第 28 卷第 1 期　　四川理工学院学报(自然科学版)　　Vol. 28　No. 1
2015 年 2 月　　Journal of Sichuan University of Science & Engineering(Natural Science Edition)　　Feb. 2015
文章编号:1673-1549(2015)01-0017-05　　DOI:10.11863/j.suse.2015.01.05

利用 HTML5 拖放技术实现多文件异步上传

刘耀钦

(郧阳师范高等专科学校计算机科学系, 湖北　十堰　442000)

摘　要:在传统的 Web 应用中实现本地文件的跨浏览器应用,不仅要编写大量冗余代码,而且很难获取良好的用户体验。HTML5 利用拖放技术将本地文件拖放至浏览器,通过 File API 的 dataTransfer 对象获取文件属性,然后将符合条件的文件添加至由 FormData 生成的模拟表单中,最后由 Ajax 的 XMLHttpRequest 将表单内容提交给服务器,进而实现本地文件的拖放和异步上传。这极大地提升了用户体验,减轻了服务器负载。基于拖放技术的文件异步上传是 Web 应用的一种新型技术形式,有较良好的应用前景。

关键词:HTML5;拖放;异步;Ajax;File API
中图分类号:TP311　　**文献标志码:**A

引　言

在 Web 应用系统中,若要获得本地文件句柄,需要通过打开对话框找到文件位置,然后获取文件路径,若需要上传的文件有多个,就需要多次重复操作,过程不仅繁琐,而且无法获得良好的用户体验。HTML5 作为 Web 前端开发的新标准超文本标记语言,不仅简化了传统 HTML 标签语法,而且提供了诸如智能表单、离线缓存、Canvas 绘图、拖放等核心技术和相应的 API,相比以前版本的 HTML,HTML5 极大地改善和增强了用户体验和开发功能[1],降低了浏览器对资源的占有率以及对插件的依赖[2],解决了 HTML4 在 Web 应用功能上的欠缺[3]。其中的拖放技术和 File API 既实现了同一 Web 页面内部不同对象之间的拖放,又建立起了本地文件与 Web 页面之间的关联,使得多文件直接拖放上传成为可能。

Ajax 的 *XmlHttpRequest*[4] 对象具有对 HTTP 协议的完全访问能力,包括对 POST、HEAD 以及 GET 等请求处理的能力,用来实现发送和接收 HTTP 请求与响应信息,经由 *XmlHttpRequest* 对象发送的请求不需要 Web 页面存在或返回一个 form 表单元素。文件的异步上传就是通过使用该对象直接与 Web 服务器进行通信[5],在页面不重载或不转向的情况下将文件上传请求发送至服务器并返回处理结果,进而实现文件异步传输。相对传统的浏览同步提交,文件拖拽的异步传输方式既提升了用户体验,又减轻了服务器负载,是 HTML5 环境下 Web 应用的一种新型技术形式,也是目前本地文件 Web 传输瓶颈的解决途径之一,有着较为广阔的应用前景。

1　原理和架构

1.1　原理

在 HTML5 以前,要实现 Web 对象的拖放操作不仅需要逐一实现 *mouseDown*、*mouseMove*、*mouseUp* 等一系列鼠标事件,而且还需要编写大量 JS 代码,编写完的应用程序体积庞大、代码冗余[6],更为主要的是这些冗余的代码仅仅能实现浏览器内部对象的相互拖放。HTML5 给文件的跨应用拖放提供了快速便捷的实现方式,只需要给被拖放元素添加“*draggable* = true”属性值即可,一

收稿日期:2014-12-05
基金项目:郧阳师范高等专科学校年重点科研项目(2014A01)
作者简介:刘耀钦(1980-),男,河南禹州人,讲师,硕士,主要从事信息安全及 Web 应用方面的研究,(E-mail)22556099@qq.com

个标记有该属性值的元素允许用户将其拖放至其他位置,同时也允许用户将其他对象拖放至自身区域内部,同时触发相应事件,用户可以从这些触发的事件中准确及时地获取元素从拖放开始至放下鼠标过程的各种状态和数据。图 1 反映了元素 A 在被拖放过程各事件响应流程。

