МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ
Проректор по учебной работе
_____ О.Г. Локтионова
«_18_» _____ 2019 г.

ИЗУЧЕНИЕ ИНТЕГРИРОВАННОЙ СРЕДЫ MS VISUAL STUDIO C#

Методические указания по выполнению лабораторной работы по дисциплинам «Программирование на языках высокого уровня», «Языки программирования» для студентов направлений подготовки 09.03.04 «Программная инженерия», 10.03.01 «Информационная безопасность», 10.05.02 «Информационная безопасность телекоммуникационных систем»

Курск 2019

УДК 681.3.06(071.8)

Составители: Т.М. Белова, В.Г. Белов

Рецензент
Кандидат технических наук, доцент кафедры программной инженерии ЮЗГУ И.Н. Ефремова

Изучение интегрированной среды MS Visual Studio C#: методические указания по выполнению лабораторной работы по дисциплинам «Программирование на языках высокого уровня», «Языки программирования» для студентов направлений подготовки 09.03.04 «Программная инженерия», 10.03.01 «Информационная безопасность», 10.05.02 «Информационная безопасность телекоммуникационных систем»/ Юго-Зап. гос. ун-т; сост. Т.М. Белова, В.Г. Белов. Курск, 2019. – 32 с.

Содержат основные теоретические положения и приемы разработки программ в интегрированной среде ***MS Visual Studio C#***, пример решения типовой задачи, индивидуальные задания и контрольные вопросы к защите лабораторной работы.

Методические указания соответствуют требованиям рабочих программ по дисциплинам «Программирование на языках высокого уровня», «Языки программирования».

Предназначены для студентов дневной и заочной форм обучения направлений подготовки 09.03.04 «Программная инженерия», 10.03.01 «Информационная безопасность», 10.05.02 «Информационная безопасность телекоммуникационных систем».

Текст печатается в авторской редакции.

Подписано в печать . Формат 60x84 1/16.
Усл. печ. л. . Уч.-изд. л. . Тираж 100 экз. Заказ . Бесплатно.
Юго-Западный государственный университет
305040, Курск, ул.50 лет Октября, 94.

ИЗУЧЕНИЕ ИНТЕГРИРОВАННОЙ СРЕДЫ MS VISUAL STUDIO C#

Цель работы – изучение основных режимов интегрированной среды *MS Visual Studio C#* и приобретение навыков работы с помощью технологии визуального программирования, ввода программ, компиляции и отладки программ.

Основные понятия

Интегрированная среда разработки

MS Visual Studio предлагает пользователю гибкий и современный инструментарий, позволяющий быстро создавать приложения. При работе с ***MS Visual Studio*** можно получить достаточно полную информацию об активном в данный момент средстве программирования, нажав клавишу [F1].

Инструментальные средства, которые поддерживают технологию визуальной разработки программ, являются интегрированными составными частями ***MS Visual Studio***. В связи с этим появился термин «интегрированная среда разработки», сокращенно ***IDE (Integrated Development Environment).*** При дальнейшем изложении ради краткости будем использовать аббревиатуру ***IDE***.

Начало работы

Для начала необходимо создать новый проект. Для создания нового проекта необходимо выбрать в меню Файл опцию *Создать* > *Проект…* и выбрать шаблон для нового проекта, его расположение и (при необходимости) имя решения, в которое будет добавлено в проект. Также можно создать проект, нажав на кнопку *Создать* проект на начальной странице ***MS Visual Studio***. Необходимый шаблон – Приложение ***Windows Forms***.

На первый взгляд

MS Visual Studio состоит из редактора исходного кода, редактора форм, панели обозревателя решений, панели свойств, панели меню. На рисунке 1 показан вид экрана после запуска ***MS Visual Studio.***

Главное окно

В верхней части экрана расположены главное меню, кнопки панели инструментов: запуска, сохранения, открытия файлов, а также параметры сборки.

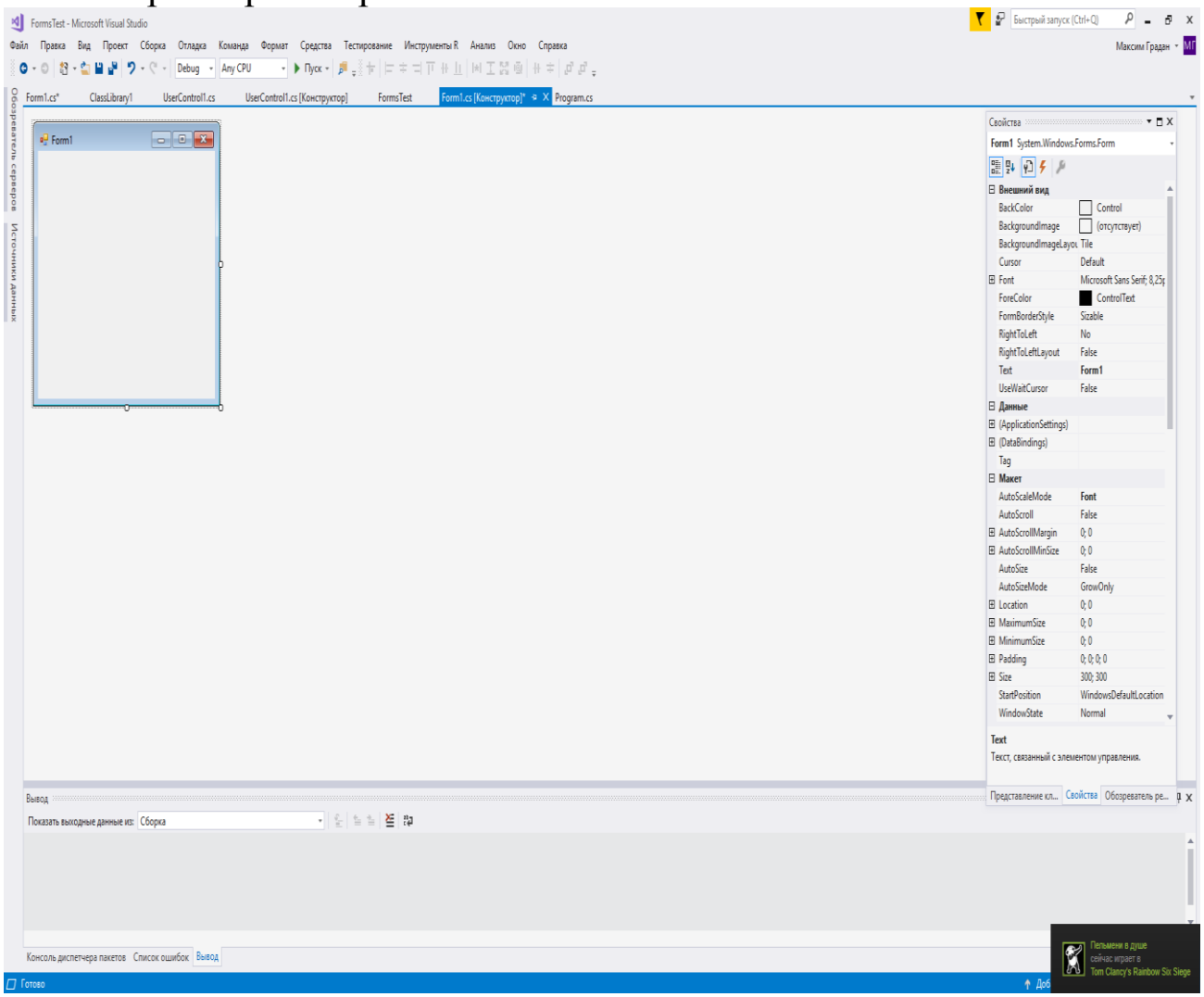

Рисунок 1 – Вид экрана компьютера после запуска ***MS Visual Studio***

Строка заголовка

В строке заголовка главного окна отображается имя открытого в данный момент решения. Решение объединяет в себе несколько проектов одного приложения. В левой части заголовка главного окна, рядом с символом ***MS Visual Studio***, находится надпись <Имя проекта> - ***Microsoft Visual Studio*** — например: ***Project - Microsoft Visual Studio***.

