n,nnnnnnnn

在 `Retrofit2` 中发送多个文件和一段文字到 `FastAPI`，可以通过使用 `Multipart` 请求来实现。下面是如何在 Kotlin 中使用 Retrofit2 实现该功能，并在 FastAPI 中处理这些数据的示例。

1. Retrofit2 (客户端部分)

首先在 `Retrofit2` 中定义接口，将文件和文字通过 `@Multipart` 注解发送。

```
interface ApiService {
    @Multipart
    @POST("upload")
    suspend fun uploadFiles(
        @Part("description") description: RequestBody,
        @Part files: List<MultipartBody.Part>
    ): Response<Any>
}
```

创建 Retrofit 实例：

```
val retrofit = Retrofit.Builder()
    .baseUrl("http://your-fastapi-server.com/")
    .addConverterFactory(GsonConverterFactory.create())
    .build()

val apiService = retrofit.create(ApiService::class.java)
```

发送文件和文字数据：

```
val descriptionText = "这是描述文字"

// 创建描述文字的 RequestBody
val description = RequestBody.create("text/plain".toMediaTypeOrNull(),
descriptionText)

// 文件部分
val files = listOf(file1, file2) // 这是你要上传的文件列表

val multipartFiles = files.map { file ->
    val requestBody = file.asRequestBody("application/octet-
stream".toMediaTypeOrNull())
    MultipartBody.Part.createFormData("files", file.name, requestBody)
}
```

```
// 调用 API
val response = apiService.uploadFiles(description, multipartFiles)
```

2. FastAPI (服务器部分)

FastAPI 中处理上传的文件和文字可以使用 `File` 和 `Form`，多个文件可以使用 `List` 来接收。

```
from fastapi import FastAPI, File, UploadFile, Form
from typing import List

app = FastAPI()

@app.post("/upload")
async def upload_files(description: str = Form(...), files:
List[UploadFile] = File(...)):
    return {
        "description": description,
        "filenames": [file.filename for file in files]
    }
```

总结

- 在 `Retrofit2` 中使用 `@Multipart` 注解发送文件和文字。
- 在 FastAPI 中使用 `Form` 和 `File` 来处理文本和文件上传。

你可以通过 `Retrofit2` 发送一个 PNG 文件、一个 MP3 文件和一段文字到 `FastAPI`，依然使用 `Multipart` 请求方式。以下是如何实现这一功能的示例。

1. Retrofit2 (客户端部分)

在 `Retrofit2` 中定义接口，将 PNG 文件、MP3 文件和文字通过 `@Multipart` 注解发送。

```
interface ApiService {
    @Multipart
    @POST("upload")
    suspend fun uploadFiles(
        @Part("description") description: RequestBody,
        @Part image: MultipartBody.Part,
        @Part audio: MultipartBody.Part
    ): Response<Any>
}
```

创建 Retrofit 实例：

```
val retrofit = Retrofit.Builder()
    .baseUrl("http://your-fastapi-server.com/")
    .addConverterFactory(GsonConverterFactory.create())
    .build()

val apiService = retrofit.create(ApiService::class.java)
```

准备数据并发送请求：

```
// 准备文字部分
val descriptionText = "这是描述文字"
val description = RequestBody.create("text/plain".toMediaTypeOrNull(),
descriptionText)

// 准备 PNG 文件
val pngFile = File("/path/to/image.png")
val pngRequestBody = pngFile.asRequestBody("image/png".toMediaTypeOrNull())
val imagePart = MultipartBody.Part.createFormData("image", pngFile.name,
pngRequestBody)

// 准备 MP3 文件
val mp3File = File("/path/to/audio.mp3")
val mp3RequestBody = mp3File.asRequestBody("audio/
mpeg".toMediaTypeOrNull())
val audioPart = MultipartBody.Part.createFormData("audio", mp3File.name,
mp3RequestBody)

// 调用 API 上传文件和文字
val response = apiService.uploadFiles(description, imagePart, audioPart)
```

2. FastAPI (服务器部分)

在 FastAPI 中处理 PNG 文件、MP3 文件和文字可以使用 `Form` 和 `File`，多个文件可以分别接收。

```python
from fastapi import FastAPI, File, UploadFile, Form

app = FastAPI()

@app.post("/upload")
async def upload_files(
    description: str = Form(...),
    image: UploadFile = File(...),
    audio: UploadFile = File(...)
):
    return {
        "description": description,
        "image_filename": image.filename,
        "audio_filename": audio.filename
    }
```

总结

- 在 `Retrofit2` 中使用 `@Multipart` 注解分别发送 PNG 文件、MP3 文件和文字。
- 在 FastAPI 中使用 `Form` 和 `File` 来接收文字和文件，并分别处理。