# Micro Focus Enterprise Developer チュートリアル

## リモート 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 を使用したリモート 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リモートマシン OS ： Red Hat Enterprise Linux Server release 7.5
- 本チュートリアルで使用したローカルマシン OS ： Windows 10 Enterprise
- リモートマシンに Micro Focus Enterprise Developer 5.0 for Linux and Unix がインストールされていること
- ローカルマシンに Micro Focus Enterprise Developer 5.0 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. リモートマシンの準備
3. Eclipse の起動
4. リモート メインフレーム COBOL プロジェクトの作成
5. プロジェクトプロパティの設定
6. ビルドの実行
7. Enterprise Server インスタンスの設定
8. Enterprise Server インスタンスの開始と確認
9. JCL の実行とデバッグ
10. Enterprise Server インスタンスの停止

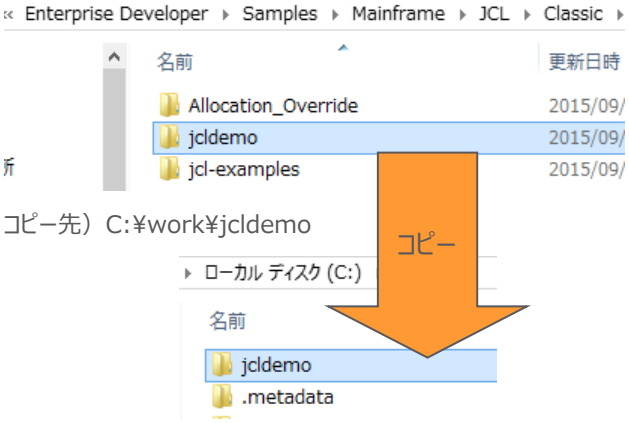
## 3.1 チュートリアルの準備

例題プログラムに関連するリソースを用意します。

1) Eclipse のワークスペースで使用する work フォルダを C ディレクトリー直下に作成します。

2) 製品をインストールしたフォルダ配下に含まれている例題プログラム jcldemo フォルダを、作成した C:¥work へコピーします。

例）C:¥Users¥Public¥Documents¥Micro Focus¥Enterprise Developer¥Samples¥Mainframe¥JCL¥Classic¥jcldemo




## 3.2 リモートマシンの準備

ここではリモートマシンの準備を行うために、リモートマシンへルートユーザーでログインします。

1） 環境変数 LANG に SJIS ロケールを設定します。

コマンド例）export LANG=ja_JP.sjis

```
#export LANG=ja_JP.sjis
```

2） COBOL を実行する環境を設定します。製品フォルダ配下の bin フォルダ内に存在する cobsetenv を実行すると、環境変数の COBDIR が設定された旨メッセージが表示されます。

コマンド例）. /opt/mf/ED50/bin/cobsetenv

```
#. /opt/mf/ED50/bin/cobsetenv
COBDIR set to /opt/mf/ED50
```

3） COBOL 作業モードを設定します。

COBOL の作業モード(32-bit または 64-bit)を指定します。 cobmode コマンドまたは環境変数 COBMODE を使用して設定します。

64-bit 設定コマンド例）export COBMODE=64

```
#export COBMODE=64
```

4） Micro Focus Directory Server (MFDS) を起動します。

Web ブラウザからリモートマシンのホスト名:86（デフォルトポート番号）を指定して、Enterprise Server Administration 画面が表示されない場合は、mfds コマンドを実行して MFDS を起動します。32-bit 環境用には mfds32 コマンド、64-bit 環境用には mfds64 コマンドを明示的に実行することも可能です。

コマンド例）mfds &

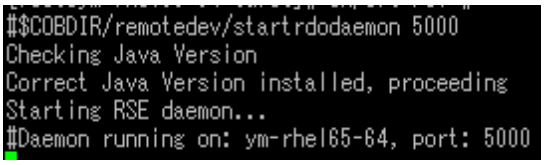
上記 “&” を付加すると、設定済の COBOL 環境変数を基に別プロセスで mfds が起動されます。


5） ローカルマシンからのアクセス方法を RSE に指定する場合は（3.4-5 項参照）、接続ポートの解放を行います。本チュートリアルでは RSE を使用しますので解放します。 SSH 接続の場合はポートの解放は必要ありません。

COBOL 環境の配下に存在する startrdodaemon を実行します。

コマンド例）$COBDIR/remotedev/startrdodaemon 5000

上記 5000 をポート番号へ指定しない場合には、デフォルトの 4075 がポート番号として指定されます。




6） ネットワーク ファイル システム（SAMBA、NFS など）を使用する際には、そのシステムを起動させる必要があります。

SAMBA 起動確認コマンド例）service smb status

SAMBA nmbd 起動コマンド例）/usr/sbin/nmbd -D

SAMBA smbd 起動コマンド例）/usr/sbin/smbd –D

また、リモートマシン共有エリアの使用権限を持つユーザーでローカルマシンからマップを行い、ローカルマシン上からリモートマシンのファイルを認識可能にする必要があります。

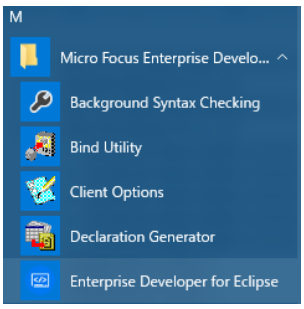
コマンド例）net use v: ¥¥tok-rhel65-64¥tarot /user:taros password

**注意**

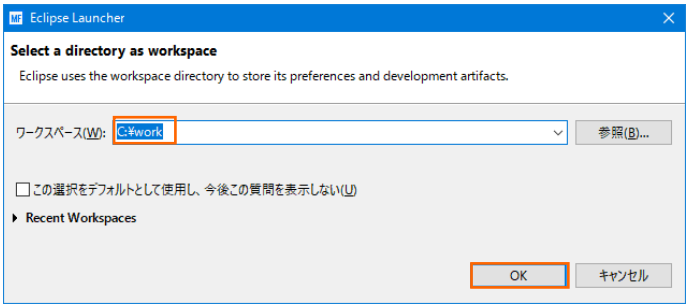
ディレクトリー配下の書き込み権限がない場合はビルド時にエラーとなります。
ls –l コマンドなどで権限を確認後、chmod コマンドなどで適切な権限設定を前もって実行してください。

## 3.3 Eclipse の起動

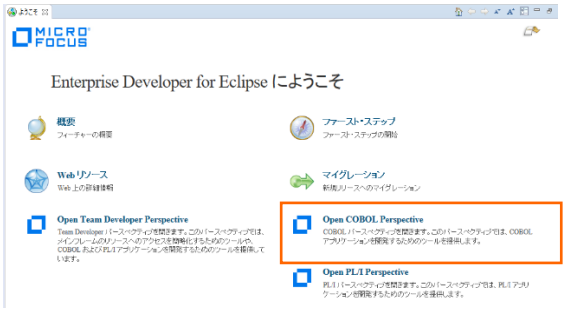
1) Micro Focus Enterprise Developer for Eclipse を起動します。



