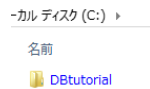


3.1 チュートリアル準備

例題プログラムに関連する資源を用意します。

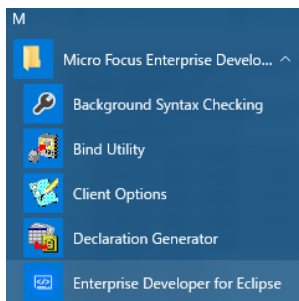
- 1) 使用する例題プログラムは、キットに添付されている DBtutorial.zip に圧縮されています。これを C:¥ 直下に解凍します。



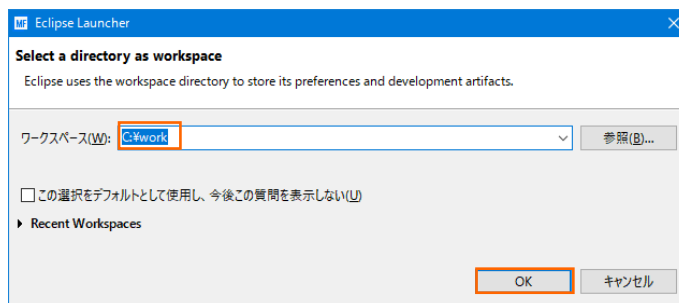
- 2) Eclipse のワークスペースで使用する work フォルダを C:¥ 直下に作成します。

3.2 Eclipse の起動

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



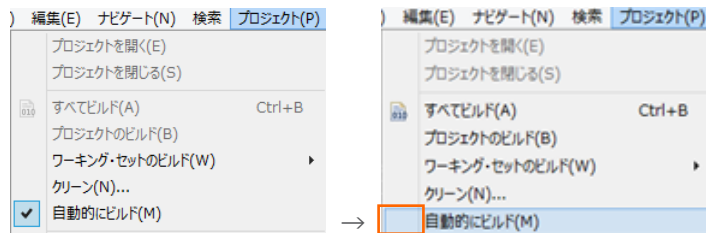
- 2) 前項で作成した C:¥work をワークスペースへ指定して、[OK] ボタンをクリックします。



- 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。

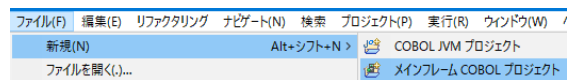


- 4) パースペクティブ表示後、[プロジェクト] プルダウンメニューの [自動的にビルド] を選択して、これをオフにします。

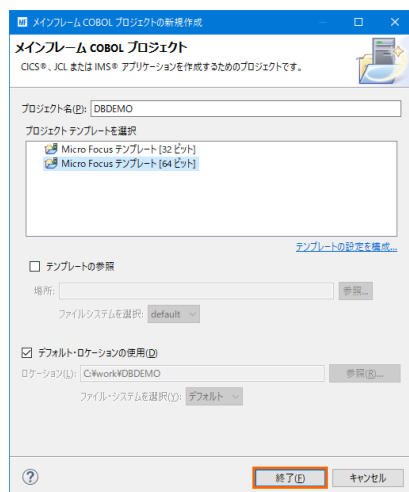


3.3 メインフレーム COBOL プロジェクトの作成

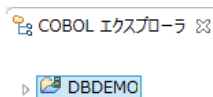
- 1) 用意した例題ソースをインポートします。[ファイル] プルダウンメニューから [新規] > [メインフレーム COBOL プロジェクト] を選択します。



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

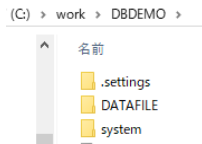


- 3) COBOL エクスプローラーへ作成したプロジェクトが表示されます。

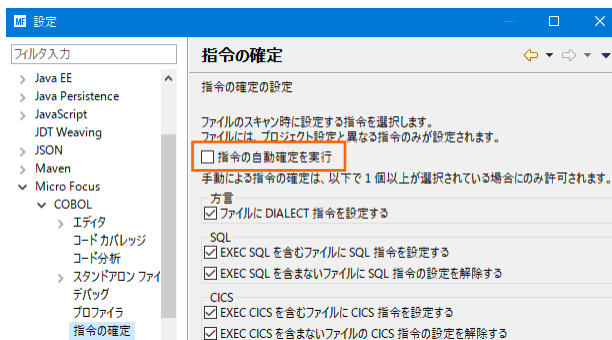


- 4) プロジェクトを作成したことにより C:\work\DBDEMO フォルダが作成されています。このフォルダ配下に JES 機能で使用するフォルダを Windows エクスプローラーからあらかじめ用意しておきます。

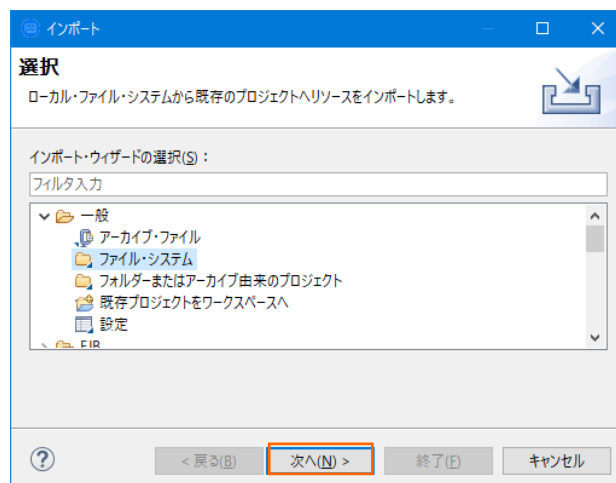
カタログファイルやスプールファイルを配置するため DATAFILE フォルダを C:\work\DBDEMO 配下へ、実行時にログなどを格納する system フォルダを C:\work\DBDEMO 配下へ作成します。



