

# Enterprise Developer チュートリアル

## メインフレーム COBOL 開発：データベース連携 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、データベース接続を使用した JCL の実行までを行い、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利用したマシン OS：Windows 11 Pro
- 利用マシンに Enterprise Developer 11J for Eclipse がインストールされていること
- 利用マシンに対象のデータベースクライアントがインストール済みで、プリコンパイラを含め動作確認済みであること

### 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. テキストファイルのエンコード指定
5. プロジェクトプロパティの設定
6. ビルドの実行
7. XA スイッチモジュールの生成
8. 文字エンコーディングの設定
9. Enterprise Server インスタンスの設定
10. Enterprise Server インスタンスの開始と確認
11. 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
12. Enterprise Server インスタンスの停止

### 4. 免責事項

### 3.1 チュートリアルの準備

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。



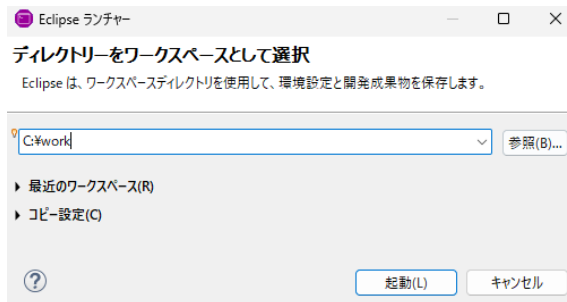
- 2) Eclipse のワークスペースで使用する work フォルダを C:¥直下に作成します。

### 3.2 Eclipse の起動

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



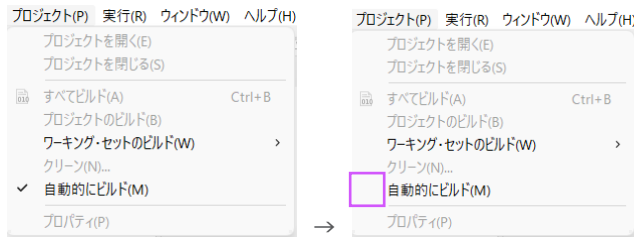
- 2) 前項で作成した C:¥work をワークスペースへ指定して、[起動] ボタンをクリックします。



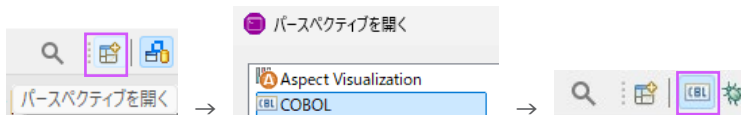
- 3) [ようこそ] タブが表示されたら [Open COBOL Perspective] をクリックして、COBOL パースペクティブを開きます。



- 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。

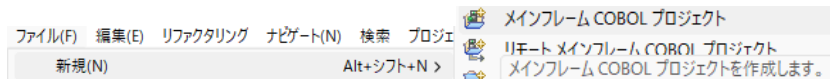


- 5) COBOL パースペクティブが開いていない場合は Eclipse 右上の [パースペクティブを開く] アイコンをクリックして表示後、[COBOL] を選択して [開く] ボタンをクリックします。

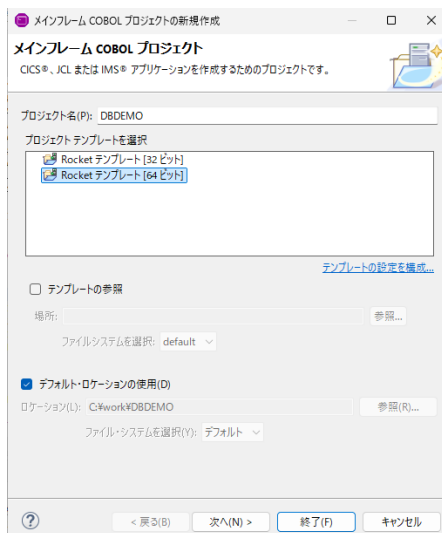


### 3.3 メインフレーム COBOL プロジェクトの作成

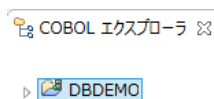
- 1) 用意した例題ソースをインポートします。[ファイル] プルダウンメニューから [新規] > [メインフレーム COBOL プロジェクト] を選択します。



- 2) [プロジェクト名] は任意ですが、ここでは 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。

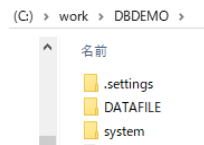


- 3) COBOL エクスプローラーへ作成したプロジェクトが表示されます。

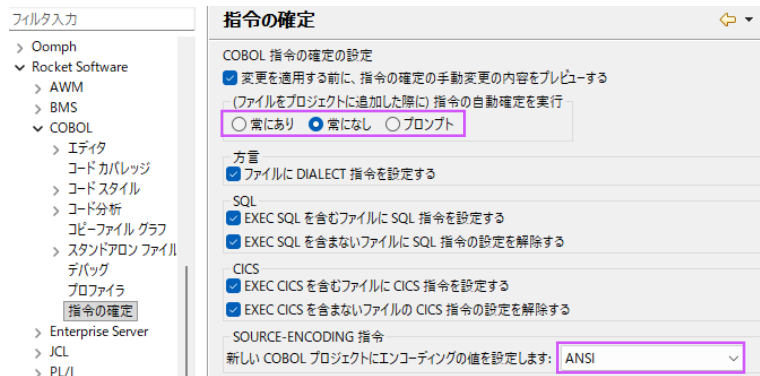


- 4) プロジェクトを作成したことにより C:\work\BDBDEMO フォルダが作成されています。このフォルダ配下に JES 機能で使用するフォルダを、Windows エクスプローラーを使用して、あらかじめ用意しておきます。

