Министерство науки и образования Украины

Харьковский национальный университет им. В.Н. Каразина

Факультет компьютерных наук

**Курсовая работа**

**"Tic Tac Toe"**

Выполнили:

студент группы КС - 21

Ридозуб Олег

студентка группы КС - 22

Струк Алена

студент группы КС - 22

Балюк Константин

Проверил:

ст. преп. кафедры искусственного интеллекта

и программного обеспечения

Литвинов Д. Н.

Харьков – 2016

**ОГЛАВЛЕНИЕ**

ВВЕДЕНИЕ.……………………………………………………...……………….3

1. АНАЛИЗ

1.1. Постановка задачи и основные требования……………..………4

1.2. Перечень основных возможностей………………………............4

1.3 Дополнительные сведения или/и системные требования……….7

1. РАЗРАБОТКА ПРОДУКТА

2.1. Руководство программиста……………………..………...............7

2.2. Диаграммы классов………………………………………………14

2.3. Используемые технологии………………………………………16

1. ОПИСАНИЕ РЕЗУЛЬТАТОВ

3.1. Руководство пользователя………………………………….……17

1. РАБОТА КОМАНДЫ

4.1. Таблица распределения обязанностей………………………….23

4.2. Не очень японская поэзия……………………………………….23

ВЫВОДЫ……………………………………………………………………….25

ПЕРЕЧЕНЬ ССЫЛОК…………………………………………………….........27

**ВВЕДЕНИЕ**

Целью данной работы является ознакомление с основными механизмами объектно-ориентированного программирования и применением их на практике посредством написания десктоп-приложения на языке java, грамотное распределение обязанностей, а также знакомство с проектированием, методами удаленной командной разработки и написанием документации.

Актуальность работы невозможно оспорить, поскольку все, исследованное командой в процессе разработки приложения, в дальнейшем будет необходимо для успешной разработки более сложных программ, уменьшения количества затрачиваемых на программирование времени и усилий.

Кроме того, игра крестики-нолики (англ. TicTacToe), выбранная нами в качестве основы для приложения, является классикой мировой истории настольных игр. Наша команда приняла решение разработать программу, предоставляющую возможность игры в неё без необходимости использования конспектов, парт и досок. Цифровая версия популярной игры не может не снискать славу среди пользователей.

Программа должна давать возможность игры двух человек на одном компьютере, игры человека против компьютера и компьютера против компьютера для демонстрации игрового процесса. Помимо этого, было решено добавить в приложение запись, хранение и вывод статистики, а также пользовательские настройки интерфейса.

1. **АНАЛИЗ**

**1.1. Постановка задачи и основные требования**

Реализации всех идей, перечисленных во введении, необходимо достичь максимально простыми и удобными для этого методами, а также разработать интуитивно-понятный пользовательский интерфейс десктоп-приложения, не вызывающий проблем в использовании.

Решено было использовать в качестве игрового поля массив кнопок, добавить возможность смены цветовой схемы окна, а также хранить в файле информацию о десяти последних играх: имена игроков, результат партии и ее длительность.

После принятия решения о хранении статистики стало необходимым добавить возможность ввода имен игроков, а также выбора очередности хода: крестик или нолик делает первый ход.

Кроме того, наша команда сочла необходимым внести в приложения правила игры, чтобы человек, не знакомый с ними, также мог использовать приложение без всяческий затруднений.

Язык, использующийся в приложении, должен быть понятен большинству пользователей, следовательно, из таких соображений для реализации игры был выбран английский.

