Jquery 15 天教程

第一章 15 Days of jQuery What, Why, When, Where, Who

what

jQuery 是一个了不起的 javascript 库，它可以是我们用很少的几句代码就可以创建出漂亮的页面效果。从网站的方面说，这使得 javascript 更加有趣。

如果你这样想：“孩子,我需要另外一个 javascript 库,就好比我 I need another hole in my head”那么加入这个俱乐部吧。这正是我第一次遇到的时候所想的。

我已经用过了 Moo.fx, Scriptaculous, TW-SACK, 和 Prototype. 我曾参与了 RICO, Yahoo YUI 和其他一些库的开发。

没有了 PHPjavascript 和我一点也不亲近了。但是我还是尽全力保持头脑清醒，并尽量保持用 AJAX 去思考。

所以当我遇到 jQuery 的时候我想:“还需要另外一个 javascript 库吗?不了,谢谢…”

why

为什么我改变我我对 jQuery 的看法，以及为什么你要考虑去使用它? 很简单,只要你看一眼过使用 jQuery 的页面你就会发现它是如此的简单易用.只用很少的几行,就能表现出很优雅的效果. 有一天当我突然看到一些用 jQuery 写的代码时我一下子豁然开朗了. 早茶的过程中,我例行公务的去翻阅我的订阅,去看每日必看的设计博客的时候我看到了一个用 jQuery 写的 javascript 的例子.事实证明,这些代 码还是有些和浏览器关联的 bug,不过这些概念还是我以前从来没有见过的.

还有那些代码…

代码看起来很简单看起来不像我以前见过的.但也不无道理.

我开始通读文档,并且惊奇的发现用一点点代码竟然能做这么多事情.

when

你应当在你需要的时候使用 jQuery.

给你一个小型的库文件 DOM 强大的控制能力 不费吹灰之力的工作,和少许的努力.

或者

快速的通过 AJAX 没有大量无用的代码 和一些基本的动画效果

但是

如果你需要超级花式效果,动画,拖放,和超级平稳动画,那么你可能想使用 Prototype.他是一个有大量动画效果的类库.

where

你可以 jQuery 的官方网站下载到他的源代码(10K).

who

jQuery was created by John Resig.

第二章　15 Days of jQuery 比 window.onload 更快一些的载入

window.onload()是传统 javascript 里一个能吃苦耐劳的家伙。它长久以来一直被程序员们作为尽快解决客户端页面载入问题的捷径。

但有时候等待页面载入还是不够快。

只 有少数大型的图片文件会被快速的载入，而大部分大型的图片文件会使 window.onload()载入的很慢。所以当我为最近的网络营销创建一个 web 应 用程序的时候我不得希望更快一点。有一些围绕 window.onload()的新研究(比如 brother cake)的代码是一种快速的方式。如果你需要，可以试试。

但是如果你要做一些 DOM(文档对象模型)javascript 的编程，那么你为什么不试试 jQuery，它就像你自己亲自制作一个蛋糕，并品尝它。（双关 Brother Cake，俏皮话）。

jQuery 有一个用来作为 DOM 快速载入 javascript 的得心应手的小函数，那就是 ready… 他在页面加载完成之前执行。

```
$(document).ready(function(){
// Your code here...
});
```

你可以用他来载入任何你想要载入的 javascript，并不一定要保留 jQuery 的编码风格。让 jQuery 同时去执行多个函数也是可以的。

你以前可能见过类似于 init()之类的函数，你会发现 jQuery 会快很多。

在以后的教程里我们会一遍又一遍的用到这个函数。

OK 你现在可以尝试一下代码：(记得把 jQuery 引用进你的页面哦，不记得的话看看上篇。)

```
$(document).ready(function(){
alert("Congratluations!");
});
```

