Лабораторная работа № 8
Создание Windows-приложений

Приступим к созданию win-приложений.

Шаг 1. Создаем новый проект – проект Windows Application. Не будем на этом подробно останавливаться, т.к. win-проект создается по аналогии с консольным.

Шаг 2. Перед нами появляется пустая поверхность win-окна, на которую можно добавлять различные элементы управления. Так же можно увидеть панели Toolbox (в ней то и находятся все элементы управления), Server Explorer (подключение к проекту БД), Solution Explorer (отображение подключенных проектов, файлов, пространств имен и т.п.), Properties (отображение свойств выбранного элемента и событий, связанных с ним), Dynamic Help и другие (в зависимости от настроек пользователя этих панелей можно не обнаружить!). Возможности, связанные с другими панелями, так же весьма интересны, но их рассмотрение не входит в цели данного методического пособия. Добавить дополнительные панели можно через меню «View».

Шаг 3. Рассмотрим более подробно win-поверхность и ее свойства. Легко заметить, что мы можем менять его размеры с помощью мыши, как и многих обыкновенных Win-программ, при этом в панели свойства (Properties), размеры (Size) будут меняться автоматически. Т.е. размеры можно менять не только с помощью мыши, но и с помощью свойств! В панели свойств так же можно найти свойства связанные с изменением цвета поверхности – BackColor (по умолчанию такого же цвета будут все элементы управления), связанные с изменением размера и стиля шрифта (Font), с изменением фоновой картинки (BackgroundImage), с изменением вида курсора (Cursor), с изменением цвета шрифта (ForeColor), стиля окна (FormBorderStyle), заголовком формы - Text (нашей поверхности), с изменением имени формы (Name), с начальной позицией (StartPosition), с кнопками максимизации и минимизации и многое другое.

Поэкспериментируйте с этими и другими свойствами, посмотрите изменения на форме, связанные с изменением свойств.

Шаг 4. Теперь, когда вы более или менее усвоили свойства связанные с формой, можно рассмотреть события. На той же панели свойств есть кнопка с пиктограммой «молния». Щелкнув на ней мы перейдем к событиям связанным с текущим (выделенного на данной момент) элемента, т.е. с формой (если она не выделена, то выделить и перейти к событиям). Отобразятся различные события, например, события связанные с мышью (движение, движение в определенных направлениях, вход или выход из определенной области), с различными нажатиями клавиши мыши, нажатием кнопки клавиатуры, с рисованием, с изменением стиля, с размером окна и многие другие (подробнее с теми или иными событиями можно ознакомиться самостоятельно, используя источники с более глубокой детализацией материала). Чтобы задействовать то или иное событие достаточно лишь дважды щелкнуть на его названии и компилятор автоматически создаст имя и тело метода, который будет обрабатывать выбранной событие. Так же можно ввести самостоятельно имя метода и сметить фокус от выбранного события – тело метода создастся автоматически с введенным именем! Существует еще и третий способ: сначала создается метод обрабатывающий событие, а затем выбирается в выпадающем списке! В теле метода можно написать все что заблагорассудится (действия связанные с обработкой события)!

Шаг 5. Пожалуй, это самое интересное! Теперь можно без разбору «накидать» на поверхность формы из Toolbox все что угодно! ☺ Посмотреть, как это выглядит, какие свойства и события связанны с тем или иным элементом, поэкспериментировать, откомпилировать, но не слишком увлекайтесь – «ждет» шаг шестой!

Шаг 6. Чтобы показать как «это чудо» работает, разработаем простенькое приложение калькулятор для сложения двух чисел.

Для этого, во-первых, удаляем все «набросанные» элементы управления, поверхность формы должна быть пустой.

Во-вторых, «выкладываем» (перетаскиваем) из Toolbox на форму два Textbox, пять Label и одну Button. Добиваемся чтобы это выглядело примерно как на рисунке 1.

