**Explanatory note**

**1. Topic of the project / program name**

Connect-4

**2. Annotation - a short description of the program and its functionality**

 Игра под названием Connect-4, создана для двоих игроков, которым присваивается определенный цвет фишек, а затем они ходят по очереди, роняя фишки в ячейки вертикальной доски. Цель игры — расположить раньше противника подряд по горизонтали, вертикали или диагонали четыре фишки своего цвета.

**3. Central repository address**

https://github.com/andreyalisa/connect-4

**4. Members of the team and their roles**

Михайловская Алиса 146: UI   
Калиниченко Андрей 143: game logic   
Калюжный Даниил 142: server-client part

**5. List of classes with short description for each class**

Class Graphics.GameObject – Класс для создания графических объектов в игре и для контроля их движения

|  |  |
| --- | --- |
| public Texture2D Sprite | Рисунок объекта |
| public Vector2 Position | Левый верхний край объекта |
| public Vector2 Center (readonly) | Середина объекта |
| public Vector2 Velocity | Скорость объекта |
| public bool IsVisible | Видимость объекта |
| public Rectangle GetBoundingBox() | Возвращает границы объекта |
| public GameObject(Texture2D loadedTexture) | Конструктор: где loadedTexture - Sprite |

Class Graphics.Chip : GameObject – Класс для фишки игры, наследуется от класса GameObject

public enum ChipTeam { Blue, Red }

|  |  |
| --- | --- |
| public ChipTeam Team | Команда фишки |
| public Chip(Texture2D loadedTexture, ChipTeam team) : base(loadedTexture) | Конструктор |

Interface Graphics.GpItem – Интерфейс для управляемых объектов игры (button, textbox и т.е.)

|  |  |
| --- | --- |
| void Update(MouseState lastMstate, MouseState currentMstate, KeyboardState lastKstate, KeyboardState currentKstate); | Метод для обновления состояния объекта, должен вызываться на каждом ходу игры |
| void Draw(SpriteBatch spriteBatch); | Метод для рисования объекта |

Class Graphics.GpButton : GpItem – Класс для создания Button

|  |  |
| --- | --- |
| public Vector2 Size | Размер объекта |
| public bool IsClicked | Нажата ли кнопка |
| public Action onClick | Событие, которое вызывается при нажатии на кнопку |
| public void SetPosition(Vector2 newPosition) | Задает позицию объекта |
| public GpButton(Texture2D texture, GraphicsDevice gd) | Конструктор, где texture – фон кнопки; gd – GraphicsDevice игры |

Class Graphics.GpLabel : GpItem – Класс для создания Label

|  |  |
| --- | --- |
| public Vector2 Position | Левый верхний край объекта |
| public string Content | Содержание объекта |
| public SpriteFont Font | Фонт текста объекта |
| public bool IsVisible | Виден ли объекта |
| public Color TextColor | Цвет текста |
| public GpLabel(Vector2 position, SpriteFont font, string content = "") | Конструктор где position – позиция; font – фонт текста; content – содержание объекта |

Class Graphics.GpTextBox: GpItem – Класс для создания TextBox с цифрами и точкой

|  |  |
| --- | --- |
| public SpriteFont Font | Фонт текста объекта |
| public string GetContent() | Возвращает текст объекта |
| public int MaxLength | Максимальная длина строки |
| public bool IsFocused() | Возвращает true, если объект фокусирован |
| public GpTextBox(Texture2D texture, Vector2 position, SpriteFont font) | Конструктор, где texture – фон; position – позиция; font – фонт текста |

Class Network.User – класс для пользователя игры

|  |  |
| --- | --- |
| public string IP | IP пользователя |
| public int Port | Port пользователя |
| public User(string ip, int port) | Конструктор |

Class Network.Server – класс для работы с сетью (одиночка)

