Практическое занятие № 5

Тема: «Разработка на основе тестирования с помощью обозревателя тестов»

Цель работы: получение практических навыков создания, модульных тестов, чтобы обеспечить правильную работу кода с помощью добавочных изменений кода.

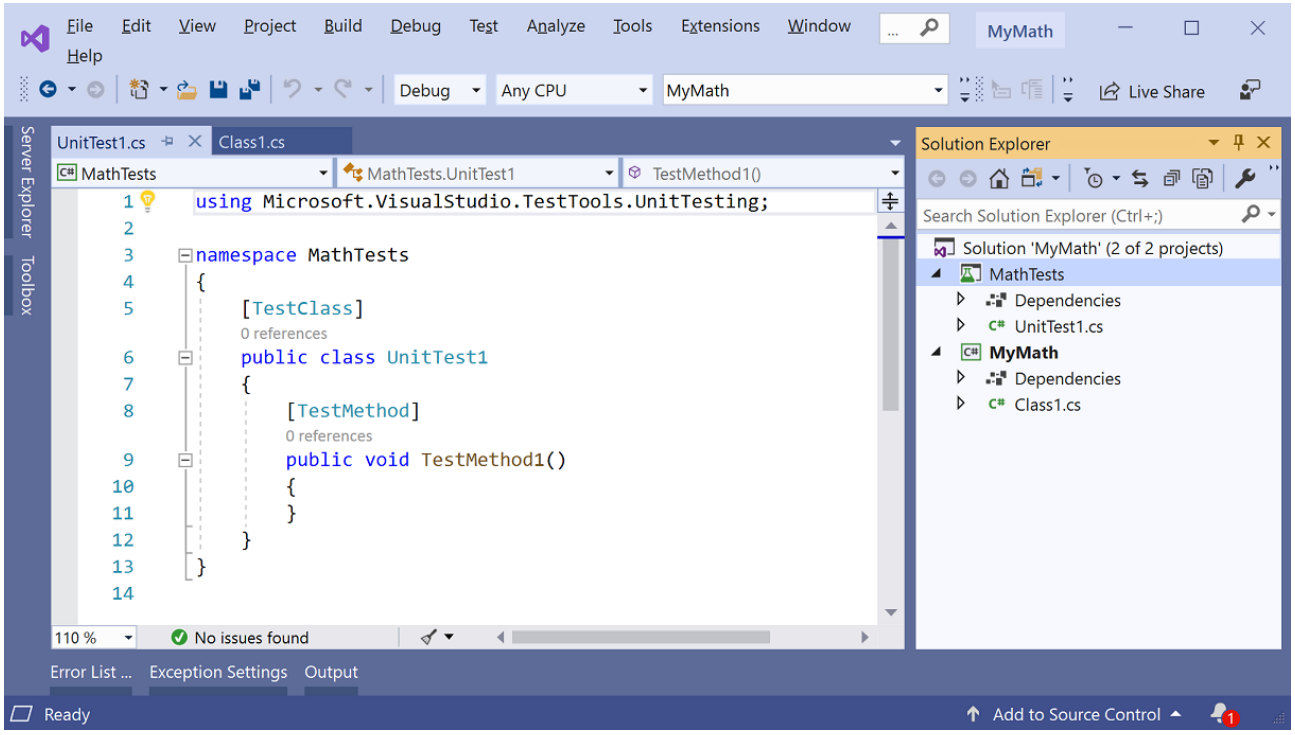
Краткие теоретические сведения

Существует несколько платформ, которые можно использовать для написания модульных тестов, в том числе разработанные третьими сторонами. Некоторые тестовые среды специализируются на тестировании на различных языках или платформах. Обозреватель тестов предоставляет единый интерфейс модульных тестов для любых таких платформ.

Создание теста и создание кода

1. Создайте проекта C# **Библиотека классов (.NET Standard)** . Данный проект будет содержать код, который мы хотим протестировать. Назовите проект **MyMath**.

2. В том же решении добавьте новый проект **Тестовый проект MSTest (.NET Core)** . Назовите тестовый проект **MathTests**.



