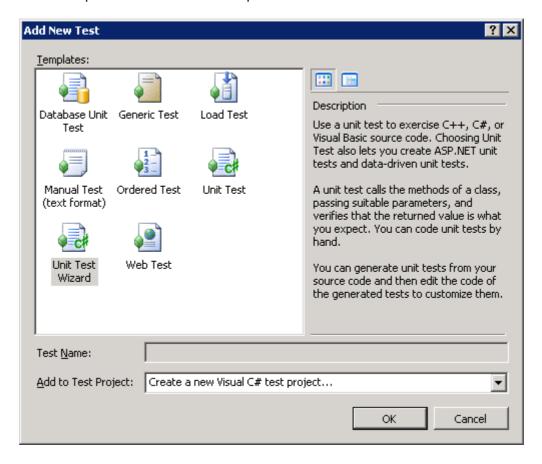
Лабораторная работа 4. Разработка модульных тестов.

На данном занятии команды должны разработать набор модульных тестов, покрывающих функциональность, разработанную на занятии предыдущем. В рамках данного занятие предполагается освоить следующие возможности MS VSTS.

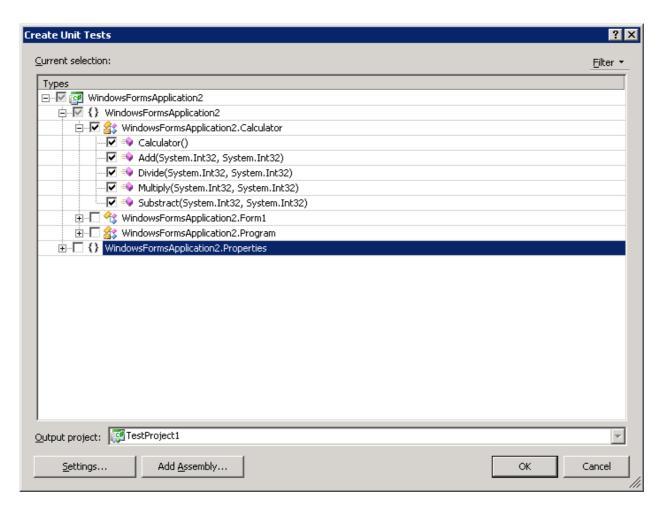
- 1. Автоматическая генерация тестов.
- 2. Наполнение тестов содержимым.
- 3. Запуск тестов и просмотр результатом.
- 4. Изменение конфигурации работы тестов.

Шаг 1. Автоматическая генерация тестов.

Для ускорения разработки тестов команды могут воспользоваться возможность Visual Studio по автоматической генерации тестов. Для этого необходимо воспользоваться командой Test/New Test и выбрать Unit Test wizard в открывшиеся окне:



После создания тестового проекта, будет предложен выбор из тех типов и методов, для тестирования которых необходимо создать заглушки:



В этом диалоге команда должна выбрать все основные классы и методы, которые планируется покрыть модульными тестами.

Шаг 2. Наполнение тестов содержимым.

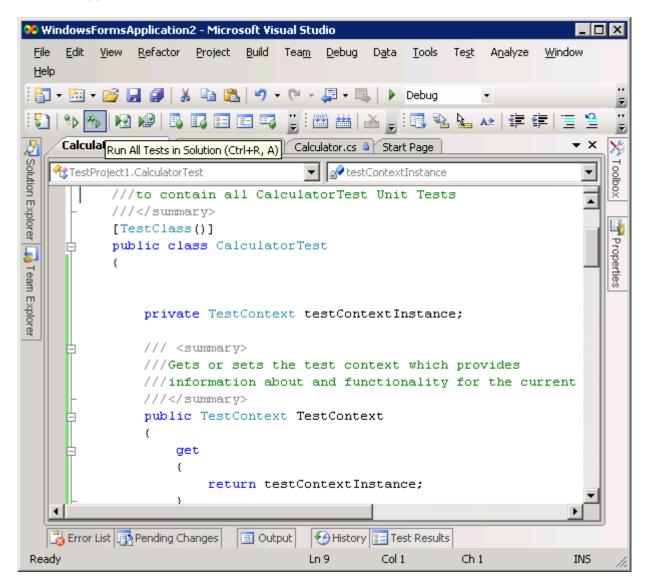
После генерации тестового покрытия команда получить набор скелетов тестов для всех методов, которые были выбраны для тестирования. Однако, эти тесты имеют достаточно простую структуру и пока лишены смысла:

```
/// <summary>
///A test for Multiply
///</summary>
[TestMethod()]
public void MultiplyTest()
{
    // TODO: Initialize to an appropriate value
    Calculator target = new Calculator();
    int a = 0; // TODO: Initialize to an appropriate value
    int b = 0; // TODO: Initialize to an appropriate value
    int expected = 0; // TODO: Initialize to an appropriate value
    int actual;
    actual = target.Multiply(a, b);
    Assert.AreEqual(expected, actual);
    Assert.Inconclusive("Verify the correctness of this test
    method.");
}
```

