JavaScript

一．JavaScript概述

1.js历史

诞生于1995年，当时是为了完成表单输入的验证。

LiveScript

1997年制定ECMAScript标准。

ECMA欧洲计算机制造商协会。

2.js组成

核心(ECMAScript):类似于JavaSE，基本语法操作

文档对象模型(DOM) ：用JS操作元素

浏览器对象模型(BOM) ：用JS操作浏览器

3.js的位置

在html页面中<script>代码</script>

引用外部的js文件<script src=”文件名”></script>

4.js延迟脚本

defer

脚本在执行时不会影响页面构造

脚本会被延迟到整个页面解析结束后在加载。

<script src=“1.js” defer=”defer”></script>

<body></body>

5.js异步加载

Async：不让页面等待两个脚本下载和执行，从而异步加载页面内容。

<script src=“1.js” async=”async”></script>

<script src=“2.js” async=”async”></script>

<body></body>

目的：都是为了让内容先渲染出来了

二．JavaScript基础

1、数据类型

原始数据类型：

Number String Boolean Undefined Null

在内存中，存放在栈中的简单数据段，也就是说，他们直接存储在变量访问的位置。

JS是弱类型语言，也就是说定义变量时，是无需声明类型的，和java相反

定义变量var a;

后期会根据变量的值来判断是什么类型的

var a1 = 1;//number

var a2 = “abc”;//String ‘ ’ “ ”

var a3 = true;//boolean

var a4; //undefined a4=undefined;

var a = null;//object

typeof 查看数据类型

typeof a1;//number

typeof(a2);//String

Undefined 未被添加类型值undefined

var obj = null; Null类型 typeof(obj);//object null其实是一个object类型java把null叫做空对象，经常是用来查看某个对象是否为空alert(obj== null);

Boolean 值true false

Boolean()将其他类型强转成boolean类型

String->Boolean只要不是空串都为true

Boolean("hello") true;

Boolean("") false;

Number->Boolean 0 false 非0为true

>>Boolean(0)

false

Undefined-> Boolean false

>>Boolean(undefined)

false

Object->Boolean null false

>>Boolean(null)

false

Number进制

10进制123

8进制070

079 自动转换为十进制的79

16进制0x12

科学计数法

1.42e9

Number的属性

浮点数的max和min

MIN\_VALUE

>Number.MIN\_VALUE

4.94065645841247E-324 这是一个无限接近于0的数，说明，js没有0

MAX\_VALUE

>>Number.MAX\_VALUE

1.79769313486232E+308

var infinity = Number.POSITIVE\_INFINITY

var ins = Number.NEGATIVE\_INFINITY;

Infinity 无穷

var box = 0/0;// NaN （not a number）

NaN是个值 NaN==NaN FALSE

var box = 12/0; Infinity

var box = 12/0\*0; NaN

检查是不是数字：

isNaN(box); // true true表示不是一个数字

任何类型

Number()

true 1 Number(true);

false 0

null 0

undefined NaN

'' 空串 0

'Lee' NaN

智能转换字符串

parseInt()

'456Lee' 456

'Lee456Lee' NaN

>>parseInt('Lee456Lee')

1.#QNAN

parseInt('1f',16);

>>parseInt('1f',16);

31

parseInt('0x1f');

>>parseInt('0x1f');

31

parseFloat()

'123.4.5' 123.4

2、运算符

+ - \* / %

+ 拼接字符串

==判断值是否相等

===判断值与类型都是否相等 恒等于

&& || 逻辑运算符

三目运算符

3、控制语句

if语句

if(条件){

}else if(条件){

}else{

}

switch语句

switch(表达式){

case 值1:

break;

case 值2:

break;

default :

break;

}

4、消息框

/\* 警告框 \*/

alert("xxx");

alert("\n"); //\n代表换行

/\* 确认框 \*/

var str1 = confirm("xxx"); //确定,返回值为true 取消,返回值为false

/\* 输入框 第二个参数可写可不写 \*/

var str2 = prompt("请输入xxx:","\*\*\*\*\*\*");

