02_SpringBoot#DAO#TK-Mybatis#使用

> 【TK-Mybatis#使用】

创建时间：2023年04月11日 23:05:12

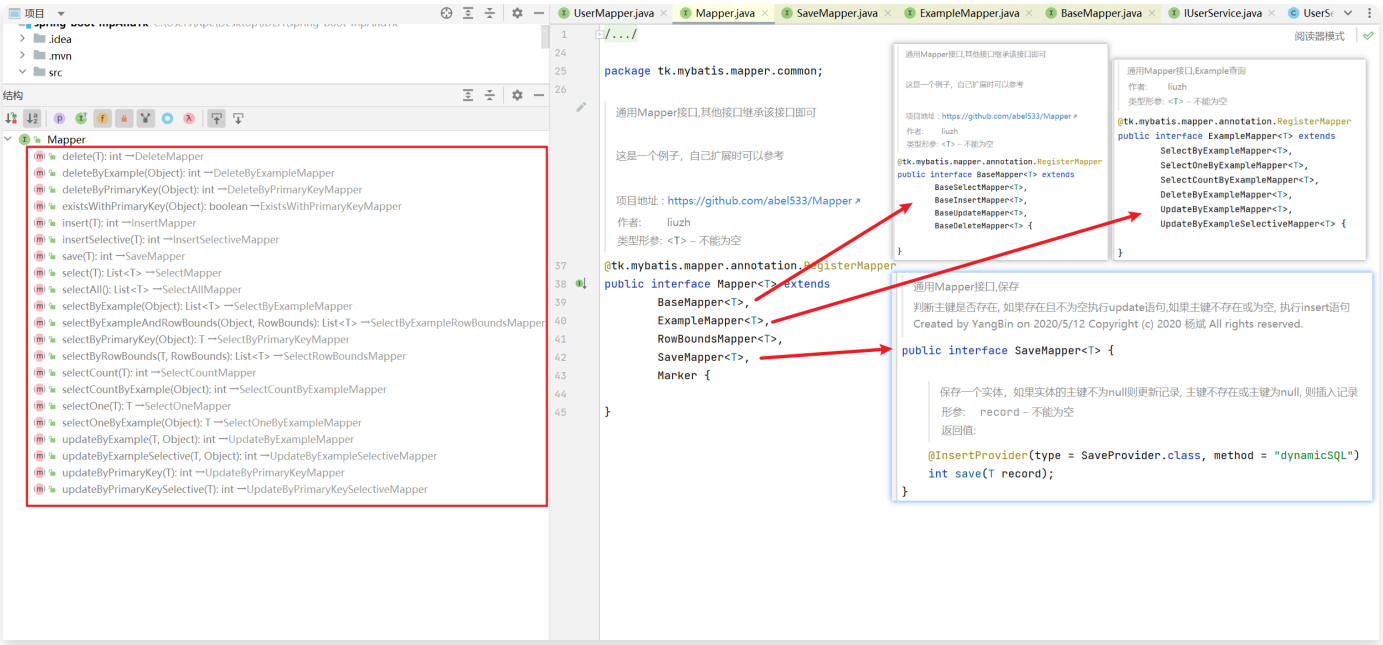
