|  |
| --- |
|  |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования  **"МИРЭА - Российский технологический университет"**  **РТУ МИРЭА** |
| Институт информационных технологий (ИТ) |
| Кафедра инструментального и прикладного программного обеспечения |

|  |  |
| --- | --- |
| **ОТЧЕТ**  **ПО ПРАКТИЧЕСКОЙ РАБОТЕ №2** | |
| **по дисциплине** |  |
| **«Архитектура клиент-серверных приложений»**  **Тема: Концепция удаленного вызова метода (RMI) в JAVA** | |
| Выполнил студент группы: ИКБО-20-21 | Костенко П.А. |
| Принял преподаватель кафедры ИиППО | Волков М.Ю. |

|  |  |  |
| --- | --- | --- |
| Практические работы выполнены | « » 2023 г. |  |
| «Зачтено» | « » 2023 г. |  |

Москва 2023

**Теоретическое введение**

RMI - это объектно-ориентированный эквивалент RPC (удаленного вызова процедур). Это называется вызовом удаленного метода. Вызов удаленного метода (RMI) позволяет объекту Java вызывать метод объекта, запущенного на другом компьютере. RMI обеспечивает удаленную связь между Java-программой.

Главной целью разработчиков RMI было предоставление возможности программистам разрабатывать распределенные Java программы, используя такие же синтаксис и семантику, как и при разработке обычных нераспределенных программ. Для этого они должны были преобразовать модель работы классов и объектов в одной виртуальной машине Java (JVM) в новую модель работы классов и объектов в распределенной (несколько JVM) вычислительной среде.

Приложение RMI можно разделить на две части. Одним из них является программой клиентом и другая программа сервер. Сервер программа создает некоторые удаленный объект, сделать их ссылки доступны для клиента для вызова метода на нем. Клиент программа делает запрос для удаленных объектов на сервере и вызвать метод на них. Заглушка и скелет - два важных объекта, используемых для связи с удаленным объектом.

Таким образом, запустив открытый сервер RMI в системе, можно разрешить внешним субъектам взаимодействовать с ним и, возможно, выполнять методы на сервере RMI. Эти методы должны быть определены в реализации Сервера. Как только они вызываются клиентом, они будут выполняться на сервере, а возвращаемые значения будут возвращены клиенту. Еще одна интересная часть заключается в том, что собственный RMI (опять же, я НЕ говорю о JMXRMI) не поддерживает большую часть безопасности, кроме шифрования соединения с использованием SSL.

Реализация RMI, по существу, состоит из трех абстрактных уровней. Первый – это уровень заглушки и скелета, расположенный непосредственно перед разработчиком. Этот уровень перехватывает вызовы методов, произведенные клиентом при помощи переменной-ссылки на интерфейс, и переадресует их в удаленную службу RMI.

Следующий уровень – уровень удаленной ссылки. Этот уровень понимает, как интерпретировать и управлять ссылками на удаленные объекты служб. В JDK 1.1 этот уровень соединяет клиентов с удаленными объектами служб, которые исполняются на сервере. Это соединение является связью типа один к одному (однонаправленное соединение). В Java 2 SDK этот уровень был расширен поддержкой активации пассивных удаленных объектов при помощи технологии Remote Object Activation.

Транспортный уровень основан на соединениях ТСР/IР между сетевыми машинами. Он обеспечивает основные возможности соединения и некоторые стратегии защиты от несанкционированного доступа.

При использовании уровневой архитектуры каждый из уровней может быть изменен или заменен без воздействия на остальную систему. Например, транспортный уровень может быть заменен протоколом UDP/IP без изменения остальных уровней.

Имена «заглушка» и «скелет» могут сбивать с толку на первый взгляд, но это просто то, как называются «клиентская» и «серверная» части удаленного объекта.

Stub - это класс, реализующий удаленный интерфейс и служащий в качестве заполнителя на стороне клиента для удаленного объекта. С другой стороны, Skeleton — это серверная сущность, которая отправляет вызовы фактической реализации удаленного объекта.

Заглушка и скелетон - два важных объекта, используемых для связи с удаленным объектом.

Уровень заглушки и скелета RMI расположен непосредственно перед разработчиком Java. На этом уровне RMI использует прокси-модель проектирования, которая описана в книге Gamma, Helm, Johnson и Vlissides «Design Patterns». В прокси-модели объект одного контекста представляется другим (прокси-объектом) в отдельном контексте. Прокси-объект знает, как направлять вызовы методов между этими объектами.

