ReSharper: Code snippets for tests.

Прочитать статью на сайте

Мне всегда было лень набирать код полными словами, писать какие-то стандартные формулировки, это просто выводит из себя или навевает вселенскую тоску. Если раньше все это было не так печально, то за последние 2 года пользования решарпером, все усугубилось настолько, что я даже public никогда полностью не наберу, а ограничусь набором “p” и нажатием пробела - ReSharper допишет слово до конца. Единственное, что я пишу полностью, так это названия методов, классов и тестов. Заметьте что только названия, а не полное объявление. Да, и переменные за меня называет тоже ReSharper. =) Вот такой я лентяй!

Хочу поделиться с вами знанием, как развить в себе такое же лентяйство и как в этом помогает ReSharper. Т.к. мы придерживаемся Test Driven Development, то и заготовки кода (code snippets) будут относиться к тестам.

Первое, с чего начинается любой тест, это объявление тестового класса.

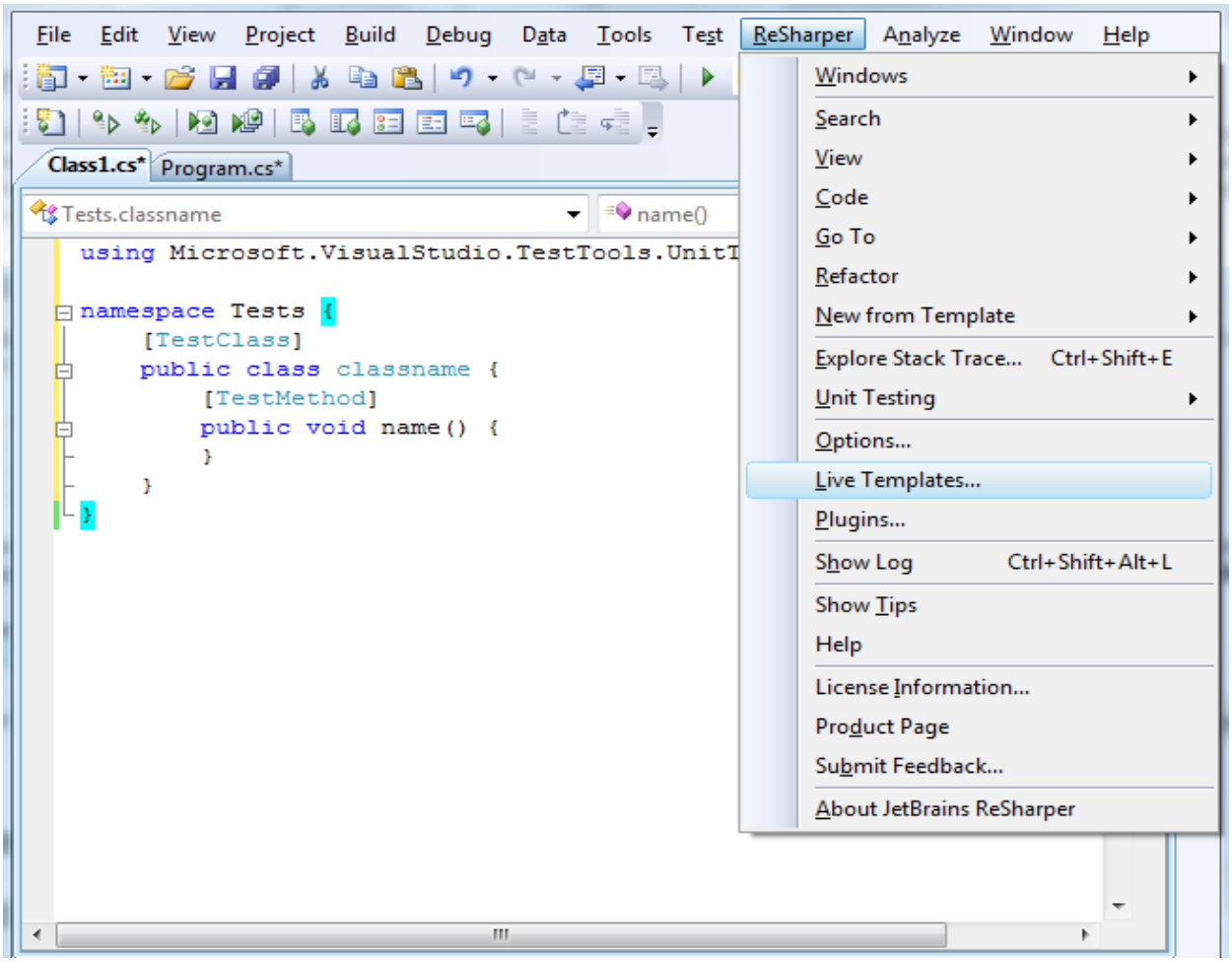
```
using Microsoft.VisualStudio.TestTools.UnitTesting;

    [TestClass]
    public class classname { }
```

