请求文件

`File` 用于定义客户端的上传文件。

!!! info "说明"

因为上传文件以「表单数据」形式发送。

所以接收上传文件，要预先安装 `python-multipart`。

例如： `pip install python-multipart`。

导入 `File`

从 `fastapi` 导入 `File` 和 `UploadFile`：

```
{!../../../docs_src/request_files/tutorial001.py!}
```

定义 `File` 参数

创建文件（ `File` ）参数的方式与 `Body` 和 `Form` 一样：

```
{!../../../docs_src/request_files/tutorial001.py!}
```

!!! info "说明"

`File` 是直接继承自 `Form` 的类。

注意，从 `fastapi` 导入的 `Query`、`Path`、`File` 等项，实际上是返回特定类的函数。

!!! tip "提示"

声明文件体必须使用 `File`，否则，`FastAPI` 会把该参数当作查询参数或请求体（`JSON`）参数。

文件作为「表单数据」上传。

如果把*路径操作函数*参数的类型声明为 `bytes`，**FastAPI** 将以 `bytes` 形式读取和接收文件内容。

这种方式把文件的所有内容都存储在内存里，适用于小型文件。

不过，很多情况下， `UploadFile` 更好用。

含 `UploadFile` 的 `File` 参数

定义 `File` 参数时使用 `UploadFile`：

```
{!../../../docs_src/request_files/tutorial001.py!}
```

`UploadFile` 与 `bytes` 相比有更多优势：

- 使用 `spooled` 文件：
  - 存储在内存的文件超出最大上限时，FastAPI 会把文件存入磁盘；
- 这种方式更适于处理图像、视频、二进制文件等大型文件，好处是不会占用所有内存；
- 可获取上传文件的元数据；
- 自带 file-like `async` 接口；
- 暴露的 Python `SpooledTemporaryFile` 对象，可直接传递给其他预期「file-like」对象的库。

`UploadFile`

`UploadFile` 的属性如下：

- `filename`：上传文件名字符串（`str`），例如，`myimage.jpg`；
- `content_type`：内容类型（MIME 类型 / 媒体类型）字符串（`str`），例如，`image/jpeg`；
- `file`：`SpooledTemporaryFile`（file-like 对象）。其实就是 Python文件，可直接传递给其他预期 `file-like` 对象的函数或支持库。

`UploadFile` 支持以下 `async` 方法，（使用内部 `SpooledTemporaryFile`）可调用相应的文件方法。

- `write(data)`：把 `data`（`str` 或 `bytes`）写入文件；
- `read(size)`：按指定数量的字节或字符（`size`（`int`））读取文件内容；
- `seek(offset)`：移动至文件 `offset`（`int`）字节处的位置；
  - 例如，`await myfile.seek(0)` 移动到文件开头；
  - 执行 `await myfile.read()` 后，需再次读取已读取内容时，这种方法特别好用；
- `close()`：关闭文件。

因为上述方法都是 `async` 方法，要搭配「await」使用。

例如，在 `async` *路径操作函数* 内，要用以下方式读取文件内容：

```
contents = await myfile.read()
```

在普通 `def` *路径操作函数* 内，则可以直接访问 `UploadFile.file`，例如：

```
contents = myfile.file.read()
```

!!! note " `async` 技术细节"

使用 `async` 方法时，**FastAPI** 在线程池中执行文件方法，并 `awiat` 操作完成。

!!! note "Starlette 技术细节"

FastAPI 的 `UploadFile` 直接继承自 **Starlette** 的 `UploadFile`，但添加了一些必要功能，使之与 **Pydantic** 及 FastAPI 的其它部件兼容。

什么是「表单数据」

与 JSON 不同，HTML 表单（`<form></form>`）向服务器发送数据通常使用「特殊」的编码。

FastAPI 要确保从正确的位置读取数据，而不是读取 JSON。

!!! note "技术细节"

不包含文件时，表单数据一般用 `application/x-www-form-urlencoded`「媒体类型」编码。

但表单包含文件时，编码为 `multipart/form-data`。使用了 `File`，**FastAPI** 就知道要从请求体的正确位置获取文件。

编码和表单字段详见 <abbr title="Mozilla Developer Network">MDN</abbr> Web 文档的 <code>POST </code> 小节。

!!! warning "警告"

可在一个*路径操作*中声明多个 `File` 和 `Form` 参数，但不能同时声明要接收 JSON 的 `Body` 字段。因为此时请求体的编码是 `multipart/form-data`，不是 `application/json`。

这不是 **FastAPI** 的问题，而是 HTTP 协议的规定。

多文件上传

FastAPI 支持同时上传多个文件。

可用同一个「表单字段」发送含多个文件的「表单数据」。

上传多个文件时，要声明含 `bytes` 或 `UploadFile` 的列表（`List`）：

```
{!../../../docs_src/request_files/tutorial002.py!}
```

接收的也是含 `bytes` 或 `UploadFile` 的列表（`list`）。

!!! note "笔记"

注意，截至 2019 年 4 月 14 日，Swagger UI 不支持在同一个表单字段中上传多个文件。详见 #4276 和 #3641.

不过，**FastAPI** 已通过 OpenAPI 标准与之兼容。

因此，只要 Swagger UI 或任何其他支持 OpenAPI 的工具支持多文件上传，都将与 **FastAPI** 兼容。

!!! note "技术细节"

也可以使用 `from starlette.responses import HTMLResponse`。

`fastapi.responses` 其实与 `starlette.responses` 相同，只是为了方便开发者调用。实际上，大多数 **FastAPI** 的响应都直接从 Starlette 调用。

小结

本节介绍了如何用 `File` 把上传文件声明为（表单数据的）输入参数。