

## Практическое занятие №1 Изучение среды разработки VISUAL STUDIO

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

### Методические указания

Профессиональное программирование требует профессиональных средств. Основным таким средством является интегрированная среда разработки (IDE – Integrated development environment), которая объединяет в себя редактор кода, визуальные средства программирования, компилятор, отладчики другие инструментальные средства.

### Скачивание и установка Visual Studio

Бесплатно скачать Visual Studio можно на сайте [visualstudio.com](https://visualstudio.com). На этой странице нужно выбрать Visual Studio Community и скачать программу для установки Visual Studio. Эта программа в начале установит Visual Studio Installer, который позволяет выбрать необходимые компоненты Visual Studio, которые вам необходимы. Поскольку Visual Studio имеет огромное количество различных компоненты, многие, из которых вам не понадобятся никогда и при этом объем этих компонент может быть весьма большим, то я рекомендую вам устанавливать в начале минимальный набор компонент, которые нужны для программирования на языке C#.

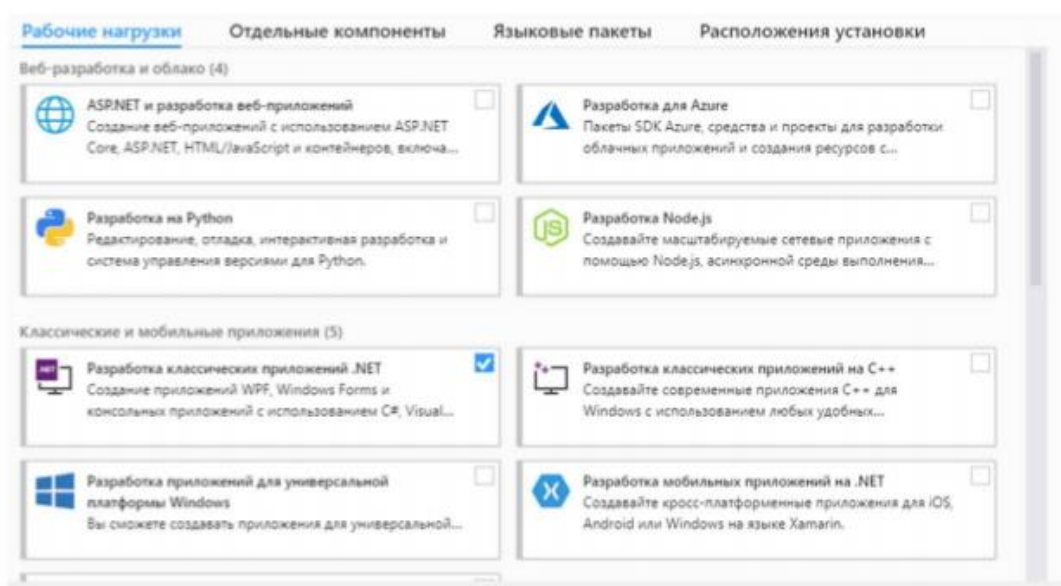


Рис. 1. Выбор компонентов Visual Studio.

На рис. 1 приведены лишь некоторые компоненты Visual Studio, которые можно выбрать при установке. Минимально нужно выбрать лишь компонент «Разработка классических приложений .NET», как указано на этом рисунке. Дальнейшая установка Visual Studio не вызывает каких-либо трудностей. Для использования Visual Studio необходимо иметь учетную запись Microsoft (Live ID), которую бесплатно нужно получить на сайте: [account.microsoft.com](http://account.microsoft.com). Для этого переходим по ссылке «Создать учетную запись Microsoft». После этого в форме, представленной на рис. 2 необходимо ввести вашу электронную почту или телефон. После чего нажать кнопку «Далее». На новой форме нужно придумать пароль, который должен удовлетворять некоторым правилам безопасности.

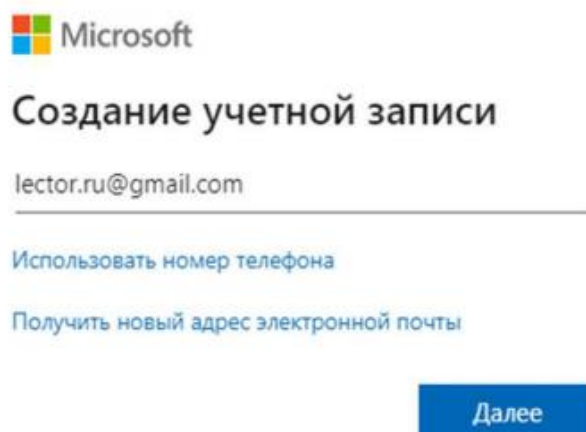


Рис. 2. Создание учетной записи Microsoft.

После ввода пароля вам нужно будет ввести страну, и дату рождения. После этого на указанный вами адрес придет письмо с кодом подтверждения, который нужно ввести в форме. Наконец, нужно будет пройти проверку, что вы не робот, для чего нужно будет ввести предложенные на картинке символы.

После этого созданную учетную запись нужно будет ввести при запуске Visual Studio.

