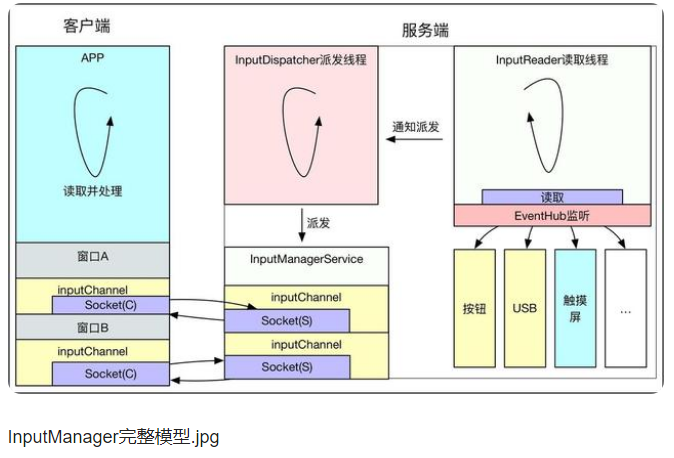
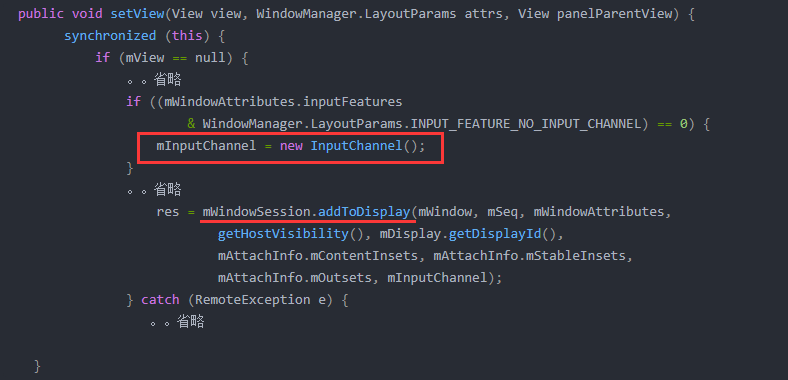
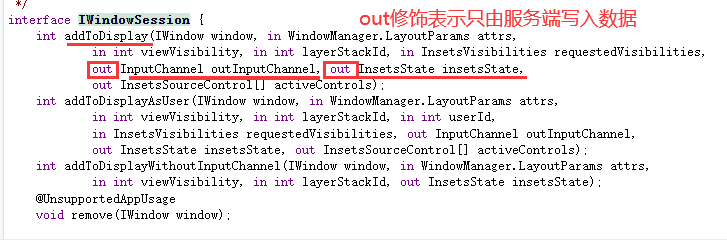
**android系统中的socket通信案例分析**



