*Практическая работа #1. Создание различных методов тестирования для проверки корректности работы программы.*

*Часть 1. Краткие теоретические сведения*

*Методы тестирования различаются по разным признакам. По знанию системы выделяют тестирование черного ящика и тестирование белого ящика.*

*При тестировании чёрного ящика, инженер по тестированию имеет доступ к ПО только через те же интерфейсы, что и заказчик или пользователь, либо через внешние интерфейсы, позволяющие другому компьютеру либо другому процессу подключиться к системе для тестирования. Например, тестирующий модуль может виртуально нажимать клавиши или кнопки мыши в тестируемой программе с помощью механизма взаимодействия процессов, с уверенностью в том, все ли идёт правильно, что эти события вызывают тот же отклик, что и реальные нажатия клавиш и кнопок мыши. Как правило, тестирование чёрного ящика ведётся с использованием спецификаций или иных документов, описывающих требования к системе.*

*При тестировании белого ящика (англ. white-boxtesting, также говорят —прозрачного ящика), разработчик теста имеет доступ к исходному коду программ и может писать код, который связан с библиотеками тестируемого ПО. Это типично для юнит-тестирования (англ. unittesting), при котором тестируются только отдельные части системы. Оно обеспечивает то, что компоненты конструкции —работоспособны и устойчивы, до определённой степени. При тестировании белого ящика используются метрики покрытия кода.*

*Сценарий тестирования (Testcase, тест кейс) - это один из основных документов, с которыми имеет дело тестировщик. По сути, упрощенное описание теста. То есть входной информации, условий и последовательности выполнения действий и ожидаемого выходного результата. Учитывая, что даже успешно прошедшие тесты выполняются неоднократно в ходе регрессионного тестирования, наличие таких описаний необходимо. Однако уровень формальных требований к их оформлению может меняться в очень широких пределах. Одно дело, если вы собираетесь использовать тесты в ходе приемочных испытаний, проводимых заказчиком, и другое — в ходе внутреннего тестирования коробочного продукта. Ниже представлен шаблон для составления сценариев тестирования.*

*Часть 2. Создание Test Case для программы.*

## Общая информация о тестировании

|  |  |
| --- | --- |
| **Название проекта** | Project.dpr |
| **Номер версии** | Med1.0 |
| **Имя тестера** | Иванов И.И. |
| **Даты тестирования** | **29.11.2018** |

## Testcase #1: (Пример)

|  |  |
| --- | --- |
| **TestCase #** | 1 |
| **Приоритет теста** | высокий |
| **Название тестирования/Имя** | Проверка функции авторизации при верном логине и пароле |
| **Резюме испытания** | В окне авторизации вводим верный логин и пароль и переходим в личный кабинет пользователя |
| **Шаги тестирования** | 1) в поле «Логин» ввести имя пользователя;  2) в поле «Пароль» ввести пароль пользователя;  3) нажать на кнопку «Вход» |
| **Данные тестирования** | 1) «Логин» = Иванов  2) «Пароль» = 1 |
| **Ожидаемый результат** | Открывается форма «Личный кабинет». В верхнем правом углу отображается Логин пользователя |
| **Фактический результат** | Открылась форма «Личный кабинет». В верхнем правом углу отображается Логин пользователя |
| **Предпосылки** | В базе данных в таблице user должна быть запись с данными тестирования |
| **Постусловия** | Активным является окно «Личный кабинет», окно авторизации закрыто |
| **Статус (Pass/Fail)** | pass |
| **Комментарии** | При вводе логина регистр можно не учитывать |