### Создание проекта

Каждая программа, написанная в Visual Studio, должна быть в рамках проекта. *Проект* это основная единица, с которой работает программист. При создании проекта можно выбрать его тип, а Visual Studio создаст каркас проекта в соответствии с выбранным типом.

Проект имеет определенное название и располагается в отдельной папке. Он содержит все исходные материалы для приложения, такие как

файлы исходного кода, ресурсов, значки, ссылки на внешние файлы, на которые опирается программа, и данные конфигурации, такие как параметры компилятора.

Кроме понятия проект часто используется более глобальное понятие – *решение (solution)*. Решение содержит один или несколько проектов, один из которых может быть указан как стартовый проект. Выполнение решения начинается со стартового проекта.

Таким образом, при создании простейшей C# программы в Visual Studio. Создается папка решения, в которой для каждого проекта создается подпапка проекта, в которой будут создаваться другие подпапки с результатами компиляции приложения.

Рассмотрим процесс создания проекта и его запуска. Для создания проекта нужно выбрать кнопку «Создание проекта» (рис.3).

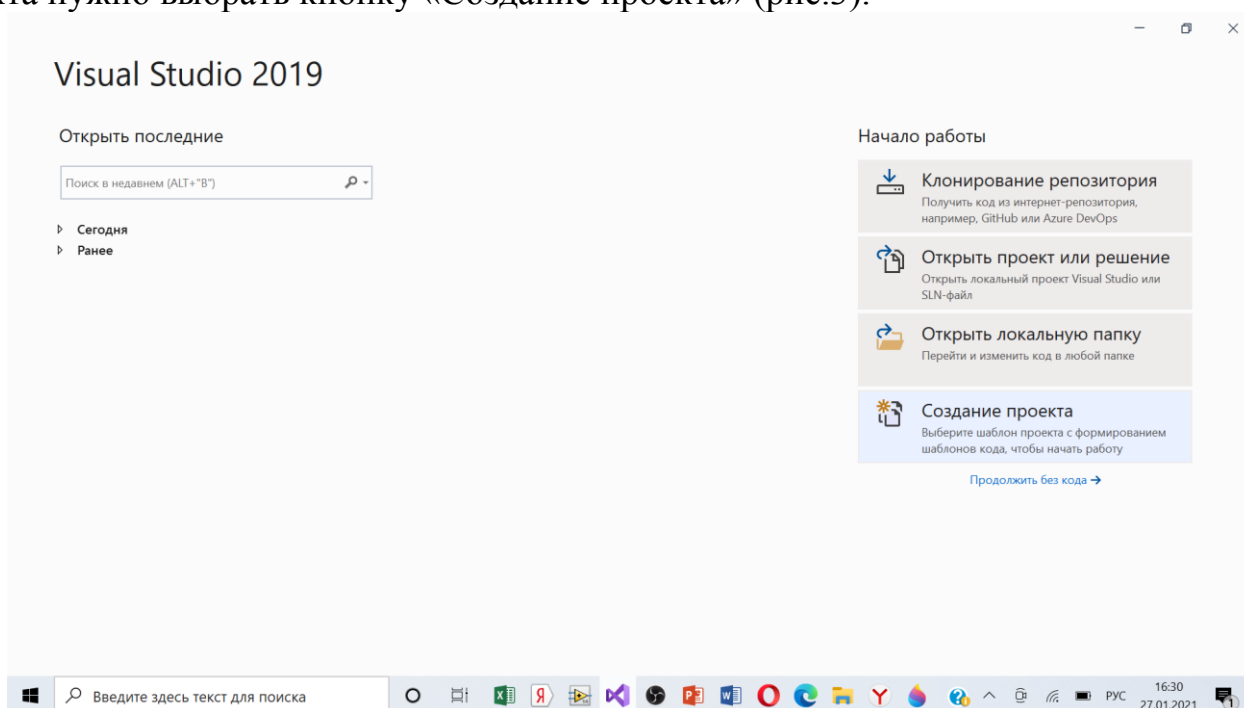


Рис.3 Форма создания проекта

После этого нужно будет выбрать шаблон проекта, см. рис. 4.

