11对象

是应用类型数据

（1）对象：是一组属性和方法（功能）的集合。

哪些是对象？

一个手机：属性：品牌、尺寸。方法：打电话，发短信

一个人：属性、年龄、姓名。方法：工作，说话

万物皆对象

（2）JS中的对象

内置对象：JS提供的

宿主对象：根据JS不同的执行环境来划分

自定义对象：自己创建的对象

（3）自定义对象

对象字面量

内置构造函数

自定义构造函数

（4）使用对象字面量创建对象

使用大括号{}创建对象

属性名和属性值之间用冒号隔开

多组属性之间用逗号隔开

属性名中引号可加可不加，如果含有特殊字符必须加引号

{属性名：属性值…}

练习：创建一个人对象，添加姓名，性别，年龄

练习；创建一个员工对象，包含编号、姓名、性别、生日、工资、所属部门编号

（5）访问对象的属性

对象属性名

对象['属性名']

可以添加之前不存在的属性；

如果属性不存在，属性值为udefined.

/\*var pople={

eid:1,

name:'hh',

sec:'nv',

birthday:'1996',

salary:20000,

'dept-id':20

}

//console.log(pople)

//访问对象中的属性值，获取对象中的属性值

console.log(pople.name)

console.log(pople['dept-id'])

//访问对象中不存在的属性返回undefined;

console.log(pople.size)

//修改对象中的属性值

pople.salary=10000;

pople['birthday']='2019-12'

console.log(pople)\*/

练习：创建一个电脑对象，包含的属性有屏幕尺寸，品牌，颜色，内存大小，获取品牌的属性值，修改颜色，添加产地

var computer={

size:5.5,

brand:'vivo',

color:'blue',

memory:'16G',

}

console.log(computer);

console.log(computer.brand);

computer.color='red'

computer.madein='成都'

console.log(computer)

（6）、使用内置构造函数创建对象

new Object(); 创建一个空的对象

需要单独添加每一个属性

对象['属性名']=‘属性值’

对象.属性名=‘属性值’

//使用内置构造函数创建对象

/\*var car=new Object();

car.id=123;

car.brand='BMW';

car['color']='black';

console.log(car)8\*/

练习：使用内置构造来创建用户对象，包含编号，用户名，密码，邮箱，手机

（7）、遍历对象的属性

访问所有的属性

for(var key in 对象){

key，//对象中所有属性名

对象[key] //通过属性名找对应的属性值

}

/\*var person={

name:'tom',

age:18,

sex:'男'

};

for(var key in person){

//key对象中每一个属性名

console.log(key,person[key])

}\*/

练习：创建对象，包含若干个成绩属性，遍历对象中属性获取总成绩

提升声明sum用于保存总和

var total={

//name:'mak',

//age:19,

//sex:'女',

math:99,

china:75,

english:78,

history:89

}

var sum=0;

for(var key in total){

//console.log(key,total[key])

//把每一个属性值加到sum中

sum+=total[key];

}

console.log(sum)

（8）、对象中的方法

方法也称为成员方法，对应的是一个匿名函数

var person={

name:'tom',

say:function(){

this.name//this指当前所在的对象

}

}；

person.say()//调用方法

创建一个圆对象，包含半径、圆周率属性；添加获取面积和获取周长的方法，分别返回对应的值。

尝试可以构造函数创建对象，并添加属性和方法。。

var circle=new Object();

circle.r=5;

circle.pi=3.14

circle.getArea=function(){

return this.pi\*this.r\*this.r;}

circle.getLength=function(){

return 2\*this.pi\*this.r;

}

console.log(circle.getArea())

console.log(circle.getLength())

（9）检测对象中是否含有某个属性

对象属性名===undefined,如果是true——>不存在，false——>存在。

比较一个属性的值是否为undefined

console.log(person.sex===undefined);

对象.hasOwnProperty('属性名');如果是true——>存在，false——>不存在

//查看是否有自己的某个属性

console.log(person.hasOwnProperty('name'));

‘属性名’in 对象；如果是true-->存在，false🡪不存在

//查看某个属性是否存在与对象中

console.log('phone'in person);

/\*var book={

id:1002,

title:'javascript高级程序设计',

price:79

}

//把book变量中存储的数据赋给book2变量

//把现在book存储的地址赋给了book2

//现在

var book2=book;

