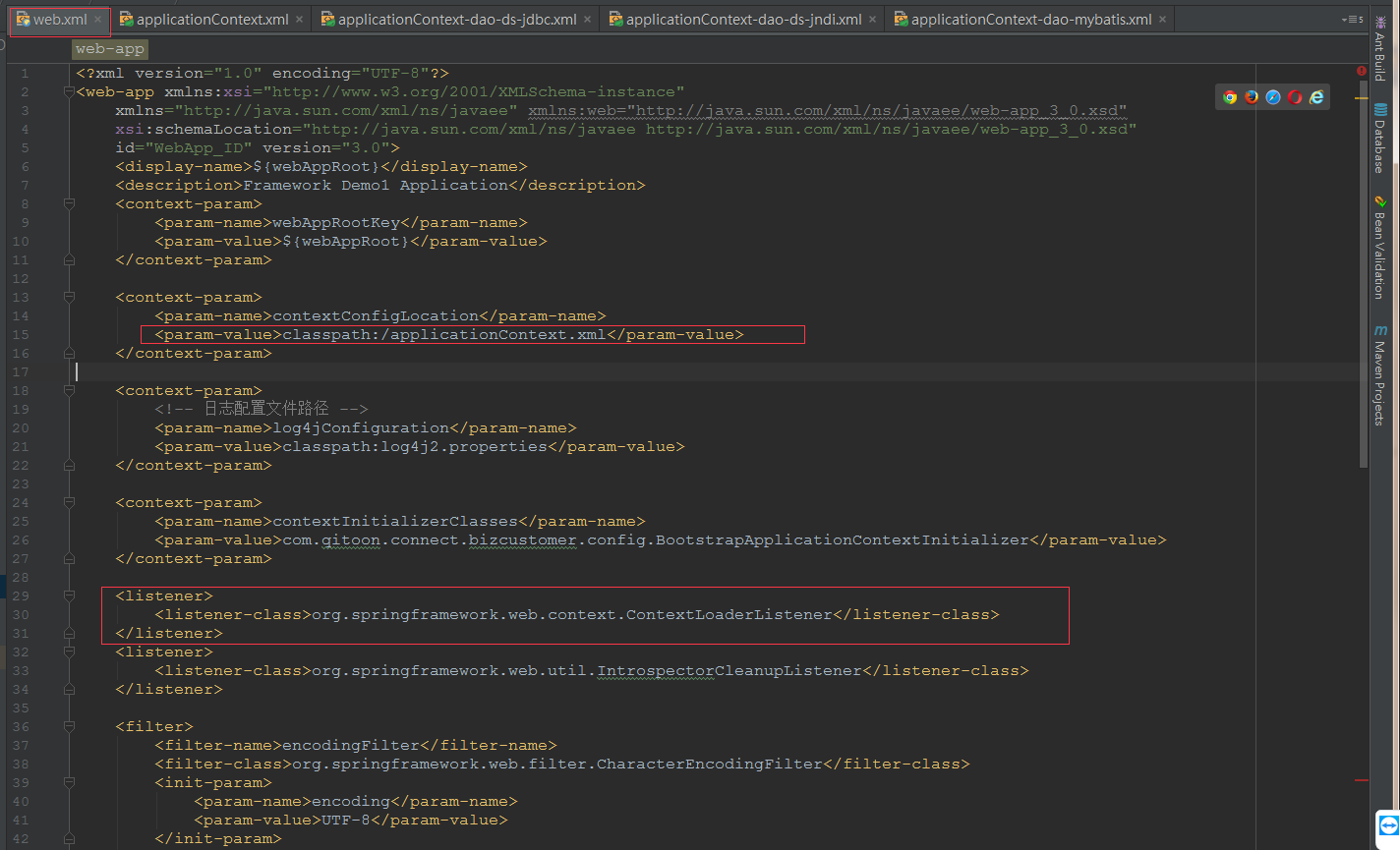
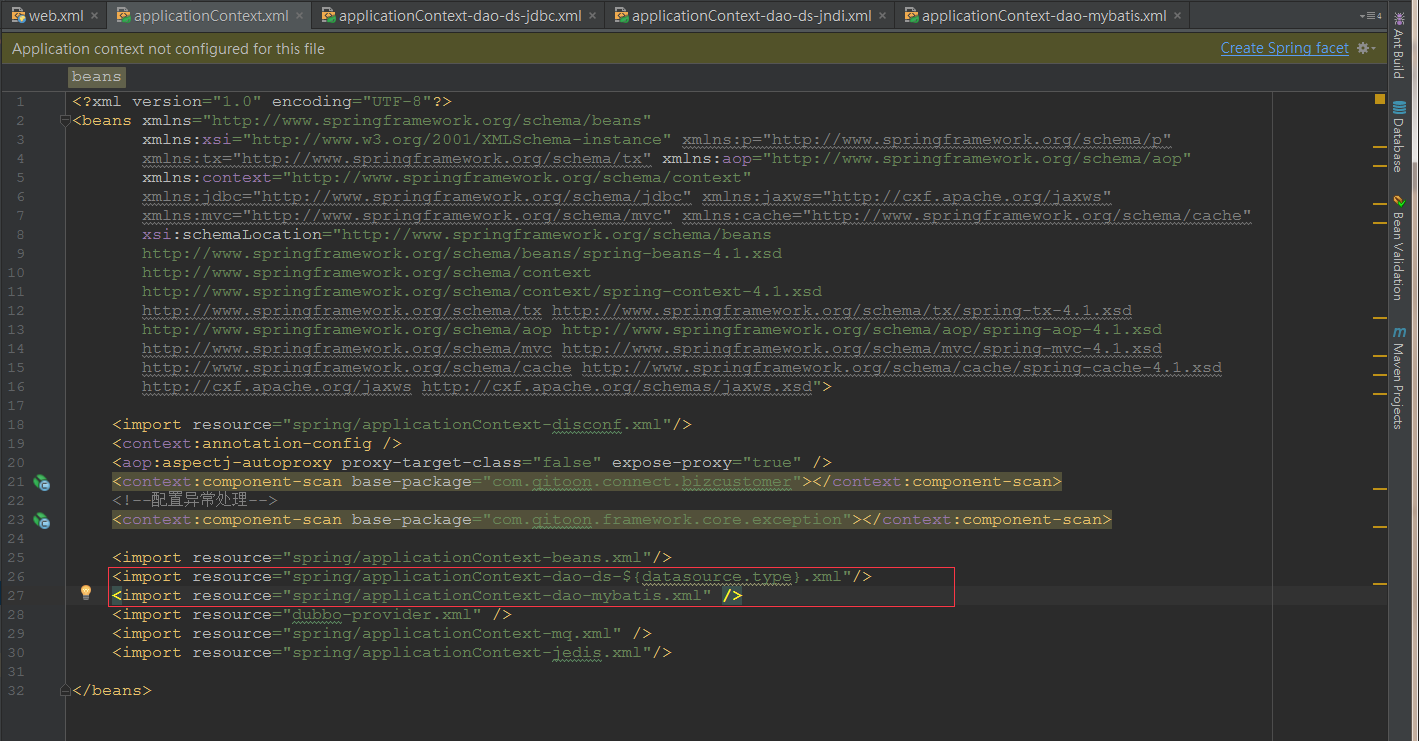
