Visual COBOL チュートリアル

Visual Studio：.NET COBOL の単体テスト

1　目的

本チュートリアルでは、.NET COBOL プログラムに対するテスト作成、実行方法、および、テスト結果を表示させる方法の習得を目的としています。

MFUnit は、Visual COBOL に搭載された xUnit 系の単体テストフレームワークです。xUnit はオブジェクト指向型の単体テストフレームワーク SUnit に起源を持つ JUnit や RUnit 等の単体テストフレームワークの総称です。MFUnit は xUnit の設計アーキテクチャーや仕組みは取り入れつつも COBOL 開発者にとって扱いやすい手続き型の COBOL を対象とした単体テストフレームワークという設計思想の下、開発されました。

MFUnit は COBOL 開発作業に以下の利点を提供します。

- テストを繰返し実行させることができるため、修正作業時などのテスト工数の削減が見込める
- Jenkins などの継続的インテグレーション（Continuous Integration）ツールと連携によりテストの自動化が行え、DevOps サイクルの導入が足がかりを作れる

2　前提

- 本チュートリアルで使用したマシン OS ： Windows 11
- Visual COBOL 10.0 for Visual COBOL 2022 がインストール済みであること

本資料は、.NET COBOL に対する単体テストフレームワークの利用方法を記載したチュートリアルです。ネイティブ COBOL の単体テスト実現方法については、別チュートリアルを参照ください。

