How to perform file upload?

I am going to use [express framework](http://codeforgeek.com/2014/10/top-5-node-packages/) and middleware called **“multer”.**This middleware is designed for handling the multipart/form-data.

In front-end script, we will set the **“target”**parameter of **FORM**to our router. for.eg  **/upload/picture**and in **app.get(‘/upload/picture’, …)**we will be able to handle our file. Let’s see how.

Our project:

This project is simple. We will have simple express web server which does the task of routing and other stuff. **Multer**will take care of the file handling and HTML for handling form input.

**package.json**

{  
  "name": "file\_upload",  
  "version": "0.0.1",  
  "dependencies": {  
    "express": "~4.10.2",  
    "multer": "~0.1.6"  
  }  
}

Installing dependency:

Save the package.json in folder and type

npm install

to install the required stuff.

**Server.js**

*/\*Define dependencies.\*/*  
  
var express=require("express");  
var multer  = require('multer');  
var app=express();  
var done=false;  
  
*/\*Configure the multer.\*/*  
  
app.use(multer({ dest: './uploads/',  
 rename: function (fieldname, filename) {  
    return filename+Date.now();  
  },  
onFileUploadStart: function (file) {  
  console.log(file.originalname + ' is starting ...')  
},  
onFileUploadComplete: function (file) {  
  console.log(file.fieldname + ' uploaded to  ' + file.path)  
  done=true;  
}  
}));  
  
*/\*Handling routes.\*/*  
  
app.get('/',function(req,res){  
      res.sendfile("index.html");  
});  
  
app.post('/api/photo',function(req,res){  
  if(done==true){  
    console.log(req.files);  
    res.end("File uploaded.");  
  }  
});  
  
*/\*Run the server.\*/*  
app.listen(3000,function(){  
    console.log("Working on port 3000");  
});

**index.html**

<[form](http://december.com/html/4/element/form.html) id        =  "uploadForm"  
     enctype   =  "multipart/form-data"  
     action    =  "/api/photo"  
     method    =  "post"  
>  
<[input](http://december.com/html/4/element/input.html) type="file" name="userPhoto" />  
<[input](http://december.com/html/4/element/input.html) type="submit" value="Upload Image" name="submit">  
</[form](http://december.com/html/4/element/form.html)>

Running project:

Copy code and save them in folder. Type

node Server.js

to run the project. Visit **localhost:3000** to view the app. Select the file and check the **uploads** folder. Your file must be present there!

Explanation :

In our Server.js file, we have configured **multer**. Multer emits event on particular situation such as we used **onFileUploadStart** which means when file start uploading, this event will be emitted.

After completing upload , **file** variable holds following array of information.

{   
  userPhoto:  
  {   
     fieldname: 'userPhoto',  
     originalname: 'banner.png',  
     name: 'banner1415699779303.png',  
     encoding: '7bit',  
     mimetype: 'image/png',  
     path: 'uploads\\banner1415699779303.png',  
     extension: 'png',  
     size: 11800,  
     truncated: false,  
     buffer: null   
 }   
}

We used **rename** function to manually pass the name of files, else by default multer use the random names for uniqueness.

As soon as **onFileUploadComplete** event emitted we have set the variable**done** to true and use it in our router to determine whether file is uploaded or not.

 if(done==true){   
    res.end("File uploaded.");  
  }

In HTML form you must mention **enctype=”multipart/form-data”** else multer will not work.

Performing file validation:

To perform validation on Server end, multer provides **limits** array parameter which have following parameters. ( Reference: **Github official page**. )

* **fieldNameSize**– integer – Max field name size (Default: 100 bytes)
* **fieldSize**– integer – Max field value size (Default: 1MB)
* **fields**– integer – Max number of non-file fields (Default: Infinity)
* **fileSize**– integer – For multipart forms, the max file size (in bytes) (Default: Infinity)
* **files**– integer – For multipart forms, the max number of file fields (Default: Infinity)
* **parts**– integer – For multipart forms, the max number of parts (fields + files) (Default: Infinity)
* **headerPairs**– integer – For multipart forms, the max number of header key=>value pairs to parse Default: 2000 (same as node’s http).

You can define it like this

limits: {  
  fieldNameSize: 100,  
  files: 2,  
  fields: 5  
}

In our code.

app.use(multer({ dest: './uploads/',  
 rename: function (fieldname, filename) {  
    return filename+Date.now();  
  },  
limits: {  
  fieldNameSize: 100,  
  files: 2,  
  fields: 5  
}  
}));

Conclusion:

Multer is very nice middleware created by Express community. It really helps us to quickly develop critical code like File uploads in [Node.js](http://codeforgeek.com/2014/11/basic-http-calls-using-node-js/) easily. I hope you find this tutorial helpful.