# 15天学会jQuery (6-10)

## 15 Days of jQuery(Day 6) --- 更安全的Contact Forms，不带CAPTCHA

这次的教程内容贴近我擅长的技术方向：安全的contact forms。

就像我在前一篇教程中提到的那样，一个最普通的contact forms可以帮助访客同你进行通信来往而不需要暴露你的电子邮件地址给那些可恶的垃圾邮件制造者们。

但如果spammer们已经盯上你，没有什么比一个不安全的contact foms更糟糕的了。想象一下你的网络空间提供商发给你一封措辞强烈的电子邮件，通知说：他们发现你的网站发送了大批量的性药广告以及其他垃圾邮件，另外，直到你停止这种行为之前，你的网站都将处于离线状态－谢谢！

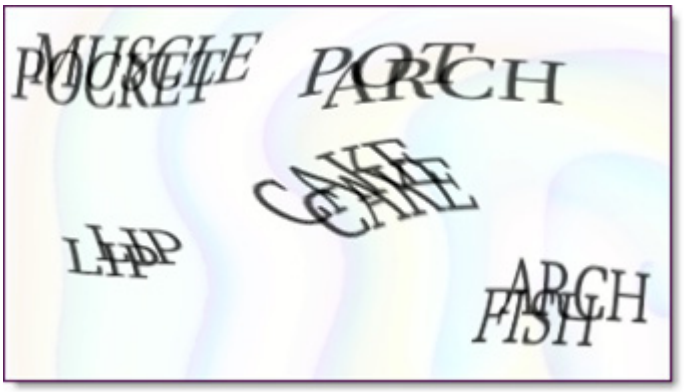
那么，今天我要在这篇教程里告诉大家的是一种在任何contact forms上添加一个额外安全层的简单方法－即使你没有使用我提供的超级安全、超级灵活的Ultimate Form Mail。

***当前状况***

你意识到spammer们已经通过远程探测技术发现了你的contact forms的弱点，而你希望他们走开。

***难点***

你不想使用CAPTCHA（Completely Automated Public Turing Test to Tell Computers and Humans Apart），因为你明白，让你的访客先去阅读那些歪七扭八的字母数字才能发送消息只能抑制他们想要互动的欲望，而不是促进它。(数字验证的缺陷)



关键点：你希望那些坏家伙们堵车到天黑，同时希望那些好孩子们一条大道通罗马。

***解决方案***

你将学会在页面加载的时候使用jQuery来给你的contact forms添加一些隐藏的标签信息。当窗体信息被提交到服务器端的时候，你可以用一些简单的php代码实现如下的步骤：

隐藏的标签被识别出来 隐藏标签的信息与你的网站访客下载到浏览器上的cookie里的某项标志相一致 隐藏标签的有效时间还未过期 换句话说，你的访客们只有在一段有限的时间内才可以填写窗体并进行发送。如果一个spammer尝试通过远程调用来提交窗体信息到你的服务器，他们将会发现自己踢到了一块又厚又硬的铁板，不付出点代价休想通过。

我将要告诉你的这种方法是从一位非常聪明的同事Chris Shiflett提供的蓝本基础上修改而成的。他是位专业的安全专家，对php程序员经常遇到的安全问题了如指掌（我怎么感觉他又要忍不住提到他的Ultimate Form Mail 了~~汗）。

*教程*

基于上次那篇《斑马线表格轻松制作》的反响良好，我决定再次制作一次类似的“手把手图文教程”。虽然要花费些时间，但很值得这么做。

手把手教程

DEMO

源代码

*银弹？* 1)

> 银弹是软件领域的说法,意为解决一切问题的方法。这个来源于欧洲的传说,说是只有银弹可以消灭狼人。

“那么，现在我的窗体就是100%安全的，可以假设任何免费的cntact forms程序，然后高枕无忧了？”

