Сетевой чат
Описание программы
Часть 1. Сервер

Раздел 1. Функциональная модель

1. Функция «C01. Запустить Сервер»

Запуск Сервера выполняется в основном потоке работы. При этом производится:

- Инициация настроек Сервера;
- Открытие и чтение файла учетных записей пользователей чата;
- Открытие лог-файла и логирование операции запуска Сервера;
- Перевод Сервера в режим ожидания, который прерывается при обращении Клиента.

Диаграмма (Рисунок 1) представляет схему исполнения функции.

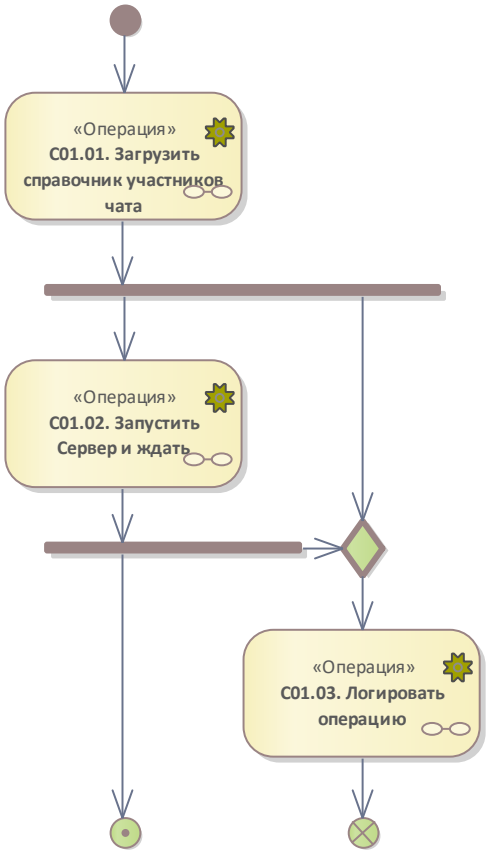


Рисунок 1 - C01. Запустить Сервер

Далее в таблице (Таблица 1) приведена детализация операций функции «C01. Запустить Сервер».

Модель программных реализаций представленных функций приведена в следующей главе.

Таблица 1 - Операции функции «C01. Запустить Сервер»

Наименование операции	Описание операции
C01.01. Загрузить справочник участников чата	Операция выполняет инициализацию необходимых статических переменных, открытие лог-файла Сервера и создание пула потоков. При выполнении операции производится чтение файла

Наименование операции	Описание операции
	справочника участников чата, разбор его записей и формирование на их основе объектов класса User. Наименование json-файла справочника участников чата и путь к этому файлу находятся в классе настроек.
С01.02. Запустить Сервер и ждать	Операция запускает Сервер, создает сокет Клиента и переходит к ожиданию обращений участников чата. Если при запуске сервера или создании сокета происходит ошибка, сообщение выводится в консоль и программа завершается. В любом случае операция логируется.
С01.03. Логировать операцию	Файл лога — это json-файл, в который дописываются соответствующие записи о выполненных операциях. В зависимости от типа операции формируется запись в логе. Сначала собираются данные об операции: • дата и время операции из объекта класса Message; для операции старта Сервера берется текущее время; • тип операции из enum OperationType передается в качестве аргумента соответствующему методу; • ник отправителя для операций регистрации и коннекта - из соответствующих сообщений, для получения и передачи - из соответствующей беседы, для операции запуска Сервера это поле не заполняется; • поле ников получателей для полученного сервером сообщения заполняется из этого сообщения, для отправляемых сообщений в это поле записывается единственный адресат, которому отправляется сообщение (для операций старта сервера, регистрации и коннекта поле остается пустым); • поле кода завершения операции передается как аргумент соответствующему методу формирования объекта лога; • поле тела сообщения остается пустым для операций старта сервера, регистрации и коннекта. Далее осуществляется преобразование объекта лога в форму json-записи и сохранение в файле.

2. Функция «C02. Обработать сообщение Клиента»

Операция обработки обращения Клиента предполагает:

- Прием сообщений от Клиента;
- Регистрацию клиента в чате;
- Деактивацию участника чата по команде “/exit”;
- Логирование всех операций.

Диаграмма (Рисунок 2) представляет схему исполнения функции.

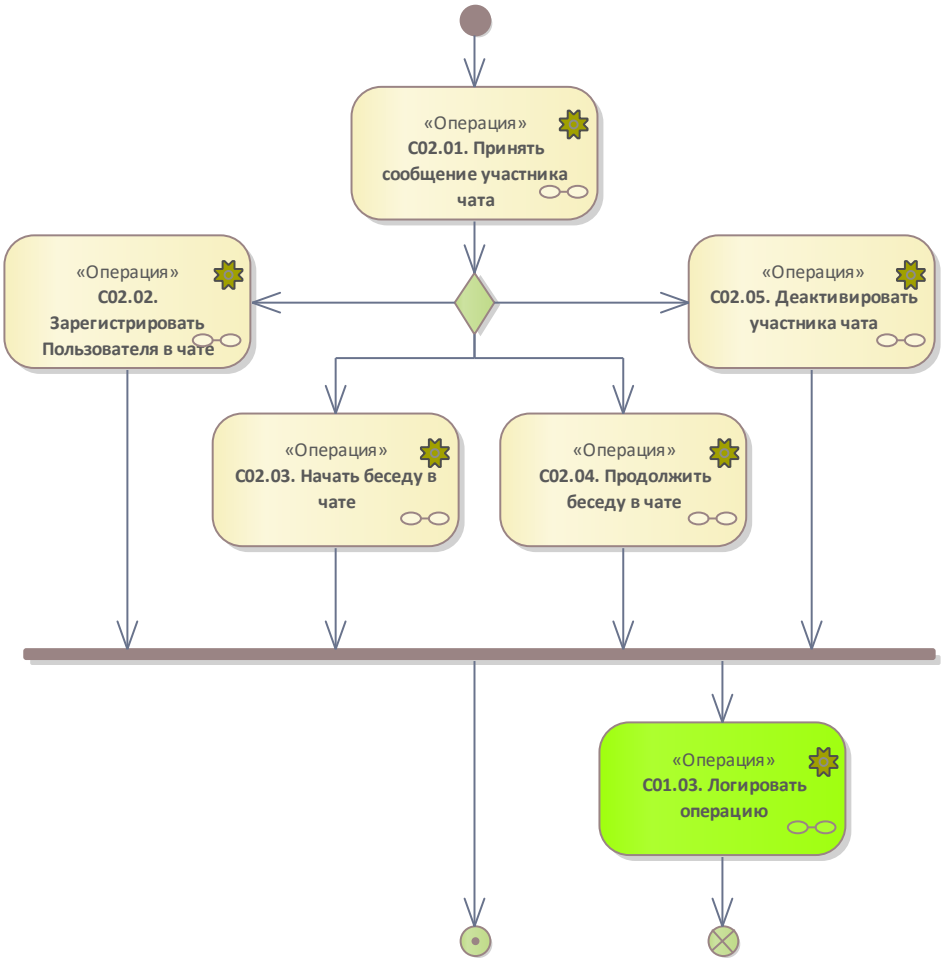


Рисунок 2 - C02. Обработать обращение Клиента

Далее в таблице (Таблица 2) приведена детализация операций функции «C02. Обработать сообщение Клиента».

Модель программных реализаций представленных функций приведена в следущей главе.

Таблица 2 - Операции функции «C02. Обработать сообщение Клиента»

Наименование операции	Описание операции
C02.01. Принять сообщение	При приеме сообщения участника чата Сервер выполняет

Наименование операции	Описание операции
участника чата	следующие действия: • пытается преобразовать строку сообщения в объект класса Message; • в случае неверного формата сообщения выставляет соответствующую ошибку; • анализирует тип сообщения и передает обработку соответствующей операции; • логирует операцию и передает обработку функции «С03. Передать сообщение Клиенту».
С02.02. Зарегистрировать Пользователя в чате	Для регистрации участника чата Сервер выполняет следующие действия: • извлекает из строки сообщения ник нового участника; • ник регистрируемого проверяется на уникальность; • если ник не уникален выставляется соответствующая ошибка (уже зарегистрирован или ник занят); • извлекает имя участника из тела сообщения; • если имя отсутствует выставляется ошибка; • формирует объект класса User соответствующий новому участнику. Далее операция логируется и обработка передается функции «С03. Передать сообщение Клиенту».
С02.03. Начать беседу в чате	Если в сообщении отсутствует идентификатор беседы - это признак запроса на коннект участника чата. При этом поле адресов получателей и тела сообщения в таком сообщении игнорируются. Сервер выполняет следующее: • проверяет наличие ника отправителя в зарегистрированных участниках; • если такого участника нет - выставляет ошибку («для участия в чате необходимо зарегистрироваться»); • сохраняет ссылки на каналы взаимодействия в соответствующем объекте класса User; • формирует задачу создания беседы и помещает ее в поток; • создает и формирует беседу (объект класса TalkShow). Далее операция логируется и обработка передается функции «С03. Передать сообщение Клиенту».
С02.04. Продолжить беседу в чате	В сообщении присутствует идентификатор беседы. При этом поле адресов получателей должно быть корректно заполнено. Сервер выполняет следующее: • проверяет наличие ника отправителя в участниках беседы; • если отправителя нет в списке - выставляет ошибку («неверный идентификатор беседы»); • проверяет наличие получателей среди участников беседы; • для тех, получателей, которых нет в списке участников беседы осуществляется проверка на участие в чате; • если среди участников чата нет каких-либо получателей, выставляется сообщение об ошибке («пользователи с никами ... не зарегистрированы в чате»);

Наименование операции	Описание операции
	• если среди получателей есть не активные в данный момент участники выставляется сообщение об ошибке («пользователи с никами ... не активны в чате»); • формирует задачу передачи сообщения и помещает ее в поток. Далее операция логируется и обработка передается функции «С03. Передать сообщение Клиенту».
С02.05. Деактивировать участника чата	Деактивация производится по получении от отправителя сообщения со строкой в теле сообщения. Поле сообщения со списком получателей игнорируется. Для деактивации сеанса участника чата Сервер выполняет: • выбирает объект класса User, соответствующий нику отправителя; • если такого ника нет среди подключенных участников, переходит к логированию операции; • в объекте User: устанавливает userCondition = false, userCanalIn = null, userCanalOut = null. Далее операция логируется. Сообщение о результате операции участнику чата не возвращается.

