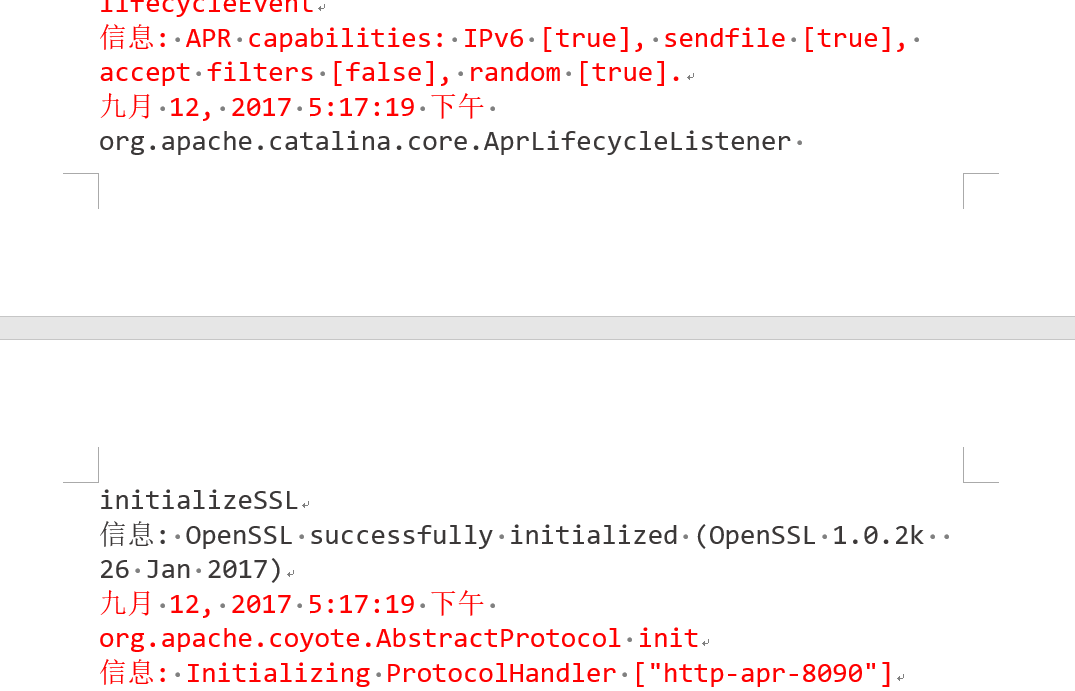
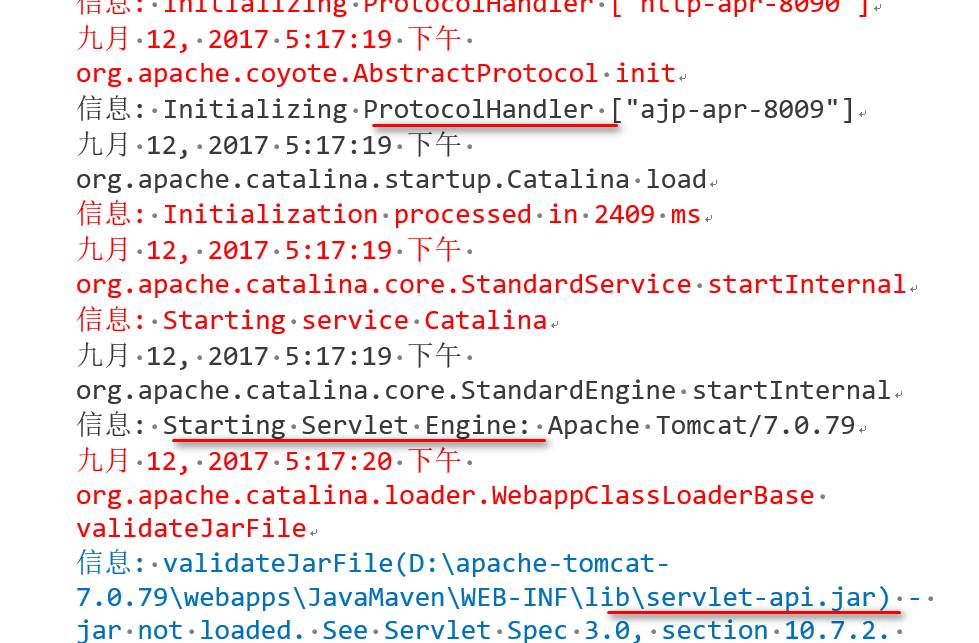
1. tomcat启动的时候首先启动 tomcat的核心服务,一个监听器