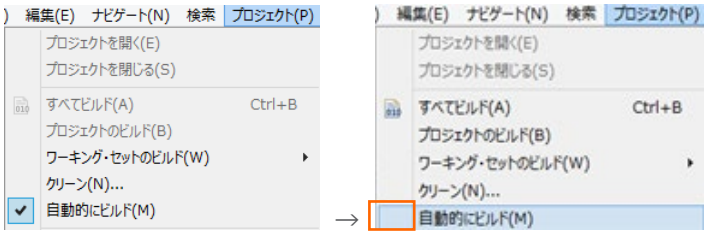

2) 前項で作成した C:¥work をワークスペースへ指定して、[OK] ボタンをクリックします。



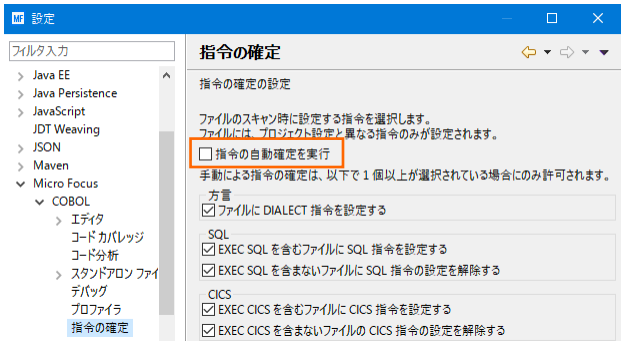

3) [ようこそ] タブが表示されますので、[Open COBOL Perspective] をクリックして、COBOL パースペクティブを開きます。



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。

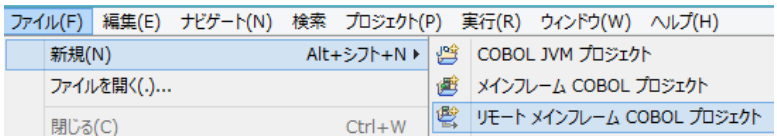



5) 既存ファイルのインポート時、自動的にコンパイル指令が指定される機能が用意されていますが、本チュートリアルではこれを解除します。 [ウィンドウ] プロダウンメニューの [設定] > [Micro Focus] > [COBOL] > [指令の確定] > [指令の自動確定を実行] チェックボックスをオフにして [OK] ボタンをクリックします。



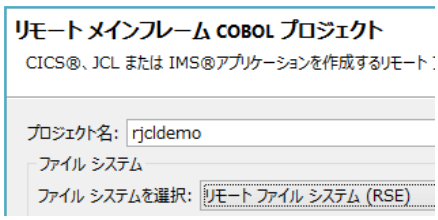

## 3.4 リモート メインフレーム COBOL プロジェクトの作成

1) 新しいプロジェクトを作成します。 [ファイル] プルダウンメニューから [新規] > [リモート メインフレーム COBOL プロジェクト] を選択します。

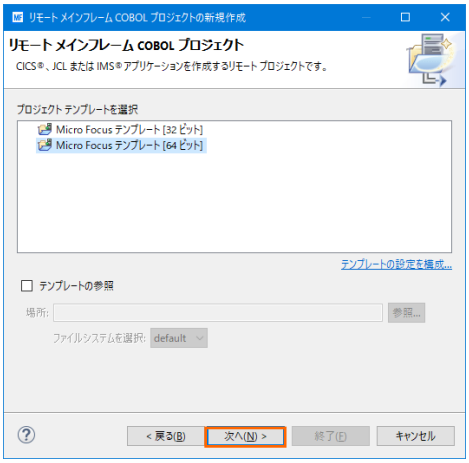



2) プロジェクト作成ウィンドウには以下のように入力します。

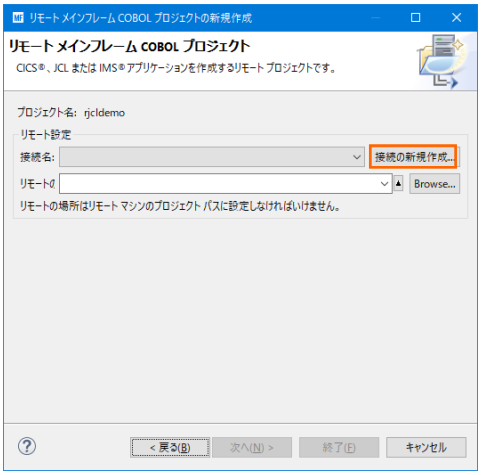
| 項目名 | 説明 |
|---|---|
| プロジェクト名 | 任意です。ここでは rjcldemo を指定します。 |
| ファイル システムを選択 | リモートマシンと接続するファイル システムを指定します。<br>ここでは [リモート ファイル システム（RSE）] を選択します。 |



