
Micro Focus Visual COBOL チュートリアル

COBOL 開発 : Visual Studio – コードカバレッジ機能

1. 目的

本チュートリアルは、ネイティブ COBOL プログラムに対するコードカバレッジを表示させる手順の習得を目的としています。コードカバレッジ機能は、COBOL 開発作業に以下の利点を提供します。

- テスト未実施箇所の検出
追加テストを適切に行う事でプログラム品質の向上が見込まれます
- テスト終了後のカバレッジ率算出
品質指標値を満たしているかの確認ができ、その証左として利用できます

なお、コードカバレッジ機能は ネイティブ COBOL でのみ利用可能です。また、マネージ COBOL プロジェクトに追加されたネイティブ COBOL プログラムへの利用もできません。

2. 前提

- 本チュートリアルで使用したマシン OS : Windows 11
- Micro Focus Visual COBOL 8.0J for Visual Studio がインストール済みであること

本資料では、Visual COBOL 8.0J for Visual Studio 2022 を使用しています。開発環境については、ホームページをご覧ください。

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

1. 目的
2. 前提
3. チュートリアル手順の概要
 - 3.1. IDE からの実行
 - 3.1.1. Visual Studio の起動
 - 3.1.2. チュートリアルプロジェクトの作成
 - 3.1.3. カバレッジを有効にしたアプリケーション実行
 - 3.1.4. 複数のカバレッジ情報のマージ
 - 3.1.5. 既存カバレッジ情報ファイルの読み込み
 - 3.2. コマンドラインからの実行
 - 3.2.1. コマンドラインからのコードカバレッジ機能の利用
 - 3.2.2. ユーティリティ機能の紹介

3. チュートリアル手順の概要

3.1. IDE からの実行

3.1.1. Visual Studio の起動

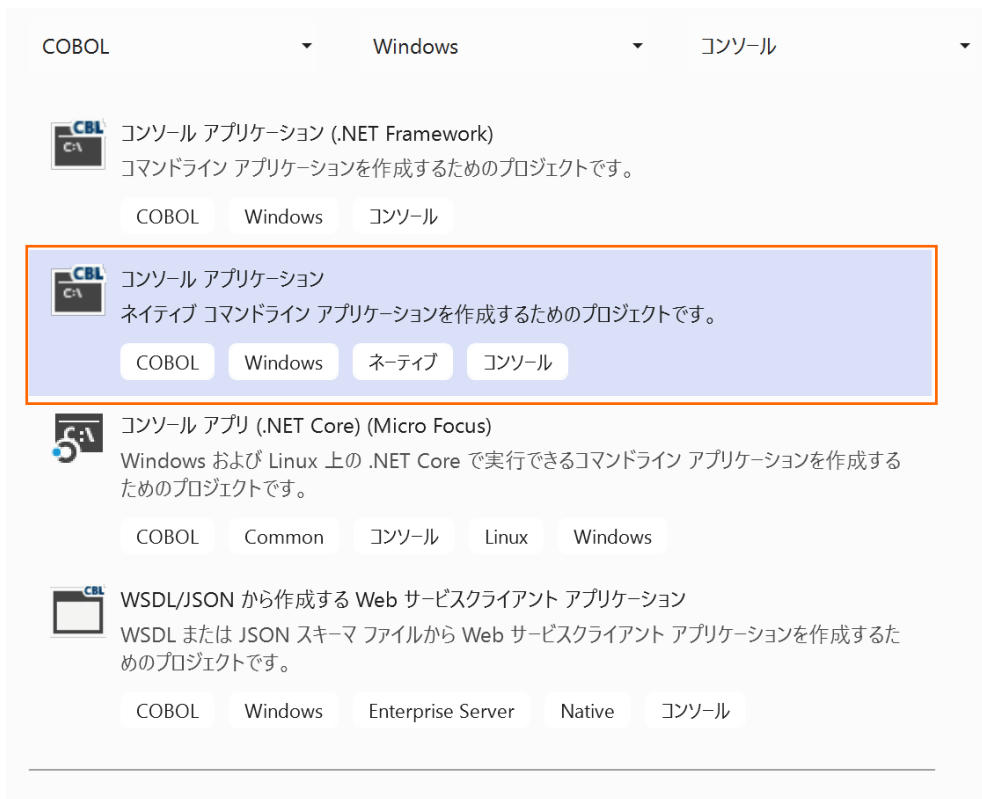
- 1) スタートメニューより、Visual Studio 2022 を起動します。

