# javascript和HTML5上传图片之前实现预览效果

## 一：FileList对象与file对象

     FileList对象表示用户选择的文件列表，在HTML4中，file控件内只允许放置一个文件，但是到了HTML5中，通过添加multiple属性，file控件内允许一次放置多个文件，控件内的每一个用户选择的文件都是一个file对象，而FileList对象是file对象的列表；

比如如下代码：

选择文件<input type="file" id="file" multiple size="80"/>

<input type="button" onclick="ShowFileName();" value="文件上传"/>

我们可以按住ctrl键选择多个文件；

JS代码如下：

function ShowFileName() {

var files = document.getElementById("file").files;

for(var i = 0; i < files.length; i+=1) {

var file = files[i];

console.log(file);

console.log(file.name);

}

}

我们先打印下file对象有哪些属性，在safari截图如下：

如上图可以看到有文件名(name)属性，文件大小(size)属性，最后修改时间，还有type属性等；

## 二：Blob对象

在HTML5中，新增一个Blob对象，代表原始二进制数据，我们上面提到的file对象也继承了这个Blob对象；

Blob对象有2个属性，size属性表示一个Blob对象的字节长度，type属性表示Blob的MIME类型，如果是未知类型，则返回一个空字符串。

如下代码：

选择文件<input type="file" id="file" multiple size="80" accept="image/\*"/>

<input type="button" onclick="ShowFileType();" value="显示文件信息"/>

文件字节长度:<span id="size"></span><br/>

文件类型:<span id="type"></span>

JS代码如下：

function ShowFileType() {

var file = document.getElementById("file").files[0];

var size = document.getElementById("size");

var type = document.getElementById("type");

// 显示文件字节长度

size.innerHTML = file.size;

// 显示文件类型

type.innerHTML = file.type;

console.log(file);

}

我们打印出file，如下图所示：

## 三：FileReader对象

    FileReader对象有5种方法，其中四种用于读取文件，另一种用来读取过程中断，需要注意的是：无论读取成功与失败，方法并不会返回读取结果，而是将结果保存在result属性中。此对象也是异步的。

FileReader对象的方法如下：

    readAsBinaryString(file): 这个方法将blob对象或文件中的数据读取为二进制字符串，通常我们将它传送到服务器端，服务器端可以通过这段字符串存储文件。

    readAsText(file,encoding): 以纯文本形式读取文件，将读取到的文本保存在result属性中，第二个参数用于指定编码类型，可选的。

    readAsDataURL(file): 读取文件并将文件以数据URL的形式保存在result属性中。

    readAsArrayBuffer(file): 读取文件并将一个包含文件内容的ArrayBuffer保存在result属性中。

FileReader对象的事件如下：

    onabort:  数据读取中断时触发

    onerror: 数据读取出错时触发

    onloadstart: 数据读取开始时触发

    onprogress: 数据读取中

    onload： 数据读取成功完成时触发

    onloadend: 数据读取完成时触发，无论成功或失败。

我们可以先看看demo，如何使用FileReader对象中前面三个方法；注意：fileReader对象读取的数据都保存在result中，如下代码：

<p>

<label>请选择一个文件：</label>

<input type="file" id="file" />

<input type="button" value="读取图像" onclick = "readAsDataURL()"/>

<input type="button" value="读取二进制数据" onclick = "readAsBinaryString()"/>

<input type="button" value="读取文本文件" onclick = "readAsText()"/>

</p>

<div id="result"></div>

JS代码如下：

<script>

var result = document.getElementById("result");

if(typeof FileReader == 'undefined') {

result.innerHTML = "抱歉，你的[浏览器](http://www.2cto.com/os/liulanqi/)不支持FileReader";

}

// 将文件以Data URL形式进行读入页面

function readAsDataURL(){

// 检查是否为图像类型

var simpleFile = document.getElementById("file").files[0];

if(!/image\/\w+/.test(simpleFile.type)) {

alert("请确保文件类型为图像类型");

return false;

