
Micro Focus Enterprise Developer チュートリアル

メインフレーム COBOL 開発 : JCL

Eclipse 編

1. 目的

本チュートリアルでは、Eclipse を使用したメインフレーム COBOL プロジェクトの作成、コンパイル、JCL の実行、デバッグまでを行い、その手順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 10 Enterprise
- 使用マシンに Micro Focus Enterprise Developer 6.0 for Eclipse がインストールされていること

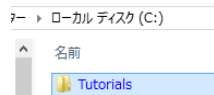
3. チュートリアル手順の概要

1. チュートリアルの準備
2. Eclipse の起動
3. メインフレーム COBOL プロジェクトの作成
4. プロジェクトプロパティの設定
5. ビルドの実行
6. Enterprise Server インスタンスの設定
7. Enterprise Server インスタンスの開始と確認
8. JCL の実行
9. プロシージャライブラリの作成
10. COBOL バッチプログラムの実行
11. COBOL バッチプログラムのデバッグ
12. Enterprise Server インスタンスの停止

3.1 チュートリアル準備

例題プログラムに関連する資源を用意します。

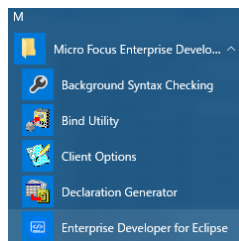
- 1) 使用する例題プログラムは、キットに添付されている Tutorials.zip に圧縮されています。これを C:¥ 直下に解凍します。



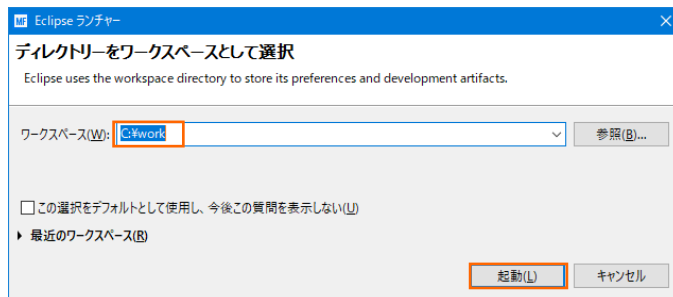
- 2) Eclipse のワークスペースで使用する「work」フォルダを C:¥ 直下に作成します。

3.2 Eclipse の起動

- 1) メニューから [Enterprise Developer for Eclipse] を起動します。



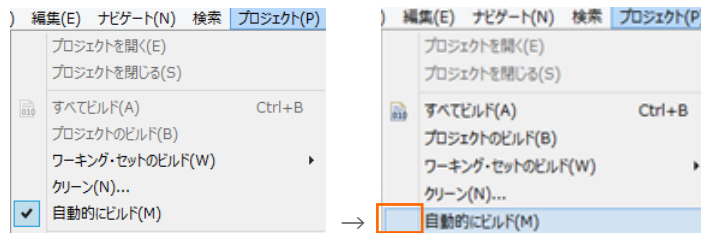
- 2) 前項で作成した C:¥work をワークスペースへ指定して、[起動] ボタンをクリックします。



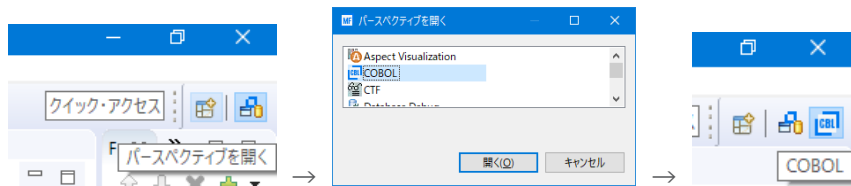
- 3) [ようこそ] タブが表示されたら [Open COBOL Perspective] をクリックして、COBOL パースペクティブを開きます。



- 4) パースペクティブ表示後、[プロジェクト] プルダウンメニューの [自動的にビルド] を選択して、これをオフにします。

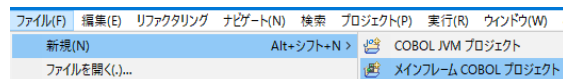


- 5) COBOL パースペクティブが開いていない場合は Eclipse 右上の [パースペクティブを開く] アイコンをクリックして表示後、[COBOL] を選択して [開く] ボタンをクリックします。

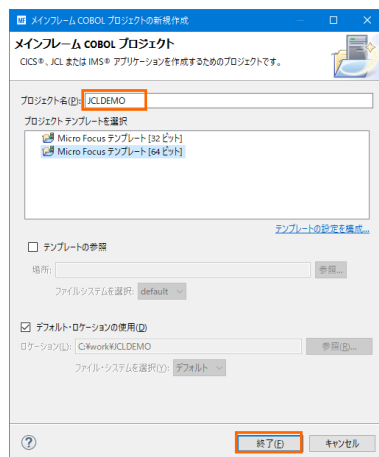


3.3 メインフレーム COBOL プロジェクトの作成

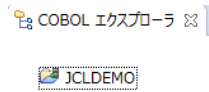
- 1) 用意した例題ソースをインポートします。[ファイル] プルダウンメニューから [新規] > [メインフレーム COBOL プロジェクト] を選択します。



- 2) [プロジェクト名] は任意ですが、ここでは JCLDEMO を入力し、テンプレートは 64 ビットを選択して [終了] ボタンをクリックします。

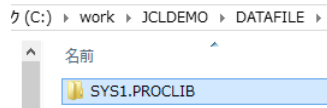


3) COBOL エクスプローラーに作成したプロジェクトが表示されます。

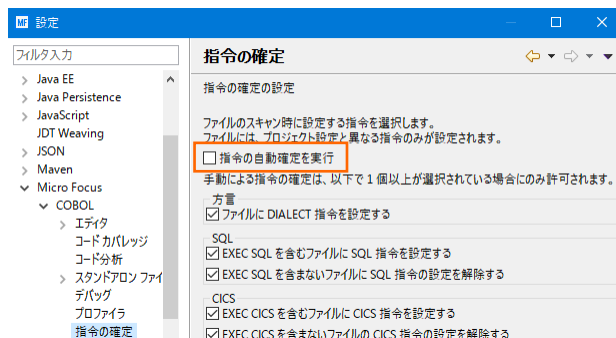


4) プロジェクトを作成したことにより C:\work\JCLDEMO フォルダが作成されています。このフォルダ配下に JES 機能で使用するフォルダを Windows エクスプローラーを使用してあらかじめ用意しておきます。

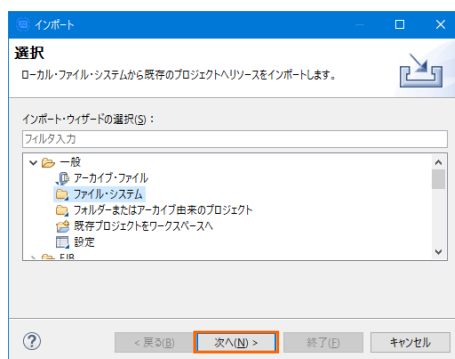
- ① カタログファイルやスプールファイルを配置するため DATAFILE フォルダを C:\work\JCLDEMO 配下へ作成します。
- ② プロシージャファイルを配置するため、プロシージャライブラリとして SYS1.PROCLIB フォルダを C:\work\JCLDEMO\DATAFILE 配下へ作成します。