Кроме названия тестового класса, всё всегда одно и то же. Я физически страдаю даже от созерцания того, как это пишут другие. В информации о себе я упоминал, что на работе мы практикуем XP и, соответственно, парное программирование как неотъемлемая его часть.

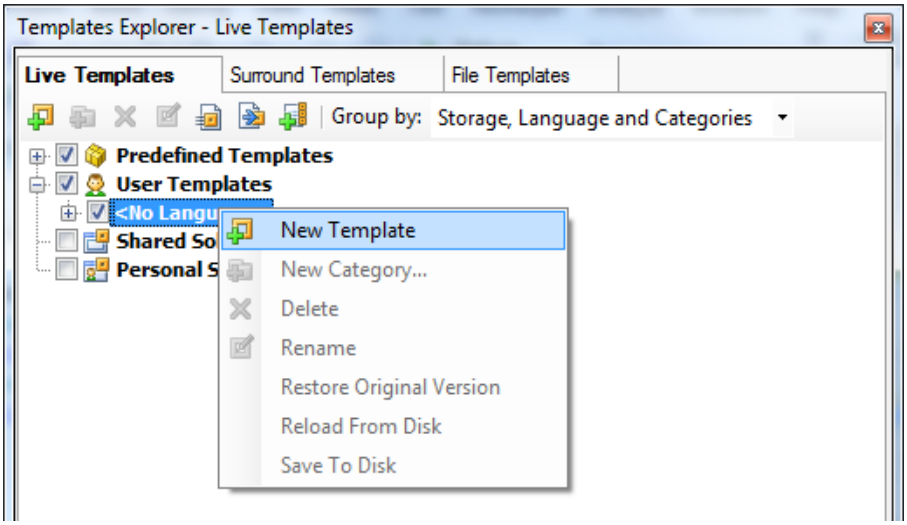
После того, как объявлен класс, как всегда идет объявление тестового метода.

```
[TestClass]
    public class classname {
        [TestMethod]
        public void name() {
        }
    }
```

Смотрите сколько текста! И осмысленного тут будет только 2 (два) слова! Руками набирать все это определенно нельзя, надо это как-то автоматизировать. С помощью решарпера все делается весьма быстро и легко.

