Национальный Исследовательский Университет

Московский Энергетический Институт

Кафедра Прикладной математики

Отчет по лабораторной работе № 14

по дисциплине

«CASE-технологии разработки программных средств»

на тему

«Применение технологии модульного тестирования»

Выполнила: Шутова Екатерина

Группа: А-13-07

Преподаватель: Куриленко И.Е.

Москва, 2012

Цель работы

Изучить средства автоматизации тестирования программных модулей в Microsoft Visual Studio 11. Применить на практике инструменты модульного тестирования на базе MSUnit.

Реализация

Методы для тестирования

```
public double Sin(double x)
{
   return Math.Sin(x);
}
public string GetTypeAnimal(int x)
{
   switch (x)
   {
      case 1:
      {
         return "Dog";
      }
      case 2:
      {
         return "Cat";
      }
      default:
      {
         throw new KeyNotFoundException("Key not found!");
      }
    }
}
public bool GetAnswer(int x)
        {
            string tmp = x.ToString();
            if (tmp.Length < 3)
            {
                return true;
            }
            else
            {
                return false;
            }
        }
```

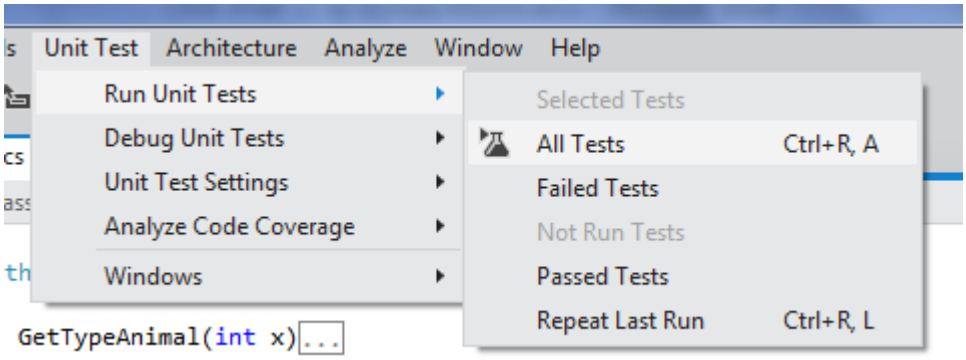
Тестовые методы

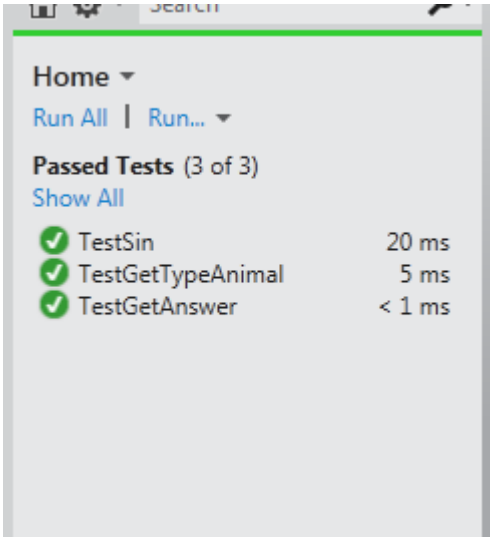
```
[TestMethod]
        public void TestSin()
        {
            ClassForTest cl = new ClassForTest();
            Assert.AreEqual(0, cl.Sin(0), "Test ok");
            Assert.AreNotEqual(0, cl.Sin(5), "Test ok");
        }
        [TestMethod]
        [ExpectedException(typeof(KeyNotFoundException))]
        public void TestGetTypeAnimal()
        {
            ClassForTest cl = new ClassForTest();
            Assert.AreEqual("Dog", cl.GetTypeAnimal(1), "Test ok");
            cl.GetTypeAnimal(0);
        }

        [TestMethod]
```

```
public void TestGetAnswer()
{
    ClassForTest cl = new ClassForTest();
    Assert.IsFalse(cl.GetAnswer(34567));
}
```

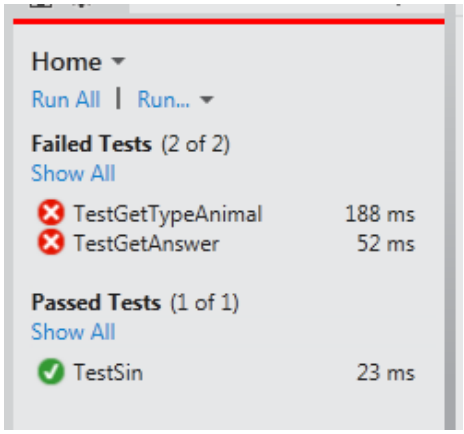
Запуск тестов





Тесты прошли успешно.

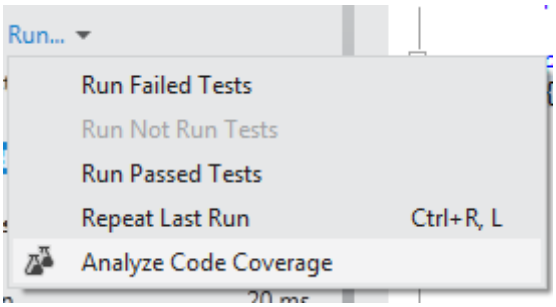
Если в тестовом методе TestGetTypeAnimal() убрать ожидаемую ошибку, то тест должен не пройти. Также проверим метод TestGetAnswer(), введя число 34.



