СОЦИАЛЬНО-ГУМАНИТАРНЫЙ КОЛЛЕДЖ
УО «МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ А.А. КУЛЕШОВА»

Дисциплина
«Конструирование программ и языки программирования»

Создание библиотеки

(2 часа)

Методические рекомендации к лабораторной работе №16

Могилев 2018

Составитель: Шлапаков А.В.

Понятия «Dynamic Link Library (DLL)», «XML-комментарий». Методические указания по лабораторной работе №16 «Конструирование программ и языки программирования». Для учащихся 3 курса очной формы обучения специальности 2–40 01 01 «Программное обеспечение информационных технологий.

Составитель: Шлапаков А.В.

Оглавление

Составитель: Шлапаков А.В.

1 Цель работы

научиться создавать библиотеки .dll..

2 Ход работы

1. Изучение теоретического материала.
2. Выполнение практических индивидуальных заданий по вариантам (вариант уточняйте у преподавателя).
3. Оформление отчета.
   3.1. Отчет оформляется индивидуально каждым студентом. Отчет должен содержать задание, алгоритм и листинг программы.
   3.2. Отчет по лабораторной работе выполняется на листах формата А4. В состав отчета входят:
      1) титульный лист;
      2) цель работы;
      3) текст индивидуального задания;
      4) выполнение индивидуального задания.
4. Контрольные вопросы.

3 Краткие теоретические сведения

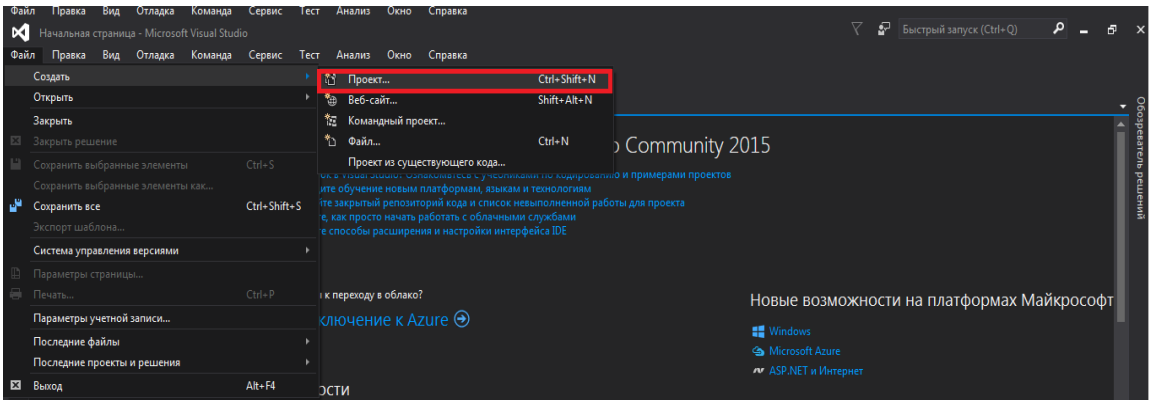
3.1 Что такое Dynamic Link Library (DLL)?

DLL (англ. Dynamic Link Library – «динамически подключаемая библиотека») – это библиотека, содержащая код и данные, которые могут использоваться несколько программами одновременно. Например, в операционных системах Windows, библиотека **Comdlg32.dll** выполняет общие функции, связанные с диалоговыми окнами. Таким образом, каждая программа может использовать функцию, которая содержится в этой библиотеке для реализации диалогового окна Открыть. Это позволяет повысить уровень повторного использования кода и эффективного использования памяти.

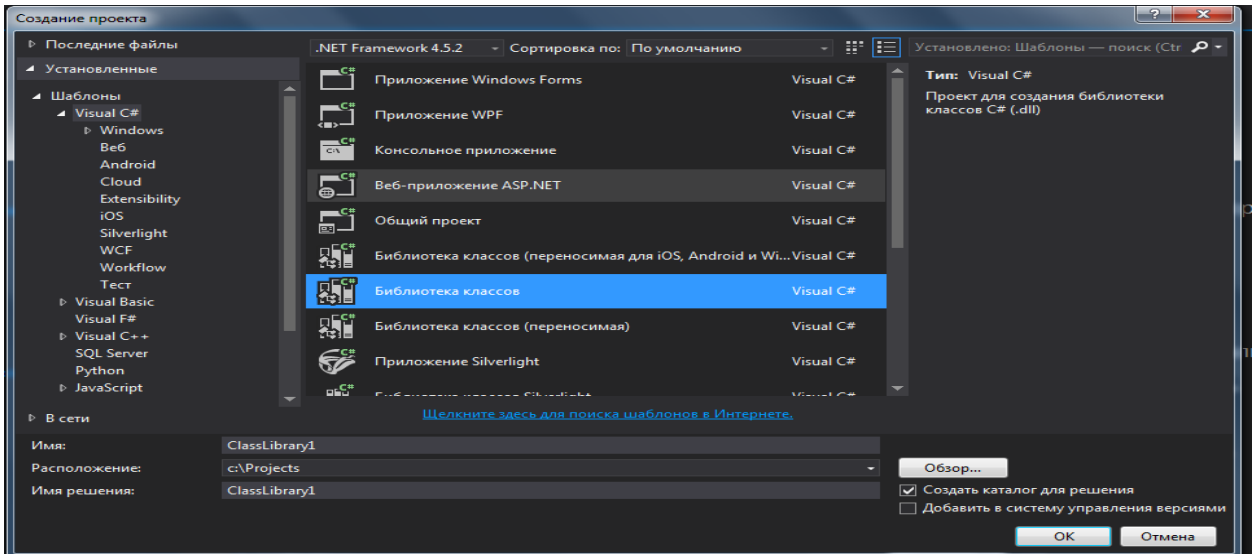
С помощью библиотек можно реализовать модульность для программы, в виде отдельных компонентов. Например, бухгалтерскую программу можно продать по модулям. Каждый модуль может быть загружен в основной программе во время выполнения установки. Отдельные модули загружается только при запросе функций заложенных в них, поэтому загрузка программы выполняется быстрее.

Кроме того обновления легче применить для каждого модуля, не влияя на другие части программы. Например, имеется программа по зарплате и надо изменить налоговые ставки за каждый год. Когда эти изменения изолированы в библиотеке, можно применить обновления без необходимости построения или установки программы целиком. Давайте рассмотрим пример создания библиотеки с самыми простыми математическими методами, такие как произведение, деление, сумма и разность.