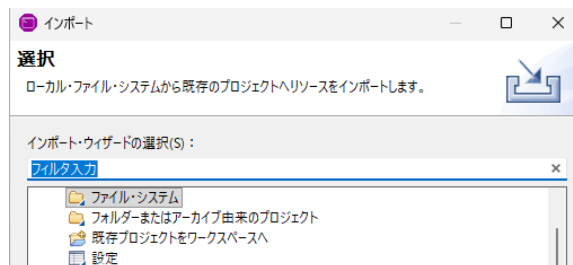
カタログファイルやスプールファイルを配置するため DATAFILE フォルダを C:\work\BDBDEMO 配下へ、実行時にログなどを格納する system フォルダを C:\work\BDBDEMO 配下へ作成します。



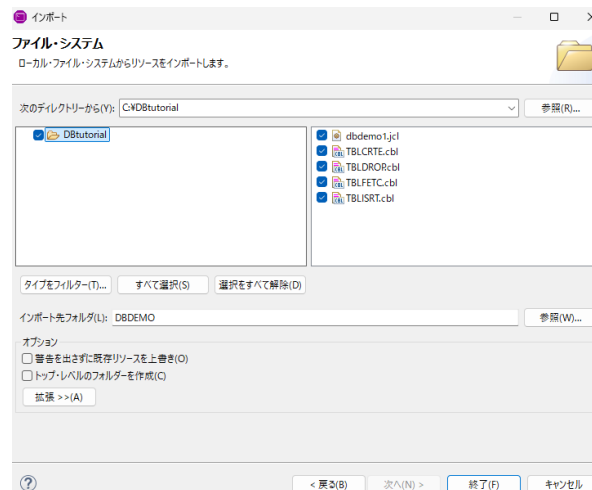
- 5) 既存ファイルのインポート時、自動的にコンパイル指令が指定される機能が用意されていますが、本チュートリアルではこれを解除します。[ウィンドウ] プルダウンメニューの [設定] > [Rocket Software] > [COBOL] > [指令の確定] > [COBOL 指令の確定の設定] では [常になし] を選択し、[SOURCE-ENCODING 指令] では [ANSI] を選択後、[適用して閉じる] ボタンをクリックします。



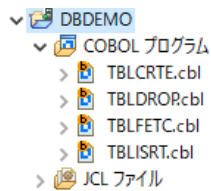
- 6) 用意した例題プログラム類をインポートします。DBDEMO プロジェクトを右クリックして [インポート] > [インポート] を選択し、インポートウィンドウにて [一般] > [ファイル・システム] を選択後 [次へ] ボタンをクリックします。



- 7) C:\BDBtutorials を [次のディレクトリから] 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。



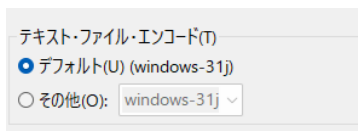
- COBOL エクスプローラー内に表示されている DBDEMO にインポートしたファイルが表示されていることを確認します。



### 3.4 テキストファイルのエンコード指定

Eclipse ではワークスペースの設定として、テキストファイルのエンコードを指定できます。ソースファイル類のエンコードに沿って適切なエンコードを指定してください。この例題では MS932 を使用します。

- Eclipse の [ウィンドウ] プルダウンメニューから [設定] を選択し、設定ウィンドウを表示します。  
左側ペインで [一般] > [ワークスペース] を選択し、右側ペインの [テキスト・ファイル・エンコード] に Windows-31j または MS932 が指定されていることを確認します。「デフォルト」がこの値ではない場合は、「その他」を選択して、これらを指定してください。



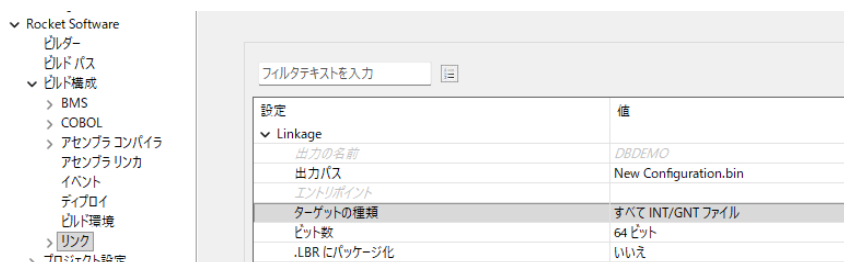
- 指定後は [適用して閉じる] ボタンをクリックします。

### 3.5 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、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るため、この方法を使用します。

