# Micro Focus Enterprise Developer チュートリアル

## メインフレーム COBOL 開発：コードカバレッジの実施

## Eclipse 編

### 1. 目的

本チュートリアルは、JCL から実行された COBOL ソースを例としたコードカバレッジの取得や結果表示を行う手順とその方法の習得を目的としています。

### 2. 前提

- 本チュートリアルで使用したマシン OS ： Windows 11 Pro
- 使用マシンに Micro Focus Enterprise Developer 8.0 for Eclipse がインストールされていること
- JCL チュートリアルのプログラムと Enterprise Server インスタンスを使用するため、JCL チュートリアルが実施済であること

  補足）完了していない場合は、先に JCL チュートリアルを実施してください。

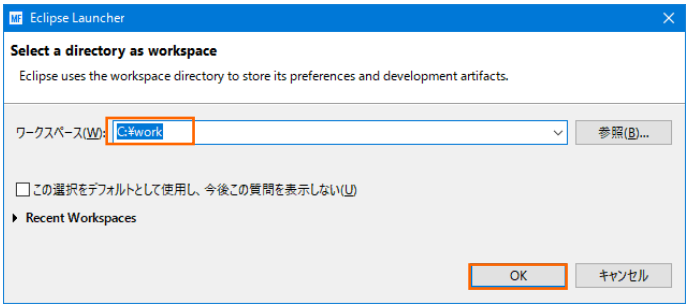
### 3. チュートリアル手順の概要

1. Eclipse の起動
2. プロジェクトプロパティの設定
3. IDE を使用した JCL 実行
4. コードカバレッジの表示
5. IDE を利用しない JCL 実行

## 3.1 Eclipse の起動

1) Micro Focus Enterprise Developer for Eclipse を起動します。


2) JCL チュートリアルで作成した C:¥work をワークスペースへ指定して、[OK] ボタンをクリックします。

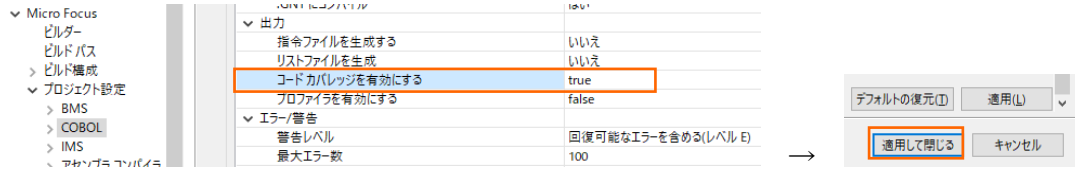



## 3.2 プロジェクトプロパティの設定

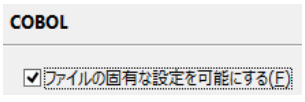
プロジェクトのプロパティを設定します。

1) COBOL エクスプローラー内の JCLDEMO プロジェクトを右クリックして [プロパティ] を選択します。

2) 左側メニューの [Micro Focus] > [プロジェクト設定] > [COBOL] を選択して、[コード カバレッジを有効にする] に ture を指定後 [適用して閉じる] ボタンをクリックしてください。保存後は再度プロジェクトのビルドを実施してください。




補足）個別ソースファイルへ指定したい場合は対象ソースファイルを右クリックし、左側メニューの [COBOL] から表示される画面で [ファイルの固有な設定を可能にする] チェックをオンにした後、コードカバレッジを有効にします。指定後は [適用して閉じる] ボタンをクリックしてください。

