我们能够如此简单的实现Ajax的文件上传要得益于两个新的对象：File和FormData对象，还有就是w3c标准化XMLHttpRequest之后的扩展。现在新的浏览器都支持这些东西。

**先来看一下File对象。**

取得File对象有两个途径：

1、用户在一个<input>元素上选择文件后返回的FileList对象。

2、来自由拖放操作生成的 DataTransfer对象。

这里主要讲的是第一个，第二个有兴趣的同学可以自行查找相关资料。

假如有这样一个inpu元素：

|  |  |
| --- | --- |
| 1 | <input id="file" type="file" name="file"> |

可以像下面那样获取到File对象：

|  |  |
| --- | --- |
| 1  2 | var input = document.getElementById("file");  console.log(input.files[0]); |

input.files是就是上面所说的FileList对象，这是一个数组对象，每一个元素都是一个File对象。

File对象的主要属性如下：

name：当前File对象所引用文件的文件名。

size：文件大小，单位是字节。

type：文件MIME类型。

还有其他一些信息，可以通过浏览器的调试命令行查看。

**接下来就是FormData对象了。**

通过这个对象可以模拟一个完整的表单。

获取FormData对象：

|  |  |
| --- | --- |
| 1 | var form = new FormData(); |

这对象就是一个表单，它可以被XMLHttpRequest直接发送。

它有一个方法append，作用就是给FormData对象添加字段。

比如：

|  |  |
| --- | --- |
| 1 | form.append("user", "hehe"); |

相当于表单里面的一个name为user，value为hehe的input元素。

它的值还可以是一个File对象：

|  |  |
| --- | --- |
| 1 | form.append("file", input.files[0]); |

相当于type为file的input元素。

上面那样是从零开始构造，也可以从一个现存的表单来构造FormData对象。

比如有一个form元素ID为myForm:

|  |  |
| --- | --- |
| 1  2 | var myForm= document.getElementById("myForm");  var form = new FormData(myForm); |

这样form元素的表单字段就全部到FormData对象了。

最后就是被w3c标准化后的XMLHttpRequest功能更强了。

不仅可以直接发送FormData对象，还可以监视上传和下载的进度。

如果是图片，需要在上传之前预览的话，那就要用FileReader对象了。

使用FileReader对象,web应用程序可以异步的读取存储在用户计算机上的文件(或者原始数据缓冲)内容,可以使用File对象或者Blob对象来指定所要处理的文件或数据。

创建一个FileReader对象：

|  |  |
| --- | --- |
| 1 | var reader = new FileReader(); |

要实现预览图片的话，主要是用它的readAsDataURL方法，这个方法读取到的文件内容是data:URL格式的字符串，可以直接放在img标签的src里面。

还有一个是它的onload事件，当读取操作成功完成时触发，可以在读取完成时把结果给img标签。

来个实例。

HTML代码清单如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <title>HTML5 File Upload Demo</title>  <script type="text/javascript" src="upload.js"></script>  <style type="text/css">     \* {      padding : 0;      margin : 0;     }     .item {      border : 2px solid #ccc;      padding: 20px;      margin: 50px;      width: 350px;     }  </style>  </head>  <body>  <div>  <h2>示例</h2>  <input id="file" type="file" name="file">  <br>  <button type="button" id="button">点击上传</button>  <div>进度:<span id="progress"></span></div>  <img id="uploadPreview" style="width: 100px; height: 100px;" src="data:image/svg+xml,%3C%3Fxml%20version%3D%221.0%22%3F%3E%0A%3Csvg%20width%3D%22153%22%20height%3D%22153%22%20xmlns%3D%22http%3A//www.w3.org/2000/svg%22%3E%0A%20%3Cg%3E%0A%20%20%3Ctitle%3ENo%20image%3C/title%3E%0A%20%20%3Crect%20id%3D%22externRect%22%20height%3D%22150%22%20width%3D%22150%22%20y%3D%221.5%22%20x%3D%221.500024%22%20stroke-width%3D%223%22%20stroke%3D%22%23666666%22%20fill%3D%22%23e1e1e1%22/%3E%0A%20%20%3Ctext%20transform%3D%22matrix%286.66667%2C%200%2C%200%2C%206.66667%2C%20-960.5%2C%20-1099.33%29%22%20xml%3Aspace%3D%22preserve%22%20text-anchor%3D%22middle%22%20font-family%3D%22Fantasy%22%20font-size%3D%2214%22%20id%3D%22questionMark%22%20y%3D%22181.249569%22%20x%3D%22155.549819%22%20stroke-width%3D%220%22%20stroke%3D%22%23666666%22%20fill%3D%22%23000000%22%3E%3F%3C/text%3E%0A%20%3C/g%3E%0A%3C/svg%3E" alt="Image preview" />  </div>  </body>  </html> |