}

var reader = new FileReader();

// 将文件以Data URL形式进行读入页面

reader.readAsDataURL(simpleFile);

reader.onload = function(e){

console.log(this.result);

result.innerHTML = '<img src="'+this.result+'" alt=""/>';

}

}

// 将文件以二进制形式读入页面

function readAsBinaryString(){

// 检查是否为图像类型

var simpleFile = document.getElementById("file").files[0];

if(!/image\/\w+/.test(simpleFile.type)) {

alert("请确保文件类型为图像类型");

return false;

}

var reader = new FileReader();

// 将文件以二进制形式进行读入页面

reader.readAsBinaryString(simpleFile);

reader.onload = function(e){

// 在页面上显示二进制数据

result.innerHTML = this.result;

}

}

// 将文件以文本形式读入页面中

function readAsText(){

var simpleFile = document.getElementById("file").files[0];

var reader = new FileReader();

// 将文件以文本形式读入页面中

reader.readAsText(simpleFile);

reader.onload = function(e){

result.innerHTML = this.result;

}

}

</script>

如上代码，如果需要演示的话，可以复制代码到文本编辑器去，预览下即可，这里就不截图了；但是如上代码，在safari浏览器就不支持了~   截图如下：

## 四：btoa方法与atob方法

在HTML5中，btoa方法与atob方法来支持base64编码，b可以被理解为一串二进制数据，a可以被理解为一个ASCLL码字符串，btoa的使用方法如下：

    var result = window.btoa(data);

该方法用于将一串字符串进行base64编码处理，该方法使用一个参数，参数值由一串二进制数据组成的Unicode字符串，该方法返回编码后的base64格式的字符串。

atob方法使用如下所示：

    var result = window.atob(data);

该方法用于将一串经过base64编码后的base64格式的字符串进行解码处理，该方法使用一个参数，参数值为一串经过base64编码后的字符串，方法返回经过解码后的一串由二进制数据组成的Unicode字符串；

浏览器支持：firefox，chrome，opera10.5+及IE10+

什么时候使用btoa方法呢？

当服务器端[数据库](http://www.2cto.com/database/)中直接保存了是图片的二进制数据及图片文件的格式时，当我们需要根据此二进制数据来渲染图片的时候非常有用；如下代码：

<p>

<label>请选择一个文件：</label>

<input type="file" id="file"/>

<input type="button" value="读取图像" onclick = "readPicture()" id="btnReadPicture"/>

</p>

<div id="result"></div>

JS代码如下：

if(typeof FileReader == 'undefined') {

result.innerHTML = "抱歉，你的浏览器不支持FileReader";

}

function readPicture(){

// 检查是否为图像类型

var simpleFile = document.getElementById("file").files[0];

if(!/image\/\w+/.test(simpleFile.type)) {

alert("请确保文件类型为图像类型");

return false;

}

var reader = new FileReader();

// 将文件以二进制文件读入页面中

reader.readAsBinaryString(simpleFile);

reader.onload = function(f){

var result = document.getElementById("result");

var src = "data:" + simpleFile.type + ";base64," + window.btoa(this.result);

result.innerHTML = '<img src ="'+src+'"/>';

}

}

当我们选择一张图像的时候，点击读取图像按钮即可生成一张图片，虽然此方法我们还可以使用我们上面介绍的readAsDataURL的方法也可以直接读取图像，但是当服务器中直接是保存的是二进制文件的话，我们可以直接使用btoa此方法生成图片即可；

下面我们可以看看atob的demo如下：

我们可以使用canvas来绘制一张图片后，点击上传图片按钮后，首先通过canvas元素的toDataURL（）方法获取该图片的url地址，最后获取该URL地址中的base64格式的字符串，最后使用atob方法将其解码为一串二进制数据，并将该二进制数据提交到服务器端；如下代码：