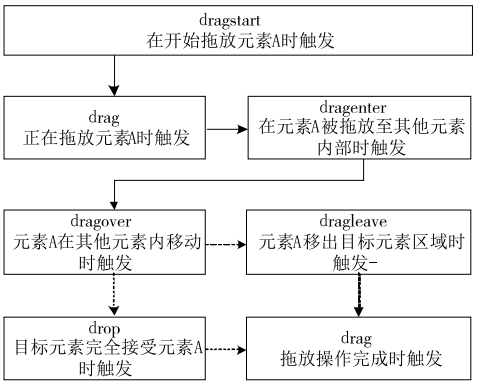

图 1 拖放过程各事件响应流程

在具体实现 Web 元素的拖放操作时,除了要逐一实现图 1 所示的各种相关拖放事件外,还要使用 preventDefault( )[7]方法通知浏览器不要执行与事件关联的默认动作。而很多 HTML 标签通常会有一个默认的动作,比如:

<a *id* = "link" *href* = "http://www.mypido.cn" > 百度

默认情况下,鼠标点击了这个链接即会打开百度首页,若在页面加入如下 JS 代码后就不会执行打开链接这个默认动作了:

var *link1* = document.getElementById("link");//获取 *id* 为 link 的元素

link1.onclick = function(*e*){//单击 *link1* 元素时触发

e.preventDefault( );//阻止或取消了 *link1* 元素的默认动作}

现在假定 Web 页中有 *red*、*blue*、*yellow* 三个 div,根据如下步骤即可实现将 *blue* 拖放至 *red*、*yellow* 任意一个 div 中。

Step 1:获取三个 div

var *yellow* = *document*.getElementById("yellow");//获取 id 为 yellow 的元素

var *blue* = *document*.getElementById("blue");//获取 id 为 blue 的元素

var *red* = *document*.getElementById("red");//获取 id 为 red 的元素

Step 2:设置手动 *blue* 时的数据

blue.ondragstart = function(*e*){//当拖动开始时触发,产生一个事件对象 *e*

e.dataTransfer.setData("*blueDiv*","blue");//设置 *blueDiv* 的值为 blue 元素}

dataTransfer 是事件对象的一个属性,用于从被拖拽元素向目标元素传递字符串格式的数据,有 setData( )和 getData( )两种方法,后者用于获取前者保存的数据。

Step 3:阻止 *yellow* 和 *red* 两元素默认事件

yellow.ondragover = function(*e*){//当有元素拖动至 *yellow* 元素上方时触发

e.preventDefault( );}

red.ondragover = function(*e*){ //当有元素拖动至 *red* 元素上方时触发

e.preventDefault( );}

Step 4:目标元素接受被拖动元素时

yellow.ondrop = addNewTag;//当 *yellow* 完全接受被拖动元素 *blue* 时触发 addNewTag

red.ondrop = addNewTag; //当 *red* 完全接受被拖动元素 *blue* 时触发 addNewTag

function addNewTag(*e*){

var *recDiv* = *document*.getElementById(*e*.dataTransfer.getData("*blueDiv*"));//获取之前保存的 *blueDiv* 数据

e.target.appendChild(*recDiv*);//在当前元素内部添加新元素 *recDiv*}

1.2 架构

HTML5 实现了本地文件的浏览器端应用,实现了 Web 应用程序集成的最终目标。接下来就需要在浏览器端通过 JS 加载和解析拖放来的文件,然后使用 Ajax 的 *XMLHttpRequest* 对象将文件异步传输至服务器。图 2 描述了多图片文件异步上传架构。

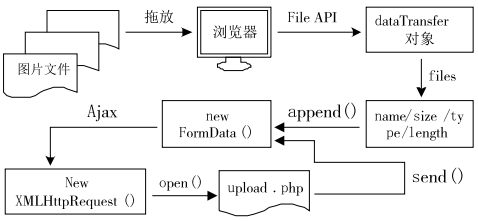

图 2 异步上传架构

2 dataTransfer 对象

在较早以前的 Web 时代, Web 应用中的本地文件访问功能需要借助 flash、silverlight[8]、ActiveX[9]等特定插件技术才能实现,例如,我们非常熟悉的 IE 浏览器就

需要借助 ActiveX 控件才能访问本地文件。使用了这些特定技术的 Web 应用的独立性和通用性都较差,而且很难实现跨平台、跨应用的统一表现。HTML5 的 File API 提供了一套完整的文件操作标准,包括 Web 应用客户端表现和文件操作对象的应用程序接口,使得基于 Web 应用的文件读写任务变得易为轻松。其中 *dataTransfer* 对象用于传输拖拽过程中的文件信息,表 1 和表 2 列出了其常用的属性和方法。

