最后一天，面向对象

变量的作用域

var a=10;//如果写在window.onload里面，他就不能算全局变量，只能算局部内最高变量

想在函数外面访问局部变量，只有用闭包

也就是说，外部函数的变量，可以被内部函数调用，这叫做变量的作用域链

也就是说，我先在外部函数内部声明一个函数，接着，在外部函数里面返回内部函数

function fun1(){

var a = 10;

function fun2(){

console.log(a)

}

return fun2;

}

然后，我在外面调用fun1的时候，会把fun2这个函数体返回出来

返回的是一串文字：

function fun2(){

console.log(a)

}

接着，我把这一串文字，赋值给了一个变量；那么那个变量就变成这个函数了；

var demo = fun1();

接着，调用那个变量，就相当于执行了fun2

demo()

声明函数立马调用：

(function a(){}());===function a(){};a();

循环里面采用闭包传参

var divs=document.getElementsByTagName('div');

for(var i=0;i<divs.length;i++){

(function(j){

divs[j].onclick=function(){

console.log(j);

}

}(i))

}

闭包：

我们可以用一个函数去访问另一个函数内部的变量

优点：不会产生全局变量，实现属性私有化

缺点：闭包中的数据，会常驻内存，再不用的时候记得删掉，否则会导致内存溢出；

面向对象编程：

什么是对象？

简单来讲，对象，就是有属性，有方法的东西，就叫做对象；

吴泽权：

属性：身高，体重，爱好。。。。

方法：能干活，能搬东西；

吴泽权.搬东西（）；

那么，吴泽权就是一个对象

手机：

属性：宽高，颜色，重量；

方法：能上网，能打电话，能玩游戏。。。

手机.发送短信（）；

万事万物都是对象；

之前学过很多函数

数学函数：Math.round()。。。

时间函数：Date.getDay()….;

之前学过的函数，都是一个方法；

方法又是什么玩意儿？

方法就是函数，也就是工具；

例如，时间函数，他就是我们获取时间的一个工具；

Alert(‘s’);

Window.alert();

也就是说，我们整个JS都是一个面向对象的语言；

document.getElementById("btn1");

getElementById就是document的方法，

document就是一个对象吧？

它有很多方法。。。。。

对象是什么？

是多个工具集合的一个工具包

由多个函数组成的对象；

知道什么是对象了：就是有属性，有方法的东西，就叫做对象；

那么，什么是面向对象？？？？？？

就是：把各种零零散散的方法，封装成一个对象，然后给别人去使用，就是面向对象；

面向对象开发：

架构师：一个产品对象，一个用户对象

用户对象：

属性：账号、密码。。。。

方法：购买、注册、登录

传一个空框架→

中级程序员：封装一个用户对象，login()登录方法；

把框架里面的东西完成；

初级程序员：

<h1></h1> h1.innerHTML=yonghu .zhanghao

<button>购买</button> btn.onclick=function(){

Login();

}

面向对象开发，第一件事，声明对象；

//声明一个对象

var obj=new Object();//实例化一个对象；

//但是，我们不推崇这个方式声明对象

一般，除非特殊情况，需要实例化一个对象，或者延迟加载数据，基本不需要用到new关键字，因为在JS里面大量的new操作改变内存地址是一个很慢的操作，所以，通常为了效率起见，应该始终采用对象符号来声明对象；

var obj1={};//叫做字面量式声明对象

字面量式声明对象和json数组很像

其实，json数组就是对象的一个实例化；

为什么说面向对象虚无而又缥缈

他不是一个方法，也不是个属性，他是啥？？？

他是一种抽象的编程思路！！！！！！！！！

为什么要用面向对象编程？

人为什么现在这么发达？

因为人不断地进化！

人类的第一部里程碑式的进化，怎么来的？

由猴子进化成类人猿？

学会了使用工具，进入了石器时代

在外后面，学会了用火。。。。。。人类才进化到现在

编程，由当初的一片茫然，到现在的热火朝天

就是因为，我们学会了使用面向对象编程，学会了创建工具包给别人程序员用，也学会了使用别人的工具包，节约了我们大量的时间；

JAVA里面，他们可以把一个对象，封装成class(类)，或者public（包）

我们有啥？我们没有class，也没有public，但是我们更先进

我们一个函数，全部搞定

面向对象三大特征：

# 封装、继承、多态

封装，重复的封装起来，继承就是字面意思，子继承父

多态？面向不同的对象，展现不同的状态（例如网购支付）；

//刚刚，我们讲了使用一个对象，那么很多个对象。就要用到封装

function Person(name,age){

var obj={};//声明对象

obj.name=name;//通过传参传进来的name赋值给对象的name;

obj.age=age;

obj.showName=function(){

alert('我叫：'+name);

}

obj.showAge=function(){

alert('我今年'+age+'岁');

}

return obj;

}

var wzq=Person('吴泽权',18);

var xhh=Person('薛欢欢',19);

console.log(wzq,xhh)

// console.log(xhh.name)

// xhh.showName();

/\*这就是封装，但是这只是函数封装，并不是对象封装\*/

你从同一个函数生成的对象，相互之间没有任何联系

为了让他们其实是来自同一个对象，我们有我们的方法，也就是我们JS真正对象的封装：

## 构造函数

所谓的构造函数，其实就是普通函数，只不过内部使用了this变量，对构造函数使用了new运算符，就能够生成实例，并且this变量会绑定在实例对象上面；