3.1.2. チュートリアルプロジェクトの作成

- 1) [新しいプロジェクトの作成] をクリックします。



- 2) 言語に“COBOL”、プロジェクトタイプに“コンソール”を選択し、「コンソール アプリケーション」を選択した上で、[次へ(N)] ボタンをクリックします。



COBOL Windows コンソール

 コンソール アプリケーション (.NET Framework)
コマンドライン アプリケーションを作成するためのプロジェクトです。

COBOL Windows コンソール

 コンソール アプリケーション
ネイティブ コマンドライン アプリケーションを作成するためのプロジェクトです。

COBOL Windows ネーティブ コンソール

 コンソール アプリ (.NET Core) (Micro Focus)
Windows および Linux 上の .NET Core で実行できるコマンドライン アプリケーションを作成するためのプロジェクトです。

COBOL Common コンソール Linux Windows

 WSDL/JSON から作成する Web サービスクライアント アプリケーション
WSDL または JSON スキーマ ファイルから Web サービスクライアント アプリケーションを作成するためのプロジェクトです。

COBOL Windows Enterprise Server Native コンソール

- 3) プロジェクト名に “VCTutCoverage” を入力し、[作成(C)] ボタンをクリックします。



新しいプロジェクトを構成します

コンソール アプリケーション COBOL Windows ネーティブ コンソール

プロジェクト名(I)

VCTutCoverage

場所(L)

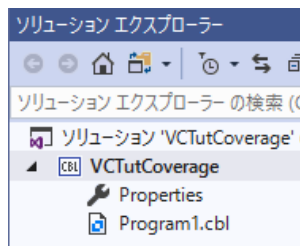
C:\work\

ソリューション名(M) ⓘ

VCTutCoverage

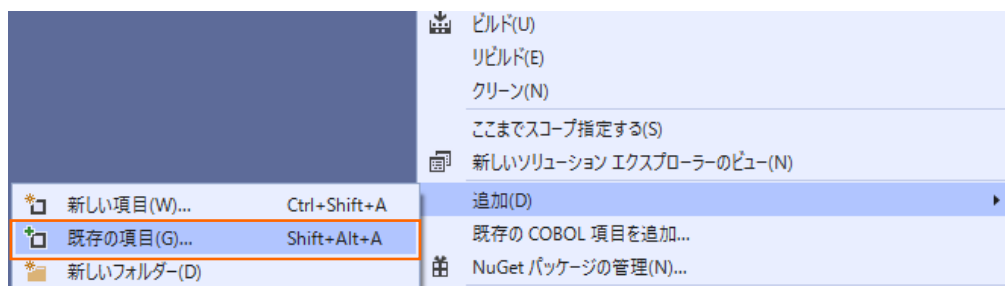
☐ ソリューションとプロジェクトを同じディレクトリに配置する(D)

