Spring Data REST 模块

要使用 **spring-data-rest-core**，可以利用 Spring Data REST 模块将 Spring Data 仓库暴露为应用程序中的 RESTful 端点。以下是一个完整的、逐步指南，帮助你入门：

---

1. 添加 Spring Data REST 依赖

要使用 Spring Data REST，需要在项目中包含其依赖项。如果你使用的是 **Spring Boot**（常见情况），将以下依赖项添加到构建文件中：

- **Maven**：

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-rest</artifactId>
</dependency>
```

- **Gradle**：

```
implementation 'org.springframework.boot:spring-boot-starter-data-rest'
```

这个启动器会带来 `spring-data-rest-core` 以及其他必要的组件。对于非 Spring Boot 项目，可以直接包含 `spring-data-rest-core`，但推荐使用 Spring Boot 启动器以简化操作。

---

2. 定义实体

通过使用持久化技术（如 JPA）定义实体类来创建域模型。例如：

```
import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.GeneratedValue;

@Entity
public class User {
    @Id
    @GeneratedValue
    private Long id;
    private String name;
```

```
    // 构造函数
    public User() {}
    public User(String name) {
        this.name = name;
    }

    // Getter 和 Setter
    public Long getId() { return id; }
    public void setId(Long id) { this.id = id; }
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
}
```

这个 `User` 实体表示数据库中的一个简单表，包含 `id` 和 `name`。

---

3. 创建仓库接口

通过扩展 Spring Data 的仓库接口（如 `JpaRepository`）为实体定义一个仓库接口。例如：

```
import org.springframework.data.jpa.repository.JpaRepository;

public interface UserRepository extends JpaRepository<User, Long> {
}
```

通过扩展 `JpaRepository`，你可以免费获得基本的 CRUD（创建、读取、更新、删除）操作。Spring Data REST 会自动将这个仓库暴露为 RESTful 端点。

---

4. 运行应用程序

添加依赖项并定义实体和仓库后，启动 Spring Boot 应用程序。Spring Data REST 会根据仓库自动生成 REST 端点。对于上述的 `UserRepository`，可以访问：

- **GET /users**：检索所有用户的列表。
- **GET /users/{id}**：通过 ID 检索特定用户。
- **POST /users**：创建新用户（使用 JSON 有效负载，例如 `{"name": "Alice"}`）。
- **PUT /users/{id}**：更新现有用户。

- **DELETE /users/{id}**：删除用户。

例如，如果你的应用程序在 `localhost:8080` 上运行，可以使用 `curl` 或浏览器进行测试：

```
curl http://localhost:8080/users
```

响应将包括 HATEOAS 链接，允许客户端动态导航相关资源。

---

5. （可选）自定义 REST 端点

可以使用注解或配置自定义仓库的暴露方式：

- **更改端点路径**：使用 `@RepositoryRestResource` 注解指定自定义路径：

  ```
  import org.springframework.data.rest.core.annotation.RepositoryRestResource;

  @RepositoryRestResource(path = "people")
  public interface UserRepository extends JpaRepository<User, Long> {
  }
  ```

  现在，端点将是 `/people` 而不是 `/users`。

- **配置全局设置**：通过实现 `RepositoryRestConfigurer` 自定义基础路径或其他设置：

  ```
  import org.springframework.data.rest.webmvc.config.RepositoryRestConfigurer;
  import org.springframework.context.annotation.Configuration;
  import org.springframework.data.rest.core.config.RepositoryRestConfiguration;

  @Configuration
  public class RestConfig implements RepositoryRestConfigurer {
      @Override
      public void configureRepositoryRestConfiguration(RepositoryRestConfiguration config) {
          config.setBasePath("/api"); // 所有端点都以 /api 开头
      }
  }
  ```

  使用此配置，用户将可以在 `/api/users` 访问。

Spring Data REST 的关键功能

- **HATEOAS 支持**：响应包括指向相关资源的超媒体链接。
- **分页和排序**：使用查询参数（如 `?page=0&size=10&sort=name,asc`）对结果进行分页和排序。
- **过滤**：在仓库中添加自定义查询方法（例如 `findByName(String name)`）以暴露额外的端点。

总结

要使用 `spring-data-rest-core`：1. 包含 Spring Data REST 依赖项（例如，对于 Spring Boot，使用 `spring-boot-starter-data-rest`）。2. 使用适当的注解定义实体（例如，对于 JPA，使用 `@Entity`）。3. 创建扩展 Spring Data 接口的仓库接口（例如，`JpaRepository`）。4. 运行应用程序以访问自动生成的 REST 端点。5. 根据需要自定义端点路径或行为。

这种方法可以让你快速构建 RESTful API，利用 Spring Data REST 的核心功能。