- 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。
- 左側ツリービューの [Rocket Software] > [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 [すべて INT/GNT ファイル] を選択します。
ビット数	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。



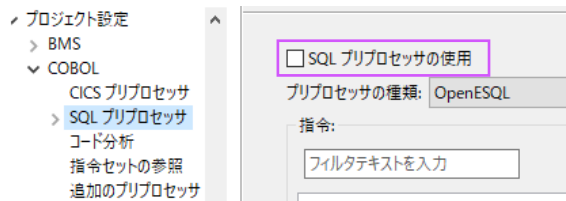
- 3) 左側ツリービューの [Rocket Software] > [プロジェクト設定] > [COBOL] を選択して、下記項目を指定します。指定後は [適用] ボタンをクリックしてください。

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。						
ソースエンコーディング	Eclipse に指定したエンコードと一致するエンコーディングを指定します。MS932 または Windows-31j と一致するように ANSI を指定します。						
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 使用時の例: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 使用時の例: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 使用時の例: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)						

- ▼ Rocket Software
  - ビルダー
  - ビルドパス
  - ▼ ビルド構成
  - ▼ プロジェクト設定
    - ▼ BMS
    - ▼ COBOL
      - CICS プリプロセッサ
      - ▼ SQL プリプロセッサ
        - コード分析
        - 指令セットの参照
        - 追加のプリプロセッサ
      - Enterprise Server
      - ▼ IMS
      - ▼ アセンブラコンパイラ
      - アセンブラリンク
      - コンテナー
      - ビルド環境
      - 指令の確定
      - ▼ 実行時構成
      - WikiText
      - タスク・タグ
      - ▼ タスク・リポジトリ
      - ビルダー
      - プロジェクト・ネーチャー

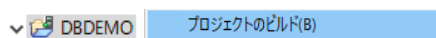
フィルタテキストを入力	
設定	値
▼ 一般	
文字セット	ASCII
ソースエンコーディング	ANSI
COBOL 方言	Enterprise COBOL for z/OS
ソースフォーマット	固定
メインフレームのコピー処理	COPY
デバッグ用にコンパイル	はい
EXIT PROGRAM を GOBACK として処理	ANSI
詳細	いいえ
.GNT にコンパイル	はい
▼ 出力	
指令ファイルを生成する	いいえ
リストファイルを生成	いいえ
コードカバレッジを有効にする	false
プロファイラを有効にする	false
▼ エラー/警告	
警告レベル	回復可能なエラーを含める(レベル E)
最大エラー数	100
▼ 追加指令	
追加指令	SQL(DBMAN=ODBC,BEHAVIOR=JCL,TARGETDB=MSSQLSERVER)

- 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 [Rocket Software] > [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。

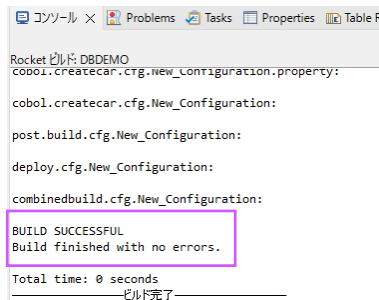


### 3.6 ビルドの実行

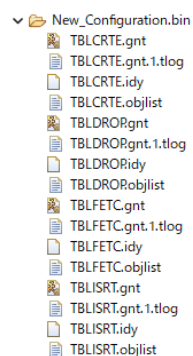
- COBOL エクスプローラー内のプロジェクトを右クリックして [プロジェクトのビルド] を選択するとビルドが実行されます。



- [コンソール] タブで成功を確認します。



- 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。



### 3.7 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 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) XA スイッチモジュールのビルドを行うため、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を許容しない。

【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)¥Rocket Software¥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 [Rocket 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 [詳細] > [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:¥xa>cblms -L
Rocket (R) COBOL - Configuration Utility for the Microsoft Build Tools & SDK
11.0.0.88 (C) 1984-2025 Rocket Software, Inc. or its affiliates.

Windows SDK
Id バージョン 場所
0] 10.0.26100.0 c:¥Program Files (x86)¥Windows Kits¥10

Microsoft Build Tools
Id バージョン 場所
```

最新バージョンに設定するコマンド)

cblms -U

```
c:¥xa>cblms -U
Rocket (R) COBOL - Configuration Utility for the Microsoft Build Tools & SDK
11.0.0.88 (C) 1984-2025 Rocket Software, Inc. or its affiliates.

Windows SDK
場所 = c:¥Program Files (x86)¥Windows Kits¥10
バージョン = 10.0.26100.0
エラー: 有効な Build Tools が見つかりません
```



特定のバージョンを指定するコマンド例):以降は-L で表示された番号を指定します。  
 Windows SDK を指定する場合)cblms -US:0  
 Build Tools を指定する場合)cblms -UB:0

```
c:\xa>cblms -US:0
Rocket (R) COBOL - Configuration Utility for the Microsoft Build Tools & SDK
11.0.0.88 (C) 1984-2025 Rocket Software, Inc. or its affiliates.

