

Visual COBOL チュートリアル

Eclipse – 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、JVM 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
- Visual COBOL 11.0 Patch Update 01 for Eclipse がインストール済みであること

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

- 1 目的
- 2 前提
- 3 チュートリアル手順
 - 3.1 IDE からの実行
 - 3.1.1 Eclipse の起動
 - 3.1.2 チュートリアルプロジェクトのインポート
 - 3.1.3 カバレッジを有効にしたアプリケーションの実行
 - 3.1.4 複数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
 - 3.1.6 カバレッジ情報のエクスポート
 - 3.2 コマンドラインからの実行
 - 3.3 ユーティリティ機能の紹介
 - 3.3.1 カバレッジ情報ファイルの HTML 形式でのレポート出力
 - 3.3.2 カバレッジ率の評価

3 チュートリアル手順

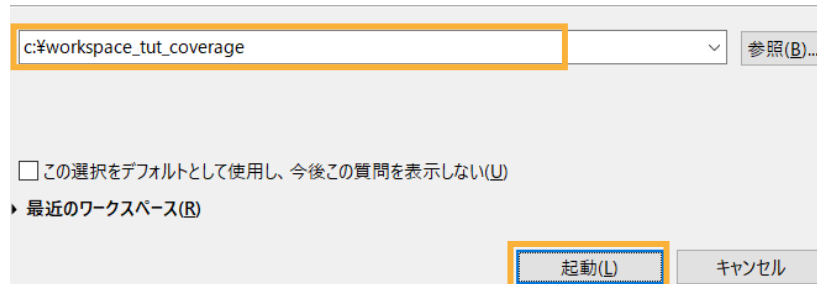
3.1 IDE からの実行

3.1.1 Eclipse の起動

- 1) スタートメニューより、Visual COBOL for Eclipse を起動します。
- 2) ワークスペースを指定し、[起動(L)] ボタンをクリックします。

ディレクトリーをワークスペースとして選択

Eclipse は、ワークスペースディレクトリを使用して、環境設定と開発成果物を保存します。



起動後、ようこそ画面は閉じてください。

3.1.2 チュートリアルプロジェクトのインポート

- 1) Eclipse IDE メニューの [ファイル(F)] > [インポート(I)] を選択します。



- 2) [一般] > [既存プロジェクトをワークスペースへ] を選択し、[次へ(N) >] ボタンをクリックします。

選択

アーカイブ・ファイルまたはディレクトリーから新規プロジェクトを作成します。



インポート・ウィザードの選択(S) :

フィルタ入力

- ▼ 一般
 - アーカイブ・ファイル
 - ファイル・システム
 - フォルダーまたはアーカイブ由来のプロジェクト
 - 既存プロジェクトをワークスペースへ**
 - 設定
- > EJB
- > Git
- > Gradle

3) 以下の設定を行い、[終了(F)] をクリックします。

- ルート・ディレクトリーの選択(T)
[参照(R)] をクリックして、ダウンロードしたサンプルファイルを展開したフォルダーを指定
- プロジェクトをワークスペースにコピー
チェックをつける

プロジェクトをインポート

既存の Eclipse プロジェクトを検索するディレクトリーを選択します。



☒ ルート・ディレクトリーの選択(T): C:\vc-coverage

☐ アーカイブ・ファイルの選択(A):

プロジェクト(P):

☒ AirportDemoTutCoverage(C:\vc-coverage)

オプション

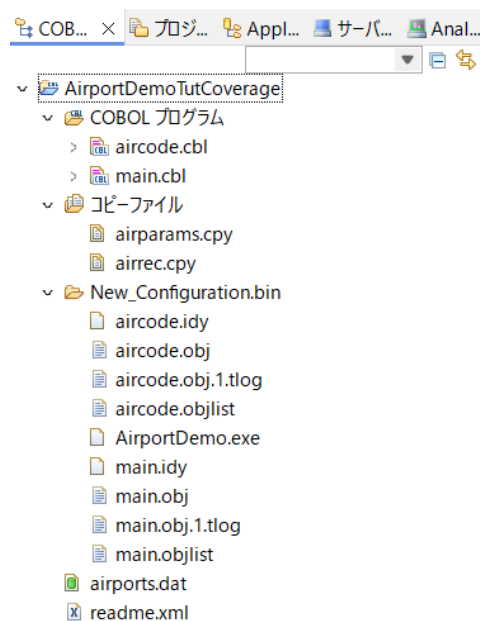
☐ ネストしたプロジェクトを検索(H)
☒ **プロジェクトをワークスペースにコピー(C)**
☐ 完了次第、新しくインポートしたプロジェクトを閉じる(o)
☐ ワークスペースに既に存在するプロジェクトを隠す(i)

