**КТ № 15. Конфигурация multer. Загрузка файлов с помощью multer**

**Время на выполнение практической работы:** 4 часа.

**Теоретические сведения к выполнению практической работы:**

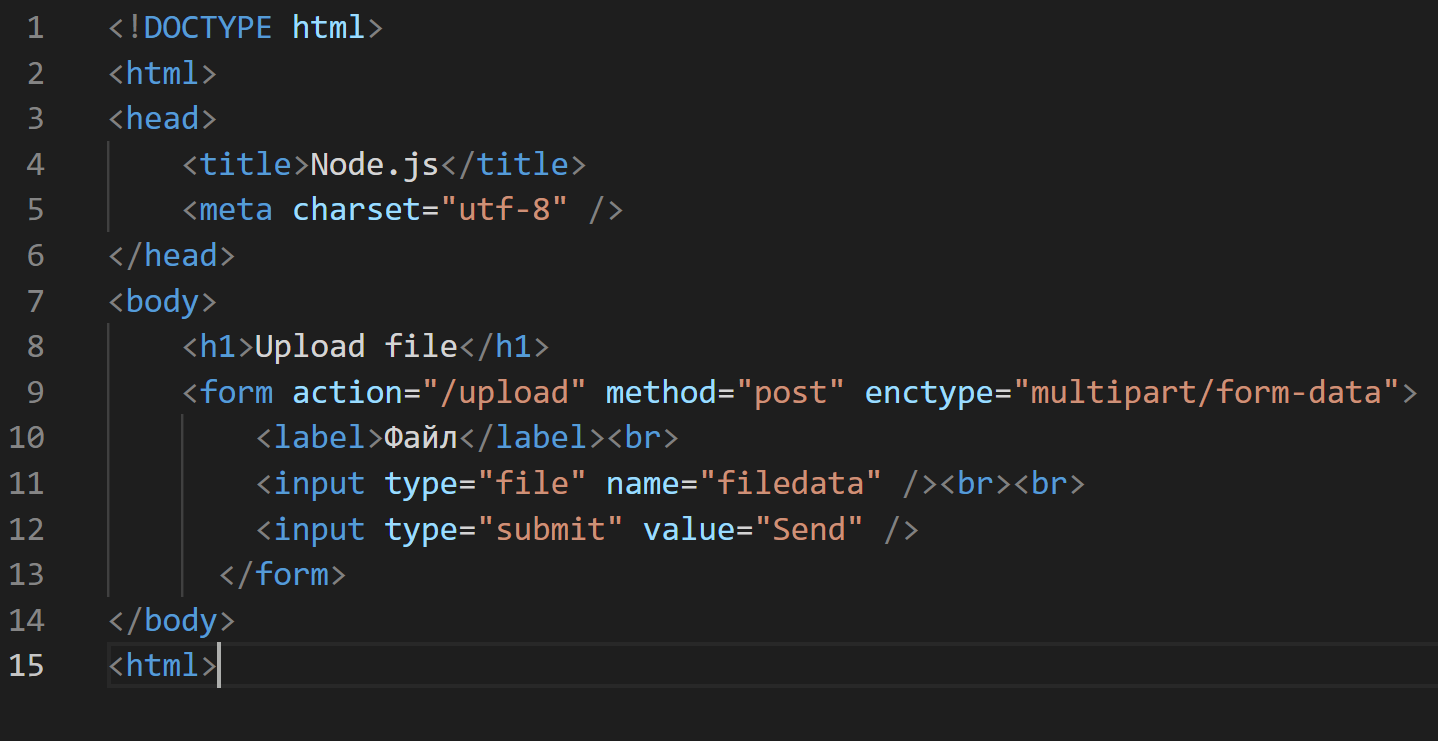
**Загрузка файлов с помощью multer**

Одной из наиболее часто встречаемых задач в Node.js является загрузка файлов на сервер. В Node.js для этого нет встроенных инструментов, однако мы можем использовать дополнительные специальные пакеты. Одним из популярных подобных пакетов является multer, который работает поверх Express.

