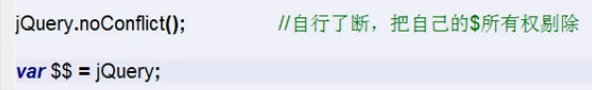
**在JQuery中，不管是页面元素选择、内置的功能函数，都是以“$”符来起始的。如下：**

1. **$(function(){ } ) 执行一个匿名函数**
2. **$(‘#box ’) 选择id=”box”的元素**
3. **$(‘#box’).css(‘color’:’red’) 执行功能函数**

**$()函数就是一个JQuery对象的制造工厂**

**$===jQuery**

**当其它库也使用$符而出现冲突时，可以var $$=jQuery; 使JQuery让出$符。**



**alert($) : 返回JQuery对象的内部**

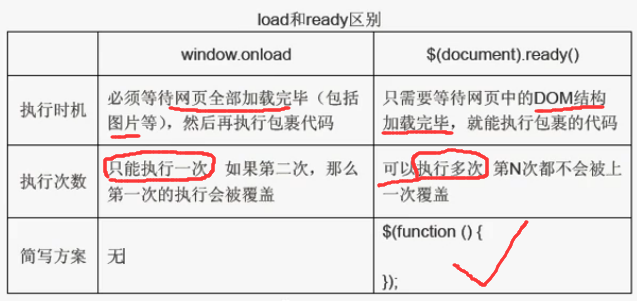
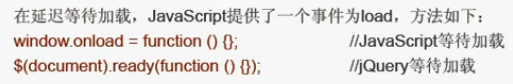
**alert($( ) ): 返回一个JQuery对象**

**alert($(‘ #box ’)): 返回一个JQuery对象**

**alert($(‘#box’).css(‘color’, ’red’)): 返回一个JQuery对象**

**加载模式：**

**JQuery库文件是在DOM元素之前被加载的，因此这时就无法对DOM内容进行操作。我们必须延迟加载库文件，使之在所有DOM元素加载后再加载JQuery。**



**Window.onload需要等待图片全部加载完毕才能执行！！！**

**通过元素获取到的JQuery对象是一个数组，从中取出一个即为DOM对象。**

**$(‘#box’).get(0) ： 使用JQuery返回第一个div对象**

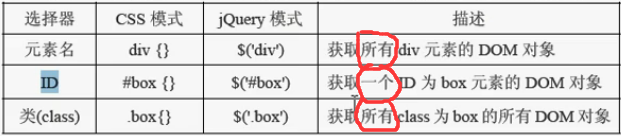
**等同于：$(‘#box’)[0];**

**常规选择器**

**JQuery最核心的组成部分就是“选择器引擎”，它继承并模拟了css的语法，可以对DOM元素的标签名、属性名、状态等进行快速的准确选择。**

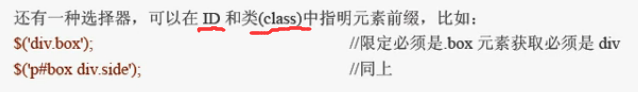
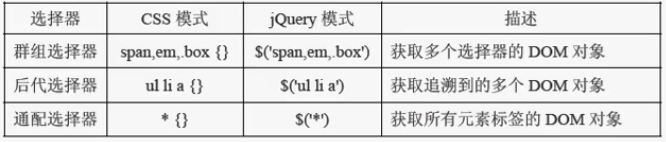
**JQuery选择器的语法与CSS选择器的语法十分相似，只是它们的功能不同。CSS找到元素后添加的是单一的样式，而JQuery则是添加的动作行为。**

**1.简单选择器**



**JQuery有很强大的容错功能**

**2．中级选择器：**



**群组选择器、后代选择器与3种基础选择器可以灵活的互相搭配，以实现极为强大的元素选择功能。**

**在全局范围内使用$(‘ ’)统通配选择器表示选择<html>及之下的所有标签，会极大地浪费资源。通配选择器一般与后代选择器搭配运用在局部的环境内。**

**3.高级选择器：**



//**后代选择器**

$('#box p').css('color','red'); 等同于： $('#box').**find('p').**css('color','blue');

//**子代选择器**

$('#box > p').css('color','red'); 等同于： $('#box').**children('p').**css('color','blue');

