1.首先在pom.xml里面引入依赖（按道理来讲是用到哪个导入哪个来的，这里直接copy学长的pom.xml引入依赖就行）

2.编写web.xml文件

①配置spring的监听器（指定spring配置文件的位置）

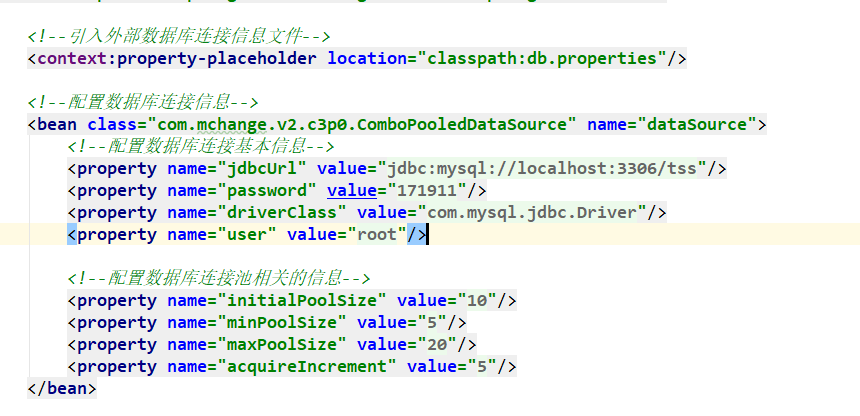
②配置springmvc的DispatcherServlet（指定springmvc配置文件的位置）

③字符编码过滤器（指定为UTF-8）

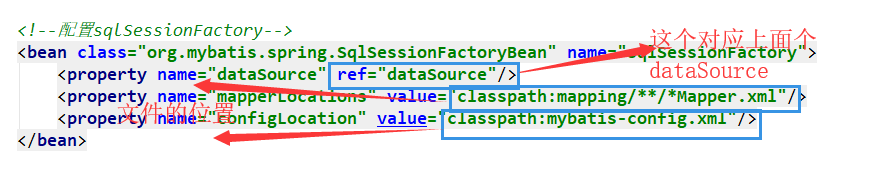
下面根据编写spring配置文件会陆续建立相应的配置文件

3.spring配置文件

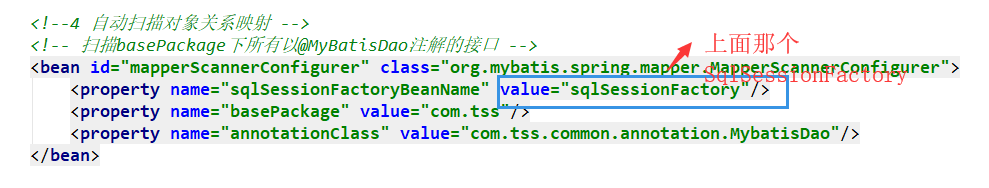
①因为要配置DataSource（数据库的连接信息），需要数据库的连接信息，所以需要编写一个db.properties文件（里面写数据库连接的信息），编写完成之后，在spring里面引入这个文件



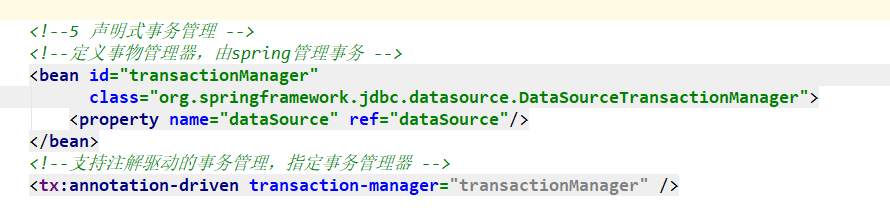
②然后配置sqlSessionFactory，（这个个是mybaits的一个类，因为spring是管理各种东西的配置文件，所以说把这个类弄成那个一个类，给他对应的值）在这里面要指定他的dataSOurce，还要指定mapper文件的位置和mybatis配置文件的位置