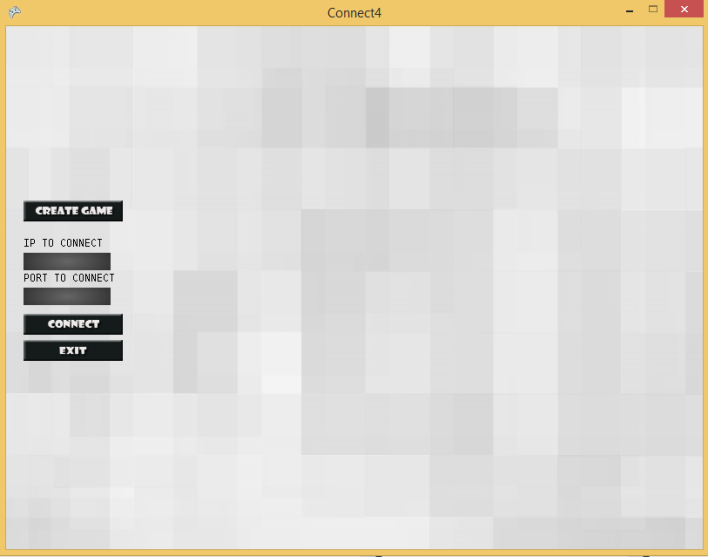
|  |  |
| --- | --- |
| public static Server getInstance() | Получает единственный объект класса |
| public void Abort() | Заканчивает все прослушки и потоки, обнуляет объект класса |
| public bool IsServerCreated() | Возвращает true, если прослушка началась и сервер создан |
| public bool IsUserConnected() | Возвращает true, если подключился пользователь, или мы подключились |
| public User getServerUser() | Возвращает текущего пользователя |
| public void Connect(string ip, string port) | Подключается к серверу по порту и ip |
| public void SendMove(int team, int column) | Отправляет сообщение о том, что сделан ход |
| public Action onUserConnected; | Событие, которое вызывается, когда подключается пользователь |
| public Action<int, int> onMakeMove; | Событие, которое вызывается, когда приходит сообщение о ходе |
| public Action onCantConnect; | Событие, которое вызывается, когда происходит ошибка при подключении к серверу |
| public Action onConnected; | Событие, которое вызывается, когда подключаемся к серверу |

Class GameLogic – класс для управления логикой игры (одиночка)

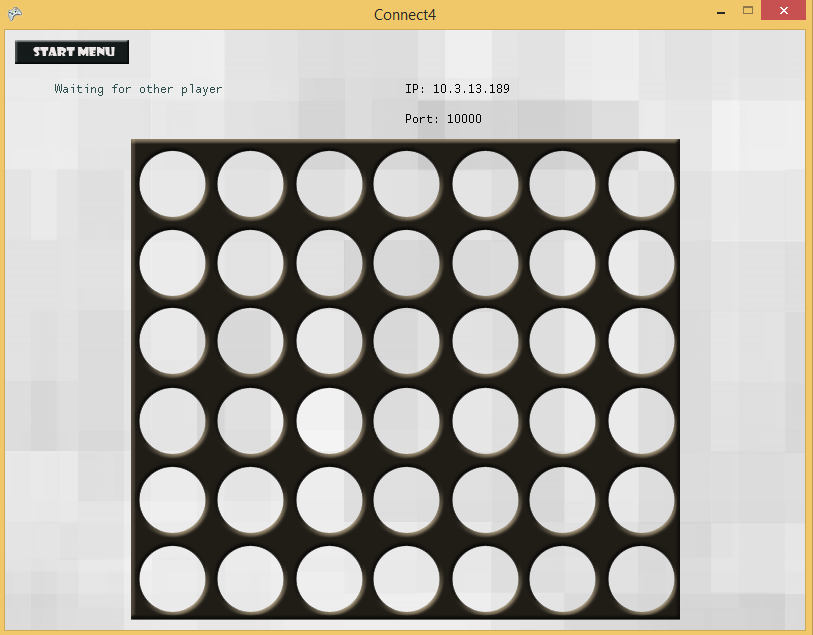
|  |  |
| --- | --- |
| public static GameLogic getInstance() | Получает единственный объект класса |
| public void AddonWonObserver(Action<int> observer) | Добавляет метод, который будет вызван при победе одного из игроков |
| public void AddonDrawObserver(Action observer) | Добавляет метод, который будет вызван при ничье |
| public void RemoveonWonObserver(Action<int> observer) | Удаляет метод, который будет вызван при победе одного из игроков |
| public void RemoveonDrawObserver(Action observer) | Удаляет метод, который будет вызван при ничье |
| public void AddonMoveObserver(Action<int, int> observer) | Добавляет метод, который будет вызван если сделан удачный ход |
| public void RemoveonMoveObserver(Action<int, int> observer) | Удаляет метод, который будет вызван если сделан удачный ход |
| public void MakeMove(int team, int column) | Сделать ход по команде и столбцу |
| public bool IsWon(int color) | Возвращает true, если color победил |
| public bool CanMove(int column) | Возвращает true, если можно сделать ход в column |
| public bool IsFull() | Возвращает true, если поле полон |
| public void Reset() | Удаляет все наблюдатели и обнуляет поле |
| public void ResetBattleFields() | Обнуляет поле |
| public void IsDraw() | Возвращает true, если ничья |

