Teste Unitário com Unit Test

fatec – senai centro | Rua Caetés 741, Belo Horizonte

Testando o Unit Test No Visual studio 2015

Rubem Cândido dos Santos

2019

Sumário

[Introdução 2](#_Toc25133964)

[Ferramentas necessárias 2](#_Toc25133965)

[Visual Studio 2](#_Toc25133966)

[Criando o projeto 2](#_Toc25133967)

[Inserindo um código de teste 4](#_Toc25133968)

[Inserindo classe UnitTest 5](#_Toc25133969)

[Requisitos de classe de teste 6](#_Toc25133970)

[TestClass 6](#_Toc25133971)

[TestMethod 7](#_Toc25133972)

[TestMethod1 7](#_Toc25133973)

[Adicionando um casos de teste 11](#_Toc25133974)

[Caso de teste 1 11](#_Toc25133975)

[AreEqual 12](#_Toc25133976)

[Executando Teste 12](#_Toc25133977)

[Caso de teste 2 13](#_Toc25133978)

[Caso de teste 3 13](#_Toc25133979)

[Caso de teste 4 14](#_Toc25133980)

[Caso de teste 5 14](#_Toc25133981)

[Caso de teste 6 14](#_Toc25133982)

[Caso de teste 7 15](#_Toc25133983)

[Executando os testes 15](#_Toc25133984)

[Github 16](#_Toc25133985)

[Conclusão 16](#_Toc25133986)

# Introdução

O teste funcional, também conhecido como teste de caixa branca é uma etapa extremamente importante para o bom desenvolvimento de sistemas. Seu objetivo é analisar as funcionalidades existentes no software, ou seja, parte dos códigos e métodos. Esse teste geralmente é realizado pelo próprio programador verificando o retorno esperado.

# Ferramentas necessárias

## Visual Studio

Visual Studio a partir do 2012

# Criando o projeto

No **Visual Studio 2015**, clique em **File** > **New** > **Project**. Selecione o **Class Library** e nomeie como **TesteUnit**.