3. Функция «C03. Передать сообщение Клиенту»

Операция обработки обращения Клиента предполагает:

- Передачу сведений о зарегистрированных в сети Пользователях чата;
- Передачу сообщения адресату;
- Логирование всех операций.

Диаграмма (Рисунок 1) представляет схему исполнения функции.

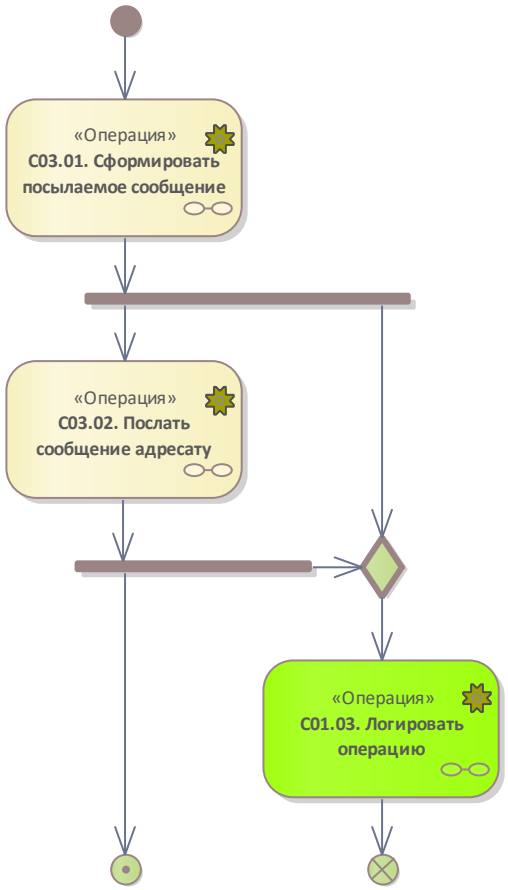


Рисунок 3 - C03. Передать сообщение Клиенту

Далее в таблице (Таблица 3) приведена детализация операций функции «C03. Передать сообщение Клиенту».

Модель программных реализаций представленных функций приведена в следующей главе.

Таблица 3 - Операции функции «C03. Передать сообщение Клиенту»

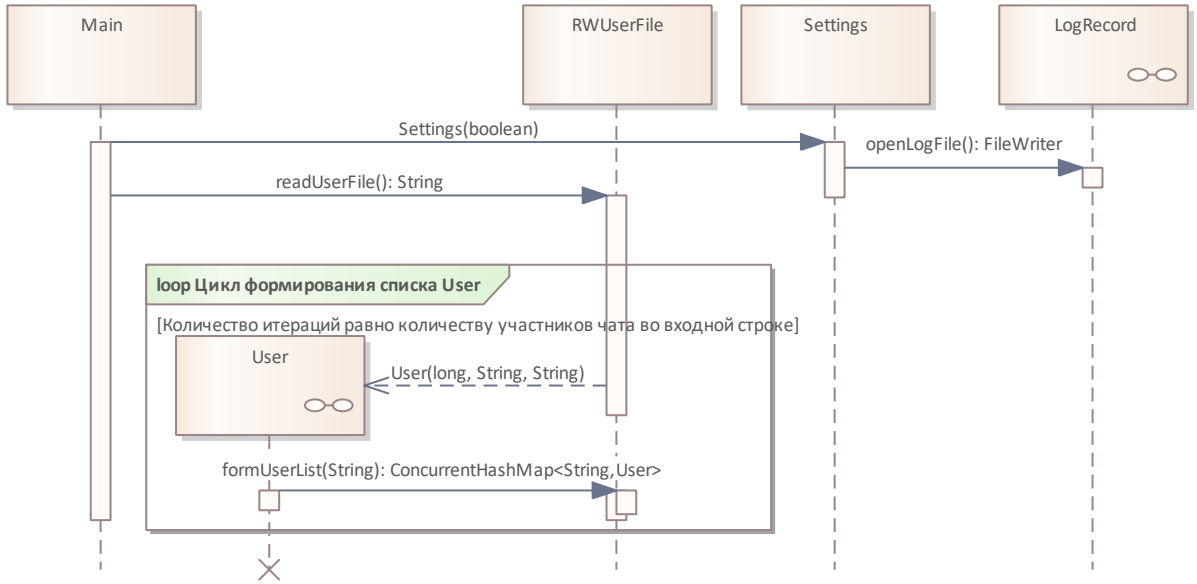
Наименование операции	Описание операции
C03.01. Сформировать посылаемое сообщение	На вход поступает обработанное при получении сообщение, а на выходе - объекты класса Message, которые должны быть отправлены адресатам. Если код завершения (поле result) меньше нуля, поступившее сообщение обработано с ошибкой. Такое сообщение должно

Наименование операции	Описание операции
	быть возвращено отправителю с указанием ошибки. Сообщение без ошибки и сообщения с незарегистрированными никами и неактивными участниками порождают сообщения для активных адресатов.
С03.02. Послать сообщение адресату	Сформированные сообщения в виде массива объектов класса Message для передачи адресатам вместе с исходным сообщением поступают на вход операции. Сервер для каждого сообщения формирует строку сообщения и отсылает адресату.

4. Модель программной реализации функции «С01. Запустить Сервер»

4.1. Операция «С01.01. Загрузить справочник участников чата»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «С01.01. Загрузить справочник участников чата»



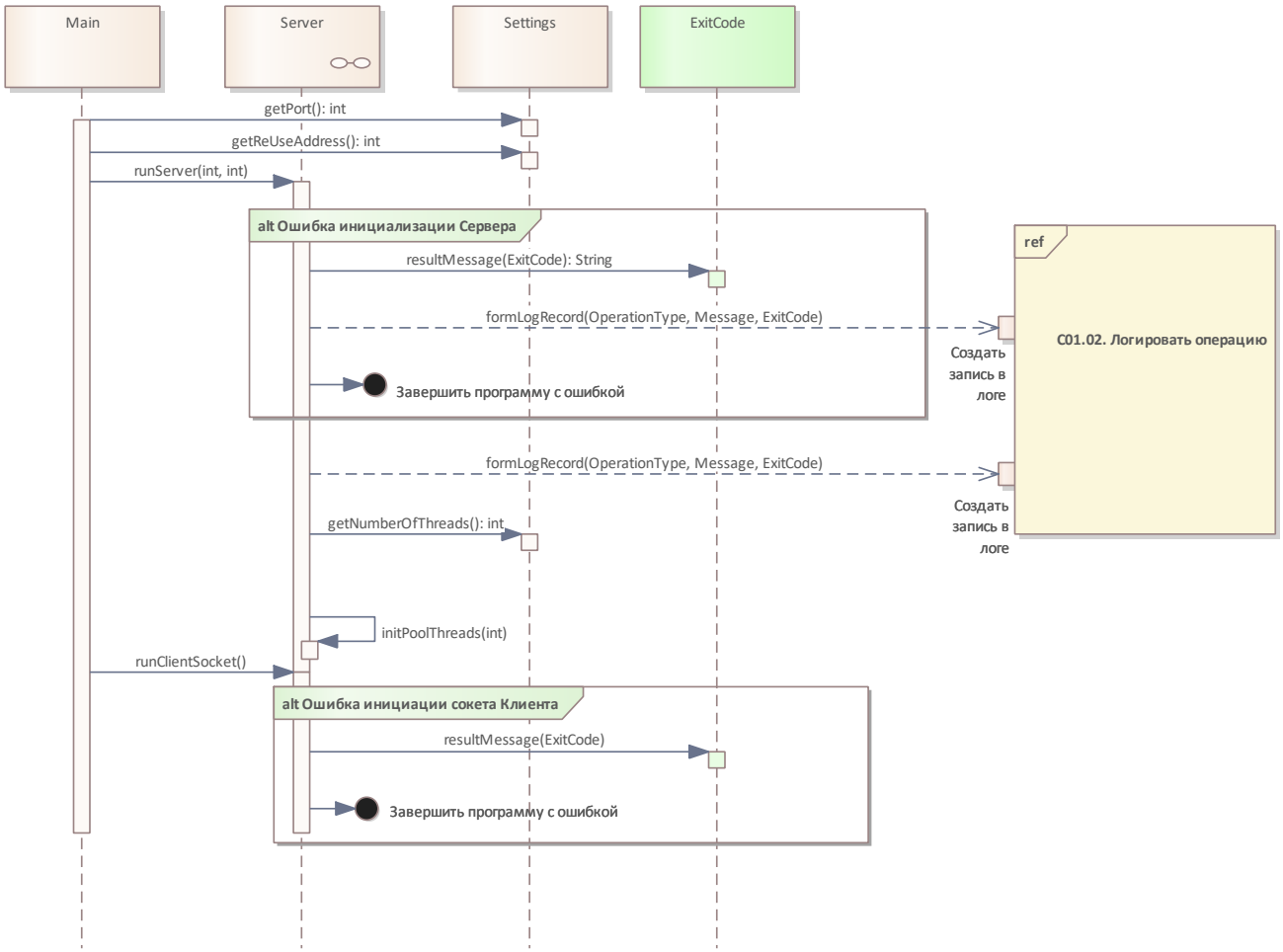
В таблице (Таблица 4) приведено описание модулей программной реализации.

Таблица 4 - Описание модулей программной реализации операции «С01.01. Загрузить справочник участников *чата»*

Модуль	Параметры	Классы	Комментарий
Settings(boolean)	begin	от: Main к:Settings	Инициализируем необходимые статические переменные
openLogFile()		от: Settings к:LogRecord	Открываем лог-файл.
readUserFile()	path, name	от: Main к:RWUserFile	Читаем файл справочника участников чата.
User(long, String, String)	userID, userNickname, userName	от: RWUserFile к:User	Создаем объект класса User.
formUserList(String)	jsonString	от: User к:RWUserFile	Поместить данные участника чата в мапу, для поиска участника по нику.

