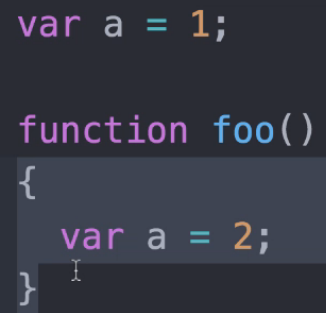
课时1 let命令

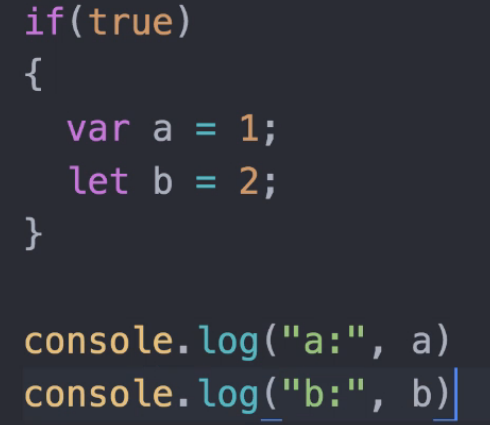
1 var的两种作用域：

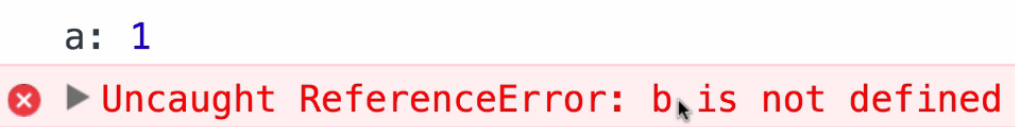
1全局

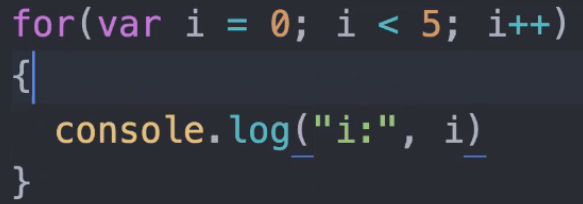
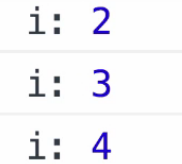
2 函数（内部）作用域

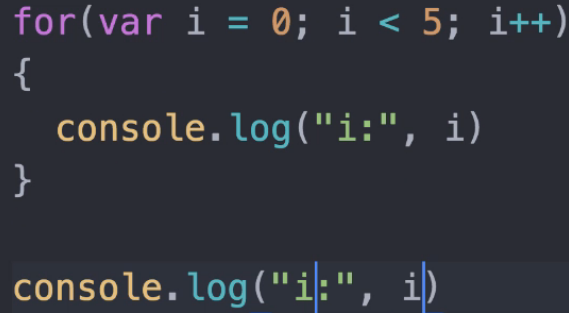
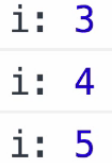


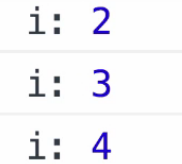
2 let只要遇到代码块就算一个域

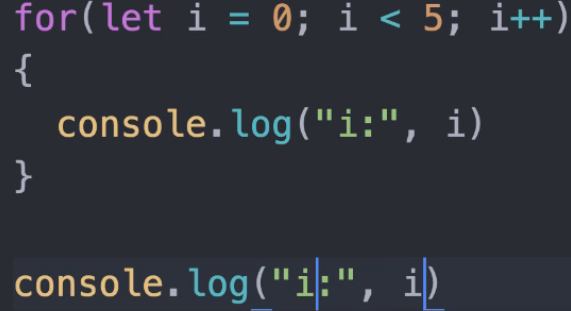
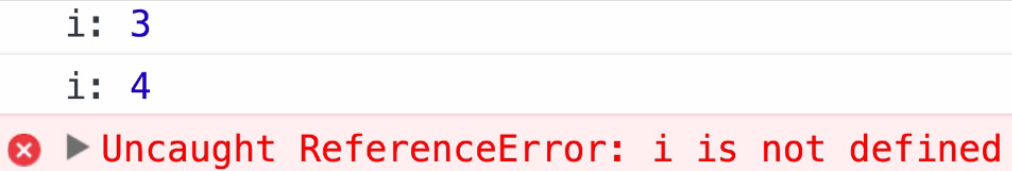




