Egg.js 上传图片到本地

主讲教师：（大地）
合作网站：www.itying.com （IT 营）
我的专栏：https://www.itying.com/category-79-b0.html

一、Egg.js 中上传单个文件

请求 body 除了可以带参数之外，还可以发送文件，一般来说，浏览器上都是通过 Multipart/form-data 格式发送文件的，框架通过内置 Multipart 插件来支持获取用户上传的文件。**egg-multipart 插件地址：**https://github.com/eggjs/egg-multipart

注意：框架通过内置 Multipart 插件，所以我们直接使用就可以了。

File 模式的文件上传:

如果你完全不知道 Nodejs 中的 Stream 用法，那么 File 模式非常合适你。

1）在 config 文件中启用 file 模式：

```
// config/config.default.js
exports.multipart = {
  mode: 'file',
};
```

2）上传 / 接收文件：

你的前端静态页面代码应该看上去如下样子：

```
<form method="POST" action="/upload?_csrf=<%=csrf%>" enctype="multipart/form-data">
  title: <input name="title" />
  file: <input name="file" type="file" />
  <button type="submit">Upload</button>
</form>
```

对应的后端代码如下：

```
const fs = require('mz/fs');  //这个库是 node.js API 各个方面的包装器
const path = require('path');
const pump = require('mz-modules/pump');

async doAdd() {
        let { ctx } = this;
        let body = ctx.request.body;
        let file = ctx.request.files[0];
        if (file) {
            const filename =file.filename;
            const targetPath = path.join('app/public/upload', filename);
            const source = fs.createReadStream(file.filepath);
            const target = fs.createWriteStream(targetPath);
            try {
                await pump(source, target);
            } finally {
                await ctx.cleanupRequestFiles();
            }
        }
        this.ctx.body = {
            body: body,
            file: file
        };
    }
```

二、Egg.js 中上传多个文件

你的前端静态页面代码应该看上去如下样子：

```
<form method="POST" action="/upload?_csrf=<%=csrf%>" enctype="multipart/form-data">
  title: <input name="title" />
  file1: <input name="file1" type="file" />
  file2: <input name="file2" type="file" />
  <button type="submit">Upload</button>
</form>
```

对应的后端代码：

```
async doAdd() {
        let { ctx } = this;
        let body = ctx.request.body;
        const files = ctx.request.files;
        try {
            for (const file of files) {
                if (file) {
                    const filename = file.filename;
                    const targetPath = path.join('app/public/upload', filename);
                    const source = fs.createReadStream(file.filepath);
                    const target = fs.createWriteStream(targetPath);
                    await pump(source, target);
                }
            }
        } finally {
            // delete those request tmp files
            await ctx.cleanupRequestFiles();
        }
        this.ctx.body = {
            body: body,
            files: files
        };
    }
```