Модульное тестирование в Visual Studio

Модульное тестирование (или Unit-тестирование) предназначено для проверки правильности выполнения небольшого блока кода, решающего свою конкретную задачу. В статье рассказывается, как проводить в модульное тестирование в Visual Studio. Разработка ведётся на языке C#.

Создание проекта программы, модули которой будут тестироваться

Разработаем проект содержащий класс, который вычисляет площадь прямоугольника по длине двух его сторон.

Создадим в Visual Studio новый проект Visual C# -> Библиотека классов. Назовём его **MathTaskClassLibrary**.

Class1 переименуем в **Geometry**.

В классе реализуем метод, вычисляющий площадь прямоугольника. Для демонстрации остановимся на работе с целыми числами. Код программы приведён ниже.

C#

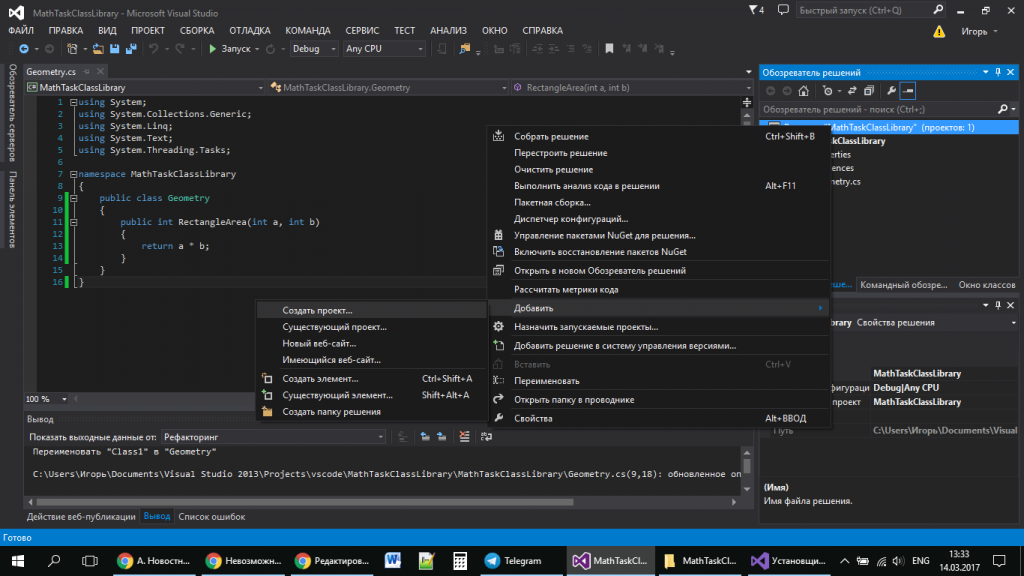
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;    namespace MathTaskClassLibrary  {      public class Geometry      {          public int RectangleArea(int a, int b)          {              return a \* b;          }      }  } |

Площадь прямоугольника, как известно, это произведение двух его сторон.

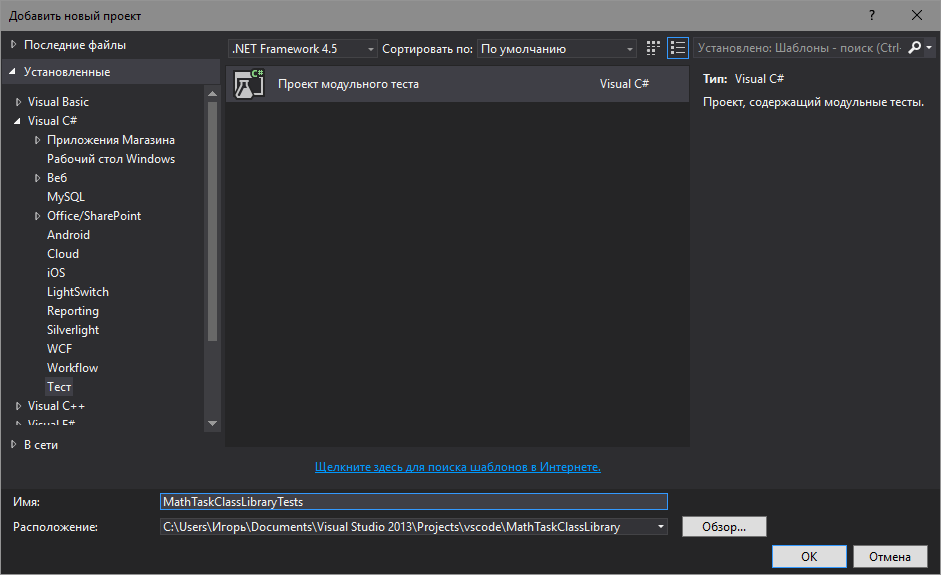
Создание проекта для модульного тестирования в Visual Studio