首先获取到上传按钮，file input和进度显示的DOM对象：

|  |  |
| --- | --- |
| 1  2  3 | var btn = document.getElementById("button");  var input = document.getElementById("file");  var pr = document.getElementById("progress"); |

然后给上传按钮添加点击上传的事件：

|  |  |
| --- | --- |
| 1 | btn.addEventListener("click", uploader, false); |

接下来实现上传的功能和进度监视的功能。

uploader上传功能的代码清单：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | var uploader = function() {      if(input.files.length == 0) {             return false;      }        //创建FormData对象和XMLHttpRequest对象。      var form = new FormData();      var xhr = new XMLHttpRequest();        xhr.upload.addEventListener("progress", updateProgress, false); //监视进度        form.append('file', input.files[0]);//把文件添加到FormData对象        xhr.open("POST", "upload.php");      xhr.send(form);//上传文件        xhr.onreadystatechange = function() {         //.....      };    }; |

XMLHttpRequest分别有四个事件可以监听：

1、progress：处理中。

2、load：传输完成。

3、error：传输失败，出错了。

4、abort：传输退出，被终止了。

如果要监听这四个事件，都必须在调用XMLHttpRequest.open方法之前设定。

上传和下载都有这个四个事件。

如果是下载，四个事件是由XMLHttpRequest触发的，也就是要设置在XMLHttpRequest对象上面：

|  |  |
| --- | --- |
| 1  2  3  4 | xhr.addEventListener("progress", updateProgress, false);  xhr.addEventListener("load", transferComplete, false);  xhr.addEventListener("error", transferFailed, false);  xhr.addEventListener("abort", transferCanceled, false); |

如果是上传的话，事件是由XMLHttpRequest.upload触发的，也就是要设置在XMLHttpRequest.upload属性上面：

|  |  |
| --- | --- |
| 1  2  3  4 | xhr.upload.addEventListener("progress", updateProgress, false);  xhr.upload.addEventListener("load", transferComplete, false);  xhr.upload.addEventListener("error", transferFailed, false);  xhr.upload.addEventListener("abort", transferCanceled, false); |

这里要注意上面这点区别。

下面是上传进度监听的函数代码清单：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | var updateProgress = function(evt) {      console.log(evt.lengthComputable);      if(evt.lengthComputable) {          var prN = Math.round( (evt.loaded / evt.total) \* 100 ) + "%";          pr.textContent = prN;      } else {          pr.textContent = "不能计算进度信息!";      }  }; |

进度信息会在事件属性里面。

upload.js的完整代码,如下：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66 | document.onreadystatechange = function(){      if(document.readyState == "complete") {          var btn = document.getElementById("button");          var input = document.getElementById("file");          var pr = document.getElementById("progress");            //进度处理回调          var updateProgress = function(evt) {              console.log(evt.lengthComputable);              if(evt.lengthComputable) {                  var prN = Math.round( (evt.loaded / evt.total) \* 100 ) + "%";                  pr.textContent = prN;              } else {                  pr.textContent = "不能计算进度信息!";              }          };            //上传方法          var uploader = function() {              if(input.files.length == 0) {                  return false;              }                //创建FormData对象和XMLHttpRequest对象。              var form = new FormData();              var xhr = new XMLHttpRequest();                xhr.upload.addEventListener("progress", updateProgress, false);//监视进度                form.append('file', input.files[0]); //把文件添加到FormData对象                xhr.open("POST", "upload.php");              xhr.send(form); //上传文件                xhr.onreadystatechange = function() {                    //.....              };            };            //图片预览          var preview = function() {              var reader = new FileReader();              var filter = /^(?:image\/bmp|image\/cis\-cod|image\/gif|image\/ief|image\/jpeg|image\/jpeg|image\/jpeg|image\/pipeg|image\/png|image\/svg\+xml|image\/tiff|image\/x\-cmu\-raster|image\/x\-cmx|image\/x\-icon|image\/x\-portable\-anymap|image\/x\-portable\-bitmap|image\/x\-portable\-graymap|image\/x\-portable\-pixmap|image\/x\-rgb|image\/x\-xbitmap|image\/x\-xpixmap|image\/x\-xwindowdump)$/i;                if(input.files.length == 0) {                  return false;              }              var file = input.files[0];              //过滤非图片              if(!filter.test(file.type)) {                  return false;              }                //文件读取完成之后，目标元素的result属性会包含有一个data: URL字符串代表文件内容              reader.addEventListener("load", function(evt){                  document.getElementById("uploadPreview").src = evt.target.result;              }, false);                reader.readAsDataURL(file); //读取文件内容          }            btn.addEventListener("click", uploader, false); //点击上传          input.addEventListener("change", preview, false); //选取文件之后，预览图片      }  } |

然后是上传服务器端的脚步upload.php:

|  |  |
| --- | --- |
| 1  2 | <?php  var\_dump($\_FILES); |

很简单，我们只要打印出上传信息就足够了。

如果在本地测试的话，速度太快，你可能会看不见上传进度百分比的变化。