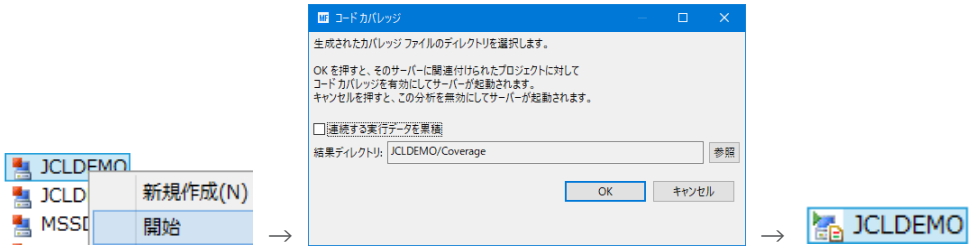


## 3.3 IDE を使用した JCL 実行

1) JCLDEMO プロジェクトに関連付けられている、JCL チュートリアルで作成した JCLDEMO インスタンスへコードカバレッジ分析を指定します。

Eclipse の [サーバーエクスプローラー] > [JCLDEMO] を右クリックし、[分析を設定] > [コード カバレッジ] を選択して有効にします。


2) 再度 [サーバーエクスプローラー] > [JCLDEMO] を右クリックし [開始] を選択するとログファイルと結果ファイルの出力パスを指定するウィンドウが表示されますので、このまま [OK] ボタンをクリックします。



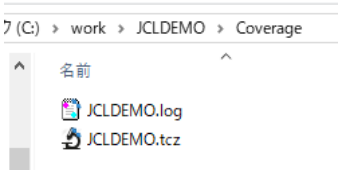

[コンソール] タブに以下のように表示されます。

```
サーバー: JCLDEMO 正常に起動されました - コード カバレッジ有効
```

3) COBOL エクスプローラー内の vsamwrt2.jcl を右クリックして [Enterprise Server へのサブミット] を選択し、この JCL を実行します。