3) テンプレート指定ウィンドウでは [Micro Focus テンプレート 64 ビット] を選択して [次へ] ボタンをクリックします。



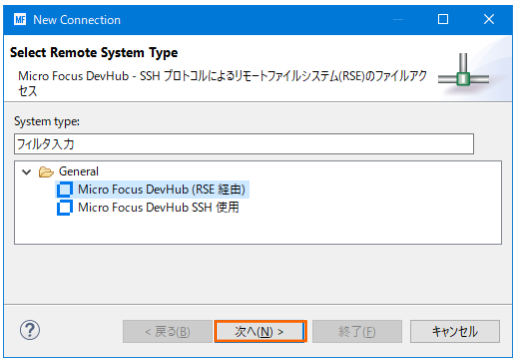

4) 新しい接続を作成するため、[接続の新規作成] ボタンをクリックします。




5) 接続タイプでは 2 種類から選択可能です。

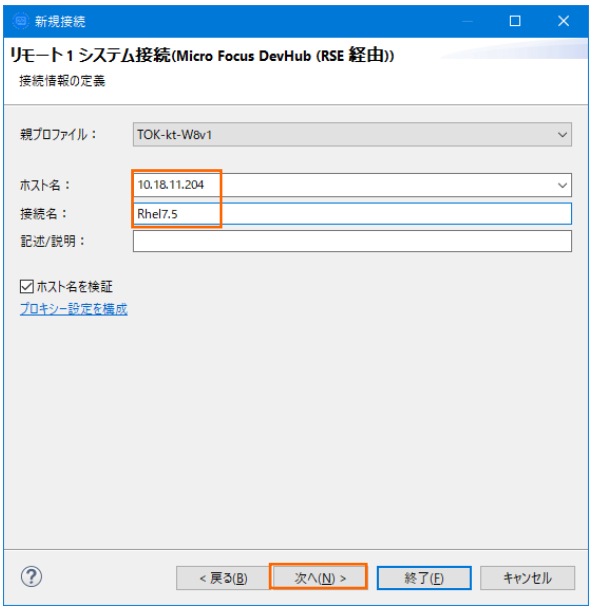
5-1) [RSE 経由] の場合

① [RSE 経由] を選択して [次へ] ボタンをクリックします。本チュートリアルでは RSE を使用しますので、こちらの手順で作成します。



② [ホスト名] へリモートマシン名または IP アドレスを指定して [次へ] ボタンをクリックします。

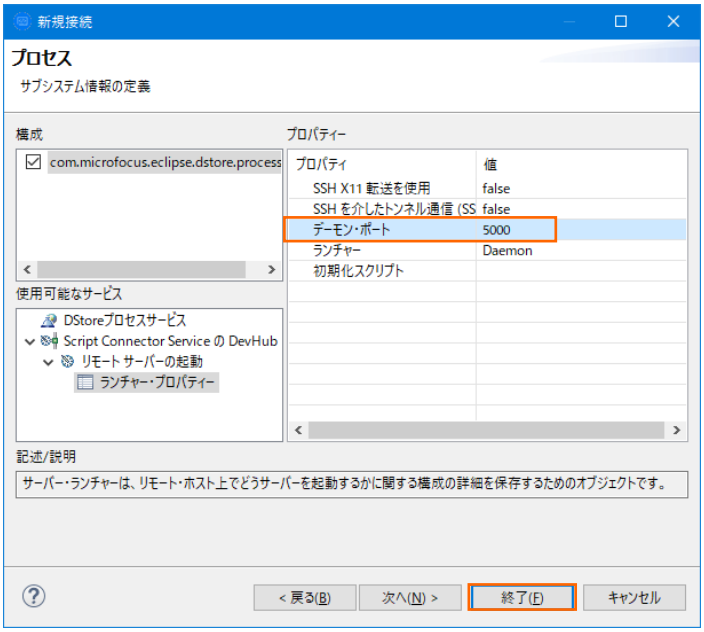
[接続名] は任意に変更可能です。




③ 下記画面の [使用可能なサービス] > [Script Connector Service の DevHub] > [リモート サーバーの起動] > [ランチャー・プロパティー] > [デーモン・ポート] 項目値を、前項で指定したリモートマシンのポート 5000 へ変更後、[終了] ボタンをクリックします。デフォルトポート番号（4075）を解放した場合は前画面で [終了] ボタンをクリックして構いません。