下記のリンクから事前にチュートリアル用のサンプルファイルをダウンロードして、任意のフォルダーに解凍しておいてください。

サンプルプログラムのダウンロード

内容

3 チュートリアル手順

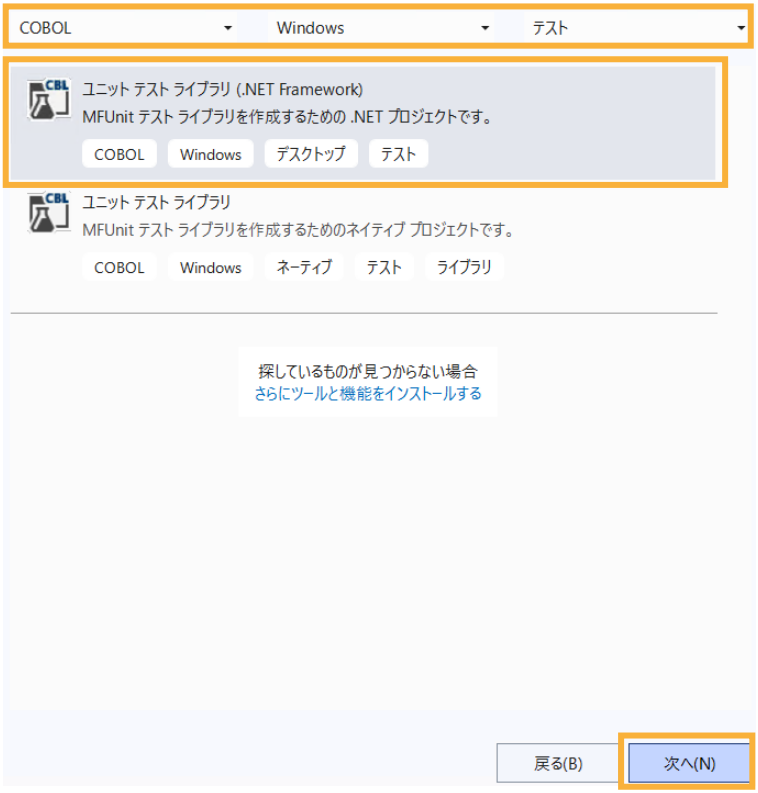
3.1 IDE からの実行

3.1.1 チュートリアルプロジェクトの作成

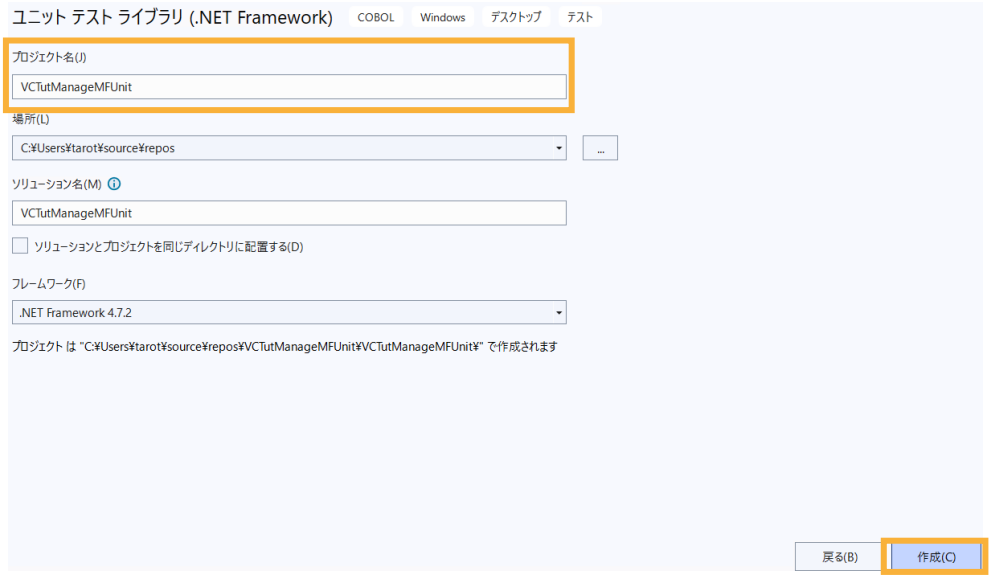
1) スタートメニューより、Visual Studio を起動します。

2) [新しいプロジェクトの作成] をクリックします。

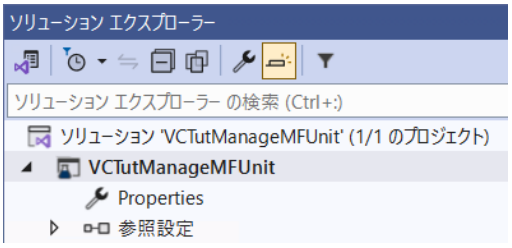
3) 言語に “COBOL”、プラットフォームに “Windows”、プロジェクトの種類に “テスト” を選択し、「ユニットテストライブラリ(.NET Framework)」を選択した上で、[次へ(N)] をクリックします。



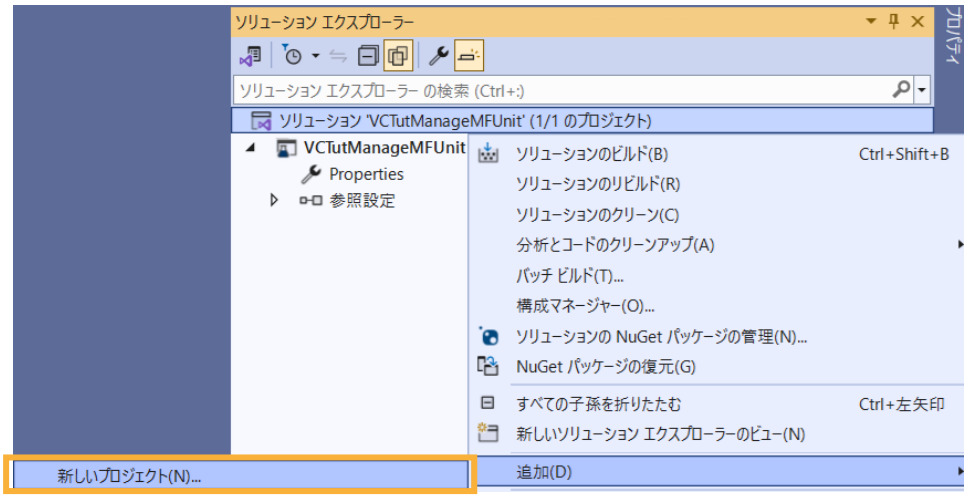
4) プロジェクト名に “VCTutManageMFUnit” を入力し、[作成(C)] をクリックします。



プロジェクトが作成されます。

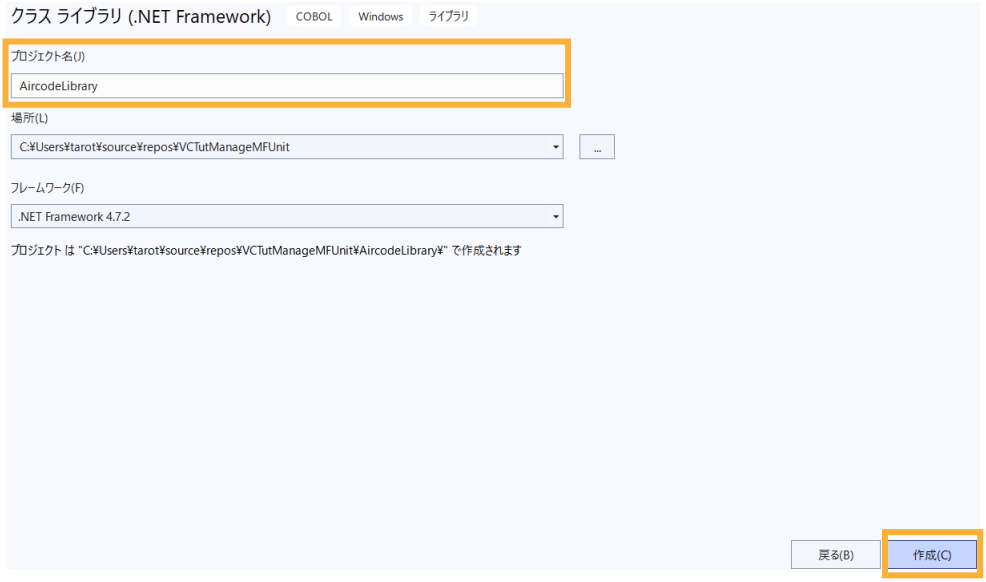


5) VCTutManageMFUnit ソリューション名を選択した状態で、マウスの右クリックにてコンテクストメニューを表示し、[追加(D)] > [新しいプロジェクト(N)] を選択します。