ワーキング・セット

☐ ワーキング・セットにプロジェクトを追加(I)

ワーキング・セット(O):

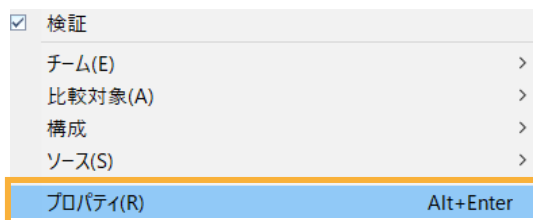
プロジェクトがインポートされます。



補足)

COBOL 開発を行うためには、COBOL パースペクティブという画面レイアウトを使用します。異なるパースペクティブを開いている場合、Eclipse IDE メニューの [ウィンドウ(W)] > [パースペクティブ(R)] > [パースペクティブを開く(O)] > [その他(O)] をクリックした上で COBOL をクリックすることで、COBOL パースペクティブを開くことができます。

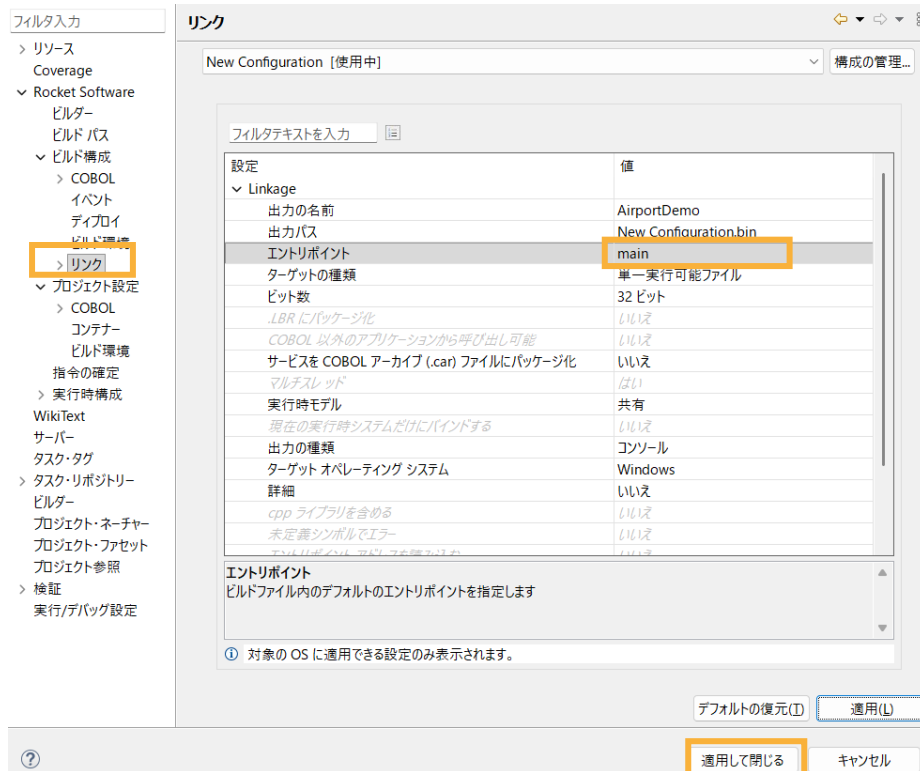
- 4) COBOL エクスプローラービュー上で AirportDemoTutCoverage プロジェクトを選択した状態で、マウスの右クリックにてコンテキストメニューを表示し、[プロパティ(R)] をクリックします。



- 5) 画面左部のメニューより、[Rocket Software] > [プロジェクト設定] > [COBOL] を選択します。その後、[コードカバレッジを有効にする] を “true” に変更したうえで、[適用(L)] をクリックします。