3. Напишите простой метод теста, который проверяет результат, полученный для конкретных входных данных. Добавьте следующий код в класс UnitTest1 :

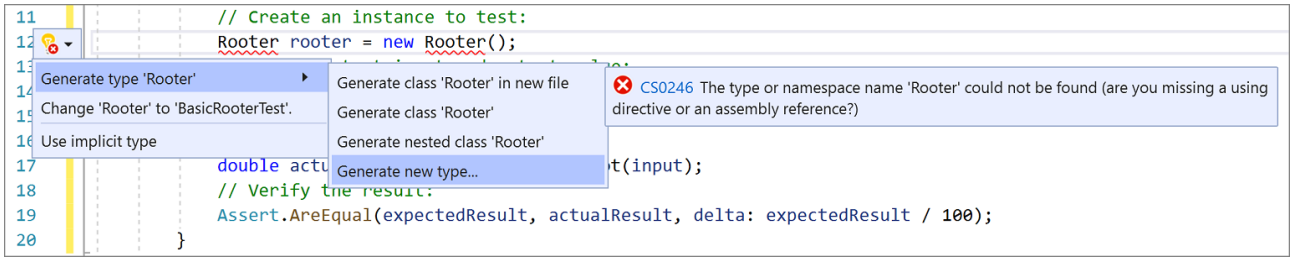
C#

```
[TestMethod]
public void BasicRooterTest()
{
  // Create an instance to test:
  Rooter rooter = new Rooter();
```

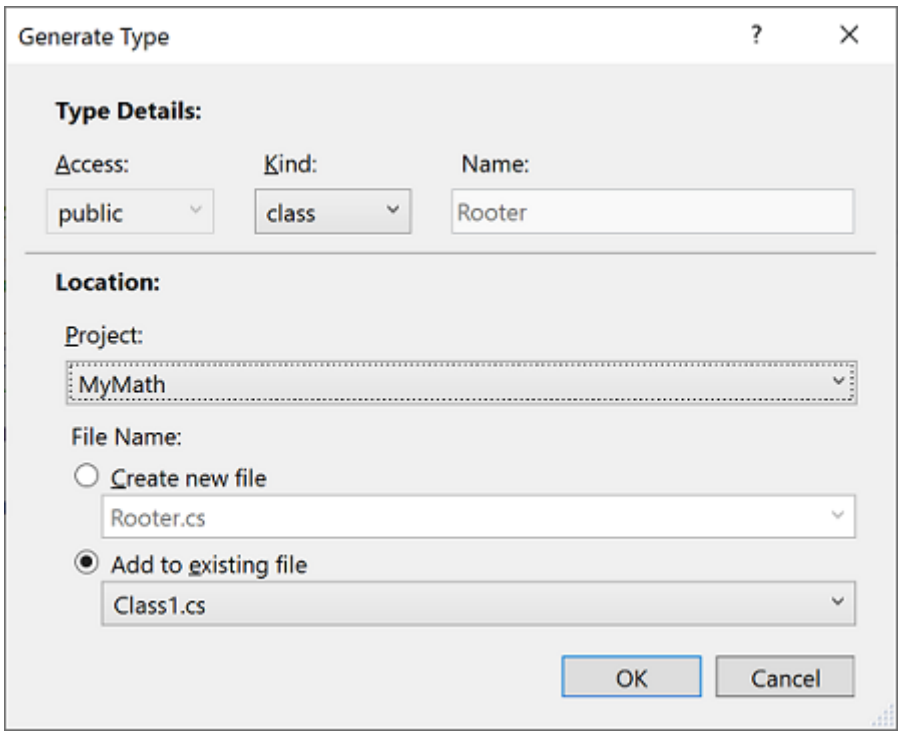
```
    // Define a test input and output value:
    double expectedResult = 2.0;
    double input = expectedResult * expectedResult;
    // Run the method under test:
    double actualResult = rooter.SquareRoot(input);
    // Verify the result:
    Assert.AreEqual(expectedResult, actualResult, delta: expectedResult / 100);
}
```

4. Создайте тип на основе кода теста.

    a. Установите курсор на Rooter, а затем в меню лампочки выберите **Создать тип "Rooter"** > **Создать новый тип**.



    b. В диалоговом окне **Создать тип** установите для параметра **Проект** значение **MyMath**, проект библиотеки классов, и нажмите **ОК**.



5. Создайте метод из кода теста. Установите курсор на SquareRoot, а затем в меню лампочки выберите **Создать метод Rooter.SquareRoot**.

6. Выполните модульный тест.

