formidable 与Node.js 多文件/图片上传

2016年06月14日 1004 声明

笔者所参与的一个APP项目中，有一个上传多个张图片的需求。虽然之前已经使用 `formidable` 模块实现了文件/图片的接收，但只能上传一张图片，要满足多张图片上传还要做一些处理。

1. 文件上传与 `formidable` 的一些介绍

1.1 HTTP文件上传

HTTP文件数据提交与接收不同于普通数据，文件基于二进制发送和接收。服务器通过 `Content-Type` 来判断内容类型，当其值为 `multipart/form-data` 时，说明收到的是文件数据。

更多关于文件上传的介绍，请参考：Node.js HTTP服务器中不依赖第三方模块的文件、图片上传 (http://itbilu.com/nodejs/core/VkK5RCoXW.html)

1.2 `formidable` 模块

`formidable` （ `node-formidable` ）是一个Node.js `form` 数据解析模块，非常适合用于文件上传的处理。本站曾介绍过使用这个模块接收单个上传文件（请参考： Node.js文件上传处理模块 `formidable` (http://itbilu.com/nodejs/npm/NkGKcF14.html)），接下来介绍用它实现多个文件/图片的上传。

初始化与数据解析

安装并引用 `formidable` 模块，要调用 `IncomingForm()` 构造函数初始模块。Node.js对HTTP请求的处理是，将用户请求数据封装到 `req` 对象中，该对象是一个 `IncomingMessage` (http://itbilu.com/nodejs/core/N1okQ7Eh.html#incomingMessage)对象实例。 `formidable` 解析用户上传数据也就是对这个对象的解析，初始化后就可以通过实例的 `.parse` 方法来解析数据。

当用户使用 `form` 表单提交数据时，表单中可能会包含两类数据：文件/图片数据、普通表单数据。 `formidable` 解析用户后，会将这两种数据分别放到 `files` 和 `fields` 两个回调参数中。

```
var form = new formidable.IncomingForm();
// req 即用户请求对象
form.parse(req, function (err, fields, files) {
  // fields 是普通表单数据
  // files 是文件数据
})
```

`files` 和 `fields` 两个数据对象结构类似如下：

```
{ fields: { field1: '111', field2: '222' },
  files:
   { file1: {/* 文件1 */},
     file2: {/* 文件2 */}
   }
}
```

`formidable` 解析完用户请求对象后，会将上传文件/图片数据放到第二个回调参数 `files` 中，每上传文件/图片是该对象一个属性。

2. `formidable` 实现多文件上传

这个文件上传功能基于Express (http://itbilu.com/nodejs/npm/EJUJrGVsg.html)框架实现，创建应用添加如下两上路由：

```
/* 显示一个上传页面 */
router.get('/', function(req, res) {});
/* 接收上传数据 */
router.post('/upload', function(req, res, next){});
```

为了测试上传功能，我们需要加载一个 `form` 表单页，用于提交上传文件，实现代码如下：

```
/* 显示一个上传页面 */
router.get('/', function(req, res) {
  res.writeHead(200, {'content-type': 'text/html'});
  res.end(
    '<form action="/upload" enctype="multipart/form-data" method="post">'+
      '<input type="text" name="field1" /> '+
      '<input type="text" name="field2" /> '+
      '<input type="file" name="file1" multiple="multiple" /> '+
      '<input type="file" name="file2" multiple="multiple" /> '+
      '<input type="submit" value="Upload" />'+
    '</form>'
  );
});
```