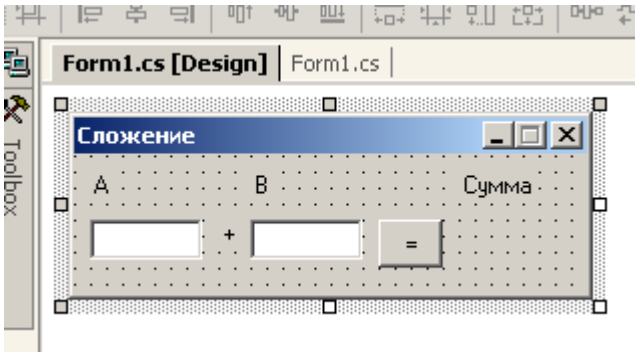

Рис. 1.

Поясним, почему именно пять Label. Первые четыре – «A», «B», «+», «Сумма», пятый – пустой ставится после кнопки под Label с текстом «Сумма».

Затем, для удобства изменим имена (не текст!) Textbox, кнопки и пятой (не видимой) метке. Первому Textbox присваиваем имя «A», второму – «B», лейблу (метке) – «C», а кнопке – «Calculate». Все это делается, если Вы еще не забыли, в панели свойств (не забывайте при этом выделить нужный элемент!). Теперь нам нужно добавить обработку лишь одного события – нажатие кнопки. Это можно сделать всеми перечисленными выше способами или просто двойным щелчком по кнопке. И сразу же автоматически появляется код приложения с телом метода, обрабатывающего событие нажатия кнопки.

Пояснение. При двойном щелчке по любому элементу управления автоматически создается метод по обработке того или иного события, но(!!!) создается всегда, то событие обработка, которого более характерна для данного элемента управления. Для кнопки – нажатие на кнопку, для формы – загрузка и т.п.

В теле метода нужно дописать код, который будет выглядеть следующим образом:

```
private void Calculate_Click(object sender, System.EventArgs e)
{
 double a, b;
 try//Обработка исключений - тема не сегодняшней лабы!
 {
 a=Convert.ToDouble(A.Text);/*Считываем текст (A.Text),
 затем конвертируем его в формат double*/
 b=Convert.ToDouble(B.Text);/*Считываем текст (B.Text),
 затем конвертируем его в формат double*/
 C.Text=Convert.ToString(a+b);/*Присваиваем тексту метки
 (C.Text) конвертированную в строковый формат сумму чисел
```

```
(a+b)полученных из двух редакторов*/
}
catch//Обработка исключений - тема не сегодняшней лабы!
{
MessageBox.Show("Проверьте правильность ввода чисел!");
}
}
```

Компилируем приложение. Если все-таки что-то не работает, проверьте, все ли вы сделали правильно! Свертись с полным листингом приложения. Если и после этого не работает – зовите препода! ☺

Так же в тело метода обработки нажатия кнопки мыши можно добавить следующий код (при этом увеличьте размеры поверхности, иначе ничего не увидите!):

```
Pen p;//Объявляем перо
   Graphics g=CreateGraphics();/*Это нужно для того,
     *  чтобы на поверхность выводился
      * рисунок. В будуще, возможно, этот
     * вопрос будет рассмотрен более  подробно*/
   Color clr = new Color();/*Ну это понятно, создается объект типа
         * Color для хранения цвета*/
    float R2,r2,k=1,R=200;
    int x,y;
    int red=20,gre=250,blu=150;
    clr=Color.FromArgb((int)(k*red),(int)(k*gre),(int)(k*blu));/*
        * Запоминаем цвет в формате RGB*/
    p=new Pen(clr);//Создаем перо
    R2=R*R;
     /*Здесь рисуем шарик окружностями
     * увеличивая его радиус и меняя цвет каждой окружности*/
    for(y=0;y<=R;y++)
     for(x=0;x<=y;x++)
     {
      r2=(float)2*x*x;
      if(r2>R2) break;
      k=1-r2/R2;
   clr=Color.FromArgb((int)(k*red),(int)(k*gre),(int)(k*blu));
       p=new Pen(clr);
       g.DrawEllipse(p,(float)(150-0.5*x),(float)(150-0.5*x),(float)x,(float)x);//  Именно здесь происходит вывод окружностей на поверхность
      }
   g.DrawArc(p,50,60,25,47,24,68);//Рисуем обыкновенную кривую
   p=System.Drawing.Pens.Fuchsia;//Выбираем стандартный цвет из библиотеки цветов
   g.DrawBezier(p,50,54,68,75,69,78,75,64);// рисуем кривую Безье
```

Этот код выводит картинку на поверхность. Подробнее вопрос о классах для рисования, возможно, будет рассмотрен в будущем. В рамках данного методического пособия пример рисования приведен лишь для выполнения задания!

