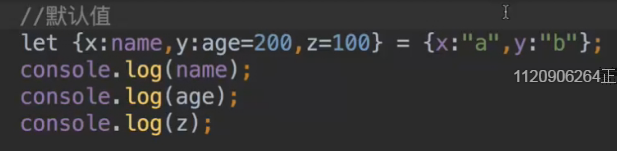
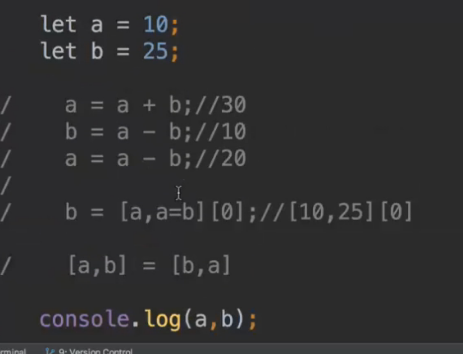
## ES6

1. 关于var、let、const的区别
   1. Var声明的变量会被挂载到window上，let不会；
   2. Var声明的变量存在变量提升，let没有变量提升；
   3. Let不允许重复声明；
   4. Let声明形成块级作用域，声明的变量只在当前的块级作用域有效；
   5. 暂时性死区，在let声明变量之前，不能使用这些变量，不然就会报错（引用错误，不声明就使用变量就会报错；暂时性死区的本质就是，只要一进入当前作用域，所要使用的变量就已经存在了，但是不可获取，只有等到声明变量的那一行代码出现，才可以获取和使用该变量。）
   6. Const可以声明一个常量，也可以声明一个对象；声明的对象只是对象所在的内存地址的指针没有变化，对象的值可以更改；object.freeze（）可以冻结对象，使其不变。
2. 解构赋值
   1. 默认值是undefined或者没有值的时候才能生效



* 1. 数组解构是根据索引来确定位置；对象解构是根据属性来确定；
  2. 小面试题：



* 1. “...”将剩余的值分配给变量

1. 箭头函数
   1. 箭头函数中的this指向定义时的this，而不是执行时的this；箭头函数this指向其父级作用域，不是调用时候的作用域（window）;（this指向的固定化，并不是因为箭头函数内部有绑定this的机制，实际原因是箭头函数根本没有自己的this，导致内部的this就是外层代码块的this。正是因为它没有this，所以也就不能用作构造函数。）
   2. 箭头函数不能作为构造函数，不能使用new关键字构建
   3. 箭头函数没有arguements（类数组--------->实参列表）
   4. 箭头函数不能使用call, apply, bind改变this指向
   5. 箭头函数没有原型属性（prototype）
   6. 不适用的场合：
      1. 定义对象的方法是不能使用this，因为对象不构成单独的作用域，所有对象里面的this会指向全局；
      2. 需要动态this的时候，也不应该使用箭头函数；实现数据监听时需要让this指向被点击时候的对象，但是箭头函数中的this会指向全局
2. Promise
   1. Promise是对异步状态管理的
   2. 特点
      1. 对象的状态不受外界影响。Promise对象代表一个异步操作，有三种状态：pending（进行中）、fulfilled（已成功）和rejected（已失效）。只有异步操作的结果，可以决定当前是哪一种状态，任何其他操作都无法改变这个状态。这也是promise这个名字的又来，他的英语意思就是“承诺”，表示其他手段无法改变。
      2. 一旦状态改变，就不会再变，任何时候都可以得到这个结果。Promise对象的状态改变，只有两种可能：从pending变为fulfilled和从pending变为rejected。只有这两种情况发生，状态就凝固了，不会再变了，会一直保持这个结果，这时就成为resolved已成型。如果改变已经发生了，你再对promise对象添加回调函数，也会立即得到这个结果。这与事件完全不同事件的特点是，如果你错过了它，再去监听，是得不到结果的。
   3. Promise.all（）接受一个数组，只有当promise的对象全部成功的时候才算成功，若有一个失败，promise就是打印失败。
   4. Promise.race（）接受一个数组，只要有一个对象成功就表示成功了。
3. Generator
   1. 需要yield表达式，返回值是yield的值
   2. 需要手动调用，通过next一步一步调用
4. 扩展运算符（接收一个数组）|| arguments是一个类数组，不是一个真正的数组
   1. 不确定参数
   2. 作为数组的一部分 let arr3 = [...1,2,3]
   3. 数组复制 let arr2 = [...arr1]。这里需要注意，arr2只能复制一维数组，不能复制二维数组，二维数组会随着arr1发生改变
   4. 合并数组 原生方法：let arr3 = arr.concat(arr2) ES6方法：let arr3 = [...arr, ...arr2]
   5. 合并对象
   6. 解构
   7. Rest 剩余的，也是用。。。，只接受剩余部分的参数，只能放在所有参数最后面，不能在中间。
5. 字符串扩展
   1. 模板字符串（反引号）： ${} 支持标签、换行 简单地数学运算
   2. Includes()（判断是否有字符串，有的话返回true，没有返回false） VS indexOf()（返回字符的索引，若没有值就返回-1）
   3. startsWith()（是否以传入的子串开始，返回布尔值） endsWith()（是否以传入的子串结束，返回布尔值）
   4. padStart（5， ‘abc’）：5是规定的长度，若字符串的长度不足5，就用abc这种字符串从前面来补齐 padEnd（5，‘abc’）：从后面补齐
   5. Repeat（5）重新5次
6. 数值扩展
   1. 二进制0B 八进制0O，用parseInt可以将其转换
   2. 是否为数字：Number.isFinite(5)
   3. Number.isNaN(NaN)
   4. Number.isInteger(5)
   5. Number.parseFloat(5)
   6. Number.parseInt(5.5)
   7. Number.isSafeInteger()
      1. Number.MAX\_SAFE\_INTEGER
      2. Number.MIN\_SAFE\_INTEGER

一些方法一直到Number对象上面，行为完全保持不变。这样做的目的，是逐步减少全局性方法，使得语言逐步模块化。

1. Class
   1. 类---------对象
   2. Constructor（）
   3. 类和模块的内部，默认就是严格模式，所以不需要使用use strict指定运行模式。只要你的代码写在类或模块之中，就只有严格模式可用。考虑到未来所有的代码，其实都是运行在模块之中，所以ES6实际上把整个语言升级到了严格模式。
   4. Extends
   5. 在类的实例上调用方法，实际上是调用原型上的方法。
2. 关于js的继承
   1. 由继承的方法创造出来的对象，若对象自身有属性，调用对象的时候，this是指向自身的属性

1591948291(1)