更新时间：2023年08月09日 20:16:26

文档：

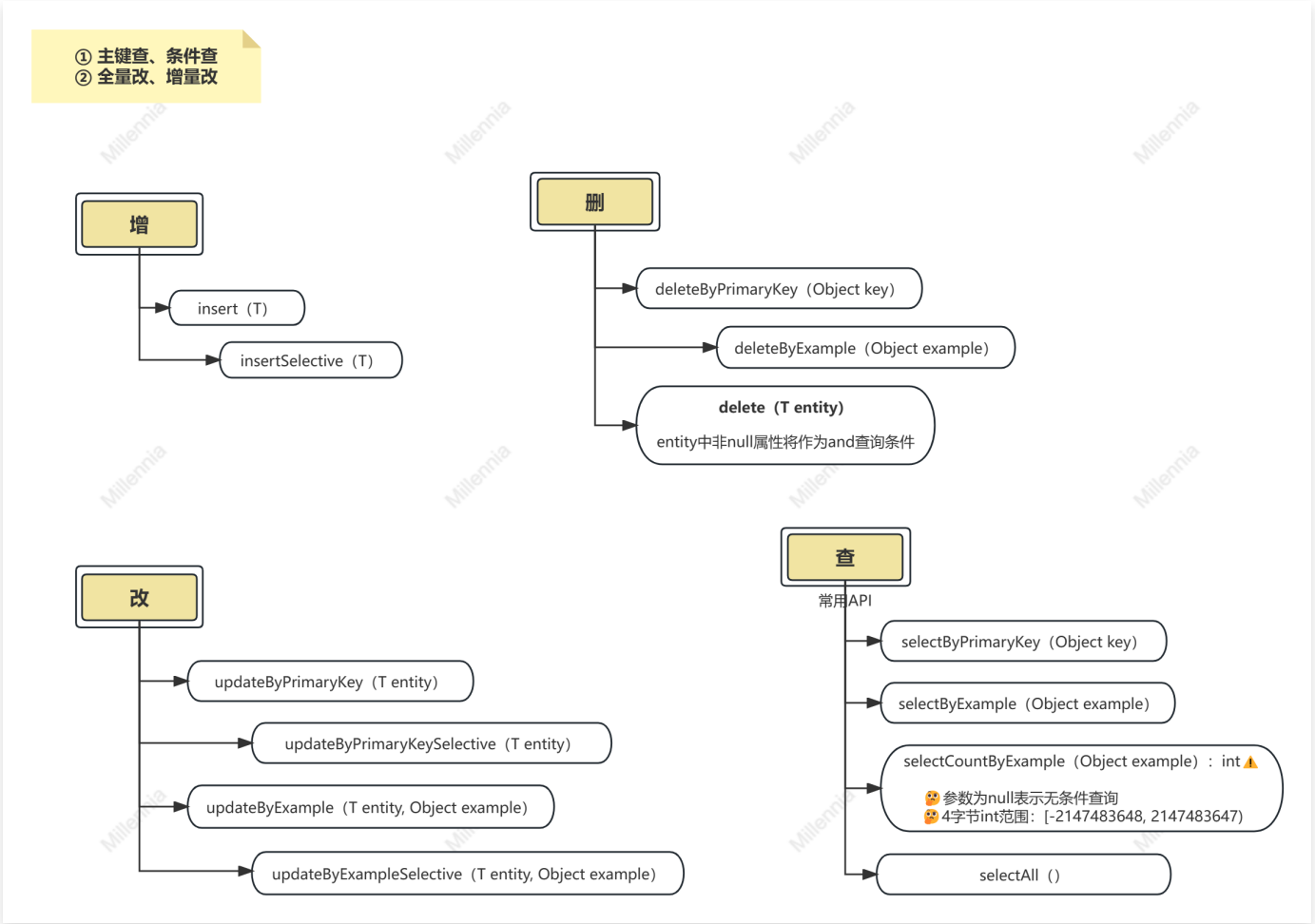
- https://gitee.com/free/Mapper/wikis/Home

