Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования
Национальный научно-исследовательский Университет ИТМО

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №6

по дисциплине

«Программирование»

Вариант № 3846274761835

Выполнил:
Студент группы P3106
Афанасьев Кирилл Александрович

Преподаватель:
Байрамова Хумай

Санкт-Петербург, 2023

Оглавление

Задание 3
Исходный код: 5
Диаграмма классов реализованной объектной модели. 5
Вывод 6

Задание

Разделить программу из лабораторной работы №5 на клиентский и серверный модули. Серверный модуль должен осуществлять выполнение команд по управлению коллекцией. Клиентский модуль должен в интерактивном режиме считывать команды, передавать их для выполнения на сервер и выводить результаты выполнения.

Необходимо выполнить следующие требования:

- Операции обработки объектов коллекции должны быть реализованы с помощью Stream API с использованием лямбда-выражений.
- Объекты между клиентом и сервером должны передаваться в сериализованном виде.
- Объекты в коллекции, передаваемой клиенту, должны быть отсортированы по умолчанию
- Клиент должен корректно обрабатывать временную недоступность сервера.
- Обмен данными между клиентом и сервером должен осуществляться по протоколу UDP
- Для обмена данными на сервере необходимо использовать **датаграммы**
- Для обмена данными на клиенте необходимо использовать **сетевой канал**
- Сетевые каналы должны использоваться в неблокирующем режиме.

Обязанности серверного приложения:

- Работа с файлом, хранящим коллекцию.
- Управление коллекцией объектов.
- Назначение автоматически генерируемых полей объектов в коллекции.
- Ожидание подключений и запросов от клиента.
- Обработка полученных запросов (команд).
- Сохранение коллекции в файл при завершении работы приложения.
- Сохранение коллекции в файл при исполнении специальной команды, доступной только серверу (клиент такую команду отправить не может).

Серверное приложение должно состоять из следующих модулей (реализованных в виде одного или нескольких классов):

- Модуль приёма подключений.
- Модуль чтения запроса.
- Модуль обработки полученных команд.
- Модуль отправки ответов клиенту.

Сервер должен работать в **однопоточном** режиме.
Обязанности клиентского приложения:

- Чтение команд из консоли.

- Валидация вводимых данных.
- Сериализация введённой команды и её аргументов.
- Отправка полученной команды и её аргументов на сервер.
- Обработка ответа от сервера (вывод результата исполнения команды в консоль).
- Команду `save` из клиентского приложения необходимо убрать.
- Команда `exit` завершает работу клиентского приложения.

Важно! Команды и их аргументы должны представлять из себя объекты классов. Недопустим обмен "простыми" строками. Так, для команды add или её аналога необходимо сформировать объект, содержащий тип команды и объект, который должен храниться в вашей коллекции.

Дополнительное задание:

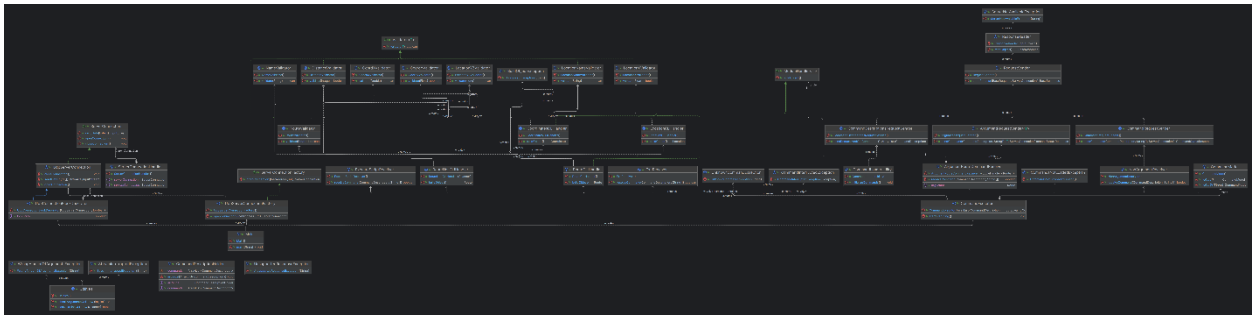
Реализовать логирование различных этапов работы сервера (начало работы, получение нового подключения, получение нового запроса, отправка ответа и т.п.) с помощью **Log4J2**

Исходный код:

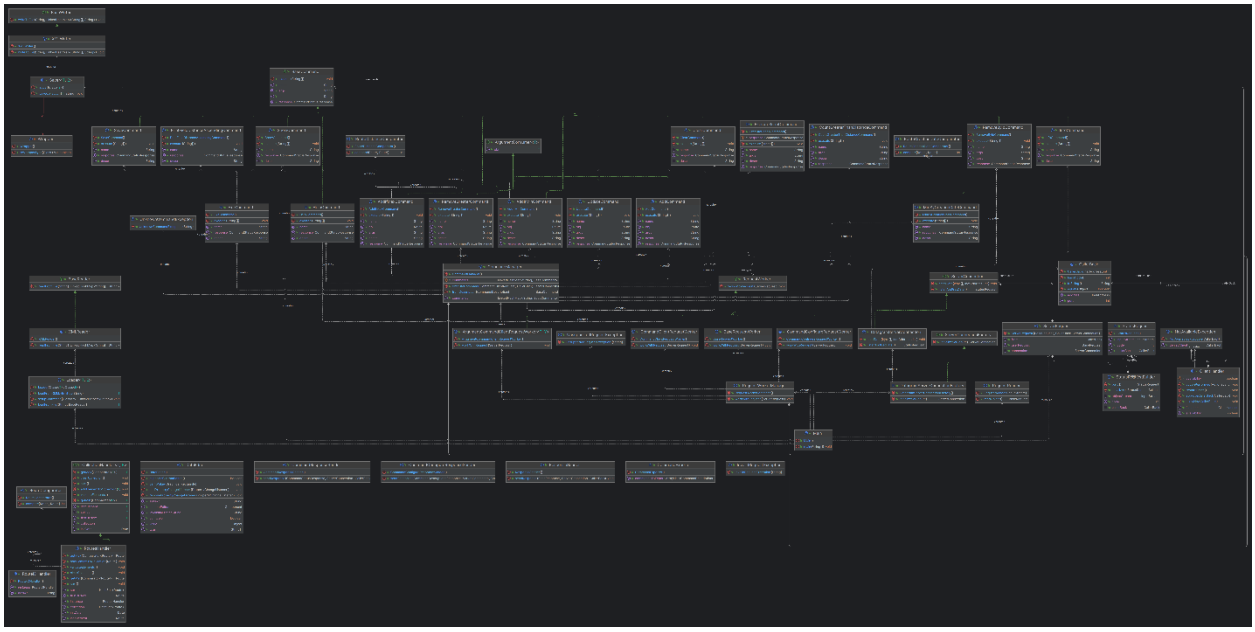
Репозиторий: https://github.com/Zerumi/no6_150323_3846274761835

Диаграмма классов реализованной объектной модели.

Клиент:



Сервер:



Вывод в UML-формате см. в репозитории.

Вывод

Во время выполнения данной лабораторной работы я

- Ознакомился с некоторыми моделями сетевого взаимодействия (одноранговые сети/клиент-серверная архитектура), семействами протоколов сетевого обмена (TCP/IP, VoIP), конкретными протоколами (TCP, UDP) для изучения способов современной организации передачи информации по сети.

- Ознакомился со способами представления объектов для сетевого обмена (сериализация).

- Ознакомился с прикладными способами простейшей реализации сетевого обмена (с использованием датаграмм и сетевого канала).

- Ознакомился с некоторыми базовыми шаблонами проектирования: Adapter, Observer, Singleton, Strategy, Factory Method, Decorator, Proxy, Facade, Command, Iterator, а также Flyweight и Interpreter.

- Применил некоторые полученные знания (в том числе полученные из прошлых работ) на практике:

- - Расширил свою предыдущую лабораторную работу на клиентскую и серверную часть, разработал новые классы следуя некоторым из шаблонов проектирования (Observer, Factory method, Decorator, Strategy, Singleton, Command), с целью дальнейшего расширения программы.

- - Реализовал «слабого» клиента: клиент не зависит от конкретных загружаемых команд, лишь от их сигнатур, а сами команды легко заменяются без необходимости обновлять клиент.

- - Изучил возможности IDE и системы сборки Gradle по запуску и компиляции нескольких приложений в одном проекте.

Полученные мною знания будут использоваться для дальнейшего изучения языка и обучения в целом.

Спасибо за внимание!