

01_SpringBoot#DAO#TK-Mybatis#对比介绍

一、差异

- 💊 1.1 名称不同、开发者不同
- 💊 1.2 MapperScan注解（定位DAO）
- 💊 1.3 实体注解（表、主键、列）
- 💊 1.4 三层模型架构中位置不同

二、tk-mybatis和mybatis-plus可以同时使用吗？

- 2.1 注意版本兼容性⚡
- 2.2 企业中内部封装的BaseService⚡
- 2.3 配置文件

三、如何同时使用tk-mybatis和mybatis-plus？

- 3.1 实际案例
 - 3.1.1 引入依赖
 - 3.1.2 MapperScan注解
 - 3.1.3 Mapper层接口继承
 - 3.1.4 Service层注入Mapper
 - 3.1.5 在Service层针对mybatis-plus提供的ServiceImpl进行二次封装▶

【TK-Mybatis#对比介绍】

时间：2023年04月24日 20:13:48

推荐阅读：

- <https://gitee.com/free/Mapper/wikis/Home>
- https://blog.csdn.net/zhong_jing/article/details/124962765

一、差异

- 💊 1.1 名称不同、开发者不同

- tk-mybatis
 - 通用mapper
 - TK-Mybatis 的开发者是一位名叫常德鹏的中国开发者。他在 2013 年开源了 TK-Mybatis，当时它的名称是 Mybatis-Helper。后来，为了更好地表达其作为通用 Mapper 框架的定位，常德鹏将其改名为 TK-Mybatis（TK 是Toolkit的缩写，表示工具包）。这是一款基于 Mybatis 的通用 Mapper 框架，旨在简化 Mybatis 的开发流程，提高开发效率。
- mybatis-plus
 - mybatis+
 - MybatisPlus 的开发者是一位名叫杨晓峰的中国开发者。他在 2016 年开源了 MybatisPlus，这是一款基于 Mybatis 的快速开发框架，旨在简化 Mybatis 的开发流程，提高开发效率。

💊 1.2 MapperScan注解（定位DAO）

- tk-mybatis 是自己实现的 @MapperScan
- mybatis-plus 是使用mybatis原生的 @MapperScan

💊 1.3 实体注解（表、主键、列）

- tk-mybatis 使用的是JPA（Java Persistence API）注解
 - @Table
 - @Id
 - @Column
- mybatis-plus 使用的是自己定义的一套实体注解
 - @TableName
 - @TableId
 - @ColumnName

💊 1.4 三层模型架构中位置不同

- tk-mybatis 专注于DAL层
- mybatis-plus 提供了DAL层接口与实现，同时提供了封装完善的Service层接口与实现

二、tk-mybatis 和 mybatis-plus 可以同时使用吗？

可以，但需要注意以下几点：

2.1 注意版本兼容性 ⚡

- tk-mybatis和mybatis-plus不是由一个作者开发，不同的版本中会依赖不同的mybatis版本
 - mybatis大版本迭代会出现明显的兼容性问题（请确保mybatis版本一致或使用各自的最新版本）

2.2 企业中内部封装的BaseService ⚡

- 部分企业使用 `tk-mybatis`，没有封装BaseService层或封装的BaseService层不完全；
 - 他们会拓展 `tk-mybatis` 的Mapper以支持批量新增与批量更新等操作。
- 部分企业使用 `mybatis-plus`，在其提供的 `ServiceImpl` 基础上封装了内部的BaseService；
 - 这是推荐的做法。