一、Mapper接口

> 基本的增删改查都已经提供了API
>
> - `2+3+4+N(4)`



tk–mybatis中Mapper接口的继承关系



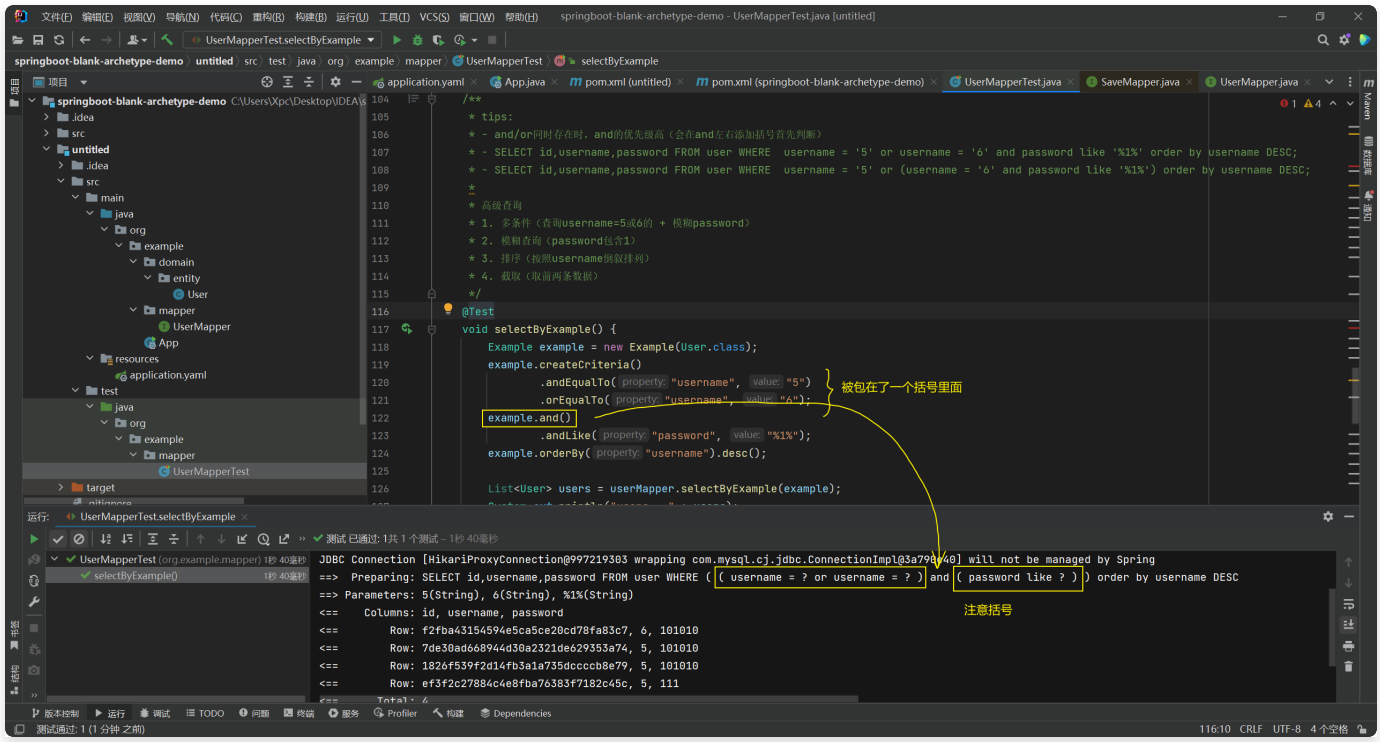
tk–mybatis常用增删改查API

二、使用 `Example` + `Criteria` 进行复杂查询

> Example
>
> - 例子；榜样
>
> Criteria
>
> - 标准；条件

2.1 常用的 `Example.Criteria`

2.1.1 使用示例



2.1.2 相等、模糊、IN、大小

```
criteria.andEqualTo("property5", "10");
criteria.andLike("property7", "%Hello%");
criteria.andIn("property6", Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9));

criteria.andGreaterThan("property1", "10");
criteria.andGreaterThanOrEqualTo("property2", "10");
criteria.andLessThan("property3", "10");
criteria.andLessThanOrEqualTo("property4", "10");
```

常用

相等、模糊、IN、大小

Java | 复制代码

```java
/**
 * 示例：多条件查询多条数据⬇️
 * 1.注意日志格式、日志锚点的位置
 * 2.注意常用的Criteria
 * 3.注意Converter的使用
 * 4.注意分页
 * 5.注意需要先查询所有记录数，放入total
 */
@Slf4j
@Service
@RequiredArgsConstructor
public class UserServiceImpl2 implements IUserService {

    private final UserConverter userConverter;
    private final UserMapper userMapper;

    @Override
    public BaseResponse<List<UserDTO>> queryUser(UserRequest userRequest)
{
        // 你需要在你的项目中统一日志输出格式，这将加速异常定位
        log.info(">>>>>>>>>|queryUser|start|request:{}", JSONObject.toJSONS
tring(userRequest));
        BaseResponse<List<UserDTO>> response = new BaseResponse<>();
        List<User> users = new ArrayList<>();
        Example example = new Example(User.class);
        Example.Criteria criteria = example.createCriteria();
        PageInfo pageInfo = userRequest.getPageInfo();
        if (pageInfo == null) {
            pageInfo = new PageInfo();
        }

        // 这里有一个问题很明显：属性参数是"魔法字符串"，这种字符串不推荐出现在代码
中；而在mybatis-plus中，属性参数被替换为了方法引用。
        criteria.andEqualTo("property5", "10");
        criteria.andLike("property7", "%Hello%");
        criteria.andIn("property6", Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8,
9));

        criteria.andGreaterThan("property1", "10");
        criteria.andGreaterThanOrEqualTo("property2", "10");
        criteria.andLessThan("property3", "10");
        criteria.andLessThanOrEqualTo("property4", "10");

        long count;
        try {
```

```
            count = userMapper.selectCountByExample(example);
            if (count < 1) {
                response.setInnerCode("000002");
                response.setInnerMessage("失败");
                response.setResult(new ArrayList<>());
                response.setTotal(0L);
                return response;
            }
            // 开启分页
            PageHelper.startPage(pageInfo.getPageNum(), pageInfo.getPageSize());
            users = userMapper.selectByExample(example);
        } catch (Exception e) {
            // 1.格式化日志
            // 2.当Service层有问题，通过自定义业务异常向Controller层抛出
            log.info(">>>>>>>>>|queryUser|failed|exception:", e);
            throw new BizException("500001", "系统错误");
        }

        response.setInnerCode("000001");
        response.setInnerMessage("成功");
        response.setResult(userConverter.UserToDTO(users));
        response.setTotal(count);

        log.info(">>>>>>>>>>>queryUser|end|success|<<<<<<<<");
        return response;
    }
}
```