**1.2. Перечень основных возможностей**

1. При запуске программы открывается главное меню с пунктами "Play", "Settings", "Stats", "Rules". Каждая из этих кнопок открывает панель соответствующих возможностей.
2. По умолчанию к окну приложения применяется цветовая схема CS\_Milk. Поменять ее можно в настройках, выбрав понравившуюся из перечня доступных.
3. При нажатии на кнопку Play будет открыта панель выбора игровых настроек. В ней можно указать тип первого и второго игрока (компьютер или человек), а также их имена, под которыми они будут занесены в статистику. После того как выбор сделан, пользователь должен нажать на кнопку “READY”.
4. При нажатии кнопки “READY” будет открыта панель с игровым полем. В зависимости от типа игры, игрок должен нажимать на те кнопки, куда хочет поставить свой знак, и ждать хода другого игрока. При нажатии на клетку, в которой знак уже стоит, ничего не произойдёт. В таком случае необходимо выбрать пустую клетку и поставить знак в неё.
5. Игра завершается, когда на поле не остается свободных клеток (ничья - *(англ.)* tie), либо когда один из игроков собрал три своих знака по вертикали, горизонтали или диагонали, что повлечет его победу[1]. При сборе выигрышной тройки соответствующие клетки тут же меняют цвет, показывая победу, поле становится неактивным, происходит небольшая задержка для наглядности, после чего демонстрируется панель результатов, где, в зависимости от исхода, будет написано “Имя\_игрока wins!” либо “Tie!”. После нажатия на панель происходит возврат в меню.
6. Кнопка меню “Stats” открывает панель статистики, на которой в виде таблицы демонстрируются результаты последних 10 партий в формате “имя\_первого\_игрока | результат | имя\_второго\_игрока | длительность партии (мс)”. Кнопка “Clear stats” безвозвратно стирает статистику, “Return to menu” - возвращает пользователя на панель меню.
7. Кнопка “Settings” открывает панель с двумя настройками визуальной части приложения. Список “Color Scheme” имеет на выбор некоторое число цветовых схем, от которых зависят цвета фонов шрифтов и даже знаков при игре, цветовая схема демонстрируется тут же. Список схем рассчитан на большое их количество, имеет полосу прокрутки при переполнении области видимости. “Settings” - пункт настроек, в котором предусматривались мелкие настройки игры. Кнопка “Return to menu” возвращает пользователя в меню, сбрасывая все изменения. Для сохранения изменений используется кнопка “Confirm”. Она вернет пользователя в меню, применив все изменения.
8. Завершает список возможностей пункт меню “Rules”. Здесь на двух языках RU/ENG описаны базовые правила игры в крестики нолики. Для возвращения в меню предусмотрена кнопка “Return to Menu”.

**1.3. Дополнительные сведения или/и системные требования**

Так как приложение написано на языке Java, его использование возможно на всех самых распространенных операционных системах (Windows, Linux, MacOS), главное требование – существование для этой операционной системы Java Runtime Environment.

Существование JRE для вашей операционной системы можно проверить на официальном сайте Oracle [1].

Требования пользователей:

* Минималистичный интерфейс;
* Низкая нагрузка на аппаратную часть;
* Простота в использовании.

**2. РАЗРАБОТКА ПРОДУКТА**

**2.1. Руководство программиста**

Программа содержит 2 пакета, 18 классов и 1 интерфейс. Далее изложено подробное описание функций и предназначения каждого:

**Пакет Graphics**

Большая часть классов в этом пакете наследует классы библиотеки javax.swing и предназначается для реализации графического контента приложения.

* **class XOButton**

Этот класс является наследником класса JButton и предназначается для реализации ячейки игрового поля. Он содержит информацию о своих координатах в массиве (используется для отправки команды игровому полю после клика пользователя), а также о своей доступности и занятости. Кроме того, имеет две иконки - крестика и нолика, которые устанавливает на себя после клика в зависимости от значка игрока.

Такая кнопка может обновлять свои цвета в соответствии с выбранной цветовой схемой и подсвечиваться в случае, если участвует в выигрышной комбинации.

* **class Menu**

Класс меню является наследником класса JFrame и содержит в себе функцию main, которая создает экземпляр этого же класса. Класс является “емкостью” для добавления панелей с различными функциями. Все эти панели создаются в конструкторе класса, занимают одно и тоже место в форме, и становятся видимыми при необходимости. В связи с этим класс содержит множество полей и методов для работы с графикой.

* **Подпакет CS**

В данном подпакете лежат все цветовые схемы, доступные в данный момент для применения к окну приложения.

* + **public abstract class ColorScheme**

Этот класс является основой для всех цветовых схем. Он содержит необходимые поля и служит своего рода напоминанием при создании наследников.

* + **public class CS\_Ginger**

Оранжево-белая цветовая схема. Класс наследует ColorScheme.