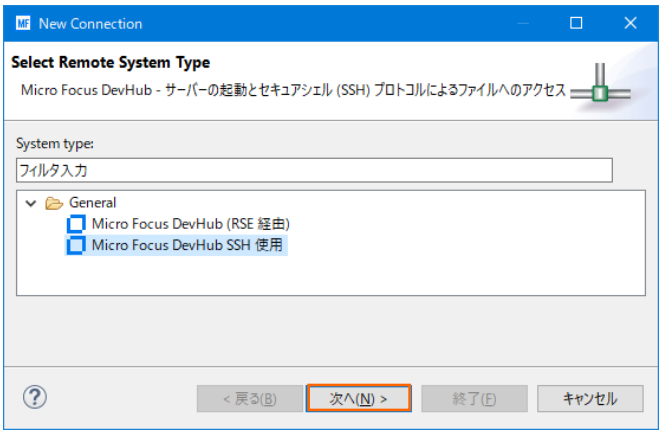
デフォルト値）4075

変更値）5000



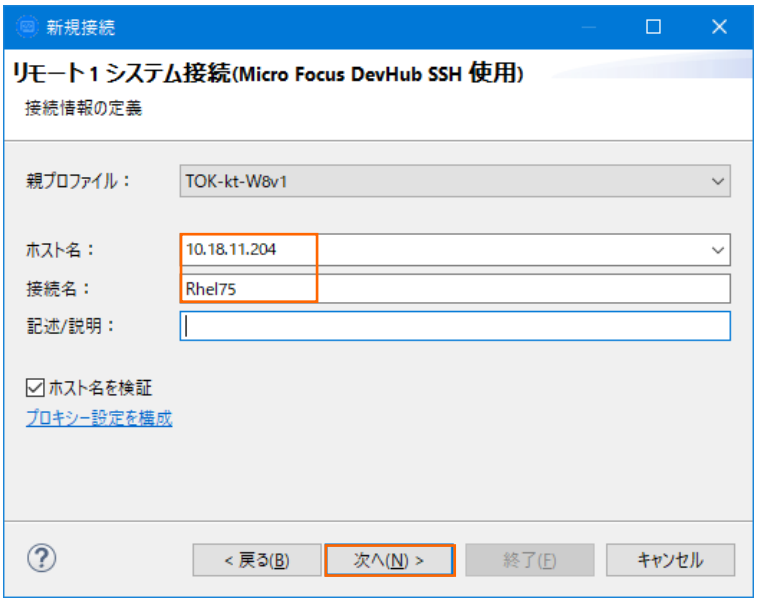
5-2)　[SSH 使用] の場合

① [SSH 使用] を選択して [次へ] ボタンをクリックします。




② [ホスト名] へリモートマシンまたは IP アドレスを指定して [次へ] ボタンをクリックします。

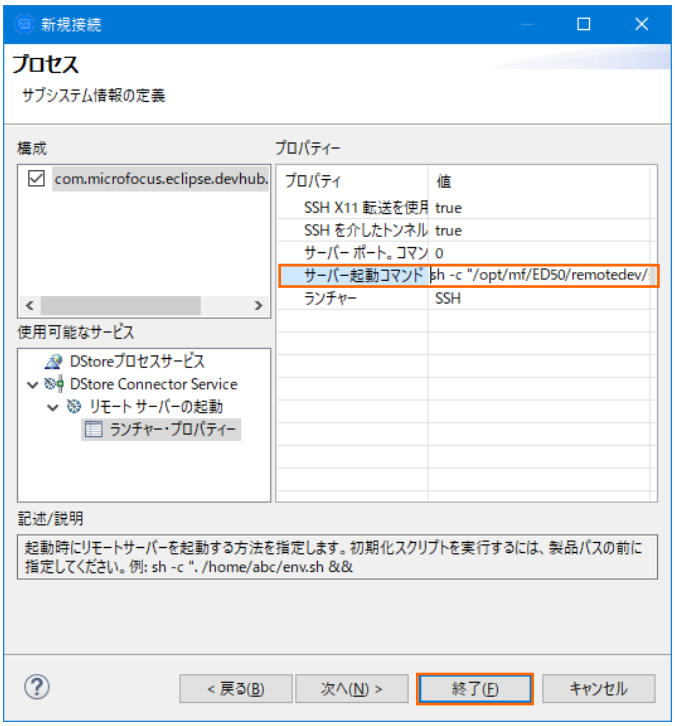
[接続名] は任意に変更可能です。



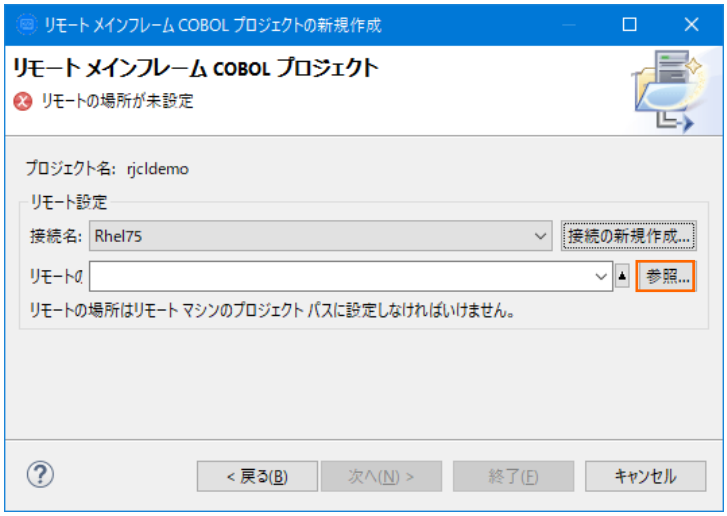

③ 下記画面の [使用可能なサービス] > [DStore Connector Service] > [ランチャー・プロパティー] > [サーバー起動コマンド] 項目値をデフォルト値からリモートマシンに実在するパスへ変更後、[終了] ボタンをクリックします。

デフォルト値）sh -c "/opt/microfocusEnterpriseDeveloper/remotedev/startrdoserver 0" &

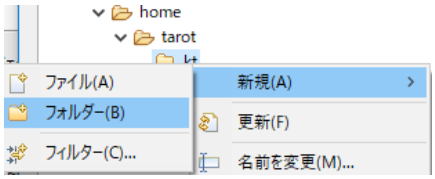
実在パス値の例）sh -c "/opt/mf/ED50/remotedev/startrdoserver 0" &



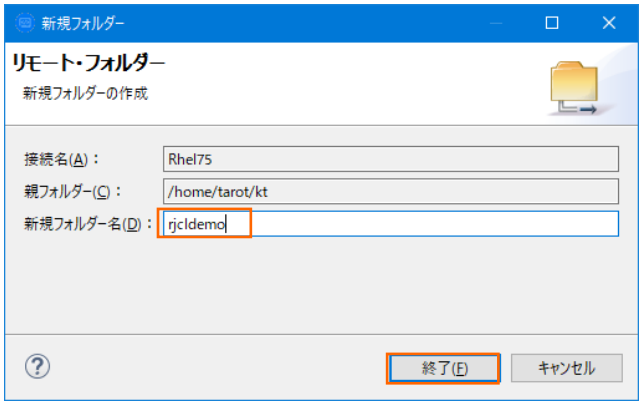

6) リモートマシンにプロジェクトを作成するロケーションを指定するため、[参照] ボタンをクリックします。リモートマシンへのログオンウィンドウが表示された場合には、権限を持つユーザー ID とパスワードを指定してアクセスしてください。



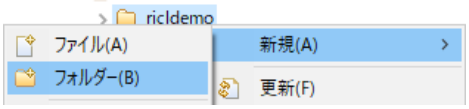

7) リモートマシンのブラウザウィンドウが表示されますので、配置したいパスへプロジェクト用のフォルダを作成します。フォルダ作成可能なロケーションを右クリックして [新規] > [フォルダー] を選択します。



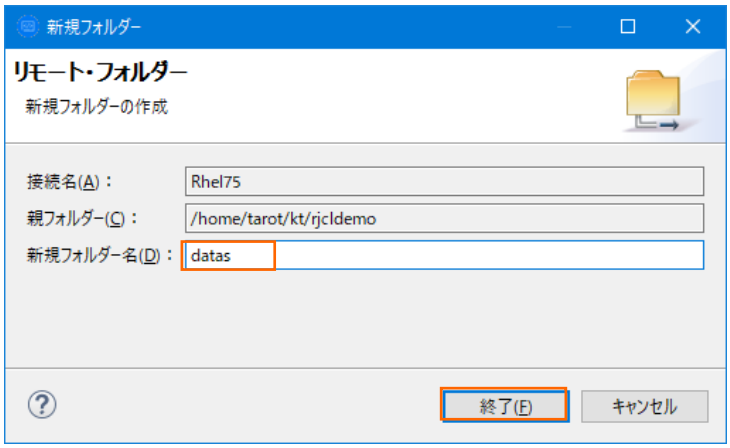
8) 新しいフォルダ名は任意ですが、ここではプロジェクト名と同様の rjcldemo を指定して [終了] ボタンをクリックします。