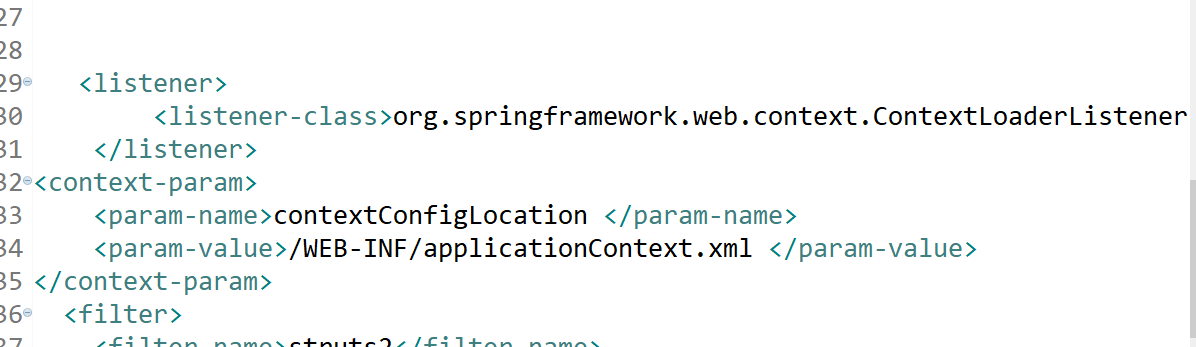
并且初始化ssl服务,说明可以远程连接到tomcat服务器,存在安全验证的问题



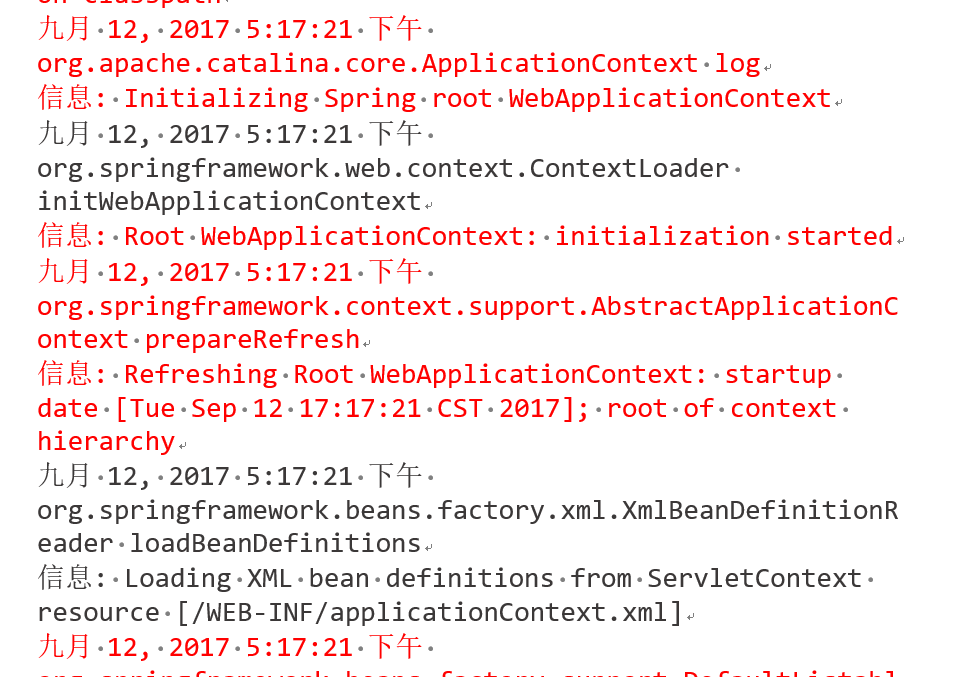
1. 当核心监听服务初始化完成后,然后初始化 协议控制器,表明tomcat可以接受什么协议的访问,如http,tls等,当协议初始化完成后,开始初始化 servlet,engine,初始化servlet的时候,会自动调用tomcats的servlet-api.api这个包,如果网页程序引入的servlet.api这个包和服务器的相同名字的包不匹配,那么可能导致冲突.