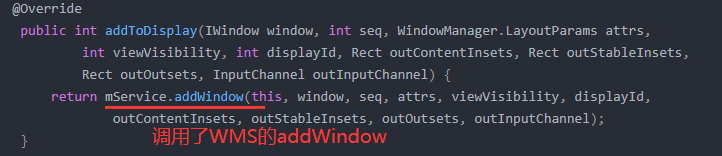
**直接看触摸事件相关源码**

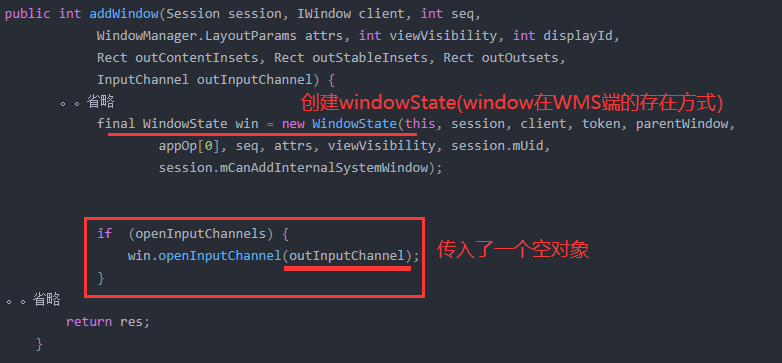


**查看该Binder接口的声明：可以看到 inputChannel 对象是由S端写入的(WMS),就是由服务端来填充**



**继续看addToDisplay：**

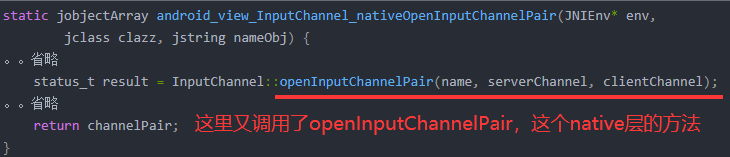




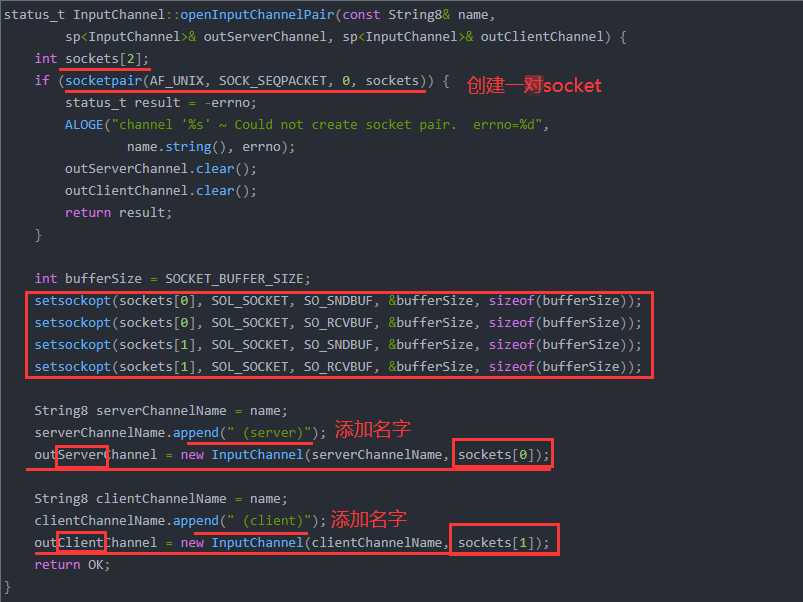


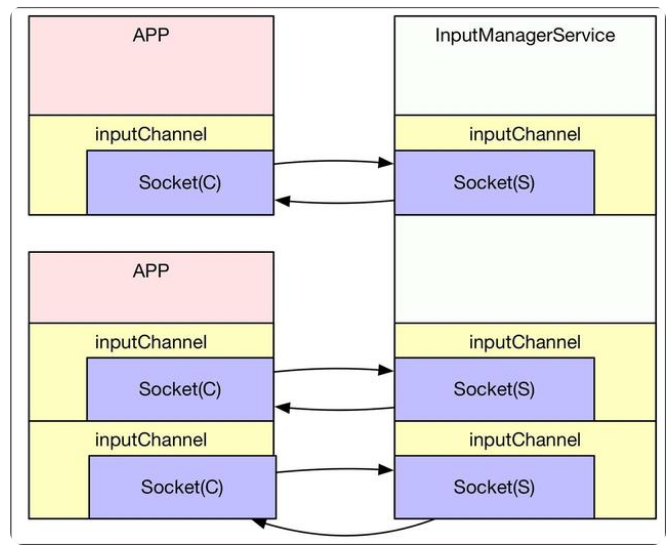
**这里应该就是将S端的inputChannel注册到inputDispatcher中，将C端的inputChannel传递给了客户端应用**