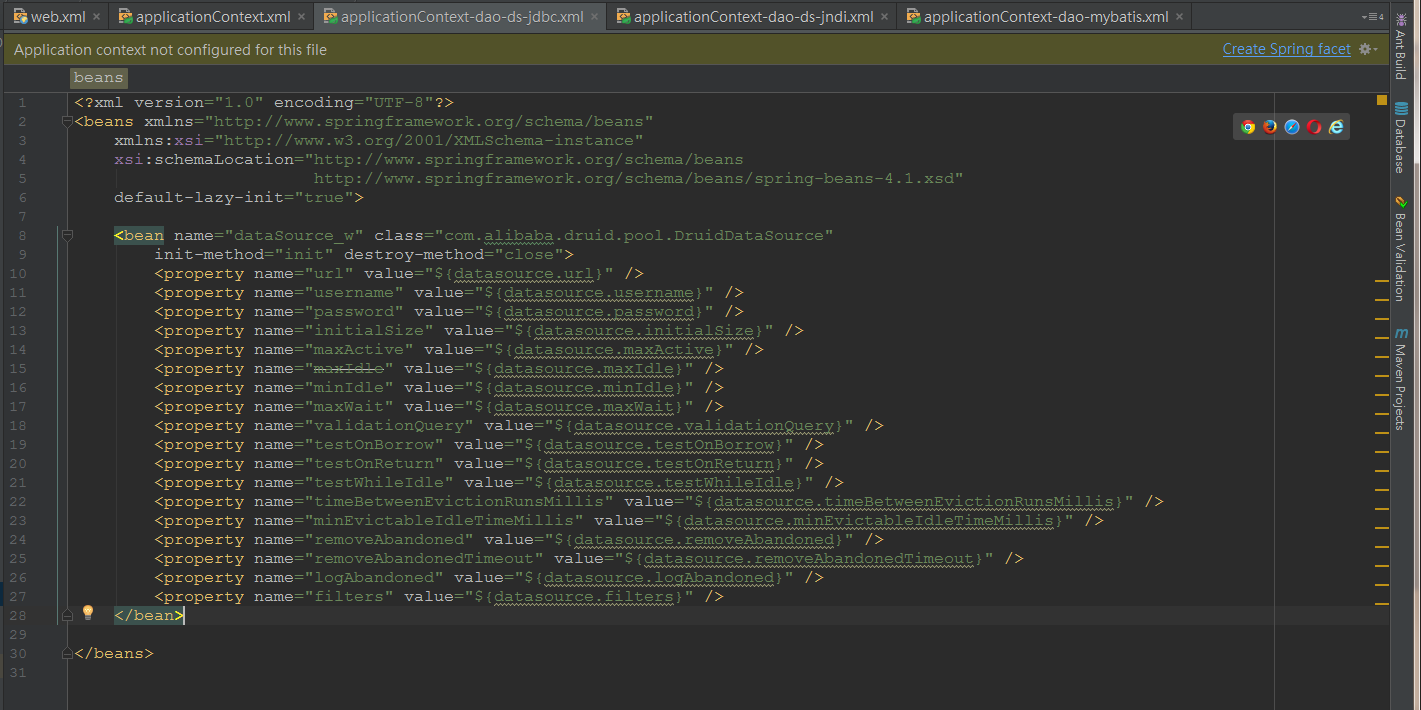
1.配置web.xml，在tomcat容器启动时，会即在spring的监听器，spring监听器会读取applicationContext.xml。

