МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ»

Г.Э. Вощинская, Е.М. Лещенко

СТРУКТУРЫ И АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ

Часть 1

Учебно-методическое пособие

Воронеж
Издательский дом ВГУ
2019

Содержание

ŋqddɣeɎ bɆ dəqɆbɎɜe d ɛɔɑbɔɤ ɈqddɎqɤè

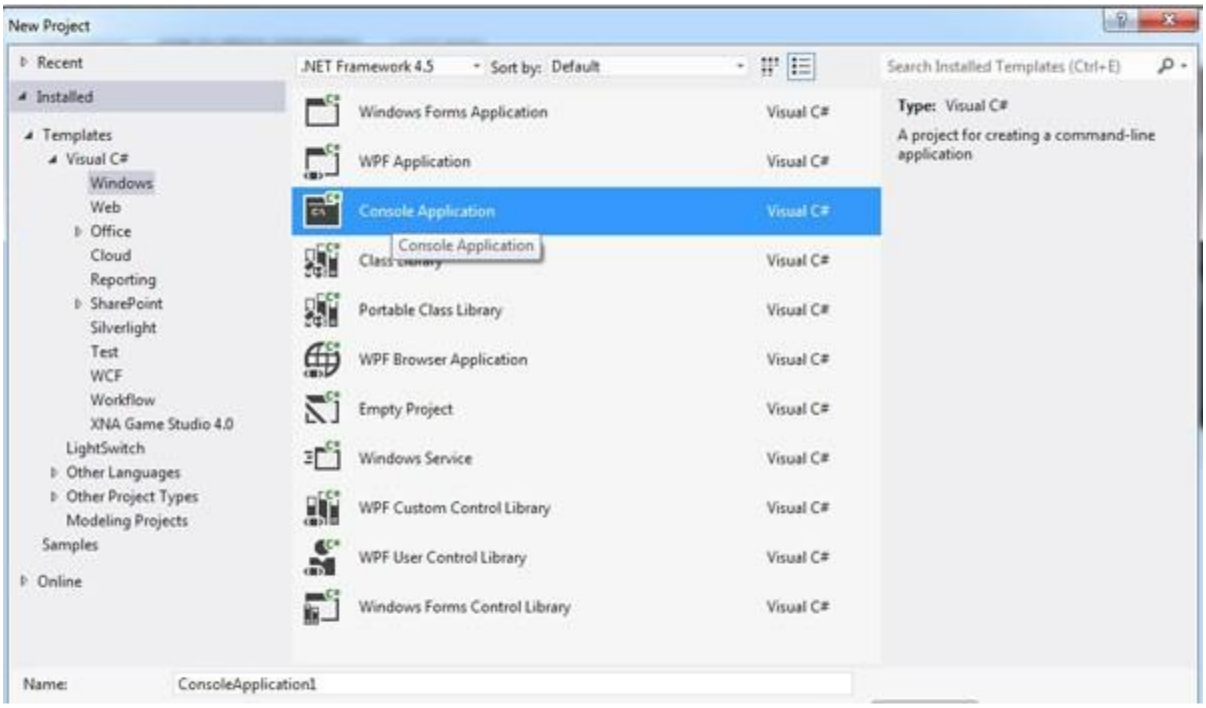

Рис. 3. Окно для создания консольного приложения.

Напишем первые строки на языке C# и запустим программу на выполнение, нажав на кнопку **"Start"** (или клавишу F5):

1 Console.WriteLine("Hello World!!!");

2 Console.ReadLine();

Если у вас всё получилось правильно, будет выведено сообщение **"Hello World"**, и код должен выглядеть как на рисунке 4.

Кратко разберем основные моменты. Ключевое слово **using** используется для подключения библиотек к нашей программе. Библиотек существует большое количество. Если мы захотим работать с файлами, то необходимо будет подключать одну библиотеку, если с базами данных — другую.