**6. Program interface**

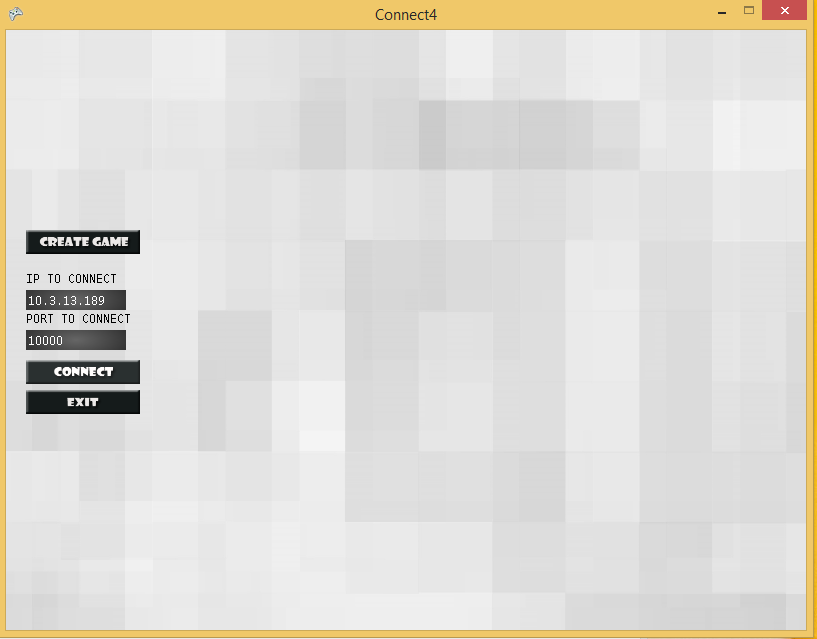
При запуске игры появляется окно, в левом углу посередине которого располагается меню игры.



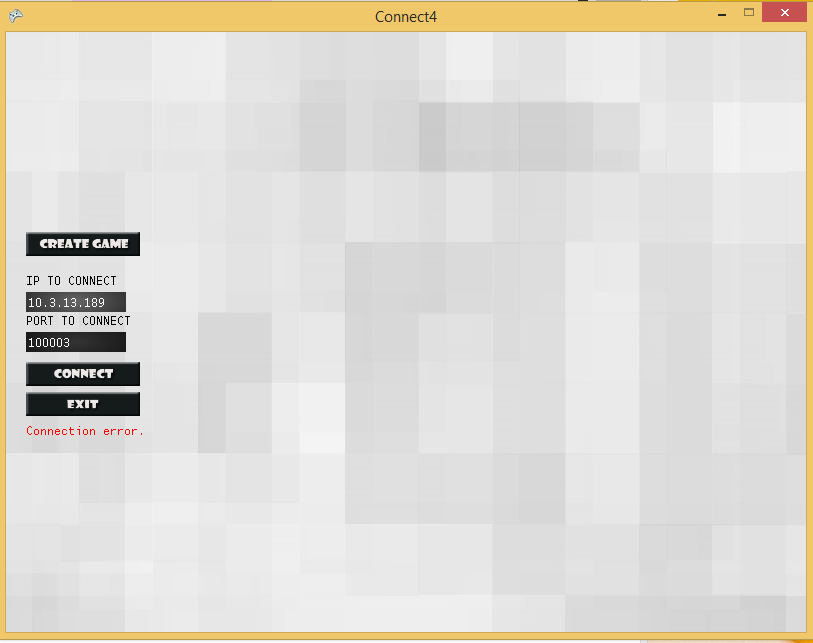
При нажатии на кнопку «Create game» перед игроком появляется доска с полями, а также IP и Port,чтобы второй игрок мог присоединится к игре. Слева имеется сообщение «Waiting for other player». Первый игрок не сможет начать ходить, пока второй игрок не подключиться.