新規プロジェクトが作成されます。

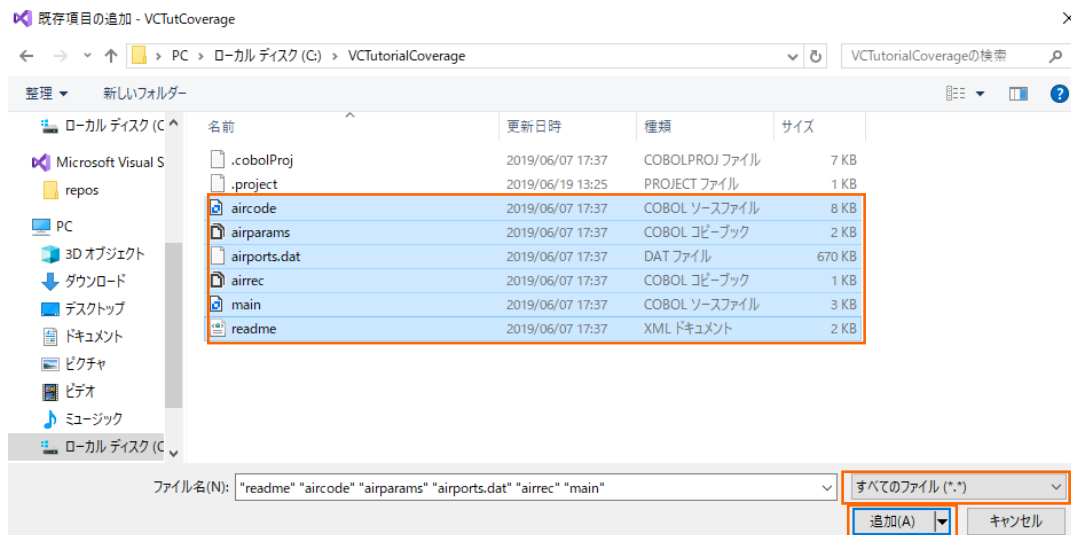


この Program1.cbl は不要なため、削除してください。

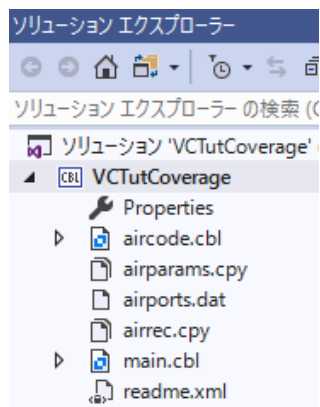
- 4) VCTutCoverage プロジェクト名を選択した状態で、マウスの右クリックにてコンテキストメニューを表示し、[追加(D)] > [既存の項目(G)] を選択します。



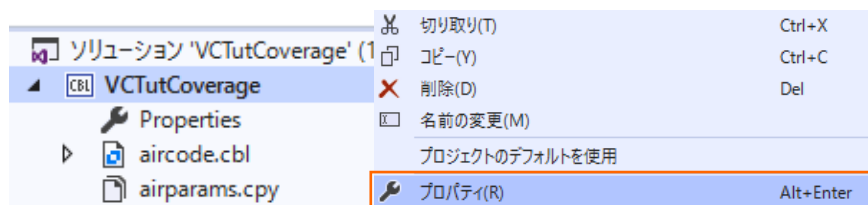
- 5) サンプルプログラムを展開したフォルダを選択し、“すべてのファイル (*.*)” を選択した後、.cobolProj, .project を除く全てのファイルを選択した上で、[追加(A)] ボタンをクリックします