2.1.3 LIMIT

> 官网没有找到LIMIT的相关API，使用分页帮助器解决

Java | 复制代码

```java
    /**
     * tips:
     * - and/or同时存在时，and的优先级高（会在and左右添加括号首先判断）
     * - SELECT id,username,password FROM user WHERE  username = '5' or username = '6' and password like '%1%' order by username DESC;
     * - SELECT id,username,password FROM user WHERE  username = '5' or (username = '6' and password like '%1%') order by username DESC;
     * <p>
     * 高级查询
     * 1. 多条件（查询username=5或6的 + 模糊password）
     * 2. 模糊查询（password包含1）
     * 3. 排序（按照username倒叙排列）
     * 4. 截取（取前两条数据）
     */
    @Test
    void selectByExample() {
        // 第一页PageNum，每页两个PageSize
        //   -> 推荐使用
        PageHelper.startPage(1, 2);

        Example example = new Example(User.class);
        example.setDistinct(true);
        example.createCriteria()
                .andEqualTo("username", "5")
                .orEqualTo("username", "6");
        example.and()
                .andLike("password", "%1%");
        example.orderBy("username").desc();

        List<User> users = userMapper.selectByExample(example);

        PageInfo<User> userPageInfo = new PageInfo<>(users);
        //  【总数、总页数】、【当前页、页大小】
        System.out.println(userPageInfo.getTotal());
        System.out.println(userPageInfo.getPages());
        System.out.println(userPageInfo.getPageNum());
        System.out.println(userPageInfo.getPageSize());

        List<User> list = userPageInfo.getList();
        System.out.println(users == list);  // true
        System.out.println(users.equals(list));  // true

        System.out.println("list = " + list);
        System.out.println("users = " + users);

```

```
    }
```

依赖：tk-mybatis、pagehelper、mysql-connector　XML　复制代码

```xml
        <dependency>
            <groupId>tk.mybatis</groupId>
            <artifactId>mapper-spring-boot-starter</artifactId>
            <version>4.2.2</version>
        </dependency>
        <dependency>
            <groupId>com.github.pagehelper</groupId>
            <artifactId>pagehelper-spring-boot-starter</artifactId>
            <version>1.4.6</version>
        </dependency>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.31</version>
        </dependency>
```

配置数据库、开启mybatis日志、配置分页帮助器 YAML 复制代码

```yaml
spring:
  datasource:
    # type: com.alibaba.druid.pool.DruidDataSource
    driver-class-name: com.mysql.cj.jdbc.Driver
    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
    username: root
    password: Mysql12345
    type: com.zaxxer.hikari.HikariDataSource
    hikari:
      ### 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
      minimum-idle: 10
      ### 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的值
      maximum-pool-size: 20
      ### 连接池名称，默认HikariPool-1
      pool-name: HikariPool-Abc-1
      ### 连接超时时间:毫秒，小于250毫秒，会被重置为默认值30秒
      connection-timeout: 10000
      ### 连接测试查询
      connection-test-query: SELECT 1
      ### 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。
      idle-timeout: 600000
      ### 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设置的超时时间短；单位ms
      max-lifetime: 1800000
      ### 自动提交从池中返回的连接，默认值为true
      auto-commit: true


mybatis:
  mapper-locations: ["classpath*:/mapper/**/*.xml"]
  configuration:
    ### 关闭日志打印org.apache.ibatis.logging.nologging.NoLoggingImpl
    log-impl: org.apache.ibatis.logging.stdout.StdOutImpl
pagehelper:
  helper-dialect: mysql
  reasonable: true
  ### 改参数决定分页时能否拿到total
  default-count: true
```

2.1.4 返回指定字段

> 🌵tk-mapper如何使用criteria返回结果中的一个字段?