<input type="button" value="上传图片" onclick="imgSave()"/><br/>

<canvas id="canvas" width="400" height="300"></canvas>

JS代码如下：

<script>

var canvas;

function draw(id) {

canvas = document.getElementById(id);

var context = canvas.getContext('2d');

context.fillStyle = 'rgb(0,0,255)';

context.fillRect(0,0,canvas.width,canvas.height);

context.fillStyle = 'rgb(255,255,0)';

context.fillRect(10,20,50,50);

}

draw('canvas');

function imgSave(){

var data = canvas.toDataURL("image/png");

data = data.replace("data:image/png;base64,","");

var xhr = new XMLHttpRequest();

xhr.open("POST","uploadImg.php");

xhr.send(window.atob(data));

}

</script>

uploadImg.php自己写；

下面我们来做一个本地图片上传前预览效果，但是safari不支持，所以在做移动端的同学可能不好了，特别是在IOS6下就不支持哦，应该还有其他办法可以解决的，我们可以看看百度的插件，网址如下：

今天为了学习FileReader，我们可以再来学习下；之前的本地上传图片预览的话，不支持IE10+, 且把页面放在服务器端在IE下会有bug，如下所示：

如上所示，在IE下 这个方法  document.selection.createRange() 不支持，因此为了修复这个bug和在IE10+以上的话，今天又特意研究了下, 在file控件下获取焦点情况下 document.selection.createRange() 将会拒绝访问，所以我们要失去下焦点。我们可以再加一句代码就可以支持了 file.blur();

IE10+的bug的话，我们可以使用[html](http://www.2cto.com/kf/qianduan/css/)5的上面介绍的文件API来解决，先判断是不是IE，如果是的话，且小于10的话，使用滤镜的方法解决图片上传的问题，如果是其他浏览器的话，就使用html5的文件上传方法；但是safari没有处理好，原因是safari不支持html5的fileReader的API，所以如果在移动端开发的图片，如果需要兼容[IOS](http://www.2cto.com/kf/yidong/iphone/)6和6+的话，请注意一下，并不支持~ 当然如果大家有更好的方案来支持IOS6的话，请留言，可以一起总结，一起学习~

## 五：上传图片之前预览效果demo(Safari不兼容)

 1. 在标准浏览器下（IE10+）使用HTML5中文件API即可解决上传图片前预览效果；在这里有2中方案可以解决，第一种是使用文件API，如下代码：

function html5Reader(file) {

var file = file.files[0];

var reader = new FileReader();

reader.readAsDataURL(file);

reader.onload = function(e){

var pic = document.getElementById("img");

pic.src=this.result;

}

}

2. 第二种方案是使用 URL.createObjectURL(fileObj) 这个方法和canvas技术,  我们先来看看 支持这个 URL.createObjectURL(fileObj) 的浏览器有哪些，我们可以点击下面的连接查看：

 实现代码如下：

function getBase64Image(img) {

var canvas = document.createElement("canvas");

canvas.width = img.width;

canvas.height = img.height;

var ctx = canvas.getContext("2d");

ctx.drawImage(img, 0, 0, img.width, img.height);

var ext = img.src.substring(img.src.lastIndexOf(".")+1).toLowerCase();

var dataURL = canvas.toDataURL("image/"+ext);

return dataURL;

}

HTML代码如下：

<input type="file" id="logo" name="logo" accept="image/\*">

<img src= '' id="img"/>

<div id="btn" style="margin-top:50px;font-size:40px;">btn</div>

<canvas id="myCanvas"></canvas>

所有的JS代码如下：

<script>

var EventUtil = {

addHandler: function(element,type,handler) {

if(element.addEventListener) {

element.addEventListener(type,handler,false);

}else if(element.attachEvent) {

element.attachEvent("on"+type,handler);

}else {

element["on" +type] = handler;