Чтобы выполнить unit-тестирование, необходимо в рамках того же самого решения создать ещё один проект соответствующего типа.

Правой кнопкой щёлкните по решению, выберите “Добавить” и затем “Создать проект…”.

[](https://vscode.ru/wp-content/uploads/2017/03/Sozdanie-proekta-dlya-modulnogo-testirovaniya.png)

В открывшемся окне в группе Visual C# щёлкните “Тест”, а затем выберите “Проект модульного теста”. Введите имя проекта **MathTaskClassLibraryTests**и нажмите “ОК”. Таким образом проект будет создан.

[](https://vscode.ru/wp-content/uploads/2017/03/Tip-proekta-dlya-Unit-testirovaniya.png)

Перед Вами появится следующий код:

C#

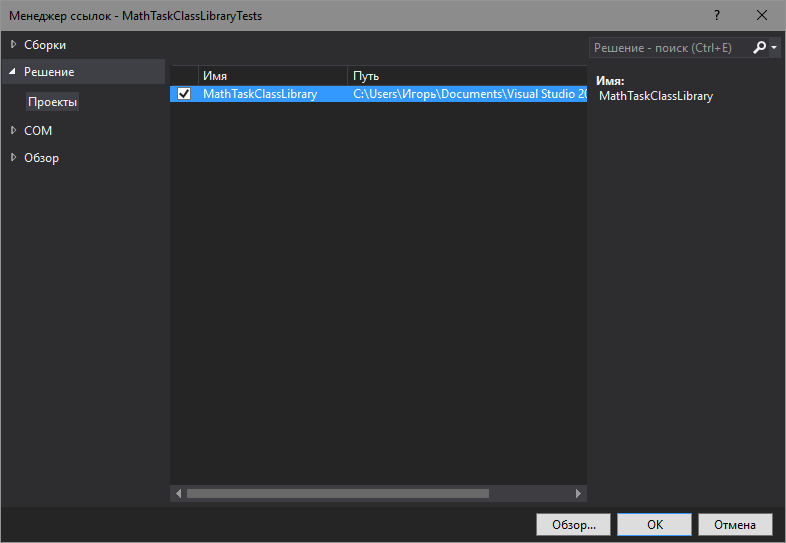
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | using System;  using Microsoft.VisualStudio.TestTools.UnitTesting;    namespace MathTaskClassLibraryTests  {      [TestClass]      public class UnitTest1      {          [TestMethod]          public void TestMethod1()          {          }      }  } |

Директива [TestMethod] обозначает, что далее идёт метод, содержащий модульный (unit) тест. А [TestClass] в свою очередь говорит о том, что далее идёт класс, содержащий методы, в которых присутствуют unit-тесты.

В соответствии с принятыми соглашениями переименуем класс UnitTest1 в **GeometryTests**.

Затем в References проекта необходимо добавить ссылку на проект, код которого будем тестировать. Правой кнопкой щёлкаем на References, а затем выбираем “Добавить ссылку…”.