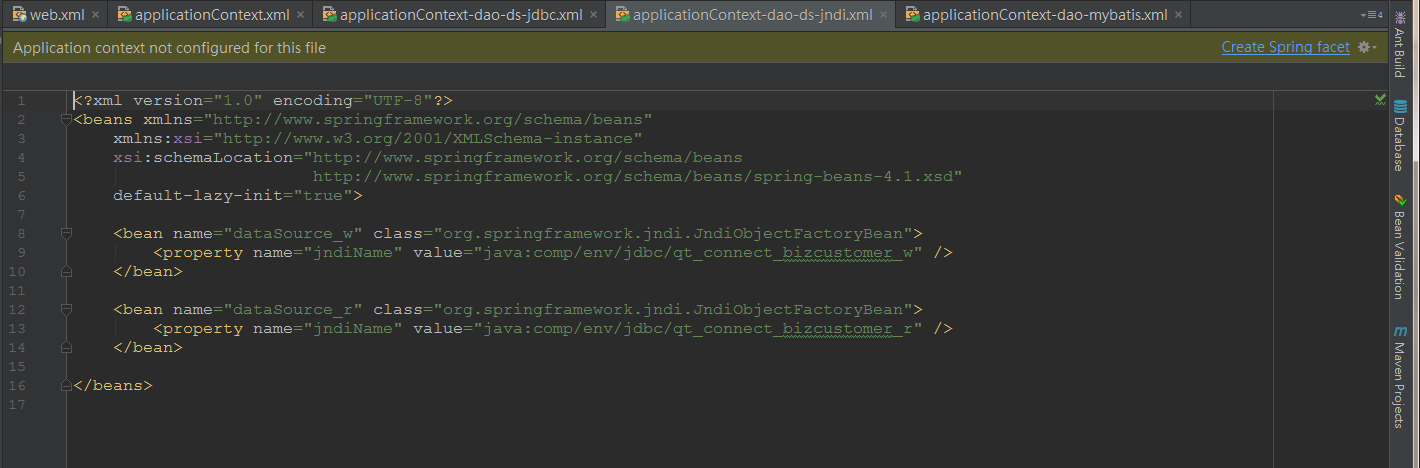


2.在applicationContext.xml中，导入了数据源连接池的配置和mybatis配置，使其都纳入spring的bean控制反转依赖注入管理体系中。



3.数据源连接池的配置。





4.配置mybatis相关的bean，主要依赖注入了刚才配置好的数据源连接池，表明mybatis的SqlSessionFactoryBean注入的数据源是刚才配置的的那个数据源bean。

进而再配置SqlSessionTemplate对应，依赖刚才的SqlSessionFactoryBean，然后当我们调用mybaties的数据库操作方法时，其实就是在使用SqlSessionTemplate进行操作，也就是使用了对应的数据源连接池在进行背后的操作。

另外在该文件中还指明了mybaties的扫描的接口包的位置。

至于事务管理，其实跟mybatis没有关系，事故管理跟数据源有关，指明那个数据源使用spring的什么事务管理器进行管理，管理时是xml管理还是注解管理，这些配置其实可以单放，不一定要放在mybatis配置文件中，可能只是一种省事的习惯吧，都是数据库相关的，所以就放一块了。

另外，这里除了指明SqlSessionFactoryBean注入的数据源的属性之外，还进一步指明了SqlSessionFactoryBean的配置依赖，通过引入外部mybatis的xml配置文件的方式，来注入这些mybaties的配置属性。