//**next选择器**，获取节点后一个同级DOM对象，如果不是所指定的没有效果

$('#box + p').css('color','red'); 等同于： $('#box').**next('p').**css('color','blue');

//**nextAll选择器**，获取节点后所有同级DOM对象

$('#box ~ p').css('color','red'); 等同于 $('#box').**nextAll('p').**css('color','blue');

//jQuery对高级选择器的补充，提供的几种方法

//同级上指定元素:一个或所有个

$('#box').prev('p').css('color','red');

$('#box').prevAll('p').css('color','blue');

//同级非指定元素:上或下

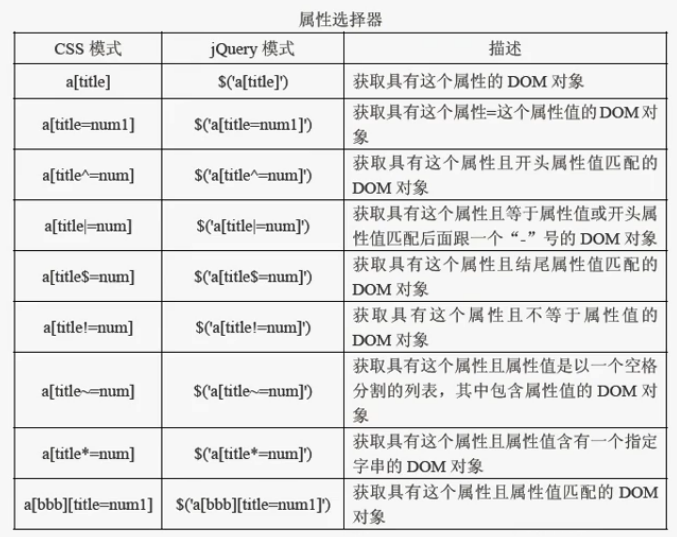
$('#box').prevUntil('p').css('color','red');

$('#box').nextUntil('p').css('color','blue');

//同级指定元素：上和下

$('#box').siblings('p').css('color','red');

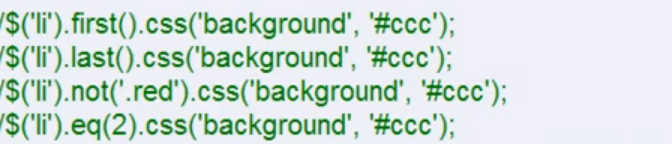
**高级选择器与JQuery的语法灵活搭配后可以有多种形式的选择方式，不同方式的选择器性能不尽相同，但上述调用方法的选择器方式是性能最高的，因此建议使用方法调用。同时此类的选择器方式具有较好的可读性。**