//修改你book中属性

book.price=86

console.log(book2)

book2.title='javascript权威指南';

console.log(book2)

//fn提升到最前边

//var fn

//function fn(){}//把一个函数赋给fn

console.log(fn)

var fn=1;//再次赋盖了fn中的值

function fn(){

}

console.log(fn);

12、数组

数组是由多个元素组成的集合，每个元素就是一个数据。

（1）创建数组

①数组字面量

['元素1'，'元素2'，…]

练习：创建数组，包含多个商品的名称；创建数组，包含多个城市名称

var mp=['牛奶','鸡蛋','牛肉','西红柿'];

//console.log(mp);

var ct=['成都','都江堰','乐山','德阳','重庆'];

//console.log(ct);

②访问数组中的元素

数组[下标] 下标是从0开始，第一个元素的下标就是0

//通过下标来获取元素

console.log(ct[4]);

ct[5]='布鲁塞尔';//添加元素

ct[4]='米兰'//修改其中的元素

console.log(ct)

③内置构造函数

new Array(元素1，元素2，…)

new Array(3) 初始化元素个数为3，也可以添加更多元素

练习：创建数组，保存若干个课程名称

练习：创建数组，初始化长度为5，添加篮球场上的五个位置

var cal=new Array('数学','语文','英语','物理','化学')

console.log(cal)

var bas=new Array(5);

bas[0]='中场';

bas[1]='后场';

bas[2]='前锋';

bas[3]='后卫';

bas[4]='主攻';

console.log(bas)

var car=new Array('奔驰','宝马','奥迪');

var car2=new Array(3);

car2[0]='凯迪拉克';

car2[1]='沃尔沃';

car2[2]='捷豹';

car2[3]='路虎';

console.log(car2)

④数组的长度

数组.length 获取数组中元素的个数

作用：在数组的末尾添加新的元素

数组[数组.length]=值；

var bas=new Array(5);

bas[0]='中场';

bas[1]='后场';

bas[2]='前锋';

bas[3]='后卫';

bas[4]='主攻';

bas[bas.length]='进攻后卫'

//console.log(bas)

//获取数组的长度或者个数

console.log(bas.length)

练习：创建一个空数组，使用数组长度添加若干个国家名称

var cat=new Array(6);

cat[cat.length]='巴黎';

cat[cat.length]='里约';

cat[cat.length]='伦敦';

cat[cat.length]='米兰';

cat[cat.length]='埃塞俄比亚';

cat[cat.length]='西班牙';

console.log(cat)

(2)数组的分类

数组分为索引数组和关联数组

索引数组以0以上的整数作为下标

关联数组义字符串作为下标，只能单独的添加元素

（3）遍历数组

①for--in

for(var key in 数组){

key 要遍历的下标

数组[key] 下标对应的元素

}

既可以遍历关联数组，也可以遍历索引数组

练习；创建数组，包含有多个学生的成绩，获取总成绩

var sum=0

var studes=[65,98,78,85,99,88,48]

//for (var key in studes )

//{sum+=studes[key];

//}

//console.log(sum)

②循环

for(var i=0;i<=数组.length;i++){

I 代表下标 0-数组长度

数组[i] 代表下标对应的元素

}

只能遍历索引数组

//使用for循环获取下标0-4，使用下标获取元素

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

//i代表下标

//studes[i]下标对应的元素

console.log(i,studes[i])

sum+=studes[i]

}

console.log(sum)

练习：创建数组，包含多个姓名，把tom全部改成'汤姆'。

参加数组

遍历得到每一个元素

判断进这个元素是否为Tom汤姆，如果是，重新赋值为汤姆在遍历之后打印数组

var name=['tom','make','jieke','linda'];

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

{

if (name[i]=='tom')

{name[i]='汤姆'

}

}console.log(name)

//创建数组，包含多个姓名，查询Tom出现的次数

//创建数组

//遍历数组

//初始化变量的次数为0

//判断每个元素是否为tom,如果是次数加1

/\*var name=['tom','mak','jiexi','tom','timu','linaad']

for (var i=0,str=0;i<=name.length ;i++ )

{if (name[i]=='tom')

