Js 知识点

JavaScript 中如何检测一个变量是一个 String 类型？请写出函数实现

方法 1、

```
function isString(obj){
   return typeof(obj) === "string"? true: false;
   // returntypeof obj === "string"? true: false;
}
```

方法 2、

```
function isString(obj){
   return obj.constructor === String? true: false;
}
```

方法 3.

```
function isString(obj){
    return Object.prototype.toString.call(obj) === "[object String]"?true:false;
}
```

如：

```
var isstring = isString('xiaoming');
console.log(isstring);  // true
```

请用 js 去除字符串空格？

方法一：使用 replace 正则匹配的方法

去除所有空格: str = str.replace(/\s*/g,"");

去除两头空格: str = str.replace(/^\s*|\s*$/g,"");

去除左空格： str = str.replace(/^\s*/, “”);

去除右空格： str = str.replace(/(\s*$)/g, "");

str 为要去除空格的字符串，实例如下：

var str = " 23 23 ";

var str2 = str.replace(/\s*/g,"");

console.log(str2); // 2323

方法二：使用 str.trim()方法

str.trim()局限性：无法去除中间的空格，实例如下：

var str = " xiao ming ";

```
var str2 = str.trim();
console.log(str2);    //xiao    ming
```

同理，str.trimLeft()，str.trimRight()分别用于去除字符串左右空格

你如何获取浏览器 URL 中查询字符串中的参数？

测试地址为：http://www.runoob.com/jquery/misc-trim.html?channelid=12333&name=xiaoming&age=23

```
function showWindowHref(){
    var sHref = window.location.href;
    var args = sHref.split('?');
    if(args[0] == sHref){
        return "";
    }
    var arr = args[1].split('&');
    var obj = {};
    for(var i = 0;i< arr.length;i++){
        var arg = arr[i].split('=');
        obj[arg[0]] = arg[1];
    }
    return obj;
}
var href = showWindowHref(); // obj
console.log(href['name']); // xiaoming
```

判断一个字符串中出现次数最多的字符，统计这个次数

```
var str = 'asdfssaaasasasasaa';
var json = {};
for (var i = 0; i < str.length; i++) {
    if(!json[str.charAt(i)]){
       json[str.charAt(i)] = 1;
    }else{
       json[str.charAt(i)]++;
    }
};
var iMax = 0;
var iIndex = '';
for(var i in json){
    if(json[i]>iMax){
         iMax = json[i];
         iIndex = i;
    }
```

```
}
console.log('出现次数最多的是:'+iIndex+'出现'+iMax+'次');
```

js 字符串操作函数

concat() – 将字符串或者数组组合起来，返回一个新的字符串。

indexOf() – 返回字符串中一个子串第一处出现的索引。如果没有匹配项，返回 -1 。

charAt() – 返回指定位置的字符。

lastIndexOf() – 返回字符串中一个子串最后一处出现的索引，如果没有匹配项，返回 -1 。

match() – 检查一个字符串是否匹配一个正则表达式。

substr() 函数 -- 返回从 string 的 startPos 位置，长度为 length 的字符串

substring() – 返回字符串的一个子串。传入参数是起始位置和结束位置。

slice() – 提取字符串的一部分，并返回一个新字符串。

replace() – 用来查找匹配一个正则表达式的字符串，然后使用新字符串代替匹配的字符串。

search() – 执行一个正则表达式匹配查找。如果查找成功，返回字符串中匹配的索引值。否则返回 -1 。

split() – 通过将字符串划分成子串，将一个字符串做成一个字符串数组。

length – 返回字符串的长度，所谓字符串的长度是指其包含的字符的个数。

toLowerCase() – 将整个字符串转成小写字母。

toUpperCase() – 将整个字符串转成大写字母。

Array 对象方法

concat() 连接两个或更多的数组，并返回结果。 // arr.concat(arr2)

join() 把数组的所有元素放入一个字符串。元素通过指定的分隔符进行分隔。 arr.join(',')

pop() 删除并返回数组的最后一个元素。

```
var arr = [2,3,4,5];var arr2 = arr.pop();
console.log(arr2); // 删除的数组的最后一个元素为：5
console.log(arr);  // 删除元素之后的数组为：[2, 3, 4]
```

shift() 删除并返回数组的第一个元素

```
var arr = [2,3,4,5];var arr2 = arr.shift();
console.log(arr2); // 删除的数组的第一个元素为：2
console.log(arr);  // 删除元素之后的数组为：[3, 4, 5]
```

push() 向数组的末尾添加一个或更多元素，并返回新的长度。

```
var arr = [2,3,4,5];var arr2 = arr.push(6);
console.log(arr2);  // 返回的数组长度：5
console.log(arr);  // [2, 3, 4, 5, 6]
```