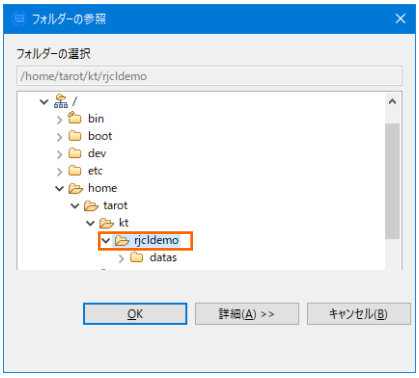

9) 同様の操作で、作成した rjcldemo フォルダ配下にデータを配置するためのフォルダを作成します。



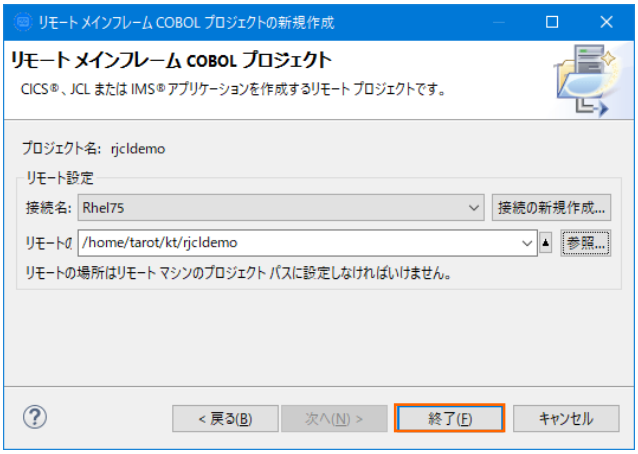

10) 新しいフォルダ名は任意ですが、ここでは datas を指定して [終了] ボタンをクリックします。



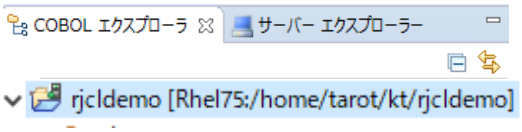

11) プロジェクトフォルダへ rjcldemo フォルダーを選択して、[OK] ボタンをクリックします。



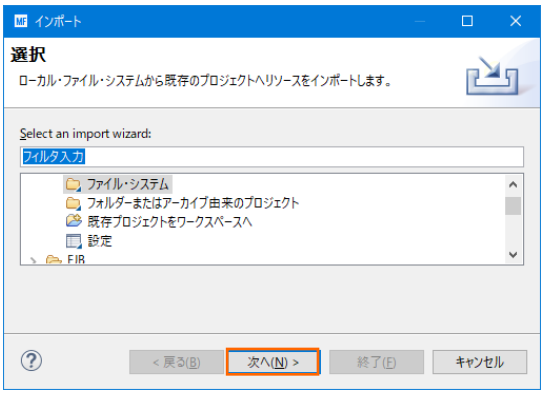
12) 指定項目を確認後、[終了] ボタンをクリックします。




13) COBOL エクスプローラーへ作成したリモートプロジェクトが表示されます。

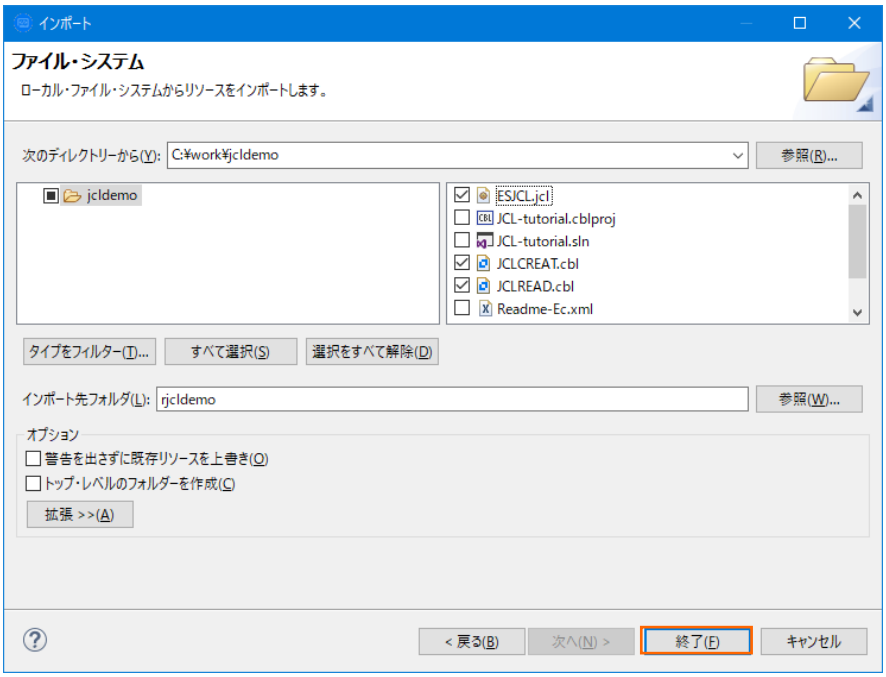



14) 用意した例題プログラム類をインポートします。 [ファイル] プルダウンメニューから [インポート] > [インポート] を選択し、インポートウィンドウにて [一般] > [ファイル・システム] を選択後 [次へ] ボタンをクリックします。



15) 前項で作成した C:¥work¥jcldemo 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、拡張子が .cbl , .jcl の合計 3 ファイルのチェックをオンにして [終了] ボタンをクリックします。

この実行により、前項で指定したリモートマシンの指定フォルダへ例題プログラムが配置されます。

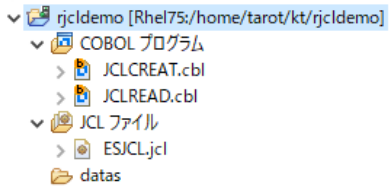



## 3.5 プロジェクトプロパティの設定

この例題はファイル操作を行う JCL と JCL から呼ばれるプログラムが含まれています。

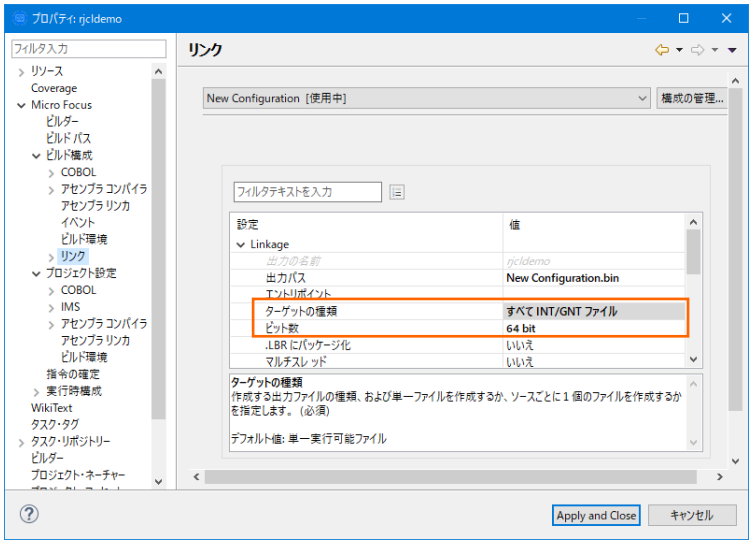
プロジェクトのプロパティを設定します。

1) インポートされた内容が COBOL エクスプローラー へ表示されていることを確認後、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。



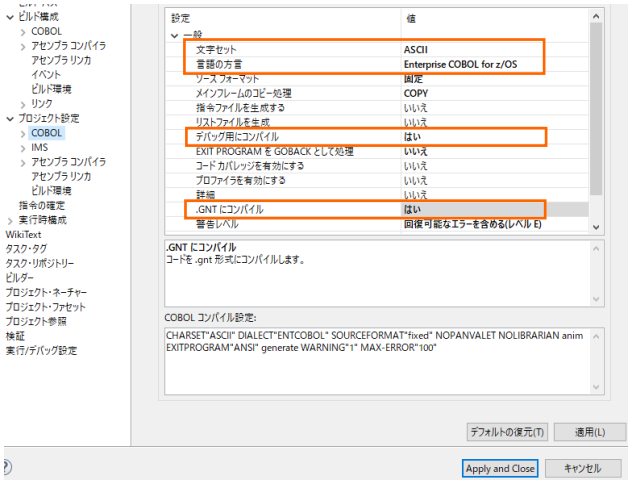
2) 左側ツリービューの [Micro Focus] > [ビルド構成] > [リンク] を選択して、下記項目を指定します。指定後は [適用] ボタンをクリックしてください。

| 項目名 | 説明 |
|---|---|
| ターゲットの種類 | 実行ファイル形式を指定。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。ここでは [64 ビット] を指定します。 |



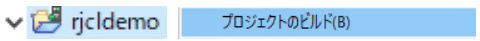

3) 左側ツリービューの [Micro Focus] > [プロジェクト設定] > [COBOL] を選択して、下記項目を指定します。指定後は [Apply and Close] ボタンをクリックしてください。

| 項目名 | 説明 |
|---|---|
| 文字集合 | EBCDIC または ASCII を指定。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。<br>ここでは [Enterprise COBOL for z/OS] を選択します。 |
| 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 'はい' を選択します。 |
| .GNT にコンパイル | 実行ファイル形式 GNT を生成するよう 'はい' を選択します。 |