//如果每个元素为tom，则次数加一

{str++

}

}console.log(str)\*/

//练习：创建数组，包含多个数字，获取这组数字地最大值

//创建数组

////遍历数组元素

//声明变量用于保存最大值，默认把第一个元素放进去

//使用之前声明的变量和每个元素比价，如果变量小于任何一个元素，咋把该元素放入到变量中

var a=[25,56,98,99,102,015,250,126,380,666]

var max=a[0]

for (var i=1;i<=a.length ;i++ )

{//用max和数组每个元素比较

//如果小于任意一个元素，则把该元素放入到max

if(max<a[i])

{max=a[i]

}

}console.log(max)

练习创建函数getAvg，传递一个参数（一组工资），返回工资的平均值

实参就是一个数组，包含多个工资；

函数体中，计算元素的和，求平均

平均=总和/元素长度

(4)数组API(方法)

API应用程序编程接口，预定义好的一些方法或者函数

toString() 将数组中的元素按照逗号组合成字符串

join('-')将数组中的元素按照指定字符组合成字符串，默认是逗号

concat(arr2,arr3) 拼接多个数组

slice(start，end) 截取数组中的元素,start开始的下标，end结束的下标，不包含end本身，如果是负数表示倒数；返回一个数组

splice(start,count,value1,value2…)删除数组中的元素，start开始的下标，count删除的数量，value删除后插入的元素；返回删除的元素，原数组会发生变化

reverse()翻转数组中的元素

sort()对数组中的元素进行排序，默认按照unicode码从小到大[]排序

sort(function(a,b){return a-b})从小到大排序

sort(function(a,b){return b-a})从大到小排序

push() 往数组的末尾添加元素，返回数组的长度

pop() 删除数组末尾的元素，返回删除的元素

unshift()往数组的开头添加元素，返回数组的长度

shift()删除数组开头的元素，返回删除的元素

/\*var arr=['tom','jerry','king'];

var arr2=['david','lucy']

var arr3=['smith','white']

//数组转为字符串

console.log(arr.toString() );

console.log(arr.join('-'))

//拼接多个数组中的元素

console.log(arr.concat(arr2,arr3))\*/

/\*var arr=['java','html','css','ajax','vue','react','angular'];

//截取

console.log(arr.slice(1,4))

console.log(arr.slice(-3,-1))\*/

练习创建数组a-g,每个字母是一个元素，分贝截取cd,f，b,把截取的结果拼接成一个新的数组

var arr=['a','b','c','d','e','f','g'];

var a=arr.slice(2,4)

var b=arr.slice(5,6)

var c=arr.slice(1,2)

console.log(a.concat(b,c))

//删除数组中的元素

/\*var arr=['html','css','js','nodejs','ajax','vue','react','angular'];

//console.log(arr.splice(2,2))

//console.log(arr.splice(-3,1))

console.log(arr.splice(-3,0,'ionic','h5','c++'))

console.log(arr)\*/

//翻转数组中的元素

var arr=['a','b','c','d'];

console.log(arr.reverse())

//对数组元素排序

var arr=[75,58,99,25,56,102,23];

//回调函数——匿名函数以实参传递

console.log(arr.sort(function(a,b){

return a-b;

}))

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

练习创建数组a-h，每个字母是一个元素，删除d,e替换f为m,在下标为1的位置插入字母z

var arr=['a','b','c','d','e','f','g','h']

console.log(arr.splice(3,2))

console.log(arr.splice(-3,1,'m'))

console.log(arr.splice(1,0,'z'))

console.log(arr)

练习使用遍历数组翻转数组中的元素

['a','b','c','d']🡪['d','c','b','a']

使用冒泡排序将数组中的数字从小到大排序

循环嵌套

/\*//遍历翻转数组

var arr=['a','b','c','d'];

//准备新数组倒着获取数组中的元素放入到新数组

var newArr=[];

//遍历数组arr

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

//对应元素的下标=长度-1-i

//0 4 4

//1 4 3

//console.log(arr.length-1-i)

//把倒着获取的元素，放入到新数组

newArr[i]=arr[arr.length-1-i];

}

console.log(newArr)\*/

//冒泡排序

var arr=[78,6,23,9,45];

//外层循环控制次数，比较4轮（循环4次）

for (var i=1;i<arr.length ;i++ ){

//1 5 4

//2 5 3

//3 5 2

//4 5 1

//内层循环：比较的次数

//循环条件：数组长度-i

for (var j=0;j<arr.length-i ;j++ ){

//如果当前的元素大于下一个元素，则元素位置交换

//当前元素下标j 下一个元素j+i

if (arr[j]>arr[j+1]){

var c=arr[j];

arr[j]=arr[j+1];

arr[j+1]=c;