Задания:

1). Создать приложение, в котором вводятся координаты окружности и меняются по нажатию кнопки.

2). Создать приложение, в котором движется окружность, по нажатию одной кнопки он останавливается, по нажатию второй кнопки считывается значения красной, зеленой и синей компонент из трех Textbox-ов и изменяется цвет окружности.

3). Создать приложение, в котором вводятся три переменные x, y, z, а затем по нажатию кнопки рисуется x – кривых, x+y – кривых Безье, x+y+z – окружностей с произвольными координатами.

4). Создать приложение, в котором при нажатии одной копки окружность начинает двигать/останавливаться, по нажатию другой - радиус окружности начинает увеличивать (до определенного момента), затем уменьшаться (до определенного момента), после повторного нажатия кнопки изменение радиуса прекращается.

5). Создать приложение, в котором окружность движется в том направлении что движется и мышь.

6). Создать приложение, в котором окружность начинает двигаться в противоположном направлению движения мыши.

7). Создать приложение, в котором направление движения окружности задается нажатием кнопок (стрелок) клавиатуры).

8). Создать приложение, в котором по нажатию правой кнопки мыши появляется окружность с произвольными координатами, по нажатию на левую – окружности исчезают по одной.

Листинг.

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;

namespace FirstWindowsApplication
{
 /// <summary>
 /// Summary description for Form1.
 /// </summary>
 public class Form1 : System.Windows.Forms.Form
 {
  private System.Windows.Forms.TextBox A;
  private System.Windows.Forms.Label label1;
  private System.Windows.Forms.Label label2;
  private System.Windows.Forms.TextBox B;
  private System.Windows.Forms.Label label4;
```

```
private System.Windows.Forms.Button Calculate;
private System.Windows.Forms.Label C;
private System.Windows.Forms.Label label5;
/// <summary>
/// Required designer variable.
/// </summary>
private System.ComponentModel.Container components = null;

public Form1()
{
 //
 // Required for Windows Form Designer support
 //
 InitializeComponent();

 //
 // TODO: Add any constructor code after InitializeComponent call
 //
}

/// <summary>
/// Clean up any resources being used.
/// </summary>
protected override void Dispose( bool disposing )
{
 if( disposing )
 {
  if (components != null)
  {
   components.Dispose();
  }
 }
 base.Dispose( disposing );
}

#region Windows Form Designer generated code
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
 this.A = new System.Windows.Forms.TextBox();
 this.label1 = new System.Windows.Forms.Label();
 this.label2 = new System.Windows.Forms.Label();
```

```
this.B = new System.Windows.Forms.TextBox();
this.label4 = new System.Windows.Forms.Label();
this.Calculate = new System.Windows.Forms.Button();
this.C = new System.Windows.Forms.Label();
this.label5 = new System.Windows.Forms.Label();
this.SuspendLayout();
//
// A
//
this.A.Location = new System.Drawing.Point(8, 32);
this.A.Name = "A";
this.A.Size = new System.Drawing.Size(56, 20);
this.A.TabIndex = 0;
this.A.Text = "";
this.A.TextAlign = System.Windows.Forms.HorizontalAlignment.Right;
//
// label1
//
this.label1.AutoSize = true;
this.label1.Location = new System.Drawing.Point(8, 8);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(12, 13);
this.label1.TabIndex = 1;
this.label1.Text = "A";
//
// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(88, 8);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(12, 13);
this.label2.TabIndex = 2;
this.label2.Text = "B";
//
// B
//
this.B.Location = new System.Drawing.Point(88, 32);
this.B.Name = "B";
this.B.Size = new System.Drawing.Size(56, 20);
this.B.TabIndex = 3;
this.B.Text = "";
this.B.TextAlign = System.Windows.Forms.HorizontalAlignment.Right;
//
// label4
//
```

