Лабораторная работа №16

Тема: Создание и использование библиотек динамической компоновки.

Цель: Научиться создавать библиотеки динамической компоновки и использовать их при разработке программ.

Выполнение работы:

Иногда при запуске какой-либо программы появляется сообщение, что не найден файл ***.dll**. Для операционных систем Microsoft Windows, большая часть функциональных возможностей операционной системы обеспечивается библиотеками динамической компоновки (DLL). Кроме того, некоторые возможности программ могут быть реализованы в библиотеках DLL. Например некоторые программы могут содержать много различных модулей и при работе использовать только часть из них. Таким образом операционная система и программы загружаются быстрее, работают быстрее и занимают меньше места на диске компьютера.

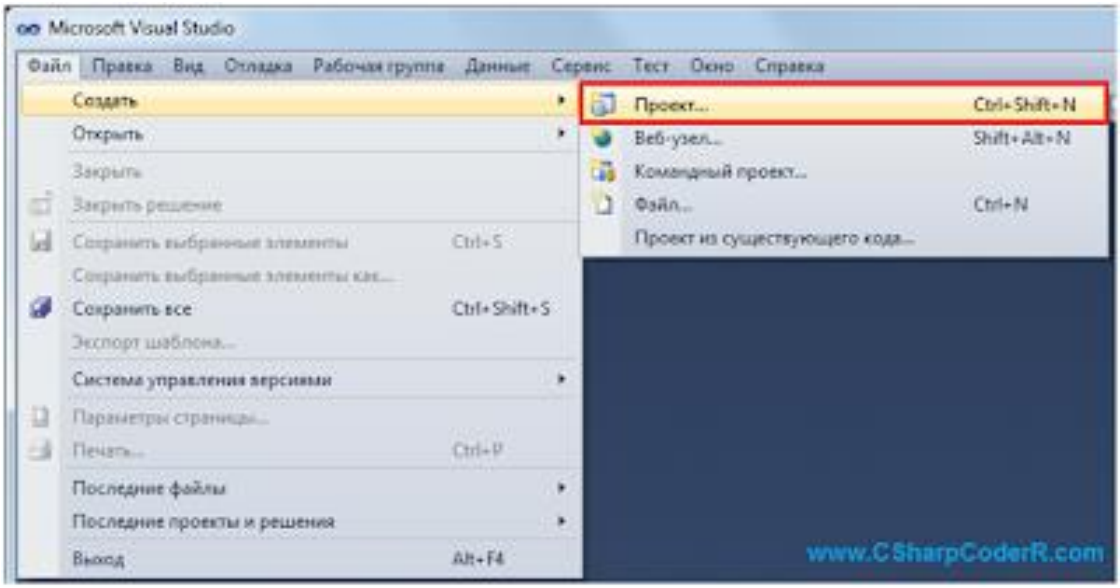
Что такое DLL?

DLL — 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.

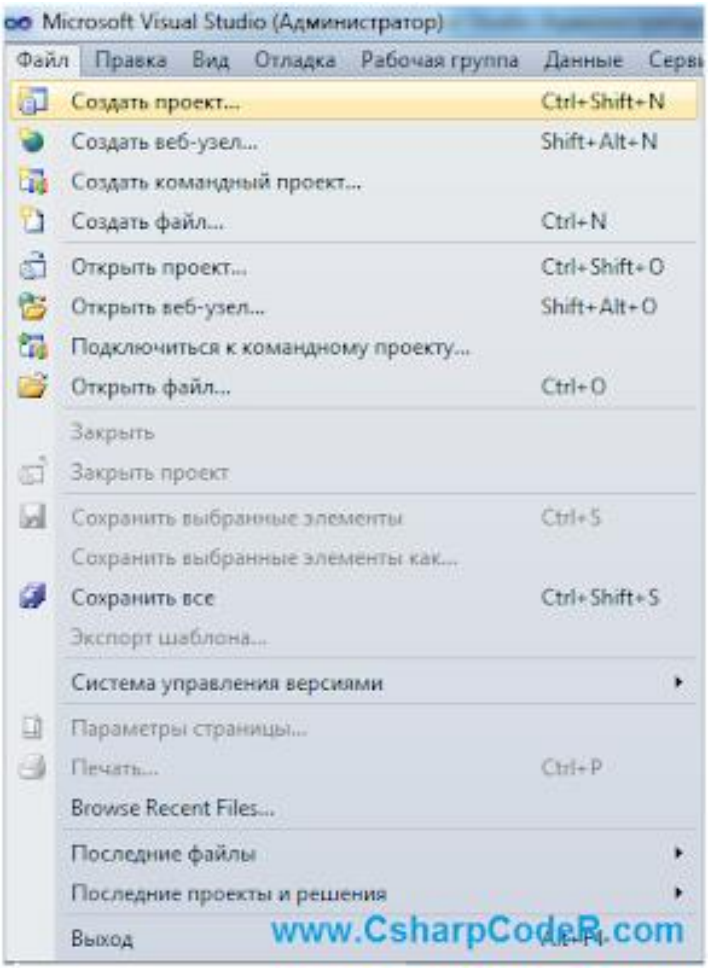
С помощью библиотек можно реализовать модульность для программы, в виде отдельных компонентов. Например бухгалтерскую программу можно продать по модулям. Каждый модуль может быть загружен в основной программе во время выполнения установки. Отдельные модули загружается только при запросе функций заложенных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

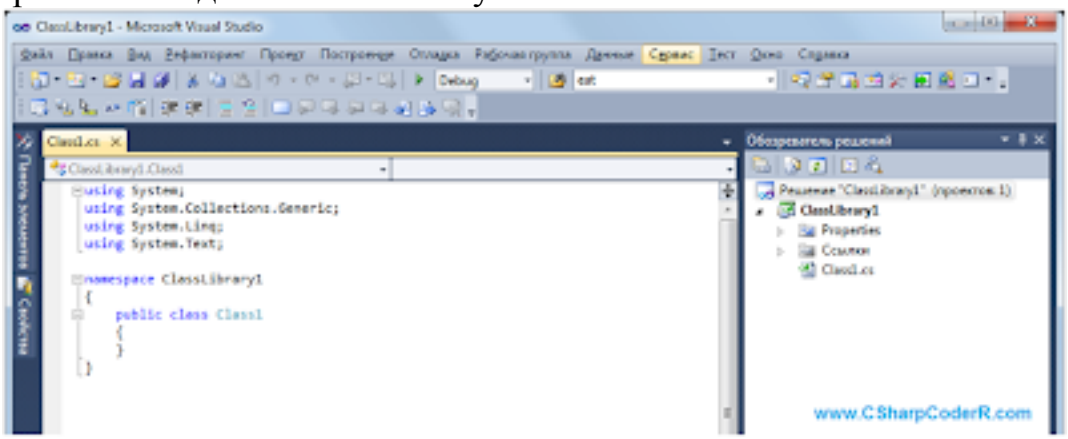
Для начала, создадим новый проект, для этого запустите **Microsoft Visual Studio** и перейдите в меню **Файл -> Создать -> Проект…** или выполните сочетание клавиш **Ctrl+Shift+N**.