}

}

}

console.log(arr)

var arr=['html','css','js']

//作用

//参数

//返回值

//在数组末尾添加元素

//console.log(arr.push('nodejs'))//push推

//删除数组末尾的元素

//console.log(arr.pop())

//往数组的开头添加元素，返回数组的长度

//console.log(arr.unshift('java'))

//删除数组开头的元素，返回删除的元素

console.log(arr.shift())

console.log(arr)

（5）二维数组

数组中的每个元素也是数组

var arr=[[],[],[]…]

访问二维数组中的元素 arr[下标][下标]

/\*var pro=['北京','广东','江苏']

//var city=['海淀','朝阳','东城','西城','广州','深圳','惠州','南京','苏州','无锡']

//二位数据

var city=[

['海淀','朝阳','东城','西城'],

['广州','深圳','惠州'],

['南京','苏州','无锡'],

];

//console.log(city[1][1])

//console.log(city[2][2])\*/

var arr=['a','b','c']

var arr2=arr;

//把arr中的地址赋给了arr2

arr.push('d')

//console.log(arr2)

//赋值为null,不在指向任何一个堆内存数据,释放一个内存空间

arr=null;

arr2=null;

console.log(typeof null)

（6）字符串

包装对象：目的是原始类型数据像引用类型一样，具有属性和方法。

js提供了3种包装对象：String Number Boolean

String 将任意数据转成字符串

new String() 返回对象，使用和字符没有区别

String()返回字符串

/\*var str1='hello';

//将数据强制转成字符串，并返回对象

var str2=new String(1);

//console.log(str2,typeof str2)

//console.log(str2+2)

//将数据转为字符串，返回字符串

var str3=String(true)

console.log(str3,typeof str3)\*/

①转义字符——\

转换字符的意义

用法：再要转义的字符前加\

console.log('it\'s a dog')；(\' 将引号转换成普通字符 )

console.log('hello \nworld');(\n 将字符n转义成换行符)

console.log('a\tb');(将字符t转义成制表符（tab键效果）).

练习打印出现welcome to chi\na

（7）字符串API

length 获取字符串的长度

charAt 获取下标对应的字符，也可以使用数组形式字符串[下标]

练习：遍历字符串，打印出每一个字符

var str='makenihaoma';

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

console.log(str[i])

}

charCodeAt() 获取某个字符的Unicode码

console.log('a'.charCodeAt)

indexOf(value,start)查找某个字符串的下标，value要查找的字符串，start开始的下标，默认是0，如果找不到返回-1

lastIndexOf 查找某个字符串最后一次出现的下标，找不到返回-1

练习声明变量保存字符串，检测该字符串中是否含有@如果有打印“合法的邮箱”，否则打印'“不合法邮箱”

var str='15481264@qq.com'

if (str.indexOf('@')>=0)

{console.log('合法邮箱')

}else{console.log('不合法邮箱')}

toUpperCase() 将英文字母转大写

toLowerCase() 将英文字母转小写

练习：声明变量保存4个英文字母，无限循环弹出提示框，输入验证码（不区分大小写），如果输入正确，结束循环

//声明变量

var str='ABcd';

//无限循环弹出提示框

while (true){

//获取输入的内容

var str1=prompt('input chaeck str');

//输入的内容和验证码是否一致

if(str1.toUpperCase()==str.toUpperCase()){

//结束循环

break;

}

}

slice(start,end) 截取字符串，start开始的下标，end结束的下标，不包含end本身；如果为负数表示倒数；end为截取到最后

var str='javascript';

//截取字符串

console.log(str.slice(4))

console.log(str.slice(4,7))

console.log(str.slice(-3-1))

练习：声明变量保存邮箱要求分别截取出邮箱的用户名和域名

tom123@163.com

查找@的下标

0--下标 用户名

下标的下一位-最后 域名

var str='tom123@163.com';

//console.log(str.indexOf())

//截取用户名：0-@的下标

console.log(str.slice(0,str.indexOf('@')))

//截取域名：@的下标下一位-最后

console.log(str.slice(str.indexOf('@')+1))

substr(start,count) 截取字符串，start 开始的下标，count截取的长度，如果count为空截取到最后，start为负数表示倒数

var str='javascript';