4.2. Операция «С01.02. Запустить Сервер и ждать»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «С01.02. Запустить Сервер и ждать»



В таблице (Таблица 5) приведено описание модулей программной реализации.

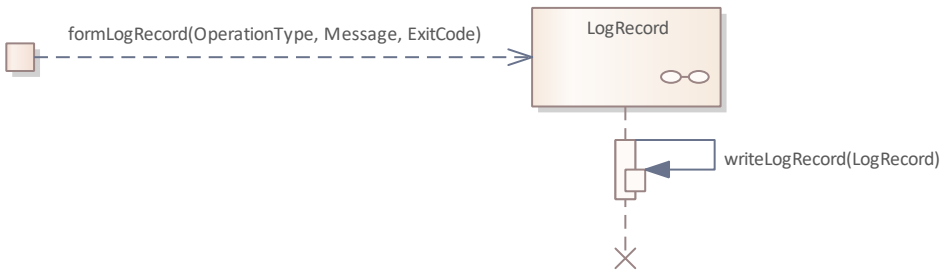
Таблица 5 - Описание модулей программной реализации операции «C01.02. Запустить Сервер и ждать»

Модуль	Параметры	Классы	Комментарий
getPort()		от: Main к:Settings	Получаем номер порта Сервера.
getReUseAddress()		от: Main к:Settings	Получаем разрешение на повторное использование адреса порта.
runServer(int, int)	port, reuseaddr	от: Main к:Server	Запускаем сервер.
resultMessage(ExitCode)	code	от: Server к:ExitCode	Выводим сообщение о неудаче инициализации Сервера в консоль.
formLogRecord(OperationType, Message, ExitCode)	oper, message, exitCode	от: Server к:Создать запись в логе	
formLogRecord(OperationType, Message, ExitCode)	oper, message, exitCode	от: Server к:Создать запись в логе	
getNumberOfThreads()		от: Server к:Settings	Получаем количество потоков для резервирования.
initPoolThreads(int)		от: Server к:Server	Резервируем потоки для обработки поступающих от участников чата сообщений.

Модуль	Параметры	Классы	Комментарий
runClientSocket()		от: Main к:Server	Переходим к ожиданию сообщений участников чата.
resultMessage(ExitCode)	code	от: Server к:ExitCode	Сообщение о неудаче запуска сокета.

4.3. Операция «C01.03. Логировать операцию»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «C01.03. Логировать операцию»



В таблице (Таблица 6) приведено описание модулей программной реализации.

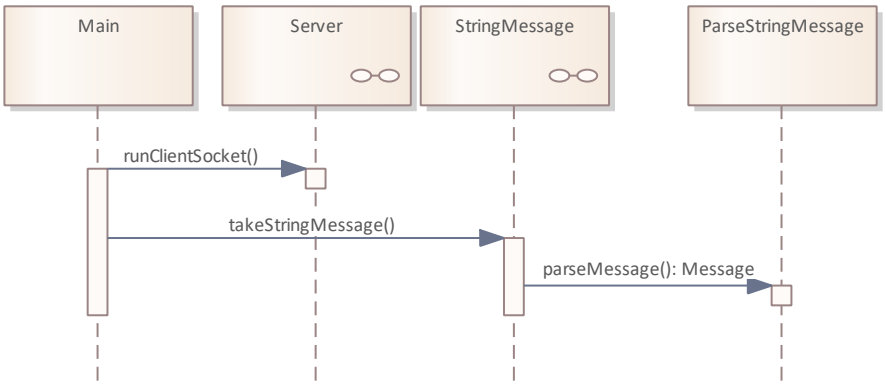
Таблица 6 - Описание модулей программной реализации операции «C01.03. Логировать операцию»

Модуль	Параметры	Классы	Комментарий
formLogRecord(OperationType, Message, ExitCode)	oper, exitCode	от: создать запись в логе к:LogRecord	Создаем объект записи в лог и заполняем атрибуты согласно типу операции.
writeLogRecord(LogRecord)	record	от: LogRecord к:LogRecord	Преобразуем объект записи лог-файла в json-строку и записываем в файл.

5. Модель программной реализации функции «C02. Обработать сообщение Клиента»

5.1. Операция «C02.01. Принять сообщение участника чата»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «C02.01. Принять сообщение участника чата»



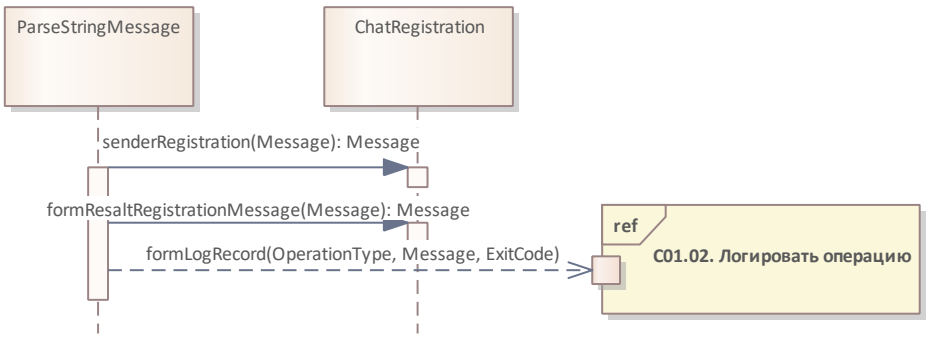
В таблице (Таблица 7) приведено описание модулей программной реализации.

Таблица 7 - Описание модулей программной реализации операции «C02.01. Принять сообщение участника *чата»*

Модуль	Параметры	Классы	Комментарий
runClientSocket()		от: Main к:Server	Получить сообщение.
takeStringMessage()		от: Main к:StringMessage	Получаем объект, содержащий строку сообщения, входной и выходной каналы.
parseMessage()		от: StringMessage к:ParseStringMessage	Проверяем формат входящего сообщения и создаем объект Message.

5.2. Операция «C02.02. Зарегистрировать Пользователя в чате»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «C02.02. Зарегистрировать Пользователя в чате»



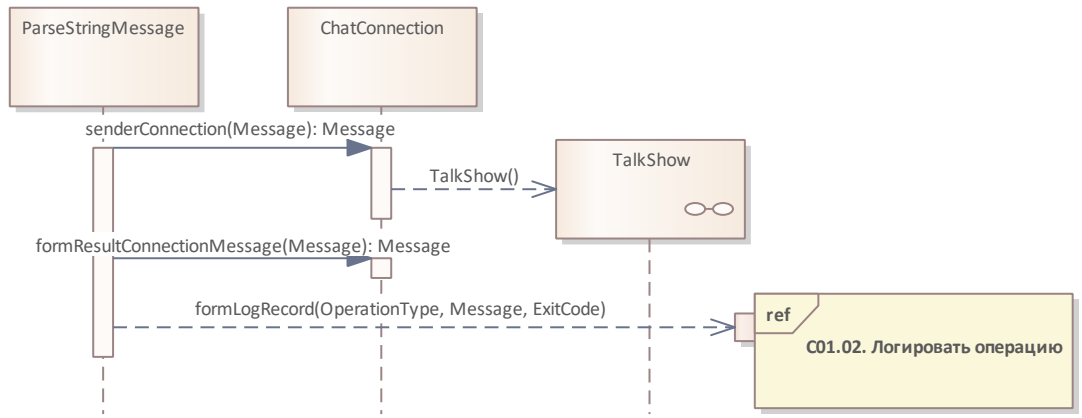
В таблице (Таблица 8) приведено описание модулей программной реализации.

Таблица 8 - Описание модулей программной реализации операции «C02.02. Зарегистрировать Пользователя в *чате»*

Модуль	Параметры	Классы	Комментарий
senderRegistration(Message)	inMessage	от: ParseStringMessage к:ChatRegistration	Проверка и регистрация участника в чате.
formResaltRegistrationMessage(Message)	inMessage	от: ParseStringMessage к:ChatRegistration	Формирование ответного сообщения пользователю о результате регистрации.
formLogRecord(OperationType, Message, ExitCode)	oper, message, exitCode	от: ParseStringMessage к:MessageEndpoint1	

5.3. Операция «C02.03. Начать беседу в чате»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «C02.03. Начать беседу в чате»



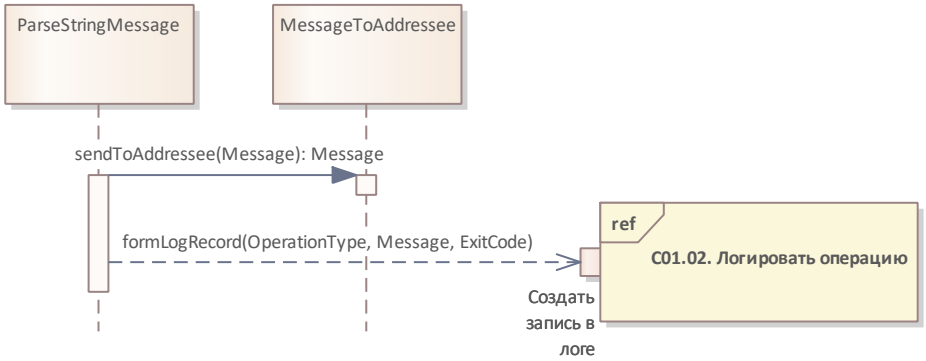
В таблице (Таблица 9) приведено описание модулей программной реализации.