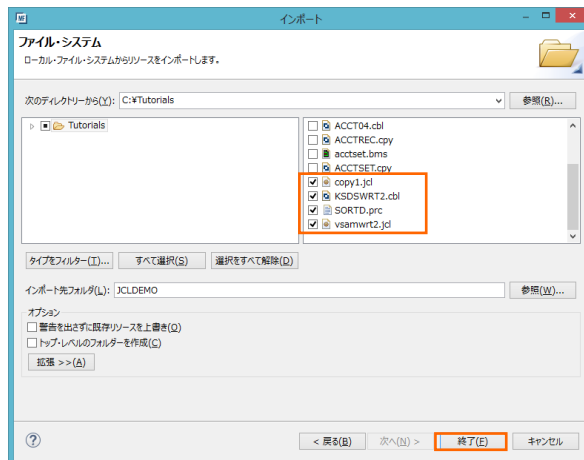
5) 既存ファイルのインポート時、自動的にコンパイル指令が指定される機能が用意されていますが、本チュートリアルではこれを解除します。[ウィンドウ] プロダクションメニューの [設定] > [Micro Focus] > [COBOL] > [指令の確定] > [指令の自動確定を実行] チェックボックスをオフにして [Apply and Close] ボタンをクリックします。



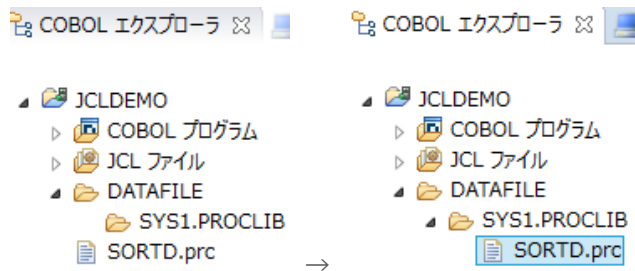
6) 用意した例題プログラム類をインポートします。JCLDEMO プロジェクトを右クリックして [インポート] > [インポート] を選択し、インポートウィンドウにて [一般] > [ファイル・システム] を選択後 [次へ] ボタンをクリックします。



- 7) C:\¥Tutorials を [次のディレクトリーから] へ指定すると内容が表示されますので、最後から 4 ファイルをオンにして [終了] ボタンをクリックします。この実行により、プロジェクトフォルダへ例題プログラムが配置されます。

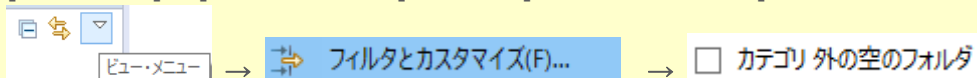


- 8) COBOL エクスプローラー内に表示されている JCLDEMO プロジェクトにインポートしたファイルが表示されていることを確認後、作成した SYS1.PROCLIB フォルダへ SORTD.prc ファイルをドラッグ & ドロップして移動します。



注意

フォルダ内が空のため作成したフォルダが表示されない場合があります。その際は COBOL エクスプローラー 右上の [ビュー・メニュー] > [フィルタとカスタマイズ] を選択後 [カテゴリ外の空のフォルダ] のチェックをオフにしてください。

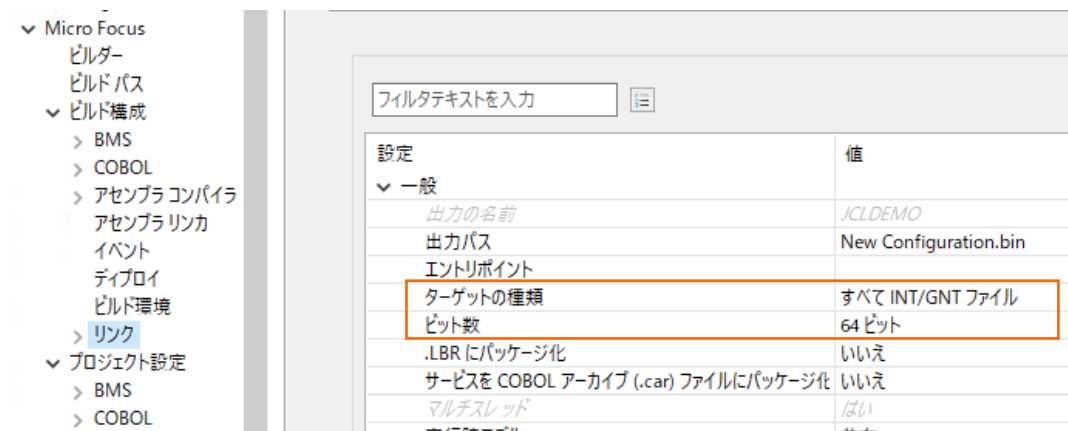


3.4 プロジェクトプロパティの設定

プログラム内容に沿ったプロジェクトのプロパティを設定します。

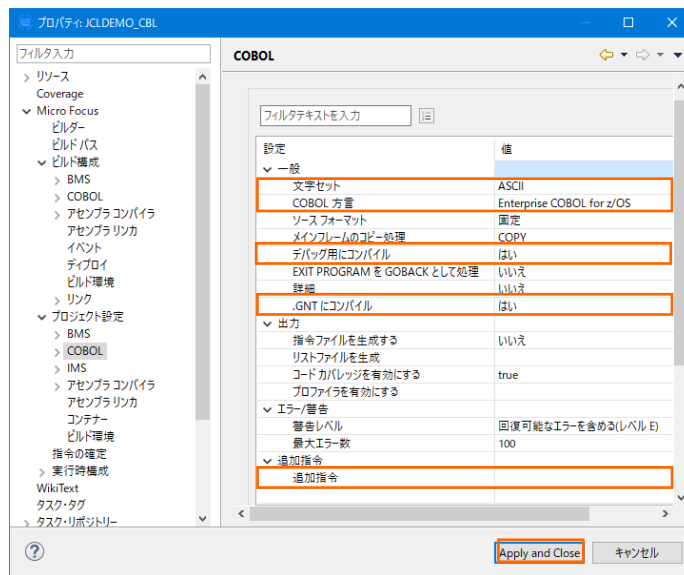
- 1) COBOL エクスプローラー内の JCLDEMO プロジェクトを右クリックして [プロパティ] を選択します。
- 2) 左側ツリービューの [Micro Focus] > [ビルド構成] > [リンク] を選択して、下記項目を指定します。指定後は [適用] ボタンをクリックしてください。

項目名	説明
ターゲットの種類	実行ファイル形式を指定します。ここでは [全て INT/GNT ファイル] を選択します。
プラットフォーム ターゲット	稼働ビット数を指定します。ここでは [64 ビット] を指定します。