В появившемся окне раскрываем группу “Решение”, выбираем “Проекты” и ставим галочку напротив проекта **MathTaskClassLibrary**. Затем жмём “ОК”.

[](https://vscode.ru/wp-content/uploads/2017/03/Dobavlenie-ssyilki-na-proekt-kotoryiy-budet-testirovatsya.png)

Также в коде необходимо подключить с помощью директивы using следующее пространство имён:

C#

|  |  |
| --- | --- |
| 1 | using MathTaskClassLibrary; |

Займёмся написание теста. Проверим правильно ли вычисляет программа площадь прямоугольника со сторонами 3 и 5. Ожидаемый результат (правильное решение) в данном случае это число 15.

Переименуем метод TestMethod1() в **RectangleArea\_3and5\_15returned()**. Новое название метода поясняет, что будет проверяться (RectangleArea – площадь прямоугольника) для каких значений (3 и 5) и что ожидается в качестве правильного результата (15 returned).

Тестирующий метод обычно содержит три необходимых компонента:

1. исходные данные: входные значения и ожидаемый результат;
2. код, вычисляющий значение с помощью тестируемого метода;
3. код, сравнивающий ожидаемый результат с полученным.

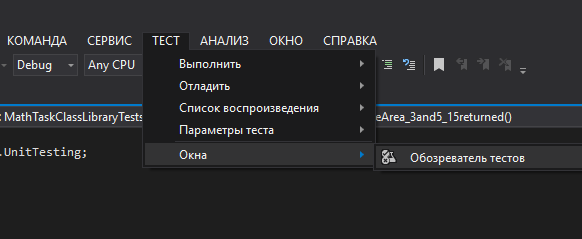
Соответственно тестирующий код будет таким:

C#

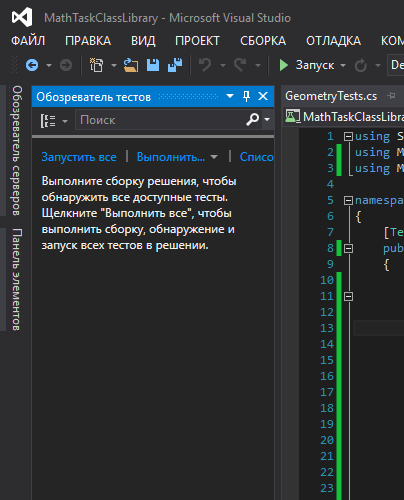
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | using System;  using Microsoft.VisualStudio.TestTools.UnitTesting;  using MathTaskClassLibrary;    namespace MathTaskClassLibraryTests  {      [TestClass]      public class GeometryTests      {          [TestMethod]          public void RectangleArea\_3and5\_15returned()          {              // исходные данные              int a = 3;              int b = 5;              int expected = 15;                // получение значения с помощью тестируемого метода              Geometry g = new Geometry();              int actual = g.RectangleArea(a, b);                // сравнение ожидаемого результата с полученным              Assert.AreEqual(expected, actual);          }      }  } |

Для сравнения ожидаемого результата с полученным используется метод **AreEqual** класса **Assert**. Данный класс всегда используется при написании unit тестов в Visual Studio.

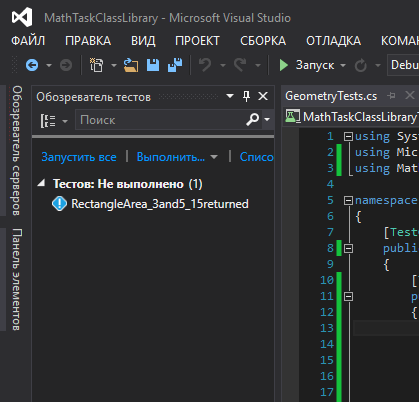
Теперь, чтобы просмотреть все тесты, доступные для выполнения, необходимо открыть окно “Обозреватель тестов”. Для этого в меню Visual Studio щёлкните на кнопку “ТЕСТ”, выберите “Окна”, а затем нажмите на пункт “Обозреватель тестов”.