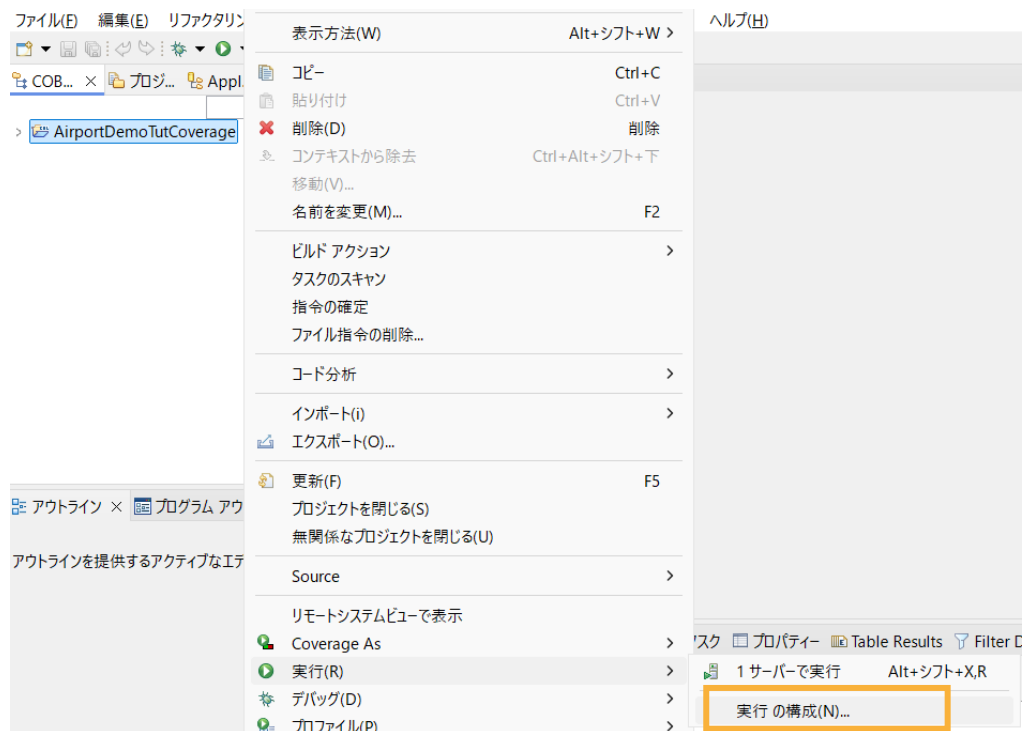


- 6) 画面左部のメニューより、[Rocket Software] > [ビルド構成] > [リンク] を選択します。エントリポイントに "main" と入力し、[適用して閉じる] ボタンをクリックします。

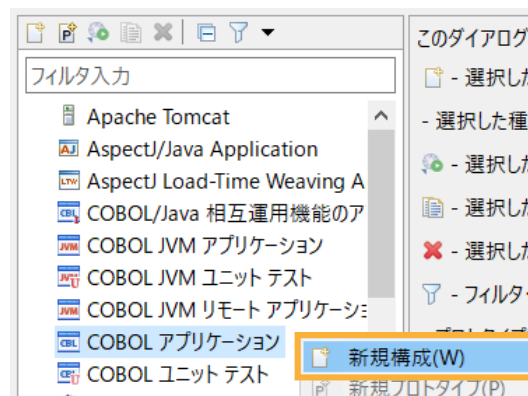


3.1.3 カバレッジを有効にしたアプリケーションの実行

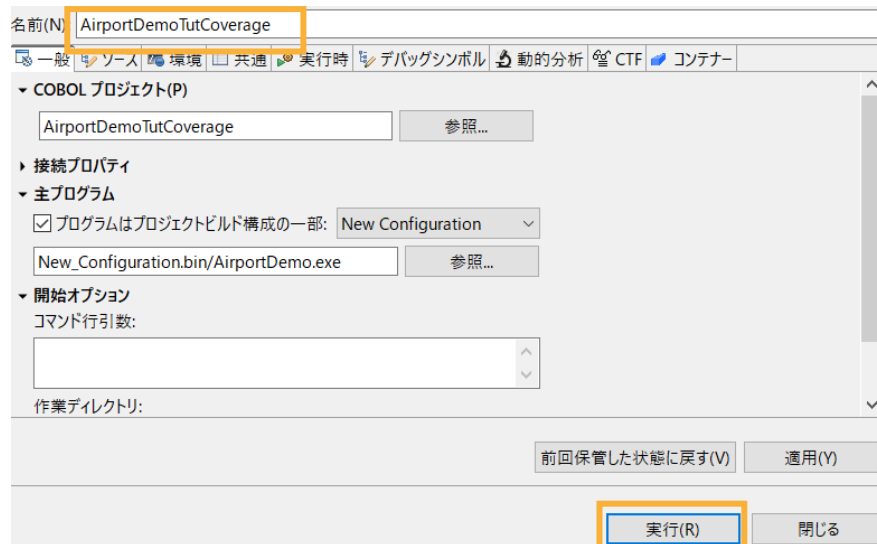
- 1) AirportDemoTutCoverage プロジェクト内の main.cbl を選択した上で、マウスの右クリックにてコンテキストメニューを表示し、[実行(R)] > [実行の構成(N)] を選択します。



- 2) 「COBOL アプリケーション」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、[新規構成(W)] を選択します。