4) [サーバーエクスプローラー] > [JCLDEMO] を右クリックし [スプール表示] を選択して JCL の実行結果を確認します。

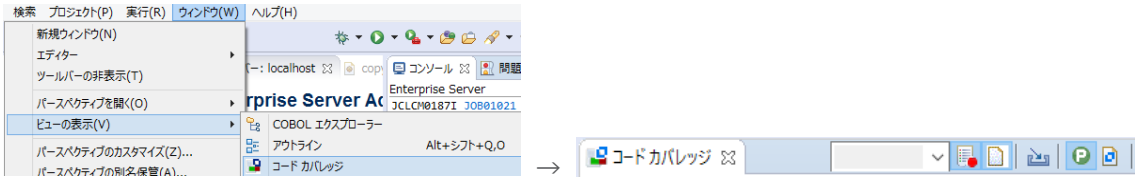

5) Windows エクスプローラーから、前項の出力パスにコードカバレッジ用のフォルダとファイルが作成されていることを確認します。



## 3.4 コードカバレッジの表示

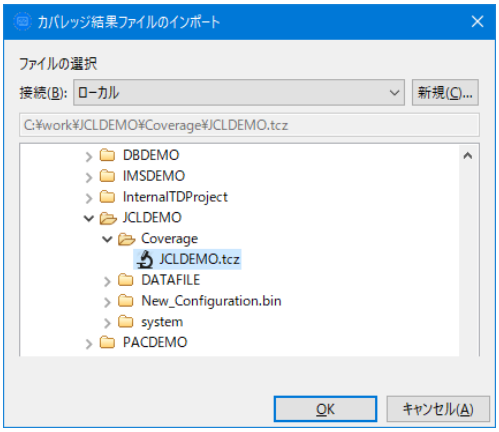
Eclipse に戻り、下記の操作を行います。

1) [ウィンドウ] プルダウンメニューから [ビューの表示] > [コード カバレッジ] を選択してコードカバレッジビューを表示します。

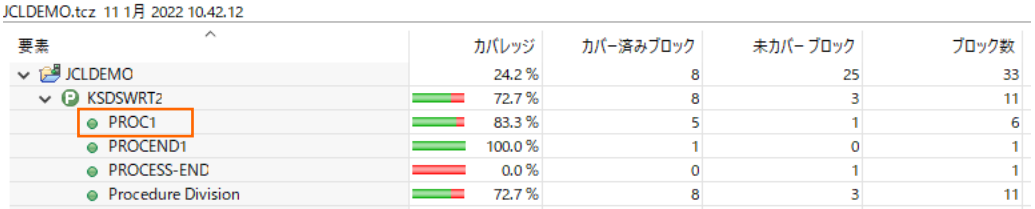



2) [コードカバレッジ] タブの [セッションをインポート] アイコンをクリックしてファイル選択ウィンドウを表示します。


3) [ドライブ] 配下から前項で生成した “JCLDEMO.tcz” ファイルを選択し、[OK] ボタンをクリックします。




4) 実行されたコードのカバレッジ率が表示されます。 [PROC1] をダブルクリックしてみます。




| 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 |
|---|---|---|---|---|
| JCLDEMO | 24.2 % | 8 | 25 | 33 |
| KSDSWRT2 | 72.7 % | 8 | 3 | 11 |
| PROC1 | 83.3 % | 5 | 1 | 6 |
| PROCEND1 | 100.0 % | 1 | 0 | 1 |
| PROCESS-END | 0.0 % | 0 | 1 | 1 |
| Procedure Division | 72.7 % | 8 | 3 | 11 |

5) 該当箇所が表示され、カバーされた箇所は緑に表示されます。

```
PROCEDURE DIVISION.
    OPEN INPUT INDATA.
    OPEN OUTPUT INDEXFILE.
    OPEN OUTPUT PRTFILE.
    IF "00" NOT = FSTAT-I OR FSTAT-K OR FSTAT-P
      THEN
        CONTINUE;
      ELSE
        PERFORM PROC1 THRU PROCEND1;
    END-IF.
PROCESS-END.
    DISPLAY "** OPEN ERROR **".
    STOP RUN.
PROC1.
    PERFORM UNTIL LOOP1
      READ INDATA
        AT END
          SET LOOP1 TO TRUE
        NOT AT END
          MOVE INREC TO KREC
          MOVE INREC TO PREC
          WRITE PREC
          WRITE KREC
            INVALID KEY DISPLAY "INVALID"
            NOT INVALID KEY CONTINUE
          END-WRITE
      END-READ
    END-PERFORM.
```

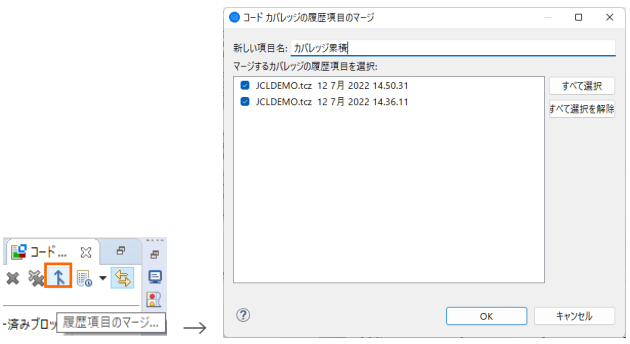
6) カバレッジが 0% の [PROCESS-END] を見ると赤で表示されており、今回の実行ではカバーされていないことがわかります。

```
PROCESS-END.
    DISPLAY "** OPEN ERROR **".
    STOP RUN.
PROC1.
```

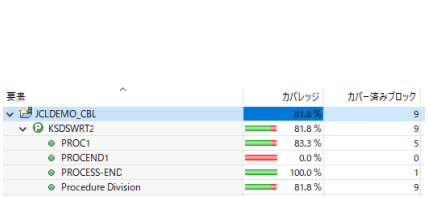
7) [PROCESS-END] を通すため、INDATA ファイルをオープン時にエラーとなるよう意図的に操作して JCL を再実行します。再度カバレッジの結果ファイルを取り込むと、正常時と逆の結果になりました。