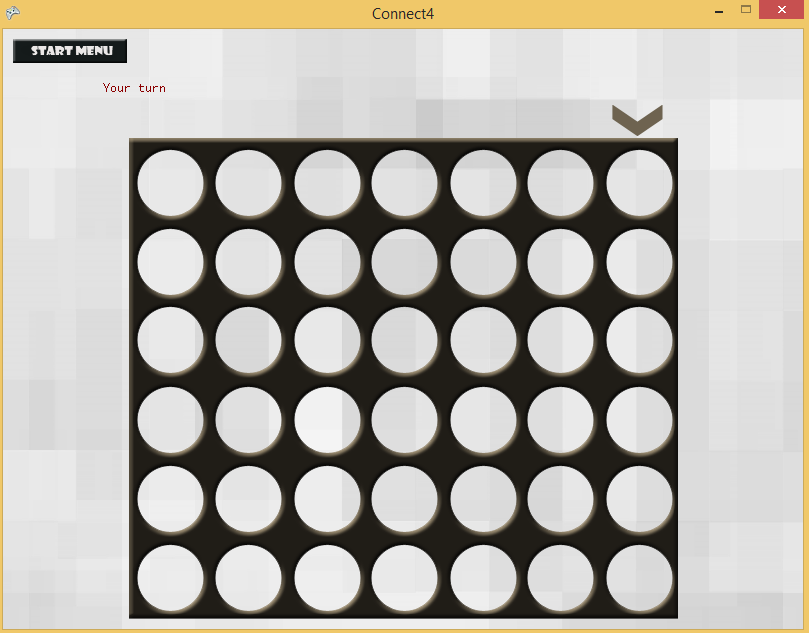
Для того чтобы второй игрок мог присоединиться к игре он должен корректно вести IP и Port первого игрока, после этого он должен нажать на кнопку «Connect».



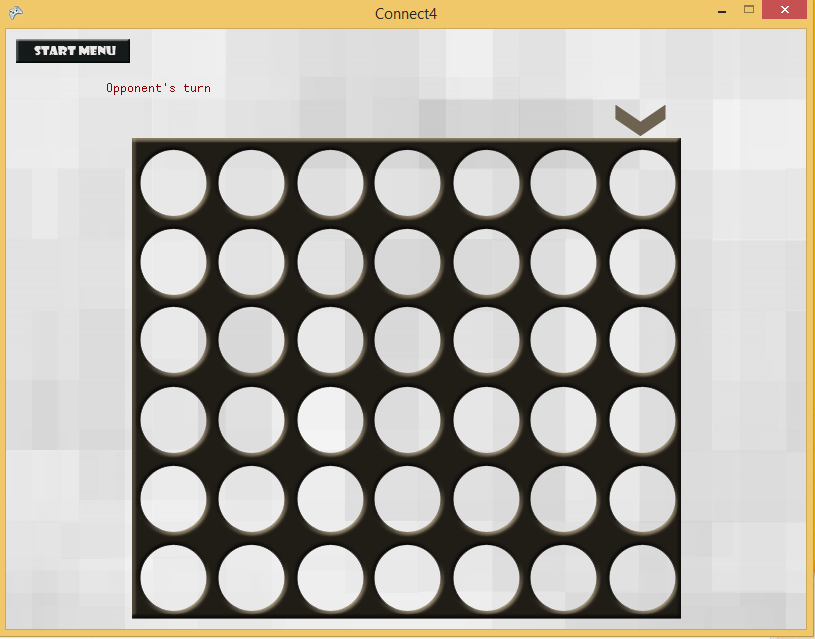
Если второй игрок неправильно ввел IP и/или Port, то при нажатии кнопки «Connect» внизу появляется сообщение «Connection error».



