js继承

1.使用构造函数实现

```
function Parent1() {
      this.name = 'parent1';
};
function Child1() {
        Parent1.call(this);// apply
        this.type='child1';
};
```

2.借助原型链实现

```
function Parent2() {
      this.name = 'parent2';
      this.play = [1,2,3];
};
function Child2() {
      this.type='child2';
};
```

3.组合方式

```
function Parent3() {
     this.name = 'parent3';
     this.play = [1,2,3];
};
function Child3() {
    Parent3.call(this);//因为创建一个子类的实例的时候，父类的构造函数执行了两次。
    this.type = 'child3';
};
Child3.prototype = new Parent3();//Parent3构造函数执行了两次
```

4.组合继承优化1

```
function Parent4() {
      this.name = 'parent4';
```

```
      this.play = [1,2,3];
};
function Child4() {
      Parent4.call(this);
      this.type = 'child4';
};
Parent4.prototype.go = function() {console.log('66')};
Child4.prototype = Parent4.prototype;
Child4.prototype.constructor = Child4;
```

5.组合集成优化2

```
function Parent5() {
     this.name = 'parent5';
     this.play = [1,2,3];
};
function Child5() {
     Parent5.call(this);
     this.type = 'child5';
};
Parent5.prototype.go = function() {console.log('66')};
Child5.prototype = Object.create(Parent5.prototype);
Child5.prototype.constructor = Child5; //*
```