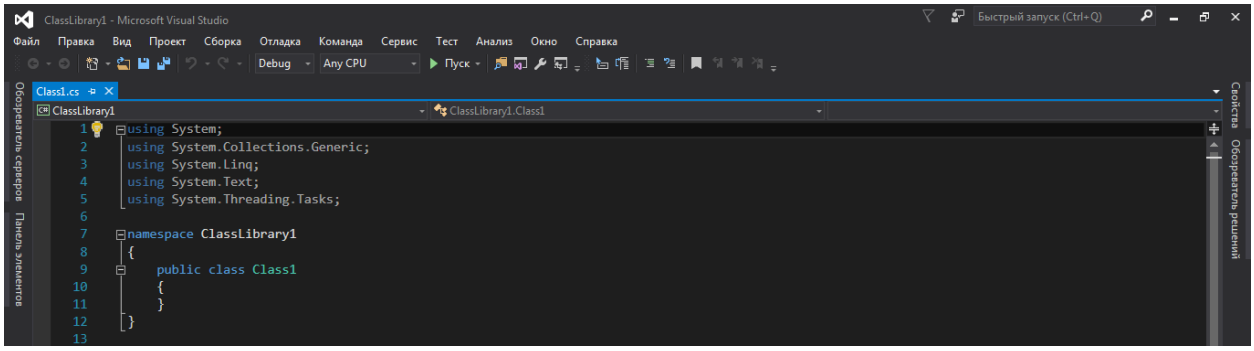
Для начала, создадим новый проект, для этого запустите **`Microsoft Visual Studio`** и перейдите в меню **`Файл -> Создать -> Проект`**... или выполните сочетание клавиш **`Ctrl+Shift+N.`**



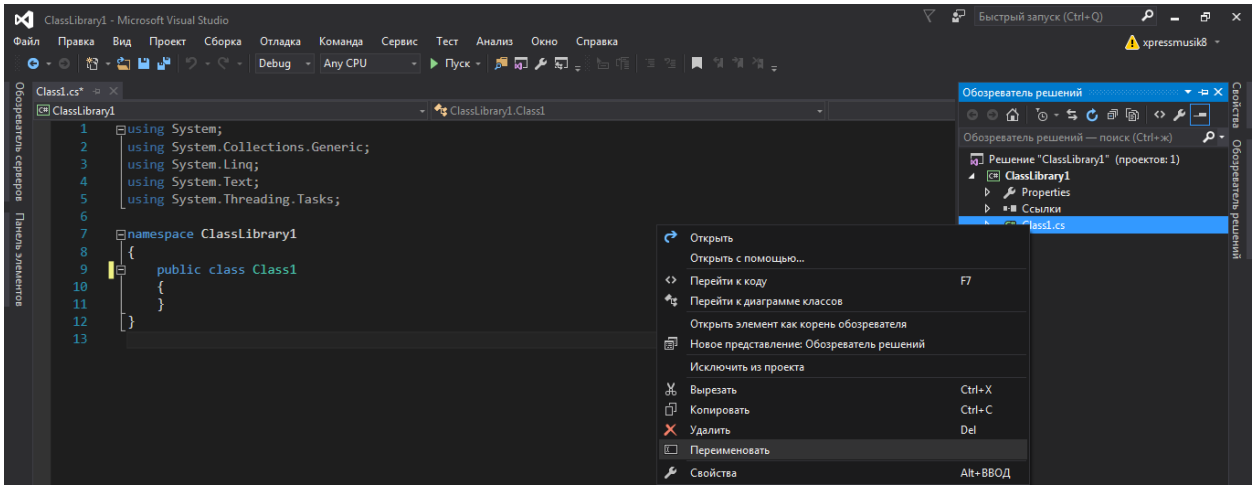
В открывшемся диалоговом окне выберете версию Framework, а в левой части **«`Установленные шаблоны`»** выберете **«`Visual C#`»**, в центральной части вам будет представлен список шаблонов, выберете **«`Библиотека классов`»** и введите имя библиотеки, можно оставить по умолчанию.



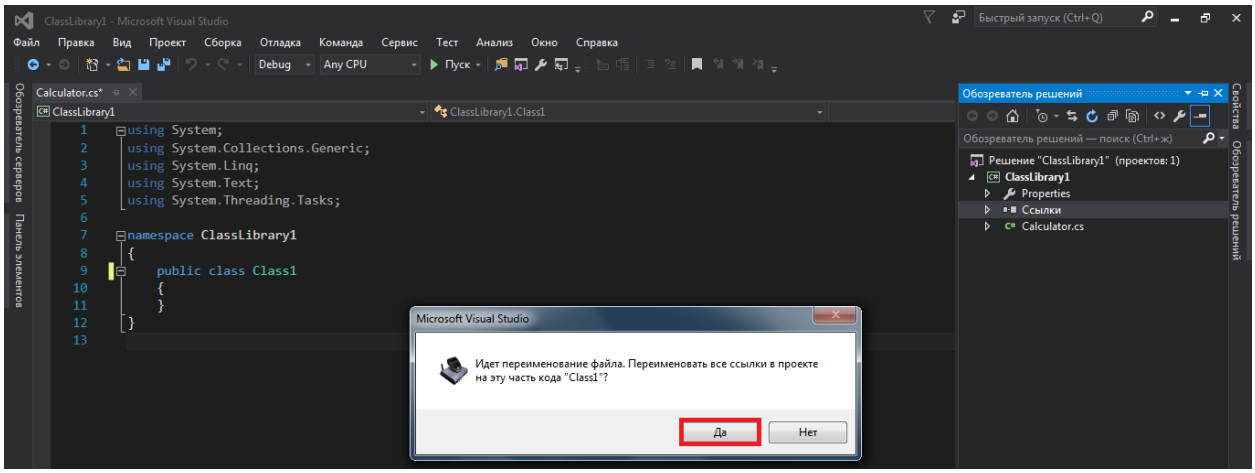
У вас откроется вкладка с классом по умолчанию.



По умолчанию создается класс **`Class1`**, переименуем его в класс **`Calculator`**. Для этого перейдите в **`Обозреватель`** решений или выполните сочетание клавиш **`Ctrl + Alt+L`**. Выберете по умолчанию созданный класс Class1, сделайте клик правой клавишей мыши по нему и выберете «**Переименовать…**».



Обратите внимание, что данное окно позволяет переименовать класс во всем проекте.

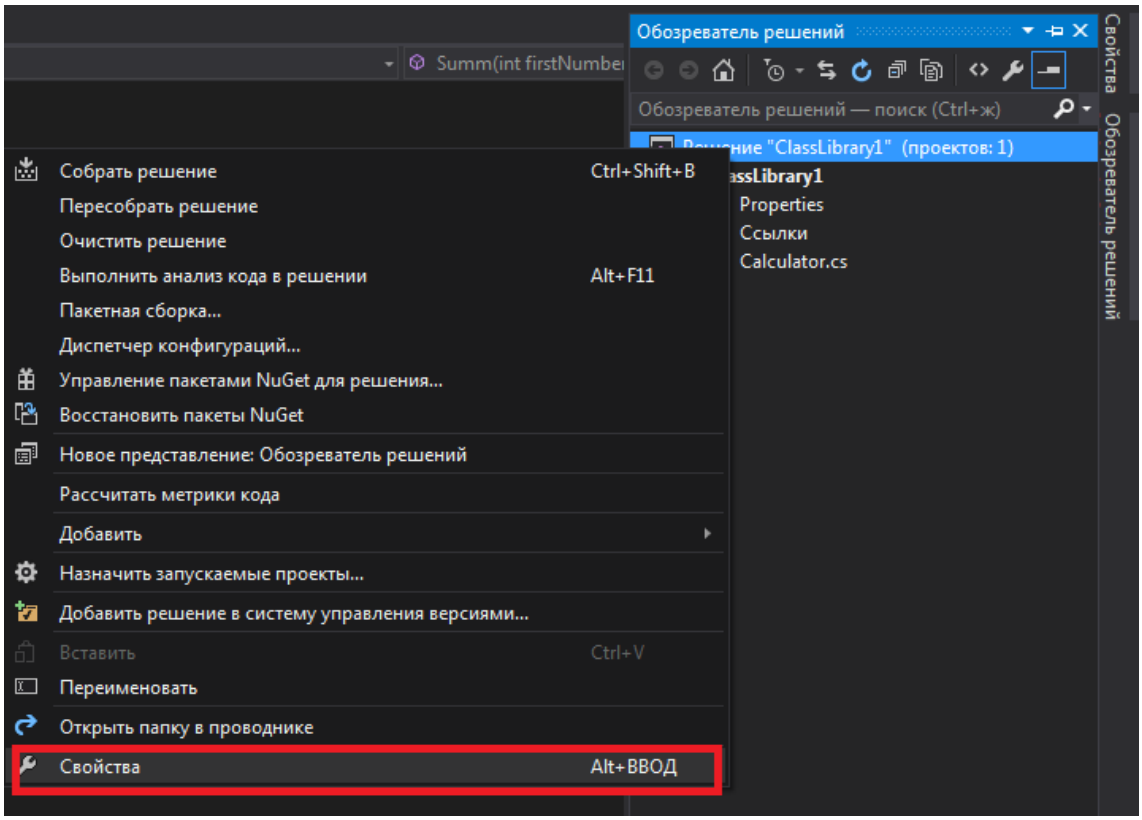


Добавим в класс **Calculator** несколько методов и добавим к ним описание.

