正课:

1. Function对象

2. 面向对象

1. Function对象

创建: 3种:

1. 声明方式:

function 函数名(形参列表){

函数体;

return 返回值

}

会被声明提前

2. 赋值方式:

var函数名=function (形参列表){

函数体;

return 返回值

}

不会被声明提前

揭示: 函数其实也是一个对象

函数名其实只是一个简单的变量

3. 用new:

var 函数名=new Function("形参1","形参2",...,"函数体")

重载overload:

什么是: 相同函数名，不同形参列表的多个函数，在调用时可根据传入参数的不同，自动选择匹配的函数执行。

为什么: 减少函数的个数，减轻调用者的负担

何时: 一件事，可能根据传入的参数不同，选择执行不同的操作时

如何:

问题: js语言默认不支持重载的语法  
 因为js不允许多个同名函数同时存在

如果同时存在，只有最有一个会覆盖之前所有

解决: 用arguments对象变通实现

arguments: 每个函数中自带的接收所有实参值的类数组对象

每个函数: 所有函数中都有arguments对象

自带: 不用自己创建，直接使用

作用: 接收所有实参值

数据结构: 类数组对象——长的像数组的对象

相同: 1. 下标 2. length 3. for循环遍历

不同: 类型不同——类数组对象是对象家孩子，无权使用数组家的API

匿名函数:

什么是: 定义函数时，不起名的函数，称为匿名函数

为什么: 节约内存！

何时: 只要一个函数，只使用一次时，都要用匿名函数，而不是有名称的函数

如何: 2种:

1. 回调函数:

比如: setInterval/setTimeout(function(){...},1000)

arr.sort(function(a,b){return a-b})

str.replace(/正则/ig,function(kw){ ... })

2. 匿名函数自调: 也称为立即调用函数IIF

什么是: 定义一个匿名函数，然后立刻执行

为什么: 临时创建一个函数作用域，避免使用全局变量

何时: 今后所有的js代码，都要放在一个巨大的匿名函数自调中。

(function(){

自己的功能代码

})()

作业:

1. function fun(){

var n=999;

nAdd=function(){ n++ };

return function(){

console.log(n);

}

}

var getN=fun();

getN()//?

nAdd()

getN()//?

2. function fun(){

for(var i=0,arr=[]; i<3;i++){

arr[i]=function(){ console.log(i); }

}  
 return arr;

}

var funs=fun();

funs[0]();//?

funs[1]();//?

funs[2]();//?