**SpringBoot 集成mybatis**

maven相关

<!-- 引入 mybatis -->  
 <!--为项目引入mybatis依赖-->  
 <dependency>  
 <groupId>org.mybatis.spring.boot</groupId>  
 <artifactId>mybatis-spring-boot-starter</artifactId>  
 <version>${mybatis-spring-boot-starter.version}</version>  
 </dependency>  
 <!--定制自己的通用mapper-->  
 <dependency>  
 <groupId>tk.mybatis</groupId>  
 <artifactId>mapper-spring-boot-starter</artifactId>  
 <version>${mapper-spring-boot-starter.version}</version>  
 </dependency>  
 <!--分页插件-->  
 <dependency>  
 <groupId>com.github.pagehelper</groupId>  
 <artifactId>pagehelper-spring-boot-starter</artifactId>  
 <version>${pagehelper-spring-boot-starter.version}</version>  
 </dependency>  
<!--引入mybatis end-->

application.yml配置

**spring:** *#-------------database----------#* **datasource:** *#数据库* **url:** jdbc:oracle:thin:@127.0.0.1:1521:orcl  
 **username:** tyjx  
 **password:** tyjx  
 **driver-class-name:** oracle.jdbc.driver.OracleDriver *#-------------database end----------#*

引入mybatis-generator

快速生成mapper，model，dao

首先在pom.xml配置

<plugin>  
 <groupId>org.mybatis.generator</groupId>  
 <artifactId>mybatis-generator-maven-plugin</artifactId>  
 <version>1.3.5</version>  
 <configuration>  
 <configurationFile>src/main/resources/generator/generatorConfig.xml</configurationFile>  
 <overwrite>true</overwrite>  
 <verbose>true</verbose>  
 </configuration>  
 <dependencies>  
 <dependency>  
 <groupId>com.oracle</groupId>  
 <artifactId>ojdbc14</artifactId>  
 <version>${ojdbc14.version}</version>  
 </dependency>  
 <dependency>  
 <groupId>tk.mybatis</groupId>  
 <artifactId>mapper</artifactId>  
 <version>3.4.4</version>  
 </dependency>  
 </dependencies>  
</plugin>

创建配置文件：

src/main/resources/generator下

**generator.properties**

# oracle  
jdbc.driverClass = **oracle.jdbc.driver.OracleDriver**jdbc.url=**jdbc**\:**oracle**\:**thin**\:**@127.0.0.1**\:**1521**\:**orcl**  
jdbc.user = **tyjx**jdbc.password = **tyjx**# mapper config  
mapper.plugin = **tk.mybatis.mapper.generator.MapperPlugin**mapper.Mapper = **com.first.study.service.mybatis.MyMapper**#指定文件生成目录,以及Java文件的package,值末尾不能有空格  
targetJavaProject = **c:/createspringboot**targetModelPackage = **com.first.study.service.mybatis.model**targetMapperPackage = **com.first.study.service.mybatis.dao**targetResourcesProject = **c:/createspringboot**targetXMLPackage =**com.first.study.service.mybatis.mapper**

**generatorConfig.xml**

参考项目里的配置

然后创建MyMapper接口，方便使用生成的mapper接口

package com.first.study.service.mybatis;  
  
import tk.mybatis.mapper.common.IdsMapper;  
import tk.mybatis.mapper.common.Mapper;  
  
public interface MyMapper<T> extends Mapper<T>, IdsMapper<T> {  
}

配置完成，还需要在系统入口添加扫描的包和mapper

**@SpringBootApplication**(scanBasePackages={**"com.first.study.\*"**})  
**@MapperScan**(**"com.first.study.\*\*.dao"**)  
public class Application {  
 //默认启动器  
 public static void main(String[] args) throws Exception {  
 SpringApplication.*run*(Application.class, args);  
 }

然后使用方式：

service：

**@Service**public class RedisService {

**@Autowired**TSBaseUserMapper tsBaseUserMapper;  
Logger logger= LoggerFactory.*getLogger*(RedisService.class);

**@Transactional**public void updateData(){  
 logger.info(**"测试INFO日志信息"**);  
 logger.debug(**"测试DEBUG是否会写到file"**);  
 logger.trace(**"测试trace是否会写到file"**);  
 logger.warn(**"测试WARN是否会写到file"**);  
 logger.error(**"测试ERROR是否会写到file"**);  
 List<TSBaseUser> BU=tsBaseUserMapper.getBU();

}

}

**Mapper：**

public interface TSBaseUserMapper extends MyMapper<TSBaseUser> {  
 List<TSBaseUser> getBU();  
}

**MapperXML:id与Mapper的方法名对应**

<select id**="getBU"** resultMap**="BaseResultMap"** >  
 select \* from t\_s\_base\_user t  
</select>

Controller：

//@Controller  
//@ResponseBody//可放在方法头部也可以放在类头部

**@RestController**//@RestController相当于@Controller+@ResponseBody  
**@RequestMapping**(**"/api/redistest/"**)  
public class RedisTestController {  
 **@Autowired** RedisService redisService;  
 public String user=**"初始化"**;  
 **@RequestMapping**(**"hello"**)public String helloworld(){  
 redisService.updateData();  
 return **"hello"**;  
 }

**多数据源配置：**