很 Easy，不是吗？ 用几分钟时间做的双色表格。

第三章 15 Days of jQuery(Day 2)---很容易的制作双色表格

这节本身没有太多的价值，重点在它提供的这个例子上。我将代码帖出来然后对重点部分注释一下：我们先来看看 Thewatchmakerproject 传统的做法：预览地址(你可以查看一下源代码)。再来看看 jQuery 是如何用 5 行代码来搞定的:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>StripingTable</title>
<script src="jquery-latest.pack.js" type="text/javascript"></script>
<!--将 jQuery 引用进来-->
<script type="text/javascript">
$(document).ready(function(){  //这个就是传说的 ready
          $(".stripe tr").mouseover(function(){
                //如果鼠标移到 class 为 stripe 的表格的 tr 上时，执行函数
                  $(this).addClass("over");}).mouseout(function(){
                                //给这行添加 class 值为 over，并且当鼠
标一出该行时执行函数
                  $(this).removeClass("over");})  //移除该行的 class
          $(".stripe tr:even").addClass("alt");
                //给 class 为 stripe 的表格的偶数行添加 class 值为 alt
});
</script>
<style>
th {
          background:#0066FF;
          color:#FFFFFF;
          line-height:20px;
          height:30px;
}

td {
          padding:6px 11px;
          border-bottom:1px solid #95bce2;
          vertical-align:top;
          text-align:center;
}

td * {
          padding:6px 11px;
}
```

```
tr.alt td {
          background:#ecf6fc;  /*这行将给所有的 tr 加上背景色*/
}

tr.over td {
          background:#bcd4ec;  /*这个将是鼠标高亮行的背景色*/
}

</style>
</head>

<body>
<table class="stripe" width="50%" border="0" cellspacing="0"
cellpadding="0">
<!--用 class="stripe"来标识需要使用该效果的表格-->
<thead>
    <tr>
      <th>姓名</th>
      <th>年龄</th>
      <th>QQ</th>
      <th>Email</th>
    </tr>
</thead>
<tbody>
    <tr>
      <td>邓国梁</td>
      <td>23</td>
      <td>31540205</td>
      <td>gl.deng@gmail.com</td>
    </tr>
    <tr>
      <td>邓国梁</td>
      <td>23</td>
      <td>31540205</td>
      <td>gl.deng@gmail.com</td>
    </tr>
    <tr>
      <td>邓国梁</td>
      <td>23</td>
      <td>31540205</td>
      <td>gl.deng@gmail.com</td>
    </tr>
    <tr>
```

```
      <td>邓国梁</td>
      <td>23</td>
      <td>31540205</td>
      <td>gl.deng@gmail.com</td>
    </tr>
    <tr>
      <td>邓国梁</td>
      <td>23</td>
      <td>31540205</td>
      <td>gl.deng@gmail.com</td>
    </tr>
    <tr>
      <td>邓国梁</td>
      <td>23</td>
      <td>31540205</td>
      <td>gl.deng@gmail.com</td>
    </tr>
</tbody>
</table>
<p>PS: 飘飘说我的 table 没有<thead>,我知错了...</p>
</body>
</html>
```

这里有一个 jQuery 的技巧不得不提一下：jQuery 的链式操作，什么是链式操作呢？ 我们来看看，本来应该写成这样子的：

```
$(".stripe tr").mouseover(function(){
          $(this).addClass("over");})
$(".stripe tr").mouseout(function(){
          $(this).removeClass("over"); })
```

但是我们写成了：

```
$(".stripe tr").mouseover(function(){
               $(this).addClass("over");}).mouseout(function(){
                   $(this).removeClass("over");})
```

因为鼠标移入移除都是发生在同一个对象上的，所以我们可以将发生在同一个对象上的动作连起来写，这样子如果有很多对象并且在他们身上发生了很多动作那么就会节省很多代码。（我暂时是这样理解的，也不知道对不对希望高手能够斧正。）

第四章 15 Days of jQuery(Day 3)---巧妙的伪装链接

今天的教程是草草完成的.我想把一些东西放在这 15 天的前面简单的讲讲,这样以来就可以使一些 js 新手不至于被一堆代码搞的晕头转向.事实上我是在即将结尾的时候才做出的这个决定.

目标

我们要使用 jQuery 去创建一小段代码,这段代码会把一个页面所有的超链接转换并且伪装起来.

为什么?

一些下属经销商认为，一部分用户有足够的悟性发现会员链接，并能尽量避免通过点击 URL 链接直接进入浏览器，从而“欺骗”下属经销商脱离代理（假设购买行为已经发生）

“老”办法

有很多下属经销商千方百计的掩饰他们的链接,避免人们通过链接找到他们.这些伎俩涉及到.htaccess 和服务器端的代码. 但对于本教程,我会将重点放到”老学校”的 javascript 上:

```
<a onMouseOver='window.status="http://www.merchant-url-here.com";
return true;' onMouseOut='window.status="Done"; return true;'
href="http://www.affiliate-url-here.com"
target="_blank">Link Text Here</a>
```

这段代码被用来在浏览器状态栏显示用户鼠标指向的链接地址.比如实际链接是 www.website.com?aff=123,则可以在状态栏显示 www.website.com.

问题

你是一个很活跃的下级经销商,你可能会以疯狂的速度给很多个页面添加链接.并且还要给每个链接添加这种效果那么你肯定会厌倦的.加入有一天你要修改任务栏里显示的链接的时候估计你会疯掉的.

jQuery 的解决办法

首先,我们想让 javascript 尽快的掩饰我们的链接所以我们应该先从这里开始:

```
<script src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
//code goes here
});
```

</script>

当 DOM 准备好的时候我们放在 ready 里的代码就开始执行了.

接下来

要给所有我们想伪装的链接添加一个 class,class 有助于 jQuery 帮我们找到需要伪装的链接而撇开其它不需要伪装的链接.title 有两个作用:当鼠标划过链接的时候会有一个小小的盒状提示显示 URL:www.affsite.com 并且同样的信息会显示在浏览器的状态栏(IE Only).

```
<p><a href="http://www.affsite.com?id=123"
title="http://www.affsite.com"
class="affLink">Super Duper Product</a></p>
```

告诉 jQuery 找到有 class=“affLink”的链接

```
$('a.affLink')
```

添加一个鼠标划过事件

```
$('a.affLink').mouseover(function(){window.status=this.title;return
true;})
```

简 单的说:找到 class=“affLink”的所有链接,当鼠标划过它们的时候改变浏览器状态栏信息为该链接 title 的内容.这个在 FireFox 并不 能正常的工作,只是在 IE 里起作用.在FireFox 的状态栏只是显示一个”Done”,或者更准确的说,鼠标划过超链接对状态栏并没有任何影响.我没有 更多的浏览器测试.

继续-mouseout jQuery 可以让我们用”链”的方式,像这样:

```
$('a.affLink').mouseover(function(){window.status=this.title;return
true;})
.mouseout(function(){window.status='Done';return true;});
```

这点代码告诉 jQuery 改变浏览器状态栏信息,或者当鼠标不再停留在链接上时返回”Done”. 如果你不适应 jQuery 的这种链的工作方式,那么你完全可以这样写:

```
$('a.affLink').mouseover(function(){window.status=this.title;return
true;});
$('a.affLink').mouseout(function(){window.status='Done';return
true;});
```

这就看你了.

把这些代码放到一起:

```
<script src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
$('a.affLink').mouseover(function(){window.status=this.title;return
true;})
.mouseout(function(){window.status='Done';return true;});
});
</script>
```

最后的想法 你们当中可能觉得今天的课程太简单了,有些还可能还是有点不太明白,因为你们不是二级经销商.

In “Days” to come you’ll see me tackle issues that get more involved and apply to almost anyone with a website - whether you monetize your traffic or not.

第五章 15 Days of jQuery 安全邮件列表

规则提到如何防止垃圾邮件:不要把你的邮件地址放到任何一个 mailto:链接中.在与垃圾邮件恶魔做斗争的过程中我们的网页设计师和程序员总结出了一些有创意的解决办法,让我们快速的看一些这些常见方法的缺点(或多或少有一些).

name [at-no-spam] website.com

问题:链接式的更方便,而且把邮件地址敲入收件人栏还有可能会出错.

联系方式

问题:你冒着这么大的风险就是因为有一个垃圾邮件借用你的帐户发送大量的垃圾邮件(除非你使用真正的安全邮件脚本).而这样就扼杀了那些只想给你发个简单邮件的用户.

javascript 加密

问题:你的邮件仍然暴露在光天化日之下,即使你使用了复杂的密码技术给它带上面具.还有谁希望为了发送一封邮件而启用第三方的解密网站,反正我是不会.

藏在一种简单的形式后面(有一个例子,但是打不开了.)http://simon.incutio.com/contact/我能想到的没有人…但是让我们看看是否我们能改进观念。

可能的解决办法:AJAX

我提供的解决方案将比目前设计师们使用的方法有如下优势:

- 易于实施
- 易于修改
- 还有一些小小的花哨的效果
- 不用第三方工具来加密邮件地址
- 没有邮件地址暴露在光天化日之下

最后我想说明一点,我认为电子邮件靠这种闪烁其词的加密手段来躲避垃圾邮件还是非常不明智的.在实践中,我认为电子邮件加密是相对安全的,但是客观事实是,电子邮件还是在 html 原代码里.

概念

1. 1.用 jQuery 从服务器上把 html 文件内容抓下来.
2. 2.把包含邮件链接的 html 文件放到好的容器中是一种简单的保护机制.

示例

我要示范一些例子来显示邮件链接地址,当访客点击一个按钮或者一个链接的时候,页面就会跳转到对应的那个例子里.

按钮点击--立即显示

链接点击--淡出

页面载入--淡出

页面载入--立即显示

(说明:所谓的立即显示,我的意思是说”没有花哨的效果而尽快的显示电子邮件地址”)

代码

这里发表非商业共创使用许可,如果你希望将代码使用在你的商业产品中时,请联系我.我正在一个新的 CMS for web designers 中使用它.

为什么这种方式比普通的 mailto 链接安全?

真 正的问题是垃圾邮件制造者会使用自动化软降从 html 源文件中寻找电子邮件链接,这种做法和 google 一样:使用相关链接. 他么就和我们大部分人一样懒惰.所以很难所他们不会拿个本子放在键盘旁边记下你的电子邮件地址. 请查看我提供的示例的源代码,你将不会在 html 里找到任何的邮件地址. 这几坚实的保证了你绝对不会收到垃圾邮件,只会从朋友或者浏览者那里收到邮件. 但是从页面中移除邮件地址,……

最后一点说明

先 仔细看看前面三个例子,你会看到我使用了 AJAX 回调函数来触发 slideDown() 和 show() 效果. 换句话说,我希望 AJAX 调用收到信息(html)时 jQuery 才开始 slideDown() 效果.把秘密粘贴到我们简单的服务段脚本并且等待服务器返回信息.

正确的方法:

```
$(document).ready(function(){
$.post('mailtoInfo.php',{
    pass: "secret"
},function(txt){
    $('div.email').html(txt);
    $('div.email').slideDown("slow");
});
});
```

错误的方法:

```
$(document).ready(function(){
$.post('mailtoInfo.php',{
    pass: "secret"
},function(txt){
    $('div.email').html(txt);
});
$('div.email').slideDown("slow");
});
```

第六章 15 Days of jQuery(Day 5)---包起来--懒人用 Jquery 生成的 HTML

这个让我们轻松的纪念日已经到来 - 我恨我在计算机前已经花了 48 个小时，我希望能够有另外一个 jQuery 来结束我的噩梦，并且让我上网更快。

当我一边“在用 Jquery 方法编写”和一边“进行复杂的文件上传”，我已经筋疲力尽。然而这两种操作的代码是一种较浅的，它只不过是你才刚刚开始解决的一些简单问题。

所以下来我开始介绍：

尽管我在我的网站用所有的 CSS 样式表去进行表格设计（也许这要花费两年半的时间或者更多），我已经用了很多我能找到的在这方面的信息。回到 2004 年 5 月(古代史)A list a part 有一篇《关于创建阴影的伟大教程(洋葱皮投影)》可以应用到任何图片,无论尺寸多大.

那片文章的应经不能再评论了,但还是有些人希望能够再出篇教程.

问题

一些 css 工程师用一些”不相干”的标记,就是为了使背景图片能够应用到每一个元素上.例如:

这里是 A list a part 用到的代码:

```
<div class="wrap1">
<div class="wrap2">
<div class="wrap3">
<img src="object.gif" alt="The object casting a shadow" />
</div>
</div>
</div>
```

所有这些 divs 充当一个给图片添加投影效果的”钩子”.这不见得好,我们可以把源代码精简成这样:

```
<img src="object.gif" class="dropshadow" alt="The object casting a shadow" />
```

按着这个思路…

目标

在这里,我要想你展示如何用 jQuery 轻而易举的将多于的标记从你的源代码中剔除.运用这个方法,让你的代码更加干净(更重要的是)将使以后变换布局容易的多.

解

这里,看看 jQuery 是如何击退这个问题的.

```
$(document).ready(function(){
$("img.dropshadow")
.wrap("<div class='wrap1'><div class='wrap2'>" +
"<div class='wrap3'></div></div></div>");
});
```

图片就可以保持这样了:

```
<img src="object.gif" class="dropshadow" alt="The object casting a
shadow" />
```

仔细看看

$(document).ready() 是 jQuery 版的 window.onload()

$(“img.dropshadow”) 告诉 jQuery 找到带有 class=“dropshadow”的图片,如果你想用一个 id 你可以这样: $(“img#dropshadow”)wrap() (wrap() tells jQuery to use the DOM (Document Object Method Model) to wrap the images with the class=”dropshadow” in the html inside the parenthesis.)

最后的结果

傻乎乎的图片,但是和 original Onion Skinned Drop Shadows 用的是一样的.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Onion Skin DropShadwo with jQuery</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1" />

<style>
.wrap0, .wrap1, .wrap2, .wrap3 {
    display:inline-table;
  /* \*/display:block;/**/
```

```
  }
.wrap0 {
  float:left;
    background:url(shadow.gif) right bottom no-repeat;
  }
.wrap1 {
    background:url(shadow180.gif) no-repeat;
  }
.wrap2 {
    background:url(corner_bl.gif) -18px 100% no-repeat;
  }
.wrap3 {
padding:10px 14px 14px 10px;
    background:url(corner_tr.gif) 100% -18px no-repeat;
  }
    body { background: #fff;}
</style>
<script src="jquery.js" type="text/javascript"></script>

<script>
$(document).ready(function(){
$("img.dropshadow")
.wrap("<div class='wrap0'><div class='wrap1'><div class='wrap2'>" +
"<div class='wrap3'></div></div></div></div>");
});
</script>
</head>

<body>
<h1>Onion Skinned - With jQuery</h1>
<p>First, the old-school, multiple divs hard coded into the html as seen
on the <a href="http://www.ploughdeep.com/onionskin/360.html">orignial
article</a>:</p>
<div class="wrap0">
<div class="wrap1">
    <div class="wrap2">
     <div class="wrap3">
      <img src="ball.jpg" alt="The object casting a shadow" />
     </div>
    </div>
</div>
</div>
```

```
<p style="clear:both;">And now, the jQuery method, which uses javascript to wrap the image at runtime:</p>
        <img src="ball.jpg" class = "dropshadow" alt="The object casting a shadow" />
        <p>View the source of this page and you'll see the huge difference in markup!</p>
</body>
</html>
```

(这里只是代码,没有图片.要看效果去这里)

jQuery 和其它解决方法的比较

jQuery 的网站上有一个到 Ajaxian 网站的链接,那里有用另外一个 javascrip 库创建的 Onion Skin Drop Shadow ,我相信他的代码复杂程度和代码量现在看来自不待言.我宁愿使用 jQuery.(怎么?你猜到了..)

平 心而论,没有一个库是对于每一个工作或每一段代码都是合适的.本教程不是为了证明 jQuery 是一切 javascrip 类库中的老大. 试试 Prototype, Scriptaculous, YUI, Rico, Behaviour, Moo.fx 和 the dozens 或者其它的.如果你找到了一个你用起来比较顺手的,那就去用它吧.

jQuery 对于我来说只是一个工具.我只是希望这个教程能够提供给你更多使用它的方法.

