

# MyBatis Plus

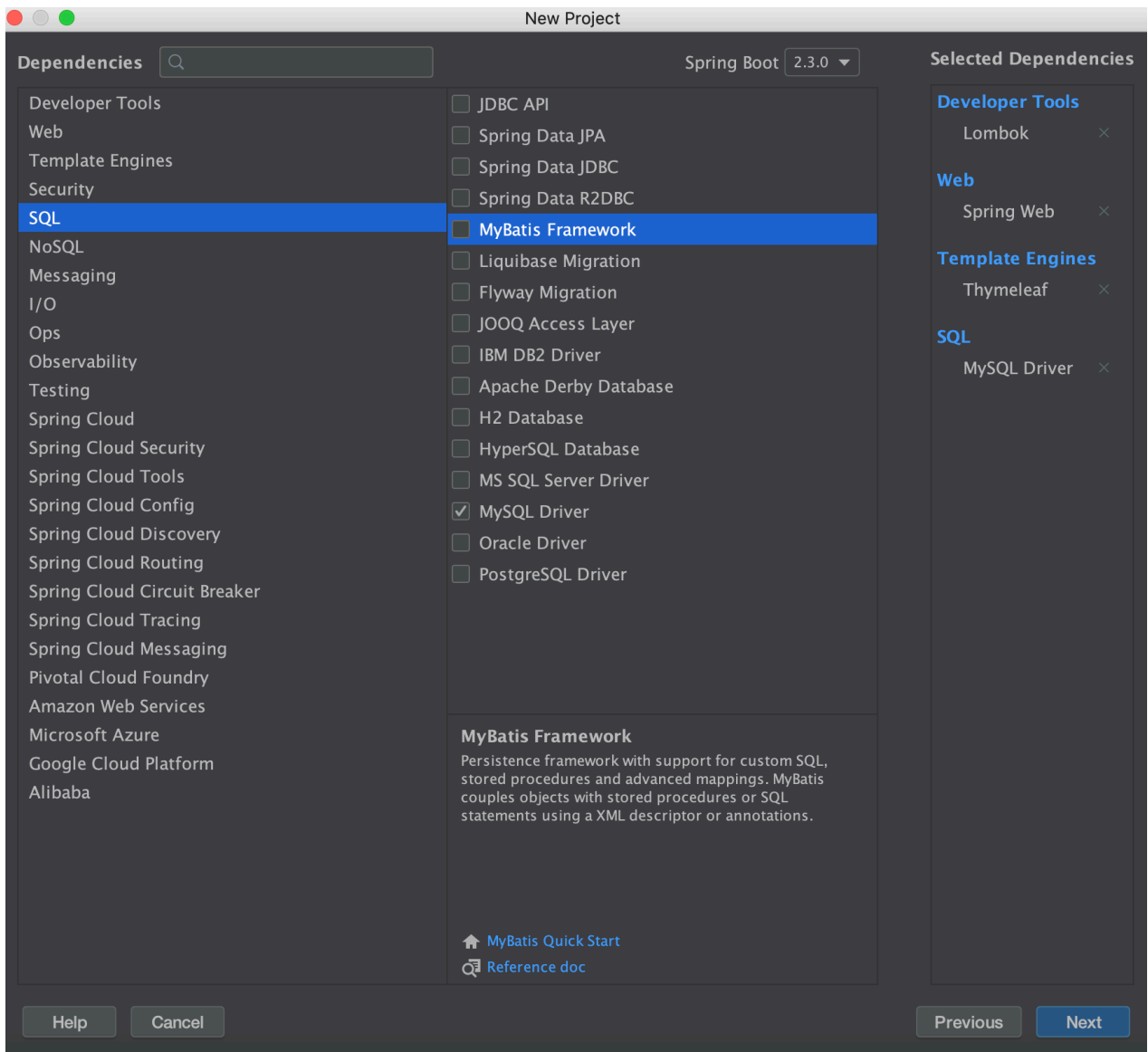
国产的开源框架，基于 MyBatis

核心功能就是简化 MyBatis 的开发，提高效率。

## MyBatis Plus 快速上手

Spring Boot(2.3.0) + MyBatis Plus（国产的开源框架，并没有接入到 Spring 官方孵化器中）

### 1、创建 Maven 工程



## 2、pom.xml 引入 MyBatis Plus 的依赖

```
<dependency>
    <groupId>com.baomidou</groupId>
    <artifactId>mybatis-plus-boot-
starter</artifactId>
    <version>3.3.1.tmp</version>
</dependency>
```