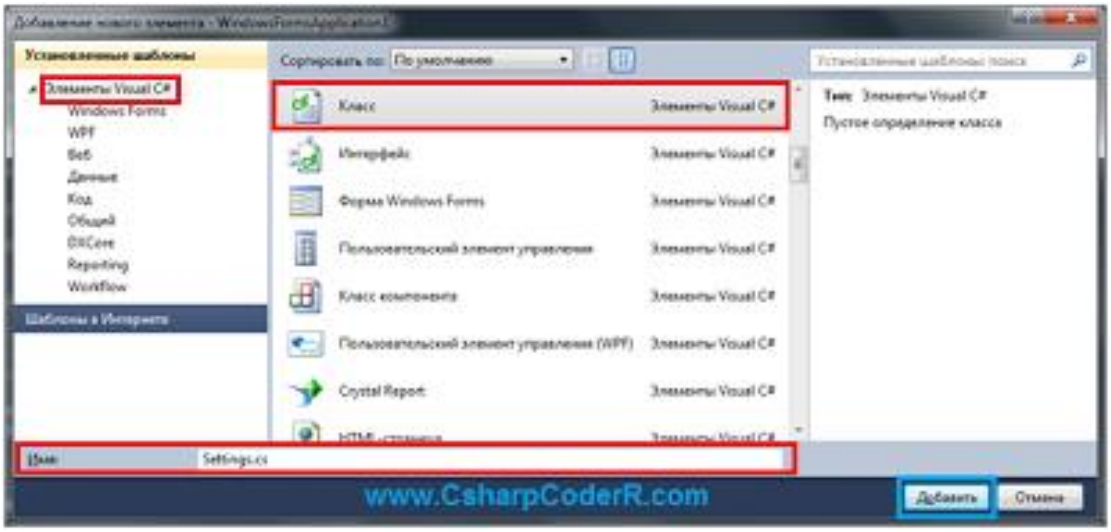
В открывшемся диалоговом окне выберете версию Framework, а в левой части **«Установленные шаблоны»** выберете **«Visual C#»**, в центральной части вам будет представлен список шаблонов, выберете **«Библиотека классов»** и введите имя библиотеки, можно оставить по умолчанию.



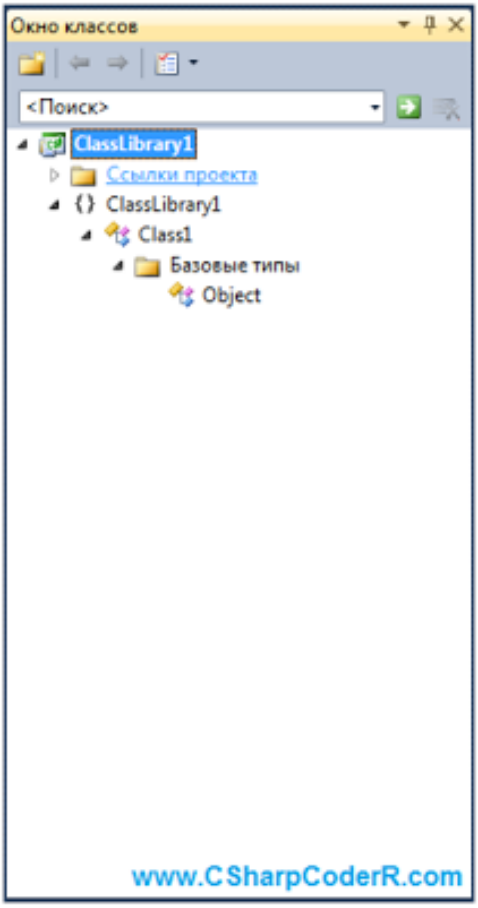
У вас откроется вкладка с классом по умолчанию.



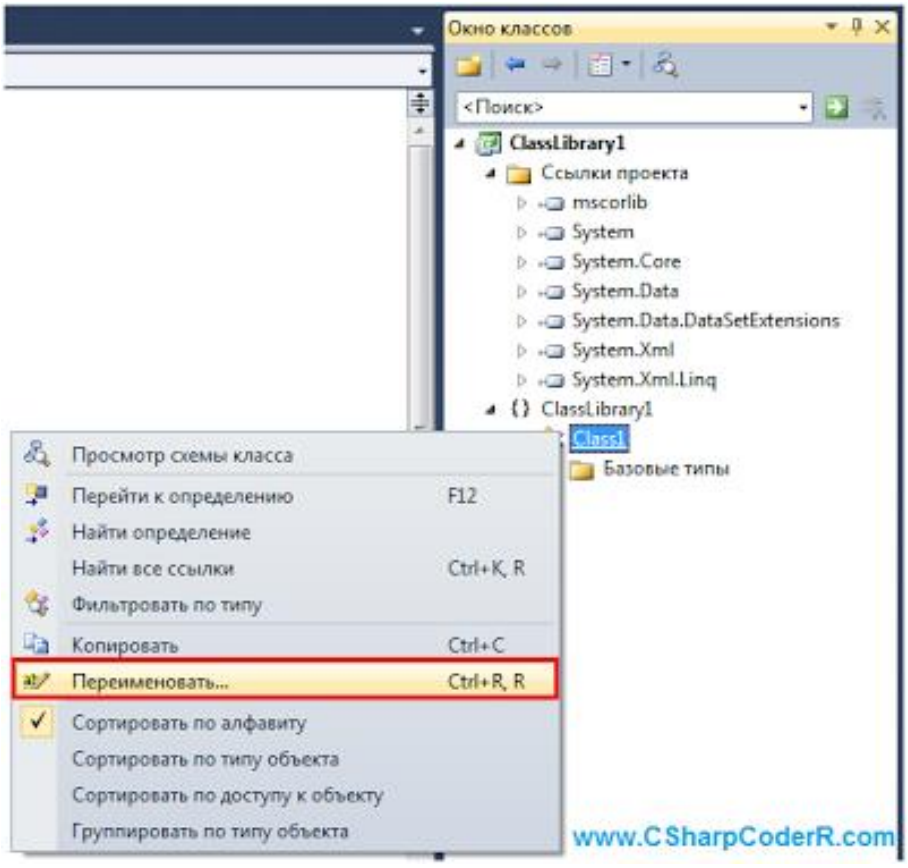
По умолчанию создается класс **Class1**, переименуем его в класс Calculator. Для этого перейдите в меню Вид – > Классы или выполните сочетание клавиш **Ctrl + W**, с последующем нажатием клавиши **C**.