* + **public class CS\_Milk**

Черно-бежевая цветовая схема, которая используется по умолчанию при открытии приложения. Класс наследует ColorScheme.

* **Подпакет Panels**

В данном подпакете все классы являются наследниками класса JPanel и используются для реализации панелей с основными возможностями приложения. Все они обладают стандартным набором методов для JPanel, а также умеют обновлять цвета своих объектов в соответствии с выбранной цветовой схемой. Каждая панель создается при запуске программы в единственном экземпляре.

* + **public class FieldPanel**

Данный класс реализует игровое поле. Содержит двумерный массив XOButton размером n x n, что есть размерность поля. Неявно наследует методы XOButton для подсветки кнопки, ее очистки от символа и нажатия на нее.

* + **public class MMenuPanel**

Эта панель - главное меню - открывается при запуске приложения и содержит 4 кнопки: "Play", "Stats", "Settings", "Rules".

* + **public class NamePanel**

Такая панель используется после нажатия на кнопку "Play the game" для ввода дополнительной информации о партии. Она содержит два идентичных блока для первого и второго игроков. В каждом блоке содержится группа JRadioButton, позволяющая выбрать тип игрока: Human или Computer, и текстовое поле для ввода имени.

При переключении между JRadioButton создаются экземпляры соответствующих классов игроков. При этом в поле для имени для человека появляется Mr. Nobody (для первого игрока) или Ms. Nobody (для второго игрока); для компьютера имя выбирается из 10 стандартных имен, предложенных нашей командой. Пользователь может изменить имя для обоих типов игроков по собственному желанию.

Также панель содержит JButton “Ready”, нажатие на которую сохраняет внесенные пользователем изменения и активирует панель игрового поля.

* + **public class OffResPanel**

На этой панели содержится всего одна кнопка, на которой появляются надписи “Tie!” или “Player\_Name Wins!” в зависимости от результата партии после ее завершения. Нажатие на кнопку возвращает пользователя в главное меню.

Обратим внимание на название панели: панель для режима игры оффлайн. В дальнейших версиях программы планируется также разработка возможности игры по сети.

* + **public class RulesPanel**

Панель предназначена для хранения правил игры и содержит многострочный (с использованием html) JLabel правил игры и кнопку возвращения в меню.

* + **public class SettingsPanel**

Объект этого класса позволяет настраивать цветовые схемы окна. Для этого нужно выбрать одну цветовую схему из перечня доступных. После клика на ее название, включается предпросмотр темы.

Также в панели настроек есть возможность выбора очередности хода: крестик, затем нолик или нолик, затем крестик. Выбор осуществляется посредством группы JRadioButton.

Чтобы сохранить настройки, пользователю необходимо нажать “Confirm”. Чтобы вернуться в меню без сохранения результатов - кнопку “Return to the menu”.

* + **public class StatsPanel**

Данная панель предназначена для вывода статистики на экран. Она содержит двумерный массив JLabel, информация для заполнения которого передается из неявно наследуемого класса Statistic, а также две кнопки: для возвращения в меню и обнуления статистики. Первая активирует панель главного меню, а вторая неявно наследует метод обнуления из класса Statistic.

Также класс содержит методы для обновления информации и очистки панели.

**Пакет Logic**

Этот пакет содержит основную часть программного кода приложения. В нем реализованы все возможности программы на данном этапе ее развития.

* **public class Field**

Данный класс реализует игровое поле и является основой для полей разных размеров. Он содержит 4 поля: n - размер игрового поля, матрицу занятости ячеек игрового поля размера n\*n типа boolean и матрицу значений (крестик или нолик) в ячейках такого же размера и типа. Содержит следующие методы:

* + static boolean getOccupancy(int y, int x) возвращает true, если клетка с координатами ( x , y ) занята; false в противном случае.
  + static boolean getElement(int y, int x) возвращает значение, лежащее в клетке с координатами ( x , y ). False для крестика, true для нолика.
  + static void setElement(int y, int x, boolean val) устанавливает в клетку с координатами ( x , y ) элемент val. Соответствующей клетки в матрице занятости присваивает значение true.
  + static boolean isFull() возвращает true, если поле заполнено полностью. False в противном случае.
  + static boolean isEmpty() возвращает true, если поле пустое. False в противном случае.
  + abstract boolean isWin(boolean value) проверяет, есть ли на поле выигрышная комбинация. Реализуется в наследниках.
* **private class Field3x3**

Этот класс наследует класс Field и реализует вариант игры с полем размера 3\*3. Не содержит собственных полей, наследует поля родителя. Переопределяет метод boolean isWin(boolean value): метод возвращает true, если на поле есть выигрышная комбинация, а также подсвечивает соответствующие кнопки на поле с игровой панелью.