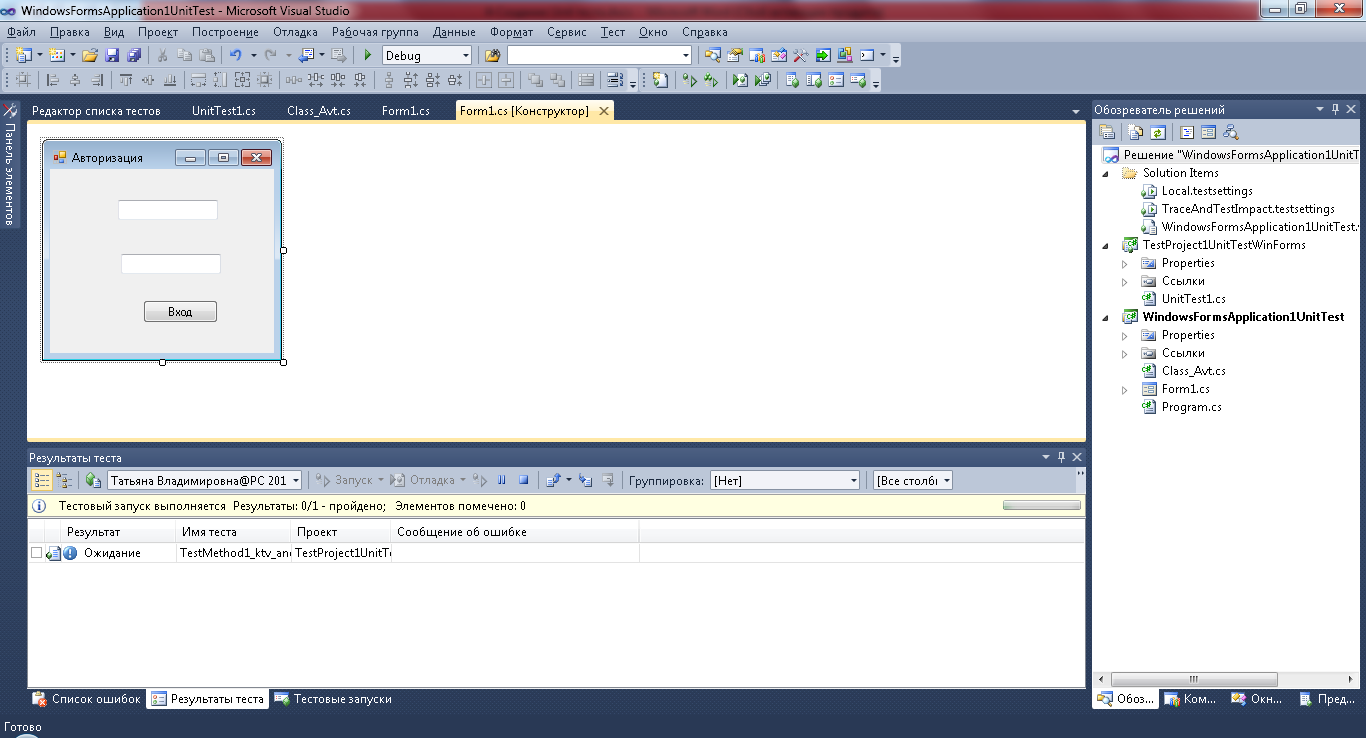
## Описание информационных полей для тестирования

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| **Наименование проекта** | Наименование проекта проверено |
| **Номер версии** | Версия проекта (первый номер можно принять как 1.0) |
| **Имя тестера** | Имя тестера, который выполнял эти тесты |
| **Даты тестирования** | Даты когда проводили тестирование – это может быть один тест или несколько. Если тесты проводили через большие промежутки времени, дата тестирования может определятся отдельными тест кейсами |
| **Test Case #** | Уникальный ID для каждого testcase.Следуйте опредленной логике именования и нумерации. Например ‘TC\_UI\_1′ указание на ‘пользовательский интерфейс testcase #1′. |
| **Приоритет тестирования** *(Малый/Средний/высокий)* | Насколько важен каждый тест. Приоритет при испытании бизнес-правил или функционала может быть средним или высоким, в то время как незначительные формы пользовательского интерфейса могут быть с низким приоритетом. |
| **Названиетестирования/Имя** | Название тестирования. Например, проверка формы авторизации с правильным логином и паролем. |
| **Резюме испытания** | Описание, чего нужно достигнуть при тестировании. |
| **Шаги тестирования** | Перечислите детально все шаги тестирования. Напишите в каком порядке должны быть выполнены эти шаги. Убедитесь что вы обеспечили настолько максимальную детализацию насколько можете. Нумерованный список – будет хорошей идей |
| **Данные тестирования** | Напишите тестовые данные используемые для этого тестирования. Таким образом актуальные данные, которые будут предложены будут использоваться для проведения тестирования. Например логин и пароль – для входа в систему. |
| **Ожидаемый результат** | Какой должен получится результат после выполнения теста? Опишите подробно ожидаемый результат включая любые сообщения и ошибки, которые должны быть выданы на экран. |
| **Фактический результат** | Какой фактический результат после выполнения теста? Опишите любое соответствующее поведение системы после выполнения тестирования. |
| **Предпосылки** | Любые предварительные действия, которые должны быть выполнены перед проведением тестирования. Перечислите предварительные условия, для успешного выполнения проекта |
| **Постусловия** | Какое состояние должно быть у системы после выполнения тестирования? |
| **Статус** *(Pass/Fail)* | Еслифактическийрезультатнесоответствуетожидаемымрезультатамотметка, что тест провалился (fail). В противном случае как прошло (pass) |
| **Комментарии** | Используйте эту область для любых дополнительных записей или комментариев. Это область нужна для поддержки полей выше (например есть какие-то особые условия, которые не могут быть описаны ни в одном из полей или есть вопросы связанные с ожидаемыми или фактическими результатами) |

