

Классы и прототипы



- Конструкторы
- Прототип и наследование свойств
- Эмуляция классов JavaScript
- Общие методы
- Наследование классов

Конструктор



```
function Rectangle(w, h) {  
    this.width = w;  
    this.height = h;  
}  
  
var rect1 = new Rectangle(2, 4);  
var rect2 = new Rectangle(8.5, 11);
```

Прототип



```
var pi = 3.1415926;
function c_perimeter() {return 2*pi*this.radius;}
function c_setRadius(r) {this.radius=r;}

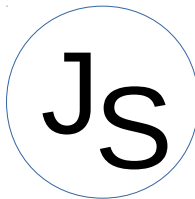
function Circle(r) {

    this.radius = r;

    this.perimeter = c_perimeter;
    this.setRadius = c_setRadius;
}

var r = new Circle(3);
r.setRadius(10);
var p = r.perimeter();
```

Прототип



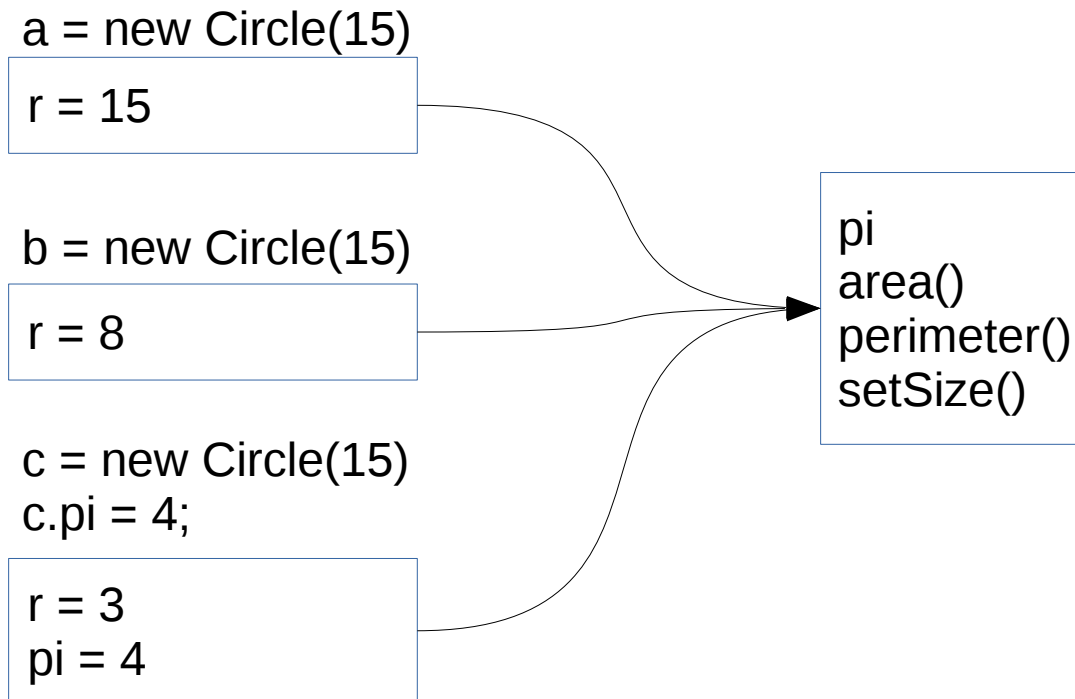
```
function Circle(r) {  
    this.radius = r;  
}
```

```
new Circle(0); //создать прототип  
Circle.prototype.pi = 3.1415926;  
Circle.prototype.perimeter = () => 2*pi*this.radius;  
Circle.prototype.setRadius = (r) => {this.radius=r;}
```

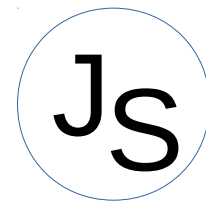
```
var r = new Circle(3);  
r.setRadius(10);  
var p = r.perimeter();
```

```
String.prototype.endsWith = function(c) {  
    return (c == this.charAt(this.length-1))  
}  
'Привет, мир'.endsWith('p'); //true
```

Наследование свойств



Эмуляция классов



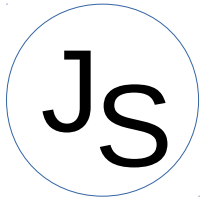
Классов нет, эмулируются конструкторами и прототипами.