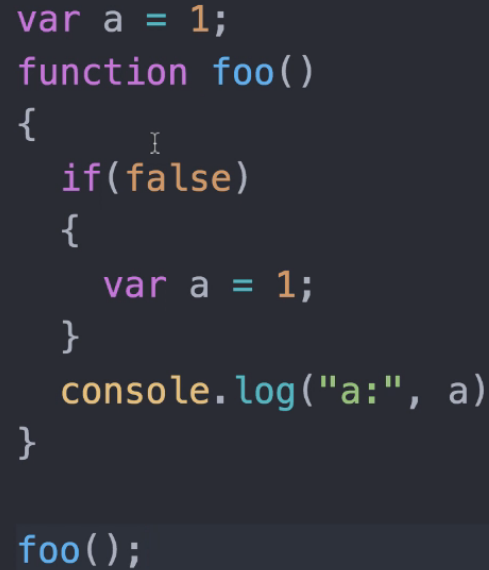
 从0-4

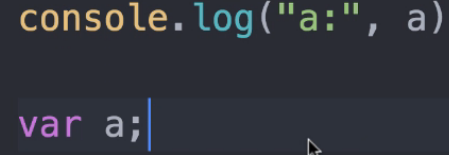
 从0-5

 从0-4

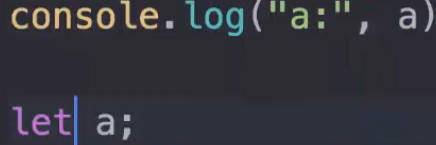
 

(是个数据类型而不是个错误，ES5为了防止错误发生迁就开发者，可以说是一个功能)

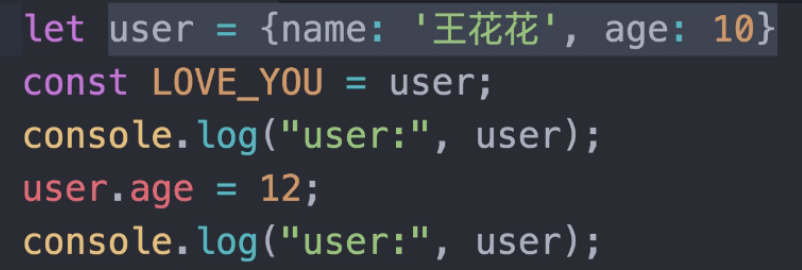




课时2 const命令

const用来定义常量，在程序运行期间常量值不会改变

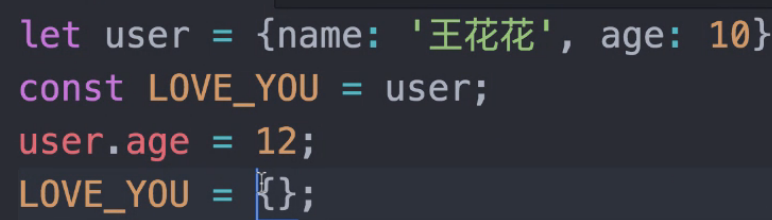
* 用const来定义的常量的名字通常要大写(方便一眼就看出这是个常量)，如：const PI=3.14 或者 SOA = ‘abc’
* const在声明常量时一定要同时赋值，不能只是声明而不赋值。
* const定义的常量的数据类型除了数值、字符串及布尔类型(true/false)以外，还可以是**对象{}，如 const LOVE = {}**

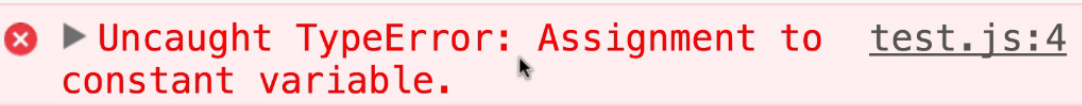






不变的是LOVE\_YOU指向user这个对象的指针，而user这个对象本身的值是可以变的。(const 常量指向内存中某个小格子的id/地址)