//截取指定的长度

console.log(str.substr(4,3));

console.log(str.substr(-3,2))

练习声明变量保存身份证号，截取出年月日和性别；打印‘1998年06月20日性别女’

110236199806202589

var str='110236199806202589'

var year=str.substr(6,4)

var moth=str.substr(10,2)

var date=str.substr(12,2)

var sex=str.substr(-2,1)%2==0?'女':'男'

console.log(year+'年'+moth+'月'+date+'日'+sex)

substring(start,end) 截取字符串，start开始的下标，end结束的下标，如果end 为截取到最后；如果下标为负数，自动转为0

对比slice和substring

slice 中下标允许使用负数，substring会自动将负数转为0；slice中下表的顺序start小于end,substring中下标不分顺序

var str='javascript'

//截取字符串

console.log(str.substring(4))

console.log(str.substring(4,7))

//不允许写负数

console.log(str.substring(3,0))

练习将一个英文单词的首字母转大写，其余的字母转小写

heLLO🡪Hello

/\*var str='heLLO'

//截取首字母

var str1=str.substring(0,1).toUpperCase();

//截取第二个字符到最后

var str2=str.substring(1).toLowerCase();

console.log(str1+str2)\*/

split(sep) 将字符串按照指定的字符分隔成数组，sep是指定的字符

tom123@163.com

使用split分隔邮箱获取用户名和域名

var email='tom123@163.com';

console.log(email.split('@'))

（8）匹配模式（掌握）

作用：用于查找、替换字符串

match （value） 查找匹配的字符串，返回所有满足条件的元素，组成数组/china/ig

i🡪ignore忽略大小写

g🡪global 全局查找

search （value）用于查找满足条件的第一个字符的下标，如果找不到返回-1

replace(value1,value2)用于查找并替换，value1要查找的字符串，value2要替换的字符串

var str='China是世界上人口做多的国家,china的互联网非常发达，CHINA有好多的程序员，welcome to china';

//查找china出现了多少次

//console.log(str.match('china'))

//使用正则表达式的写法查找

//修饰符

//i-->ignore 忽略大小写

//g-->global 全局查找

//console.log(str.match(/china/ig))

//search 满足条件的第一个下标

//console.log(str.search(/chinas/i))

//查找并替换

console.log(str.replace(/china/ig,'中国'))

12、Math对象

不需要使用new创建，可以直接使用

PI 取圆周率

abs()取绝对值

floor()向下取整

ceil()向上取整

round()四舍五入取整

max()去一组数的最大值

min()去一组数的最小值

pow()求x的y次幂

random()取随机数

//圆周率

//console.log(Math.PI)

//绝对值

//console.log(Math.abs(18-20))//absolute

//取整

//console.log(parseInt(3.9))

//向下取整

//console.log(Math.floor(3.9))

//向上取整

//console.log(Math.ceil(3.9))

//四舍五入取整

//console.log(Math.round(3.9))

//获取最大值

//console.log(Math.max(23,6,78,9,45))

//获取最小值

//console.log(Math.min(23,6,78,9,45))

//获取x的y次幂

//console.log(Math.pow(4,2))

//获取随机 0~1 >=0 <1

//console.log(Math.random())var str=Math.random()\*10;

console.log(Math.floor(str))

练习：随机产生0-9之间的一个整数

随机0-1 \*10 0-10 向下取整

var str=Math.random()\*10;

console.log(Math.floor(str))

练习：创建一个数组，包含若干个元素，每次随机取出1个元素；数组有10个元素，随机取 0-9，作为下标

var str=['a','b','c','d','e','f','g','h','i']

console.log(str[Math.floor(Math.random()\*10)])

创建数组，包含所有英文字母（大写，小写）和数字（0-9），随机取4个元素（循环4次），放入一个新的数组push,如何不重复

/\*var arr=['a','b','c','e','f','g','h','i','j','k','l','n','m','o','p','q','r','s','t','u','v','w','x','y','z','A','B','C','E','F','G','H','I','J','K','L','N','M','O','P','Q','R','S','T','U','V','W','X','Y','Z']

var arr2=[0,1,2,3,4,5,6,7,8,9]

var arr3=arr.concat(arr2)

var arr4=[]

for (var i=0,sum=0;i<4 ;i++ )

{var index=Math.floor(Math.random()\*arr3.length)

arr4.push(arr3[index])

