Iframe的name是否和form中的target 值一样

JavaScript code

[?](http://bbs.csdn.net/topics/391002313#clipboardWindow)

|  |  |
| --- | --- |
| 1  2  3  4  5 | var fd = new FormData();      var fileList=e.dataTransfer.files;      for (var i = 0; i < fileList.length; i++) {       fd.append(file[], fileList[i]);  }  Iframe中的src=”javascript:false” ????????????? |

这里 fd.append("file[]", fileList[i]); 记得加引号

var oData = new FormData(document.getElementById("myForm"));

<form id="myForm"....

<input type="file" name="file1" />

<input type="file" name="file2" />

<input type="file" name="file3" />

</form>

表单里放几个就可以传几个.

如果你是通过 append 添加上去的, 只要多执行几次就可以了.  
比如下面这样:

//oData 为 new FormData();

oData.append("file1", document.getElementById('fileinfo1').files[0], 'newfile1.txt');

oData.append("file2", document.getElementById('fileinfo2').files[0], 'newfile2.txt');

oData.append("file3", document.getElementById('fileinfo3').files[0], 'newfile3.txt');

[html5多文件上传预览及进度条 思路](http://blog.csdn.net/dys1990/article/details/19481049)

2014-02-19 11:56 1969人阅读 [评论](http://blog.csdn.net/dys1990/article/details/19481049#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/dys1990/article/details/19481049#report)

http://static.blog.csdn.net/images/category_icon.jpg 分类：

个人（10） http://static.blog.csdn.net/images/arrow_triangle%20_down.jpg

版权声明：本文为博主原创文章，未经博主允许不得转载。

本文只考虑html5环境。因为本人的编程信条是，一切都要尽可能的简单。因此面对老旧的html4我是持摒弃态度。

html5的file控件支持multiple属性，但是可惜它的功能实在太有限了，它的multiple的意思是在同一次选择文件对话框中，可以多选文件，而不是通过可以打开多个文件选择对话框来选择多个文件。这样导致如果要多文件的话，必须这些文件在同一个文件夹下才能被多选上。因此这个multiple属性不靠谱。

关于文件上传进度条，使用XMLHttpRequest对象来上传文件并提供上传过程信息给浏览器客户端。

综合以上的条件限制，解决思路就产生了：

首先文件必须先于表单提交，而不是与表单同步。否则不仅用户体验很差，而且前端实现起来也非常麻烦（因为这样每选择一个文件，就需要再动态生成一个隐藏的新file控件）。

每当用户点击file控件时，使用change事件监测用户选择文件行为。在该事件中，将文件立刻进行ajax上传。这些文件存在于临时文件表中(我用[**数据库**](http://lib.csdn.net/base/14)存文件，原因就是因为简单)，上传成功后会返回该文件的主键到页面中，以便作为表单提交时文件参数。

至于预览，在html5中就太容易了，因为html5有个文件读取api，直接从用户磁盘上读取所选取的文件。而且通过判断文件的类型，如果是图片的话，就可以将它转成base64 URL型图片，直接显示在页面上。

当在表单中删除已选择的文件时，直接从页面表单提交参数中删除对应的主键即可，无需再从数据库中删除。因为我的临时文件表会定期清除过期文件。

# XMLHttpRequest upload属性

时间 2013-08-13 13:51:37  [WEB前端开发](http://www.tuicool.com/sites/aUbI3i)

原文  [http://www.css119.com/archives/1476](http://www.css119.com/archives/1476?utm_source=tuicool&utm_medium=referral)

主题 [JavaScript](http://www.tuicool.com/topics/11060004)

新版本的XMLHttpRequest对象，传送数据的时候，有一个progress事件，用来返回进度信息。

它分成上传和下载两种情况

１）下载的progress事件属于XMLHttpRequest对象

２）上传的progress事件属于XMLHttpRequest.upload对象。

[***我***](http://www.css119.com/archives/category/aboutme) 们先定义progress事件的回调函数。

xhr.onprogress = updateProgress;

xhr.up**load**.onprogress = updateProgress;

然后，在回调函数里面，使用这个事件的一些属性。

function updateProgress(**event**) {

**if**(**event**.lengthComputable) {

**var** percentComplete = **event**.loaded / **event**.total;

}

}

上面的代码中，event.total是需要传输的总字节，event.loaded是已经传输的字节。如果event.lengthComputable不为真，则event.total等于0。

与progress事件相关的，还有其他五个事件，可以分别指定回调函数：   
\* load事件：传输成功完成。

\* abort事件：传输被用户取消。

\* error事件：传输中出现错误。

\* loadstart事件：传输开始。

\* loadEnd事件：传输结束，但是不知道成功还是失败。

# [Angularjs]ng-file-upload上传文件

时间 2015-09-04 14:06:00  [博客园\_Wolfy](http://www.tuicool.com/sites/6rQRJrQ)

原文  [http://www.cnblogs.com/wolf-sun/p/4781673.html](http://www.cnblogs.com/wolf-sun/p/4781673.html?utm_source=tuicool&utm_medium=referral)

主题 [AngularJS](http://www.tuicool.com/topics/11060094)[HTML](http://www.tuicool.com/topics/11060000)

### 写在前面

最近在弄文档库的H5版，就查找了下相关的上传组件，发现了ng-upload的东东，推荐给大家。

### 系列文章

[***[Angularjs]ng-select和ng-options***](http://www.cnblogs.com/wolf-sun/p/4614532.html)

[***[Angularjs]ng-show和ng-hide***](http://www.cnblogs.com/wolf-sun/p/4620150.html)

[***[Angularjs]视图和路由（一）***](http://www.cnblogs.com/wolf-sun/p/4621854.html)

[***[Angularjs]视图和路由（二）***](http://www.cnblogs.com/wolf-sun/p/4638029.html)

[***[Angularjs]视图和路由（三）***](http://www.cnblogs.com/wolf-sun/p/4640725.html)

[***[Angularjs]视图和路由（四）***](http://www.cnblogs.com/wolf-sun/p/4656216.html)

[***[Angularjs]ng-class,ng-class-even,ng-class-odd***](http://www.cnblogs.com/wolf-sun/p/4676116.html)

[***[Angularjs]单页应用之分页***](http://www.cnblogs.com/wolf-sun/p/4713028.html)

[***[Angularjs]国际化***](http://www.cnblogs.com/wolf-sun/p/4732942.html)

[***[Angularjs]ng-repeat中使用ng-model遇到的问题***](http://www.cnblogs.com/wolf-sun/p/4749854.html)

[***[Angularjs]过滤器***](http://www.cnblogs.com/wolf-sun/p/4752004.html)

[Angularjs]ng-file-upload上传文件

### ng-file-upload

angular-file-upload 是一款轻量级的 AngularJS 文件上传工具，为不支持浏览器的 FileAPI polyfill 设计，使用 HTML5 直接进行文件上传。

#### 特性

* 支持上传进度，在上传的时候，可以取消或者中止，支持文件拖拽（HTML5），目录拖拽（weikit），CORS， PUT(html5) / POST 方法
* 支持使用 Flash polyfill [***FileAPI***](https://github.com/mailru/FileAPI) 跨浏览器上传 ( HTML5 和  non-HTML5 ) 。允许客户端在上传之前验证或者修改文件。
* 当文件的内容类型使用 $upload.http() 时，支持直接上传到 CouchDB，imgur 等等。支持 Angular http POST / PUT 请求的进度事件，更多内容请看 [***#88(comment)***](https://github.com/danialfarid/angular-file-upload/issues/88#issuecomment-31366487)
* Separate shim file loaded on demand for non-HTML5 code meaning no extra load/code if you just need HTML5 support. (Note that html5-shim.js is still needed for  progress event in  HTML5 browsers)
* 轻量级，使用常规的 $http 来上传（支持非 HTML5 浏览器），所以提供所有 Angular  $http 功能。

### 一个例子

需要的js文件，可以去这里下载： [***https://github.com/danialfarid/ng-file-upload***](https://github.com/danialfarid/ng-file-upload)

**<!DOCTYPE html>**

<html ng-app="app">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>文件上传</title>

<meta charset="utf-8" />

<script src="JS/angular.min.js"></script>

<script src="JS/ng-file-upload.min.js"></script>

<script src="JS/ng-file-upload-shim.min.js"></script>

<script>

**var** app = angular.module('app', ['ngFileUpload']);

app.controller('FileController', **function** ($scope, Upload) {

$scope.uploadImg = '';

*//提交*

$scope.submit = **function** () {

$scope.upload($scope.file);

};

$scope.upload = **function** (file) {

$scope.fileInfo = file;

Upload.upload({

*//服务端接收*

url: 'Ashx/UploadFile.ashx',

*//上传的同时带的参数*

data: { 'username': $scope.username },

file: file

}).progress(**function** (evt) {

*//进度条*

**var** progressPercentage = parseInt(100.0 \* evt.loaded / evt.total);

console.log('progess:' + progressPercentage + '%' + evt.config.file.name);

}).success(**function** (data, status, headers, config) {

*//上传成功*

console.log('file ' + config.file.name + 'uploaded. Response: ' + data);

$scope.uploadImg = data;

}).error(**function** (data, status, headers, config) {

*//上传失败*

console.log('error status: ' + status);

});

};

});

</script>

</head>

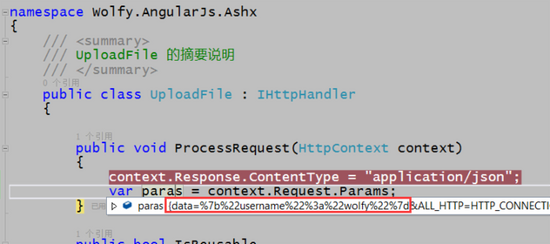
<body>

<form ng-controller="FileController">

<img src="{{uploadImg}}"/>

当前上传用户:<input type="text" placeholder="请输入您的名称" name="name" ng-model="username"/><div class="button" ngf-select ng-model="file" name="file" ngf-pattern="'image/\*" accept="image/\*" ngf-max-size="20MB" ngf-min-height="100">Select</div><button type="submit" ng-click="submit()">submit</button> {{fileInfo.name}}<br /> {{fileInfo.size}} </form></body></html>

简单测试



其中data中存的为我们上传文件的同时，需要的参数。

完整的例子，上传成功并在页面上进行预览。

**public** **class** UploadFile : IHttpHandler

{

**public** **void** ProcessRequest(HttpContext context)

{

context.Response.ContentType = "application/json";