**查看openInputChannelPair:**

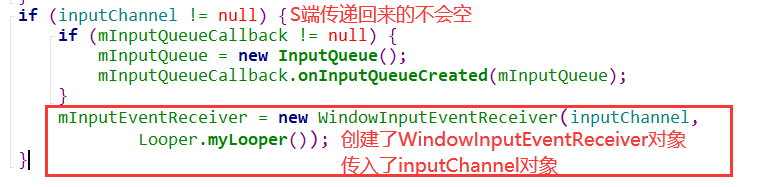


**InputChannel:openInputChannelPair: native层**



**outClientChannel最后通过IPC通信传递给了客户端**

**接着看客户端拿到了这个inputChannel之后做了什么：**



**WindowInputEventReceiver**





**nativeInit 接收一个long类型的返回值:**

clipboard.png

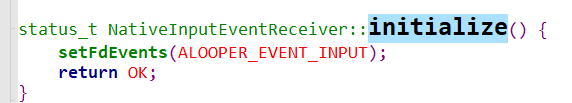
**查看C++代码：nativeInit:**



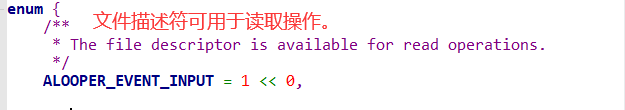
**NativeInputEventReceiver构造方法：**



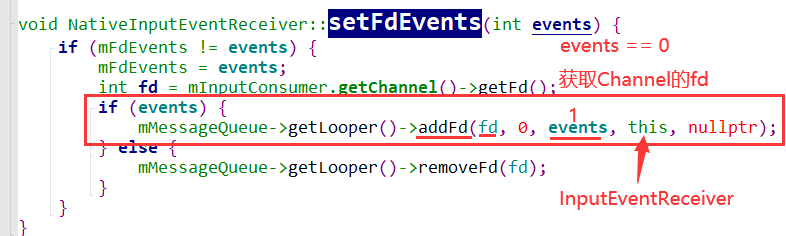
**initialize:**



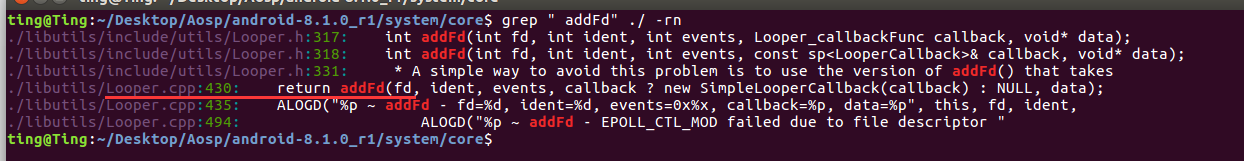
**setFdEvents 传入的参数是ALOOPER\_EVENT\_INPUT 有点像无限循环的标志。**