Таблица 9 - Описание модулей программной реализации операции «C02.03. Начать беседу в чате»

Модуль	Параметры	Классы	Комментарий
senderConnection(Message)	inMessage	от: ParseStringMessage к:ChatConnection	
TalkShow()		от: ChatConnection к:TalkShow	Создаум беседу - объект, который соответствует сеансу общения участника чата с другими участниками.
formResultConnectionMessage(Message)	inMessage	от: ParseStringMessage к:ChatConnection	
formLogRecord(OperationType, Message, ExitCode)	oper, message, exitCode	от: ParseStringMessage к:MessageEndpoint2	

5.4. Операция «C02.04. Продолжить беседу в чате»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «C02.04. Продолжить беседу в чате»



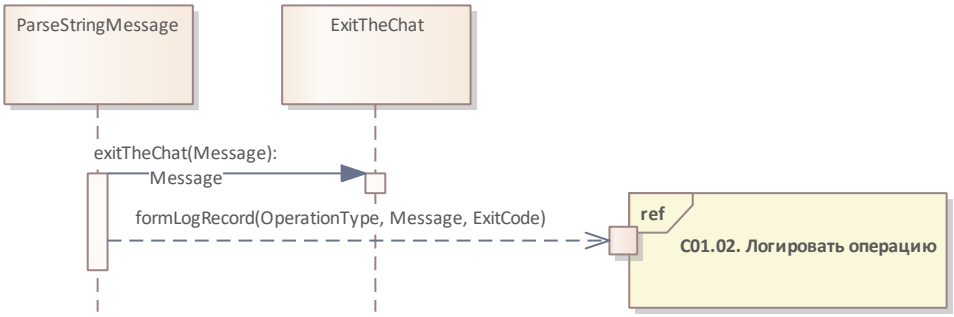
В таблице (Таблица 10) приведено описание модулей программной реализации.

Таблица 10 - Описание модулей программной реализации операции «C02.04. Продолжить беседу в чате»

Модуль	Параметры	Классы	Комментарий
sendToAddressee(Message)	mesg	от: ParseStringMessage к:MessageToAddressee	
formLogRecord(OperationType, Message, ExitCode)	oper, message, exitCode	от: ParseStringMessage к:Создать запись в логе	

5.5. Операция «C02.05. Деактивировать участника чата»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «C02.05. Деактивировать участника чата»



В таблице (Таблица 11) приведено описание модулей программной реализации.

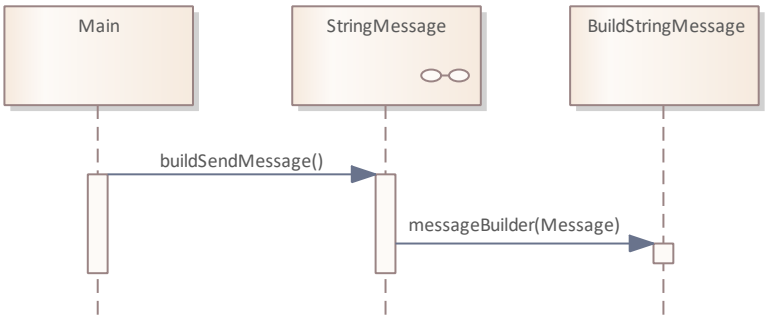
Таблица 11 - Описание модулей программной реализации операции «C02.05. Деактивировать участника чата»

Модуль	Параметры	Классы	Комментарий
exitTheChat(Message)	mesg	от: ParseStringMessage к:ExitTheChat	
formLogRecord(OperationType, Message, ExitCode)	oper, message, exitCode	от: ParseStringMessage к:	

6. Модель программной реализации функции «C03. Передать сообщение Клиенту»

6.1. Операция «C03.01. Сформировать посылаемое сообщение»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «C03.01. Сформировать посылаемое сообщение»



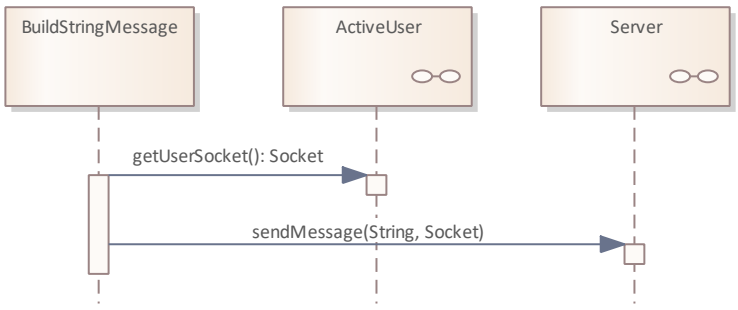
В таблице (Таблица) приведено описание модулей программной реализации.

Таблица 12 - Описание модулей программной реализации операции «C03.01. Сформировать посылаемое *сообщение»*

Модуль	Параметры	Классы	Комментарий
buildSendMessage()		от: Main к:StringMessage	Запускаем формирование и отправку соответствующих исходящих сообщений.
messageBuilder(Message)	mesg	от: StringMessage к:BuildStringMessage	Сформировать строковое сообщение на основе обработанного поступившего сообщения.

6.2. Операция «C03.02. Послать сообщение адресату»

Диаграмма последовательности представляет основные классы объектов и модули программной реализации операции «C03.02. Послать сообщение адресату»



В таблице (Таблица) приведено описание модулей программной реализации.

Таблица 13 - Описание модулей программной реализации операции «C03.02. Послать сообщение

адресату»

Модуль	Параметры	Классы	Комментарий
getUserSocket()		от: BuildStringMessage к:ActiveUser	Получить ссылку на сокет, связанный с адресатом.
sendMessage(String, Socket)	message, out	от: BuildStringMessage к:Server	

Раздел 2. Информационная модель

В информационной модели приведены описания основных классов объектов сетевого чата.

Объекты разложены по следующим пакетам:

- server
- stringmsg
- talkshow
- user
- log.

Для классов приведены следущие описания:

- наименование и определение класса;
- определения атрибутов класса;
- определения методов класса.

Схему взаимодействия классов представляет диаграмма (Рисунок 4).

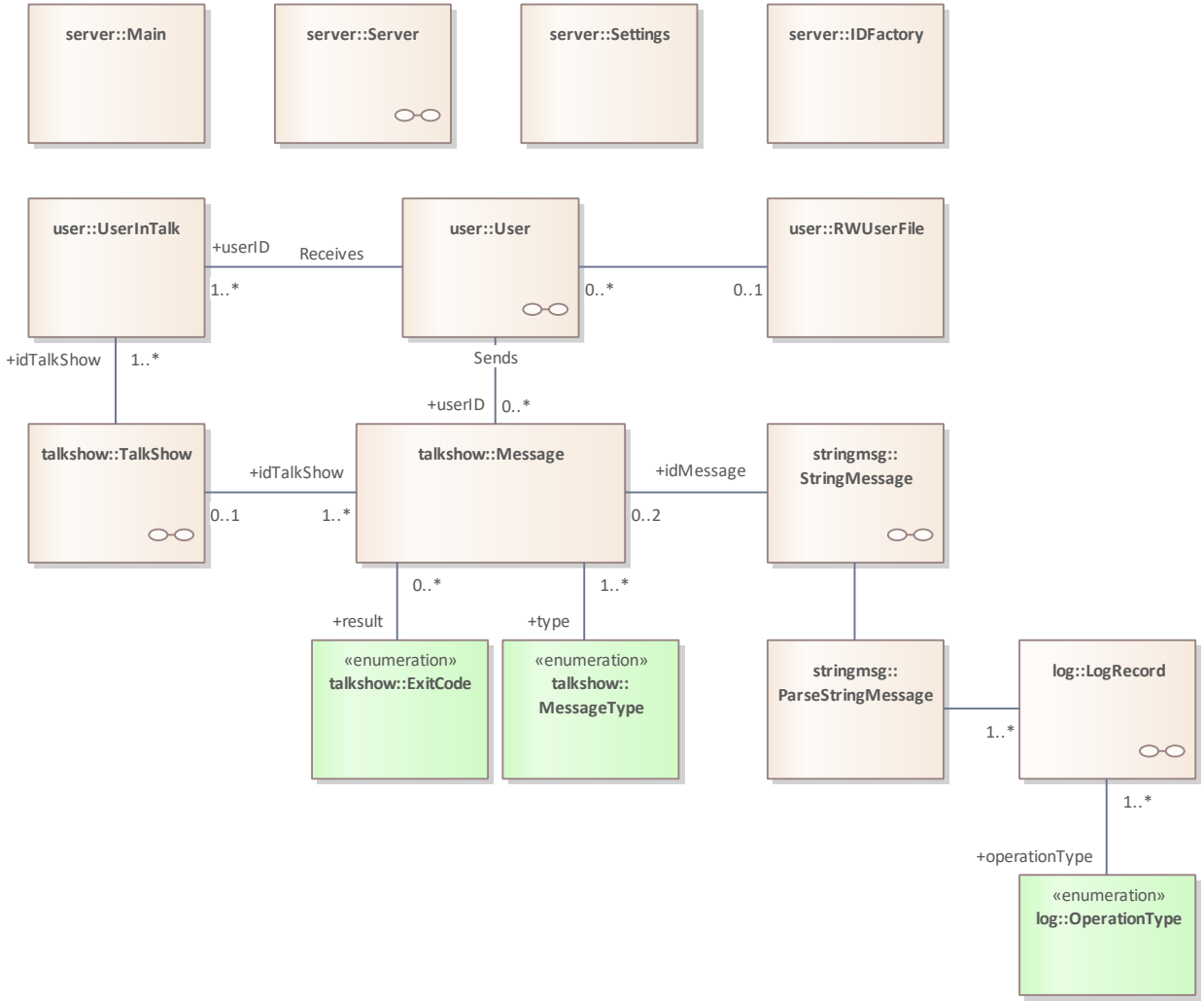


Рисунок 4 - Информационная модель сетевого чата

7. Пакет server

В пакет входят основные общие классы, обеспечивающие запуск программы, основные настройки и создание объектов.

Схему взаимодействия классов представляет диаграмма (Рисунок 5).

Рисунок 5 - Классы пакета server

7.1. Класс Main

Класс входа в программу сетевого чата.

7.2. Класс Server

Класс обслуживает операции взаимодействия Сервера с Клиентом.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 14, Таблица 15).

