springboot 集成mybatis-plus

创建项目

create-springboot-project: https://iovhm.com/book/books/bbcbf/page/create-springboot-project

集成数据库

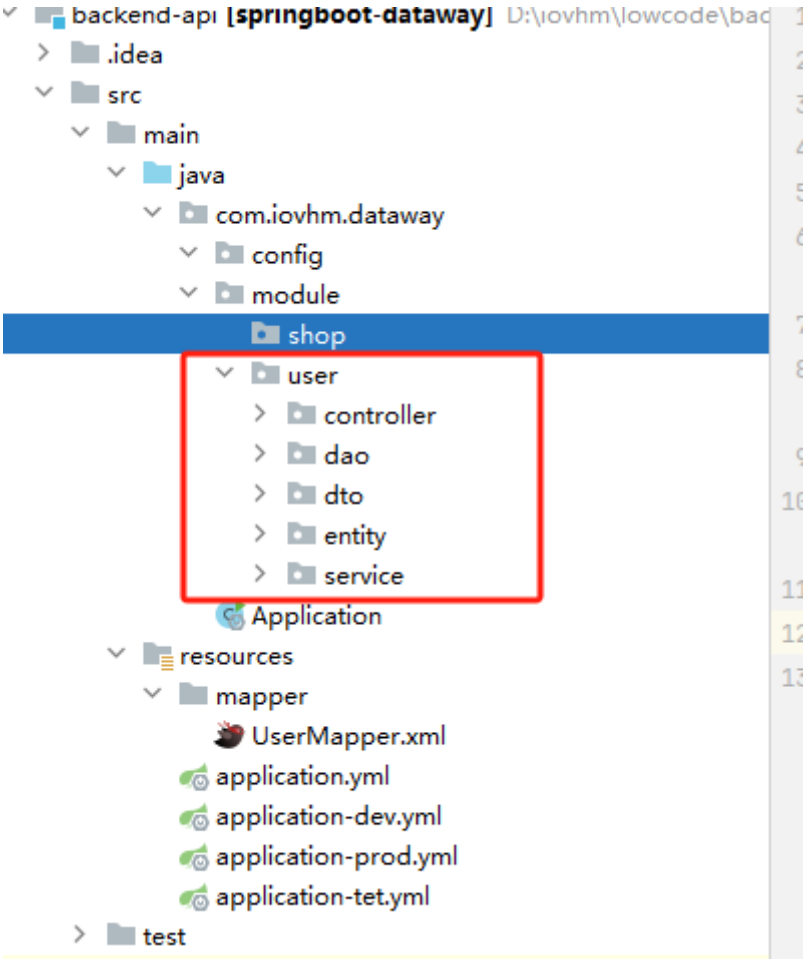
注意：如果你是按**create-springboot-project**创建的项目，需要删除配置

```
# application.yml
spring:
  autoconfigure:
    # 阻止Spring Boot自动配置数据源
    exclude: org.springframework.boot.autoconfigure.jdbc.DataSourceAutoConfiguration
```

约定与名词解释

- entity（实体），与数据表，代表一张真实的数据表
- DAO（数据访问对象），访问数据库的接口或实例，在spring-boot中，有时候有喜欢取名为Mapper
- DTO（数据传输对象），含业务领域的数据（既DTO包含entity的字段，但是应该多于entity），不包含业务逻辑，负责在系统与系统之间、分层与分层之间传递数据。

确定项目结构



□□pom□□

```
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>com.baomidou</groupId>
            <artifactId>mybatis-plus-bom</artifactId>
            <version>3.5.10</version>
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>

    <!--mysql□□-->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.30</version>
```

```
        </dependency>
        <!--mybatis□□-->
        <!--□□□□ mybatis plus □□ mybatis □□ □□□ mybatis plus □□□□□□□-->
        <dependency>
            <groupId>org.mybatis.spring.boot</groupId>
            <artifactId>mybatis-spring-boot-starter</artifactId>
            <version>3.0.3</version>
        </dependency>
        <dependency>
            <groupId>com.baomidou</groupId>
            <artifactId>mybatis-plus-spring-boot3-starter</artifactId>
            <version>3.5.10</version>
        </dependency>
```

□□□□□□□□

```
spring:
  datasource:
    url:  jdbc:mysql://127.0.0.1:33306/dataway?charset=utf8mb4&serverTimezone=Asia/Shanghai
    username:  <root>
    password:  <password>
    driver-class-name:  com.mysql.cj.jdbc.Driver
```

□□□□

名称	类型	长度	小数点	不是 null	虚拟	键	注释
id	int	11		☑	☐	🔑 1	
name	varchar	255		☑	☐		
age	int	11		☐	☐		
email	varchar	255		☐	☐		

□□Entity□□□□□□□□□□□□□

@Data @TableName("user") 注解

```
@Data
@TableName("user")
public class UserEntity {
    private Integer id;
    private String name;
    private Integer age;
    private String email;
}
```

创建DTO

由于DTO主要负责业务数据传输，DTO可以先直接继承于Entity，再根据业务需要增加字段，注意 @Data 注解

```
@Data
public class UserDTO extends UserEntity {

}
```

创建DAO

注意继承于MybatisPlus的**BaseMapper**，以及Mapper里面的Entity。DAO是用来操作数据库的类，使用的数据结构应该是Entity，虽然继承于EEntity的DTO也可以

```
@Mapper
public interface UserDao extends BaseMapper<UserEntity> {
}
```

创建Service

```
public interface UserService {
```

```
    List<UserDTO> selectList();

    List<UserDTO> selectList2();
}


@Service
public class UserServiceImpl implements UserService {

    private final UserMapper userMapper;
    private final UserDao userDao;

    public UserServiceImpl(UserMapper userMapper, UserDao userDao) {
        this.userMapper = userMapper;
        this.userDao = userDao;
    }

    @Override
    public List<UserDTO> selectList() {
        List list = userDao.selectList(null);
        return list;
    }

    @Override
    public List<UserDTO> selectList2() {
        return userMapper.selectList();
    }

}
```

□□Controller

```
// □□□□□□
@RequestMapping("/")
// □□□□□□Controller□
```

```
@RestController
@Slf4j
public class HomeController {

    private UserService userService;

    public HomeController(UserService userService) {
        this.userService = userService;
    }

    @RequestMapping("/")
    public Object hello() {
        List list = userService.selectList();
        List list2 = userService.selectList2();
        return Map.of("list", list, "list2", list2);
    }
}
```

□□Application

[`@MapperScan("com.iovhm.dataway.**.dao")`

```
@SpringBootApplication
@MapperScan("com.iovhm.dataway.**.dao")
public class Application {
    public static void main(String[] args) {
        SpringApplication.run(Application.class);
    }
}
```

□□□ #8
□ □□□ □□□ 28 □□ 2025 18:11:45
□ □□□ □□□ 29 □□ 2026 16:40:23