

# Стандартные свойства и методы



## propertyIsEnumerable()

```
var o = { x:1 };  
o.propertyIsEnumerable("x");           // true  
o.propertyIsEnumerable("y");           // false  
o.propertyIsEnumerable("valueOf");     // false
```

## isPrototypeOf()

```
var o = new Object( );  
Object.prototype.isPrototypeOf(o);     // true
```

# Массив



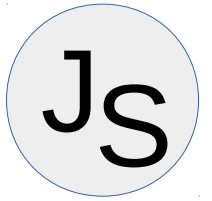
## Конструктор

```
var a = new Array();  
var a = new Array(1, 7, 15, 'элемент', true);  
var a = new Array(10);
```

## Литерал

```
var a = [1, 7, 15, 'элемент', true];  
var b = [[1, {x:1, y:2}], [2, {x:3, y:4}]];
```

# Элементы массива



## Доступ к элементам

```
value = a[9];  
a[3] = 3.14;  
a[i+1] = 'привет';  
a[a[i]] = a[0];  
me['salary'] *= 1.5;
```

## Добавление элементов

```
a[1001] = 9;
```

```
var fruits = ['Apple', 'Orange', 'Pineapple'];  
fruits.push('Kiwi');
```

```
var c = new Point(-1,1);  
c[0] = 'это новый элемент';
```



# Длина массива



```
var a = new Array( );    // a.length == 0
a = new Array(10);      // a.length == 10
a = new Array(1,2,3);    // a.length == 3
a = [4, 5];             // a.length == 2
a[5] = -1;              // a.length == 6
a[49] = 0;              // a.length == 50
```

```
for(var i = 0; i < fruits.length; i++){
    if (fruits[i] != undefined){
        alert(fruits[i]);
    }
}
```

```
a.length = 2;
a.toString();           // 4,5
```

# Методы массива



## join

```
[1,2,3].join();           // '1,2,3'.split(',')  
[1,2,3].join('*');       // '1*2*3'
```

## reverse

```
[1,2,3].reverse();       // [3,2,1]
```

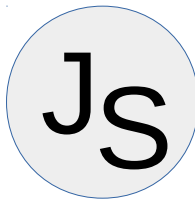
## sort

```
['Kiwi','Apple','Orange'].sort();  
                        // ['Apple','Kiwi','Orange']  
[33, 4, 1111, 222].sort() // [1111,222,33,4]  
a.sort( function(a,b){return a-b;} );
```

## concat

```
[1,2,3].concat(4,5,6);  
[1,2,3].concat([4,5],[6]);
```

# Методы массива



## slice, splice

```
a=[1,2,3,4,5];  
a.slice(0,3);           // [1,2,3]  
a.splice(1,2,11,12);    // [2,3,4]; a:[1,11,12,4,5]
```

## push, pop

```
a.pop();                // 5 ; a:[1,11,12,4]  
a.push(7);              // 5 ; a:[1,11,12,4,7]
```

## shift, unshift

```
a.shift();              // 1 ; a:[11,12,4,7]  
a.unshift(51,52);       // 5 ; a:[51,52,11,12,4,7]
```

## toString, toLocaleString

```
["a", "b", "c"].toString() // 'a,b,c'  
[1, [2, 'c']].toString()   // '1,2,c'
```



- Объект
- Свойства объекта
- Объект как массив
- Стандартные свойства и методы объектов
- Массив
- Доступ к элементам массива
- Добавление нового элемента в массив
- Длина массива
- Методы массивов



## Конструктор

```
let o = new Object( );  
let now = new Date( );  
var new_year_evening = new Date(2017, 11, 31);
```

## Литерал (JSON)

```
let circle = {x:0, y:0, radius:2};
```

```
let person = {  
  name: "Jack Daniel",  
  age: 64,  
  married: true,  
  occupation: "businessman",  
  email: "jack.daniel@google.com"  
};
```

## Свойства объекта



```
table.caption = "Employees";  
table.body = new Object();  
table.body.rows = [  
    {name:"Иванов", salary:1000},  
    {name:"Петров", salary:1700}  
]  
table.body.color = "gray";
```

```
let names = "";  
for(let name in obj){  
    names += name + "\n";  
}  
alert(names);
```

```
delete table.body;  
alert(table.body);
```