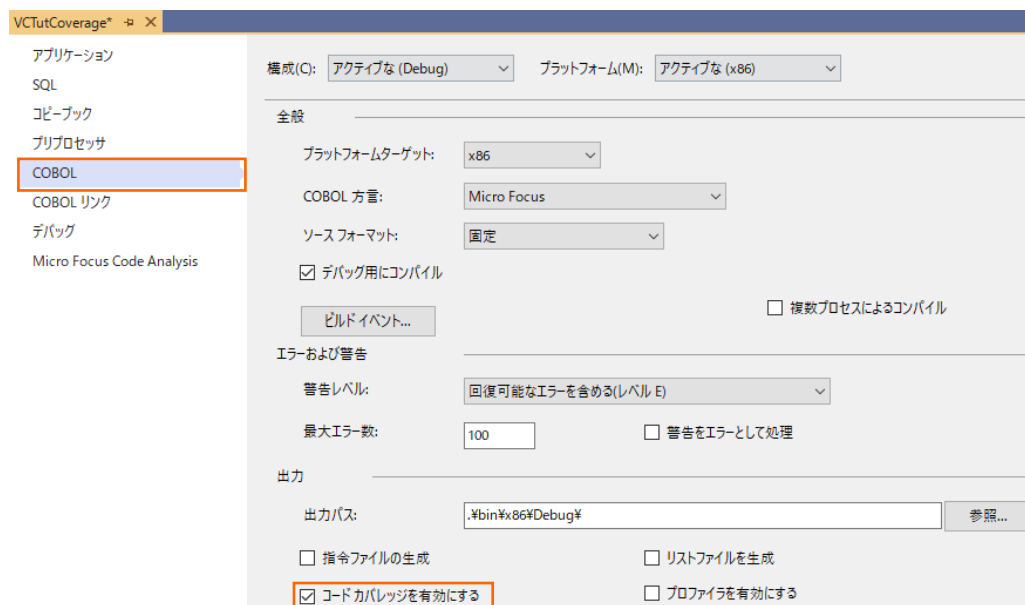
プロジェクトが、以下のようになります。



- 6) VCTutCoverage プロジェクト名を選択し、マウスの右クリックにてコンテキストメニューを表示し、[プロパティ(R)] を選択します。

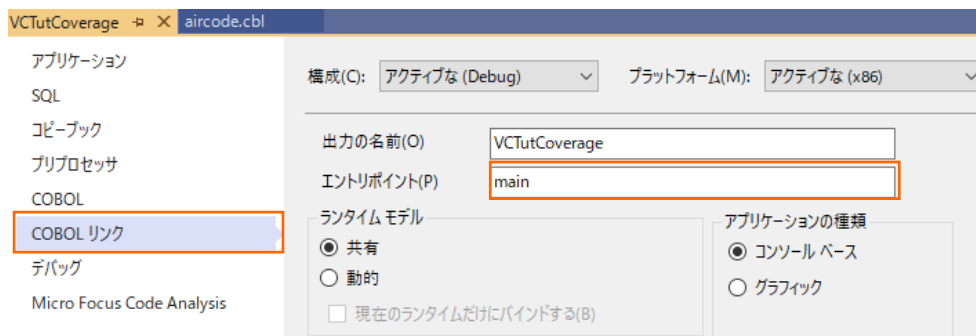


- 7) COBOL タブを選択し、「コードカバレッジを有効にする」にチェックを付けます。



- 8) COBOL リンクタブを選択し、以下の入力を行った後、保存します。

エントリーポイント： main

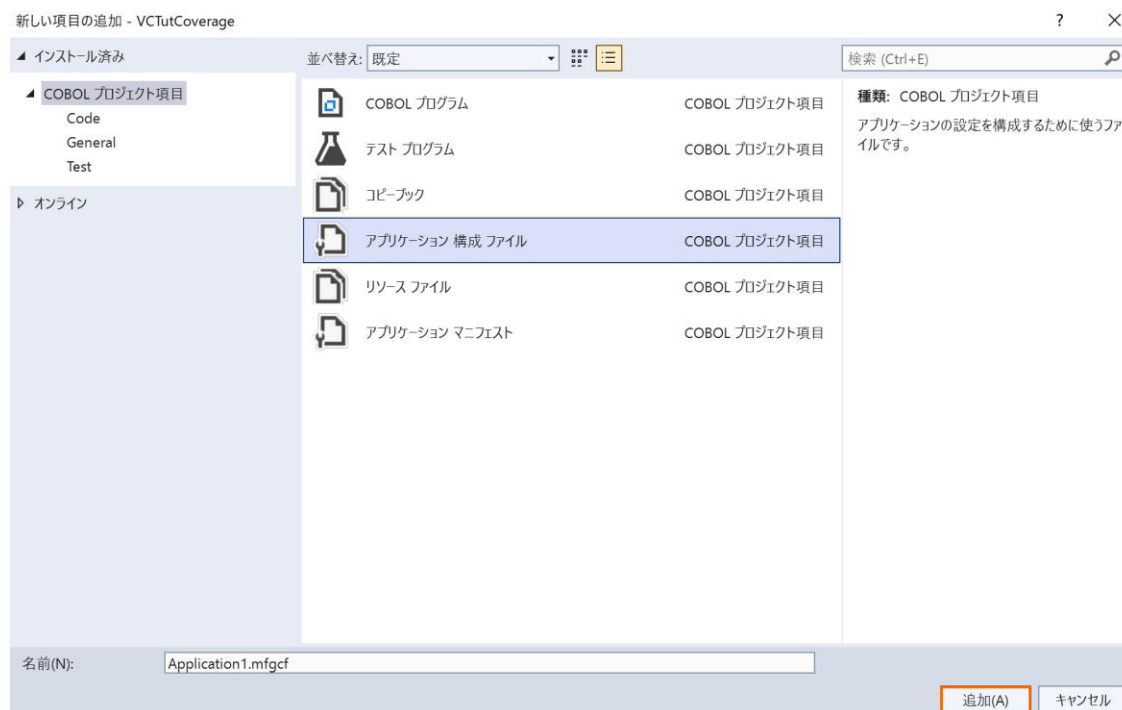


3.1.3. カバレッジを有効にしたアプリケーション実行

- 1) VCTutCoverage 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キストメニューを表示し、[追加(D)] > [新しい項目(W)] を選択します。



