**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики и информатики**

**КОЗЕЛ ЮЛИЯ ВАЛЕНТИНОВНА**

**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

студентки 2 курса 13 группы

**Руководитель**

*Орешко И.Г.*

Минск 2016**Содержание**

[1. Отчет по лабораторной работе №1 3](#_Toc451863266)

[2. Отчет по лабораторной работе №2 4](#_Toc451863267)

[3. Отчет по лабораторной работе №3 5](#_Toc451863268)

[4. Отчет по лабораторной работе №4 6](#_Toc451863269)

[5. Отчет по лабораторной работе №5 7](#_Toc451863270)

[6. Отчет по лабораторной работе №6 8](#_Toc451863271)

[7. Отчет по лабораторной работе №7 9](#_Toc451863272)

[8. Отчет по лабораторной работе №8 10](#_Toc451863273)

[9. Отчет по лабораторной работе №9 11](#_Toc451863274)

[10. Отчет по лабораторной работе №10 13](#_Toc451863275)

[11. Отчет по лабораторной работе №11 15](#_Toc451863276)

1. Отчет по лабораторной работе №1

Постановка задачи:

* создать класс, реализующий интерфейс Shape для рисования указанной линии.
* создать приложение или апплет для тестирования и демонстрации разработанных классов.
* выполнить указанные в задании перемещения указанной в задании фигуры с помощью аффинного преобразования координат.
* выполнить рисунок в окне апплета или фрейма с выбранной толщиной границы фигуры, цветом границы и цветом внутренней области.

Особенности реализации:

Перемещения данной фигуры происходят с помощью аффинного преобразования координат. Это происходит, благодаря методу, который мы переопределяем в классе-наследнике.

Вращение происходит вокруг одного из концов отрезка. Был создан класс, реализующий Shape, являющийся оберткой над Line2D.

Результат работы программы:

Данное приложение было разработано с использованием графического интерфейса, результат работы которого представлен на рисунке 1.

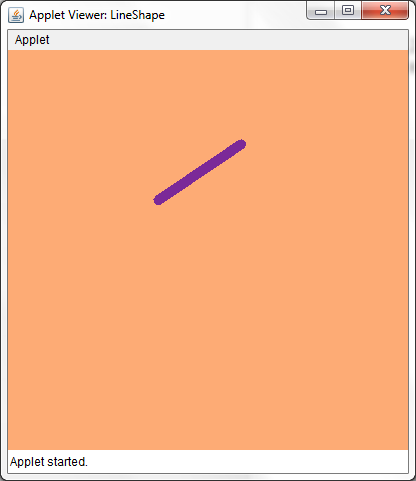


Рисунок 1 – Скриншот работы приложения. Вращающаяся линия

1. Отчет по лабораторной работе №2

Постановка задачи:

* для изображения указанной в задании фигуры создать класс, реализующий интерфейс Shape;
* создать указанный фильтр изображения; при тестировании вывести фигуру без фильтра и с фильтром (аналогично фильтрам из примеров);
* моделировать освещение и тень от объекта при помощи альфа-канала и/или механизма обработки изображения;
* при рисовании использовать сглаживание, внеэкранный буфер и преобразования координат.

Особенности реализации:

При рисовании использовалось сглаживание, внеэкранный буфер и преобразования координат. При изменении объекта, от которого отбрасывается тень, тень не должна изменяться. Поддержка возможности освещения с разных углов.

Результат работы программы:

Данное приложение было разработано с использованием графического интерфейса, результат работы которого представлен на рисунке 2.