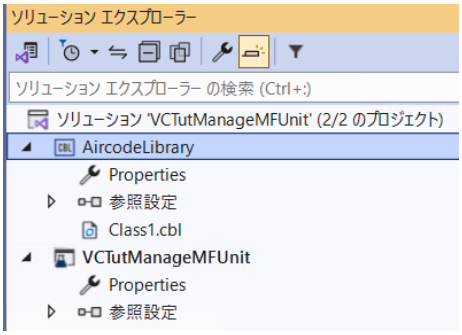


6) 言語に “COBOL” を選択し、プラットフォームに “Windows”、プロジェクトの種類に “ライブラリ” を選択し、表示された一覧から「クラスライブラリ（.NET Framework）」を選択し、[次へ(N)] をクリックします。

7) プロジェクト名に “AircodeLibrary” と入力し、[作成(C)] をクリックします。

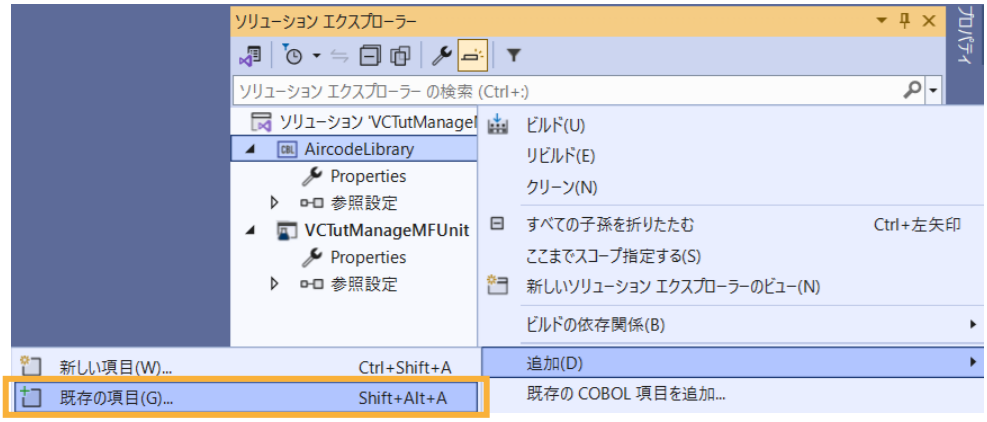


プロジェクトが作成されます。

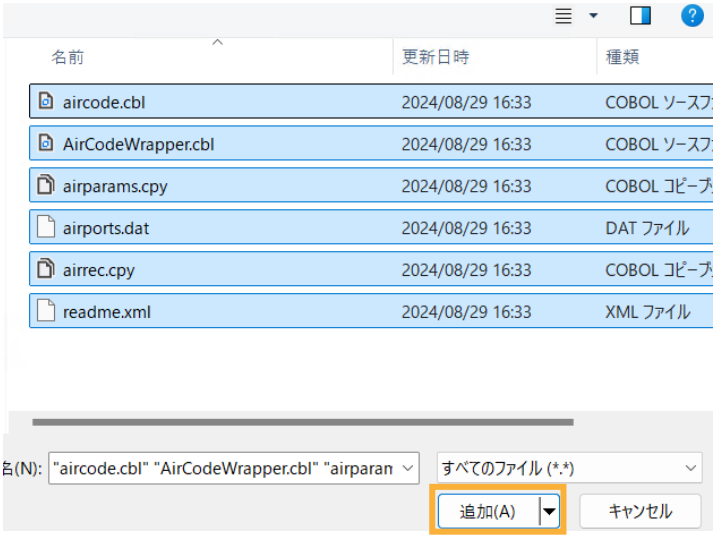


AircodeLibrary プロジェクト配下の Class1.cbl は不要のため、削除してください。

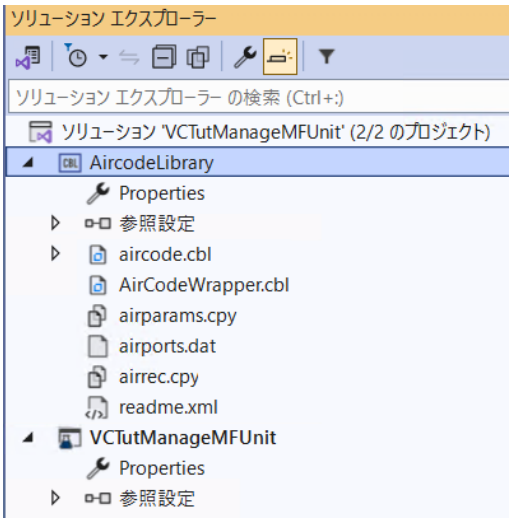
8) AircodeLibrary プロジェクト名を選択した状態で、マウスの右クリックにてコンテクストメニューを表示し、[追加(D)] > [既存の項目(G)] を選択します。