**var** paras = context.Request.Params["data"];

JObject jobj = JObject.Parse(paras);

**string** strUserName = jobj["username"].ToString();

HttpFileCollection files = context.Request.Files;

**if** (files.Count > 0)

{

**var** file = files[0];

**string** fileExt = Path.GetExtension(file.FileName);

**string** fileNewName = Guid.NewGuid() + fileExt;

**string** strRelativeDir = "/Upload/" + strUserName;

**string** strDir = context.Request.MapPath(strRelativeDir);

**if** (!Directory.Exists(strDir))

{

Directory.CreateDirectory(strDir);

}

**string** strSavePath = Path.Combine(strDir, fileNewName);

file.SaveAs(strSavePath);

context.Response.Write(Path.Combine(strRelativeDir, fileNewName));

}

}

**public** **bool** IsReusable

{

**get**

{

**return** **false**;

}

}

}



http://img2.tuicool.com/QfMzuum.png!web

### 总结

使用ng-file-upload可以很好的与angularjs结合。在使用的时候，查找了一下angularjs相关的文件上传的例子，如果浏览器支持html5，那也可以很方便的制作进度条，另外该组件也支持多文件上传。推荐给大家。

你可以去这里进行下载： [***https://github.com/danialfarid/ng-file-upload***](https://github.com/danialfarid/ng-file-upload)

[**AngularJS 文件上传工具 angular-file-upload**](http://www.oschina.net/p/angular-file-upload)

angular-file-upload 是一款轻量级的 AngularJS 文件上传工具，为不支持浏览器的 FileAPI polyfill 设计，使用 HTML5 直接进行文件上传。

[在线演示](https://angular-file-upload.appspot.com/)

* 支持上传进度，在上传的时候，可以取消或者中止，支持文件拖拽（HTML5），目录拖拽（weikit），CORS，PUT(html5)/POST 方法
* 支持使用 Flash polyfill [FileAPI](https://github.com/mailru/FileAPI)  跨浏览器上传 (HTML5 和 non-HTML5) 。允许客户端在上传之前验证或者修改文件。
* 当文件的内容类型使用 $upload.http()时，支持直接上传到 CouchDB，imgur 等等。支持 Angular httpPOST/PUT 请求的进度事件，更多内容请看 [#88(comment)](https://github.com/danialfarid/angular-file-upload/issues/88#issuecomment-31366487)
* Separate shim file loaded on demand for non-HTML5 code meaning no extra load/code if you just need HTML5 support. (Note that html5-shim.js is still needed for progress event in HTML5 browsers)
* 轻量级，使用常规的 $http 来上传（支持非 HTML5 浏览器），所以提供所有 Angular $http 功能

HTML:

<!-- shim is needed to support upload progress/abort for HTML5 and non-HTML5 FormData browsers.--><!-- angular-file-upload-html5-shim.js could be used instead of angular-file-upload-shim if your app targets HTML5 browsers only (not IE8-9) --><!-- Note: shim.js MUST BE PLACED BEFORE angular.js and angular-file-upload.js AFTER angular.js--><script src="angular-file-upload-shim.min.js"></script>

<script src="angular.min.js"></script>

<script src="angular-file-upload.min.js"></script>

<div ng-controller="MyCtrl">

  <input type="text" ng-model="myModelObj">

  <input type="file" ng-file-select="onFileSelect($files)">

  <input type="file" ng-file-select="onFileSelect($files)" multiple accept="image/\*">

  <div class="button" ng-file-select="onFileSelect($files)" data-multiple="true"></div>

  <div ng-file-drop="onFileSelect($files)" ng-file-drag-over-class="optional-css-class-name-or-function"

        ng-show="dropSupported">drop files here</div>

  <div ng-file-drop-available="dropSupported=true"

        ng-show="!dropSupported">HTML5 Drop File is not supported!</div>

  <button ng-click="upload.abort()">Cancel Upload</button>

</div>

JS:

//inject angular file upload directives and service.angular.module('myApp', ['angularFileUpload']);var MyCtrl = [ '$scope', '$upload', function($scope, $upload) {

  $scope.onFileSelect = function($files) {    //$files: an array of files selected, each file has name, size, and type.

    for (var i = 0; i < $files.length; i++) {      var file = $files[i];

      $scope.upload = $upload.upload({

        url: 'server/upload/url', //upload.php script, node.js route, or servlet url

        //method: 'POST' or 'PUT',

        //headers: {'header-key': 'header-value'},

        //withCredentials: true,

        data: {myObj: $scope.myModelObj},

        file: file, // or list of files ($files) for html5 only

        //fileName: 'doc.jpg' or ['1.jpg', '2.jpg', ...] // to modify the name of the file(s)

        // customize file formData name ('Content-Disposition'), server side file variable name.

        //fileFormDataName: myFile, //or a list of names for multiple files (html5). Default is 'file'

        // customize how data is added to formData. See #40#issuecomment-28612000 for sample code

        //formDataAppender: function(formData, key, val){}

      }).progress(function(evt) {        console.log('percent: ' + parseInt(100.0 \* evt.loaded / evt.total));

      }).success(function(data, status, headers, config) {        // file is uploaded successfully

        console.log(data);

      });      //.error(...)

      //.then(success, error, progress);

      // access or attach event listeners to the underlying XMLHttpRequest.

      //.xhr(function(xhr){xhr.upload.addEventListener(...)})

    }    /\* alternative way of uploading, send the file binary with the file's content-type.       Could be used to upload files to CouchDB, imgur, etc... html5 FileReader is needed.        It could also be used to monitor the progress of a normal http post/put request with large data\*/

    // $scope.upload = $upload.http({...})  see 88#issuecomment-31366487 for sample code.

  };

}];

# [input file类型，文件类型的限制](http://www.cnblogs.com/Joans/p/3158582.html)

直接限制input type='file'的文件类型限制，通过accept属性进行设定,多个类型用逗号分隔开,因为accept是html5的新特性，所以火狐和IE的支持就显得单薄了，

如:

accept可以指定如下信息：

|  |
| --- |
|  |
| \*.3gpp | audio/3gpp, video/3gpp | 3GPP Audio/Video |
| \*.ac3 | audio/ac3 | AC3 Audio |
| \*.asf | allpication/vnd.ms-asf | Advanced Streaming Format |
| \*.au | audio/basic | AU Audio |
| \*.css | text/css | Cascading Style Sheets |
| \*.csv | text/csv | Comma Separated Values |
| \*.doc | application/msword | MS Word Document |
| \*.dot | application/msword | MS Word Template |
| \*.dtd | application/xml-dtd | Document Type Definition |
| \*.dwg | image/vnd.dwg | AutoCAD Drawing Database |
| \*.dxf | image/vnd.dxf | AutoCAD Drawing Interchange Format |
| \*.gif | image/gif | Graphic Interchange Format |
| \*.htm | text/html | HyperText Markup Language |
| \*.html | text/html | HyperText Markup Language |
| \*.jp2 | image/jp2 | JPEG-2000 |
| \*.jpe | image/jpeg | JPEG |
| \*.jpeg | image/jpeg | JPEG |
| \*.jpg | image/jpeg | JPEG |
| \*.js | text/javascript, application/javascript | JavaScript |
| \*.json | application/json | JavaScript Object Notation |
| \*.mp2 | audio/mpeg, video/mpeg | MPEG Audio/Video Stream, Layer II |
| \*.mp3 | audio/mpeg | MPEG Audio Stream, Layer III |
| \*.mp4 | audio/mp4, video/mp4 | MPEG-4 Audio/Video |
| \*.mpeg | video/mpeg | MPEG Video Stream, Layer II |
| \*.mpg | video/mpeg | MPEG Video Stream, Layer II |
| \*.mpp | application/vnd.ms-project | MS Project Project |
| \*.ogg | application/ogg, audio/ogg | Ogg Vorbis |
| \*.pdf | application/pdf | Portable Document Format |
| \*.png | image/png | Portable Network Graphics |
| \*.pot | application/vnd.ms-powerpoint | MS PowerPoint Template |
| \*.pps | application/vnd.ms-powerpoint | MS PowerPoint Slideshow |
| \*.ppt | application/vnd.ms-powerpoint | MS PowerPoint Presentation |
| \*.rtf | application/rtf, text/rtf | Rich Text Format |
| \*.svf | image/vnd.svf | Simple Vector Format |
| \*.tif | image/tiff | Tagged Image Format File |
| \*.tiff | image/tiff | Tagged Image Format File |
| \*.txt | text/plain | Plain Text |
| \*.wdb | application/vnd.ms-works | MS Works Database |
| \*.wps | application/vnd.ms-works | Works Text Document |
| \*.xhtml | application/xhtml+xml | Extensible HyperText Markup Language |
| \*.xlc | application/vnd.ms-excel | MS Excel Chart |
| \*.xlm | application/vnd.ms-excel | MS Excel Macro |
| \*.xls | application/vnd.ms-excel | MS Excel Spreadsheet |
| \*.xlt | application/vnd.ms-excel | MS Excel Template |
| \*.xlw | application/vnd.ms-excel | MS Excel Workspace |
| \*.xml | text/xml, application/xml | Extensible Markup Language |
| \*.zip | aplication/zip | Compressed Archive |

