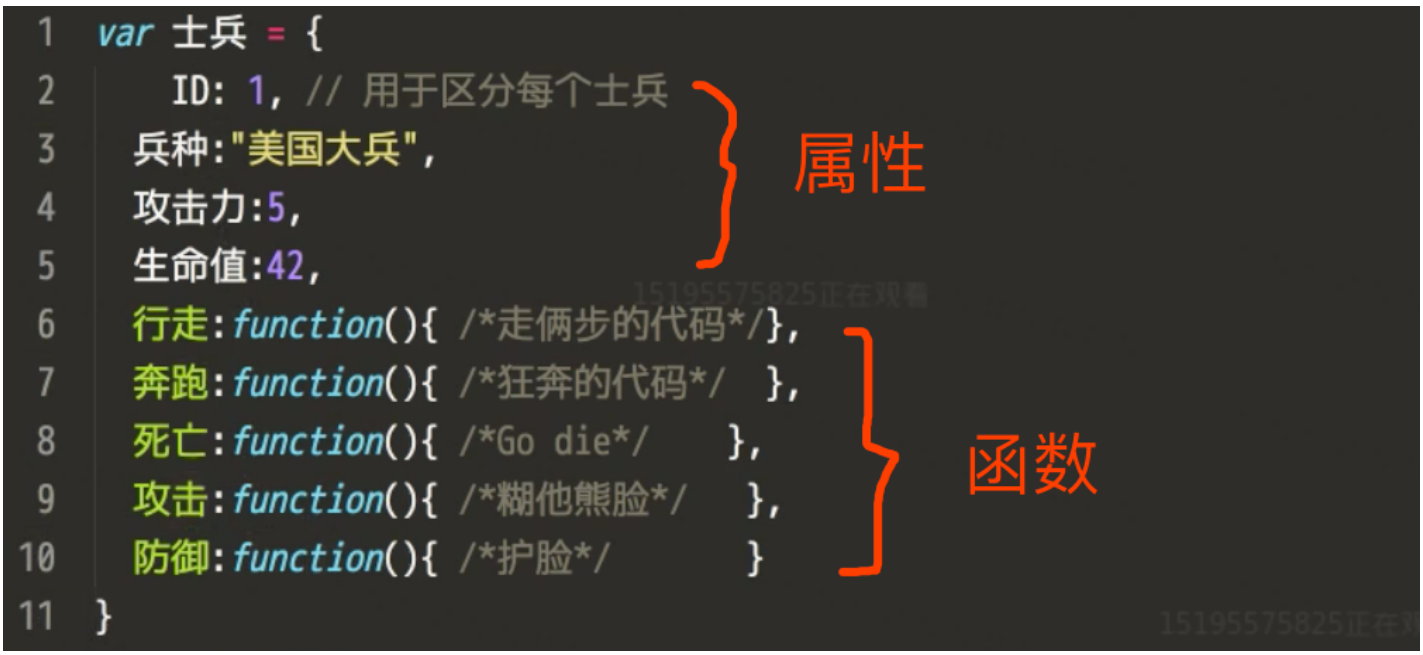
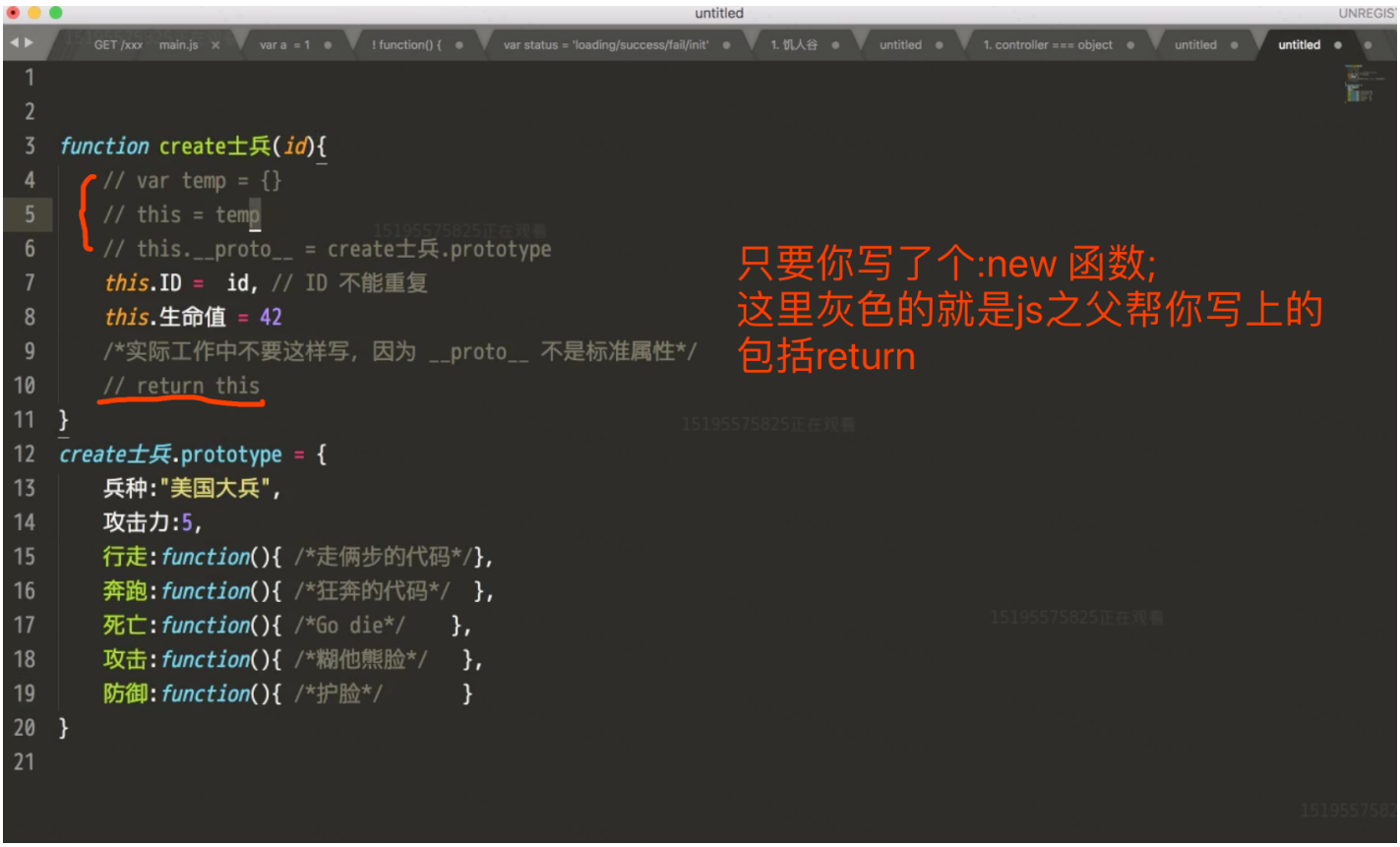
总结

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

对象里面要么是属性要么是函数;



new做了什么:



```
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

```
var object = new Object()

自有属性 空
object.__proto__ === Object.prototype

var array = new Array('a','b','c')

自有属性 0:'a' 1:'b' 2:'c',length:3
array.__proto__ === Array.prototype
Array.prototype.__proto__ === Object.prototype


var fn = new Function('x', 'y', 'return x+y')
自有属性 length:2, 不可见的函数体: 'return x+y'
fn.__proto__ === Function.prototype

Array is a function
Array = function(){...}
Array.__proto__ === Function.prototype
```