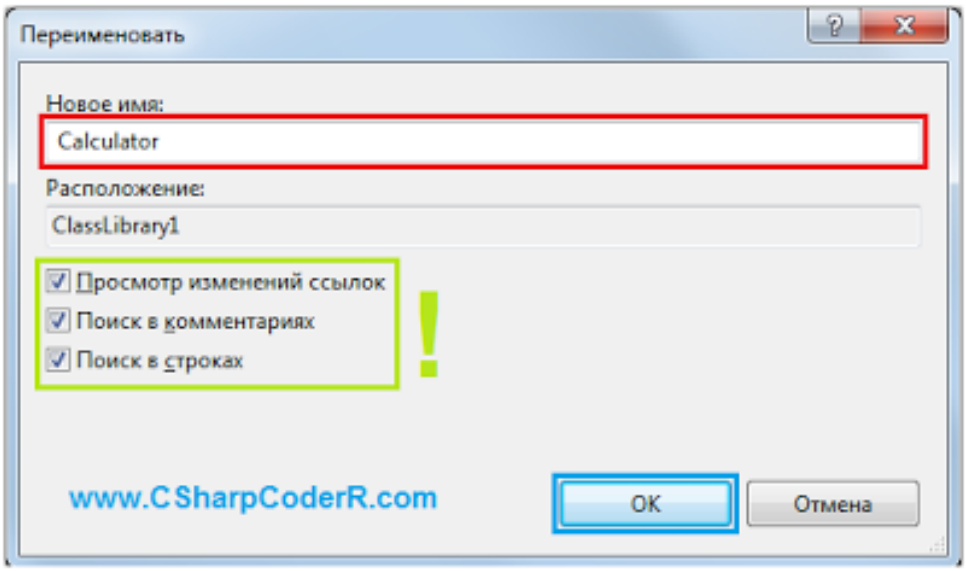
В правой части программы у вас откроется вкладка **«Окно классов»**.



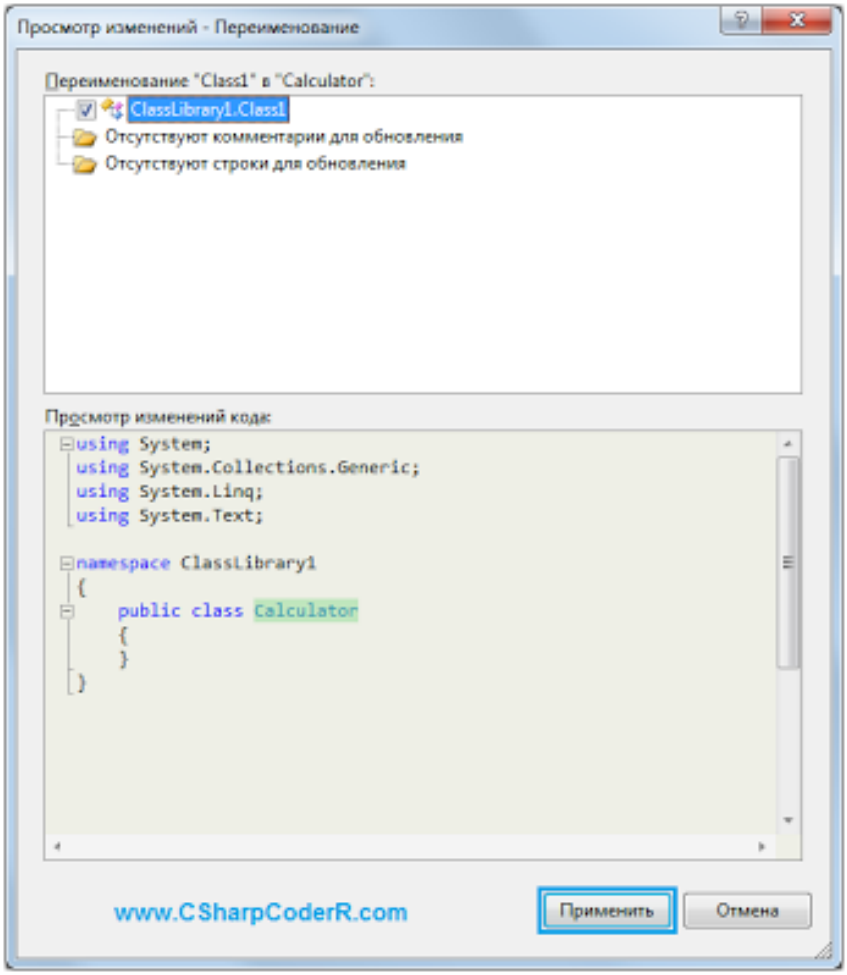
Выберете по умолчанию созданный класс Class1, сделайте клик правой клавишей мыши по нему и выберете «**Переименовать…**».



В открывшемся окне введите новое имя класса Calculator и нажмите кнопку OK, обратите внимание, что данное окно позволяет переименовать класс во всем проекте.



После нажатия клавиши OK, вам будет предложено просмотреть список изменений, которые будут внесены в проект, а так же просмотр изменений кода.

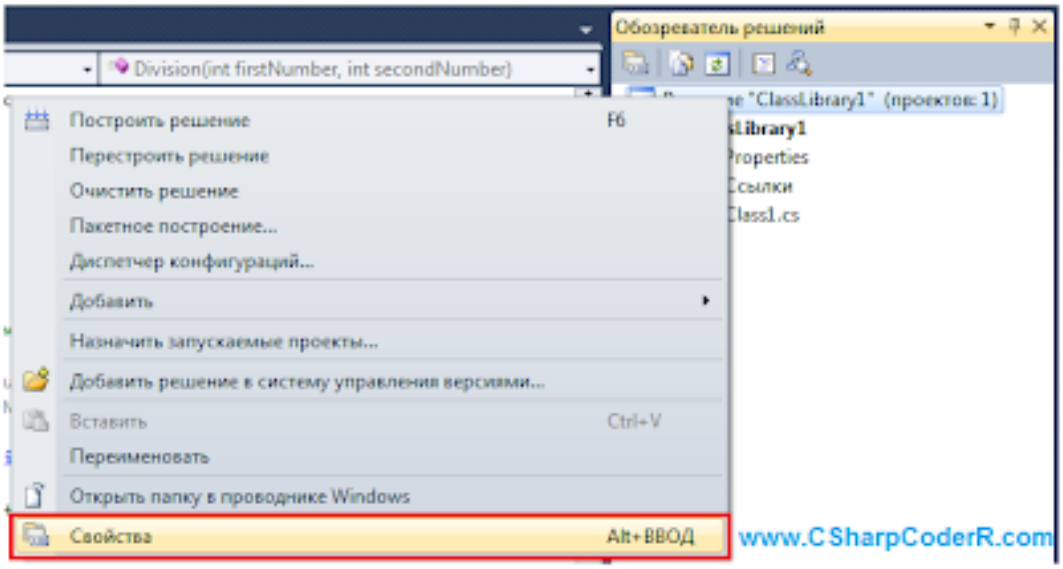


Добавим в класс **Calculator** несколько методов и добавим к ним описание.