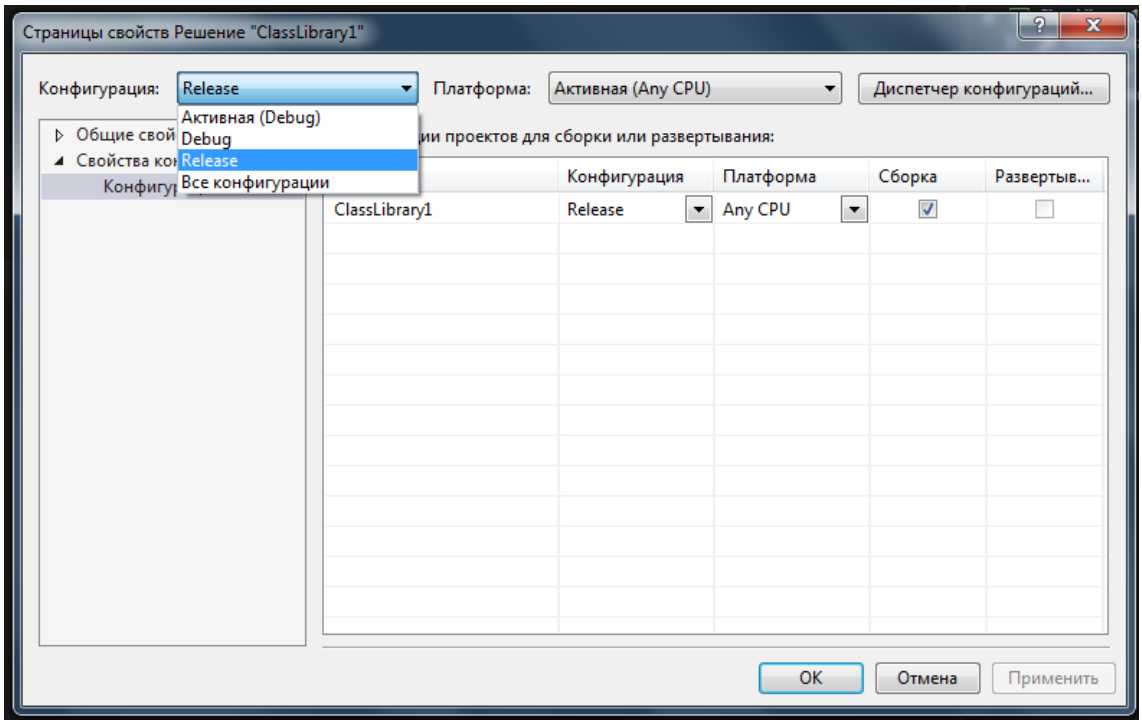
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ClassLibrary1
{
    /// <summary>
    /// Математический класс
    /// </summary>
    public class Calculator
    {
        /// <summary>
        /// Метод возвращает сумму двух целых чисел
        /// </summary>
        /// <param name="firstNumber"></param>/// <param name="secondNumber"></param>/// <returns></returns>
        public static int Summ(int firstNumber, int secondNumber)
        {
            return firstNumber + secondNumber;
        }
        /// <summary>
        /// Метод возвращает разность двух целых чисел
        /// </summary>
        /// <param name="firstNumber"></param>/// <param name="secondNumber"></param>/// <returns></returns>
        public static int Division(int firstNumber, int secondNumber)
        {
            return firstNumber - secondNumber;
        }
        /// <summary>
        /// Метод возвращает произведение двух чисел
        /// </summary>
        /// <param name="x"></param>/// <param name="y">
```