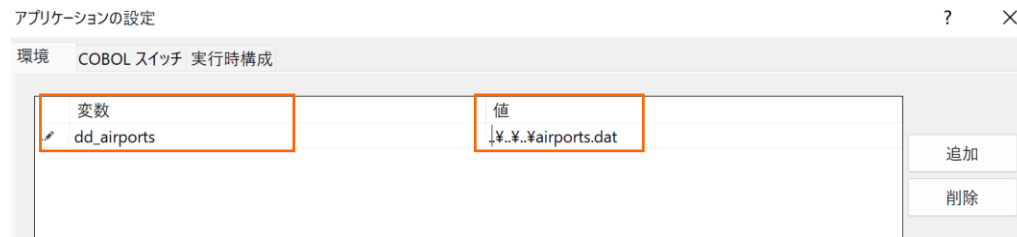
- 2) 「アプリケーション構成ファイル」を選択し、[追加(A)] ボタンをクリックします。



- 3) 追加された「Application1.mfgcf」をクリックし、表示されたダイアログ上で、以下の入力を行い、[OK] ボタンをクリックします。

変数: “dd_airports”

値: “..¥..¥..¥airports.dat”



- 4) Visual Studio IDE メニューより、[デバッグ(D)] > [デバッグの開始(S)] を選択します。



アプリケーション起動後、以下の入力を行います。

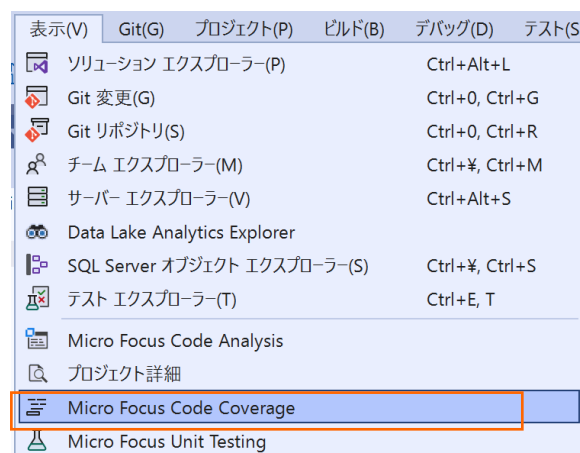
1 回目: “HND△LHR” を入力して Enter (羽田・ロンドンヒースロー空港間の距離を表示)

2 回目: 何も入力せず Enter (プログラムの終了)

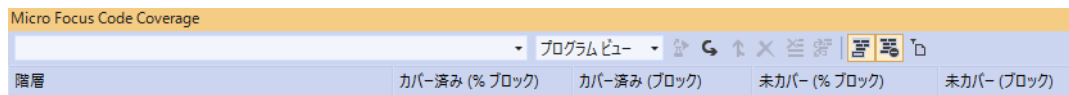
補足) △は半角スペースを表します。



- 5) Visual Studio IDE メニューより、[表示(V)] > [Micro Focus Code Coverage] を選択します。



現時点では、カバレッジ情報が何も表示されていないことを確認します。



コードカバレッジ機能を有効化するためには、アプリケーション実行時にも指示が必要となります。

- 6) Visual Studio IDE メニューより、[デバッグ(D)] > [Micro Focus Code Coverage で開始] を選択します。



再度、アプリケーションが起動するため、以下の入力を行います。

1 回目： “HND△LHR” を入力して Enter（羽田・ロンドンヒースロー空港間の距離を表示）

2 回目： 何も入力せず Enter（プログラムの終了）

補足）△は半角スペースを表します。

実行後、コードカバレッジビューに、カバレッジ情報が表示されることを確認します。

Micro Focus Code Coverage				
Results_2019-06-27_15_22_53.tcz		プログラムビュー		
階層	カバレッジ (% ブロック)	カバレッジ (ブロック)	未カバレッジ (% ブロック)	未カバレッジ (ブロック)
Results_2019-06-27_15_22_53.tcz	67.86 %	38	32.14 %	18
VCtutCoverage	67.86 %	38	32.14 %	18
プログラム: aircode	65.91 %	29	34.09 %	15
節: main	62.5 %	5	37.5 %	3
節: lookup-one-airport	0 %	0	100 %	3