Таблица 14 - Описание атрибутов класса Server

Наименование	Тип данных	Комментарий
poolTask	ArrayList<Future<Integer>>	Список запущенных задач по обработке сообщений участников чата.
threadPool	ExecutorService	Пул потоков.

Наименование	Тип данных	Комментарий
readMessageTask	ArrayBlockingQueue<Future<StringMessage>>	Очередь задач на чтение приходящих сообщений.
parseMessageTask	ArrayBlockingQueue<Future<Message>>	Очередь задач на разбор поступивших сообщений.
stringMessages	ArrayBlockingQueue<StringMessage>	Очередь поступивших необработанных сообщений.
sendLogTask	ArrayBlockingQueue<Future<Message>>	Сообщение в лог.
ABQ_CAPACITY	int	Размер очереди необработанных сообщений.
log	LogRecord	Запись в лог-файл.
serverSocket	ServerSocket	

Таблица 15 - Описание методов класса Server

Наименование	Параметры	Комментарий
runServer	void	Инициflизируем сокет Сервера, указав номер порта и разрешение на повторное использование адреса. В случае неудачной инициализации выводим ошибку и завершаем работу. Также инициализируем пул потоков.
runClientSocket	void	Инициализируем клиентский сокет, помещаем его в пул потоков и переходим в ожидание сообщений от участников чата.
initPoolThreads	void	Инициализируем пул потоков.
sendMessage	void	Отправка сообщения участнику чата.

7.3. Класс IDFactory

Фабрика по работе с идентификаторами объектов. Выполняет следующие операции:

- формирование идентификатора;
- сохранение объекта в потокобезопасной мапе (ключ – идентификатор);
- выбор объекта по идентификатору из мапы.

Также формирует в потокобезопасной мапе необходимые связи между объектами. Для этого выполняет следующее:

- сохраняет пары идентификаторов связанных объектов;
- по запросу передает идентификаторы объектов, связанных с заданным объектом;
- по запросу передает объекты заданного класса, связанные с каким-либо объектом.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 16, Таблица 17).

Таблица 16 - Описание атрибутов класса IDFactory

Наименование	Тип данных	Комментарий
nextID	AtomicLong	Общий счетчик идентификаторов для всех типов объектов.
objectList	ConcurrentHashMap< Long, Object >	Мапа для хранения объектов: ключ – идентификатор, значение – объект.
objectConnectList	ConcurrentHashMap<long,Set<Long>>	

Таблица 17 - Описание методов класса IDFactory

Наименование	Параметры	Комментарий
initFactory	void	Инициализация мапы и счетчика идентификаторов. Запускается из Server.
buildID	long	Запрос на формирование идентификатора объекта и сохранение объекта в мапе.
objectConnection	void	Формируем мапу для хранения связей объектов: • ключ – идентификатор, • значение - множество идентификаторов связанных объектов.
connectionObjects	Set<Long>	Найти объекты, связанные с объектом, заданным идентификатором. Если объекта, заданного идентификатором нет или связанные объекты не найдены, возвращается null.
getObject	Object	Найти объект в мапе по идентификатору. Если объект не найден, возвращается null.
getRelatedObjects	Set<Object>	Найти объекты заданного класса, связанные с объектом с указанным идентификатором. Если объект не найден, возвращается null.

7.4. Класс Settings

Класс настроек Сервера.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 18, Таблица 19).

Таблица 18 - Описание атрибутов класса Settings

Наименование	Тип данных	Комментарий
port	int	Номер порта Сервера.
autoFlash	boolean	Если true, то методы println, printf или format будут очищать выходной буфер.
reUseAddress	int	Разрешено повторное использование адреса.
messageSeparator	String	Символьная строка, обозначающая начало сообщения. Сразу за messageHead следует блок адресов участников беседы.
talkShowExit	String	Символьная строка - команда завершения беседы в чате. Если у участника чата бесед не осталось, то он помечается как отсутствующий в чате.
pathToFiles	String	Путь к файлам, необходимым для работы чата.

Наименование	Тип данных	Комментарий
logFile	String	Файл логирования операций в чате.
numberOfThreads	int	Количество резервируемых потоков в пуле потоков.
listOfUsersFile	String	JSON-файл справочника участников чата.
calendar	GregorianCalendar	
referenceToLogFile	FileWriter	Ссылка на лог файл.
formatter	DateFormat	

Таблица 19 - Описание методов класса Settings

Наименование	Параметры	Комментарий
Settings		Конструктор класса Settings, инициализирующий необходимые статические переменные, используя специальные конструкторы соответствующих классов.
getReUseAddress	int	Разрешено повторное использование адреса.
getMessageSeparator	String	Символьная строка, обозначающая начало сообщения. Сразу за messageHead следует блок адресов участников беседы.
getPort	int	Номер порта Сервера.
getTalkShowExit	String	Символьная строка - команда завершения беседы в чате. Если у участника чата бесед не осталось, то он помечается как отсутствующий в чате.
getNumberOfThreads	int	Количество резервируемых потоков в пуле потоков.
getPathToFiles	String	Путь к файлам, необходимым для работы чата.
getReferenceToLogFile	FileWriter	Ссылка на лог файл.
getLogFile	String	Файл логирования операций в чате.
getListOfUsersFile	String	JSON-файл справочника участников чата.

8. Пакет stringmsg

Объекты классов пакета обеспечивают обработку полученного и передаваемого строкового сообщения.

Схему взаимодействия классов представляет диаграмма (Рисунок 6).

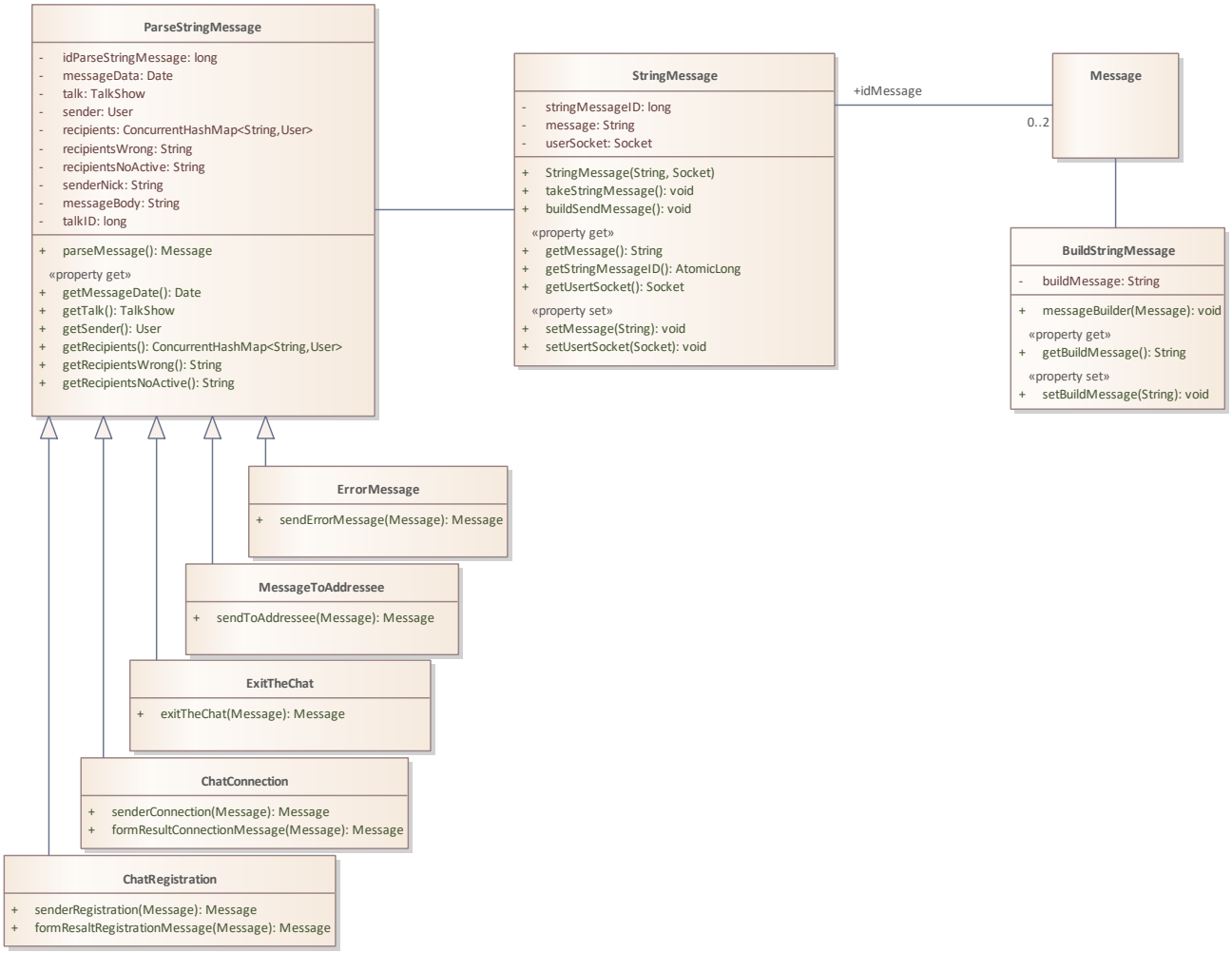


Рисунок 6 - Классы пакета stringmsg

8.1. Класс StringMessage

Класс, представляющий получаемое или отправляемое сообщение.

Каждое получаемое сообщение представляет собой строку, включающую:

- разделитель - подстрока messageSeparator и строковое представления даты и времени его отправки;
- далее за разделителем идет идентификатор беседы (idTalkShow) и разделитель; если идентификатор не определен, то подряд идут два разделителя;
- следующий блок - адрес отправителя, представленный ником отправителя

(userNickname) и разделитель;

- затем располагается блок с никами получателей сообщения, представленными никами участников беседы; ники в этом блоке разделены «точкой с запятой»; блок завершается разделителем;
- после блока получателей идет собственно тело сообщения; при регистрации участника чата в теле сообщения передается его имя;
- конец сообщения обозначен разделителем.