Windows SDK
場所 = c:\Program Files (x86)\Windows Kits\10
バージョン = 10.0.26100.0
```

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
Rocket (R) COBOL - Configuration Utility for the Microsoft Build Tools & SDK
11.0.0.88 (C) 1984-2025 Rocket Software, Inc. or its affiliates.

Windows SDK
場所 = c:\Program Files (x86)\Windows Kits\10
バージョン = 10.0.26100.0

Microsoft Build Tools
場所 = C:\Program Files (x86)\Rocket Software\Enterprise Developer\Microsoft
バージョン = 14.40.33807
```

利用可能なオプションを表示するコマンド)

cblms -H



- 1) 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 ora21)

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

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。

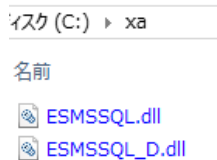
#### ③ SQL Server

##### 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。

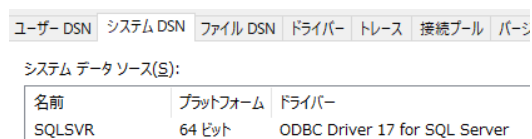
コマンド)build mssql17



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。

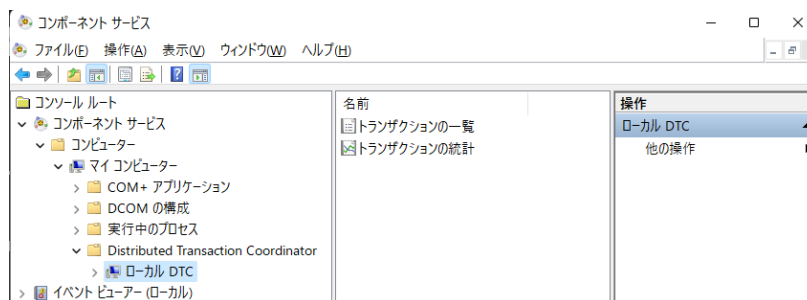
## B) ODBC の追加

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] > [Windows 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。

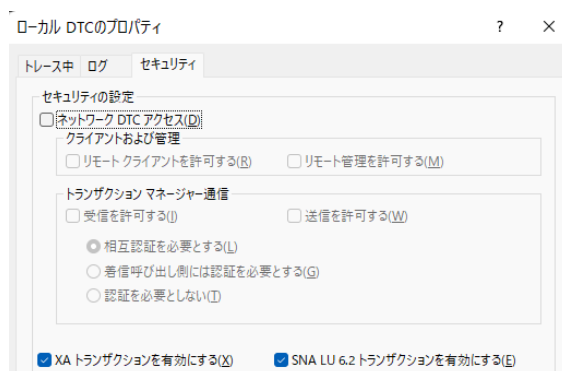


## C) XA トランザクションの有効化

Windows の [コントロールパネル] > [Windows ツール] > [コンポーネントサービス] > [コンピューター] > [マイコンピューター] > [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。

### 3.8 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。  
<https://download.boulder.ibm.com/ibmdl/pub/software/dw/java/cdctables.zip>

- 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)\Rocket Software\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。

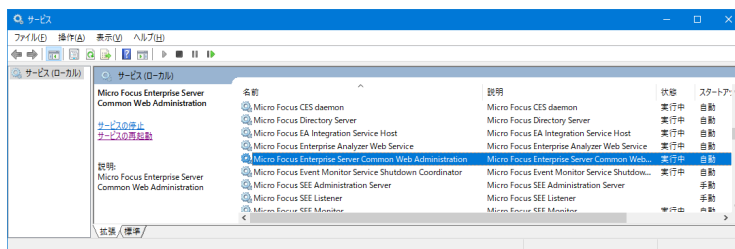
- 4) 展開した Package2 フォルダに含まれる IBM-932.zip を展開します。

- 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。

名前	種類	サイズ
03A434B0.MU-R-A2	MU-R-A2 ファイル	28 KB
03A434B0.MU-R-A3	MU-R-A3 ファイル	28 KB
03A434B0.MU-R-D	MU-R-D ファイル	28 KB
03A434B0.PACKAGE	PACKAGE ファイル	5 KB
03A434B0.TPMAP11A	TPMAP11A ファイル	329 KB
03A434B0.TPMAP12A	TPMAP12A ファイル	329 KB
03A434B0.TPMAP110	TPMAP110 ファイル	329 KB
03A434B0.UPMAP12A	UPMAP12A ファイル	446 KB
03A434B0.UPMAP13A	UPMAP13A ファイル	490 KB
03A434B0.UPMAP120	UPMAP120 ファイル	447 KB
34B003A4.RPMAP12A	RPMAP12A ファイル	336 KB
34B003A4.RPMAP120	RPMAP120 ファイル	336 KB
34B003A4.UM-E-A21	UM-E-A21 ファイル	54 KB
34B003A4.UM-E-D12	UM-E-D12 ファイル	54 KB