```
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(192, 8);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(40, 13);
this.label4.TabIndex = 5;
this.label4.Text = "Сумма";
//
// Calculate
//
this.Calculate.Location = new System.Drawing.Point(152, 32);
this.Calculate.Name = "Calculate";
this.Calculate.Size = new System.Drawing.Size(32, 24);
this.Calculate.TabIndex = 6;
this.Calculate.Text = "=";
this.Calculate.Click += new System.EventHandler(this.Calculate_Click);
//
// C
//
this.C.AutoSize = true;
this.C.Location = new System.Drawing.Point(192, 32);
this.C.Name = "C";
this.C.Size = new System.Drawing.Size(0, 13);
this.C.TabIndex = 7;
//
// label5
//
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(72, 32);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(11, 13);
this.label5.TabIndex = 8;
this.label5.Text = "+";
//
// Form1
//
this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
this.ClientSize = new System.Drawing.Size(256, 69);
this.Controls.AddRange(new System.Windows.Forms.Control[] {
                        this.label5,
                        this.C,
                        this.Calculate,
                        this.label4,
                        this.B,
                        this.label2,
                        this.label1,
```

```
                                this.A});
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;
            this.MaximizeBox = false;
            this.Name = "Form1";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Сложение";
            this.ResumeLayout(false);

        }
        #endregion

        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.Run(new Form1());
        }

        private void Calculate_Click(object sender, System.EventArgs e)
        {
            double a, b;
            try//Обработка исключений - тема не сегодняшней лабы!
            {
            a=Convert.ToDouble(A.Text);/*Считываем текст (A.Text),
              затем конвертируем его в формат double*/
            b=Convert.ToDouble(B.Text);/*Считываем текст (B.Text),
              затем конвертируем его в формат double*/
            C.Text=Convert.ToString(a+b);/*Присваиваем тексту метки
            (C.Text) конвертированную в строковый формат сумму чисел
            (a+b)полученных из двух редакторов*/
            }
            catch//Обработка исключений - тема не сегодняшней лабы!
            {
            MessageBox.Show("Проверьте правильность ввода чисел!");
            }
        }