有关 jQuery 的工具

jQuery 用难以置信的简单来操作 DOM. 你应该花些时间看看 jQuery 能用来做什么,用下 append(), prepend(), before(), after(), html(), and remove().

来自 jQuery Docs

arap(String html)

把所有匹配的元素用其他元素的结构化标记包装起来。这种包装对于在文档中插入额外的结构化标记最有用，而且它不会破坏原始文档的语义品质。

这个函数的原理是检查提供的第一个元素（它是由所提供的 HTML 标记代码动态生成的），并在它的代码结构中找到最上层的祖先元素－－这个祖先元素就是包装元素。

当 HTML 标记代码中的元素包含文本时无法使用这个函数。因此，如果要添加文本应该在包装完成之后再行添加。

示例:

```
$("p").wrap("<div class='wrap'></div>");
```

HTML

```
<p>Test Paragraph.</p>
```

结果

```
<div class='wrap'><p>Test Paragraph.</p></div>
```

第七章 Days of jQuery 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！

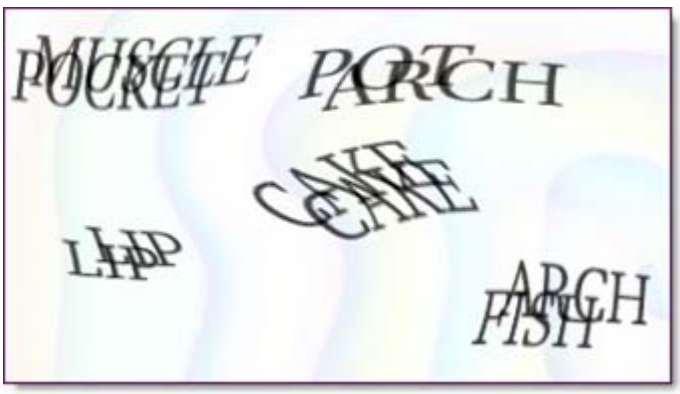
那么，今天我要在这篇教程里告诉大家的是一种在任何 contact forms 上添加一个额外安全层的简单方法－即使你没有使用我提供的超级安全、超级灵活的 Ultimate Form Mail。

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。

第八章 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);
}

$(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。

第九章 Days of jQuery 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()
});
```

第十章 Days of jQuery 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。

第十一章 15 Days of jQuery 使用 jQuery 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省略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。

第十二章 15 Days of jQuery 使用不苛刻的 javascript 代码实现多文件上传

好几个月以前，当我在追逐互联网上 AJAX 热潮的时候，我在 FiftyFourEleven 网站上发现了一篇使用创新的 javascript 代码实现当时正在困扰我的“ 单文件元素实现多文件上传”的文章。

所以当我想写作《15 天漫游 jQuery》的时候，我第一个想到的就是用 jQuery 实现这个功能。

接触易用性狂热爱好者

几天前当我检查网站记录的时候，发现了一条遗漏的文章 trackback。跟过去看的时候我发现我的两篇 jQuery 文章被作者引用来证明他为什么讨厌 javascript。

根据这个人的说法，任何工具或技术如果没有将易用性放在第一位都将成为垃圾。

尽管我很不同意这位仁兄一杆子打死的态度，但他还是让我对这篇详细教程有所留意。当我在编写一个简单网页效果的时候，我会尽量小心谨慎的处理。这样如果网站访客们决定关闭 javascript 代码执行功能的时候，他们仍然可以正常使用网站的功能。

关于第一价值的两个教程

- 使用一个文件输入元素实现多文件上传，并让整个交互过程流畅舒适。
- 让多文件上传更加人性化，但要避免以牺牲可用性为代价。关键在于使用不苛刻的 javascript 代码制作多文件输入区域。

演示

- 只有一个文件输入元素，但添加了 jQuery 和其他代码实现较为亲近用户的多文件上传功能。

演示一地址

- 在页面(x)html 代码中使用了多个文件输入元素，但通过 jQuery 调整为与第一个演示类似的显示页面效果。优点是代码是不苛刻的。。。即使关闭了 javascript 执行，用户也能上传多个文件。

演示二地址

解释

单文件输入框

jQuery 的$(document).ready() 函数的工作有两个：

在文档下载量最大的时候创建一个 div 元素。 查找文件上传框（假设这里只有一个），然后给它附上一个 onChange 事件。