5、循环语句

for语句

1). /\*for (变量=开始值;变量<=结束值;变量=变量+步进值) { 需执行的代码 } \*/

2). /\* for(var 变量名 in 数组名){ 数组名[变量名]; } \*/

while语句

/\* while (变量<=结束值) { 需执行的代码 } \*/

do-while语句

/\* do { 需执行的代码 } while (变量<=结束值); \*/

break; //跳出当前循环

continue; //跳出本次循环,开启新的循环

作业：小九九 或 金字塔

6、javascript函数

JavaScript设计的最出色的的就是它的函数的实现，它几乎接近完美。

函数类似于C语言中的函数，可以直接定义调用，但调用时，函数必须先声明好。

函数的声明：使用function关键字

function 函数名(){

}

无参函数

function name(){ ... }

单参函数

function name(arg){ ... }

多参函数

function name(a1,a2,a3){...}

在函数里面，会将这些参数放在arguments对象里，arguments是一个数组

argumnets对象 function a(){ return arguments[0]; }

返回值函数

function name(){ return }

递归函数

function name(){ name() }

匿名函数

function(){} 标签对象.事件 = function(){}

构造函数

构造函数和普通的函数一样，但是具有以下两个特殊性质。

1).通常构造函数的首字母是大写的（让识别构造函数变得更容易）。

2).构造函数通常要和 new 操作符结合，用来构造新对象。

function Fun(){

//var 变量 局部变量

var name = "张三";

var age = 12;

//属性

this.username="里斯";

}

var f = new Fun(); //创建对象

alert(f.username); //里斯

alert(f.name); //undefined 未定义

//全局变量

var a = 20;

function fun(){

var a = 10; //局部变量

alert(this.a); //20 this相当于Window对象

alert(a); //10

}

fun();

注意：

js的this

1.全局作用域下的函数中的 this===window。Window代表的是全局对象，也是窗口对象

2.当函数作为对象obj的方法调用时，函数中的 this===obj。

3.构造函数的this为新创建的对象。

4.嵌套函数中的this不会继承上层函数的this，嵌套函数中的 this===window。

7、JavaScript对象

1). String 字符串

单引号和双引号

var box = 'box’;

常用方法：

str.length

str.concat(str1)

str.indexOf(substr)

str.substr(num1,[num2]) //substr(start ,count)

>>box.substr(1)

"ox"

>>box.substr(1,1)

"o"

注：num1表示从第几个开始切。Num2表示切几个，如果不写，就默认切到最后

Str.toLowerCase()

Str.toUpperCase()

>>box.toUpperCase();

"BOX"

Str.replace(str1,str2)

>>box.replace('b','c');

"cox"

2). object类型

对象的创建：

var obj = {} //空的object

>>var obj = {};

Undefined

>>typeof obj

"object"

obj = null //空对象

空的对象是对象存在，只是没有自己定义的属性和方法

空对象是对象根本不存在

var obj = new Object();

例如：new String()

>>var obj = new String();

Undefined

>>typeof obj;

"object"

引用数据类型：

在内存中，存放在堆中的对象，就是说，存储在变量处的值是一个指针，指向存储对象的内存处。

var person = new Object();

person.name="Kity";

person.age=14;

对象字面量表示法,属性名可以为字符串

var person = {

"name" : "kity",

age : 14

}

var person = {}; //new person();

对象属性的访问

person.age;

person["name"] ;

3).Array类型

可以存放任意类型，数组长度可变，数组的创建,可以省略new

var arr = new Array();

var arr = new Array(20);

数组字面量表示法

var arr = ["abc","bcd","cde"];

数组的读取和设置

var color = ["red","green","blue"];

color[0] ; //"red"

color[2]="yellow"; //替换掉blue

color[3]="grey"; //新增"grey"

color.length; //4

color[color.length]="pick"; //尾部添加一项

操作方法

数组对象.concat(数组1,数组2,…)

