js高级总结:

day03:函数,重overload ,匿名函数, 作用域和作用域链 闭包

**函数:function的本质是new Function**

**一.函数**

**1.创建函数**

1.1.function 函数名(形参){函数体,返回值return}**注意:函数会整体被声明提前 (同名函数会替换)**

1.2.v**ar 函数名=function(){函数体,return返回值}** 注意**函数名var 函数名被声明提前,=function(){}留在原位置(常用可以避免声明提前)**

1.3. 函数名=new Function("形参变量1","形参变量",...,"函数体; return 返回值");

**js函数的本质: 函数也是一个对象**

**其实function 只是 new Function()的简写而已！**

**function等效于new Function()**

**2.声明提前**

Js引擎会先将var声明的变量和function声明的函数，提前到当前作用域的顶部集中创建

**2.1 var 声明提前**

var a=1ex:var a 声明提前 值留在原地1

ex: var a

a=1

**2.2 函数声明提前**

赋值(=)留在原地！

=前的声明部分被提前，而函数定义留在原地。

函数其实也是一个对象，只不过内部存的不是数据，而是代码段而已

函数名仅仅是一个普通的变量。函数名通过保存函数对象的地址，来引用这函数对象

**2.3强调:**

引用类型存在多个值,在内存中只会保存变量的地址 如果使用同一个数组/对象 内存会复制一个地址新的变量,因为地址是指向同一个地址 所以修改任何一个都会影响所有的变量的值

原始类型:内存只可以存储一个值,所以只有重新赋值才会改变变量的值 另外要赋值个另外一个变量,内存会复制一个副本赋值给新德变量,不影响原始的值

**二.重载overload**

js不支持重载的效果(多个相同名的函数会覆盖为最后一个,声明提前)

解决: js中是通过一个特殊arguments对象来变通实现重载效果 arguments属于类数组对象,有一个length属性但不能使用数组家的函数

相同: 1. 下标i, 2. length, 3. 可用for循环遍历

arguments: { 0: 值1, 1:值2,... , length:元素个数 }

不同: 类数组对象不是数组家孩子！无权使用数组家任何函数！

使用:不要定义函数里不要定义形参,直接使用 arguments.length

ex:手机支付 function(){if(if(arguments.length==0){}else if(.....

**三.匿名函数以及匿名函数自调**

作用:防止全局污染

function(){} 匿名函数

强调:几乎所有回调函数都用匿名函数:

(function(){})();匿名函数自调

强调: 匿名函数自调前后的语句必须都加";"，不能偷懒!

**四.作用域(scope)和作用域链(scope chain)**

作用域:

在函数内var出来的变量，是局部变量

形参变量也会成为局部变量

1.全局作用域对象 属于全局变量 优点:可重用 缺点:造成全局污染

全局作用域存储在window中

2.函数作用域对象 属于局部变量/形参也是局部变量 优点:不会造成全局污染,缺:不可重用

2.1强调:先局部后全局

2.2调用函数时,会创建临时作用域对象 储存 函数体内的局部变量,以及形参,

2.3调用函数时,临时作用域对象会把地址给 给离他最近的"好友列表" 之后每次都会先找临时函数作用域对象,没有再找window, 两者都没找到才会报错

**五.闭包**

闭包形成的原因: 外层函数调用后，外层函数的作用域对象被返回出来的内层函数引用着，无法释放！就行了闭包对象!

本质:外层函数的作用域对象被内层函数应用着无法释放，这样的作用域对象就称为闭包对象

闭包3步:

(1). 用外层函数包裹要保护的变量和内层函数

(2). 外层函数将内层函数return到外部

(3). 外部的使用者需要调用外层函数，才能获得返回出来的内层函数对象。