Обработка сообщения порождает ответное сообщение отправителю, если в сообщении обнаружены ошибки. Строка ответного сообщения об ошибке:

- начальный разделитель;
- представления даты и времени отправки ответного сообщения;
- далее за разделителем помещается код ошибки обработки исходного сообщения и разделитель;
- следующий блок - нераспознанные ники и ники не зарегистрированных в чате адресатов (если есть) и разделитель;
- затем располагается блок с никами неактивных пользователей, которым не удалось отправить сообщение (если есть); ники в этом блоке разделены «точкой с запятой»; блок завершается разделителем;
- далее идет блок тела сообщения - повтор полученного сообщения для визуального сопоставления ответа с исходным сообщением;
- конец сообщения обозначен разделителем.

Передаваемые Сервером сообщения адресатам в точности дублируют исходное сообщение, за исключением блока получателей - в этом блоке только один ник.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица, Таблица).

Таблица 20 - Описание атрибутов класса StringMessage

Наименование	Тип данных	Комментарий
stringMessageID	long	
message	String	Строка полученного или отправляемого сообщения.
userSocket	Socket	Объект (сокет), определяющий каналы взаимодействия Сервера и каждого участника чата.

Таблица 21 - Описание методов класса StringMessage

Наименование	Параметры	Комментарий
StringMessage		Конструктор класса. Принимает необработанное сообщение в виде строки для дальнейшей обработки и ссылки на входной и выходной каналы.
getMessage	String	Строка полученного или отправляемого сообщения.
setMessage	void	Строка полученного или отправляемого сообщения.
takeStringMessage	void	Метод запуска потока на обработку поступивших сообщений: ждет, когда в очереди появятся результаты потока чтения сообщений, забирает сообщение из очереди и запускает поток обработки.
getStringMessageID	AtomicLong	Получить ИД записи (например, для привязки объекта Message).
getUsertSocket	Socket	Объект (сокет), определяющий каналы взаимодействия Сервера и каждого участника чата.
buildSendMessage	void	Метод ждет результатов работы потоков обработки входящих сообщений и запускет потоки формирования и отправки соответствующих исходящих сообщений.
setUsertSocket	void	Объект (сокет), определяющий каналы взаимодействия Сервера и каждого участника чата.

8.2. Класс ParseStringMessage

Класс, выполняющий операции разбора строки поступившего сообщения.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 22, Таблица 23).

Таблица 22 - Описание атрибутов класса ParseStringMessage

Наименование	Тип данных	Комментарий
idParseStringMessage	long	
messageData	Date	Поле даты сообщения в формате Date.
talk	TalkShow	Идентифицированная беседа, полученная в сообщении.
sender	User	
recipients	ConcurrentHashMap< String,User>	Адресаты сообщения - массив ников.
recipientsWrong	String	Неопознанные или незарегистрированные получатели.
recipientsNoActive	String	Адресаты сообщения, которых нет в чате.
senderNick	String	Ник отправителя сообщения.
messageBody	String	
talkID	long	

Таблица 23 - Описание методов класса ParseStringMessage

Наименование	Параметры	Комментарий
getMessageDate	Date	Поле даты сообщения в формате Date.
getTalk	TalkShow	Идентифицированная беседа, полученная в

Наименование	Параметры	Комментарий
		сообщении.
getSender	User	
getRecipients	ConcurrentHashMap<String,User>	Адресаты сообщения - массив ников.
getRecipientsWrong	String	Неопознанные или незарегистрированные получатели.
getRecipientsNoActive	String	Адресаты сообщения, которых нет в чате.
parseMessage	Message	Разобрать строку сообщения и первично сформировать объект Message.

8.3. Класс ChatRegistration

Класс для осуществления корректной регистрации участника чата.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 24).

Таблица 24 - Описание методов класса ChatRegistration

Наименование	Параметры	Комментарий
senderRegistration	Message	Проверка корректности сообщения с запросом о регистрации. В случае отсутствия ошибок регистрация участника чата.
formResaltRegistrationMessage	Message	Формирование ответного сообщения как об ошибке в формате сообщения, так и успешной регистрации или коннекте.

8.4. Класс ChatConnection

Класс для осуществления корректного присоединения (коннекта) участника чата к беседе.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 25).

Таблица 25 - Описание методов класса ChatConnection

Наименование	Параметры	Комментарий
senderConnection	Message	Проверка корректности сообщения с запросом о коннекте. В случае отсутствия ошибок присоединение участника чата к беседе (если она есть) или создание беседы. На входе предварительно сформированный объект входящего сообщения с заполненными полями: • dataOfMessage • idStringMessage • body • direction (true).
formResultConnectionMessage	Message	

8.5. Класс MessageToAddressee

Класс для подготовки дальнейшей отправки полученных сообщений.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 26).

Таблица 26 - Описание методов класса MessageToAddressee

Наименование	Параметры	Комментарий
sendToAddressee	Message	Отправить сообщение адресату.

8.6. Класс ErrorMessage

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 27).

Таблица 27 - Описание методов класса ErrorMessage

Наименование	Параметры	Комментарий
sendErrorMessage	Message	Отправить сообщение об оштбке.

8.7. Класс ExitTheChat

Класс для осуществления выхода из чата.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 28).

Таблица 28 - Описание методов класса ExitTheChat

Наименование	Параметры	Комментарий
exitTheChat	Message	Выйти из чата.

8.8. Класс BuildStringMessage

Класс, отвечающий за сборку строки посылаемого сообщения.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 29, Таблица 30).

Таблица 29 - Описание атрибутов класса BuildStringMessage

Наименование	Тип данных	Комментарий
buildMessage	String	Собранное посылаемое сообщение.

Таблица 30 - Описание методов класса BuildStringMessage

Наименование	Параметры	Комментарий
getBuildMessage	String	Собранное посылаемое сообщение.
setBuildMessage	void	Собранное посылаемое сообщение.
messageBuilder	void	Метод-сборщик строки посылаемого сообщения. На входе: объект класса Message, на основании данных которого полностью собирается сообщение. На выходе: объект класса BuildStringMessage, который содержит собранное сообщение.

9. Пакет talkshow

Пакет включает в себя классы, отвечающие за работу с разобранным и очищенным сообщением, и обеспечивают связь с классом участника чата.

Схему взаимодействия классов представляет диаграмма (Рисунок 7).

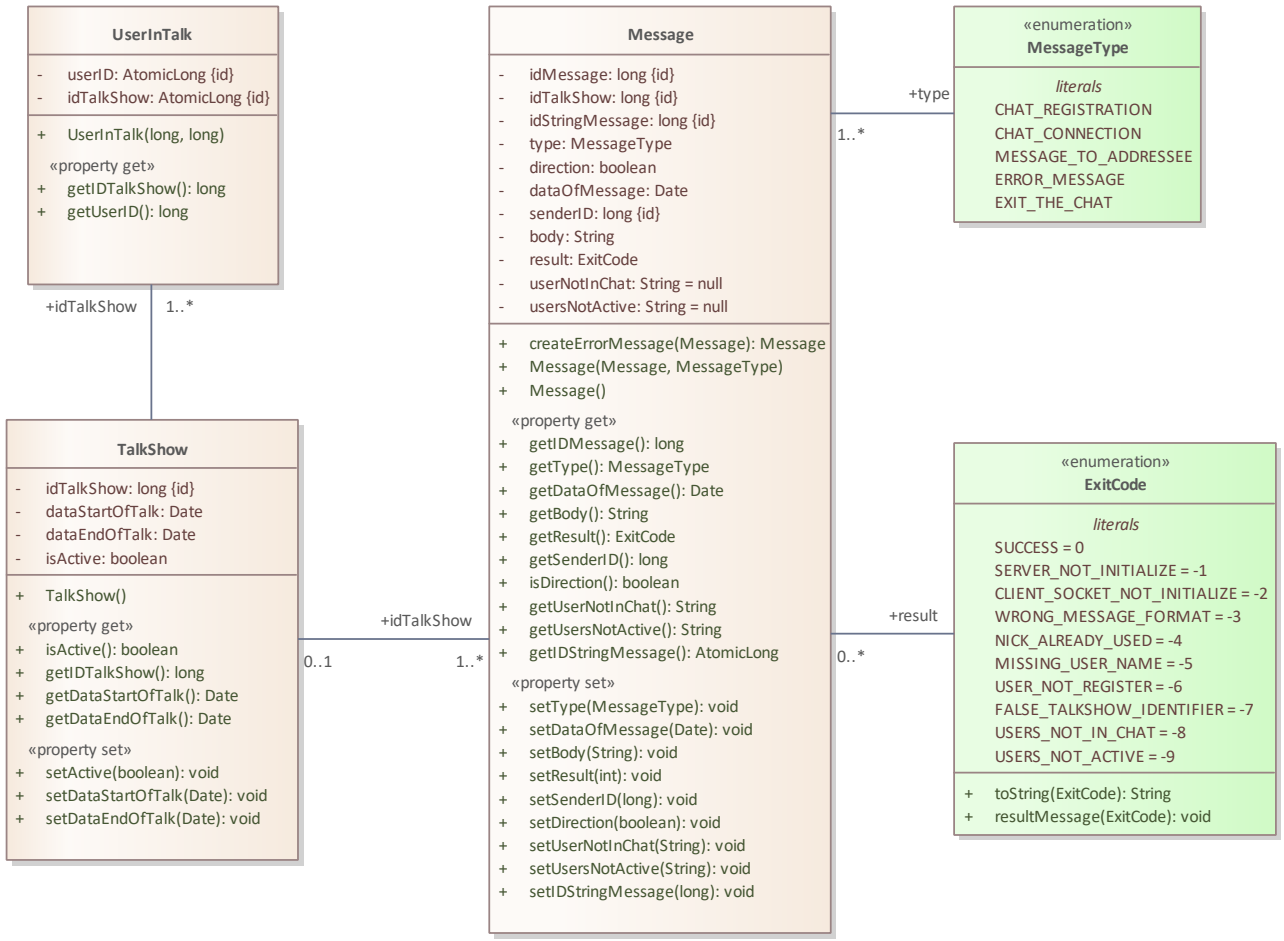


Рисунок 7 - Классы пакета talkshow

9.1. Класс Message

Класс, описывающий сообщение в чате.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 31, Таблица 32).