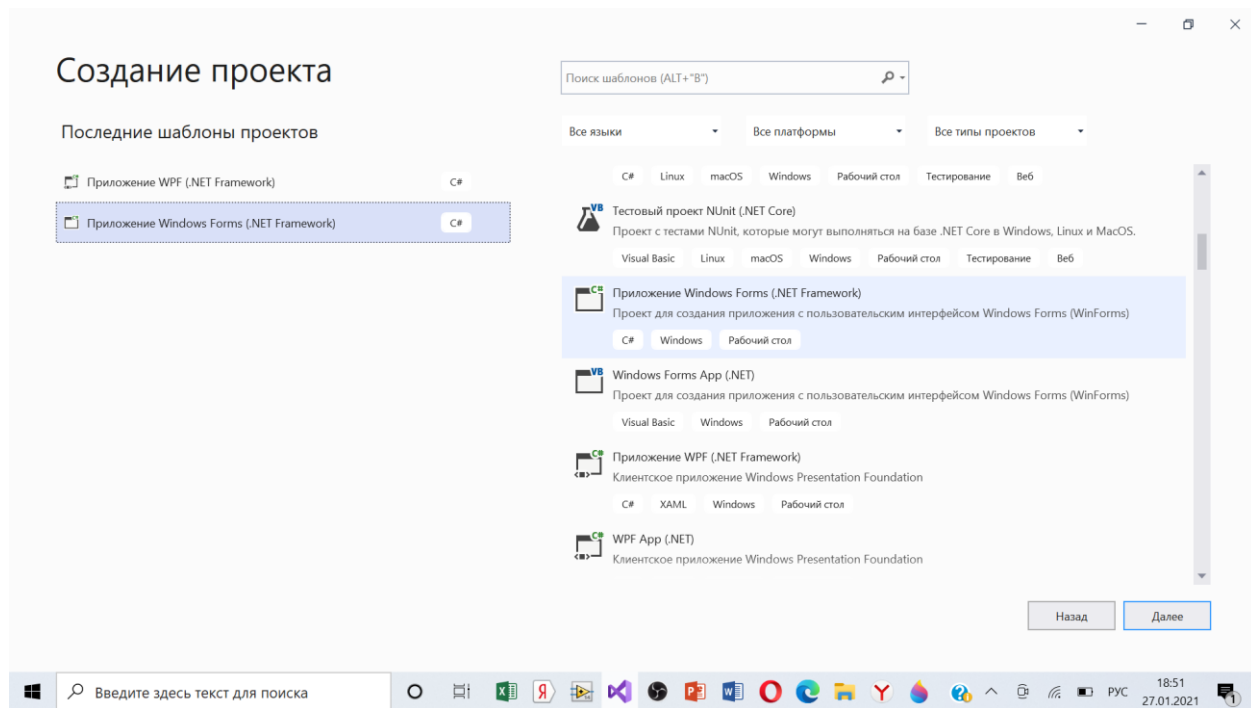


Рис.4 Выбор шаблона проекта

В списке шаблонов приведены языки программирования, которые поддерживает данная версия Visual Studio: убедитесь, что там выделен раздел C#. Также справа приведены типы проектов, которые можно создать. Нами будут использоваться два типа проектов:

- 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.

После нажатия кнопки «Создать» у вас откроется проект, который будет выглядеть как на рис.5.

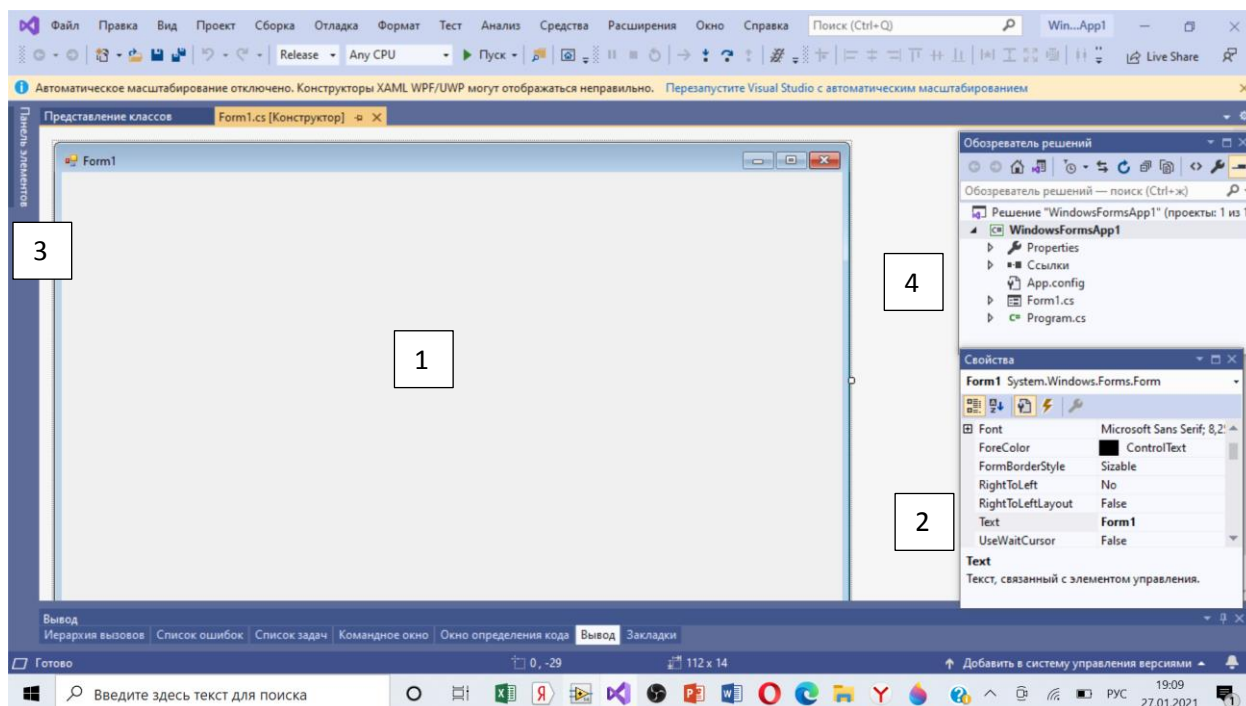




Рис.5 Открытый проект

В главном окне Visual Studio присутствует несколько основных элементов, которые будут помогать нам в работе. Прежде всего, это **форма** (1) – будущее окно нашего приложения, на котором будут размещаться элементы управления. При выполнении программы помещенные элементы управления будут иметь тот же вид, что и на этапе проектирования.