作者：Joan

出处：<http://www.cnblogs.com/Joans/>

本文版权归作者和博客园共有，欢迎转载，但未经作者同意必须保留此段声明，且在文章页面明显位置给出原文连接，否则保留追究法律责任的权利。

### List of MIME Types by Content Type

### Applications

| **Application** | **MIME Type** | **File Extension** |
| --- | --- | --- |
| Corel Envoy | application/envoy | evy |
| fractal image file | application/fractals | fif |
| Windows print spool file | application/futuresplash | spl |
| HTML application | application/hta | hta |
| Atari ST Program | application/internet-property-stream | acx |
| BinHex encoded file | application/mac-binhex40 | hqx |
| Word document | application/msword | doc |
| Word document template | application/msword | dot |
|  | application/octet-stream | \* |
| binary disk image | application/octet-stream | bin |
| Java class file | application/octet-stream | class |
| Disk Masher image | application/octet-stream | dms |
| executable file | application/octet-stream | exe |
| LHARC compressed archive | application/octet-stream | lha |
| LZH compressed file | application/octet-stream | lzh |
| CALS raster image | application/oda | oda |
| ActiveX script | application/olescript | axs |
| Acrobat file | application/pdf | pdf |
| Outlook profile file | application/pics-rules | prf |
| certificate request file | application/pkcs10 | p10 |
| certificate revocation list file | application/pkix-crl | crl |
| Adobe Illustrator file | application/postscript | ai |
| postscript file | application/postscript | eps |
| postscript file | application/postscript | ps |
| rich text format file | application/rtf | rtf |
| set payment initiation | application/set-payment-initiation | setpay |
| set registration initiation | application/set-registration-initiation | setreg |
| Excel Add-in file | application/vnd.ms-excel | xla |
| Excel chart | application/vnd.ms-excel | xlc |
| Excel macro | application/vnd.ms-excel | xlm |
| Excel spreadsheet | application/vnd.ms-excel | xls |
| Excel template | application/vnd.ms-excel | xlt |
| Excel worspace | application/vnd.ms-excel | xlw |
| Outlook mail message | application/vnd.ms-outlook | msg |
| serialized certificate store file | application/vnd.ms-pkicertstore | sst |
| Windows catalog file | application/vnd.ms-pkiseccat | cat |
| stereolithography file | application/vnd.ms-pkistl | stl |
| PowerPoint template | application/vnd.ms-powerpoint | pot |
| PowerPoint slide show | application/vnd.ms-powerpoint | pps |
| PowerPoint presentation | application/vnd.ms-powerpoint | ppt |
| Microsoft Project file | application/vnd.ms-project | mpp |
| WordPerfect macro | application/vnd.ms-works | wcm |
| Microsoft Works database | application/vnd.ms-works | wdb |
| Microsoft Works spreadsheet | application/vnd.ms-works | wks |
| Microsoft Works word processsor document | application/vnd.ms-works | wps |
| Windows help file | application/winhlp | hlp |
| binary CPIO archive | application/x-bcpio | bcpio |
| computable document format file | application/x-cdf | cdf |
| Unix compressed file | application/x-compress | z |
| gzipped tar file | application/x-compressed | tgz |
| Unix CPIO archive | application/x-cpio | cpio |
| Photoshop custom shapes file | application/x-csh | csh |
| Kodak RAW image file | application/x-director | dcr |
| Adobe Director movie | application/x-director | dir |
| Macromedia Director movie | application/x-director | dxr |
| device independent format file | application/x-dvi | dvi |
| Gnu tar archive | application/x-gtar | gtar |
| Gnu zipped archive | application/x-gzip | gz |
| hierarchical data format file | application/x-hdf | hdf |
| internet settings file | application/x-internet-signup | ins |
| IIS internet service provider settings | application/x-internet-signup | isp |
| ARC+ architectural file | application/x-iphone | iii |
| JavaScript file | application/x-javascript | js |
| LaTex document | application/x-latex | latex |
| Microsoft Access database | application/x-msaccess | mdb |
| Windows CardSpace file | application/x-mscardfile | crd |
| CrazyTalk clip file | application/x-msclip | clp |
| dynamic link library | application/x-msdownload | dll |
| Microsoft media viewer file | application/x-msmediaview | m13 |
| Steuer2001 file | application/x-msmediaview | m14 |
| multimedia viewer book source file | application/x-msmediaview | mvb |
| Windows meta file | application/x-msmetafile | wmf |
| Microsoft Money file | application/x-msmoney | mny |
| Microsoft Publisher file | application/x-mspublisher | pub |
| Turbo Tax tax schedule list | application/x-msschedule | scd |
| FTR media file | application/x-msterminal | trm |
| Microsoft Write file | application/x-mswrite | wri |
| computable document format file | application/x-netcdf | cdf |
| Mastercam numerical control file | application/x-netcdf | nc |
| MSX computers archive format | application/x-perfmon | pma |
| performance monitor counter file | application/x-perfmon | pmc |
| process monitor log file | application/x-perfmon | pml |
| Avid persistant media record file | application/x-perfmon | pmr |
| Pegasus Mail draft stored message | application/x-perfmon | pmw |
| personal information exchange file | application/x-pkcs12 | p12 |
| PKCS #12 certificate file | application/x-pkcs12 | pfx |
| PKCS #7 certificate file | application/x-pkcs7-certificates | p7b |
| software publisher certificate file | application/x-pkcs7-certificates | spc |
| certificate request response file | application/x-pkcs7-certreqresp | p7r |
| PKCS #7 certificate file | application/x-pkcs7-mime | p7c |
| digitally encrypted message | application/x-pkcs7-mime | p7m |
| digitally signed email message | application/x-pkcs7-signature | p7s |
| Bash shell script | application/x-sh | sh |
| Unix shar archive | application/x-shar | shar |
| Flash file | application/x-shockwave-flash | swf |
| Stuffit archive file | application/x-stuffit | sit |
| system 5 release 4 CPIO file | application/x-sv4cpio | sv4cpio |
| system 5 release 4 CPIO checksum data | application/x-sv4crc | sv4crc |
| consolidated Unix file archive | application/x-tar | tar |
| Tcl script | application/x-tcl | tcl |
| LaTeX source document | application/x-tex | tex |
| LaTeX info document | application/x-texinfo | texi |
| LaTeX info document | application/x-texinfo | texinfo |
| unformatted manual page | application/x-troff | roff |
| Turing source code file | application/x-troff | t |
| TomeRaider 2 ebook file | application/x-troff | tr |
| Unix manual | application/x-troff-man | man |
| readme text file | application/x-troff-me | me |
| 3ds Max script file | application/x-troff-ms | ms |
| uniform standard tape archive format file | application/x-ustar | ustar |
| source code | application/x-wais-source | src |
| internet security certificate | application/x-x509-ca-cert | cer |
| security certificate | application/x-x509-ca-cert | crt |
| DER certificate file | application/x-x509-ca-cert | der |
| public key security object | application/ynd.ms-pkipko | pko |
| zipped file | application/zip | zip |

### Sound Files

| **Application** | **MIME Type** | **File Extension** |
| --- | --- | --- |
| audio file | audio/basic | au |
| sound file | audio/basic | snd |
| midi file | audio/mid | mid |
| media processing server studio | audio/mid | rmi |
| MP3 file | audio/mpeg | mp3 |
| audio interchange file format | audio/x-aiff | aif |
| compressed audio interchange file | audio/x-aiff | aifc |
| audio interchange file format | audio/x-aiff | aiff |
| media playlist file | audio/x-mpegurl | m3u |
| Real Audio file | audio/x-pn-realaudio | ra |
| Real Audio metadata file | audio/x-pn-realaudio | ram |
| WAVE audio file | audio/x-wav | wav |

### Image Files

| **Application** | **MIME Type** | **File Extension** |
| --- | --- | --- |
| Bitmap | image/bmp | bmp |
| compiled source code | image/cis-cod | cod |
| graphic interchange format | image/gif | gif |
| image file | image/ief | ief |
| JPEG image | image/jpeg | jpe |
| JPEG image | image/jpeg | jpeg |
| JPEG image | image/jpeg | jpg |
| JPEG file interchange format | image/pipeg | jfif |
| scalable vector graphic | image/svg+xml | svg |
| TIF image | image/tiff | tif |
| TIF image | image/tiff | tiff |
| Sun raster graphic | image/x-cmu-raster | ras |
| Corel metafile exchange image file | image/x-cmx | cmx |
| icon | image/x-icon | ico |
| portable any map image | image/x-portable-anymap | pnm |
| portable bitmap image | image/x-portable-bitmap | pbm |
| portable graymap image | image/x-portable-graymap | pgm |
| portable pixmap image | image/x-portable-pixmap | ppm |
| RGB bitmap | image/x-rgb | rgb |
| X11 bitmap | image/x-xbitmap | xbm |
| X11 pixmap | image/x-xpixmap | xpm |
| X-Windows dump image | image/x-xwindowdump | xwd |

### Mail Message Files

| **Application** | **MIME Type** | **File Extension** |
| --- | --- | --- |
| MHTML web archive | message/rfc822 | mht |
| MIME HTML file | message/rfc822 | mhtml |
| Windows Live Mail newsgroup file | message/rfc822 | nws |

### Text Files

| **Application** | **MIME Type** | **File Extension** |
| --- | --- | --- |
| Cascading Style Sheet | text/css | css |
| H.323 internet telephony file | text/h323 | 323 |
| HTML file | text/html | htm |
| HTML file | text/html | html |
| Exchange streaming media file | text/html | stm |
| NetMeeting user location service file | text/iuls | uls |
| BASIC source code file | text/plain | bas |
| C/C++ source code file | text/plain | c |
| C/C++/Objective C header file | text/plain | h |
| text file | text/plain | txt |
| rich text file | text/richtext | rtx |
| Scitext continuous tone file | text/scriptlet | sct |
| tab separated values file | text/tab-separated-values | tsv |
| hypertext template file | text/webviewhtml | htt |
| HTML component file | text/x-component | htc |
| TeX font encoding file | text/x-setext | etx |
| vCard file | text/x-vcard | vcf |

[Ads](http://webdesign.about.com/od/multimedia/a/mime-types-by-content-type.htm)