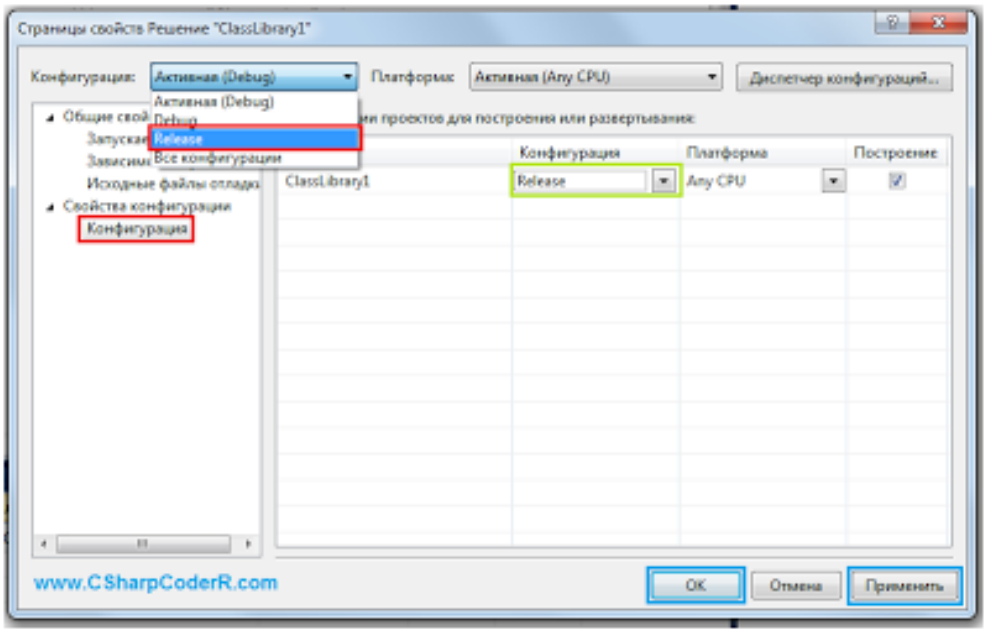
```
namespace ClassLibrary1
{
///
/// Математический класс
///
public class Calculator
{
///
/// Метод возвращает сумму двух целых чисел
///
/// /// ///
public static int Summ(int firstNumber, int secondNumber)
{
return firstNumber + secondNumber;
}
///
/// Метод возвращает разность двух целых чисел
///
/// /// ///
public static int Division(int firstNumber, int secondNumber)
{
return firstNumber - secondNumber;
}
///
/// Метод возвращает произведение двух чисел
///
/// /// ///
public static long Multiply(long x, long y)
{
return (x * y);
}
```

```
///
/// Метод возвращает деление двух чисел
///
/// /// ///
public static int Residual(int firstNumber, int secondNumber)
{
return (firstNumber / secondNumber);
}
}
}
```

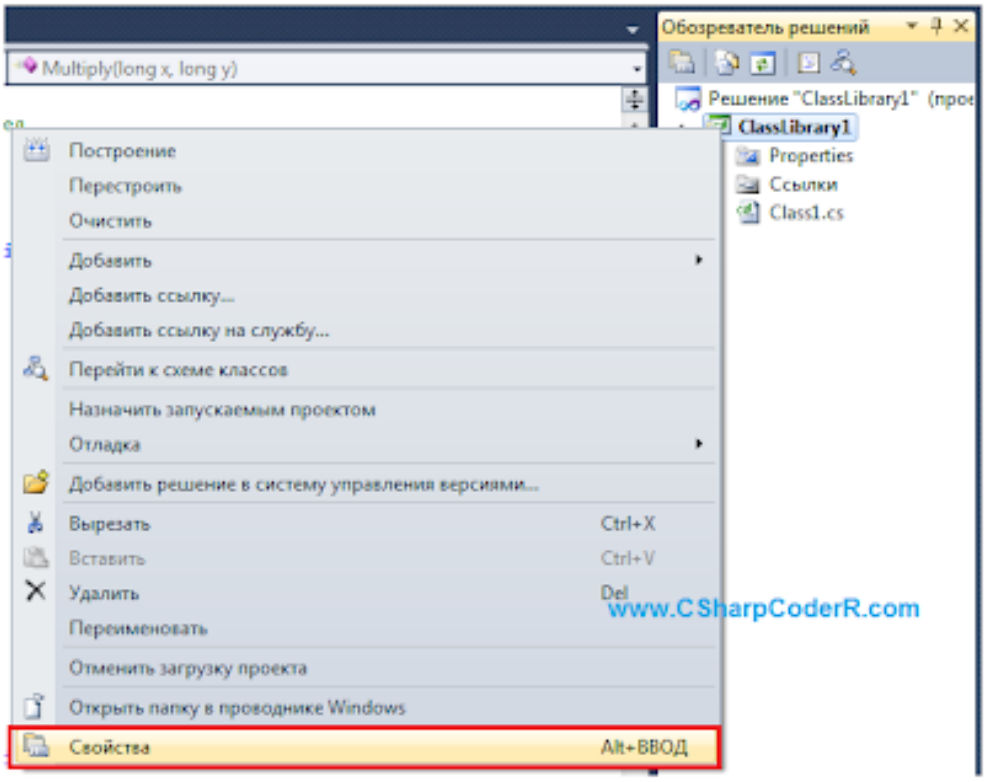
По умолчанию для всех проектов стоит режим построения **Debug**(режим отладки), переведем проект в режим построения конечной версии (**Release**). Для этого перейдите в обозреватель решений и сделав клик правой клавишей мыши по названию проекта, выберете в открывшемся контекстном меню пункт «**Свойства**».



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

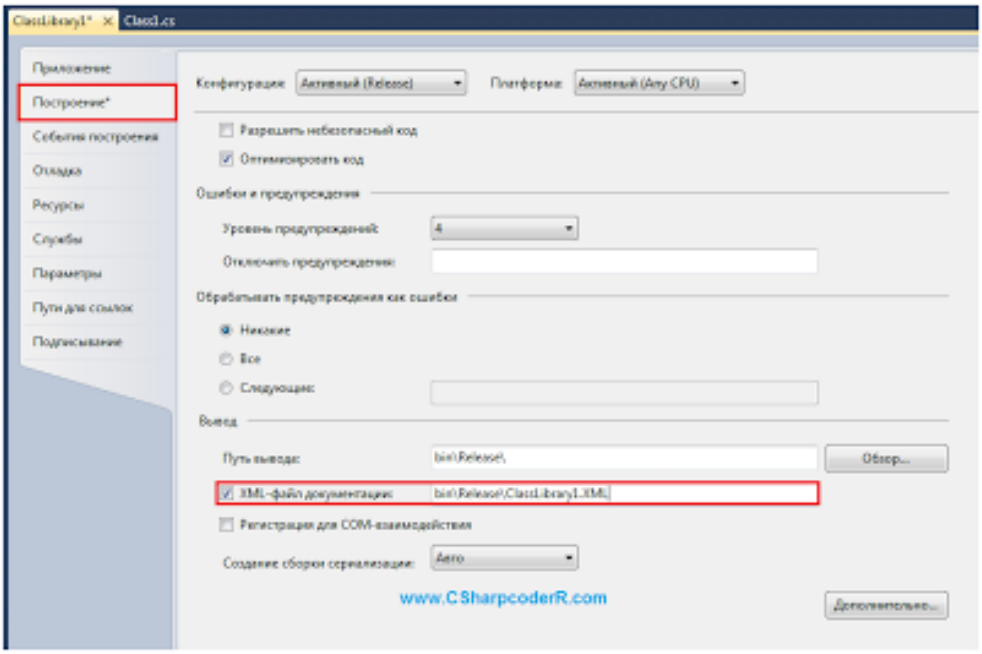


Наверно все замечали, что при наборе кода, появляется подсказка к методам или функциям. Мы задали такую подсказку в тегах. Но если сейчас просто выполнить построение библиотеки, то при подключении к другим проектам никаких подсказок видно не будет. Что бы устранить данную проблему, нам необходимо сформировать 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**».