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。

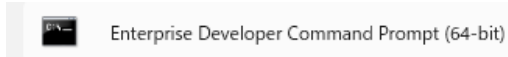


### 3.9 Enterprise Server インスタンスの設定

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) ESCWA は、製品が提供する VSAM 外部セキュリティマネージャー(ESM)モジュールによるセキュリティがデフォルトで有効になっており、すべての処理において実行ユーザーの認証が行われます。

まずはデフォルトユーザーと初期パスワードを取得するために、スタートメニューから [Enterprise Developer] を選択し、[Enterprise Developer コマンドプロンプト] を起動します。



Enterprise Developer コマンドプロンプトから次のコマンドを実行して、デフォルトユーザーと初期パスワードを取得します。

```
mfsecretsadmin read microfocus/temp/admin
```

```
C:\Users\tarot\Documents>mfsecretsadmin read microfocus/temp/admin
{"mfUser": "SYSAD", "mfPassword": "CKfa+xWt"}
```

上記例の場合、SYSAD がデフォルトユーザー、CKfa+xWt が初期パスワードです。

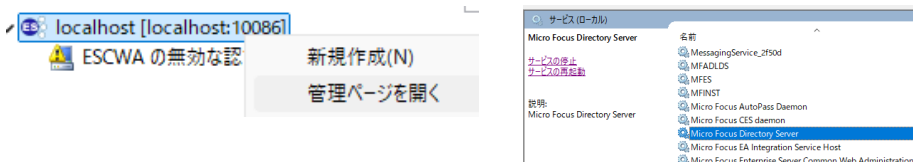
この情報は ESCWA のログオン時に使用しますので、記憶しておいてください。

デフォルトセキュリティを無効にすることもできますが、安全を確認後に実施してください。

詳しくは製品マニュアルの [ここからはじめよう] > [Getting Started] にある [デフォルトセキュリティの構成] チュートリアルをご参照ください。

- 2) 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 [localhost] を右クリックして [管理ページを開く] を選択します。表示されていない場合は、Windows サービスの [Micro Focus Directory Server] が開始されているかを確認し、停止している場合は開始してください。



- 3) Web ブラウザが立ち上がり、ユーザー認証を求められます。

パスワードを変更してログオンします。

[パスワード変更] をクリックし、前項で取得したデフォルトユーザーと初期パスワード、新しいパスワードを入力して [サブミット] をクリックします。

変更後のパスワードはご自身の責任で管理してください。

パスワード変更後、ESCWA がタイムアウトした場合は新しいパスワードを使用してログオンしてください。

Enterprise Server Administration

⚠ Rocket Software Enterprise Serverでは、インストール時に基本的なセキュリティ機能がデフォルトで有効になっています。


[詳細情報](#)

ユーザー名

パスワード

パスワード変更 ログイン

認証情報は、次のセキュリティ マネージャを使用  
して検証されます: VSAM ESM



Enterprise Server Administration

⚠ Rocket Software Enterprise Serverでは、インストール時に基本的なセキュリティ機能がデフォルトで有効になっています。

[詳細情報](#)


ユーザー名

パスワード

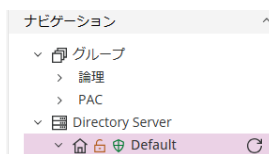
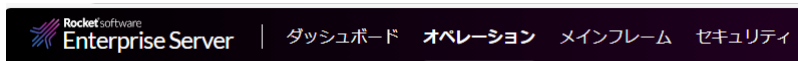
新しいパスワード\*  パスワードの確認\*

キャンセル サブミット

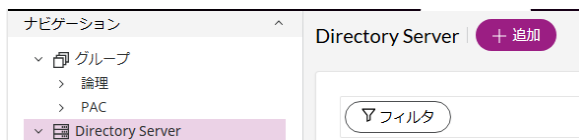
認証情報は、次のセキュリティ マネージャを使用  
して検証されます: VSAM ESM



- 4) メニューで【オペレーション】を選択後、【ナビゲーション】に【Default】という名前の接続が表示されます。表示されない場合は新規に作成します。表示されている場合はこの項目はスキップしてください。



左側ペインで【Directory Server】をクリックし、右側ペインの【追加】ボタンをクリックします。



追加ウィンドウの【名前】は任意ですが、ここでは Default を、  
【ホスト】は ホスト名またはマシンの IP アドレスを、  
【ポート】は Micro Focus Directory Server の  
デフォルトポートである 86 を入力して【保存】ボタンをクリックします。