unshift() 向数组的开头添加一个或更多元素，并返回新的长度。

```
var arr = ['xiao','ming','qiqi','aiming'];var arr1 = arr.unshift('lang');
console.log(arr1);  // 返回的数组的长度：  5
console.log(arr);  //向数组开头添加元素返回的结果：["lang", "xiao", "ming", "qiqi", "aiming"]
```

reverse() 颠倒数组中元素的顺序。

```
var arr = [2,3,4,5];
arr.reverse();
console.log(arr);   //  [5, 4, 3, 2]
```

slice() 从某个已有的数组返回选定的元素

```
var arr = [2,3,4,5];var arr2 = arr.slice(1,3);
console.log(arr2);  // 截取区间返回的数组为：[3, 4]
console.log(arr);  // [2, 3, 4, 5]
```

sort() 对数组的元素进行排序

splice() 删除元素，并向数组添加新元素。

toSource() 返回该对象的源代码。

toString() 把数组转换为字符串，并返回结果。

toLocaleString() 把数组转换为本地数组，并返回结果。

编写一个方法 去掉一个数组的重复元素

方法一：

```
var arr = [0,2,3,4,4,0,2];
var obj = {};
var tmp = [];
for(var i = 0 ;i< arr.length;i++){
   if( !obj[arr[i]] ){
      obj[arr[i]] = 1;
      tmp.push(arr[i]);
   }
}
console.log(tmp);
```

结果如下： [0, 2, 3, 4]

方法二：

```
var arr = [2,3,4,4,5,2,3,6],
   arr2 = [];
for(var i = 0;i< arr.length;i++){
   if(arr2.indexOf(arr[i]) < 0){
```

```
        arr2.push(arr[i]);
    }
}
console.log(arr2);
```

结果为：[2, 3, 4, 5, 6]

方法三：

```
var arr = [2,3,4,4,5,2,3,6];
var arr2 = arr.filter(function(element,index,self){
return self.indexOf(element) === index;
});
console.log(arr2);
```

结果为：[2, 3, 4, 5, 6]

求数组的最值？

方法一：求数组最大值：Math.max.apply(null,arr);

```
var arr = [3,43,23,45,65,90];
var max = Math.max.apply(null,arr);console.log(max);// 90
```

求数组最小值：Math.min.apply(null,arr);

```
var arr = [3,43,23,45,65,90];
var min = Math.min.apply(null,arr);console.log(min);// 3
```

方法二：Array.max = function(arr){} / Array.min = function(arr){}

```
var array = [3,43,23,45,65,90];

Array.max = function( array ){
   return Math.max.apply( Math, array );
};
Array.min = function( array ){
   return Math.min.apply( Math, array );
};
var max = Array.max(array);console.log(max);  // 90
var min = Array.min(array);console.log(min);  // 3
```

方法三：Array.prototype.max = function(){};Array.prototype.min = function(){};

求数组最大值(基本思路：将数组中的第一个值赋值给变量 max ,将数组进行循环与 max 进行比较，将数组中的大值赋给 max,最后返回 max;)

```
var arr = [3,43,23,45,65,90];
Array.prototype.max = function() {
    var max = this[0];
    var len = this.length;
    for (var i = 0; i < len; i++){
        if (this[i] > max) {
            max = this[i];
        }
    }
    return max;
}
var max = arr.max();
console.log(max);  // 90
```

求数组最小值：

```
var arr = [3,43,23,45,65,90];
Array.prototype.min = function() {
    var min = this[0];
    var len = this.length;
    for(var i = 0;i< len;i++){
        if(this[i] < min){
            min = this[i];
        }
    }
    return min;
}
var min = arr.min();console.log(min);  // 3
```

数组排序相关

数组由小到大进行排序：sort,sortnum

```
var arr = [3,43,23,45,65,90];
function sortnum(a,b){
    return a-b;
}
arr = arr.sort(sortnum);
console.log(arr);
// [3, 23, 43, 45, 65, 90]
```

数组由大到小进行排序：sort,sortnum;

```
var arr = [3,43,23,45,65,90];
function sortnum(a,b){
    return b-a;
}
arr = arr.sort(sortnum);
console.log(arr);
// [90, 65, 45, 23, 43, 3]
```

数组的翻转

方法一：

```
var arr = [1,2,3,4];
var arr2 = [];
while(arr.length) {
    var num = arr.pop(); //删除数组最后一个元素并返回被删除的元素
    arr2.push(num);
}
console.log(arr2);
// [4, 3, 2, 1]
```