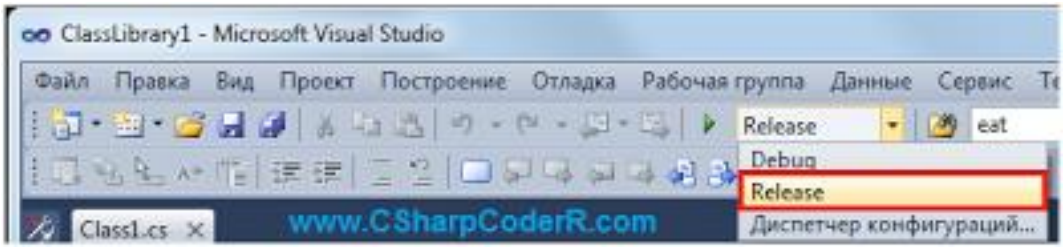


После этих действий у вас откроется новая вкладка со свойствами проекта **ClassLibrary1**. Выберете в вкладку «**Построение**» и найдите раздел

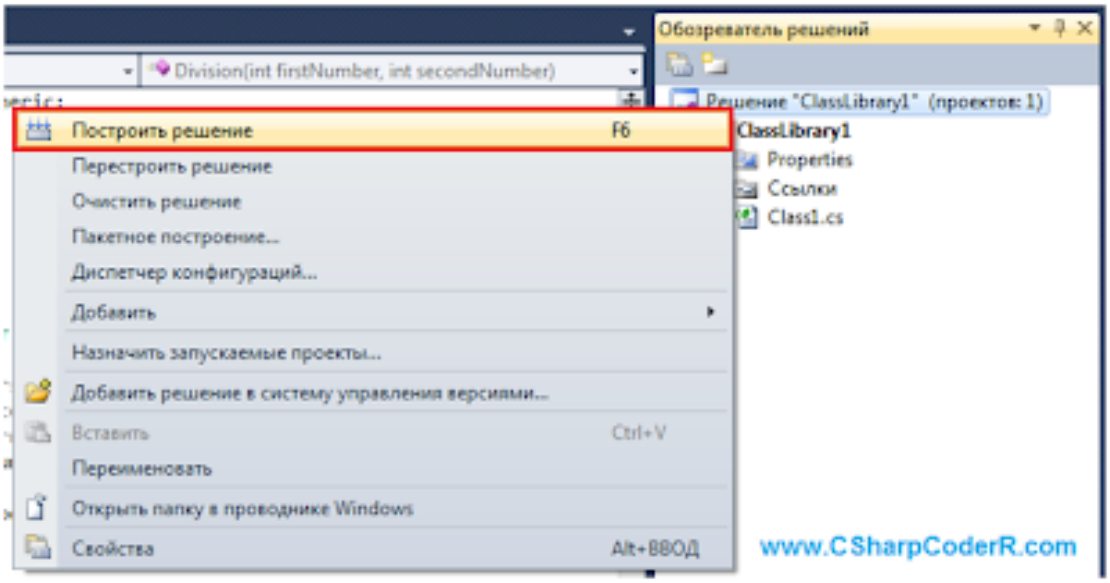
«Вывод», там вам будет предложено ввести путь куда будет выполнено построение конечной версии библиотеки и пункт необходимый нам для построения xml файла документации, тут вам необходимо просто поставить галочку как показано на скриншоте ниже. Тут важно чтобы библиотека и файл документации находились в одном месте, поэтому проверьте чтобы их путь вывода совпадал.



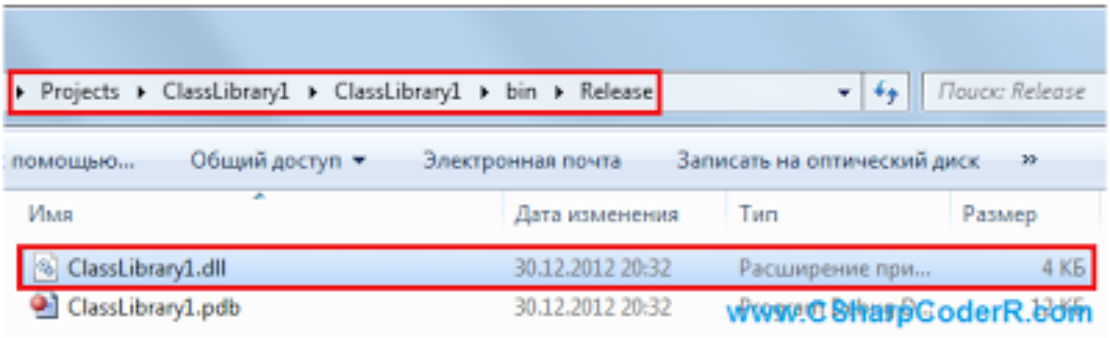
Остались последние шаги и мы получим готовую для использования библиотеку. И так продолжим, вам необходимо в верхней части программы выбрать режим конфигурации **Release** как показано на скриншоте ниже.



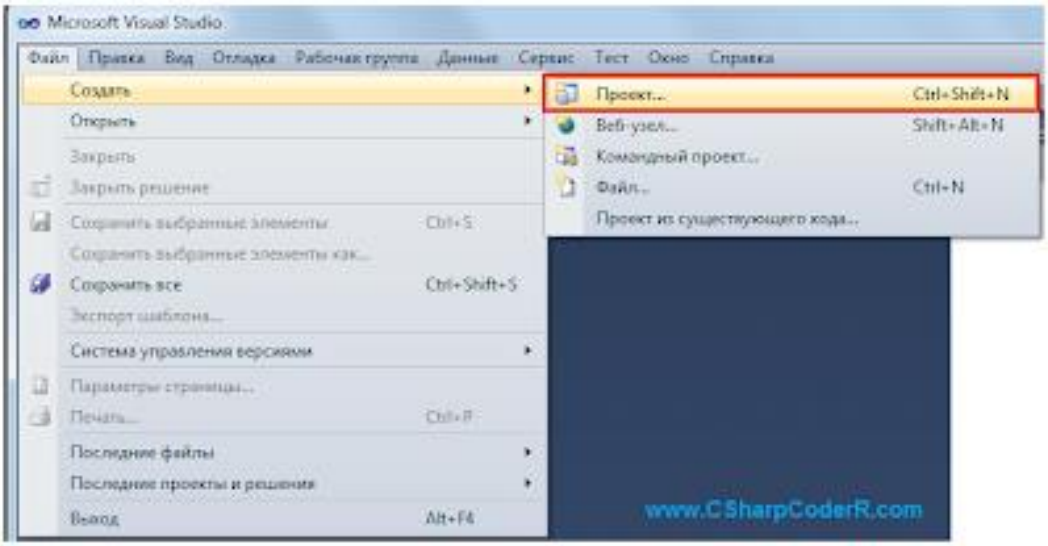
После этого, выполним построение решения. Нажав на клавиатуре клавишу **F6.**