formidable 会将上传文件放到第二个回调参数 files 中，我们一般会将上传文件放到一个临时目录，检查完上传文件合法性后，再将其移动到目标目录。接收多个上传文件处理也类型，在本例中我们简单将文件移动到目录并返回一个包含文件Url的列表。实现代码如下：

```
/* 接收上传数据 */
router.post('/upload', function(req, res, next){
  var form = new formidable.IncomingForm();
  form.uploadDir = '/tmp';   //文件保存在系统临时目录
  form.maxFieldsSize = 1 * 1024 * 1024;  //上传文件大小限制为最大1M
  form.keepExtensions = true;        //使用文件的原扩展名

  var targetDir = path.join(__dirname, '../public/upload');
  // 检查目标目录，不存在则创建
  fs.access(targetDir, function(err){
    if(err){
      fs.mkdirSync(targetDir);
    }
    _fileParse();
  });

  // 文件解析与保存
  function _fileParse() {
    form.parse(req, function (err, fields, files) {
      if (err) throw err;
          var filesUrl = [];
          var errCount = 0;
          var keys = Object.keys(files);
      keys.forEach(function(key){
        var filePath = files[key].path;
        var fileExt = filePath.substring(filePath.lastIndexOf('.'));
        if (('.jpg.jpeg.png.gif').indexOf(fileExt.toLowerCase()) === -1) {
                errCount += 1;
        } else {
                //以当前时间戳对上传文件进行重命名
            var fileName = new Date().getTime() + fileExt;
            var targetFile = path.join(targetDir, fileName);
            //移动文件
            fs.renameSync(filePath, targetFile);
            // 文件的Url（相对路径）
            filesUrl.push('/upload/'+fileName)
        }
      });

      // 返回上传信息
      res.json({filesUrl:filesUrl, success:keys.length-errCount, error:errCount});
    });
  }
});
```

在上面代码中，我们使用 fs.renameSync (http://itbilu.com/nodejs/core/E1Abosjbe.html)来移动上传文件。该方法是一个同步方法，如果使用异步方法时需要借Async (http://itbilu.com/nodejs/npm/NyC1zNUE.html)等库来进行异步流程控制。

下一篇：bluebird与原生Promise对象及bluebird模块的中文API文档 (/nodejs/npm/VJHw6ScNb.html)

上一篇：redis - Node.js Redis客户端模块 (/nodejs/npm/EkiI9PG4-.html)

文章分类

- 包、应用

阅读排行

- Sequelize 中文API文档－1. 快速入门、Seq... (/nodejs/npm/VkYIaRPz-.html) (31764)
- Sequelize 中文API文档－2. Model 的定... (/nodejs/npm/V1PExztfb.html) (30614)
- 解决类似 /usr/lib64/libstdc++.so.... (/linux/management/NymXRUieg.html) (12624)
- Sequelize 中文API文档－3. 模型（表）之间的... (/nodejs/npm/41qaV3czb.html) (9860)
- Sequelize 中文API文档－4. 查询与原始查询 (/nodejs/npm/VJIR1CjMb.html) (8982)
- HTTP请求方法：GET、HEAD、POST、PUT、DE... (/other/relate/EkwKysXII.html) (6236)
- Linux升级安装GCC (/linux/management/V1vdnt9ll.html) (4735)
- Redis设置认证密码 Redis使用认证密码登录 在Re... (/database/redis/Ey_r7mWR.html) (4121)
- MQTT协议－MQTT协议简介及协议原理 (/other/relate/4kHBsx_Pg.html) (4081)
- ［ES6］Promise对象Promise.all()方法... (/javascript/js/41KMSZ9a.html) (3777)

最新文章

- Linux split 大文件分割与 cat合并文件 (/linux/man/Nkz2hoeNm.html)
  当需要将较大的数据上传到服务器，或从服务器下载较大的日志文件时，往往会因为网络或其它原因而导致传输...
- curl 命令行工具的使用及命令参数说明 (/linux/man/4yZ9qH_7X.html)
  `curl` 是一个开源的用于数据传输的命令行工具与库，它使用URL语法格式，...
- Sequelize 中文API文档－10. Migrati... (/nodejs/npm/VyqgRUVf7.html)
  Sequelize `2.0.0` 引入了一个新的CLI（命令行工具），其基于...
- 高性能分布式队列系统 Beanstalkd 介绍及使用 (/other/relate/VkBat8I-X.html)
  `Beanstalkd` 是一个简单、高效的工作队列系统，其最初设计目的是通过...
- CSS 选择器与JavaScript DOM元素查找方法 ... (/javascript/js/4J5hj63Cz.html)
  querySelector()及querySelectorAll()是Document DOM对象...

交流群：564850876