字符串拼接-不同方式下的性能问题纠错

这是我当初在学习javascript初期时总结的一句话：

数组是一种性能出色的数据结构，因为它是一段线性分配的内存。不幸的是，js的数组只是array-like的对象而已，它比一个真正的数组明显慢。尽管如此，在大量拼接字符串的场景下，我们也应该使用这种array-like的数组去操作，因为字符串一旦创建不可更改的特性，会更加消耗性能。

刚在javascript语言精粹中看到对Array.join()方法的这样一段阐述：

如果你想把大量的字符串片段组装成一个字符串，把这些片段放到一个数组中并用join方法连接起来通常比用+元素运算符连接这些片段要快。

这段话是符合当时IE6/7浏览器占据大量市场份额条件下的。我结合下面译者的译注和上网查找得出的结论大概是这样的：对于IE/7，使用Aarry.join()连接大量字符串的效率确实优于使用+元素运算符，归根到底是由于IE6/7的垃圾回收机制的缺陷导致的。但目前主流的浏览器，包括IE8以后的版本，都对垃圾回收机器进行了改造，对+元素运算符连接字符串做了特别优化，性能已经显著高于Array.join()。所以，我们可以得出这样的结论，在目前的大多数情况下，建议链接字符串首选使用+元素运算符。

其次，在我们循环字符串拼接是，应该把循环获取的length或者变量保存到for(;;)第一个分号的前面，进行缓存，防止极大循环次数情况下的循环获取变量长度有损效率。下面是我在chrome浏览器下的测试贴图：