Второй по важности объект – это **окно свойств** (2), в котором приведены все основные свойства выделенного элемента управления или окна. С помощью кнопки  можно просматривать свойства элемента управления, а кнопка  переключает окно в режим просмотра событий. Если этого окна на экране нет, его можно активировать в меню Вид -> Окно свойств.

Сами элементы управления можно брать на **панели элементов** (3). Все элементы управления разбиты на логические группы, что облегчает поиск нужных элементов. Если панели нет на экране, её нужно активировать командой Вид -> Панель элементов.

Наконец, **обозреватель решений** (4) содержит список всех файлов, входящих в проект, включая добавленные изображения и служебные файлы. Активируется командой Вид -> Обзорщик решений.

**Окно текста** программы предназначено для просмотра, написания и редактирования текста программы. Переключаться между формой и текстом программы можно с помощью команд Вид -> Код (F7) и Вид -> Конструктор (Shift+F7). При первоначальной загрузке в окне текста программы находится текст, содержащий минимальный набор операторов для нормального функционирования пустой формы в качестве Windows-окна. При помещении

элемента управления в окно формы, текст программы автоматически дополняется описанием необходимых для его работы библиотек стандартных

программ (раздел using) и переменных для доступа к элементу управления (в скрытой части класса формы).

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.

## Настройка формы

Настройка формы начинается с настройки размера формы. С помощью мыши, «захватывая» одну из кромок формы или выделенную строку заголовка отрегулируйте нужные размеры формы.

Для настройки будущего окна приложения задаются свойства формы. Для задания любых свойств формы и элементов управления на форме используется окно свойств.

Новая форма имеет одинаковые имя (Name) и заголовок (Text) - Form1.

Для изменения заголовка перейдите в окно свойств и щелкните кнопкой мыши на форме. В форме инспектора объектов найдите и щелкните мышью на строчке с названием Text. В выделенном окне наберите “Практическая работа N1. Иванов И.И.”. Для задания цвета окна используйте свойство BackColor.

## Первая программа на языке C#

Язык программирования C# Язык программирования C# является современным универсальным языком, который пользуется большой популярностью благодаря своей мощности и внутренней красоте. Этот язык является одним из основных языков программирования не только для Windows, но для веб-программирования и мобильных приложений. Язык C# является строго типизированным, объектно-ориентированным языком программирования. Также как в языках C++ и Pascal каждую переменную перед использованием необходимо определять, указывая ее тип. При этом в дальнейшем переменная не может менять свой тип. Эти ограничения позволяют C# являться безопасным и комфортным языком программирования. Идеология C# неразрывно связана с платформой .NET Framework, которая изначально была для операционной системы Windows, но сейчас развивается и в Linux, а также на мобильных устройствах. Программа, написанная на C# исполняется в среде .NET, хотя внешне скомпилированная программа и выглядит как обычный .EXE файл. Поэтому для работы такой программы необходимо наличие среды .NET Framework, которая, как мы уже отмечали, всегда присутствует в современных версиях

Windows. Именно наличие этой платформы делает язык C# таким мощным и удобным.

### Графический интерфейс пользователя

Почти любая программа должна иметь интерфейс пользователя, с помощью которого она принимает от пользователя входную информацию и выводит результат своей работы. Различают два вида пользовательского интерфейса: консольный, когда вывод осуществляется в текстовом виде, и графический.

Мы будем изучать и использовать графический интерфейс пользователя, чтобы создавать программы, аналогичные которыми мы пользуемся при работе с Windows. Графический интерфейс пользователя основан на понятии окна. Окном называется, как правило, прямоугольная область, которая имеет следующие элементы:

- рамку;
- заголовок окна;
- меню окна;
- кнопки управления окном (свернуть, развернуть, закрыть);
- область окна.

Эти элементы показаны на рис.6.



Рис.6 Элементы окна

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. Для



выполнения практических работ потребуются элементы управления из группы Стандартные элементы управления.