```
</param>/// <returns></returns>
            public static long Multiply(long x, long y)
            {
                return (x * y);
            }
            /// <summary>
            /// Метод возвращает деление двух чисел
            /// </summary>
            ///  <param name="firstNumber"> </param>///  <param
name="secondNumber"> </param>/// <returns></returns>
            public  static  int  Residual(int  firstNumber,  int
secondNumber)
            {
                return (firstNumber / secondNumber);
            }
        }
    }
```

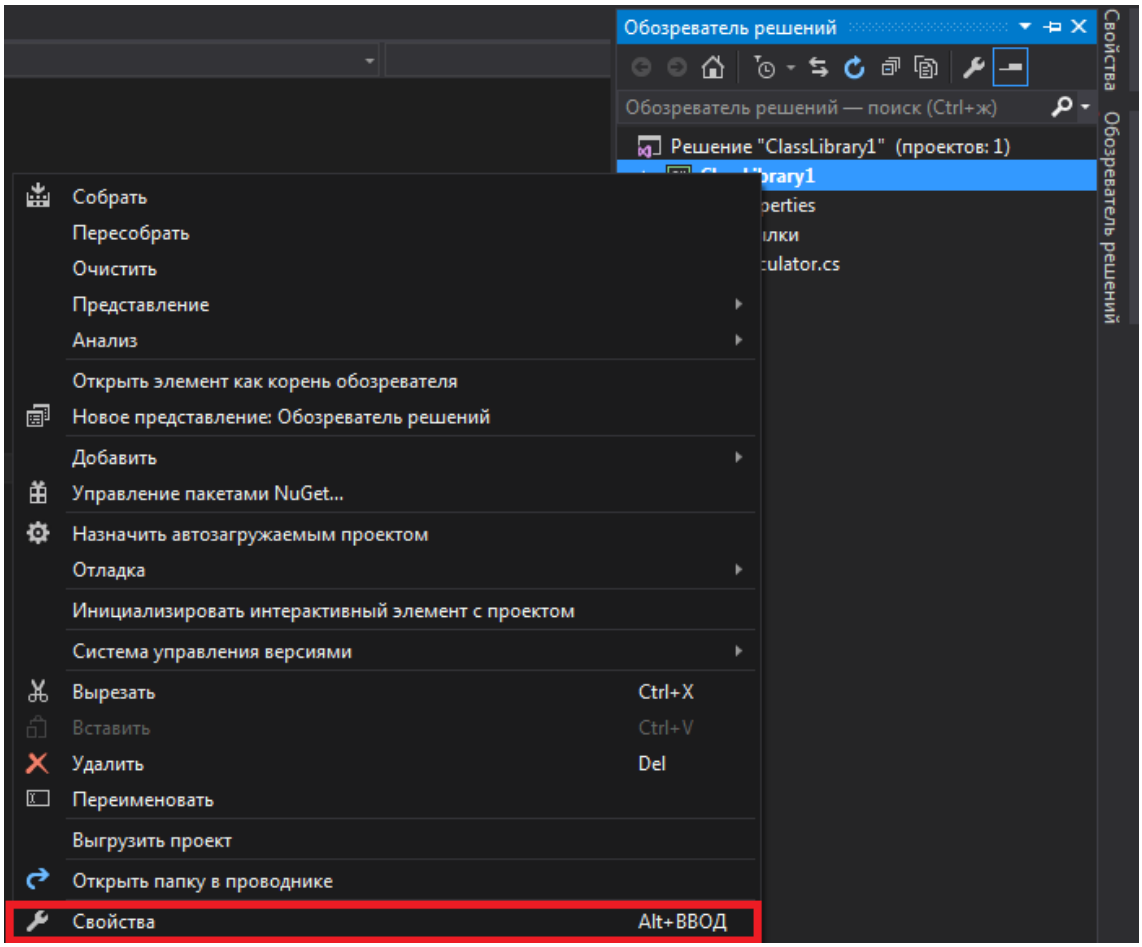
По умолчанию для всех проектов стоит режим построения **Debug**(режим отладки), переведем проект в режим построения конечной версии (**Release**). Для этого перейдите в обозреватель решений и, сделав клик правой клавишей мыши по названию проекта, выберете в открывшемся контекстном меню пункт «**Свойства**».



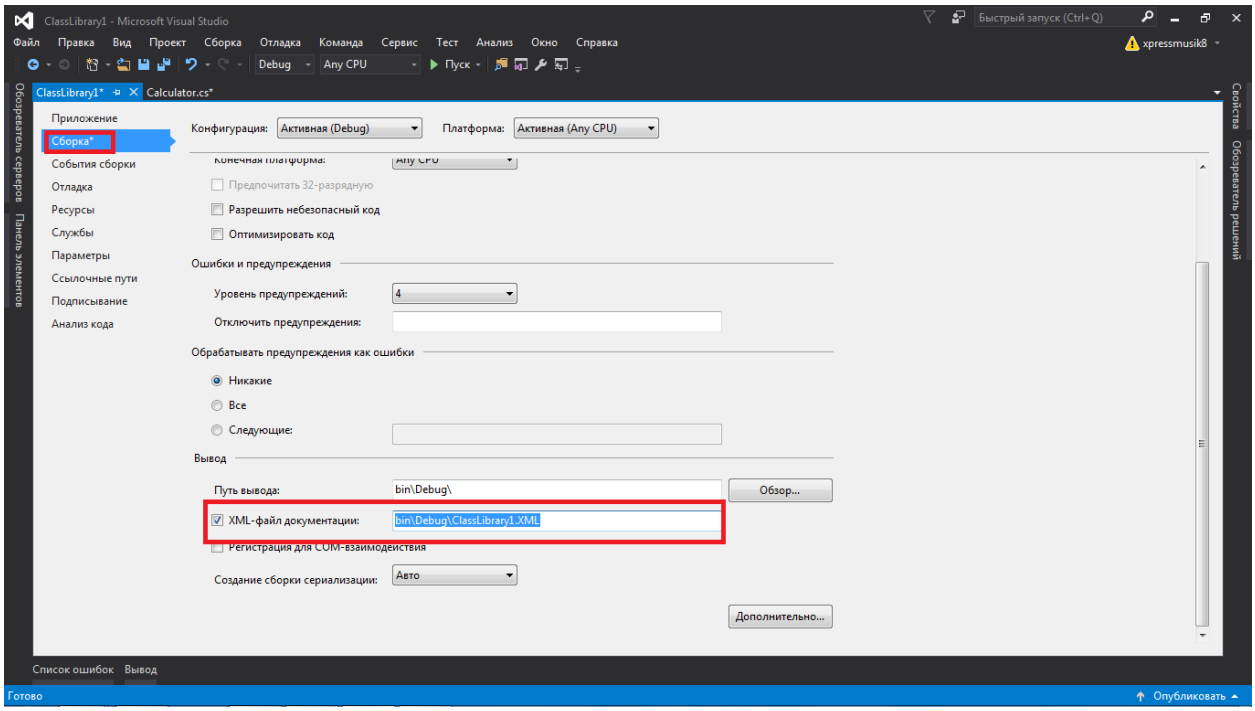
В открывшемся окне «Страницы свойств Решение "**ClassLibrary1**"» выберете везде конфигурацию Release, как показано на скриншоте ниже.



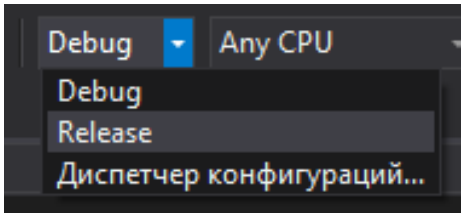
Наверно все замечали, что при наборе кода, появляется подсказка к методам или функциям. Мы задали такую подсказку в тегах. Но если сейчас просто выполнить построение библиотеки, то при подключении к другим проектам никаких подсказок видно не будет. Что бы устранить данную проблему, нам необходимо сформировать XML файл документации к проекту. Для этого в обозревателе решений выполните клик правой клавишей мыши по названию библиотеки и в открывшемся контекстном меню выберете пункт «**Свойства**».



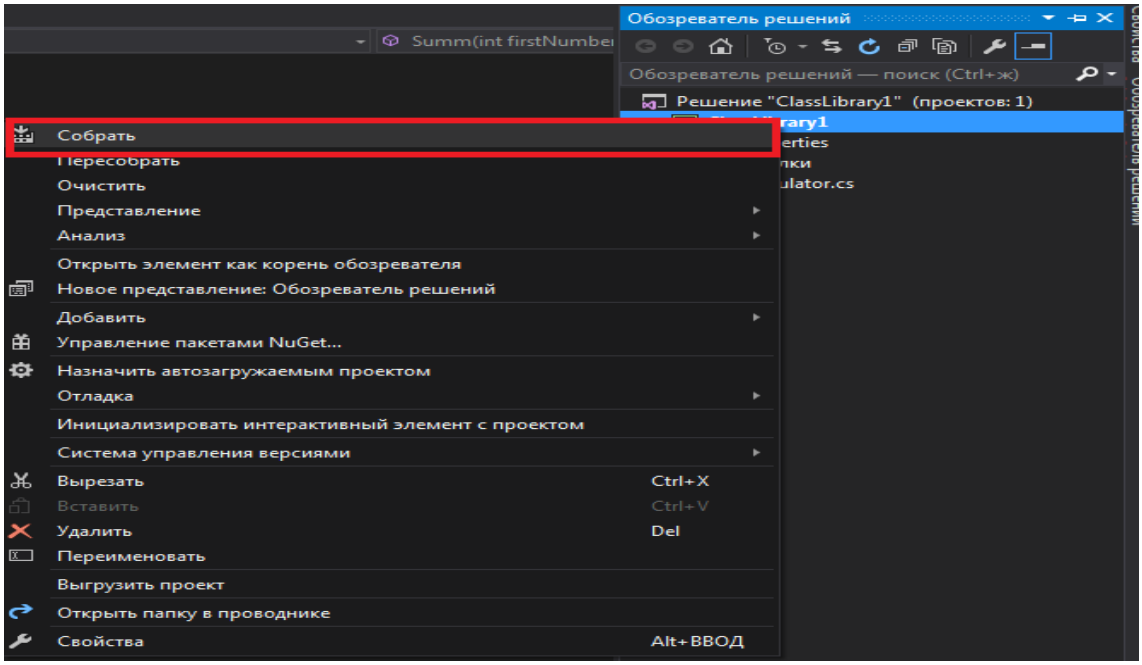