* [Convert PDF File to HTML](http://webdesign.about.com/z/js/o.htm?k=convert%20pdf%20file%20to%20html&SUName=webdesign&d=Convert%20PDF%20File%20to%20HTML&r=http%3A%2F%2Fwebdesign.about.com%2Fod%2Fmultimedia%2Fa%2Fmime-types-by-content-type.htm&title=Find%2520MIME%2520Types%2520by%2520Content%2520Type%2520(Web%2520Design)&google_rt=ChBXHOdsAAp8rwq9EVOpAUGbEhhDb252ZXJ0IFBERiBGaWxlIHRvIEhUTUwaCJUrdadB1WTkKAFSEwjB6PzRzKfMAhWMJr0KHRhPBcg)
* [File Extension PDF](http://webdesign.about.com/z/js/o.htm?k=file%20extension%20pdf&SUName=webdesign&d=File%20Extension%20PDF&r=http%3A%2F%2Fwebdesign.about.com%2Fod%2Fmultimedia%2Fa%2Fmime-types-by-content-type.htm&title=Find%2520MIME%2520Types%2520by%2520Content%2520Type%2520(Web%2520Design)&google_rt=ChBXHOdsAAp8wAq9EVOpAUGbEhJGaWxlIEV4dGVuc2lvbiBQREYaCFUMZ6Nnpe7wKAFSEwjB6PzRzKfMAhWMJr0KHRhPBcg)
* [Are File](http://webdesign.about.com/z/js/o.htm?k=are%20file&SUName=webdesign&d=Are%20File&r=http%3A%2F%2Fwebdesign.about.com%2Fod%2Fmultimedia%2Fa%2Fmime-types-by-content-type.htm&title=Find%2520MIME%2520Types%2520by%2520Content%2520Type%2520(Web%2520Design)&google_rt=ChBXHOdsAAp8wwq9EVOpAUGbEghBcmUgRmlsZRoITGHDW5gDHNwoAVITCMHo_NHMp8wCFYwmvQodGE8FyA)
* [Binary File Viewer](http://webdesign.about.com/z/js/o.htm?k=binary%20file%20viewer&SUName=webdesign&d=Binary%20File%20Viewer&r=http%3A%2F%2Fwebdesign.about.com%2Fod%2Fmultimedia%2Fa%2Fmime-types-by-content-type.htm&title=Find%2520MIME%2520Types%2520by%2520Content%2520Type%2520(Web%2520Design)&google_rt=ChBXHOdsAAp8xgq9EVOpAUGbEhJCaW5hcnkgRmlsZSBWaWV3ZXIaCBKd_ReLnuLGKAFSEwjB6PzRzKfMAhWMJr0KHRhPBcg)
* [JPEG File Format](http://webdesign.about.com/z/js/o.htm?k=jpeg%20file%20format&SUName=webdesign&d=JPEG%20File%20Format&r=http%3A%2F%2Fwebdesign.about.com%2Fod%2Fmultimedia%2Fa%2Fmime-types-by-content-type.htm&title=Find%2520MIME%2520Types%2520by%2520Content%2520Type%2520(Web%2520Design)&google_rt=ChBXHOdsAAp8yAq9EVOpAUGbEhBKUEVHIEZpbGUgRm9ybWF0GghPqu8cmo7mWCgBUhMIwej80cynzAIVjCa9Ch0YTwXI)

### Video Files

| **Application** | **MIME Type** | **File Extension** |
| --- | --- | --- |
| MPEG-2 audio file | video/mpeg | mp2 |
| MPEG-2 audio file | video/mpeg | mpa |
| MPEG movie file | video/mpeg | mpe |
| MPEG movie file | video/mpeg | mpeg |
| MPEG movie file | video/mpeg | mpg |
| MPEG-2 video stream | video/mpeg | mpv2 |
| Apple QuickTime movie | video/quicktime | mov |
| Apple QuickTime movie | video/quicktime | qt |
| Logos library system file | video/x-la-asf | lsf |
| streaming media shortcut | video/x-la-asf | lsx |
| advanced systems format file | video/x-ms-asf | asf |
| ActionScript remote document | video/x-ms-asf | asr |
| Microsoft ASF redirector file | video/x-ms-asf | asx |
| audio video interleave file | video/x-msvideo | avi |
| Apple QuickTime movie | video/x-sgi-movie | movie |

### Virtual World Files

| **Application** | **MIME Type** | **File Extension** |
| --- | --- | --- |
| Flare decompiled actionscript file | x-world/x-vrml | flr |
| VRML file | x-world/x-vrml | vrml |
| VRML world | x-world/x-vrml | wrl |
| compressed VRML world | x-world/x-vrml | wrz |
| 3ds max XML animation file | x-world/x-vrml | xaf |
| Reality Lab 3D image file | x-world/x-vrml | xof |

# js判断文件格式及大小

//判断照片大小

function getPhotoSize(obj){  
photoExt=obj.value.substr(obj.value.lastIndexOf(".")).toLowerCase();//获得文件后缀名  
if(photoExt!='.jpg'){  
alert("请上传后缀名为jpg的照片!");  
return false;  
}  
var fileSize = 0;  
var isIE = /msie/i.test(navigator.userAgent) && !window.opera;   
if (isIE && !obj.files) {   
var filePath = obj.value;   
var fileSystem = new ActiveXObject("Scripting.FileSystemObject");   
var file = fileSystem.GetFile (filePath);   
fileSize = file.Size;   
}else {   
fileSize = obj.files[0].size;   
}  
fileSize=Math.round(fileSize/1024\*100)/100; //单位为KB  
if(fileSize>=10){  
alert("照片最大尺寸为10KB，请重新上传!");  
return false;  
}  
}  
  
jsp页面：

<input type="file" id="imgFile" name="upload" style="width:150px;" onchange="getPhotoSize(this)" />

function checkImgType(ths) {  
 if (ths.value == "") {  
 alert("请上传图片");  
 return false;  
 } else {  
 if (!/\.(gif|jpg|jpeg|png|GIF|JPG|PNG)$/.test(ths.value)) {  
 alert("图片类型必须是.gif,jpeg,jpg,png中的一种");  
 this.value="";  
 return false;  
 }  
 }  
 return true;  
 }

### [javascript判断上传图片文件类型（后缀名）](http://java110eye.iteye.com/blog/1141982)

**博客分类：**

* [JavaScript](http://java110eye.iteye.com/category/168020)

[javascript](http://www.iteye.com/blogs/tag/javascript)[文件判断](http://www.iteye.com/blogs/tag/%E6%96%87%E4%BB%B6%E5%88%A4%E6%96%AD)[图片判断](http://www.iteye.com/blogs/tag/%E5%9B%BE%E7%89%87%E5%88%A4%E6%96%AD)[后缀名判断](http://www.iteye.com/blogs/tag/%E5%90%8E%E7%BC%80%E5%90%8D%E5%88%A4%E6%96%AD)

1、根本是正则表达式,不存在浏览器问题

**Js代码  [收藏代码](javascript:void())**

1. /\*
2. \* 判断图片类型
3. \*
4. \* @param ths
5. \*          type="file"的javascript对象
6. \* @return true-符合要求,false-不符合
7. \*/
8. **function** checkImgType(ths){
9. **if** (ths.value == "") {
10. alert("请上传图片");
11. **return** **false**;
12. } **else** {
13. **if** (!/\.(gif|jpg|jpeg|png|GIF|JPG|PNG)$/.test(ths.value)) {
14. alert("图片类型必须是.gif,jpeg,jpg,png中的一种");
15. ths.value = "";
16. **return** **false**;
17. }
18. }
19. **return** **true**;
20. }

 2、以下判断像素的在firefox下不行,主要是这句代码document.body.**insertAdjacentElement("beforeEnd"**, img);

暂时未找到解决办法....

**Js代码  [收藏代码](javascript:void())**

1. /\*
2. \* 判断图片大小
3. \*
4. \* @param ths
5. \*          type="file"的javascript对象
6. \* @param width
7. \*          需要符合的宽
8. \* @param height
9. \*          需要符合的高
10. \* @return true-符合要求,false-不符合
11. \*/
12. **function** checkImgPX(ths, width, height) {
13. **var** img = **null**;
14. img = document.createElement("img");
15. document.body.insertAdjacentElement("beforeEnd", img); // firefox不行
16. img.style.visibility = "hidden";
17. img.src = ths.value;
18. **var** imgwidth = img.offsetWidth;
19. **var** imgheight = img.offsetHeight;
21. alert(imgwidth + "," + imgheight);
23. **if**(imgwidth != width || imgheight != height) {
24. alert("图的尺寸应该是" + width + "x"+ height);
25. ths.value = "";
26. **return** **false**;
27. }
28. **return** **true**;
29. }

# [HTML5 之文件操作(file)](http://blog.csdn.net/oscar999/article/details/37499743)

2014-07-09 08:11 34303人阅读 [评论](http://blog.csdn.net/oscar999/article/details/37499743#comments)(5) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/oscar999/article/details/37499743#report)

http://static.blog.csdn.net/images/category_icon.jpg 分类：

Web-编程语言（152） http://static.blog.csdn.net/images/arrow_triangle%20_down.jpg

版权声明：本文为博主原创文章，未经博主允许不得转载。

目录[(?)[+]](http://blog.csdn.net/oscar999/article/details/37499743)

### 前言

在 HTML 文档中 <input type="file"> 标签每出现一次，一个 FileUpload 对象就会被创建。

该元素包含一个文本输入字段，用来输入文件名，还有一个按钮，用来打开文件选择对话框以便图形化选择文件。

该元素的 value 属性保存了用户指定的文件的名称，但是当包含一个 file-upload 元素的表单被提交的时候，浏览器会向服务器发送选中的文件的内容而不仅仅是发送文件名。

当用户选择或编辑一个文件名，file-upload 元素触发 onchange 事件句柄。

看个简单例子：

**[html]** [view plain](http://blog.csdn.net/oscar999/article/details/37499743) [copy](http://blog.csdn.net/oscar999/article/details/37499743)

[在CODE上查看代码片](https://code.csdn.net/snippets/422893)

1. <!-- oscar999  -->
2. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"**>**
3. **<html>**
4. **<head>**
5. **<meta** http-equiv="content-type" content="text/html; charset=utf-8"**>**
6. **<meta** name="author" content="oscar999"**>**
7. **<title></title>**
8. **<script>**
9. function  handleFiles(files)
10. {
11. if(files.length)
12. {
13. var file = files[0];
14. var reader = new FileReader();
15. reader.onload = function()
16. {
17. document.getElementById("filecontent").innerHTML = this.result;
18. };
19. reader.readAsText(file);
20. }
21. }
22. **</script>**
24. **</head>**
25. **<body>**
27. **<input** type="file" id="file" onchange="handleFiles(this.files)"**/>**
28. **<div** id="filecontent"**></div>**
29. **</body>**
30. **</html>**

这里读取一个文件， 显示在div 中。

(在IE8 中 无效， this.files 无法读取文件。这个属于html5 的特性)

当选择了一个文件时，就会把包含这个文件的列表（一个FileList对象）作为参数传给handleFiles()函数了。  
这个FileList对象类似一个数组，可以知道文件的数目，而它的元素就是File对象了。  
从这个File对象可以获取name、size、lastModifiedDate和type等属性。  
把这个File对象传给FileReader对象的读取方法，就能读取文件了。

### HTML5 Drag and Drop File

HTML5 支持的File 的操作不仅仅是文件的选择，

在HTML5 之前需要使用 Applet 和 SilverLight 才能达到的文件拖拽功能，在HTML5 中也能轻松的实现，

看代码：

**[html]** [view plain](http://blog.csdn.net/oscar999/article/details/37499743) [copy](http://blog.csdn.net/oscar999/article/details/37499743)

[在CODE上查看代码片](https://code.csdn.net/snippets/422893)

1. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"**>**
2. **<html>**
3. **<head>**
4. **<meta** http-equiv="content-type" content="text/html; charset=utf-8"**>**
5. **<meta** name="author" content="oscar999"**>**
6. **<title></title>**
7. **</head>**
8. **<body>**
9. **<div** id="dropbox"**>** Drop Here **</div>**
10. **<div** id="filecontent"**></div>**
11. **<script>**
12. var dropbox = document.getElementById("dropbox");
13. dropbox.addEventListener("dragenter", dragenter, false);
14. dropbox.addEventListener("dragover", dragover, false);
15. dropbox.addEventListener("drop", drop, false);
17. function dragenter(e) {
18. e.stopPropagation();
19. e.preventDefault();
20. }
22. function dragover(e) {
23. e.stopPropagation();
24. e.preventDefault();
25. }
27. function drop(e) {
28. e.stopPropagation();
29. e.preventDefault();
30. var dt = e.dataTransfer;
31. var files = dt.files;
32. if(files.length)
33. {
34. var file = files[0];
35. var reader = new FileReader();
36. reader.onload = function()
37. {
38. document.getElementById("filecontent").innerHTML = this.result;
39. };
40. reader.readAsText(file);
41. }
42. }
44. **</script>**
45. **</body>**
46. **</html>**

这里通过事件对象的 dataTransfer 可以得到文件。

### 读取文件内容

在第一个例子中， 我们使用 FileReader类来读取文件的内容，

在 W3C 草案中，File 对象只包含文件名，文件类型等只读属性;FileReader用于内容读取和监控读取状态。

（在firefox 中， 可以直接使用 var fileBinary = file.getAsBinary();  读取文件的二进制源码）

FileReader提供的方法包括：

1. readAsBinaryString

2. readAsDataURL

3. readAsText

4. abort

.........

以下，举一个 使用 FileReader 将用户选择的图片不通过后台即时显示出来的例子。

**[html]** [view plain](http://blog.csdn.net/oscar999/article/details/37499743) [copy](http://blog.csdn.net/oscar999/article/details/37499743)

[在CODE上查看代码片](https://code.csdn.net/snippets/422893)

1. function handleFiles(files){
2. for (var i = 0; i **<** **files.length**; i++) {
3. var file = files[i];
4. var imageType = /image.\*/;
6. if (!file.type.match(imageType)) {
7. continue;
8. }
10. var img = document.createElement("img");
11. img.classList.add("obj");
12. img.file = file;
13. preview.appendChild(img);
15. var reader = new FileReader();
16. reader.onload = (function(aImg){
17. return function(e){
18. aImg.src = e.target.result;
19. };
20. })(img);
21. reader.readAsDataURL(file);
22. }
23. }

### 同后端的交互

在一般的HTML  中，使用方式是把file input 放在form 中， 以POST 方式把文件传递到后端。

在 HTML5 中， 也可以通过 FileReader 的 readAsBinaryString 方法读取到文件的二进制码，然后通过 XMLHttpRequest 的 sendAsBinary 方法将其发送出去。

**[javascript]** [view plain](http://blog.csdn.net/oscar999/article/details/37499743) [copy](http://blog.csdn.net/oscar999/article/details/37499743)

[在CODE上查看代码片](https://code.csdn.net/snippets/422893)

1. **var** xhr = **new** XMLHttpRequest();
2. xhr.open("POST", "url");
3. xhr.overrideMimeType('text/plain; charset=x-user-defined-binary');
4. <pre code\_snippet\_id="422893" snippet\_file\_name="blog\_20140709\_4\_2106578" **class**="sh\_javascript sh\_sourceCode" name="code">xhr<span **class**="sh\_symbol">.</span><span **class**="sh\_function">sendAsBinary</span><span **class**="sh\_symbol">(</span>binaryString<span **class**="sh\_symbol">);</span></pre><br>

### 参考

* [W3C File API](http://www.w3.org/TR/FileAPI/)
* [MDC File](https://developer.mozilla.org/en/DOM/File)

### 浏览器支持

适用于 Firefox 3.6+ ，Chrome ，部分适用于其他支持 HTML 5 接口的浏览器，完全不适用于 IE8 及以下版本

[**Web 前沿——HTML5 Form Data 对象的使用**](http://www.cnblogs.com/lhb25/p/html5-formdata-tutorials.html)

　　XMLHttpRequest Level 2 添加了一个新的接口——FormData。利用 FormData 对象，我们可以通过 [**JavaScript**](http://www.cnblogs.com/lhb25/category/146074.html) 用一些键值对来模拟一系列表单控件，我们还可以使用 XMLHttpRequest 的 send() 方法来异步的提交表单。与普通的 Ajax 相比，使用 FormData 的最大优点就是我们可以异步上传二进制文件。

您可能感兴趣的相关文章

* [10大流行 Metro UI Bootstrap 主题和模板](http://www.cnblogs.com/lhb25/archive/2013/04/11/10-metro-bootstraps-ui-templates.html)
* [精选12款优秀 jQuery Ajax 分页插件和教程](http://www.cnblogs.com/lhb25/archive/2012/08/08/ajax-jquery-pagination-plugin-tutorial.html)
* [10大流行的 Metro UI 风格 Bootstrap 主题](http://www.cnblogs.com/lhb25/archive/2013/04/11/10-metro-bootstraps-ui-templates.html)
* [8款效果精美的 jQuery 加载动画和进度条插件](http://www.cnblogs.com/lhb25/archive/2012/10/24/jquery-loading-animation-and-progress-bar-plugins.html)
* [推荐35款精致的 CSS3 和 HTML5 网页模板](http://www.cnblogs.com/lhb25/archive/2012/08/03/5-creative-html5-and-css3-templates.html)

**创建一个FormData对象**

　　你可以先创建一个空的 FormData 对象,然后使用 append() 方法向该对象里添加字段，如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | var oMyForm = new FormData();    oMyForm.append("username", "Groucho");  oMyForm.append("accountnum", 123456); // 数字123456被立即转换成字符串"123456"    // fileInputElement中已经包含了用户所选择的文件  oMyForm.append("userfile", fileInputElement.files[0]);    var oFileBody = "<a id="a"><b id="b">hey!</b></a>"; // Blob对象包含的文件内容  var oBlob = new Blob([oFileBody], { type: "text/xml"});    oMyForm.append("webmasterfile", oBlob);    var oReq = new XMLHttpRequest();  oReq.open("POST", "<http://foo.com/submitform.php>");  oReq.send(oMyForm); |

**注：**字段 "userfile" 和 "webmasterfile" 的值都包含了一个文件。通过 [FormData.append()](https://developer.mozilla.org/zh-CN/DOM/XMLHttpRequest/FormData#append()) 方法赋给字段 "accountnum" 的数字被自动转换为字符（字段的值可以是一个 [Blob](https://developer.mozilla.org/zh-CN/docs/DOM/Blob) 对象，[File](https://developer.mozilla.org/zh-CN/docs/DOM/File)对象或者字符串，剩下其他类型的值都会被自动转换成字符串）。

　　在该例子中，我们创建了一个名为 oMyForm 的 FormData 对象，该对象中包含了名为"username"，"accountnum"，"userfile" 以及 "webmasterfile" 的字段名，然后使用XMLHttpRequest的 [send()](https://developer.mozilla.org/zh-CN/DOM/XMLHttpRequest#send()) 方法把这些数据发送了出去。"webmasterfile" 字段的值不是一个字符串,还是一个 [Blob](https://developer.mozilla.org/zh-CN/DOM/Blob) 对象。

**使用HTML表单来初始化一个FormData对象**

　　可以用一个已有的 form 元素来初始化 FormData 对象，只需要把这个 form 元素作为参数传入 FormData 构造函数即可：

|  |  |
| --- | --- |
| 1 | var newFormData = new FormData(someFormElement); |

　　例如：

|  |  |
| --- | --- |
| 1  2  3  4 | var formElement = document.getElementById("myFormElement");  var oReq = new XMLHttpRequest();  oReq.open("POST", "submitform.php");  oReq.send(new FormData(formElement)); |

　　你还可以在已有表单数据的基础上，继续添加新的键值对，如下：

|  |  |
| --- | --- |
| 1  2  3  4 | var formElement = document.getElementById("myFormElement");  formData = new FormData(formElement);  formData.append("serialnumber", serialNumber++);  oReq.send(formData); |

　　你可以通过这种方式添加一些不想让用户编辑的固定字段,然后再发送.

**使用FormData对象发送文件**

　　你还可以使用 FormData 来发送二进制文件.首先在 HTML 中要有一个包含了文件输入框的 form 元素：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <form enctype="multipart/form-data" method="post" name="fileinfo">    <label>Your email address:</label>    <input type="email" autocomplete="on" autofocus name="userid" placeholder="email" required size="32" maxlength="64" /><br />    <label>Custom file label:</label>    <input type="text" name="filelabel" size="12" maxlength="32" /><br />    <label>File to stash:</label>    <input type="file" name="file" required />  </form>  <div id="output"></div>  <a href="javascript:sendForm()">Stash the file!</a> |

　　然后你就可以使用下面的代码来异步的上传用户所选择的文件：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | function sendForm() {    var oOutput = document.getElementById("output");    var oData = new FormData(document.forms.namedItem("fileinfo"));      oData.append("CustomField", "This is some extra data");      var oReq = new XMLHttpRequest();    oReq.open("POST", "stash.php", true);    oReq.onload = function(oEvent) {      if (oReq.status == 200) {        oOutput.innerHTML = "Uploaded!";      } else {        oOutput.innerHTML = "Error " + oReq.status + " occurred uploading your file.<br \/>";      }    };      oReq.send(oData);  } |

　　你还可以不借助 HTML 表单，直接向 [FormData](https://developer.mozilla.org/zh-CN/docs/DOM/XMLHttpRequest/FormData) 对象中添加一个 [File](https://developer.mozilla.org/zh-CN/docs/DOM/File) 对象或者一个 [Blob](https://developer.mozilla.org/zh-CN/docs/DOM/Blob) 对象：

|  |  |
| --- | --- |
| 1 | data.append("myfile", myBlob); |

　　如果 FormData 对象中的某个字段值是一个 [Blob](https://developer.mozilla.org/zh-CN/docs/DOM/Blob) 对象，则在发送 HTTP 请求时，代表该 Blob 对象所包含文件的文件名的 "Content-Disposition" 请求头的值在不同的浏览器下有所不同，Firefox使用了固定的字符串"blob"，而 Chrome 使用了一个随机字符串。