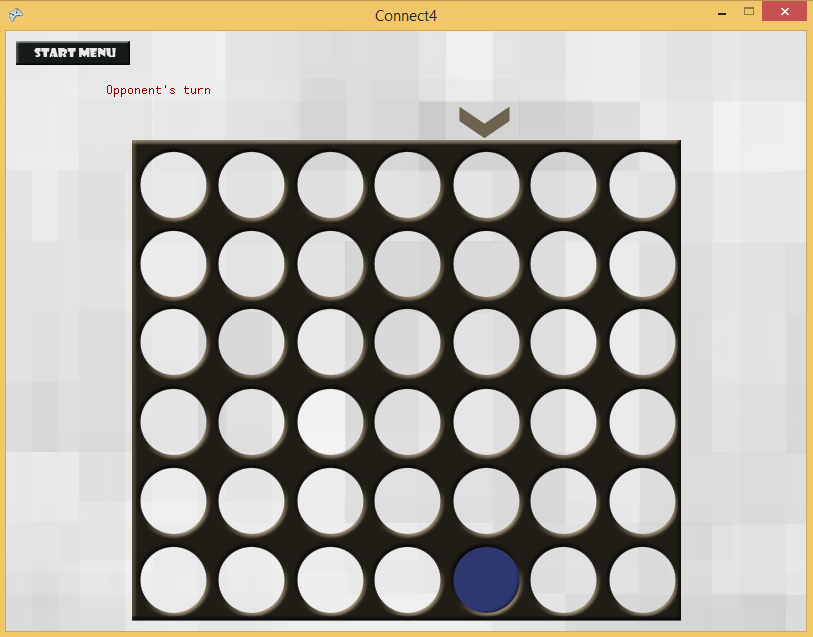
В случае, когда игрок второй игрок ввел правильно IP и Port и подключился к игре, то у первого игрока появляется сообщение «Your turn» и стрелочка над доской для выбора вертикальной ячейки.



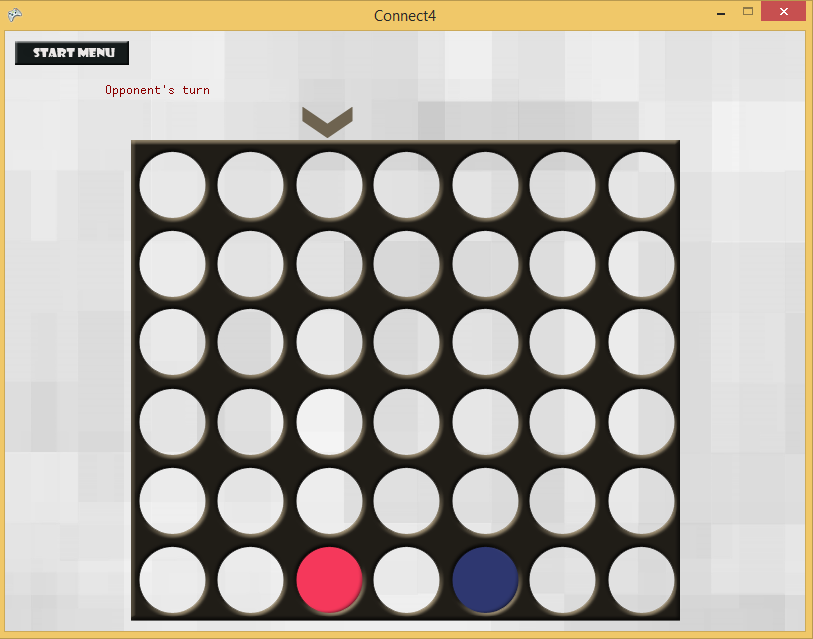
В тоже время у второго игрока появляется сообщение «Opponent’s turn» и стрелочка, но при этом он ходить не может.