В прокси-модели, используемой в RMI, роль прокси выполняет класс заглушки, а роль RealSub j ect выполняет класс, реализующий удаленную службу.

Скелет является вспомогательным классом, который создается для использования RMI. Скелет понимает, как взаимодействовать с заглушкой при RMI соединении. Скелет поддерживает общение с заглушкой; он читает параметры для вызова метода из соединения, производит вызов объекта, реализующего удаленную службу, принимает возвращаемое значение и записывает его обратно в заглушку.

В реализации RMI Java 2 SDK новый протокол связи сделал классы скелетов не нужными. RMI использует отражение для установления соединения с объектом удаленной службы. Вы должны использовать классы и объекты скелетов только в JDK 1.1 и совместимых с ним реализациях систем.

**Постановка задачи**

Использую информацию из описания данной практической работы, необходимо реализовать удалённый метод решения квадратных уравнений общего вида ax2 + bx + c = 0. При этом условие уравнения передавать на сервер, а клиентская часть должна получать результат в виде объектов пользовательского класса. Клиент и сервер должны работать на одном хосте.

**Программный код**

На рисунке 1 представлен интерфейс калькулятора.

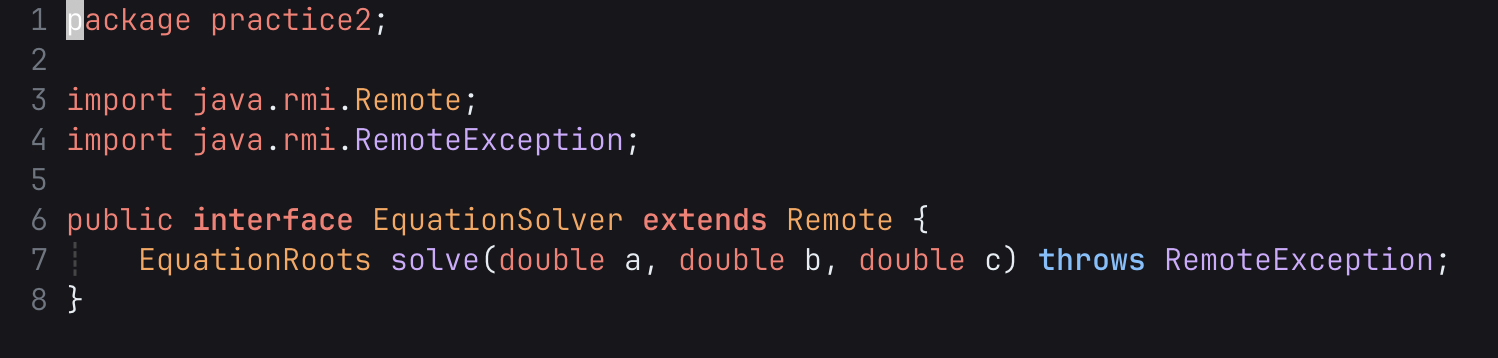
****

Рисунок 1 – Интерфейс

На рисунке 2 представлена реализация интерфейса.

