Lab 5.2: Adding a clean shutdown to our producer

Welcome to the session 5 lab 2. The work for this lab is done in `~/kafka-training/labs/lab5.2`. In this lab, you are going to create a clean shutdown for our advanced Producer.

Lab Adding an orderly shutdown flush and close

Shutdown Producer Nicely

Let's write some code to shut the Producer down nicely.

The shutdown code will happen if you are running the producer example from a terminal and type ctrl-C so shutdown from Java occurs. We will write some code to shutdown thread pool and wait. Then we will flush the producer to send any outstanding batches if using batches (producer.flush()). Lastly, we will close the producer using producer.close and wait five seconds for the producer to shutdown. (Note that closing the producer also flushes it.)

To this we will add shutdown hook to Java runtime. Then to test we will start the StockPriceKafkaProducer, and then you can stop it using CTRL-C or by pressing the stop button in your IDE.

~/kafka-training/labs/lab5.2/src/main/java/com/fenago/kafka/producer/StockPriceKafkaProducer.java

Kafka Producer: StockPriceKafkaProducer shutdown hook for clean shutdown

```
public class StockPriceKafkaProducer {

    ...

    private static final Logger logger =
LoggerFactory.getLogger(StockPriceKafkaProducer.class);



    public static void main(String... args) throws Exception {
        //Create Kafka Producer
        final Producer<String, StockPrice> producer = createProducer();
        //Create StockSender list
        final List<StockSender> stockSenders = getStockSenderList(producer);

        //Create a thread pool so every stock sender gets it own.
        // Increase by 1 to fit metrics.
        final ExecutorService executorService =
                Executors.newFixedThreadPool(stockSenders.size() );

        //Run each stock sender in its own thread.
        stockSenders.forEach(executorService::submit);


        //Register nice shutdown of thread pool, then flush and close producer.
        Runtime.getRuntime().addShutdownHook(new Thread(() -> {
            executorService.shutdown();
```

```
            try {
                executorService.awaitTermination(200, TimeUnit.MILLISECONDS);
                logger.info("Flushing and closing producer");
                producer.flush();
                producer.close(10_000, TimeUnit.MILLISECONDS);
            } catch (InterruptedException e) {
                logger.warn("shutting down", e);
            }
        }));
    }

...
}
```

Notice we add a shutdown hook using Runtime.getRuntime().addShutdownHook and this shutdown hook that shuts down the thread pool, then calls flush on the producer and then closes the producer whilst waiting 10 seconds for the close to happen.

ACTION - EDIT src/main/java/com/fenago/kafka/producer/StockPriceKafkaProducer.java and add a shutdown hook to main.

ACTION - RUN this StockPriceKafkaProducer and try shutting it down.