在使用 tk-mapper 的 Criteria 进行查询时，可以使用 `selectProperties()` 方法来指定查询返回的字段，例如：

Java | 复制代码

```java
import tk.mybatis.mapper.entity.Example.Criteria;

Criteria criteria = example.createCriteria();
criteria.andEqualTo("id", 1);
example.selectProperties("name");
List<User> userList = userMapper.selectByExample(example);
```

上述代码中，`example` 是一个 `tk.mybatis.mapper.entity.Example` 对象，通过 `createCriteria()` 方法创建一个 Criteria 对象，然后使用 `andEqualTo()` 方法设置查询条件。在查询结果中，我们只需要返回 `name` 字段的值，因此使用 `selectProperties()` 方法指定返回的字段。

如果想要在查询结果中返回多个字段，可以在 `selectProperties()` 方法中指定多个字段名，用逗号隔开即可。例如：

Java | 复制代码

```java
example.selectProperties("id", "name", "age");
```

这样就可以返回 id、name 和 age 三个字段的值了。

> 🌵selectProperties中指定的字符串是实体类中的属性名还是数据库中的字段名?

`selectProperties()` 方法中指定的字符串应该是实体类中的属性名，而不是数据库中的字段名。如果实体类中的属性名和数据库中的字段名不相同，可以使用 `@Column` 注解来指定它们之间的对应关系。例如：

```java
@Table(name = "user")
public class User {
    @Id
    private Long id;

    @Column(name = "user_name")
    private String userName;

    private Integer age;

    // 省略 getter 和 setter 方法
}
```

在上述代码中，`@Column(name = "user_name")` 注解指定了 `userName` 属性对应的数据库字段名为 `user_name`。在使用 `selectProperties()` 方法查询时，应该使用实体类中的属性名来指定返回的字段，而不是数据库中的字段名。例如：

```java
example.selectProperties("id", "userName", "age");
```

这样就可以返回 id、user_name 和 age 三个字段的值了。

2.1.5 在查询条件中指定括号

- 查询【name=A且age=27】或【name=B且age=37】的记录

```
c.e.s.d.m.UserMapper.selectByExample     : ==>  Preparing: SELECT id,name,age,create_time,update_time,stop_time,memo FROM user WHERE ( ( name = ? and age = ? ) or ( name = ? and age = ? ) )
c.e.s.d.m.UserMapper.selectByExample     : ==> Parameters: A(String), 27(Integer), B(String), 37(Integer)
c.e.s.d.m.UserMapper.selectByExample     : <==      Total: 2
```

- 查询【name=A且age=27】或【name=B且age=37】的记录

```java
@GetMapping("/demo07")
public void demo07() {
    Example userEx = new Example(User.class);
    userEx.createCriteria()
        .andEqualTo(User.Fields.name, "A").andEqualTo("age", 27);
    userEx.or()
        .andEqualTo(User.Fields.name, "B").andEqualTo("age", 37);
    List<User> users = userMapper.selectByExample(userEx);
    JSONObject.toJSONString(users, JSONWriter.Feature.PrettyFormat);
}
```

