Unit Test In .NET Core Application Using MSTest

Jignesh Trivedi Feb 11 2018 Article

9 11 12.4k

Unittest.zip
Download Free .NET & JAVA Files API

Introduction

There are three different test frameworks which are supported by the unit test with asp.net core: MSTest, xUnit, and NUnit, which allow us to test our code in a consistent way. In this article, I will explain about the unit test in asp.net core using MSTest.

To demonstrate the example of the unit test, I have created an MVC project, solution and unit test project by using CLI (Command Line Interface). To create the MVC and Test project, I am following the steps given below.

Create Solution file using the following command. This command creates the empty solution.

```
dotnet new sln -n MVCUnittest
```

Create MVC Project: Using the following command, MVC project will be created

```
>dotnet new MVC
```

Adding this project to solution: Using the following command we can add project to solution

```
>dotnet sln add Unittest\Unittest.csproj
```

Create MsTest Project: Using the following command, we can create an MSTest project.

```
>dotnet new mstest
```

This command creates MSTest Project and the generated template configures test runner into .csproj file

```
<ItemGroup>
  <PackageReference Include="Microsoft.NET.Test.Sdk" Version="15.3.0" />
  <PackageReference Include="MSTest.TestAdapter" Version="1.1.18" />
  <PackageReference Include="MSTest.TestFramework" Version="1.1.18" />
</ItemGroup>
```

The generated code also has a dummy unit test file. It looks as follows.

```
namespace Testproject
{
    [TestClass]
    public class UnitTest1
        public void TestMethod1()
        {

        }
    }
}
```

The TestClass attribute denotes the class which contains unit tests and TestMethod attribute denotes that a method is a test method.

Adding test project to solution

```
>dotnet sln add TestProject\Testproject.csproj
```

To demonstrate the concept, I have created a method within HomeController class (GetEmployeeName). This method accepts empId as a parameter and based on this, it will return the name of the employee or "Not Found" hard code string.

HomeController

```
public string GetEmployeeName(int empId)
{
    string name;
    if (empId == 1)
    {
        name = "Jignesh";
    }
    else if (empId == 2)
    {
        name = "Rakesh";
    }
    else
    {
        name = "Not Found";
    }
    return name;
}
```

In the following test method, I have passed a hardcoded value and checked the result using Assert class.

Unittest1.cs

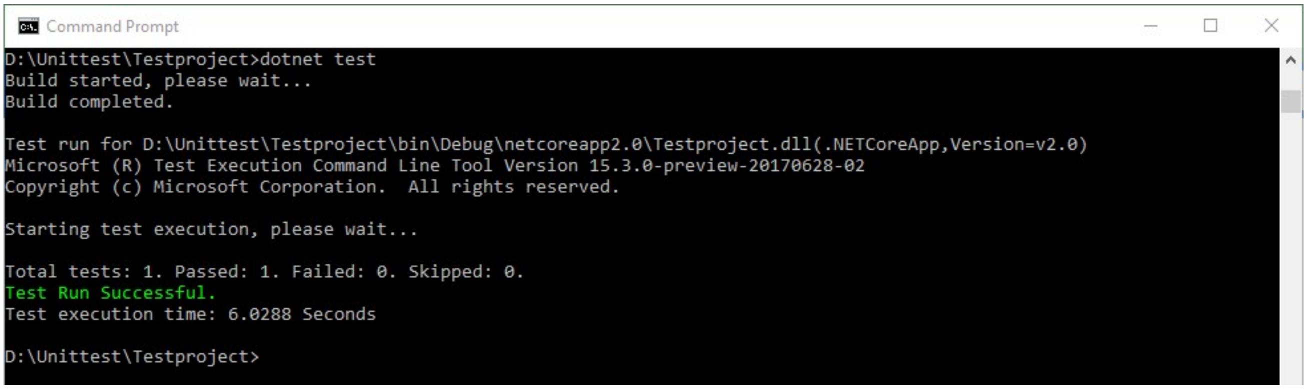
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Unittest.Controllers;

namespace TestProject1
{
    [TestClass]
    public class UnitTest1
    {
        [TestMethod]
```