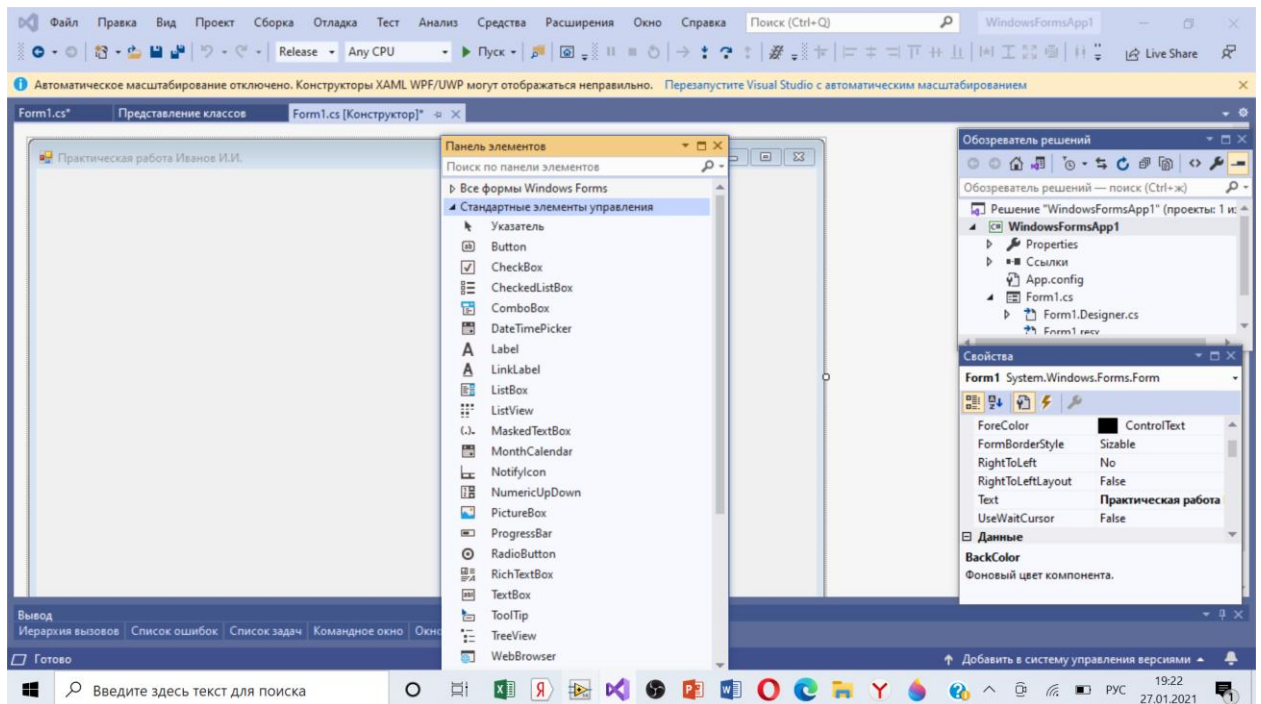




Рис. 6 Панель элементов

Щёлкните на нужном элементе управления, а затем щёлкните в нужном месте формы – элемент появится на форме. Элемент можно перемещать по форме схватившись за него левой кнопкой мышки (иногда это можно сделать лишь за появляющийся при нажатии на элемент квадрат со стрелками ). Если элемент управления позволяет изменять размеры, то на соответствующих его сторонах появятся белые кружки, ухватившись за которые и можно изменить размер. После размещения элемента управления на форме, его можно выделить щелчком мыши и при этом получить доступ к его свойствам в окне свойств.

### ***Размещение строки ввода (TextBox)***

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.


В данной программе с помощью однострочного редактора будут вводиться переменные  $x$ ,  $y$  типа `double` или `int`.

Выберите на панели элементов пиктограмму , щелкните мышью в том месте формы, где вы хотите ее поставить. Вставьте три элемента TextBox в форму. Захватывая их “мышью” отрегулируйте размеры окон и их положение. Обратите внимание на то, что теперь в тексте программы можно использовать переменные `textBox1`, `textBox2` которые соответствуют каждому добавленному элементу управления. В каждой из этих переменных в свойстве `.Text` будет содержаться строка символов (тип `string`) и отображаться в соответствующем окне TextBox.



С помощью инспектора объектов установите шрифт и размер символов отражаемых в строке TextBox (свойство Font).

### ***Размещение надписей (Label)***


На форме могут размещаться пояснительные надписи. Для нанесения таких надписей на форму используется элемент управления Label. Выберите на панели элементов пиктограмму  **A**, щелкните на ней мышью. После этого в нужном месте формы щелкните мышью, появится надпись label1. Прделайте это для четырех надписей. Для каждой надписи, щелкнув на ней мышью, отрегулируйте размер и, изменив свойство Text в окне свойств, введите строку, например “Введите значение X:”, а также выберите размер символов (свойство Font).

Обратите внимание, что в тексте программы теперь можно обращаться к четырём новым переменным типа Label. В них хранятся пояснительные строки, которые можно изменять в процессе работы программы.

### ***Написание программы обработки события***

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 быть 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 создаётся обработчик события нажатия.

#### ***Написание программы обработки события нажатия кнопки (Click)***

Поместите на форму кнопку, которая описывается элементом управления Button, для чего выберем пиктограмму  **ab**. С помощью окна свойств измените заголовок (Text) на слово “Выполнить” или другое по вашему желанию. Отрегулируйте положение и размер кнопки.

