LicenseHandler

Introduction

一个适用Netty的，用于限制流量的handler

Installation

`maven`

```
<dependency>
    <groupId>tongji.sse.bookish-meme</groupId>
    <artifactId>license-handler</artifactId>
    <version>1.0.0</version>
</dependency>
```

Usage

定义规则，即决定哪些是消息是需要这个handler去响应的，哪些是不需要这个handler响应的

```
MessageFilter<AnyTypeObject> messageFilter = new MessageFilter<AnyTypeObject>() {
    //定义这个对象是否应该响应
    @Override
    public Boolean shouldFilter(AnyTypeObject obj) {
        if(obj should be filte) return true;
        else return false;
    }
};
```

定义需要使用到的limiter类型

```
int maxCount = 5;
TZLicense tzLicense = new SumCountRtLimiter(maxCount);
```

构造limitHandler

```
LicenseHandler<AnyTypeObject> licenseHandler = new LicenseHandler<AnyTypeObject>(m
essageFilter, tzLicense) {
    //当该类型的消息被响应且被许可时
    @Override
    public void messageAgree(AnyTypeObject msg) {
        //做你的业务逻辑
    }
          //当该类型的消息被响应且不被许可时
    @Override
    public void messageDisagree(AnyTypeObject msg) {
        //做你的业务逻辑
    }
          //当该类型的消息不被响应
    @Override
    public void messageIgnore(AnyTypeObject msg) {
        //做你的业务逻辑
    }
};
```

加入到Netty的pipeline中

```
public class ChatServerInitializer extends ChannelInitializer<SocketChannel> {

    @Override
    protected void initChannel(SocketChannel socketChannel) throws Exception {
        ChannelPipeline pipeline = socketChannel.pipeline();
        pipeline.addLast("framer", new DelimiterBasedFrameDecoder(8192, Delimiters
.lineDelimiter()))
                .addLast("decoder", new StringDecoder())
                .addLast("encoder", new StringEncoder())
                .addLast("limit",new licenseHandler());
    }

}
```