**Sizzle是一个纯JavaScript CSS选择器引擎。jquery1.3开始使用sizzle，Sizzle一反传统采取了相反的Right To Left的查询匹配方式,效率提高.Sizzle是jQuery作者John Resig新写的DOM选择器引擎,速度号称业界第一.Sizzle完全独立于jQuery，若不想用jQuery,可只用Sizzle实现,压缩3K多。**

**第一，多用ID选择器 , 总是从#id选择器来继承**

多用ID选择器，这是一个明智的选择。即使添加"在"ID选择器，也可以从父级元素中添加一个ID选择器，这样就会缩短节点访问的路程。

这是jQuery选择器的一条黄金法则。jQuery选择一个元素最快的方法就是用ID来选择了

$('#content').hide();

或者从ID选择器继承来选择多个元素

$('#content p').hide();

再如

$("#container").find("div.robotarm");

效率更高，那是因为$("#container")是不需要经过Sizzle选择器引擎处理的，jquery对仅含id选择器的处理方式是直接使用了浏览器的内置函数document.getElementById()，所以其效率是非常之高的。

##### 特征性

使一个选择器的右边更具有特征，相对而言，选择器的左边可以少一些特征性。

**// unoptimized　　优化前**

**$( "div.data .gonzalez" );**

**// optimized 优化后**

**$( ".data td.gonzalez" );**

　　再选择器的右边尽可能使用"tag.class"类型的选择符，在选择器的左边直接使用标签选择符或类选择符即可。

　　(类似于css选择器，其匹配算法是从右至左的)

##### 避免过度的约束

$(".data table.attendees td.gonzalez");

// better: drop the middle if possible 尽可能移除掉中间的

$(".data td.gonzalez");

一个更为“扁平”的DOM结构，会使得选择器引擎在寻找元素时经过的层次数更少，因此这样也是有利于提高选择器的性能的。

##### 避免使用全局的选择器

一个会被在多处地方成功匹配的选择器可能会消耗更多的性能

$(".buttons > \*"); // extremely expensive

$(".buttons").children(); // much better

$(".gender :radio"); // implied universal selection

$(".gender \*:radio"); // same thing, explicit now

$(".gender input:radio"); // much better

**第二，少直接使用Class选择器。**

可以使用复合选择器，例如使用tag.class代替.class。文档的标签是有限的，但是类可以拓展标签的语义，那么大部分情况下，使用同一个类的标签也是相同的。

当然，应该摒除表达式中的冗余部分，对于不必要的复合表达式就应该进行简化。例如，对于#id2 #id1 或者 tag#id1表达式，不妨直接使用#id1即可，因为ID选择器是惟一的，执行速度最快。使用复合选择器，相反会增加负担。

在class前面使用tag

jQuery中**第二快**的选择器就是tag选择器（如$(‘head’)），因为它和直接来自于原生的Javascript方法getElementByTagName()。所以最好总是用tag来修饰class(并且不要忘了就近的ID)

var receiveNewsletter = $('#nslForm input.on');

jQuery中**class选择器是最慢的**，因为在IE浏览器下它会遍历所有的DOM节点。尽量避免使用class选择器。也不要用tag来修饰ID。下面的例子会遍历所有的div元素来查找id为’content’的那个节点：

var content = $('div#content'); **// 非常慢，不要使用**

用ID来修饰ID也是**画蛇添足**：

var traffic\_light = $('#content #traffic\_light'); **// 非常慢，不要使用**

**第三，多用父子关系，少用嵌套关系。**

例如，使用parent>child代替parent child。因为">"是child选择器，只从子节点里匹配，不递归。而" "是后代选择器，递归匹配所有子节点及子节点的子节点，即后代节点。

**$parent.find('.child')**

这条是最快的语句。.find()方法会调用浏览器的原生方法（getElementById，getElementByName，getElementByTagName等等），所以速度较快。

**$('.child', $parent)**

给定一个DOM对象，然后从中选择一个子元素。jQuery会自动把这条语句转成$.parent.find('child')，这会导致一定的性能损失。它比最快的形式慢了5%-10%。

$('.child', $('#parent'))

jQuery内部会将这条语句转成$('#parent').find('.child')，比最快的形式慢了23%。

$parent.children('.child')

这条语句在jQuery内部，会使用$.sibling()和javascript的nextSibling()方法，一个个遍历节点。它比最快的形式大约慢50%

$('#parent > .child')

 jQuery内部使用Sizzle引擎，处理各种选择器。Sizzle引擎的选择顺序是从右到左，所以这条语句是先选.child，然后再一个个过滤出父元素#parent，这导致它比最快的形式大约慢70%。

$('#parent .child')

这条语句与上一条是同样的情况。但是，上一条只选择直接的子元素，这一条可以于选择多级子元素，所以它的速度更慢，大概比最快的形式慢了77%。