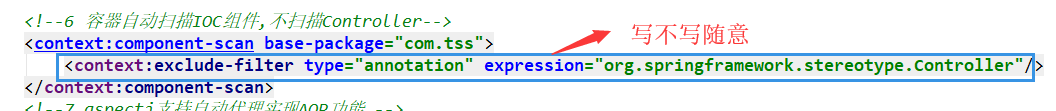
③然后扫描@MybatisDao注解的接口（这个注解是那个类定义的）



④配置声明式事物（数据库中的事务还记得吗）



⑤配置自动扫描IOC组件（也就是你用@Service和@Controller等注解的那些东西），我此处写的是把@Controller排除了，因为@Controller注解是应该springmvc上扫描的东西（教程上教的，学长并没有分那么细，不分也没大事，就是多扫描一遍罢了）



⑥AOP（面向切面），emm，配置上把



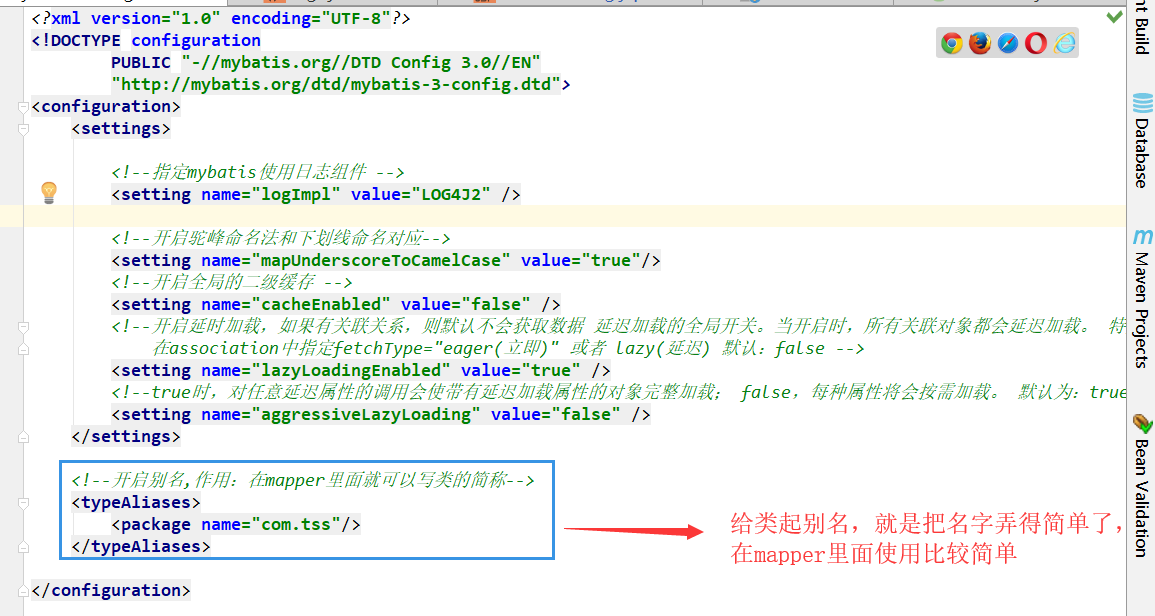
⑦工具类要用的，就是工具类要用的



4.mybatis-config.xml

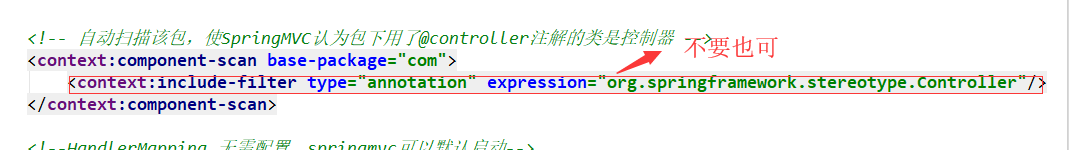
这个配置文件可以没有，因为所有能够在这里面配置的东西，在spring里面的SqlSessionFactory里面进行配置，所以说可以没有这个配置文件