9) サンプルファイルを展開したフォルダー内の AirportDemoMFUnit フォルダー配下を選択し、”すべてのファイル(*.*)” を選択した結果、表示される全てのファイルを選択した上で、[追加(A)] をクリックします。

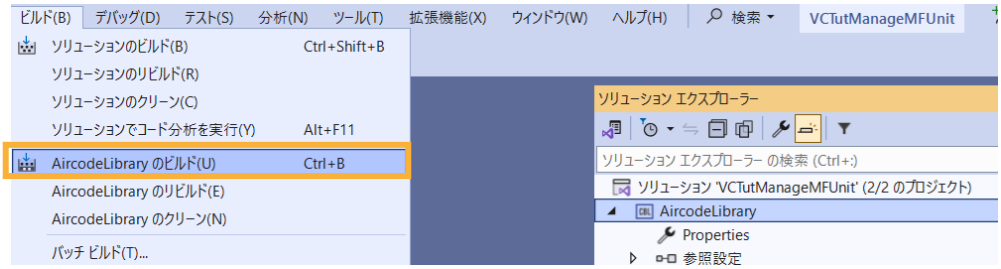


プロジェクトが、以下のようになります。

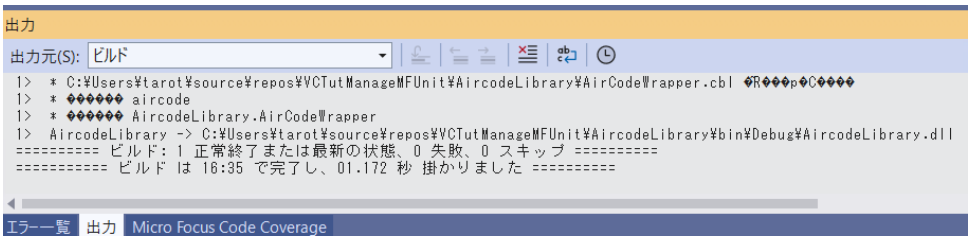


aircode.cbl はレガシーなネイティブ COBOL プログラムです。 AirCodeWrapper.cbl は .NET 言語から容易に COBOL プログラムを呼び出すための Wrapper クラスです。

10) AircodeLibrary プロジェクト名を選択した状態で、Visual Studio IDE メニューより、[ビルド(B)] > [AircodeLibrary のビルド(U)] を選択します。