****

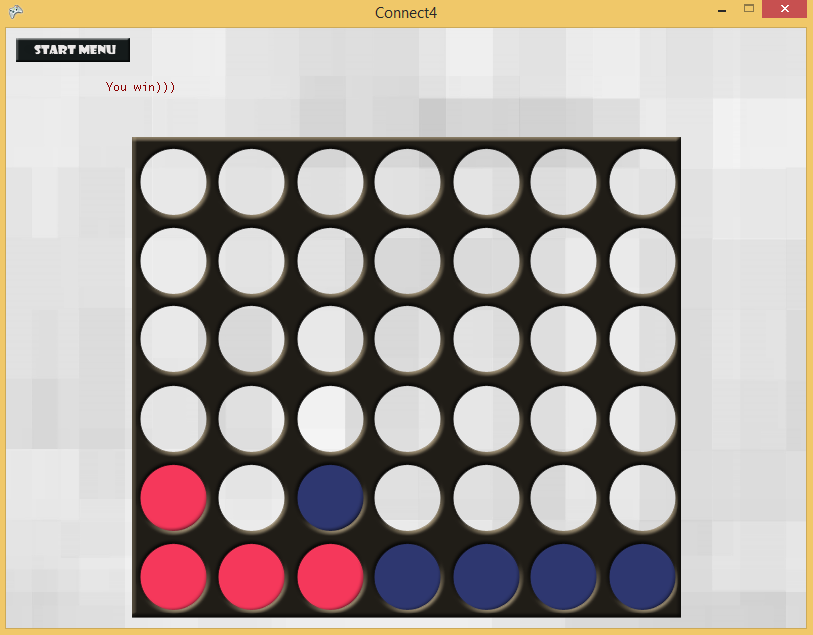
Когда первый игрок походил,ему автоматически была присвоена фишка с синим цветом и у него теперь появляется сообщение «Opponent’s turn» и соответственно он ходить не сможет, пока не сходит второй игрок.

****

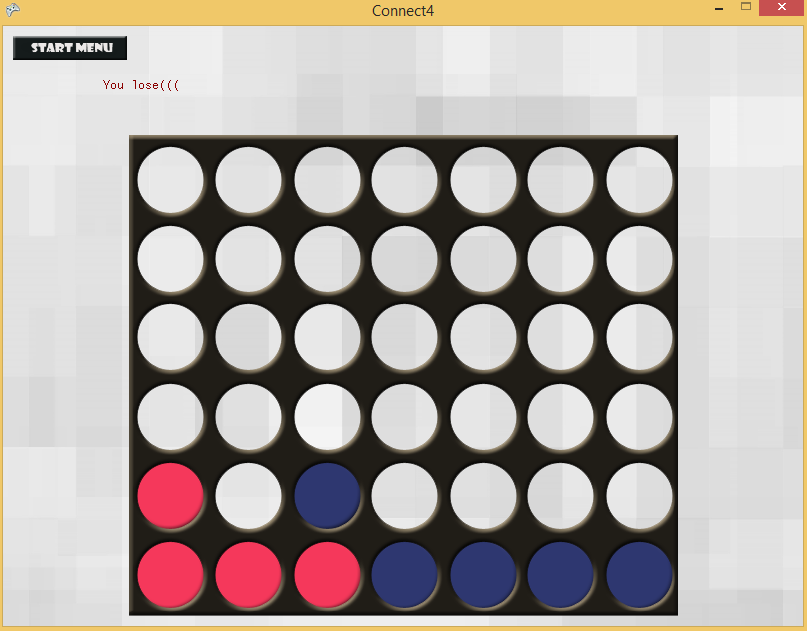
Второму игроку присваивается другой цвет фишки.