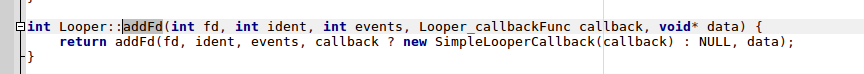
**可以看到ALOOPER\_EVENT\_INPUT就是等于1**



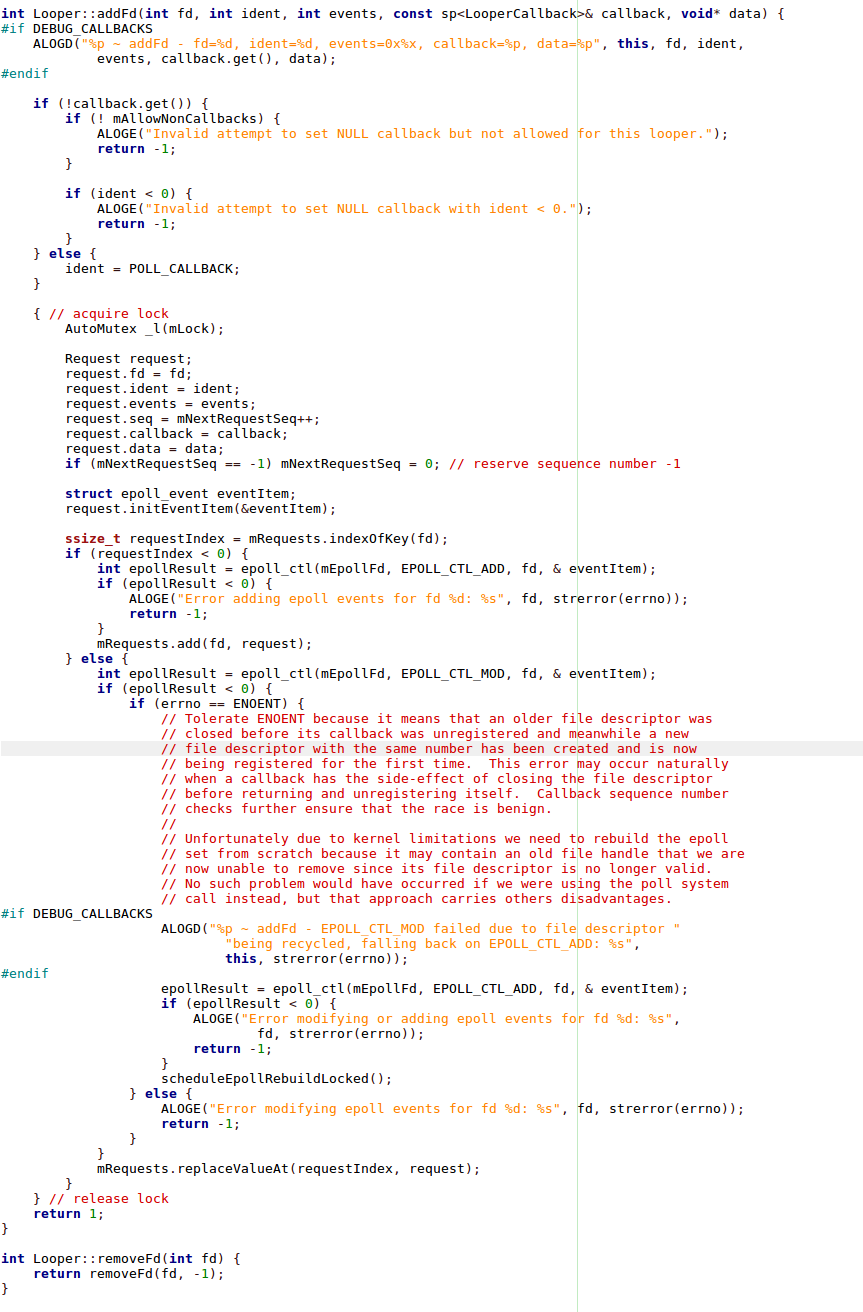
**查看addFd:**



**addFd**



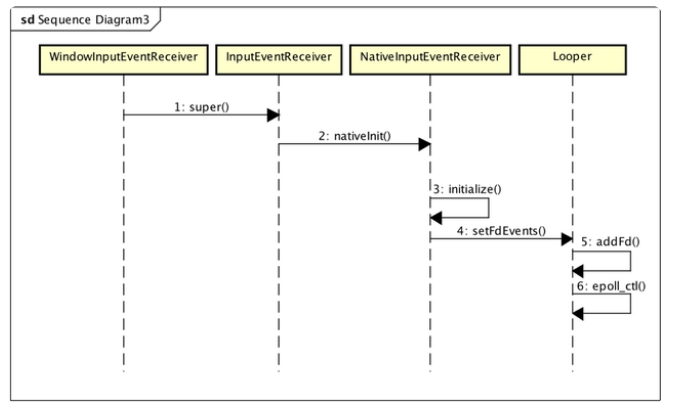
**addFd**



**可以看到该方法就是对我们传入进来的fd进行的epoll的监听操作**

**也就是initialize方法就是加了将fd添加到了epoll中进行监听，如果监听到了消息，就将其回调传递给客户都安的主线程进行处理**

**总结：**



**这里也看出来了其实native层的Loop本质也是用了epoll来实现的。**

**https://www.baidu.com/link?url=M4qwL93y7k1XmLjCbW67unFLgr-YtsAY16ZlfhYsB9UoQtAg-4qk1r2XsmAAcZKsezi0bkZVbuq\_aM8KsIhikFbgbQW92sOmX7KX\_tjbEHG&wd=&eqid=f8f046d3000051750000000662ea3fc4**