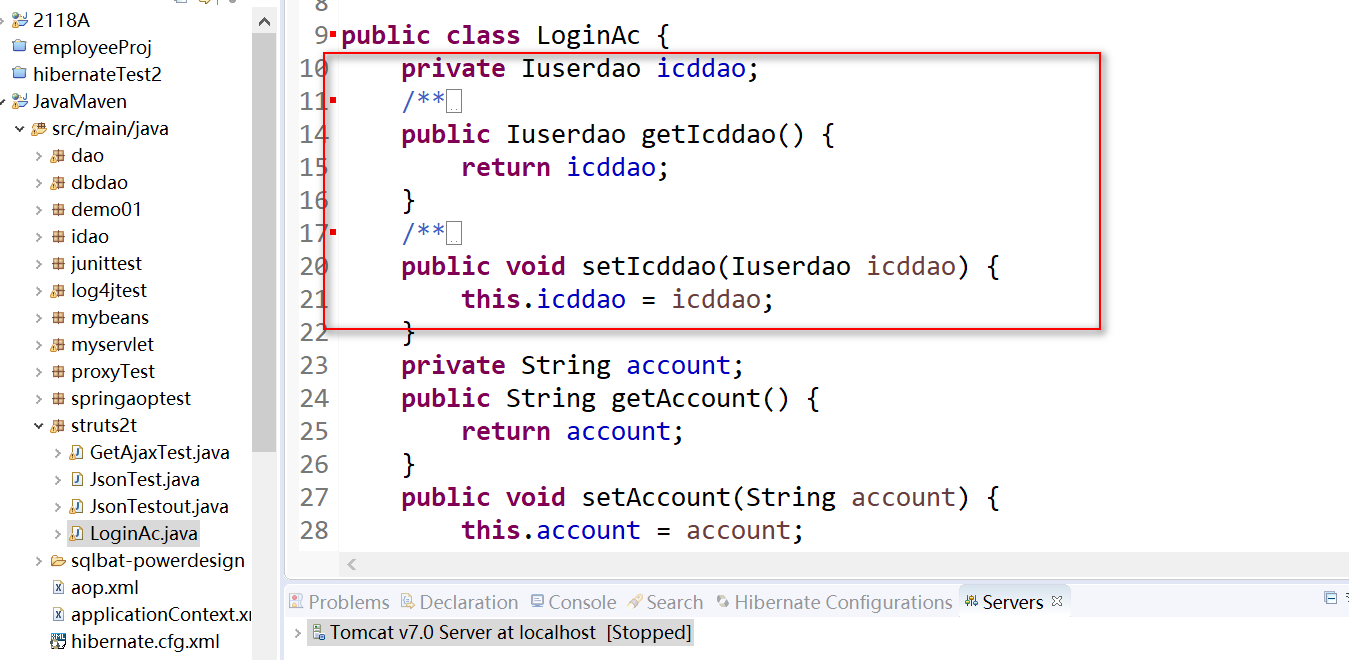


1. 开始初始化springframework-context这对象,这个对象默认会读取 web-inf/applicationContext.xml这个配置文件,当然也可以在web.xml,这个配置文件的位置在web.xml中进行设置

