МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний технічний університет України  
“Київський політехнічний інститут”

Лабораторна робота 5.

Обробка подій віддалених об'єктів засобами .Net Remoting.

Підготував:

ст. 4-го курсу ТЕФ гр. ТМ-21

Олійник Ю.Р.

Перевірив:

Стативка Ю.І.

Київ-2016

**Лабораторна робота 5.**

**Обробка подій віддалених об'єктів засобами .Net Remoting.**

**Мета роботи:**

1. Розробити віддалений клас з визначенням події.

2. Розробити серверний та клієнтський застосунки для організації віддаленої взаємодії.

3. Реалізувати вказану функціональність віддаленим класом.

**Варіант:6.**

Гра " Камінь-ножиці-папір ".

**Код програми:**

**Класс Game:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace RPSGame

{

public class Game : MarshalByRefObject

{

List<Player> players;

List<string> namesOfPlayers;

List<Player> winners;

bool gameStatus = false;

public event ShowPlayerHandler ShowPlayer;

public List<Player> Winners

{

get { return winners; }

}

public List<Player> Players

{

get { return players; }

}

public Player ConnectNewPlayer(Player p)

{

if (!gameStatus)

{

if (players == null)

{

players = new List<Player>();

namesOfPlayers = new List<string>();

}

if (!namesOfPlayers.Contains(p.Name))

{

p.Number = players.Count;

players.Add(p);

namesOfPlayers.Add(p.Name);

return p;

}

else

throw new ArgumentException("Player with this name is already in a game");

}

else

throw new EntryPointNotFoundException("Game is in progress. Wait few seconds.");

}

public void Disconnect(Player p)

{

try

{

players.Remove(p);

namesOfPlayers.Remove(p.Name);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

public List<Player> StartGame()

{

if (winners != null)

{

winners.Clear();

}

if (players.Count > 1)

{

// Waiting for players to make moves;

bool ready = true;

foreach (Player p in players)

{

if (p.Moved == false)

ready = false;

}

if (ready)

{

gameStatus = true;

// Getting players figures;

bool rock = false;

bool paper = false;

bool scissors = false;

foreach (Player p in players)

{

if (p.Figure == "rock")

rock = true;

if (p.Figure == "paper")

paper = true;

if (p.Figure == "scissors")

scissors = true;

}

// Check if there are any winner;

if (IsAnyWinner(rock, paper, scissors))

{

winners = GetWinners(rock, paper, scissors, players);

foreach (Player p in winners)

p.Win();

foreach (Player p in players)

OnShowPlayer(p);

}

foreach (Player p in players)

{

OnShowPlayer(p);

ClearPlayersFigure();

}

}

}

gameStatus = false;

return winners;

}

public List<Player> GetWinners(bool rock, bool paper, bool scissors, List<Player> participiants)

{

List<Player> winnersList = new List<Player>();

if (rock && paper)

{

foreach (Player p in players)

{

if (p.Figure == "paper")

winnersList.Add(p);

}

}

if (rock && scissors)

{

foreach (Player p in players)

{

if (p.Figure == "rock")

winnersList.Add(p);

}

}

if (scissors && paper)

{

foreach (Player p in players)

{

if (p.Figure == "scissors")

winnersList.Add(p);

}

}

return winnersList;

}

public bool IsAnyWinner(bool rock, bool paper, bool scissors)

{

if (rock && paper && scissors)

{

return false;

}

else

{

if ((rock && !paper && !scissors) || (!rock && !paper && scissors) || (!rock && paper && !scissors))

return false;

else

return true;

}

}

public void ClearPlayersFigure()

{

foreach (Player p in players)

{

p.Moved = false;

}

}

// згенерувати подію

public void OnShowPlayer(Player p)

{

if (ShowPlayer != null)

ShowPlayer(p);

}

//Оновити (перемалювати) форму усіх гравців

public void ShowAllPlayers()

{

foreach (Player p in players)

OnShowPlayer(p);

}

}

}

**Клас Player:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Drawing;

namespace RPSGame

{

public class Player : MarshalByRefObject

{

string name;

Color playerColor;

string figure;

int score;

bool moved;

int number;

Game game;

public Player(string \_name, Color color, Game \_game)

{

name = \_name;

playerColor = color;

score = 0;

game = \_game;

moved = false;

}

public string Name

{

get { return name; }

set

{

if (name.Length < 12)

name = value;

else

throw new ArgumentOutOfRangeException("Name is too long");

}

}

public int Number

{

get { return number; }

set { number = value; }

}

public bool Moved

{

get { return moved; }

set { moved = value; }

}

public string Figure

{

get { return figure; }

}

public Color Color

{

get { return playerColor; }

set { playerColor = value; }

}

public void MakeMove(string input)

{

if (input == "rock" || input == "paper" || input == "scissors")

{

figure = input;

moved = true;

game.StartGame();

}

else

throw new ArgumentOutOfRangeException("Wrong figure name");

}

public void ResetPlayer()

{

figure = "";

}

public void Win()

