第9回
BAU Study Session

【テーマ】

- **Unit testing in .NET Core**

日時　：2020年7月22日(水)

参加者：古川・森(発表者)・有川

Contents

下記３つのテストプロジェクトを試してみる

- MSTest
- xUnit
- NUnit

- MSTest

MSTest：① ディレクトリ構成

/UnitTestSample
    /PrimeService
        PrimeService.cs
        PrimeService.csproj
    /PrimeService.Tests.MSTest
        PrimeService_IsPrimeShould.cs
        PrimeService.Tests.MSTest.csproj
    UnitTestSample.sln

MSTest：② 任意の場所でディレクトリ＆sln作成

[任意の場所] $ mkdir UnitTestSample

[任意の場所] $ cd UnitTestSample/

[UnitTestSample] $ dotnet new sln

MSTest：③ ライブラリのプロジェクト作成

[UnitTestSample] $ mkdir PrimeService

[UnitTestSample] $ cd PrimeService/

[PrimeService] $ dotnet new classlib

■ MSTest：④ クラスファイルの作成＆編集

[PrimeService] $ vim PrimeService.cs

```
using System;

namespace Prime.Services
{
    public class PrimeService
    {
        public bool IsPrime(int candidate)
        {
            if (candidate == 1)
            {
                return false;
            }
            throw new NotImplementedException("Please create a test first.");
        }
    }
}
```

素数かどうかチェックするメソッド
引数で渡ってきた値が1ならfalse

MSTest：⑤ ライブラリprojをslnに追加

[PrimeService] $ cd ..

[UnitTestSample] $ dotnet sln add PrimeService/PrimeService.csproj

MSTest：⑥ テストプロジェクトを作成

[UnitTestSample] $ mkdir PrimeService.Tests.MSTest

[UnitTestSample] $ cd PrimeService.Tests.MSTest

[PrimeService.Tests.MSTest] $ dotnet new mstest

MSTest：⑦ テストprojにライブラリ参照を追加

[PrimeService.Tests.MSTest] $ dotnet add reference ../PrimeService/PrimeService.csproj

MSTest：⑧ テストprojをslnに追加

[PrimeService.Tests.MSTest] $ cd ..

[UnitTestSample] $ dotnet sln add PrimeService.Tests.MSTest/PrimeService.Tests.MSTest.csproj

MSTest：⑨ テストクラスを作成

[UnitTestSample] $ cd PrimeService.Tests.MSTest

[PrimeService.Tests.MSTest] $ vim PrimeService_IsPrimeShould.cs

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

namespace Prime.UnitTests.MSTest.Services
{
    [TestClass]
    public class PrimeService_IsPrimeShould
    {
        private readonly PrimeService primeService;

        public PrimeService_IsPrimeShould()
        {
            this.primeService = new PrimeService();
        }

        [TestMethod]
        public void IsPrime_InputIs1_ReturnFalse()
        {
            var result = this.primeService.IsPrime(1);