* **public interface Player**

Интерфейс Player используется в качестве шаблона для классов PC и Human. Он содержит основные методы (String getName, void SetName, void toMove: вернуть имя, назначить имя и походить. соответственно), необходимые для существования игрока, а также позволяет использование полиморфизма в процессе игры.

* **public class Human**

Данный класс реализует интерфейс Player и предназначается для человека. Он содержит единственное поле - имя игрока. В методе toMove разблокирует для нажатия кнопки поля посредством неявного наследования статического метода из класса Menu.

* **public class PC**

Класс PC реализует интерфейс Player и предназначается для игрока-искусственного интеллекта. Также содержит поле с именем игрока. В методе toMove посредством определенной последовательности операций определяет оптимальную позицию для установки символа, после чего разблокирует поле для себя и нажимает на соответствующую кнопку.

* **public class Result**

Этот класс предназначается для сохранения пакета результатов игры. Содержит 4 поля: имя первого игрока, имя второго, результат игры в формате:

“количество\_побед\_первого\_игрока : количество\_побед\_второго”

и длительность партии. Кроме этого два конструктора: конструктор по умолчанию, инициализирующий строки пустыми, а время нулем, и конструктор для заполнения определенными данными.

* **public class Statistic**

Класс предназначается для работы с файлом, который сохраняет в себе результаты последних 10 партий. Содержит массив объектов класса Result, поле size, в котором сохраняется количество объектов в массиве, строку с названием файла, а также следующие методы:

* + public void Clear() удаляет файл со статистикой.
  + public int getSize() возвращает количество объектов класса Result в массиве.
  + public void addElement(Result res) добавляет в начало массива объект res, предварительно смещая все имеющиеся объекты на одну позицию к концу массива, и увеличивает size на единицу. В случае, если количество объектов в массиве равно его длине, затирает объект, лежащий в последней ячейке.
  + public void writeStats() записывает имеющийся массив в файл, при этом для каждого объекта используя метод writeResult.
  + private void writeResult(RandomAccessFile file, int pos, Result t) открывает соответствующий файл, записывает в необходимую позицию информацию из объекта res и закрывает файл.
  + public Result[] readStats() возвращает массив объектов, которые считывает из файла. Для этого использует метод getOne.
  + private Result getOne(String dir, int pos) считывает из файла с названием dir информацию из необходимой позиции, заполняет объект класса Result и возвращает его.
* **public class ToPlay**

Класс призван управлять ходом игры, передавая эстафету от одного игрока к другому, проверяя, не стоит-ли завершить игру. Его методы:

* pubic toPlay(Player p1, Player p2, int sideN)для своих нужд просит через параметры о передаче ссылок на игроков, принимает также количество элементов, составляющих сторону поля. В зависимости от значения sideN конструктор создает класс, наследник класса Field (сейчас только Field3x3), и вызывает свой метод nextTurn()
* public static void nextTurn() - метод, вызывающий отключение поля, меняющий очередь хода (boolean turn), проверяющий состояние поля и завершающий игру методом endOfGame() либо продолжающий её методом toMove().
* public static void toMove() проверяет поле turn класса toPlay и в зависимости от ей значения вызывает toMove() метод у игрока\_1 (true) / игрока\_2 (false).
* public static void setElement(int i, int j), не является классическим setter’ом, он перенаправляет свои обязанности к одноименному методу класса Field, устанавливая тем самым в его матрице занятости поля значение turn по индексу [i][j].
* public static void endOfGame(boolean win) - метод завершения игры. Его работа происходит в отдельном потоке, что было необходимо в силу особенностей архитектуры GUI. Он проверяет win, в зависимости от которого меняется второе значение, передаваемое в Menu.showWin(), обозначая то, что кто-то победил или была ничья. В случае ничьей на место третьего параметра передается пустая строка, иначе - имя победившего игрока в зависимости от turn. Первый параметр здесь всегда true и обозначает offline режим. Все нужные данные об игре заносятся в статистику и вызывается вызов панели результатов “4” offrespanel.