Строка меню

Строка меню содержит команды, необходимые для разработки и тестирования приложений, а также управления ими. Команды меню подробно рассмотрены в разделе «Меню».

Панель инструментов

Панель инструментов (рисунок 2) содержит кнопки, которые соответствуют определенным командам меню, например командам *Файл, Вид* и др. Щелчок на какой-либо из кнопок приводит к тому же результату, что и выбор соответствующей команды в главном меню.

Например, для открытия файла необходимо выбрать команду Открыть меню *Файл* или выполнить щелчок на кнопке *Открыть* файл панели инструментов *Стандартная*. Кнопки панели инструментов снабжены всплывающими подсказками (*hints* или *tooltips*). Если подвести курсор мыши к какой-либо кнопке, на экране будет отображено имя команды, вызываемой данной кнопкой.

Рисунок 2 – Панель инструментов

Пользователь может изменить конфигурацию любой панели инструментов. С этой целью в ***MS Visual Studio*** используется всплывающее окно *Настройка*, посредством которого можно отображать или скрывать на экране панели инструментов, а также добавлять, удалять кнопки панелей инструментов и изменять их расположение. Окно *Настройка* можно открыть, выполнив щелчок правой кнопкой мыши на панели инструментов. В результате на экране будет отображено контекстное меню (рисунок 3). Выберите команду *Настройка…*, после чего на экране должно появиться окно настройки.

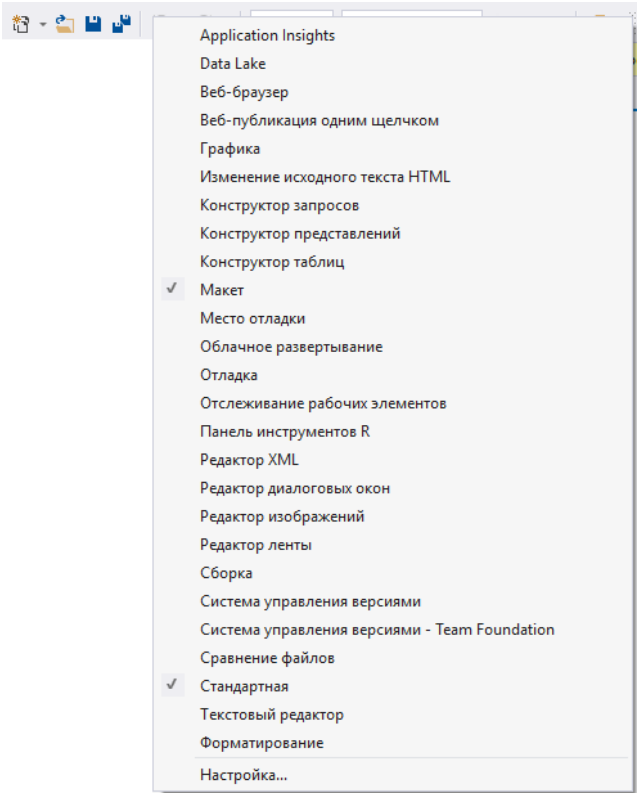

Рисунок 3 – Контекстное меню панели инструментов

Панель элементов

Панели элементов (*Вид -> Панель* элементов) отображаются компоненты, с помощью которых пользователь создает свои приложения. Компоненты являются основными элементами каждого ***MS Visual Studio***-приложения и, одновременно, основой библиотеки визуальных компонентов.

Windows Forms

Этот шаблон позволяет создавать пользовательский интерфейс прикладных программ. Пиктограммы стандартных компонентов ***MS Visual Studio***, в соответствии с выполняемыми ими функциями, разделены на группы; чтобы получить доступ к компонентам группы, нужно щелкнуть левой кнопкой мыши по ее названию.

Компоненты можно переименовывать, удалять, добавлять новые, менять их расположение на страницах. Кроме того, сам порядок страниц может быть изменен. Все эти изменения производятся при помощи диалогового окна Панель элементов (рисунок 4).

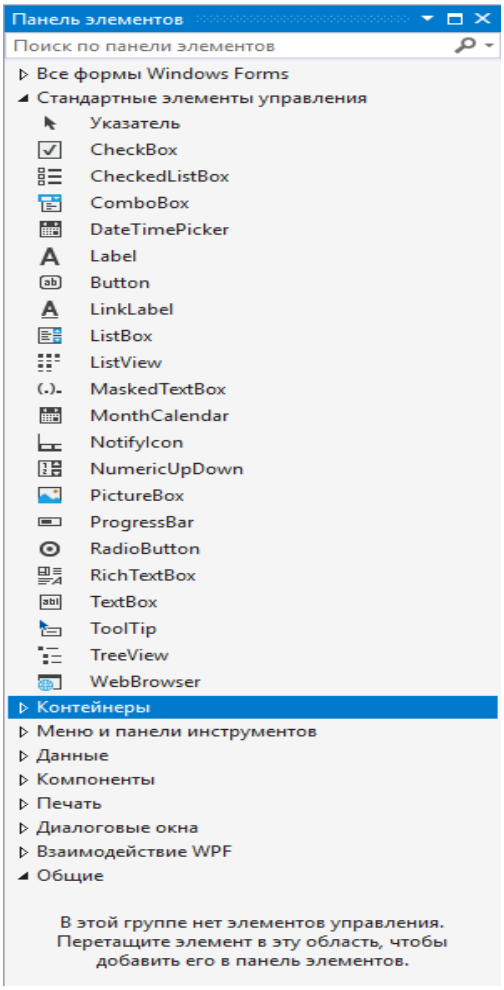

Рисунок 1.4 – Диалоговое окно **Панель элементов**

Диалоговое окно Панель элементов может быть открыто через меню Вид.

Конструктор форм

Каждое Windows-приложение выполняется в собственном окне — главном окне соответствующего приложения. MS Visual Studio назначает главное окно для каждого приложения автоматически. Разработка приложения всегда начинается с создания нового проекта. Для каждого нового проекта в IDE автоматически отображается окно проектировщика формы, которое представляет собой главное окно вашего будущего приложения и по умолчанию имеет имя Form1. Главное окно — это первое, что видит пользователь после запуска приложения. Если пользователь закрывает это окно, тем самым он закрывает приложение.

Для разработчика главное окно — это форма, отображаемая при разработке приложения в конструкторе, в котором создается графический интерфейс пользователя разрабатываемого приложения.