方法二：

```
var arr = [1,2,3,4];
var arr2 = [];
while(arr.length){
    var num = arr.shift(); //删除数组第一个元素并返回被删除的元素
    arr2.unshift(num);
}
console.log(arr2);
```

$(this) 和 this 关键字在 jQuery 中有何不同？

$(this) 返回一个 jQuery 对象，你可以对它调用多个 jQuery 方法，比如用 text() 获取文本，用 val() 获取值等等。

而 this 代表当前元素，它是 JavaScript 关键词中的一个，表示上下文中的当前 DOM 元素。你不能对它调用 jQuery 方法，直到它被 $() 函数包裹，例如 $(this)。

jquery 怎么移除标签 onclick 属性？

获得 a 标签的 onclick 属性：$("a").attr("onclick")

删除 onclick 属性：$("a").removeAttr("onclick");

设置 onclick 属性：$("a").attr("onclick","test();");

jquery 中 addClass,removeClass,toggleClass 的使用。

$(selector).addClass(class)：为每个匹配的元素添加指定的类名

$(selector).removeClass(class)：从所有匹配的元素中删除全部或者指定的类，删除 class 中某个值；

$(selector).toggleClass(class)：如果存在（不存在）就删除（添加）一个类

$(selector).removeAttr(class);删除 class 这个属性；

jQuery 中的 Delegate()函数有什么作用？

delegate()会在以下两个情况下使用到：

如果你有一个父元素，需要给其下的子元素添加事件，这时你可以使用 delegate()了，代码如下：

```
$("ul").delegate("li", "click", function(){ $(this).hide(); });
```

$(document).ready()方法和 window.onload 有什么区别？

(1)、window.onload 方法是在网页中所有的元素(包括元素的所有关联文件)完全加载到浏览器后才执行的。

(2)、$(document).ready() 方法可以在 DOM 载入就绪时就对其进行操纵，并调用执行绑定的函数。

jquery 中$.get()提交和$.post()提交有区别吗？

相同点：都是异步请求的方式来获取服务端的数据；

异同点：

1、请求方式不同：$.get() 方法使用 GET 方法来进行异步请求的。$.post() 方法使用 POST 方法来进行异步请求的。

2、参数传递方式不同：get 请求会将参数跟在 URL 后进行传递，而 POST 请求则是作为 HTTP 消息的实体内容发送给 Web 服务器的，这种传递是对用户不可见的。

3、数据传输大小不同：get 方式传输的数据大小不能超过 2KB 而 POST 要大的多

4、安全问题： GET 方式请求的数据会被浏览器缓存起来，因此有安全问题。

px 和 em 的区别

相同点：px 和 em 都是长度单位；

异同点：px 的值是固定的，指定是多少就是多少，计算比较容易。em 得值不是固定的，并且 em 会继承父级元素的字体大小。浏览器的默认字体高都是 16px。所以未经调整的浏览器都符合: 1em=16px。那么 12px=0.75em, 10px=0.625em。

sessionStorage 、localStorage 和 cookie 之间的区别

共同点：用于浏览器端存储的缓存数据

不同点：(1)、存储内容是否发送到服务器端：当设置了 Cookie 后，数据会发送到服务器端，造成一定的宽带浪费；

web storage,会将数据保存到本地，不会造成宽带浪费；

(2)、数据存储大小不同：Cookie 数据不能超过 4K,适用于会话标识；web storage 数据存储可以达到 5M;

(3)、数据存储的有效期限不同：cookie 只在设置了 Cookid 过期时间之前一直有效，即使关闭窗口或者浏览器；sessionStorage,仅在关闭浏览器之前有效；localStorage,数据存储永久有效；

(4)、作用域不同：cookie 和 localStorage 是在同源同窗口中都是共享的；sessionStorage 不在不同的浏览器窗口中共享，即使是同一个页面；

Storage 与 Cookie 相比存在的优势：

(1)、存储空间更大：IE8 下每个独立的存储空间为 10M，其他浏览器实现略有不同，但都比 Cookie 要大很多。

(2)、存储内容不会发送到服务器：当设置了 Cookie 后，Cookie 的内容会随着请求一并发送的服务器，这对于本地存储的数据是一种带宽浪费。而 Web Storage 中的数据则仅仅是存在本地，不会与服务器发生任何交互。

(3)、更多丰富易用的接口：Web Storage 提供了一套更为丰富的接口，如 setItem,getItem,removeItem,clear 等,使得数据操作更为简便。cookie 需要自己封装。

(4)、独立的存储空间：每个域（包括子域）有独立的存储空间，各个存储空间是完全独立的，因此不会造成数据混乱。

Ajax 的优缺点及工作原理？

优点

1.减轻服务器的负担,按需取数据,最大程度的减少冗余请求

2.局部刷新页面,减少用户心理和实际的等待时间,带来更好的用户体验

3.基于 xml 标准化,并被广泛支持,不需安装插件等,进一步促进页面和数据的分离

缺点

1.AJAX 大量的使用了 javascript 和 ajax 引擎,这些取决于浏览器的支持.在编写的时候考虑对浏览器的兼容性.