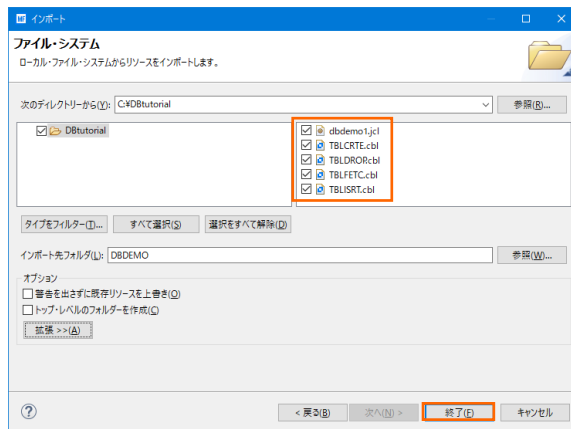
- 5) 既存ファイルのインポート時、自動的にコンパイル指令が指定される機能が用意されていますが、本チュートリアルではこれを解除します。[ウィンドウ] プロダクションメニューの [設定] > [Micro Focus] > [COBOL] > [指令の確定] > [指令の自動確定を実行] チェックボックスをオフにして [適用して閉じる] ボタンをクリックします。



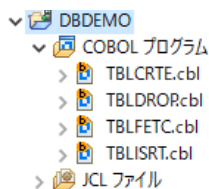
- 6) 用意した例題プログラム類をインポートします。DBDEMO プロジェクトを右クリックして [インポート] > [インポート] を選択し、インポートウィンドウにて [一般] > [ファイル・システム] を選択後 [次へ] ボタンをクリックします。



- 7) C:\¥DBtutorials 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、全ての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、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 DBDEMO 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します。

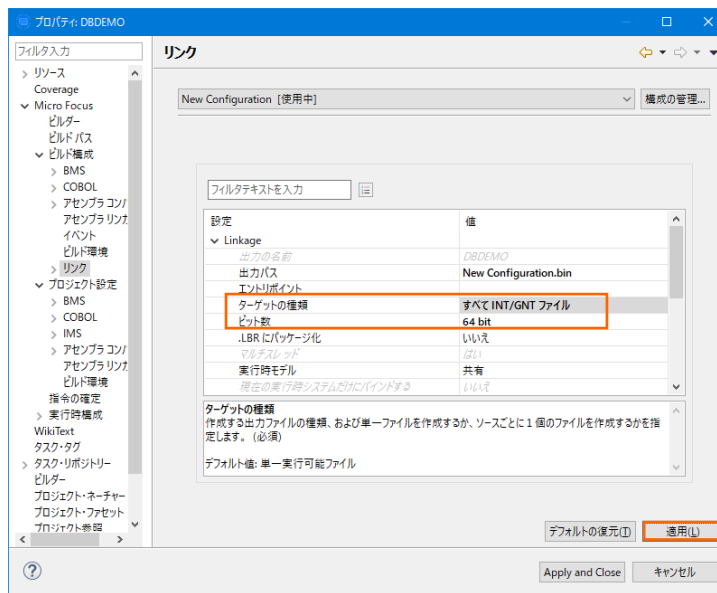


3.4 プロジェクトプロパティの設定

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。埋め込み SQL 付き COBOL ソースは、予め Micro Focus 形式の COBOL ソースにプリコンパイルしてから使用することも可能ですが、製品にはプリプロセッサ機能からプリコンパイラを呼び出して内部的にプリコンパイルする機能があります。これを使用することにより、オリジナルソースイメージのままデバッグが可能となり管理も容易になります。ここでは後者の方法を紹介します。

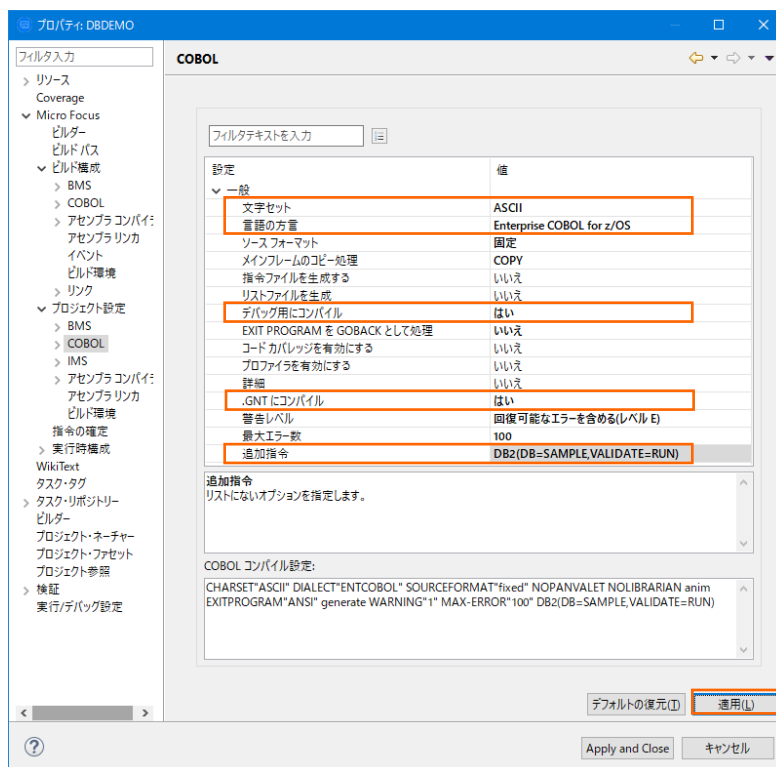
- 1) COBOL エクスプローラー内の DBDEMO プロジェクトを右クリックして [プロパティ] を選択します。
- 2) 左側ツリービューの [Micro Focus] > [ビルド構成] > [リンク] を選択して、下記項目を指定します。指定後は [適用] ボタンをクリックしてください。