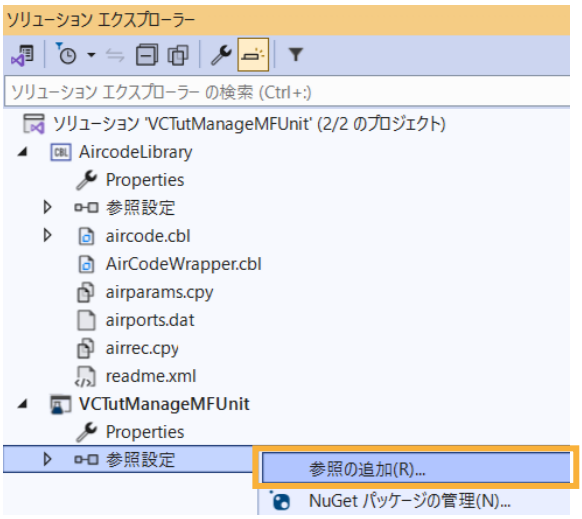
プロジェクトのビルド処理が行われ、DLL が作成されます。



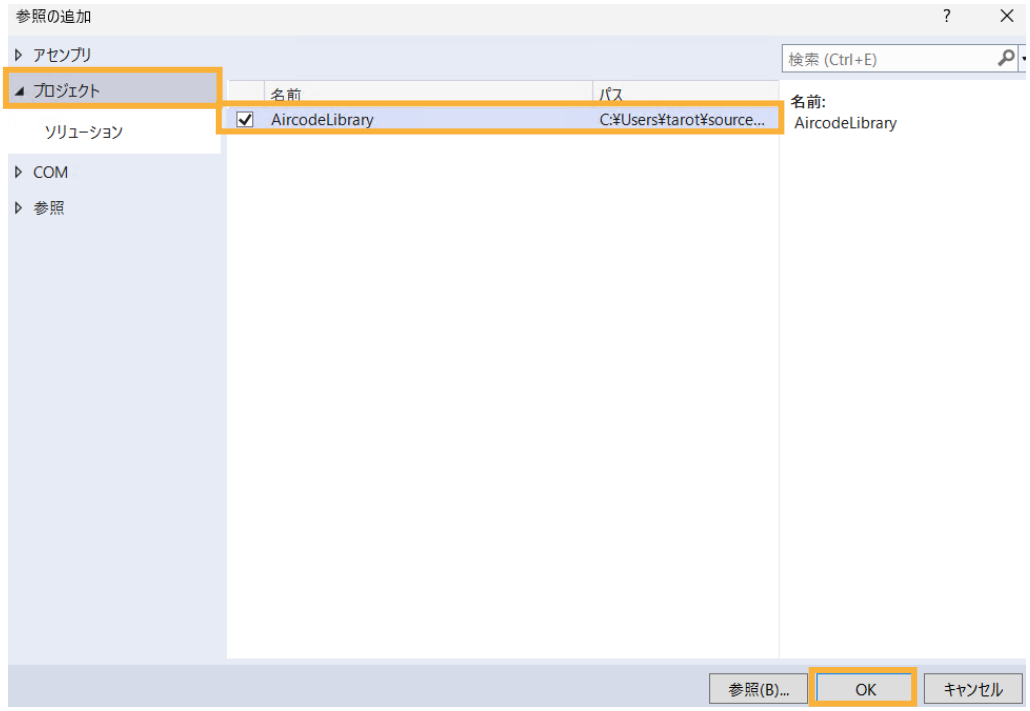
3.1.2 MFUnit テストの作成

さきほど追加した AircodeWrapper.cbl に対する単体テストを作成します。

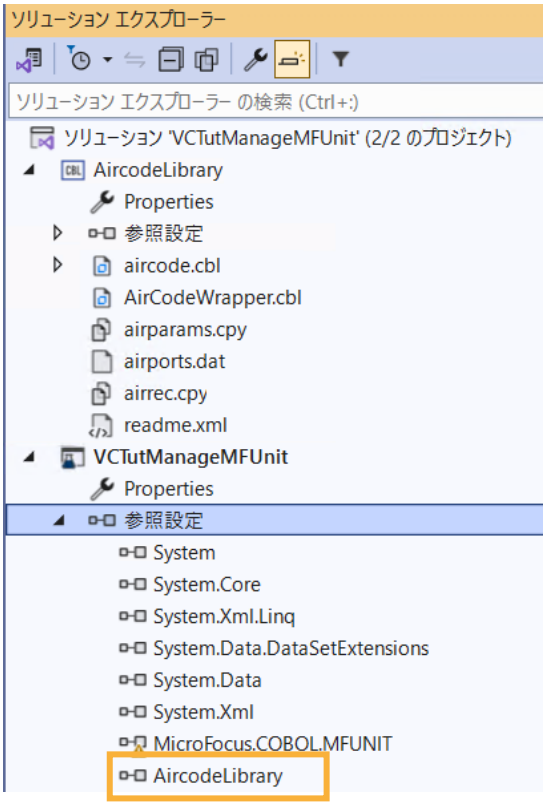
1) VCTutManageMFUnit プロジェクト配下の「参照設定」を選択した状態で、マウスの右クリックにてコンテクストメニューを表示し、[参照の追加(R)] を選択します。



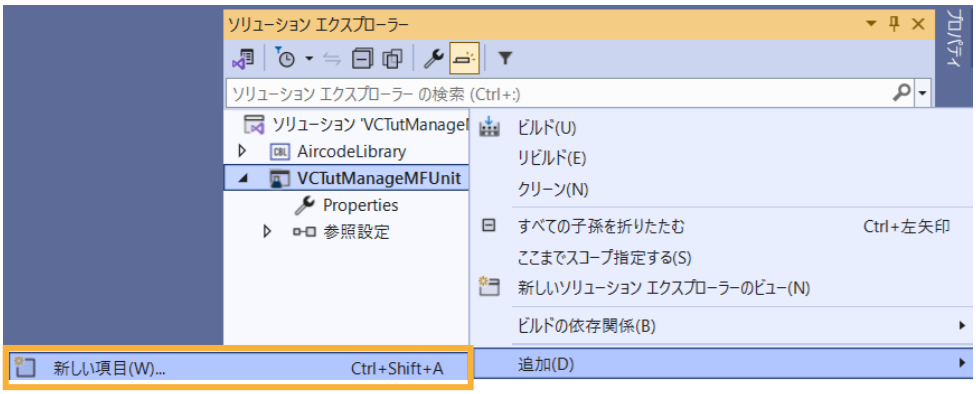
2) [プロジェクト] を選択し、AircodeLibrary 項目のチェックを行った後、[OK] をクリックします。