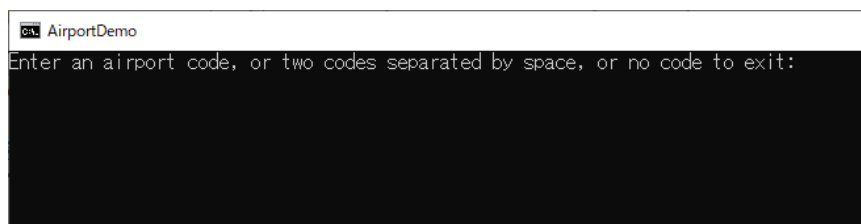


- 3) [名前(N)] に "AirportDemoTutCoverage" を入力した上で、[実行(R)] をクリックします。

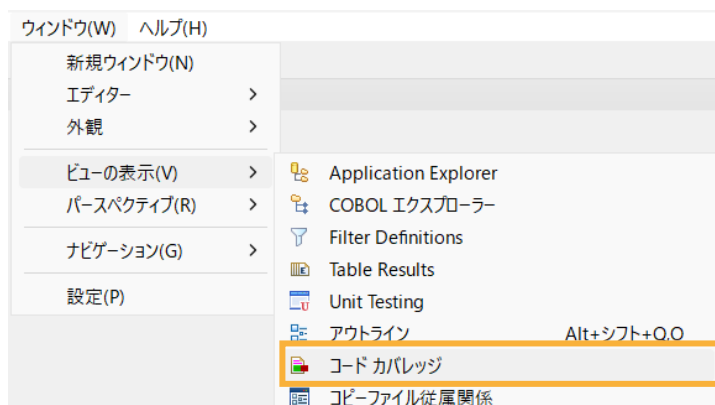


4) アプリケーション画面上に以下の入力を行います。

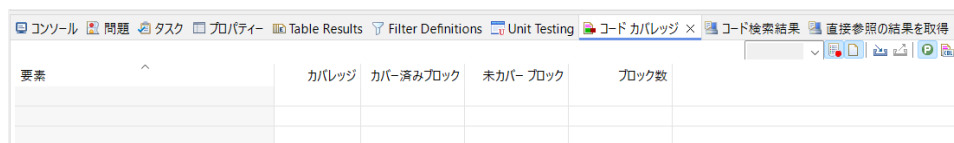
- 1 回目： “HND△LHR” を入力して Enter（羽田・ロンドンヒースロー 空港間の距離を表示）
- 2 回目： 何も入力せず Enter（プログラムの終了）



5) Eclipse IDE メニューより、[ウィンドウ(W)] > [ビューの表示(V)] > [コードカバレッジ] 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ません。



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) Eclipse IDE メニューより、[実行(R)] > [実行構成(N)] を選択します。



- 7) AirportDemoTutCoverage の実行構成が選択されている状態で、[動的分析] タブを選択し、[コードカバレッジを有効にする] をチェックします。その後、[実行(R)] をクリックします。



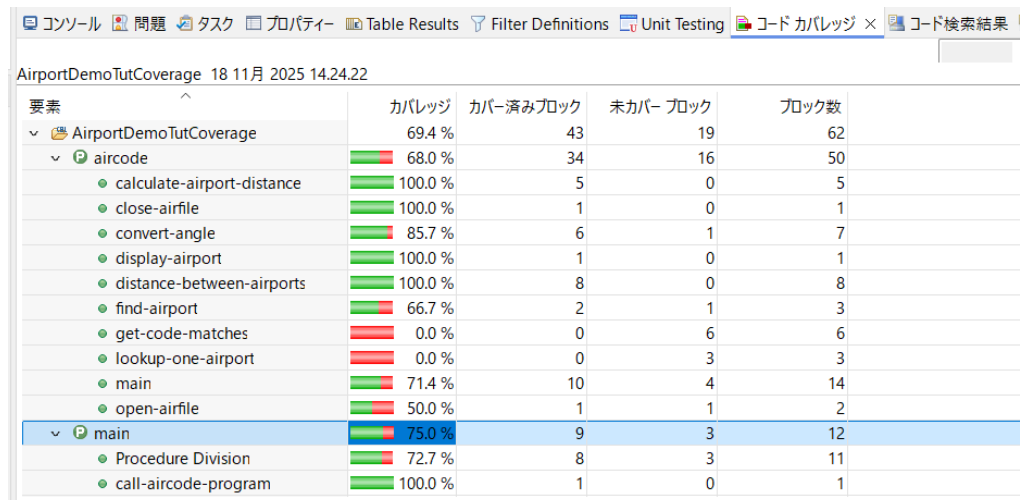
- 8) アプリケーション画面が起動されるため、前回同様、下記入力を行ってください。

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)