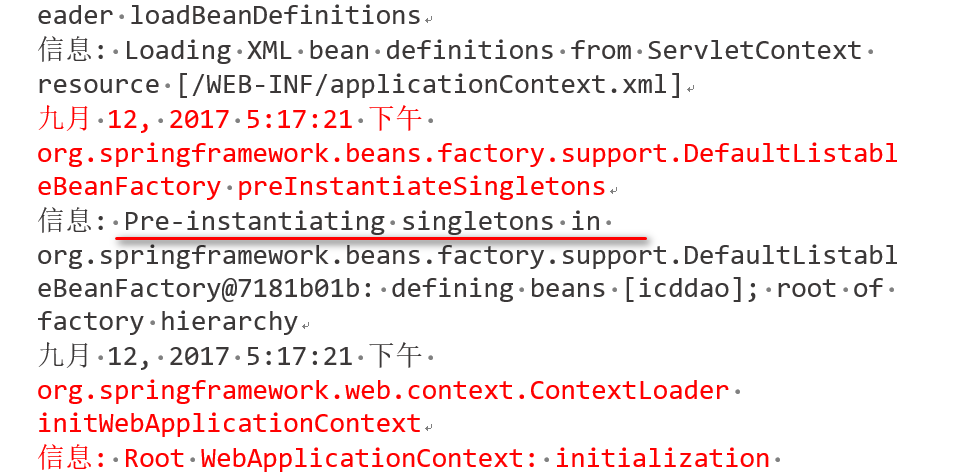


Springframe-context对象使用applicationcontext.xml这个文件,将相应的bean的类使用classloader放置到内存中,为下一步,structs集成spring中,使用 applicationcontext.xml中的bean定义做了铺垫