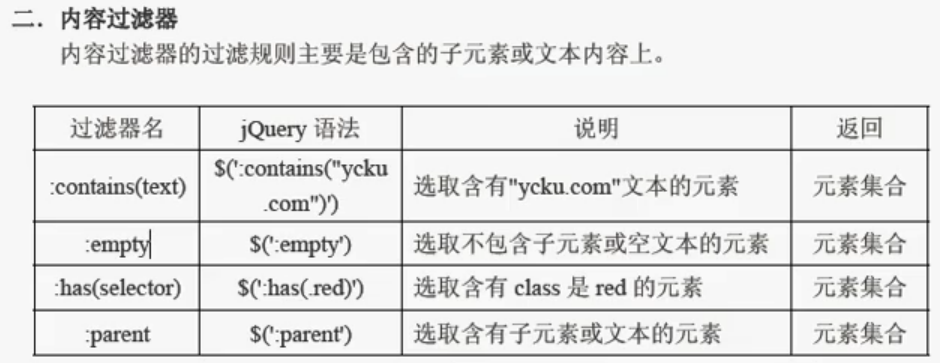
**属性选择器通常用在局部范围内，在全局范围中使用较少。**

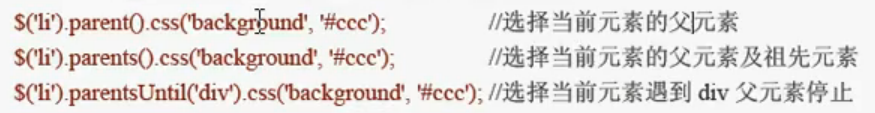
**4.过滤选择器**

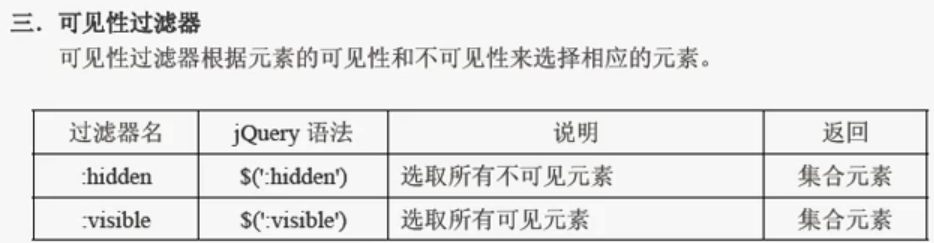
**过滤选择器简称：过滤器。它其实也是一种选择器，这种选择器类似于CSS3中的伪类。**

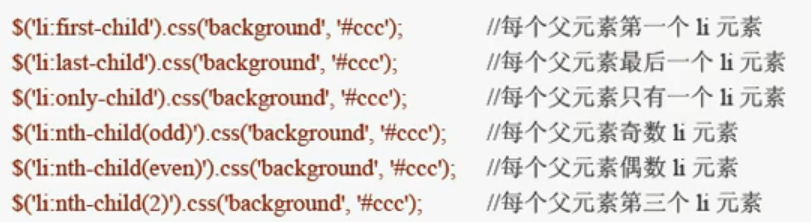
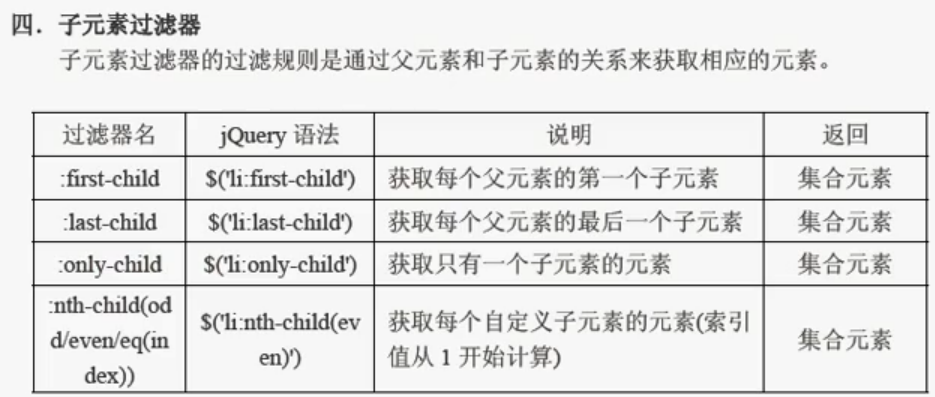


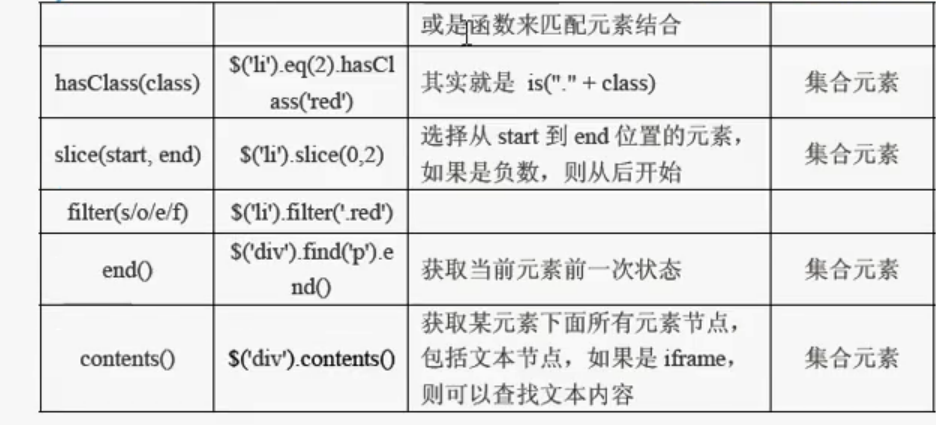
**：后面的内容是对前面的获取的元素对象的筛选。而并非是这些元素对象的子标签。：左右两边是平级的。**