Для получения представления о визуальном проектировании приложений в ***MS Visual Studio***, изменим размер формы и разместим в ней некоторые компоненты. Форма (*Form1*) является Windows-окном, поэтому она может быть увеличена, уменьшена, увеличена до размеров экрана или свернута до пиктограммы.

Теперь разместим на этой форме следующие компоненты: *Label* и *TextBox*. Эти компоненты находятся на странице Стандартные элементы управления панели элементов. Найти указанные компоненты в панели элементов помогут всплывающие подсказки (*hints*).

Для размещения компонента на форме необходимо сначала выполнить щелчок на пиктограмме компонента в панели элементов, а затем выполнить щелчок в том месте проектировщика формы, где должен быть расположен компонент.

При выполнении приложения форма будет по умолчанию отображаться в том же виде, как и при проектировании, то есть будет иметь те же размеры и содержать те же компоненты. Пользователь может изменить размер формы, ее расположение на экране и т.д. в соответствии со своими требованиями (например, воспользовавшись для этого системным меню *Windows*).

Компонент *Label* представляет собой текст (например, название управляющего элемента). Компонент *TextBox* — это поле ввода, где пользователь может ввести строку текста.

Проектировщик форм содержит контекстное меню, посредством которого можно выполнять определенные команды редактирования компонентов. Его можно вызвать, выполнив в любом месте проектировщика формы щелчок правой кнопкой мыши.

Окно редактора кода

Окно редактора кода (*Code Editor*) имеет заголовок Form1.cs (рисунок 5).

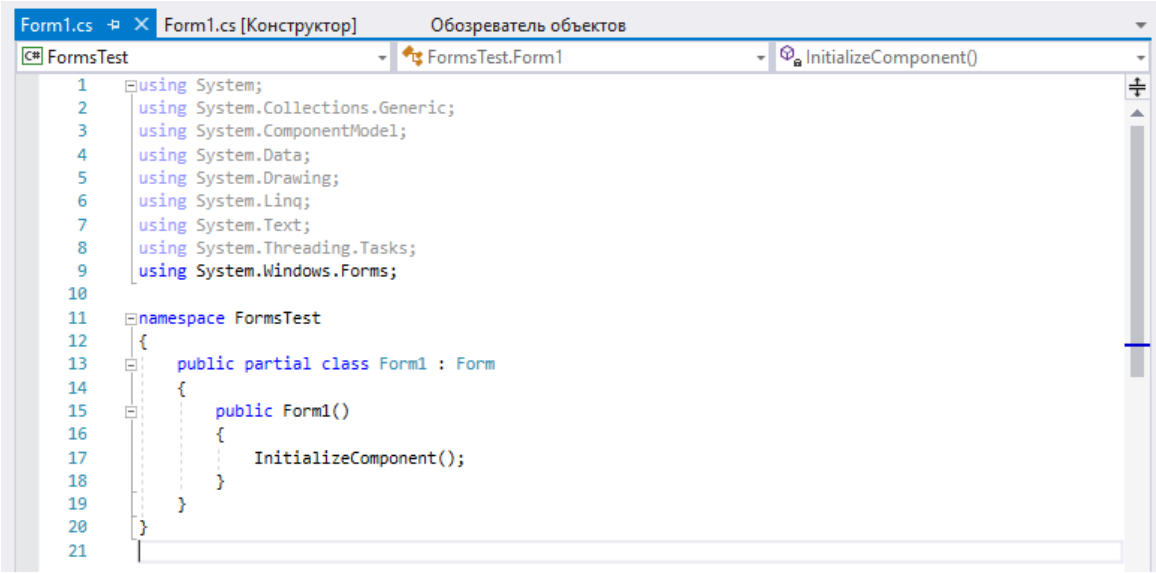

Рисунок 5 – Модуль *Form1.cs* в окне редактора кода

Редактор кода и проектировщик форм тесно связаны между собой. В редакторе кода могут быть открыты несколько файлов. Каждый открытый файл размещается на отдельной странице, а его название отображается в верхней части окна на отдельной вкладке. Выполнив щелчок на имени нужного файла, можно активизировать страницу с исходным кодом модуля, который следует отредактировать или просмотреть. Имя файла, страница которого активизирована, отображается в строке заголовка окна редактора кода.

В правой части окна *IDE* располагается окно представления классов (*Class Representation*). После запуска *IDE* окно представления классов объединено с окнами свойств и обозревателя решений. Однако при желании окно представления классов можно сделать независимым от окна редактора. Для этого достаточно перетащить окно представления классов за пределы окна свойств и пр. При необходимости это окно может быть повторно открыто при помощи команды *Классы* меню *Вид*.

Представление классов упрощает поиск информации в коде программных модулей и автоматизирует создание новых классов.
нижней части окна IDE расположена строка состояния (таблица 1).

Таблица 1 – Информация в строке состояния окна редактора кода

Информация в строке состояния	Значение
Строка 7 Столбец 1	Позиция курсора в тексте (строка и столбец)
ЗАМ	Указывает на то, что редактор находится в режиме замены
ВСТ	Указывает на то, что редактор находится в режиме вставки

Окно редактора кода никогда не бывает пустым. Если открытых файлов нет, окно закрыто. Таким образом, если закрыть все вкладки, то закроется и окно редактора кода. Однако окно редактора кода всегда может быть открыто снова двойным нажатием на элемент в *Обозревателе решений*.

Окно свойств

При разработке приложения часто используется окно *Свойства (Properties)*. Если окна свойств нет на экране, его можно открыть при помощи команды *Окно свойств* меню *Вид*. Поскольку с помощью окна свойств задаются и редактируются свойства и обработчики событий компонентов, его удобно постоянно "иметь под рукой". Окно свойств можно активизировать посредством нажатия клавиши [F4].

Окно *Свойства* содержит две страницы, каждую из которых можно активизировать, выполнив щелчок на пиктограмме с соответствующим названием. Первая страница имеет название *Свойства*. Левая колонка этой страницы содержит список всех свойств редактируемого компонента, доступных во время проектирования. Вторая страница называется *События*. В ее левой колонке перечислены все имеющиеся обработчики событий компонента. В правых колонках обеих страниц могут устанавливаться значения соответствующих свойств или обработчиков событий. Функциональные возможности компонента, используемого в проектируемом приложении, определяются путем присвоения свойствам компонента определенных значений и связывания с обработчиками событий определенных функций.

Таким образом, инспектор объектов является инструментом, который используется для формирования внешнего вида и

функциональных возможностей формы и компонентов в процессе разработки приложения.

Свойства, отображенные на странице *Свойства*, имеют начальные значения. Это стандартные значения данных свойств. Значения в правой колонке могут быть изменены пользователем во время проектирования приложения. Кроме того, значения свойств компонентов могут быть изменены или заданы и в процессе выполнения приложения, но для этого необходимо написать соответствующий программный код. В инспекторе объектов приведены только те свойства, которыми обладает данный компонент во время проектирования приложения (*Design time*). Полный список свойств, которыми данный компонент обладает во время выполнения приложения (*Run time*), можно получить с помощью системы подсказок ***MS Visual Studio***. Некоторые из этих свойств предназначены только для чтения (т.е. имеют статус *Readonly).*

Порядок действий для создания пользовательского интерфейса приложения

Для создания пользовательского интерфейса приложения необходимо выполнить следующие действия:

- выбрать в панели элементов необходимые компоненты и расположить их на форме;
- изменить внешний вид компонентов, задавая в инспекторе объектов значения свойств;
- задать в инспекторе объектов функции обработки событий;
- написать программный код для заданных функций обработки событий.

