**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

К курсовой работе

по курсу «Программирование на языке Java»

на тему «Текстовый чат»

Выполнил студент группы 21ВВП1:

Кабачек Н.А.

Приняла:

Юрова О.В.

# Содержание

# Введение

«Клиент - сервер» — вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Фактически клиент и сервер — это программное обеспечение. Обычно эти программы расположены на разных вычислительных машинах и взаимодействуют между собой через вычислительную сеть посредством сетевых протоколов, но они могут быть расположены также и на одной машине. Программы-серверы ожидают от клиентских программ запросы и предоставляют им свои ресурсы в виде данных или в виде сервисных функций (например, работа с электронной почтой, общение посредством систем мгновенного обмена сообщениями или просмотр web-страниц во всемирной паутине). Поскольку одна программа-сервер может выполнять запросы от множества программ-клиентов, её размещают на специально выделенной вычислительной машине, настроенной особым образом, как правило, совместно с другими программами-серверами, поэтому производительность этой машины должна быть высокой. Из-за особой роли такой машины в сети, специфики её оборудования и программного обеспечения, её также называют сервером, а машины, выполняющие клиентские программы, соответственно, клиентами.

# 1. Постановка задачи

Необходимо разработать систему программ клиент-серверной архитектуры, позволяющую пользователю играть, получать информацию с сервера и взаимодействовать с ней.

Функции клиента:

● Обеспечение интерфейса с пользователем

● Взаимодействие с сервером

● Регистрация на сервере

Функции сервера:

● Управление трафиком данных

● Взаимодействие с программами-клиентами

Приложение должно обладать графическим интерфейсом и использовать следующие технологии:

1) Java Collections Framework

2) Механизм обработки исключительных ситуаций

3) Java Stream API

4) Java Multithreading

5) Сетевое взаимодействие.

ОС – Windows. Язык программирования: Java. Среда разработки NetBeans.

# 2. Выбор решения

Важной частью курсовой работы является передача сообщений от клиента к серверу. Для реализации этого используются TCP-сокеты. TCP гарантирует доставку пакетов, их очередность, автоматически разбивает данные на пакеты и контролирует их передачу, в отличии от UDP. ПротоколTCP был выбран так как нам необходимо гарантировать доставку данных, UDP.

Для взаимодействия между компьютерами используются адреса и порты. Адрес представляет собой 32-битную структуру. Номер порта – целое число от 0 до 65535.33

# 3. Описание протокола взаимодействия.

Запускаем сервер и создаем сокет. Устанавливаем локальную точку для прослушивания подключений. Запускаем прослушивание входящих подключений и принимаем новых клиентов и запросы от них в бесконечном цикле. После того, как клиент или запрос был принят, другой запрос отправляется обратно клиенту. На этом установление соединения между клиентом и сервером заканчивается. После сервер в бесконечном цикле ожидает других команд от клиента.

Диаграмма последовательности находиться в Приложение Б.5.

Всего будет реализовано два типа сообщений протокола обмена клиента с сервером. Первый состоит только из команды:

“/close” – клиент может отправить такую команду серверу, когда требуется закрыть соединение.

“/online” – клиент запрашивает список пользователей, находящихся в сети.

“#leave” – сервер получает сообщение о том, что клиент вернулся на страницу авторизации.

Второй тип протокола обмена состоит из команды и сообщения, разделенные специальным символом ‘|’:

“/online|name,name…:id,id…” – после авторизации клиент получает список пользователей, находящихся онлайн.

“#new|Name:Id” – клиент получает уведомление о том, что появился новый пользователь в сети.

“#reg|name login password” – клиент отправляет информацию серверу с данными, указанными в окне регистрации.

“#log|login password” – клиент отправляет информацию серверу с данными, указанными в окне авторизации.

“#reg|accept” и “#reg|reject” – сервер отправляет флаг об успешной/неудачной регистрации.

“#log|online” – сервер отправляет клиенту флаг о том, что пользователь с указанными данными уже находится в сети.

“#log|Name” и “#log|reject” – сервер отправляет имя клиенту в случае успешной авторизации, иначе - “reject”.

“#leave|Id” – клиент получает уведомление о том, что пользователь вышел из сети.

“%Message” – клиент получил сообщение от сервера.

“&Id|Message” – клиент получил сообщение от пользователя.

“@TargetId:SenderId|Message” – клиент получил сообщение от пользователя в общем чате.

“@Id|message” – отправка конкретному пользователю по Id. Такое сообщение клиент отправляет серверу.

# 4. Архитектура сервера

В курсовой работе для организации взаимодействия сервера с клиентами используется следующий подход: при запуске сервера происходит подключение к базе данных, затем сервер начинает ждать ответ от каждого клиента.