После этих действий у вас откроется новая вкладка со свойствами проекта **`ClassLibrary1`**. Выберете в вкладку **«Построение»** и найдите раздел **«Вывод»,** там вам будет предложено ввести путь куда будет выполнено построение конечной версии библиотеки и пункт необходимый нам для построения xml файла документации, тут вам необходимо просто поставить галочку как показано на скриншоте ниже. Тут важно чтобы библиотека и файл документации находились в одном месте, поэтому проверьте чтобы их путь вывода совпадал.



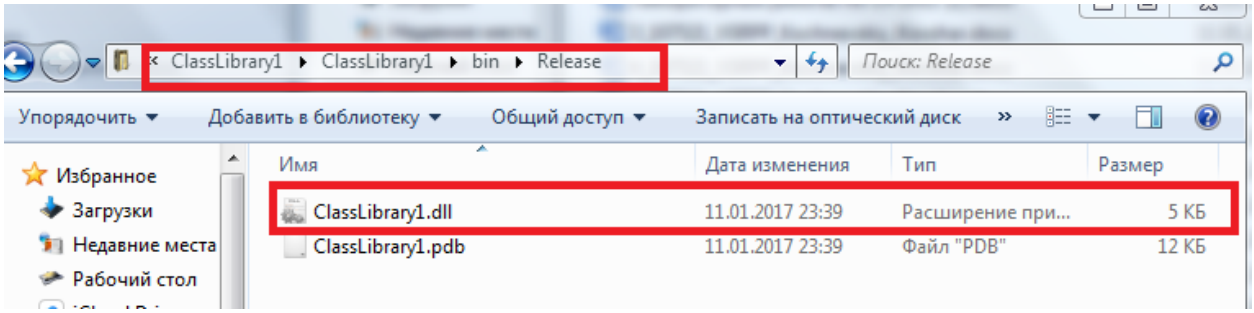
Остались последние шаги, и мы получим готовую для использования библиотеку. И так продолжим, вам необходимо в верхней части программы выбрать режим конфигурации **Release** как показано на скриншоте ниже.



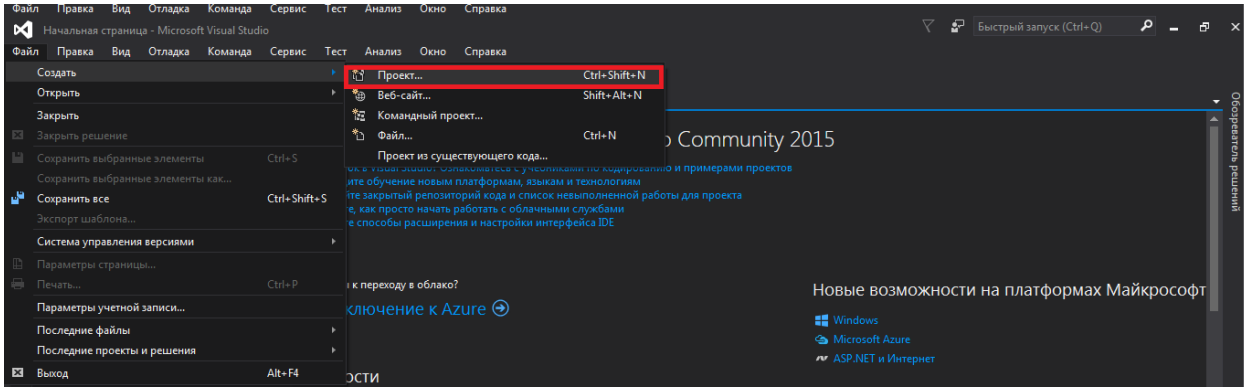
После этого, выполним построение решения. Нажав на клавиатуре клавишу **F6.**



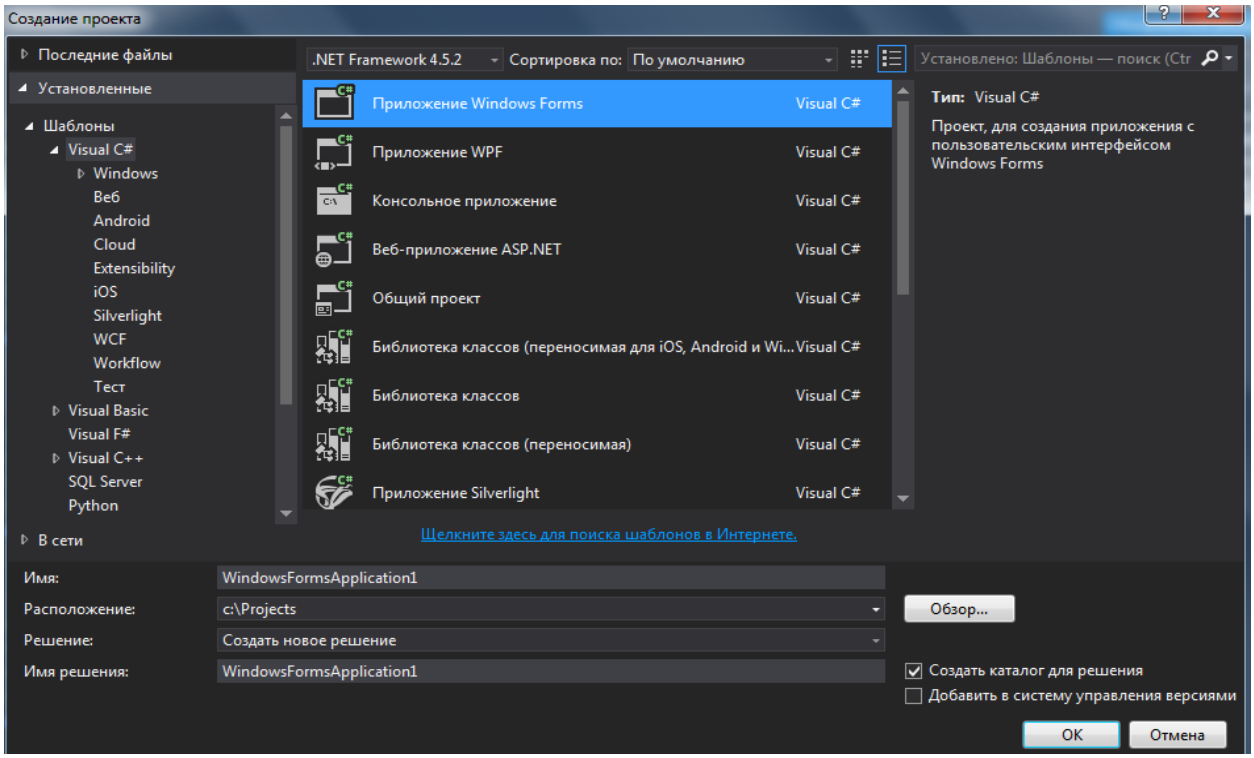
Как только программа закончит построение, можно перейти в директорию с проектом и посмотреть что получилось. На этом этапе закончилось создание библиотеки.



Для проверки работоспособности библиотеки создадим тестовый проект. Выполните **Файл -> Создать -> Проект…**

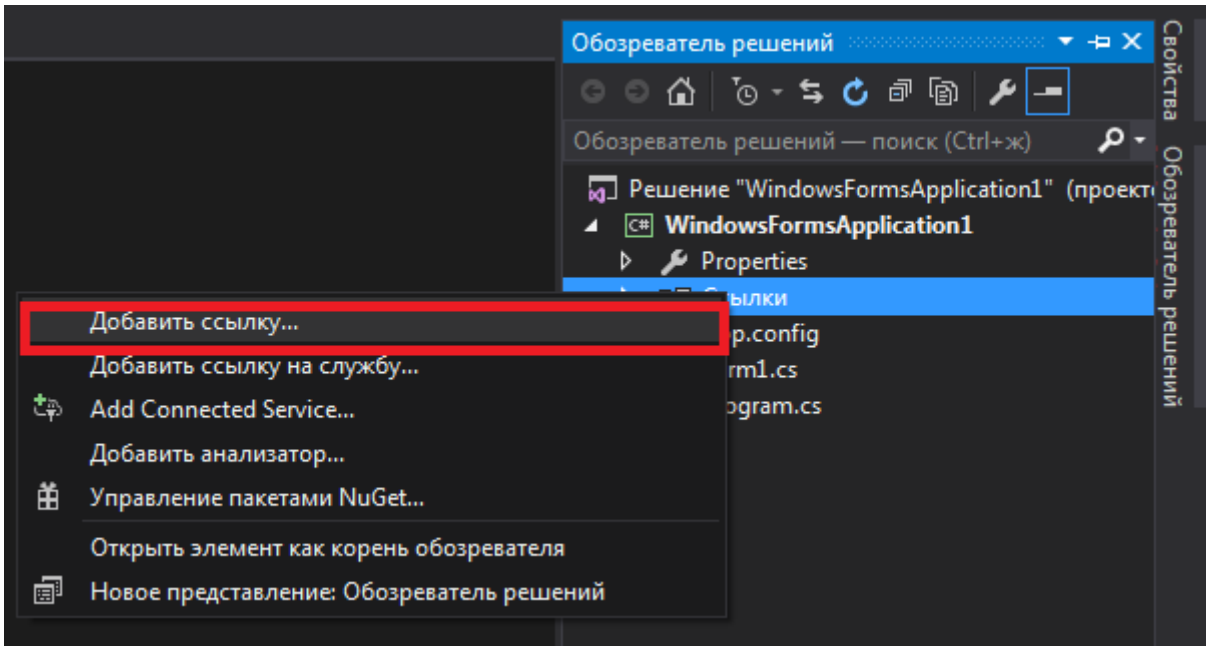


Выберете из предложенных шаблонов, шаблон «**`Приложение Windows Forms Visual C#`**». Задайте имя проекта и нажмите кнопку **`OK`**.

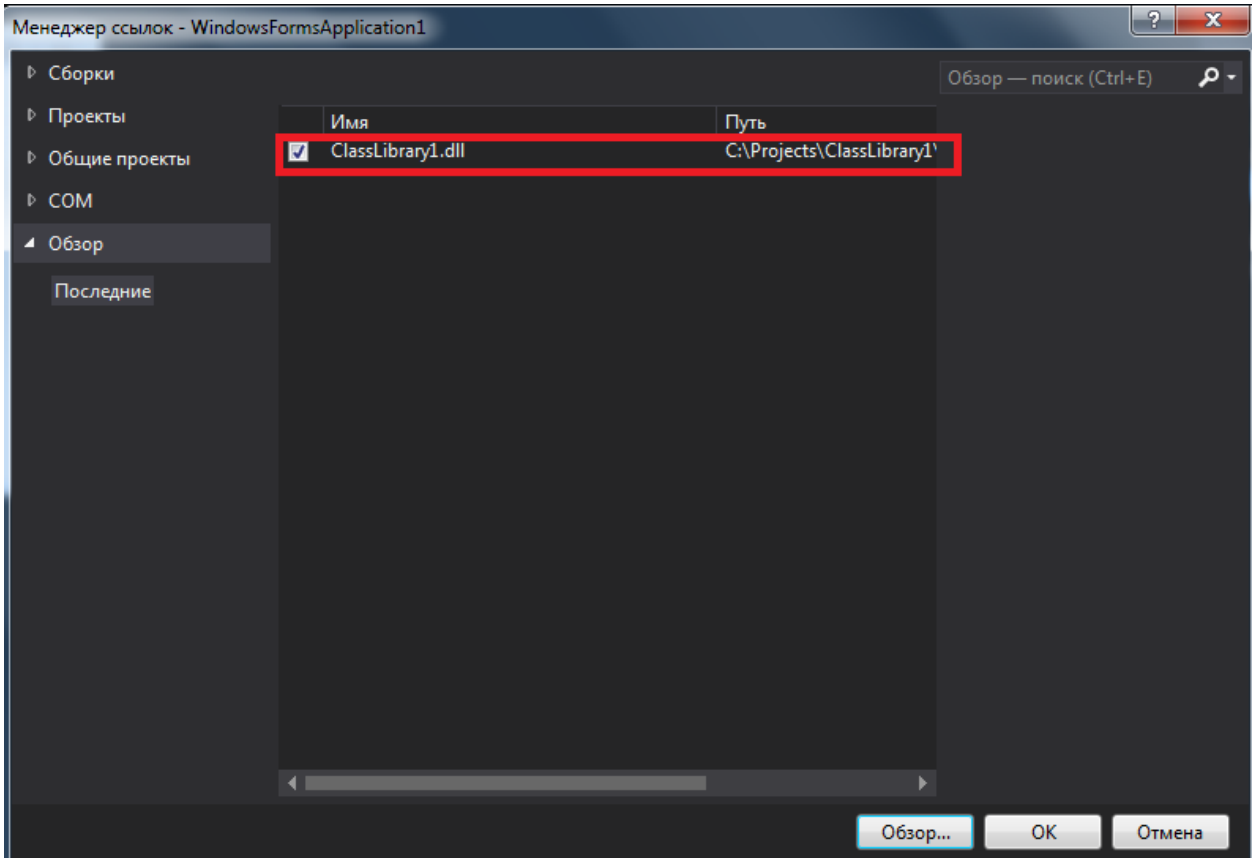


После создания проекта, в обозревателе решений сделайте клик правой клавишей

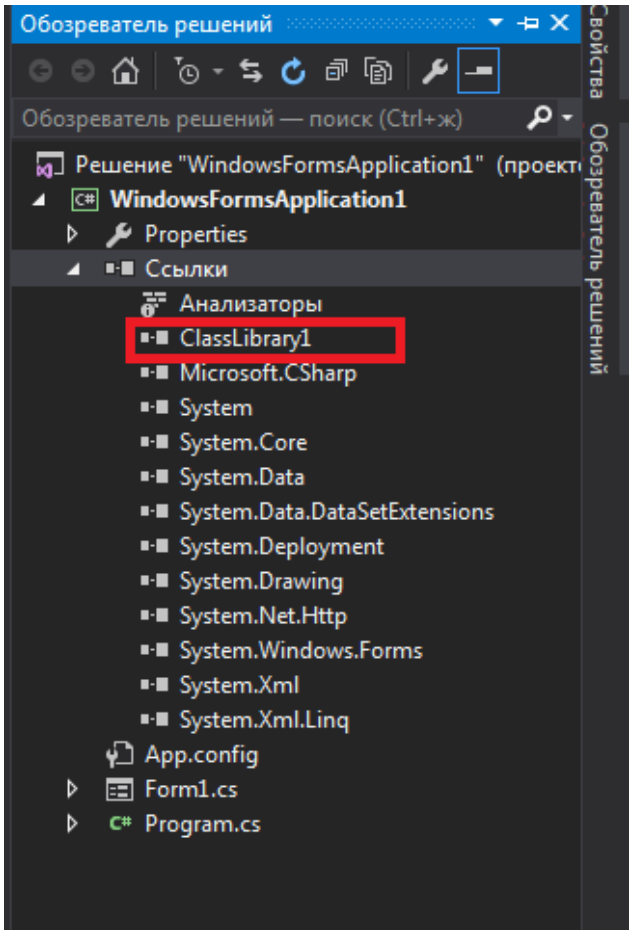
мыши по разделу «**Ссылки**» и выберете в появившемся контекстном меню пункт «**Добавить ссылку**...».



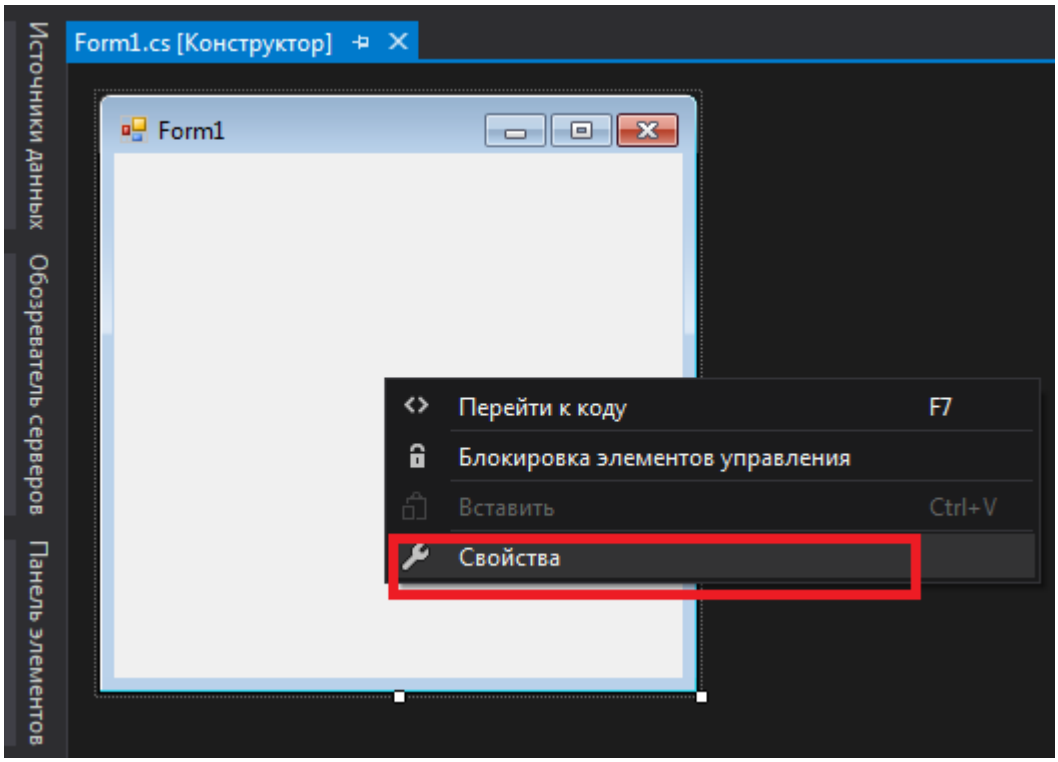
Выберете вкладку «**Обзор**» и укажите вашу библиотеку.



Если вы все успешно выполнили, в разделе «**Ссылки**» у вас появится название вашей библиотеки.

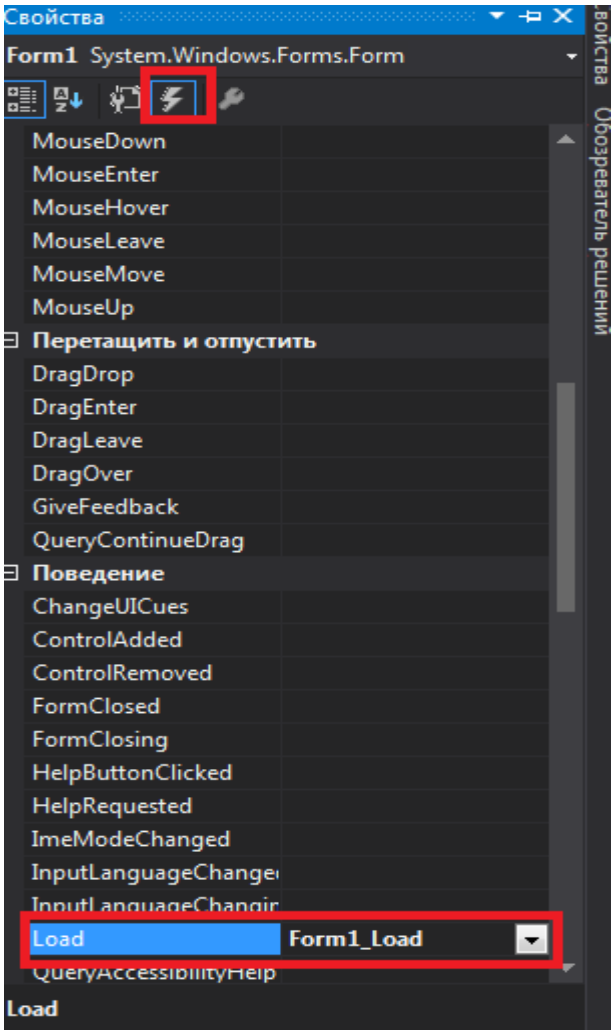


Сделайте клик правой клавишей мыши по главной форме вашего проекта и в открывшемся контекстном меню выберете пункт «**Свойства**».

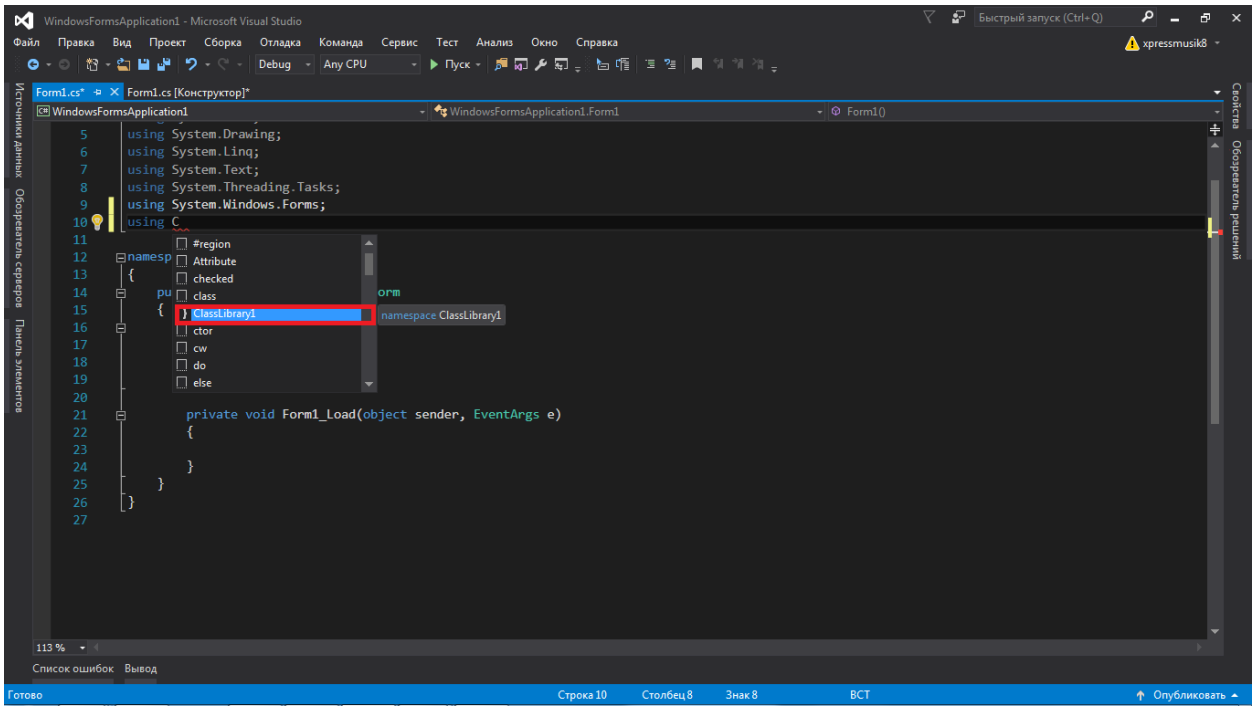