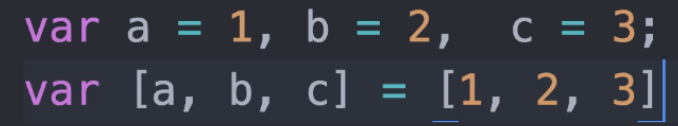


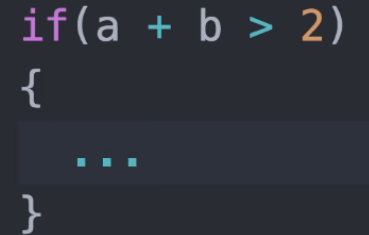


其他方面如作用域等的用法都和let相同，只能作用在自己的代码块里，是一个块级作用域

课时3 变量的解构赋值（数组）

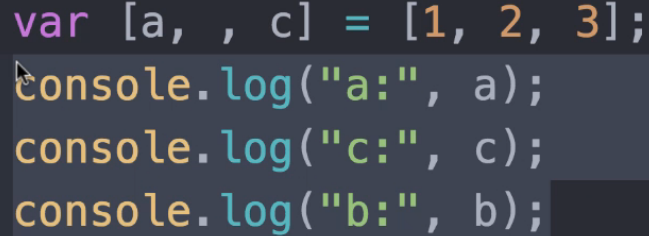
1 常规赋值

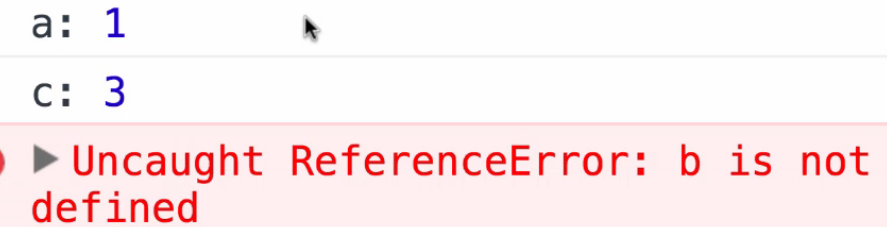






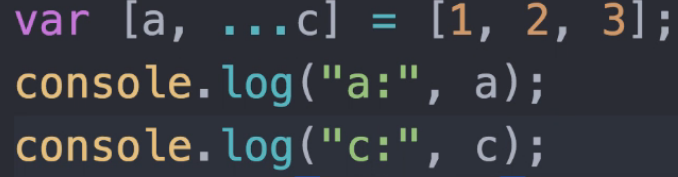
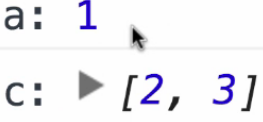
2 跳着赋值