## 3、创建实体类

```
package com.southwind.mybatisplus.entity;

import lombok.Data;

@Data
public class User {
    private Integer id;
    private String name;
    private Integer age;
}
```

#### 4、创建 Mapper 接口

```
package com.southwind.mybatisplus.mapper;

import
com.baomidou.mybatisplus.core.mapper.BaseMapper;
import
com.southwind.mybatisplus.entity.User;

public interface UserMapper extends
BaseMapper<User> {

}
```

#### 5、application.yml

```
spring:
  datasource:
    driver-class-name:
com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/db?
useUnicode=true&characterEncoding=UTF-8
    username: root
    password: root
mybatis-plus:
  configuration:
    log-impl:
org.apache.ibatis.logging.stdout.StdoutImpl
```

6、启动类需要添加 @MapperScan("mapper所在的包"), 否则无法加载 Mppaer bean。

```
package com.southwind.mybatisplus;

import
org.mybatis.spring.annotation.MapperScan;
import
org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.Spr
ingBootApplication;

@SpringBootApplication
```

```
@MapperScan("com.southwind.mybatisplus.mapper")
public class MybatisplusApplication {

    public static void main(String[] args)
    {

        SpringApplication.run(MybatisplusApplication.class, args);
    }

}
```

## 7、测试

```
package com.southwind.mybatisplus.mapper;

import org.junit.jupiter.api.Test;
import
org.springframework.beans.factory.annotation.Autowired;
import
org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest
class UserMapperTest {
```

```
@Autowired
private UserMapper mapper;

@Test
void test() {

    mapper.selectList(null).forEach(System.out
::println);
    }

}
```

## 常用注解

| @TableName

映射数据库的表名

```
package com.southwind.mybatisplus.entity;

import
com.baomidou.mybatisplus.annotation.TableName;
import lombok.Data;

@Data
@TableName(value = "user")
public class Account {
    private Integer id;
    private String name;
    private Integer age;
}
```

## @TableId

设置主键映射，value 映射主键字段名

type 设置主键类型，主键的生成策略，

```
AUTO(0),
NONE(1),
INPUT(2),
ASSIGN_ID(3),
ASSIGN_UUID(4),
```

```
/** @deprecated */
@Deprecated
ID_WORKER(3),
/** @deprecated */
@Deprecated
ID_WORKER_STR(3),
/** @deprecated */
@Deprecated
UUID(4);
```

值	描述
AUTO	数据库自增
NONE	MP set 主键，雪花算法实现
INPUT	需要开发者手动赋值
ASSIGN_ID	MP 分配 ID，Long、Integer、String
ASSIGN_UUID	分配 UUID，String

INPUT 如果开发者没有手动赋值，则数据库通过自增的方式给主键赋值，如果开发者手动赋值，则存入该值。

AUTO 默认就是数据库自增，开发者无需赋值。

ASSIGN\_ID MP 自动赋值，雪花算法。



ASSIGN\_UUID 主键的数据类型必须是 String，自动生成 UUID 进行赋值

## @TableField

映射非主键字段，value 映射字段名

exist 表示是否为数据库字段 false，如果实体类中的成员变量在数据库中没有对应的字段，则可以使用 exist，VO、DTO

select 表示是否查询该字段

fill 表示是否自动填充，将对象存入数据库的时候，由 MyBatis Plus 自动给某些字段赋值，create\_time、update\_time

- 1、给表添加 create\_time、update\_time 字段
- 2、实体类中添加成员变量

```
package com.southwind.mybatisplus.entity;  
  
import  
com.baomidou.mybatisplus.annotation.FieldFill;  
11;
```

```
import
com.baomidou.mybatisplus.annotation.TableField;
import
com.baomidou.mybatisplus.annotation.TableId;
import
com.baomidou.mybatisplus.annotation.TableName;
import lombok.Data;

import java.util.Date;

@Data
@TableName(value = "user")
public class User {
    @TableId
    private String id;
    @TableField(value = "name",select = false)
    private String title;
    private Integer age;
    @TableField(exist = false)
    private String gender;
    @TableField(fill = FieldFill.INSERT)
    private Date createTime;
```

```
        @TableField(fill =  
FieldFill.INSERT_UPDATE)  
        private Date updateTime;  
    }
```