//在原数组中删除取到的元素

//pop shift splice

arr3.splice(index,1)

}

console.log(arr4)\*/

//获取每个单词，按空格将字符串分割为数组

//遍历数组，将每个单词首字母取出转大写，其他的字母取出转小写，拼接起来替换之前的字符串

//获取每个单词，按空格将字符串分割为数组

var arr='hOw aRE yOu'

var str=(arr.split(' '))

//历数组，将每个单词首字母取出转大写，其他的字母取出转小写

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

var first=str[i].slice(0,1).toUpperCase();

//取其余字母

var last=str[i].slice(1).toLowerCase();

str[i]=first+last

}

//数组arr转字符串

console.log(str.join(' '))

13、Date对象

用于对日期时间进行存储和计算

①创建Date对象

//获取当前的系统时间

var d1=new Date('2019/4/19 10:20:30');

var d2=new Date(2019,11,19,10,20,30)//第二个参数月份范围0-11

//获取当前的系统时间

var d3=new Date();

//存储计算机元年的毫秒数

//1970-1-10 0:0:0

var d4=new Date(0)

var d5=new Date(5480000000000)

console.log(d5) //存储距离计算机元年的毫秒数对应的日期时间2143-08-27T22:13:20.000Z

②获取Date对象中的日期时间

getFullYear获取对象中的年份

getMonth获取对象中的月份（0-11）

getDate获取天数

getHours获取小时数

getMinutes获取分钟数

getSeconds获取秒

getMilliseconds获取毫秒数

getDay获取星期数0-6

getTime获取距离计算机元年的毫秒数

③转为本地字符串

toLocaleDateString //年-月-日

toLocaleTimeString //时：分：秒

toLocaleString //年-月-日 时：分：秒

④修改Date对象中的日期时间

setFullYear()//修改年

setMonth()//修改月

setDate()///修改日

setTime()/

setHours()//修改小时

setMinutes()//修改秒

setMilliseconds()

/\*var now=new Date();

//获取年月日

console.log(now.toLocaleDateString())

//获取时分秒

console.log(now.toLocaleTimeString())

//完整

console.log(now.toLocaleString())\*/

/\*var d1=new Date('2019/4/18');

//复制对象

var d2=new Date(d1)

//把d2的年份修改为3年后的指

//获取当前的年份，在加3，把结果修改为当前的年份

d2.setFullYear(d2.getFullYear()+3);

//console.log(d1.getHours())

//修改年份为2020

d1.setFullYear(2020);

//修改月份

d1.setMonth(7);

//修改日

d1.setDate(3);

d1.setTime(1380000000000)

console.log(d1.toLocaleString())

console.log(d2.toLocaleString())\*/

练习：创建一个Date对象，保存员工的入职时间‘2019-4-18’3年后合同到期，计算到期时间；合同到期前一个月续签合同，如果周末，提前到周五，计算续签时间

var d1=new Date('2019/04/18');

var d2=new Date(d1)

d2.setFullYear(d2.getFullYear()+3)

var d3=new Date(d2)

d3.setMonth(d3.getMonth()-1)

console.log(d3.getDay())

if (d3.getDay==6){

d3.setDate(d3.getDate()-1);

}else if(d3.getDay==0){

d3.setDate(d3.getDate()-2)

}

console.log('入职时间'+d1.toLocaleString(d1))

console.log('到期时间'+d2.toLocaleString(d1))

console.log('续签时间'+d3.toLocaleString(d3))

14、Number对象

new Number()将数据转为数值型，返回对象

Number()将数据转为数值型，返回数值

Number.MAX\_VALUE获取Number的最大值

Number.MIN\_VALUE获取Number的最小值

toFixed()保留小数点后n位

toSntrig()将数值转为字符串，n转换的进制

var num=new Number(true)

var num2=Number('la')

//console.log(num2)

//console.log(num,typeof num)

//console.log(num+3)

//console.log(Number.MAX\_VALUE)

//console.log(Number.MIN\_VALUE)

var num3=2\*5\*3.14

console.log(num3.toFixed(1))

var num1=0.1+0.2

console.log(num1.toFixed(1))

//数字转字符串

var num4=27;

console.log(num4.toStrig(16))