В боковой панели откроются «**Свойства**» формы. Найдите **метод Load** и сде-

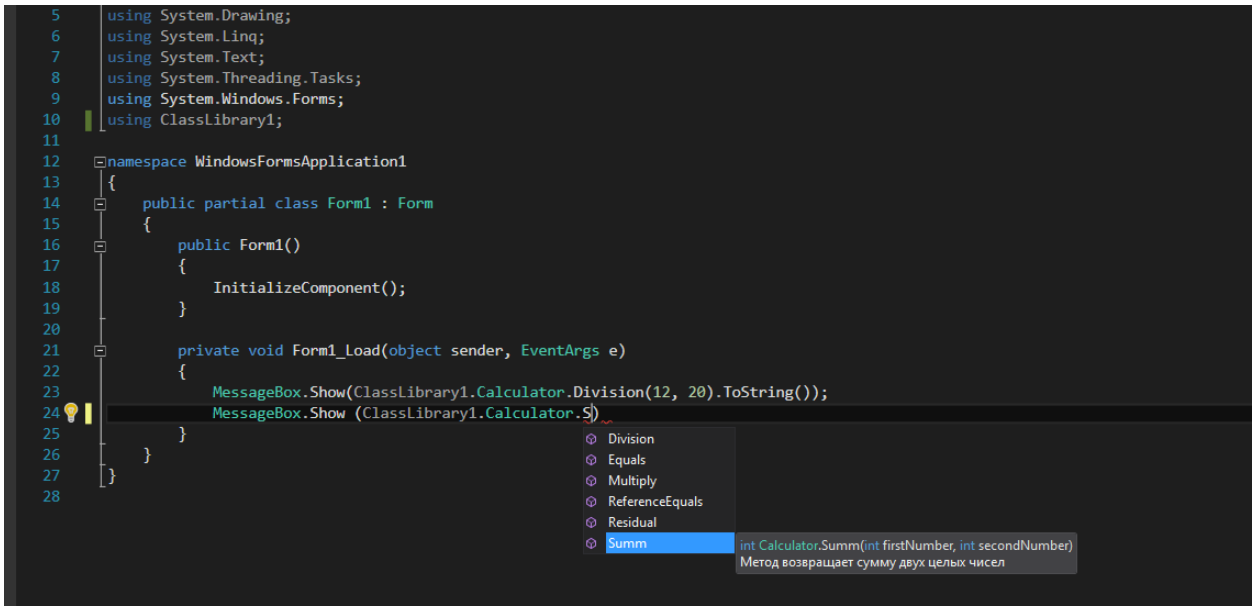
лайте двойной клик левой клавишей мыши по нему, у вас откроется новая вкладка с добавленным методом **`Form1_Load`**



Добавим пространство имен с названием нашей библиотеки: `using ClassLibrary1;`



Пропишите вызов нескольких методов из библиотеки и вывод результата в диалоговом окне «**MessageBox**». Обратите внимание что при выборе методом из библиотеки у вас показывается подсказка которую мы прописывали. Если такого не происходит то, xml файл документации отсутствует в директории с библиотекой. Пример подсказок представлен на скриншоте ниже.



Запустите проект, нажав клавишу **F5**. И вы увидите результат выполнения методов прописанных в библиотеке, с параметрами которые вы передали при вызове.

4 Задания

1. Выполнить пример из лабораторной работы.
2. Выполнить задание по вариантам:

Вариант 1

Разработать DLL для работы с массивами целых чисел. В библиотеку включить метод поиска максимального элемента массива, метод поиска минимального элемента массива и метод обмена значениями максимального и минимального элементов массива. В программе предусмотреть создание и печать массива, обмен значениями максимального и минимального элементов и печать нового массива.

Вариант 2

Разработать DLL для работы с целыми и вещественными числами. В библиотеку включить методы печати чисел в форматах E, F и C. В программе предусмотреть анализ на соответствие выводимого числа заданному формату и печать числа после выполнения любого метода DLL. Числа задавать в режиме диалога.

Вариант 3

Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать значение года, месяца и дня недели. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.

Вариант 4

