21.3.30学习记录

1、循环语句

通过循环语句可以反复的执行一段代码多次

while循环：

语法：

```
while（条件表达式）{
    循环体；
}
```

条件表达式写死为true，则为死循环；

可以用break终止循环

循环三步骤：

创建初始化一个变量

在循环中设置一个条件表达式

定义一个更新表达式，每次更新初始化变量；

do 。。。 while循环：

语法：

```
do{
    循环体；
}while(条件表达式)
```

先执行后判断；所以循环体至少执行一次；

for循环：

语法：

```
for（初始化表达式；条件表达式；更新表达式）{
    循环体；
}
```

三个部分都可以省略，也可以写在外部

不写任何表达式，则是死循环;

双层循环实现金字塔;

break终止循环；

label：循环语句

使用break语句时，可以再break后跟着一个label，这样break将会结束指定的循环，而不是最近的；

continue：跳过当次循环！

console.time();,定义一个名字

console.timeEnd()用来停止一个计时器，传入计时器的名字

String Number Boolean Null Undefined

对象 Object

前言：基本数据类型都是单一的值，彼此之前无任何的联系；

介绍：对象是复合类型，可以保存多个不同数据类型

分类：

1、内建对象

--由ES标准中定义的对象，在任何的ES的实现中都可以使用

--比如： Math String Number Function等等都可以直接使用

2、宿主对象

--由JS的运行环境提供的对象，目前来讲主要指浏览器提供的对象

--比如BOM，DOM

3、自建对象

--由开发人员自己创建的对象

对象中的值叫属性

添加属性：

对象.属性名 = 属性值;

属性名的命名不强制要求遵守标识符的规范；

如果使用特殊的属性名，不能用.的方式，需要：

对象["属性名"] = 属性值;

obj["123"] = 789;

！！！中括号方式比较灵活，可以直接传递一个变量，变量值是多少就会读取哪个属性！！！！

读取属性：

对象.属性名

如果读取对象中没有的属性，不会报错而是会返回undefined；

修改属性：

对象.属性名 = 新的属性值；

删除属性：

delete 对象.属性名

属性值：

JS对象的属性值，可以是任意数据类型，包括对象

注意：in运算符：通过该运算符可以检查一个对象中是否含有指定的属性，有则返回true；　　语法：“属性名” in 对象

引用数据类型

前言：JS中的变量都是保存在栈内存中

基本数据类型的值也是在栈中，复制的时候各个值是独立的

引用数据类型的值是在堆内存中，而在栈中存储的是创建的地址，复制的时候就只是建立了地址的联系；而如果对某个对象赋值为null，实际是断了与某个堆内存的联系。

比较两个引用数据类型时，比较的是内存地址！！！

对象字面量创建对象：

var obj = {};

这种方法可以在创建的同时指定对象中的属性：

{属性名1：属性值，属性名2：属性值，.....}

这里的属性名一般不加引号，但是如果用特殊的属性名则就必须加引号；

函数

函数也是一个对象，但是区别于普通对象，函数可以封装一些功能，在需要的时候调用；调用时才会触发。

1、构造函数

new Function（“”）；

2、函数声明

function 函数名（【形参1，...】）{函数体}

3、使用函数表达式

var 函数名 = **function（）{函数体} 右边为匿名函数**

形参

可以在函数的额（）中指定一个或多个形参，相当于在函数内部声明了对应的变量；

在调用函数的时候，可以在（）中指定实参；此时实参会赋值给形参

传参的时候，函数解析器不会检查实参的类型；所以要注意是否有可能接收非法的参数，可以对类型检查；

传参的时候，函数解析器不会检查实参的数量，多余的实参不会被赋值，而实参少于形参时，没有对应实参的形参就是undefined；

return； 返回的时undefined；

return后面可以加条件表达式，会让代码更优美

return后可以是任意的数据类型，包括对象和函数！！！

参数也可以是对象！参数还可以是函数，完全可以把匿名对象传入；

--func（）

调用函数，相当于使用函数返回值

--func

函数对象，相当于直接使用函数对象

立即执行函数

用（）括号括起来表示这个匿名函数变为了一个函数对象，所以可以直接调用，也是用括号

立即执行函数往往只会执行一次

对象的方法：

就是函数作为一个对象的属性；

枚举对象，就用for..in语句。每次执行时会将对象中的一个属性的名字赋值

这样**取属性值**的话，就可以用【】方便的提取赋值变量的属性值了

作用域

--作用域指一个变量的作用的范围

分类：

--全局作用域{

--直接在script标签中的js代码，都在全局作用域

--全局作用域在页面打开时创建，在页面关闭时销毁

--在全局作用域中有一个全局对象window，可以直接使用

--在全局作用域中，创建的变量都会作为window对象的属性保存

--在全局作用域中，创建的函数都会作为window对象的方法保存

--全局变量在任意部分都可以访问到

}

变量的声明提前{

--使用var关键字声明的变量，会在所有的代码执行之前被声明，但是并不会被赋值；

--如果不适用var的话，则不会声明提前；

}

函数的声明提前{

--使用函数声明的方式创建的函数会在所有代码执行之前被创建

--但是使用函数表达式创建的函数不会声明提前

--声明提前才可以在函数声明前调用函数

}

--局部作用域{

--即函数作用域，调用函数时创建函数作用域，执行完毕后作用域消除

--每调用一次函数就会创建一个新的函数作用域，他们之间是相互独立的

--在函数作用域中可以访问到全局作用域的变量，反之不行；如果变量重名，而想访问全局变量，那就加window；

--在函数中不使用var的变量，是全局变量

}

debug调试！！

解析器在调用函数每次都会想函数内部传递一个隐含的参数，这个隐含的参数就是this；这个对象称为函数执行的上下文对象，根据函数调用方式的不同，this会指向不同的对象。

--以函数的形式调用时，this永远是window；

--以方法的形式调用时，this永远指向方法所属的那个对象

复习

*表单：收集用户信息

*组成：表单域、表单控件（表单元素）、提示信息

表单域：包含表单元素的区域

<form action="url" method="" name="">

表单元素：<input type="" />

属性值：value\checked\maxlength\name

type:text\passwor\radio\checkbox\submit\submit\reset\button\file

*<lable>标签 用于绑定一个表单元素

*select 下拉列表 <select></select>

*文本域 textarea

*列表：布局

*无序列表、有序列表、自定义列表

*无序列表：

*有序列表：

*自定义列表：<dl>

<dt名词</dt>

```
        <dd>名词解释</dd>
    </dl>
```

表格：

table标签{ align：表格位置 cellpadding：表格内间距　cellspacing：表格外间距}

thead标签{ 表格标题第一栏}

tbody标签{表格内容栏}

tr标签：行；　　th标签或td标签：列；

合并单元格{

跨行合并：rowspan

跨列合并：colspan

}

并集选择器用逗号分开；可以同时选择多个元素

后代选择器用空格分开；可以精准选择一个元素

类选择器：　用class名区分，class名可以重复，所以一个标签可以有多个类名