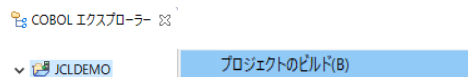
- 3) 左側ツリービューの [Micro Focus] > [プロジェクト設定] > [COBOL] を選択して、下記項目を指定します。指定後は [Apply and Close] ボタンをクリックしてください。

項目名	説明
文字集合	EBCDIC または ASCII を指定します。ここでは [ASCII] を選択します。
言語方言	COBOL 言語方言を指定します。 例題プログラムは IBM Enterprise COBOL の方言を使用しているため、ここでは [Enterprise COBOL for z/OS] を指定します。
デバッグ用にコンパイル	デバッグ実行時に使用するファイルを生成するようにチェックをオンに指定します。
.GNT にコンパイル	実行ファイル形式を GNT に指定します。
追加指令	ここでは指定しません。

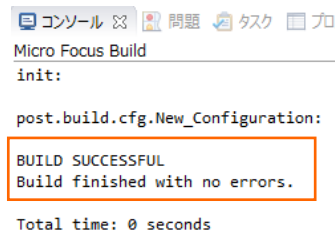


3.5 ビルドの実行

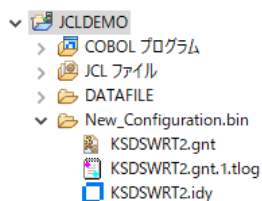
- 1) COBOL エクスプローラー内のプロジェクトを右クリックして [プロジェクトのビルド] を選択するとビルドが実行されます。



- 2) コンソールタブで成功を確認します。



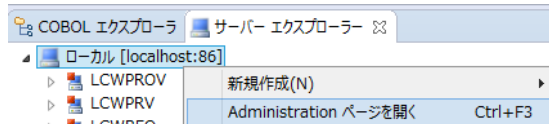
- 3) COBOL エクスプローラーのプロジェクト内に存在する New_Configuration.bin フォルダ配下の実行ファイル (.gnt ファイル) が作成されていることを確認してください。



3.6 Enterprise Server インスタンスの設定

Enterprise Developer は JES のエミュレーション機能を搭載している開発用 Enterprise Server インスタンスを内包しており、各開発者がこのインスタンスを占有してメインフレームアプリケーションのテスト実行やデバッグを行うことができます。本番環境にはコンパイラなどを含まない実行環境製品 Enterprise Server をインストールし、本番用インスタンス上でアプリケーションを稼働させます。

- 1) Enterprise Server インスタンスを作成します。サーバー エクスプローラタブの [ローカル] を右クリックして [Administration ページを開く] を選択します。デフォルトポート番号は 86 です。



- 2) Enterprise Server Administration 画面に Enterprise Server インスタンス一覧が表示されますので、画面の左下にある [追加] ボタンをクリックします。



- 3) サーバー名には JCLDEMO を入力、動作モードは 64-bit を指定して [次へ] ボタンをクリックします。

サーバーの追加 (ページ 1 / 3):

サーバー名: JCLDEMO

作業モード:

☐ 32ビット ☒ 64ビット

サーバーの作成後に、その作業モードの選択の変更は不可能です。ピーまたはインポートをおこなう際、作業モードの変更は可能です。

キャンセル 次へ >>



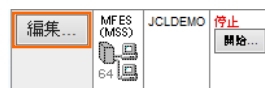
重要

実行ファイル生成に指定した稼働ビット数 = Enterprise Server インスタンス稼働ビット数である必要があります。

- 4) 画面の Page 2/3 ではそのまま [次へ] ボタンを、Page 3/3 では [TN3270 リスナーの作成] チェックボックスをオフにして [追加] ボタンをクリックすると、JCLDEMO という名前の 64 ビットアプリケーション稼働用 Enterprise Server インスタンスが追加されます。



- 5) 左にある [編集] ボタンをクリックします。



6) [サーバー] > [プロパティ] > [一般] タブ内の下記項目を設定します。

- ① [動的デバッグを許可] チェックボックスをオンにします。この指定により、Eclipse から動的デバッグが可能になります。

開始オプション:

共有メモリページ数:	512	サービス実行プロセス:	2
共有メモリクッション:	32	要求ライセンス:	10
ローカルコンソールを表示:	☐	動的デバッグを許可:	☒
Start on System Start:	☐	64-Bit Working Mode:	☒
以前のログを削除:	☐	コンソールログサイズ (K):	0

- ② [適用] ボタンをクリックします。

7) [サーバー] > [プロパティ] > [MSS] > [JES] タブで表示される画面の各項目を設定します。入力後は [Apply] ボタンをクリックします。

項目名	説明
メインフレーム サブシステム サポート有効	[MSS] タブ配下の設定をオン、オフ指定します。ここではオンに指定します。
ジョブ入力サブシステム 有効	[JES] タブ配下の設定をオン、オフ指定します。ここではオンに指定します。
JES プログラム パス	COBOL アプリケーション実行ファイルが存在するパスを指定します。
システムカタログ	カタログファイルを出力するパスと、そのファイル名称を指定します。
データセットの省略時ロケーション	ジョブ実行時に生成されるスプールデータやカタログされるデータセットのデフォルトパスを指定します。
システムプロシージャライブラリ	プロシージャライブラリの名前を指定します。 ここでは SYS1.PROCLIB を入力します。

メインフレーム サブシステム サポート有効: ☒

CICS (✓) JES... (✓) IMS... PL/I

一般 イニシエータ (1) プリンター (0)

ジョブ入力サブシステム有効: ☒

JESプログラムパス:
C:\work\JCLDEMO\New_Configuration.bin

システムカタログ:
C:\work\JCLDEMO\DATAFILE\catalog.dat

データセットの省略時ロケーション:
C:\work\JCLDEMO\DATAFILE

システムプロシージャライブラリ:
SYS1.PROCLIB

Fileshare 構成 ロケーション:

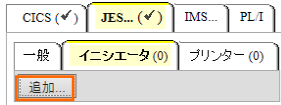
Apply



重要

入力値は全て半角英数字で指定してください。
これらのフィールドでは改行を入れないように注意してください。

- 8) [サーバー] > [プロパティ] > [MSS] > [JES] > [イニシエータ] タブを表示し、左下の [追加] ボタンをクリックします。



- 9) 下記画面のように入力して [追加] ボタンをクリックします。この指定により JCLDEMO インスタンスが開始時にイニシエータが稼働し、ジョブクラス A,B,C のジョブが実行可能になります。