**2.2 Диаграммы классов**

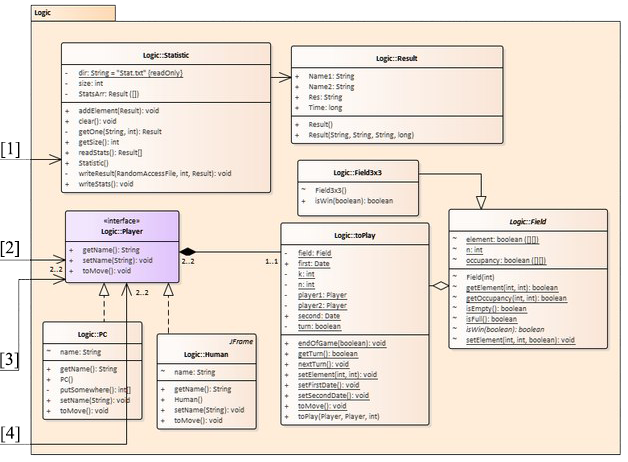


Рис. 2.2.1. Диаграмма пакета Logic

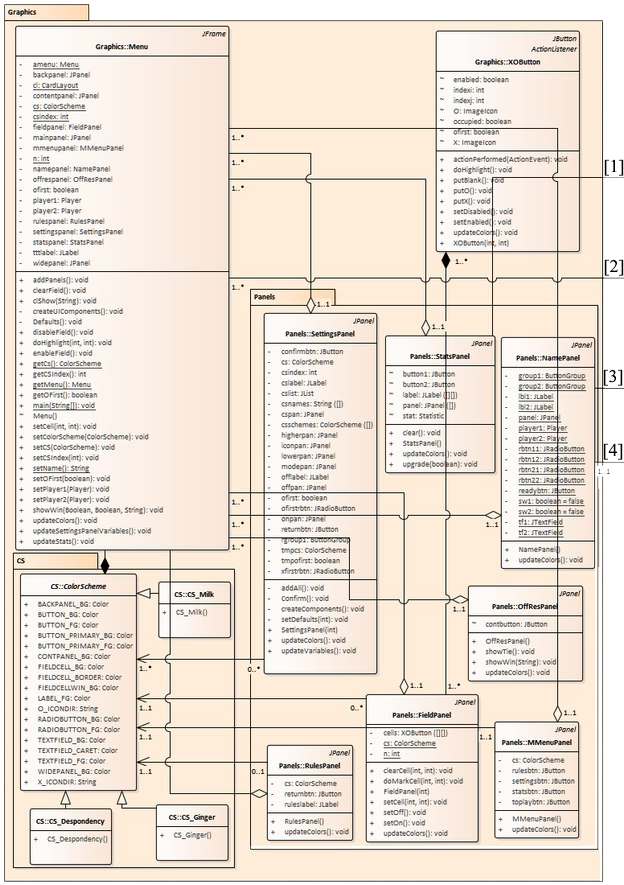


Рис. 2.2.2. Диаграмма пакета Graphic

**2.3. Используемые технологии**

Для написания игры использовались такие технологии:

* Язык программирования Java
* JDK 1.8 [1]
* Intellij IDEA 2016.2.5 [2]
* Git extensions [3]
* GitHub [4]

**3. ОПИСАНИЕ РЕЗУЛЬТАТОВ**

**3.1. Руководство пользователя**

При запуске игры пользователь видит следующее окно (рис 3.1.1).



Рис 3.1.1. Панель MMenuPanel

В нем он может нажать на одну из четырех представленных кнопок и перейти от панели главного меню к другим панелям. Нажатие на первую кнопку (кнопка Play) откроет панель NamePanel, в которой пользователю предложат ввести дополнительные данные, которые будут использоваться для взаимодействия с ним на протяжение игры, а также будут использованы для занесения в статистику (рис 3.1.2).