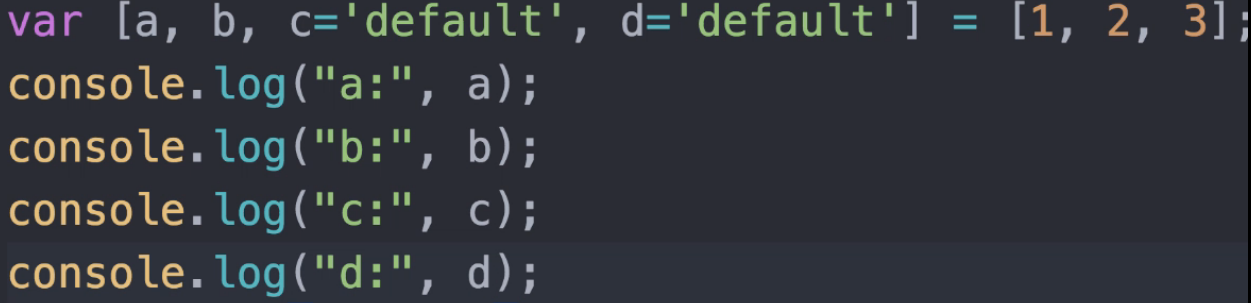


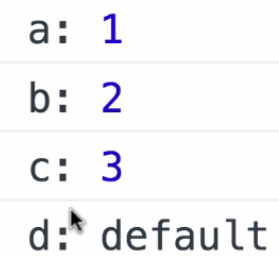
即可以跳着赋值

3 将第一个元素赋给a，其他的元素都赋值给c

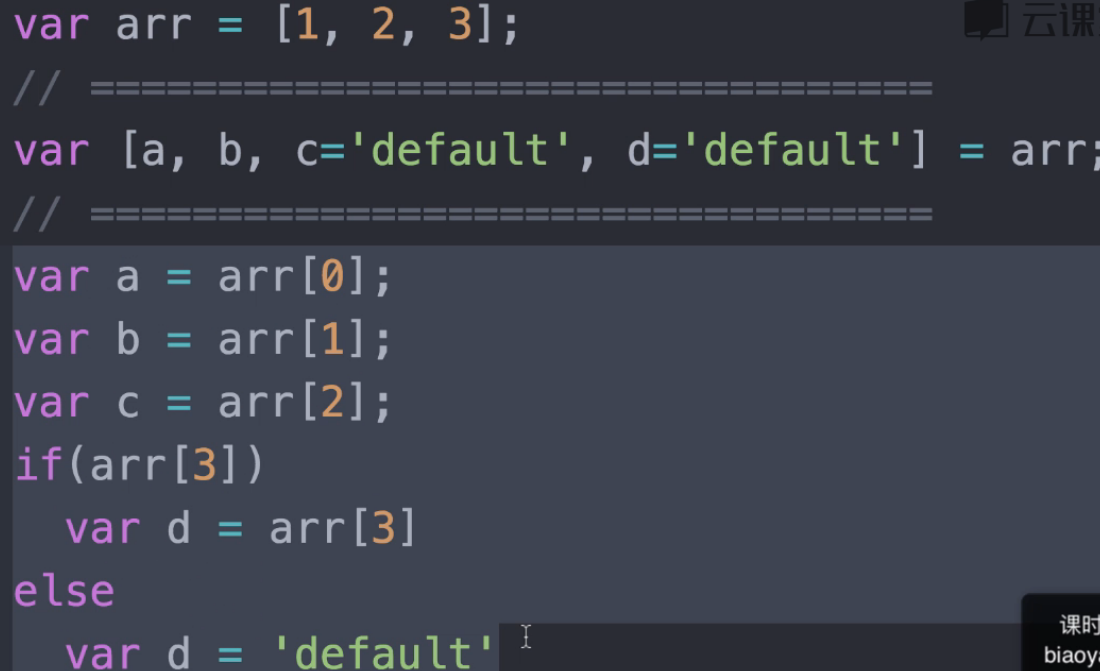
 

4 默认值default





5 ES6 VS ES6以前



6 不存在的值会指定为undefined

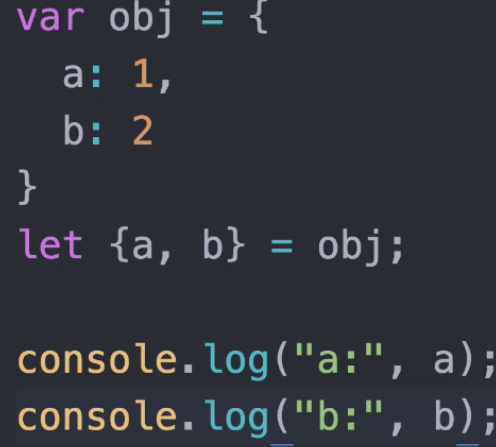
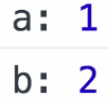


console.log(‘c:’,c )



课时4 变量的解构赋值（对象）

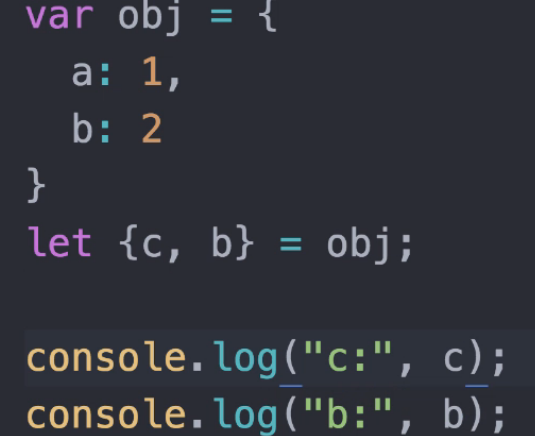
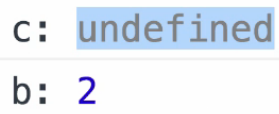
1 常规赋值

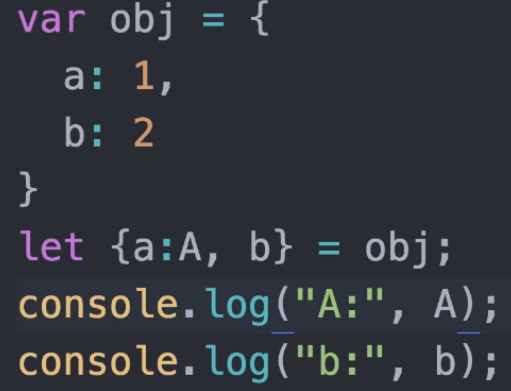
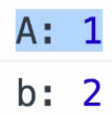
2 对象属性的两种寻址方式

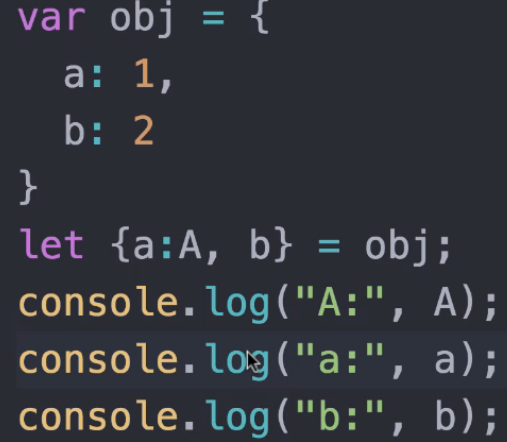
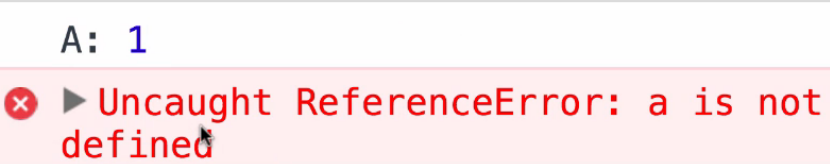
3 改变变量名a-c:

被赋值的变量名要与对象中的key的名字相同！不然就按下面的方法：

即先找到对象obj里的a，再将其命名为A并赋值给A

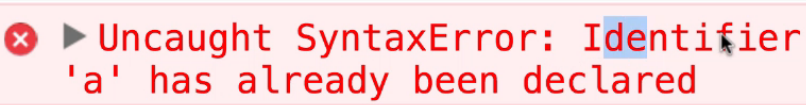
 

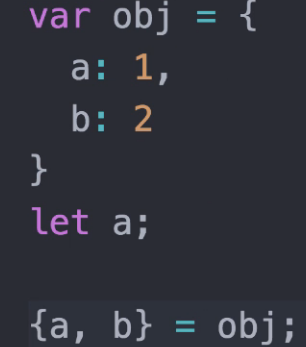
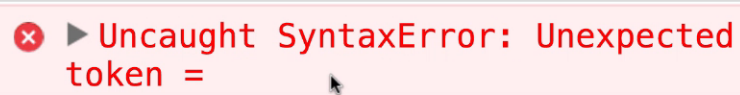
常规赋值其实相当于：



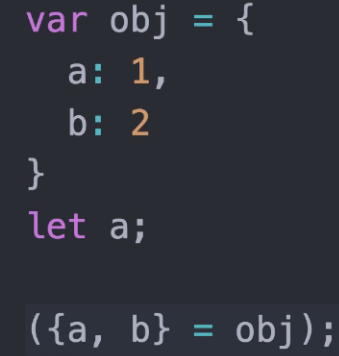
找到a然后命名成a，只不过省略了

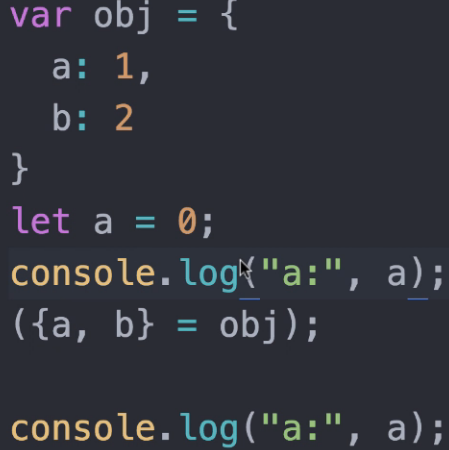
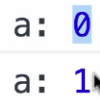
4 语法报错

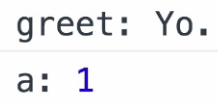
 

原因：花括号用在解构赋值时不能使用在一行的最前面，不然JS解析器会把它当成一个代码块。而事实上它不是个代码块，而是个结构语法。此时我们加一对圆括号()把这个结构赋值包起来就好了。即

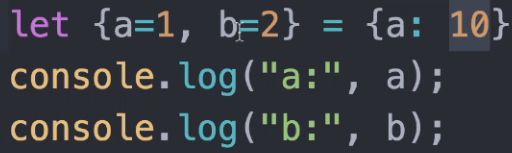
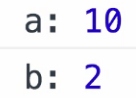
 这样控制台没有报错

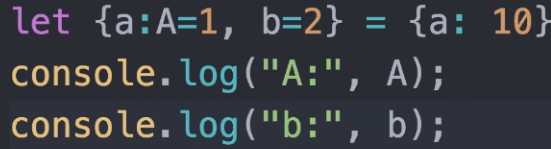
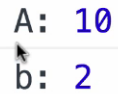
5 更复杂一点的对象解构

 (这种情况还是比较少的)