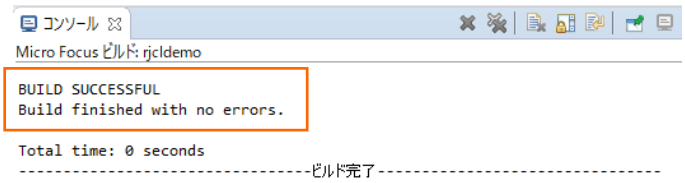


## 3.6 ビルドの実行

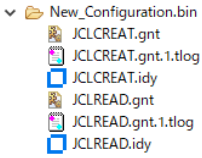
1) COBOL エクスプローラー内のプロジェクトを右クリックして [プロジェクトのビルド] を選択するとビルドが実行されます。




2) [コンソール] タブで成功を確認します。

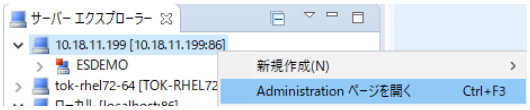



3) プロジェクトの New_Configuration.bin フォルダ配下に実行ファイルが作成されていることを確認してください。



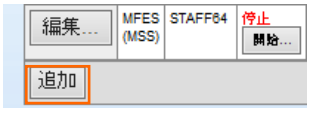

## 3.7 Enterprise Server インスタンスの設定

1) JCL を実行するためのエンジンを搭載した Enterprise Server インスタンスを作成します。[サーバー エクスプローラー] タブのリモートマシンを右クリックして [Administration ページを開く] を選択します。

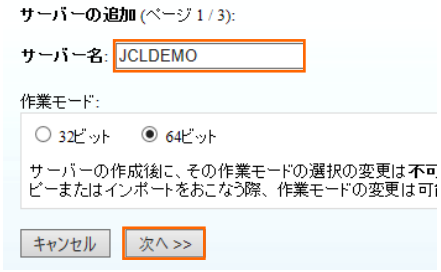



2) Administration ページが表示されない場合は、リモートマシン上から Micro Focus Directory Server を起動してください。（3.2-4 項参照）

3) 表示された画面の左下にある [追加] ボタンをクリックします。



4) 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。

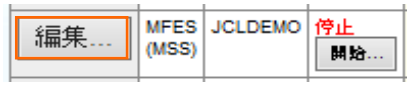



**重要**

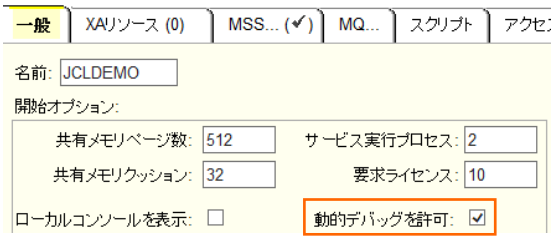
実行ファイル生成に指定した稼働ビット数 = Enterprise Server インスタンス稼働ビット数である必要があります。

5) 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 が追加されます。


6) 左にある [編集] ボタンをクリックします。




7) [サーバー] > [プロパティ] > [一般] タブの [動的デバッグを許可] チェックボックスをオンにして、[適用] ボタンをクリックします。この指定により、Eclipse からの動的デバッグが可能になります。



8) [サーバー] > [プロパティ] > [MSS] > [JES] > [一般] 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。ここではオンにします。 |
| 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。ここではオンにします。 |
| JES プログラム パス | 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配置するファイル名までのフルパスを指定します。 |
| データセットの省略時ロケーション | JCL で使用するファイルのデフォルトパスを指定します。 |
| システムプロシジャーライブラリ | プロシジャーライブラリ名を指定します。本チュートリアルでは使用しません。使用する際は名前を指定してカタログする必要があります。 |


9) [サーバー] > [プロパティ] > [MSS] > [JES] > [イニシエータ] タブで、[追加] ボタンをクリックし、イニシエータ定義を作成します。入力後は [追加] ボタンをクリックします。

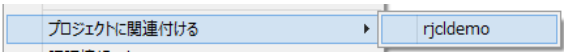
| 項目名 | 説明 |
|---|---|
| 名称 | 任意。ここでは INIT1 を指定します。 |
| クラス | 実行させるクラスを指定します。ここではクラス A~G までを指定します。 |

10) 画面左上の [Home] をクリックして一覧画面に戻ります。


## 3.8 Enterprise Server インスタンスの開始と確認

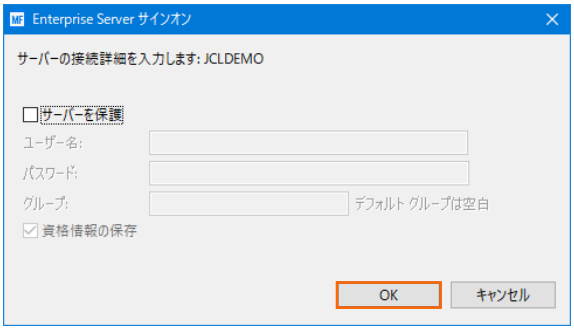
1) サーバーエクスプローラー内のリモート環境に JCLDEMO インスタンスが表示されていることを確認します。表示されていない場合はリモート接続名を右クリックし、[更新] を選択してリフレッシュしてください。

2) サーバーエクスプローラー内の 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] > [rjcldemo] を選択します。これにより rjcldemo プロジェクトから実行される JCL は JCLDEMO インスタンスで処理されることになります。