但一般都留一个这个配置文件，就算不写东西。。。也得这个留一个，这里面常用的配置就是开启mybatis的二级缓存机制了

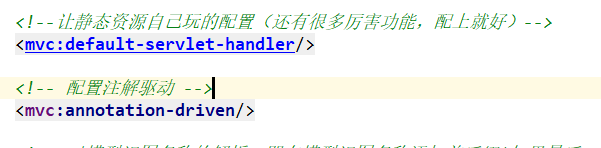


5.spring-mvc配置文件

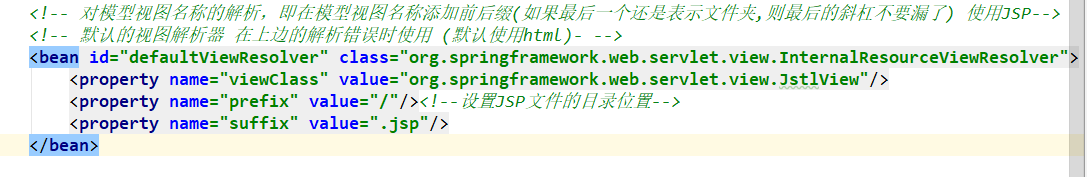
①自动扫描包（此处我写的是只扫描@Controller注解，跟上面理由相对应）



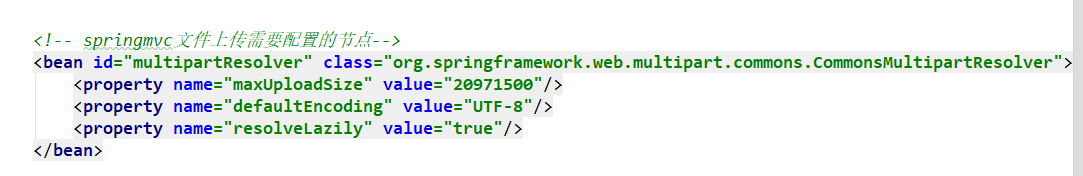
②然后每次必须配置的两个配置（这个用处有点多，根本无法描述）



③视图解析器，大家都配过这个，我就不说了



④文件上传（暂时还没用上，配上也行）

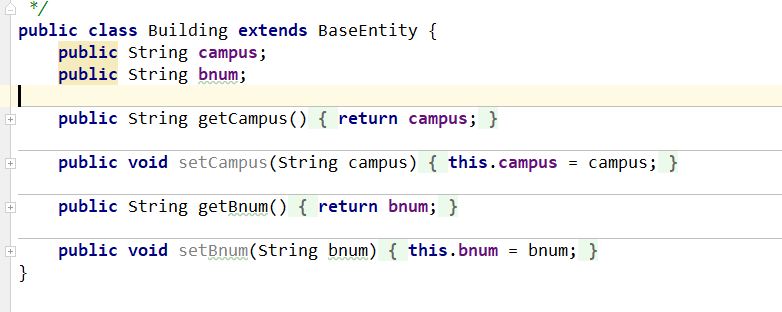


6.log4j2.xml

这个文件我是直接copy的学长的那个，这个就是指定在控制台打印一些日志信息（应该大概是这个意思，我也不太懂具体的配置）

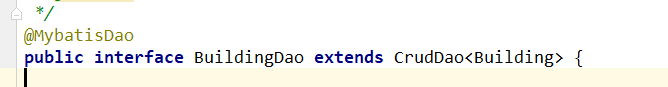
下面开始以building为例

1.根据数据库建立出来实体类，实体类继承BaseEntity，因为父类entity里面有id还有get和set方法，所以说不用写id

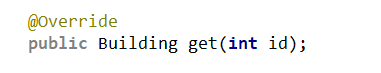


注意：类中的属性和数据库中的字段名（因为数据库不区分大小写，所以说数据库中用的大些，类中使用驼峰命名法）要一样（其实是get和set方法后面的那部分一样，现在记住就是一样就行了）

2.写完entity之后，写dao，dao是一个接口，此处命名为BuildingDao，继承自CrudDao接口，此处会有一个范型，范型直接写刚刚建立号的那个entity（Building），然后用@MybatisDao注解注解在类上。



3.我们此处只需要把数据库中的数据查出来即可，所以说，我们只需要先写一个get方法



然后分析一波这个get方法，这个东西把，返回值是一个Building对象，是因为查出来的数据他给你自动封装成一个对象，然后传入的参数是id，这个方法的意思就是根据主键id来查出整条数据。