### 3、创建自动填充处理器

```
package com.southwind.mybatisplus.handler;  
  
import  
com.baomidou.mybatisplus.core.handlers.Meta  
ObjectHandler;  
import  
org.apache.ibatis.reflection.MetaObject;  
import  
org.springframework.stereotype.Component;  
  
import java.util.Date;  
  
@Component  
public class MyMetaObjectHandler implements  
MetaObjectHandler {  
    @Override  
    public void insertFill(MetaObject  
metaObject) {
```

```
    this.setFieldValByName("createTime", new
Date(), metaObject);

    this.setFieldValByName("updateTime", new
Date(), metaObject);
}

@Override
public void updateFill(MetaObject
metaObject) {

    this.setFieldValByName("updateTime", new
Date(), metaObject);
}
}
```

## @Version

标记乐观锁，通过 version 字段来保证数据的安全性，当修改数据的时候，会以 version 作为条件，当条件成立的时候才会修改成功。

version = 2

线程 1:update ... set version = 2 where version = 1

线程2： update ... set version = 2 where version = 1

1、数据库表添加 version 字段，默认值为 1

2、实体类添加 version 成员变量，并且添加 @Version

```
package com.southwind.mybatisplus.entity;

import
com.baomidou.mybatisplus.annotation.*;
import lombok.Data;

import java.util.Date;

@Data
@TableName(value = "user")
public class User {
    @TableId
    private String id;
    @TableField(value = "name",select =
false)
    private String title;
    private Integer age;
    @TableField(exist = false)
    private String gender;
    @TableField(fill = FieldFill.INSERT)
    private Date createTime;
```

```
    @TableField(fill =  
FieldFill.INSERT_UPDATE)  
    private Date updateTime;  
    @Version  
    private Integer version;  
}
```

### 3、注册配置类

```
package com.southwind.mybatisplus.config;  
  
import  
com.baomidou.mybatisplus.extension.plugins.  
OptimisticLockerInterceptor;  
import  
org.springframework.context.annotation.Bean  
;  
import  
org.springframework.context.annotation.Conf  
iguration;  
  
@Configuration  
public class MyBatisPlusConfig {  
  
    @Bean  
    public OptimisticLockerInterceptor  
optimisticLockerInterceptor(){
```

```
        return new  
OptimisticLockerInterceptor();  
    }  
  
}
```

## @EnumValue

1、通用枚举类注解，将数据库字段映射成实体类的枚举类型成员变量

```
package com.southwind.mybatisplus.enums;  
  
import  
com.baomidou.mybatisplus.annotation.EnumValue;  
  
public enum StatusEnum {  
    WORK(1, "上班"),  
    REST(0, "休息");  
  
    StatusEnum(Integer code, String msg) {  
        this.code = code;  
        this.msg = msg;  
    }  
}
```

```
    @EnumValue
    private Integer code;
    private String msg;
}
```

```
package com.southwind.mybatisplus.entity;

import
com.baomidou.mybatisplus.annotation.*;
import
com.southwind.mybatisplus.enums.StatusEnum;
import lombok.Data;

import java.util.Date;

@Data
@TableName(value = "user")
public class User {
    @TableId
    private String id;
    @TableField(value = "name",select =
false)
    private String title;
    private Integer age;
    @TableField(exist = false)
    private String gender;
    @TableField(fill = FieldFill.INSERT)
```



```
    private Date createTime;
    @TableField(fill =
FieldFill.INSERT_UPDATE)
    private Date updateTime;
    @Version
    private Integer version;
    private StatusEnum status;
}
```

application.yml

```
type-enums-package:
    com.southwind.mybatisplus.enums
```

## 2、实现接口

```
package com.southwind.mybatisplus.enums;

import
com.baomidou.mybatisplus.core.enums.IEnum;

public enum AgeEnum implements
IEnum<Integer> {
    ONE(1, "一岁"),
    TWO(2, "两岁"),
    THREE(3, "三岁");
}
```

```
private Integer code;
private String msg;

AgeEnum(Integer code, String msg) {
    this.code = code;
    this.msg = msg;
}

@Override
public Integer getValue() {
    return this.code;
}
}
```