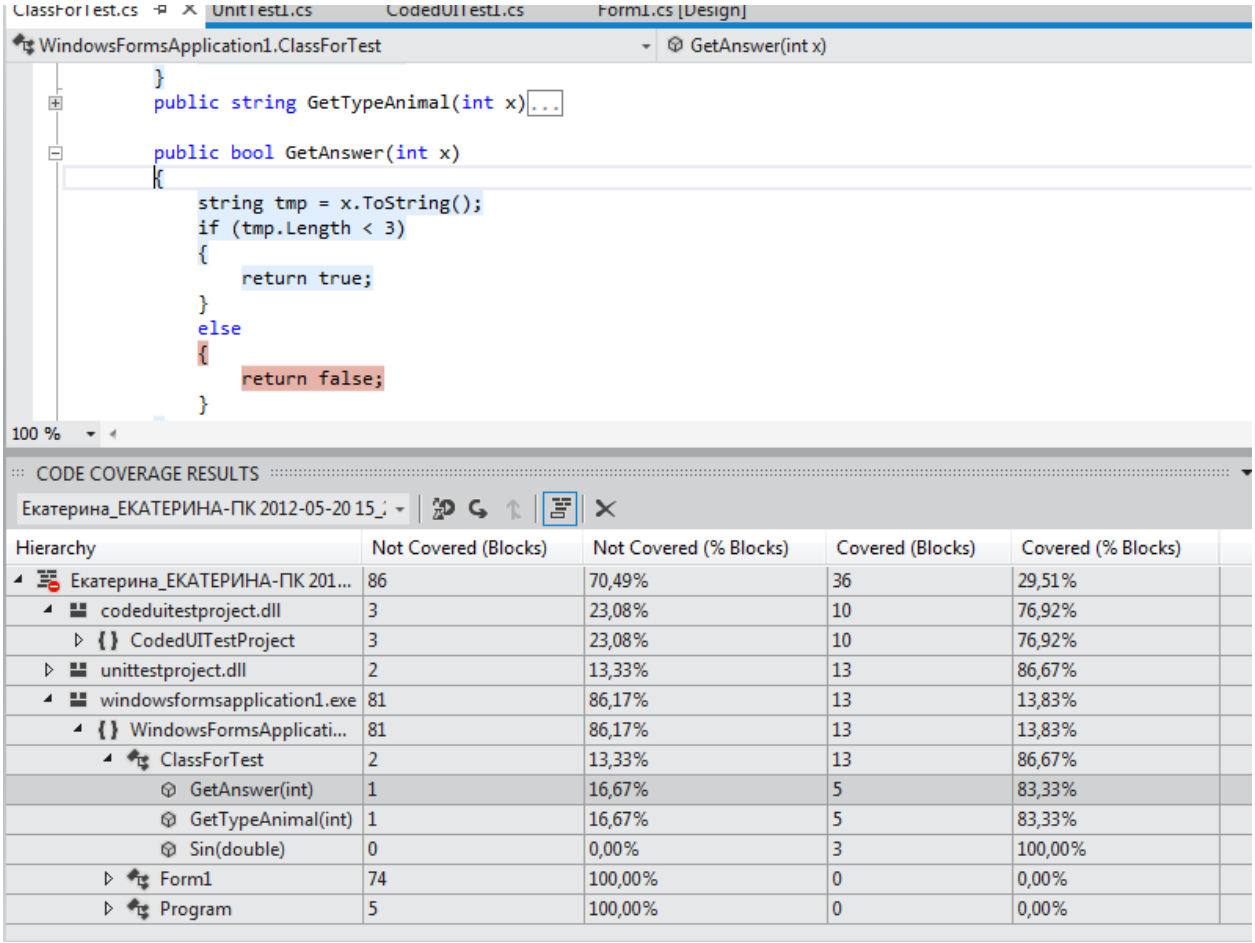
Как и должно было быть, измененные тесты не прошли.

Покрытие кода тестами

В среде есть возможность анализа того, на сколько код покрыт тестами. Для этого в Unit Test Explorer нужно нажать на стрелку рядом с элементом Run и выбрать пункт Analyze Code Coverage.



Результат будет отображаться в виде иерархического списка, для каждого элемента которого можно посмотреть процент покрытия.



Hierarchy	Not Covered (Blocks)	Not Covered (% Blocks)	Covered (Blocks)	Covered (% Blocks)
Екатерина_ЕКАТЕРИНА-ПК 201...	86	70,49%	36	29,51%
codeduitestproject.dll	3	23,08%	10	76,92%
CodedUITestProject	3	23,08%	10	76,92%
unittestproject.dll	2	13,33%	13	86,67%
windowsformsapplication1.exe	81	86,17%	13	13,83%
WindowsFormsApplicati...	81	86,17%	13	13,83%
ClassForTest	2	13,33%	13	86,67%
GetAnswer(int)	1	16,67%	5	83,33%
GetTypeAnimal(int)	1	16,67%	5	83,33%
Sin(double)	0	0,00%	3	100,00%
Form1	74	100,00%	0	0,00%
Program	5	100,00%	0	0,00%

На рисунке выше видно, что среда подсказывает разработчику, что ветка else тестами не покрыта вообще.