**NUnit and Moq**

**1. NUnit-Handson:-**

**CalcLibrary:-**

namespace CalcLibrary

{

public class Calculator

{

public int Add(int a, int b) => a + b;

public int Subtract(int a, int b) => a - b;

public int Multiply(int a, int b) => a \* b;

public int Divide(int a, int b) => b != 0 ? a / b : 0;

}

}

**CalculatorTests.cs :-**

using NUnit.Framework;

using CalcLibrary;

namespace CalculatorTests

{

[TestFixture]

public class CalculatorTests

{

private Calculator calculator;

[SetUp]

public void SetUp()

{

calculator = new Calculator();

}

[TestCase(2, 3, 5)]

[TestCase(-1, -1, -2)]

[TestCase(0, 0, 0)]

public void AddTests(int firstNumber, int secondNumber, int expectedSum)

{

int actualSum = calculator.Add(firstNumber, secondNumber);

Assert.That(actualSum, Is.EqualTo(expectedSum), $"Expected {expectedSum} but got {actualSum}");

}

[TestCase(5, 3, 2)]

public void SubtractTests(int firstNumber, int secondNumber, int expectedDifference)

{

int actualDifference = calculator.Subtract(firstNumber, secondNumber);

Assert.That(actualDifference, Is.EqualTo(expectedDifference), $"Expected {expectedDifference} but got {actualDifference}");

}

}

}

**OUTPUT :-**

A black and white line

AI-generated content may be incorrect.

1. **Moq-Handson**

**Write Testable Code with Moq:-**

1. IMailSender Interface:-

namespace CustomerCommLib

{

public interface IMailSender

{

bool SendMail(string toAddress, string message);

}

}

2. MailSender Class:-

using System.Net;

using System.Net.Mail;

namespace CustomerCommLib

{

public class MailSender : IMailSender

{

public bool SendMail(string toAddress, string message)

{

MailMessage mail = new MailMessage();

SmtpClient smtpServer = new SmtpClient("smtp.gmail.com");

mail.From = new MailAddress("krishna@gmail.com");

mail.To.Add(toAddress);

mail.Subject = "Test Mail";

mail.Body = message

smtpServer.Port = 587;

smtpServer.Credentials = new NetworkCredential("username", "password");

smtpServer.EnableSsl = true;

smtpServer.Send(mail);

return true;

}

}

}

**3. CustomerComm Class :-**

namespace CustomerCommLib

{

public class CustomerComm

{

private readonly IMailSender \_mailSender;

public CustomerComm(IMailSender mailSender)

{

\_mailSender = mailSender;

}

public bool SendMailToCustomer()

{

string email = "vamsi@abc.com";

string message = "Some Message";

\_mailSender.SendMail(email, message);

return true;

}

}

}

**4. Unit Test using Moq:-**

using NUnit.Framework;

using Moq;

using CustomerCommLib;

namespace CustomerComm.Tests

{

[TestFixture]

public class CustomerCommTests

{

private CustomerCommLib.CustomerComm customerComm;

[OneTimeSetUp]

public void Setup()

{

var mockMailSender = new Mock<IMailSender>();

mockMailSender.Setup(m => m.SendMail(It.IsAny<string>(), It.IsAny<string>())).Returns(true);

customerComm = new CustomerCommLib.CustomerComm(mockMailSender.Object);

}

[TestCase]

public void SendMailToCustomer\_ReturnsTrue()

{

bool result = customerComm.SendMailToCustomer();

Assert.That(result, Is.True);

}

}

}