名前:
INITABC

Class:
ABC

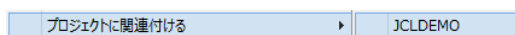
説明:
A、B、Cクラスのイニシエータ

- 10) 画面左上の [Home] をクリックして一覧画面に戻ります。

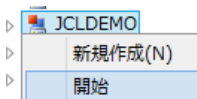


3.7 Enterprise Server インスタンスの開始と確認

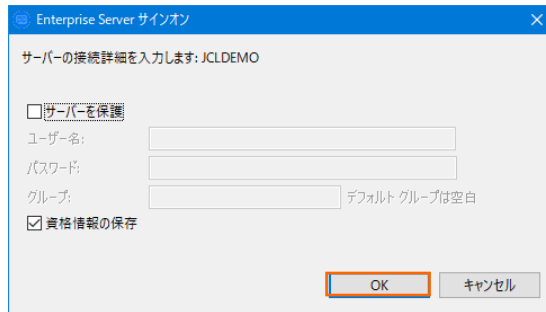
- 1) サーバークリエイター内に JCLDEMO インスタンスが表示されていることを確認します。表示されていない場合は [localhost] を右クリックし、[更新] を選択してリフレッシュしてください。
- 2) サーバークリエイター内の JCLDEMO インスタンスを右クリックし、[プロジェクトに関連付ける] > [JCLDEMO] を選択します。これにより JCLDEMO プロジェクトから実行されるアプリケーションは JCLDEMO インスタンスで処理されることになります。



- 3) JCLDEMO インスタンスを右クリックして [開始] を選択します。



- 4) 下記ウィンドウが表示された場合は、ここではユーザーによる制限を行わないため [OK] ボタンをクリックします。



Enterprise Server サインオン

サーバーの接続詳細を入力します: JCLDEMO

☐ サーバーを保護

ユーザー名:

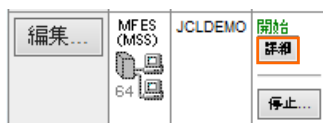
パスワード:

グループ: デフォルト グループは空白

☒ 資格情報の保存

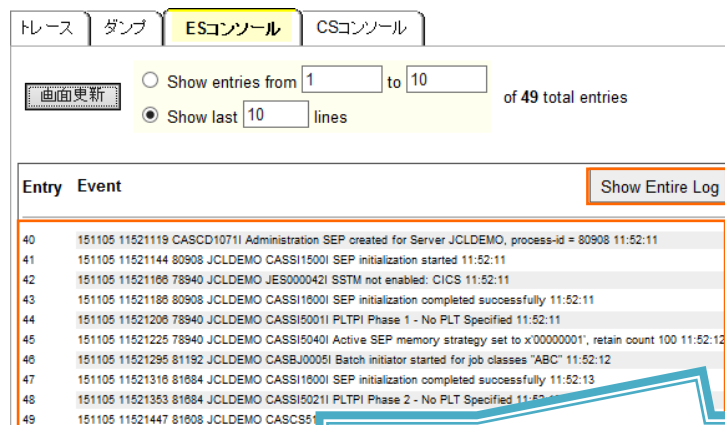
OK キャンセル

- 5) Enterprise Server Administration 画面へ移動して開始状態であることを確認後、[詳細] ボタンをクリックします。



- 6) [サーバー] > [診断] > [ES コンソール] で JCLDEMO インスタンスのコンソールログをリアルタイムにチェックすることができます。また [Show Entire Log] をクリックしてログ全体を表示させることも可能です。

正常に開始されたことを確認します。



トレース ダンプ ESコンソール CSコンソール

画面更新

Show entries from 1 to 10 of 49 total entries

Show last 10 lines

Entry Event Show Entire Log

40	151105 11521119 CASCD10711 Administration SEP created for Server JCLDEMO, process-id = 80908 11:52:11
41	151105 11521144 80908 JCLDEMO CASSI15001 SEP initialization started 11:52:11
42	151105 11521166 78940 JCLDEMO JES0000421 SSTM not enabled: CICS 11:52:11
43	151105 11521186 80908 JCLDEMO CASSI16001 SEP initialization completed successfully 11:52:11
44	151105 11521206 78940 JCLDEMO CASSI50011 PLTPI Phase 1 - No PLT Specified 11:52:11
45	151105 11521225 78940 JCLDEMO CASSI50401 Active SEP memory strategy set to x'00000001', retain count 100 11:52:12
46	151105 11521295 81192 JCLDEMO CASBJ00051 Batch initiator started for job classes "ABC" 11:52:12
47	151105 11521316 81684 JCLDEMO CASSI16001 SEP initialization completed successfully 11:52:13
48	151105 11521353 81684 JCLDEMO CASSI50211 PLTPI Phase 2 - No PLT Specified 11:52:13
49	151105 11521447 81608 JCLDEMO CASCS51

【JES 機能の正常開始ログ抜粋】

```
JES000051I Job Entry Subsystem (JES) services initialized 17:24:39
JES000050I JES 5 digit job numbering support enabled 17:24:39
CASCD1080I JES Initiator created for Server JCLDEMO, process-id = 5912 17:24:39
CASBJ0023I Batch initiator INITABC: class(es) "ABC" 17:24:39
CASBJ0006I Batch initiator initialization started 17:24:40
```



注意

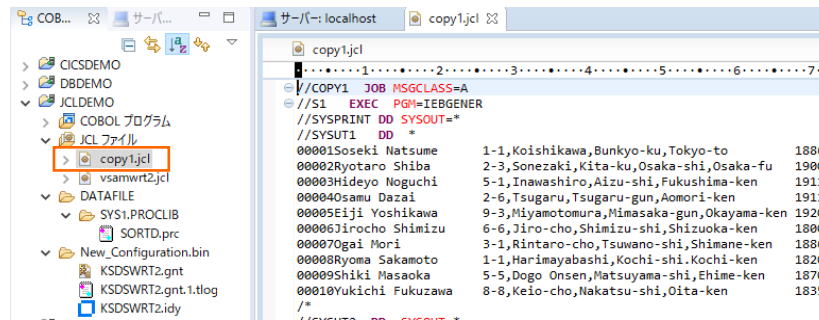
いくつかのサービス開始が失敗してもインスタンスは開始されますので、ログ内容を必ず確認してください。

- 7) 画面左上の [Home] をクリックして一覧画面に戻ります。

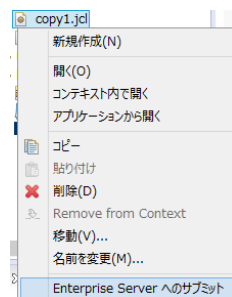
3.8 JCL の実行

現在 JCLDEMO インスタンスが稼働していますので、例題プログラムを実行することができます。まずは簡単な JCL を実行してみます。

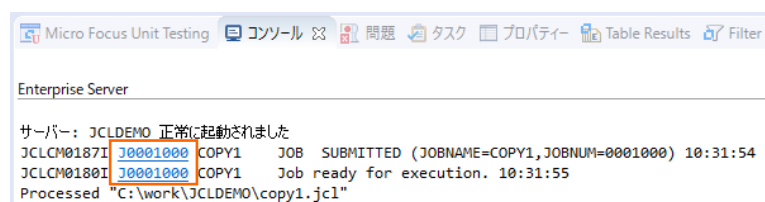
- 1) COBOL エクスプローラー内にある JCLDEMO プロジェクト配下の copy1.jcl をダブルクリックし、エディタで内容を確認します。この JCL は IEBGENER ユーティリティを使用して、JCL 内に書かれたインラインデータを SYSOUT に書き出しています。



