Subiendo archivos con Multer

DigitalHouse >
Coding School

Para **subir archivos** a través de un formulario con el objetivo de almacenarlos en un servidor, se requiere de un proceso específico.

¿Cómo funciona Multer?

Hasta el momento tenemos instalado **Multer** y también hemos creado el **HTML** para poder subir nuestro archivo. Veamos qué necesitamos del lado del servidor:

require 'multer'

Requerimos el **módulo multer** para poder utilizar todos los métodos que nos provee.

```
{}  const multer = require('multer');
```

¿Cómo funciona Multer?

Multer ofrece la opción de almacenar archivos en el disco, por lo que usaremos el método `.diskStorage()`. Este método recibe como parámetro un **objeto literal** con dos propiedades: `destination` y `filename`.

{}

```
var storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, '/uploads')
  },
  filename: function (req, file, cb) {
    cb(null, file.fieldname + '-' + Date.now())
  }
})

var upload = multer({ storage: storage })
```

.diskStorage()

destination

Permite definir la carpeta donde se va a almacenar el archivo.

```
{}
var storage = multer.diskStorage({
  destination: function (req, file, cb) {
    //...
  },
  filename: function (req, file, cb) {
    //...
  }
})
```

filename

Permite indicar con qué nombre se guardará ese archivo en el servidor.

destination

En destination usaremos el **callback** (**cb**) para definir la carpeta en donde queremos almacenar los archivos. El primer parámetro será `null`, el segundo, **la ruta hacia la carpeta de destino**. Si no se proporciona ningún destino, se utiliza el directorio predeterminado del sistema operativo para archivos temporales.

```
{}
var storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, 'public/img/uploads')
  },
  filename: function (req, file, cb) {
    cb(null, file.fieldname + '-' + Date.now())
  }
})
var upload = multer({ storage: storage })
```

filename

En filename, usaremos el **callback** (**cb**) para definir el nombre con el que guardaremos el archivo. El primer parámetro será `null`, el segundo, el nombre del archivo. Por ejemplo, podemos usar la variable `file` en el paquete `path` para crear el nombre del archivo, junto con el método `extname()` del paquete, pasándole como parámetro el nombre original del archivo para que nos devuelva únicamente su extensión.

{}

```
var storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, 'public/img/uploads')
  },
  filename: function (req, file, cb) {
   cb(null, file.fieldname + '-' + Date.now() + path.extname(file.originalname))
  }
})
var upload = multer({ storage: storage })
```

Configurando la ruta

Ya vimos cómo indicar dónde se almacenará y cómo se llamará el archivo que estamos subiendo al servidor. Ahora veamos cómo configuramos la ruta que se encargará de manejar la petición de la subida de archivos, es decir, la ruta que indicamos en nuestro `action` del formulario.

Importante: deberemos implementar todo el código de configuración en cada archivo en donde exista una ruta que esté implementando la subida de archivos.

```
{}
var upload = multer({ storage: storage });
app.post('/register', upload.single('avatarFile'), (req, res) => {
  console.log(req.file) // Nos devuelve un objeto con la información del archivo
  res.send('Archivo subido correctamente')
})
```

Subiendo un archivo

Para indicar que vamos a subir un archivo usamos `.single('nombre')`, donde nombre debe coincidir con el atributo name del input del formulario.

```
var upload = multer({ storage: storage });
app.post('/register', upload.single('avatarFile'), (req, res) => {
  console.log(req.file) // Nos devuelve un objeto con la información del archivo
  res.send('Archivo subido correctamente')
})
```

Subiendo múltiples archivos

Para indicar que vamos a subir más de un archivo usamos `.array('nombre')`, donde `nombre` debe coincidir con el atributo name del input del formulario, y no nos olvidemos que el input debe tener la propiedad `multiple`.

html

```html
<input type="file" name="avatarFiles" id="file"  multiple>
```

{}

```
var upload = multer({ storage: storage });
app.post('/register', upload.array('avatarFiles'), function (req, res, next) {
  console.log(req.files)
  res.send('Archivos subidos correctamente)
})
```

En la **documentación oficial de la librería**, podemos encontrar toda esta configuración ya definida para copiar e incorporar a nuestro proyecto.


DigitalHouse>
Coding School