### 1.初识springboot

@SpringBootApplication注解它是声明当前类为sprinboot的入口类，应该放在底层包文件中，而一个springboot项目内有且只能有一个这个注解存在。

### 2.springboot整合mybatis的两种方式

orm框架的本质是简化编程中操作数据库的编码，发展到现在基本上就剩两家了，一个是宣称可以不用写一句SQL的hibernate，一个是可以灵活调试动态sql的mybatis,两者各有特点，在企业级系统开发中可以根据需求灵活使用。发现一个有趣的现象：传统企业大都喜欢使用hibernate,互联网行业通常使用mybatis。

mybatis-spring-boot-starter主要有两种解决方案，一种是使用注解解决一切问题，一种是简化后的老传统。

引入mybatis-spring-boot-starter的pom文件：

<dependency>

<groupId>org.mybatis.spring.boot</groupId>

<artifactId>mybatis-spring-boot-starter</artifactId>

<version>1.1.1</version>

</dependency>

无配置文件注解版

1 添加相关maven文件

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.mybatis.spring.boot</groupId>

<artifactId>mybatis-spring-boot-starter</artifactId>

<version>1.1.1</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<optional>true</optional>

</dependency></dependencies>

2、application.properties 添加相关配置

mybatis.type-aliases-package=com.jxp.bean

spring.datasource.driverClassName=com.mysql.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/test1?useUnicode=true&characterEncoding=utf-8

spring.datasource.username = root

spring.datasource.password = root

springboot会自动加载spring.datasource.\*相关配置，数据源就会自动注入到sqlSessionFactory中，sqlSessionFactory会自动注入到Mapper中，对了你一切都不用管了，直接拿起来使用就行了。

在启动类中添加对mapper包扫描@MapperScan

@SpringBootApplication

@MapperScan("com.jxp.mapper")

public class Application {

public static void main(String[] args) {

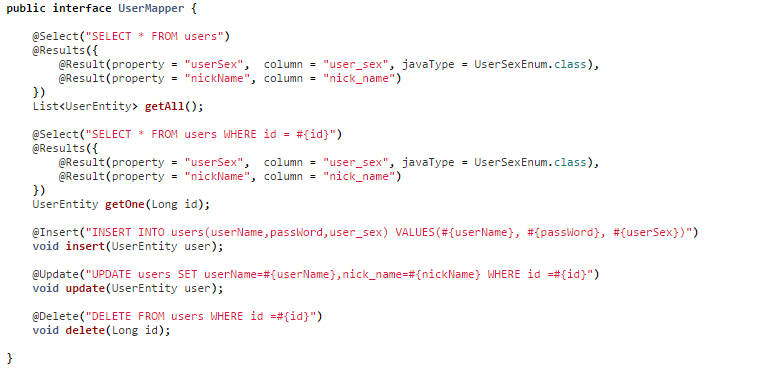
SpringApplication.run(Application.class, args);

}}

或者直接在Mapper类上面添加注解@Mapper,建议使用上面那种，不然每个mapper加个注解也挺麻烦的。

3、开发Mapper

第三步是最关键的一块，sql生产都在这里



为了更接近生产我特地将user\_sex、nick\_name两个属性在数据库加了下划线和实体类属性名不一致，另外user\_sex使用了枚举

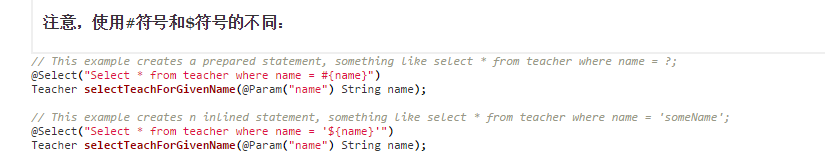
@Select 是查询类的注解，所有的查询均使用这个

@Result 修饰返回的结果集，关联实体类属性和数据库字段一一对应，如果实体类属性和数据库属性名保持一致，就不需要这个属性来修饰。

@Insert 插入数据库使用，直接传入实体类会自动解析属性到对应的值

@Update 负责修改，也可以直接传入对象

@delete 负责删除



极简xml版本

极简xml版本保持映射文件的老传统，优化主要体现在不需要实现dao的是实现层，系统会自动根据方法名在映射文件中找对应的sql.