**Directory Server**

名前\*

ホスト\*

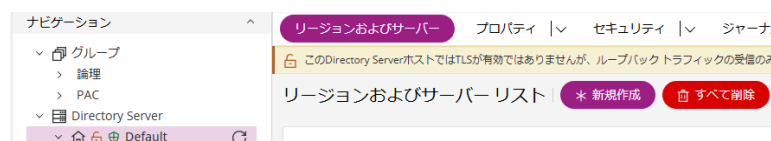
ポート\*

説明

\* 入力必須の項目です

保存
戻る

- 5) Enterprise Server インスタンスを追加します。  
【Default】をクリックし、右側ペインの【新規作成】ボタンをクリックします。



- 6) [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

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。

名前\*  
DBDEMO

説明  
DBチュートリアル用

☒ 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と Enterprise Server インスタンス稼働ビット数はイコールにします。

- 7) 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。

ア...	名前	タ...	ステータス	64ビット	MSS有効	セキュ...	アクション
	DBDEMO	Region	Stopped	✓	✓	デフォルト	

- 8) 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。

ア...	名前	タ...	ステータス	64ビット	MSS有効	セキュ...	アクション
	DBDEMO	Region	Stopped	✓	✓	デフォルト	

編集

- 9) 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、[リージョンの機能] の [JES 有効] にチェック、[動的デバッグを許可] にチェックします。

一般的なプロパティ 適用 削除

**開始オプション** \* 入力必須の項目です

名前\* システムディレクトリ  
DBDEMO C:\work\DBDEMO\system

共有メモリ ページ数 512 ページ数(4k): 共有メモリ クッション 32 ページ数(4k):  
SEP数 2 コンソール ログ サイズ 0 k

☐ ローカル コンソールを表示 ☒ 動的デバッグを許可

☒ 64ビット作業モード ☐ 以前のログを削除

☐ システム起動時に開始する

**リージョンの機能**

☒ MSS有効 ☒ JES有効

☐ IMS有効 ☐ MQ有効

**Windowsの監視と管理**

☐ パフォーマンス モニターを使用

イベント ログ:

☐ 情報 ☐ 警告

- 10) [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

- 11) 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。 ここでは何も指定しません。

**JESの構成** 適用

---

JES プログラム パス ⓘ  
\$proj\New\_Configuration.bin

システム カタログ ⓘ  
\$proj\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 ⓘ

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。

- 12) [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

- 13) 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。

**JESイニシエータ**

名前\* ⓘ  
INITABC

クラス ⓘ  
ABC

説明 ⓘ  
A,B,C クラスのイニシエータ

\* 入力必須の項目です 保存 戻る

### ❌ 重要

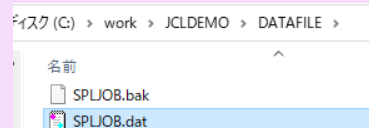
バージョン 7.0 では、パフォーマンス向上の観点から JES 関連ファイルである SPLJOB.DAT のフォーマットが改善されています。そのため、旧バージョンのファイルを 7.0 以降で利用する場合は mfsplcnv コマンドを使用して新フォーマットにコンバートする必要があります。コンバートを実行すると、古いフォーマットのファイルは SPLJOB.bak として保存されます。

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。

- 14) 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  【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 * c:\xa\ESDB2XA.DLL  【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 * c:\xa\ESMSSQL_D.dll
OPEN 文字列	対象データベースのオープン文字列を指定します。  【Oracle 使用時の例】 ORACLE_XA+SesTm=100+SqlNet=orcl-19c+Acc=P/scott/tiger を入力します。  【Db2 使用時の例】 DB=DEMODB,uid=tarot,pwd=password,AXLIB=casaxlib.dll を入力します。



	<p>Windows の場合)</p> <p>AXLIB に指定するモジュールのパスが PATH 環境変数に指定されていない場合はフルパスを指定してください。ビット数により bin64、bin のパスが異なりますのでご注意ください。ファイル拡張子は省略可能です。</p> <p>64 ビットフルパスの例: %COBDIR%\bin64\CASAXLIB.dll</p> <p>Linux/UNIX の場合)</p> <p>AXLIB に指定するモジュールのパスが LD_LIBRARY_PATH または LIBPATH 環境変数に指定されていない場合はフルパスを指定してください。ビット数により casaxlib64.so、casaxlib.so とファイル名が異なりますのでご注意ください。ファイル拡張子は必須です。</p> <p>64 ビットフルパスの例: \$COBDIR/lib/casaxlib64.so</p> <p>静的登録の場合は末尾に SREG=T を指定します。</p> <p>【SQL Server 使用時の例】</p> <p>ODBC 名を指定します。</p> <p>DSN=SQLSVR を入力します。</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。

