jquery

```
jQuery(document).ready( function(){

    $("button").click( function(){  /*获取元素标签*/

       $("p").css("color","red").hide(5000);

       $("< h3>PHP 中文网< /h3>").appendaTo(document.body);/*appendTo 添加元素*/

    });

});

/*当页面元素加载完毕时开始执行*/
```

可以简化为：

```
$(function(){

    $("button").click( function(){  /*获取元素标签*/

       $("p").css("color","red").hide(5000);

       $("< h3>PHP 中文网< /h3>").appendaTo(document.body);/*appendTo 添加元素*/

    });

});
```

jQuery 对象与 DOM 对象相互转换

```
document.getElementTagName('li')[2].style.color="red";

/*将第三个 li 标签的字体设置为红色*/

可以变为：

$(document.getElementTagName('li')[2]).css("color","red");

$("li").get(2).style.color="red";

$("li")[2].style.color="red";
```

HTML 属性的查询与设置，移除

```
< img src="xxx.jpg" alt="photo" width="400">
```

```
$("img").attr("alt")  /*输出 img 标签下的 alt 属性*/

$("img").attr("src","xxa.jpg"); /*更改属性*/

$("img").removeAttr("属性名")

/*操作仅对当前有效，刷新页面后将恢复,原始的文档结构并未发生变化*/

$('img').css() /*可以修改,查询 css 的样式属性值*/
```

css 类设置

```
<script src="jquery.js"></script>
```

< img src="xxx.jpg" alt="hello w" width="240"> < style> .red{ border-radius:20px; box-shadow:10px 10px 5px #f00; } < /style> 设置： $("img").addClass("red"); 添加 $("img").removeClass("red"); 删除 $("img").toggleClass("red"); 添加/删除 $("img").isClass()/hasClass();判断是否存在该样式

HTML 内容查询与设置

```
< ul>

< li>01< /li>

< li>02< /li>

< li>03< /li>

< /ul>
```

查询：

```
$('li').eq(1).text();
```

修改：

```
$('li').eq(1).text("xx");
```

获取：

```
$('li').eq(2).html();
```

设置：

```
$('li').eq(2).html("xax");
```

jQuery 选择器

```
/*简单选择器*/

常规过滤器：

标签选择符 $("tag").css("bgcolor","red")

类选择符    $(".class").css("width","50")

ID 选择符    $("#id").css("bgcolor","green")



属性过滤器：

$("[style]").css();        /*属性为 style 的标签*/

$("[title='xxx']").css(); /*属性 title 值为 xxx 的标签*/

$("[title^='ap']").css(); /*属性值为 ap 开头的标签*/

$("[title!='abc']").css();/*属性值不为 abc 的标签*/

$("[title*='redis']").css();/*模糊查找，属性值中有 redis 的标签*/

$("[title~='PHP']").css();/*单词匹配，字符串前或后有空格分隔的认为是单词*/

$("[title][class]").css();/*同时具有 title 和 class 属性的标签*/



表单过滤器：

$("input:text").css();

$("input:password").attr("placeholder","至少 8 到 16 位");

$("input:checkbox").css("box-shadow","2px 2px 2px green");/*将复选框设置绿色阴影*/

$("input:checked").css("box-shadow","2px 2px 2px red");/*将选中的复选框阴影设置为红色*/

$(":button").css();

$(":disabled").css();



位置过滤器：
```

```
$("li:eq(索引)").css(); /*eq(0)含义为索引等于0*/

$("li:qt(索引)").css();/*qt(7)表示为所以大于7*/

$("li:lt(索引)").css();/*qt(2)表示为所以小于2*/

$("li:first").css();/*第一个li元素*/

$("li:last").css(); /*最后一个li元素*/

$("li:even").css(); /*选择所有偶数li元素*/

$("li:odd").css();  /*选择所有奇数li元素*/

$("li:nth-child(n)").css();/*全选：参数:n，n的取值(1~max)，n从1开始*/

/*2n(even)(因为n从1开始，所以实际选中的为奇数)*/

/*2n+1(odd)为选中奇数，实际显示为偶数*/
```