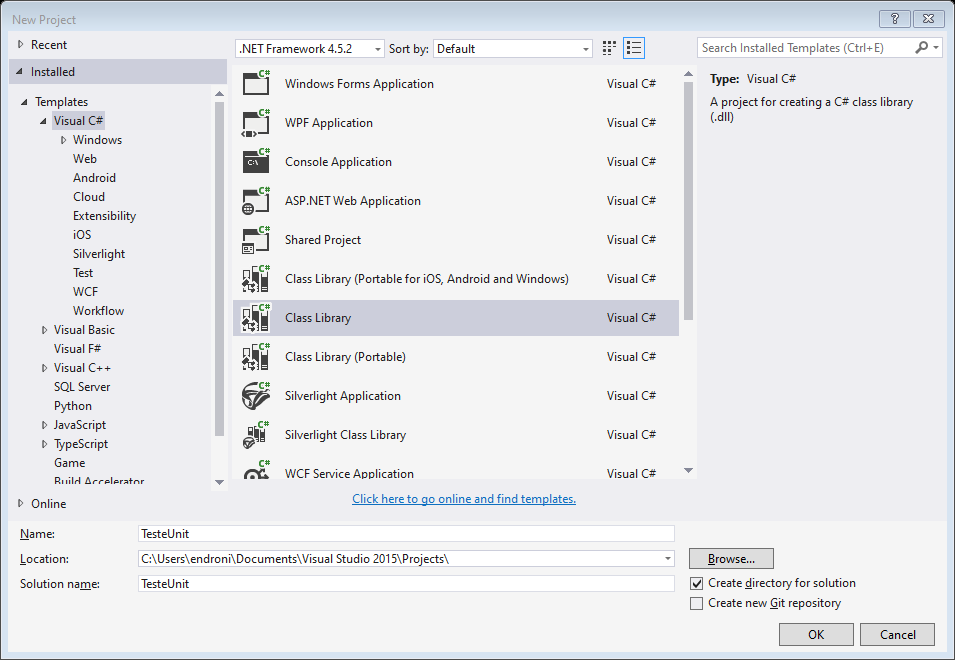


Figura 1. Criando projeto TesteUnit

Renomeie o class1.cs para Desconto.cs

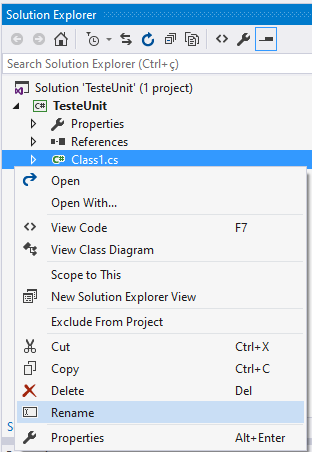


Figura 2. Rename Class1.cs

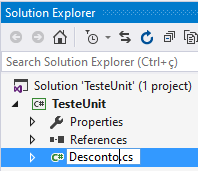


Figura 3. Desconto.cs

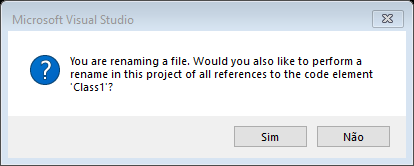


Figura 4. Aceitando a alteração

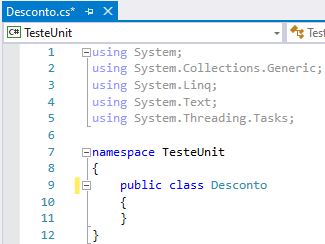


Figura 5. Desconto.cs

# Inserindo um código de teste

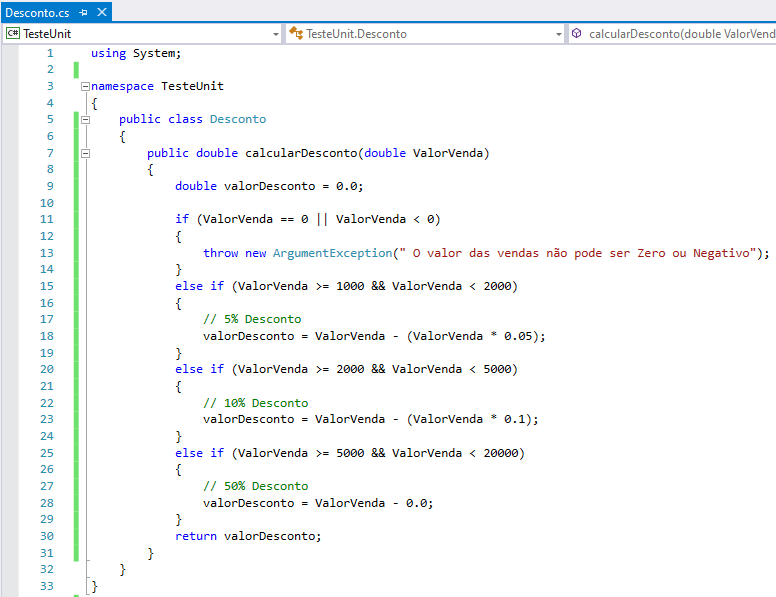


Figura 6. Código Desconto.cs

A classe desconto foi projetada para aplicar um percentual de desconto dependendo do valor da venda através do método