Разработать DLL для работы с переменными типа Char. В библиотеку включить методы, позволяющие по значению заданной переменной типа Char, определить является ли заданный символ цифрой, пробелом и управляющим символом. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.

Вариант 5

Разработать DLL для работы с массивами целых чисел. В библиотеку включить метод сортировки только четных значений элементов массива, метод сортировки только нечетных значений элементов массива и метод сортировки всех значений массива. Все сортировки выполнять по убыванию. В программе предусмотреть создание и печать исходного массива, печать массива после любого вида сортировки.

Вариант 6

Разработать DLL для работы с целыми и вещественными числами. В библиотеку включить методы печати чисел в форматах C, N и X. В программе предусмотреть анализ на соответствие выводимого числа заданному формату и печать числа после выполнения любого метода DLL. Числа задавать в режиме диалога.

Вариант 7

Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать значение часа, минуты и секунды. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.

Вариант 8

Разработать DLL для работы с переменными типа Char. В библиотеку включить методы, позволяющие по значению заданной переменной типа Char, определить является ли заданный символ буквой, прописной буквой, латинской буквой. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.

Вариант 9

Разработать DLL для работы с массивами целых чисел. В библиотеку включить метод сортировки только положительных значений элементов массива, метод сортировки только отрицательных значений элементов массива и метод сортировки всех значений массива. Все сортировки выполнять по убыванию. В программе предусмотреть создание и печать исходного массива, печать массива после любого вида сортировки.

Вариант 10

