Протокол взаимодействия сервера и клиентов в sockchat.

Diagram

Description automatically generated

Общение между сервером и клиентами производится посредством сообщений. Sockchat поддерживает 4 вида сообщений, определенных в перечислении MessageType (message.py):

class MessageType(IntEnum):  
 LOGIN = auto()  
 TEXT = auto()  
 ACK = auto()  
 DISCONNECT = auto()

Для каждого вида собщения есть отдельный класс: LoginMessage, TextMessage, AckMessage и DisconnectMessage. Перед отправкой по сети, сообщения кодируются в формате JSON. Подробнее о порядке отправки сообщений далее.

Оглавление

[LOGIN 2](#_Toc110978066)

[TEXT и ACK 2](#_Toc110978067)

[DISCONNECT 3](#_Toc110978068)

# LOGIN

LOGIN – сообщение отправляемое при подключении клиента к серверу.

# TEXT и ACK

TEXT – сообщение отправляемое при отправке текста от клиента к серверу. Сервер пересылает это сообщение всем остальным клиентам.

ACK – сообщение отправляемое клиентом при прочтении предыдущего TEXT сообщения. Сервер пересылает это сообщение клиенту-автору изначального TEXT сообщения.

Пример пересылки текста от пользователя изображен на следующем рисунке.

Diagram

Description automatically generated

Порядок отправки сообщений:

1. Клиент 1 передает серверу TEXT сообщение со именем пользователя, содержанием и id текста.
2. Сервер пересылает сообщение остальным клиентам.
3. После отображения текста пользователю, клиент 2 уведомляет сервер о том что текст был прочтен, сообщение содержит имя автора изначального текста, имя пользователя прочитавшего текст и id изначального текста.
4. Сервер пересылает уведомление клиенту 1, сервер находит адрес клиента 1 по имени из поля text\_by\_username.

# DISCONNECT

DISCONNECT – сообщение отправляемое клиентом серверу при закрытии клиента. Сервер так же отправляет DISCONNECT в ответ.

Diagram

Description automatically generated

Необходимость в этом обмене сообщениями обусловлена организацией клиента. Клиент состоит из двух потоков: один получает сообщения пользователя из консольного интерфейса и отправляет их по сокету на сервер, второй поток читает сообщения от сервера из сокета. Сигнал Ctrl+C от пользователя заврешает лишь один поток, второй поток продолжает ждать сообщение от сервера, именно поэтому сервер должен отправить DISCONNECT, после чего все потоки клиента будут завершены и приложение закроется.

Альтернативные решения для завершения приложения:

1. Использовать чтение из сокета с timeout. Если поток читающий сообщения от сервера будет переодически прерывать ожидание сообщений и проверять некоторый глобальный флаг устанавливаемый другим потоком, завершение работы клиента сможет происходить без участия сервера. Этот подход не используется потому что интерфейс makefile() используемый в sockchat не поддерживает timeout.
2. Вызов функции os.\_exit(). Это решение может привести к утечке ресурсов программы (сокетов, файлов и их дескрипторов) поэтому является оптимальным.