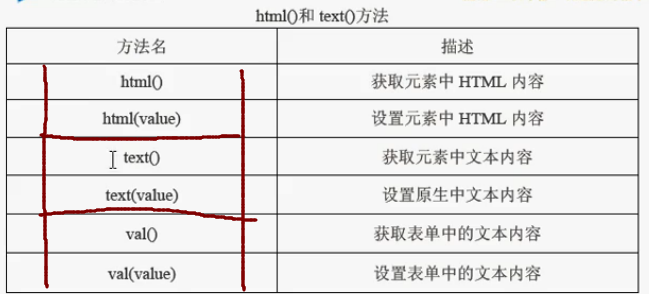




**DOM的基础操作：**

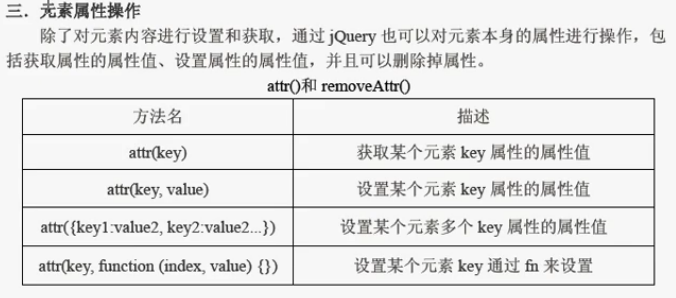
**我们通过前面学习的各种选择器、各种过滤器得到了我们想要操作的元素。这时，我们就可以对这些元素进行DOM操作。最常用的操作就是对元素内容进行获取和修改。**



**注意：始终都要带括号**

**html(value)方法能够解析value中的标签。但text(value)方法始终将value当做字符串处理**

**对于设置表单中文本内容的val(value)方法，如果想设置多个选项的选定状态，比如下拉列表、单选复选框等等，就可以通过数组来传递操作。**



**Key，value都需要带引号 ’ ’。**

**class属性不建议通过attr(key,value)来设置**

**ctrl+F5 ： 底层刷新**

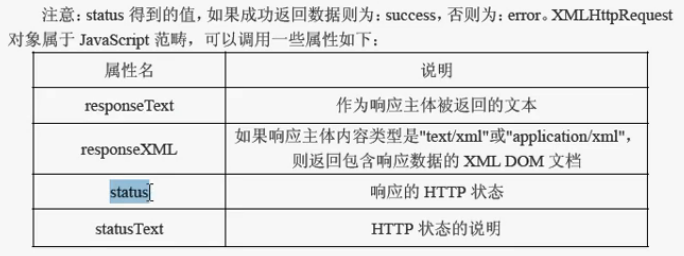
**JQuery-Ajax概述：**

**Jquery对ajax做了大量的封装，我们使用起来也较为方便，不需要去 浏览器兼容性。对于封装的方式，JQuery采用了三层级别的封装：**

1. **最底层的封装方法为：$.ajax( )**
2. **封装了$.ajax( )的第二层方法：.load( )、$.get( )、$.post( )**
3. **最高层是$.getScript( )和$.getJSON( )**

**除了.load( )方法为局部方法，其它均为全局方法。**

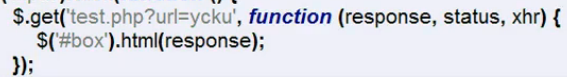
**.load( )局部方法需要一个JQuery对象作为前缀才能调用，而除此之外的全局方法不需要JQuery对象作为前缀就可以直接使用。对于需要动态传参到服务器页面的，$.get( )，$.post( )等更加合适**



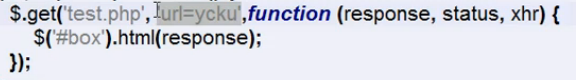
**<textarea>文本框中无法解析任何html标签**

**以get方式提交时，参数值有三种添加形式，但最终都将转换并附加到url的？后面。但是，post提交不能在url中以？来添加**

1. 直接在url后面？紧跟着传参。$.get( url , function() )



1. 将要传递的参数写成get( )的第二个参数data



1. 将第二个参数写成js对象的形式进行传参。

