jQuery的开篇声明里有一段非常重要的话:jQuery是为了改变javascript的编码方式而设计的.  
从这段话可以看出jQuery本身并不是UI组件库或其他的一般AJAX类库.  
jQuery改变javascript编码方式!  
那么它是如何实现它的声明的呢?这里,用以下的一段简短的使用流程:  
1)查找(创建)jQuery对象:$(”selector”);  
2)调用jQuery对象的方法完成我们需要完成的工作:$(”selector”).doOurWork();  
ok,jQuery就是以这种可以说是最简单的编码逻辑来改变javascript编码方式的.这两个步骤是jQuery的编码逻辑核心!  
要实现这种简洁编码方式,创建jQuery对象这一环节至关重要.因此,jQuery的dom元素查找能力相当强悍.此外,jQuery对象的方法肯定是有限的,有限的方法满足不了日益增长各有所需的要求,所以,必须提供jQuery对象方法的扩展能力.  
强悍的dom元素查找能力,以及随心所欲的方法扩展,这两点正是jQuery的核心所在!  
来一个简单的示例,来说明jQuery是如何工作的:

|  |
| --- |
| < !DOCTYPE html PUBLIC ”-//W3C//DTD XHTML 1.0 Transitional//EN” ”http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd”> |
| <html xmlns=”http://www.w3.org/1999/xhtml” > |
| <head> |
| <title>无标题页</title> |
| </head> |
| <body id=”bd”> |
| <a href=”http://www.baidu.com”>baidu</a> |
| </body> |
| </html> |
| <script type=”text/javascript” src=”../Script/jquery.js”></script> |
| <script type=”text/javascript” > |
| $(function(){ |
| $(“a“).click(function(e){//1)查找$(”a”);2)jQuery对象事件click;3)jQuery对象方法hide |
| $(this).hide(“slow“); |
| return false; |
| }); |
| }); |
| </script> |

jQuery中有一个\”配置”的思想,这一点使得对象的属性/事件等设置变得容易理解且十分简便,如下一个拖拽组件的初始化:

|  |
| --- |
| $(document).ready( |
| function() |
| { |
| $(’#drag1′).Draggable({ |
| handle: “ax“, //属性设置 |
| onStart: function(el,x,y){ //事件监听器设置 |
| } |
| }); |
| } |
| ); |

可以看到,$(”#drag1″)是查找并创建一个jquery对象,然后调用Draggable方法进行拖拽初始化,在此方法调用时,传递一个\”配置”对象,进行拖拽操作的初始化配置.  
这一”配置”的思想,极大简化了一些编码步骤,并相当直观和易懂.

以下我进行三个问答:  
1) 问:为什么$(selector)之后,返回的是jQuery对象?  
答:从jQuery的源代码中,我们可以知道:var $ = jQuery.因此当我们$(selector)操作时,其实就是jQuery(selector),创建的是一个jQuery对象.当然正确的写法应该是这样的:var jq = new $(selector);而jQuery使用了一个小技巧在外部避免了new,在jquery方法内部:if ( window == this ) return new jQuery(selector);  
2) 问:为什么创建一个jQuery对象之后,我们可以这样写$(selector).each(function(index){…});进行遍历操作呢?  
答:其实jQuery(selector)方法调用时,在jQuery(selector)方法内部,最后返回的是一个数组:return this.setArray(a);而each方法体内部是一个for循环,在循环体内是这样调用的:method.call(this[i],i).  
3) 问:为什么jQuery能做到jQuery对象属性/方法/事件的插件式扩展?  
答:如果您有一些javasciprt的面向对象方面的知识,就会知道,jQuery.prototype原型对象上的扩展属性/方法和事件,将会给jQuery的对象\”扩展”.基于这一点,jQuery是这样写的:jQuery.fn = jQuery.prototype.所以,当我们扩展一个插件功能时,如下:

|  |
| --- |
| jQuery.fn.check = function() { |
| return this.each(function() { |
| this.checked = true; |
| }); |
| }; |

其实就是:

|  |
| --- |
| jQuery.prototype.check = function() { |
| return this.each(function() { |
| this.checked = true; |
| }); |
| }; |

综上所述,jQuery给我们带来了一个简洁方便的编码模型(1>创建jQuery对象;2>直接使用jQuery对象的属性/方法/事件),一个强悍的dom元素查找器($),插件式编程接口(jQuery.fn),以及插件初始化的”配置”对象思想.