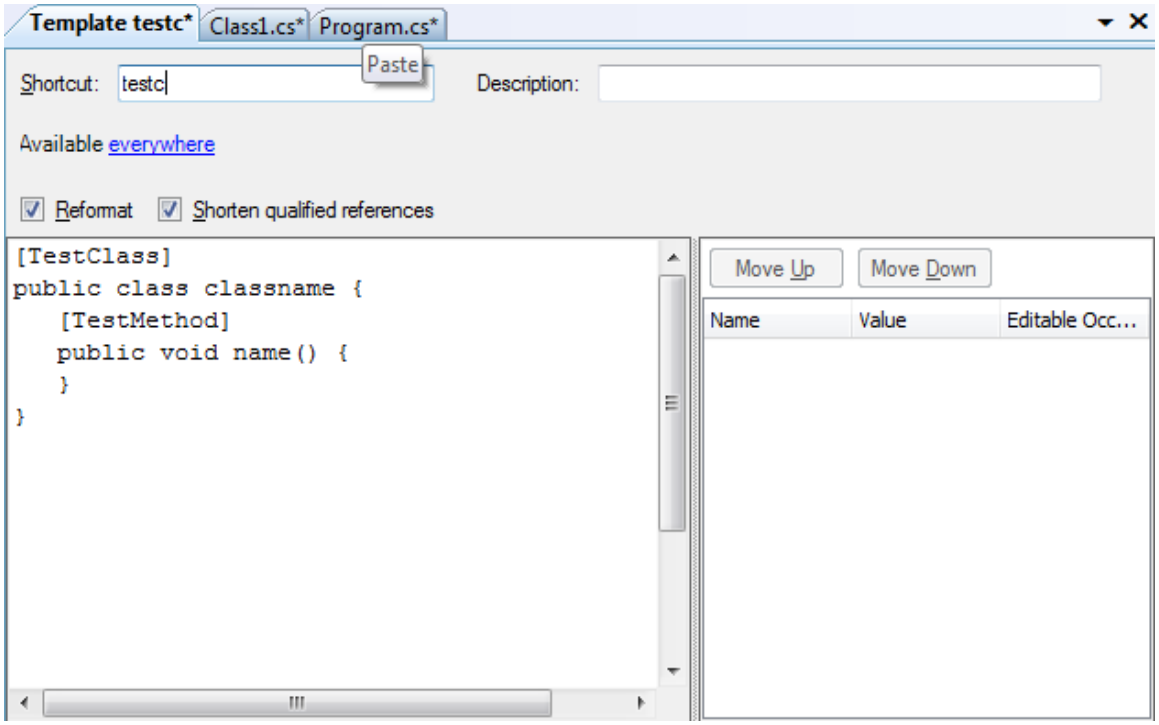


Открываем “ReSharper > Live Templates…”, появляется окно, где мы можем создавать заготовки кода. Раскрываем узел “User Templates”, вызываем контекстное меню для него или пункта “<No Languages>” и выбираем “New template”.



После этого в основной области студии появится закладка, где и будем оформлять заготовку.

Копируем код, который хотим видеть в качестве заготовки, вставляем в окно, называем заготовку “**testc**” и оформляем переменные части. Т.е. на данный момент должно быть примерно так:



Переменные в решарпере маркируются знаком доллара с двух сторон. В заготовке у нас будет 2 переменные:

- Classname
- Testname

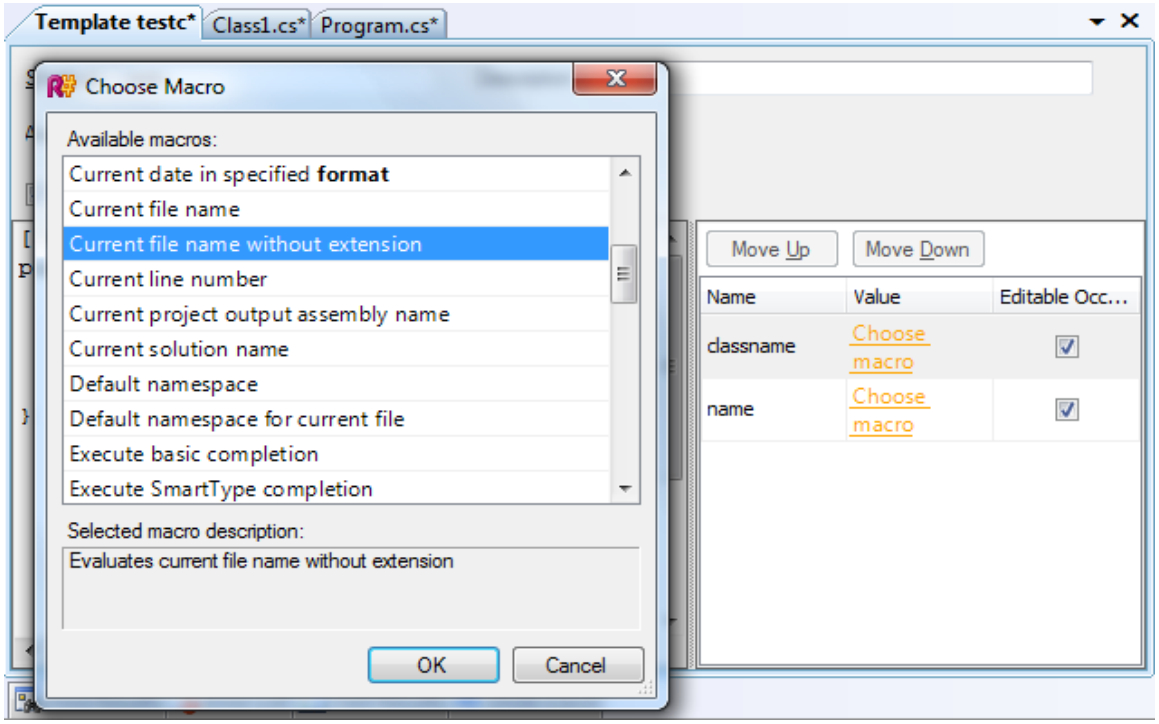
Так же надо обозначить место, где установить курсор после вставки заготовки. Для этого есть специальная переменная END.

Код приобретает такой вид:

```
[TestClass]
public class $classname$ {
   [TestMethod]
   public void $name$() {
         $END$
   }
}
```

В правой части окна можно видеть используемые переменные. Им можно назначить наиболее подходящий тип данных или сразу значение. Так же можно указать, останавливаться ли на редактировании данного элемента после вставки, с помощью галки.

Я думаю, что многие придерживаются правила «Один класс – один файл». Если да, то в этом случае можно назвать тестовый класс, так же как и файл. Для этого нажимаем на “Choose macro” напротив перменной в правой половине экрана и выбираем “Current file name without extension”.



Снимаем галочку напротив classname, т.к. название уже будет задано.

Сохраняем все, и теперь можно попробовать в действии. Создаем новый файл в тестовом проекте. Открываем его, пишем **testc** и нажимаем Tab. ReSharper вставит заготовку кода, подставит имя файла, подсветит имя метода, предлагая его ввести. Вводите имя теста, еще раз жмете Tab и оказываетесь в теле метода. Можно начинать писать осмысленный код.

Как правило, в одном тестовом классе находится несколько методов, и не все они похожи по начальным значениям. Или вам религия не позволяет копи-пастить код вообще. На этот случай можно сделать еще одну заготовку кода, которая вставляет только тестовый метод.

```
[TestMethod]
public void $name$(){
      $END$
}
```

При написании тестов без Assert не обойтись, так что можно автоматизировать и эту операцию. Самые популярные конструкции у меня:

- Assert.AreEqual
- Assert.IsTrue
- Assert.IsFalse

Сокращения для них я делаю по первым буквам. Ну и заготовки для них простые будут

```
Assert.AreEqual($END$,);
```

```
Assert.IsTrue($END$);
```

```
Assert.IsFalse($END$);
```

В заготовки кода так же можно занести метод с аттрибутом [TestInitialize]. Вообще советую активно пользоваться заготовками, как своими, так и встроенными. Думаю, вы на примере

убедились, что писать их легко и просто. Я не считаю, что это отучит вас быстро печатать на клавиатуре, но позволит вом сэкономить время для проверки новых подходов, решений задачи.

Единственный минус из всего этого, что без решарпера будет огромнейшая ломка. Может дойти до отказа от работы без этого инструмента. =)

Видео всего процесса и использования.

О том, как еще ускорить и облегчить создание кода я расскажу в следующей статье.

Hard’n’heavy!