При поступлении запросов от клиентов, он ищет совпадение с базой данных и отвечает на основе существующей у него информации.

Диаграмма классов сервера находиться в Приложение Б.2.

# 5. Архитектура клиента

После запуска клиента, он сразу же пытается установить соединение с сервером. После того, как соединение было установлено, пользователь может войти в свой аккаунт или создать новый.

После входа в аккаунт пользователь может отправить сообщение в общий чат или выбрать собеседника из списка и отправить ему личное сообщение. Так же клиент может вернуться на страницу авторизации, выйдя из сети из текущего профиля.

Диаграмма классов клиента находиться в Приложение Б.2.

# 6. Описание работы программы

Ниже представлена работа клиент-серверного приложения.

Для начала необходимо запустить сервер:

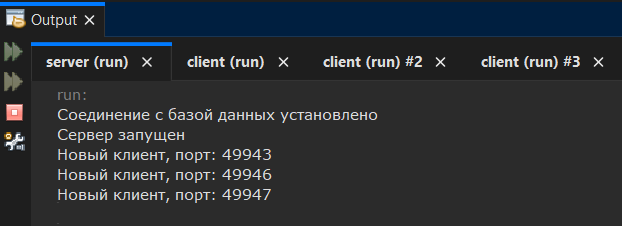
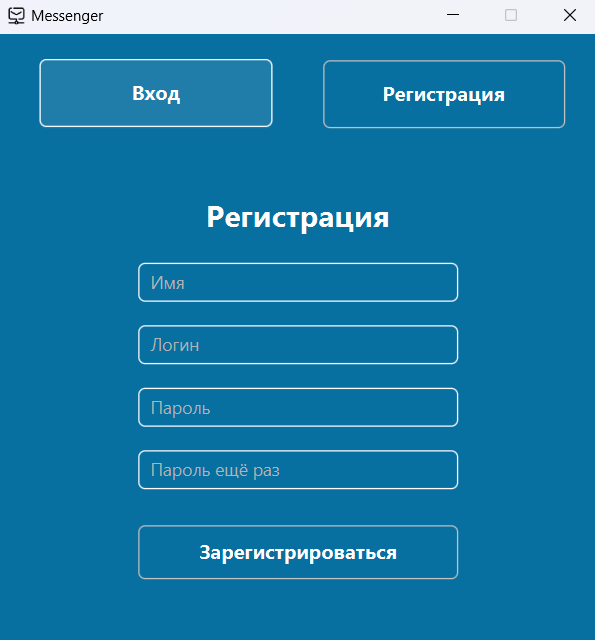
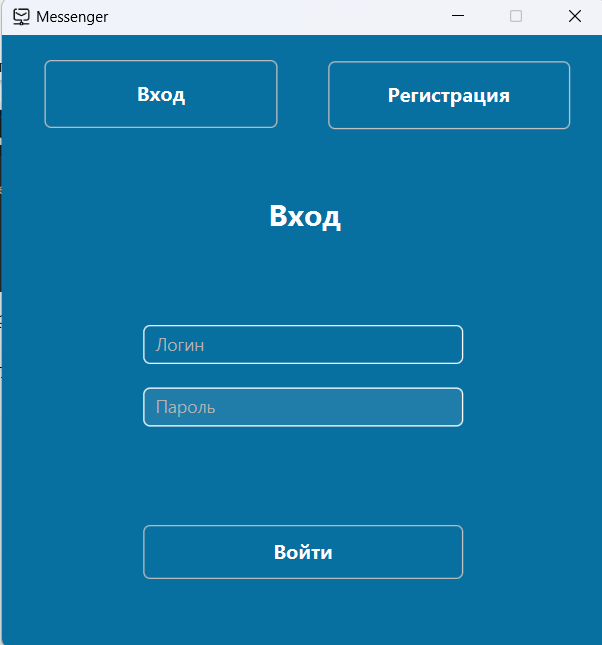


Рисунок 1 - Сервер запущен

Запускаем клиента(-ов). Открывается окно авторизации:



# Заключение

В ходе выполнения курсового проектирования были изучены принципы работы протоколов TCP/IP. Были получены навыки разработки приложения с пользовательским интерфейсом.

В итоге было разработано клиент-серверное приложение для передачи текстовых сообщений.

# Литература

1. "Head First Java, Изучаем Java", Кэти Сьерра, Берт Бэйтс
2. "Java. Руководство для начинающих", Герберт Шилдт
3. "Java для чайников", Барри Бёрд
4. "Java. Полное руководство", Герберт Шилдт

# Приложение А. Листинг программы