Составные свойства

Некоторые свойства могут содержать в себе другие свойства, в этом случае они называются *составными,* а содержащиеся в них свойства — *вложенными.* Составные свойства помечены в инспекторе объектов значком "+" (слева от имени свойства). Необходимо щелкнуть на значке "+". Значок плюс изменится на минус "−", и на экране отобразятся вложенные свойства. Теперь можно выбрать свойства, значения которых нужно изменить.

Чтобы снова скрыть список вложенных свойств, необходимо выполнить щелчок на значке "–" или повторный двойной щелчок на имени свойства.

Обозреватель решений

Обозреватель решений — незаменимый инструмент, используемый для выполнения различных операций над файлами. Обозреватель решений (рисунок 6) может быть открыт посредством команды *Обозреватель решений* меню *Вид*.

Обозреватель решений всегда открывается сразу же при открытии некоторого решения (расположен справа).

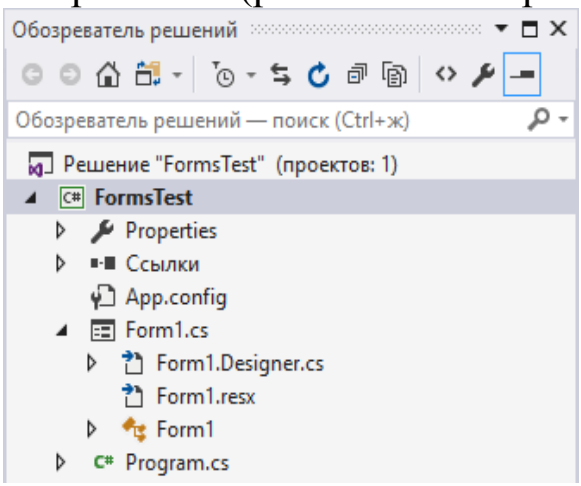

Рисунок 1.6 – Окно Обозреватель решений

Решения

При нажатии правой кнопкой мыши на решение в обозревателе решений открывается контекстное меню, при помощи которого можно добавить:

Таблица 1.2 – Кнопки панели инструментов

Кнопка	Функция
Новый проект	Открыть меню шаблонов для добавления нового проекта
Существующий проект	Указать путь к существующему проекту
Новый/существующий элемент	Добавить отдельный элемент в папку *Solution Items*

При нажатии правой кнопкой мыши на проект в решении также становятся доступны следующие опции в подменю Добавить:

Создать элемент...

Открыть меню доступных заготовок нового элемента проекта.

Существующий элемент…

Указать путь к уже существующему элементу для добавления его в проект.

Удалить

Посредством этой кнопки можно удалить выбранный проект из решения, но не из папки решения (проект можно будет снова добавить, правой кнопкой мыши нажав на решение, затем Добавить>Существующий проект…)

Выгрузить проект

При выполнении этой команды проект не удаляется из решения, но становится недоступен для модификации. Проект можно перезагрузить, щелкнув на него правой кнопкой мыши и выбрав Перезагрузить проект.

Управление файлами

В начале работы над проектом ***MS Visual Studio*** фактически предоставляет в распоряжение пользователя готовую программу. Эта программа состоит из одного окна с заголовком *Form1* и обладает функциональными возможностями стандартного *Windows*-окна (т.е. имеет системное меню, а также кнопки закрытия, минимизации, максимизации и восстановления).

При выборе команды *Запуск* без отладки меню *Отладка* проект будет откомпилирован и запущен на выполнение. В этом заключается важная особенность *IDE*: весь процесс написания, компилирования, компоновки, выполнения, тестирования и отладки программы происходит в одной и той же среде разработки. Таким образом, пользователь может легко переходить от разработки к выполнению программы, что позволяет ему увидеть и протестировать результат своей работы, не выходя из *IDE*.

Кроме того, во время выполнения программы в распоряжении пользователя остаются средства разработки. Это значительно облегчает отладку программ. Например, можно производить изменения в программном коде непосредственно во время выполнения программы. Разумеется, эти изменения проявят себя только после перекомпилирования проекта, которое, однако, происходит очень быстро. На рисунке 1.7 показана стандартная программа, разрабатываемая в *IDE* и находящаяся в режиме выполнения. Открытое меню в форме *Form1* — это стандартное системное меню *Windows*, с помощью которого пользователь может изменять внешний вид окна.

Программный код стандартного приложения генерируется автоматически. При размещении в форме компонентов или их изменении код программы будет соответственно изменен. Кроме этого, ***MS Visual Studio*** автоматически создает некоторые файлы, используемые для различных целей.

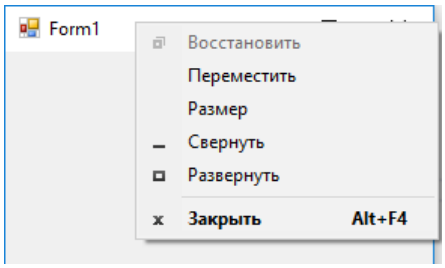

Рисунок 17 – Windows-окно, автоматически создаваемое ***MS Visual Studio***

Автоматически создаваемые файлы

Если при создании стандартной программы были выполнены все описанные ранее действия, ее можно запустить на выполнение при помощи команды *Запуск без отладки* меню *Отладка*.

При этом ***MS Visual Studio*** по умолчанию сохраняет как скомпилированный файл проекта (например, *Project1.exe*), так и некоторые дополнительные файлы (файлы конфигурации и отладочных данных) в подкаталоге *bin/Debug* основного каталога проекта в решении (по умолчанию при инсталляции основным каталогом для решений принимается каталог *C:\Users\User\source\repos*. Для изменения места хранения проекта следует изменить значение параметра *Расположение* при создании

проекта: в этом окне необходимо выбрать каталог, в котором будут созданы папка решения (если это необходимо) и папки проектов.

Для сохранения файлов проекта на стандартной панели инструментов присутствуют кнопки *Сохранить* (сохраняет изменения, произведенные только в текущем файле, открытом в редакторе кода) и *Сохранить все* (сохранить изменения во всех файлах, в которых происходит работа). Обе эти функции доступны также в меню *Файл*.

Кроме того, в меню сборка присутствуют опции *Собрать решение* (выполнить компиляцию всех проектов в решении) и *Построить <Project>* (выполнить компиляцию только текущего проекта). При использовании данных опций также выполняется сохранение изменений в файлах исходного кода проекта.

MS Visual Studio при создании проекта автоматически добавляет в него файлы, соответствующие выбранному шаблону проекта.

Файл с методом Main

В начале работы над новым проектом создается файл *Program.cs*. Он содержит код главной программы, написанной на языке *C#*. В файле проекта содержатся ссылки на все формы проекта и относящиеся к ним модули. В нем также содержится код инициализации приложения. Ниже приведен исходный код файла нового проекта.