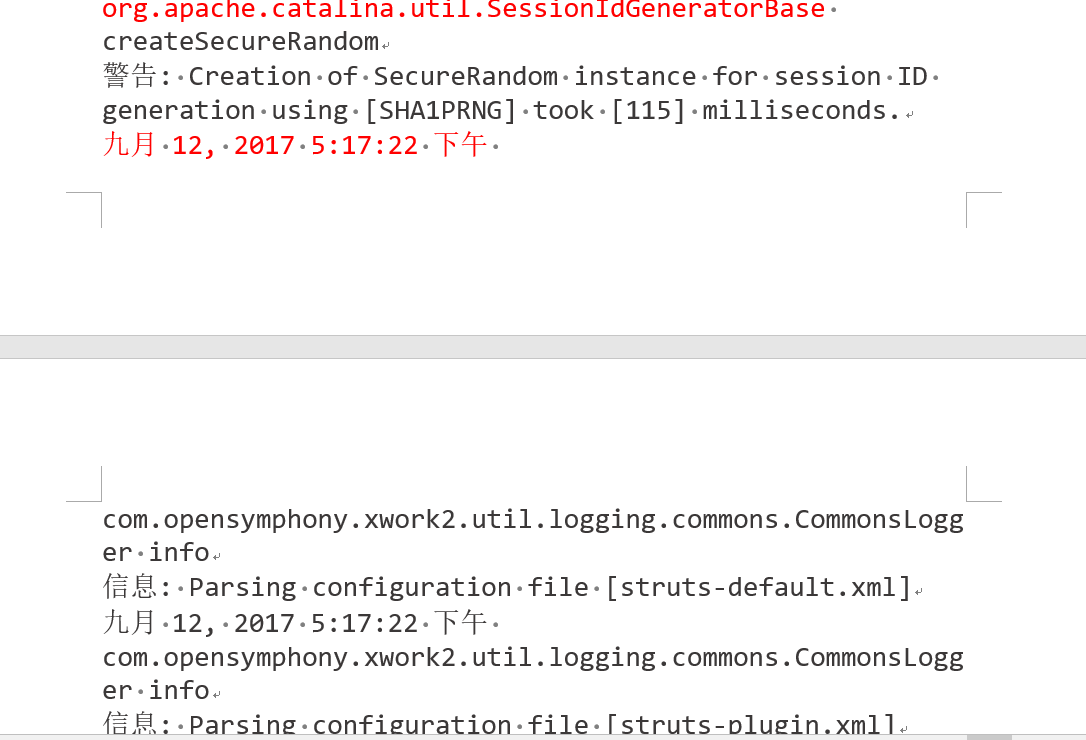




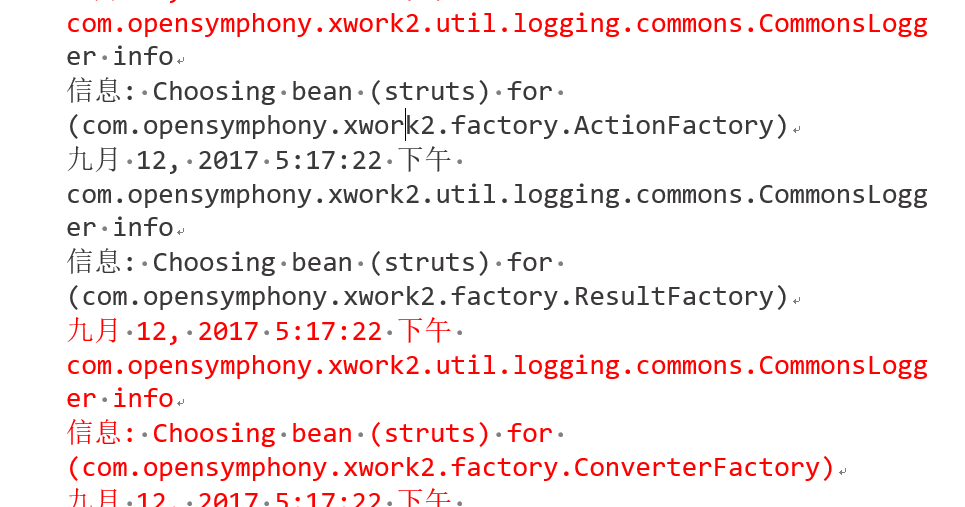
Spring-context对象使用单利模式创建bean



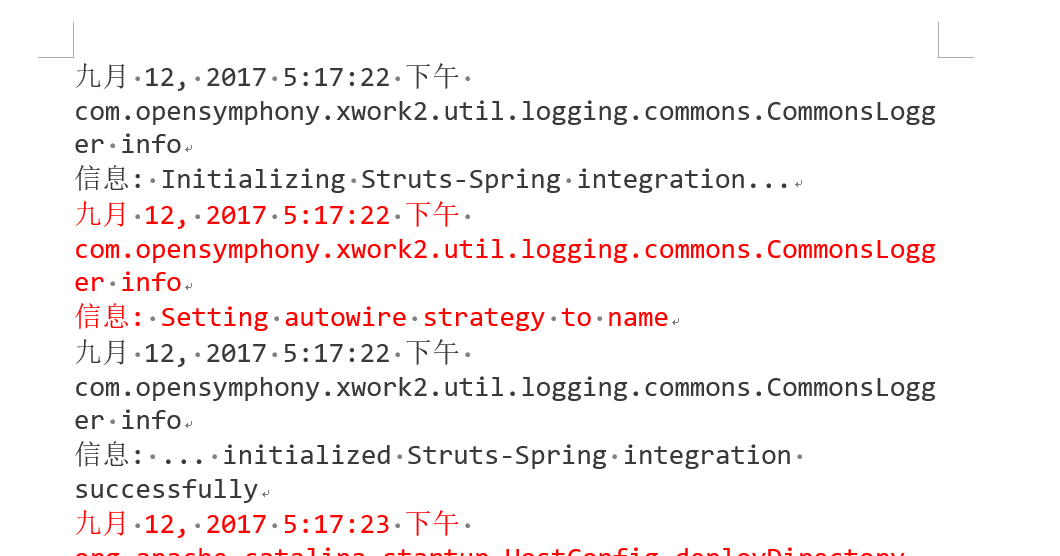
1. 当spring-context对象,加载完bean之后,便开始初始化struct框架了,这里值得注意的是使用一个sessionid作为spring和struct相关联的安全机制