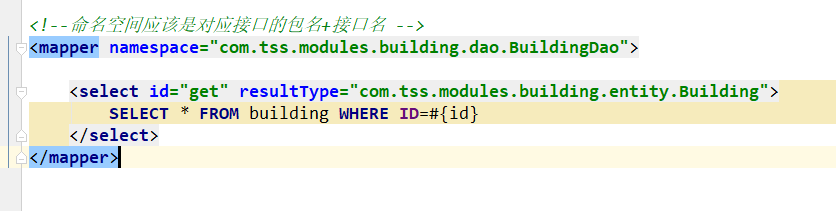
然后上面的@Override注解是为了防止自己重载了这个方法（也就是参数写的不同了，导致这个方法不是重写的父类中的方法）

4.然后一个dao就对应一个mapper文件，dao是接口，然后mapper配置文件是这个接口的实现。mapper配置文件都放在mapping下面（我的是放在了这下面，如果你不是，你需要去修改spring中那个配置去标注你的位置）

然后建立mapper文件，开头是这个文件要声明的一些东西，是固定的，在开头加上下面的代码（可以去看官方文档）

*<?***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"*>***

如上所示，



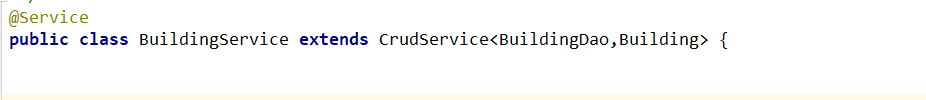
如上所示，先写一个mapper标签，mapper里面的namespace里面标注的是哪个dao对应的mapper文件，

然后下面的select标签对应的是那个get方法（当然，对应的删除，修改，和插入分别是delete，update和insert），然后其中的id指定是方法名，然后resultType是指定方法的返回值类型

（如果是很复杂的表，比如说学生表里面有个班级，那么如果你想要查这个学生的时候把这个学生的班级也查出来，你就要自己写一个ResultMap，然后里面你自定义返回值中实体属性和表中字段的对应关系，当然你也可以写表的连接的sql’语句，来查，那样的话，sql语句会写的复杂一些，上面嘟嘟的啥我也不知道，自己瞅瞅把，resultMap我可能说不清楚，去百度一下把，而且学长以前发的学习的文档好像也有这个的，我看过呢还）

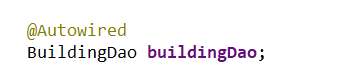
然后里面就写对应的sql语句，他会自动把你从数据库中查出对应的字段给你封装成一个Building对象，然后此时你的那个dao接口就有了实现，所以下面写service

5.因为此处我们并没有什么业务逻辑，所以说，在service只需要调用dao里面的方法返回就行

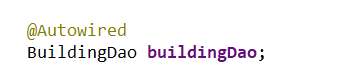


建立一个类继承CrudService，其中有两个范型，第一个写刚刚的dao，第二个写刚刚的entity

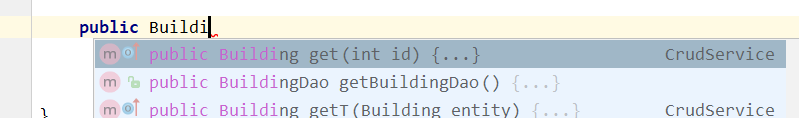
，然后@Service注解标注这是一个service

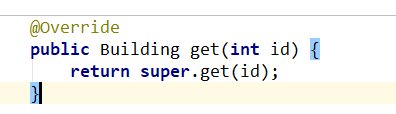


因为下面的方法调用中需要用到BuildingDao，所以说需要自动注入一个BudingDao（都是这样，这是固定套路，在service里面自动注入dao，在controller里面自动注入service）