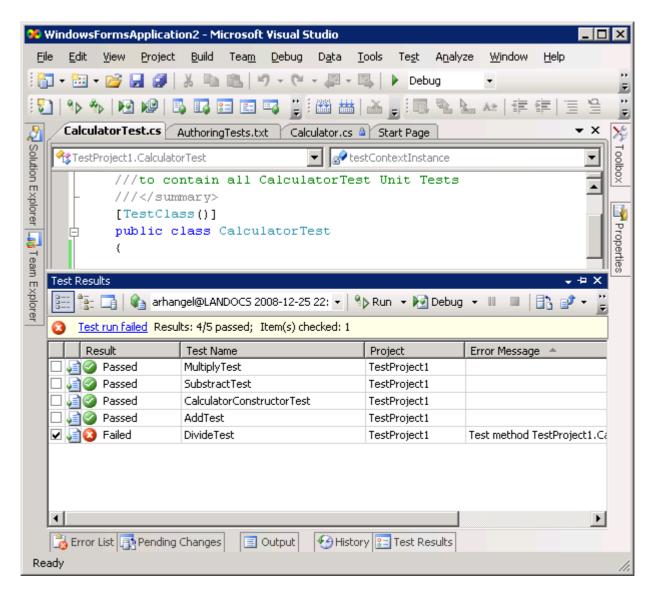
На следующем шаге команда должна заполнить эти тесты необходимым содержимым, используя функциональность по валидации (Assert), предоставляемую тестовой платформой.

Шаг 3. Запуск тестов

Для того, чтобы исполнить созданные тесты необходимо использовать соответствующую панель инструментов:



После этого результаты выполнения тестов будут видны в окне результатов:

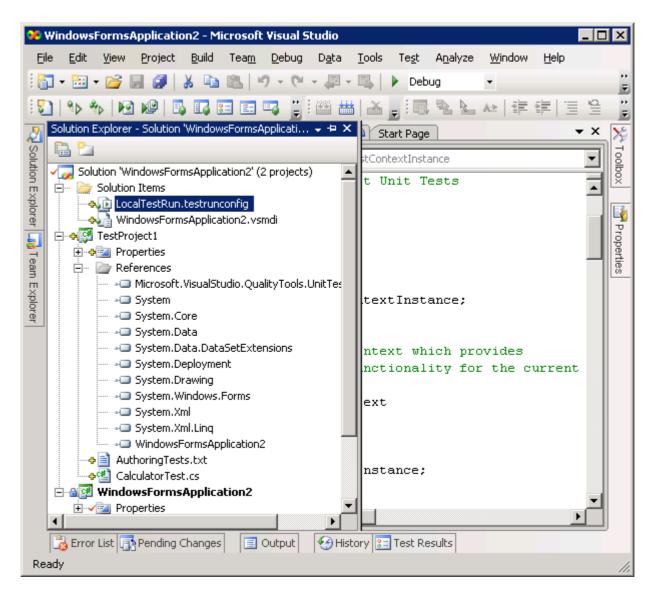


Команды должны добиться того, чтобы все разработанные тесты проходили успешно.

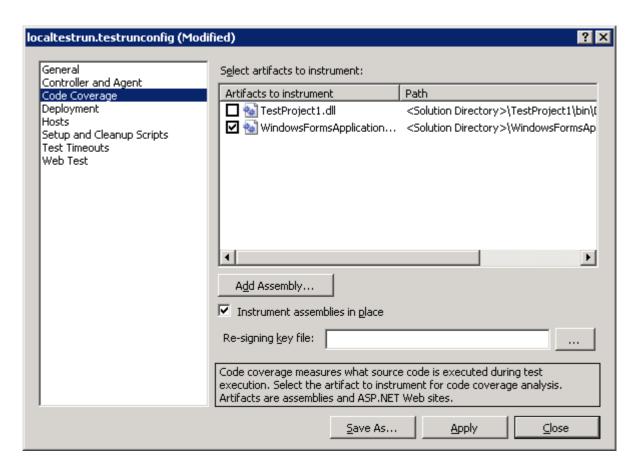
Шаг 4. Изменение конфигурации тестов.

Для того, чтобы проанализировать качество разработанных тестов, команда должна вычислить тестовое покрытие. Для этого её необходимо:

1. Открыть файл конфигурации запуска тестов, автоматически добавленный к решению при создании тестов:



2. На открывшемся диалоге выбрать вкладку Code Coverage и установить то, какие именно проекты нужно анализировать:



3. После сохранения конфигурации запустить тесты и активировать опцию Show Code Coverage Coloring:

```
WindowsFormsApplication2 - Microsoft Visual Studio
 <u>File Edit View Refactor Project Build Team Debug Data Tools Test Analyze Window Help</u>
🏥 - 🛅 - 🚰 - 🖟 🕽 🖟 🎒 🧸 🖺 🖺 🕒 - 🕮 - 📮 - 🗒 🕨 Debug
                                                                      ▼ Any CPU
| 🛂 | 🦫 🍨 | 🗠 🕬 | 🖺 🖫 🔠 📰 🗐
      CalculatorTest.cs AuthoringTests.txt Calculator.cs
                                                                                                ▼ X
                                           Show Code Coverage Coloring
Solution Explorer Light Team Explorer
    stration 2. Calculator stration 2. Calculator
                                                 Add(int a, int b)
                                                                                                 ▾
      □ using System;
        using System.Collections.Generic;
                                                                                                     Properties
        using System.Linq;
       └using System.Text;
      □ namespace WindowsFormsApplication2
             class Calculator
      白
                  public int Add(int a, int b)
                       return a + b;
                  public int Divide(int a, int b)
                       return a / b;
                  public int Multiply(int a, int b)
    🚡 Error List 🚮 Pending Changes
                                           History = Test Results
                                ☐ Output
                                                                  Col 18
                                                                               Ch 18
                                                                                                INS
```