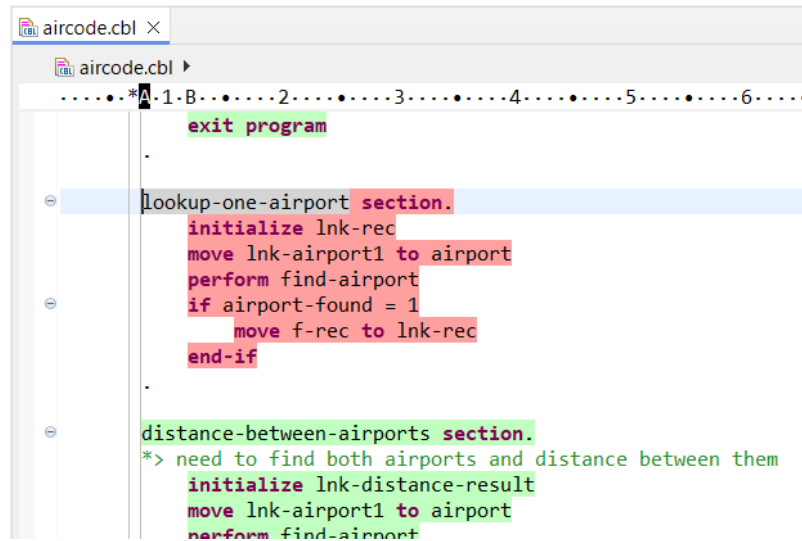
補足) △は半角スペースを表します。

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。



要素	カバレッジ	カバー済みブロック	未カバー ブロック	ブロック数
▼ AirportDemoTutCoverage	69.4 %	43	19	62
▼ aircode	68.0 %	34	16	50
calculate-airport-distance	100.0 %	5	0	5
close-airfile	100.0 %	1	0	1
convert-angle	85.7 %	6	1	7
display-airport	100.0 %	1	0	1
distance-between-airports	100.0 %	8	0	8
find-airport	66.7 %	2	1	3
get-code-matches	0.0 %	0	6	6
lookup-one-airport	0.0 %	0	3	3
main	71.4 %	10	4	14
open-airfile	50.0 %	1	1	2
▼ main	75.0 %	9	3	12
Procedure Division	72.7 %	8	3	11
call-aircode-program	100.0 %	1	0	1

- 9) 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。



```

aircode.cbl
...1.B...2...3...4...5...6...
exit program
.
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

```

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。

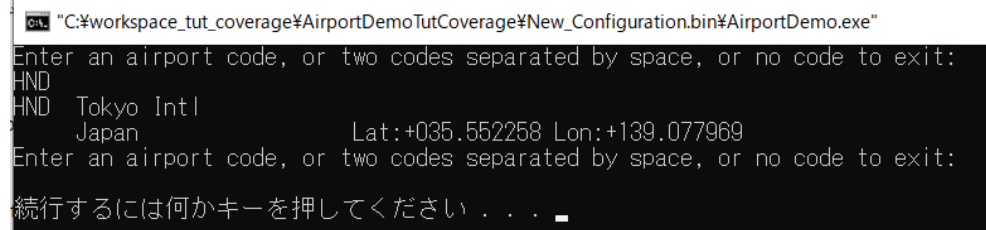
3.1.4 複数カバレッジ情報のマージ

一般的に、1回のテスト実行で品質指標として十分なカバレッジ情報を得ることはできません。本節では、複数回のアプリケーション実行によるカバレッジ情報をマージする方法を学びます。

- 1) 再度、AirportDemoTutCoverage アプリケーションを実行し、以下の入力を行ってください。

1 回目: “HND” と入力した後で Enter (羽田空港の情報を表示)

2 回目: 何も入力せず Enter (プログラムの終了)



```

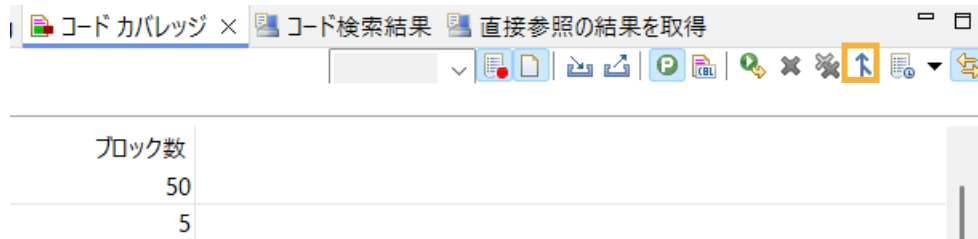
"C:\workspace_tut_coverage\AirportDemoTutCoverage\New_Configuration.bin\AirportDemo.exe"
Enter an airport code, or two codes separated by space, or no code to exit:
HND
HND Tokyo Intl
Japan Lat:+035.552258 Lon:+139.077969
Enter an airport code, or two codes separated by space, or no code to exit:
続行するには何かキーを押してください . . .