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。

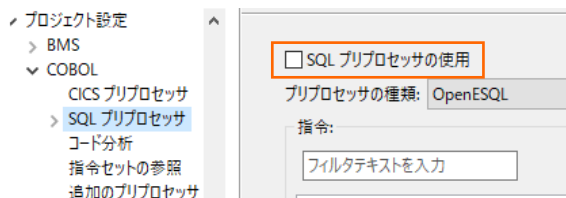


- 3) 左側ツリービューの [Micro Focus] > [プロジェクト設定] > [COBOL] を選択して、下記項目を指定します。指定後は [適用] ボタンをクリックしてください。

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	<p>使用するデータベース製品に合わせ、[追加指令] 欄へ埋め込み SQL 対応のプリプロセッサの設定を追加します。</p> <p>【Oracle 使用時：COBSQL プリプロセッサを使用します。】 P(COBSQL) ENDP</p> <table border="1"> <tr> <td>追加指令</td> <td>P(COBSQL) ENDP</td> </tr> </table> <p>【Db2 使用時：ECM プリプロセッサを使用します。】 DB2(DB=DEMODB,VALIDATE=RUN)</p> <table border="1"> <tr> <td>追加指令</td> <td>DB2(DB=DEMODB,VALIDATE=RUN)</td> </tr> </table> <p>【SQL Server 使用時：OpenESQL を使用します。】 SQL(DBMAN=ODBC,BEHAVIOR=JCL,TARGETDB=MSSQLSERVER)</p> <table border="1"> <tr> <td>追加指令</td> <td>SQL(DBMAN=ODBC,BEHAVIOR=JCL,TARGETDB=MSSQLSERVER)</td> </tr> </table>	追加指令	P(COBSQL) ENDP	追加指令	DB2(DB=DEMODB,VALIDATE=RUN)	追加指令	SQL(DBMAN=ODBC,BEHAVIOR=JCL,TARGETDB=MSSQLSERVER)
追加指令	P(COBSQL) ENDP						
追加指令	DB2(DB=DEMODB,VALIDATE=RUN)						
追加指令	SQL(DBMAN=ODBC,BEHAVIOR=JCL,TARGETDB=MSSQLSERVER)						

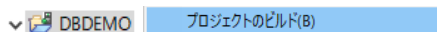


- 4) 前項で SQL 対応のプリプロセッサを使用する追加指令を指定しているため、左側ツリービューの [Micro Focus] > [プロジェクト設定] > [COBOL] > [SQL プリプロセッサ] を選択して、[SQL プリプロセッサの使用] のチェックをオフにします。最後に [適用して閉じる] ボタンをクリックしてください。

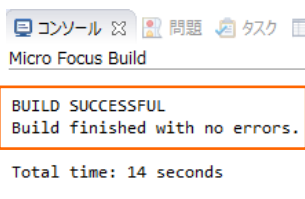


3.5 ビルドの実行

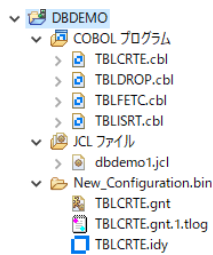
- 1) COBOL エクスプローラー内のプロジェクトを右クリックして [プロジェクトのビルド] を選択するとビルドが実行されます。



- 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 フォルダ配下にプログラム本数分の実行ファイル (.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 XA スイッチモジュールの生成

実行するプログラムは XA スイッチモジュール経由でデータベースと接続するため、使用するデータベース製品に合わせた XA スイッチモジュールを作成します。本チュートリアルでは JCL バッチからの使用方法として紹介していますが、CICS や IMS プログラムからのデータベース連携を XA リソース方式で行う場合も同様の手順となります。

- 1) ビルドを行うため、製品フォルダに含まれている次のフォルダを書き込み権限があるフォルダ配下へ Windows エクスプローラーを使用してコピーします。本チュートリアルでは C:¥ 直下へコピーします。

【理由 1】 Oracle のプリコンパイラはパスに英数字とアンダースコア以外は許容しない。

【理由 2】 製品関連フォルダの書き込み権限によるトラブルを避ける。

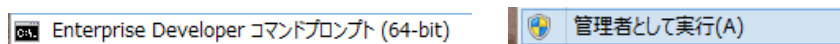
【コピー元フォルダ例】

C:¥Program Files (x86)¥Micro Focus¥Enterprise Developer¥src¥enterpriseserver¥xa

【コピー先フォルダの例】 C:¥xa



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



- 3) コピーした C:¥xa パスへ移動します。

```
C:¥Users¥tarot¥Documents>cd c:¥xa
c:¥xa>
```


- 4) DLL を生成するために、Windows SDK が必要になります。リンクエラーを避けるために、これがインストールされているかご確認ください。また、複数の SDK や Microsoft Build Tools がインストールされている場合には、COBOL 環境が使用するバージョンを指定することもできます。

使用可能な SDK と Microsoft Build Tools の確認コマンド)

cblms -L

```
c:\>cblms -L
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Windows SDK
Id バージョン 場所
0] 10.0.18362.0 c:\Program Files (x86)\Windows Kits\10
1] 10.0.19041.0 c:\Program Files (x86)\Windows Kits\10
2] 10.0.20348.0 c:\Program Files (x86)\Windows Kits\10

Microsoft Build Tools
Id バージョン 場所
0] 14.29.30133 c:\Program Files (x86)\Microsoft Visual Studio\2019\Professional
```

最新バージョンに設定するコマンド)

cblms -U

```
c:\>cblms -U
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Windows SDK
場所 = c:\Program Files (x86)\Windows Kits\10
バージョン = 10.0.20348.0

Microsoft Build Tools
場所 = c:\Program Files (x86)\Microsoft Visual Studio\2019\Professional
バージョン = 14.29.30133
```

