|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА**  ***Институт информационных технологий***  ***Кафедры Вычислительной техники*** |
|  |

**КУРСОВАЯ РАБОТА**

По дисциплине «Разработка клиент-серверных приложений»

**Тема курсовой работы** Клиент-сервеный чат на основе Netty\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Студент группы** ИКБО-05-18 Шмаков Глеб Ильич \_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(учебная группа, Фамилия Имя Отчество студента). (подпись студента)

**Руководитель курсовой работы** ст. преп. ВТ, Воронков С.О., \_\_\_\_\_\_\_\_\_\_\_\_\_

Должность, звание, ученая степень (подпись руководителя)

Работа представлена к защите «\_26\_» \_декабря\_ 2020 г.

Допущен к защите «\_26\_» \_ декабря\_ 2020 г.

**Задание**

Написать клиент-серверный чат, используя неблокирующую среду Netty. Чат должен поддерживать подключение нескольких пользователей, сообщения одного пользователя видны остальным, проверять наличие дублируемых имён пользователей, сообщать о подключении и отключении пользователей.

**Реализация**

**Сервер**

В основе сервера чата лежит обычный сервер Netty. В потоке обработки данных(pipeline) используются стандартные кодеры и декодеры строк (StringEncoder, StringDecoder), а также пользовательский класс ChatHandler.

ChatHandler представляет собой класс, наследующий дженерик с типом String SimpleChannelInboundHandler. В классе реализованы методы channelActive, channelRead0 и channelInactive. Также класс имеет поля users(словарь <канал пользователя, имя пользователя>) и messages(список всех сообщений). Экспериментальным путём было установлено, что для каждого подключившегося клиента создаётся новый экземпляр класса ChatHandler, поэтому эти поля являются статическими.

channelActive запускается при первом подключении клиента и отправляет клиенту сообщение с запросом имени пользователя.

В channelRead0 реализуется основная логика чата. Если канала пользователя нет среди ключей словаря users, то подразумевается что он находится в стадии регистрации в чате и должен был отправить имя пользователя. Если отправленное клиентом имя пользователя уже занято другим клиентом чата, то клиенту отправляется ответ с просьбой ввести другое имя пользователя. В противном случае имя пользователя связывается с его каналом в словаре users, новому пользователю отправляется вся история чата, и все пользователи в чате видят сообщение о присоединении нового пользователя. Если пользователь уже зарегистрирован, то присланная им строка публикуется в чате.

Если пользователь вышел из чата, то активируется функция channelInactive, которая удаляет пользователя из словаря users и отправляет всем пользователям уведомление о выходе.

Отправка сообщений и уведомлений о входе и выходе реализована в функции sendMessage. Она активируется в коде после каждых перечисленных ранее действий, и отправляет сообщение всем клиентам из словаря users.

**Клиент**

В основе клиента чата лежит стандартный сервер Netty. В потоке обработки данных(pipeline) используются декодер использующий делиметры (DelimiterBasedFrameDecoder), стандартные кодеры и декодеры строк (StringEncoder, StringDecoder), а также пользовательский класс ClientHandler. Ввод сообщений клиентом осуществляется в цикле посредством объекта канала, полученного в результате установления соединения.

ClientHandler представляет собой класс, наследующий дженерик с типом String SimpleChannelInboundHandler. В классе реализованы методы channelRead0 и channelReadComplete. channelRead0 отображает присланные сервером сообщения. channelReadComplete очищает буфер клиента.

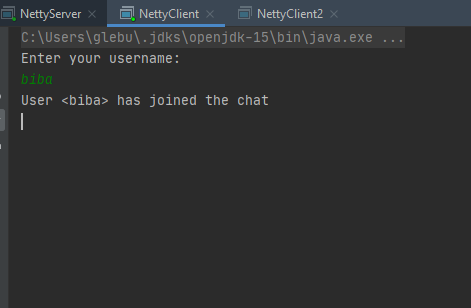
Сервер в качестве агрументов командной строки принимает на вход номер порта, а клиент – адрес хоста и номер порта.

**Контейнеризация**

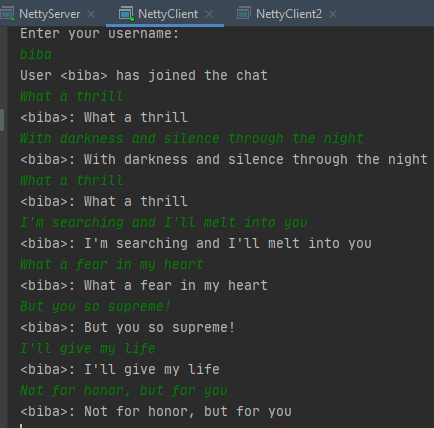
Для того чтобы упаковать в контейнеры клиент и сервер я разбил их на два проекта. Каждый проект по отдельности собирается с помощью maven с подгрузкой зависимостей, затем с помощью Dockerfile собирается образ на основе jdk-15 содержащий исполняемый файл .jar .

**Демонстрация работы**

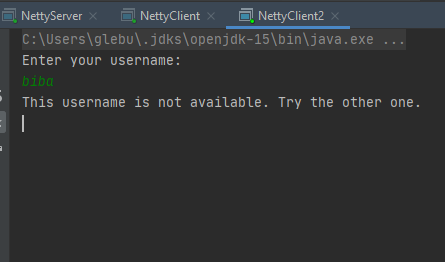
Регистрация пользователя в чате



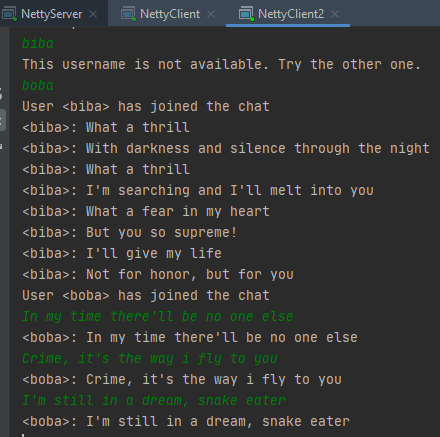
Отправка пользователем сообщений



Попытка подключиться к чату, используя уже занятое имя



Успешное подключение и отправка сообщений вторым пользователем



Первый пользователь покинул чат

