

<https://zhuanlan.zhihu.com/p/23987456>

对象里面要么是属性要么是函数;

```

1  var 士兵 = {
2      ID: 1, // 用于区分每个士兵
3      兵种:"美国大兵",
4      攻击力:5,
5      生命值:42,
6      行走:function(){ /*走俩步的代码*/},
7      奔跑:function(){ /*狂奔的代码*/ },
8      死亡:function(){ /*Go die*/ },
9      攻击:function(){ /*糊他熊脸*/ },
10     防御:function(){ /*护脸*/ }
11 }

```

属性

函数

new做了什么:

```

1
2
3  function create士兵(id){
4      // var temp = {}
5      // this = temp
6      // this.__proto__ = create士兵.prototype
7      this.ID = id, // ID 不能重复
8      this.生命值 = 42
9      /*实际工作中不要这样写, 因为 __proto__ 不是标准属性*/
10     // return this
11 }
12 create士兵.prototype = {
13     兵种:"美国大兵",
14     攻击力:5,
15     行走:function(){ /*走俩步的代码*/},
16     奔跑:function(){ /*狂奔的代码*/ },
17     死亡:function(){ /*Go die*/ },
18     攻击:function(){ /*糊他熊脸*/ },
19     防御:function(){ /*护脸*/ }
20 }
21

```

只要你写了个:new 函数;  
这里灰色的就是js之父帮你写上的  
包括return

```

function create士兵(id){
  // var temp = {}
  // this = temp
  // this.__proto__ = create士兵.prototype
  { this.ID = id, // ID 不能重复
    this.生命值 = 42 // 自有属性
  }
  /*实际工作中不要这样写, 因为 __proto__ 不是标准属性*/
  // return this
}

create士兵.prototype = {
  // 共有属性
  兵种:"美国大兵",
  攻击力:5,
  行走:function(){ /*走俩步的代码*/},
  奔跑:function(){ /*狂奔的代码*/},
  死亡:function(){ /*Go die*/},
  攻击:function(){ /*糊他熊脸*/},
  防御:function(){ /*护脸*/}
}

```

自有属性

共有属性

new + 函数:----->写了一个函数,这个函数指定了两个属性(自有属性和共有属性)

```

function 士兵(id){
  // var temp = {} // 1
  // this = temp // 2
  // 士兵.prototype = { constructor: 士兵 }
  // this.__proto__ = 士兵.prototype // 3
  this.ID = id, // ID 不能重复
  this.生命值 = 42 // 自有属性
  /*实际工作中不要这样写, 因为 __proto__ 不是标准属性*/
  // return this // 4
  // 语法糖
}

士兵.prototype = {
  // 共有属性
  兵种:"美国大兵",
  攻击力:5,
  行走:function(){ /*走俩步的代码*/},
  奔跑:function(){ /*狂奔的代码*/ },
  死亡:function(){ /*Go die*/ },
  攻击:function(){ /*糊他熊脸*/ },
  防御:function(){ /*护脸*/ }
}

```

这是默认的, 但是其实是被下面的2给替代了

2

```
function 士兵(id){
  // var temp = {} // 1
  // this = temp // 2
  // 士兵.prototype = { constructor: 士兵 }
  // this.__proto__ = 士兵.prototype // 3
  this.ID = id, // ID 不能重复
  this.生命值 = 42 // 自有属性
  /*实际工作中不要这样写，因为 __proto__ 不是标准属性*/
  // return this // 4
  // 语法糖
}
```

```
士兵.prototype.兵种 = "美国大兵"
士兵.prototype.攻击力 = 5
士兵.prototype = {
  // 共有属性
  constructor: 士兵,
  兵种: "美国大兵",
  攻击力: 5,
  行走: function(){ /*走俩步的代码*/ },
  奔跑: function(){ /*狂奔的代码*/ },
  死亡: function(){ /*Go die*/ },
  攻击: function(){ /*糊他熊脸*/ },
  防御: function(){ /*护脸*/ }
}
```

推荐这样写

## 总结

```
1  var object = new Object()
2
3  自有属性 空
4  object.__proto__ === Object.prototype
5
6  var array = new Array('a','b','c')
7
8  自有属性 0:'a' 1:'b' 2:'c',length:3
9  array.__proto__ === Array.prototype
10 Array.prototype.__proto__ === Object.prototype
11
12
13 var fn = new Function('x', 'y', 'return x+y')
14 自有属性 length:2, 不可见的函数体: 'return x+y'
15 fn.__proto__ === Function.prototype
16
17 Array is a function
18 Array = function(){...}
19 Array.__proto__ === Function.prototype
```