Таблица 31 - Описание атрибутов класса Message

Наименование	Тип данных	Комментарий
idMessage	long	Уникальный идентификатор сообщения.
idTalkShow	long	Идентификатор беседы с точки зрения Сервера.
idStringMessage	long	Ссылка на исходное полученное сообщение.
type	MessageType	Тип сообщения, представленный enum MessageType.
direction	boolean	Направление сообщения относительно обработчика (Сервера или Клиента): • true - входящее;

Наименование	Тип данных	Комментарий
		• false - исходящее.
dataOfMessage	Date	Дата и время отправки сообщения.
senderID	long	Идентификатор отправителя сообщения.
body	String	Строка тела сообщения.
result	ExitCode	Возвращаемый Сервером код завершения операции из enum ExitCode.
userNotInChat	String	
usersNotActive	String	

Таблица 32 - Описание методов класса Message

Наименование	Параметры	Комментарий
getIDMessage	long	Уникальный идентификатор сообщения.
getType	MessageType	Тип сообщения, представленный enum MessageType.
setType	void	Тип сообщения, представленный enum MessageType.
getDataOfMessage	Date	Дата и время отправки сообщения.
setDataOfMessage	void	Дата и время отправки сообщения.
getBody	String	Строка тела сообщения.
setBody	void	Строка тела сообщения.
getResult	ExitCode	Возвращаемый Сервером код завершения операции из enum ExitCode.
setResult	void	Возвращаемый Сервером код завершения операции из enum ExitCode.
getSenderID	long	Идентификатор отправителя сообщения.
setSenderID	void	Идентификатор отправителя сообщения.
isDirection	boolean	Направление сообщения относительно обработчика (Сервера или Клиента): • true - входящее; • false - исходящее.
setDirection	void	Направление сообщения относительно обработчика (Сервера или Клиента): • true - входящее; • false - исходящее.
createErrorMessage	Message	Создаем сообщение об ошибке и в конструкторе формируем поля: • idMessage • type = ERROR_MESSAGE • direction = false • dataOfMessage = <current> • senderID = <отправитель входящего сообщения> • body = <тело входящего сообщения> • UserNotInChat и UserNotActive переносятся из исходного сообщения.
getUserNotInChat	String	
setUserNotInChat	void	
getUsersNotActive	String	

Наименование	Параметры	Комментарий
setUsersNotActive	void	
Message		Конструктор, создающий ответное сообщение указанного типа.
Message		Конструктор для двух режимов: • true - инициирует счетчик идентификаторов; • false - создает новый объект.
setIDStringMessage	void	Ссылка на исходное полученое сообщение.
getIDStringMessage	AtomicLong	Ссылка на исходное полученое сообщение.

9.2. Класс TalkShow

Сеанс взаимодействия нескольких участников чата (беседа).

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 33, Таблица 34).

Таблица 33 - Описание атрибутов класса TalkShow

Наименование	Тип данных	Комментарий
idTalkShow	long	Уникальный идентификатор беседы (talkshow).
dataStartOfTalk	Date	Дата и время начала беседы.
dataEndOfTalk	Date	Дата и время окончания беседы.
isActive	boolean	Признак активности беседы: • true - активна • false - не активна.

Таблица 34 - Описание методов класса TalkShow

Наименование	Параметры	Комментарий
isActive	boolean	Признак активности беседы: • true - активна • false - не активна.
setActive	void	Признак активности беседы: • true - активна • false - не активна.
getIDTalkShow	long	Уникальный идентификатор беседы (talkshow).
getDataStartOfTalk	Date	Дата и время начала беседы.
setDataStartOfTalk	void	Дата и время начала беседы.
getDataEndOfTalk	Date	Дата и время окончания беседы.
TalkShow		Конструктор, осуществляющий присвоение идентификатора и сохранение объекта в мапе objectList.
setDataEndOfTalk	void	Дата и время окончания беседы.

9.3. Класс MessageType

Перечисление типов сообщений.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 35).

Таблица 35 - Описание атрибутов класса MessageType

Наименование	Тип данных	Комментарий
CHAT_REGISTRATION		
CHAT_CONNECTION		
MESSAGE_TO_ADDRESSEE		
ERROR_MESSAGE		
EXIT_THE_CHAT		

9.4. Класс ExitCode

Коды завершения операций в чате.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 36, Таблица 37).

Таблица 36 - Описание атрибутов класса ExitCode

Наименование	Тип данных	Комментарий
SUCCESS	int	Операция успешно исполнена
SERVER_NOT_INITIALIZE	int	Ошибка инициализации Сервера
CLIENT_SOCKET_NOT_INITIALIZE	int	Ошибка инициализации сокета Клиента
WRONG_MESSAGE_FORMAT	int	Неверный формат сообщения
NICK_ALREADY_USED	int	Пользователь с таким ником уже зарегистрирован
MISSING_USER_NAME	int	Для регистрации в чате необходимо указать имя пользователя
USER_NOT_REGISTER	int	Для участия в чате необходимо зарегистрироваться
FALSE_TALKSHOW_IDENTIFIER	int	Неверный идентификатор беседы
USERS_NOT_IN_CHAT	int	Пользователи со следующими никами не зарегистрированы в чате:
USERS_NOT_ACTIVE	int	Пользователи со слудующими никами не активны в чате:

Таблица 37 - Описание методов класса ExitCode

Наименование	Параметры	Комментарий
toString	String	По коду возвращает строку сообщения.
resultMessage	void	

10.Пакет user

Пакет содержит классы, определяющие каждого участника чата, так и обслуживающие весь справочник участников чата.

Схему взаимодействия классов представляет диаграмма (Рисунок 8).

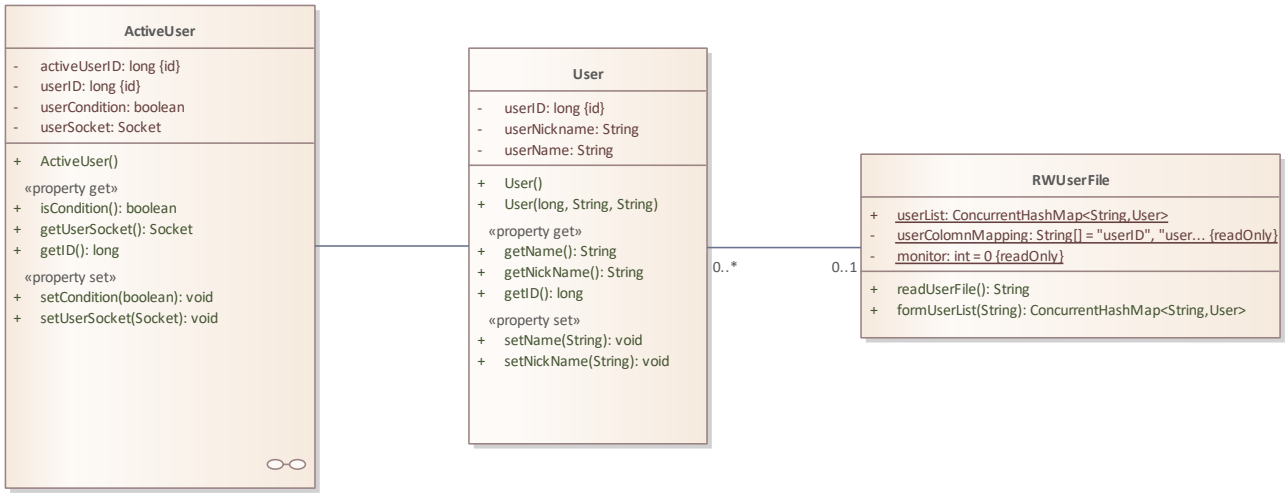


Рисунок 8 - Классы пакета user

10.1. Класс User

Данные Пользователя чата.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 38, Таблица 39).

Таблица 38 - Описание атрибутов класса User

Наименование	Тип данных	Комментарий
userID	long	Уникальный идентификатор Пользователя.
userNickname	String	Уникальный ник Пользователя чата, начинающийся с @.
userName	String	Имя Пользователя чата.

Таблица 39 - Описание методов класса User

Наименование	Параметры	Комментарий
getName	String	Получить имя участника чата.
setName	void	Сохранить имя объекта.
getNickName	String	Получить уникальный ник участника чата, начинающийся с @.
setNickName	void	Сохранить ник участника чата.
User		Конструктор, создающий объект в потоковом режиме.
getID	long	Получить уникальный идентификатор объекта.
User		Конструктор для формирования пользователя при чтении справочника пользователей из файла при инициализации Сервера.

10.2. Класс ActiveUser

Дополнительные данные участника чата, характеризующие его активность.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 40, Таблица 41).

Таблица 40 - Описание атрибутов класса ActiveUser

Наименование	Тип данных	Комментарий
activeUserID	long	Уникальный идентификатор записи.
userID	long	Идентификатор участника чата, с которым связан ActiveUser.
userCondition	boolean	Состояние Пользователя в чате: • true - есть в чате; • false - нет в чате.
userSocket	Socket	Объект (сокет), определяющий каналы взаимодействия Сервера и каждого участника чата.

Таблица 41 - Описание методов класса ActiveUser

Наименование	Параметры	Комментарий
isCondition	boolean	Участник чата в активном состоянии (true)?
setCondition	void	Установить состояние объекта: • true - активен; • false - неактивен.
getUserSocket	Socket	Объект (сокет), определяющий каналы взаимодействия Сервера и каждого участника чата.
setUserSocket	void	Объект (сокет), определяющий каналы взаимодействия Сервера и каждого участника чата.
getID	long	Получить уникальный идентификатор объекта.
ActiveUser		Конструктор.

10.3. Класс UserInTalk

Участник беседы. Класс реализующий многомножественную связь между классами User и ActiveUser.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 42, Таблица 43).