**calculcaDesconto(double ValorVenda)**;

# Inserindo classe UnitTest

Clique com o botão direito na solução depois em **Add** e **New Project**

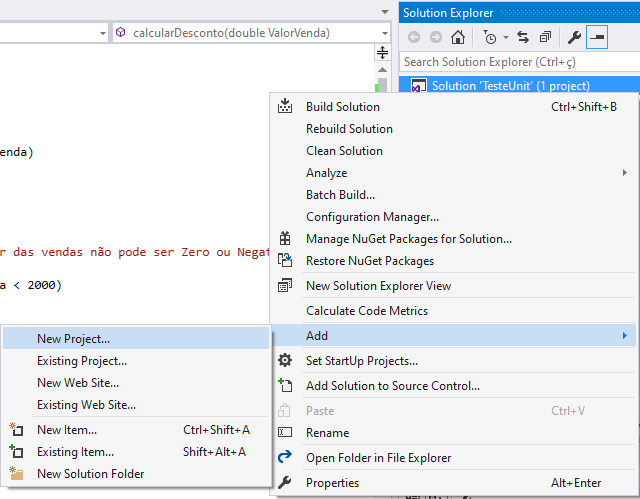


Figura 6. New Project

Vamos chama-lo de **UnitTesteDesconto.**

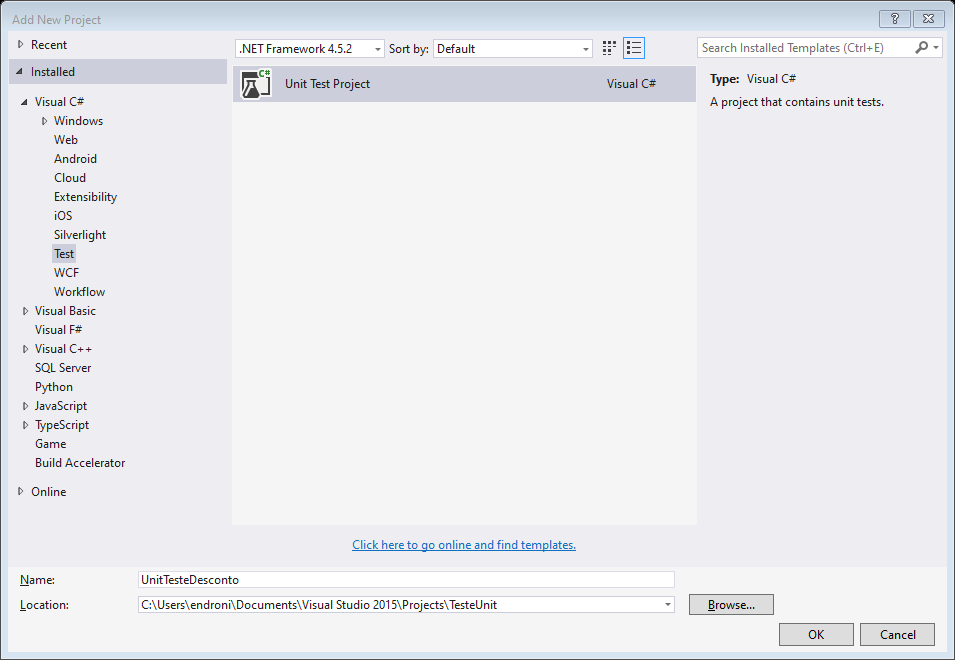


Figura 7. Unit Test Project

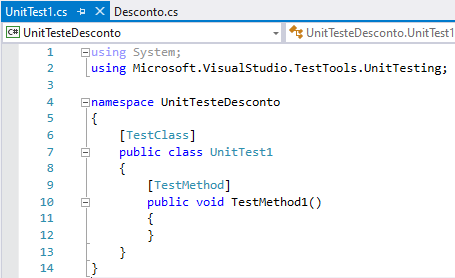


Figura 8. UnitTeste1.cs

# Requisitos de classe de teste

## TestClass

O atributo [TestClass] é necessário em qualquer classe que contenha métodos de teste de unidade que você queira executar no Gerenciador de Testes.

## TestMethod

Cada método de teste que você deseja reconhecer com o Gerenciador de Testes precisa ter o atributo [TestMethod].

## TestMethod1

O método TestMethod1 padrão, pode ser excluído pois iremos criar nossos próprios métodos.

