ИЗУЧЕНИЕ СРЕДЫ РАЗРАБОТКИ VISUAL STUDIO.

**Цель лабораторной работы:** изучить среду быстрой разработки приложений Visual Studio. Научиться размещать и настраивать внешний вид элементов управления на форме.

**Задание:** разместить на форме ряд полей ввода (TextBox). Создайть обработчики события нажатия кнопкой мыши на данные поля ввода, которые будут выводить в текущее поле ввода его номер. Создайть обработчик события изменения размера формы (Resize), который будет очищать все поля ввода.

**Теория:**

Среда Visual Studio визуально реализуется в виде одного окна с несколькими панелями инструментов. Количество, расположение, размер и вид панелей может меняться программистом или самой средой разработки в зависимости от текущего режима работы среды или пожеланий программиста, что значительно повышает производительность работы.

*Приложение Windows Forms* – данный тип проекта позволяет создать полноценное приложение с окнами и элементами управления (кнопками, полями ввода и пр.) Такой вид приложения наиболее привычен большинству пользователей.

*Консольное приложение* – в этом типе проекта окно представляет собой текстовую консоль, в которую приложение может выводить тексты или ожидать ввода информации пользователя. Консольные приложения часто используются для вычислительных задач, для которых не требуется сложный или красивый пользовательский интерфейс.

*Форма* - рабочее поле, будущее окно приложения.

*Окно свойств* - окно с приведенными свойствами.

Для размещения различных элементов управления на форме используется *панель элементов*. Панель элементов содержит элементы управления, сгруппированные по типу. Каждую группу элементов управления можно свернуть, если она в настоящий момент не нужна.

*Обозреватель решений* содержит список всех файлов, входящих в проект, включая добавленные изображения и служебные файлы.

*Окно текста программы* предназначено для просмотра, написания и редактирования текста программы.

Программа на языке C# составляется как описание алгоритмов, которые необходимо выполнить, если возникает определенное событие, связанное с формой (например, щелчок «мыши» на кнопке – событие Click, загрузка формы – Load). Для каждого обрабатываемого в форме события, с помощью окна свойств, в тексте программы организуется метод, в котором программист записывает на языке C# требуемый алгоритм.

Если необходимо ввести из формы в программу или вывести на форму информацию, которая вмещается в одну строку, используют окно однострочного редактора текста, представляемого элементом управления TextBox.

На форме могут размещаться пояснительные надписи. Для нанесения таких надписей на форму используется элемент управления Label.

С каждым элементом управления на форме и с самой формой могут происходить события во время работы программы. Например, с кнопкой может произойти событие – нажатие кнопки, а с окном, которое проектируется с помощью формы, может произойти ряд событий: создание окна, изменение размера окна, щелчок мыши на окне и т. п. Эти события могут обрабатываться в программе. Для обработки таких событий необходимо создать обработчики события – специальный *метод*. Для создания обработчика события существует два способа.

Первый способ – создать обработчик для события по умолчанию (обычно это самое часто используемое событие данного элемента управления). Например, для кнопки таким образом создается обработчик события нажатия.

Второй способ создания обработчика события заключается в выборе соответствующего события для выделенного элемента на форме. При этом используется окно свойств и его закладка в виде молнии.

Каждый элемент управления имеет свой набор обработчиков событий, однако некоторые из них присущи большинству элементов управления. Наиболее часто применяемые события описаны ниже:

 Activated: форма получает это событие при активации.

 Load: возникает при загрузке формы. В обработчике данного события следует задавать действия, которые должны происходить в момент создания формы, например установка начальных значений.

 KeyPress: возникает при нажатии кнопки на клавиатуре. Параметр e.KeyChar имеет тип char и содержит код нажатой клавиши (клавиша Enter клавиатуры имеет код #13, клавиша Esc – #27 и т. д.). Обычно это событие используется в том случае, когда необходима реакция на нажатие одной из клавиш. 

KeyDown: возникает при нажатии клавиши на клавиатуре. Обработчик этого события получает информацию о нажатой клавише и состоянии клавиш Shift, Alt и Ctrl, а также о нажатой кнопке мыши. Информация о клавише передается параметром e.KeyCode, который представляет собой перечисление Keys с кодами всех клавиш, а информацию о клавишах-модификаторах Shift и др. можно узнать из параметра e.Modifiers.

 KeyUp: является парным событием для KeyDown и возникает при отпускании ранее нажатой клавиши.

 Click: возникает при нажатии кнопки мыши в области элемента управления.

 DoubleClick: возникает при двойном нажатии кнопки мыши в области элемента управления.

Запустить программу можно, выбрав в меню Отладка команду Начать отладку. При этом происходит трансляция и, если нет ошибок, компоновка программы и создание единого загружаемого файла с расширением .exe. На экране появляется активное окно программы.

