МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики  
Кафедра информационных технологий

Отчет по индивидуальному заданию №3

**«Создание клиент-серверного приложения “Морской бой”»**

Работу выполнил  
студент 42 группы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Б.М.Ибрагимов

Направление подготовки 01.03.02 Прикладная математика и информатика

Проверил  
канд. техн. наук, доц.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А.Полупанов

Краснодар 2020

**Цель работы:** разработать клиент-серверную игру «Морской бой» на C#.

**Задача 1.** Разработать клиент-серверную игру «Морской бой», с интерфейсом «человек-человек». Корабли расставляются по игровому полю с помощью рандома для каждого из игроков. До начала игры каждый из игроков имеет возможность перестроить свое игровое поле с помощью соответствующей кнопки.

**ЛИСТИНГ ПРОГРАММЫ**

**СЕРВЕР**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Net;

using System.Net.Sockets;

namespace BattleShip\_Server {

public partial class Form1 : Form {

Socket sListener, handler;

IPEndPoint ipEndPoint;

static string data = null, name = null;

public int sizePole = 10;

static Button[,] YourBoard, OpponentBoard;

List<Point> AcceptPoints = new List<Point>();

Regex regex = new Regex(@"\d \d");

int isWiner = 0;

public Form1() {

InitializeComponent();

}

private void button3\_Click(object sender, EventArgs e) {

}

private void button4\_Click(object sender, EventArgs e) {

int heightButtons = groupBoxYourBoard.Height / sizePole - 3;

Size buttonSize = new Size(groupBoxYourBoard.Width / sizePole - 3, heightButtons);

Font buttonsFont = new Font("Arial", heightButtons / 2);

YourBoard = new Button[sizePole, sizePole];

OpponentBoard = new Button[sizePole, sizePole];

for (int i = 0; i < sizePole; i++) {

for (int j = 0; j < sizePole; j++) {

Button button = new Button() {

Name = i + " " + j,

Font = buttonsFont,

Size = buttonSize,

Location = new Point(10 + j \* buttonSize.Width, 20 + i \* buttonSize.Height),

Enabled = false,

BackColor = Color.Blue,

Tag = 0

};

YourBoard[i, j] = button;

groupBoxYourBoard.Controls.Add(button);

AcceptPoints.Add(new Point(i, j));

Button button2 = new Button() {

Name = i + " " + j,

Font = buttonsFont,

Size = buttonSize,

Location = new Point(10 + j \* buttonSize.Width, 20 + i \* buttonSize.Height),

Enabled = false,

BackColor = Color.Blue,

Tag = 0

};

button2.Click += ButtonPoleClick;

OpponentBoard[i, j] = button2;

groupBoxOpponentBoard.Controls.Add(button2);

}

}

YourBoard = ArrangeShips(YourBoard, true);

reBuildYourBoard.Enabled = true;

listeningButton.Text = "Слушается...";

listeningButton.Enabled = false;

backgroundWorker1.RunWorkerAsync();

}

private void ButtonPoleClick(object sender, EventArgs e) {

name = sender.GetType().GetProperty("Name").GetValue(sender).ToString();

byte[] msg = Encoding.UTF8.GetBytes(name);

int bytesSent = handler.Send(msg);

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

if (OpponentBoard[i, j].BackColor == Color.Blue) OpponentBoard[i, j].Enabled = false;

}

private void StartServer(object sender, EventArgs e) {

IPHostEntry ipHost = Dns.GetHostEntry("localhost");

IPAddress ipAddr = ipHost.AddressList[0];

ipEndPoint = new IPEndPoint(ipAddr, 11000);

sListener = new Socket(ipAddr.AddressFamily, SocketType.Stream, ProtocolType.Tcp);

startButton.Enabled = false;

listeningButton.Enabled = true;

stopButton.Enabled = true;

}

private void button2\_Click(object sender, EventArgs e) {

sListener.Close();

sListener.Dispose();

listeningButton.Enabled = false;

stopButton.Enabled = false;

startButton.Enabled = true;