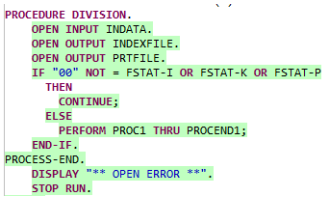
```
PROCEDURE DIVISION.
    OPEN INPUT INDATA.
    OPEN OUTPUT INDEXFILE.
    OPEN OUTPUT PRTFILE.
    IF "00" NOT = FSTAT-I OR FSTAT-K OR FSTAT-P
      THEN
        CONTINUE;
      ELSE
        PERFORM PROC1 THRU PROCEND1;
    END-IF.
PROCESS-END.
    DISPLAY "** OPEN ERROR **".
    STOP RUN.
PROC1.
    PERFORM UNTIL LOOP1
      READ INDATA
        AT END
          SET LOOP1 TO TRUE
        NOT AT END
          MOVE INREC TO KREC
          MOVE INREC TO PREC
          WRITE PREC
          WRITE KREC
            INVALID KEY DISPLAY "INVALID"
```

8) Eclipse の [コードカバレッジ] タブの [履歴項目のマージ] アイコンをクリックしてマージウィンドウを表示します。項目名に任意の値を入力後、2 回の実行結果にチェックして [OK] ボタンをクリックします。

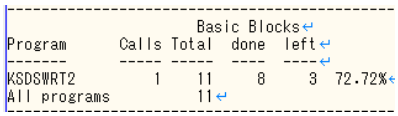


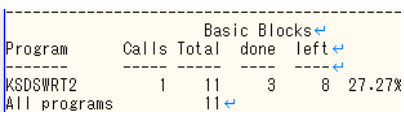
9) [コードカバレッジ] タブにはマージされた結果が表示され、ソースコードも緑になりました。



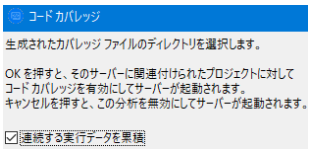



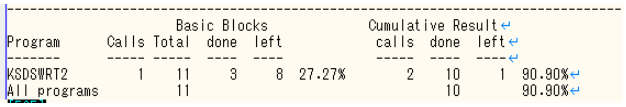

10) カバレッジ出力フォルダにはログが生成されますので、こちらでも結果を確認することができます。



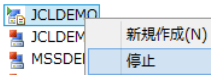




11) Enterprise Server 開始時に [連続する実行データを累積] へチェックし、複数回実行後のカバレッジの結果ファイルを取り込むと、マージされた後と同じように表示されます。




12) [JCLDEMO] インスタンスを停止します。




## 3.5 IDE を利用しない JCL 実行

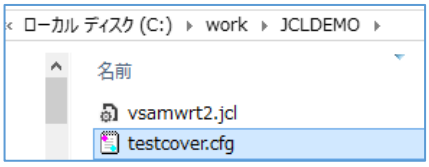
次に、コマンドラインから JCL を実行する際のコードカバレッジを実施します。

1) 結果の出力先を指定するコードカバレッジ用の構成ファイルを作成します。

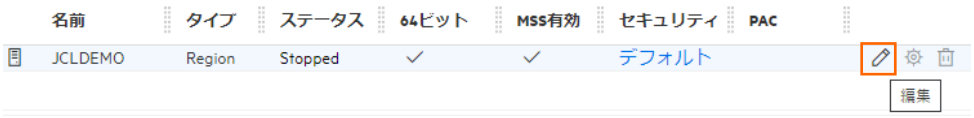
下記内容をテキストエディターへコピーして JCLDEMO フォルダ配下へ保存します。ここではファイル名を testcover.cfg とします。 RESULT 文字に続く jclcover.tcz ファイルが実行により出力される結果ファイルです。

【構成ファイル内容】

[TESTCOVER]
RESULT C:¥work¥JCLDEMO¥jclcover.tcz ACCUMULATE
ECHOLOG NO
LOGNAME C:¥work¥JCLDEMO¥testcover.log