- 2) COBOL エクスプローラー内の copy1.jcl を右クリックして [Enterprise Server へのサブミット] を選択すると、この JCL が実行されます。



- 3) コンソールタブに JOB 実行ログと JOB 番号が表示されますので、リンクをクリックします。



- 4) この JOB 番号にかかわるスプール一覧が表示されます。先頭の [JESYSMEG] をクリックしてジョブログを確認します。

J0001000	Name: COPY1	Status: Complete		
Hold	Class: A	Priority: 00		
Update	User: JESUSER	COND: 0000		
Delete	File: \$TXRFDIR/t000000015.t			
JCLCM0188I J0001000 COPY1 JOB STARTED 10:31:56				
JCLCM0182I J0001000 COPY1 JOB ENDED - COND CODE 0000 10:31:56				
Status Class DD Name Step Nbr. Proc Step Records				
Details	Hold	A JESYSMSG	0	31
Details	Ready	A SYSPRINT	1	4
Details	Ready	A SYSUT2	1	10

- 5) ジョブログの内容を確認すると、この JOB が正常に終了していることが確認できます。

```
JCLCM0182I JOB ENDED - COND CODE 0000
```

- 6) 右クリックで [前へ戻る] を選択し、スプール一覧から [SYSPRINT] をクリックすると、IEBGENER ユーティリティの実行ログが記録されていることが確認できます。

```
Micro Focus MFJGENER Utility Version ED5.0_001
Copyright (C) Micro Focus 1997-2019. All rights reserved.

JCLGN0110I(00) - 0000000010 RECORDS COPIED FROM SYSUT1 TO SYSUT2
```

- 7) 右クリックで [前へ戻る] を選択し、スプール一覧から [SYSUT2] をクリックすると、出力されたスプールの内容が確認できます。

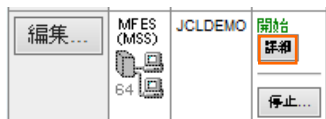
```
00001Soseki Natsume      1-1,Koishikawa,Bunkyo-ku,Tokyo-to      1886
00002Ryotaro Shiba       2-3,Sonezaki,Kita-ku,Osaka-shi,Osaka-fu    1900
00003Hideyo Noguchi      5-1,Inawashiro,Aizu-shi,Fukushima-ken     1911
00004Osamu Dazai         2-6,Tsugaru,Tsugaru-gun,Aomori-ken        1911
00005Eiji Yoshikawa      9-3,Miyamotomura,Mimasaka-gun,Okayama-ken 1920
00006Jirocho Shimizu     6-6,Jiro-cho,Shimizu-shi,Shizuoka-ken      1800
00007Ogai Mori           3-1,Rintaro-cho,Tsumano-shi,Shimane-ken    1886
00008Ryoma Sakamoto      1-1,Harimayabashi,Kochi-shi,Kochi-ken      1820
00009Shiki Masaoka       5-5,Dogo Onsen,Matsuyama-shi,Ehime-ken     1870
00010Yukichi Fukuzawa    8-8,Keio-cho,Nakatsu-shi,Oita-ken         1835
```

3.9 プロシージャライブラリの作成

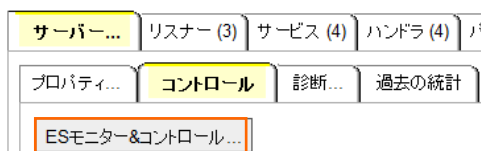
プロシージャを使用する JCL を実行するために、プロシージャライブラリを作成します。Enterprise Server インスタンスではプロシージャを区分データセットのメンバーとして配置します。前項で作成したプロシージャライブラリとなるフォルダをカタログします。

```
▼ DATAFILE
  ▼ SYS1.PROCLIB
    SORTD.prc
```

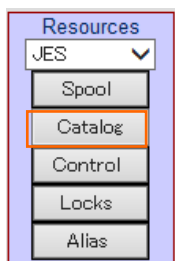
- 1) Enterprise Server Administration 画面へ移動して JCLDEMO インスタンスの [詳細] ボタンをクリックします。



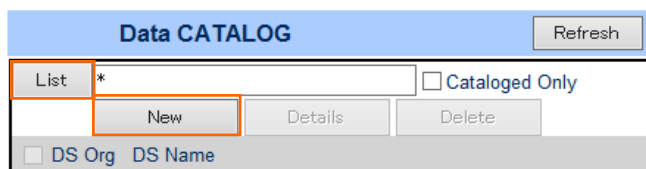
- 2) [サーバー] > [コントロール] > [ES モニター & コントロール] ボタンをクリックします。



- 3) 画面左の中央部にある [Resources] 直下のプロダクションメニューから [JES] を選択後、表示された [Catalog] ボタンをクリックします。前項で確認したスプールについても [Spool] ボタンをクリックすることにより、全てが参照可能になります。

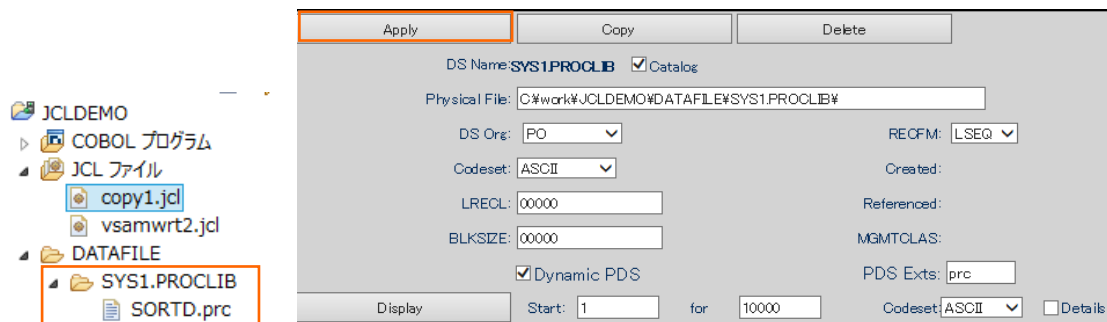


- 4) [List] ボタンをクリックして、カタログ情報の一覧を表示すると、現在は何も登録されていないことが確認できます。新規に作成するため [New] ボタンをクリックします。