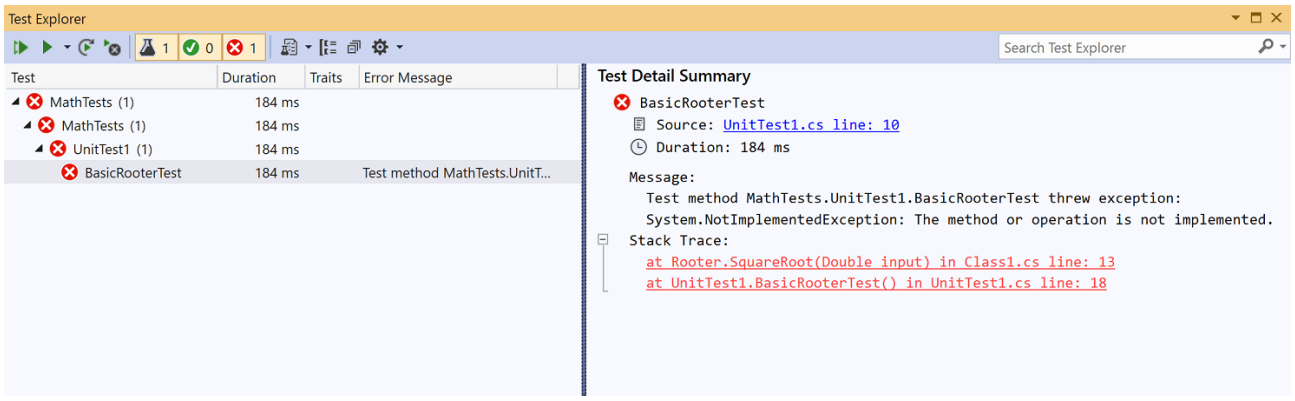
    a. Чтобы открыть **Обозреватель тестов**, в меню **Тест** выберите **Windows** > **Обозреватель тестов**.

b. В **обозревателе тестов** выберите **Запустить все**, чтобы запустить тест.

Выполняется сборка решения, тест запускается и завершается ошибкой.

7. Выберите имя теста.

Дополнительные сведения о тесте появятся на панели **Сводка теста**.



8. Перейдите по верхней ссылке в разделе **Трассировка стека**, чтобы перейти к расположению, в котором произошел сбой теста.

На данном этапе создан тест и заглушка, которые будут изменены таким образом, что тест будет успешно пройден.

Проверка изменения кода

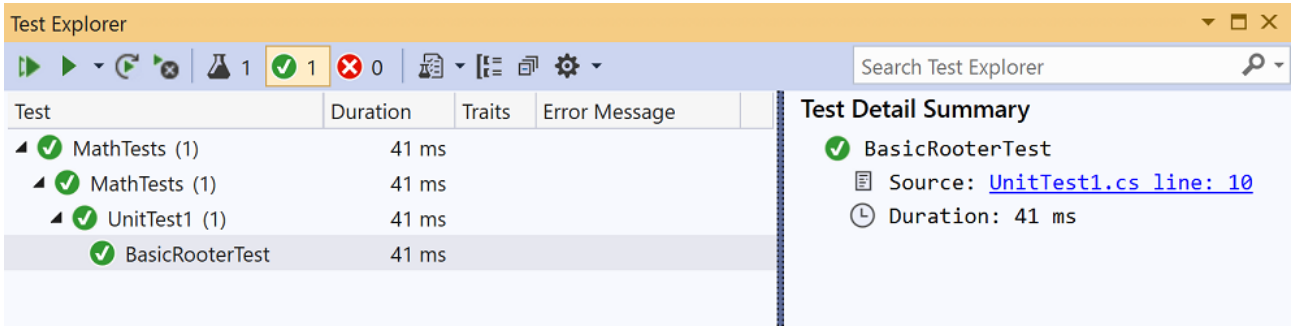
1. В файле *Class1.cs* улучшите код SquareRoot:

C#

```
public double SquareRoot(double input)
{
    return input / 2;
}
```

2. В обозревателе тестов выберите **Запустить все.**

Выполняется сборка решения, тест запускается и завершается успешно.



Расширение диапазона входных данных

Для уверенности, что код работает во всех случаях, добавьте тесты, которые используют более широкий диапазон входных значений.

Совет

Избегайте изменения существующих успешно выполненных тестов. Вместо этого добавьте новые тесты. Изменяйте существующие тесты только в тех случаях, когда меняются пользовательские требования. Такой подход позволяет не потерять существующие функциональные возможности при работе с расширенным кодом.

1. В тестовом классе добавьте следующий тест, который использует диапазон входных значений:

C#

```
[TestMethod]
public void RooterValueRange()
{
    // Create an instance to test.
    Rooter rooter = new Rooter();

    // Try a range of values.
    for (double expected = 1e-8; expected < 1e+8; expected *= 3.2)
    {
        RooterOneValue(rooter, expected);
    }
}

private void RooterOneValue(Rooter rooter, double expectedResult)
{
    double input = expectedResult * expectedResult;
    double actualResult = rooter.SquareRoot(input);
    Assert.AreEqual(expectedResult, actualResult, delta: expectedResult /
1000);
}
```

2. В **обозревателе тестов** выберите **Запустить все**.

Новый тест завершается неудачей (несмотря на то, что первый тест по-прежнему завершается успешно). Чтобы найти точку сбоя, выберите тест, который не пройден, и просмотрите сведения на панели **Сводка теста**.