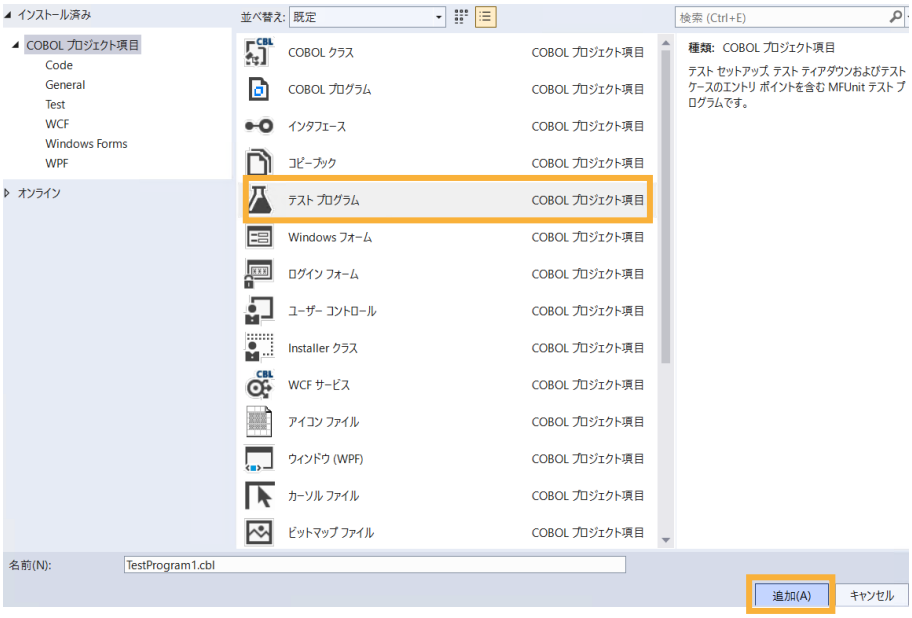
参照設定に、AircodeLibrary が追加されます。



3) VCTutManageMFUnit プロジェクト名を選択し、[追加(D)] > [新しい項目(W)] を選択します。



4) [テスト プログラム] を選択し、[追加(A)] をクリックします。



5) 新規のテストケース（羽田・ロンドンヒースロー空間間の距離（km）のテスト）を追加した上で、実行を行ないます。サンプルファイルを展開したフォルダー内の TestProgram1.cbl で、現在の TestProgram1.cbl を上書きしてください。

これは、テストケース “testDistance” を途中まで作成したものになります。プログラムを確認すると、MFU-TC-SETUP-PREFIX, MFU-TC-PREFIX, MFU-TC-TEARDOWN-PREFIX から始まる “testDistance” の 3 entry が定義されていることが分かります。MFUnit では、テストを下記のように決められた手順で実行しています。

① entry MFU-TC-SETUP-PREFIX & “testDistance”

② entry MFU-TC-PREFIX & “testDistance”

③ entry MFU-TC-TEARDOWN-PREFIX & “testDistance”

MFU-TC-SETUP-PREFIX で始まる entry にて、テストの前処理を定義できます。前処理の代表例としては、ファイルをあらかじめオープンしておくなどが考えられます。一方、MFU-TC-TEARDOWN-PREFIX で始まる entry では、テスト実行後の処理を定義できます。前処理でオープンしたファイルをクローズするような処理が該当します。前処理、後処理ともに省略可能です。

注意）

2 5行目に、airports.dat へのパスが指定されているため、環境に合わせて修正してください。

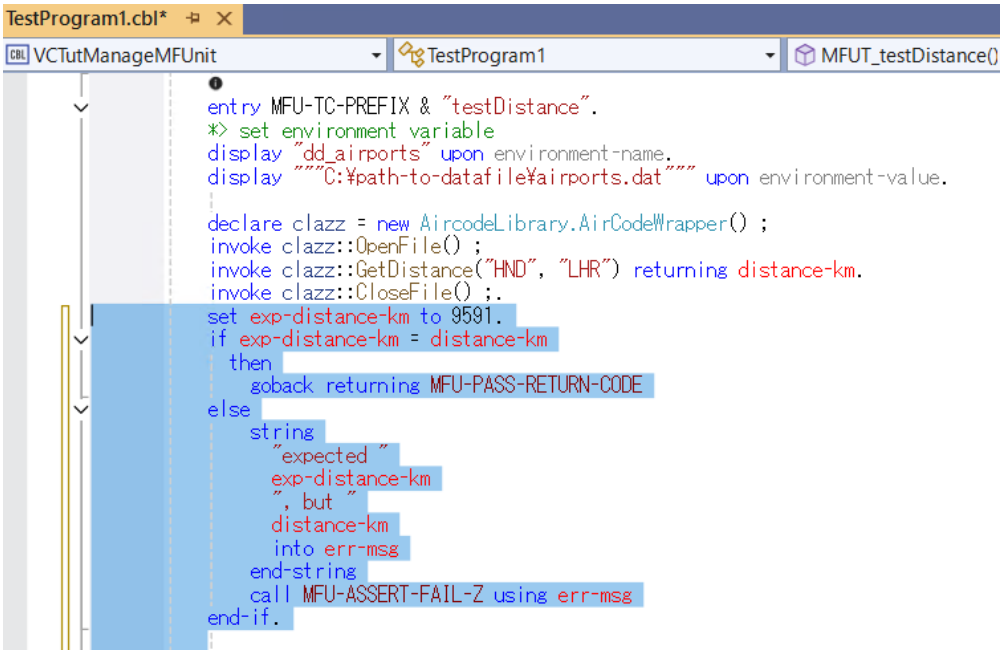
テスト本体である MFU-TC-PREFIX を確認すると、下記のように結果検証コードが実装されていません。

```
entry MFU-TC-PREFIX & "testDistance".
*> set environment variable
display "dd_airports" upon environment-name.
display """C:¥path-to-datafile¥airports.dat""" upon environment-value.

declare clazz = new AircodeLibrary.AirCodeWrapper() ;
invoke clazz::OpenFile() ;
invoke clazz::GetDistance("HND", "LHR") returning distance-km.
invoke clazz::CloseFile() ;.
```