}

private void groupBoxYourBoard\_Enter(object sender, EventArgs e) {

}

private void reBuildYourBoard\_Click(object sender, EventArgs e) {

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

YourBoard[i, j].BackColor = Color.Blue;

YourBoard = ArrangeShips(YourBoard, true);

}

private void backgroundWorker1\_DoWork(object sender, DoWorkEventArgs e) {

try {

sListener.Bind(ipEndPoint);

sListener.Listen(1);

handler = sListener.Accept();

MessageBox.Show("Соединение установлено! Первый ход противника");

reBuildYourBoard.Invoke(new Action(() => reBuildYourBoard.Enabled = false));

while (true) {

data = null;

byte[] bytes = new byte[1024];

int bytesRec = handler.Receive(bytes);

data += Encoding.UTF8.GetString(bytes, 0, bytesRec);

if (data == "Вы проиграли") {

MessageBox.Show(data);

data = "<>";

}

if (data == "true" || data == "false") {

string[] splitName = name.Split(' ');

int[] indexes = new int[] { int.Parse(splitName[0]), int.Parse(splitName[1]) };

if (data == "true") {

OpponentBoard[indexes[0], indexes[1]].Invoke(new Action(() => OpponentBoard[indexes[0], indexes[1]].BackColor = Color.Red));

isWiner++;

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

if (OpponentBoard[i, j].BackColor == Color.Blue) OpponentBoard[i, j].Invoke(new Action(() => OpponentBoard[i, j].Enabled = true));

if (isWiner == 20) {

MessageBox.Show("Вы победили");

byte[] msg = Encoding.UTF8.GetBytes("Вы проиграли");

handler.Send(msg);

break;

}

}

else {

OpponentBoard[indexes[0], indexes[1]].Invoke(new Action(() => OpponentBoard[indexes[0], indexes[1]].BackColor = Color.White));

}

}

if (regex.IsMatch(data)) {

string[] splitData = data.Split(' ');

int[] indexes = new int[] { int.Parse(splitData[0]), int.Parse(splitData[1]) };

string answer = null;

if (Convert.ToInt32(YourBoard[indexes[0], indexes[1]].Tag) == 1) {

YourBoard[indexes[0], indexes[1]].Invoke(new Action(() => YourBoard[indexes[0], indexes[1]].Tag = 0));

YourBoard[indexes[0], indexes[1]].Invoke(new Action(() => YourBoard[indexes[0], indexes[1]].BackColor = Color.Red));

answer = "true";

}

else {

YourBoard[indexes[0], indexes[1]].Invoke(new Action(() => YourBoard[indexes[0], indexes[1]].BackColor = Color.White));

answer = "false";

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

if (OpponentBoard[i, j].BackColor == Color.Blue) OpponentBoard[i, j].Invoke(new Action(() => OpponentBoard[i, j].Enabled = true));

}

byte[] msg = Encoding.UTF8.GetBytes(answer);

handler.Send(msg);

}

if (data == "<>") {

handler.Shutdown(SocketShutdown.Both);

handler.Close();

listeningButton.Invoke(new Action(() => listeningButton.Text = "Слушать"));

listeningButton.Invoke(new Action(() => listeningButton.Enabled = true));

stopButton.Invoke(new Action(() => stopButton.Enabled = true));

groupBoxOpponentBoard.Invoke(new Action(() => groupBoxOpponentBoard.Controls.Clear()));

groupBoxYourBoard.Invoke(new Action(() => groupBoxYourBoard.Controls.Clear()));

MessageBox.Show("Соединение прервано");

break;

}

}

}

catch (Exception ex) {

MessageBox.Show(ex.ToString());

}

}

private Button[,] ArrangeShips(Button[,] board, bool myBoard) {

Button[,] boardCopy = board;

List<int> ships = new List<int>() { 1, 2, 3, 4, 1, 2, 3, 1, 2, 1 };