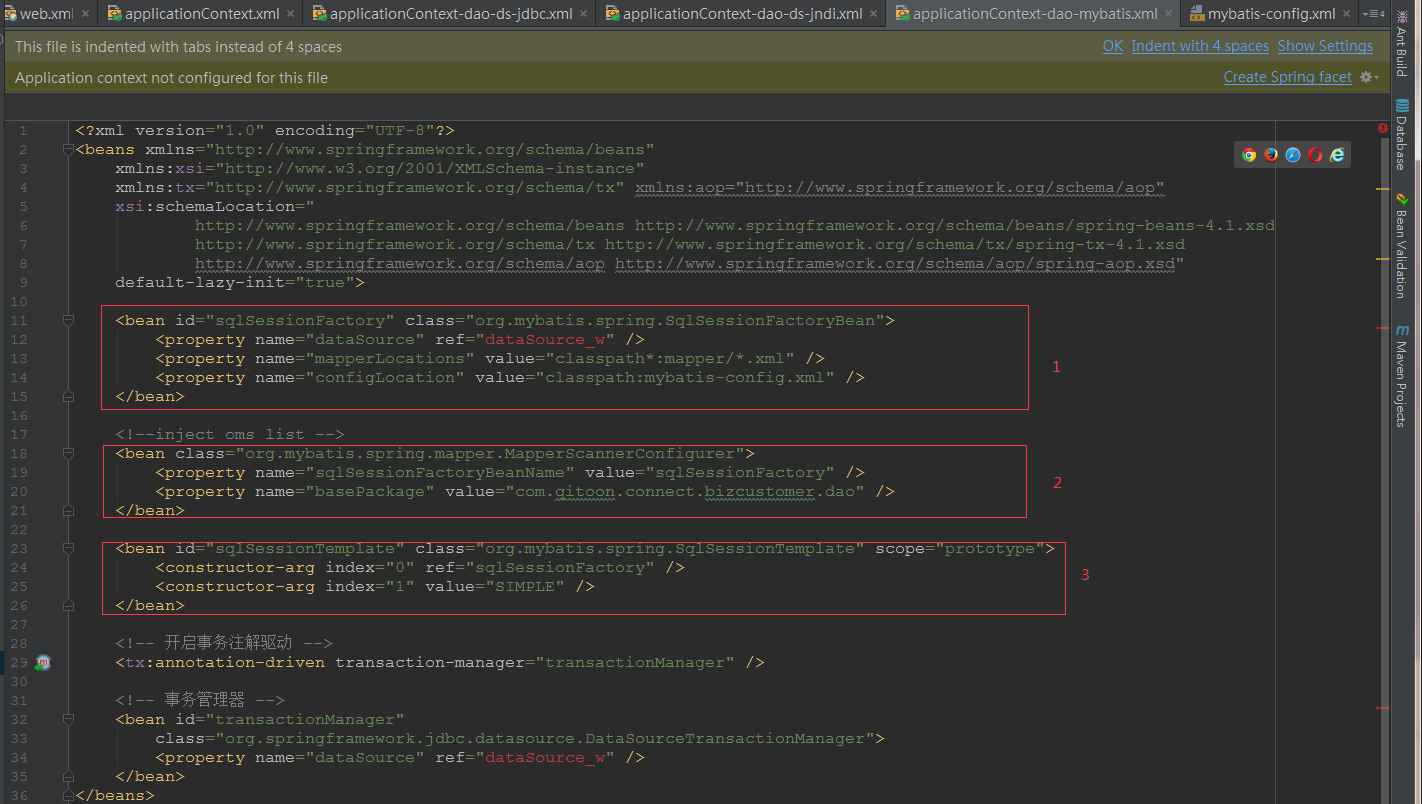
另外，这里还指明了SqlSessionFactoryBean的扫描mapper.xml的路径。这些都是mybatis工作所不可或缺的。

你想了，一个连接数据库的工具要想工作，他都要知道什么呀，肯定要知道我的数据库连接从哪里拿，我自己配置了哪些特性，如果做到一些自动映射，还要知道哪些类哪些方法是需要我去动态注入实现类和实现方法的，注入实现类的时候，我去那找我的sql语句去进行动态实现。

mybatis主流有两种工作方式。

1.配置自动映射，只要指明dao层接口类的扫描位置，mybaties自动进行扫描映射对应的mapper.xml进行方法的调用，下图中1和2配合即可。

2.指定sqlSessionTemplate，然后自己写dao层接口类的实现类，在实现类中使用sqlSessionTemplate对象调用mybatis的api来进行数据库操作，一般读写分离都是使用这种方式，比较推荐。即下图中的1和3配合即可。这个sqlSessionTemplate对象可以在这里配置多个，比如一个读一个写。



5.mybatis属性配置文件。