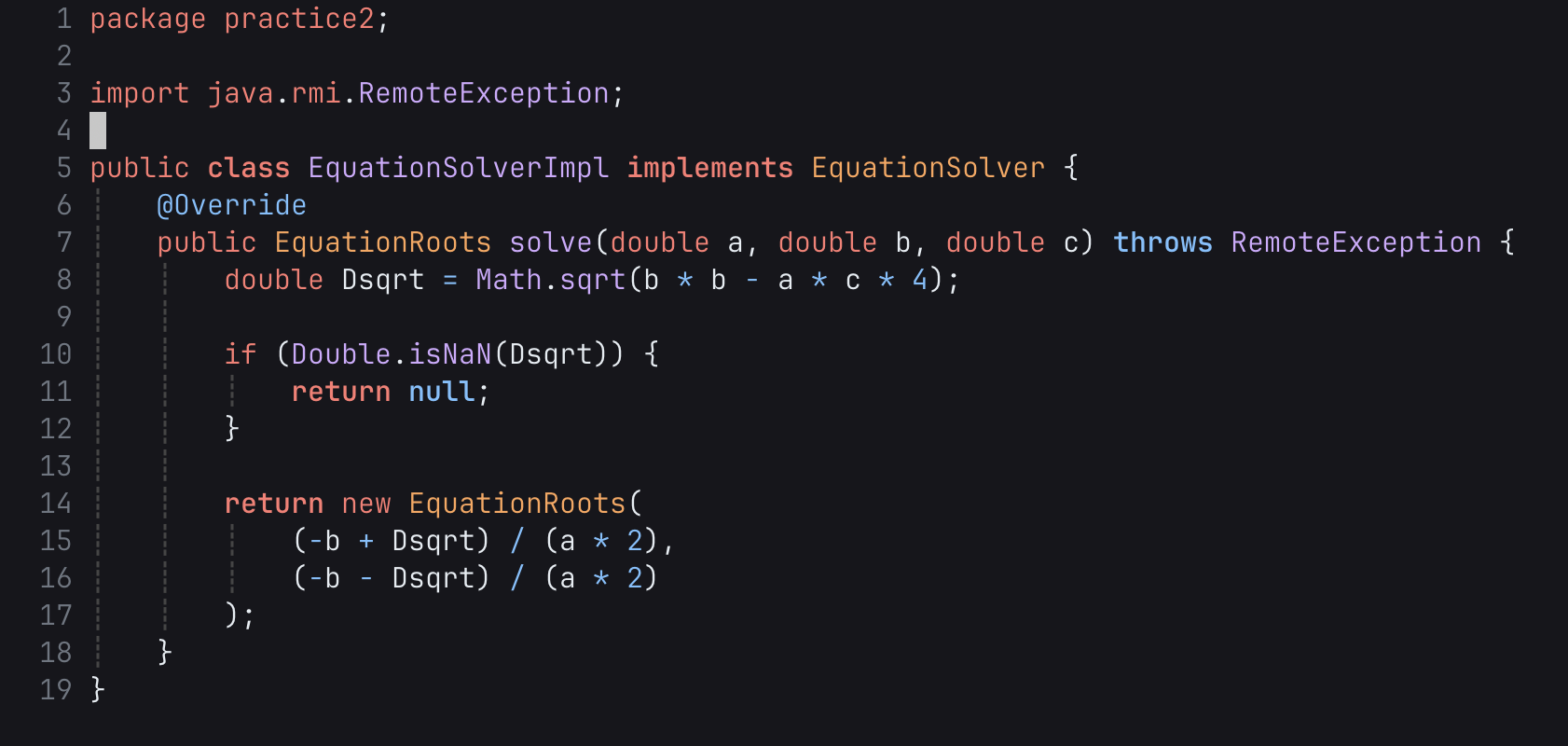
****

Рисунок 2 – Реализация интерфейса

На рисунке 3 представлен класс ответа сервера.

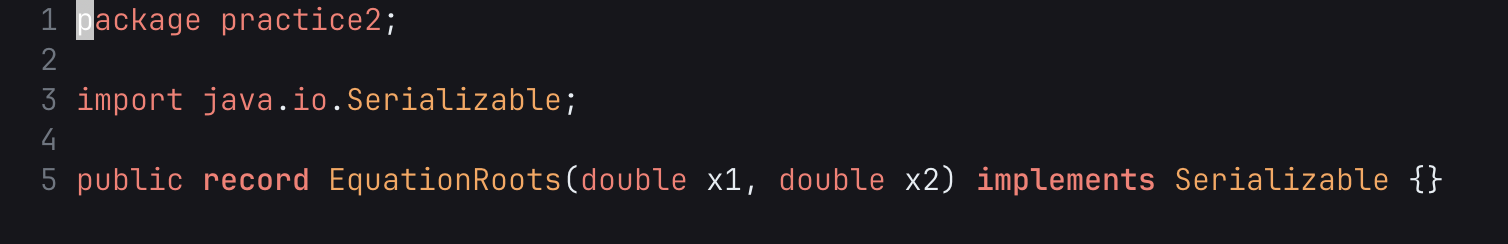


Рисунок 3 – class Answer

На рисунке 4 представлен класс Main сервера.



Рисунок 4 – Класс Main сервера

На рисунке 5 представлен класс Main клиента.