　　你还可以使用 jQuery 来发送 FormData，但必须要正确的设置相关选项：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | var fd = new FormData(document.getElementById("fileinfo"));  fd.append("CustomField", "This is some extra data");  $.ajax({    url: "stash.php",    type: "POST",    data: fd,    processData: false,  // 告诉jQuery不要去处理发送的数据    contentType: false   // 告诉jQuery不要去设置Content-Type请求头  }); |

**浏览器兼容性**

　　桌面端：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Feature** | **Chrome** | **Firefox (Gecko)** | **Internet Explorer** | **Opera** | **Safari** |
| Basic support | 7+ | 4.0 (2.0) | 10+ | 12+ | 5+ |
| 支持filename参数 | (Yes) | 22.0 (22.0) | ? | ? | ? |

　　移动端：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Feature** | **Android** | **Chrome for Android** | **Firefox Mobile (Gecko)** | **IE Mobile** | **Opera Mobile** | **Safari Mobile** |
| Basic support | 3.0 | ? | 4.0 (2.0) | ? | 12+ | ? |
| 支持filename参数 | ? | ? | 22.0 (22.0) | ? | ? | ? |

**参考文献：**

* [MDN：使用 XMLHttpRequest](https://developer.mozilla.org/zh-CN/docs/DOM/XMLHTTPRequest)
* [MDN：XMLHttpRequest FormData](https://developer.mozilla.org/zh-CN/docs/DOM/XMLHttpRequest/FormData)
* [XMLHttpRequest 2 新技巧](http://www.html5rocks.com/zh/tutorials/file/xhr2/)
* [MDN：使用 FormData 对象](https://developer.mozilla.org/zh-CN/docs/DOM/XMLHttpRequest/FormData/Using_FormData_Objects)
* [W3C：XMLHttpRequest Level 2](http://www.w3.org/TR/XMLHttpRequest2/#interface-formdata)

## Html5中允许使用FormData表单提交数据

首先来认识一下FormData表单

var formData = new FormData();

formData.append('name', 'zhangsan');

formData.append('age', 20);

formData.append('gender', 'M');

//防止跨域，注意，该字符串由服务端生成后一部分发送到session中，一部分放在form隐藏域中

formData.append('csrf', 'yty98db128swdx827dcd3jdxh');

或者使用另一种方式

<form id='registerform' name='user/register' action='user/register'>

<input type='text' name='username' value='张三'>

<input type='email' name='email' value='zhangsan@example.com'>

<input type='number' name='birthDate' value='1940'>

<input type='submit' onclick='sendForm(this.form); return false;'>

</form>

var formData = new FormData(document.forms['registerform']);

formData.append('csrf', 'yty98db128swdx827dcd3jdxh');

然后构建Ajax

**function** **sendForm**(formData) {

**var** xhr = **new** XMLHttpRequest();

xhr.open('POST', form.action, true);

xhr.onload = **function**(e) { *// ... };*

*// 指定通信过程中状态改变时的回调函数*

xhr.onreadystatechange = **function**()

{

*// 通信成功时，状态值为4*

**var** completed = 4;

**if**(xhr.readyState === completed)

{

**if**(xhr.status === 200)

{

*// 处理服务器发送过来的数据*

}**else**{ *// 处理错误*

}

}

};

xhr.send(formData);

}

使用FormData进行文件上传

**function** **uploadFiles**(url, files) {

**var** formData = **new** FormData();

**for** (**var** i = 0, file; file = files[i]; ++i)

{

formData.append(file.name, file);

}

**var** xhr = **new** XMLHttpRequest();

xhr.open('POST', url, true);

xhr.onload = **function**(e) { ... };

xhr.send(formData); *// multipart/form-data }*

document.querySelector('input[type="file"]').addEventListener('change', **function**(e) {

uploadFiles('/server', **this**.files); },

false);

**function** **upload**(blobOrFile)

{

**var** xhr = **new** XMLHttpRequest();

xhr.open('POST', '/server', true);

xhr.onload = **function**(e) { ... }; *// Listen to the upload progress.*

**var** progressBar = document.querySelector('progress');

xhr.upload.onprogress = **function**(e) {

**if** (e.lengthComputable) {

progressBar.value = (e.loaded / e.total) \* 100;

progressBar.textContent = progressBar.value; *// Fallback for unsupported browsers.*

}

};

xhr.send(blobOrFile);

}

upload(**new** Blob(['hello world'], {type: 'text/plain'}));

# 使用FormData对象

利用[FormData](https://developer.mozilla.org/zh-CN/DOM/XMLHttpRequest/FormData)对象,你可以使用一系列的键值对来模拟一个完整的表单,然后使用XMLHttpRequest发送这个"表单".

## 创建一个FormData对象

你可以先创建一个空的FormData对象,然后使用[append()](https://developer.mozilla.org/zh-cn/DOM/XMLHttpRequest/FormData#append%28%29)方法向该对象里添加字段,如下:

var oMyForm = new FormData();

oMyForm.append("username", "Groucho");

oMyForm.append("accountnum", 123456); // 数字123456被立即转换成字符串"123456"

// fileInputElement中已经包含了用户所选择的文件

oMyForm.append("userfile", fileInputElement.files[0]);

var oFileBody = "<a id="a"><b id="b">hey!</b></a>"; // Blob对象包含的文件内容

var oBlob = new Blob([oFileBody], { type: "text/xml"});

oMyForm.append("webmasterfile", oBlob);

var oReq = new XMLHttpRequest();

oReq.open("POST", "http://foo.com/submitform.php");

oReq.send(oMyForm);

**注:** 字段"userfile"和"webmasterfile"的值都包含了一个文件.通过 [FormData.append()](https://developer.mozilla.org/zh-CN/DOM/XMLHttpRequest/FormData#append())方法赋给字段"accountnum"的数字被自动转换为字符(字段的值可以是一个[Blob](https://developer.mozilla.org/zh-CN/docs/DOM/Blob)对象,一个[File](https://developer.mozilla.org/zh-CN/docs/DOM/File)对象,或者一个字符串,剩下其他类型的值都会被自动转换成字符串).

在该例子中,我们创建了一个名为oMyForm的FormData对象,该对象中包含了名为"username", "accountnum", "userfile" 以及 "webmasterfile"的字段名,然后使用XMLHttpRequest的[send()](https://developer.mozilla.org/zh-CN/DOM/XMLHttpRequest#send())方法把这些数据发送了出去."webmasterfile"字段的值不是一个字符串,还是一个[Blob](https://developer.mozilla.org/zh-CN/DOM/Blob)对象.

## 使用HTML表单来初始化一个FormData对象

可以用一个已有的[<form>](https://developer.mozilla.org/zh-CN/docs/HTML/Element/form)元素来初始化FormData对象,只需要把这个form元素作为参数传入FormData构造函数即可:

var newFormData = new FormData(someFormElement);

例如:

var formElement = document.getElementById("myFormElement");

var oReq = new XMLHttpRequest();

oReq.open("POST", "submitform.php");

oReq.send(new FormData(formElement));

你还可以在已有表单数据的基础上,继续添加新的键值对,如下:

var formElement = document.getElementById("myFormElement");

formData = new FormData(formElement);

formData.append("serialnumber", serialNumber++);

oReq.send(formData);

你可以通过这种方式添加一些不想让用户编辑的固定字段,然后再发送.

## 使用FormData对象发送文件

你还可以使用FormData来发送二进制文件.首先在HTML中要有一个包含了文件输入框的form元素:

<form enctype="multipart/form-data" method="post" name="fileinfo">

<label>Your email address:</label>

<input type="email" autocomplete="on" autofocus name="userid" placeholder="email" required size="32" maxlength="64" /><br />

<label>Custom file label:</label>

<input type="text" name="filelabel" size="12" maxlength="32" /><br />

<label>File to stash:</label>

<input type="file" name="file" required />

</form>

<div id="output"></div>

<a href="javascript:sendForm()">Stash the file!</a>

然后你就可以使用下面的代码来异步的上传用户所选择的文件:

function sendForm() {

var oOutput = document.getElementById("output");

var oData = new FormData(document.forms.namedItem("fileinfo"));

oData.append("CustomField", "This is some extra data");

var oReq = new XMLHttpRequest();

oReq.open("POST", "stash.php", true);

oReq.onload = function(oEvent) {

if (oReq.status == 200) {

oOutput.innerHTML = "Uploaded!";

} else {

oOutput.innerHTML = "Error " + oReq.status + " occurred uploading your file.<br \/>";

}

};

oReq.send(oData);

}

你还可以不借助HTML表单,直接向[FormData](https://developer.mozilla.org/zh-CN/docs/DOM/XMLHttpRequest/FormData)对象中添加一个[File](https://developer.mozilla.org/zh-CN/docs/DOM/File)对象或者一个[Blob](https://developer.mozilla.org/zh-CN/docs/DOM/Blob)对象:

data.append("myfile", myBlob);

如果FormData对象中的某个字段值是一个[Blob](https://developer.mozilla.org/zh-CN/docs/DOM/Blob)对象,则在发送http请求时,代表该Blob对象所包含文件的文件名的"Content-Disposition"请求头的值在不同的浏览器下有所不同,Firefox使用了固定的字符串"blob,"而Chrome使用了一个随机字符串.