- 7) コードカバレッジビュー上から「lookup-one-airport」をダブルクリックすることで、エディターが開き、カバレッジ結果を表示します。

```
lookup-one-airport section.
  initialize lnk-rec
  move lnk-airport1 to airport
  perform find-airport
  if airport-found = 1
    move f-rec to lnk-rec
  end-if
.

distance-between-airports section.
*> need to find both airports and distance between them
  initialize lnk-distance-result
  move lnk-airport1 to airport
  perform find-airport
  if airport-found = 1
    perform display-airport
    move f-rec to al-rec

    move lnk-airport2 to airport
    perform find-airport
    if airport-found = 1
```

背景色が赤色の箇所が未実行箇所、緑色の箇所が実行済み箇所となります。

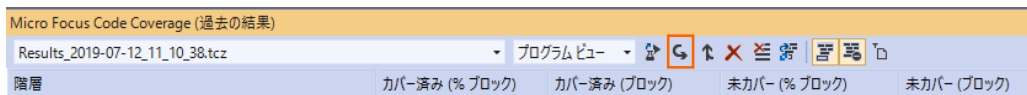
3.1.4. 既存カバレッジ情報ファイルの読み込み

過去に作成したカバレッジ情報ファイルを IDE 上に読み込む方法について学びます。なお、カバレッジ情報ファイルは、IDE 上からの実行だけではなく、後述するコマンドラインから実行された結果も表示できます。

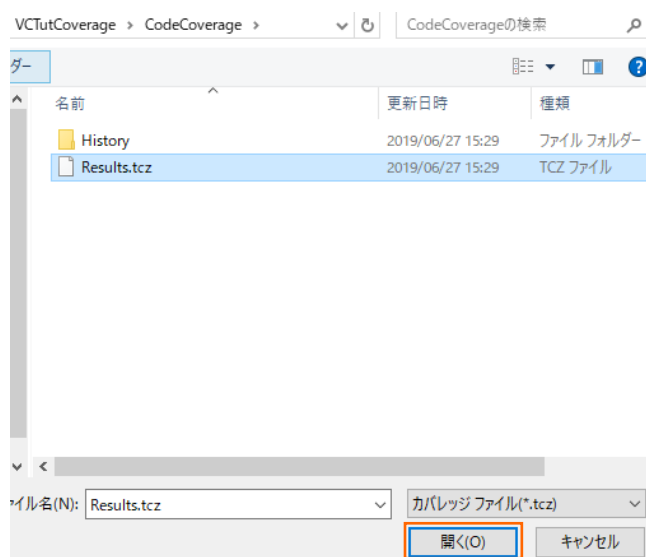
- 1) Visual Studio IDE メニューより、[表示(V)] > [Micro Focus Code Coverage] を選択して、コードカバレッジビューを開きます。



- 2) コードカバレッジビュー上の [カバレッジデータをインポートする] アイコンをクリックします。



- 3) ファイルダイアログ上で、インポートするカバレッジファイルを選択し、[開く(O)] ボタンをクリックします。



カバレッジ情報が更新されることを確認します。

Micro Focus Code Coverage (過去の結果)				
Results_2019-06-27_15_29_48.tcz				
階層	カバー済み (% ブロック)	カバー済み (ブロック)	未カバー (% ブロック)	未カバー (ブロック)
節: main	75 %	6	25 %	2
節: lookup-one-airport	100 %	3	0 %	0
節: distance-between-airports	0 %	0	100 %	8
節: calculate-airport-distance	0 %	0	100 %	5

3.2. コマンドラインからの実行

コードカバレッジ機能は、Visual Studio 上からだけではなく、コマンドラインから実行できます。Jenkins などの CI ツールと連携することで、他言語開発と同様、プログラム開発 > コンパイル > テスト実行 > カバレッジ結果の確認 > プログラム開発 > ... といった開発サイクルを、COBOL 開発に導入することができます。