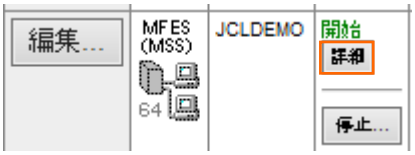



3) JCLDEMO インスタンスを右クリックして [開始] を選択します。


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。

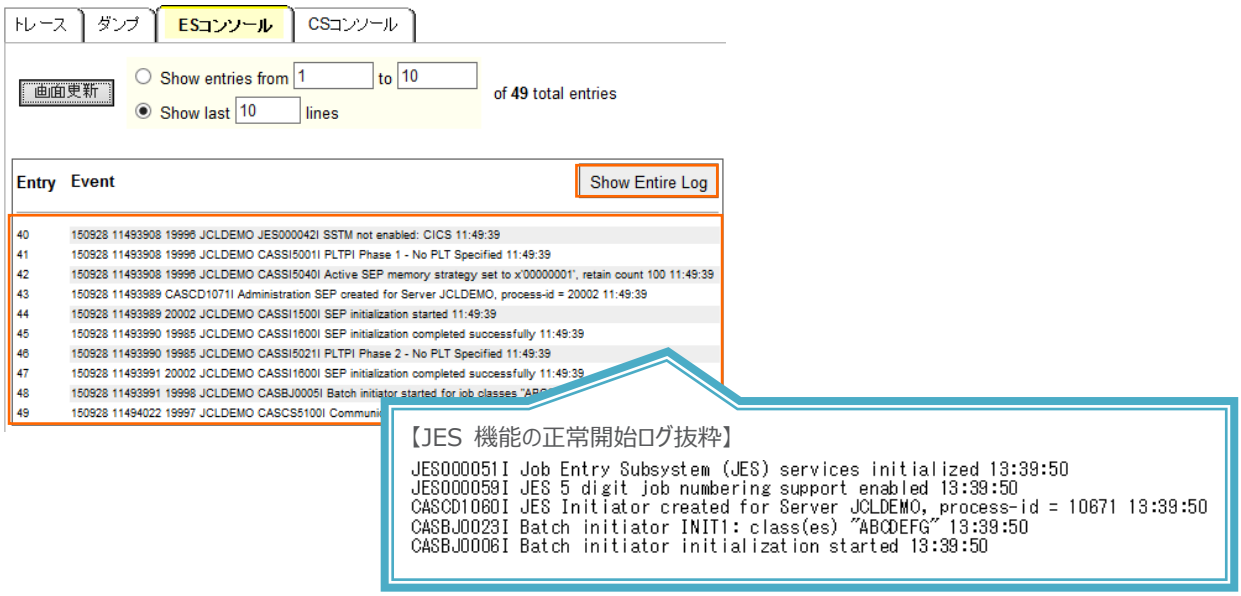



5) Enterprise Server Administration 画面へ移動して開始状態であることを確認後、[詳細] ボタンをクリックします。



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。




**注意**

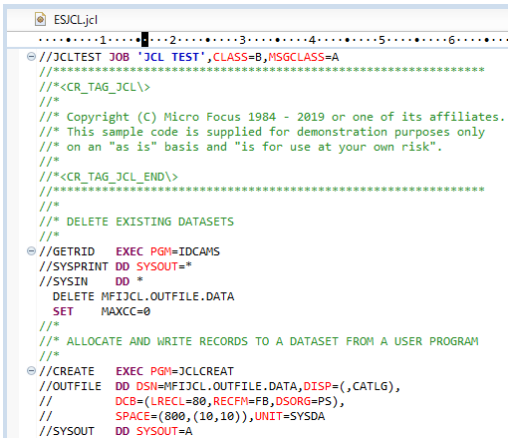
いくつかのサービス開始が失敗してもインスタンスは開始されますので、ログ内容を必ず確認してください。

## 3.9 JCL の実行とデバッグ

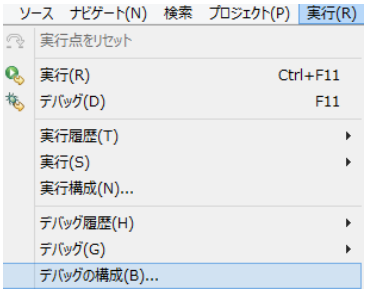
ローカルマシンの Eclipse から、リモートマシンに存在する COBOL 実行ファイルと JCLDEMO インスタンスを使用して、JCL の実行とデバッグを行います。

1) COBOL エクスプローラー内に存在する rjcldemo プロジェクトの ESJCL.jcl をダブルクリックして内容を表示します。IDCAMS などのユーティリティを使用してファイルを操作したのち、後続ステップでプログラムを実行していることがわかります。

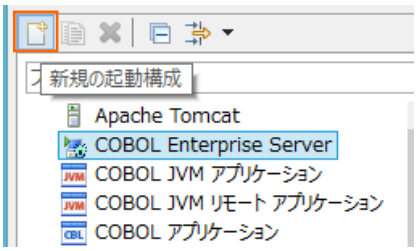
この JCL から呼ばれるプログラムをデバッグ実行します。



2) [実行] プロダウンメニューから [デバッグの構成] を選択します。

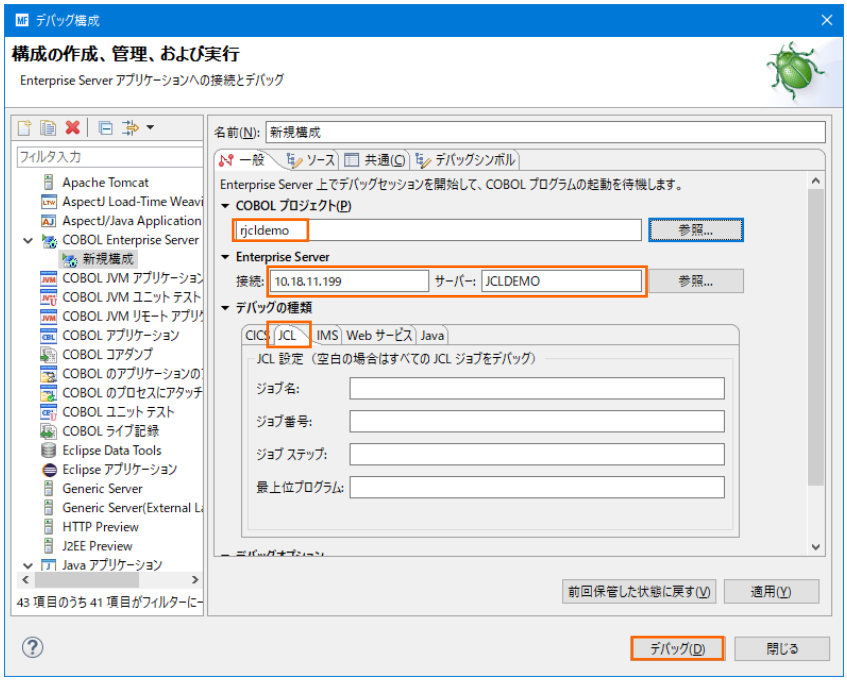



3) 左側のメニューから [COBOL Enterprise Server] を選択して、左上の [新規の起動構成] アイコンをクリックします。

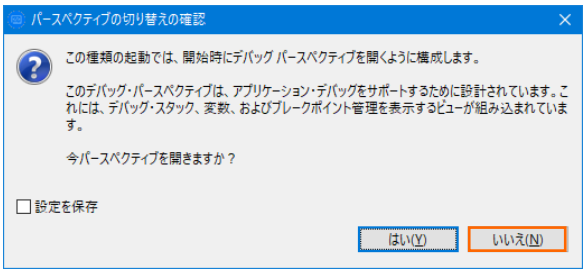



4) [COBOL プロジェクト] へ対象となる rjcldemo プロジェクトを入力し、[Enterprise Server] へ実行させるリモート環境に存在する JCLDEMO インスタンスを指定します。

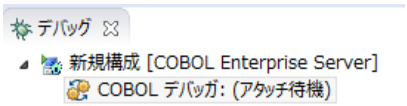
[デバッグの種類] は「JCL」タブを選択した状態で、[デバッグ] ボタンをクリックします。



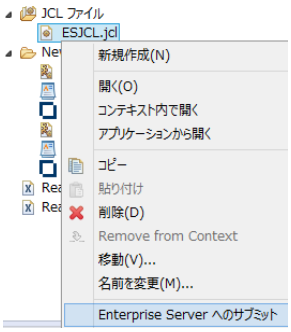
5) COBOL エクスプローラーから JCL を実行するため、パースペクティブの切り替え確認ウィンドウでは [いいえ] ボタンをクリックします。