Класс и экземпляр (instance).

```
c = new Circle();
```

свойство экземпляра	c.r	прототип
метод экземпляра	c.area()	прототип
свойство класса	Circle.PI	класс
метод класса	Circle.max (c1, c2)	класс

Наследование классов



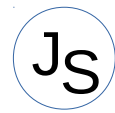
```
function Disk(r) {  
    this.radius = r;  
}  
  
Disk.prototype = new Circle(0);  
  
Disk.prototype.area = function() {  
    return Circle.PI * this.r * this.r;  
}  
  
Disk.prototype.constructor = Disk;  
                                     // ВОССТАНОВИЛИ
```

Классы и прототипы



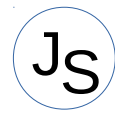
- Конструкторы
- Прототип и наследование свойств
- Эмуляция классов JavaScript
- Общие методы
- Наследование классов

Конструктор



```
function Rectangle(w, h){  
    this.width = w;  
    this.height = h;  
}  
  
var rect1 = new Rectangle(2, 4);  
var rect2 = new Rectangle(8.5, 11);
```

Прототип



```
var pi = 3.1415926;
function c_perimeter(){return 2*pi*this.radius;}
function c_setRadius(r){this.radius=r;}

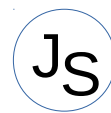
function Circle(r){

    this.radius = r;

    this.perimeter = c_perimeter;
    this.setRadius = c_setRadius;
}

var r = new Circle(3);
r.setRadius(10);
var p = r.perimeter();
```

Прототип



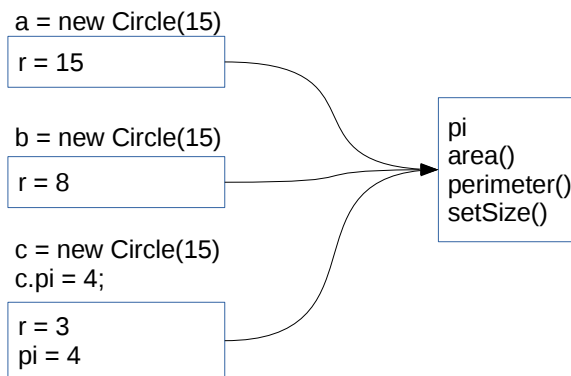
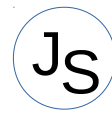
```
function Circle(r){
    this.radius = r;
}

new Circle(0); //создать прототип
Circle.prototype.pi = 3.1415926;
Circle.prototype.perimeter = () => 2*pi*this.radius;
Circle.prototype.setRadius = (r) => {this.radius=r;}

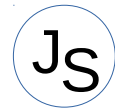
var r = new Circle(3);
r.setRadius(10);
var p = r.perimeter();
```

```
String.prototype.endsWith = function(c) {
    return (c == this.charAt(this.length-1))
}
'Привет, мир'.endsWith('p'); //true
```

Наследование свойств



Эмуляция классов



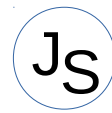
Классов нет, эмулируются конструкторами и прототипами.

Класс и экземпляр (instance).

```
c = new Circle();
```

свойство экземпляра	c.r	прототип
метод экземпляра	c.area()	прототип
свойство класса	Circle.PI	класс
метод класса	Circle.max (c1, c2)	класс

Наследование классов



```
function Disk(r) {  
    this.radius = r;  
}  
  
Disk.prototype = new Circle(0);  
  
Disk.prototype.area = function() {  
    return Circle.PI * this.r * this.r;  
}  
  
Disk.prototype.constructor = Disk;  
// ВОССТАНОВИЛИ
```