```

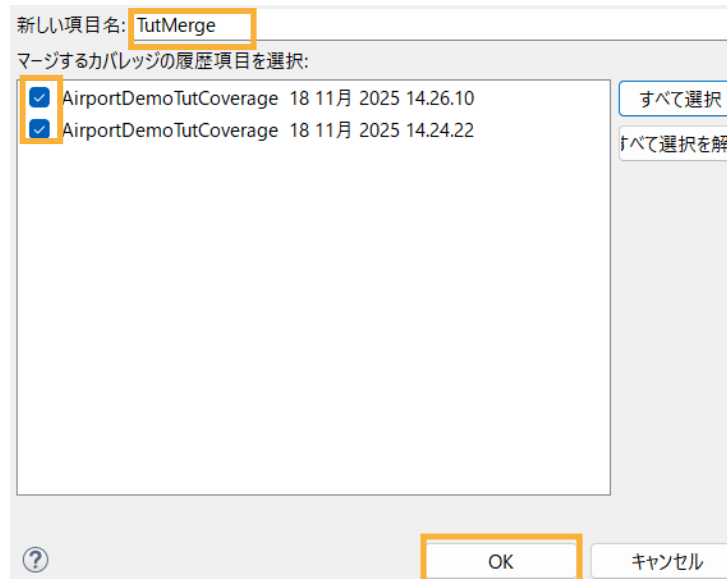
コードカバレッジビューより、さきほどの実行結果と異なっていることを確認します。例えば、lookup-one-airport や distance-between-airports 節のカバレッジ率が真逆になっていることが確認できます。

AirportDemoTutCoverage 18 11月 2025 14.26.10					
要素	カバレッジ	カバー済みブロック	未カバー ブロック	ブロック数	
aircode	40.0 %	20	30	50	
calculate-airport-distance	0.0 %	0	5	5	
close-airfile	100.0 %	1	0	1	
convert-angle	0.0 %	0	7	7	
display-airport	100.0 %	1	0	1	
distance-between-airports	0.0 %	0	8	8	
find-airport	66.7 %	2	1	3	
get-code-matches	0.0 %	0	6	6	
lookup-one-airport	100.0 %	3	0	3	
main	85.7 %	12	2	14	
open-airfile	50.0 %	1	1	2	

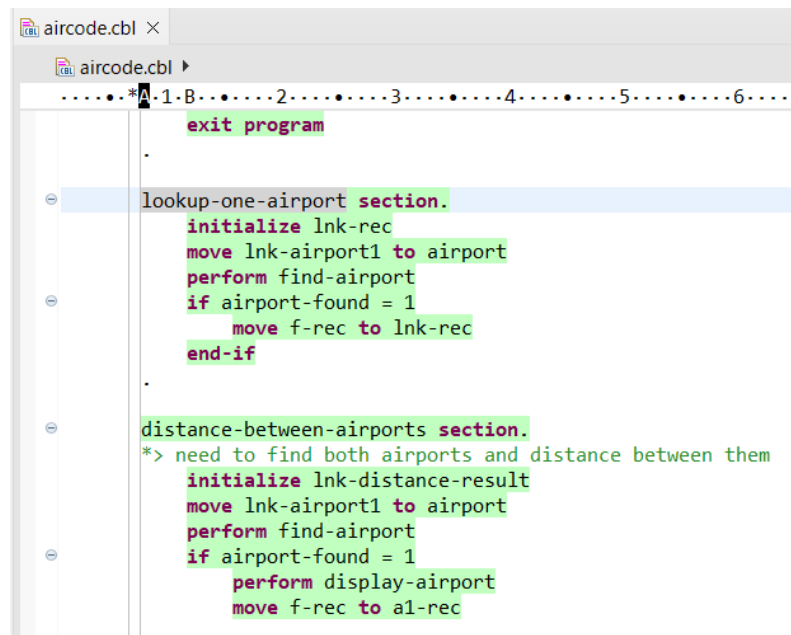
- 2) コードカバレッジビュー上の右上にある「履歴項目のマージ」アイコンをクリックします。