## Объект как массив



```
table['caption'] = "Employees";  
table['body'].color = "gray";
```

```
var addr = "";  
for(i = 0; i < 4; i++) {  
    addr += customer["address" + i] + '\n';  
}
```

```
let box = {l:10, w:8, h:3};  
for (dim in box){  
    box[dim] *= 1.2;  
}
```

## Стандартные свойства и методы



### constructor

```
let c = new Complex(3,-4);  
c.constructor == Complex;           //true
```

### toString(), toLocaleString(), valueOf()

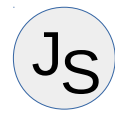
```
alert('c=' + c);                     // c=5 (valueOf)  
alert('c=' + c.toString());          // c={3,4}
```

### hasOwnProperty()

```
Math.hasOwnProperty("cos");           // true  
c.hasOwnProperty("toString");         // false
```



## Стандартные свойства и методы



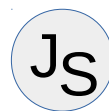
### propertyIsEnumerable()

```
var o = { x:1 };  
o.propertyIsEnumerable("x");           // true  
o.propertyIsEnumerable("y");           // false  
o.propertyIsEnumerable("valueOf");     // false
```

### isPrototypeOf()

```
var o = new Object( );  
Object.prototype.isPrototypeOf(o);      // true
```

# Массив



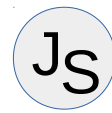
## Конструктор

```
var a = new Array();  
var a = new Array(1,7,15,'элемент',true);  
var a = new Array(10);
```

## Литерал

```
var a = [1,7,15,'элемент',true];  
var b = [[1,{x:1, y:2}], [2, {x:3, y:4}]];
```

## Элементы массива



### Доступ к элементам

```
value = a[9];  
a[3] = 3.14;  
a[i+1] = 'привет';  
a[a[i]] = a[0];  
me['salary'] *= 1.5;
```

### Добавление элементов

```
a[1001] = 9;
```

```
var fruits = ['Apple', 'Orange', 'Pineapple'];  
fruits.push('Kiwi');
```

```
var c = new Point(-1,1);  
c[0] = 'это новый элемент';
```

## Длина массива



```
var a = new Array( );    // a.length == 0
a = new Array(10);       // a.length == 10
a = new Array(1,2,3);    // a.length == 3
a = [4, 5];              // a.length == 2
a[5] = -1;               // a.length == 6
a[49] = 0;               // a.length == 50
```

```
for(var i = 0; i < fruits.length; i++){
    if (fruits[i] != undefined){
        alert(fruits[i]);
    }
}
```

```
a.length = 2;
a.toString();           // 4,5
```

## Методы массива



### join

```
[1,2,3].join();           // '1,2,3'.split(',')  
[1,2,3].join('*');       // '1*2*3'
```

### reverse

```
[1,2,3].reverse();       // [3,2,1]
```

### sort

```
['Kiwi','Apple','Orange'].sort();  
                        // ['Apple','Kiwi','Orange']  
[33, 4, 1111, 222].sort() // [1111,222,33,4]  
a.sort( function(a,b){return a-b;} );
```

### concat

```
[1,2,3].concat(4,5,6);  
[1,2,3].concat([4,5],[6]);
```

## Методы массива



### slice, splice

```
a=[1,2,3,4,5];  
a.slice(0,3);           // [1,2,3]  
a.splice(1,2,11,12);    // [2,3,4]; a:[1,11,12,4,5]
```

### push, pop

```
a.pop();                // 5 ; a:[1,11,12,4]  
a.push(7);              // 5 ; a:[1,11,12,4,7]
```

### shift, unshift

```
a.shift();              // 1 ; a:[11,12,4,7]  
a.unshift(51,52);       // 5 ; a:[51,52,11,12,4,7]
```

### toString, toLocaleString

```
["a", "b", "c"].toString( )    // 'a,b,c'  
[1, [2,'c']].toString( )       // '1,2,c'
```