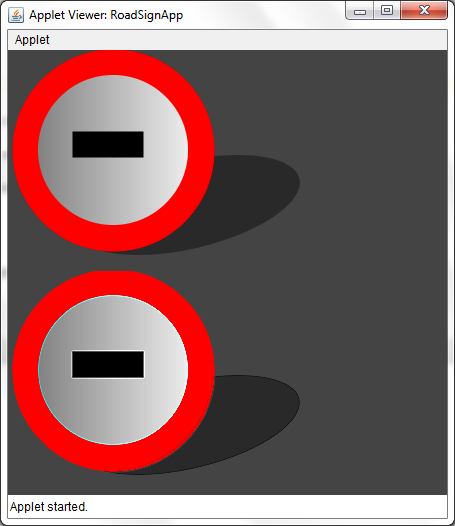


Рисунок. 2 – Скриншот работы приложения

1. Отчет по лабораторной работе №3

Постановка задачи:

* Разработайте пользовательский класс Shape реализующий рисование указанной алгебраической линии.
* Разработайте пользовательский класс Stroke для отображения указанного контура
* Создайте приложение или апплет для тестирования и демонстрации разработанных классов.

Особенности реализации:

Для рисования фигуры (строфоида), использовался класс, поддерживающий Shape. Был написан свой класс Stroke в виде зубьев, для отображения контура.

Результат работы программы:

Данное приложение было разработано с использованием графического интерфейса, результат работы которого представлен на рисунке 3.

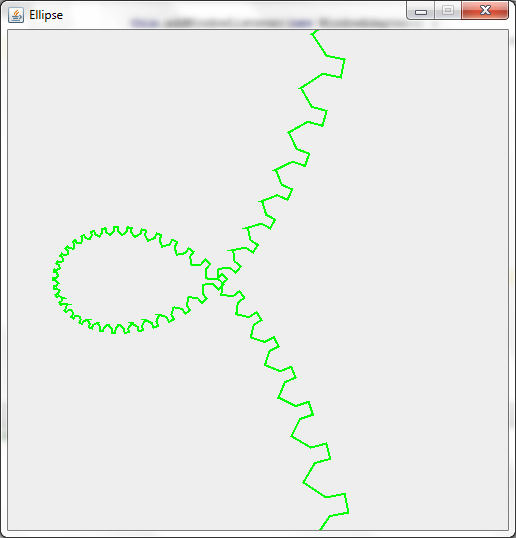


Рисунок. 3 – Скриншот работы приложения. Алгебраическая линия

1. Отчет по лабораторной работе №4

Постановка задачи:

* Разработайте пользовательский класс Shape реализующий рисование указанной алгебраической линии.
* Разработайте пользовательский класс Stroke для отображения указанного контура
* Создайте приложение или апплет для тестирования и демонстрации разработанных классов.
* Распечатать исходный код Класса, который строит алгебраическую линию.

Особенности реализации:

Для рисования фигуры (строфоида), использовался класс, поддерживающий Shape. Был написан свой класс Stroke в виде зубьев.

Для печати в файл использовался класс, поддерживающий интерфейс Printable. Для каждой страницы читались строки из файла с исходным кодом и выводились построчно.

Результат работы программы:

Рисунок с подписью представлен на рисунке 4.