### SQL Server の例)

#### XA リソースの構成

ID\*

名前\*

XADB

XADB

モジュール\*

C:\xa\ESMSSQL\_D.dll

☒ 有効

OPEN 文字列

DSN=SQLSVR

CLOSE 文字列

説明

\* 入力必須の項目です

保存 戻る

- 15) 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

- 16) 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。

▼

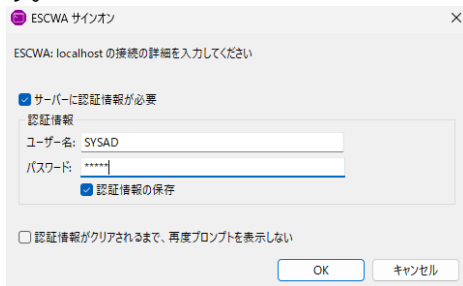
Directory Server

▼

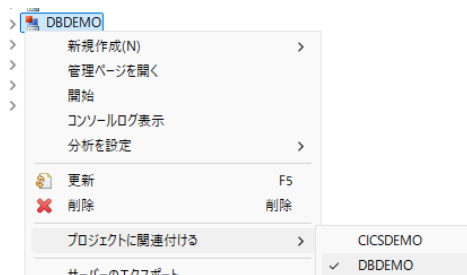
Default

### 3.10 Enterprise Server インスタンスの開始と確認

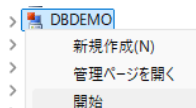
- 1) Eclipse に戻り、サーバーエクスプローラー内の [localhost] をクリックすると、ESCWA への認証ウィンドウが表示されますので、前述で指定した SYSAD ユーザーとそのパスワードを指定して [OK] ボタンをクリックします。



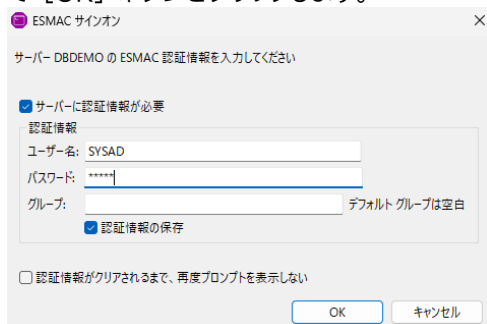
- 2) [Default] を展開して作成した DBDEMO インスタンスが表示されていることを確認します。表示されていない場合は [Default] を右クリックし、[更新] を選択してリフレッシュしてください。
- 3) サーバー エクスプローラー内の DBDEMO インスタンスを右クリックし、[プロジェクトに関連付ける] > [DBDEMO] を選択します。これにより DBDEMO プロジェクトから実行されるアプリケーションは DBDEMO インスタンスで処理されることになります。



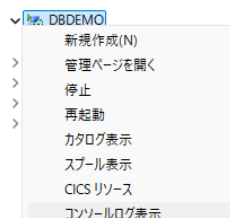
- 4) DBDEMO インスタンスを右クリックして [開始] を選択します。



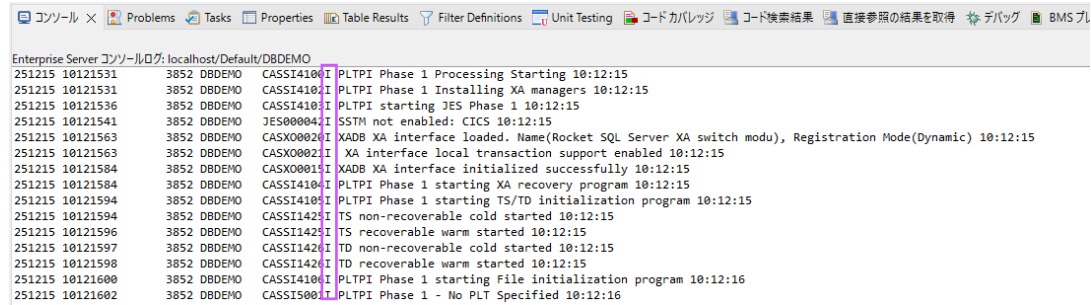
- 5) 開始ユーザーの認証ウィンドウが表示されますので、前述で指定した SYSAD ユーザーとそのパスワードを指定して [OK] ボタンをクリックします。



- 6) 開始後、DBDEMO を右クリックして [コンソールログ表示] を選択すると、[コンソール] タブに開始ログが表示されます。



メッセージコードの最終桁のログレベルが I はインフォメーション、W は警告、S や E の場合はエラーです。



```

Enterprise Server コンソールログ: localhost/Default/DBDEMO
251215 10121531 3852 DBDEMO CASSI4100I PLTPI Phase 1 Processing Starting 10:12:15
251215 10121531 3852 DBDEMO CASSI4100I PLTPI Phase 1 Installing XA managers 10:12:15
251215 10121536 3852 DBDEMO CASSI4100I PLTPI starting JES Phase 1 10:12:15
251215 10121541 3852 DBDEMO JES00004I SSTH not enabled: CICS 10:12:15
251215 10121563 3852 DBDEMO CASX00020I XADB XA interface loaded. Name(Rocket SQL Server XA switch modu), Registration Mode(Dynamic) 10:12:15
251215 10121563 3852 DBDEMO CASX00021I XA interface local transaction support enabled 10:12:15
251215 10121584 3852 DBDEMO CASX00015I XADB XA interface initialized successfully 10:12:15
251215 10121584 3852 DBDEMO CASSI4100I PLTPI Phase 1 starting XA recovery program 10:12:15
251215 10121594 3852 DBDEMO CASSI4100I PLTPI Phase 1 starting TS/TD initialization program 10:12:15
251215 10121594 3852 DBDEMO CASSI4125I TS non-recoverable cold started 10:12:15
251215 10121596 3852 DBDEMO CASSI4125I TS recoverable warm started 10:12:15
251215 10121597 3852 DBDEMO CASSI4125I TD non-recoverable cold started 10:12:15
251215 10121598 3852 DBDEMO CASSI4125I TD recoverable warm started 10:12:15
251215 10121600 3852 DBDEMO CASSI4100I PLTPI Phase 1 starting File initialization program 10:12:16
251215 10121602 3852 DBDEMO CASSI5000I PLTPI Phase 1 - No PLT Specified 10:12:16
  