- 3) 新しい項目名に “TutMerge” を入力し、マージ対象となる履歴項目にチェックした上で、[OK] ボタンをクリックします。



カバレッジ情報がマージされ、情報が更新されていることを確認してください。



TutMerge				
要素	カバレッジ	カバー済みブロック	未カバー ブロック	ブロック数
▼ AirportDemoTutCoverage	83.9 %	52	10	62
▼ aircode	80.0 %	40	10	50
● calculate-airport-distance	100.0 %	5	0	5
● close-airfile	100.0 %	1	0	1
● convert-angle	85.7 %	6	1	7
● display-airport	100.0 %	1	0	1
● distance-between-airports	100.0 %	8	0	8
● find-airport	66.7 %	2	1	3
● get-code-matches	0.0 %	0	6	6
● lookup-one-airport	100.0 %	3	0	3
● main	92.9 %	13	1	14
● open-airfile	50.0 %	1	1	2
▼ main	100.0 %	12	0	12
● Procedure Division	100.0 %	11	0	11
● call-aircode-program	100.0 %	1	0	1

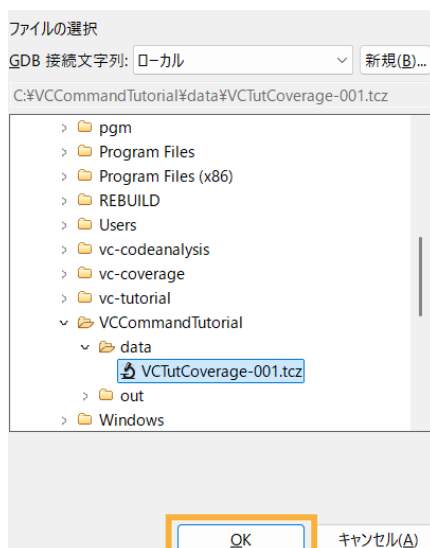
3.1.5 既存カバレッジ情報ファイルの読み込み

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、読み込み対象となる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にも対応しています。本手順では、後述する 3.2 のカバレッジ結果を利用しますので、先に 3.2 を実施してください。

- 1) コードカバレッジビュー上の右上より [セッションをインポート] アイコンをクリックします。

Unit Testing		コード カバレッジ	コード検索結果	直接参照の結果を取得
未カバー ブロック	ブロック数			
10	62			
10	50			
0	5			

- 2) 表示したいカバレッジ情報ファイルを選択し、[OK] をクリックします。



コードカバレッジビュー情報が対象のカバレッジ情報ファイルのものに更新されます。

/CTutCoverage-001.tcz 18 11月 2025 14:35:10				
要素	カバレッジ	カバー済みブロック	未カバー ブロック	ブロック数
> aircode	68.0 %	34	16	50
> main	75.0 %	9	3	12

3.1.6 カバレッジ情報のエクスポート

カバレッジ情報はエクスポートすることでファイルに保存しておくことができます。

- 1) カバレッジビューの右上より [エクスポート] アイコンをクリックすると保存先が選択できます。

Unit Testing		コード カバレッジ ×	コード検索結果	直接参照の結果を取得
未カバー ブロック		ブロック数		
16		50		
3		12		

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、Eclipse 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 > ... という開発サイクルを、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。

- 1) コマンドライン実行用のフォルダーを用意します。

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

- 2) 以下の内容で、カバレッジ設定ファイルを VCCCommandTutorial¥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、[Rocket Visual COBOL] > [Visual COBOL コマンドプロンプト (32-bit)] を選択します。
- 4) さきほど用意した作業フォルダーに移動します。

```
C:¥Users¥tarot¥Documents>cd ¥VCCCommandTutorial
C:¥VCCCommandTutorial>
```

5) 以下のコマンドを実行します。

- set ECLIPSE_WORKSPACE=c:¥workspace_tut_coverage
- cobol %ECLIPSE_WORKSPACE%¥AirportDemoTutCoverage¥aircode.cbl gnt,¥ ANIM TESTCOVER COBIDY(.¥);
- cobol %ECLIPSE_WORKSPACE%¥AirportDemoTutCoverage¥main.cbl gnt,¥ ANIM TESTCOVER COBIDY(.¥);

注意)

ECLIPSE_WORKSPACE は各環境に合わせて修正してください。

```
C:¥VCCCommandTutorial>set ECLIPSE_WORKSPACE=c:¥workspace_tut_coverage
C:¥VCCCommandTutorial>cobol %ECLIPSE_WORKSPACE%¥AirportDemoTutCoverage¥aircode.c
bl gnt,¥ ANIM TESTCOVER COBIDY(.¥);
Rocket (R) COBOL
Version 11.0 (C) 1984-2025 Rocket Software, Inc. or its affiliates.
* 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¥AirportDemoTutCoverage¥aircode
* Data:          1824      Code:          5408      Literals:          811
C:¥VCCCommandTutorial>cobol %ECLIPSE_WORKSPACE%¥AirportDemoTutCoverage¥main.cbl
gnt,¥ ANIM TESTCOVER COBIDY(.¥);
Rocket (R) COBOL
Version 11.0 (C) 1984-2025 Rocket Software, Inc. or its affiliates.
* 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¥AirportDemoTutCoverage¥main
* Data:           576      Code:          1344      Literals:           328
C:¥VCCCommandTutorial>
```