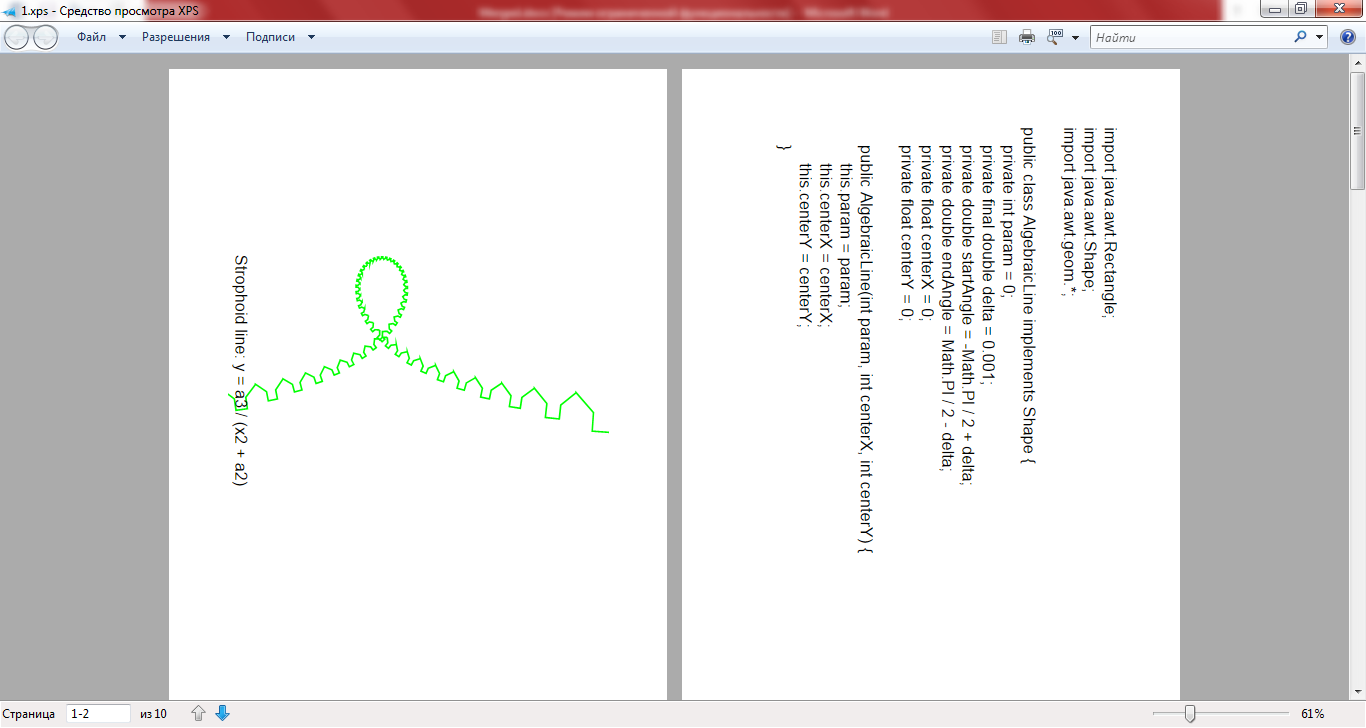


Рисунок. 4 – Скриншот работы приложения. Файл для печати

1. Отчет по лабораторной работе №5

Постановка задачи:

* Разработайте пользовательский класс Shape реализующий рисование указанной алгебраической линии.
* Разработайте пользовательский класс Stroke для отображения указанного контура.
* Создайте тестовое приложение, добавьте в ваш класс рисования алгебраической линии возможность «перетаскивание» (drag-and-drop). Реализуйте необходимые интерфейсы в классе и в приложении для демонстрации «перетаскивания» алгебраической линии между несколькими копиями тестового приложения

Особенности реализации:

Для рисования фигуры (строфоида), использовался класс, поддерживающий Shape. Был написан свой класс Stroke в виде зубьев, для отображения контура. Для «перетаскивания» был реализован класс Scribble поддерживающий следующие интерфейсы: Shape, Transferable, Serializable, Cloneable, а также класс ScribbleDragAndDrop, который обрабатывает жесты Gesture Recognizer.

Результат работы программы:

Данное приложение было разработано с использованием графического интерфейса, результат работы которого представлен на рисунке 5.