Итак, для загрузки файлов создадим новый проект. Установим в него пакеты express и multer:



Определим в проекте новый файл index.html, в котором будет располагаться форма для загрузки файла:



Здесь важно отметить, что, чтобы на стороне сервера multer мог подхватить загруженный файл, у формы атрибут enctype должен иметь значение multipart/form-data.

В главном файле приложения app.js используем multer для получения отправленного файла:



В данном случае multer добавляется в виде компонента middleware. Для конфигурации в функцию multer передается объект, в котором параметр dest указывает на путь, по которому будет загружаться файл. В данном случае это папка uploads:



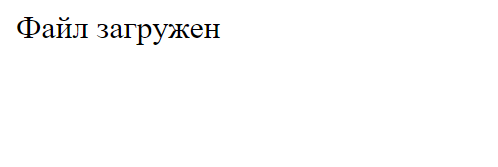
Если внутри проекта такой папки нет, то она автоматически будет создана.

Далее вызывается функция single() указывает, что загружаться будет один файл. Собственно, на форме в index.html мы имеем поле для загрузки одного файла. В этот метод передается название поля, которое используется на форме для загрузки файла. Соответствующее поле в файле index.html называется "filedata", поэтому в функцию single() передается соответствующее значение.

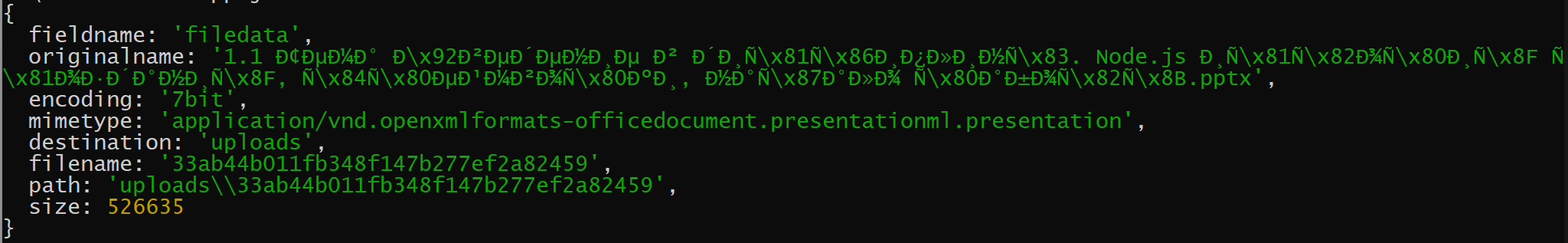
Настроив multer, мы можем при обработке запроса получить файл через req.file.

Для тестирования приложения запустим приложение, обратимся к странице index.html и загрузим какой-нибудь файл:



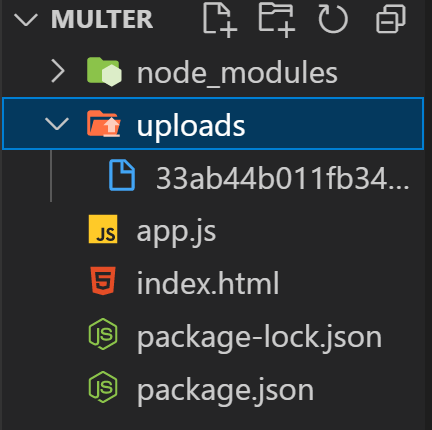


После этого на консоли мы увидим что-то наподобие:

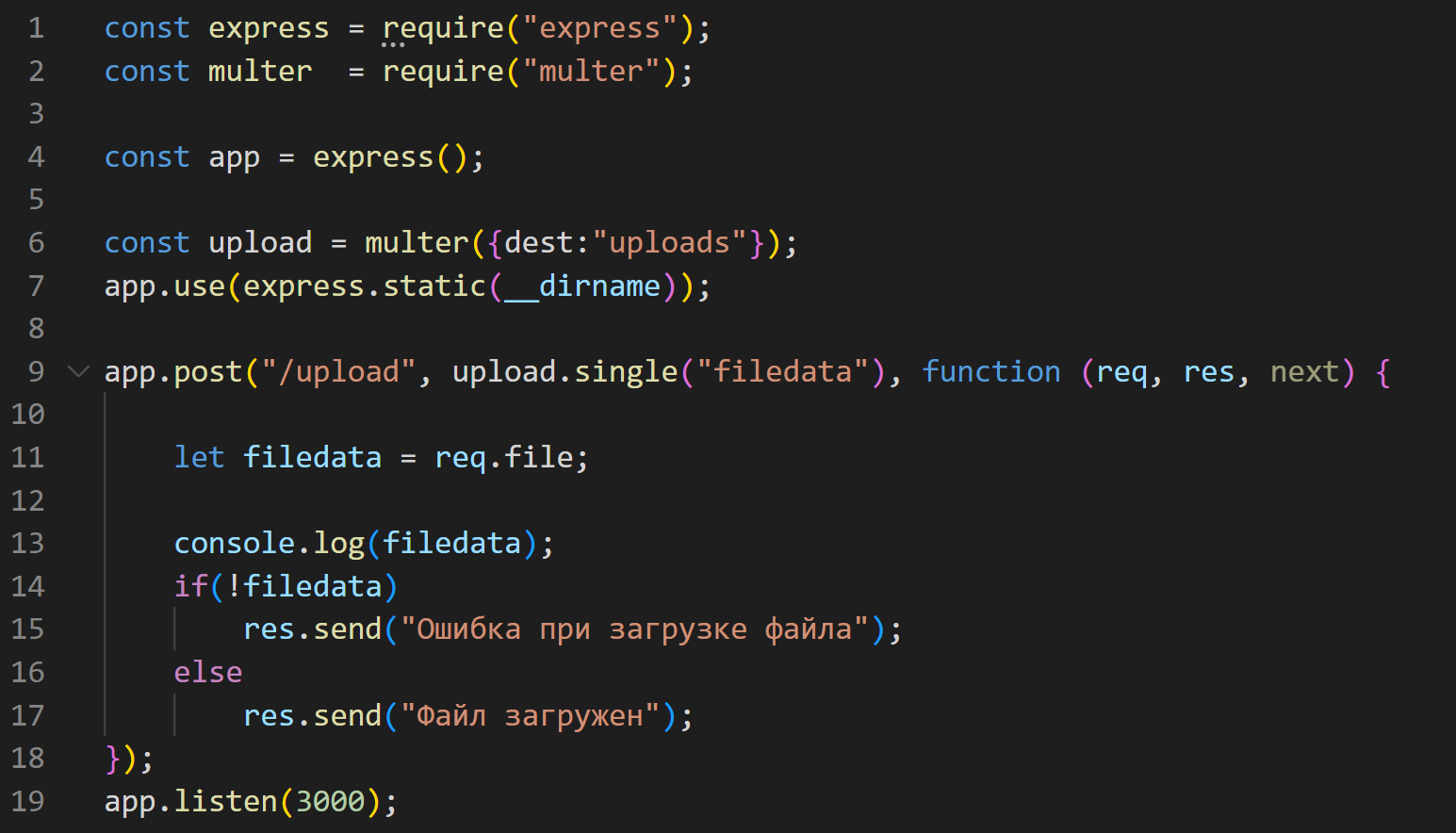


Здесь мы видим, какие метаданные получает сервер при загрузке файла, которые мы можем получить и как-нибудь использовать по своему усмотрению.

В результате в папке проекта будет создана подкаталог uploads, где мы найдем загруженный файл:



Также нам необязательно встраивать multer глобально в конвейер обработки запросов в виде компонента middleware. Вместо этого мы можем определить использование multer только для отдельных функций обработки запросов. Например:

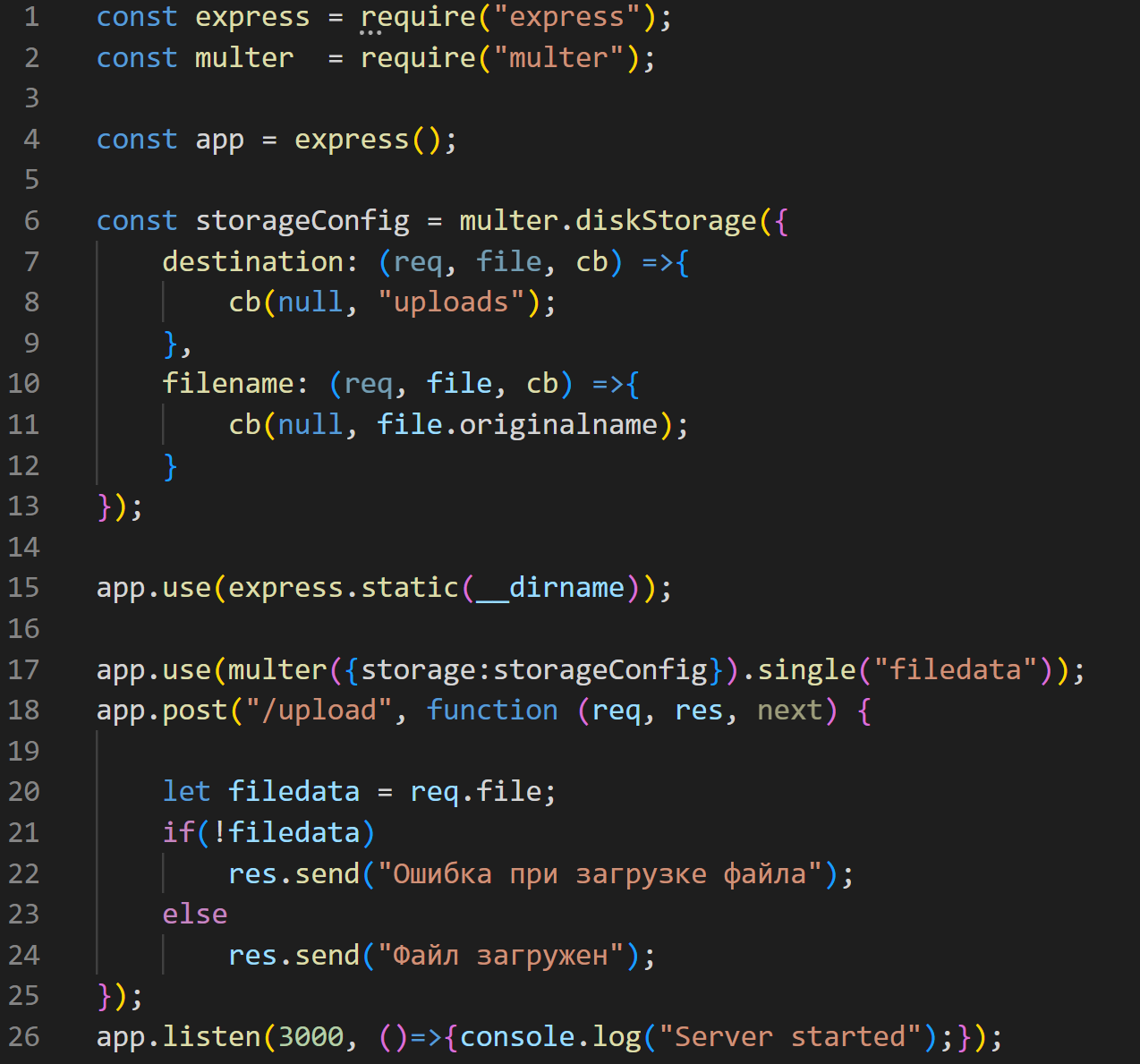


Здесь в функции app.post("/upload"... второй параметр upload.single("filedata") опять же устанавливает использование multer для загрузки одного файла.

**Настройка multer**

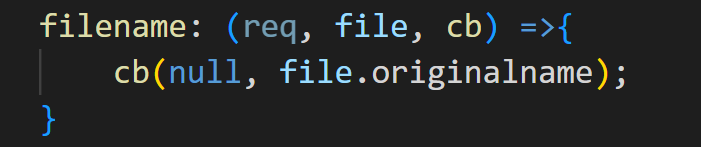
**Настройка параметров сохранения файла**

Ранее при загрузке имени файла присваивается некоторое произвольное значение. Тем не менее, возможно, нас такие имена файлов не устраивают, и мы хотим сами контролировать процесс именования файлов. В этом случае мы можем выполнить более точную настройку multer:



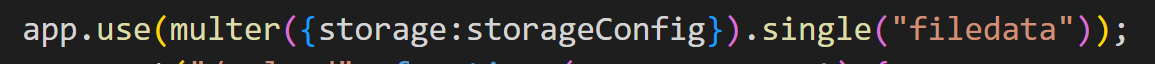
Для настройки сохранения файлов применяется функция multer.diskStorage(). Она принимает объект с двумя параметрами, каждый из которых представляет функцию:

* destination: определяет место для сохранения загруженных файлов - в данном случае папка "uploads".
* filename: определяет имя для загруженных файлов - в данном случае это непосредственно имя загруженного файла - file.originalname. Хотя здесь при необходимости можно использовать другие методы, например, добавить к имени файла текущую дату:



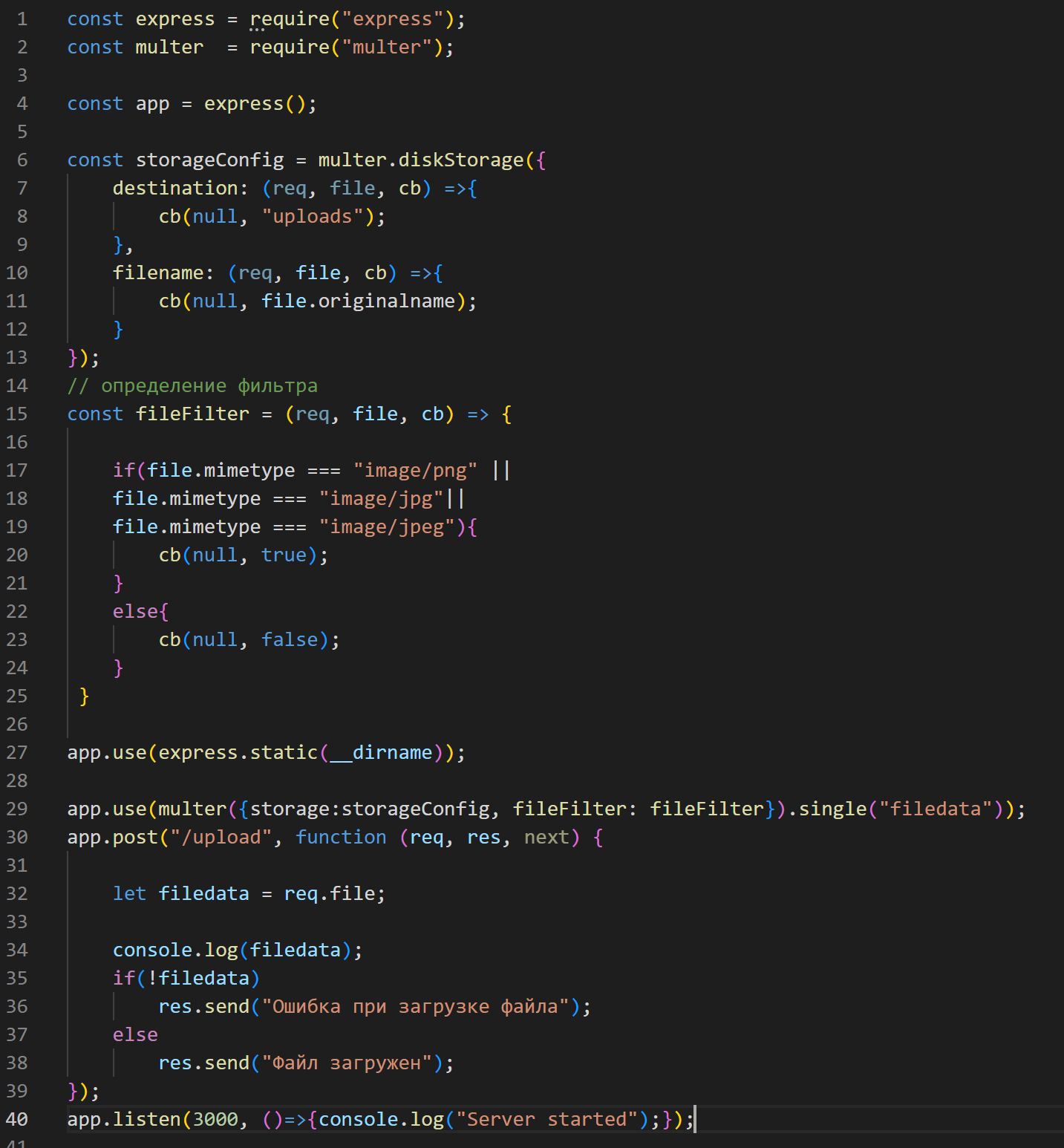
Оба параметра получаю объект запроса req, из которого при необходимости мы можем получить какие-то другие данные запроса и использовать их при сохранении файла.