呃。。。非也。

这种安全模式基于一个关键的假定：**Spammer们总是会拿软柿子捏，浪费时间去解决一个狡猾的对手对他们来说就是浪费金钱。**

现在， 好好听着，我的朋友们：

这个技术，尽管相当健壮，但仍然不是解决目前脆弱的窗体处理程序问题的灵丹妙药。

我的这些安全建议的目的是为了让spammer们知难而退。小偷们入室盗窃之前总会进行仔细踩点，他们只对那些可以用最小代价获取最大利益的房间感兴趣。

换句话说，如果在他们动手之前有99%的机会挡住他们的试探，而且实现起来相当容易，为什么不试一试呢？这才是此项技术要实现的目标。

但这还是治标不治本，不能解决所有问题。

---

## 15 Days of jQuery(Day 7) --- 样式表切换

我第一次看到样式表切换器是在A List Apart或者Simple Bits，那是两个设计师最应该去的网站。

从那以后，我找到了很多可以让访客通过鼠标点击某个地方切换样式表的方法。但最近我要写一篇如何 使用jQuery编写简单代码实现它的教程。

我将向你们逐步解说整个的过程，不仅仅因为要展示jQuery代码的简介，同时也要揭示jQuery库中的若干高级特性。

*首先，代码*

```
$(document).ready(function()
{
$('.styleswitch').click(function()
{
switchStylestyle(this.getAttribute("rel"));
return false;
});
var c = readCookie('style');
if (c) switchStylestyle(c);
});
```

```
function switchStylestyle(styleName)
{
$('link[@rel*=style]').each(function(i)
{
this.disabled = true;
if (this.getAttribute('title') == styleName) this.disabled = false;
});
createCookie('style', styleName, 365);
}
```

其他这里没有提到的部分是你将在后面看到的创建和读取cookie的函数。

***熟悉的开篇***

```
$(document).ready(function()
{
$('.styleswitch').click(function()
```

告诉jQuery“以最快的速度查找所有包含对象名‘styleswitch’的元素，并在他们被鼠标点击时执行一个函数”。

看起来不错。当鼠标点击预先指定的元素时，switchStylestyle函数将被调用。从现在开始是重点。

这句话什么意思？

第一次看到这句代码的时候我的脑子有些卡壳：

$('link[@rel*=style]').each(function(i) {

在互联网上搜索了一下后我空手而归。最后不得不找到了jQuery的作者John Resig，向他咨询。

他直接给了我一个jQuery网站的页面地址，里面讲解了若干jQuery提供的高级特性(xpath)，可以用来查找并操作页面中的若干元素。

如果你看过这些东西你就能明白上面那句神秘的代码的含义是告诉jQuery“查找所有带rel属性并且属性值字符串中包含‘style’的link链接元素”。

嗯？

让我们看看如何编写包含一个主样式表，两个备用样式表的页面：

```
<link rel="stylesheet" type="text/css" href="styles1.css" title="styles1" media="screen" />
<link rel="alternate stylesheet" type="text/css" href="styles2.css" title="styles2" media="screen" />
<link rel="alternate stylesheet" type="text/css" href="styles3.css" title="styles3" media="screen" />
```

我们可以看到所有样式表都含有一个包含‘style’字串的rel属性。

所以结果一目了然，jQuery轻松定位了页面中的样式表链接。

***下一步？***

each()函数将遍历所有这些样式表链接，并执行下一行中的代码：

```
this.disabled = true;
if (this.getAttribute('title') == styleName) this.disabled = false;
```

“首先禁用所有的样式表链接，然后开启任何title属性值与switchStylestyle函数传递过来的字串相同的样式表”

一把抓啊，不过很有效。

现在我们需要保证的是那些样式表存在并且有效。

***完整代码和演示***

既然 Kelvin Luck已经编写了这些代码，我就不在这里重复了。

DEMO

我相信Kelvin的灵感是从 这个网站那里得到的，我们正好可以看看使用其他工具实现这个功能是否要比jQuery更加复杂冗长。

---

## 15 Days of jQuery(Day 8) --- 使用Javascript（jQuery）实现圆角边框

当我看到这些实现圆角边框的HTML源代码的时候，我发现这很适合用来写一篇jQuery教程 - 使用wrap()、prepend()、append() 函数。

这里是原先的HTML代码，我们将从这里开始：

```
<div class="dialog">
  <div class="hd">
    <div class="c"></div>
  </div>
  <div class="bd">
    <div class="c">
      <div class="s">
        <main
          content goes here >
      </div>
    </div>
  </div>
  <div class="ft">
    <div class="c"></div>
  </div>
</div>
```

现在我们怎么使用jQuery来精简这段代码呢？

首先，我们需要一个“钩子”，一个特殊的HTML元素，或者一个id，或者一个对象名 - 来告诉jQuery处理的目标。

现在我们改成了这个样子：

 <main content goes here >  下一步，我们使用jQuery来将剩下的代码添加进去：

```
$(document).ready(function(){ $("div.roundbox") .wrap('<div
class="dialog">'+
'<div class="bd">'+
'<div class="c">'+
'<div class="s">'+
'</div>'+
'</div>'+
'</div>'+
'</div>');
});
```

