JSB

1.转换函数

转换成字符串 toString()

转换成整型 parseInt(数据)

转换成浮点型 parseFloat(数据)

转换成Number Number(数据)

2. 全局函数（7个）

parseInt()

parseFloat()

Number()

isFinite()

encodeURI()

decodeURI()

eval()

JSC——数组的API

1. 创建数组的三种情况

var 变量=[];

var 变量=[值1,值2,...];

var 变量=new Array(n);

2. 数组转为字符串: 2种:

1. var str=String(arr)。

2. var str=arr.join("自定义连接符")

3. 拼接: 将其它元素或其它数组拼接到当前数组末尾,返回新数组

var newArr=arr1.concat(值1,值2,arr2,......)

4. 选取: 复制原数组中指定位置的元素组成新数组

var subArr=arr1.slice(starti,endi+1);

5. 删除:从arr中starti开始，删除n个元素

arr.splice(starti,n)

6. 插入:在arr中starti位置插入: 值1,值2,...

arr.splice(starti,0,值1,值2,...)

7. 替换:在arr中先删除starti位置后的n个元素

再在starti位置插入新元素

arr.splice(starti,n,值1,值2,...)

8. 翻转数组: arr.reverse();

9. 排序API: arr.sort();

10. 栈: 一端封闭，只能从另一端进出的数组

1. 从结尾出入栈:

入: arr.push(值) => arr[arr.length]=值

出: var last=arr.pop()

2. 从开头出入栈:

入: arr.unshift(值)

出: var first=arr.shift()

11.队列: 只能从一端进入，从另一端出的数组

1. 结尾入: 入: arr.push(值)

2. 开头出: var first=arr.shift()

——String的API

1. 都转大写str.toUpperCase()

都转小写 str.toLowerCase()

2. 获得指定位置的字符: str[i]

var char=str.charAt(i);

3.获得指定字符的unicode号:

var unicode=str.charCodeAt(i);

4.将unicode号反向转回文字

var char=String.fromCharCode(unicode);

5.选取子字符串: str.slice(starti,endi+1)

6. 查找一个固定的关键词出现的位置:

var i=str.indexOf("关键词",fromi)

7. 专门找最后一个关键词的位置:

var i=str.lastIndexOf("关键词")

8. 使用正则表达式查找指定的一类关键词的位置:

var i=str.search(/正则表达式/);

9. 使用正则表达式查找指定的一类关键词的内容:

var arr=str.match(/正则/ig);

10. 即找所有关键词内容，又找每个关键词的位置

reg.exec();

11. 基本替换:

str=str.replace(/正则/ig,“替换值”);

12. 切割: 将原字符串，按指定字符，分隔为多个子字符串

var substrs=str.split(/正则/)

13. 验证: 检查字符串是否完全符合正则表达式的要求

var bool=reg.test(待检测字符串)

14. 查找关键词: 第四种情况: 即找内容，又找位置

var arr=reg.exec(待查找的完整字符串)

15. 取整:

上取整: Math.ceil(num)

下取整: Math.floor(num)

四舍五入取整: Math.round(num)

16. 乘方: Math.pow(底数, 幂)

开平方: Math.sqrt(n)

17. 最大值和最小值:

Math.max(值1,值2,...);

Math.min(值1,值2,...);

18. 随机数:

0<=r<1 随机小数 ： Math.random()

从min~max之间取随机整数:

Math.floor(Math.random()\*(max-min+1)+min)

从0~n之间去随机:

Math.floor(Math.random()\*(n+1));

19. 创建日期对象，并自动获得当前客户端系统时间

var now=new Date();

20. 当地时间的完整时间格式：date.toString()

当地时间的简化版格式 ： date.toLocaleString()

当地时间的日期部分：date.toLocaleDateString()

当地时间的时间部分: date.toLocaleTimeString()

标准时区的标准时间 : date.toGMTString()

1. 判断自有属性或共有属性:

自有: var bool=obj.hasOwnProperty("属性名")

共有: 不是自有! 且 子对象可访问到！

1. 用判断father是否是child的父级对象的方式判断原型对象

var bool=father.isPrototypeOf(child)

1. 判断构造函数: obj.constructor==Array

4.获得class属性值: Object.prototype.toString();-

5. 修改单个对象的继承关系: obj.\_\_proto\_\_=father;

6. 批量修改多个对象的继承关系: obj.prototype=father

7.数据属性:

四大特性:{

value: 值,

writable: true/false,

enumerable: true/false,

configurable: true/false

}

获取四大特性:

var attrs=Object.getOwnPropertyDescriptor(obj,"属性名");

attrs:{ value: , writable: , ...}

修改四大特性:

Object.defineProperty(obj,"属性名",{

特性: 值,

configurable:false

})

同时修改多个属性的特性:

Object.defineProperties(obj,{

属性名1:{

四大特性:

},

...

})

8.访问器属性:

四大特性: {

get:function(){ return 属性值; },

set:function(value){ 属性名=value;

//否则

报错

},

enumerable: true/false,

configurable: true/false

}

添加访问器属性:

Object.defineProperty(obj,"属性名",{

get:function(){return xxx},

set:function(value){ xxx=value},

enumerable: true/false,

configurable: true/false

});

9.判断obj是否可扩展: Object.isExtensible(obj)

10.阻止对obj的任何扩展: Object.preventExtensions(obj)

11.判断obj是否密封: Object.isSealed(obj)

12.设置obj密封: Object.seal(obj)

13.判断obj是否被冻结: Object.isFrozen(obj)

14.冻结obj: Object.freeze(obj)

15.仅创建子对象: var child=Object.create(father);

16.创建子对象时，为子对象扩展自有属性:

　　　var child=Object.create(father,{

属性名:{

四大特性

},

...

})

17.检查数组中每个元素是否都满足条件: arr.every()

18.检查数组中是否包含满足条件的元素: arr.some()

19.对原数组中每个元素执行相同的操作: arr.forEach():