/*组合选择器*/

后代组合符："空格"和">"

A B: 在A元素所有子孙元素中查询B元素

A>B: 仅在A元素中的子元素中查询B，不含孙元素

兄弟组合符：

A~B:查找A元素后面所有与A有着共同父级的兄弟元素

A+B:查询A元素后面的第一个兄弟元素

/*选择器组*/

用","分隔的选择器列表

```
$("h3,h4,h5").css();/*直接使用标签*/

$("p.green,div.red").css();/*带过滤的简单选择器 选择 p 标签 class 属性为 green 和 div 标签 class 属性为 red 的*/
```

组合选择器：

```
$(p>span,div strong).css();

/*p 下的直接子元素 span 和 div 的后代元素 strong*/
```

选择方法：

```
1.根据结果集中元素位置选择：

$("li").first().css();

$("li").last().css();

$("li").eq().css();/*从零开始*/

$("li").slice(a,b).css();

/*slice(起始索引，结束索引)，索引从零开始，结果不包括结束索引数据*/

/*支持负数，从倒数开始选择元素，例：slice(-3),选择最后三个*/



2.根据结果集中元素的自身特征来选择：filter(),not().

选择元素与自身属性相关可以用 filter().

$("li").filter(".red").css();/*选择 li 标签下 class 属性为 red 的元素*/  等价于

$("li.red").css();

filter 方法可以传入一个回调判断函数，参数就是索引，

例：获取偶数行：

$("li").filter(function(index){

    return index%2==0;

}).css();   等价于：

$("li:even").css();$("li:nth-child(2n+1)").css();

用 not 来确定奇数行：
```

```
$("li").not(function(index){
    return index%2==0;
}).css();   等价于：
$("li:add").css();$("li:nth-child(2n)").css();
```

3.将选中元素集合当做上下文.(其实就是一个定位标志)

方法：

```
1.find 2.children 3.contents 4.next 5.prev
6.nextAll 7.prevAll 8.parent 9.parents
10.closest 11.parentsUntil
/*查询所有 li 元素下面的 strong 元素,可以深入多级*/
$("li").find("strong").css();
/*获取当前元素的下一个元素*/
$("#info").next().css();
/*获取从当前元素开始到结束的全部元素，它会将子孙元素全部选中,当前元素为 id="info"*/
$("#info").nextAll().css();
/*获取当前元素的前一个元素*/
$("#info").prev().css();
/*获取当前元素前的全部元素*/
$("#info").prevAll().css();
/*获取当前元素的父级元素*/
$("#info").parent().css();
<script src="...">  </script>
<script>
//$(this):返回当前 jQuery 对象，与上下文相关
```

```
$(document).ready(function(){

    $("img").click(function(event){

      alter($(this).attr("alt"));

    });

});

</script>
```

```
$(document).ready(function(){

    $("#sitename").bind("click",function(){

        //append():将新元素添加到当前的 jQuery 对象中

        //bind 绑定事件

        $(this).addend("(xxxx)");

        //after():将新元素添加到当前对象的后面

        $(this).after("< p>  PHP< /p>");

        prepend():将新元素添加到当前元素的前面，它还在当前元素中

        $(this).prepend("< a href='http://.....'>点击访问< /a>");

        //before():将新元素添加到当前元素的前面

        $(this).before("< h4>PHP is the best< /h4>");

        //replaceWith():替换当前元素

        $(this).replaceWith("< h1>"+$(this).text()+"< /h1>");

    });

});

< h3 id="sitename">PHP< /h3>
```

- 克隆元素，仅适用于页面上存在的元素

```
$(document).ready(function(){
```

```
    //向 body 中添加一个 div 元素
  $(document.body).append("< div id='xx'>links< /div>");
    //将当前页面的所有元素，全部克隆到新创建的< div>中
  $("a").clone.appendTo("#xx");
    //给< div>中的所有元素后添加一个< br>
  $("#xx > a").after("< br>");
});
```