特定のバージョンを指定するコマンド例)

cblms -US:1

```
c:\>cblms -US:1
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Windows SDK
場所 = c:\Program Files (x86)\Windows Kits\10
バージョン = 10.0.19041.0

c:\>cblms -UB:0
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Microsoft Build Tools
場所 = c:\Program Files (x86)\Microsoft Visual Studio\2019\Professional
バージョン = 14.29.30133
```

COBOL 環境が使用するバージョンを表示するコマンド)

cblms -Q

```
c:\%xa>cblms -Q
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Windows SDK
場所 = c:\Program Files (x86)\Windows Kits\10
バージョン = 10.0.19041.0

Microsoft Build Tools
場所 = c:\Program Files (x86)\Microsoft Visual Studio\2019\Professional
バージョン = 14.29.30133
```

利用可能なオプションを表示するコマンド)

cblms -H

- 5) 使用するデータベース製品に合わせた XA スイッチモジュールを build コマンドで作成します。正常終了すると C:\%xa 配下に対象データベースの XA スイッチモジュールが作成されます。

① Oracle

コマンド) build ora19 (対象バージョンにより ora18)

```
c:\%xa>build ora19
Building 64-bit switch module...
Micro Focus COBOL
Version 7.0 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

* Cobsql Integrated Preprocessor
* CSQL-I-018: Oracle プリコンパイラトランスレータを起動します。
* CSQL-I-020: Oracle プリコンパイラの出力を処理中。
* CSQL-I-001: COBSQL: チェッカへの引き渡しを完了しました。
* 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 ESORAXA_D
* Data:      34752      Code:      78794      Literals:      4000
Micro Focus COBOL - CBLINK utility
Version 7.0.0.75 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.
```

ディスク (C:) > xa

名前

- ESORAXA.dll
- ESORAXA_D.dll

ファイル名.dll は静的登録用、ファイル名_D.dll は動的登録用です。

② Db2

コマンド) build db2

```
c:\%xa>build db2
Building 64-bit switch module...
Micro Focus COBOL
Version 7.0 (C) Copyright 1984-2021 Micro Focus or one of 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 ESDB2XA
* Data:      40416      Code:      69426      Literals:      4144
Micro Focus COBOL - CBLINK utility
Version 7.0.0.75 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.
```

ディスク (C:) > xa

名前

- ESDB2XA.DLL
- ESDB2XA_S.DLL

ファイル名_S.DLL は静的登録用、ファイル名.DLL は動的登録用です。

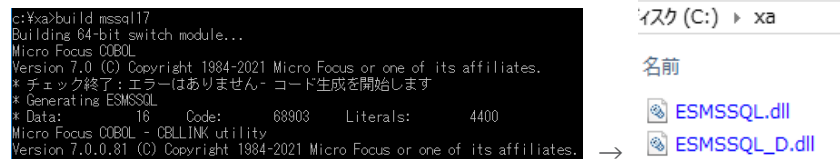
③ SQL Server

A) ビルドの実行

SQL Server に使用する ODBC 17.3 ドライバー以降、Microsoft 社は XA プロトコルを変更したため、ODBC 17.3 以降を使用する場合は、mssql17 ビルドオプションを使用します。また、ご利用の Windows に合わせた SDK をインストールしておいてください。

本チュートリアルでは ODBC 17.8 を利用するため mssql17 を使用してビルドします。

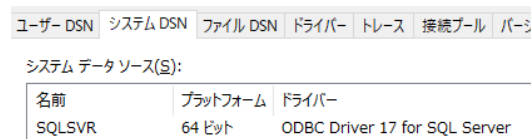
コマンド) build mssql17



ファイル名.dll は静的登録用、ファイル名_D.dll は動的登録用です。

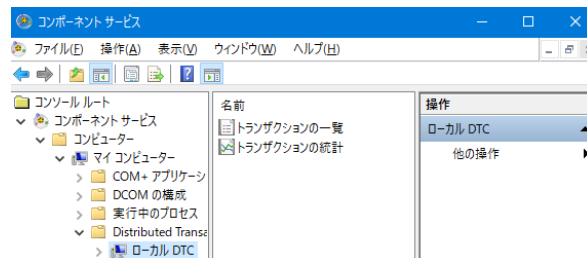
B) ODBC の追加

使用ビット数に合わせた ODBC データソースを Windows の [コントロールパネル] > [管理ツール] > [ODBC データソース] から追加します。ここで指定する ODBC データソースの名前が Enterprise Server インスタンスへ登録する XA リソース定義の OPEN 文字列で使用する DSN 名となります。

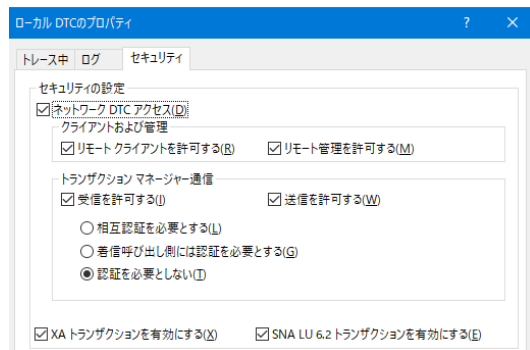


C) XA トランザクションの有効化

Windows の [コントロールパネル] > [管理ツール] > [コンポーネントサービス] > [コンピューター] > [マイ コンピュータ] > [Distributed Transaction Coordinator] > [ローカル DTC] を展開します。



[ローカル DTC] を右クリックして [プロパティ] を選択し、[セキュリティ] タブへ移動します。[XA トランザクションを有効にする] のチェックがオンであることを確認、もしくはオンにして [OK] ボタンをクリックします。