表 1　*dataTransfer* 对象主要方法

方　法	描　述	参　数
setData(*sFormat*,*sData*)	将 *sData* 存储于 *dataTransfer* 对象	*sFormat*:*URL*,*Text*
getData(*sFormat*)	从 *dataTransfer* 对象获取数据	*sFormat*:*URL*,*Text*
clearData([*sFormat*])	清空 *dataTransfer* 对象中的数据	*sFormat*:*Text*,*URL*,*File*,*HTML*,*Image*
setDragImage(*element*,*x*,*y*)	设置拖放操作的自定义图标	*element*:自定义图标,*x* - *y*:水平垂直方向的距离
addElement(*element*)	添加自定义图标	*element*:自定义图标

表 2　*dataTransfer* 对象主要属性

方　法	描　述	参　数
dropEffect[=*sCursorStyle*]	设置或获取被允许的操作效果类别	可选的,*copy*、*link*、*move*、*none*
effectAllowed[=*sEffect*]	设置或获取当前选定的操作效果类别	可选的,*copy*、*link*、*move*、*copylink*、*link*kmove、*all*、*none*、*uninitialized*
items	返回 *dataTransferItemLsit* 对象	无
types	返回 dragstart 事件中设置的数据格式	无
files	返回被拖动文件的清单 *fileList*	无

在文件拖拽过程中,可以通过 *dataTransfer* 对象来传输文件数量、类型、大小等数据,以供拖拽操作结束后对这些数据进行判断、显示等操作。下面程序以上传多张本地图片文件为例详细描述了拖放判断的过程。

```
var picContainer = document.getElementById('pic_container'); //获取显示图片的容器 DIV
picContainer.addEventListener("drop",function(e){ //给容器添加拖拽结束后事件监听
e.preventDefault(); //取消默认行为
var fileList = e.dataTransfer.files; //获取传输的图片文件列表
var n = fileList.length;//获取图片文件数量
for(i=0;i<n;i++){//判断图片大小及格式
if(fileList[i].size>1024*1000) error+='第'+(i+1)+'个图片体积过大\n';if(!/image/i.test(fileList[i].type)) error+='第'+(i+1)+'个不是图片格式\n';}
if(error){//如果有不符合条件的图片,则显示错误信息,并终止程序
alert(error);return false;}
for(i=0;i<n;i++){//显示拖拽来的图片
var img = window.webkitURL.createObjectURL(fileList[i]);//创建第 i+1 个图片的 url
var str = "<img src='"+img+"'><p>图片名称:"+fileList[i].name+"</p><p>大小:"+Math.floor((fileList[i].size)/1024)+"KB</p>";}
MYM("#pic_container").html(str);//将 str 字符串放置在 id 为 pic_container 容器内
```

3　异步处理

Web 应用的异步处理是指不向服务器提交完整页面的情况下实现 Web 页面数据的局部更新和提交。Ajax 可以构建更为动态和响应更灵敏的 Web 应用程序。其中的 *XMLHttpRequest* 对象提供了对 HTTP 协议的完整访问,包括对 POST 和 GET 请求的访问能力[5],异步或同步的返回 Web 服务器的响应,它不局限于对 XML 的操作,可以接收任何形式的数据,是基于 Ajax 的 Web 应用构架的核心组件。异步传输其实就相当于在客户端和服务器之间架设了一个中间层,可以把同步传输过程中的部分请求负担分摊给客户端处理,减轻了服务器的承载压力,缩短了响应时间,提高了程序的可操作性,增强了界面的友好性与直观性[10]。

XMLHttpRequest 对象中用于异步传输的核心方法有 open() 和 send(),分别用于初始化和发送 HTTP 参数和请求。open() 方法的具体结构如图 3 所示。