{

score += 10;

}

public void ResetScore()

{

score = 0;

}

}

}

**Клас форми:**

using System;

using System.Drawing;

using System.Windows.Forms;

using System.Runtime.Remoting.Channels;

using System.Runtime.Remoting.Channels.Tcp;

using System.Runtime.Remoting.Channels.Http;

using System.Collections;

using System.Runtime.Serialization.Formatters;

using System.Runtime.Remoting;

namespace RPSGame

{

public delegate void ShowPlayerHandler(Player player);

public partial class ClientForm : Form

{

Game game = null;

Player player;

int port;

public ClientForm()

{

InitializeComponent();

}

public void StartServer()

{

int sPort = 8900;

ChannelServices.RegisterChannel(CreateChannel(sPort), false);

game = new Game();

RemotingServices.Marshal(game, "GameObject");

Console.WriteLine("Server status is ... OK");

Console.WriteLine("Port = " + sPort);

Console.ReadLine();

}

private TcpChannel CreateChannel(int port)

{

BinaryServerFormatterSinkProvider

sp = new BinaryServerFormatterSinkProvider();

sp.TypeFilterLevel = TypeFilterLevel.Full; // Дозволити

//передачу делегатів

BinaryClientFormatterSinkProvider

cp = new BinaryClientFormatterSinkProvider();

IDictionary props = new Hashtable();

props["port"] = port;

props["typeFilterLevel"] = TypeFilterLevel.Full;

props["name"] = "Channel" + port.ToString(); // here enter unique channel name

return new TcpChannel(props, cp, sp);

}

public void ConnectToServer()

{

// Створити та зареєструвати канал для взаємодії з сервером

ChannelServices.RegisterChannel(CreateChannel(port), false);

// Отримати посилання на віддалений об'ект-гру

try

{

game = (Game)Activator.GetObject(typeof(Game), String.Format("tcp://localhost:8900/GameObject"));

game.ShowPlayer += new ShowPlayerHandler(PrintResult);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

public void PrintResult(Player player)

{

string gameResult = "";

using (Graphics g = CreateGraphics())

{

Font drawFont = new Font("Arial", 10);

SolidBrush drawBrush = new SolidBrush(Color.White);

if (game.Winners != null)

{

if (game.Winners.Count == 0)

{

gameResult += "Игра закончилась: \n";

gameResult += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \n";

gameResult += "Поздравляем всех, у вас Ничья! \n";

foreach (Player p in game.Players)

gameResult += "Игрок " + p.Name + " фигура - " + p.Figure + "\n";

gameResult += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \n";

g.Clear(Color.Black);

g.DrawString(gameResult, drawFont, drawBrush, 11, 50);

}

else

{

if (game.Winners.Count == game.Players.Count)

{

gameResult += "Игра закончилась: \n";

gameResult += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \n \n";

gameResult += "Поздравляем всех, у вас Ничья! \n";

foreach (Player p in game.Players)

gameResult += "Игрок " + p.Name + " фигура - " + p.Figure + "\n";

gameResult += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \n";

g.Clear(Color.Black);

g.DrawString(gameResult, drawFont, drawBrush, 11, 50);

}

else

{

gameResult += "Игра закончилась: \n";

gameResult += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \n";

if (game.Winners.Count == 1)

gameResult += "Победил: \n";

else

{

gameResult += "Победили: \n";

}

foreach (Player p in game.Winners)

gameResult += "Игрок " + p.Name + " фигура - " + p.Figure + "\n";

gameResult += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \n";

g.Clear(Color.Black);

g.DrawString(gameResult, drawFont, drawBrush, 11, 50);

}

}

}

else

{

gameResult += "Игра закончилась: \n";

gameResult += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \n";

gameResult += "Поздравляем всех, у вас Ничья! \n";

foreach (Player p in game.Players)

gameResult += "Игрок " + p.Name + " фигура - " + p.Figure + "\n";

gameResult += "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \n";

g.Clear(Color.Black);

g.DrawString(gameResult, drawFont, drawBrush, 11, 50);

}

}

}

public void RegPlayerOnServer()

{

player = game.ConnectNewPlayer(player);

}

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

{

//StartServer();

}

private void btEnterName\_Click(object sender, EventArgs e)

{

try

{

PlayerName form = new PlayerName();

form.Text = "Player Name";

form.ShowDialog();

player = new Player(form.PName, Color.Red, game);

RegPlayerOnServer();

this.Text = player.Name;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void btChoose\_Click(object sender, EventArgs e)

{

if (player != null)

{

if (!player.Moved)

{

try

{

ChooseFigureForm form = new ChooseFigureForm();

form.Text = "Player Figure";

form.ShowDialog();

player.MakeMove(form.PlayerChoose);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

else

MessageBox.Show("Вы уже походили");

}

else

MessageBox.Show("Сначала введите имя");

}

private void ввестиПортToolStripMenuItem\_Click(object sender, EventArgs e)

{

try

{

PortForm form = new PortForm();

form.Text = "Port";

form.ShowDialog();

port = Convert.ToInt32(form.Port);