2.3 配置文件

使用mybatis默认的配置文件的属性即可，但是👇

- ⚠️mybatis-plus对于mapperXML文件的定位无法使用mybatis为前缀的配置：
 - 所以推荐将 `mapperXML` 文件 放到 `/resources/<DAO包名层级>/` 目录下，确保编译后 `mapperXML` 文件 会和对应 `Mapper`对象 处于同一目录中。
- 日志输出，`mybatis-plus` 和 `tk-mybatis` 对于下方配置均生效。

```

1  mybatis:
2  mapper-locations: ["classpath*:mapper/**/*.xml"]
3  configuration:
4      ### 关闭日志打印org.apache.ibatis.logging.noLogging.NoLoggingImpl
5      log-impl: org.apache.ibatis.logging.stdout.StdoutImpl
6  pagehelper:
7      helper-dialect: mysql
8      reasonable: true
9      ### 改参数决定分页时能否拿到total
10     default-count: true
11
12  spring:
13     datasource:
14         # type: com.alibaba.druid.pool.DruidDataSource
15         driver-class-name: com.mysql.cj.jdbc.Driver
16         url: jdbc:mysql://192.168.204.101:23306/dev_test?autoReconnect=true&useServerPreparedStmts=true&cachePrepStmts=true&rewriteBatchedStatements=true&allowMultiQueries=true&characterEncoding=utf8&serverTimezone=Asia/Shanghai&allowPublicKeyRetrieval=true
17         username: root
18         password: Mysql12345
19         type: com.zaxxer.hikari.HikariDataSource
20         hikari:
21             ### 最小空闲连接，默认值10，小于0或大于maximum-pool-size，都会重置为maximum-pool-size
22             minimum-idle: 10
23             ### 最大连接数，小于等于0会被重置为默认值10；大于零小于1会被重置为minimum-idle
24             的值
25             maximum-pool-size: 20
26             ### 连接池名称，默认HikariPool-1
27             pool-name: HikariPool-Abc-1
28             ### 连接超时时间: 毫秒，小于250毫秒，会被重置为默认值30秒
29             connection-timeout: 10000
30             ### 连接测试查询
31             connection-test-query: SELECT 1
32             ### 空闲连接超时时间，默认值600000（10分钟），大于等于max-lifetime且max-lifetime>0，会被重置为0；不等于0且小于10秒，会被重置为10秒。
33             idle-timeout: 600000
34             ### 连接最大存活时间. 不等于0且小于30秒，会被重置为默认值30分钟. 设置应该比mysql
35             设置的超时时间短；单位ms
36             max-lifetime: 1800000
37             ### 自动提交从池中返回的连接，默认值为true
38             auto-commit: true

```

三、如何同时使用 `tk-mybatis` 和 `mybatis-plus` ?

首先明确：前者在Mapper层进行抽象，后者在Mapper层和Service层进行抽象。

- 我们可以在Mapper层同时使用二者提供的接口，在Service层上基于 `mybatis-plus` 进行二次封装；
- 同时可以在 `tk-mybatis` 的Mapper层进行拓展。

3.1 实际案例

`SpringBoot` + `tk-mybatis` + `mybatis-plus`

3.1.1 引入依赖

注意事项（lombok和mapstruct）：

```
<dependency>
  <groupId>org.mapstruct</groupId>
  <artifactId>mapstruct</artifactId>
  <version>1.5.3.Final</version>
</dependency>
<dependency>
  <groupId>org.mapstruct</groupId>
  <artifactId>mapstruct-processor</artifactId>
  <version>1.5.3.Final</version>
</dependency>
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
</dependency>
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok-mapstruct-binding</artifactId>
  <version>0.2.0</version>
</dependency>
```

Lombok 1.8.16+版本
请添加该依赖，
否则mapstruct概率失效！

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.
  w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://ma
  ven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5   <parent>
6     <groupId>org.springframework.boot</groupId>
7     <artifactId>spring-boot-starter-parent</artifactId>
8     <version>2.7.11</version>
9     <relativePath/> <!-- lookup parent from repository -->
10  </parent>
11  <groupId>com.example</groupId>
12  <artifactId>spring-boot-mpAndTk</artifactId>
13  <version>0.0.1-SNAPSHOT</version>
14  <name>spring-boot-mpAndTk</name>
15  <description>spring-boot-mpAndTk</description>
16  <properties>
17    <java.version>1.8</java.version>
18  </properties>
19
20  <dependencies>
21    <!-- web -->
22    <dependency>
23      <groupId>org.springframework.boot</groupId>
24      <artifactId>spring-boot-starter-web</artifactId>
25    </dependency>
26
27    <!-- mybatis-plus -->
28    <!-- tk-mybatis -->
29    <!-- pagehelper -->
30    <!-- mysql-connector-java -->
31    <dependency>
32      <groupId>com.baomidou</groupId>
33      <artifactId>mybatis-plus-boot-starter</artifactId>
34      <version>3.5.3.1</version>
35    </dependency>
36    <dependency>
37      <groupId>tk.mybatis</groupId>
38      <artifactId>mapper-spring-boot-starter</artifactId>
39      <version>4.2.2</version>
40    </dependency>
41    <dependency>
42      <groupId>com.github.pagehelper</groupId>
43      <artifactId>pagehelper-spring-boot-starter</artifactId>
```