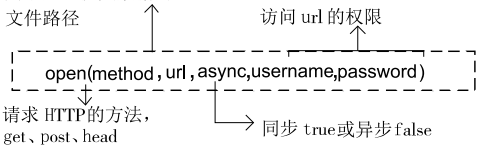

图 3　open() 方法结构

如果 open( )方法指定了 POST 或 GET 请求方式,send( )方法要指定具体的请求体。比如,要以 POST 方式提交上传的文件,就需要将文件所在的表单内容 *formData* 作为请求体,然后再 send(*formData*)。对于拖拽来的文件,可以通过使用 *FromData* 对象的 append( )方法将其逐一添加至动态模拟的表单控件中,即 *formData*. append("files[ ]", *e*. dataTransfer. files[*i*])。以下程序段将第 2 章中拖拽来的图片文件异步上传至服务器[11]。

xhr = new *XMLHttpRequest*( );//生成 *XMLHttpRequest* 对象的一个实例

xhr. open("post","/upload. php", true);//以 post 方式异步打开 upload. php 文件,该文件存储于服务器,用于处理客户端提交的表单数据

var *formData* = new *FormData*( );//生成 *FormData* 对象的一个实例来模拟表单

for(*i* = 0;*i* < *n*;*i* + +){//将拖拽来的文件逐一添加至 *formData* 中

formData. append("myfiles[ ]",fileList[*i*]);

}

xhr. send(*formData*);//发送表单数据

4 结 论

HTML5 的拖放技术不仅实现了页面内元素的相互拖拽,而且使得本地文件直接拖拽至浏览器成为可能。当文件拖拽操作结束时触发目标区域的 ondrop 事件,进而通过使用 HTML5 的 File API 中 *dataTransfer* 对象获取拖拽文件的 name/size/type 等属性,并将符合条件的文件显示在目标区域,同时逐一将这些文件添加在由 *FormData* 生成的模拟表单中,最后由 *XMLHttpRequest* 对象异步上传到服务器。相比传统方法,基于 HTML5 的拖放技术具有以下 3 个优点:

(1) Web 前端响应较快:基于客户端主机的判断验证减少了服务器验证返回结果的响应过程。

(2)容错能力较强:Ajax 使用异步交互技术,提高了网页的连续性和响应速度[12],即使进程中断也不会导致系统崩溃。

(3)用户体验较良好:减少了繁琐的操作过程,提高了传输效率,降低了用户等待时间。

尽管 HTML5 新技术具有较强的移植性和跨平台运行性,有效提高了程序可用性,但是由于目前浏览器技术的兼容性和支持性不够完善,再加上缺少一种统一的数据描述标准,使得这些新技术还不能较好的共享通用。然而,随着规范的逐步完善,HTML5 必将成为移动互联网领域的主宰者,带来更加丰富的网络应用。

参 考 文 献:

[1] 陶国荣.HTML5 实战[M].北京:机械工业出版社,2011.

[2] 刘华星,杨庚.HTML5 下一代 Web 开发标准研究[J].计算机技术与发展,2011,21(8):54-58 + 62.

[3] 童丽霞,何加铭,陈恳,等.基于 HTML5 技术的 Widget 引擎内容缓存模型及实现[J].计算机应用研究,2011,28(12):4625-4628.

[4] Network Working Group. Hypertext Transfer Protocol-HTTP/1.1 [EB/OL]. (2004-09-01) [2014-08-16]. http://www.w3.org/Protocols/rfc2616/rfc2616.html

[5] 吕国勇,史祥龙.基于嵌入式 Linux 和 Ajax 技术的 Web 异步交互设计[J].计算机应用,2013,33(S1):247-251.

[6] 邹梦丽,刘小勇.基于 Ajax 技术创建的异步方法应用研究[J].计算机时代,2014(3):33-35.

[7] 刘华煜,黄绍龙.通过 HTML5 的拖放简化网站的文件管理[J].洛阳师范学院学报,2014,33(5):72-73.

[8] 谭淇.基于 WCF 服务框架与 Silverligh 的 Web 应用研究[J].计算机与现代化,2011(1):79-81.

[9] 杨丁宁,肖晖,张玉清.基于 Fuzzing 的 ActiveX 控件漏洞挖掘技术研究[J].计算机研究与发展,2012,49(7):1525-1532.

[10] 戴维,蒋玉芳.基于 Ajax 技术实现 Web 异步树的应用研究[J].计算机与现代化,2011(2):148-149,153.

[11] 郭兆良.B/S 架构 Web 程序中 Ajax 异步传输技术的应用研究[J].电脑与信息技术,2013,21(6):41-43.

[12] 屈展,李婵.JSON 在 Ajax 数据交换中的应用研究[J].西安石油大学学报:自然科学版,2011,26(1):95-98.

(下转第 30 页)