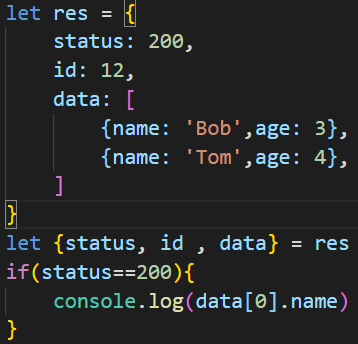
6设置变量默认值

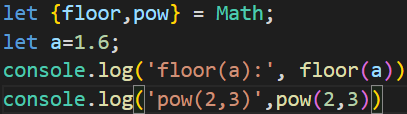
重命名之后再赋值

7 实践运用（解构服务器返回来的数据）

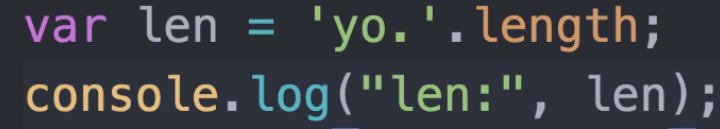
8解构方法

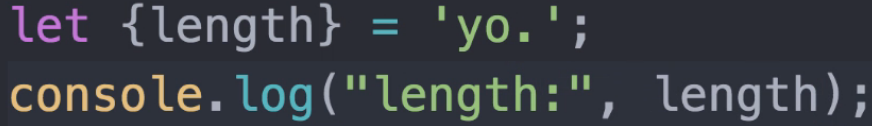
 

PS: floor是去除小数点的方法，pow是求幂函数的方法，第一个参数为底数，第二个参数为幂

课时5 变量的解构赋值（其他）

1 直接查看字符串的长度

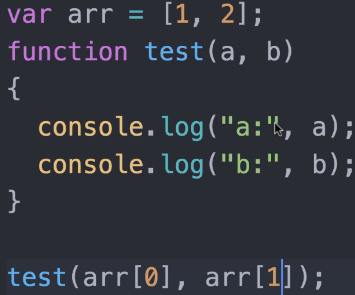
length是字符串的一个属性

2 解构字符串

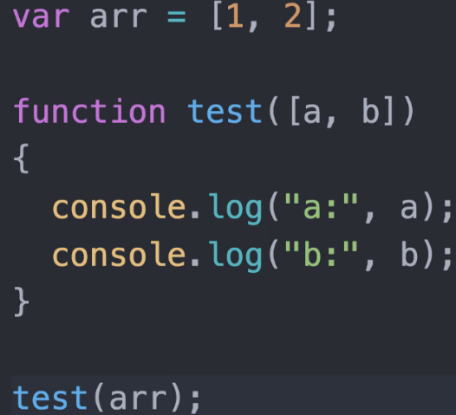
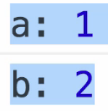
 