你还可以使用[**jQuery**](http://lib.csdn.net/base/22)来发送FormData,但必须要正确的设置相关选项:

var fd = new FormData(document.getElementById("fileinfo"));

fd.append("CustomField", "This is some extra data");

$.ajax({

url: "stash.php",

type: "POST",

data: fd,

processData: false, // 告诉jQuery不要去处理发送的数据

contentType: false // 告诉jQuery不要去设置Content-Type请求头

});

# XHR——XMLHttpRequest对象

## 创建XMLHttpRequest对象

与之前众多DOM操作一样,创建XHR对象也具有兼容性问题:IE6及之前的版本使用ActiveXObject,IE7之后及其它浏览器使用XMLHttpRequest

不但IE6及其之前的版本将XHR作为一个ActiveXObject运行,而且还存在众多版本:一开始是Microsoft.XMLHTTP 之后变成Msxml2.XMLHTTP及更新版的Msxml3.XMLHTTP

function XHR() {

var xhr;

try {xhr = new XMLHttpRequest();}

catch(e) {

var IEXHRVers =["Msxml3.XMLHTTP","Msxml2.XMLHTTP","Microsoft.XMLHTTP"];

for (var i=0,len=IEXHRVers.length;i< len;i++) {

try {xhr = new ActiveXObject(IEXHRVers[i]);}

catch(e) {continue;}

}

}

return xhr;

}

## 使用XHR

var xhr = XHR();

//open方法 创建一个新的http请求，并指定此请求的方法、URL以及验证信息(用户名/密码)

xhr.open("get","test.txt",true);

/\*第一个参数是请求方式,一般用get与post方法,与form标签的method类似

第二个参数是请求的URL

第三个参数是请求是同步进行还是异步进行,true表示异步

调用了open方法仅仅是传递了参数而已\*/

xhr.send(null);//调用了send方法后才会发出请求

//并且get方式发送请求时send参数是null

在服务器环境中执行上面的脚本,并且给一个php或asp脚本发送请求,会发现服务器端脚本其实会被执行

//PHP脚本

$fp =fopen("a.txt","wb");

fwrite($fp,"PHP文件在后台执行了");

fclose($fp);

echo "返回内容!";

上面PHP脚本的返回内容不会直接在页面上显示出来,必需要用JS通过XHR对象接收

var xhr = XHR();

xhr.open("get","test.php",true);

xhr.onreadystatechange = callback;//在readystatechange事件上绑定一个函数

//当接收到数据时,会调用readystatechange事件上的事件处理函数

xhr.send(null);

functon callback() {

//在这里面没有使用this.readyState这是因为IE下面ActiveXObject的特殊性

if (xhr.readyState == 4) {//readyState表示文档加载进度,4表示完毕

alert(xhr.reponseText);//responseText属性用来取得返回的文本

}

}

## XHR对象参考

#### readyState属性 返回当前请求的状态

* 0 (未初始化) 对象已建立，但是尚未初始化（尚未调用open方法）
* 1 (初始化) 对象已建立，尚未调用send方法
* 2 (发送数据) send方法已调用，但是当前的状态及http头未知
* 3 (数据传送中) 已接收部分数据，因为响应及http头不全，这时通过responseBody和responseText获取部分数据会出现错误
* 4 (完成) 数据接收完毕,此时可以通过通过responseBody和responseText获取完整的回应数据

var xhr =XHR();

alert(xhr.readyState);//0

xhr.open("get","test.htm",true);

alert(xhr.readyState);//1

xhr.send(null);

alert(xhr.readyState);//IE下会是4,而FF下会是2

//可以通过readystatechange事件监听

xhr = XHR();

xhr.onreadystatechange = function () {

alert(xhr.readyState);//FF下会依次是1,2,3,4但最后还会再来个1

//IE下则是1,1,3,4

};

xhr.open("get","test.txt",true);

xhr.send(null);

从上面可以看到,对于readyState这个属性,各个浏览器看法也不一样,但其实我们只需要知道当状态为4的时候可以获取response就行了!

#### status 返回当前请求的http状态码

status属性返回当前请求的http状态码,此属性仅当数据发送并接收完毕后才可获取。完整的HTTP状态码如下:

* 100 Continue 初始的请求已经接受，客户应当继续发送请求的其余部分
* 101 Switching Protocols 服务器将遵从客户的请求转换到另外一种协议
* 200 OK 一切正常，对GET和POST请求的应答文档跟在后面。
* 201 Created 服务器已经创建了文档，Location头给出了它的URL。
* 202 Accepted 已经接受请求，但处理尚未完成。
* 203 Non-Authoritative Information 文档已经正常地返回，但一些应答头可能不正确，因为使用的是文档的拷贝
* 204 No Content 没有新文档，浏览器应该继续显示原来的文档。如果用户定期地刷新页面，而Servlet可以确定用户文档足够新，这个状态代码是很有用的
* 205 Reset Content 没有新的内容，但浏览器应该重置它所显示的内容。用来强制浏览器清除表单输入内容
* 206 Partial Content 客户发送了一个带有Range头的GET请求，服务器完成了它
* 300 Multiple Choices 客户请求的文档可以在多个位置找到，这些位置已经在返回的文档内列出。如果服务器要提出优先选择，则应该在Location应答头指明。
* 301 Moved Permanently 客户请求的文档在其他地方，新的URL在Location头中给出，浏览器应该自动地访问新的URL。
* 302 Found 类似于301，但新的URL应该被视为临时性的替代，而不是永久性的。
* 303 See Other 类似于301/302，不同之处在于，如果原来的请求是POST，Location头指定的重定向目标文档应该通过GET提取
* 304 Not Modified 客户端有缓冲的文档并发出了一个条件性的请求（一般是提供If-Modified-Since头表示客户只想比指定日期更新的文档）。服务器告诉客户，原来缓冲的文档还可以继续使用。
* 305 Use Proxy 客户请求的文档应该通过Location头所指明的代理服务器提取
* 307 Temporary Redirect 和302（Found）相同。许多浏览器会错误地响应302应答进行重定向，即使原来的请求是POST，即使它实际上只能在POST请求的应答是303时才能重定向。由于这个原因，HTTP 1.1新增了307，以便更加清除地区分几个状态代码：当出现303应答时，浏览器可以跟随重定向的GET和POST请求；如果是307应答，则浏览器只能跟随对GET请求的重定向。
* 400 Bad Request 请求出现语法错误。
* 401 Unauthorized 客户试图未经授权访问受密码保护的页面。应答中会包含一个WWW-Authenticate头，浏览器据此显示用户名字/密码对话框，然后在填写合适的Authorization头后再次发出请求。
* 403 Forbidden 资源不可用。
* 404 Not Found 无法找到指定位置的资源
* 405 Method Not Allowed 请求方法（GET、POST、HEAD、Delete、PUT、TRACE等）对指定的资源不适用。
* 406 Not Acceptable 指定的资源已经找到，但它的MIME类型和客户在Accpet头中所指定的不兼容
* 407 Proxy Authentication Required 类似于401，表示客户必须先经过代理服务器的授权。
* 408 Request Timeout 在服务器许可的等待时间内，客户一直没有发出任何请求。客户可以在以后重复同一请求。
* 409 Conflict 通常和PUT请求有关。由于请求和资源的当前状态相冲突，因此请求不能成功。
* 410 Gone 所请求的文档已经不再可用，而且服务器不知道应该重定向到哪一个地址。它和404的不同在于，返回407表示文档永久地离开了指定的位置，而404表示由于未知的原因文档不可用。
* 411 Length Required 服务器不能处理请求，除非客户发送一个Content-Length头
* 412 Precondition Failed 请求头中指定的一些前提条件失败
* 413 Request Entity Too Large 目标文档的大小超过服务器当前愿意处理的大小。如果服务器认为自己能够稍后再处理该请求，则应该提供一个Retry-After头
* 414 Request URI Too Long URI太长
* 416 Requested Range Not Satisfiable 服务器不能满足客户在请求中指定的Range头
* 500 Internal Server Error 服务器遇到了意料不到的情况，不能完成客户的请求
* 501 Not Implemented 服务器不支持实现请求所需要的功能。例如，客户发出了一个服务器不支持的PUT请求
* 502 Bad Gateway 服务器作为网关或者代理时，为了完成请求访问下一个服务器，但该服务器返回了非法的应答
* 503 Service Unavailable 服务器由于维护或者负载过重未能应答。例如，Servlet可能在数据库连接池已满的情况下返回503。服务器返回503时可以提供一个Retry-After头
* 504 Gateway Timeout 由作为代理或网关的服务器使用，表示不能及时地从远程服务器获得应答
* 505 HTTP Version Not Supported 服务器不支持请求中所指明的HTTP版本

事实上,我们只需要知道状态为200的时候(OK)才读取response就行了!

#### responseText与responseXML

responseText 将响应信息作为字符串返回 . XMLHTTP尝试将响应信息解码为Unicode字符串，XMLHTTP默认将响应数据的编码定为UTF-8，如果服务器返回的数据带BOM(byte-order mark)，XMLHTTP可以解码任何UCS-2 (big or little endian)或者UCS-4 数据。注意，如果服务器返回的是xml文档，此属性并不处理xml文档中的编码声明。你需要使用responseXML来处理。

responseXML 将响应信息格式化为Xml Document对象并返回 . 如果响应数据不是有效的XML文档，此属性本身不返回XMLDOMParseError，可以通过处理过的DOMDocument对象获取错误信息。

#### 其它一些XHR对象的方法

abort 取消当前请求

getAllResponseHeaders 获取响应的所有http头 每个http头名称和值用冒号分割，并以\r\n结束。当send方法完成后才可调用该方法。

getResponseHeader 从响应信息中获取指定的http头 当send方法成功后才可调用该方法。如果服务器返回的文档类型为"text/xml", 则这句话xmlhttp.getResponseHeader("Content-Type");将返回字符串"text/xml"。可以使用getAllResponseHeaders方法获取完整的http头信息。

setRequestHeader 单独指定请求的某个http头 如果已经存在已此名称命名的http头，则覆盖之。此方法必须在open方法后调用。

## 请求方式

GET 请求

//JS

var xhr = XHR();

xhr.open("get","test.php?qs=true&userName=abc&pwd=123456",true);

xhr.onreadystatechange = function () {

if (xhr.readyState==4 && xhr.status ==200) {

alert(xhr.responseText);

}

};

xhr.send(null);

//PHP

print\_r($\_GET);

POST 请求

//JS

var xhr = XHR();

xhr.open("post","test.php",true);

xhr.onreadystatechange = function () {

if (xhr.readyState==4 && xhr.status ==200) {

alert(xhr.responseText);

}

};

//比GET请求多了一步

xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

//另外,数据是通过send方法发送的

xhr.send("qs=true&userName=abc&pwd=123456");

//PHP

header("Cache-Control: no-cache, must-revalidate");//可以让浏览器不缓存结果

print\_r($\_POST);

Email:gaojun\_le@163.com

# [HTML5文件上传还有进度条](http://bbs.9ria.com/thread-219147-1-1.html)

以下是自学it网--中级班上课笔记。 网址：[www.zixue.it](http://www.zixue.it/)   
需要在chrome,ff,IE10下运行  
  