***其他Div标记去哪里了？***

仔细观察代码，你就会发现它们都跑到了js代码里面，在wrap函数执行时它们将嵌套在“钩子Div”的内部。

细心的观众会发现我漏掉了部分代码。这是因为jQuery中的wrap()函数要求div标签必须严格对称嵌套才能工作。

具体的，我去掉了下面两个部分：

```
<div class="hd"><div class="c"></div></div>
<div class="ft"><div class="c"></div></div>
```

*添加和预置一体化*

下一步我们将会通过prepend和append函数将这两段代码添加进带有dialog对象名的div标记内部，并且使用“连锁”方法。

```
$('div.dialog').prepend('<div class="hd">'+
'<div class="c"></div>'+
'</div>')
.append('<div class="ft">'+
'<div class="c"></div>'+
'</div>');
```

*示例及代码*

我已经在网上放置了一个演示页面供大家查看。建议你看一下页面的源代码，体会jQuery给页面代码带来的清爽和便捷。

这些代码来自 Schillmania的一篇文章，个人推荐大家下载包含点缀图片的zip打包，非常精美。

*不使用图片的圆角边框*

有很多方法可以实现圆角边框 - 有些方法甚至不需要图片。

在jQuery的网站上有一个用来制作无图圆角边框的插件。虽然不是适合所有人（或者说所有程序），但也值得学习。

看看它的漂亮代码吧（使用时）：

```
$(document).bind("load", function(){
$("#box1").corner()
});
```

---

## 15 Days of jQuery(Day 9) --- 快速和略显粗劣的AJAX视频教程

今天我的想法有点改变。近段时间以来我一直考虑注册一个YouTube帐号来上传一些教程录像，现在我终于做出了决定并上传了一个。在这里我将手把手的向大家演示为你的网站添加一些AJAX基本应用的方法。

录像很短，因为YouTube对上传影片的长度有限制（10分钟以内）。当然由于制作仓促，错误在所难免。比如在某个地方我称CGI为“服务器端脚本”，而更准确的说法应该是“服务器端语言”。

这是AJAX，还是AHAH，抑或AXAH？

你将看到的东西其实更接近AHAH而不是纯粹的AJAX。

有什么区别么？AJAX中的“X”代表着XML。但更多时候人们喜欢使用简单的文本或者javascript代码或者单独文件而不是那种复杂冗长的XML。对此有篇文章有详细论述：AJAX vs. AHAH。

至于AXAH。。。 Cody Lindley的文章可以解释一切。对AJAX的一些工作理念有兴趣的读者可以看一下。

***教程录像***

这个页面上有我提供的演示。

---

## 15 Days of jQuery(Day 10) --- 使用jQuery Javascript 库实现“即点即改”的AJAX化

以前我在Quirksmode网站见过这种代码，后来又在24 Ways网站看到了一个更具Web 2.0风格的方案。这次我将为大家展示两种使用jQuery实现相同功能（甚至更好）的方法。

***目标***

一个用AJAX（或AHAH）技术设计的页面，访问者无需离开就可以在看到的(x)HTML 页面上编辑内容。

***方案***

点击需要编辑的文本，变幻出一个带有保存和取消按钮的textarea。修改的部分将通过AHAH传送至服务器端的一个PHP脚本文件，用来更新数据库（MySQL或普通文件）。

***演示***

AJAX式即点即改演示一

在这第一个演示中，我使用了一个id为“editinplace”的div元素。当鼠标划过这里时，背景颜色将变成浅黄色。点击文本将启动一些DOM操作，div元素被一个textarea元素取代 - 内中包含原先的文本。

点击保存按钮将向服务器端的PHP脚本文件发送新的HTML内容，并重新输出收到的新文本内容（通过$_POST）。

在真实应用环境下，你还应当添加一个安全性检测，然后才能更新数据库并返回更新后的页面内容，同事告知jQuery执行成功的信息。

但在这个例子中，所有的修改都是成功的，发送给PHP脚本的信息将原封不动的返回到jQuery代码，显示到一个普通的警告窗口里。

***解释***

开头部分说了很多次了，如果你不想使用jQuery提供的document.ready函数，尽可以选择你自己中意的init()函数。

```
$(document).ready(function(){
setClickable();
});
```