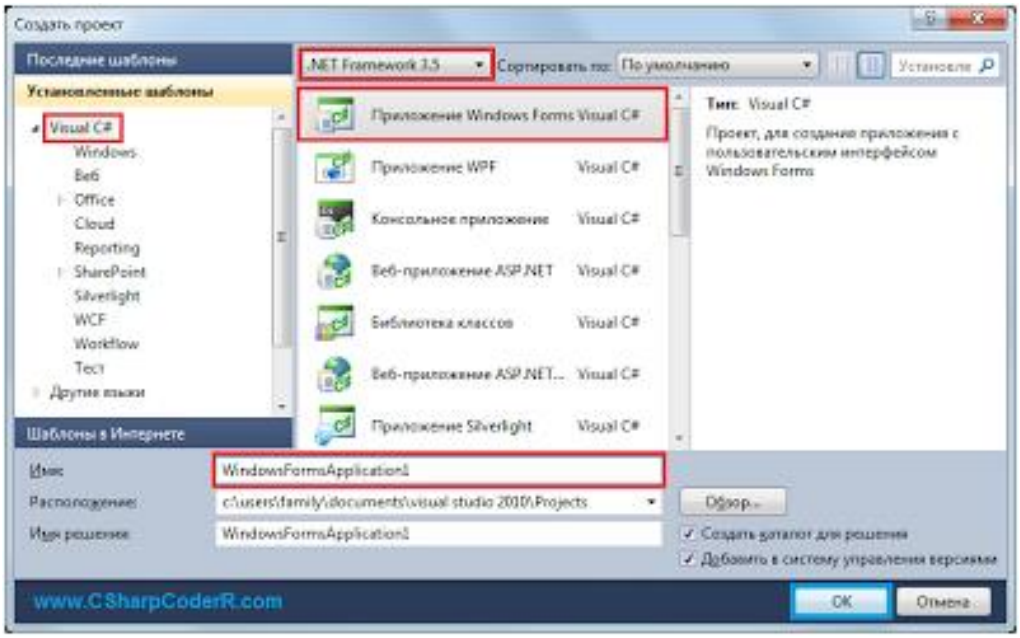
Как только программа закончит построение, можно перейти в директорию с проектом и посмотреть что получилось. На этом этапе закончилось создание библиотеки.



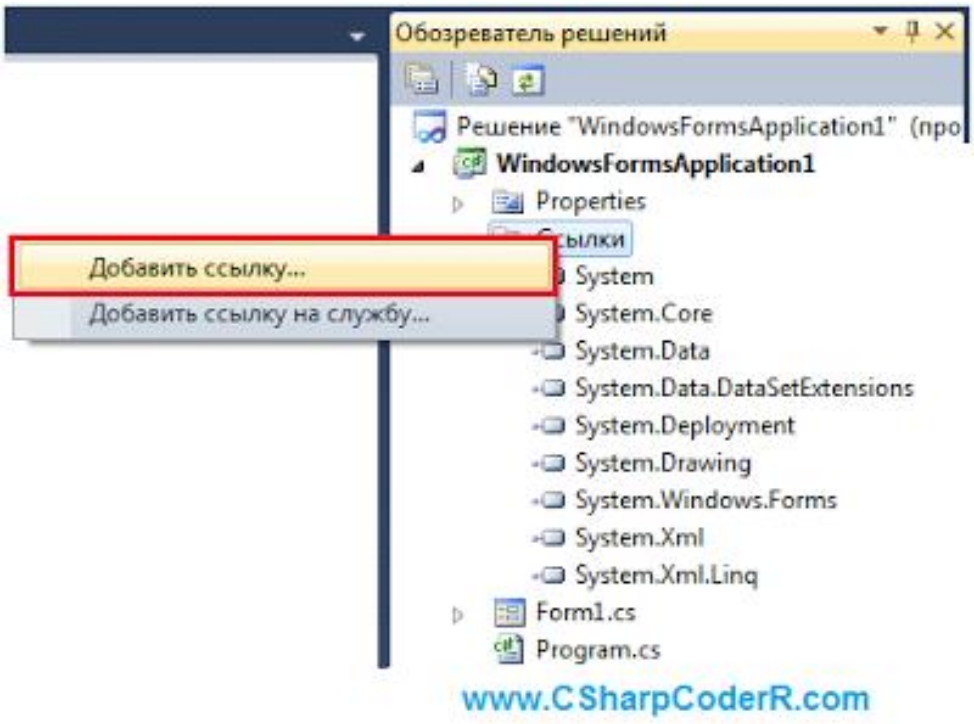
Для проверки работоспособности библиотеки создадим тестовый проект. Выполните **Файл -> Создать -> Проект…**



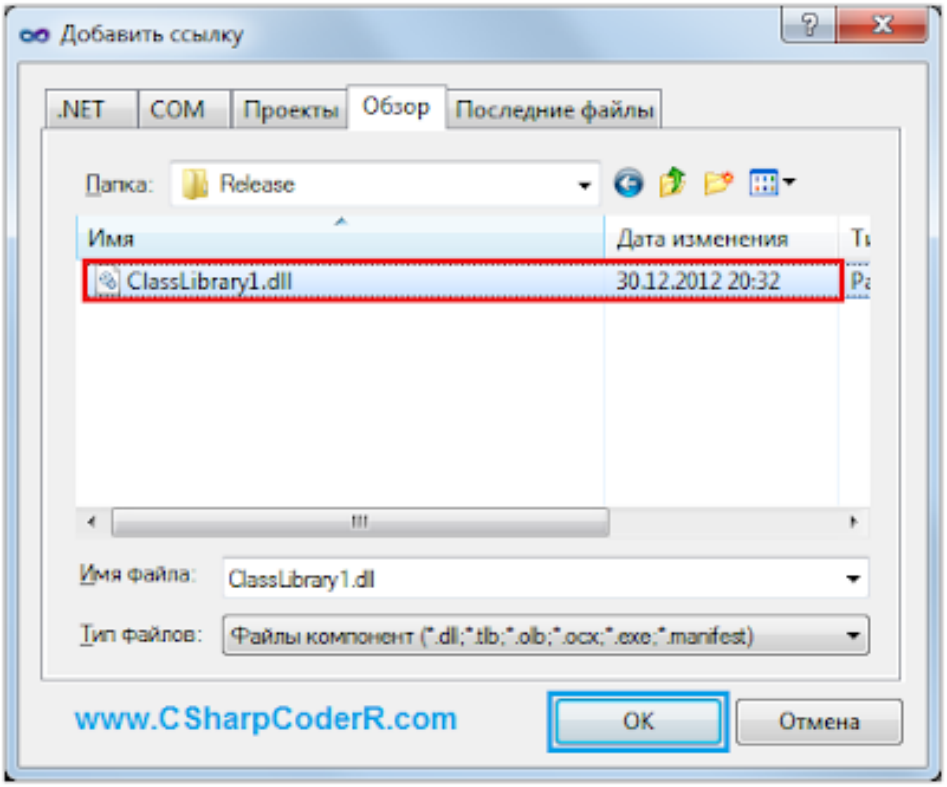
Выберете из предложенных шаблонов, шаблон «Приложение Windows Forms Visual C#». Задайте имя проекта и нажмите кнопку OK.