[html](http://www.2cto.com/kf/qianduan/css/)页面

1. [html]
2. <!DOCTYPE html>
3. <html lang="zh-CN">
4. <head>
5. <title>HTML5 文件上传进度条</title>
6. <meta charset="utf-8" />
7. <meta name="description" content="" />
8. <meta name="keywords" content="" />
9. <script type="text/javascript">
10. /\*
11. 想得到传送文件的进度条
12. 1:要能把文件内容打包并发送  FormData对象
13. 2: 发送过程中,要能不断检测 已发送/全部 progress.onprogress
14. \*/
15. </script>
16. <style type="text/css">
17. #progress{
18. border: 1px solid blue;
19. width:500px;
20. height:20px;
21. }
22. #bar {
23. background:green;
24. height:20px;
25. width:0%;
26. }
27. </style>
28. </head>
29. <body>
30. <h1>在chrome,ff,IE10运行</h1>
31. <form>
32. <input type="file" name="pic" /><br />
33. </form>
34. <div id="progress"><div id="bar"></div></div>
35. </body>
36. <script>
37. // 负责ajax发送数据
38. function up(fd) {
39. var xhr = new XMLHttpRequest();
40. xhr.open('POST','upfile.php',true); // 异步传输
41. // xhr.upload 这是html5新增的api,储存了上传过程中的信息
42. xhr.upload.onprogress = function (ev) {
43. var percent = 0;
44. if(ev.lengthComputable) {
45. percent = 100 \* ev.loaded/ev.total;
46. //document.getElementById('progress').innerHTML = percent;
47. document.getElementById('bar').style.width = percent + '%';
48. }
49. }
50. xhr.send(fd);
51. }
52. document.getElementsByTagName('input')[0].onchange = function() {
53. //alert('你选择文件了');
54. //alert(this.files[0]); // 文件对象,html5新增的api
55. var fd = new FormData(); // html5新增的对象,可以包装字符,二进制信息
56. fd.append(this.name,this.files[0]);
57. up(fd);
58. }
59. </script>
60. </html>

复制代码

upfile.php页面

1. [php]
2. echo move\_uploaded\_file($\_FILES['pic']['tmp\_name'],'./upload/' . $\_FILES['pic']['name']) ? 'OK':'fail';

复制代码

原文：<http://www.2cto.com/kf/201306/220934.html>

# [新鲜出炉：基于HTML5的jquery文件上传插件（多文件选择、带进度条、完整可用）](http://piperzero.iteye.com/blog/1811357)

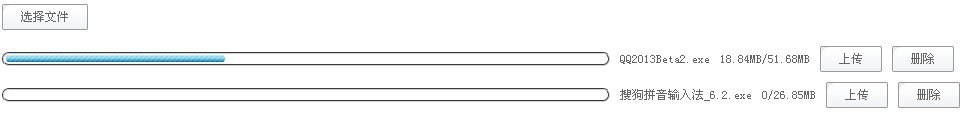
一、前言

首先，为什么要写这个插件呢？分析下现在大家用的比较多的上传插件，大多是flash版本的，事实上我在前段时间一直用的是一个叫做uploadify的插件，官网http://www.uploadify.com.其官网提供了两个版本，flash的，免费，html5的，收费。其flash版本的做的挺不错，很多可以配置的参数，但是用flash的缺点也是很明显的：1.加载时间长；2.不支持flash的设备（ipad）不能使用。关于这个html5版本的，这里必须要鄙视一下，竟然要收费！太不地道了。而这个正是我写这个插件的动力：做一个免费的插件，而且要跟你的功能一模一样！于是，照着官方的api，开始动手了。。。uploadify,我要断你财路了大笑

html5标准对file控件做了很多扩展，包括可以在客户端读取文件信息等，可参考http://www.w3.org/TR/file-upload/。另外一个就是关于xmlHttpRequest对象，也做了新的扩展，可以实时反馈上传文件的进度。可参考http://www.w3.org/TR/XMLHttpRequest/。利用这些新特性，做一个上传插件就轻而易举了。搜一下网上关于html5上传的文章，不少，有些是简单的一些介绍，有些是做出了插件，但是功能不太多。在此必须提一下@张鑫旭（http://www.zhangxinxu.com）的一篇文章，给了我很大帮助，事实上，我就是在他所写的核心文件基础上做的，所以ZXXFILE这个名字也给留着没改。

二、先拉出来遛遛

是骡子是马，先拉出来看看吧。先上截图：



这个骡子不错吧。插件有以下特性：

1.支持多文件上传

2.带进度条

3.可以显示文件名称

4.可以实时显示已上传的大小

5.可以从队列中删除

6.各元素的样式都可以自己控制

此demo的全部代码如下：

**Html代码  [收藏代码](javascript:void())**

1. <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"**>**
2. **<html** xmlns="http://www.w3.org/1999/xhtml"**>**
3. **<head>**
4. **<meta** http-equiv="Content-Type" content="text/html; charset=utf-8" **/>**
5. **<title>**无标题文档**</title>**
6. **<link** rel="stylesheet" type="text/css" href="html5uploader.css"**/>**
7. **<script** type="text/javascript" src="jquery.js"**></script>**
8. **<script** type="text/javascript" src="jquery.html5uploader.js"**></script>**
9. **<style** type="text/css"**>**
10. **</style>**
11. **<script** type="text/javascript"**>**
12. $(function(){
13. $('#upload').html5uploader({
14. auto:false,
15. multi:true,
16. removeTimeout:9999999,
17. url:'upload.php',
18. onUploadStart:function(){
19. alert('开始上传');
20. },
21. onInit:function(){
22. alert('初始化');
23. },
24. onUploadComplete:function(){
25. alert('上传完成');
26. }
27. });
28. });
29. **</script>**
30. **</head>**
32. **<body>**
33. **<div** id="upload"**></div>**
34. **</body>**
35. **</html>**

dom中只需有一个div容器即可。

三、插件的使用

参加demo页面就可以啦！

四、关于配置参数

上面说了，这个插件是照着uploadify的api文档写的，那么，我就省了写文档啦，哈哈。各参数如何配置，参看他官方的文档就可以了。等于我这个是山寨产品，免费使用。

下载页面：http://download.csdn.net/detail/never\_say\_goodbye/5077796

# [localresizeimg](http://www.jincon.com/tag/localresizeimg/)-概述

* 通常压缩图片需要上传到后端，由后端处理。
* 但是如果要上传的图片很大，特别是手机当场拍摄下来的照片（约2M+），那样效率会很低，用户也不会愿意等待。
* 现在能够由前端本地压缩的话，效率将会极大的提升。

这个玩意很强大的地方就在于他可以在HTML5 前端压缩 ，大大加快了上传速度，以及节约流量。

直接上写好的demo了

HTML

|  |  |  |
| --- | --- | --- |
| 01 | <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" | |
| 02 | "<http://www.w3.org/TR/html4/loose.dtd>"> |

|  |  |
| --- | --- |
| 03 | <html> |
| 04 | <head> | |

|  |  |
| --- | --- |
| 05 | <title> New Document </title> |
| 06 | <meta name="Generator" content="EditPlus"> | |

|  |  |
| --- | --- |
| 07 | <meta name="Author" content=""> |
| 08 | <meta name="Keywords" content=""> | |

|  |  |
| --- | --- |
| 09 | <meta name="Description" content=""> |
| 10 | <script src="lrz.mobile.min.js"></script> | |

|  |  |  |
| --- | --- | --- |
| 11 | </head> | |
| 12 |  |

|  |  |  |
| --- | --- | --- |
| 13 | <body> | |
| 14 |  |

|  |  |  |
| --- | --- | --- |
| 15 | <input type="file" capture="camera" /> | |
| 16 | <script> |

|  |  |  |
| --- | --- | --- |
| 17 | var input = document.querySelector('input'); | |
| 18 | input.onchange = function () { |

|  |  |
| --- | --- |
| 19 | lrz(this.files[0], {width: 100}, function (results) { |
| 20 | // 你需要的数据都在这里，可以以字符串的形式传送base64给服务端转存为图片。 |

|  |  |  |
| --- | --- | --- |
| 21 | console.log(results); | |
| 22 |  |

|  |  |  |
| --- | --- | --- |
| 23 | // 发送到后端 | |
| 24 | var xhr = new XMLHttpRequest(); |

|  |  |
| --- | --- |
| 25 | var data = { |
| 26 | base64: results.base64, | |

|  |  |  |
| --- | --- | --- |
| 27 | size: results.base64.length // 校验用，防止未完整接收 | |
| 28 | }; |

|  |  |
| --- | --- |
| 29 | xhr.open('POST', '1.php'); |
| 30 | xhr.setRequestHeader('Content-Type', 'application/json; charset=utf-8'); | |

|  |  |
| --- | --- |
| 31 | xhr.onreadystatechange = function () { |
| 32 | if (xhr.readyState === 4 && xhr.status === 200) { |

|  |  |  |
| --- | --- | --- |
| 33 | var result = JSON.parse(xhr.response); | |
| 34 | result.error |

|  |  |
| --- | --- |
| 35 | ? alert('服务端错误，未能保存图片') |
| 36 | //: demo\_report('服务端实存的图片', result.src, result.size); |

|  |  |  |
| --- | --- | --- |
| 37 | : alert('上传OK'); | |
| 38 | } |

|  |  |
| --- | --- |
| 39 | }; |
| 40 | xhr.send(JSON.stringify(data)); // 发送base64 | |

|  |  |  |
| --- | --- | --- |
| 41 | }); | |
| 42 | } |

|  |  |  |
| --- | --- | --- |
| 43 | </script> | |
| 44 | </body> |

|  |  |
| --- | --- |
| 45 | </html> |

重要的是很多的人不知道这个工具是后端该如何处理，所以贴出php的代码：

|  |  |
| --- | --- |
| 01 | <?php |
| 02 | $base64=file\_get\_contents("<php://input>"); //获取输入流 | |

|  |  |  |
| --- | --- | --- |
| 03 | $base64=json\_decode($base64,1); | |
| 04 | $data = $base64['base64']; |

|  |  |  |
| --- | --- | --- |
| 05 | preg\_match("/data:image\/(.\*);base64,/",$data,$res); | |
| 06 | $ext = $res[1]; |

|  |  |  |
| --- | --- | --- |
| 07 | if(!in\_array($ext,array("jpg","jpeg","png","gif"))){ | |
| 08 | echo json\_encode(array("error"=>1));die; |

|  |  |
| --- | --- |
| 09 | } |
| 10 | $file=time().'.'.$ext; | |

|  |  |
| --- | --- |
| 11 | $data = preg\_replace("/data:image\/(.\*);base64,/","",$data); |
| 12 | if (file\_put\_contents($file,base64\_decode($data))===false) { |

|  |  |  |
| --- | --- | --- |
| 13 | echo json\_encode(array("error"=>1)); | |
| 14 | }else{ |

|  |  |  |
| --- | --- | --- |
| 15 | echo json\_encode(array("error"=>0)); | |
| 16 | } |

下载地址：

https://github.com/think2011/localResizeIMG3/releases