**[jquery几种事件绑定方式的比较](https://www.cnblogs.com/DemoJin/p/4794372.html)**

比较和联系:  
1.bind()函数只能针对已经存在的元素进行事件的设置；但是live(),on(),delegate()均支持未来新添加元素的事件设置；  
  
2.bind()函数在jquery1.7版本以前比较受推崇，1.7版本出来之后，官方已经不推荐用bind()，替代函数为on(),这也是1.7版本新添加的函数，同样，可以  
  
　用来代替live()函数，live()函数在1.9版本已经删除；  
  
3.live()函数和delegate()函数两者类似，但是live()函数在执行速度，灵活性和CSS选择器支持方面较delegate()差些，想了解具体情况，请戳这：  
  
　http://kb.cnblogs.com/page/94469/  
  
4.bind()支持Jquery所有版本；live()支持jquery1.9-；delegate()支持jquery1.4.2+；on()支持jquery1.7+；

**推荐使用.on()方法绑定，原因有两点**：

**1.on()方法可以绑定动态添加到页面元素的事件**

比如动态添加到页面的DOM元素，用.on()方法绑定的事件不需要关心注册该事件的元素何时被添加进来，也不需要重复绑定。有的同学可能习惯于用.bind()、.live()或.delegate()，查看源码就会发现，它们实际上调用的都是.on()方法，并且.live()方法在jQuery1.9版本已经被移除。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | bind: function( types, data, fn ) {      return this.on( types, null, data, fn );  },    live: function( types, data, fn ) {      jQuery( this.context ).on( types, this.selector, data, fn );      return this;  },    delegate: function( selector, types, data, fn ) {      return this.on( types, selector, data, fn );  } |

移除.on()绑定的事件用.off()方法。

**2.on()方法绑定事件可以提升效率**

很多文章都提到了利用事件冒泡和代理来提升事件绑定的效率，大多都没列出具体的差别，所以为了求证，我做一个小测试。

假设页面添加了5000个li，用chrome开发者工具Profiles测试页面载入时间。

(事件冒泡参考 http://www.cnblogs.com/webflash/archive/2009/08/23/1552462.html)

**普通**

|  |  |
| --- | --- |
| 1  2  3 | $('li').click(function(){      console.log(this)  }); |

绑定过程的执行时间

[2013-08-13_190358](https://images0.cnblogs.com/blog/551140/201308/14154950-3bdab7181c0849fc97cf64f647ee2811.png)

普通绑定相当于在5000li上面分别注册click事件，内存占用约4.2M，绑定时间约为72ms。

**.on()绑定**

|  |  |
| --- | --- |
| 1  2  3 | $(document).on('click', 'li', function(){      console.log(this)  }) |

绑定过程的执行时间

[2013-08-13_191010](https://images0.cnblogs.com/blog/551140/201308/14154950-8cdeab526b6d4e688ef1fbd2a76d6573.png)

.on()绑定利用事件代理，只在document上注册了一个click事件，内存占用约2.2M，绑定时间约为1ms。

**多个事件绑定同一个函数**

$(document).ready(function(){

  $("p").on("mouseover mouseout",function(){

    $("p").toggleClass("intro");

  });

});

**多个事件绑定不同函数**

$(document).ready(function(){

  $("p").on({

    mouseover:function(){$("body").css("background-color","lightgray");},

    mouseout:function(){$("body").css("background-color","lightblue");},

    click:function(){$("body").css("background-color","yellow");}

  });

});

用on()方法绑定多个选择器，多个事件则可以这样写：  
  
$(document).on({  
    mouseenter: function() {  
        // Handle mouseenter...  
    },  
    mouseleave: function() {  
        // Handle mouseleave...  
    },  
    click: function() {  
        // Handle click...  
    }  
}, '#header .fixed-feedback-bn, #sb-sec .feedback-bn');

绑定自定义事件

$(document).ready(function(){

  $("p").on("myOwnEvent", function(event, showName){

    $(this).text(showName + "! What a beautiful name!").show();

  });

  $("button").click(function(){

    $("p").trigger("myOwnEvent",["Anja"]);

  });

});

传递数据到函数

function handlerName(event)

{

  alert(event.data.msg);

}

$(document).ready(function(){

  $("p").on("click", {msg: "You just clicked me!"}, handlerName)

});

适用于未创建的元素

$(document).ready(function(){

  $("div").on("click","p",function(){

    $(this).slideToggle();

  });

  $("button").click(function(){

    $("<p>This is a new paragraph.</p>").insertAfter("button");

  });

});

tip:如果你需要移除on()所绑定的方法，可以使用off()方法处理。

$(document).ready(function(){

　　$("p").on("click",function(){

　　　　$(this).css("background-color","pink");

　　});

　　$("button").click(function(){

　　　　$("p").off("click");

　　});

});

在需要为较多的元素绑定事件的时候，优先考虑事件委托，可以带来性能上的好处。将click事件绑定在document对象上，页面上任何元素发生的click事件都冒泡到document对象上得到处理。