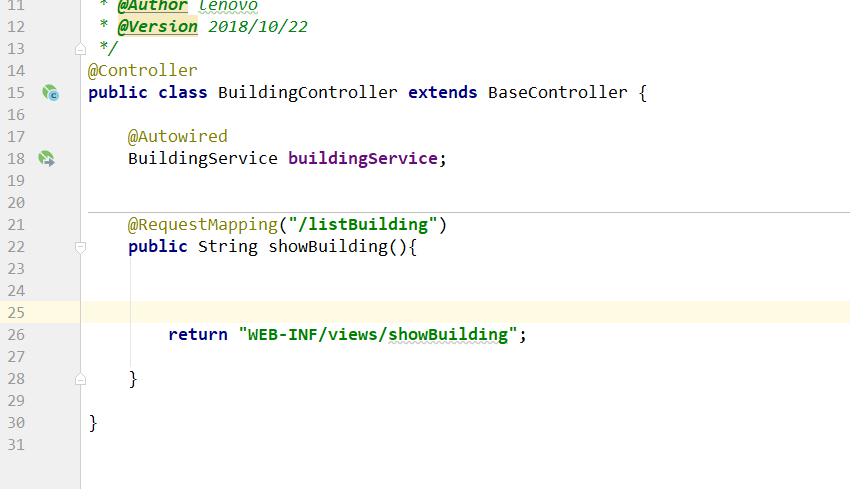
然后下面就写相应的方法



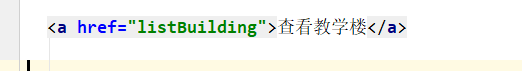


其中@Override还是为了防止重载方法

6.下面写controller，下面的controller什么的跟上面套路一样，我光说不一样的



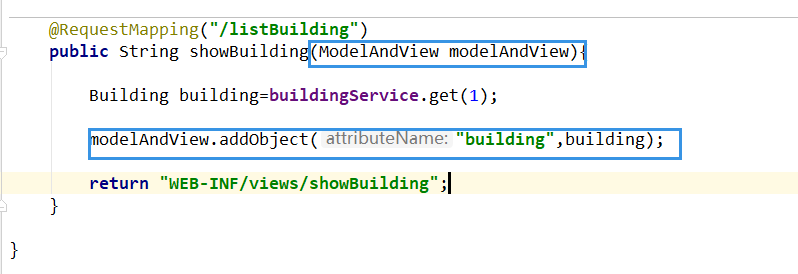
首先先写号跳转逻辑，@RequestMapping是指定他拦截哪个请求，然后学长要求在index里面访问，所以就在index.jsp里面写上

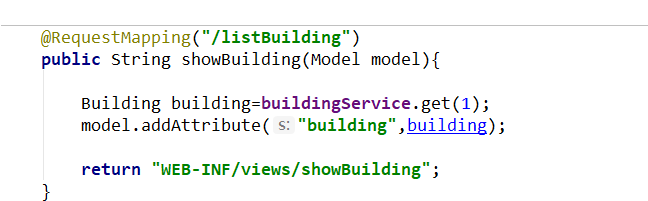


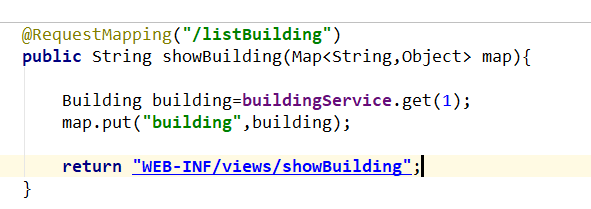
然后下面的返回值和springmvc的配置文件中的试图解析器拼接起来就是他要访问的jsp页面的地址，然后就需要写在views下面建立一个showBuilding的jsp页面

而此时，只是完成了页面的跳转逻辑

下面完成从service层获取Building对象，并且把这个对象放入到请求域里面，然后在页面里用这个对象，可以采用下面几种方式

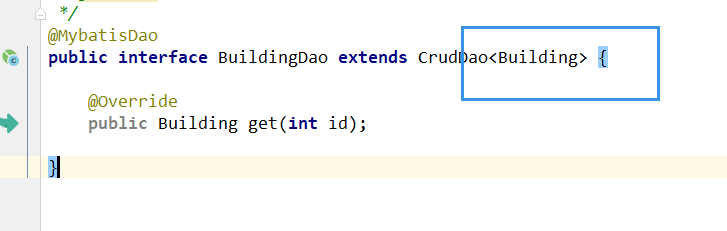






个人喜欢最后一种，因为自己不懂前两个，所以喜欢map

最后说一说范型把，那个范型

就看上面这张图，它上面的T是后来传进来的，这里指定的是Building也就在那个T，传入什么就当成什么，就比如说，上面的那个T get(int id)，也就是变成了Building get(int id)，就这么回事，，，，，

Map<String,Object>，这个东西也是一个范型，String就代表这个东西的第一个参数只能是String类型的，第二个参数只能是Object类型的

还有和Map、List等集合类结合起来的用法和更牛的用法和实现的原理就自行百度把