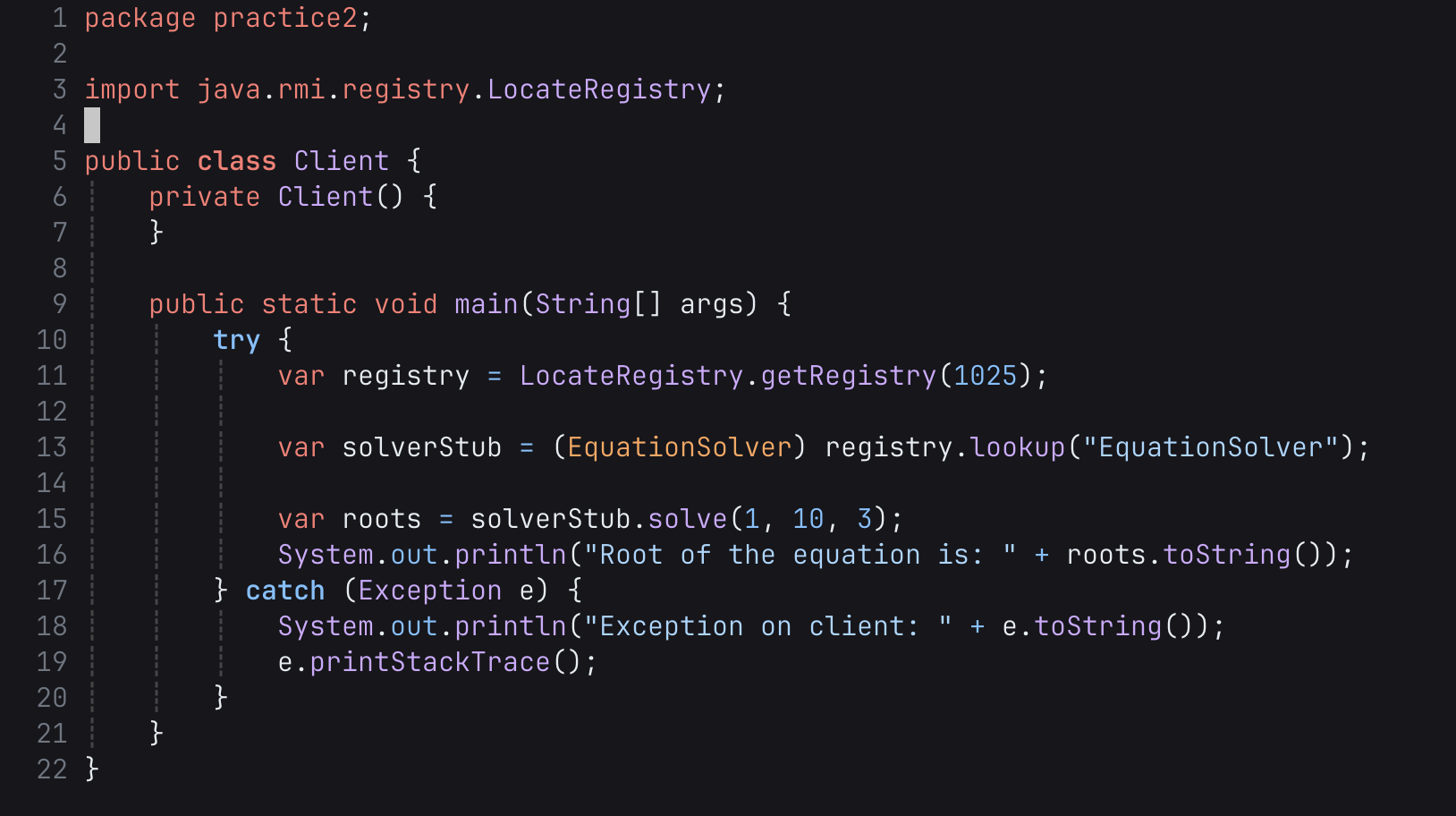


Рисунок 5 – класс Main клиента

**Вывод программы**

На рисунке 6 показан запуск сервера.