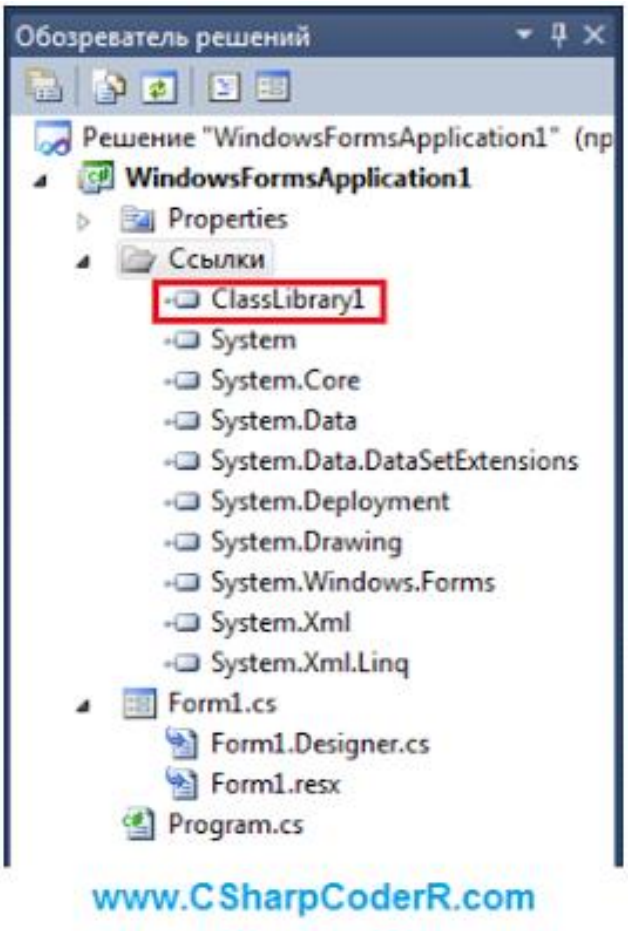
После создания проекта, в обозревателе решений сделайте клик правой клавишей мыши по разделу «**Ссылки**» и выберете в появившемся контекстном меню пункт «**Добавить ссылку…**».



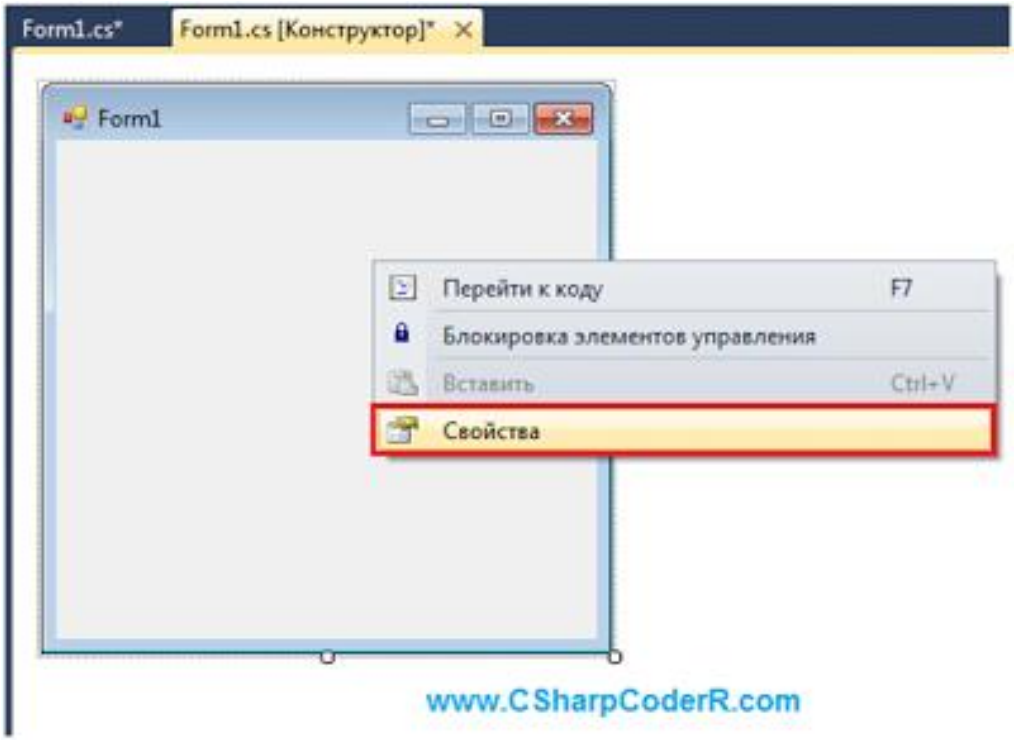
Выберете вкладку «Обзор» и укажите вашу библиотеку.



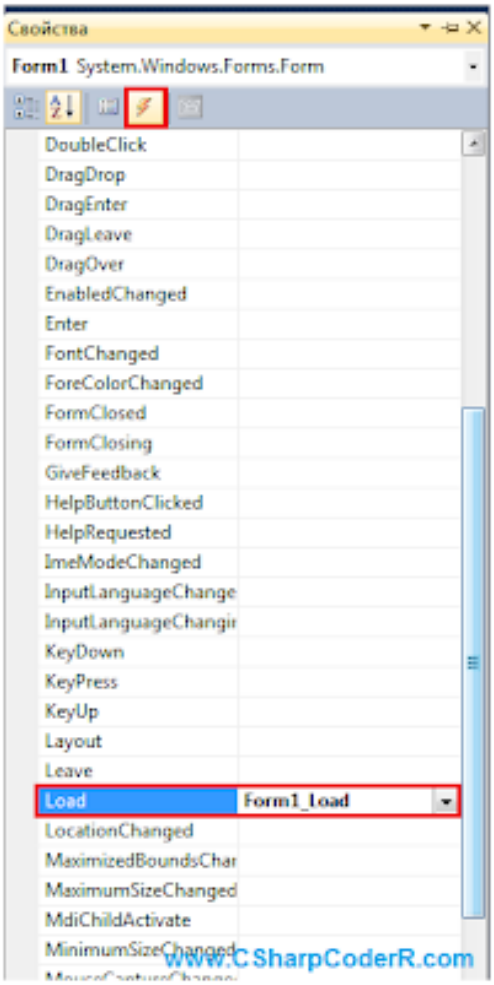
Если вы все успешно выполнили, в разделе «**Ссылки**» у вас появится название вашей библиотеки.