После этого два раза щелкните мышью на кнопке, появится текст программы:


```
private void button1_Click(object sender, EventArgs e)
{

}
```

Это и есть обработчики события нажатия кнопки. Вы можете добавлять свой код между скобками { }. Например, наберите:

MessageBox.Show("Привет!");

### **Написание программы обработки события загрузки формы (Load)**

Второй способ создания обработчика события заключается в выборе соответствующего события для выделенного элемента на форме. При этом используется окно свойств и его закладка . Рассмотрим этот способ. Перейдите на форму, в окне свойств найдите событие **Load**. Щелкните по данной строчке дважды мышкой. Появится метод:

```
private void Form1_Load(object sender, EventArgs e)
{

}
```

Между скобками { } вставим текст программы:

```
BackColor = Color.AntiqueWhite;
```

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

Событие	Описание события
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 OnKeyDown 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
-------------	---

## Запуск программы

Для запуска проекта нужно переключить конфигурацию проекта на «Release» и нажать зеленую кнопку «Пуск». Или 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. **Для завершения работы программы и возвращения в режим проектирования формы не забудьте закрыть окно программы!**

## *Динамическое изменение свойств*

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 Мир";

<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. Эти имена могут быть замены 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.

## Пример написания элементарного калькулятора

Пусть элементарный калькулятор имеет следующий вид:

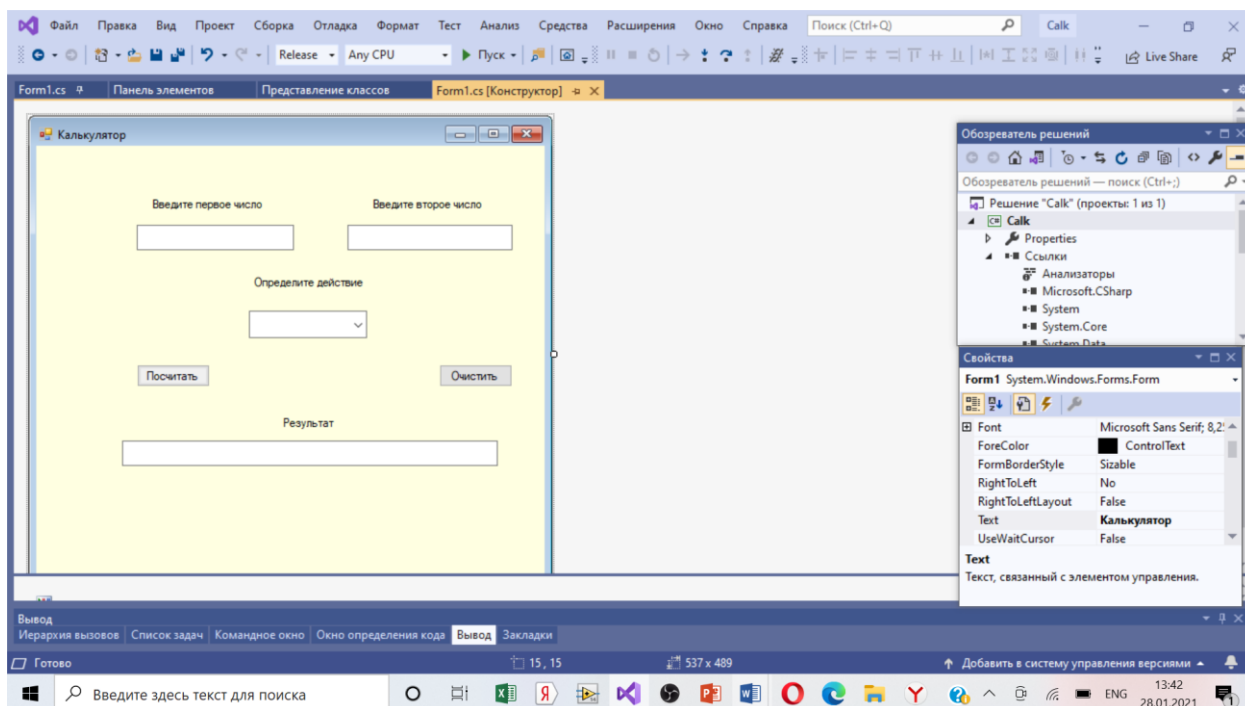


Рис.7 Проект калькулятор

Алгоритм работы калькулятора следующий. Пользователь вводит два числа, выбирает арифметическое действие из выпадающего списка нажимает кнопку «Посчитать» и видит результат вычисления. У него есть возможность очистить все поля с помощью кнопки «Очистить». Программа предусматривает обработку ошибки деление на ноль.

Форма выглядит следующим образом (см. рис 7). Для организации ввода первого числа, второго числа и вывода результата использованы элементы **textBox1**, **textBox2**, **textBox3** соответственно. Для определения действия взят элемент **comboBox1**. В свойстве этого элемента **Items** в таблице (Коллекция) перечислены возможные арифметические операции (+, -, \*, /). Для размещения надписей использовались элементы **Label1**, **Label2**, **Label3**, **Label4**. На форме присутствуют две управляющие кнопки **button1** – посчитать и **button2** – очистить.

Обработчик событий по нажатию кнопки «Посчитать» приведен на следующем рисунке.