Renomeie **UnitTeste1.cs** para **CasoTeste.cs**

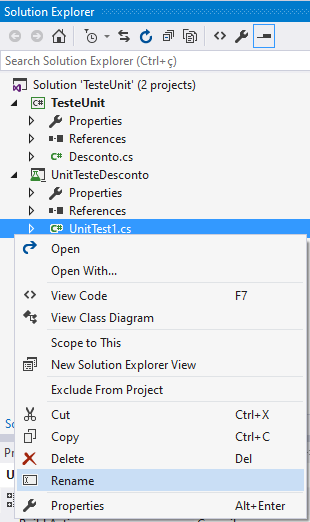


Figura 9. Rename

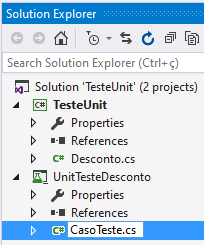


Figura 10. CasoTeste.cs

Aceite a alteração da classe

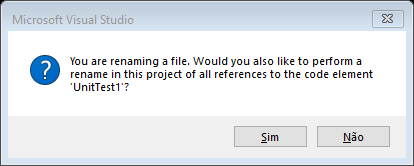


Figura 11. Alteração UnitTeste1 para CasoTeste



Figura 12. CasoTeste.cs

Referenciando a classe CasoTeste em Desconto

Clique com o botão direito em **UnitTesteDesconto** > **Add** > **Reference**

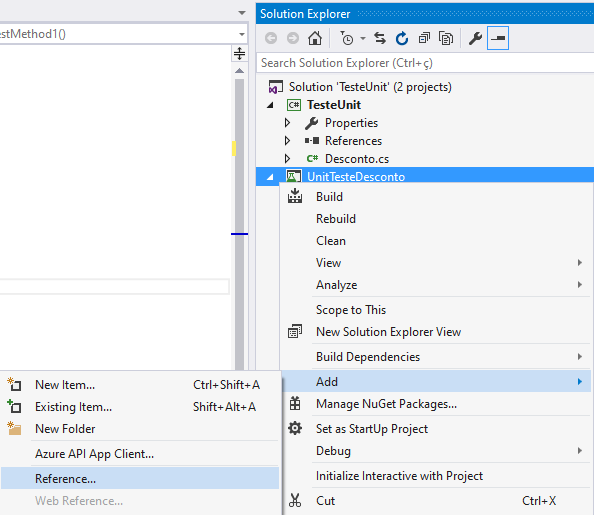


Figura 14. Reference

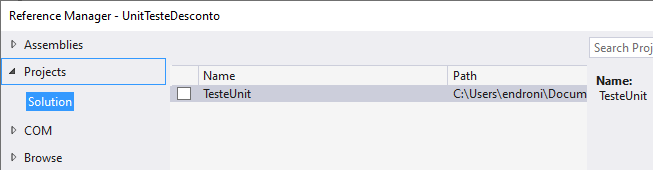


Figura 15. TesteUnit

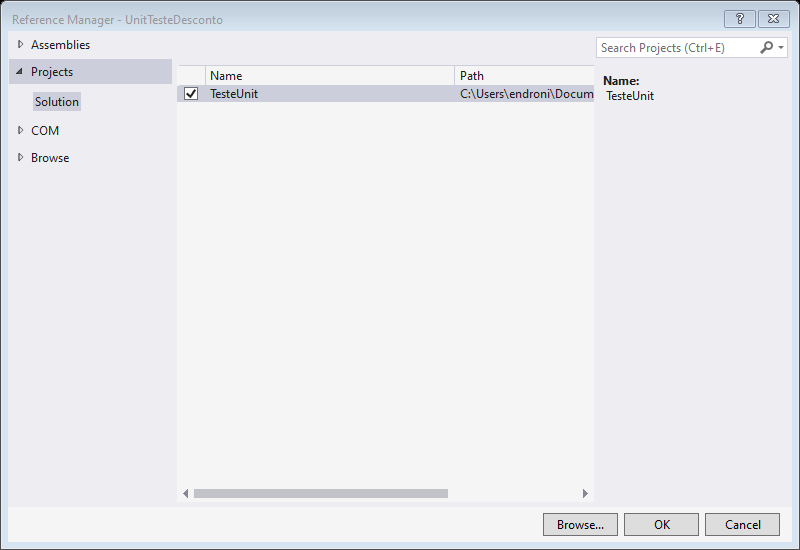


Figura 16. TesteUnit sendo referenciado

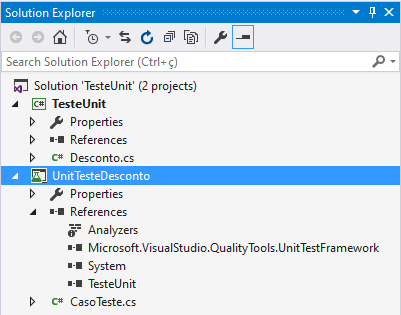


Figura 17. References

Acrescente a referência ao **TesteUnit** manualmente no código

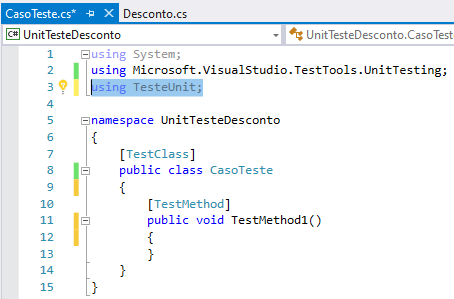


Figura 18. Using TesteUnit

# Adicionando um casos de teste

Os métodos de teste precisa ser público e retornar void. No nosso primeiro caso nomeamos como **MaiorOuIgual\_1000()**

