

第十一天

变量提升（声）【运行前提前声明的意思】

全局

在代码执行前，把其中带var 和 function关键字的先提前声明；var**只声明不定义**默认值是undefined；function**声明加定义**；
但是如果在条件判断语句中，带var 的不变，还是只声明不定义，但是function变成了只声明不定义；（低版本浏览器不变，还是声明+定义）

局部(函数内)

在函数执行前，也同上先进行变量提升
注意：跟全局比函数内执行前在变量提升之前多了一个形参赋值，然后再变量提升；

var 和 let 的区别

- var 可以重复声明 但是 let不可以
- var 可以变量提升 但是 let 不可以
 - 注意：let 存在暂时性死区，（如果有let的话，不管是全局还是局部作用域，调用变量只能在let后边，不然就报错）
- var 可以成为window属性，但是 let 不会
- let 可以识别块级作用域；但是var 不识别
- const：声明的只是一个常量，不能被重新赋值；

作用域

全局作用域

定义：整个window；（只有一个）

- 全局变量：在全局作用域声明的变量

局部作用域/私有作用域

定义：函数内的作用范围（可以有多个函数）

- 局部/私有变量：在局部作用域声明的变量，（函数内的形参也属于局部变量）

块级作用域（ES6）

定义：（ES新加的）作用于语句中{大括号}包裹的区域被称为**块** 例如 `for{}、if{}、`

只有let 和 const 会识别块级作用域，var不会识别

作用域链

定义：变量的查找机制

过程：先在本级私有作用域查找某个变量，若没有，则去**上级作用域**查找。。。。一直查找
到window，若window也没有该变量，则就报错；

- **上级作用域**只跟在哪儿定义的有关系；跟在哪儿调用的没关系；

面试题

```
var a = {  
  x: 1  
};  
var b = a;  
// 点的优先级 比 赋值 要高  
b.c = a = {  
  y: 1  
};  
console.log(a.c,b.c);
```

The diagram illustrates the variable lookup process for the code. It shows two objects: one with 'x:1' and 'c:null' (pointed to by 'a') and another with 'y:1' (pointed to by 'b'). Red arrows show the lookup path for 'a.c' (a -> c -> null) and 'b.c' (b -> y:1). A red 'X' is over the 'a' pointer to the first object.

```
console.log(num, str);  
var num = 18;  
var str = "lily";  
function fn2() {  
  console.log(str, num);  
  num = 19;  
  str = "candy";  
  var num = 14;  
  console.log(str, num);  
}  
fn2();  
console.log(str, num);
```

The diagram illustrates the variable lookup process for the code. It shows three scopes: 'window' (num=18, str='lily', fn2()), 'fn2()' (num=19, str='candy', num=14), and 'fn2()' (num=14). Red arrows show the lookup path for 'num' and 'str' in the 'fn2()' scope, highlighting the 'candy' value for 'str' and the '14' value for 'num'.

小总结

1. 全局作用域

执行栈，在script标签内第一层js代码

(1) 如果当前script中的全局没有某个变量，这个时候还会向上面的script中去查找（只会向上找，默认不会向下找）有就输出，否则报错。

(2) 如果有多个script标签，上面的script中的代码报错是不会影响，下面script标签内的代码执行的。

(3) 全局的this为window

(4) 从作用域链的角度来说，最终会找到window下有没有某个属性(var的情况)

(5) 多个script标签如果都用了let，那么同样走let特性(不能有重名变量)

(6) 使用var的时候等同于在window下注册了一个属性，并且在没赋值之前为undefined

不过在chrome|FF下 在变量的上方打印window的时候会有属性值结果，要注意的是，显示出来的是骗人的（跟undefined走）

(7) 函数默认也是挂在window身上的

(8) 变量必须加var或者let来声明，不然在变量没赋值之前访问这个变量就报错

2. 私有作用域

在函数执行栈中运行代码，函数中的变量和参数会默认处理在函数内部，不会被外界所干扰。

如果函数内的计算或者逻辑处理需要被外界所接受一般使用return。

防止全局污染（封闭空间，教科书上说它就是闭包，但是，这样不一定是我们眼中的闭包）

3. 块级作用域

{ }

`let`、`const` 识别块级作用域

`var`不识别

要小心`function() {}`

4. 作用域链

1. 当前域没有会去它的上级域查找，知道`window`结束，`window`都没有就报

错