3. Проверьте тестируемый метод, чтобы узнать, что не так. Измените код SquareRoot, как показано:

C#

```
public double SquareRoot(double input)
{
  double result = input;
  double previousResult = -input;
  while (Math.Abs(previousResult - result) > result / 1000)
  {
    previousResult = result;
    result = result - (result * result - input) / (2 * result);
  }
  return result;
}
```

4. В **обозревателе тестов** выберите **Запустить все**.

Теперь оба теста завершаются успешно.

Добавление тестов для исключительных случаев

1. Добавьте новый тест для отрицательных входных значений:

C#

```
[TestMethod]
public void RooterTestNegativeInputx()
{
    Rooter rooter = new Rooter();
    try
    {
        rooter.SquareRoot(-10);
    }
    catch (System.ArgumentOutOfRangeException)
    {
        return;
    }
    Assert.Fail();
}
```

2. В **обозревателе тестов** выберите **Запустить все**.

Тестируемый метод зацикливается, его необходимо отменить вручную.

3. На панели инструментов **обозревателя тестов** нажмите **Отмена**.

Выполнение теста останавливается.

4. Исправьте код SquareRoot, добавив следующую инструкцию if в начало метода:

C#

```
public double SquareRoot(double input)
{
    if (input <= 0.0)
    {
        throw new ArgumentOutOfRangeException();
    }
    ...
```

5. В **обозревателе тестов** выберите **Запустить все**.

Все тесты завершаются успешно.

Рефакторинг тестируемого кода

Выполните рефакторинг кода, но не изменяйте тесты.

Совет

Рефакторинг — это изменение, которое делает код более производительным или более понятным. Это действие не предназначено для изменения поведения кода, поэтому тесты не изменяются.

Рекомендуется выполнять рефакторинг отдельно от расширения функциональности. Неизменяемость тестов уменьшает шансы случайных ошибок во время рефакторинга.

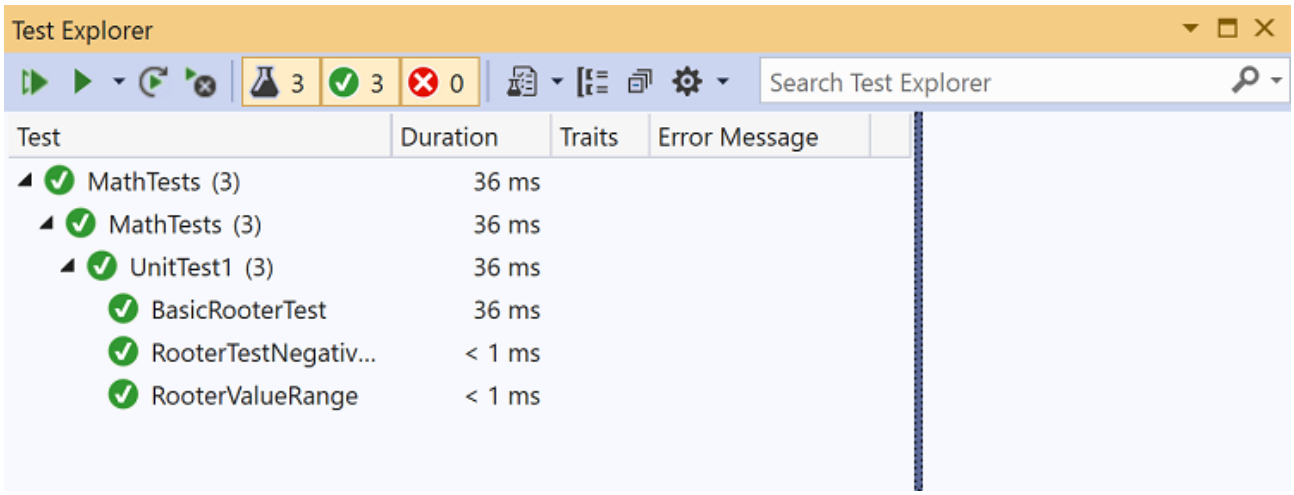
1. Измените строку, которая вычисляет result в методе SquareRoot, следующим образом:

C#

```
public double SquareRoot(double input)
{
    if (input <= 0.0)
    {
        throw new ArgumentOutOfRangeException();
    }

    double result = input;
    double previousResult = -input;
    while (Math.Abs(previousResult - result) > result / 1000)
    {
        previousResult = result;
        result = (result + input / result) / 2;
        //was: result = result - (result * result - input) / (2*result);
    }
    return result;
}
```

2. Выберите **Выполнить все** и убедитесь, что все тесты по-прежнему завершаются успехом.



Оборудование, материалы

Персональный компьютер с установленной ОС. Текстовый редактор. MS Visual Studio.

Порядок выполнения задания

1. Изучить краткие теоретические сведения.
2. Выполнить приведённые в тексте задания.

Источники

https://docs.microsoft.com/ru-ru/visualstudio/test/quick-start-test-driven-development-with-test-explorer?view=vs-2017