XA スイッチモジュールのビルド詳細に関しては製品マニュアルをご参照ください。

3.7 文字エンコーディングの設定

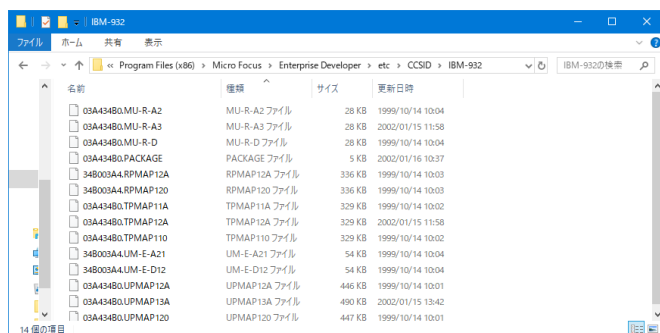
Enterprise Server インスタンスを運用、管理する Enterprise Server Common Web Administration (以降 ESCWA) 機能では、スプールやデータ内容などに含まれる日本語を正しく表示させるために、事前に文字セットを所定のフォルダへ展開します。製品マニュアルの「リファレンス > コードセットの変換 > CCSID 変換テーブルのインストール > CCSID 変換テーブルをインストールするには」を参照しながら進めてください。

- 1) CCSID 変換テーブルをインストールします。

製品マニュアルにリンクされている下記の IBM CCSID 変換テーブルを、Web ブラウザから任意のフォルダへダウンロードします。アドレスは変更される可能性がありますので、製品マニュアルにてご確認ください。

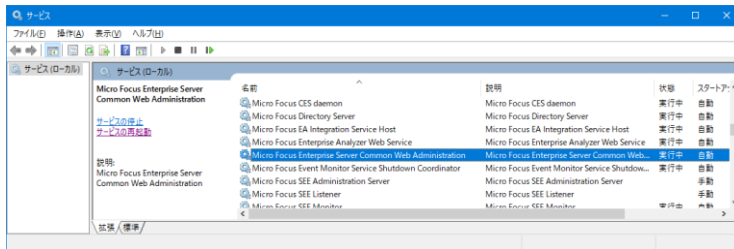
<http://www.microfocus.com/docs/links.asp?vc=cdctables>

- 2) 製品インストールフォルダ配下の etc フォルダに CCSID フォルダがない場合はこれを作成します。
例) C:\Program Files (x86)\Micro Focus\Enterprise Developer\etc\CCSID
- 3) ダウンロードファイルに含まれている Package2.zip を展開します。
- 4) 展開した Package2 フォルダに含まれる IBM-932.zip を展開します。
- 5) 展開した IBM-932 フォルダを切り取り、作成した CCSID フォルダ配下へ貼り付け、14 ファイルが含まれていることを確認します。



詳細については、製品マニュアルの「デプロイ > 構成および管理 > Enterprise Server の構成および管理 > Enterprise Server Common Web Administration > [Native] > [Directory Servers] > リージョンとサーバー > リージョン > エンタープライズ サーバー リージョンの文字エンコーディングのサポート」をご参照ください。

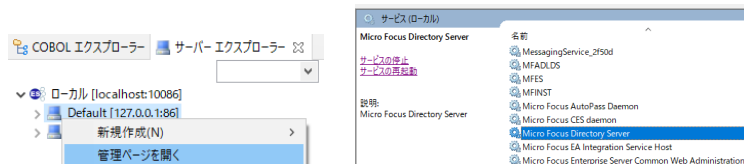
- 6) Windows サービスとして起動している Micro Focus Enterprise Server Common Web Administration を再起動し、インストールした CCSID をロードさせます。



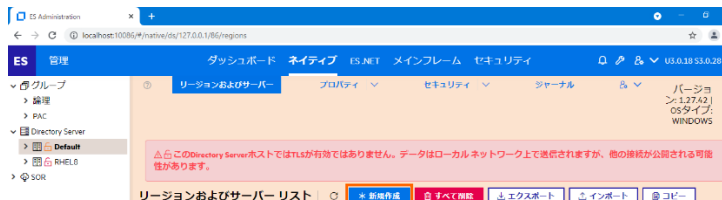
3.8 Enterprise Server インスタンスの設定

Enterprise Server インスタンスには JES をエミュレーションする機能が搭載されており、この開発用インスタンスを使用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います。本番環境には実行製品である 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) 実行する開発用 Enterprise Server インスタンスを作成します。Eclipse の [サーバー エクスプローラー] タブの [ローカル] > [Default] を右クリックして [管理ページを開く] を選択します。Default に登録されているインスタンスが表示エラーになる場合は、Windows の Micro Focus Directory Server サービスが開始されているか確認し、停止している場合は開始してください。



- 2) ブラウザが立ち上がり ESCWA が表示されます。画面の中央にある [新規作成] ボタンをクリックします。



- 3) [リージョンの新規作成] 項目の [名前]、[説明] は任意ですが、ここでは名前に DBDEMO、説明に DB チュートリアル用と入力します。Eclipse の実行可能ファイルは 64 ビットを指定してコンパイルしたため、稼働させる Enterprise Server インスタンスも同様に [64ビット作業モード] ヘチェックを入れます。これにより警告が表示されますが無視して先に進んでください。[MSS 有効] にチェックが入っていることを確認し、[TN3270 リスナーの作成] のチェックを外して [保存] ボタンをクリックします。

リージョンの新規作成

名前
DBDEMO

説明
DB チュートリアル用

▲ Directory Serverとリージョンの作業モードが一致しません。プラットフォームによっては、起動時に不具合が発生する可能性があります。

64ビット作業モード

MSS有効

TN3270リスナーの作成

TN3270リスナーポート 0

・入力必須の項目です

保存 戻る

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) 64 ビットアプリケーション稼働用の DBDEMO インスタンスが作成され、一覧に表示されます。

リージョンおよびサーバー リスト | * 新規作成 すべて削除 エクスポート

名前 (DBDEMO)	説明	PAC	エンドポイント	タイプ	ステータス	64ビット	M
名前	タイプ	ステータス	64ビット	MSS有効	セキュリティ		
DBDEMO	Region	Stopped	✓	✓	デフォルト		