```
string result = home.GetEmploye
Assert.AreEqual(result, "Jignesh
        }
```

The final step is to run the Unit test. Using the following command, we can run all our test cases.

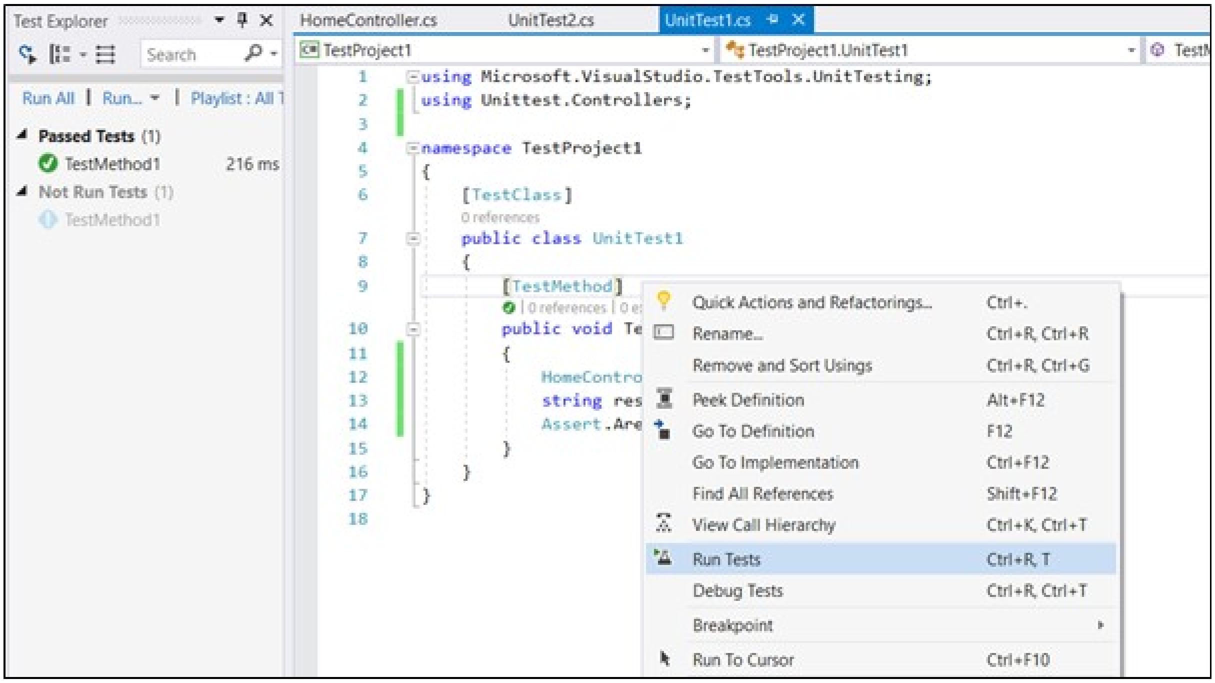
```
>dotnet test
```

Result



```
D:\Unittest\Testproject>dotnet test
Build started, please wait...
Build completed.

Test run for D:\Unittest\Testproject\bin\Debug\netcoreapp2.0\Testproject.dll(.NETCoreApp,Version=v2.0)
Microsoft (R) Test Execution Command Line Tool Version 15.3.0-preview-20170628-02
Copyright (c) Microsoft Corporation.  All rights reserved.

Starting test execution, please wait...

Total tests: 1. Passed: 1. Failed: 0. Skipped: 0.
Test Run Successful.
Test execution time: 6.0288 Seconds

D:\Unittest\Testproject>
```

We also run all test cases or individual tests within Visual Studio using Test Explore.

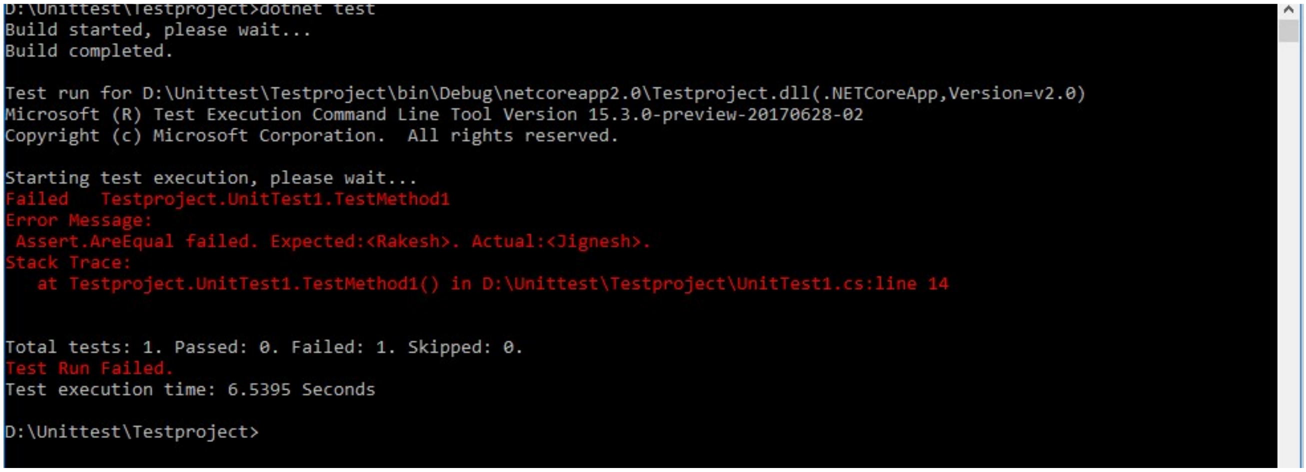


