

CalcLibary Project (Fully Implemented)

Step 1:

namespace CalcLibary

{

public interface ICalcOperations

{

float Add(float n1,float n2);

}

}

Step2:

namespace CalcLibary

{

public class CalcOperations : ICalcOperations

{

public float Add(float n1, float n2)

{

return n1 + n2;

}

}

}

MoneyExchangeLibrary Project (Partially Implemented)

Step3:

namespace MoneyExchangeLibrary

{

public interface IConversion

{

int GetUSDRateforINR();

//int Signature1();

//int Signature2();

//int Signature3();

}

}

Banking Library (Testable Project)

//Added...

using CalcLibary;

using MoneyExchangeLibrary;

namespace BankLibrary

{

public class BankOperations

{

private readonly IConversion \_conv;

private readonly ICalcOperations \_calc;

public BankOperations()

{

//Empty constructor

}

//In this Constructor, we are doing Dependency Injection...

public BankOperations(ICalcOperations calc, IConversion ExternalObjRef)

{

\_calc = calc;

\_conv = ExternalObjRef;

}

public float CalculateTotalIncome(float IncomeFromBusiness,float IncomeFromJob)

{

return \_calc.Add(IncomeFromBusiness, IncomeFromJob);

}

public float CalculateINRvalueforUSDSalary(int SalaryinUSD)

{

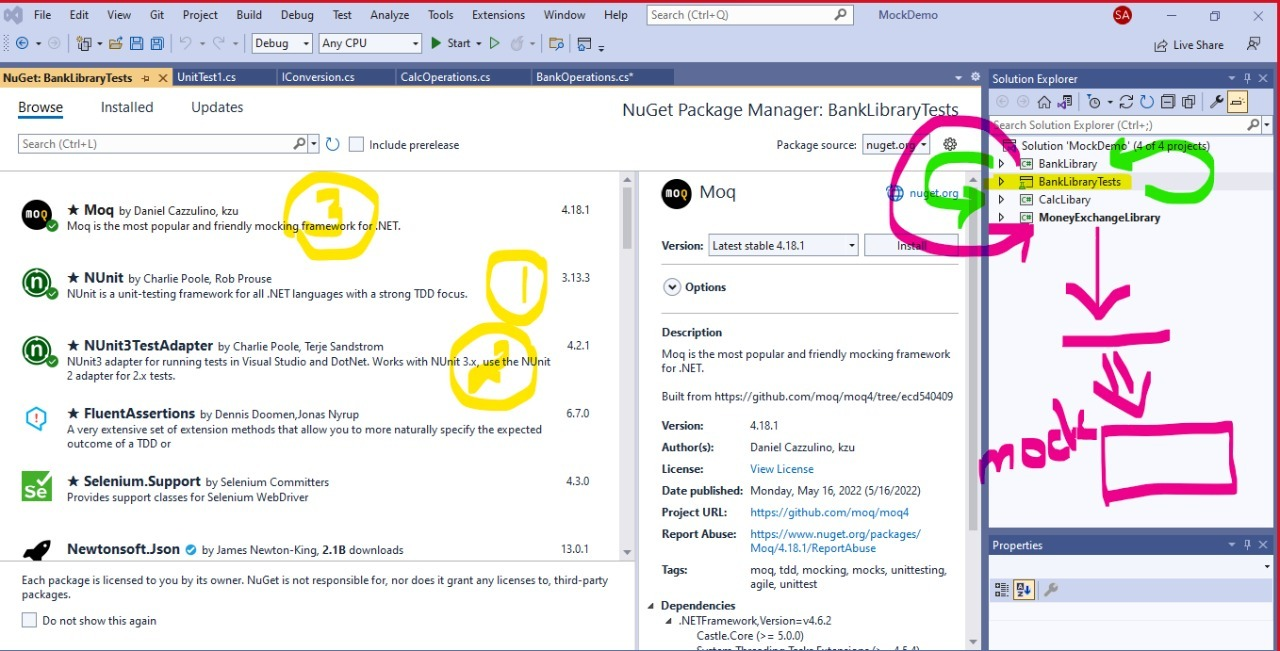
return SalaryinUSD \* \_conv.GetUSDRateforINR();

}

}

}

BankLibraryTests (Testing Project)



using System;

//using Microsoft.VisualStudio.TestTools.UnitTesting;

//Added..

using NUnit.Framework;

using Moq;

using CalcLibary;

using BankLibrary;

using MoneyExchangeLibrary;

namespace BankLibraryTests

{

[TestFixture]

public class BankOperationsTests

{

[Test]

public void CalculateTotalIncome\_Accepts2Incomes\_ReturnsTotalIncome()

{

//Arrange

CalcOperations calc = new CalcOperations();

int ExpectedResult = 20000;

//Act

BankOperations ops1 = new BankOperations(calc,null);

float ActualResult = ops1.CalculateTotalIncome(10000, 10000);

//Assert

Assert.AreEqual(ExpectedResult, ActualResult);

}

[Test]

public void CalculateINRvalueforUSDSalary\_AcceptsSalaryinUSD\_ReturnINRValue()

{

//Arrage

Mock<IConversion> mockObject = new Mock<IConversion>();

mockObject.Setup(meth => meth.GetUSDRateforINR()).Returns(70); //for Time being It will return 70

//mockObject.Setup(meth => meth.Signature1()).Returns(1);

//mockObject.Setup(meth => meth.Signature2()).Returns(2);

//mockObject.Setup(meth => meth.Signature3()).Returns(3);

float expectedResult = 70000.0f;

//Act

BankOperations ops = new BankOperations(null,mockObject.Object);

float actualResult = ops.CalculateINRvalueforUSDSalary(1000);

//Assert

Assert.AreEqual(expectedResult, actualResult);

}

}

}