本節では、コマンドラインからコードカバレッジ機能を利用する手順を習得します。

3.2.1. コマンドラインからのコードカバレッジ機能の利用

- 1) コマンドライン実行用のフォルダを用意します。

```
C:¥>mkdir VCCommandTutorial
```

- 2) 以下の内容で、カバレッジ設定ファイルを VCCommandTutorial¥testcoverage.ini で作成します。

```
[TESTCOVER]
RESULT data¥VCTutCoverage-????.tcz
ECHOLOG NO
```

補足)

実行毎にカバレッジファイルが data フォルダ配下に VCTutCoverage-001.tcz, VCTutCoverage-002.tcz, ... というように作成されます。テスト形態によっては、複数回の実行結果を 1 ファイルに最初からマージしたい場合があります。これを行うには、カバレッジファイル名に続いて、ACCUMULATE 指令を追加します。

```
RESULT data¥VCTutCoverage.tcz ACCUMULATE
```

- 3) Windows 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」を選択します。
- 4) さきほど用意した作業フォルダに移動します。

```
C:¥Users¥Public¥Documents>cd ¥
C:¥>cd VCCommandTutorial
C:¥VCCommandTutorial>
```

- 5) 以下のコマンドを実行します。

- set VS_SOLUTION_PATH=c:¥vs_solution_path
- cobol %VS_SOLUTION_PATH%¥VCTutCoverage¥aircode.cbl gnt,.¥ ANIM TESTCOVER COBIDY(.¥);
- cobol %VS_SOLUTION_PATH%¥VCTutCoverage¥main.cbl gnt,.¥ ANIM TESTCOVER COBIDY(.¥);

注意)

VS_SOLUTION_PATH は、各環境に合わせて修正してください。

```
C:\VCCCommandTutorial>set VS_SOLUTION_PATH=c:\vs_solution_path
C:\VCCCommandTutorial>cobol %VS_SOLUTION_PATH%\VCTutCoverage\aircode.cbl
gnt,.¥ ANIM TESTCOVER COBIDY(.¥);
Micro Focus COBOL
Version 5.0 Copyright (C) Micro Focus 1984-2019. All rights reserved.
* チェック終了 : エラーはありません- コード生成を開始します
* Generating c:\workspace_tut_coverage\VCTutCoverage\aircode
* Data:          1456      Code:          6920      Literals:          384
C:\VCCCommandTutorial>cobol %VS_SOLUTION_PATH%\VCTutCoverage\main.cbl
gnt,.¥ ANIM TESTCOVER COBIDY(.¥);
Micro Focus COBOL
Version 5.0 Copyright (C) Micro Focus 1984-2019. All rights reserved.
* チェック終了 : エラーはありません- コード生成を開始します
* Generating c:\workspace_tut_coverage\VCTutCoverage\main
* Data:           208      Code:          1713      Literals:          192
C:\VCCCommandTutorial>dir
ドライブ C のボリューム ラベルがありません。
ボリューム シリアル番号は 969A-9F98 です

C:\VCCCommandTutorial のディレクトリ

2022/06/27  15:46    <DIR>          .
2022/06/27  15:46    <DIR>          ..
2022/06/27  15:45             10,673 aircode.gnt
2022/06/27  15:45             59,904 aircode.idy
2022/06/27  15:45              3,571 main.gnt
2022/06/27  15:45             46,080 main.idy
2022/06/27  15:44              58 testcoverage.ini
                5 個のファイル             120,286 バイト
                2 個のディレクトリ 37,005,160,448 バイトの空き領域
```

6) アプリケーションを以下のコマンドで実行します。

- set TESTCOVER=.¥testcoverage.ini
- set dd_airports=%VS_SOLUTION_PATH%\VCTutCoverage\airports.dat
- runw main