Таблица 42 - Описание атрибутов класса UserInTalk

Наименование	Тип данных	Комментарий
userID	AtomicLong	Идентификатор участника.
idTalkShow	AtomicLong	Идентификатор беседы.

Таблица 43 - Описание методов класса UserInTalk

Наименование	Параметры	Комментарий
UserInTalk		Конструктор класса, связывающий участника чата и беседу
getIDTalkShow	long	Идентификатор беседы.

Наименование	Параметры	Комментарий
getUserID	long	Идентификатор участника.

10.4. Класс RWUserFile

Класс, выполняющий операции ведения и сохранения справочника пользователей.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 44, Таблица 45).

Таблица 44 - Описание атрибутов класса RWUserFile

Наименование	Тип данных	Комментарий
userList	ConcurrentHashMap< String,User>	Мапа для поиска участников чата по имени.
userColomnMapping	String[]	Список имен атрибутов класса User для формирования json-файла.
monitor	int	Вспомогательная константа для использования в качестве монитора в блоке синхронизации потоков.

Таблица 45 - Описание методов класса RWUserFile

Наименование	Параметры	Комментарий
readUserFile	String	Метод читает файл справочника участников чата. На вход поступает путь к файлу и его имя. Результат представлен строкой данных json.
formUserList	ConcurrentHashMap<String,User>	Метод выполняет разбор поступившей на вход json-строки и формирует список ArrayList<User> учетных записей участников чата. При формировании списка создаются экземпляры класса User и определяется максимальный идентификатор экземпляра. В атрибут класса nextUserID помещается следующее за максимальным значение.

11.Пакет log

Пакет, содержащий классы, отвечающие за обеспечение логирования операций чата.

Схему взаимодействия классов представляет диаграмма (Рисунок 9).

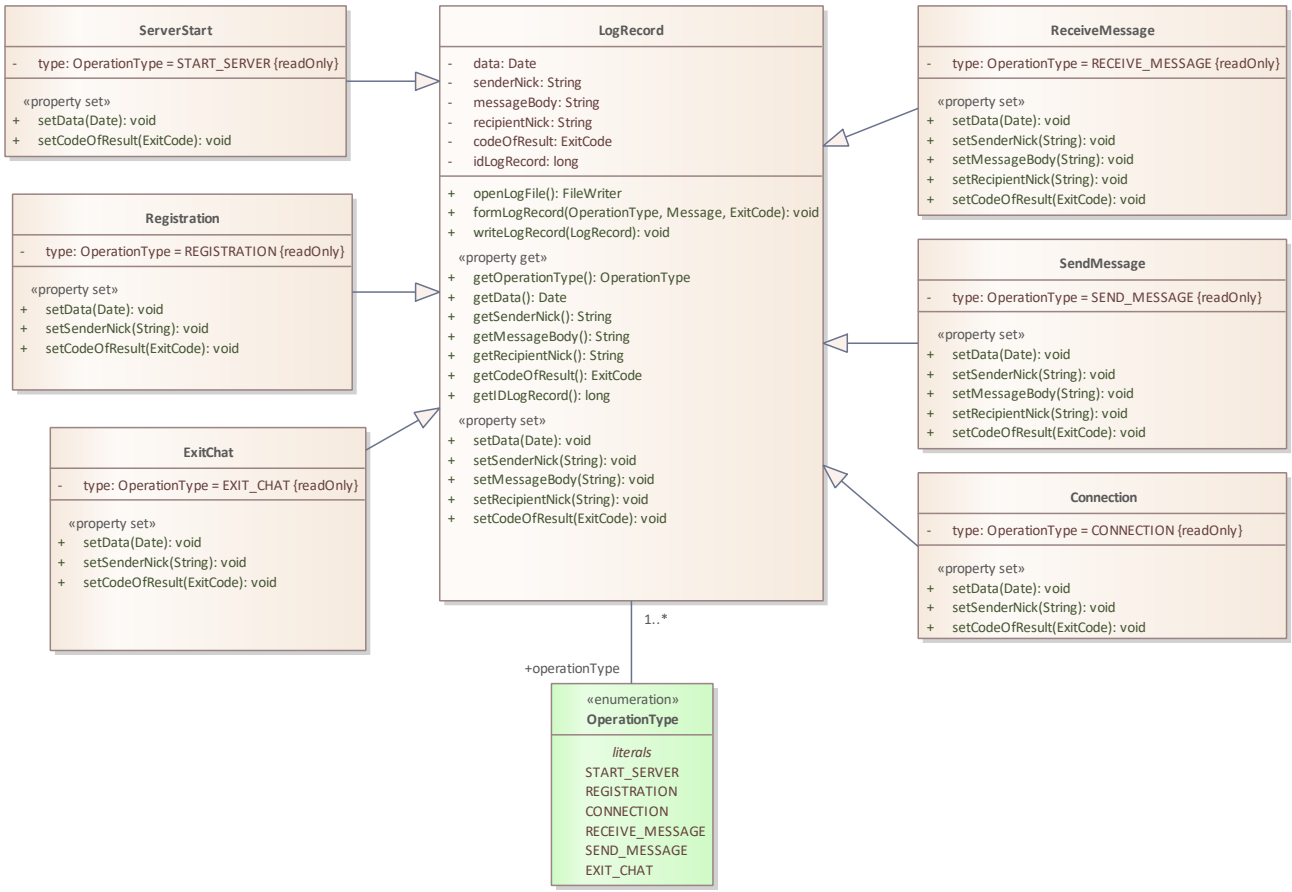


Рисунок 9 - Классы пакета log

11.1. Класс LogRecord

Структура записи лог-файла.

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 46, Таблица 47).

Таблица 46 - Описание атрибутов класса LogRecord

Наименование	Тип данных	Комментарий
data	Date	Дата и время операции.
senderNick	String	Ник отправителя сообщения.
messageBody	String	Тело сообщения.
recipientNick	String	Строка с никами получателей. В качестве разделителей используется ведущий @.
codeOfResult	ExitCode	Код завершения операции
idLogRecord	long	

Таблица 47 - Описание методов класса LogRecord

Наименование	Параметры	Комментарий
openLogFile	FileWriter	Лог-файл представляет собой файл формата json с записями, соответствующими логированным операциям чата. Метод открывает файл для дописывания записей в конец файла.
formLogRecord	void	Запись о запуске сервера содержит только дату и тип операции START_SERVER. Остальные поля пусты.
writeLogRecord	void	
getOperationType	OperationType	Тип операции.
getData	Date	Дата и время операции.
setData	void	Дата и время операции.
getSenderNick	String	Ник отправителя сообщения.
setSenderNick	void	Ник отправителя сообщения.
getMessageBody	String	Тело сообщения.
setMessageBody	void	Тело сообщения.
getRecipientNick	String	Строка с никами получателей. В качестве разделителей используется ведущий @.
setRecipientNick	void	Строка с никами получателей. В качестве разделителей используется ведущий @.
getCodeOfResult	ExitCode	Код завершения операции
getIDLogRecord	long	
setCodeOfResult	void	Код завершения операции

11.2. Класс Registration

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 48, Таблица 49).

Таблица 48 - Описание атрибутов класса Registration

Наименование	Тип данных	Комментарий
type	OperationType	

Таблица 49 - Описание методов класса Registration

Наименование	Параметры	Комментарий
setData	void	Дата и время операции.
setSenderNick	void	Ник отправителя сообщения.
setCodeOfResult	void	Код завершения операции

11.3. Класс Connection

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 50, Таблица 51).

Таблица 50 - Описание атрибутов класса Connection

Наименование	Тип данных	Комментарий
type	OperationType	

Таблица 51 - Описание методов класса Connection

Наименование	Параметры	Комментарий
setData	void	Дата и время операции.
setSenderNick	void	Ник отправителя сообщения.
setCodeOfResult	void	Код завершения операции

11.4. Класс ReceiveMessage

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 52, Таблица 53).

Таблица 52 - Описание атрибутов класса ReceiveMessage

Наименование	Тип данных	Комментарий
type	OperationType	

Таблица 53 - Описание методов класса ReceiveMessage

Наименование	Параметры	Комментарий
setData	void	Дата и время операции.
setSenderNick	void	Ник отправителя сообщения.
setMessageBody	void	Тело сообщения.
setRecipientNick	void	Строка с никами получателей. В качестве разделителей используется ведущий @.
setCodeOfResult	void	Код завершения операции

11.5. Класс SendMessage

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 54, Таблица 55).

Таблица 54 - Описание атрибутов класса SendMessage

Наименование	Тип данных	Комментарий
type	OperationType	

Таблица 55 - Описание методов класса SendMessage

Наименование	Параметры	Комментарий
setData	void	Дата и время операции.
setSenderNick	void	Ник отправителя сообщения.
setMessageBody	void	Тело сообщения.
setRecipientNick	void	Строка с никами получателей. В качестве разделителей используется ведущий @.
setCodeOfResult	void	Код завершения операции

11.6. Класс ExitChat

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 56, Таблица 57).

Таблица 56 - Описание атрибутов класса ExitChat

Наименование	Тип данных	Комментарий
type	OperationType	

Таблица 57 - Описание методов класса ExitChat

Наименование	Параметры	Комментарий
setData	void	Дата и время операции.
setSenderNick	void	Ник отправителя сообщения.
setCodeOfResult	void	Код завершения операции

11.7. Класс ServerStart

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 58, Таблица 59).

Таблица 58 - Описание атрибутов класса ServerStart

Наименование	Тип данных	Комментарий
type	OperationType	

Таблица 59 - Описание методов класса ServerStart

Наименование	Параметры	Комментарий
setData	void	Дата и время операции.
setCodeOfResult	void	Код завершения операции

11.8. Класс OperationType

Далее в таблицах приведены описания атрибутов и методов класса (Таблица 60).

Таблица 60 - Описание атрибутов класса OperationType

Наименование	Тип данных	Комментарий
START_SERVER		
REGISTRATION		
CONNECTION		
RECEIVE_MESSAGE		
SEND_MESSAGE		
EXIT_CHAT		