            Assert.IsFalse(result, "1 should not be prime.");
        }
    }
}
```

ライブラリのメソッドに
1を渡してfalseが返ってくるのは
正しいよね？
というテストを用意

MSTest：⑩ テスト実行

[PrimeService.Tests.MSTest] $ dotnet test

xxxx/UnitTestSample/PrimeService.Tests.MSTest/bin/Debug/netcoreapp3.1/
PrimeService.Tests.MSTest.dll(.NETCoreApp,Version=v3.1) のテスト実行

Microsoft (R) Test Execution Command Line Tool Version 16.6.0
Copyright (c) Microsoft Corporation. All rights reserved.

テスト実行を開始しています。お待ちください...

合計 1 個のテスト ファイルが指定されたパターンと一致しました。

テストの実行に成功しました。
テストの合計数: 3
    成功: 3
合計時間: 1.2531 秒

- xUnit

xUnit：① ディレクトリ構成

/UnitTestSample
    /PrimeService
        PrimeService.cs
        PrimeService.csproj
    /PrimeService.Tests.Xunit
        PrimeService_IsPrimeShould.cs
        PrimeService.Tests.Xunit.csproj
    UnitTestSample.sln

xUnit：② テストプロジェクトを作成

[UnitTestSample] $ mkdir PrimeService.Tests.Xunit

[UnitTestSample] $ cd PrimeService.Tests.Xunit

[PrimeService.Tests.Xunit] $ dotnet new unit

xUnit：③ テストprojにライブラリ参照を追加

[PrimeService.Tests.Xunit] $ dotnet add reference ../PrimeService/PrimeService.csproj

xUnit：④ テストprojをslnに追加

[PrimeService.Tests.Xunit] $ cd ..

[UnitTestSample] $ dotnet sln add PrimeService.Tests.Xunit/PrimeService.Tests.Xunit.csproj

xUnit：⑤ テストクラスを作成

[UnitTestSample] $ cd PrimeService.Tests.Xunit

[PrimeService.Tests.Xunit] $ vim PrimeService_IsPrimeShould.cs

```
using Xunit;
using Prime.Services;

namespace Prime.UnitTests.Xunit.Services
{
    public class PrimeService_IsPrimeShould
    {
        private readonly PrimeService primeService;

        public PrimeService_IsPrimeShould()
        {
            this.primeService = new PrimeService();
        }

        [Fact]
        public void IsPrime_InputIs1_ReturnFalse()
        {
            var result = this.primeService.IsPrime(1);

            Assert.False(result, "1 should not be prime.");
        }
    }
}
```

xUnit：⑥ テスト実行

[PrimeService.Tests.MSTest] $ dotnet test

xxxx/UnitTestSample/PrimeService.Tests.Xunit/bin/Debug/netcoreapp3.1/
PrimeService.Tests.Xunit.dll(.NETCoreApp,Version=v3.1) のテスト実行

Microsoft (R) Test Execution Command Line Tool Version 16.6.0
Copyright (c) Microsoft Corporation. All rights reserved.

テスト実行を開始しています。お待ちください...

合計 1 個のテスト ファイルが指定されたパターンと一致しました。

テストの実行に成功しました。
テストの合計数: 3
    成功: 3
合計時間: 1.5856 秒

- NUnit

NUnit：① ディレクトリ構成

/UnitTestSample
    /PrimeService
        PrimeService.cs
        PrimeService.csproj
    /PrimeService.Tests.Nunit
        PrimeService_IsPrimeShould.cs
        PrimeService.Tests.Nunit.csproj
    UnitTestSample.sln

NUnit：② テストプロジェクトを作成

[UnitTestSample] $ mkdir PrimeService.Tests.Nunit

[UnitTestSample] $ cd PrimeService.Tests.Nunit

[PrimeService.Tests.Nunit] $ dotnet new unit

NUnit：③ テストprojにライブラリ参照を追加

[PrimeService.Tests.Nunit] $ dotnet add reference ../PrimeService/PrimeService.csproj

NUnit：④ テストprojをslnに追加

[PrimeService.Tests.Nunit] $ cd ..

[UnitTestSample] $ dotnet sln add PrimeService.Tests.Nunit/PrimeService.Tests.Nunit.csproj

NUnit：⑤ テストクラスを作成

[UnitTestSample] $ cd PrimeService.Tests.Nunit

[PrimeService.Tests.Nunit] $ vim PrimeService_IsPrimeShould.cs

```
using NUnit.Framework;
using Prime.Services;

namespace Prime.UnitTests.Nunit.Services
{
    [TestFixture]
    public class PrimeService_IsPrimeShould
    {
        private PrimeService primeService;

        [SetUp]
        public void Setup()
        {
            this.primeService = new PrimeService();
        }

        [Test]
        public void Test1()
        {
            var result = this.primeService.IsPrime(1);

