### **let、const**

1. let 定义的变量不会被变量提升，const 定义的常量不能被修改，let 和 const 都是块级作用域
2. const 定义的常量不能被修改，如果const定义的是对象，对象中的内容可以变
3. var关键字声明变量。无论声明在何处，都会被视为声明在函数的最顶部

**symbol**

**--新增的数据类型**

### **import、export**

import导入模块、export导出模块

1. exports default一个文件中有且只有一个默认
2. exports一个文件中有可以有多个
3. import('文件路径')按需加载模块

### **class、extends、super**

类、继承、父级

### **箭头函数**

写法：=>  
箭头函数时，函数体内的this对象，就是定义时所在的对象

### **模板字符串**

写法：字符串${变量}

let str1 = 'bbb'

str2 = `ccc${str1}`

console.log(str2)

### **includes方法**

1.判断字符串

let str='sjfddshgljd';

console.log(str.includes('d'));//true

2.判断数组

let arr = ['aaa', 'bbb', 'ccc'];console.log(arr.includes('aaa'))

### **函数默认参数**

function foo(name='yayaya'){

console.log(name)}// 如果方法不传参，默认输出yayayafoo()

### **展开运算符**

写法：...obj

let obj1 = {a: 3, b: 4}let obj2 = {

...obj1,

c: 5

### **Object.assign()**

使用方法：Object.assign({}, objA, objB)  
参数1为目标对象，一般传入{}

1. Object.assign()是浅拷贝，如果源对象某个属性的值是对象，那么目标对象拷贝得到的是这个对象的引用

let obj1 = {a: 3, b: 4}let obj2 = {c: 3, d: 4}let obj3 = {e: 3, f: 4}const obj = Object.assign({}, obj1, obj2, obj3)console.log(obj) // { a: 3, b: 4, c: 3, d: 4, e: 3, f: 4 }

### **Promise**

用同步的方式去写异步代码,避免回调地狱

function fn() {

new Promise((resolve,reject) =>{

if(true){

resolve('str')

}

}).then(res => {

// somecoding

}).catch({})

**Generator**

Generator 是 ES6 中新增的语法，和 Promise 一样，都可以用来异步编程

// 使用 \* 表示这是一个 Generator 函数

// 内部可以通过 yield 暂停代码

// 通过调用 next 恢复执行

function\* test() {

let a = 1 + 2;

yield 2;

yield 3;

}

let b = test();

console.log(b.next()); // > { value: 2, done: false }

console.log(b.next()); // > { value: 3, done: false }

console.log(b.next()); // > { value: undefined, done: true }

**es7**