Пример

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace FormsTest
{
   static class Program
   {
     // Главная точка входа для приложения.
     // </summary>
    [STAThread]
```

```
    static void Main()
    {
      Application.EnableVisualStyles();
      Application.SetCompatibleTextRenderingDefault(false);
      Application.Run(new Form1());
    }
   }
}
```

Первые три строки содержат директивы *using*, открывающие доступ к классам указанных в данных директивах пространств имен.

Далее идет объявление пространства имена и главного класса приложения, в котором расположен метод *Main*, вызываемый при запуске приложения.

В методе *Main* содержится собственно главный код программы, в котором создается форма и содержится команда запуска основного цикла приложения.

Файлы, создаваемые пользователем

Помимо файлов, создаваемых ***MS Visual Studio*** , в проект могут быть включены ваши собственные файлы, например файлы, написанные на языке *C#* более ранних версий или на других языках программирования. Для доступа к классам в библиотеках и файлах исходного кода необходимо прописать директивы **using** для пространств имен, в которых эти классы содержатся.

В файле проекта обычно указываются все файлы, необходимые для компиляции и/или компоновки EXE- или DLL-файлов. ***MS Visual Studio*** периодически обновляет файл проекта на протяжении всего времени разработки приложения.

Меню

Если не изменять установки, которые были сделаны при инсталляции ***MS Visual Studio*** , панель инструментов будет находиться под строкой меню окна ***MS Visual Studio***. Однако панель элементов является самостоятельным окном и может быть удалена с экрана. По умолчанию главное меню расположено в верхней части экрана (рисунок 8).

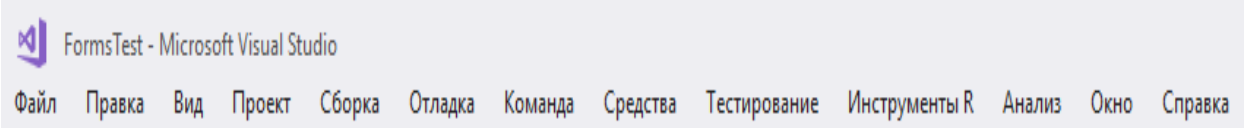

Рисунок 8 – Главное окно ***MS Visual Studio*** со строкой меню

Активизация строки меню

В строке меню (рисунок 1.8) содержатся все команды, необходимые для создания приложений, а также для управления ***MS Visual Studio***. Главное меню ***MS Visual Studio*** используется так же, как любое стандартное Windows-меню.

В подменю *Файл* рядом с командой *Сохранить* расположена надпись *Ctrl+S*. Она свидетельствует о том, что при одновременном нажатии клавиш *[Ctrl+S]* будет выполнена команда *Сохранить* и при этом нет необходимости открывать меню ***Файл***. Такие сочетания клавиш предусмотрены не для всех элементов меню.

Команды меню

На рисунке 8 изображена строка меню со всеми содержащимися в ней элементами меню.

Для построения меню *Windows* предлагает специальный стандарт: *CUA*. Эта аббревиатура расшифровывается как *Common Users Access*. Структура главного меню ***MS Visual Studio*** отвечает данному стандарту. В соответствии с ним первым в строке меню расположено меню *Файл*. Этот стандарт вы можете использовать для собственных приложений. Ниже будут описаны все элементы меню ***MS Visual Studio***, а также, при необходимости, отдельные команды.

Меню Файл

Меню *Файл* имеет шесть основных групп команд:

- в первой группе находятся команды создания или добавления нового элемента;
- во второй группе содержатся команды закрытия;
- в третьей группе располагаются команды сохранения файлов модулей, проектов и файлов других типов;
- четвертая группа включает команды печати;

- пятая группа включает команды доступа к последним файлам;
- в шестой группе находится команда выхода из ***MS Visual Studio***.

Команда Создать

Посредством этой команды открывается подменю, при помощи которого можно создать новый проект, файл или веб-сайт. Каждая из опций открывает окно создания, в котором находятся шаблоны, один из которых можно использовать для нового элемента. Все шаблоны объединены в группы, иерархия которых отображается слева.

Окно **Создать файл** показано на рисунке 9.

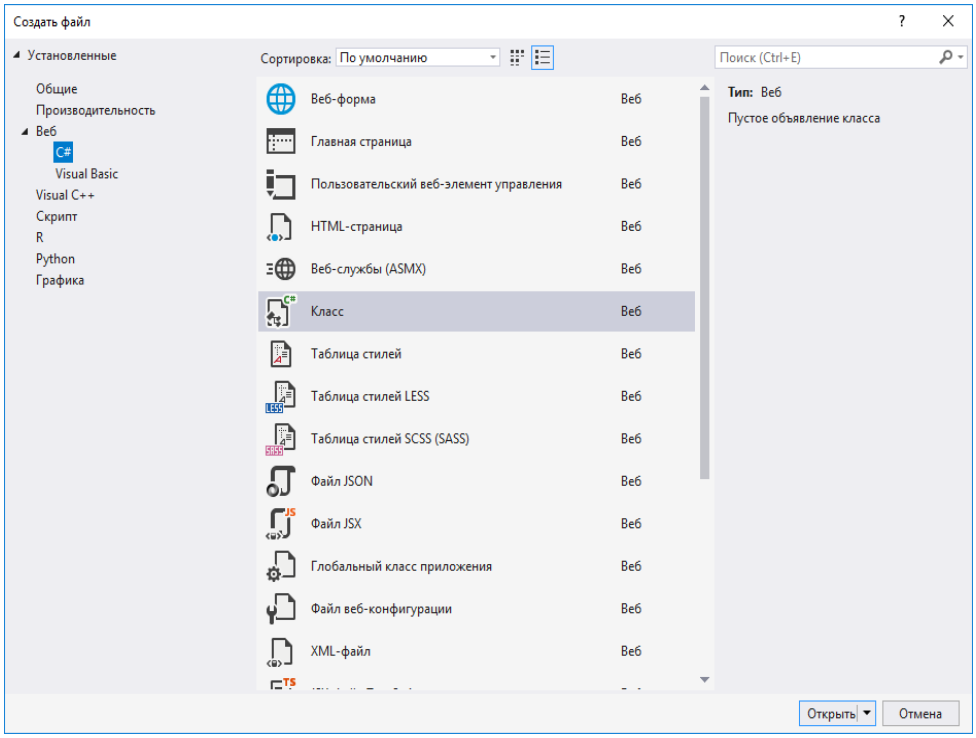

Рисунок 9 – Окно **Создать файл**

Команда Открыть

Эта команда открывает подменю *Открыть* (рисунок 10), в котором выбирается тип открываемого файла, затем вызывается диалоговое окно открытия элемента.

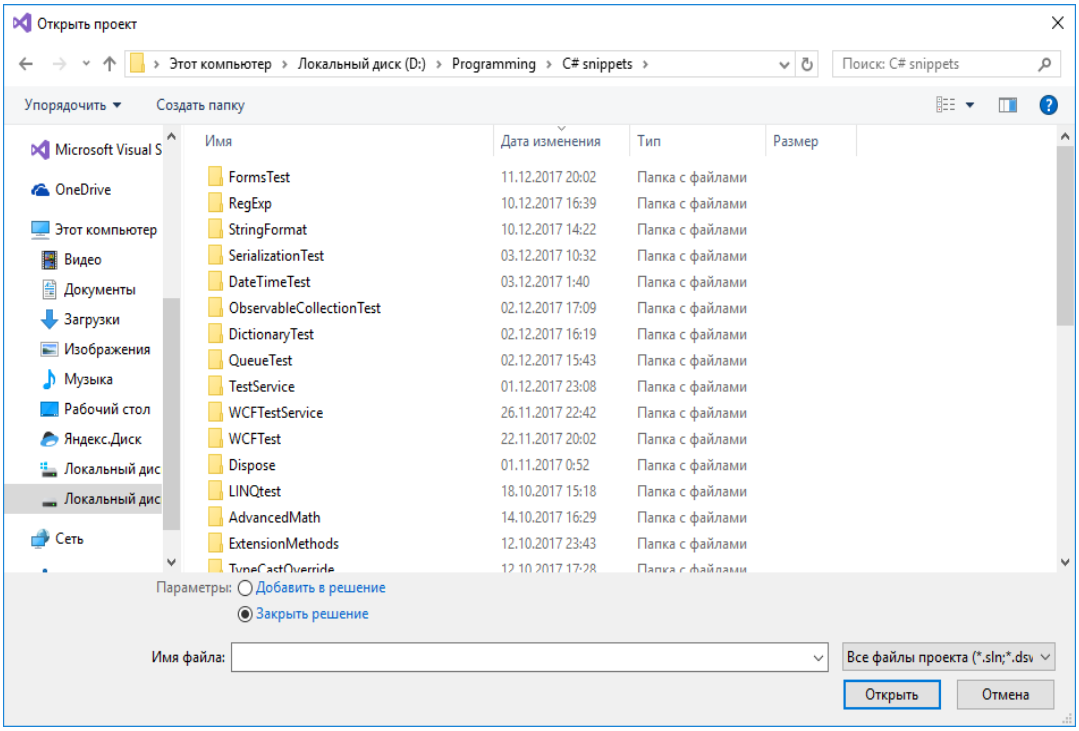

Рисунок 10 – Диалоговое окно *Открыть проект*

По умолчанию открываемые файлы — это **.sln*, **.csproj*, **.dsw*, **.vcw* для проектов или любые расширения для файлов. При открытии файлов исходного кода *(*.cs)* и проекта *(*.csproj)* их текст также будет показан в редакторе кода. Если же открываются файлы решения *(*.sln)*, то *IDE* переходит к редактированию проектов данного решения.

В случае, когда текущий активный проект не был сохранен после внесения изменений, при открытии нового проекта при помощи команды *Открыть* проект на экране появится запрос о том, следует ли сохранять произведенные изменения.

Команда Открыть проект или решение

Используется для открытия файлов проекта *(*.csproj)* или решения *(*.sln)*.

Команда Открыть файл

Используется для открытия файла в редакторе кода.

Команда Сохранить

Сохраняет активный в редакторе кода файл.

Команда Сохранить как

Сохраняет активный файл под новым именем. В результате появляется копия первоначального файла, которая заменяет его в проекте. Эту команду следует использовать в случае, если требуется включить в проект файл другого проекта, внеся в него некоторые изменения. При этом, прежде чем изменить файл, его следует сохранить его под новым именем.

Команда Сохранить все

Все относящиеся к проекту (решению) файлы сохраняются под их текущими именами.

Команда Закрыть

Закрывает активный файл. Даже если будут закрыты все файлы, составляющие проект, проект все равно остается открытым и его имя по-прежнему будет отображаться в заголовке главного окна ***MS Visual Studio***.

Команда Закрыть решение

Данная команда позволяет закрыть все файлы активного проекта. Если в этих файлах были сделаны изменения, на экране появится запрос о том, следует ли их сохранять. Отправляет пользователя на начальную страницу ***MS Visual Studio***.

Команда Выход

Согласно стандарту *CUA*, эта команда должна быть последней в меню *Файл*. С помощью команды Выход пользователь выходит из ***MS Visual Studio***. Если до этого в каком-либо файле были сделаны изменения, на экране появится запрос о том, следует ли сохранить эти изменения.

Меню Правка

- Команды меню *Правка* разделены на группы:
- поиск и замена;
- повторение или отмена команд;
- общие команды редактирования;
- дополнительные возможности;
- переход по полям и методам класса.

Команда Отменить

Последнее действие, выполненное с помощью мыши или клавиатуры в редакторе кода, либо в окне *Свойства*, может быть отменено при помощи команды *Отменить*. Следует помнить, что буфер команды *Отменить* обновляется всякий раз, когда ***MS Visual Studio*** самостоятельно создает программный код. При работе в проектировщике форм отменить можно только удаление компонента.

Команда Вернуть