2.AJAX 只是局部刷新,所以页面的后退按钮是没有用的.

3.对流媒体还有移动设备的支持不是太好等

浏览器是如何渲染页面的？

1.解析 HTML 文件，创建 DOM 树。

自上而下，遇到任何样式（link、style）与脚本（script）都会阻塞（外部样式不阻塞后续外部脚本的加载）。

2.解析 CSS。优先级：浏览器默认设置<用户设置<外部样式<内联样式<HTML 中的 style 样式；

3.将 CSS 与 DOM 合并，构建渲染树（Render Tree）

4.布局和绘制，重绘（repaint）和重排（reflow）

请说出四种减低页面加载时间的方法

压缩 css、js 文件

合并 js、css 文件，减少 http 请求

外部 js、css 文件放在最底下

减少 dom 操作，尽可能用变量替代不必要的 dom 操作

前端开发，如何提高页面性能优化？

内容方面：

1.减少 HTTP 请求 (Make Fewer HTTP Requests)

2.减少 DOM 元素数量 (Reduce the Number of DOM Elements)

3.使得 Ajax 可缓存 (Make Ajax Cacheable)

针对 CSS：

1.把 CSS 放到代码页上端 (Put Stylesheets at the Top)

2.从页面中剥离 JavaScript 与 CSS (Make JavaScript and CSS External)

针对 JavaScript ：

1. 脚本放到 HTML 代码页底部 (Put Scripts at the Bottom)

2. 从页面中剥离 JavaScript 与 CSS (Make JavaScript and CSS External)

4. 移除重复脚本 (Remove Duplicate Scripts)

面向图片(Image)：

1.优化图片

2 不要在 HTML 中使用缩放图片

3 使用恰当的图片格式

如何理解闭包？

1、定义和用法：当一个函数的返回值是另外一个函数，而返回的那个函数如果调用了其父函数内部的其它变量，如果返回的这个函数在外部被执行，就产生了闭包。

2、表现形式：使函数外部能够调用函数内部定义的变量。

3、实例如下：

(1)、根据作用域链的规则，底层作用域没有声明的变量，会向上一级找，找到就返回，没找到就一直找，直到 window 的变量，没有就返回 undefined。这里明显 count 是函数内部的 flag2 的那个 count 。

```
var count=10;   //全局作用域 标记为 flag1
function add(){
    var count=0;    //函数全局作用域 标记为 flag2
    return function(){
        count+=1;   //函数的内部作用域
        alert(count);
    }
}
var s = add()
s();//输出 1
s();//输出 2
```

4、变量的作用域

要理解闭包，首先必须理解 Javascript 特殊的变量作用域。

变量的作用域分类：全局变量和局部变量。

特点：

1、函数内部可以读取函数外部的全局变量；在函数外部无法读取函数内的局部变量。

2、函数内部声明变量的时候，一定要使用 var 命令。如果不用的话，你实际上声明了一个全局变量！

5、使用闭包的注意点

1）滥用闭包，会造成内存泄漏：由于闭包会使得函数中的变量都被保存在内存中，内存消耗很大，所以不能滥用闭包，否则会造成网页的性能问题，在 IE 中可能导致内存泄露。解决方法是，在退出函数之前，将不使用的局部变量全部删除。

2）会改变父函数内部变量的值。所以，如果你把父函数当作对象（object）使用，把闭包当作它的公用方法（Public Method），把内部变量当作它的私有属性（private value），这时一定要小心，不要随便改变父函数内部变量的值。

什么是跨域？

由于浏览器同源策略，凡是发送请求 url 的协议、域名、端口三者之间任意一与当前页面地址不同即为跨域。存在跨域的情况：

网络协议不同，如 http 协议访问 https 协议。

端口不同，如 80 端口访问 8080 端口。

域名不同，如 qianduanblog.com 访问 baidu.com。

子域名不同，如 abc.qianduanblog.com 访问 def.qianduanblog.com。

域名和域名对应 ip,如 www.a.com 访问 20.205.28.90.

简述同步和异步的区别

同步就是指一个进程在执行某个请求的时候，若该请求需要一段时间才能返回信息，那么这个进程将会一直等待下去，直到收到返回信息才继续执行下去；
异步是指进程不需要一直等下去，而是继续执行下面的操作，不管其他进程的状态。当有消息返回时系统会通知进程进行处理，这样可以提高执行的效率。