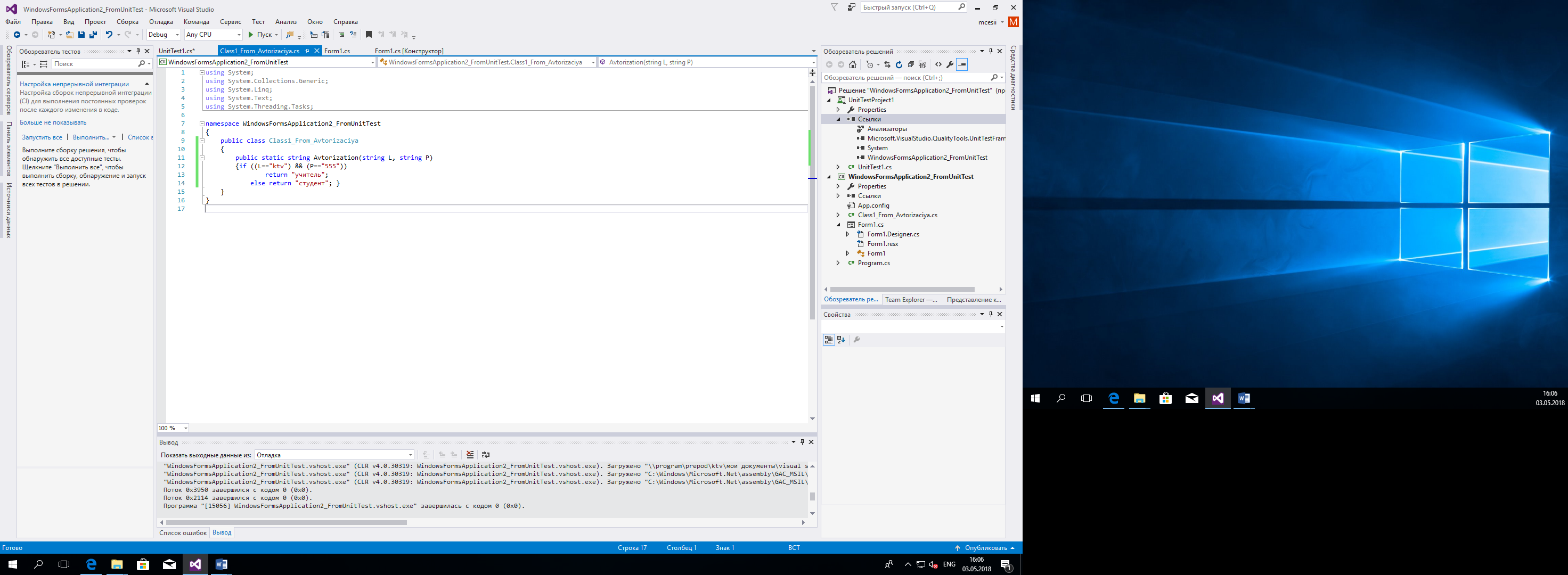
Напишите два сценария тестирования методом тестирования черного ящика для программы Test (программа находится в папке «Ресурсы»).