Разработать DLL для исследования операций сравнения <=, >=, ==, != при работе с целыми и вещественными числами. В библиотеку включить методы для каждой операции сравнения. В программе предусмотреть печать результатов сравнения после выполнения любого метода DLL. Числа задавать в режиме диалога.

Вариант 11

Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать секунды, миллисекунды и тики. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.

Вариант 12

Разработать DLL для работы с переменными типа Char. В библиотеку включить методы, позволяющие по значению заданной переменной типа Char, определить является ли заданный символ буквой, строчной буквой, кириллицей. В программе предусмотреть печать результатов (или необходимых комментарий) после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.

Вариант 13

Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать количество дней до «Нового года», номер дня в году и год, месяц, день. В программе предусмотреть печать результатов после выполнения любого метода DLL. Зна-

чение переменных задавать в режиме диалога.

Вариант 14

Разработать DLL для исследования логических операций &&, || и ! при работе с двумя логическими переменными. В библиотеку включить методы для каждой логической операции, при этом необходимо перебирать все логические комбинации двух переменных. В программе предусмотреть печать результатов после выполнения любого метода DLL.

Вариант 15

Разработать DLL для работы с переменными типа DateTime. В библиотеку включить методы, позволяющие по значению заданной переменной типа DateTime отдельно печатать название месяца, номер месяца в году и год. В программе предусмотреть печать результатов после выполнения любого метода DLL. Значение переменных задавать в режиме диалога.

5 Контрольные вопросы

1. Что такое DLL?
2. Зачем нужны динамически подключаемые библиотеки?
3. Какой тип проекта имеют динамически подключаемые библиотеки?
4. Как DLL подключаются к проекту?