[](https://vscode.ru/wp-content/uploads/2017/03/Okno-Obozrevatel-testov.png)

В студии появится следующее окно:

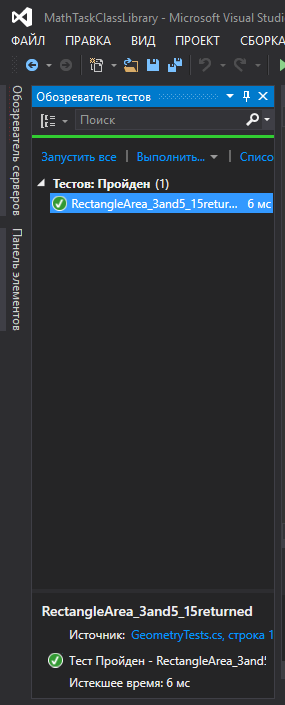
[](https://vscode.ru/wp-content/uploads/2017/03/Obozrevatel-testov.png)

В данный момент список тестов пуст, поскольку решение ещё ни разу не было собрано. Выполним сборку нажатием клавиш Ctrl + Shift + B. После её завершения в “Обозревателе тестов” появится наш тест.

[](https://vscode.ru/wp-content/uploads/2017/03/Unit-test.png)

Синяя табличка с восклицательным знаком означает, что указанный тест никогда не выполнялся. Выполним его.

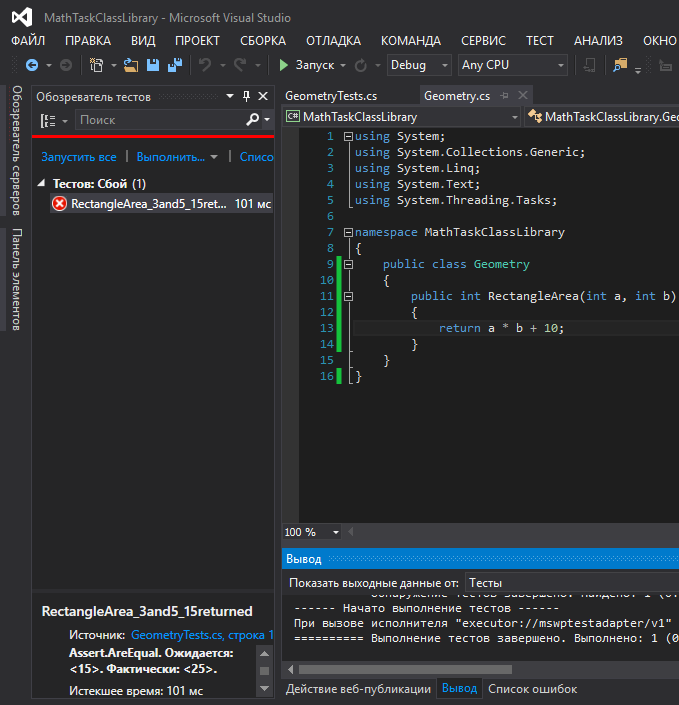
Для этого нажмём правой кнопкой мыши на его имени и выберем “Выполнить выбранные тесты”.

[](https://vscode.ru/wp-content/uploads/2017/03/Uspeshnyiy-modulnyiy-test.png)

Зелёный кружок с галочкой означает, что модульный тест успешно пройден: ожидаемый и полученный результаты равны.

Изменим код метода **RectangleArea**, вычисляющего площадь прямоугольника, чтобы сымитировать провал теста и посмотреть, как поведёт себя Visual Studio. Прибавим к возвращаемому значению 10.

Запустим unit-тест.

[](https://vscode.ru/wp-content/uploads/2017/03/Provalennyiy-unit-test.png)

Как Вы видите, красный круг с крестиком показывает провал модульного теста, а ниже указано, что при проверке ожидалось значение 15, а по факту оно равно 25.