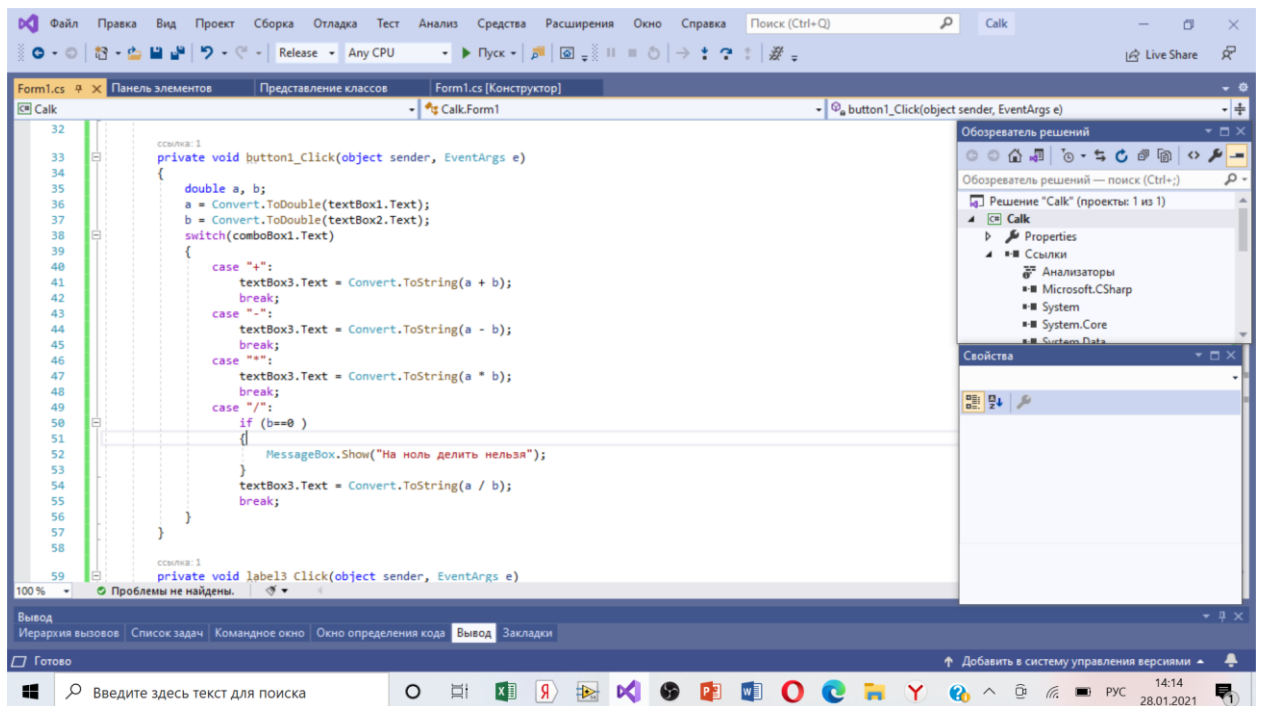


Рис.8 Обработчик событий кнопки «Посчитать».

В программе также присутствует кнопка «Очистить». Обработчик текст программы этого обработчика событий представлен на рисунке 9.

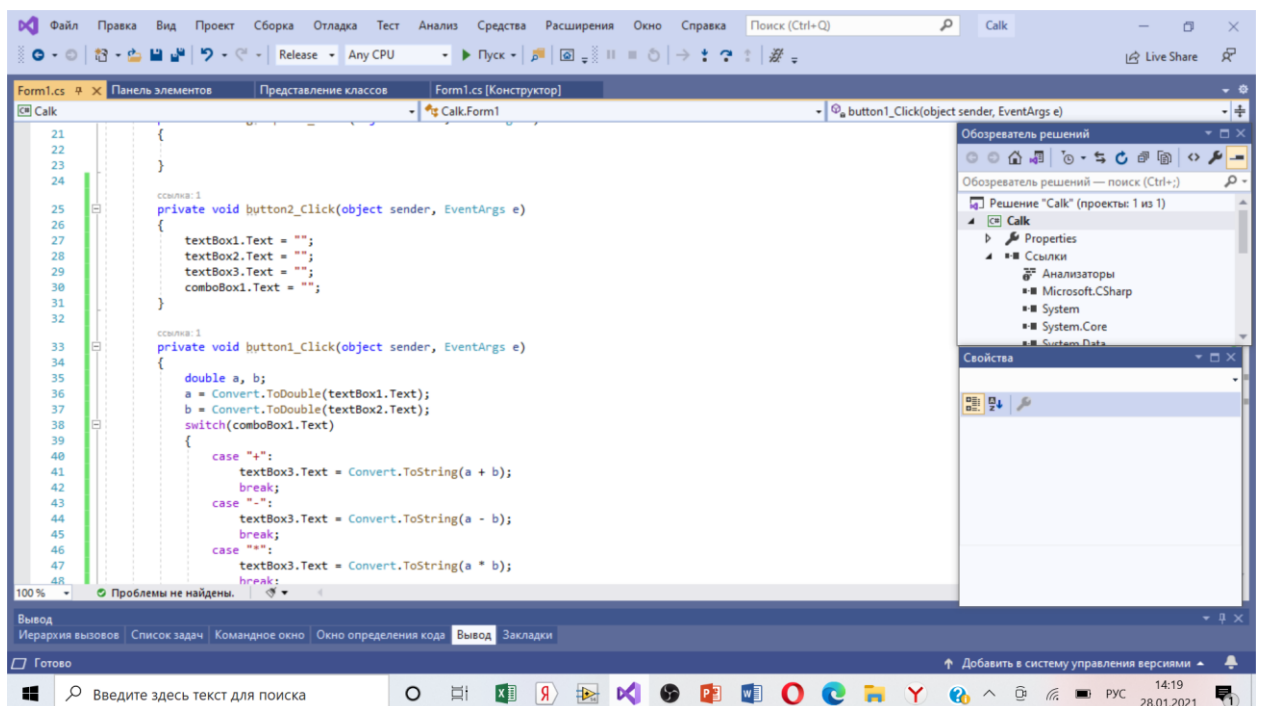


Рис.8 Обработчик событий кнопки «Очистить».