Если в программе есть ошибки, то в окне Список ошибок появятся найденные ошибки, а программа обычно не запускается.

Свойства элементов на окне могут быть изменены динамически во время выполнения программы. Например, можно изменить текст надписи или цвет формы. Изменение свойств происходит внутри обработчика события (например, обработчика события нажатия на кнопку). Для этого используют оператор присвоения вида:

<имя элемента>.<свойство> = <значение>;

Например:

label1.Text = "Привет";

<Имя элемента> определяется на этапе проектирования формы, при размещении элемента управления на форме. Например, при размещении на форме ряда элементов TextBox, эти элементы получают имена textBox1, textBox2, textBox3 и т. д., которые могут быть заменены в окне свойств в свойстве (Name) для текущего элемента. Допускается использование латинских или русских символов, знака подчеркивания и цифр (цифра не должна стоять в начале идентификатора). Список свойств для конкретного элемента можно посмотреть в окне свойств, а также в приложении к данным методическим указаниям. Если требуется изменить свойства формы, то никакое имя элемента перед точкой вставлять не нужно, как и саму точку. Например, чтобы задать цвет формы, нужно просто написать:

BackColor = Color.Green;

**Блок-схема:**

Начало

public Form1()

Конец

InitializeComponent();

Начало

private void textBox1\_Click(object sender, EventArgs e)

Конец

textBox1.Text = "1";

Конец

textBox2.Text = "2";

Начало

private void textBox2\_Click(object sender, EventArgs e)

Конец

textBox2.Text = "2";

Начало

private void textBox2\_Click(object sender, EventArgs e)

Начало

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

Конец

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

**Код программы:**

using System;

using System.Windows.Forms;

namespace WindowsFormsApplication1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void textBox1\_Click(object sender, EventArgs e)

{

textBox1.Text = "1";

}

private void textBox2\_Click(object sender, EventArgs e)

{

textBox2.Text = "2";

}

private void textBox3\_Click(object sender, EventArgs e)

{

textBox3.Text = "3";

}

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

{

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

}

}

}

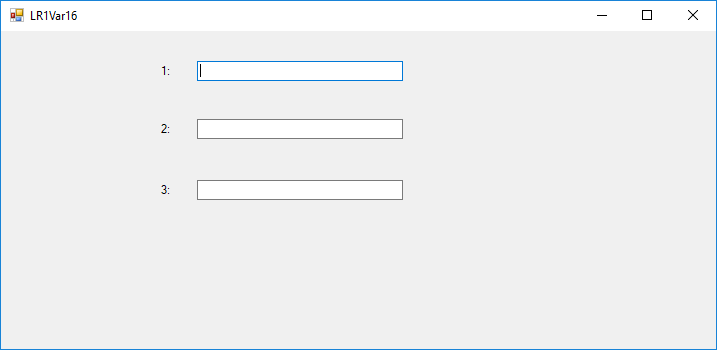


Рис 1. Внешний вид программы.

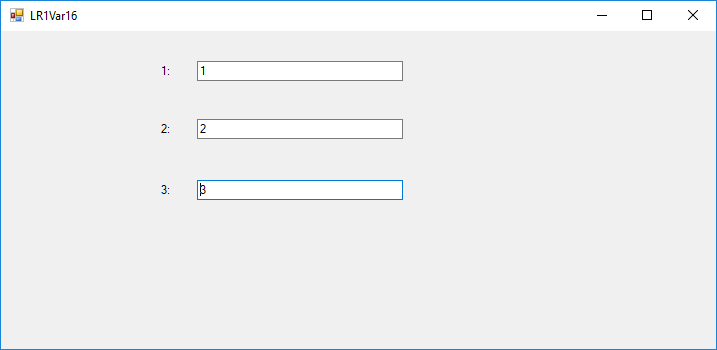


Рис 2. Работа программы. Появление порядковых номеров текстовых полей при нажатии на них.

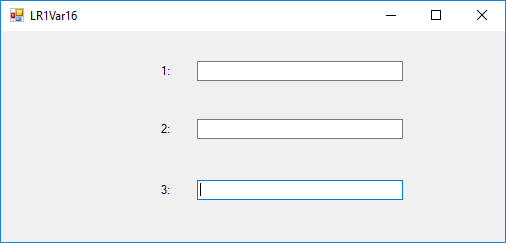


Рис 3. Работа программы. Очистка текстовых полей при изменении размера окна.

**Вывод:**

Выполнили задание заданное преподавателем. Изучили заданную теорию по Microsoft Visual Studio Windows Forms на языке C#. Оформили отчет по проделанной работе.