    }
}
```

Задания для лабораторной работы

Вариант 1

Создать меню с командами Input, Calc и Exit.

При выборе команды Input открывается диалоговое окно, содержащее:

• три поля типа TextBox для ввода длин трех сторон треугольника;

• группу из двух флажков (Периметр и Площадь) типа CheckBox;

• кнопку типа Button.

Обеспечить возможность:

• ввода длин трех сторон треугольника;

• выбора режима с помощью флажков: подсчет периметра и/или площади треугольника.

При выборе команды Calc открывается диалоговое окно с результатами. При выборе команды Exit приложение завершается.

Вариант 2

Создать меню с командами Size, Color, Paint, Quit.

Команда Paint недоступна. При выборе команды Quit приложение завершается. При выборе команды Size открывается диалоговое окно, содержащее:

• два поля типа TextBox для ввода длин сторон прямоугольника;

• группу из трех флажков (Red, Green, Blue) типа CheckBox;

• кнопку типа Button.

Обеспечить возможность:

• ввода длин сторон прямоугольника в пикселах в поля ввода;

• выбора его цвета с помощью флажков.

После задания параметров команда Paint становится доступной. При выборе команды Paint в главном окне приложения выводится прямоугольник заданного размера и сочетания цветов или выдается сообщение, если введенные размеры превышают размер окна.

Вариант 3

Создать меню с командами Input, Work, Exit.

При выборе команды Exit приложение завершает работу. При выборе команды Input открывается диалоговое окно, содержащее:

• три поля ввода типа TextBox с метками Radius, Height, Density;

• группу из двух флажков (Volume, Mass) типа CheckBox;

• кнопку типа Button.

Обеспечить возможность:

• ввода радиуса, высоты и плотности конуса;

• выбора режима с помощью флажков: подсчет объема и/или массы конуса.

При выборе команды Work открывается окно сообщений с результатами.

Вариант 4

Создать меню с командами Input, Calc, Draw, Exit.

При выборе команды Exit приложение завершает работу. При выборе команды Input открывается диалоговое окно, содержащее:

• поле ввода типа TextBox с меткой Radius;

• группу из двух флажков (Square, Length) типа CheckBox;

• кнопку типа Button.

Обеспечить возможность:

• ввода радиуса окружности;

• выбора режима с помощью флажков: подсчет площади круга (Square) и/или длины окружности (Length).

При выборе команды Calc открывается окно сообщений с результатами. При выборе команды Draw в центре главного окна выводится круг введенного радиуса или выдается сообщение, что рисование невозможно (если диаметр превышает размеры рабочей области).

Вариант 5

Создать меню с командами input, Calc, About.

При выборе команды About открывается окно с информацией о разработчике. При выборе команды Input открывается диалоговое окно, содержащее:

• три поля ввода типа TextBox с метками Number 1, Number 2, Number 3;

• группу из двух флажков (Summ, Least multiple) типа CheckBox;

• кнопку типа Button.

Обеспечить возможность ввода трех чисел и выбора режима вычислений с помощью флажков: подсчет суммы трех чисел (Summ) и/или наименьшего общего кратного двух первых чисел (Least multiple). При выборе команды Calc открывается диалоговое окно с результатами.

Вариант 6

Создать меню с командами Input, Calc, Quit.

Команда Calc недоступна. При выборе команды Quit приложение завершается. При выборе команды Input открывается диалоговое окно, содержащее:

• два поля ввода типа TextBox с метками Number 1, Number 2;

• группу из трех флажков (Summa, Max divisor, Multiply) типа CheckBox;

• кнопку типа Button.

Обеспечить возможность:

• ввода двух чисел;

• выбора режима вычислений с помощью флажков (можно вычислять в любой комбинации такие величины, как сумма, наибольший общий делитель и произведение двух чисел). При выборе команды Calc открывается окно сообщений с результатами.

Вариант 7

Создать меню с командами Begin, Help, About.

При выборе команды About открывается окно с информацией о разработчике. При выборе команды Begin открывается диалоговое окно, содержащее:

• поле ввода типа TextBox с меткой input;

• метку типа Label для вывода результата;

• группу из трех переключателей (2, 8, 16) типа RadioButton;

• две кнопки типа Button — Do и OK.

Обеспечить возможность:

• ввода числа в десятичной системе в поле input;

• выбора режима преобразования с помощью переключателей: перевод в двоичную, восьмеричную или шестнадцатеричную систему счисления.

При щелчке на кнопке Do должен появляться результат перевода.

Вариант 8

Создать меню с командами Input color, Change, Exit, Help.

При выборе команды Exit приложение завершает работу. При выборе команды Input color открывается диалоговое окно, содержащее:

• три поля ввода типа TextBox с метками Red, Green, Blue;

• группу из двух флажков (Left, Right) типа CheckBox;

• кнопку типа Button.

Обеспечить возможность ввода RGB-составляющих цвета. При выборе команды Change цвет главного окна изменяется на заданный (левая, правая или обе половины окна в зависимости от установки флажков).