数组对象.join(“#”);

**堆栈操作：**

push()//在末尾添加

pop()//删除末尾

shift()//删除顶端

unshift()//在顶端添加

重排序方法

reverse() //反转数组项

sort() //默认调用toString()

var arr = [0,1,5,10,15]

arr.sort();//0,1,10,15,5

//定义比较方法

function compare(v1,v2){

if(v1<v2){

return -1;

}else if(v1>v2){

return 1;

}else{

return 0;

}

}

arr.sort(compare);

4).Math

常量：

E PI

方法：

abs()

cos()

sin()

max(a,b)

min(a,b)

pow(a,b)

random()

round(a)

sqrt()

5). Date类型从1970- 1- 1 00:00:00开始记

var today = new Date()

today.getFullYear()

.getMonth()

getDate()

getDay()

getHours

getMinutes

getSeconds

toLocaleString()

toLocaleDateString()

toLocaleTimeString()

6). 逻辑Boolean()

var b1=new Boolean();

var b2=new Boolean(0);

var b3=new Boolean(null);

var b4=new Boolean("");

var b5=new Boolean(false);

var b6=new Boolean(NaN);

7). 正则表达式 RegExp

var str = "hello world";

var r1 = new RegExp("a");

document.write(r1.test(str)+"<br/>"); //test() 包含返回true;不包含返回false

var r2 = new RegExp("o","g"); //在全局中去查询是否包含指定字符

//exec() 包含返回查询的数据;不包含返回null

do{

var b = r2.exec(str);

document.write(b+"<br/>");

}while(b!=null);

8、事件

鼠标事件

mouseover mousemove mousedown mouseup mouseout mousewheel click dbclick

键盘事件

keydown keyup keypress

keyCode //键盘码

其他事件

focus blur change

事件监听

事件：

1.事件源： 谁去触发了此事件 DOM对象

2.事件类型： 触发的是什么样的事件 鼠标事件，键盘事件其他事件

3.事件处理： 如何去处理事件 function(){ };

方式一：

window.onload = function(){

标签对象.事件=function(){

}

}

方式二：

在标签中直接使用事件

<form onsubmit=”return xxx() ”></form>

<div id="d2" onmouseover="mouseoverFun(this.id)" onmouseout="mouseoutFun(this)"></div>

案例：完成一个遮罩效果

9、Browser 对象

/\* 跳转到某个html页面 \*/

window.location.href="xxx.html";

<input type="button" onclick="JavaScript:window.location.href='xxx.html'" value="跳转"/>

/\* 返回上一页 \*/

window.history.back();

<input type="button" value="返回" onclick="window.history.back()"/>

10、DOM操作 document object model

1）文档树的介绍

获取标签对象的方法

getElementById();

getElementsByClassName()[index];

getElementsByName()[index];

getElementsByTagName()[index];

2）图片image处理

document.images; //获取文档中<img>元素集合

document.images[0].src="xxx.png";

表单处理

document.forms; //获取文档中<form>元素集合

document.forms[0].submit();

3）定时器

window.setInterval(函数名,1000); //clearInterval

延时器

window.setTimeout(函数名,2000);

案例：图片轮播

4）//给下拉菜单select添加option节点

function addOption(){

//1.创建option节点

var o = document.createElement("option");

//2.给option节点赋值

o.text = "上海";

//3.将option添加至select 1).null默认添加到末尾2).s.selectedIndex当前选中的option位置

var s = document.getElementById("s");

s.add(o, s.selectedIndex);

}

//给下拉菜select单移除option节点

function removeOption(){

var s = document.getElementById("s");

//删除选中的option

//s.remove(s.selectedIndex);

//删除最后一个

if(s.length>0){

s.remove(s.length-1);

}

}

作业：

1.作业：进行表单**非空验证**，其中要求用户名进行正则表达式验证，**不能包含非法字符**。

2. 设计简单计时器

3. 模仿input中的placeholder属性特点，并且不能超过15个字符