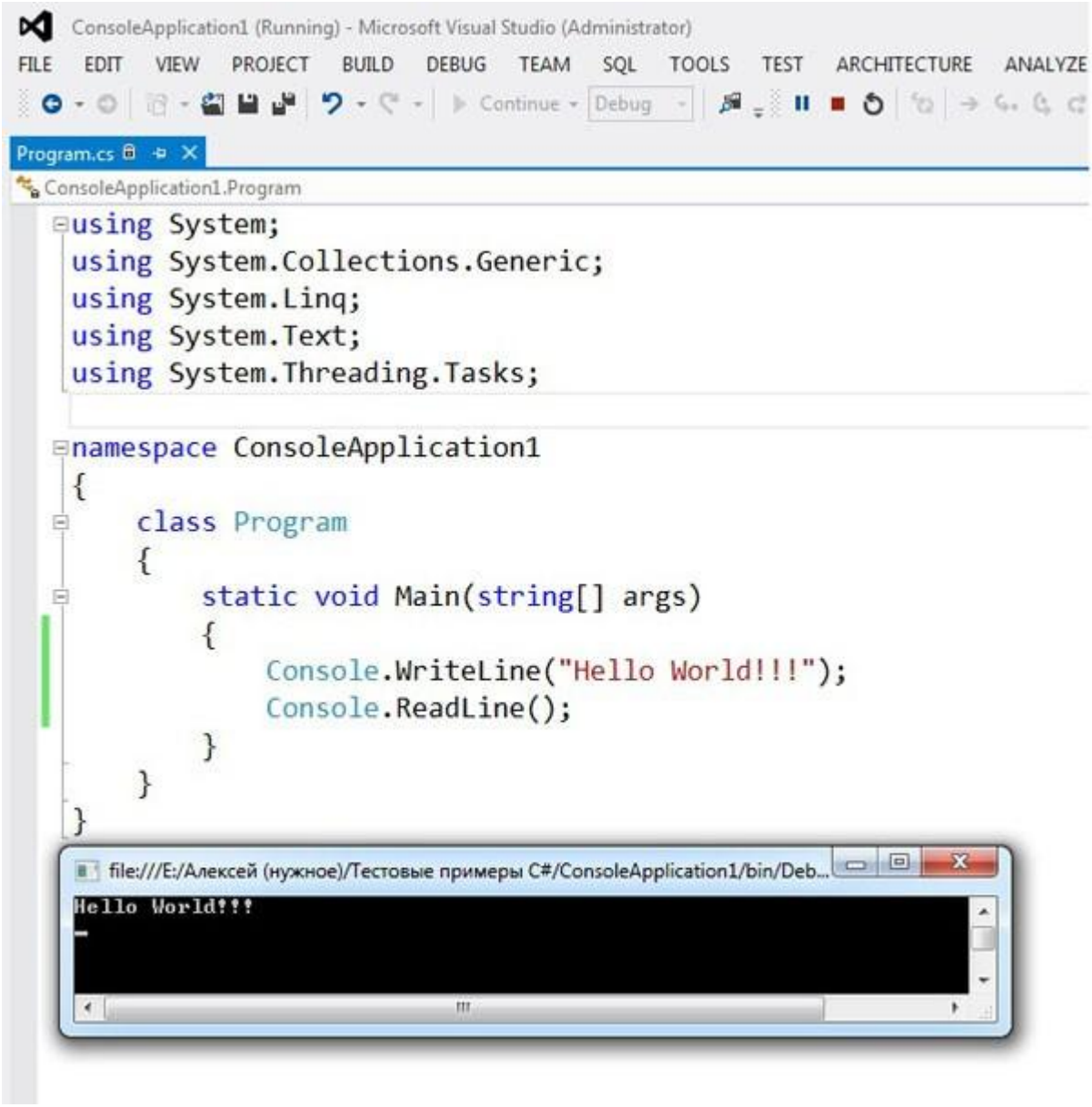

Рис.4. Пример программы.

namespace — это пространство имен. Обычно оно совпадает с названием приложения, хотя можно задать любое. По умолчанию среда Visual Studio назначает классу, определяющему метод **Main()** имя **Program**, хотя данное имя можно поменять. Метод **Main()** используется для обозначения точки входа в программу. Обратите внимание на ключевое слово **static** перед методом **Main().** Это статический член, область действия которого охватывает уровень класса в целом, а не уровень объектов. Ключевое слово

void() означает, что возвращаемого значения не будет (т.е. в конце не будет стоять **return**).