При помощи команды *Вернуть* можно отменить действие последней команды *Отменить*. При этом следует помнить, что между вызовами команд *Отменить* и *Вернуть* не должны выполняться другие действия. Команда *Вернуть* действует только непосредственно после команды *Отменить*.

Команды Вырезать, Копировать, Удалить и Вставить

Эти команды редактирования работают в соответствии с *Windows*-стандартом. Особенность заключается в том, что они могут применяться для компонентов форм. В случае, если ни один из компонентов не активен, команды *Вырезать, Копировать* и *Удалить* отмечаются серым цветом и неактивны. Если "карман" *Windows* пуст, неактивна и выделена серым цветом также команда *Вставить*.

Команда Выделить все

При выполнении этой команды все компоненты активной формы будут отмечены. Это может быть полезно, например, в случае, если необходимо расположить все компоненты в форме с выравниванием, например, по левому краю. Если в данный момент активно окно редактора кода, то будет отмечен весь код активного модуля.

Команды Найти в файлах и Заменить в файлах

Команды *Найти* в файлах и *Заменить* в файлах позволяют осуществлять поиск фрагментов текста в файлах проекта и их замену на другие фрагменты. Окно *Найти* в файлах позволяет осуществить мгновенный сдвиг каретки к следующему вхождению

фрагмента текста в файлах проекта, а команда *Найти все* выведет все места вхождения данного фрагмента текста.

Окно *Заменить* в файлах позволяет заменять одни фрагменты текста на другие. Можно заменить как текущее вхождение фрагмента, найденное последней операцией *Найти* далее, так и все вхождения фрагмента операцией *Заменить* все.

Команда Перейти

Открывает окно, позволяющее перейти к файлу или элементу исходного кода по введенным первым символам его названия или к указанной строке.

Меню Вид

Меню *Вид* состоит из нескольких команд.

При помощи команд первой группы можно открывать и закрывать окна. Большое количество открытых окон сильно затрудняет работу. Поэтому оставлять открытыми на экране следует только те окна, которые необходимы при выполнении определенных задач и которые удобно постоянно "иметь под рукой".

Также данное меню имеет команды развертывания редактора кода на весь экран и меню редактирования панели инструментов.

Команды Обозреватель решений, Окно свойств и Классы

Данные команды открывают соответственно окна обозревателя решений, свойств и представления классов, которые были рассмотрены ранее.

Команда Панель элементов

С помощью этой команды можно открыть окно *Панель элементов*, которое содержит сгруппированные по разделам элементы для размещения на формах

Команда Панели инструментов

При установке курсора на команду *Панели инструментов* открывается меню, включающее в себя множество опций и одну команду *Настройка*. Указанные опции определяют, будут ли отображаться в главном окне соответствующие им кнопки панели инструментов. Команда *Настройка* открывает окно *Настроить*, в котором можно быстро выполнить установку сразу для нескольких

опций данного меню (страница *Панели инструментов*), изменить состав кнопок панели инструментов (страница *Команды*).

Меню Проект

Меню *Проект* содержит команды, предназначенные для управления текущим проектом. Команды этого меню разбиты на следующие группы:

- команды первой группы позволяют добавлять к активному проекту файлы модулей;
- команды второй группы позволяют подключать к проекту службы и зависимости;
- команда *Экспорт шаблона* позволяет создать шаблон проекта из текущего проекта;
- команда *Управление пакетами NuGet*... позволяет проводить поиск в установленных и доступных для установки библиотеках.

Команда Добавить форму Windows

Добавить к проекту новую форму *Windows Forms*.

Команда Добавить пользовательский элемент управления

Создать собственный элемент управления для размещения на формах.

Команда Добавить компонент

Добавить новый компонент к текущему проекту.

Команда Добавить класс

Добавить файл исходного кода с классом на языке C#.

Команда Добавить новый элемент

Открывает окно *Добавление нового элемента*, в котором можно выбрать шаблон элемента, добавляемого в текущий проект. Все шаблоны распределены по группам, которые отображены в левой части окна.

Команда Добавить ссылку

Используется для подключения к проекту библиотек, открывает окно *Менеджер ссылок*, в котором необходимо отметить галочкой необходимые для подключения сборки.

Команда Свойства

Открывает окно свойств проекта, которое позволяет задать для проекта опции сборки, отладки, анализа кода и др.

Меню Сборка

В данном меню присутствуют команды, позволяющие выполнить компиляцию текущего проекта или всех проектов решения, а также задать конфигурацию для компиляции проектов.

Команда Собрать решение

Выполнить сборку всех проектов в решении.

Команда Пересобрать решение

Выполнить пересборку всех проектов в решении.

Команда Построить <Project>

Выполнить сборку только текущего проекта.

Команда Перестроить <Project>

Выполнить пересборку только текущего проекта.

Команда Диспетчер конфигураций

Открывает окно *Диспетчер конфигураций*, в котором можно задать параметры сборки для каждого проекта в решении. Конфигурация определяет, с какой целью производится компиляция – с целью отладки (конфигурация *Debug*) или с целью выпуска (конфигурация *Release*). В первом случае программа будет содержать отладочные данные и не будет оптимизирована компилятором для максимального сходства с исходным кодом. Во втором случае программа не будет содержать отладочной информации и будет оптимизирована.

Меню Отладка

Команда Запуск без отладки

Данная команда компилирует, компонует и затем запускает на выполнение созданный *EXE*-файл приложения. Какие-либо изменения, проведенные в исходном коде программы, не будут отображаться при ее выполнении в данном режиме. Кроме того, невозможно выполнить сборку приложения, пока оно работает.

Команда Параметры

Данная команда открывает окно *Параметры*. В этом окне могут быть заданы параметры отладки.

Команды Шаг с обходом, Шаг с заходом

Эти четыре команды применяются при отладке. Их можно использовать, если выполнение программы происходит под контролем встроенного отладчика, т.е. если программа запушена командой *Начать отладку* меню *Отладка.*

Команды *Шаг с обходом* и *Шаг с заходом* позволяют выполнять программный код пошагово. Разница между ними в том, что по команде *Шаг с обходом* процедуры и функции выполняются как один оператор, а по команде *Шаг с заходом* трассировка будет производиться и внутри тела процедуры или функции.

С помощью команды *Шаг с обходом* можно контролировать ход выполнения программы, если есть уверенность в том, что содержащиеся в ней процедуры и функции корректны. С помощью команды *Шаг с заходом* также можно контролировать выполнение процедур и функций, но лишь в том случае, если модули, в которых они содержатся, были откомпилированы с отладочной информацией. Несколько операторов, содержащихся в строке программы, рассматриваются отладчиком как один оператор, т. е. отладить эти операторы по отдельности нельзя. В то же время отдельный оператор, размещенный на нескольких строках, воспринимается как единая строка программы.

