文档说明：本文档用来记录一些平时学习到的东西

2014.11.6

// javascript

function show(a){

alert(2 < a < 3);

}

show(1);

输出结果居然是true，不理解，查了资料记录下原理：

jS是有符号优先级的，如果相同符号，则自左向右计算

2<1<3

这个表达式的计算过程：

1. 首先计算2<1，结果为false，

2. 然后计算 false<3；

3. js自动转换数据类型以做比较，false就是0

4. 0<3的结果自然是true

如果是2<(1<3)结果就不一样了，试了下输出结果是false

针对jQuery的优化方法：

基于Class的选择性的性能相对于Id选择器开销很大，因为需遍历所有DOM元素。

\*频繁操作的DOM，先缓存起来再操作。用Jquery的链式调用更好。

比如：var str=$("a").attr("href");

\*for (var i = size; i < arr.length; i++) {}

for 循环每一次循环都查找了数组 (arr) 的.length 属性，在开始循环的时候设置一个变量来存储这个数字，可以让循环跑得更快：

for (var i = size, length = arr.length; i < length; i++) { }

jQuery 中如何将数组转化为json字符串，然后再转化回来？

jQuery中没有提供这个功能，所以你需要先编写两个jQuery的扩展：

$.fn.stringifyArray = function(array) {

return JSON.stringify(array)

}

$.fn.parseArray = function(array) {

return JSON.parse(array)

}

然后调用：

$("").stringifyArray(array)

JavaScript 伪协议

<a href="JavaScript:popUp('http://padding.me')">Example</a>

这条语句在支持 "JavaScript:" 伪协议的浏览器中正常运行，较老的浏览器则会去尝试打开那个链接但失败，支持这种伪协议但禁用了 JavaScript 功能的浏览器则会什么也不做。

"JavaScript:" 伪协议调用 JavaScript 代码的做法非常不好。

性能考虑

尽量少访问 DOM 和尽量减少标记

不管是什么时候，只要是查询 DOM 中的某些元素，浏览器都会搜索整个 DOM 树，从中查找可能匹配的元素。

而过多不必要的元素标记只会增加 DOM 树的规模，进而增加遍历 DOM 树以查找特定元素的时间。

合并和放置脚本

将多个脚本合并为一个脚本，以减少浏览器向服务器端发送求，将脚本放在 `</body>`之前，为什么呢？这样 DOM 操作可以在浏览器将网页全部加载渲染完毕之后开始进行，以减少浏览器对网页的重绘，提升网页加载速度和性能。

压缩脚本

将脚本压缩减小脚本体积，减缓带宽压力，更快的将脚本下载下来

异步加载的方式有哪些？

(1) defer，只支持IE

(2) async：

(3) 创建script，插入到DOM中，加载完毕后callBack

new操作符具体干了什么呢?

1、创建一个空对象，并且 this 变量引用该对象，同时还继承了该函数的原型。

2、属性和方法被加入到 this 引用的对象中。

3、新创建的对象由 this 所引用，并且最后隐式的返回 this 。

var obj = {};

obj.\_\_proto\_\_ = Base.prototype;

Base.call(obj);

js延迟加载的方式有哪些？

defer和async、动态创建DOM方式（用得最多）、按需异步载入js

defer的定义和用法

defer属性规定是否对脚本执行进行延迟，直到页面加载为止。

有的 javascript 脚本 document.write 方法来创建当前的文档内容，其他脚本就不一定是了。

如果您的脚本不会改变文档的内容，可将 defer 属性加入到 <script> 标签中，以便加快处理文档的速度。因为浏览器知道它将能够安全地读取文档的剩余部分而不用执行脚本，它将推迟对脚本的解释，直到文档已经显示给用户为止。浏览器支持方面，只有 Internet Explorer 支持 defer 属性。至于js中defer的用法，可以参看这篇：<http://www.cnblogs.com/happygx/archive/2011/05/24/2055023.html>

Js中defer的作用