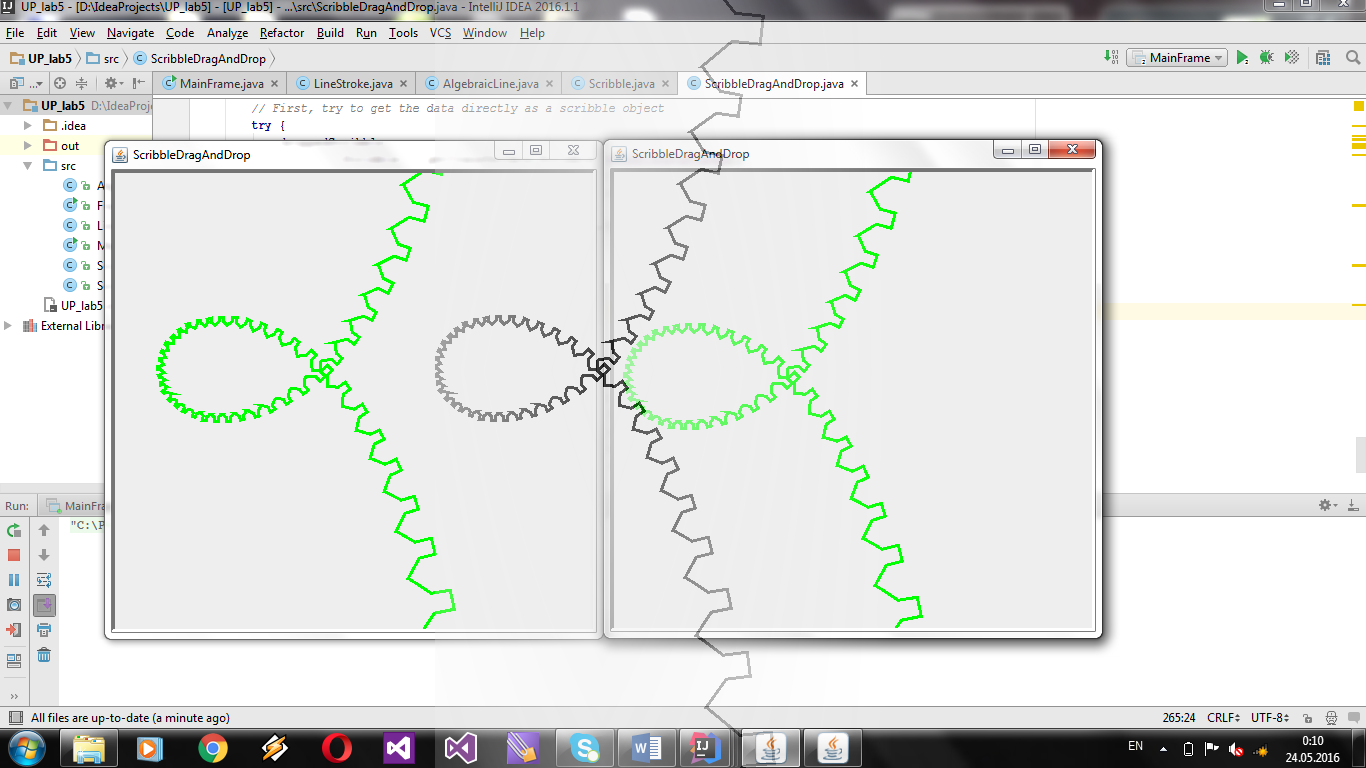


Рисунок. 5 – Скриншот работы приложения. Алгебраическая линия

1. Отчет по лабораторной работе №6

Постановка задачи:

* разработайте систему классов/интерфейсов для картотеки отдела кадров. Данные необходимо упорядочить в виде дерева.
* разработайте графическое приложение для ввода/отображения данных. При отображении структуры данных в виде дерева реализуйте интерфейс javax.swing.Tree.TreeModel. Листья дерева отображайте в виде таблицы, для этого реализуйте интерфейс javax.swing.table.TableModel. (пример похожего приложения – Проводник Windows)
* при реализации интерфейса следуйте рекомендациям стандарта CUI (Common User Interface).

Особенности реализации:

Были реализованы: класс, наследующийся от JTree, класс реализующий интерфейс MyTreeModel, TreeModel, класс MyTableModel реализующий интерфейс TableModel, класс Node наследующийся от DefaultMutableTreeNode, класс для хранения данных о сотрудниках.

**Результат работы программы:**

Данное приложение было разработано с использованием графического интерфейса, результат работы которого представлен на рисунке 6.