1. struct在初始化的使用工厂模式,创建了一些列的流程的工厂,在不同的场景中,只要使用工厂创建不同的对象即可产生不同的控制效果

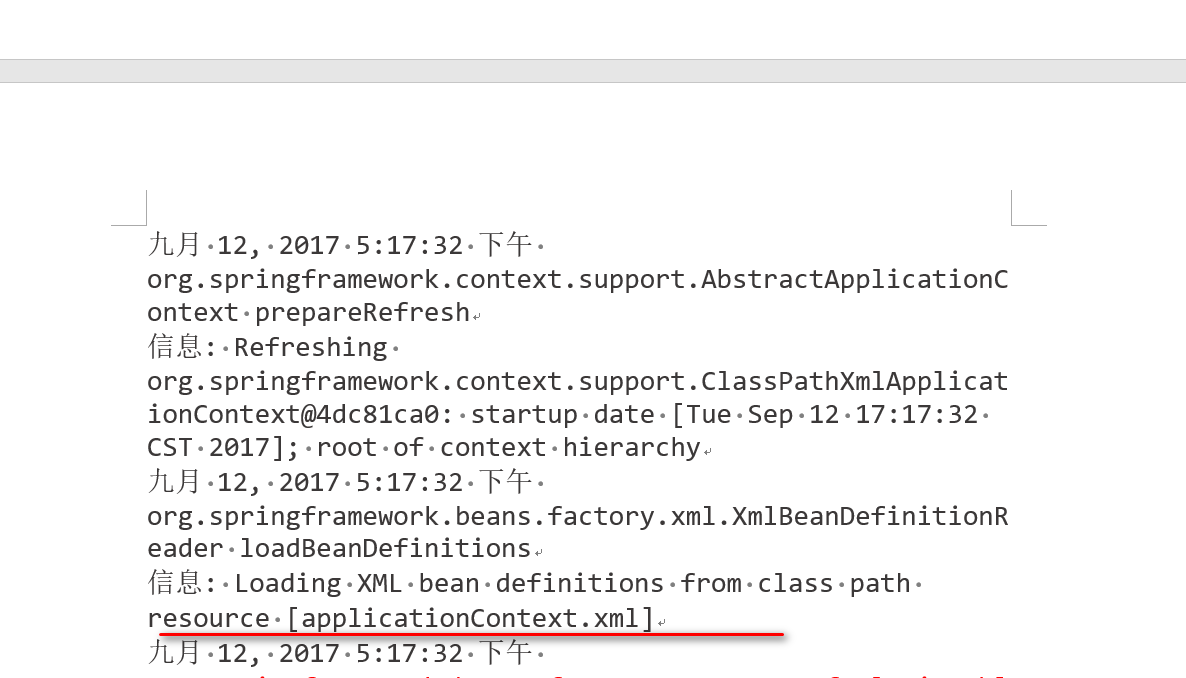


1. structs初始化完成后便开始和spring集成,注意的是 structs集成spring而不是spring集成struct

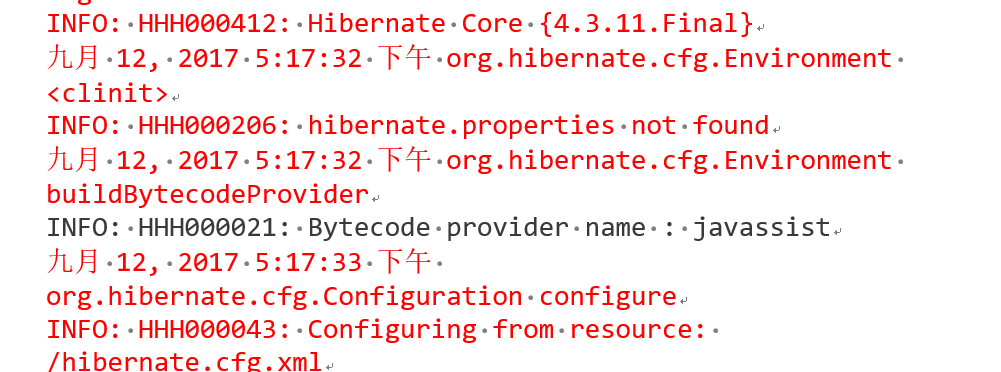


1. 集成完成后,开始初始化web程序,如果web程序中需要读取resource文件,如application.xml,那么会在这里开始读取

这里的applicationContext.xml和上文中的不是一个文件,是手动引入的,这样的好处是,一些bean是公共的可以直接初始化,但是一些bean如果值想被某些类共享,那么可以放在这个xml定义