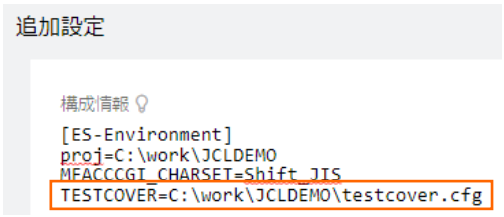


2) ESCWA から [JCLDEMO] インスタンスの [編集] アイコンをクリックして設定画面を表示します。

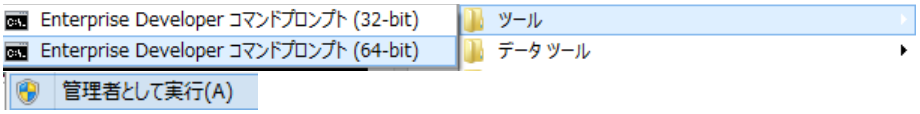



3) [構成情報] へ構成ファイル名までのパスを指定します。

TESTCOVER=C:¥work¥JCLDEMO¥testcover.cfg

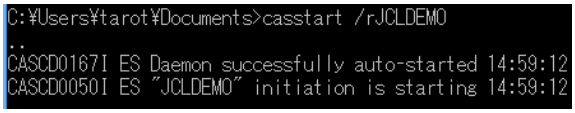



4) プログラムの [Micro Focus Enterprise Developer] > [ツール] > [Enterprise Developer コマンドプロンプト（64-bit）] を右クリックし [管理者として実行] を選択します。




5) [JCLDEMO] インスタンスを開始するコマンドを実行します。

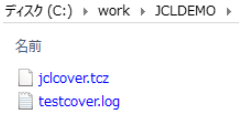
コマンド）casstart /rJCLDEMO



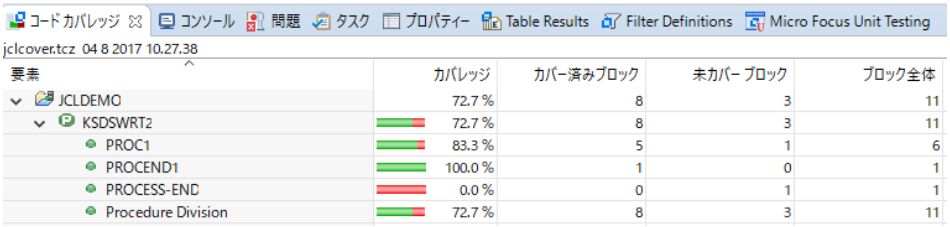

6) C:¥work¥JCLDEMO フォルダへ移動して前項と同じ JCL をコマンドから実行します。

コマンド）cassub /rJCLDEMO /jvsamwrt2.jcl

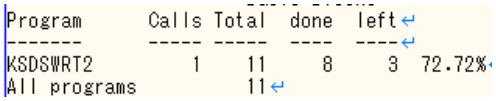

7) エクスプローラーを表示して、指定パスにコードカバレッジ用のフォルダとファイルが作成されていることを確認します。



8) Eclipse の [コードカバレッジ] タブの [セッションをインポート] アイコンをクリックして生成されたファイル内容を確認すると前項と同じ結果であることがわかります。




9) 出力された testcover.log ファイルをテキストエディターで表示すると、ブロック単位のカバレッジ結果を確認できます。




10) [JCLDEMO] インスタンスを停止するコマンドを実行します。

コマンド）casstop /rJCLDEMO


## 4. 免責事項

本チュートリアルの例題ソースコードは機能説明を目的としたサンプルであり、無謬性を保証するものではありません。例題ソースコードは弊社に断りなくご利用いただけますが、本チュートリアルに関わる全てを対象として、二次的著作物に引用する場合は著作権法の精神に基づき適切な扱いを行ってください。

## 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。