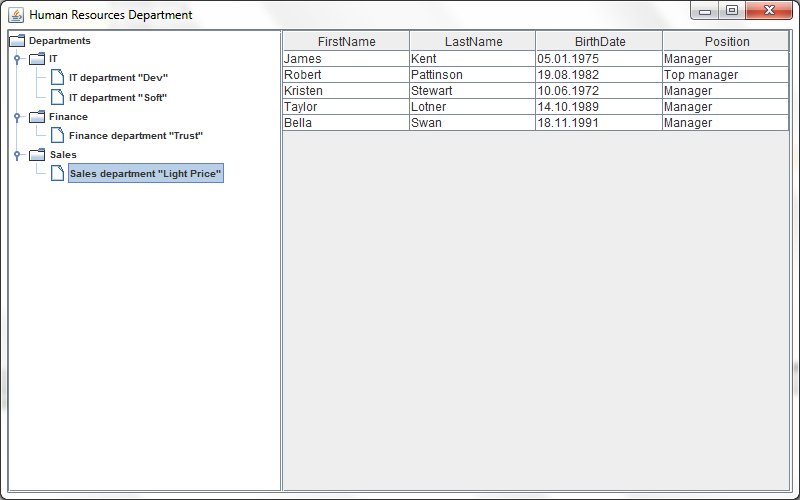


Рисунок. 6 – Скриншот работы приложения

1. Отчет по лабораторной работе №7

Постановка задачи:

* Разработайте простой компонент 3D горизонтальная линия (свойства: ширина) на базе класса Canvas.
* Создайте файл манифеста и упакуйте компонент вместе с исходным кодом разработанных классов.
* Создайте тестовое приложение в NetBeans с использованием вашего компонента.

Особенности реализации:

Для того чтобы подготовить компонент к использованию в контейнере, необходимо было упаковать его вместе с необходимыми ему файлами и ресурсами в файл JAR. Поскольку один компонент может иметь много вспомогательных файлов и поскольку файл JAR может содержать несколько компонентов, в манифесте JAR-файла должно быть описано, какие элементы файла являются компонентами.

**Результат работы программы:**

Данное приложение было разработано с использованием графического интерфейса, результат работы которого представлен на рисунке 7.

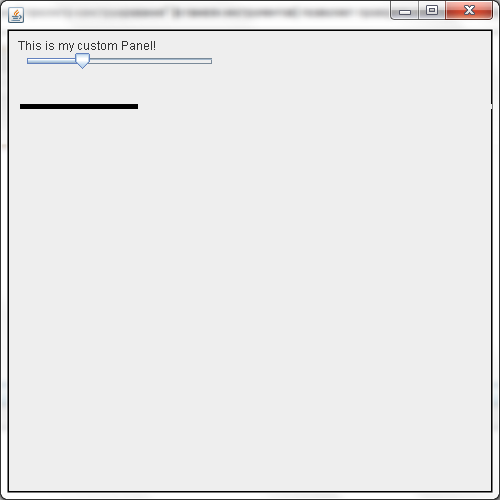


Рисунок. 7 – Скриншот работы приложения. Разработанный компонент

1. Отчет по лабораторной работе №8

Постановка задачи:

* Разработать свой компонент: однострочный статический текст, две зависимых радио-кнопки и обычная кнопка.
* Свойства: текст, текст кнопки, текст радио-кнопок. Событие генерируется при нажатии на обычную кнопку. Событие передаёт ещё и состояние радио-кнопок. Тест должен передаваться по нажатию кнопки и по нажатию клавиши “Enter”
* Компонент должен реализовать класс BeanInfo с информацией о компоненте.
* Создать тестовое приложение в NetBeans с использованием компонента.

Особенности реализации:

Был реализован класс, генерирующий собственное событие, который наследуется от стандартного события. Затем компонент был скомпилирован, создан манифест, который собрал класс в архив jar, который используется для добавления компоненты в тестовое приложение. В тестовом приложении появляется возможность добавить собственное событие и обработать его

Результат работы программы:

Данное приложение было разработано с использованием графического интерфейса, результат работы которого представлен на рисунке 8.

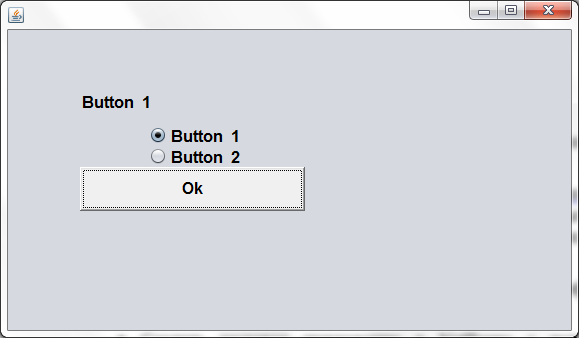


Рисунок. 8 – Скриншот работы приложения. Разработанный компонент

1. Отчет по лабораторной работе №9

Постановка задачи:

* Разработать свой компонент: однострочный статический текст, две зависимых радио-кнопки и обычная кнопка.
* Свойства: текст, текст кнопки, текст радио-кнопок. Событие генерируется при нажатии на обычную кнопку. Событие передаёт ещё и состояние радио-кнопок. Тест должен передаваться по нажатию кнопки и по нажатию клавиши “Enter”
* Компонент должен реализовать класс BeanInfo с информацией о компоненте.
* Создать тестовое приложение в NetBeans с использованием компонента.
* Реализовать собственный редактор свойств. Каждый редактор свойства ограничивает возможные значения свойств выбором из трёх – пяти допустимых значений.
* Регистрировать редакторы в классе BeanInfo компонента.

Особенности реализации:

Был реализован класс, генерирующий собственное событие, который наследуется от стандартного события. Затем компонент был скомпилирован, создан манифест, который собрал класс в архив jar, который используется для добавления компоненты в тестовое приложение. В тестовом приложении появляется возможность добавить собственное событие и обработать его.

Был создан класс для свойства однострочного текста в панели и его редактор. Редактор реализует интерфейс PropertyEditorSupport и в методе getTags возможные для выбора значения заданные в выпадающем меню. Редактор регистрируется в классе BeanInfo компонента, в методе setPropertyEditorClass. В тестовом приложении появляется возможность редактировать свои свойства и выбирать из возможных значений заданных в выпадающем меню.

Результат работы программы:

Свойства компонента, которые можно задать, представлены на рисунке 9. Сам компонент представлен на рисунке 10.