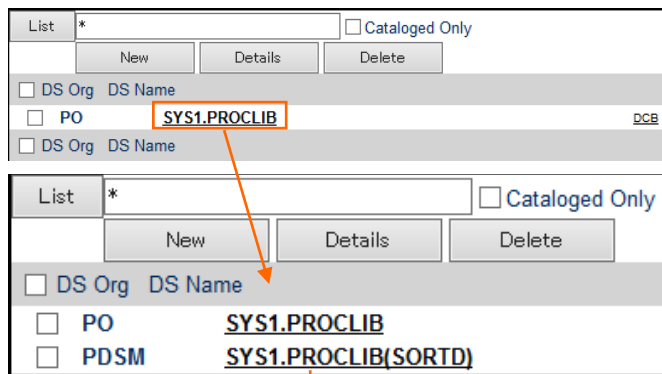


- 5) カタログエントリの画面が表示されますので、以下のように入力し [Apply] ボタンをクリックします。

項目名	説明
DS Name	SYS1.PROCLIB を入力します。
Physical File	物理パスを指定します。ここでは前項で作成したフォルダパスを入力します。
DS Org	区分データセットである PO を選択します。
RECFM	行順である LSEQ を選択します。
Dynamic PDS	プロシージャファイルをフォルダ配下に保持する動的 PDS の場合にオンにします。ここではオンを指定します。
PDS Exts	プロシージャファイル拡張子を指定します。ここでは prc を入力します。



- 6) カタログ一覧に戻り [List] ボタンをクリックすると作成した PO が表示されます。[SYS1.PROCLIB] をクリックすると配置されているメンバーが確認できます。



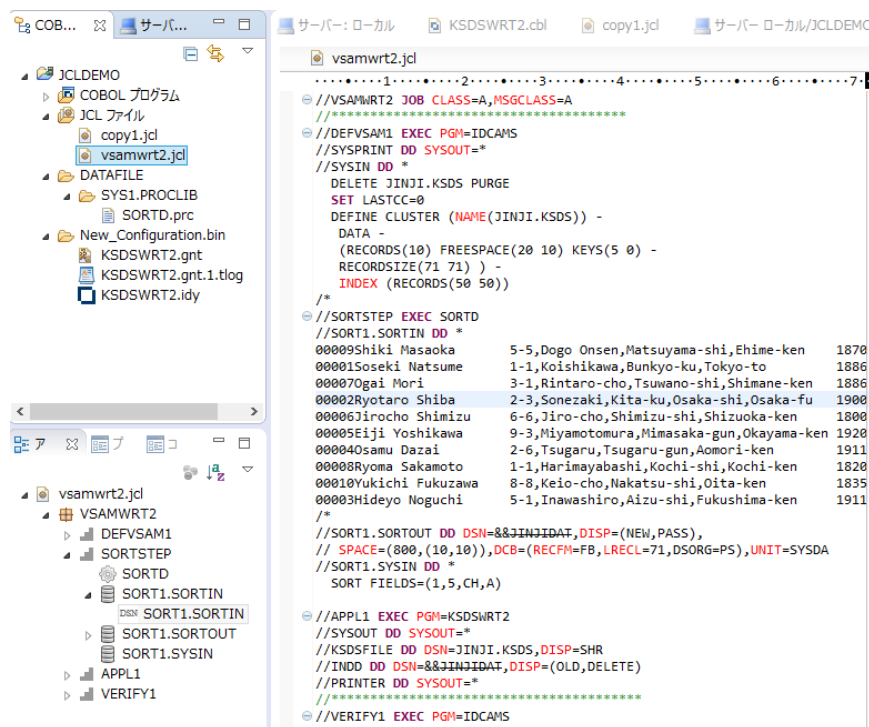
[SYS1.PROCLIB(SORTD)] をクリックすると内容が確認できます。

```
//SORTD PROC
//SORT1 EXEC PGM=SORT
//SYSOUT=*
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(10,10))
// PEND
```

3.10 COBOL バッチプログラムの実行

COBOL プログラムを含む JOB を実行してみます。

- 1) COBOL エクスプローラー内にある JCLDEMO プロジェクト配下の vsamwrt2.jcl をダブルクリックし、エディタで内容を確認します。



① ステップ 1 : DEFVSAM1

IDCAMS を使用して VSAM データセット JINJI.KSDS を削除し、クラスターを持つ KSDS として再作成します。

② ステップ 2 : SORTSTEP

前項で登録した SORTD.prc を使用して、JINJI.KSDS ファイルへの書き込み用データをソートします。

③ ステップ 3 : APPL1

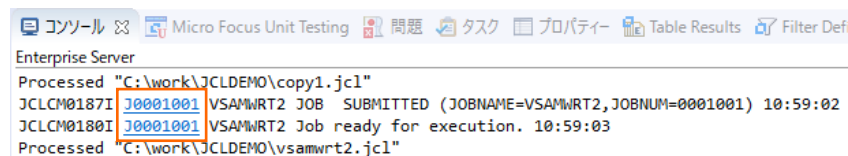
アプリケーション KSDSWRT2 を呼び出しステップ 2 でソートされたデータをステップ 1 で定義した VSAM クラスターに書き込みを行います。同時に書き込まれたデータを DD=PRINTER へ出力します。

④ ステップ 4 : VERIFY1

出力内容確認のため、IDCAMS の REPRO で内容を出します。

2) 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] を選択して、この JCL を実行します。

3) コンソールタブに JOB 実行ログと JOB 番号が表示されますので、リンクをクリックします。



- 4) この JOB 番号にかかわるスプルー一覧が表示されます。

[COND CODE] には [0008] が返却されていますが、これは初回実行時、ステップ 1 で削除該当ファイルが見つからないためです。問題ありません。

JOB01006	Name: VSAMMRT2	Status: Output Hold				
Release	Class: A	Priority: 00				
Update	User: JESUSER	COND: 00008				
JCLCM0188I JOB01006 VSAMMRT2 JOB STARTED 14:58:44						
JCLCM0182I JOB01006 VSAMMRT2 JOB ENDED - COND CODE 0008 14:58:44						
Status	Class	DD Name	Step	Nbr.	Proc Step	Records
Details	Hold	A	JESYSMSG	0		76
Details	Ready	A	SYSPRINT	1		18
Details	Ready	A	SYSOUT	2	SORT1	12
Details	Ready	A	SYSOUT	3		1

- 8) [JESYSMSG] の内容を確認すると、各ステップの COND CODE が確認できます。ジョブが異常終了した場合にはここでエラーの原因を調査することができます。

```
JCLCM0191I STEP ENDED VERIFY1 - COND CODE 0000
JCLCM0182I JOB ENDED - COND CODE 0008
```

- 9) 右クリックで [前へ戻る] を選択し、スプルー一覧から DEFVSAM1 ステップの [SYSPRINT] をクリックして内容を確認します。

```
Micro Focus MFJAMS Utility Version ED5.0_001
Copyright (C) Micro Focus 1997-2019. All rights reserved.

DELETE JINJI.KSDS PURGE
JCLAM0115E(08) - ENTRYNAME NOT CATALOGED [JINJI.KSDS]

SET LASTCC=0
JCLAM0140I(00) - LASTCC set to 0.

DEFINE CLUSTER (NAME(JINJI.KSDS)) -
  DATA -
  (RECORDS(10) FREESPACE(20 10) KEYS(5 0) -
  RECORDSIZE(71 71) ) -
  INDEX (RECORDS(50 50))
JCLAM0113I(00) - ENTRYNAME DEFINED [JINJI.KSDS]
```