```
$("input[@type=file]").change(function(){
doIt(this, fileMax);
});
```

doit()函数（简单又好记，呵呵~）检查是否达到了最大文件数量限制，如果不是，它会隐藏当前文件输入框，在父 div 里添加一个新的文件输入框，将输入框内的文件名使用 id “files_list”作为标记，在最后添加一个“删除”按钮。

在 DOM 树中导航，我使用 jQuery 的 parent()函数，然后用 remove()函数移除元素。我还使用了 append()和 prepend()函数分别添加文件名和新的输入框。

两个关键点

- 最大文件上传数量设定：

```
var fileMax = 3;
```

- 输入框必须有适当的定位措施：

```
<input type="file" class="upload" name="fileX[]"/>
```

这样弄以后输入框可疑由访问者决定添加还是删除，没有任何关于 id 或名称的操作。当这个窗体代码发送给服务器端脚本的时候，相关信息就已经被存放在了一个数组中了。

多文件输入框

首先，文件允许上传的数量由页面中的文件输入框的数量决定。其次，你仍然需要通过某种方法为每个输入框接收到的内容用一个数组存放。

```
<input type="file" class="upload" name="fileX[]"/>
```

第二个演示跟前面的比起来最大的不同在于，我遍历了每个文件输入框并在其内容有改动时执行 doit()函数。通过遍历每一个输入框，我可以为我的代码添加有用的额外信息：输入框内容在“堆栈”中的顺序。

换句话说，当这段代码执行时，它会特别指定第一个输入框，或者第二个，抑或第三个。

代码见下：

```
$("input[@type=file]:nthoftype("+
n+")")
```

jQuery 的灵活性允许我们使用 CSS 和 XPath 描述语句定位指定的元素位置。

你会发现当一个文件被选中时，文件输入框都会被文件名称覆盖。点击文件名就可以选择其他不同的文件。

第十三章 15 Days of jQuery Lightbox （插件）

Cody Lindley 移植的第一版“ Thickbox”让我第一次感受到了 jQuery 的魅力。后来他又做了一些 代码升级以修复若干跨浏览器的兼容性问题。

一些需要注意的地方

$(document).ready 取代了 TB_init() 函数，作用是在每个包含对象名“thickbox”的链接上附加一个 onClick 事件。

```
function TB_init(){
$("a.thickbox").click(function(){
var t = this.title || this.innerHTML || this.href;
TB_show(t,this.href);
this.blur();
return false;
});
```

当这些链接被点击时，TB_show()函数就将执行。

```
$("body")
.append("<div id='TB_overlay'></div><div id='TB_window'></div>");
$("#TB_overlay").click(TB_remove);
$(window).resize(TB_position);
$(window).scroll(TB_position);
$("#TB_overlay").show();
$("body").append("<div id='TB_load'><div id='TB_loadContent'><img
src='images/circle_animation.gif' /></div></div>");
```

如你所见，在文档 body 元素前添加了两个 div 元素。换句话说，这两个 div 元素将被添加在页面 html 代码的 body 关闭元素前。

覆 盖的 div 将使用一个特定的包含不透明外表的 CSS 文件指定表现。TB_window 的代码用来通过 AHAH 在页面中放置一张图片或者加入另一个页面。$(window).resize 和$(window).scroll 告诉 jQuery 在用户重新调整窗口大小或者拖动页面翻页的时候执行 TB_position 函数。这是保证 Thickbox 始终保持在窗口中心部位的手 段。

接下来，Cody 查询 url 的后缀。

```
var urlString =
/.jpg|.jpeg|.png|.gif|.html|.htm|.php|.cfm|.asp|.aspx|.jsp|.jst|.rb|.
txt/g;
var urlType = url.match(urlString);
```