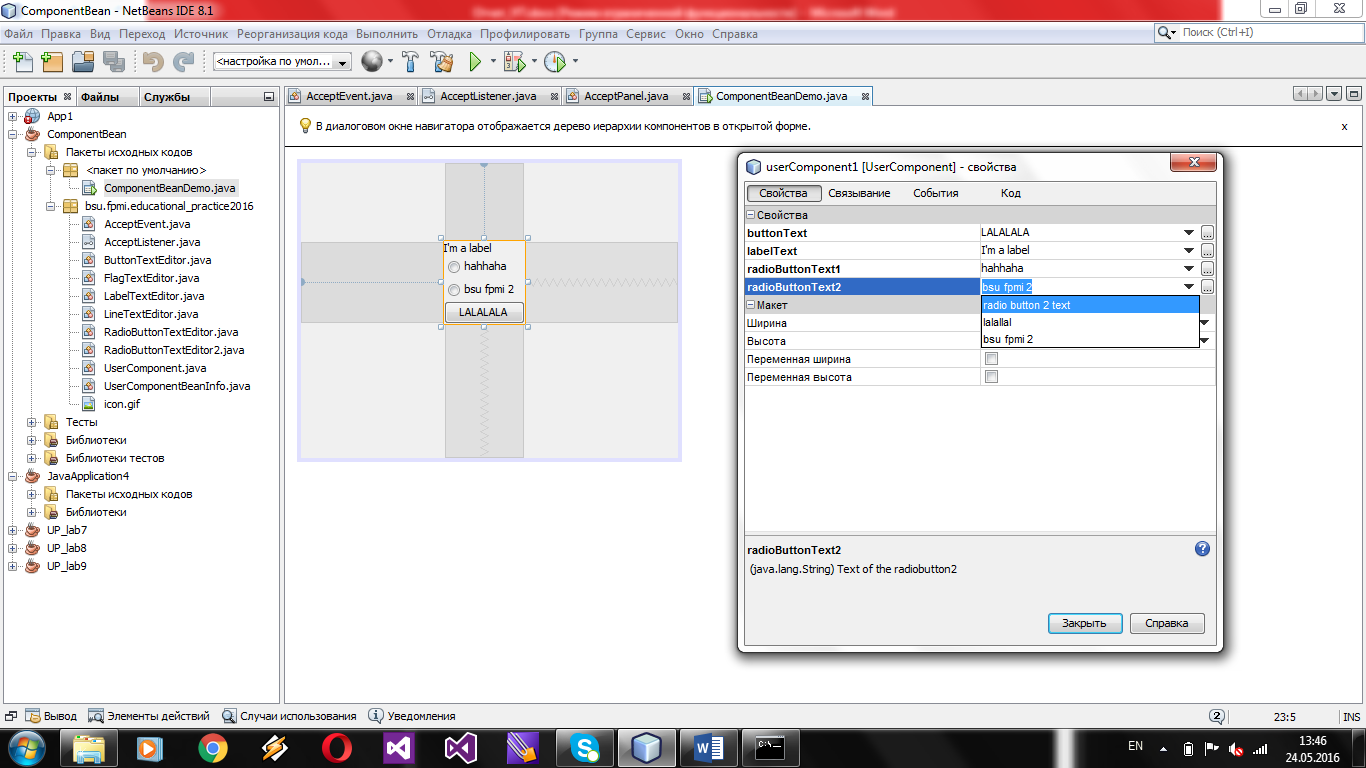


Рисунок. 9 – Скриншот работы приложения. Среда разработки с компонентом

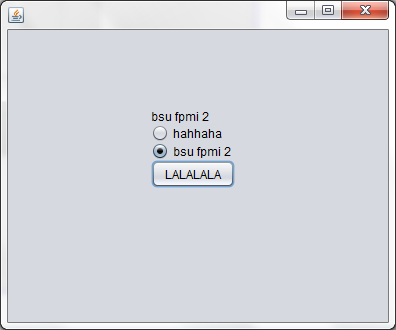


Рисунок. 10 – Скриншот работы приложения. Компонент

1. Отчет по лабораторной работе №10

Постановка задачи:

* Разработать классы для представления данных и протокол взаимодействия клиента и сервера для решения указанной задачи. Создать на основе сокетов многопоточное клиент/серверное приложение.
* Игра крестики-нолики по сети.

Особенности реализации:

Сервер постоянно находится в состоянии ожидания, он прослушивает сеть, ожидая запросов от клиентов. Клиент связывается с сервером и посылает ему запрос с описанием услуги, например, имя нужного файла.

Запросы клиента и ответы сервера формируются по строгим правилам, совокупность которых образует протокол связи (реализован в Protocol.java). Для обеспечения передачи данных по протоколу TCP основным классом является java.net.Socket. Socket(InetAddress address, int port) - создает соединение и подключает его к заданному порту по заданному IP-адресу.

Сервер содержит цикл, ожидающий нового клиентского соединения. Каждый раз, когда клиент просит соединения, сервер создает новый поток.

Результат работы программы:

Скриншоты работы данного приложения представлены на рисунках 11, 12 и 13.

