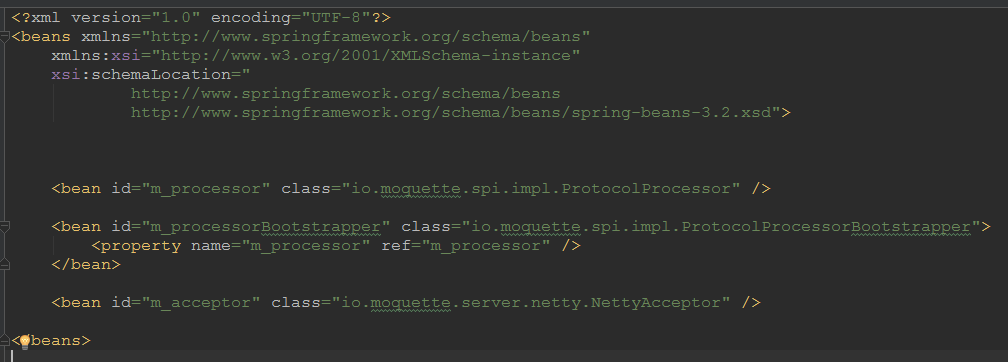
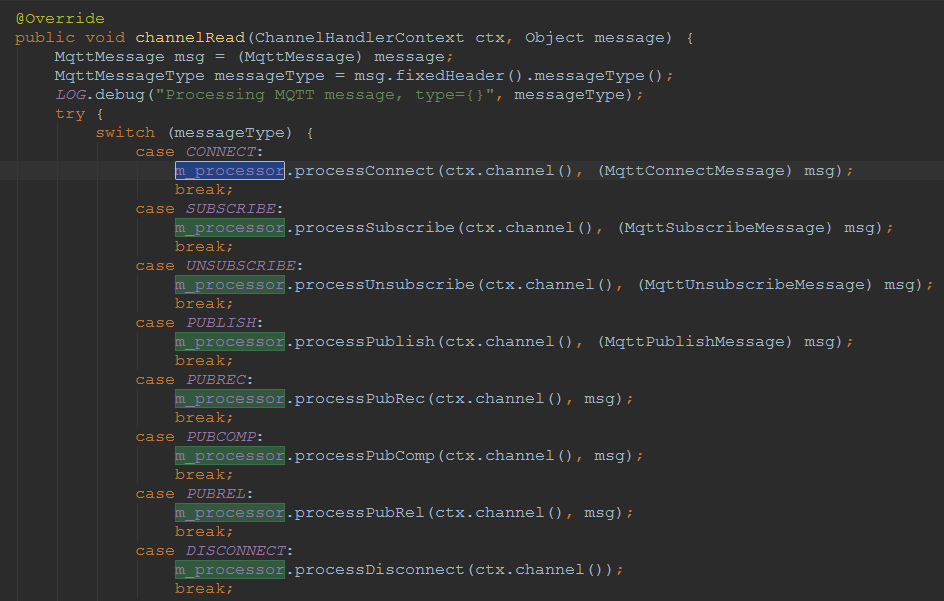
1. 程序入口类：io.moquette.server.Server

整个代码流程在培训的时候已经全部讲了一遍，当时也有讨论，这里就不展开分析，着重关注我们需要修改的地方。

1. 几个重要的全局类实例，我已经全部抽取在distribution\src\main\resources\spring-bean.xml里面，在需要使用的地方直接通过Spring的getBean去获取资源即可



1. 框架处理MQTT消息类：io.moquette.server.netty.NettyMQTTHandler



图中选中的m\_processor已经在第2点已经全局化了，在任意地方都可以引用，从图中我们可以看到处理所有的MQTT消息，如设备连接过来有CONNETCT消息，设备发送消息后会有PUBLISH消息，设备订阅的群组会有SUBSCRIBE，有了这些消息我们就可以做很多事情，比如想控制发言权限，那我们就可以在PUBLISH那进行拦截，判断这条消息的发布者是否权限发言，如有则继续；如没有就直接return返回，即达到发言权限控制的逻辑。

1. 培训所说的两大全局数据结构类实例

1）记录所有设备的连接关系：io.moquette.spi.impl.ProtocolProcessor的ConnectionDescriptorStore connectionDescriptors，注意ProtocolProcessor我也已经全局化了，想使用的时候随时可以引用。

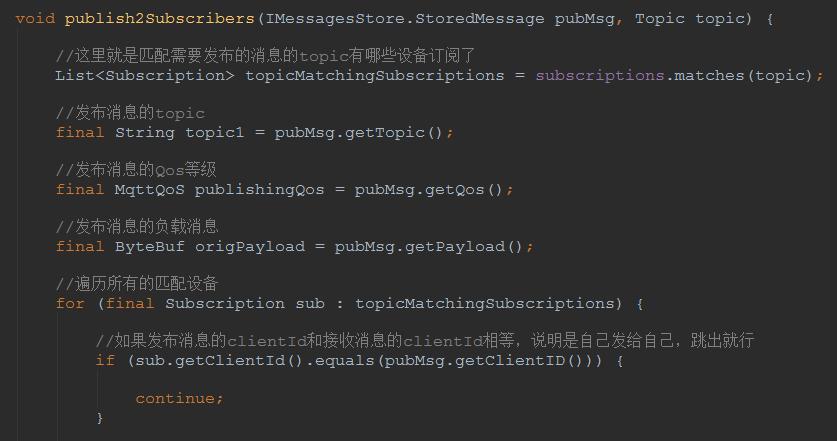
这个connectionDescriptors里面是保存着所有设备的连接关系，这个结构体跟我们后面的业务有关系，比如强制下线，统计在线的设备数；强制下线我们只要把对应设备的连接链路给关掉，设备自然就下线了，统计在线设备数编译connectionDescriptors整个结构体就可以做到。

2）记录订阅群组以及某一群组订阅的设备clientId：io.moquette.spi.impl.ProtocolProcessor的SubscriptionsDirectory subscriptions，该subscriptions记录了群组及成员的关系，如某一群组有多少设备订阅，分别是哪些设备。比如我们需要群组拉人和踢人，只要在对应的群组以成员的关系二叉树上面做对应的删除和增加就行，统计群组数也就是编译整个群组结构体就可以拿到。

5、对讲机需改动的点

1. 自发自收

io.moquette.spi.impl.MessagesPublisher



如注释所示，添加后即可去掉自发自收功能

1. 强制下线：设备连接关系结构体
2. 组拆群组：群组订阅关系结构体
3. 群组拉人踢人：群组订阅关系结构体
4. 发言权限控制：关注MQTT消息中的PUBLISH消息
5. 上线下线通知

只要MQTT消息中的CONNECT和DISCONNECT消息的数据统计

1. 在线数据统计：设备数、群组数：修改第4点的两大结构体

6、在Moquette框架判断逻辑关系尽量使用缓存，如Redis，不建议直接读取数据库，主要是出于效率的因素，缓存和数据库之间用同步机制进行同步更新缓存数据。