**5 ЛАБОРАТОРНАЯ РАБОТА №5**

**РАЗРАБОТКА СИСТЕМНЫХ ПРОГРАММ СРЕДСТВАМИ**

**ПЛАТФОРМЫ .NET FRAMEWORK И ЯЗЫКА C#.**

**Цель работы:** ознакомиться с технологией .NET, изучить основы языка C#, получить практические навыки вызова системных функций при помощи .NET C#.

**5.1 Текст программы заданной на самостоятельную работу**

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace laba5norm {

    public partial class Form1 : Form {

        public Form1(){

            InitializeComponent();

        }

        private void button1\_Click(object sender, EventArgs e){

            SyscallWraper.ShowWindow(this.Handle, 3);

        }

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

            StringBuilder s = new StringBuilder("SetWindowText");

            SyscallWraper.SetWindowText(this.Handle, s);

        }

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

            SyscallWraper.system\_time sys\_time = new SyscallWraper.system\_time();

            SyscallWraper.GetSystemTime(ref sys\_time);

            StringBuilder s = new StringBuilder(sys\_time.hour.ToString() + ':' +

                sys\_time.minute.ToString() + ":" +

                sys\_time.second.ToString());

            SyscallWraper.SetWindowText(this.Handle, s);

        }

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

            IntPtr hwnd = new IntPtr(0);

            StringBuilder txt = new StringBuilder("pokemon");

            StringBuilder capt = new StringBuilder("pikachu");

            SyscallWraper.showMsg(hwnd, txt, capt, 0);

        }

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

        {

            SyscallWraper.WINDOWINFO win\_info = new SyscallWraper.WINDOWINFO();

            StringBuilder fileName = new StringBuilder(170);

            SyscallWraper.GetWindowModuleFileName(this.Handle, fileName, 170);

            SyscallWraper.GetWindowInfo(this.Handle, ref win\_info);

            richTextBox1.AppendText(fileName.ToString());

            richTextBox1.AppendText("\n");

            richTextBox1.AppendText(win\_info.rcWindow.ToString());

        }

    }

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Runtime.InteropServices;

namespace laba5norm

{

    class SyscallWraper

    {

        [DllImport("user32.dll")]

        public extern static int ShowWindow(IntPtr hWnd, int cmd);

        [DllImport("user32.dll")]

        public extern static int SetWindowText(IntPtr hWnd, StringBuilder text);

        public struct system\_time

        {

            public ushort year;

            public ushort month;

            public ushort dayOfWeek;

            public ushort day;

            public ushort hour;

            public ushort minute;

            public ushort second;

            public ushort milliseconds;

        }

        [DllImport("kernel32.dll")]

        public extern static void GetSystemTime(ref system\_time st);

        [DllImport("user32.dll", EntryPoint = "MessageBoxW",

        CharSet = CharSet.Unicode,

        CallingConvention = CallingConvention.Winapi)]

        public extern static void showMsg(IntPtr hwnd,

            StringBuilder txt,

            StringBuilder capt,

            uint type);

        [StructLayout(LayoutKind.Sequential)]

        public struct RECT

        {

            public int Left, Top, Right, Bottom;

            public override string ToString()

            {

                return string.Format(System.Globalization.CultureInfo.CurrentCulture, "{{Left={0},Top={1},Right={2},Bottom={3}}}", Left, Top, Right, Bottom);

            }

        }

        [StructLayout(LayoutKind.Sequential)]

        public struct WINDOWINFO

        {

            public uint cbSize;

            public RECT rcWindow;

            public RECT rcClient;

            public uint dwStyle;

            public uint dwExStyle;

            public uint dwWindowStatus;

            public uint cxWindowBorders;

            public uint cyWindowBorders;

            public ushort atomWindowType;

            public ushort wCreatorVersion;

            public WINDOWINFO(Boolean? filler)

                : this()   // Allows automatic initialization of "cbSize" with "new WINDOWINFO(null/true/false)".

            {

                cbSize = (UInt32)(Marshal.SizeOf(typeof(WINDOWINFO)));

            }

        }

        [return: MarshalAs(UnmanagedType.Bool)]

        [DllImport("user32.dll", SetLastError = true)]

        public static extern bool GetWindowInfo(IntPtr hwnd, ref WINDOWINFO pwi);

        [DllImport("user32.dll")]

        public extern static int GetWindowModuleFileName(IntPtr hWnd, StringBuilder fileName, uint max\_len);

    }

}

**Выводы**

В ходе лабораторной работы были изучены основы языка C#. Также были изучены механизмы вызова системных функций WINAPI средствами языка С#. Было создано приложение которое вызывает системные функции GetWindowInfo и GetModuleFileName. Основной трудностью при вызове системных функций является приведение типов WINAPI к типам языка C#.