## Caso de teste 1

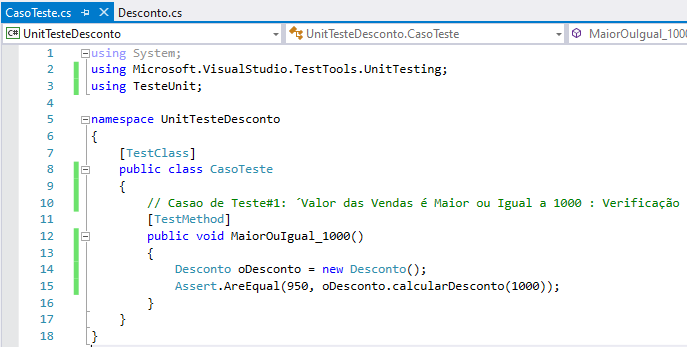


Figura 19. Definindo valores para o primeiro Caso de Teste

O método MaiorOuIgual\_1000 apenas define um novo objeto **Desconto** e em seguida retira um valor válido.

## AreEqual

O método **AreEqual** para verificar se o saldo final é conforme o esperado.

**AreEqual(Object,Object)** – Testa se os objetos especificados são iguais e lança uma exceção se os dois objetos não forem iguais. Diferentes tipos numéricos são tratados como desiguais, mesmo que os valores lógicos sejam iguais. 42L não é igual a 42. Fonte: docs.microsoft.com

# Executando Teste

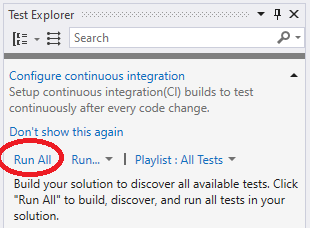


Figura 20. Run All

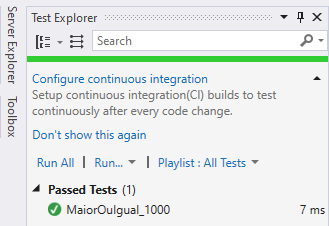


Figura 21. Resultado do Caso de teste

## Caso de teste 2

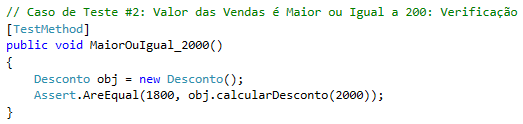


Figura 22. Caso de Teste 2

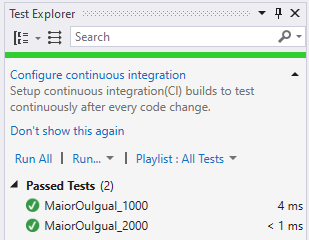


Figura 23. Resultado dos dois Casos de Teste

## Caso de teste 3

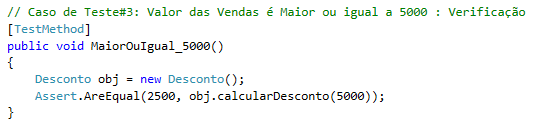


Figura 24. Caso de teste 3.