```



## 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 SQL Server 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。

```

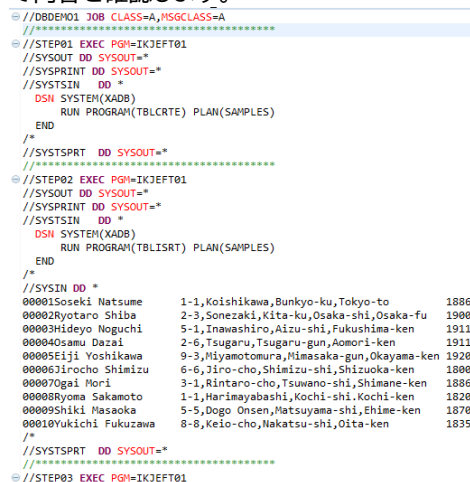
CASX00020I XADB XA interface loaded. Name(Rocket SQL Server XA switch modu), Registration Mode(Dynamic) 10:12:15
CASX00021I XA interface local transaction support enabled 10:12:15
CASX00015I XADB XA interface initialized successfully 10:12:15
  
```

- 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。

## 3.11 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。プログラム内容を各自ご確認ください、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
/*
//SYSTSIN DD *
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,Hiyamotomura,Himasaka-gun,Okayama-ken      1920
00006Jirocho Shimizu     6-8,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
/*
//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。

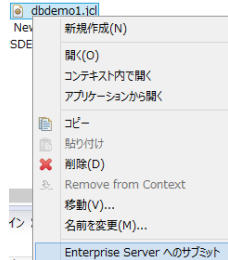
### ③ STEP03

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。

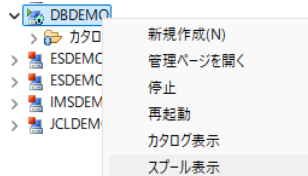
### ④ STEP04

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) サーバエクスプローラー内の DBDEMO インスタンスを右クリックし、[スプール表示] を選択します。



実行された JCL の結果を確認すると、条件コードはゼロで正常終了したことがわかります。

該当行をダブルクリックして詳細を表示します。

スプール: DBDEMO				
名前	ジョブ ID	クラス	ユーザー	条件コード
DBDEMO1	J0001000	A	SYSAD	0000

DD エントリをダブルクリックすると内容が確認できます。

スプール: DBDEMO
ジョブ ID: J0001000

ジョブの詳細

ジョブ ID: J0001000  
ユーザー: SYSAD  
COND: 0000  
優先度: 0

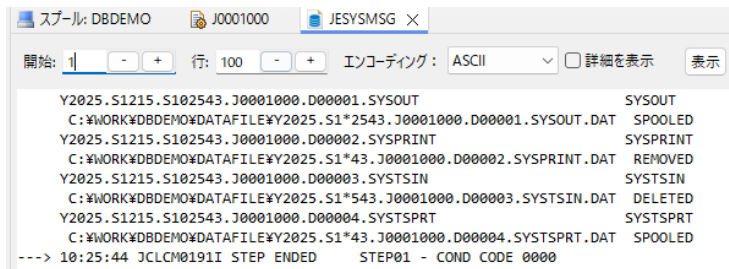

ジョブ名: DBDEMO1  
状態: Complete  
クラス: A  
ファイル: STXRFDIR/SYSAD10254354.t

メッセージ

JCLCP01881 J0001000 DBDEMO1 JOB STARTED 10:25:43  
JCLCP01882 J0001000 DBDEMO1 JOB ENDED - COND CODE 0000 10:25:44

DD エントリ

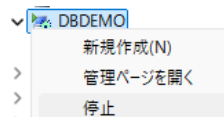
状態	クラス	DD 名	ステップ	ステップ番号	PROC ステップ	レコード数
Hold	A	JESYSMSG		0		112
Ready	A	SYSOUT	STEP01	1		1
Ready	A	SYSTSPRT	STEP01	1		5
Ready	A	SYSOUT	STEP02	2		11
Ready	A	SYSTSPRT	STEP02	2		5
Ready	A	SYSOUT	STEP03	3		11
Ready	A	SYSTSPRT	STEP03	3		5
Ready	A	SYSOUT	STEP04	4		1
Ready	A	SYSTSPRT	STEP04	4		5

- 4) 他のステップも確認し、テーブルが正常に作成、削除されていることを確認してください。
- 5) 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。

### 3.12 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。

## 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。

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。