### Индивидуальные задания

1. Разместите на форме четыре кнопки (Button). Сделайте на кнопках следующие надписи: красны, зеленый, синий, желтый. Создайте

- четыре обработчика события нажатия на данные кнопки, которые будут менять цвет формы в соответствии с текстом на кнопках.
2. Разместите на форме две кнопки (Button) и одну метку (Label). Сделайте на кнопках следующие надписи: привет, до свидания. Создайте обработчики события нажатия на данные кнопки, которые будут менять текст метки, на слова: привет, до свидания. Создайте обработчик события создания формы (Load), который будет устанавливать цвет формы и менять текст метки на строку «Начало работы».
  3. Разместите на форме две кнопки (Button) и одну метку (Label). Сделайте на кнопках следующие надписи: скрыть, показать. Создайте обработчики события нажатия на данные кнопки, которые будут скрывать или показывать метку. Создайте обработчик события создания формы (Load), который будет устанавливать цвет формы и менять текст метки на строку «Начало работы».
  4. Разместите на форме три кнопки (Button) и одно поле ввода (TextBox). Сделайте на кнопках следующие надписи: скрыть, показать, очистить. Создайте обработчики события нажатия на данные кнопки, которые будут скрывать или показывать поле ввода. При нажатии на кнопку «очистить» текст из поля ввода должен быть удален.
  5. Разместите на форме две кнопки (Button) и одно поле ввода (TextBox). Сделайте на кнопках следующие надписи: заполнить, очистить. Создайте обработчики события нажатия на данные кнопки, которые будут очищать или заполнять поле ввода знаками «\*\*\*\*\*». Создайте обработчик события создания формы (Load), который будет устанавливать цвет формы и менять текст в поле ввода на строку «++++++».
  6. Разработайте игру, которая заключается в следующем. На форме размещены пять кнопок (Button). При нажатии на кнопку какие то кнопки становятся видимыми, а какие то невидимыми. Цель игры скрыть все кнопки.
  7. Разработайте игру, которая заключается в следующем. На форме размещены четыре кнопки (Button) и четыре метки (Label). При нажатии на кнопку часть надписей становится невидимыми, а часть наоборот становятся видимыми. Цель игры скрыть все надписи.
  8. Разместите на форме ряд кнопок (Button). Создайте обработчики события нажатия на данные кнопки, которые будут делать неактивными текущую кнопку. Создайте обработчик события изменение размера формы (Resize), который будет устанавливать все кнопки в активный режим.
  9. Разместите на форме ряд кнопок (Button). Создайте обработчики события нажатия на данные кнопки, которые будут делать неактивными следующую кнопку. Создайте обработчик события нажатия кнопки мыши на форме (Click), который будет устанавливать все кнопки в активный режим.



10. Разместите на форме три кнопки (Button) и одно поле ввода (TextBox). Сделайте на кнопках следующие надписи: \*\*\*\*\*, +++++, 00000. Создайте обработчики события нажатия на данные кнопки, которые будут выводить текст, написанный на кнопках, в поле ввода. Создайте обработчик события создания формы (Load), который будет устанавливать цвет формы и менять текст в поле ввода на строку «Готов к работе».
11. Разместите на форме ряд полей ввода (TextBox). Создайте обработчики события нажатия кнопкой мыши на данные поля ввода, которые будут выводить в текущее поле ввода его номер. Создайте обработчик события изменение размера формы (Resize), который будет очищать все поля ввода.
12. Разместите на форме поле ввода (TextBox), метку (Label) и кнопку (Button). Создайте обработчик события нажатия на кнопку, который будет копировать текст из поля ввода в метку. Создайте обработчик события нажатия кнопки мыши на форме (Click), который будет устанавливать цвет формы и менять текст метки на строку «Начало работы» и очищать поле ввода.
13. Разместите на форме поле ввода (TextBox), и две кнопки (Button) с надписями: заблокировать, разблокировать. Создайте обработчики события нажатия на кнопки, которые будут делать активным или неактивным поле ввода. Создайте обработчик события нажатия кнопки мыши на форме (Click), который будет устанавливать цвет формы и делать невидимыми все элементы.
14. Реализуйте игру минер на поле 3x3 из кнопок (Button). Первоначально все кнопки не содержат надписей. При попытке нажатия на кнопку на ней либо показывается количество мин, либо надпись «мина!» и меняется цвет окна.
15. Разместите на форме четыре кнопки (Button). Напишите для каждой обработчик события, который будет менять размеры и местоположение на окне других кнопок.