```
if(urlType == '.jpg' || urlType == '.jpeg' || urlType == '.png' || urlType == '.gif'){//code to show images
```

如果这是一个图片文件，则 jQuery 的 append 函数会添加 html 代码到适当位置。

```
$("#TB_window").append("<a href='' id='TB_ImageOff' title='Close'><img id='TB_Image' src='"+url+"' width='"+imageWidth+"'
height='"+imageHeight+"'
alt='"+caption+"'/></a>"
+ "<div id='TB_caption'>"+caption+"</div><div
id='TB_closeWindow'><a href='#'
id='TB_closeWindowButton'>close</a></div>");
$("#TB_closeWindowButton").click(TB_remove);
```

另外，远程文件将使用 jQuery 的 load()函数导入。

```
$("#TB_ajaxContent").load(url, function(){
```

第十四章 15 Days of jQuery jQuery 表格

一位叫 Klaus 的朋友编写了一个小插件， 用 jQuery 实现可用性极佳的 javascript 表格。

设置好正确的(x)HTML 和 CSS 后，你可以像下面那样创建表格：

$.tabs(”container”); *first tab on by default 如果你像在默认位置“上方”再添加一个表格： $.tabs(”container”, 2);* second tab on

Klaus 这里 示例，你可以看看最终效果。

我的改版

我稍微修改了 Klaus 的代码，添加了一个简单的表单用来生成表格的表头。

用法：

非常简单。只需要输入每个表格的表头（最多 5 个），然后点击表单下方的按钮。下一个页面将生成结果 HTML 代码，你可以复制然后粘贴到文件中。

你还需要 下载 Klaus 网站的 CSS 文件，做些你自己的修改，当然还要上传 jQuery 框架库到你的服务器上。

这里是表格生成器的地址。

第十五章 15 Days of jQuery Javascript 工具提示

Cody Lindley ，Thickbox 的作者，日前发布了 jTip – jQuery 工具提示。

我对其中很多想法和思路拍案叫绝。我知道你已经看过很多类似的工具提示代码了。但是，Cody 的方法已经在我的工作中显露出了闪光点。

当我检查 HTML 代码时，我发现了一个大问题，可访问性。链接在 javascript 关闭的时候无法工作。我并不是倾向于一定要实现全面的可访问性，只是在这里我认为可以有其他更具亲和力的方式实现相同的功能。

尤其是，我个人不喜欢那种为了可访问性而去牺牲可用性来实现在提示框上链接另一个页面链接的方法。我喜欢这个提示框 – 不是对 Cody 不尊重，只是在我这里我“需要”它能够在各种情况下工作。

今天我要提供给大家的是 Cody 的工具提示代码的小小修改。如果你不是 Cody 工具提示的爱好者的话，我的改版对你来说也许不是很在意。但如果你喜欢他的作品同时希望它可以在 javascript 关闭的时候照常工作，这个也许是你需要的。

我的改动

让我产生修改想法的，是他的代码在 Yahoo 上的应用。我不喜欢他使用的代码：

```
<a href="yahoo.htm?width=175&amp;link=http://www.yahoo.com"
name="Before You Click..."
id="yahooCopy"
class="jTip">Go To Yahoo</a>
```

所以我重写了他的部分代码，成了现在这个样子：

```
<a href="http://www.yahoo.com"
rel="yahoo.htm?width=175&link=yahoo&name=Before%20
%20You%20Click..."
id="yahooCopy"
class="jTip">
Go To Yahoo</a>
```

我的示例

改进：HTML 标准校验

我的代码可以通过 w3.org 的测试

改进：命名

在我修改 Cody 的代码的时候我发现他使用了一个用来存储链接名称的叫做“title”的变量名，这会导致一些混淆。

我标出了这个命名问题，即使我认为这不过是个小小的失误。

改进：可用性

使用我的代码，你可以让每个提示框都含有真实链接地址到另一个文档，不管内部的还是外部的。或者你只是想要那个提示框，不想关心可用性，你同样可以让链接部分留空。

选择权在你。

感谢

Cody 提供了伟大的代码，帮助我节省了大量的时间和精力。我的修改只是对原有代码的轻微“调整”，希望朋友们喜欢。