15、Boolean对象

五种转为false

0 NaN '' undefined null

new Boolean()将数据转为布尔型，返回对象

Boolean()将数据转为布尔型，返回布尔型

!!隐式将数据转为布尔型

toString()将布尔型数据转为字符串

16、错误处理

Syntax Error:语法错误，错误的使用了中文，缺少括号等，出现后所有的代码都不执行。

ReferenceError：引用错误，使用了为声明的变量，属于运行时的错误，影响后续代码的执行

TypeError：类型错误，错误的使用了数据，例如把一个变量、数组当做函数使用，属于运行时的错误，会影响后续代码的执行。

RangeError:参数的使用超出了范围，属于运行时的错误，影响后续代码的执行

try{

尝试执行的代码，可能产生错误

}catch（）{

err捕获到错误

具体处理错误的内容

}

练习：声明一个变量add为赋值；尝试调用add函数；如果执行错误，给add赋值一个匿名函数，然后在调用

var add;

try

{

add();

}

catch (err){

add=function(){

console.log(3);

}

add();

}

17、ES6

ECMAScript6

（1）块级作用域

使用let 关键字声明的变量，只能在块级作用域下使用，不能被外部访问，不存在变量提升

块级作用域：{ } for、while、do、if…

防止污染全局

（2）箭头函数

是回调函数的另1种写法，和之前的匿名函数不完全一样

sort((a,b)=>{

raturn a-b

}))

如果箭头函数的函数体中只有一行代码，并且是return形式的，可以简化为sort((a,b)=>a-b)

//箭头函数

/\*var arr=[23,6,78,45]

//从小到大排序

//console.log(arr.sort(function(a,b){

// return a-b

//}))

console.log(arr.sort((a,b)=>{

return a-b;

}))

console.log(arr.sort((a,b)=> b-a))\*/

练习：创建函数add，传递两个参数，每个参数都是回调函数，在回调函数中返回一个数字；在函数add中计算两个数字相加得和

function add(a,d){

console.log(a()+d())

}

//add(function(){

//return 7},function(){

//return 8})

add(()=>7,()=>8)

ES6

(1)函数中的参数

ES6允许为参数设置默认值，如果没有传递实参，自动调用形参的默认值

（2）模板字符串

`一组字符串，可以直接写JS ${js表达式}`

var title='华为p30'

var color='红色'

//创建模板字符串

var str=`

商品的名称:${title}

商品的价格:${price>20?(price+5):price}

商品的颜色:${color}

${'欢迎购买'}

`;

console.log(str)\*/

创建一个对象，保存一个员工的对象（姓名，性别0/1，生日，工资）使用模板字符串打印员工的信息，性别0显示女，1显示男

var emp={name:'kaka',

sex:'1',

brithday:'1995-9-9',

salary:'6500'

}

str=`

姓名:${emp.name}

性别:${emp.sex==0?'女':'男'}

生日:${emp.brithday}

工资:${emp.salary}

`

console.log(str)

练习：要求计算2019年5月1日10点30分0秒距离2019年12月25日相差的天数，小时，分钟，秒

（/\*var str=new Date('2019/5/1 10:30:0')

var d=new Date(str)

d.getTime()

//console.log(d.toLocaleString())

var str2=new Date('2019/12/25')

var d1=new Date(str2)

d1.getTime()

//console.log(d.toLocaleString())

//console.log(d1.getTime())

//console.log(d1.toLocaleString())

d2=(d1-d)/1000

var d3=Math.floor(d2/(3600\*24))

var d4=Math.floor((d2-(d3\*3600\*24))/3600)

var d5=(d2-((d4\*3600)+(d3\*3600\*24)))/60

console.log(d3+'天'+d4+'时'+d5+'分')\*/）

var d1=new Date('2019/5/1 10:30:23')

var d2=new Date('2019/12/25')

//计算相差的毫秒数

var d=Math.abs(d1-d2)

//转成秒

d=Math.floor(d/1000)

//获取相差的天数

var day=Math.floor (d/(24\*3600))

//去除天数，剩下不满一天的秒数

var hours=d%(24\*3600)

//获取相差的小时

hours=Math.floor(hours/3600)

//去除小时，剩下不满一小时的秒数

var minutes=d%3600;

//获取相差得分钟

minutes=Math.floor(minutes/60)

//去除分钟，身下不满一分钟的秒数

var seconds=d%60

console.log(day+'天'+hours+'小时'+minutes+'分钟'+seconds+'秒')