List<Point> orientations = new List<Point> { new Point(0, -1), new Point(-1, 0), new Point(0, 1), new Point(1, 0) };

Random random = new Random();

HashSet<Point> forbiddenPoints = new HashSet<Point>();

while (ships.Count != 0) {

Point orientation = orientations[random.Next(orientations.Count - 1)];

int ship = ships[random.Next(ships.Count - 1)];

int indentA = (ship - 1) \* Math.Abs(orientation.X);

int indentB = (ship - 1) \* Math.Abs(orientation.Y);

int x = random.Next(indentA, sizePole - indentA);

int y = random.Next(indentB, sizePole - indentB);

Point positionStart = new Point(x, y);

bool permission = true;

for (int i = 0; i < ship; i++)

if (forbiddenPoints.Contains(new Point(positionStart.X + i \* orientation.X, positionStart.Y + i \* orientation.Y))) {

permission = false;

break;

}

if (permission) {

for (int i = 0; i < ship; i++) {

Point p = new Point(positionStart.X + i \* orientation.X, positionStart.Y + i \* orientation.Y);

if (myBoard) boardCopy[p.X, p.Y].BackColor = Color.Yellow;

boardCopy[p.X, p.Y].Tag = 1;

forbiddenPoints.Add(p);

forbiddenPoints.Add(new Point(p.X, p.Y - 1));

forbiddenPoints.Add(new Point(p.X, p.Y + 1));

forbiddenPoints.Add(new Point(p.X - 1, p.Y));

forbiddenPoints.Add(new Point(p.X + 1, p.Y));

forbiddenPoints.Add(new Point(p.X - 1, p.Y - 1));

forbiddenPoints.Add(new Point(p.X + 1, p.Y + 1));

forbiddenPoints.Add(new Point(p.X + 1, p.Y - 1));

forbiddenPoints.Add(new Point(p.X - 1, p.Y + 1));

}

ships.Remove(ship);

}

}

return boardCopy;

}

}

}

**КЛИЕНТ**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Net;

using System.Net.Sockets;

namespace BattleShip\_Client {

public partial class Form1 : Form {

Socket socketSender;

public int sizePole = 10;

Button[,] YourBoard, OpponentBoard;

List<Point> AcceptPoints = new List<Point>();

static string data = null, name = null;

Regex regex = new Regex(@"\d \d");

static bool isCancel = false;

int isWiner = 0;

public Form1() {

InitializeComponent();

socketTextBox.Text = "localhost 11000";

}

private void button5\_Click(object sender, EventArgs e) {

stopConnectButton.Enabled = true;

isCancel = false;

string[] internalSocket = socketTextBox.Text.Split(' ');

IPHostEntry ipHost = Dns.GetHostEntry(internalSocket[0]);

IPAddress ipAddr = ipHost.AddressList[0];

IPEndPoint ipEndPoint = new IPEndPoint(ipAddr, int.Parse(internalSocket[1]));

socketSender = new Socket(ipAddr.AddressFamily, SocketType.Stream, ProtocolType.Tcp);

socketSender.Connect(ipEndPoint);

reBuildYourBoard.Enabled = false;

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

OpponentBoard[i, j].Enabled = true;

backgroundWorker1.RunWorkerAsync();

}

private void button2\_Click(object sender, EventArgs e) {

isCancel = true;

backgroundWorker1.Dispose();

stopConnectButton.Enabled = false;

connectButton.Enabled = true;

socketSender.Send(Encoding.UTF8.GetBytes("<>"));

socketSender.Shutdown(SocketShutdown.Both);

socketSender.Close();

groupBoxOpponentBoard.Controls.Clear();

groupBoxYourBoard.Controls.Clear();

}

private void reBuildYourBoard\_Click(object sender, EventArgs e) {

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

YourBoard[i, j].BackColor = Color.Blue;

YourBoard = ArrangeShips(YourBoard, true);