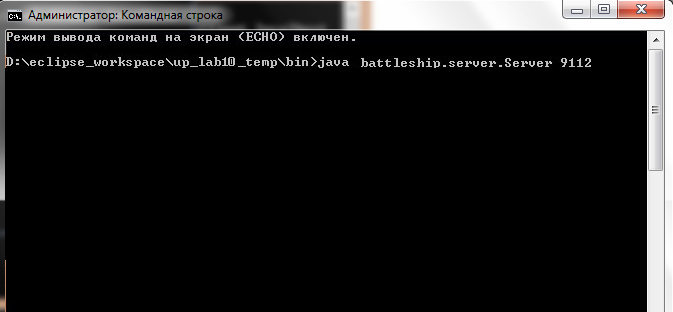


Рисунок. 11 – Скриншот работы приложения. Серверное приложение

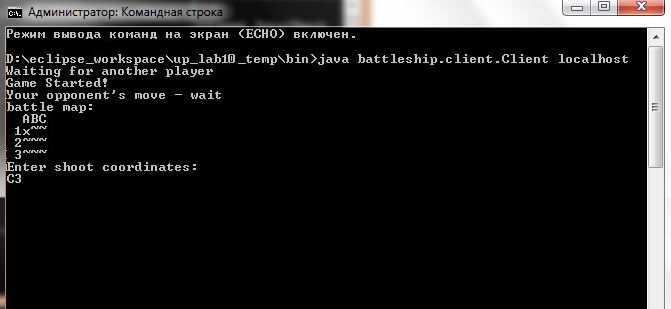


Рисунок. 12 – Скриншот работы приложения. Клиентское приложение 1

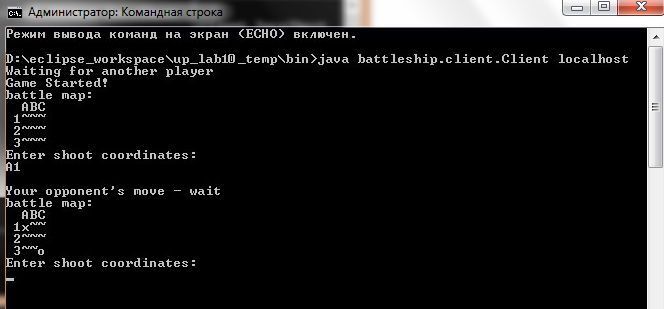


Рисунок. 13 – Скриншот работы приложения. Клиентское приложение 2

1. Отчет по лабораторной работе №11

Постановка задачи:

* проанализируйте ваш вариант задания. Можно ли его реализовать как часть MUD системы (например, в одной из комнат MudPlace), требуется ли для этого внести изменения в парадигму MUD? При реализации, по возможности, используйте парадигму MUD и классы примера 2 при реализации вашего варианта задания.
* создайте на основе технологии RMI клиент/серверное приложение:

Игра по сети в “Крестики-нолики”. Игра между двумя клиентами через сервер. Клиент предлагает поиграть и ждёт, пока другой клиент согласится на игру. Сервер организует связь между играющими клиентами.

Особенности реализации:

Вызов удаленных методов является мощной технологией для разработки сетевых приложений, освобождающей программиста от необходимости заботиться о деталях реализации сетевых соединений на нижнем уровне.

Каждый объект Room представляет собой отдельную комнату внутри MUD. У каждой комнаты есть имя, описание, список находящихся в ней предметов и людей (пользователей), выходы из этой комнаты и другие комнаты, в которые эти выходы ведут.

Результат работы программы:

Окно клиента 1 представлено на рисунке 14, клиента 2 на рисунке 15. Окно сервера представлено на рисунке 16.

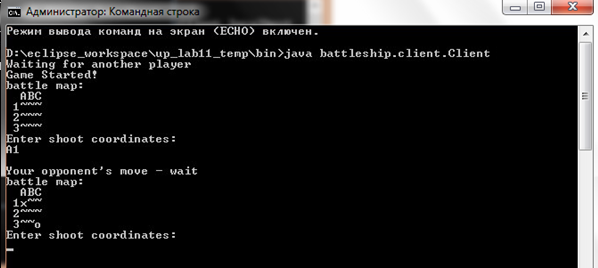


Рисунок. 14 – Скриншот работы приложения. Клиентское приложение 1

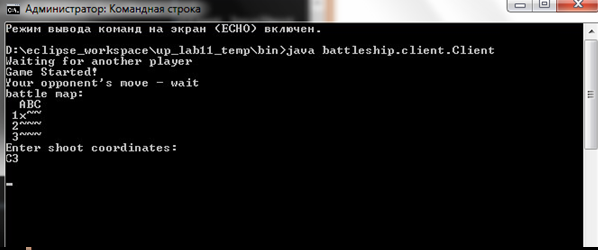


Рисунок. 15 – Скриншот работы приложения. Клиентское приложение 2

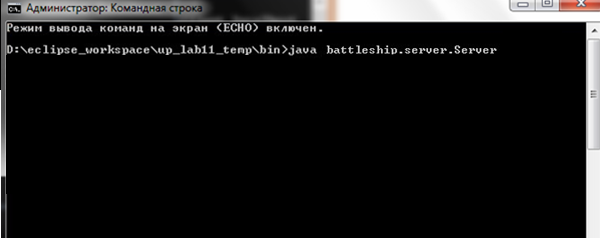


Рисунок. 16 – Скриншот работы приложения. Серверное приложение