页面上第一个被执行的就是这个setClickable()函数。它的任务就是做以下内容：

查找包含id为“editinplace”的div元素，然后告诉jQuery在这些div被点击时执行某些操作。

```
function setClickable() {
$('#editInPlace').click(function() {
```

读取div内部的HTML代码的任务将交给jQuery的html()函数来完成。这些HTML将会额外添加若干代码以组成textarea里的保存和取消按钮。

```
var textarea = '<div><textarea rows="10" cols="60">'+$(this).html()+'</textarea>';
var button = '<div><input type="button" value="SAVE"
class="saveButton" /> OR <input type="button" value="CANCEL"
class="cancelButton" /></div></div>';
var revert = $(this).html();
```

同样还是这些在div内部找到的HTML代码将会赋值给一个叫做“revert”的变量。这个变量将用来在取消按钮被按下的事件中输出原始文本。

```
var revert = $(this).html();
```

jQuery的DOM函数“after”用来将新生的textarea　HTML代码放置在我们指定的div元素后。我在后面紧跟着连锁上了一个remove()方法 来移除div元素以节省代码。

```
$(this).after(textarea+button).remove();
```

在使用jQuery的时候，我通过对象名来定位保存和取消按钮对象。我指示jQuery在任一按钮按下时触发最后一个函数“saveChanges”。我告诉了jQuery在div元素被点击时做什么事情，但我没有在最后加上分号因为我希望在这个div操作语句后面连锁其他方法。

```
$('.saveButton').click(function(){saveChanges(this, false);});
$('.cancelButton').click(function(){saveChanges(this, revert);});
})
```

我再连锁了一个简单的mouseover和mouseout事件，告诉jQuery在鼠标指针掠过我们指定的div元素（id=editInPlace）的时候添加和移除一个对象。

```
.mouseover(function() {
$(this).addClass("editable");
})
.mouseout(function() {
$(this).removeClass("editable");
});
};//end of function setClickable
```

函数“saveChanges”将以按钮对象做为第一个参数，而cancel参数则取两种值，false或者保存在revert变量中的html代码内容。

```
function saveChanges(obj, cancel) {
```

如果cancel为假，则函数将保存更改并使用html格式发送给服务器端的php脚本。我在这里使用了jQuery内置的一个DOM函数实现对textarea内容的提取操作：parent()和siblings()。

```
if(!cancel) {
var t = $(obj).parent().siblings(0).val();
```

DOM基础超出了本系列教程的范围，但在这个应用中我只是告诉了jQuery“对象（保存按钮）有一个父元素（div）。。。去找到它。那个元素拥有一个或多个DOM树同级对象。。。我只想找到其中的第一个。然后提取那个对象的所有内容。”

（稍等。。。如果你对DOM风格的代码不是很熟悉的话，前面我的注释可能并不好理解。我还是建议你之前google一下“DOM javascript”或者其他相关的信息。）

这些html赋值给了t变量，现在要通过POST方法把它发送给test2.php。

```
$.post("test2.php",{
content: t
},function(txt){
alert( txt);
});
}
```

如果cancel有一个值，那么必然是保存在revert变量中的原始html内容。所以，在这个时候我希望变量t变为原始html内容。

```
else {
var t = cancel;
}
```

下一步是通过jQuery提供的DOM函数放置一个新的div元素，id为“editInPlace”，在这之后包含了textarea元素。。。然后删除掉这个div元素。

```
$(obj).parent().parent().after('<div id="editInPlace">'+t+'</div>').remove()
```

在果壳中，这将告诉jQuery“在DOM树中上跃两次。将HTML代码附在到达位置的对象之后，然后移除那个对象。”

最后，我们再次调用setClickable函数并关闭saveChange()函数。重调setClickable()函数的含义是重新设置onMouseover,onMouseout,和onClick事件到初始状态。

```
setClickable();
}
```

### *第二个示例*

第二个方法非常类似但也有点复杂。

示例二

没有用到庞大的单独div元素，这个示例将每个段落p标签变换成单独的可编辑区域。

这里的难度在于你如何在向服务器端脚本发送数据时指定正确的段落p标签。

在这里我通过为每个p标签编号并将这个编号一同发送给服务器端的php脚本的方式解决了问题。你会在alert窗口中看到php脚本是如何“知道”哪个p标签里的内容被修改的。

### *已知的问题*

现实的应用中，你在使用类似的功能时首先需要验证更改的内容的合法性，然后才能将数据发送到服务器端。显然在这里我们刻意把这些内容忽略掉了。

---

1) 解决一切问题的方法