- 5) DBDEMO インスタンスにカーソルを合わせ、[編集] アイコンをクリックします。

名前	タイプ	ステータス	64ビット	MSS有効	セキュリティ	
DBDEMO	Region	Stopped	✓	✓	デフォルト	<input checked="" type="checkbox"/> 編集

- 6) DBDEMO インスタンスのログなどが出力される [システムディレクトリ] には前項で作成した system フォルダを指定して、[リージョンの機能] の [JES 有効] をチェックします。指定後は [適用] ボタンをクリックします。

開始オプション

名前: DBDEMO

システムディレクトリ: C:\work\DBDEMO\system

リージョンの機能

MSS有効

JES有効

MQ有効

- 7) 表示画面の下にある [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 からの動的デバッグが可能になります。

ローカルコンソールを表示
 動的デバッグを許可
 システム起動時に開始する

64ビット作業モード
 以前のログを削除

- 8) [追加設定] の [構成情報] 欄に、文字エンコーディングを指定する MFACCCGI_CHARSET 環境変数に IBM-932 を認識させるための値である Shift_JIS と、プロジェクトのパスを指定する環境変数を設定し、最後に [適用] ボタンをクリックします。

入力値)

[ES-Environment]

proj=C:\work\DBDEMO
MFACCCGI_CHARSET=Shift_JIS

追加設定

構成情報 ⓘ

[ES-Environment]
proj=C:\work\DBDEMO
MFACCCGI_CHARSET=Shift_JIS

- 9) 画面上部の [JES] プルダウンメニューから [構成] を選択し、表示される画面の各項目を設定します。構成情報に指定した proj 環境変数を使用して値を入力後、[適用] ボタンをクリックします。

項目名	説明
JES プログラム パス	COBOL アプリケーションの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。

JESの構成 | ⓘ 適用

JES プログラムパス ⓘ C:\work\DBDEMO\New_Configuration.bin ⓘ	システムカタログ ⓘ Sproj\DATAFILE\catalog.dat ⓘ
データセットの省略時ロケーション ⓘ \$proj\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 ⓘ	



重要

入力値は全て半角英数字で指定してください。

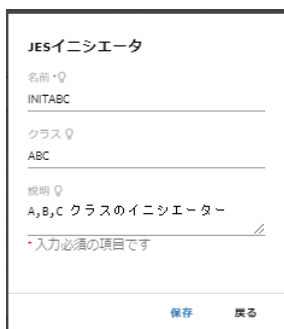
これらのフィールドでは改行を入れないように注意してください。

- 10) [イニシエータ] の [新規作成] ボタンをクリックします。

イニシエータ | ⓘ * 新規作成

名前 ↓	クラス ↓	説明 ↓
名前
...
...
合計	0	

- 11) 下記画面のように入力して [保存] ボタンをクリックします。この指定により DBDEMO 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。



重要

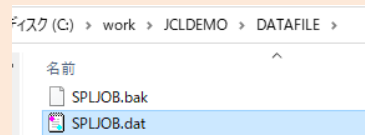
バージョン 7.0 では、パフォーマンス向上の観点から JES 関連ファイルである SPLJOB.DAT のフォーマットが改善されています。そのため、旧バージョンのファイルを 7.0 で利用する場合は mfsplcnv コマンドを使用して新フォーマットにコンバートする必要があります。コンバートを実行すると、古いフォーマットのファイルは SPLJOB.bak として保存されます。

対象ファイルの特定には MFSYSCAT 環境変数を利用して、カタログファイルを指定します。

例)

```
set MFSYSCAT=C:¥work¥JCLDEMO¥DATAFILE¥catalog.dat
```

```
mfsplcnv -2
```



詳しくは製品マニュアルをご参照ください。

- 12) 前項で作成した XA リソースを登録します。画面上部の [一般] プルダウンメニューから [XA リソース] を選択し、表示される画面で [新規作成] ボタンをクリックします。

項目名	説明
ID	プログラムや JCL の IKJEFT ユーティリティに渡す DSN TSO コマンドの SYSTEM パラメタへ指定する ID を指定します。ここでは XADB を指定します。 ID * XADB
名前	XA リソース名として任意の名前を指定します。Oracle の場合は Oracle_XA 固定です。 名前 * ORACLE_XA
モジュール	前項で作成した XA スイッチモジュールのパスとファイル名を指定します。 【Oracle 使用時の例】 動的登録の C:¥xa¥ESORAXA_D.dll を入力します。 モジュール * C:\xa\ESORAXA_D.dll

	<p>【Db2 使用時の例】 動的登録の C:\xa\ESDB2XA.dll を入力します。 モジュール名 c:\xa\ESDB2XA.DLL</p> <p>【SQL Server 使用時の例】 動的登録の C:\xa\ESMSSQL_D.dll を入力します。 モジュール名 c:\xa\ESMSSQL_D.dll</p>
再接続試行	再接続の試行回数を指定します。デフォルトは 1 で 0 は指定できません。-1 は継続的に試行します。ここではデフォルトの 1 を使用します。
OPEN 文字列	<p>対象データベースのオープン文字列を指定します。</p> <p>【Oracle 使用時の例】 ORACLE_XA+SeSTm=100+SqlNet=orcl-19c+Acc=P/scott/tiger を入力します。</p> <p>【Db2 使用時の例】 DB=DEMODB,uid=tarot,pwd=password,AXLIB=casaxlib を入力します。 静的登録の場合は末尾に SREG=T を指定します。デフォルトは動的です。</p> <p>【SQL Server 使用時の例】 DSN=SQLSVR を入力します。(=ODBC名)</p>
有効	有効、無効切り替えチェックを指定します。ここではオンを指定します。