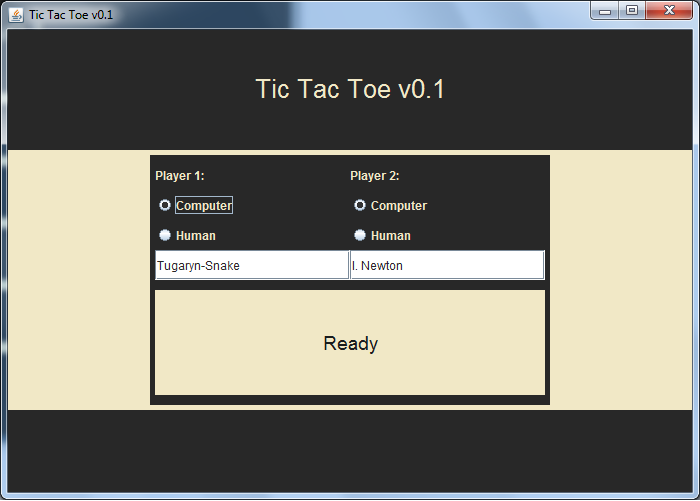


Рис 3.1.2. Панель NamePanel

Когда пользователь подтвердит свою готовность, данные будут считаны и сохранены, а перед пользователем откроется панель FieldPanel с игровым полем (рис 3.1.3).

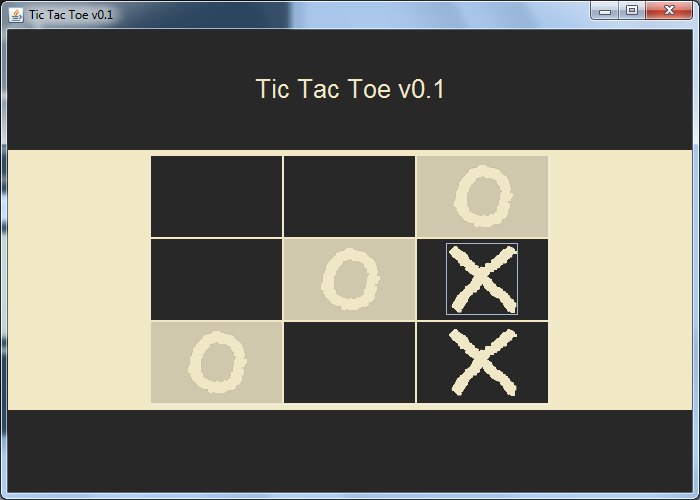


Рис 3.1.3. Панель FieldPanel

После клика на кнопку, на ней появляется соответствующий игроку значок. На рис. 3.1.3 мы можем видеть результат партии между двумя игроками класса PC. Выигрышная комбинация подсвечена более светлым цветом. Через 3 секунды после завершения партии открывается OffResPanel, показывающее результат партии (рис 3.1.4).



Рис 3.1.4. Панель OffResPanel

По нажатию на кнопку пользователь возвращается в главное меню (рис 3.1.1). Рассмотрим другие панели. По нажатию на кнопку “Statistic” в главном меню, пользователь увидит StatsPanel (рис 3.1.5), на которой отображаются результаты последних 10 партий.



Рис 3.1.5. Панель StatsPanel

Как мы можем видеть на текущем примере (рис. 3.1.5), партий было сыграно меньше 10, соответственно, на экран выводится текущее их количество. Также на этой панели пользователь может видеть две ранее описанных кнопки.

Далее рассмотрим панель настроек - SettingsPanel (рис. 3.1.6). Минимальное количество объектов на панели дает возможность понять практически любому пользователю принципы пользования ею: он видит в окне приложения перечень цветовых схем и группу JRadioButton, а также две кнопки: “Вернуться в меню” и “Применить”.

