Streaming files

> info **Note** This chapter shows how you can stream files from your **HTTP application**. The examples presented below do not apply to GraphQL or Microservice applications.

There may be times where you would like to send back a file from your REST API to the client. To do this with Nest, normally you'd do the following:

```
@Controller('file')
export class FileController {
  @Get()
  getFile(@Res() res: Response) {
    const file = createReadStream(join(process.cwd(), 'package.json'));
    file.pipe(res);
  }
}
```

But in doing so you end up losing access to your post-controller interceptor logic. To handle this, you can return a `StreamableFile` instance and under the hood, the framework will take care of piping the response.

Streamable File class

A `StreamableFile` is a class that holds onto the stream that is to be returned. To create a new `StreamableFile`, you can pass either a `Buffer` or a `Stream` to the `StreamableFile` constructor.

> info **hint** The `StreamableFile` class can be imported from `@nestjs/common`.

Cross-platform support

Fastify, by default, can support sending files without needing to call `stream.pipe(res)`, so you don't need to use the `StreamableFile` class at all. However, Nest supports the use of `StreamableFile` in both platform types, so if you end up switching between Express and Fastify there's no need to worry about compatibility between the two engines.

Example

You can find a simple example of returning the `package.json` as a file instead of a JSON below, but the idea extends out naturally to images, documents, and any other file type.

```
import { Controller, Get, StreamableFile } from '@nestjs/common';
import { createReadStream } from 'fs';
import { join } from 'path';

@Controller('file')
export class FileController {
  @Get()
```

```
  getFile(): StreamableFile {
    const file = createReadStream(join(process.cwd(), 'package.json'));
    return new StreamableFile(file);
  }
}
```

The default content type is `application/octet-stream`, if you need to customize the response you can use the `res.set` method or the `@Header()` decorator, like this:

```
import { Controller, Get, StreamableFile, Res } from '@nestjs/common';
import { createReadStream } from 'fs';
import { join } from 'path';
import type { Response } from 'express';

@Controller('file')
export class FileController {
  @Get()
  getFile(@Res({ passthrough: true }) res: Response): StreamableFile {
    const file = createReadStream(join(process.cwd(), 'package.json'));
    res.set({
      'Content-Type': 'application/json',
      'Content-Disposition': 'attachment; filename="package.json"',
    });
    return new StreamableFile(file);
  }

  // Or even:
  @Get()
  @Header('Content-Type', 'application/json')
  @Header('Content-Disposition', 'attachment; filename="package.json"')
  getStaticFile(): StreamableFile {
    const file = createReadStream(join(process.cwd(), 'package.json'));
    return new StreamableFile(file);
  }
}
```