- 10) 右クリックで [前へ戻る] を選択し、スプルー一覧から SORTSTEP の [SYSOUT] をクリックしてソート内容を確認します。

```
Content-Type: text/plain

Micro Focus MFJSORT ユーティリティ 3.0.00

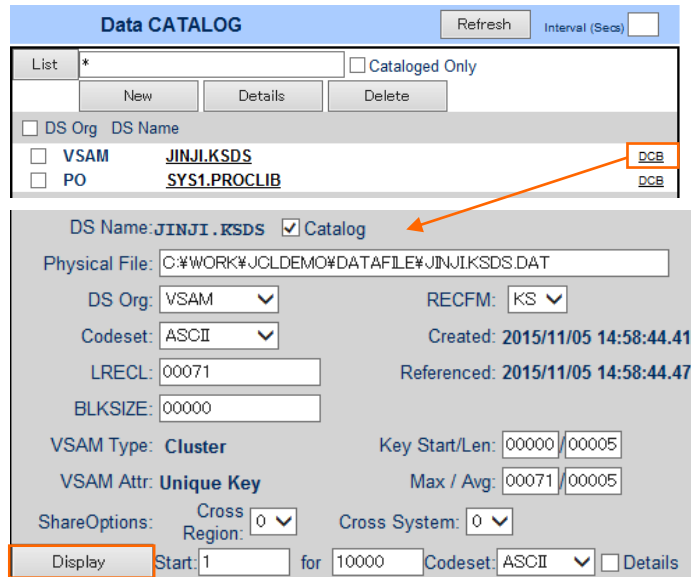
SORT FIELDS=(1,5,CH,A)

SORT204I: ***** ソート結果 *****
SORT205I: INPUT ファイル 'SORTIN'
          入力レコード          10 件
          使用レコード          10 件
SORT206I: OUTPUT ファイル 'SORTOUT'
          入力レコード          10 件
          使用レコード          10 件
          出力レコード          10 件
SORT399I: Micro Focus MFJSORT ユーティリティ終了
```

11) 右クリックで [前へ戻る] を選択し、スプルー一覧から他ステップに関しても確認してみてください。

12) 前項と同様の手順で、この JOB によってカタログされた情報を確認します。カタログ一覧で [List] ボタンをクリックすると、VSAM ファイル JINJI.KSDS がカタログされていることが確認できます。

右端の [DCB] をクリックすると登録情報が表示されます。



Data CATALOG Refresh Interval (Secs) []

List: * ☐ Cataloged Only

New Details Delete

☐ DS Org DS Name

☐ VSAM	JINJI.KSDS	DCB
☐ PO	SYS1.PROCLIB	DCB

DS Name: JINJI.KSDS ☒ Catalog

Physical File: C:\WORK\JCLDEMO\DATAFILE\JINJI.KSDS.DAT

DS Org: VSAM RECFM: KS

Codeset: ASCII Created: 2015/11/05 14:58:44.41

LRECL: 00071 Referenced: 2015/11/05 14:58:44.47

BLKSIZE: 00000

VSAM Type: Cluster Key Start/Len: 00000 / 00005

VSAM Attr: Unique Key Max / Avg: 00071 / 00005

ShareOptions: Cross Region: 0 Cross System: 0

Display Start: 1 for 10000 Codeset: ASCII ☐ Details

[Display] ボタンをクリックすると、ファイルの内容が表示されます。

Content-Type: text/plain

00001Soseki Natsume	1-1,Koishikawa,Bunkyo-ku,Tokyo-to	1886
00002Ryotaro Shiba	2-3,Sonezaki,Kita-ku,Osaka-shi,Osaka-fu	1900
00003Hideyo Noguchi	5-1,Inawashiro,Aizu-shi,Fukushima-ken	1911
00004Osamu Dazai	2-6,Tsugaru,Tsugaru-gun,Aomori-ken	1911
00005Eiji Yoshikawa	9-3,Miyatomomura,Mimasaka-gun,Okayama-ken	1920
00006Jirocho Shimizu	6-6,Jiro-cho,Shimizu-shi,Shizuoka-ken	1800
00007Gai Mori	3-1,Rintaro-cho,Tsuwano-shi,Shimane-ken	1886
00008Ryoma Sakamoto	1-1,Harimayabashi,Kochi-shi,Kochi-ken	1820
00009Shiki Masaoka	5-5,Dogo Onsen,Matsuyama-shi,Ehime-ken	1870
00010Yukichi Fukuzawa	8-8,Keio-cho,Nakatsu-shi,Oita-ken	1835

3.11 COBOL バッチプログラムのデバッグ

JCL から実行される COBOL プログラムをデバッグします。

1) [実行] ブルダウンメニューの [デバッグの構成] を選択します。



ソース ナビゲート(N) 検索 プロジェクト(P) 実行(R)

実行点をリセット

実行(R) Ctrl+F11

デバッグ(D) F11

実行履歴(T) ▶

実行(S) ▶

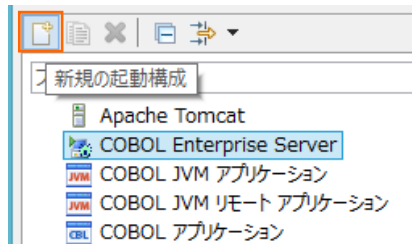
実行構成(N)... ▶

デバッグ履歴(H) ▶

デバッグ(G) ▶

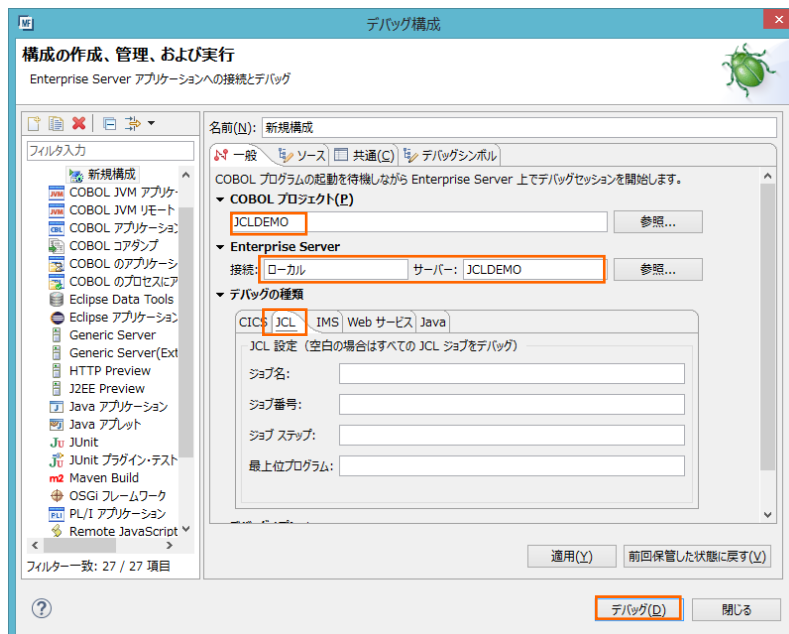
デバッグの構成(B)...