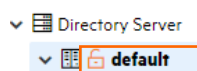
XA リソースの構成		XA リソースの構成		XA リソースの構成	
ID	名前	ID	名前	ID	名前
XADB	ORACLE_XA	XADB	DB2_XA	XADB	SQLSVR_XA
モジュール名 c:\xa\ESORAXA_D.dll		モジュール名 c:\xa\ESDB2XA.DLL		モジュール名 c:\xa\ESMSSQL_D.dll	
<input checked="" type="checkbox"/> 有効	再接続試行 1	<input checked="" type="checkbox"/> 有効	再接続試行 1	<input checked="" type="checkbox"/> 有効	再接続試行 1
OPEN 文字列 ORACLE_XA+SeSTm=100+SqlNet=orcl-19c+Acc=P/oracle/oracle		OPEN 文字列 DB=DEMODB,uid=tarot,pwd=password,AXLIB=casaxlib		OPEN 文字列 DSN=SQLSVR	

- 13) セキュリティ観点から、Web リスナーのデフォルトステータスは [Disabled] になっています。安全を確認したうえで、[一般] プルダウンメニューから [リスナー] を選択し、表示された Web リスナーのステータスを [Stopped] へ変更後、[適用] ボタンをクリックします。

TLS設定

ステータス	ステータスの設定	実際のアドレス
Disabled	Stopped	tcp:0.0.0.0:0

- 14) 画面左側ペインの [Default] をクリックして一覧画面に戻ります。

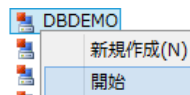


3.9 Enterprise Server インスタンスの開始と確認

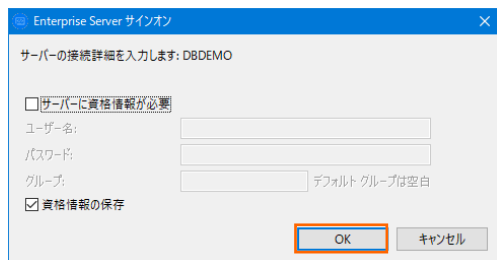
- 1) Eclipse へ戻り、サーバー エクスプローラー内に DBDEMO インスタンスが表示されていることを確認します。表示されていない場合は [Default] を右クリックし、[更新] を選択してリフレッシュしてください。
- 2) サーバー エクスプローラー内の DBDEMO インスタンスを右クリックし、[プロジェクトに関連付ける] > [DBDEMO] を選択します。これにより DBDEMO プロジェクトから実行されるアプリケーションは DBDEMO インスタンスで処理されることとなります。



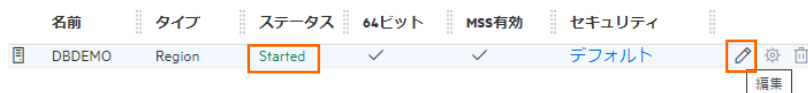
- 3) DBDEMO インスタンスを右クリックして [開始] を選択します。



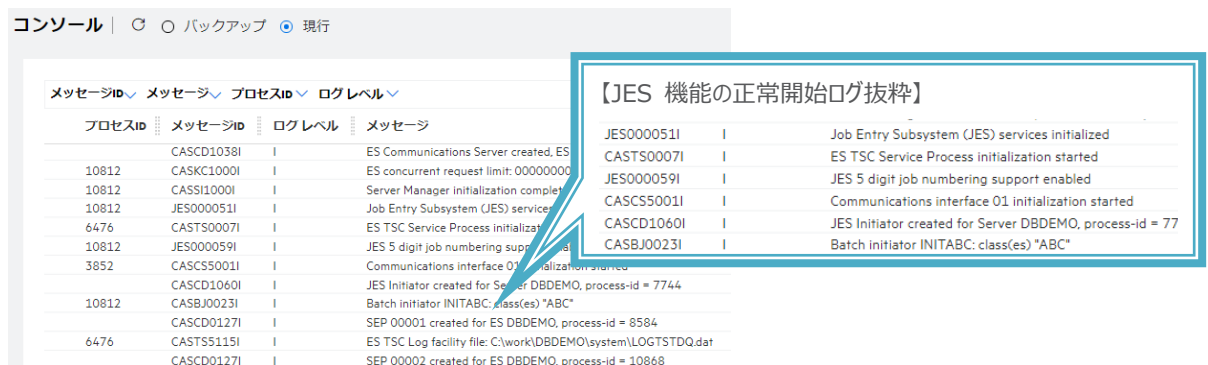
- 4) 下記ウィンドウが表示された場合は、ここではユーザーによる制限を行わないため [OK] ボタンをクリックします。



- 5) ESCWA 画面へ移動して開始状態であることを確認後、[編集] アイコンをクリックします。



- 6) 画面上部の [モニター] プルダウンメニューから [ログ] > [コンソールログ] を選択し、正常に開始されたことを確認します。メッセージ ID の最終文字が S や E の場合はエラー表示されます。



【JES 機能の正常開始ログ抜粋】

プロセスID	メッセージID	ログレベル	メッセージ
JES000051I		I	Job Entry Subsystem (JES) services initialized
CASTS0007I		I	ES TSC Service Process initialization started
JES000059I		I	JES 5 digit job numbering support enabled
CASCSS001I		I	Communications interface 01 initialization started
CASCD1060I		I	JES Initiator created for Server DBDEMO, process-id = 774
CASBJ0023I		I	Batch initiator INITABC, class(es) "ABC"
CASCD0127I		I	SEP 00001 created for ES DBDEMO, process-id = 8584
CASTS5115I		I	ES TSC Log facility file: C:\work\DBDEMO\system\LOGTSTDQ.dat
CASCD0127I		I	SEP 00002 created for ES DBDEMO, process-id = 10868