```

44         <version>1.4.6</version>
45     </dependency>
46 <dependency>
47     <groupId>mysql</groupId>
48     <artifactId>mysql-connector-java</artifactId>
49     <version>8.0.31</version>
50 </dependency>
51
52 <!-- commons-lang3 -->
53 <!-- fastjson2 -->
54 <!-- lombok -->
55 <!-- mapstruct -->
56 <dependency>
57     <groupId>org.apache.commons</groupId>
58     <artifactId>commons-lang3</artifactId>
59     <version>3.12.0</version>
60 </dependency>
61 <dependency>
62     <groupId>com.alibaba.fastjson2</groupId>
63     <artifactId>fastjson2</artifactId>
64     <version>2.0.27</version>
65 </dependency>
66 <dependency>
67     <groupId>org.projectlombok</groupId>
68     <artifactId>lombok</artifactId>
69     <optional>true</optional>
70 </dependency>
71 <dependency>
72     <groupId>org.mapstruct</groupId>
73     <artifactId>mapstruct</artifactId>
74     <version>1.5.3.Final</version>
75 </dependency>
76 <dependency>
77     <groupId>org.mapstruct</groupId>
78     <artifactId>mapstruct-processor</artifactId>
79     <version>1.5.3.Final</version>
80 </dependency>
81
82 <!-- test -->
83 <dependency>
84     <groupId>org.springframework.boot</groupId>
85     <artifactId>spring-boot-starter-test</artifactId>
86     <scope>test</scope>
87 </dependency>
88 </dependencies>
89
90 <build>
91     <plugins>

```

```

92         <plugin>
93             <groupId>org.springframework.boot</groupId>
94             <artifactId>spring-boot-maven-plugin</artifactId>
95             <configuration>
96                 <excludes>
97                     <exclude>
98                         <groupId>org.projectlombok</groupId>
99                         <artifactId>lombok</artifactId>
100                     </exclude>
101                 </excludes>
102             </configuration>
103         </plugin>
104     </plugins>
105 </build>
106
107 </project>

```

3.1.2 MapperScan 注解

添加在启动类上，同时配置二者的 `@MapperScan`

```

Java | 复制代码

1  import org.mybatis.spring.annotation.MapperScan;
2  import org.springframework.boot.SpringApplication;
3  import org.springframework.boot.autoconfigure.SpringBootApplication;
4
5  @SpringBootApplication
6  @MapperScan({"com.example.springbootmpandtk.dal.persistence"})
7  @tk.mybatis.spring.annotation.MapperScan({"com.example.springbootmpandtk.dal.persistence"})
8  public class SpringBootMpAndTkApplication {
9
10     public static void main(String[] args) {
11         SpringApplication.run(SpringBootMpAndTkApplication.class, args);
12     }
13
14 }

```

3.1.3 Mapper层接口继承

同时继承 `mybatis-plus` 的 `BaseMapper<T>` 和 `tk-mybatis` 的 `Mapper<T>`


```

1  import com.baomidou.mybatisplus.core.mapper.BaseMapper;
2  import com.example.springbootmpandtk.dal.entity.User;
3  import tk.mybatis.mapper.common.Mapper;
4
5  public interface UserMapper extends BaseMapper<User>, Mapper<User> {
6  }

```

3.1.4 Service层注入Mapper

此时，你可以使用 `tk-mybatis` 和 `mybatis-plus` 同时提供的多种API实现

`@Service`
`@RequiredArgsConstructor` Service层，同时使用tk-mybatis、mybatis-plus实现的接口

```

public class UserServiceImpl2 implements IUserService {

    private final UserConverter userConverter;

    private final UserMapper userMapper;

    1 个用法
    @Override
    public BaseResponse<List<UserDTO>> getUser(UserRequest userRequest) {
        BaseResponse<List<UserDTO>> response = new BaseResponse<>();
        PageInfo pageInfo = userRequest.getPageInfo();
        if (pageInfo == null) {
            pageInfo = new PageInfo();
        }
        // todo
        userMapper.delete
    }
}

```

`tk-mybatis`

<code>delete(Wrapper<User> queryWrapper)</code>	int
<code>delete(User record)</code>	int
<code>deleteById(User entity)</code>	int
<code>deleteByMap(Map<String, Object> columnMap)</code>	int
<code>deleteById(Serializable id)</code>	int
<code>deleteByExample(Object example)</code>	int
<code>deleteBatchIds(Collection<?> idList)</code>	int
<code>deleteByPrimaryKey(Object key)</code>	int

按 Ctrl+., 选择所选 (或第一个) 建议, 然后插入点 下一提示

3.1.5 在Service层针对 `mybatis-plus` 提供的 `ServiceImpl` 进行二次封装

```
1 import com.baomidou.mybatisplus.core.mapper.BaseMapper;
2 import com.baomidou.mybatisplus.extension.service.IService;
3 import com.baomidou.mybatisplus.extension.service.impl.ServiceImpl;
4
5 public class BaseService<M extends BaseMapper<T>, T> extends ServiceImpl<M,
  T> implements IService<T> {
6
7 }
```

继承二次封装的BaseService

```
@Service
@RequiredArgsConstructor
public class UserServiceImpl1 extends BaseService<UserMapper, User> implements IUserService {

    private final UserConverter userConverter;

    1 个用法
    @Override
    public BaseResponse<List<UserDTO>> getUser(UserRequest userRequest) {
        PageInfo pageInfo = userRequest.getPageInfo();
        if(pageInfo==null) {
            pageInfo = new PageInfo();
        }
    }
}
```

END