Затем необходимо установить параметр storage:

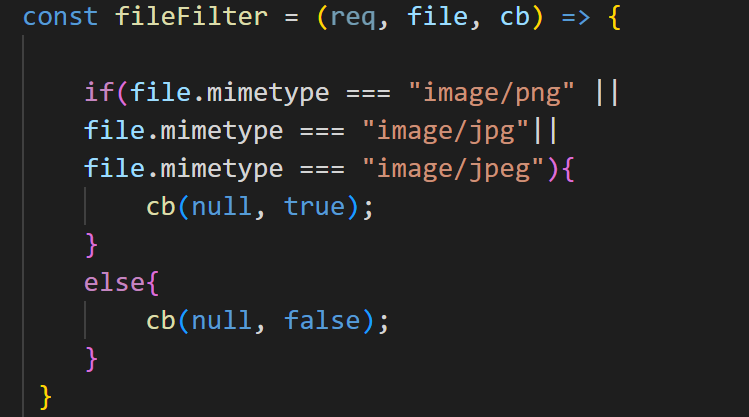


**Фильтрация файлов**

При сохранении файлов мы можем столкнуться с необходимостью их фильтрации по типу. Возможно, нам нужны только файлы изображений или файлы pdf или какие-то другие типы файлов. В этом случае мы можем определить фильтр. Например, будем принимать только файлы изображений:

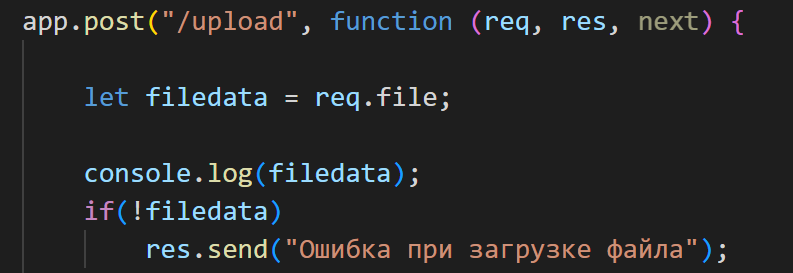


Фильтр фактически представляет функцию, которая принимает три параметра: req (объект запроса), file (загруженный файл) и функция cb():



С помощью значения file.mimetype мы можем проверить MIME-тип файла. Далее вызывается функция cb(). Если MIME-тип подходит, то есть мы хотим сохранить данный файл, то в качестве второго параметра в функцию cb передается true. Если же мы хотим отклонить файл, то передается значение false.

Когда выполняется вызов cb(null, false), то при получении multer не устанавливает значение req.file, то есть фактически оно равно undefined. Соответственно эту ситуацию мы можем обработать при получении запроса:



И чтобы применить данный фильтр, его необходимо передать в объект multer параметру fileFilter:



**Этапы выполнения:**

1. ***Создайте новый проект;***
2. ***Загрузите в него пакеты express и multer;***
3. ***Создайте форму для загрузки файла;***
4. ***Используйте multer для получения отправленного файла;***
5. ***Протестируйте работоспособность программы, загрузите какой-либо файл;***
6. ***Продемонстрируйте реакцию консоли на загрузку файла;***
7. ***Продемонстрируйте подкаталог uploads и загруженный файл;***
8. ***Проконтролируйте процесс именования файлов;***
9. ***Отфильтруйте файлы по типу.***