}

}

};

var btn = document.getElementById("btn");

var pic = document.getElementById("img");

function getBase64Image(img) {

var canvas = document.createElement("canvas");

canvas.width = img.width;

canvas.height = img.height;

var ctx = canvas.getContext("2d");

ctx.drawImage(img, 0, 0, img.width, img.height);

var ext = img.src.substring(img.src.lastIndexOf(".")+1).toLowerCase();

var dataURL = canvas.toDataURL("image/"+ext);

return dataURL;

}

var ua = navigator.userAgent.toLowerCase();

EventUtil.addHandler(btn,'click',function(){

var file = document.getElementById("logo");

var ext=file.value.substring(file.value.lastIndexOf(".")+1).toLowerCase();

// gif在IE浏览器暂时无法显示

if(ext!='png'&&ext!='jpg'&&ext!='jpeg'){

alert("图片的格式必须为png或者jpg或者jpeg格式！");

return;

}

if(/msie ([^;]+)/.test(ua)) {

var lowIE10 = RegExp["$1"]\*1;

if(lowIE10 == 6){

// IE6浏览器设置img的src为本地路径可以直接显示图片

file.select();

// 在file控件下获取焦点情况下 document.selection.createRange() 将会拒绝访问，所以我们要失去下焦点。

file.blur();

var reallocalpath = document.selection.createRange().text;

pic.src = reallocalpath;

}else if(lowIE10 > 6 && lowIE10 < 10){

// IE7~9 IE10+按照html5的标准去处理

file.select();

// 在file控件下获取焦点情况下 document.selection.createRange() 将会拒绝访问，所以我们要失去下焦点。

file.blur();

var reallocalpath = document.selection.createRange().text;

// 非IE6版本的IE由于安全问题直接设置img的src无法显示本地图片，但是可以通过滤镜来实现

pic.style.filter = "progid:DXImageTransform.Microsoft.AlphaImageLoader(sizingMethod='image',src=\"" + reallocalpath + "\")";

// 设置img的src为base64编码的透明图片 取消显示浏览器默认图片

pic.src = '';

}else if(lowIE10 >= 10) {

html5Reader(file);

}

}else {

html5Reader(file);

}

});

function html5Reader(file) {

var fileObj = file.files[0],

img = document.getElementById("img");

// URL.createObjectURL safari不支持

img.src = URL.createObjectURL(fileObj);

img.onload =function() {

var data = getBase64Image(img);

console.log(data); // 打印出base64编码

}

/\*

var file = file.files[0];

var reader = new FileReader();

reader.readAsDataURL(file);

reader.onload = function(e){

var pic = document.getElementById("img");

pic.src=this.result;

}\*/

}

</script>

# 预览 input file选中的图片，在img里面赋值src实现预览

代码如下利用html5实现:几乎兼容所有主流浏览器，当然IE必须是IE 6以上

【jquery代码】

$(function() {

$("#file\_upload").change(function() {

var $file = $(this);

var fileObj = $file[0];

var windowURL = window.URL || window.webkitURL;

var dataURL;

var $img = $("#preview");

if(fileObj && fileObj.files && fileObj.files[0]){

dataURL = windowURL.createObjectURL(fileObj.files[0]);

$img.attr('src',dataURL);

}else{

dataURL = $file.val();

var imgObj = document.getElementById("preview");

// 两个坑:

// 1、在设置filter属性时，元素必须已经存在在DOM树中，动态创建的Node，也需要在设置属性前加入到DOM中，先设置属性在加入，无效；

// 2、src属性需要像下面的方式添加，上面的两种方式添加，无效；

imgObj.style.filter = "progid:DXImageTransform.Microsoft.AlphaImageLoader(sizingMethod=scale)";

imgObj.filters.item("DXImageTransform.Microsoft.AlphaImageLoader").src = dataURL;

}

});

});

【html代码】：

<input id="file\_upload" type="file" />

<div class="image\_container">

<img id="preview" width="60" height="60">

</div>