}

private void Form1\_Load(object sender, EventArgs e) {

int heightButtons = groupBoxYourBoard.Height / sizePole - 3;

Size buttonSize = new Size(groupBoxYourBoard.Width / sizePole - 3, heightButtons);

Font buttonsFont = new Font("Arial", heightButtons / 2);

YourBoard = new Button[sizePole, sizePole];

OpponentBoard = new Button[sizePole, sizePole];

for (int i = 0; i < sizePole; i++) {

for (int j = 0; j < sizePole; j++) {

Button button = new Button() {

Name = i + " " + j,

Font = buttonsFont,

Size = buttonSize,

Location = new Point(10 + j \* buttonSize.Width, 20 + i \* buttonSize.Height),

Enabled = false,

BackColor = Color.Blue,

Tag = 0

};

YourBoard[i, j] = button;

groupBoxYourBoard.Controls.Add(button);

AcceptPoints.Add(new Point(i, j));

Button button2 = new Button() {

Name = i + " " + j,

Font = buttonsFont,

Size = buttonSize,

Location = new Point(10 + j \* buttonSize.Width, 20 + i \* buttonSize.Height),

BackColor = Color.Blue,

Enabled = false,

Tag = 0

};

button2.Click += ButtonPoleClick;

OpponentBoard[i, j] = button2;

groupBoxOpponentBoard.Controls.Add(button2);

}

}

YourBoard = ArrangeShips(YourBoard, true);

}

private void ButtonPoleClick(object sender, EventArgs e) {

name = sender.GetType().GetProperty("Name").GetValue(sender).ToString();

byte[] msg = Encoding.UTF8.GetBytes(name);

int bytesSent = socketSender.Send(msg);

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

if (OpponentBoard[i, j].BackColor == Color.Blue) OpponentBoard[i, j].Enabled = false;

}

private void backgroundWorker1\_DoWork(object sender, DoWorkEventArgs e) {

try {

while (true) {

data = null;

byte[] bytes = new byte[1024];

int bytesRec = socketSender.Receive(bytes);

data += Encoding.UTF8.GetString(bytes, 0, bytesRec);

if (data == "Вы проиграли") {

MessageBox.Show(data);

data = "<>";

}

if (data == "true" || data == "false") {

string[] splitName = name.Split(' ');

int[] indexes = new int[] { int.Parse(splitName[0]), int.Parse(splitName[1]) };

if (data == "true") {

OpponentBoard[indexes[0], indexes[1]].Invoke(new Action(() => OpponentBoard[indexes[0], indexes[1]].BackColor = Color.Red));

isWiner++;

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

if (OpponentBoard[i, j].BackColor == Color.Blue) OpponentBoard[i, j].Invoke(new Action(() => OpponentBoard[i, j].Enabled = true));

if (isWiner == 20) {

MessageBox.Show("Вы победили");

byte[] msg = Encoding.UTF8.GetBytes("Вы проиграли");

socketSender.Send(msg);

break;

}

}

else {

OpponentBoard[indexes[0], indexes[1]].Invoke(new Action(() => OpponentBoard[indexes[0], indexes[1]].BackColor = Color.White));

}

}

if (regex.IsMatch(data)) {

string[] splitData = data.Split(' ');

int[] indexes = new int[] { int.Parse(splitData[0]), int.Parse(splitData[1]) };

string answer = null;

if (Convert.ToInt32(YourBoard[indexes[0], indexes[1]].Tag) == 1) {

YourBoard[indexes[0], indexes[1]].Invoke(new Action(() => YourBoard[indexes[0], indexes[1]].Tag = 0));

YourBoard[indexes[0], indexes[1]].Invoke(new Action(() => YourBoard[indexes[0], indexes[1]].BackColor = Color.Red));

answer = "true";

}

else {

YourBoard[indexes[0], indexes[1]].Invoke(new Action(() => YourBoard[indexes[0], indexes[1]].BackColor = Color.White));

answer = "false";

for (int i = 0; i < sizePole; i++)

for (int j = 0; j < sizePole; j++)

if (OpponentBoard[i, j].BackColor == Color.Blue) OpponentBoard[i, j].Invoke(new Action(() => OpponentBoard[i, j].Enabled = true));