2.2 ✨ TK-Mybatis 使用 mapperXML

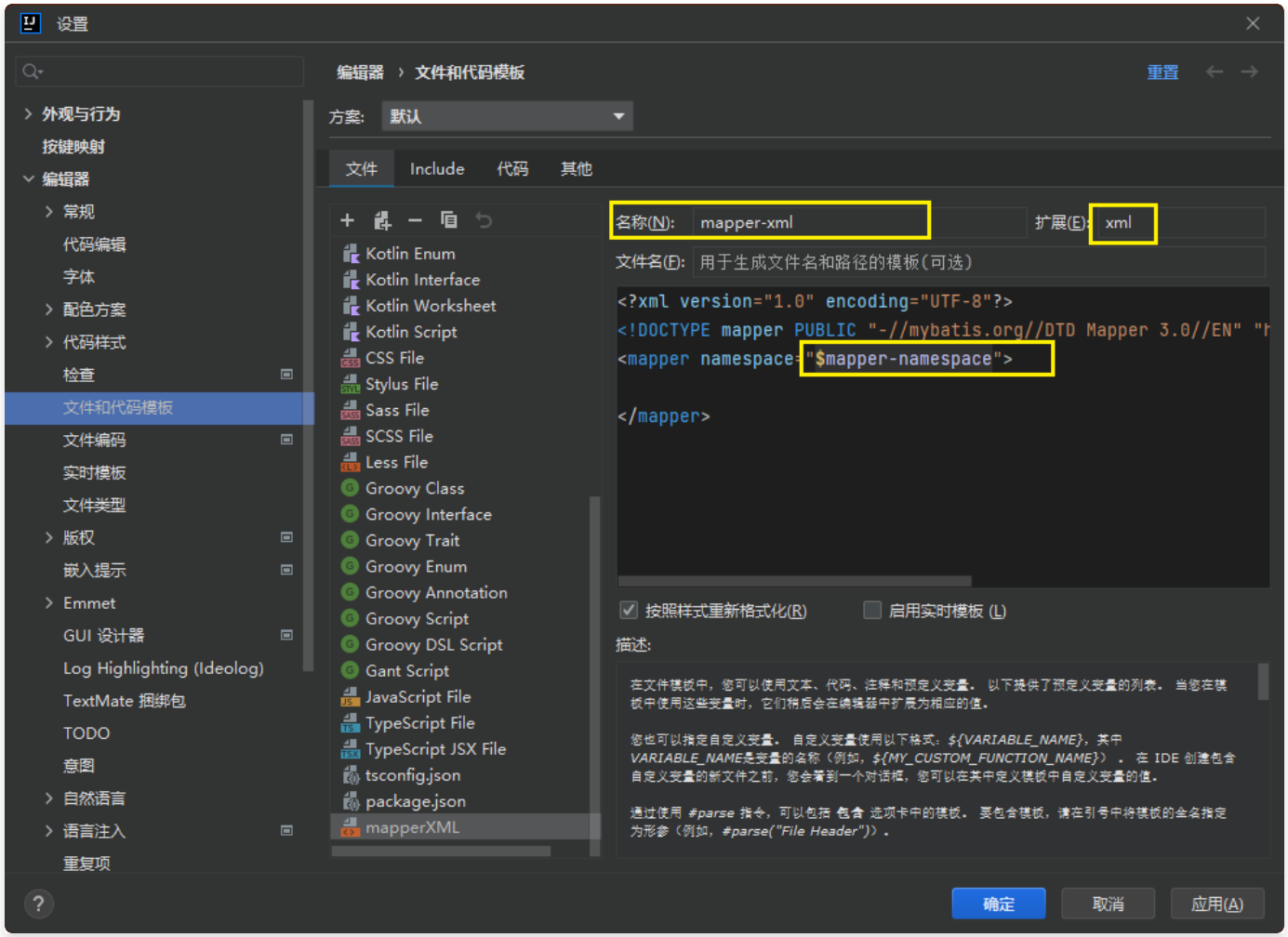
> TK-Mybatis 提供了常用的CRUD操作，如果部分需求存在一定复杂性，请手动编写MapperXML

```
5 个用法
public interface UserMapper extends Mapper<User> {
    2 个用法
    Integer selectCountX(@Param("usernameList") List<String> usernameArray);
}
```

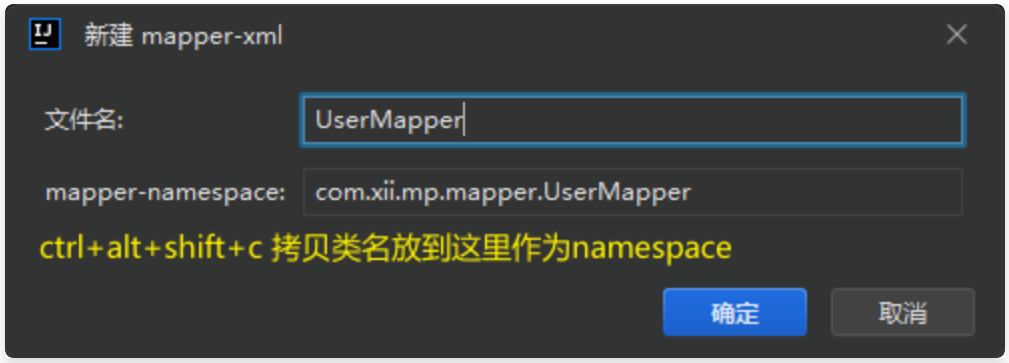
userMapper.xml　　XML　复制代码

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="org.example.mapper.UserMapper">


    <select id="selectCountX" resultType="java.lang.Integer">
        SELECT COUNT(*)
            FROM `user`
        WHERE username IN
            <foreach collection="usernameList" item="item" separator="," open="(" close=")">
                #{item}
            </foreach>
        LIMIT 2;
    </select>

</mapper>
```

2.3 ✨IDEA配置生成MapperXML的模板



IDEA配置mapper-xml文件模板




```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.xii.mp.mapper.UserMapper">