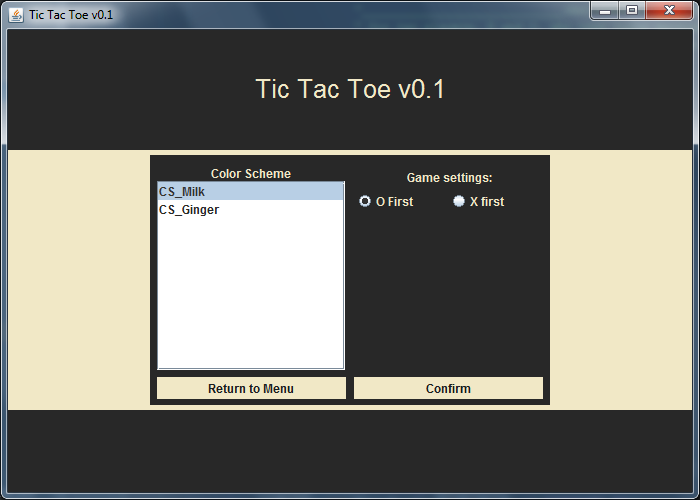


Рис 3.1.6. Панель SettingsPanel в цветовой схеме CS\_Milk

На следующем примере (рис 3.1.7) мы можем видеть, как применяются цветовые схемы, а именно CS\_Ginger.

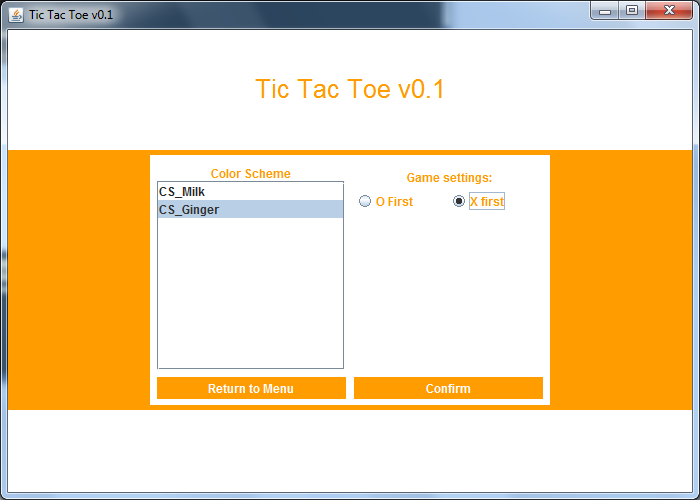


Рис 3.1.7. Панель SettingsPanel в цветовой схеме CS\_Ginger

Рассмотрим последнюю доступную пользователю панель - RulesPanel (рис 3.1.8). Открыть ее он может посредством нажатия на соответствующую кнопку из главного меню приложения (рис. 3.1.1). На ней мы видим все ранее описанное: два объекта JLabel и JButton. Примечательно, что правила было решено написать на двух языках для расширения целевой аудитории пользователей.



Рис 3.1.8. Панель RulesPanel

Таким образом мы наглядно рассмотрели программу глазами пользователя и попробовали “на ощупь” все ее возможности.

**4. РАБОТА КОМАНДЫ**

**4.1. Таблица распределения обязанностей**

|  |  |
| --- | --- |
| **Участник** | **Обязанности** |
| Струк Алена | 1. Создание базового алгоритма работы класса PC. 2. Написание логики поля. 3. Создание и улучшение всех аспектов статистики. 4. Написание панели выбора никнейма и типа игры. 5. Макетирование GUI, его создание и отладка. 6. Генерирование идей, участие в написании документации, унижение остальных участников. 7. Работа с гитом. |
| Ридозуб Олег | 1. Работа с архитектурой классов, рефакторинг 2. Отладка, дебаггинг, предотвращение ошибок 3. Создание системы выбора цветовых схем. 4. Написание панели настроек. 5. Написание GUI игрового поля. 6. Собирание проекта в кучу, участие в написании документации, информирование и организация. 7. Работа с гитом. |
| Балюк Константин | 1. Организация работы с документацией, налаживание бумажных дел проекта. 2. Написание дополнительных цветовых схем. 3. Участие и поддержка. 4. Ручное тестирование программного продукта. |

Табл. 4.1.1. Распределение обязанностей

**4.2. Не очень японская поэзия**

ஜ═════════ஜ۩۞۩ஜ═════════ஜ

Мрачные тучи застыли у пика Фудзиямы.  
Пронзающим холодом веет ветер с горных склонов.  
Покой и умиротворение найдёт самурай во сне,  
Пустив Push на Remote с кодом рабочим.