}

byte[] msg = Encoding.UTF8.GetBytes(answer);

socketSender.Send(msg);

}

if (isCancel) break;

}

}

catch (Exception ex) {

MessageBox.Show(ex.ToString());

}

}

private Button[,] ArrangeShips(Button[,] board, bool myBoard) {

Button[,] boardCopy = board;

List<int> ships = new List<int>() { 1, 2, 3, 4, 1, 2, 3, 1, 2, 1 };

List<Point> orientations = new List<Point> { new Point(0, -1), new Point(-1, 0), new Point(0, 1), new Point(1, 0) };

Random random = new Random();

HashSet<Point> forbiddenPoints = new HashSet<Point>();

while (ships.Count != 0) {

Point orientation = orientations[random.Next(orientations.Count - 1)];

int ship = ships[random.Next(ships.Count - 1)];

int indentA = (ship - 1) \* Math.Abs(orientation.X);

int indentB = (ship - 1) \* Math.Abs(orientation.Y);

int x = random.Next(indentA, sizePole - indentA);

int y = random.Next(indentB, sizePole - indentB);

Point positionStart = new Point(x, y);

bool permission = true;

for (int i = 0; i < ship; i++)

if (forbiddenPoints.Contains(new Point(positionStart.X + i \* orientation.X, positionStart.Y + i \* orientation.Y))) {

permission = false;

break;

}

if (permission) {

for (int i = 0; i < ship; i++) {

Point p = new Point(positionStart.X + i \* orientation.X, positionStart.Y + i \* orientation.Y);

if (myBoard) boardCopy[p.X, p.Y].BackColor = Color.Yellow;

boardCopy[p.X, p.Y].Tag = 1;

forbiddenPoints.Add(p);

forbiddenPoints.Add(new Point(p.X, p.Y - 1));

forbiddenPoints.Add(new Point(p.X, p.Y + 1));

forbiddenPoints.Add(new Point(p.X - 1, p.Y));

forbiddenPoints.Add(new Point(p.X + 1, p.Y));

forbiddenPoints.Add(new Point(p.X - 1, p.Y - 1));

forbiddenPoints.Add(new Point(p.X + 1, p.Y + 1));

forbiddenPoints.Add(new Point(p.X + 1, p.Y - 1));

forbiddenPoints.Add(new Point(p.X - 1, p.Y + 1));

}

ships.Remove(ship);

}

}

return boardCopy;

}

}

}

**РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ**

До начала игры один из пользователей запускает серверную часть игры, поле чего он с помощью соответствующей кнопки запускает процесс принятия входных подключений. Затем этот пользователь настраивает свое игровое поле, рандомно переставляя кораблики (желтые клетки среди синих) на игровом поле. В это время, другой игрок запускает клиентскую часть игры, в соответствующее поле вписывает хост и порт для подключения к серверу и настраивает таким же образом свое игровое поле. Когда оба пользователя настроили свои игровые поля, у каждого игрока появляется их копия поля противника и начинается игра. Первым ход делает клиент, нажимая на любую доступную кнопку поля противника и отправляя противнику координаты нажатой кнопки. После этого он ждет ответа от противника (попал или нет). Если он попал по кораблю, то эта ячейка окрашивается красным цветом у обоих игроков в соответствующих полях и этот игрок делает ход еще раз. Иначе – эта ячейка окрашивается в белый цвет и ход переходит противнику. При победе одного из игроков, показываются соответствующие сообщения соответствующим игрокам.