</mapper>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="$mapper-namespace">

</mapper>
```

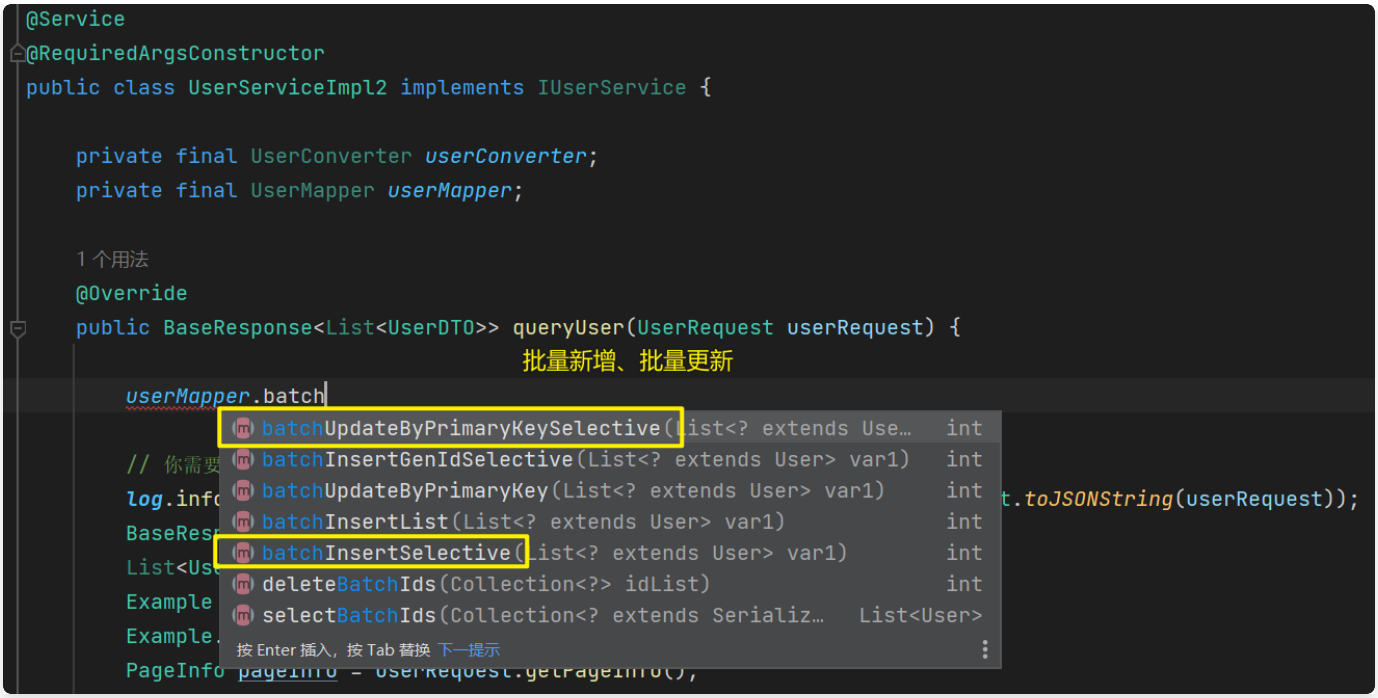
2.4 ✨显示SQL

```yaml
logging:
  level:
    com.example.springbootdateformat.dal.mapper.*: debug
```

三、如何拓展 `TK-Mybatis`

> 基本的Mapper没有提供批量新增、批量更新的API（继承Mapper接口）
>
> - ➕ 批量添加API
> - ➕ 批量更新API

3.1 使用场景



3.2 具体封装

📎tk-mybatis拓展（支持MySQL批量插入与更新）.zip

四、使用 `spring-boot-test` 测试 `MapperAPI`

springboot测试依赖 + junit　　XML | 复制代码

```xml
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>
<dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <scope>test</scope>
</dependency>
```

Java | 复制代码

```java
package org.example.mapper;

import com.github.pagehelper.PageHelper;
import com.github.pagehelper.PageInfo;
import org.example.App;
import org.example.domain.entity.User;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.test.annotation.Rollback;
import tk.mybatis.mapper.entity.Example;

import java.util.ArrayList;
import java.util.List;

@SpringBootTest(classes = App.class)
@Rollback(false)
class UserMapperTest {

    @Autowired
    private UserMapper userMapper;

    /**
     * 主键存在性检测
     */
    @Test
    void existsWithPrimaryKey() {
        boolean b = userMapper.existsWithPrimaryKey("123123");
    }