## Caso de teste 4

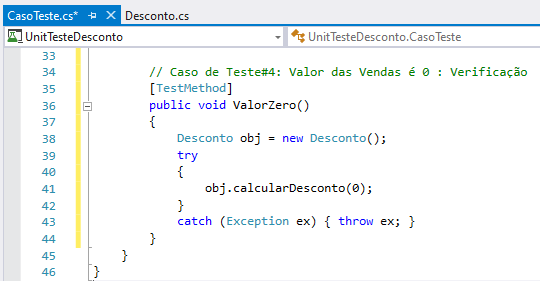


Figura 25. Caso de teste 4

## Caso de teste 5

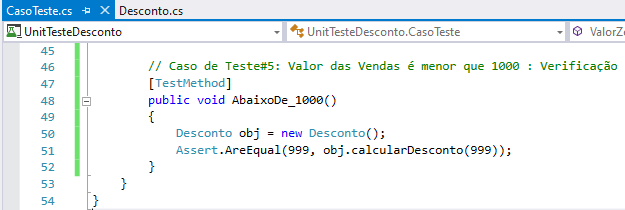


Figura 26. Caso de teste 5

## Caso de teste 6

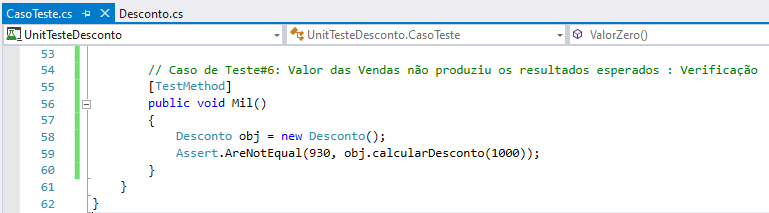


Figura 27. Caso de teste 6

## Caso de teste 7

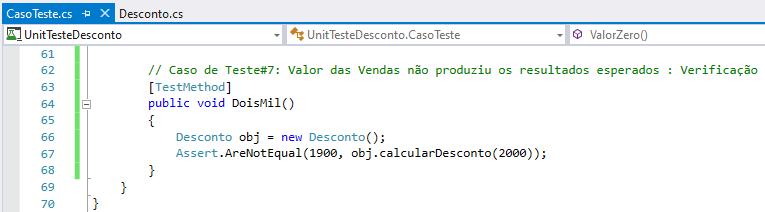


Figura 28. Caso de teste 7

# Executando os testes

Clique em **Run All**

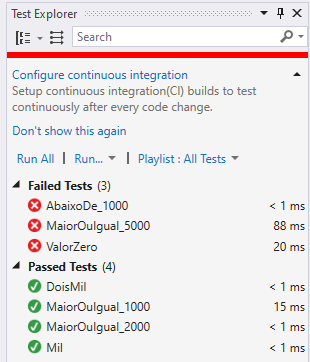


Figura 29. Resultado dos testes

Assim como na teoria. Precisamos testar os caso válidos e os casos inválidos.

# Github

https://github.com/endroni/TesteUnit\_Desconto.git

# Conclusão

O Unit Test é uma ferramenta muito útil para se realizar testes unitários no C#.

**BIBLIOGRAFIA BÁSICA:**

PRESSMAN, Roger S. **Engenharia de Software: uma abordagem profissional**. 7. ed. São Paulo: Pearson Makron Books, 2011.

SOMMERVILLE, Ian. **Engenharia de Software**. 8. ed. São Paulo: Pearson Addison Wesley, 2007.

TONSIG, Sergio Luiz. **Engenharia de Software: análise e projeto de sistemas**. 2. ed. Rio de Janeiro: Ciência Moderna, 2008.

**BIBLIOGRAFIA COMPLEMENTAR:**

BEZERRA, Eduardo. **Princípios de Análise e Projeto de Sistemas com UML**. Rio de Janeiro: Elsevier, 2007.

Macoratti. **C# Usando NUnit (testes unitários).** Macoratti.net: Desenvolvimento com CSharp. Disponível em: <http://www.macoratti.net/13/09/c\_unit1.htm/>. Acesso em: 19 de novembro de 2019.

Documentos do Visual Studio. **Passo a passo: criar e executar testes de unidade para código gerenciado.** Microsoft: Testes de unidade. Disponível em: <https://docs.microsoft.com/pt-br/visualstudio/test/walkthrough-creating-and-running-unit-tests-for-managed-code?view=vs-2019/>. Acesso em: 20 de novembro de 2019.