ConnectToServer();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void запуститьСерверToolStripMenuItem1\_Click(object sender, EventArgs e)

{

StartServer();

}

}

}

-

**Результати роботи програми:**

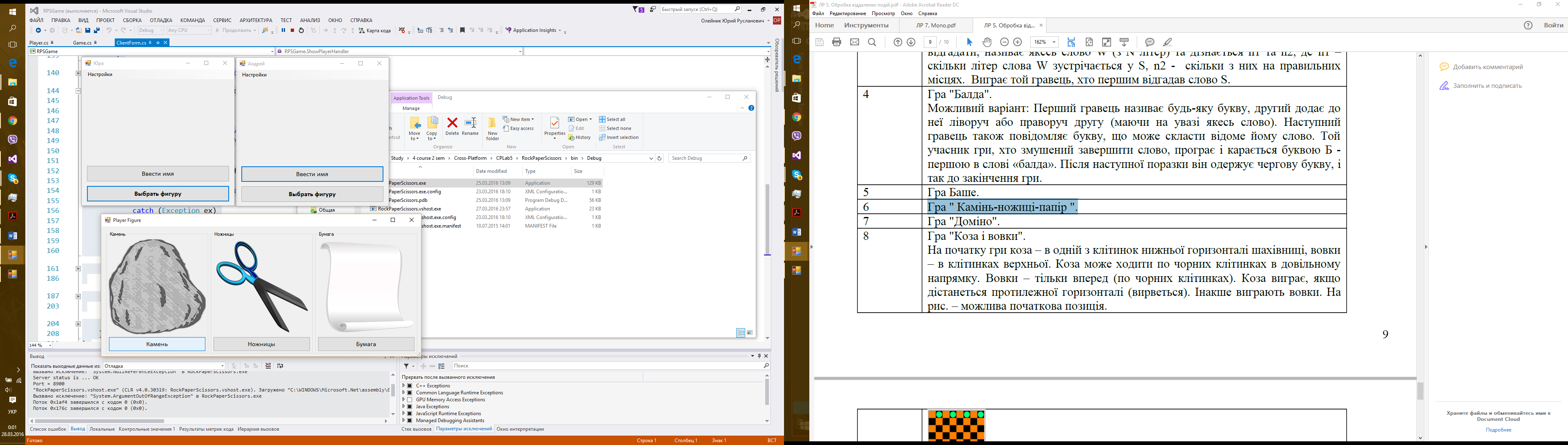


Рис 1. Вибір фігури

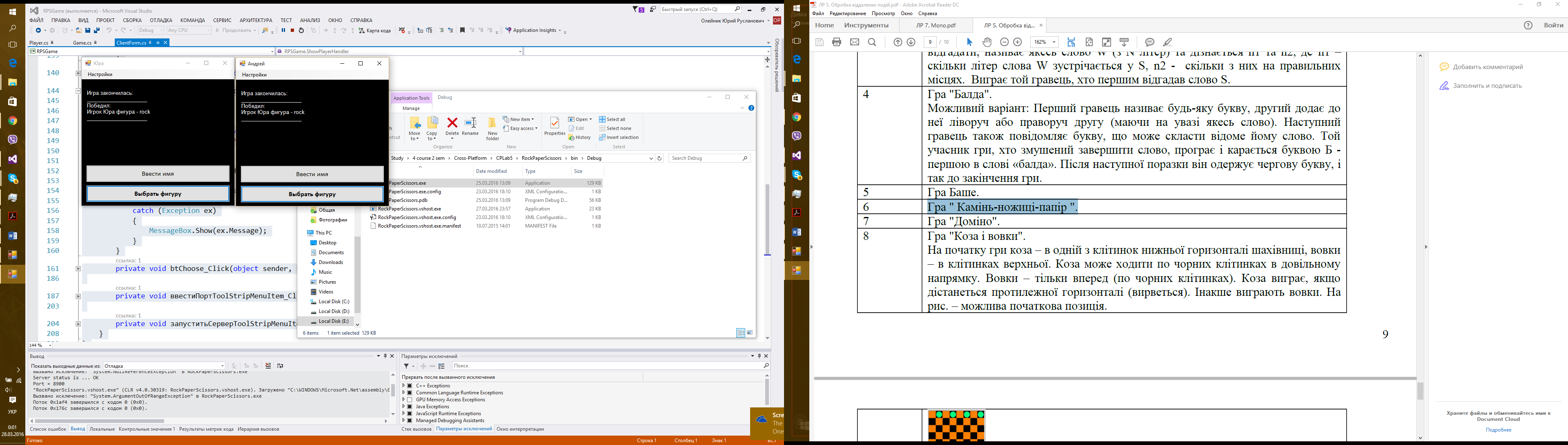


Рис2. Кінець Гри

**Висновки:** в процесі лабораторної роботи було досліджено:

1. елементи технології .Net Remoting;
2. розроблено алгоритм програми відповідно до варіанта;
3. написано програму