**Домашнее задание по 4 занятию Java.**

Создайте новый проект, используя для сборки либо maven, либо gradle.

Добавьте в качестве зависимости проекта dummy-connector.jar. Напоминаю, что для того, чтобы иметь возможность получить эту зависимость из локального mvn-репозитория, вам нужно сделать maven install для этой библиотеки.

Теперь вы можете использовать библиотеку, которую я подготовил. Эта библиотека приблизительно эмулирует поведение клиентской библиотеки ActiveMQ, позволяющей вам подключаться к очередям сообщений и отправлять и принимать из них сообщения.

Вы можете исследовать библиотеку самостоятельно, начиная с интерфейса Connection.

Говоря просто, вам нужен объект типа Connection, чтобы создать объект типа Session, объект типа Session, чтобы создать объект типа Destination и, с его помощью – объект типа Producer. Producer умеет отправлять сообщения в очереди. Сообщения у нас – просто строки; на деле наш Producer просто выводит текст сообщения в консоль, никуда ничего не отправляет.

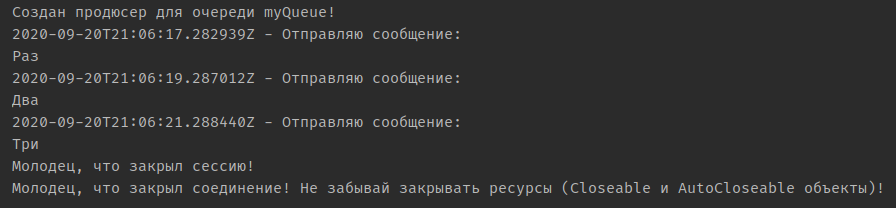
Вы можете посмотреть, как пользоваться настоящей библиотекой здесь: <https://activemq.apache.org/hello-world>. Моя, во-первых, конечно, не выполняет никаких функций, кроме вывода некоторой информации в консоль, а во-вторых, интерфейсы упрощены, но принцип работы такой же, выполните задание – с настоящей отправкой тоже справитесь.

**Задания**

**1. Отправка сообщений из списка.**

Приложение при запуске создаёт список сообщений “Четыре”, “Пять”, “Шесть”. Затем с использованием итератора проходит по списку и отправляет каждое из сообщений, выжидая по 2 секунды между отправкой.

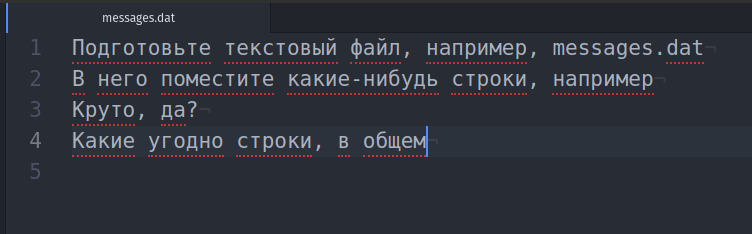
Результат будет выглядеть примерно так:



**2. Отправка сообщений из файла.**

**\*Для выполнения задания рекомендую скопировать предыдущий проект в новую папку и переделать.**

Подготовьте текстовый файл, например, messages.dat. В него поместите какие-нибудь строки, например:

Теперь ваше приложение должно считывать этот файл и отправлять каждую строчку с интервалом в 2 секунды. Файл заканчивается – приложение должно закончить работу.

**2\*.** Теперь ваше приложение должно считывать строки из файла, путь к которому передаётся в качестве аргумента командной строки.

**2\*\*.** Теперь ваше приложение при достижении конца файла должно возвращаться на его начало, отправляя одни и те же строки по кругу, пока не будет остановлено вручную.