- 2) 左側のメニューから [COBOL Enterprise Server] を選択して、左上の [新規の起動構成] アイコンをクリックします。

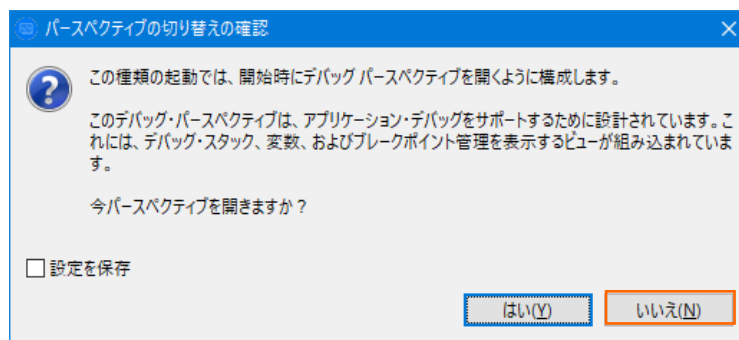


- 3) [COBOL プロジェクト] へ対象となる JCLDEMO プロジェクトを入力し、[Enterprise Server] へ実行させる JCLDEMO インスタンスを指定します。

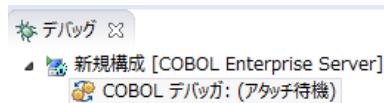
[デバッグの種類] は「JCL」タブを選択した状態で、[デバッグ] ボタンをクリックします。



- 4) パースペクティブの切り替え確認ウィンドウが表示されますが、COBOL エクスプローラーから JCL を実行するため、ここでは [いいえ] ボタンをクリックします。

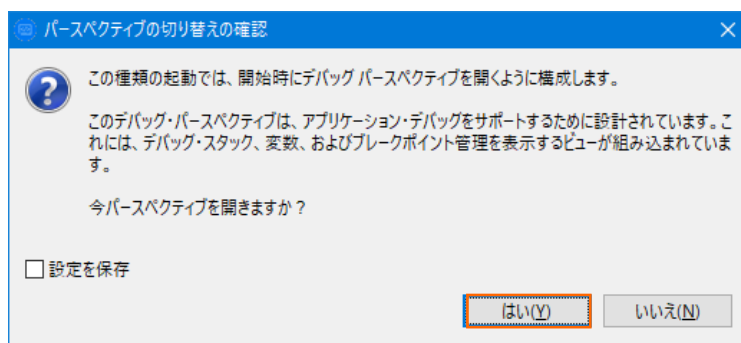


- 5) デバッグタブで [アタッチ待機] 状態になったことを確認します。

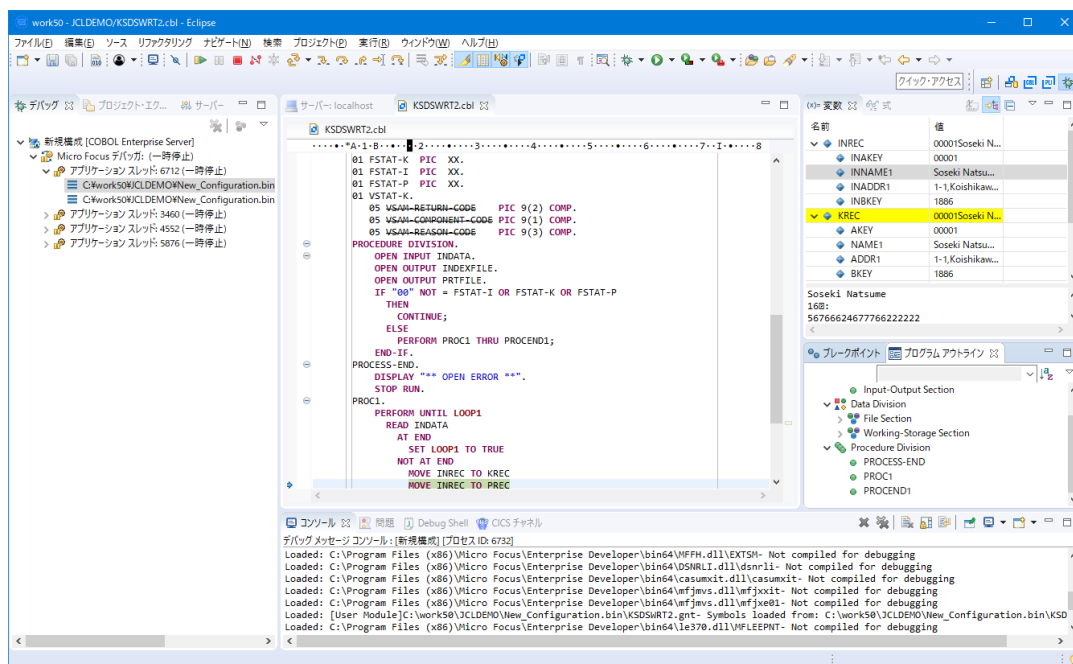


- 6) COBOL エクスプローラー内の vsamwrt2.jcl を右クリックして [Enterprise Serverへのサブミット] を選択して、JCL を実行します。

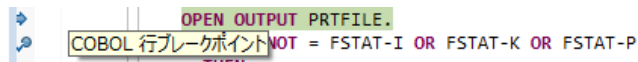
- 7) 再度、パースペクティブの切り替え確認ウィンドウが表示されますので、ここでは [はい] ボタンをクリックし、デバッグ用のパースペクティブを開きます。



- 8) 少し待つとデバッグセッションが開始して、プログラムのステップ実行が可能になります。[F5] キーもしくは [実行] ブルダウンメニューから [ステップイン] を選択してステップを進めることができ、変数タブでは使用している変数の値が確認できます。



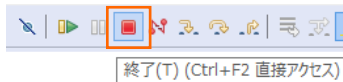
- 9) 希望のステップの左端をダブルクリックすることにより、ブレークポイントを設定することも可能です。



- 10) 先に進める場合は画面上部の再開アイコンをクリックします。

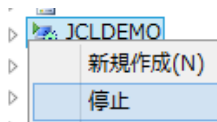


- 11) デバッグを終了させるため、画面上部の終了アイコンをクリックします。

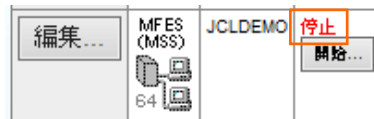


3.12 Enterprise Server インスタンスの停止

- 1) COBOL パースペクティブへ戻り、JCLDEMO インスタンスを停止します。



- 2) JCLDEMO インスタンスの停止を確認後、Eclipse を終了します。



WHAT'S NEXT

- リモート メインフレーム COBOL 開発 : CICS Eclipse 編
- 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。