            Assert.IsFalse(result, "1 should not be prime.");
        }
    }
}
```

NUnit：⑥ テスト実行

[PrimeService.Tests.MSTest] $ dotnet test

xxxx/UnitTestSample/PrimeService.Tests.Nunit/bin/Debug/netcoreapp3.1/
PrimeService.Tests.Nunit.dll(.NETCoreApp,Version=v3.1) のテスト実行

Microsoft (R) Test Execution Command Line Tool Version 16.6.0
Copyright (c) Microsoft Corporation. All rights reserved.

テスト実行を開始しています。お待ちください...

合計 1 個のテスト ファイルが指定されたパターンと一致しました。

テストの実行に成功しました。
テストの合計数: 3
    成功: 3
合計時間: 1.2930 秒

- テストケースを増やしてみよう

ライブラリを修正

まずはライブラリのメソッドを下記のように修正

```
using System;

namespace Prime.Services
{
    public class PrimeService
    {
        public bool IsPrime(int candidate)
        {
            if (candidate < 2)
            {
                return false;
            }
            throw new NotImplementedException("Please create a test first.");
        }
    }
}
```

引数で渡ってきた値が
2より小さい場合はfalse

MSTestのテストメソッドを修正

テストメソッドを下記のように修正

```
[DataTestMethod]
[DataRow(-1)]
[DataRow(0)]
[DataRow(1)]
public void IsPrime_ValuesLessThan2_ReturnFalse(int value)
{
    var result = this.primeService.IsPrime(value);

    Assert.IsFalse(result, $"{value} should not be prime.");
}
```

こんな感じでDataRow()に
-1, 0, 1を指定すると
３つの値でテストをしてくれる

xUnitのテストメソッドを修正

テストメソッドを下記のように修正

こんな感じでInlineData()に
-1, 0, 1を指定すると
３つの値でテストをしてくれる

```
[Theory]
[InlineData(-1)]
[InlineData(0)]
[InlineData(1)]
public void IsPrime_ValuesLessThan2_ReturnFalse(int value)
{
    var result = this.primeService.IsPrime(value);

    Assert.False(result, $"{value} should not be prime");
}
```

NUnitのテストメソッドを修正

テストメソッドを下記のように修正

```
[TestCase(-1)]
[TestCase(0)]
[TestCase(1)]
public void IsPrime_ValuesLessThan2_ReturnFalse(int value)
{
    var result = this.primeService.IsPrime(value);

    Assert.IsFalse(result, $"{value} should not be prime");
}
```

こんな感じでTestCase()に
-1, 0, 1を指定すると
３つの値でテストをしてくれる

おまけ

第6回で実施した四則演算にNUnitのテストプロジェクトを追加しました。

https://github.com/ymdevx3/try-arithmetic-operations-csharp

テストケースを配列にして
その配列の変数名を
TestCaseSourceAttribute()に
文字列で渡してあげることも！

```
[TestCaseSourceAttribute("CalcCases")]
public void 結果が正しいか確認2(string expression, double result)
{
    var value = Calculator.Calculate(expression);
    //Assert.AreEqual(value, result);
    Assert.That(value, Is.EqualTo(result));
}
static object[] CalcCases =
{
    new object[] { "10+20*30+40-50*2/4 * 2", 600 },
    new object[] { "23+31", 54 },
    new object[] { "43-4+45*23", 1074 },
    new object[] { "32+54*27*4/13", 480.61538461538464 },
};
```

引用

- MSTest

  https://docs.microsoft.com/ja-jp/dotnet/core/testing/?pivots=mstest

- xUnit

  https://docs.microsoft.com/ja-jp/dotnet/core/testing/?pivots=xunit

- NUnit

  https://docs.microsoft.com/ja-jp/dotnet/core/testing/?pivots=nunit

フリートーク　～ 思いつくままに ～

- vi コマンドのレクチャー

  古川氏に依頼し、初心者向けのコマンドを教えていただいた。

  まとめはまた後日UPする予定。

次回のBSS

- 日程　：2020年7月29日(水)
- 司会者：古川
- テーマ：Dockerについて