Transformando o Projeto NoteApp em uma API REST com Spark Java

Samuel Lopes Soares: 323126781

Fernando Ferreira Soares: 32313336

1. Introdução

Este documento descreve passo a passo como transformar o projeto Java 'NoteApp' em uma API REST utilizando a biblioteca Spark Java. Essa abordagem é ideal para projetos simples, especialmente quando não se utiliza ferramentas como Maven ou Gradle para gerenciar dependências.

2. Requisitos

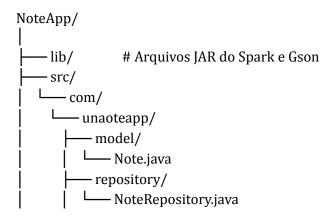
Antes de começar, você precisará dos seguintes arquivos JAR:

- spark-core-2.9.4.jar (para a criação da API REST)
- gson-2.10.1.jar (para conversão de objetos Java em JSON)

Coloque esses arquivos dentro de uma pasta chamada 'lib/' no seu projeto.

3. Estrutura Recomendada do Projeto

Uma estrutura de diretórios recomendada para o projeto seria:



```
NoteApi.java # Classe principal da API
out/ # Arquivos compilados (.class)
```

4. Criando a Classe NoteApi.java

A classe 'NoteApi' será responsável por disponibilizar os endpoints REST. Utilizamos o Spark Java para criar os endpoints e o Gson para serializar os objetos em JSON.

Exemplo de código para NoteApi.java:

```
public class NoteApi {
  static NoteRepository repo = new NoteRepository();
  static Gson gson = new Gson();
  public static void main(String[] args) {
    port(4567);
    get("/notes", (req, res) -> {
      res.type("application/json");
      return gson.toJson(repo.getAllNotes());
    });
    post("/notes", (req, res) -> {
      Note note = gson.from[son(req.body(), Note.class);
      repo.addNote(note.getTitle(), note.getContent());
      res.status(201);
      return "Nota criada com sucesso!";
    });
    get("/notes/:id", (req, res) -> {
      int id = Integer.parseInt(req.params(":id"));
      Note note = repo.getNoteById(id);
      if (note == null) {
        res.status(404);
        return "Nota não encontrada!";
      }
      res.type("application/json");
      return gson.toJson(note);
    });
    delete("/notes/:id", (req, res) -> {
      int id = Integer.parseInt(req.params(":id"));
```

```
if (repo.deleteNote(id)) {
    return "Nota deletada!";
} else {
    res.status(404);
    return "Nota não encontrada!";
}
});
}
```

5. Compilação e Execução da API

Use os seguintes comandos no terminal para compilar e rodar a API:

```
Linux/Mac:
```

```
javac -cp "lib/*" -d out src/com/unaoteapp/*.java NoteApi.java
java -cp "lib/*:out" NoteApi
Windows:
javac -cp "lib/*" -d out src\com\unaoteapp\*.java NoteApi.java
java -cp "lib/*;out" NoteApi
```

6. Testando a API

Você pode testar os endpoints utilizando ferramentas como o Postman ou Insomnia.

Exemplos:

- GET http://localhost:4567/notes
- POST http://localhost:4567/notes (body JSON: { "title": "Minha Nota", "content": "Texto aqui" })
- GET http://localhost:4567/notes/1
- DELETE http://localhost:4567/notes/1

7. Conclusão

Com esses passos, o projeto NoteApp passa a oferecer uma interface REST para integração com outras aplicações. Essa abordagem facilita a criação de clientes web, mobile ou integração com outras APIs.