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) XA モジュールが正常にロードされ、オープンされると以下のようなログが出力されます。下記は Db2 の例ですが、動的登録を指示しているため Dynamic と出力されています。静的登録の場合は Static が出力されます。

CASXO0020I	I	XADB XA interface loaded. Name(DB2 for WINDOWS), Registration Mode(Dynamic)
CASXO0021I	I	XADB XA interface local transaction support enabled
CASXO0015I	I	XADB XA interface initialized successfully

- 8) 画面左側ペインの [Default] をクリックして一覧画面に戻ります。

3.10 データベースアクセスを含む COBOL バッチプログラムの実行

現在 DBDEMO インスタンスが稼働していますので、Eclipse から例題プログラムを実行することができます。まずはプログラム内容を各自ご確認いただき、JCL を実行します。

- 1) COBOL エクスプローラー内にある DBDEMO プロジェクト配下の dbdemo1.jcl をダブルクリックし、エディタで内容を確認します。

```

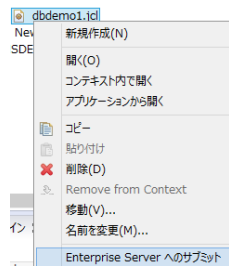
//DBDEMO1 JOB CLASS=A,MSGCLASS=A
//*****
//STEP01 EXEC PGM=IKJEFT01
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(XADB)
  RUN PROGRAM(TBLCRTE) PLAN(SAMPLES)
END
/*
//SYSTSPT DD SYSOUT=*
//*****
//STEP02 EXEC PGM=IKJEFT01
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
DSN SYSTEM(XADB)
  RUN PROGRAM(TBLISRT) PLAN(SAMPLES)
END
/*
//SYSDIN DD *
00001Soseki Natsume      1-1,Koishikawa,Bunkyo-ku,Tokyo-to      1886
00002Ryotaro Shiba      2-3,Sonezaki,Kita-ku,Osaka-shi,Osaka-fu  1908
00003Hideyo Neguchi     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
00007Ogai Mori          3-1,Rintaro-cho,Tsuwano-shi,Shimane-ken  1886
00008Ryoma Sakamoto     1-1,Harimayabashi,Kochi-shi,Kochi-ken   1828
00009Shiki Masaoka      5-5,Dogo Onsen,Natsuyama-shi,Ehime-ken  1878
00010Yukichi Fukuzawa   8-8,Keio-cho,Nakatsu-shi,Oita-ken      1835
/*
//SYSTSPT DD SYSOUT=*
//*****
//STEP03 EXEC PGM=IKJEFT01

```

この JCL は 4 ステップから構成されています。

- ① STEP01
データベーステーブルを新規作成します。
- ② STEP02
SYSIN データを作成したテーブルへ挿入します。
- ③ STEP03
挿入したデータをテーブルから全件読み込み、SYSOUT へ出力します。
- ④ STEP04
作成したテーブルをデータベースから削除します。

- 2) COBOL エクスプローラー内の dbdemo1.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。



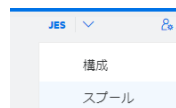
- 3) コンソールタブに JOB の実行ログと JOB 番号が表示されますので、ESCWA からスプールを確認します。

```

SJTJ00 J3532025 1144 DBDEMO JCGH0T8SI J000T002 DBDEWOT JOB EMDDED - COMD CODE 0000 J3:53:20
SJTJ00 J3532022 1144 DBDEMO JCGH0T8BI J000T002 DBDEWOT JOB 2144LLED J3:53:40
SJTJ00 J3532044 J08T5 DBDEMO JE2000004I J000T002 DBDEWOT JOB DI204LCHED J3:53:40
SJTJ00 J3532030 SJ20 DBDEMO JCGH0T8BI J000T002 DBDEWOT JOB L69QL L0L EXECN470U J3:53:40
SJTJ00 J3532030 SJ20 DBDEMO JCGH0T8LI J000T002 DBDEWOT JOB 2NBHILLED (JOBNAME=DBDEWOT*JOBNAME=000T002) J3:53:40

```

- 4) ESCWA で DBDEMO インスタンスを選択後、[JES] プロダクションメニューから [スプール] を選択します。



- 5) スプーラー一覧が表示されます。フィルタ機能で [完了] が指定されていることを確認します。



- 6) 実行した JOB 番号のスプールをダブルクリックして内容を表示します。

DD名	ステップ	PROCステップ	状態	クラス	レコード数	
DBDEMO1	J0001005	A	mfuser	0000		
状態	クラス	DD名	ステップ	ステップ番	PROCステ	レコード数
Hold	A	JESYSMSG		0		96
Ready	A	SYSOUT	STEP01	1		1
Ready	A	SYSTSPRT	STEP01	1		5
Ready	A	SYSOUT	STEP02	2		11
Ready	A	SYSTSPRT	STEP02	2		5
Ready	A	SYSOUT	STEP03	3		1

- 7) 先頭の [JESYSMEG] をダブルクリックしてジョブログを確認すると、正常に終了していることが確認できます。

```
---> 13:23:50 JCLCM0191I STEP ENDED      STEP04 - COND CODE 0000
---> 13:23:50 JCLCM0182I JOB  ENDED      - COND CODE 0000
```

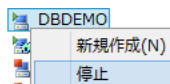
- 8) 右上にある [戻る] ボタンをクリックしてスプルー一覧に戻り、STEP03 の SYSOUT を表示すると、作成したテーブルからインサート済データが正常に FETCH できていることが確認できます。

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

- 9) JCL チュートリアルに記載しているように、バッチプログラムのデバッグも可能です。

3.11 Enterprise Server インスタンスの停止

- 1) Eclipse のサーバーエクスプローラーから DBDEMO インスタンスを停止します。



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

名前	タイプ	ステータス	64ビット	MSS有効
DBDEMO	Region	Stopped	✓	✓

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に基づき適切な扱いを行ってください。

WHAT'S NEXT

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