****

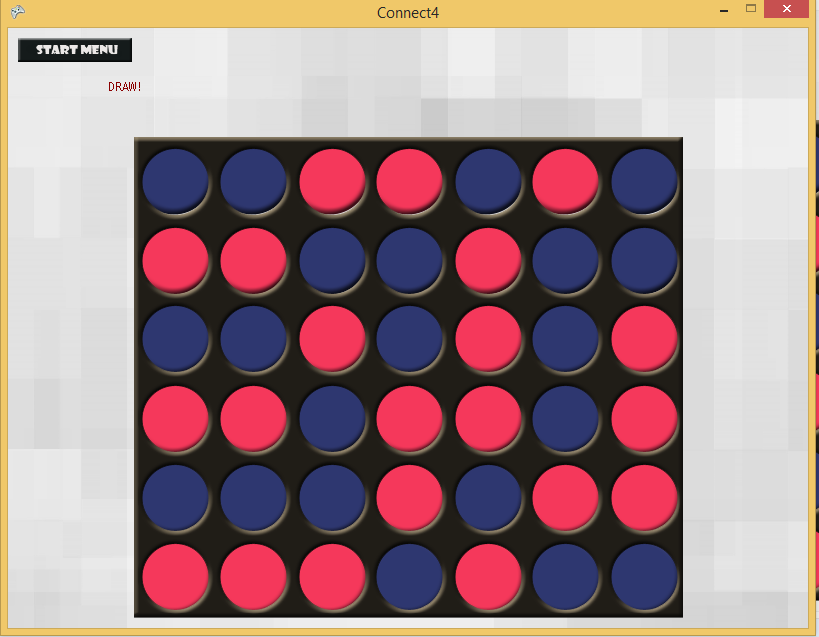
Было сделано несколько ходов, у первого игрока с синими фишками горизонтально подряд выстроилось 4 фишки его цвета, в данном случае первый игрок победил и у него появляется сообщение «You win)))».



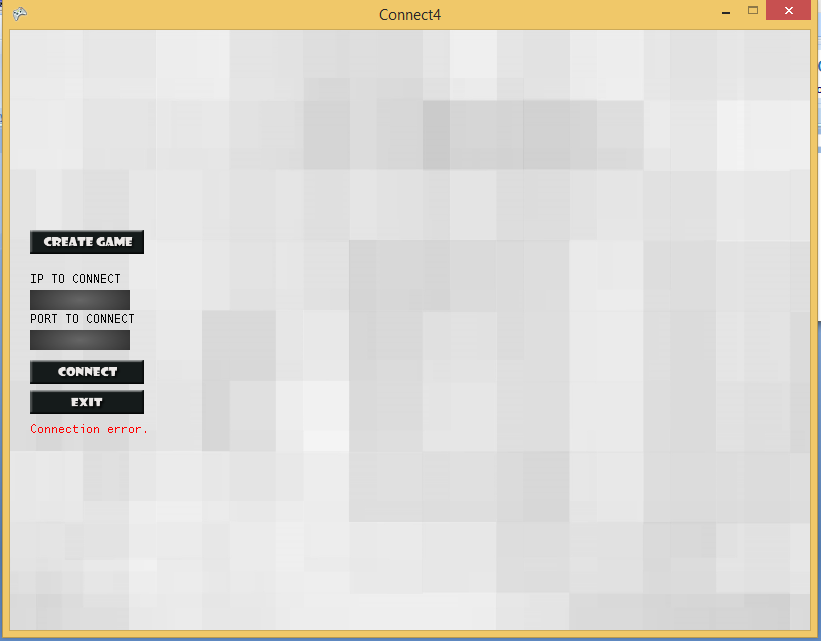
У второго игрока появляется сообщение «You lose(((»



В случае, когда ничья у обоих игроков в окне показывается сообщение «Draw!».



Если кто-то из игроков вышел из игры(по своему желанию нажав на «Start menu» или не по своему желанию), то у оппонента при нажатии мышки будет показываться окно с меню и сообщение «Connection error» .



При нажатии кнопки «Exit» в меню или клавиши Escape игра закрывается.