На рисунке 1 изображен сервер до начала игры:

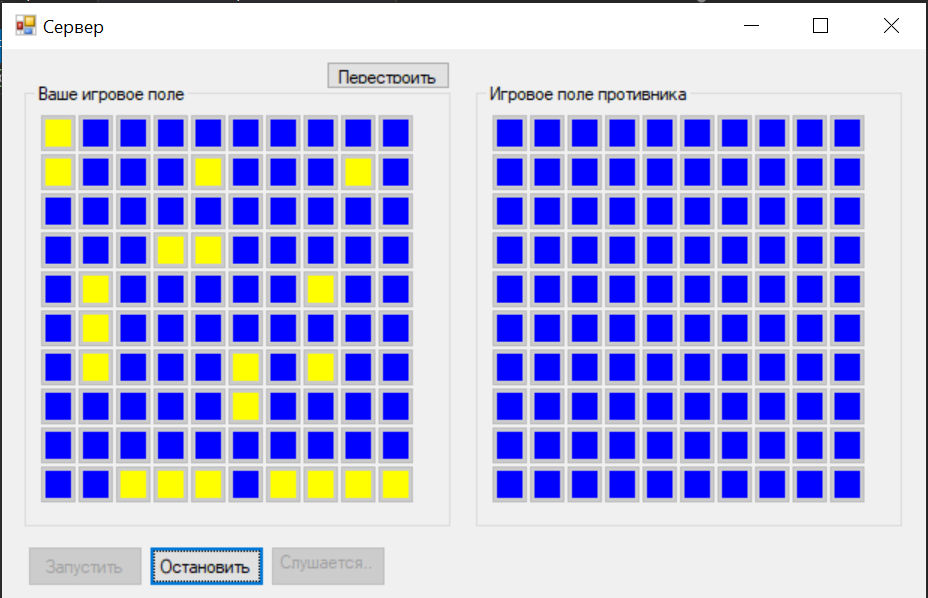


Рисунок 1 – игровое поле сервера до начала игры

На рисунке 2 изображен сервер до начала игры:

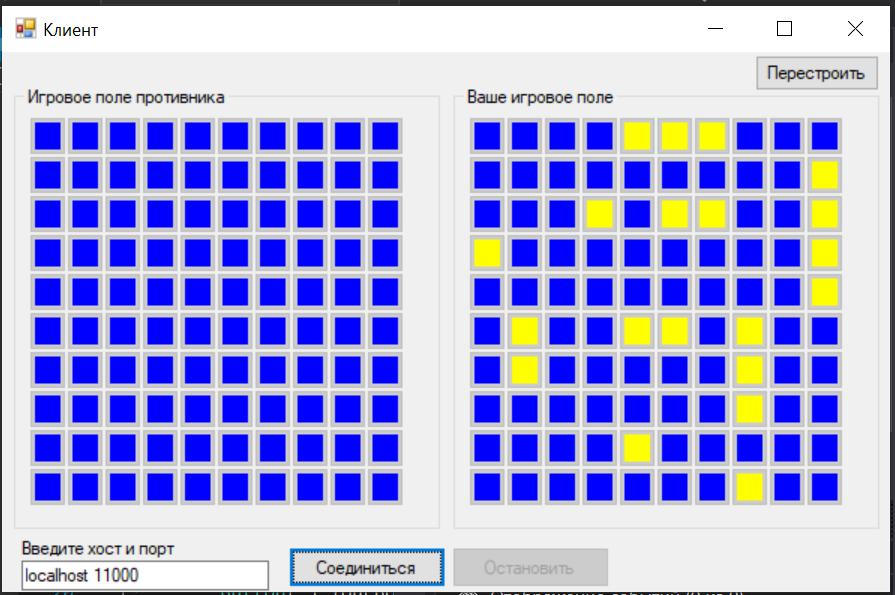
****

Рисунок 2 – игровое поле клиента до начала игры

На рисунке 3 показано состояние игрового поля сервера в процессе игры:

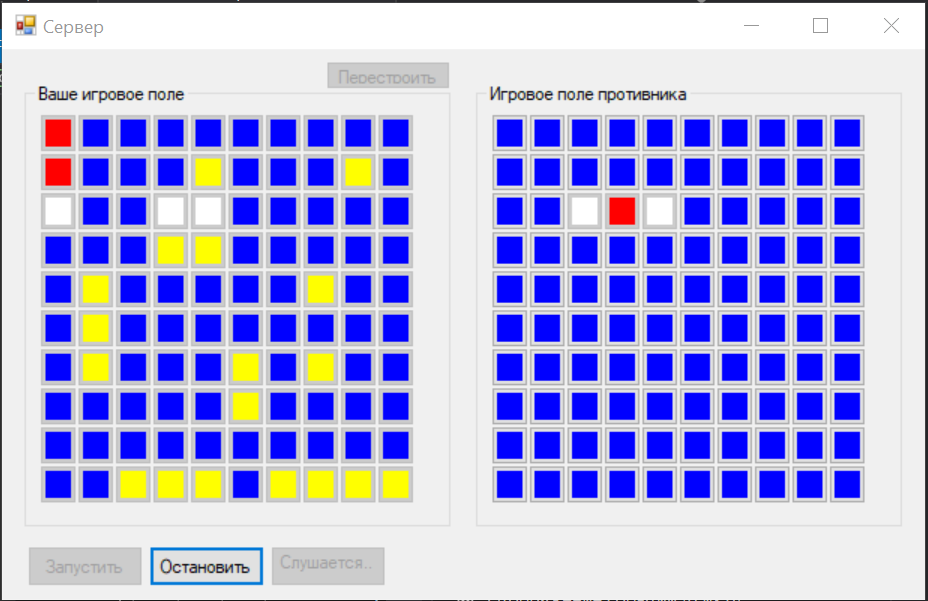


Рисунок 3 – игровое поле сервера в процессе игры

На рисунке 4 показано состояние игрового поля клиента в процессе игры:

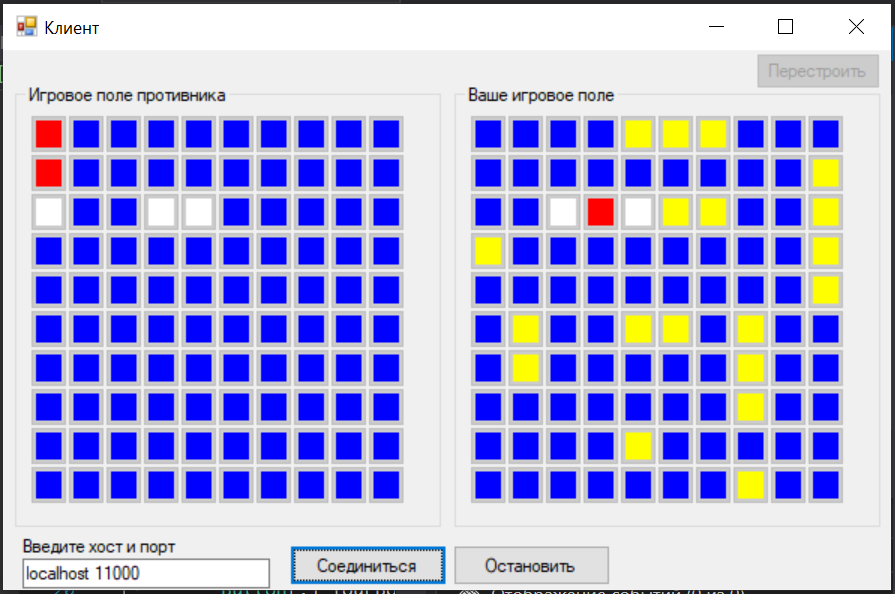


Рисунок 4 – игровое поле клиента в процессе игры

**Вывод:** была написана клиент-серверная игра «Морской бой» пользователя с пользователем, где корабли расставляются по игровому полю с помощью рандома. До начала игры каждый из пользователей имеет возможность перестроить свое игровое поле с помощью соответствующей кнопки. По окончанию игры выводятся сообщения о победе и проигрыше соответствующим игрокам.