このテストを実装するため、以下のコードを上記 entry 句の最終位置に含まれるよう、挿入してください。

```
set exp-distance-km to 9591.
if exp-distance-km = distance-km
  then
    goback returning MFU-PASS-RETURN-CODE
else
    string
      "expected "
      exp-distance-km
      ", but "
      distance-km
      into err-msg
    end-string
    call MFU-ASSERT-FAIL-Z using err-msg
end-if.
```



補足）

テスト失敗時の記述方法として、成功時同様に、戻り値で返す方法は、以下の通りです。

```
if exp-distance-km = distance-km
then
    goback returning MFU-PASS-RETURN-CODE
else
    string
        "expected "
        exp-distance-km
        ", but "
        distance-km
        into err-msg
```

```
                end-string
                display err-msg
                goback returning MFU-FAIL-RETURN-CODE
            end-if.
```

戻り値 MFU-FAIL-RETURN-CODE を利用する場合、テスト結果を確認するためにエラー情報を、display などで出力する必要があります。

3.1.3 MFUnit の実行

本テスト対象のプログラムは、環境変数で設定された空港情報が保存されたデータファイルを参照するため、手順内で設定を行います。

1) Visual Studio IDE メニューより、[表示(V)] > [Micro Focus Unit Testing] を選択します。

2) Micro Focus Unit Testing ビューより、[すべて実行] をクリックします。

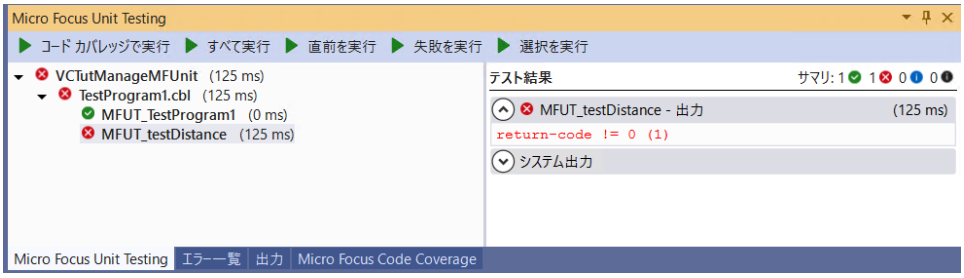
以下のように全て緑色のマークが設定されます。緑色は、テストに成功したことを示します。

3) エラーケースを確認します。「TestProgram1.cbl」をエラーとなるように修正した上で、再度、Micro Focus Unit Testing ビューより、[すべて実行] をクリックします。

なお、本例では、テストプログラム内に記載されていた期待値 9591 を 9590 に修正しています。

```
declare clazz = new AircodeLibrary.AirCodeWrapper() .
invoke clazz::OpenFile() .
invoke clazz::GetDistance("HND", "LHR") returning distance-km.
invoke clazz::CloseFile() .
set exp-distance-km to 9590.
if exp-distance-km = distance-km
then
    goback returning MFU-PASS-RETURN-CODE
else
    string
        "expected "
        exp-distance-km
```

MFUT_testDistance のテストで、エラーが発生したことが一覧から判断できます。



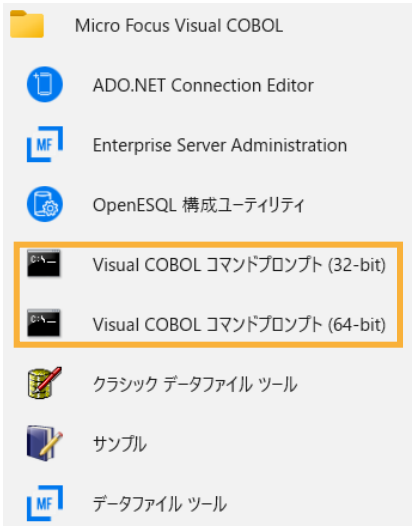
4) エラーケースの確認用に修正したコードをもとに戻し、保存します。

3.2 コマンドラインからの実行

MFUnit によるテストは、Visual Studio 上の画面からではなく、コマンドライン上からも行なうことができます。従来のスタイルでのテスト作業の効率化を図ることができ、Jenkins などの CI ツールと連携する事で、テストの自動実行を行なえるため、品質担保や作業工数の削減が見込めます。