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=%ECLIPSE_WORKSPACE%¥AirportDemoTutCoverage¥airports.dat

runw main.gnt

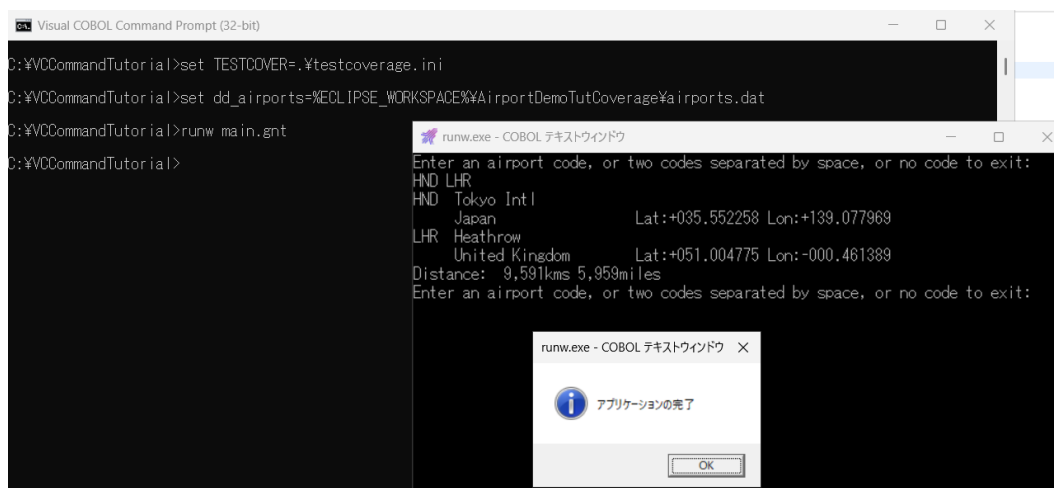
以下の入力を行ってください。

1 回目 : "HND△LHR" を入力後 Enter

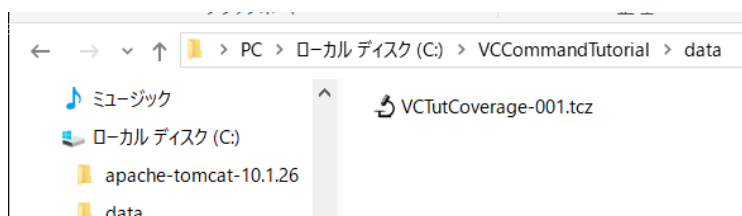
2 回目 : 何も入力せず Enter

補足)

△は半角スペースを表します。



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。



3.3 ユーティリティ機能の紹介

3.3.1 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)

```
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
C:¥VCCCommandTutorial>
```

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。

283		4
284	compute out-angle = fa-degs + (fa-mins / 60)	
285	if fa-sign = "-"	
286	multiply -1 by out-angle	1
287	end-if	
288	compute out-angle = (out-angle * function pi) / 180	
289	.	4
290		
291	display-airport section.	
292	display f-code " " f-name	
293	display " " f-country	
294	" Lat:" f-lat-sign f-lat-degs "." f-lat-mins	2
295	" Lon:" f-long-sign f-long-degs "." f-long-mins	
296	.	
297		
298	get-code-matches section.	
299	move 0 to idx	
300	initialize aircode-array	
301	move 0 to prefix-length	
302	inspect lnk-prefix-text tallying prefix-length	
303	for characters before space	-
304	move lnk-prefix-text to f-code	
305	start airfile key >= f-code	
306	not invalid key	

補足 1)

コマンドラインからの実行により作成されたカバレッジ情報を IDE 上に表示する方法は、3.1.5 を参照ください。

補足 2)

IDE 同様、複数カバレッジ情報ファイルのマージした結果レポートを出力することができ、以下の 2 通りの方法があります。

リストファイルを用いてマージ対象ファイルを指定する

C:\¥VCCCommandTutorial>type resultfiles.txt

data¥VCTutCoverage-001.tcz

data¥VCTutCoverage-002.tcz

C:\¥VCCCommandTutorial>tcreport @resultfiles.txt 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

tcreport コマンド実行時引数に半角カンマで対象ファイルを指定する

C:\¥VCCCommandTutorial>tcreport data¥VCTutCoverage-001.tcz,data¥VCTutCoverage-002.tcz 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

3.3.2 カバレッジ率の評価

- 1) 以下のコマンドを実行します。

tcutil -t <しきい値> カバレッジファイルへのパス

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
```

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。