Case Study: Java Engineer

Provided

Together with this document we will provide a Exercise.zip file containing an eclipse project. This project is the basis for the task.

Expectation

Implement the task as described below. You can start whenever you like and send us the result once you are done.
Add documentation like you would in a real project. Bring the code to a production standard (not more, not less).
Do not spend more than 2 hours on this exercise.

Return

After you have finished the task, send the code back to the provided e-mail. Please tell us how much time you have spent on the task. Also provide any external library that you have used.

Scenario

There is a producer (Producer) of price updates for stocks.
This producer will generate constant price updates for a fix number of stocks.
The producer should not block, every price update should be consumed as quickly as possible.

Furthermore there is a load handler (LoadHandler) which will consume the price updates of the producer.
In the current implementation the load handler is just passing on the update to a consumer. This should be changed.

The consumer (Consumer) will receive the price updates from the load handler.
(The current implementation will just print out all price updates for convenience sake.)
The consumer is representing a legacy system that cannot consumer more than a certain number of price updates per second. Otherwise it would fall over.

Task

The task of this exercise is to extend the LoadHandler to limit the updates per second to the consumer to a certain given number (MAX_PRICE_UPDATES).
In order to achieve this, it is allowed to drop price updates, since otherwise the application will run out of memory, if the application will keep all of them.
It is important that, if a price update will be send to the consumer, it has to be the most recent price.

Example:

Updates arrive in this order from the consumer:

```
Apple - 97.85
Google - 160.71
Facebook - 91.66
Google - 160.73
```

The load balancer has received all updates and is going to send them out to the consumer like this now:

```
Apple - 97.85
Google - 160.73
Facebook - 91.66
```

So the first Google update (160.71) has been dropped.

In order to limit the number of updates per second to the consumer, it will be necessary to write some sort of scheduler/timer. It is acceptable to send the updates as bulk once per second.
Ideally the load should be spread out into smaller chunks during that second.

Please consider that the number of stocks might be bigger than the number of allowed updates per second to the consumer.
Make sure that the application will not run out of memory, even if the number of stocks or updates per second might be bigger than MAX_PRICE_UPDATES.

Please implement the *hashCode* and *equals* in PriceUpdate,
since those methods might be relevant for the task.

It is fine to create additional classes and tests.

You can use all features of Java 8 as well as any additional library as long as it is open source and will be provided with the solution.