*Часть 2. Создание unit тестов для программы типа Windows Forms*

1. Создайте новый проект типа Windows Forms;
2. Спроектируйте внешний вид главной формы как показано на скриншоте:

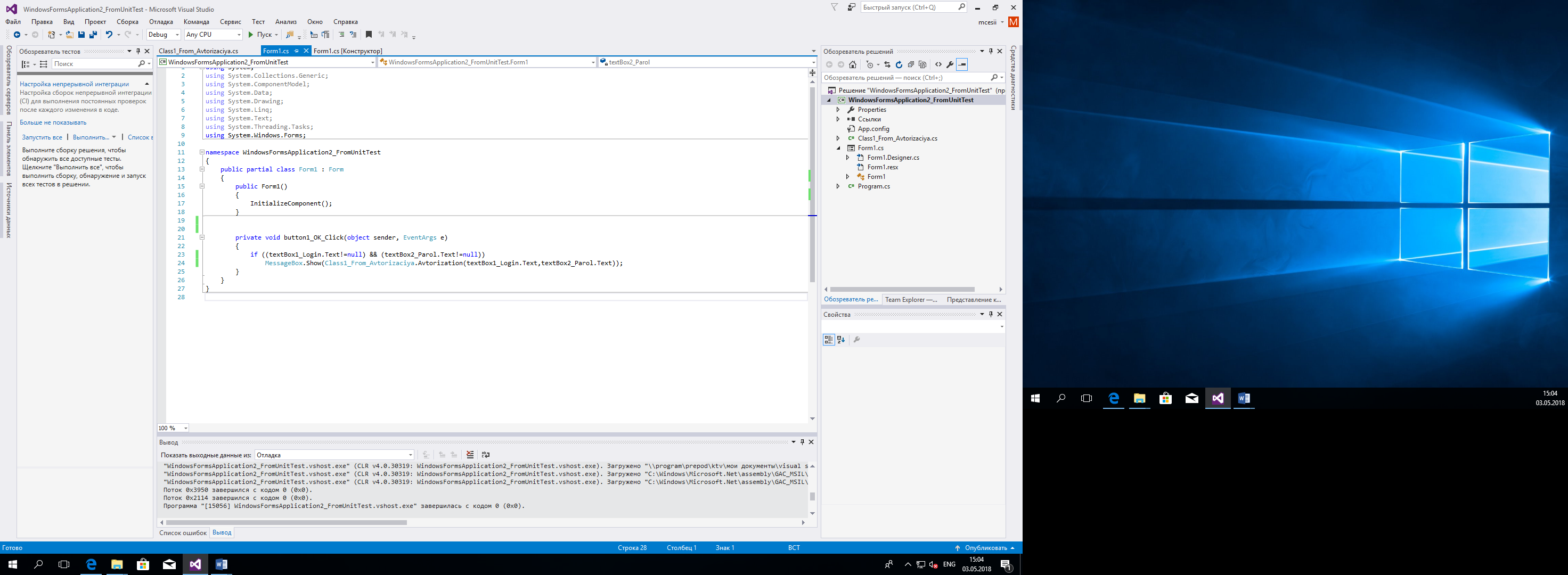


1. Создадим свой класс (ПКМ по названию проекта – Добавить – Класс) и пропишем в него метод авторизации:

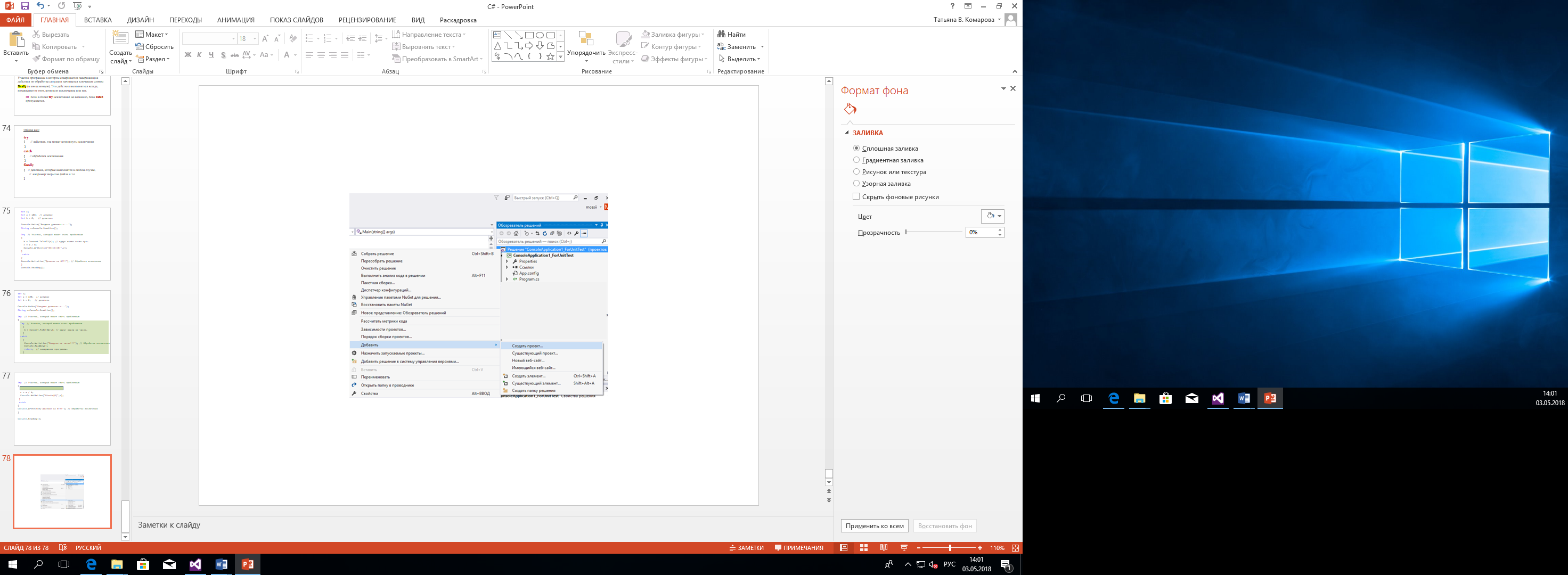


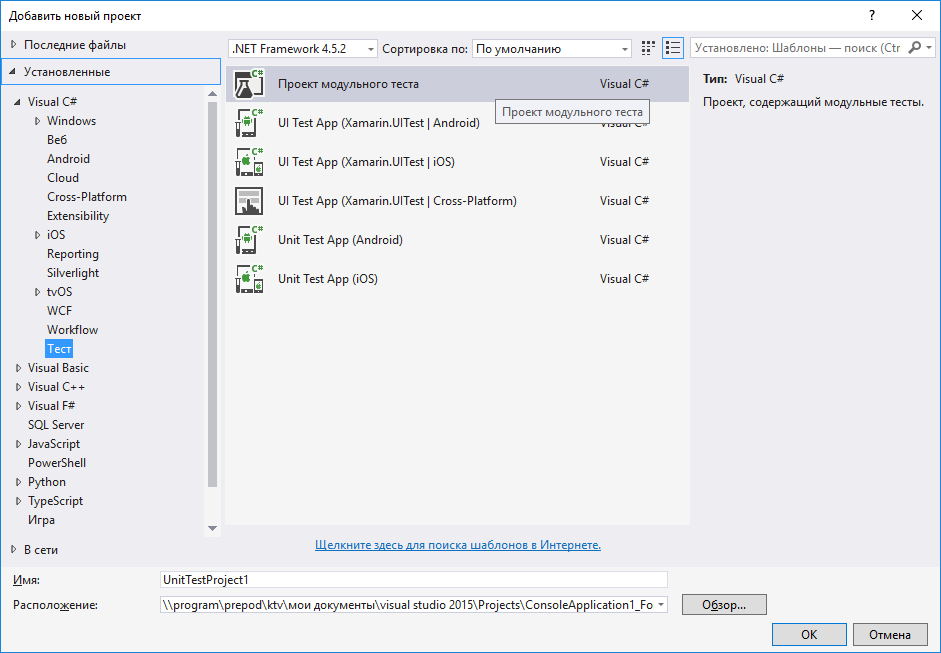
ВАЖНО: при разработке unit тестов для программы ОБЯЗАТЕЛЬНО указывается идентификатор доступа public для тех методов и классов, которые мы хотим тестировать.

1. Напишем программный код для кнопки «Вход»:

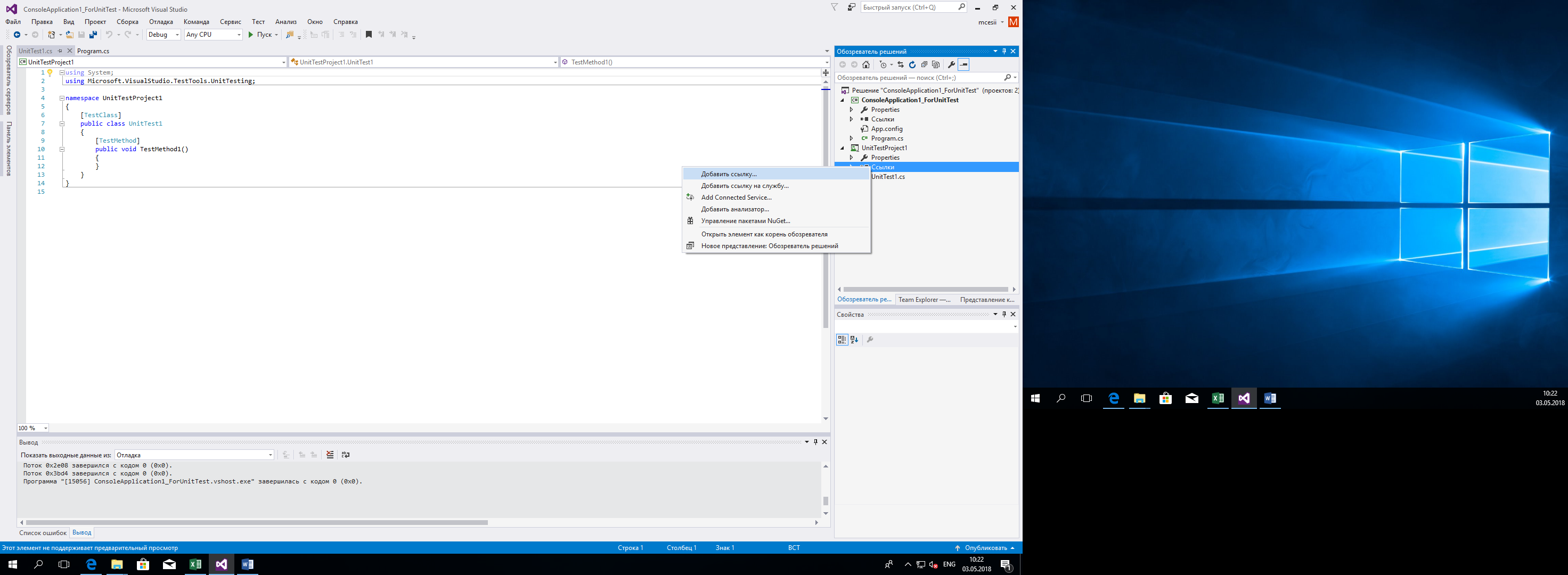


1. Добавим к РЕШЕНИЮ новый проект для написания unit тестов:

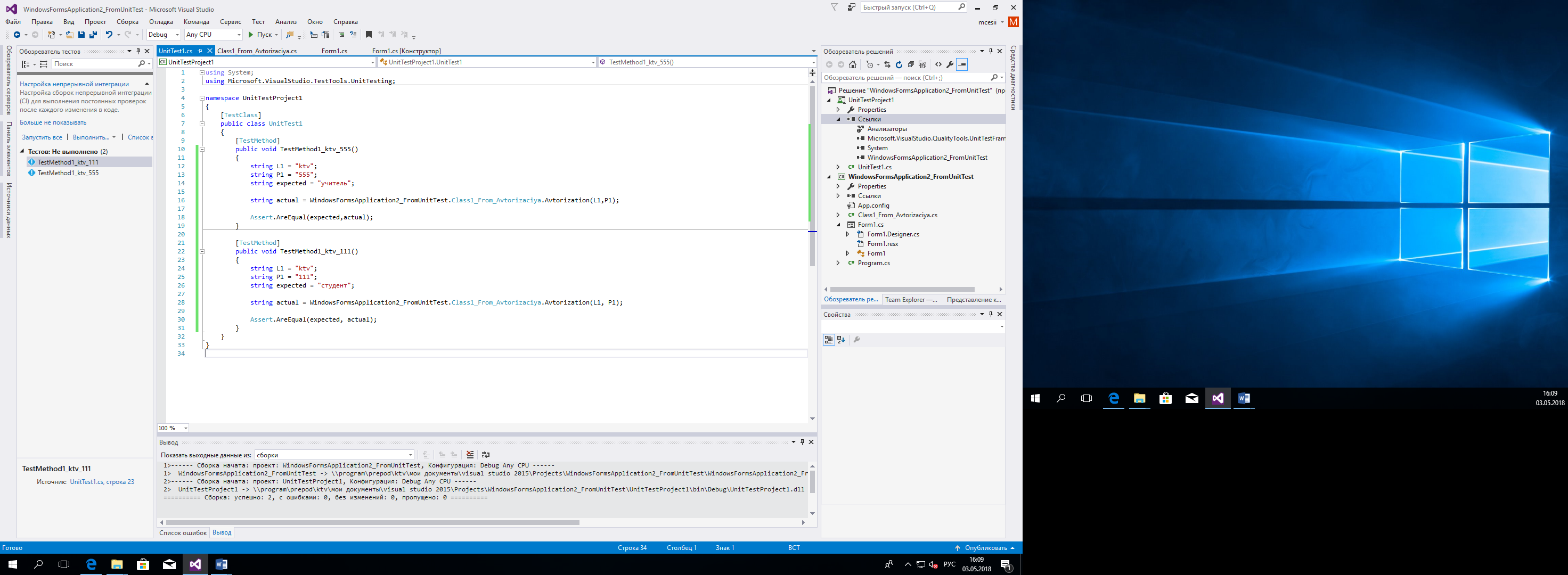




В ТЕСТОВОМ проекте надо добавить ссылку на проект, для которого будут созданы unit тесты. Для этого в ОБОЗРЕВАТЕЛЕ РЕШЕНИЙ в проекте UnitTestProject1 к имеющимся ссылкам добавить новую ссылку через контекстное меню. И выбрать имя тестируемого проекта.

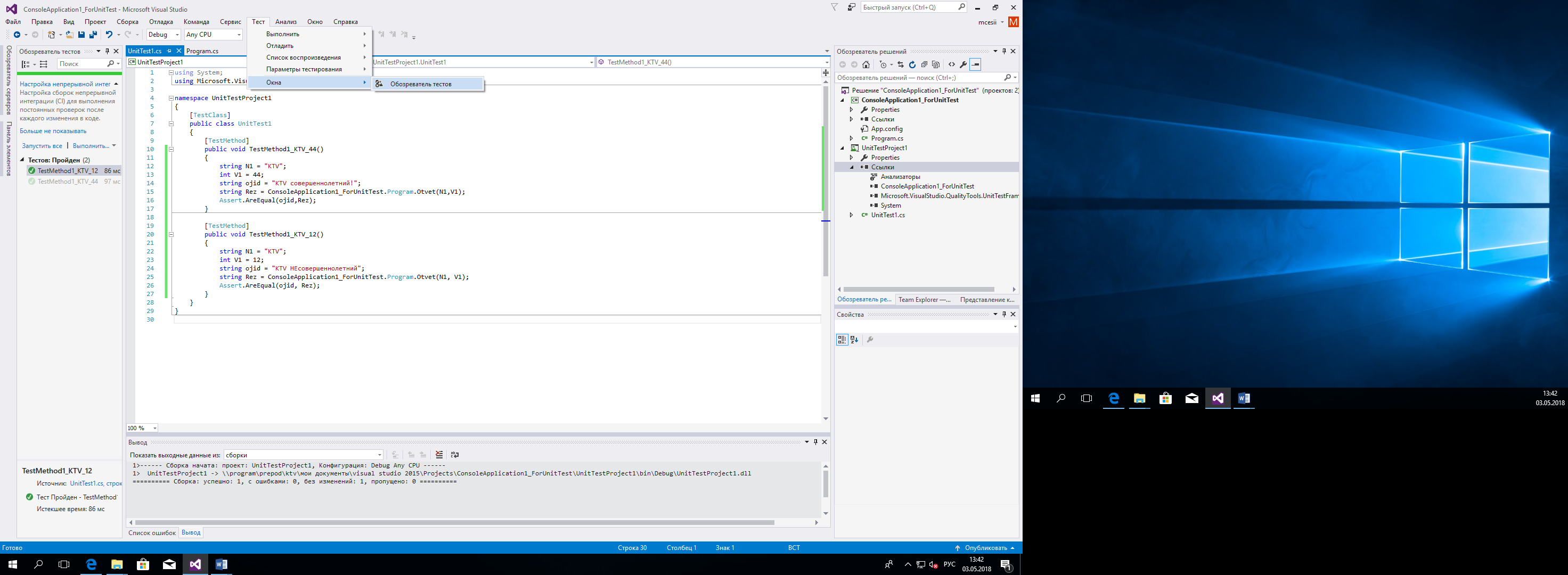


1. Откроем наш проект UnitTestProject1 и создадим два теста:



После написания тестов необходимо их запустить и проверить правильность выполнения.

1. Откроем «Обозреватель тестов»:



1. Выберем пункт меню «Сборка – Собрать решение». После этого все созданные тесты появятся в окне «Обозреватель тестов»;
2. Выберем любой тест и через контекстное меню выберем команду «Запустить выбранные тесты».