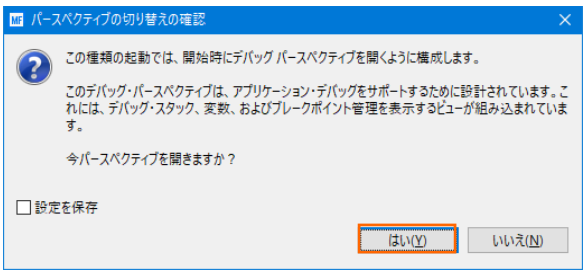

6) デバッグタブで [アタッチ待機] 状態になったことを確認します。



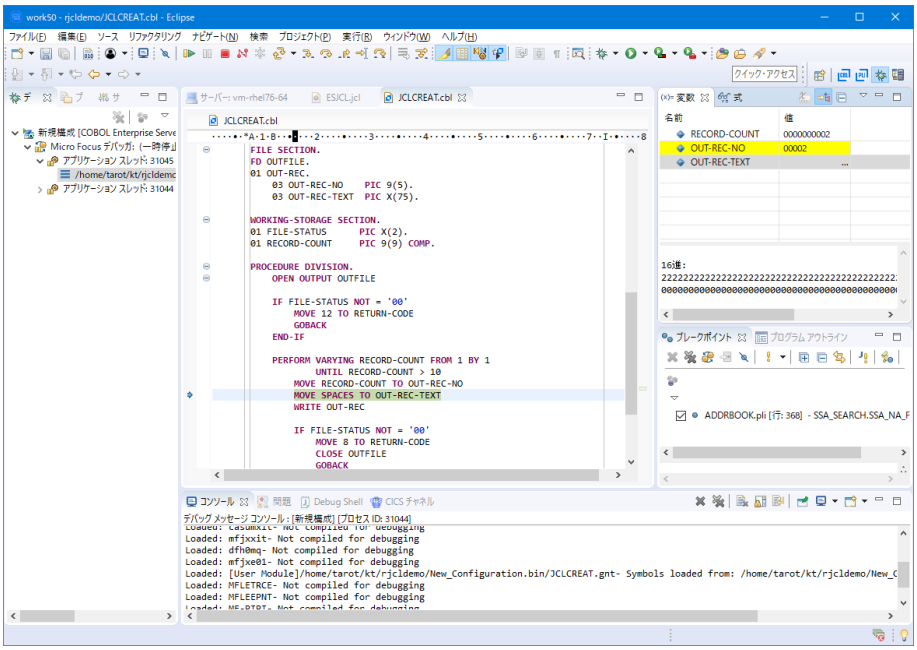

7) COBOL エクスプローラー内の ESJCL.jcl を右クリックし、[Enterprise Server へのサブミット] を選択して、JCL を実行します。




8) 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。



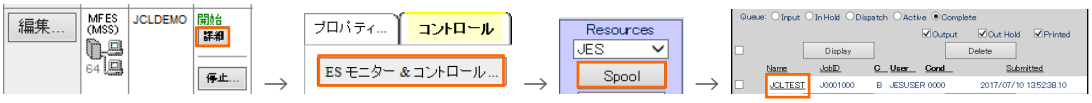
9) プログラムのステップ実行が可能になります。[F5] キーでステップ実行が可能で、変数タブでは使用している変数の値が確認できます。

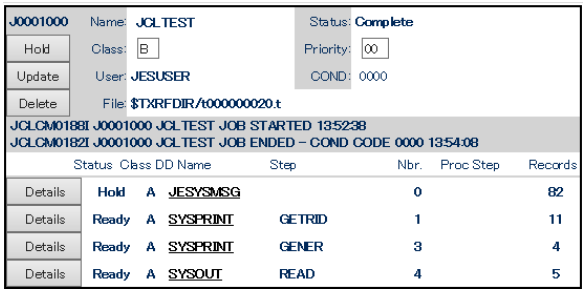



10) 再開ボタン（緑三角ボタン）を２回クリックして全てを実行させたのちデバッグを終了させます。


11) COBOL パースペクティブへ戻ります。画面右上の COBOL アイコンをクリックします。


12) Enterprise Server Administration 画面へ移動してスプールファイルを確認します。



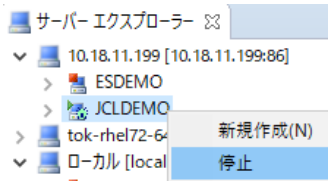
13) [JESYSMSG] をクリックしてジョブログ内容を表示し、正常終了を確認します。

```
---> 13:38:58 JCLCM0191I STEP ENDED     CREATE - COND CODE 0000

     13:38:58 JCLCM0190I STEP STARTED   GENER
     13:38:59 JCLCM0199I Program MFJGENER is COBOL       ASCII  Big-Endian    NOAMODE.
     MFE2015.S0928.S133543.J01002.D00004.SYSPRINT                        SYSPRINT
      /home/tarot/keit/rCBLJCL/New_Con*3543.J01002.D00004.SYSPRINT.DAT  SPOOLED
     MFE2015.S0928.S133543.J01002.D00005.SYSUT1                          SYSUT1
      /home/tarot/keit/rCBLJCL/New_Con*133543.J01002.D00005.SYSUT1.DAT  DELETED
     MFE2015.S0928.S133543.J01002.ANDAND.JTEMP                           SYSUT2
      /home/tarot/keit/rCBLJCL/New_Con*S133543.J01002.ANDAND.JTEMP.DAT  PASSED
---> 13:38:59 JCLCM0191I STEP ENDED     GENER - COND CODE 0000

     13:38:59 JCLCM0190I STEP STARTED   READ
     13:38:59 JCLCM0220I Job restart will not be permitted at this step.
     13:38:59 JCLCM0199I Program JCLREAD  is COBOL VSC2  ASCII  Big-Endian    NOAMODE.
     MFE2015.S0928.S133543.J01002.ANDAND.JTEMP                           INFILE
      /home/tarot/keit/rCBLJCL/New_Con*S133543.J01002.ANDAND.JTEMP.DAT  DELETED
     MFE2015.S0928.S133543.J01002.D00006.SYSOUT                          SYSOUT
      /home/tarot/keit/rCBLJCL/New_Con*133543.J01002.D00006.SYSOUT.DAT  SPOOLED
---> 13:39:08 JCLCM0191I STEP ENDED     READ - COND CODE 0000

---> 13:39:08 JCLCM0182I JOB  ENDED    - COND CODE 0000
```

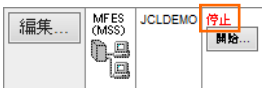
14) 同様に [SYSOUT] の内容などを確認します。

## 3.10 Enterprise Server インスタンスの停止

1) 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。



3) 必要であれば、リモートマシンで使用した Samba の終了や、使用したポートの遮断をルートユーザーで行います。

Samba nmb 終了コマンド例）service nmb stop

Samba smb 終了コマンド例）service smb stop

ポート遮断コマンド例）$COBDIR/remotedev/stoprdodaemon 5000

4) 必要であれば、Micro Focus Directory Server を停止します。

Administration ページの左側メニューから [シャットダウン] をクリックして、[OK] ボタンをクリックします。


## WHAT'S NEXT

- 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。