# 常量

**ES5中常量写法**

***Object***.defineProperty(***window***,**"PI2"**,{  
 **value**:3.1415926,  
 **writable**:**false**})  
***console***.log(***window***.**PI2**)

**ES6中常量写法**

**const *PI*** = 3.1415926

# 作用域

ES5:

**var *callbacks*** = [];  
**for** (**var *i*** = 0; ***i*** < 2; ***i***++) {  
 ***callbacks***[***i***] = **function** () {  
 **return *i*** \* 2;  
 }  
}  
***console***.table([  
 ***callbacks***[0](),  
 ***callbacks***[1](),  
 ***callbacks***[2]()   
])

结果是6,6,6

ES6

将var改成let后，是正常的0,2,4

{  
 **function** *foo*() {  
 **return** 1  
 }  
 ***console***.log(**"foo()===1"**,*foo*() === 1)  
 {  
 **function** *foo*() {  
 **return** 2  
 }  
 ***console***.log(**"foo()===2"**,*foo*() === 2)  
 }  
}

结果都是true

# 箭头函数

ES3,ES5

**var *evens*** = [1,2,3,4]  
**var *odds*** = ***evens***.map(**function** (v) {  
 **return** v + 1  
})  
***console***.log(***odds***)

ES6

**var *odds*** = ***evens***.map((v) => {  
 **return** v + 1  
})  
***console***.log(***odds***)

结果一样

箭头函数和普通function的this绑定不同

ES5:

**var** *factory* = **function** () {  
 **this**.**a** = **'a'**;  
 **this**.**b** = **'b'**;  
 **this**.**c** = {  
 **a**:**'a+'**,  
 b: **function** () {  
 **return this**.**a** }  
 }  
}  
***console***.log(**new** *factory*().**c**.b())

结果是a+

ES6:

**var** *factory* = **function** () {  
 **this**.**a** = **'a'**;  
 **this**.**b** = **'b'**;  
 **this**.**c** = {  
 **a**:**'a+'**,  
 b: () => {  
 **return this**.**a** }  
 }  
}  
***console***.log(**new** *factory*().**c**.b())

结果是a

# 默认参数

ES5：

**function** *f*(x,y) {  
 **if**(y == **undefined**){  
 y = 1;  
 }  
 *///*}

ES6:

**function** *check*() {  
 **throw new *Error***(**"Can't be empty"**)  
}  
**function** *f*(x = *check*(),y=1) {  
 *///*}

当x没有赋值时，会抛出异常

## 可变参数

**function** *f1*(...a) {  
 **var** sum = 0;  
 a.forEach(item => {  
 sum += item;  
 })  
 **return** sum;  
}

# 合并数组

ES5

**var *params*** = [**'hello'**,**true**,7]  
**var *other*** = [1,2].concat(***params***)

ES6

**var *params*** = [**'hello'**,**true**,7]  
**var *other*** = [  
 1,2,...***params***]

# 代理对象

**let *Person*** = {  
 **name**: **'es6'**,  
 **sex**: **"male"**}  
**let *person*** = **new *Proxy***(***Person***,{  
 get(target,key){  
 **return** target[key]  
 },  
 set(target,key,value){  
 **if**(key != **'sex'**){  
 target[key] = value;  
 }  
 }  
})  
***console***.table({  
 **name**:***person***.**name**,  
 **sex**:***person***.**sex**})  
  
**try**{  
 ***person***.**sex** = **'femal'**}**catch** (e) {  
 ***console***.log(e)  
}