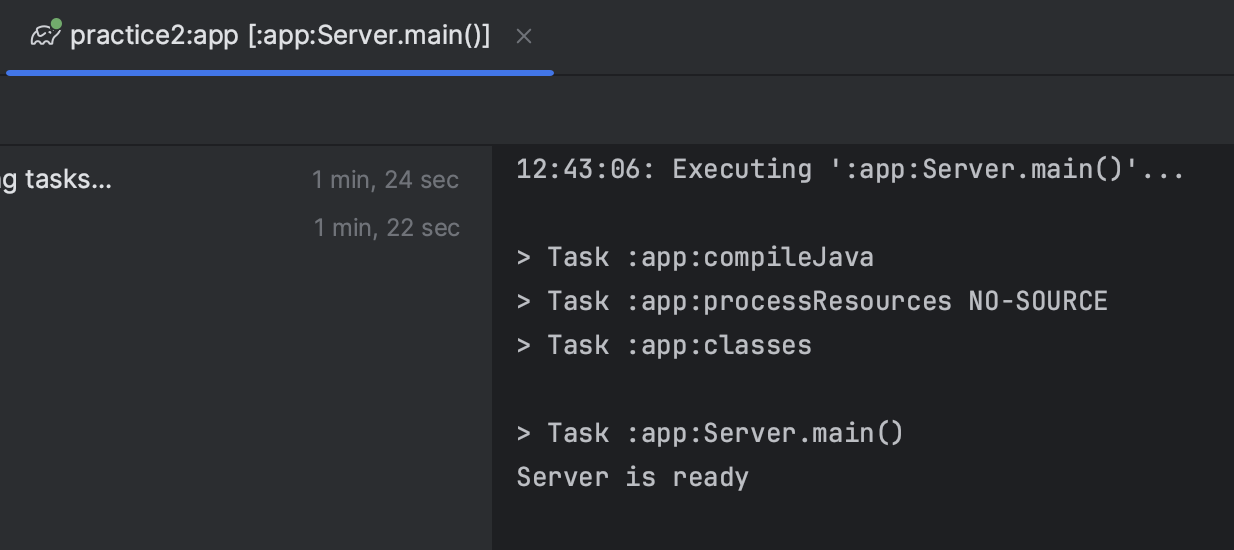


Рисунок 6 – Работа сервера

На рисунке 7 представлена работа клиента.

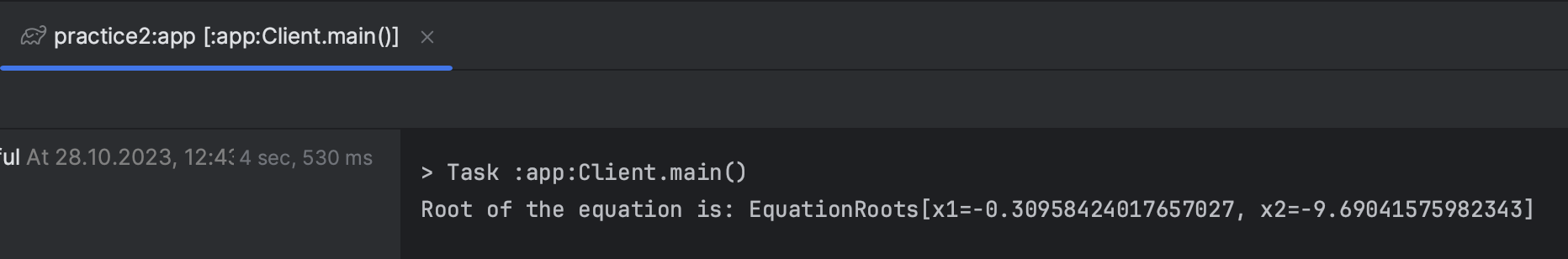


Рисунок 7 – Работа клиента

**Вывод**

В результате выполнения практической работы были получены навыки работы с механизмами удаленного вызова процедур в JAVA. И был реализован удалённый метод решения квадратных уравнений.

**Список литературы**

1. Документация Java 17 от Oracle [Электронный ресурс]: URL – <https://docs.oracle.com/en/java/javase/17/docs/api/index.html> [Дата обращения 25.09.2023];
2. Руководство по языку программирования Java [Электронный ресурс]: URL – <https://metanit.com/java/tutorial/> [Дата обращения 25.09.2023];
3. RMI: практика использования [Электронный ресурс]: URL – <https://javarush.com/groups/posts/2283-rmi-praktika-ispoljhzovanija> [Дата обращение 25.09.2023];
4. RMI (Remote Method Invocation) [Электронный ресурс]: URL – <https://habr.com/ru/articles/74639/> [Дата обращение 25.09.2023];