    @Test
    void insert() {
    }

    @Test
    void insertSelective() {

    }

    @Test
    void delete() {
        int delete = userMapper.delete(new User().setUsername("1"));
        System.out.println("delete = " + delete);
    }

```

```
46      @Test
47      void deleteByPrimaryKey() {
48          int i = userMapper.deleteByPrimaryKey("11");
49          System.out.println("i = " + i);
50      }
51
52      @Test
53      void deleteByExample() {
54      }
55
56      @Test
57      void updateByPrimaryKey() {
58      }
59
60      @Test
61      void updateByPrimaryKeySelective() {
62      }
63
64      @Test
65      void updateByExample() {
66      }
67
68      @Test
69      void updateByExampleSelective() {
70      }
71
72      @Test
73      void selectAll() {
74          List<User> users = userMapper.selectAll();
75          System.out.println(users.size());
76      }
77
78      /**
79       * 根据主键查询（@Id注解修饰的属性）
80       */
81      @Test
82      void selectByPrimaryKey() {
83          User user = userMapper.selectByPrimaryKey("6");
84          System.out.println("user = " + user);
85          User user1 = userMapper.selectByPrimaryKey("e90e4d85dac142e287b07
    d80991460b9");
86          System.out.println("user1 = " + user1);
87      }
88
89      /**
90       * 查询数量
91       * - tk提供了基本的CRUD，如果需求存在一定复杂性，请手动编写mapperXML
92       * ⬇⬇⬇
```

```
     */
    @Test
    void selectCount() {
        int i = userMapper.selectCount(new User().setUsername("6"));
        System.out.println("i = " + i);
    }

    /**
     * 查询数量(自定义MapperXML)
     */
    @Test
    void selectCountX() {
        ArrayList<String> usernames = new ArrayList<>();
        usernames.add("5");
        usernames.add("6");
        Integer integer = userMapper.selectCountX(usernames);
        System.out.println("integer = " + integer);
    }

    /**
     * 所有不为null的属性，都将作为where条件用and连接
     */
    @Test
    void select() {
        List<User> select = userMapper.select(new User().setUsername("6")
);
        System.out.println("select = " + select);
        List<User> select1 = userMapper.select(new User().setUsername("6"
).setPassword("777"));
        System.out.println("select1 = " + select1);
    }

    /**
     * 查询一个数据，当有多个数据时会报错
     */
    @Test
    void selectOne() {
        // 大于一个数据会报错
        User user = userMapper.selectOne(new User().setUsername("6"));
        System.out.println("user = " + user);
    }

    /**
     * tips:
     * - and/or同时存在时，and的优先级高（会在and左右添加括号首先判断）
     * - SELECT id,username,password FROM user WHERE  username = '5' or u
sername = '6' and password like '%1%' order by username DESC;
```

```
138         * - SELECT id,username,password FROM user WHERE  username = '5' or
    (username = '6' and password like '%1%') order by username DESC;
139         * <p>
140         * 高级查询
141         * 1. 多条件（查询username=5或6的 + 模糊password）
142         * 2. 模糊查询（password包含1）
143         * 3. 排序（按照username倒叙排列）1
144         * 4. 截取（取前两条数据）
145         */
146        @Test
147        void selectByExample() {
148            PageHelper.startPage(1, 2);
149
150            Example example = new Example(User.class);
151            example.setDistinct(true);
152            example.createCriteria()
153                    .andEqualTo("username", "5")
154                    .orEqualTo("username", "6");
155            example.and()
156                    .andLike("password", "%1%");
157            example.orderBy("username").desc();
158
159            List<User> users = userMapper.selectByExample(example);
160
161            PageInfo<User> userPageInfo = new PageInfo<>(users);
162            //  【总数、总页数】、【当前页、页大小】
163            System.out.println(userPageInfo.getTotal());
164            System.out.println(userPageInfo.getPages());
165            System.out.println(userPageInfo.getPageNum());
166            System.out.println(userPageInfo.getPageSize());
167
168            List<User> list = userPageInfo.getList();
169            System.out.println(users == list);
170            System.out.println(users.equals(list));
171            System.out.println("list = " + list);
172
173            System.out.println("users = " + users);
174
175        }
176
177        @Test
178        void selectCountByExample() {
179        }
180
181        @Test
182        void selectOneByExample() {
183        }
184
```

```
185     @Test
186     void selectByExampleAndRowBounds() {
187     }
188
189     @Test
190     void selectByRowBounds() {
191     }
```