3 对函数的传参进行解构

原始方法：

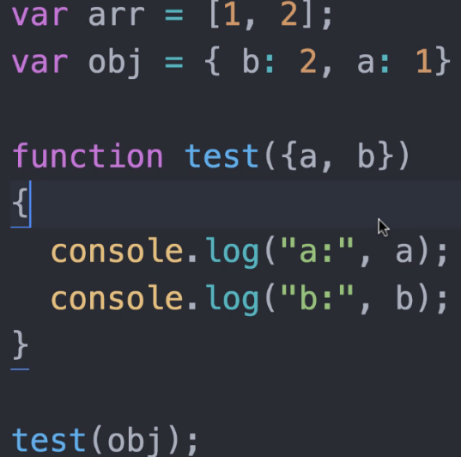
ES6：

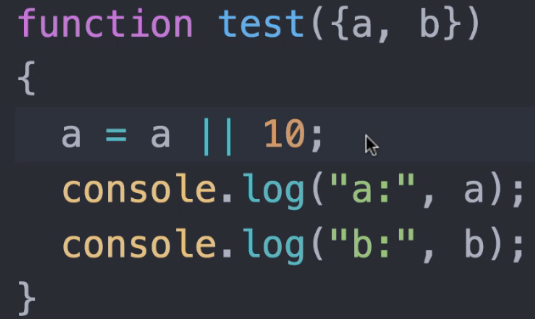
更好的方法：

解构对象，与变量的顺序无关

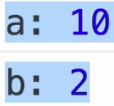
 

原始默认值赋值方法：



只有当传入的参数a为真时才将其赋值给左边声明的a，否则就将默认值10赋值给a

ES6简便方法：

课时6 新增字符串方法

1 测试一个字符串中是否包含另一个字符串

老方法: indexOf 用法：string.indexOf(substring),返回子字符串第一次出现的位置，如果没有匹配到则返回-1



（完全不等于-1）



ES6 includes方法：string.includes (substring)，返回布尔值（true/false）

2 判断字符串的开头字母:**startsWith**

3 判断字符串的结尾字母:**endsWith**

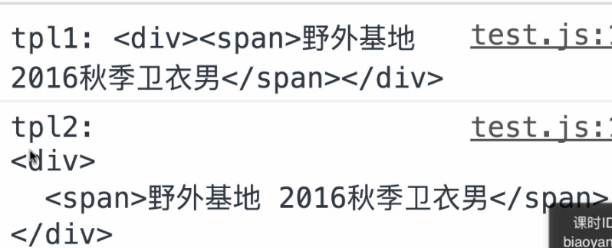
4 重复显示字符串:**repeat**

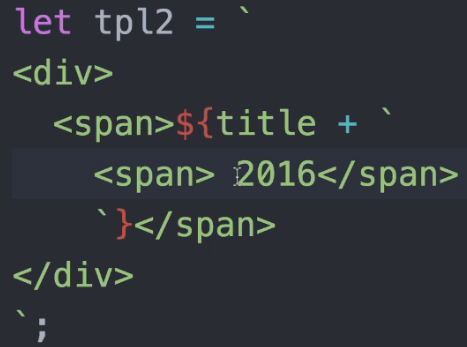
课时7 模板字符串

模板字符串（反引号``）中嵌入变量，需要将变量名写在${}中，如：

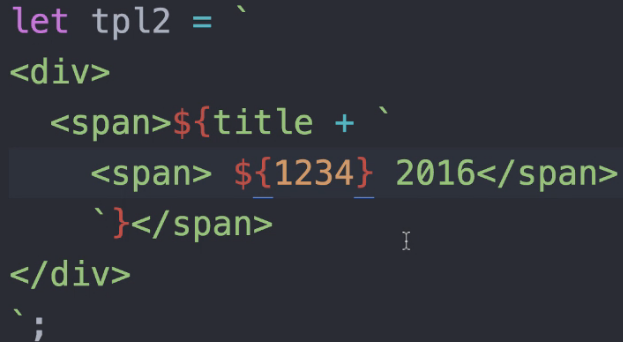
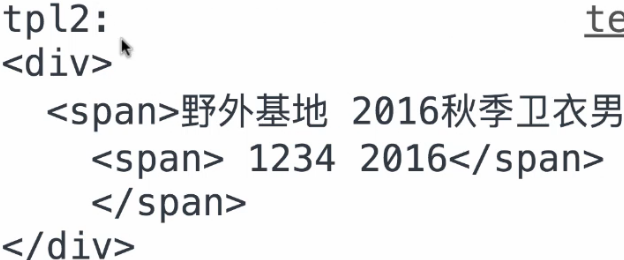
‘hello’+name+’welcome’= `hello${name}welcome`