ஜ═════════ஜ۩۞۩ஜ═════════ஜ

Не видно за тучами пика священной горы.

Легкий ветер шатает стены храма и ломает сакуру.

Найти покой во сне самураю не светит:

Дедлайн через два дня

**ВЫВОДЫ**

Игра крестики-нолики во все времена была популярна и вряд ли наступит время, когда подобного рода приложения потеряют свою актуальность. Создав рабочее приложение, не уступающее своему бумажному аналогу, мы уверены, что после некоторых доработок оно сможет заинтересовать некоторую аудиторию, пускай даже лишь на пару партий. Архитектура, продуманная нами, позволит быстро и без серьезных усилий создавать дополнения, а частые обновления - залог успеха игры.

В процессе разработки приложения члены команды осознали, что проектирование является важной частью работы, занимает много времени, требует определенных навыков и трудозатрат, но затем значительно облегчает процесс написания программного кода. Отсутствие опыта в проектировании приложений привело к трудностям в разработке и отставанию по срокам, что стало хорошим уроком для нашей команды.

Создание приложения было пошаговым. Разбиение процесса на разработку отдельных возможностей позволило работать над несколькими функциями одновременно разным людям. Распределение ответственностей позволило избежать серьезных разногласий при разработке.

Главными проблемами были незнание многих возможностей пакета Swing и усложнение архитектуры приложения с добавлением новых частей программы. Разработка показала всю важность своевременного осмысления структуры программы.

Созданная нами программа удовлетворяет нашим планам на 35-40%, чего и следовало ожидать от первого совместного проекта. Это приложение имеет ряд недостатков в плане архитектуры и некоторые ошибки при работе. Несмотря на это, общий стиль кода мы старались выдержать в максимально приемлемом виде. Наша программа не держится на волоске от гибели и пригодна к расширению. Потратив небольшое количество времени на правку багов, её можно сделать вполне играбельной.

Трудности работы в команде подстерегали нас на всём пути разработки. Система контроля версий требовала привыкания к ней, а разные обстоятельства приводили к серьезному отставанию по графику. Однако, рабочая версия программы с реализацией львиной доли от запланированного показала, что командные усилия имели свои плоды. Наша команда может похвастаться активной командной работой и относительно слаженным процессом разработки.

Приложение имеет перспективы на развитие. В будущем планируется добавить возможность игры в режиме online с чатом между играющими пользователями, реализовать выбор количества игр в партии и уровни искусственного интеллекта (легкий, средний и сложный).

**ПЕРЕЧЕНЬ ССЫЛОК**

### [Крестики-нолики — Википедия](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B5%D1%81%D1%82%D0%B8%D0%BA%D0%B8-%D0%BD%D0%BE%D0%BB%D0%B8%D0%BA%D0%B8) - [Электронный ресурс]. - Режим доступа: [https://ru.wikipedia.org/wiki/%D0%9A%D1%80%](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%25)D0%B5%D1 %81%D1%82%D0%B8%D0%BA%D0%B8-%D0%BD% D0%BE%D0%BB%D0%B8%D0%BA%D0%B8

### [Java SE Runtime Environment 8 - Downloads - Oracle](http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html) - [Электронный ресурс]. - Режим доступа: <http://www.oracle.com/technetwork/ja va/javase/downloads/jre8-downloads-2133155.html> .

### [IntelliJ IDEA : Download Latest Version of IntelliJ IDEA - JetBrains](https://www.jetbrains.com/idea/download/) - [Электронный ресурс]. - Режим доступа: <https://www.jetbrains.сom /idea/download/#section=windows>

### [Git Extensions - Visual Studio Marketplace](https://marketplace.visualstudio.com/items?itemName=HenkWesthuis.GitExtensions) - [Электронный ресурс]. - Режим доступа: <https://marketplace.visualstudio.com/items?itemName =HenkWesthuis.GitExtensions>

### [How people build software · GitHub](https://github.com/) - [Электронный ресурс]. - Режим доступа: https://github.com

### 