## @TableLogic

### 映射逻辑删除

- 1、数据表添加 deleted 字段
- 2、实体类添加注解

```
package com.southwind.mybatisplus.entity;

import
com.baomidou.mybatisplus.annotation.*;
```

```
import
com.southwind.mybatisplus.enums.AgeEnum;
import
com.southwind.mybatisplus.enums.StatusEnum;
import lombok.Data;

import java.util.Date;

@Data
@TableName(value = "user")
public class User {
    @TableId
    private String id;
    @TableField(value = "name",select =
false)
    private String title;
    private AgeEnum age;
    @TableField(exist = false)
    private String gender;
    @TableField(fill = FieldFill.INSERT)
    private Date createTime;
    @TableField(fill =
FieldFill.INSERT_UPDATE)
    private Date updateTime;
    @Version
    private Integer version;
    @TableField(value = "status")
```

```
private StatusEnum statusEnum;  
@TableLogic  
private Integer deleted;  
}
```

### 3、application.yml 添加配置

```
global-config:  
  db-config:  
    logic-not-delete-value: 0  
    logic-delete-value: 1
```

## 查询

```
//mapper.selectList(null);  
QueryWrapper wrapper = new QueryWrapper();  
//      Map<String,Object> map = new  
HashMap<>();  
//      map.put("name","小红");  
//      map.put("age",3);  
//      wrapper.allEq(map);  
//      wrapper.gt("age",2);  
//      wrapper.ne("name","小红");  
//      wrapper.ge("age",2);
```

```
//like '%小'
//          wrapper.likeLeft("name","小");
//like '小%'
//          wrapper.likeRight("name","小");

//inSQL
//          wrapper.inSql("id","select id
from user where id < 10");
//          wrapper.inSql("age","select age
from user where age > 3");

//          wrapper.orderByDesc("age");

//          wrapper.orderByAsc("age");
//          wrapper.having("id > 8");

mapper.selectList(wrapper).forEach(System.o
ut::println);
```

```
//
System.out.println(mapper.selectById(7));
//
mapper.selectBatchIds(Arrays.asList(7,8,9))
.forEach(System.out::println);
```

```
//Map 只能做等值判断, 逻辑判断需要使用 Wrapper  
来处理  
  
//          Map<String,Object> map = new  
HashMap<>();  
//          map.put("id",7);  
//  
mapper.selectByMap(map).forEach(System.out:  
:println);  
  
QueryWrapper wrapper = new QueryWrapper();  
wrapper.eq("id",7);  
/////  
System.out.println(mapper.selectCount(wrapper));  
//  
//          //将查询的结果集封装到Map中  
//  
mapper.selectMaps(wrapper).forEach(System.o  
ut::println);  
//          System.out.println("-----  
-----");  
//  
mapper.selectList(wrapper).forEach(System.o  
ut::println);  
  
//分页查询
```

```
//          Page<User> page = new Page<>
(2,2);
//          Page<User> result =
mapper.selectPage(page,null);
//
System.out.println(result.getSize());
//
System.out.println(result.getTotal());
//
result.getRecords().forEach(System.out::pri
ntln);

//          Page<Map<String,Object>> page =
new Page<>(1,2);
//
mapper.selectMapsPage(page,null).getRecords
().forEach(System.out::println);

//
mapper.selectObjs(null).forEach(System.out:
:println);

System.out.println(mapper.selectOne(wrapper
));
```

# 自定义 SQL（多表关联查询）

```
package com.southwind.mybatisplus.entity;

import lombok.Data;

@Data
public class ProductVO {
    private Integer category;
    private Integer count;
    private String description;
    private Integer userId;
    private String userName;
}
```



```
package com.southwind.mybatisplus.mapper;

import
com.baomidou.mybatisplus.core.mapper.BaseMa
pper;
import
com.southwind.mybatisplus.entity.ProductVO;
import
com.southwind.mybatisplus.entity.User;
import
org.apache.ibatis.annotations.Select;

import java.util.List;

public interface UserMapper extends
BaseMapper<User> {
    @Select("select p.*,u.name userName
from product p,user u where p.user_id =
u.id and u.id = #{id}")
    List<ProductVO> productList(Integer
id);
}
```