```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Unittest.Controllers;

namespace TestProject1
{
    [TestClass]
    0 references
    public class UnitTest1
    {
        [TestMethod]
        | 0 references | 0 e
        public void Te
        {
            HomeContro
            string res
            Assert.Are
        }
    }
}
```

In the preceding example, my test result (actual) is matched with the expected result. In the following example, my actual result is not matched with the expected result.

```
[TestMethod]
public void TestMethod2()
{
    HomeController home = new HomeController();
```

Result

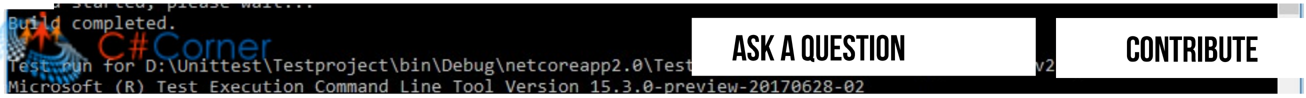


```
D:\Unittest\Testproject>dotnet test
Build started, please wait...
Build completed.

Test run for D:\Unittest\Testproject\bin\Debug\netcoreapp2.0\Testproject.dll(.NETCoreApp,Version=v2.0)
Microsoft (R) Test Execution Command Line Tool Version 15.3.0-preview-20170628-02
Copyright (c) Microsoft Corporation.  All rights reserved.

Starting test execution, please wait...
Failed   Testproject.UnitTest1.TestMethod1
Error Message:
 Assert.AreEqual failed. Expected:<Rakesh>. Actual:<Jignesh>.
Stack Trace:
   at Testproject.UnitTest1.TestMethod1() in D:\Unittest\Testproject\UnitTest1.cs:line 14


Total tests: 1. Passed: 0. Failed: 1. Skipped: 0.
Test Run Failed.
Test execution time: 6.5395 Seconds

D:\Unittest\Testproject>
```

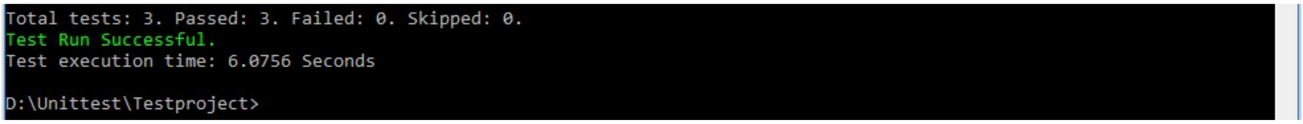
To unit test every block of code, we require more test data. We can add more test methods using TestMethod attribute, but it is a very tedious job. The MSTest project is also supported with another attribute which enables us to write a suite for a similar test. DataTestMethod attributes represent a suite of tests which executes the same code with different input arguments. A DataRow attribute can be used for specifying the values for those inputs. Instead of creating a new test, we can use these two attributes: DataTestMethod and DataRow to create a single data-driven test.

```csharp
using Microsoft.VisualStudio.TestTools.UnitTesting;
using Unittest.Controllers;

namespace TestProject1
{
    [TestClass]
    public class UnitTest2
    {
        [DataTestMethod]
        [DataRow(1, "Jignesh")]
        [DataRow(2, "Rakesh")]
        [DataRow(3, "Not Found")]
        public void TestMethod1(int empId, string name)
        {
            HomeController home = new HomeController();
            string result = home.GetEmployeeName(empId);
            Assert.AreEqual(result, name);
        }
    }
}
```

Result



```
Build completed.
Test run for D:\Unittest\Testproject\bin\Debug\netcoreapp2.0\Test
Microsoft (R) Test Execution Command Line Tool Version 15.3.0-preview-20170628-02
```



```
Total tests: 3. Passed: 3. Failed: 0. Skipped: 0.
Test Run Successful.
Test execution time: 6.0756 Seconds

D:\Unittest\Testproject>
```

Unit test with ILogger

The .net core support built-in dependency injection. So, whatever services we want to use during the execution of the code are injected as a dependency. One of the best examples is ILogger service. Using the following code, we can configure ILogger service in our asp.net core project.

Configure ILogger in Program.cs

