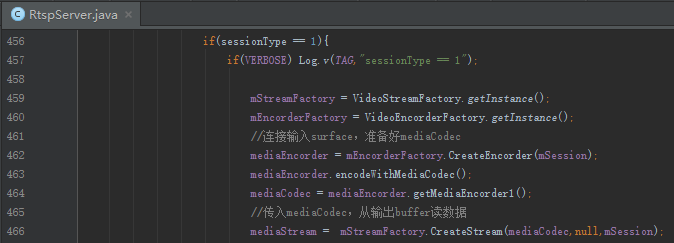
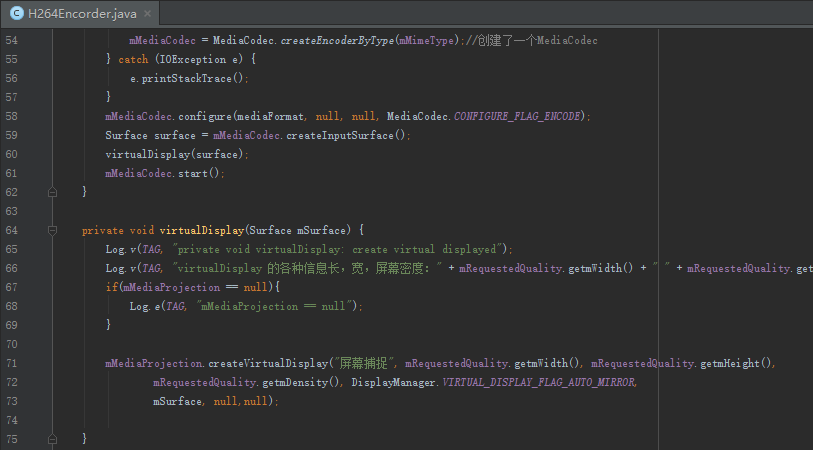
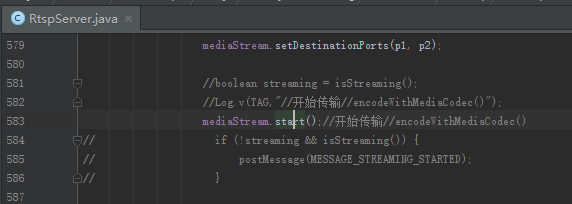
准备各种类，互相引用。



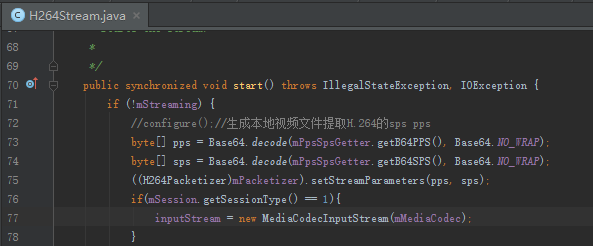
其中将mediacodec与mediaprojection关联，当程序启动时，屏幕数据就能送到meidacodec中编码。