## 添加

```
User user = new User();
user.setTitle("小明");
user.setAge(22);
mapper.insert(user);
System.out.println(user);
```

## 删除

```
//mapper.deleteById(1);
//
mapper.deleteBatchIds(Arrays.asList(7,8));
//          QueryWrapper wrapper = new
QueryWrapper();
//          wrapper.eq("age",14);
//          mapper.delete(wrapper);

Map<String,Object> map = new HashMap<>();
map.put("id",10);
mapper.deleteByMap(map);
```

## 修改

```
//          //update ... version = 3 where
version = 2
//          User user = mapper.selectById(7);
//          user.setTitle("一号");
//
//          //update ... version = 3 where
version = 2
//          User user1 =
mapper.selectById(7);
//          user1.setTitle("二号");
//
//          mapper.updateById(user1);
//          mapper.updateById(user);

User user = mapper.selectById(1);
user.setTitle("小红");
QueryWrapper wrapper = new QueryWrapper();
wrapper.eq("age", 22);
mapper.update(user, wrapper);
```

## MyBatisPlus 自动生成

根据数据表自动生成实体类、Mapper、Service、ServiceImpl、Controller

## 1、pom.xml 导入 MyBatis Plus Generator

```
<dependency>
    <groupId>com.baomidou</groupId>
    <artifactId>mybatis-plus-generator</artifactId>
    <version>3.3.1.tmp</version>
</dependency>

<dependency>
    <groupId>org.apache.velocity</groupId>
    <artifactId>velocity</artifactId>
    <version>1.7</version>
</dependency>
```

Velocity (默认) 、Freemarker、Beetl

## 2、启动类

```
package com.southwind.mybatisplus;

import
com.baomidou.mybatisplus.annotation.DbType;
import
com.baomidou.mybatisplus.generator.AutoGenerator;
```

```
import
com.baomidou.mybatisplus.generator.config.D
ataSourceConfig;
import
com.baomidou.mybatisplus.generator.config.G
lobalConfig;
import
com.baomidou.mybatisplus.generator.config.P
ackageConfig;
import
com.baomidou.mybatisplus.generator.config.S
trategyConfig;
import
com.baomidou.mybatisplus.generator.config.r
ules.NamingStrategy;

public class Main {
    public static void main(String[] args)
    {
        //创建generator对象
        AutoGenerator autoGenerator = new
AutoGenerator();
        //数据源
        DataSourceConfig dataSourceConfig =
new DataSourceConfig();

        dataSourceConfig.setDbType(DbType.MYSQL);
```

```
dataSourceConfig.setUrl("jdbc:mysql://ip:306/db?useUnicode=true&characterEncoding=UTF-8");

dataSourceConfig.setUsername("root");

dataSourceConfig.setPassword("root");

dataSourceConfig.setDriverName("com.mysql.cj.jdbc.Driver");

autoGenerator.setDataSource(dataSourceConfig);

    //全局配置
    GlobalConfig globalConfig = new GlobalConfig();

    globalConfig.setOutputDir(System.getProperty("user.dir")+"/src/main/java");
    globalConfig.setOpen(false);

    globalConfig.setAuthor("southwind");

    globalConfig.setServiceName("%sService");
```

```
autoGenerator.setGlobalConfig(globalConfig
);

    //包信息
    PackageConfig packageConfig = new
PackageConfig();

    packageConfig.setParent("com.southwind.myb
atisplus");

    packageConfig.setModuleName("generator");

    packageConfig.setController("controller");

    packageConfig.setService("service");

    packageConfig.setServiceImpl("service.impl
");

        packageConfig.setMapper("mapper");
        packageConfig.setEntity("entity");

    autoGenerator.setPackageInfo(packageConfig
);

    //配置策略
    StrategyConfig strategyConfig = new
StrategyConfig();
```

```
strategyConfig.setEntityLombokModel(true);

strategyConfig.setNaming(NamingStrategy.underline_to_camel);

strategyConfig.setColumnNaming(NamingStrategy.underline_to_camel);

autoGenerator.setStrategy(strategyConfig);

        autoGenerator.execute();
    }
}
```

## Spring Boot + MyBatis Plus 打包应用，直接发布 阿里云 上云