ここでは、IDE で作成したテストプログラムを使用して、コマンドラインからテストを実行する方法について紹介します。

1) スタートメニューより、実行環境に合わせた [Visual COBOL コマンドプロンプト] をクリックします。



2) 作業フォルダーを作成し、作成したフォルダーに移動します。

```
C:¥>mkdir VCCommandTutorial && cd VCCommandTutorial
C:¥VCCommandTutorial>
```

3) IDE で作成したソリューションが保存されているフォルダーを VS_SOLUTION_PATH に指定した上で、下記コマンドを実行します。

- set VS_SOLUTION_PATH=c:¥vs_solution_path
- cobol %VS_SOLUTION_PATH%¥AircodeLibrary¥AircodeWrapper.cbl ilsource(%VS_SOLUTION_PATH%¥AircodeLibrary¥aircode.cbl) iloutput(.) ilgen(sub);
- cobol %VS_SOLUTION_PATH%¥VCTutManageMFUnit¥TestProgram1.cbl iloutput(.) ilgen(sub) sourceformat(variable) ilref(AircodeWrapper.dll);

注意）
VS_SOLUTION_PATH は、各環境に合わせて修正してください。

```
C:¥VCCommandTutorial>set VS_SOLUTION_PATH=c:¥vs_solution_path

C:¥VCCommandTutorial>cobol %VS_SOLUTION_PATH%¥AircodeLibrary¥AircodeWrapper.cbl ils
ource(%VS_SOLUTION_PATH%¥AircodeLibrary¥aircode.cbl) iloutput(.) ilgen(sub);
Micro Focus COBOL
Version 10.0 (C) Copyright 1984-2024 Micro Focus or one of its affiliates.
* チェック終了：エラーはありません

C:¥VCCommandTutorial>cobol %VS_SOLUTION_PATH%¥VCTutManageMFUnit¥TestProgram1.c
bl iloutput(.) ilgen(sub) sourceformat(variable) ilref(AircodeWrapper.dll);
Micro Focus COBOL
Version 10.0 (C) Copyright 1984-2024 Micro Focus or one of its affiliates.
* チェック終了：エラーはありません

C:¥VCCommandTutorial>
```

2 つの DLL ファイルが作成されます。

```
C:¥VCCommandTutorial>dir
 ドライブ C のボリューム ラベルがありません。

 C:¥VCCommandTutorial のディレクトリ

2024/08/30  09:54    <DIR>          .
2024/08/30  09:53            14,848 AircodeWrapper.dll
2024/08/30  09:54             8,192 TestProgram1.dll
               2 個のファイル              23,040 バイト
C:¥VCCommandTutorial>
```

4) プロンプト上で下記コマンドを実行し、MFUnit を実行します。

```
set dd_airports=%VS_SOLUTION_PATH%¥AircodeLibrary¥airports.dat
mfurunil -report:junit -outdir:mfunit_result TestProgram1.dll
```

補足）
.NET COBOL プログラムに対する単体テストは、mfurun ではなく、mfurunil を使用します。

```
C:¥VCCommandTutorial>set dd_airports=%VS_SOLUTION_PATH%¥AircodeLibrary¥airports.da
t

C:¥VCCommandTutorial>mfurunil -report:junit -outdir:mfunit_result TestProgram1.dll
Micro Focus COBOL - mfurunil Utility
Unit Testing Framework for Windows/.Net/64

Fixture : TestProgram1

Test Run Summary
 Overall Result             Passed
 Tests run                   2
 Tests passed                2
 Tests failed               0
 Total execution time       130

C:¥VCCommandTutorial>
```

outdir オプションにより、出力結果は mfunit_result フォルダーに保存されます。

```
C:¥VCCommandTutorial>dir mfunit_result
 ドライブ C のボリューム ラベルがありません。
C:¥VCCommandTutorial¥mfunit_result のディレクトリ
2024/08/30   09:59     <DIR>            .
2024/08/30   09:59     <DIR>            ..
2024/08/30   09:59                467 TEST-MFUT_testDistance.xml
2024/08/30   09:59                273 TEST-MFUT_TestProgram1.xml
2024/08/30   09:59              1,180 TestProgram1-report.txt
               3 個のファイル                 1,920 バイト
C:¥VCCommandTutorial>
```

免責事項

ここで紹介したソースコードは、機能説明のためのサンプルであり、製品の一部ではございません。ソースコードが実際に動作するか、御社業務に適合するかなどに関しまして、一切の保証はございません。 ソースコード、説明、その他すべてについて、無謬性は保障されません。
ここで紹介するソースコードの一部、もしくは全部について、弊社に断りなく、御社の内部に組み込み、そのままご利用頂いても構いません。
本ソースコードの一部もしくは全部を二次的著作物に対して引用する場合、著作権法の精神に基づき、適切な扱いを行ってください。