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

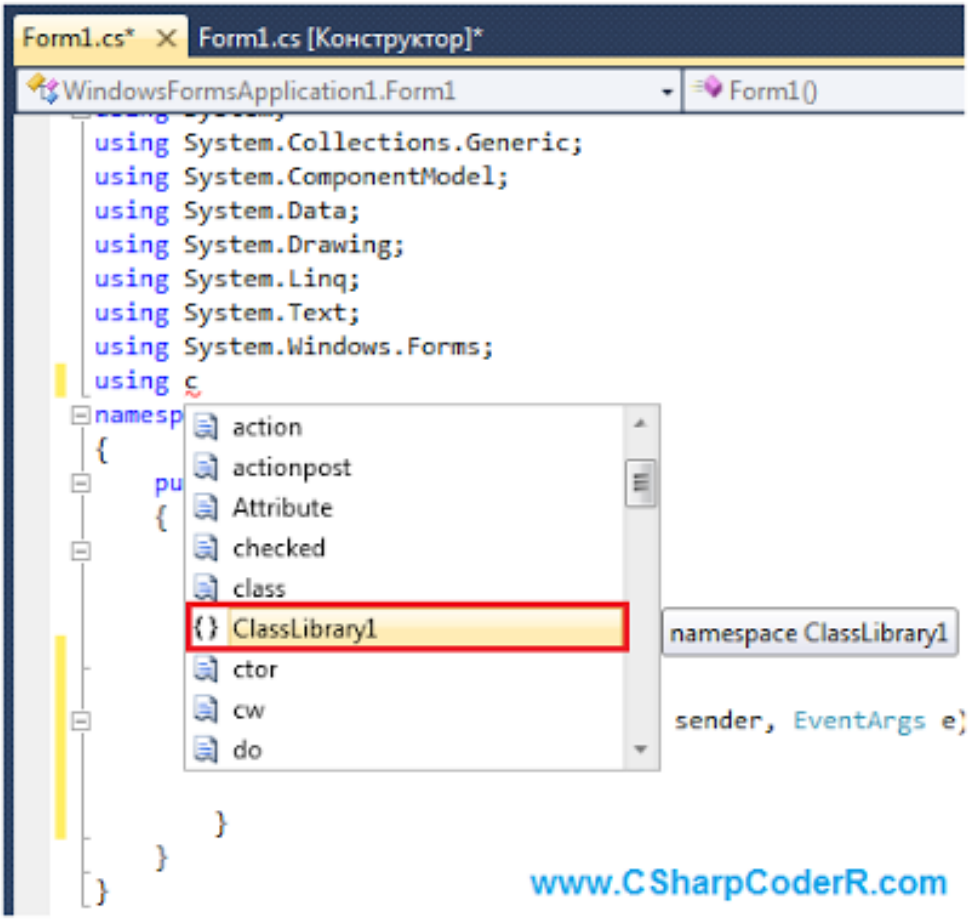


В боковой панели откроются «**Свойства**» формы. Найдите **метод Load** и сделайте двойной клик левой клавишей мыши по нему, у вас откроется новая вкладка с добавленным методом **Form1_Load**.

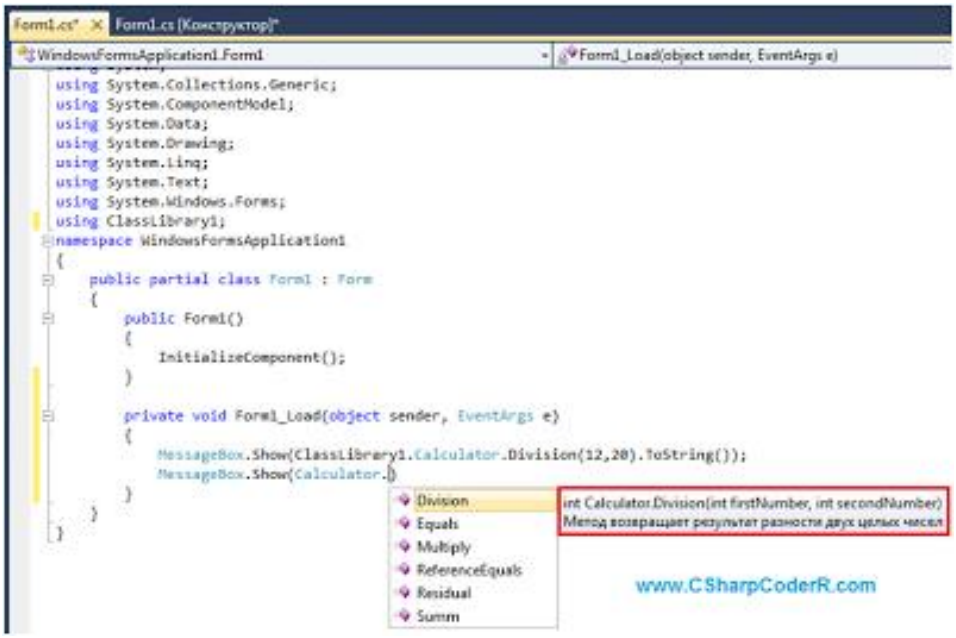


Добавим пространство имен с названием нашей библиотеки.

```
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 вы можете посмотреть на скриншоте ниже.



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

Индивидуальные задания:

Задание 1. Модифицируйте разрабатываемое программное средство, добавив на форму необходимые элементы, получив полноценный калькулятор.

Задание 2. Разработайте программное средство, в котором будет использоваться разработанная вами библиотека. Библиотека должна реализовать Задание 2 из лабораторной работы№4.

Шкала оценивания индивидуальных заданий

Примеры	1-4 баллов
Примеры + Задание 1	1-7 баллов
Примеры + Задание 1+ Задание 2	1-10 баллов

Содержание отчета:

1. Номер и тема лабораторной работы.
2. Цель лабораторной работы.
3. Техническое оснащение.
4. При выполнении примеров, необходимо в отчет внести скриншоты готовых программ.
5. При выполнении индивидуальных заданий в отчет внести изображение кода программы и окно выполнения программы.
6. Вывод по лабораторной работе.

Контрольные вопросы:

1. Что такое библиотека динамической компоновки?
2. Для чего предназначена DLL?
3. Какие существуют способы подключения DLL?
4. В чем разница DLL и статической библиотеки?
5. Как организованы память и DLL?
6. Преимущества использования DLL?