用ES6的写法保留了换行等格式，非常方便，并且还可以在模板字符串中再嵌套模板字符串，如：

再跟一个变量：

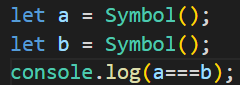
 

课时8 Symbol类型

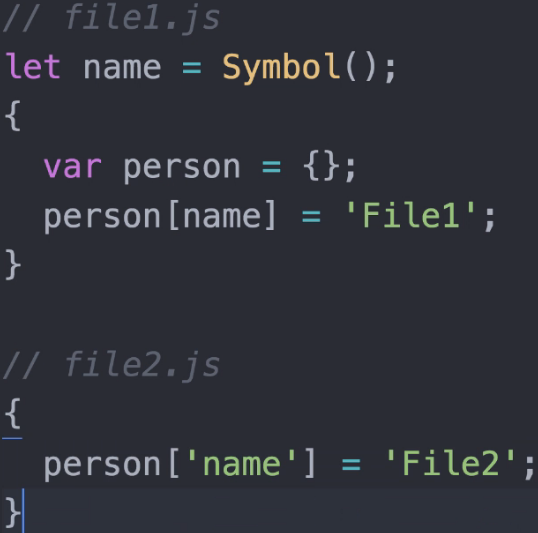
1 ES数据类型（七大类）



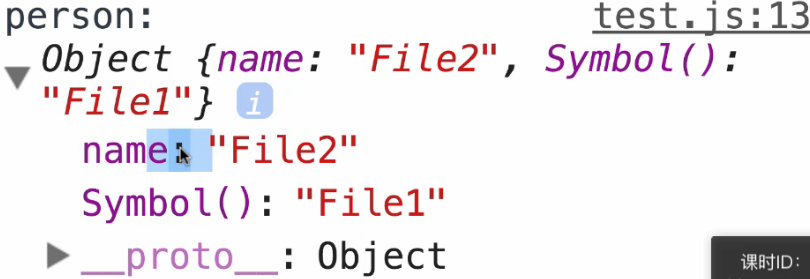
Symbol:每次创造一个Symbol,它的值都是不一样的

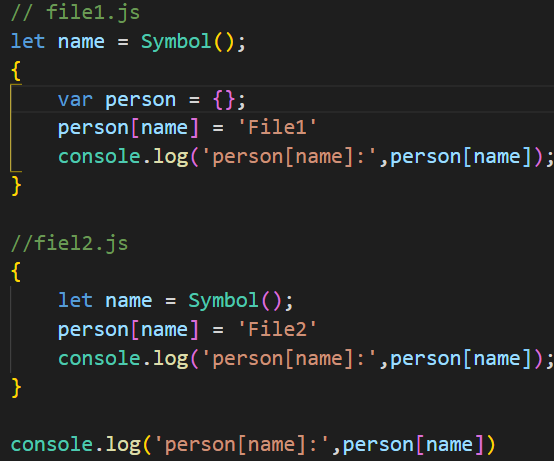
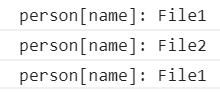
 

2 Symbol的用处：作为对象的属性名称，防止（下游对上游）对象的属性被重写

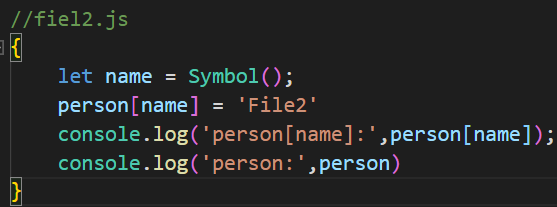
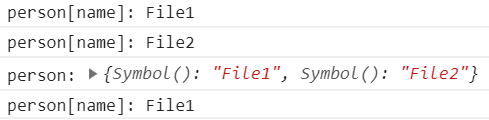






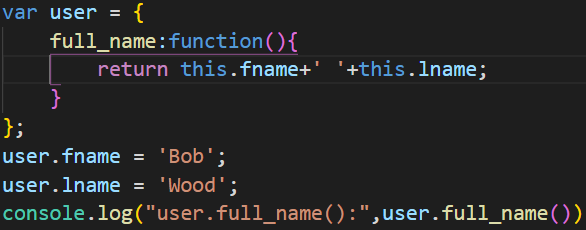
可以在局部中临时改变所引用的插件的变量，但不能在全局中改掉。

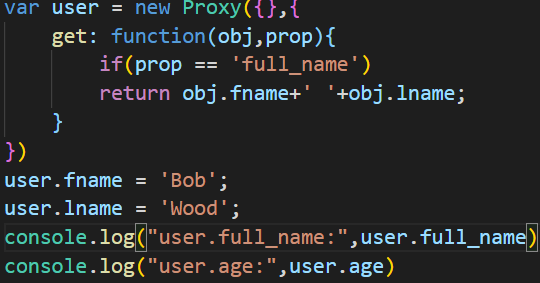
课时9 Proxy（代理）

用处：用于在语言层面上去操作一个对象

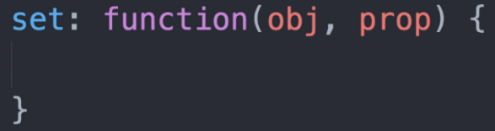
ES5：

ES6:

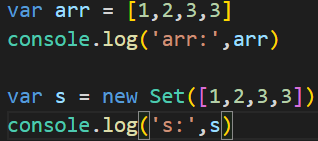
 

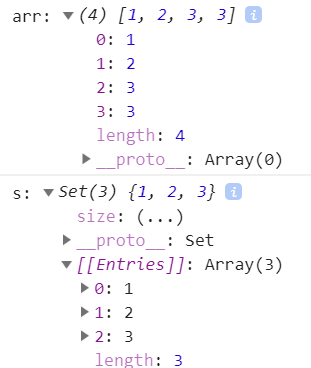
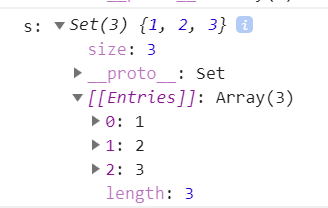
还有set



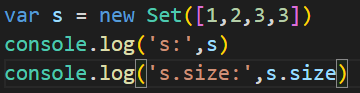
课时10 Set（新的数据结构）

1 基本概念：Set里面的值每一个都是唯一的

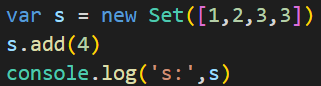
 

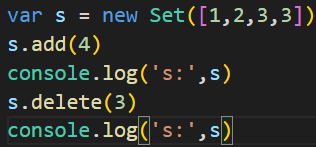
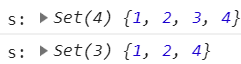
 

2 Set的属性：size长度，相当于数组的length属性

3 Set的几个常用方法：add(添加)、delete(删除)、has(是否有某个值)、clear(全部清空)

Axios 传参问题：

