# 加载和执行

* 将所有的SCRIPT标签尽量放在body底部的位置，尽量减少对整个页面下载的影响。
* 尽量少的SCRIPT标签，原因就是因为SCRIPT标签会阻塞界面。Yahoo Combo,TEngine Contact.
* 保持JS文件小，并限制HTTP请求数量，只能算第一步，如果功能越多，一个大的JS文件下载下来也会阻塞界面。需要向页面中逐步添加Javascript文件。非阻塞的秘密在于，等页面加载完成后，再加载JS源码，这意味着

一般在Window的load事件发出之后下载代码。

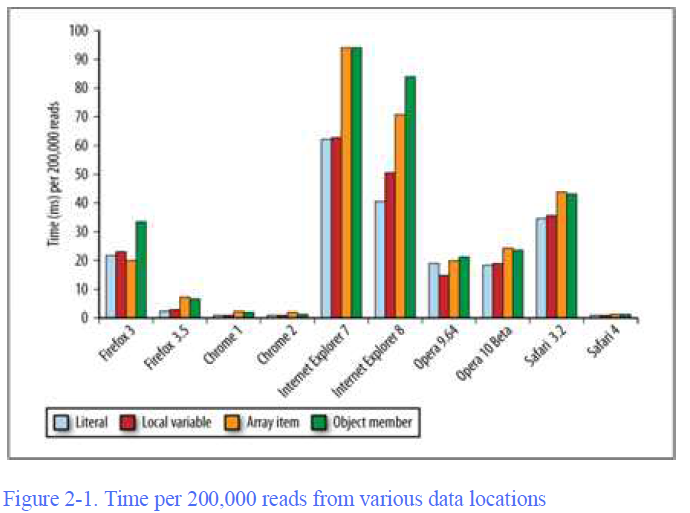
* 增加defer
* 动态脚本加载.可以创建多个动态脚本，但是这些脚本的执行顺序并不是按照指定的顺序。因此会有模块的依赖。
* XHR下载远程脚本处理
* 推荐的方式：包含动态加载JS代码，加载页面初始化所需要的除JS之外的部分。用它来加载其他的代码。

框架：

* LazyLoad
* LABjs

# 数据访问

## 作用域



直接量，本地变量，数组和对象在各个浏览器下的测试情况。

* 尽量使用直接量、本地变量，限制使用数组和对象的访问。

每个Javascript函数都是一个对象，他是一个Function的实例，既然他是一个对象，就像其他对象一样，有可以访问的属性，而这些属性中，有些只能被Javascript引擎访问到，不能被外界程序访问。其中一个属性就是SCOPE.

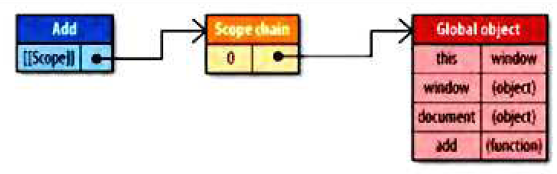
SCOPE属性包含了一个函数对象被创建时的作用域中对象的集合，此集合被称作是函数的作用域链。作用域链中的每个对象以键值的形式存在，称作为可变对象。

Function add(num1, num2){

Var sum = num1 + num2;

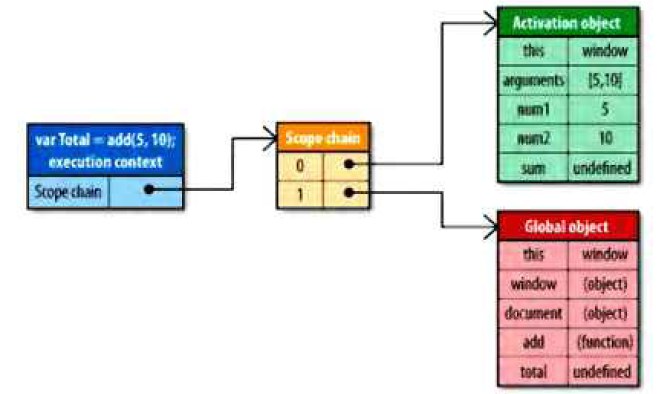
Return sum;

}



函数SCOPE与全局对象之间的关系

运行add函数时，会创建运行期上下文，他定义了一个函数运行时的环境，每次函数的运行，上下文都是不一样的，多次调用会导致多次创建上下文运行环境。运行期上下文创建时，相应的作用域链也就创建好了。运行期上下文中包含了当前的激活对象，他包括局部变量，命名参数，参数集合，this，此对象会被加入到作用域链的最前端。如下所示：



函数运行过程中，会从这些作用域中查找对应的变量值，激活对象中找不到，就回去全局变量中搜索，实际上全局变量永远处于作用域链的最后一个位置。所以**解决方案是**：

将全局变量的引用存储在一个局部变量中，然后使用这个局部变量来代替全局变量。40