前回同様の入力を行ってください。

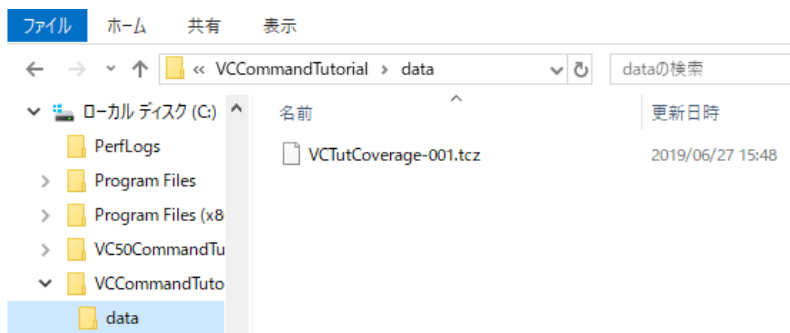
1 回目：“HND△LHR” を入力後 Enter

2 回目：何も入力せず Enter

補足：△は半角スペースを表します。

```
C:\VCCCommandTutorial>set TESTCOVER=.%testcoverage.ini
C:\VCCCommandTutorial>set dd_airports=%VS_SOLUTION_PATH%\VCTutCoverage\airports.dat
C:\VCCCommandTutorial>runw main
C:\VCCCommandTutorial>runw.exe - COBOL テキストウィンドウ
Enter an airport code, or two codes separated by space, or no code to exit:
```

testcoverage.ini の RESULT 項目で指定した data フォルダ配下にカバレッジ情報ファイルが作成されていることを確認してください。



3.2.2. ユーティリティ機能の紹介

カバレッジ情報ファイルの HTML 形式でのレポート出力

1) 以下のコマンドを実行し、HTML レポート形式で出力を行います。

- tcreport data¥VCTutCoverage-001.tcz browse html reportloc(out)

Visual COBOL コマンドプロンプト (64-bit)

```
C:\¥VCCCommandTutorial>tcreport data¥VCTutCoverage-001.tcz browse html reportloc(out)
プログラム aircode.gnt 正常に処理されました
プログラム main.gnt 正常に処理されました
生成されたレポートのリストは以下にあります:
C:\¥VCCCommandTutorial¥out¥TCIndex.htm
```

IDE と同様に通過した行は緑、未通過は赤色で表示されます。

行番号		回数
70	set open-file to true	
71	perform call-aircode-program	1
72		
73	perform until exit	1
74	display "Enter an airport code, or two codes " &	
75	"separated by space, or no code to exit:"	
76	accept user-input	
77	unstring user-input delimited by space	
78	into ls-airport1, ls-airport2	2
79		
80	if ls-airport1 = spaces	
81	exit perform	
82	end-if	1
83		
84	if ls-airport2 not = spaces	1
85	set get-distance to true	
86	perform call-aircode-program	1
87	display "Distance: " distance-km "kms"	
88	distance-miles "miles"	1
89	else	
90	move " " to ap-code of ls-rec	
91	set get-details to true	
92	perform call-aircode-program	-
93		
94	if ap-code OF ls-rec <> " "	
95	then	-
96	set display-record to true	
97	perform call-aircode-program	
98	end-if	
99	end-if	-
100	end-perform	

カバレッジ率の到達・未達判定

1) 以下のコマンドを実行します。

しきい値に達している場合)

- tcutil -t 60 data¥VCTutCoverage-001.tcz

```
C:\¥VCCCommandTutorial>tcutil -t 60 data¥VCTutCoverage-001.tcz
```

```
C:\¥VCCCommandTutorial>
```

しきい値に達していない場合)

- tcutil -t 70 data¥VCTutCoverage-001.tcz

```
C:¥VCCCommandTutorial>tcutil -t 70 data¥VCTutCoverage-001.tcz
```

カバレッジのしきい値に達しませんでした

```
C:¥VCCCommandTutorial>
```

補足)

しきい値の達成有無を判定するには、tcutil コマンドの終了コードを確認します。%ERRORLEVEL% に終了コードが設定されているため、この値が 0 の場合に達している、0 以外の場合に達していないと判断できます。

```
C:¥VCCCommandTutorial>tcutil -t 60 data¥VCTutCoverage-001.tcz
```

```
C:¥VCCCommandTutorial>echo %ERRORLEVEL%
```

```
0
```

```
C:¥VCCCommandTutorial>tcutil -t 70 data¥VCTutCoverage-001.tcz
```

カバレッジのしきい値に達しませんでした

```
C:¥VCCCommandTutorial>echo %ERRORLEVEL%
```

```
-1
```

WHAT'S NEXT

- 本チュートリアルで学習した技術の詳細については製品マニュアルをご参照ください。

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、著作権法に基づき、適切な扱いを行ってください。