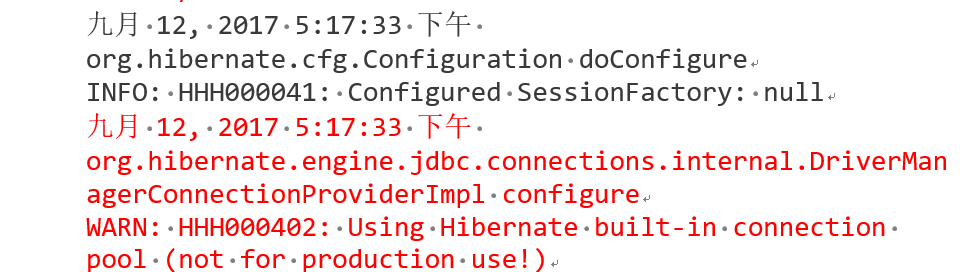
1. spring和struct的工作完成后,开始初始化hibernate



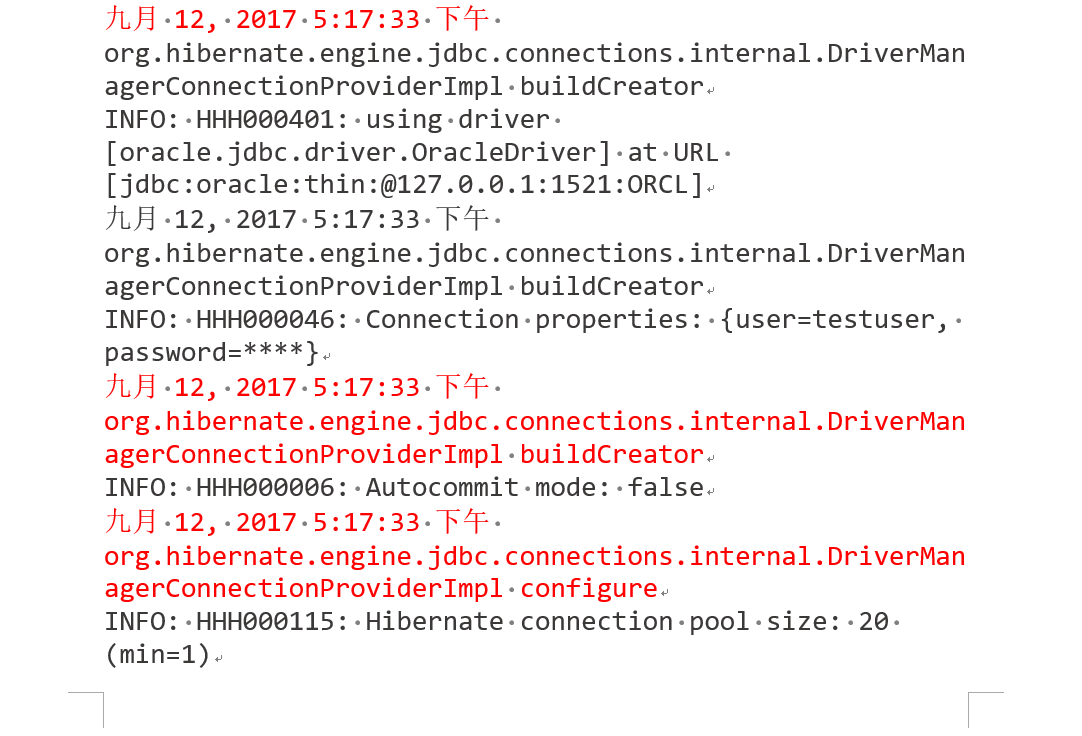
根据hibernate.cfg.xml中的mapping设置关联映射类



初始化数据库连接



Hibernate初始化数据库连接默认的设置,可以看到数据库的位置,提交模式和默认的连接池里面线程的数量



1. hibenate 自动生成sql语句查询数据库