```csharp
using Microsoft.AspNetCore;
using Microsoft.AspNetCore.Hosting;
using Microsoft.Extensions.Logging;

namespace Unittest
{
    public class Program
    {
        public static void Main(string[] args)
        {
            BuildWebHost(args).Run();
        }

        public static IWebHost BuildWebHost(string[] args) =>
            WebHost.CreateDefaultBuilder(args)
                .ConfigureLogging((hostingContext, logging) =>
                {
                    logging.AddConfiguration(hostingContext.Configuration.GetSection("Logging")
                    logging.AddConsole();
                    logging.AddDebug();
                })
                .UseStartup<Startup>()
                .Build();
    }
}
```

TestController.cs

```csharp
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Logging;

namespace Unittest.Controllers
{
    public class TestController : Controller
    {
        private readonly ILogger _logger;
        public TestController(ILogger<TestController> logger)
        {
            _logger = logger;
        {
            _logger.LogDebug("Index Method Ca
            return "Hi! Reader";
```

```
}
```

Unit Test Method

For unit test controller which have the dependency on ILogger service, we have to pass ILogger object or null value to the constructor. To create these type of dependencies, we can create an object of a service provider and help with the service provided, we can create the object of such services.

In the following code, I have created service provider object and created the ILogger object.

```
[TestMethod]
public void TestMethod4()
{
    var serviceProvider = new ServiceCollection()
        .AddLogging()
        .BuildServiceProvider();

    var factory = serviceProvider.GetService<ILoggerFactory>();

    var logger = factory.CreateLogger<TestController>();
    TestController home = new TestController(logger);
    string result = home.GetMessage();
    Assert.AreEqual(result, "Hi! Reader");
}
```

Summary

A unit test is a code that helps us in verifying the expected behavior of the other code in isolation. Here “In isolation" means there is no dependency between the tests. This is a better idea to test the application code before it goes for quality assurance (QA).

7 18.8m 6 2

9 11

Type your comment here and press Enter Key (Minimum 10 characters)

My code is. driver.FindElement(By.ClassName("IBM")).Click();(but this is not returni g any result. Can you please help me

Shivangi Patel Nov 19, 2018

1739 6 0 0 1 Reply

First just I want to clarify this is unittest that run on server side. So it is validating only C# method only. It is not able to run you UI code. There is different framework available to run unit test from UI. Hope this will make sense.

Jignesh Trivedi Nov 21, 2018

7 55.1k 18.8m 0

My code is

Shivangi Patel Nov 19, 2018

1739 6 0 0 0 Reply

In my application i have one search pleace where i enter text and click method but i am not getting result

Shivangi Patel Nov 19, 2018

1739 6 0 0 0 Reply

Currently i am working on project and i have just started automation with visual studio c# . Unit testing with MS test framework.

Shivangi Patel Nov 19, 2018

1739 6 0 0 0 Reply

Hello, I am shivangi patel. I need help for writing code.

Shivangi Patel Nov 19, 2018

1739 6 0 0 0 Reply

Jignesh: thanks for nice article. will you please post an article on unit test EF based repository in memory instead of mock library. thanks

Tridip Bhattacharjee Apr 26, 2018

818 1.2k 12.5k 0 0 Reply

Jignesh Trivedi
Apr 26, 2018
7 55.1k 18.8m
1

Please show how we can test EF data access code using mstest.

Tridip Bhattacharjee
Feb 12, 2018
818 1.2k 12.5k
0 0 Reply

I have one request. sir can you please write step-by-step article on in-memory repository testing in asp.net mvc5.

Tridip Bhattacharjee
Feb 08, 2018
818 1.2k 12.5k
2 0 Reply

cus

C# Corner

How to protect yourself from online hacking

C#Corner

ASK A QUESTION

CONTRIBUTE

TRENDING UP

01 Create And Build A Facebook Bot Using Microsoft Bot Framework

02 Hosting Static Website With Azure Storage Service

03 SharePoint Framework - Project Upgrade

04 SharePoint Framework - Call Azure AD Secured Function

05 Create And Build FAQ Chatbot Using Azure Bot Service Without Any Coding

06 Office 365 CLI Overview

07 SharePoint Framework - Call Azure Function

08 Creating And Embedding A Bot Application In Your Blogger

cus

C# Corner

How to protect yourself from online hacking

10 Getting Started With Machine Learning .NET For

C# Corner

ASK A QUESTION

CONTRIBUTE

View All ( )

cdata
Know SQL?
Connect to 100+ data sources (SaaS, NoSQL, Big Data, etc.) through standard SQL queries.
ODBC | JDBC | ADO.NET | ODATA
SAP
LEARN MORE

Philadelphia
New York
London
Delhi

JOIN C# CORNER

and millions of developer friends worldwide.

Enter your email address

Sign Up

Learn ASP.NET MVC Learn ASP.NET Core Learn Python Learn JavaScript Learn Xamarin Learn Oracle More...

cus

C# Corner

How to protect yourself from online hacking