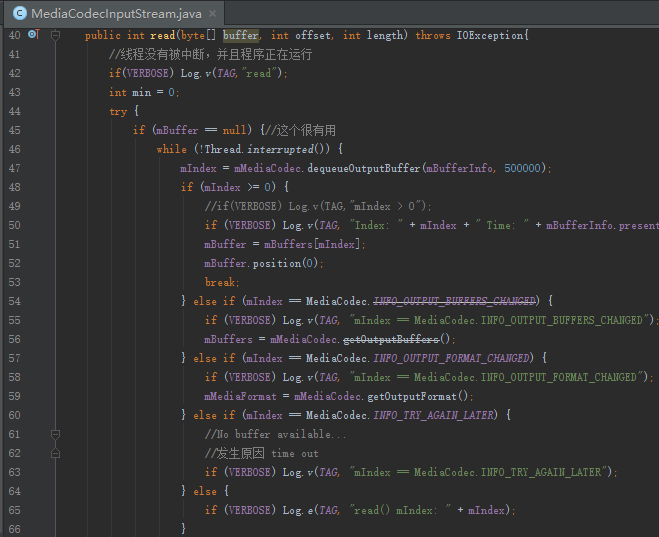
当程序执行到这时，数据开始传输。



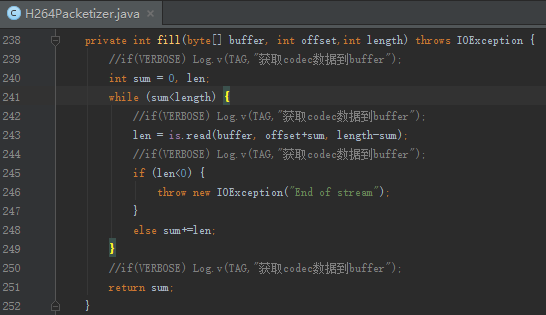
Start的功能：将之前配置好的mediacodec（此时，mediacodec已经获取到mediaprojection中的数据）传给MediaCodecInputStream类中。



MediaCodecInputStream的read方法的功能：从mediacodec中读取视频数据。

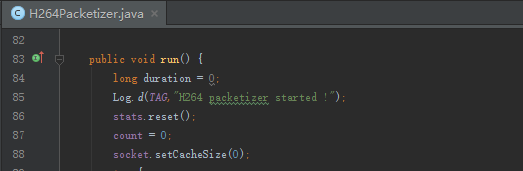


Fill方法调用了read。把stream中的数据存入数组



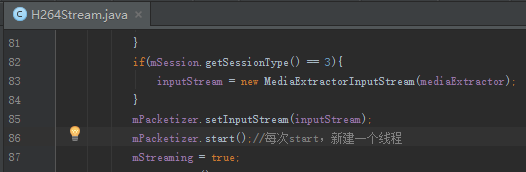
Fill由当前文件中的send方法调用，send由当前文件的run方法调用。

run方法的实现如下图。



H264Stream中的Start方法（第70行的start）启动了上面的run()（第86行的start）；

H264Stream是mediastream的子类。所以这里是真正的开始传输数据的地方。



上图中开启了线程。在这个线程中作为一个生产者线程。

H264Packetizer与rtpsocket是生产者-消费者的关系。

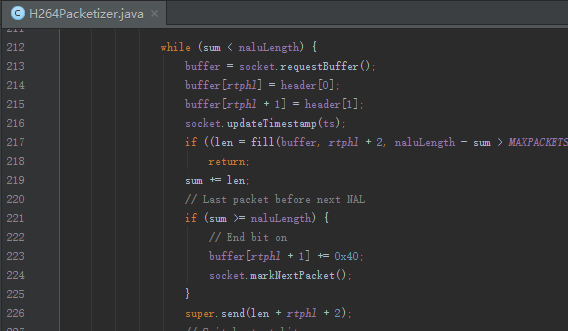
Rtpsocket是消费者线程。

Rtpsocket是最底层的数据传输了，使用UDP的方式，下图中的socket就是。

Rtpsocket里面有许多数组，构成环形数组。使用信号量控制同步。

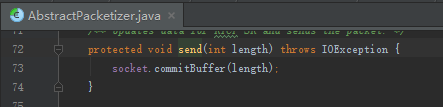
每次H264Packetizer从rtpsocket获取一个数组，向里填入数据。

Rtpsocket每次从自己的众多数组中选择一个数组，开启一个线程发送这个数组。Super.send用于发送。

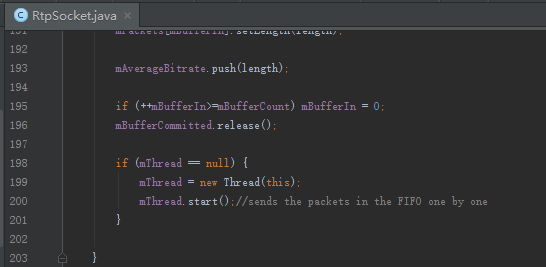


上图中requestBuffer获取rtpsocket类中环形缓存的数组。

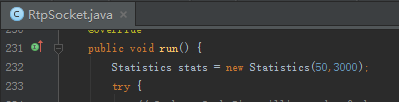
Super.send（）实现如下。

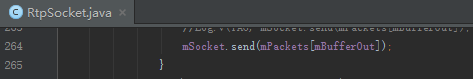


CommitBuffer实现如下，开启了一个线程。



线程的具体内容





mSocket为MultiCastSocket。Udp发送了数据。