# JS基本语法笔记

JavaScript简称为JS

JavaScript是什么?

是一门脚本语言:不需要编译,直接运行

是一门解释性的语言:遇到一样代码就解释一行代码

C#语言是一门面向对象的语言,也是编译语言,是.net技术使用的

是一门动态类型的语言---不解析

是一门基于对象的语言---不解释

是一门弱类型的语言,声明变量的时候都使用var

强类型的语言:C#就是----->

存储一个数字: int num=10; float num2=90.3;

JS分三个部分:

1. ECMAScript标准: js的基本的语法

2. DOM

3. BOM

JS用了将近10天来发明的

变量---作用:操作数据(存储数据,读取数据)

变量的声明:

var 变量名字;

var num;

变量的初始化

var 变量名字=值;---->字面量的方式赋值

var str="您好啊";

js中大小写是区分的

js中的字符串可以使用单引号,也可以使用双引号---暂时推荐使用双引号

js中每行代码结束要有分号 ;

数据类型:

js中的原始数据类型: number,string,boolean,undefined,null,object

null和undefined数据是没有太大意义的,null是很有意义的---对象的位置讲

NaN----不是一个数字,不是一个数字和一个数字计算--->结果就是NaN

isNaN()--->判断这个变量或者这个值是不是 不是一个数字---如果不是一个数字结果是 true,如果是一个数字结果false

number数据类型----无论是整数还是小数都是number数据类型的

string数据类型----字符串,获取字符串的长度:变量名.length

boolean数据类型---两个值,true，false

null----只有一个,null

undefined----只有一个,undefined,一个变量声明了,没有赋值

object---对象-----面向对象的时候讲解

类型转换:

其他类型转数字

parseInt()---->转整数

parseFloat()--->转小数

Number()---->转数字-----要比上面两个严格

其他类型转字符串

.toString()

String();

其他类型转布尔类型

Boolean()---->要么是true要么是false

运算符:----操作符

算术运算符: + - / %

复合运算符: += -= = /= %=

赋值运算符: = 优先级别是最低的

一元运算符: ++ --

三元运算符: ? :

关系运算符: > < >= <= == === != !==

逻辑运算符: && || !

关系运算符的结果是布尔类型

逻辑运算符的结果是布尔类型

表达式1&&表达式2

如果有一个为false,整个结果就是false

表达式1||表达式2

只有有一个为true,整个结果为true

!表达式---结果就是取反

一元运算符: ++ --

如果不参与运算, ++在前面还是在后面结果都是一样的 +1

如果不参与运算, --在前面还是在后面结果都是一样的 -1

var num=10;

num++; 后+

++num; 前+

无论是前+还是后+只要参与运算,结果可能就不太一样

如:

var num=10;

var sum= num++ +10;

++在后面的时候, ,然后自身加1

var num=10;

var sum=++num +10;

++在前面的时候,先自身加1,然后再参与运算

流程控制: 代码的执行过程

1. 顺序结构:代码的执行的顺序,从上到下,从左到右(不严谨)

2. 分支结构: if,if-else ,if-esle if,switch-case ,三元表达式

3. 循环结构:while 循环,do-while,for循环 for-in循环

总结分支:如果只有一个分支,就用if

如果有两个分支,就用if-else

如果多个分支,一般是范围的,推荐使用if-else if

如果多个分支,一般是具体的值,推荐使用switch-case

总结循环:

while:先判断后循环,有可能一次循环都不执行

do-while:至少执行一次循环体,再判断

for循环:知道了循环的次数,推荐使用for循环

调试:是为了找代码的错误和问题所在,

断点,不加断点,就不方便调试

函数:把一些重复的代码封装在一个地方,在需要的时候直接调用这个地方的代码就可以 了

函数作用:代码重用

函数的参数:

1.形参:函数定义的时候,函数名字后面的小括号里的变量

2.实参:函数调用的时候,函数名字后面的小括号里的变量或者值

返回值:

函数中有return，函数有返回值

函数中没有return,函数没有返回值

没有明确返回值:函数中没有return或者return后面没有任何内容

如果一个函数没有明确的返回值,接收这个函数,结果是undefined

无参数无返回值的函数

无参数有返回值的函数

有参数无返回值的函数

有参数有返回值的函数

arguments----->可以获取函数调用的时候,传入的实参的个数

arguments是一个对象,是一个伪数组

arguments.length--->是实参的个数

arguments[索引]---->实参的值

作用域:变量的使用范围

全局作用域:全局变量在任何位置都可以使用的范围

局部作用域:局部变量只能在某个地方使用---函数内

作用域链:在一个函数中使用一个变量,先在该函数中搜索这个变量,找到了则使用,找不 到则继续向外面找这个变量,找到则使用,一直找到全局作用域,找不到则是undefined

全局变量:只要是在函数外面声明的变量都可以看成或者是理解成是全局变量

局部变量:在函数中定义的变量

预解析:在执行代码之前做的事情

变量的声明和函数的声明被提前了,变量和函数的声明会提升到当前所在的作用域的最 上面

函数中的变量声明,会提升到该函数的作用域的最上面(里面)

如果有多对的script标签都有相同名字的函数,预解析的时候是分段的,互不影响

编程思想:

面向过程:凡事亲力亲为,所有的事情的过程都要清楚,注重的是过程

面向对象:提出需求,找到对象,对象解决这个问题,我们要结果,注重的是结果

面向对象的特性:封装,继承,多态,(抽象性)

JS是一门什么样的语言?

是一门解释性的语言

是一门脚本语言

是一门弱类型的语言

是一门基于对象的语言

是一门动态类型的语言

对象:有属性和方法,具体特指的某个事物

对象:js中就是一组无序的属性的集合

属性----特征

方法----行为

创建的对象的方式:

1.通过调用系统的构造函数创建对象 new Object()

var obj1=new Object();

2.自定义构造函数创建对象

var obj2=new 自定义构造函数();

3.字面量的方式创建对象

var obj3={};

变量 instanceof 对象------->布尔类型,判断这个变量是不是这个类型的

JSON格式的数据,都是键值对,成对的数据

var obj={

name:"小明"

};

var json={

"name":"小明"

};

json的数据实际上就是格式化后的一组字符串的数据

对象设置属性的值的写法

对象.属性名字=值;----点语法

对象["属性的名字"]=值;-----

对象获取属性的值的写法

对象.属性

对象["属性"]

遍历对象

for(var key in 对象){ key---是一个变量,这个变量中存储的是遍历的对象的属性的名字}

原始数据类型:number,string,boolean,null,undefined,object

基本类型(简单类型,值类型):number,string,boolean

复杂类型(引用类型):object

空类型:undefined,null

基本类型的值在栈上

复杂类型的对象在堆上,地址(引用)在栈上

值类型之间传递的是值

引用类型之间传递的是引用(地址)

对象分三种:内置对象,自定义对象,浏览器对象

内置对象:系统提供的

自定义对象：自己写的

浏览器对象:浏览器的

Math 是一个对象,但不是一个函数

Math对象下的属性和方法都是静态

方法:

Math.ceil()---向上取整

Math.floor()---向下取整

Math.Pi----圆周率的值

Math.Max()---一组数字中的最大值

Math.Min()---一组数字中的最小值

Math.abs()----绝对值

Math.random---随机数字

Math.sqrt()----开平方

Math.pow()----一个数字的多少次幂

new 的执行过程:----->new的时候，系统做了什么事?

1. 开辟空间,存储创建的新的对象

2. 把this设置为当前的对象

3. 设置属性和方法的值

4. 返回当前的新的对象