

JavaScript事件委托的技术原理

分享

分享

分享

分享

分享

如今的JavaScript技术界里最火热的一项技术应该是‘事件委托(event delegation)’了。使用事件委托技术能让你避免对特定的每个节点添加事件监听器；相反，事件监听器是被添加到它们的父元素上。事件监听器会分析从子元素冒泡上来的事件，找到是哪个子元素的事件。基本概念非常简单，但仍有很多人理解事件委托的工作原理。这里我将要解释事件委托是如何工作的，并提供几个纯JavaScript的基本事件委托的例子。

假定我们有一个 `UL` 元素，它有几个子元素：

```
<ul id="parent-list">
  <li id="post-1">Item 1</li>
  <li id="post-2">Item 2</li>
  <li id="post-3">Item 3</li>
  <li id="post-4">Item 4</li>
  <li id="post-5">Item 5</li>
  <li id="post-6">Item 6</li>
</ul>
```

我们还假设，当每个子元素被点击时，将会有各自不同的事件发生。你可以给每个独立的 `li` 元素添加事件监听器，但有时这些 `li` 元素可能会被删除，可能会有新增，监听它们的新增或删除事件将会是一场噩梦，尤其是当你的监听事件的代码放在应用的另一个地方时。但是，如果你将监听器安放到它们的父元素上呢？你如何能知道是那个子元素被点击了？

简单：当子元素的事件冒泡到父 `ul` 元素时，你可以检查事件对象的 `target` 属性，捕获真正被点击的节点元素的引用。下面是一段很简单的JavaScript代码，演示了事件委托的过程：

```
// 找到父元素，添加监听器...
document.getElementById("parent-list").addEventListener("click",function(e) {
  // e.target是被点击的元素!
  // 如果被点击的是li元素
  if(e.target && e.target.nodeName == "LI") {
    // 找到目标，输出ID!
    console.log("List item "+e.target.id.replace("post-","")+" was clicked!");
  }
});
```

第一步是给父元素添加事件监听器。当有事件触发监听器时，检查事件的来

源，排除非 `li` 子元素事件。如果是一个 `li` 元素，我们就找到了目标！如果不是一个 `li` 元素，事件将被忽略。这个例子非常简单，`ul` 和 `li` 是标准的父子搭配。让我们试验一些差异比较大的元素搭配。假设我们有一个父元素 `div`，里面有很多子元素，但我们关心的是里面的一个带有“`classA`” CSS类的A标记：

```
// 获得父元素DIV, 添加监听器...
document.getElementById("myDiv").addEventListener("click",function(e) {
    // e.target是被点击的元素
    if(e.target && e.target.nodeName == "A") {
        // 获得CSS类名
        var classes = e.target.className.split(" ");
        // 搜索匹配!
        if(classes) {
            // For every CSS class the element has...
            for(var x = 0; x < classes.length; x++) {
                // If it has the CSS class we want...
                if(classes[x] == "classA") {
                    // Bingo!
                    console.log("Anchor element clicked!");

                    // Now do something here....

                }
            }
        }
    }
});
```

上面这个例子中不仅比较了标签名，而且比较了CSS类名。虽然稍微复杂了一点，但还是很具代表性的。比如，如果某个A标记里有一个 `span` 标记，则这个 `span` 将会成为target元素。这个时候，我们需要上溯DOM树结构，找到里面是否有一个 `A.classA` 的元素。

因为大部分程序员都会使用jQuery等工具库来处理DOM元素和事件，我建议大家都使用里面的事件委托方法，因为这里工具库里都提供了高级的委托方法和元素甄别方法。

希望这篇文章能帮助你理解JavaScript事件委托的幕后原理，希望你也感受到了事件委托的强大用处！

(英文：davidwalsh.)

◀ JAVASCRIPT ▶ 事件委托

分享这篇文章：

分享

分享

分享

分享

分享

扫一扫二维码分享：



云服务器

弹性扩展，安全稳定，简单易用

40.8元起

详情

域名

超过1000万域名在阿里云注册


4元起


详情


相关文章：


- 1. 用JavaScript检测离线/在线状态
- 2. 5种JavaScript和CSS交互的方法
- 3. 使用jQuery加载js脚本
- 4. 使用 jQuery 复制文字到剪贴板的技巧
- 5. JavaScript DocumentFragment：更快捷的操作DOM的途径
- 6. 使用CSS标注外部链接的巧妙方法
- 7. 用CSS制造出光泽一闪而过的图片效果
- 8. 使用HTML5里页面可见性接口判断用户是否正在观看你的页面
- 9. HTML拼音/音标注释标记ruby和它的子元素rt/rp
- 10. JavaScript手机振动API

4 参与讨论

 **wannianchuan**
2014年4月13日 下午12:11
理解了

 **wikies.wan**
2014年4月29日 下午4:43
谢谢，博主继续

 **张龙冬**
2014年11月14日 上午11:30
点赞，理解了什么叫做事件委托

 **上单**
2016年5月15日 下午3:07
与.net 的委托比起来 就是渣渣