Модули динамических библиотек и *Windows Forms* (библиотеки визуальных компонентов) откомпилированы без отладочной информации. Поэтому их нельзя отлаживать в режиме *Шаг с заходом*.

С помощью команды *Завершить отладку* можно прервать выполнение приложения в режиме отладки. Командой *Перезапустить* можно сразу же начать отладку заново, если это необходимо.

Программа в режиме отладки выполняется до точки останова, после чего автоматически ставится на паузу. Далее можно проследить пошаговое выполнение программы с помощью

вышеописанных команд *Шаг с обходом* и *Шаг с заходом*. Чтобы поставить точку останова, необходимо щелкнуть по серому полю рядом с номером той строки, в которой находится оператор, дойдя до которого программа должна быть поставлена на паузу (строка обязательно должна быть непустой).

Во время отладки в нижней части *IDE* становятся доступны две панели. Первая панель показывает состояние переменных в текущей точке останова. С ее помощью можно изменить значение переменной во время выполнения (щелчок правой кнопкой по переменной вызывает контекстное меню, в котором надо выбрать опцию *Изменить значение*). Новое значение должно быть совместимо с типом переменной. Можно изменять отдельные поля объектов и структур данных. Вторая панель состоит из нескольких вкладок, в которых отображаются стек вызовов (позволяет проследить цепочку вызовов функций от внутренней к внешней), точки останова, параметры исключений, командное окно (позволяет управлять *IDE* при помощи консольных команд), окно интерпретации и лог, в который записываются выходные данные разных составляющих *IDE* (например, данные о процессе компиляции).

Если при отладке возникает необработанное исключение, то выполнение ставится на паузу перед операцией, которая вызвала исключение. Если не добавить код обработки исключения, попытка продолжения выполнения вызовет завершение отладки.

Команда Прервать все

Команда *Прервать все* временно приостанавливает выполнение программы. Можно продолжить выполнение командой *Продолжить* или отследить пошаговое выполнение командами *Шаг с обходом* и *Шаг с заходом*.

Команда Program Reset

При необходимости можно прервать выполнение программы и освободить память. Для этого применяется команда *Program Reset.* Эта команда закрывает все открытые файлы программы и освобождает все ресурсы.

При прекращении выполнения программы может случиться так, что ресурсы *Windows* будут освобождены не полностью. В

этом случае следует выйти из ***MS Visual Studio*** и запустить ее снова.

Меню Справка

С помощью меню *Справка* можно обратиться в техническую поддержку ***MS Visual Studio***, зарегистрировать продукт или посмотреть данные о текущей лицензии и учетной записи пользователя, отправить отзыв или сообщить о проблеме, узнать о специальных комбинациях клавиатуры, а также просмотреть список установленных компонентов *IDE*.

Вставка компонентов в приложение

Компоненты являются «кирпичиками», из которых строятся приложения *Windows*. Они создают видимую пользователю часть программы. Процесс разработки пользовательского интерфейса в основном состоит из размещения компонентов в форме. Форма — это компонент, который представляет собой окно *Windows* и может содержать другие компоненты.

После размещения компонентов в форме разработчик создает лишь процедуры обработки событий, которые могут возникать при выполнении приложения. Для разработчика программы компоненты представляются графическими объектами, свойства которых становятся доступными в окне *Свойства*, как только компонент помещается в форму (в режиме разработки).

Для пользователя программы компоненты — это графические объекты на экране, с помощью которых он взаимодействует с программой.

Какие бывают компоненты

Пиктограммы компонентов, инсталлированных в ***MS Visual Studio*** по умолчанию, содержатся в панели элементов. Если поместить на изображение компонента указатель мыши, не нажимая кнопки, то через некоторое время появится всплывающая подсказка с именем и описанием компонента. Компоненты разделены на группы. Чтобы увидеть список содержащихся в группе компонентов, щелкните по названию группы.

Панель элементов

С помощью панели элементов можно разместить в форме все необходимые компоненты. Порядок действий при этом следующий:

1) Выберите группу, в которой находится нужный компонент.

2) Выберите компонент, который вы хотите разместить в форме, выполнив на нем щелчок мышью.

3) Выполните щелчок мышью в проектировщике формы в том месте, где вы хотите поместить данный компонент. Двойной щелчок на компоненте в панели элементов используется для размещения данного компонента в левом верхнем углу формы.

Контрольные вопросы к защите лабораторной работы

1. Какие этапы работы по созданию приложений можно выполнять с помощью *IDE*?
2. Для чего предназначено окно свойств?
3. Каким образом используется *Панель элементов*?
4. Для чего используется панель инструментов?
5. Каким образом перейти от редактирования формы к редактированию кода?
6. Каким образом вставляется обработчик событий в код программы?
7. Как создать новый проект?
8. Как сохранить проект?
9. Каким образом открыть существующий проект?
10. Какие основные части содержит простая программа в ***MS Visual Studio***?
11. Верно ли, что каждая форма должна иметь свой файл исходного кода?
12. Верно ли, что каждый файл исходного кода должен иметь форму?
13. Каким образом запустить проект только на компиляцию?
14. Каким образом запустить проект на компиляцию и выполнение?
15. Как используются точки останова для отладки программ?

16. Как используется окно *Состояния переменных* в процессе отладки?
17. Как использовать пошаговый режим при отладке программ?
18. Каким образом можно использовать компонент *TextBox?*
19. Каким образом можно использовать компонент *Label?*
20. Каким образом можно использовать компонент *Button?*

Индивидуальные задания

Разработать программу сложения двух чисел.

Примечание. На основе данного примера можно выполнить другие задания, отличающиеся операциями над переменными: умножение, деление, вычитание, возведение в квадрат и так далее.

1. Разместить на форме три компонента *Label*, три компонента *TextBox* и три компонента *Button* (рисунок 11).

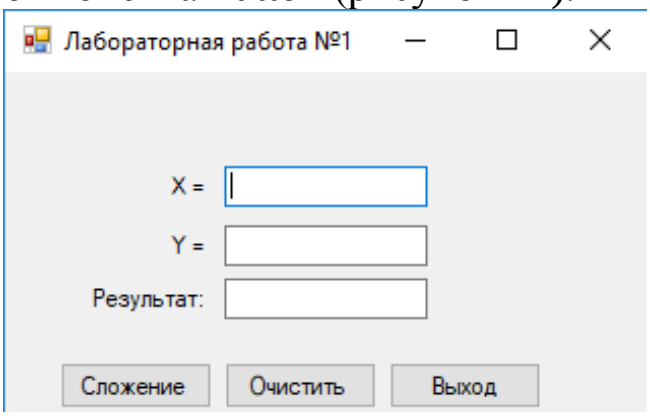

Рисунок 11 – Вид формы программы

2. Изменить свойства Text этих компонентов (таблица 4).

Таблица 4 – Используемые компоненты

Имя компонента	Настраиваемые свойства	Значение
Form1	Text	Лабораторная работа №1
TextBox1	Text	
TextBox2	Text	
TextBox3	Text	
Label1	Text	X=

Имя компонента	Настраиваемые свойства	Значение
Label2	Text	Y=
Label3	Text	Результат
Button1	Text	Сложение
Button2	Text	Отмена
Button3	Text	Выход

3. С кнопкой *Button1* связать обработчик события *Click* (выполнить двойной щелчок левой кнопкой мыши на кнопке *Button1*) и в теле обработчика события поместить следующие команды:

```
private void Button1_Click(object sender, EventArgs e)
    {
       double x, y, f;
       // Преобразование текста из TextBox1 в вещественное число.
       x = double.Parse(TextBox1.Text);
       y = double.Parse(TextBox2.Text);
       // Вычисление результата.
       f = x + y;
// Преобразование числа f в строку и вывод результата в TextBox3
       TextBox3.Text = f.ToString();
    }
```

4. С кнопкой *Button2* связать обработчик события *Click* (выполнить двойной щелчок левой кнопкой мыши на кнопке *Button2*) и в теле обработчика события поместить следующие команды:

```
private void Button2_Click(object sender, EventArgs e)
    {
       // Очищаем textBox1.
       TextBox1.Text = "";
       // Очищаем textBox2.
       TextBox2.Text = "";
       // Очищаем textBox3.
       TextBox3.Text = "";
       }
    }
```

5. С кнопкой *Button3* связать обработчик события *Click* (выполнить двойной щелчок левой кнопкой мыши на кнопке *Button3*) и в теле обработчика события поместить команду *Close()*.

```
private void Button3_Click(object sender, EventArgs e)
    {
      Close();
    }
```

6. Сохранить проект.

7. Откомпилировать проект.

8. Запустить проект на выполнение. Ввести исходные данные в *TextBox1* и *TextBox2*. Убедиться, что при щелчке левой кнопкой мыши на кнопке *Button1* в *TextBox3* выводится результат вычислений.

9. Убедиться, что при щелчке левой кнопкой мыши на кнопке *Button2* поля *TextBox* очищаются.

10. Убедиться, что при щелчке левой кнопкой мыши на кнопке Button3 проект закрывается.

11. Оформить отчет.

12. Подготовиться по контрольным вопросам к защите работы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Подбельский В.В. Язык C#. Базовый курс. [Текст]/ В.В. Подбельский. – М.: Финансы и статистика, 2013. – 427 с.

2. Троелсен, Эндрю. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е изд. [Текст]/ Эндрю Троелсен – М.: ООО «И.Д. Вильямс», 2013. – 1312 с.

3. Макконнелл, Стив. Совершенный код. Мастер-класс. [Текст]/ Пер. с англ. – М. : Издательство «Русская редакция», 2010. – 896 с.

4. Техническая документация, материалы по API и примеры кода [Электронный ресурс]// Режим доступа – https://docs.microsoft.com (дата обращения: 26.05.2019).

5. METANIT.COM – Сайт о программировании [Электронный ресурс]// Режим доступа – https://metanit.com (дата обращения: 6.06.2019).

6. MSDN – сеть разработчиков Microsoft [Электронный ресурс]// Режим доступа – https://msdn.microsoft.com (дата обращения: 27.05.2019).

7. Шилдт, Г. C# 4.0: полное руководство. Пер. с англ. [Текст]/ Герберт Шилдт. – М.: ООО "И.Д. Вильяме", 2011. - 1056 с.

8. Википедия – свободная энциклопедия [Электронный ресурс]// Режим доступа – https://ru.wikipedia.org/wiki/Заглавная_страница (дата обращения: 24.05.2019).