https://blog.csdn.net/z69183787/article/details/52621417

**Netty 之 Netty心跳之IdleStateHandler**

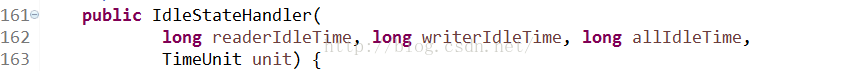
2016年09月22日 11:50:43 [OkidoGreen](https://me.csdn.net/z69183787" \t "_blank) 阅读数：13447

Netty提供了对心跳机制的天然支持，心跳可以检测远程端是否存活，或者活跃

今天我们就一起初识一下Netty4的心跳机制

Netty4.0提供了一个类，名为IdleStateHandler，这个类可以对三种类型的[心跳检测](https://www.baidu.com/s?wd=%E5%BF%83%E8%B7%B3%E6%A3%80%E6%B5%8B&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd" \t "_blank)

这个类的构造参数是这样的：



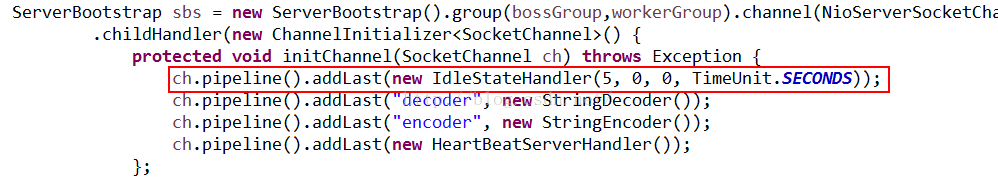
前三个的参数解释如下：

1）readerIdleTime：为读超时时间（即测试端一定时间内未接受到被测试端消息）

2）writerIdleTime：为写超时时间（即测试端一定时间内向被测试端发送消息）

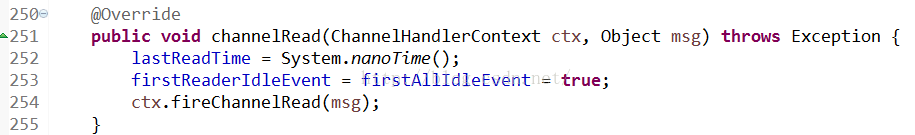
3）allIdleTime：所有类型的超时时间

这个类主要也是一个ChannelHandler，也需要被载入到ChannelPipeline中，加入我们在服务器端的ChannelInitializer中加入如下的代码：



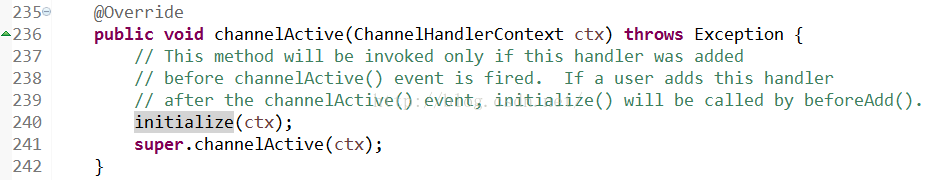
我们在channel链中加入了IdleSateHandler，第一个参数是5，单位是秒，那么这样做的意思就是：在服务器端会每隔5秒来检查一下channelRead方法被调用的情况，如果在5秒内该链上的channelRead方法都没有被触发，就会调用userEventTriggered方法：

初步地看下IdleStateHandler源码，先看下IdleStateHandler中的channelRead方法：



请注意254行代码其实表示该方法只是进行了透传，不做任何业务逻辑处理，让channelPipe中的下一个handler处理channelRead方法，但是记录了一下这里的调用时间

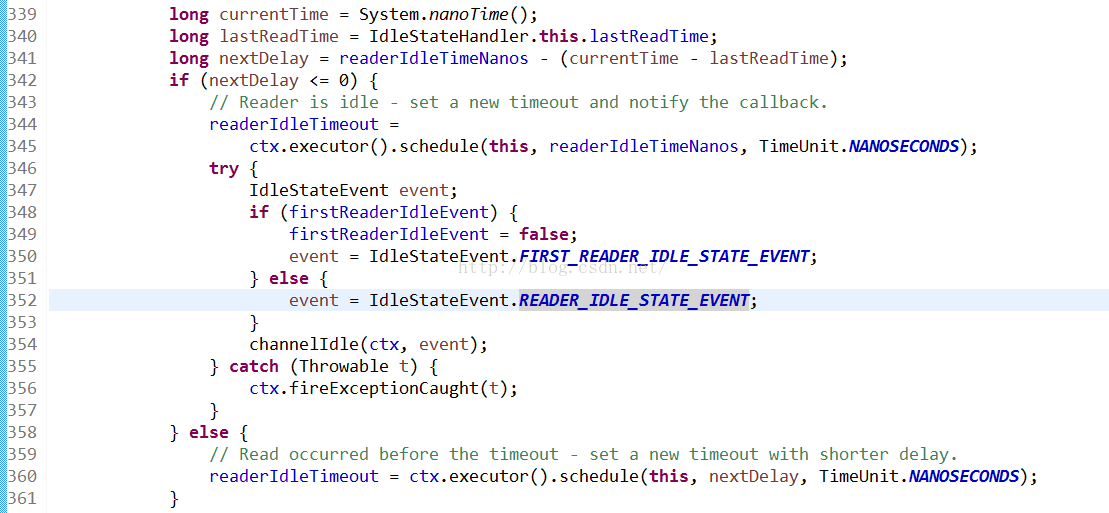
我们再看看channelActive方法：



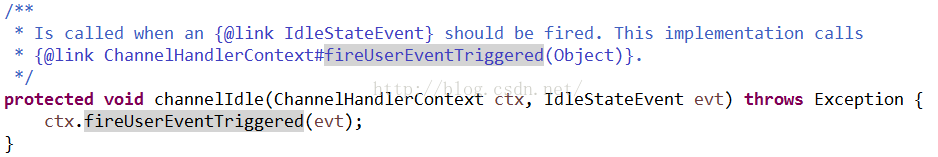
这里有个initialize的方法，这是IdleStateHandler的精髓，接着探究：



这边会触发一个Task，ReaderIdleTimeoutTask，这个task是部分源码是这样的：



341行是这样的，用当前时间减去最后一次channelRead方法调用的时间，假如这个结果是6s，说明最后一次调用channelRead已经是6s之前的事情了，你设置的是5s，那么nextDelay则为-1，说明超时了，那么354行则会触发userEventTriggered方法：



如果没有超时则不触发userEventTriggered方法

初略地看下就是这么多了，这就是IdleStateHandler的基本原理了

简而言之：

IdleStateHandler这个类会根据你设置的超时参数的类型和值，循环去检测channelRead和write方法多久没有被调用了，如果这个时间超过了你设置的值，那么就会触发对应的事件，read触发read，write触发write，all触发all

如果超时了，则会调用userEventTriggered方法，且会告诉你超时的类型

如果没有超时，则会循环定时检测，除非你将IdleStateHandler移除Pipeline