Вариант 9

Создать меню с командами Input size, Choose, Change, Exit.

При выборе команды Exit приложение завершает работу. Команда Change недоступна. При выборе команды Input size открывается диалоговое окно, содержащее:

• два поля ввода типа TextBox с метками Size x, Size y;

• кнопку типа Button.

При выборе команды Choose открывается диалоговое окно, содержащее:

• группу из двух переключателей (Increase, Decrease) типа RadloButton;

• кнопку типа Button.

Обеспечить возможность ввода значений в поля Size x и Size y. Значения интерпретируются как количество пикселов, на которое надо изменить размеры главного окна (увеличить или уменьшить в зависимости от положения переключателей). После ввода значений команда Change становится доступной. При выборе этой команды размеры главного окна увеличиваются или уменьшаются на введенное количество пикселов.

Вариант 10

Создать меню с командами Begin, Work, About.

При выборе команды About открывается окно с информацией о разработчике. При выборе команды Begin открывается диалоговое окно, содержащее:

• поле ввода типа TextBox с меткой Input word;

• группу из двух переключателей (Upper case, Lower case) типа RadioButton;

• кнопку типа Button.

Обеспечить возможность ввода слова и выбора режима перевода в верхний или нижний регистр в зависимости от положения переключателей. При выборе команды Work открывается диалоговое окно с результатом перевода.

Вариант 11

Создать меню с командами Input color, Change, Clear.

При выборе команды Input color открывается диалоговое окно, содержащее:

• группу из двух флажков (Up, Down) типа CheckBox;

• группу из трех переключателей (Red, Green, Blue) типа RadioButton;

• кнопку типа Button.

Обеспечить возможность:

• выбора цвета с помощью переключателей;

• ввода режима, определяющего, какая область закрашивается; все окно, его верхняя или нижняя половина.

При выборе команды Change цвет главного окна изменяется на заданный (верхняя, нижняя или обе половины в зависимости от введенного режима). При выборе команды Clear восстанавливается первоначальный цвет окна.

Вариант 12

Создать меню с командами Translate, Help, About, Exit.

При выборе команды Exit приложение завершает работу. При выборе команды Translate открывается диалоговое окно, содержащее:

• поле ввода типа TextBox с меткой Binary number;

• поле ввода типа TextBox для вывода результата (read-only);

• группу из трех переключателей (8, 10, 16) типа RadioButton;

• кнопку Do типа Button.

Обеспечить возможность:

• ввода числа в двоичной системе в поле Binary number;

• выбора режима преобразования с помощью переключателей: перевод в восьмеричную, десятичную или шестнадцатеричную систему счисления. При щелчке на кнопке Do должен появляться результат перевода.

Вариант 13

Создать меню с командами Reverse, About, Exit.

При выборе команды About открывается окно с информацией о разработчике. При выборе команды Reverse открывается диалоговое окно, содержащее:

• поле ввода типа TextBox с меткой Input;

• группу из двух переключателей (Upper case, Reverse) типа CheckBox;

• кнопку ОК типа Button.

Обеспечить возможность ввода фразы и выбора режима: перевод в верхний регистр и/или изменение порядка следования символов на обратный в зависимости от состояния переключателей. Результат преобразования выводится в исходное поле ввода.

Вариант 14

Создать меню с командами Begin, Work, About.

При выборе команды About открывается окно с информацией о разработчике. При выборе команды Begin открывается диалоговое окно, содержащее:

• два поля ввода типа TextBox;

• группу из двух переключателей (First letter, All letters) типа RadioButton;

• кнопку типа Button.

Обеспечить возможность ввода предложения и выбора режима его преобразования: либо начинать с прописной буквы каждое слово (First letter), либо перевести все буквы в верхний регистр (All letters). При выборе команды Work открывается диалоговое окно с результатом преобразования.

Вариант 15

Создать меню с командами Input, Change, Exit.

При выборе команды Exit приложение завершает работу. Команда Change недоступна. В центре главного окна выведен квадрат размером 100 x 100 пикселов.

При выборе команды Input открывается диалоговое окно, содержащее:

• два поля ввода типа TextBox с метками Size x, Size y;

• группу из двух переключателей (Increase, Decrease) типа RadioButton;

• кнопку типа Button.

Обеспечить возможность ввода значений в поля Size x и Size y. Значения интерпретируются как количество пикселов, на которое надо изменить размеры квадрата, выведенного в главное окно (увеличить или уменьшить в зависимости от положения переключателей). После ввода значений команда Change становится доступной. При выборе этой команды размеры квадрата увеличиваются или уменьшаются на введенное количество пикселов. Если квадрат выходит за пределы рабочей области окна, выдается сообщение.