Для вывода сообщения на экран используется класс **Console** (пространство имен System) со статическим методом **WriteLine()**, с помощью которого можно вывести строку текста на экран. Метод **Console.ReadLine()** нужен для того, чтобы мы видели результат работы программы до тех пор, пока не нажата клавиша "Enter".

С помощью специального окна Properties (Свойства) справа Visual Studio предоставляет нам удобный интерфейс для управления свойствами элемента.

Большинство этих свойств оказывает влияние на визуальное отображение формы. Рассмотрим некоторые из них:

- **Name**: устанавливает имя формы — точнее имя класса, который наследуется от класса Form.
- **BackColor**: указывает на фоновый цвет формы. Щелкнув на это свойство, можно выбрать тот цвет из списка предложенных цветов или цветовой палитры.
- **BackgroundImage**: указывает на фоновое изображение формы.
- **BackgroundImageLayout**: определяет, как изображение, заданное в свойстве BackgroundImage, будет располагаться на форме.
- **ControlBox**: указывает, отображается ли меню формы. В данном случае под меню понимается меню самого верхнего уровня, где находятся иконка приложения, заголовок формы, а также кнопки минимизации формы и крестик. Если данное свойство имеет значение false, то не будет видна ни иконка, ни крестик, с помощью которого обычно закрывается форма.

- **Cursor**: определяет тип курсора, который используется на форме.
- **Enabled**: если данное свойство имеет значение false, то форма не сможет получать ввод от пользователя, то есть нельзя будет нажать на кнопки, ввести текст в текстовые поля и т.д.
- **Font**: задает шрифт для всей формы и всех помещенных на нее элементов управления. Однако, задав у элементов формы свой шрифт, можно переопределить его.
- **ForeColor**: цвет шрифта на форме.
- **FormBorderStyle**: указывает, как будет отображаться граница формы и строка заголовка. Устанавливая данное свойство в None можно создавать внешний вид приложения произвольной формы.
- **HelpButton**: указывает, отображается ли кнопка справки формы.
- **Icon**: задает иконку формы.
- **Location**: определяет исходное положение формы по отношению к верхнему левому углу экрана, если для свойства StartPosition установлено значение Manual.
- **MaximizeBox**: указывает, будет ли доступна кнопка максимизации окна в заголовке формы.
- **MinimizeBox**: указывает, будет ли доступна кнопка минимизации окна.
- **MaximumSize**: задает максимальный размер формы.
- **MinimumSize**: задает минимальный размер формы.
- **Opacity**: задает прозрачность формы.

ŋqdqɏeɎ bɆ dədɆbɎɜə d ɕɔɑbɔɏ Ɉqddɏqɏè

- **Size**: определяет начальный размер формы.
- **StartPosition**: указывает на начальную позицию, с которой форма появляется на экране.
- **Text**: определяет заголовок формы.
- **TopMost**: если данное свойство имеет значение true, то форма всегда будет находиться поверх других окон.
- **Visible**: видима ли форма. Если нужно скрыть форму от пользователя, то данное свойство устанавливается в false.
- **WindowState**: указывает, в каком состоянии форма будет находиться при запуске: в нормальном, максимизированном или минимизированном.

Программная настройка свойств

С помощью значений свойств в окне Свойства можно изменить по своему усмотрению внешний вид формы, но все то же самое можно сделать динамически в коде. Перейдем к коду, для этого нажмем правой кнопкой мыши на форме и выберем в появившемся контекстном меню View Code (Просмотр кода). В результате открывается файл кода *Form1.cs*. Изменим его следующим образом:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```