

## Visual COBOL チュートリアル

### JCA による JBOSS EAP 連携の設定と開発

#### 1 目的

Visual COBOL に付属する COBOL 専用のアプリケーションサーバー「Enterprise Server」は、ネイティブにコンパイルした COBOL のビジネスロジックを EJB として再利用し、Java EE ライアントから呼び出す機能を提供しています。EJB コンポーネントとして呼び出しを行う場合、Java アプリケーションサーバー上の Java EE クライアントは JCA の仕様にもとづいたリクエストを Enterprise Server に渡し、COBOL のビジネスロジックが処理をして結果を返します。また、Enterprise Server は、XA に準拠しているのと同じく XA に準拠しているデータベースや他のシステムと協調してトランザクション処理を行うことができます。

このドキュメントでは Windows Server 上の JBOSS EAP 7.4 と Enterprise Server を JCA による連携を行い、Eclipse で開発する方法を説明します。

#### 2 前提

本チュートリアルは、下記の環境を前提に作成されています。サポートしているプラットフォームであれば他の Linux/UNIX でも利用可能です。

##### ● アプリケーションサーバー ソフトウェア

OS	Windows 11
Java	Adoptium 21.0.7
AP サーバー製品	JBoss Enterprise Application Server 8.1

##### ● 開発クライアント ソフトウェア

OS	Windows 11
COBOL 環境製品	Visual COBOL 11.0 for Eclipse
PostgreSQL ODBC Driver	psqlodbc 16.00.00.07
Windows SDK	10.0.26100.6901

本チュートリアルでは、アプリケーションサーバーと開発クライアントは同一環境としていますが、別環境構成も可能です。

なお、本チュートリアルでは、データベースに PostgreSQL を使用していますので、PostgreSQL 環境が必要です。  
チュートリアルを実施するにあたり、必要なテーブル、レコードについては、以下のチュートリアル用サンプルプログラム配下の `setup_database.sql` に記載しておりますので、事前にテーブル作成、レコード挿入を実施してください。

- チュートリアル用サンプルプログラム

下記のリンクから事前にチュートリアル用のサンプルファイルをダウンロードして、任意のフォルダーに解凍しておいてください。

[サンプルプログラムのダウンロード](#)

## 内容

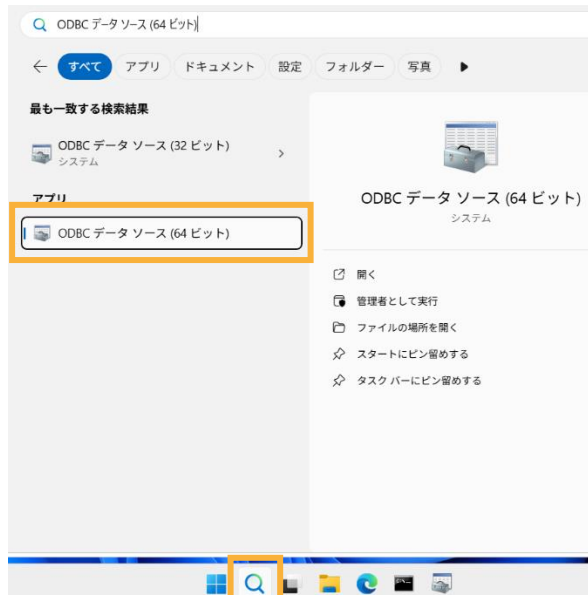
- 1 目的
- 2 前提
- 3 チュートリアル手順
  - 3.1 データソースの作成
  - 3.2 ODBC 用の XA スイッチモジュールの作成
  - 3.3 Visual COBOL for Eclipse を起動してソースコードのインポート
  - 3.4 JCA 用のアプリケーションプロファイルの作成
  - 3.5 デploy先の変更
  - 3.6 Deployしたアプリケーションのデバッグ

### 3 チュートリアル手順

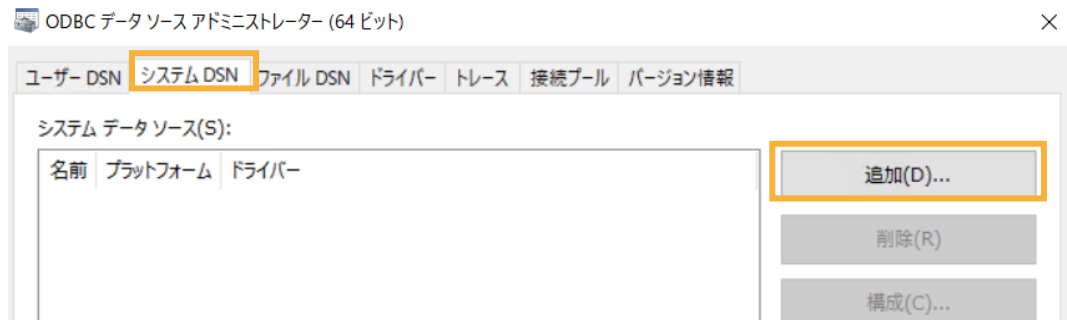
#### 3.1 データソースの作成

##### 1) データソースの作成

- ① タスクバーアイコンなどから検索を起動し、[ODBC データ ソース (64 ビット)] を検索して、それを選択します。



- ② [システム DSN] をクリックし、[追加(D)] をクリックします。



[PostgreSQL Unicode(x64)] を選択し、[完了] をクリックします。

データソースの新規作成

×

セットアップするデータソースのドライバーを選択してください(S)

名前	バージョン	会社名
PostgreSQL ANSI	16.00.00.00	Postgre
PostgreSQL ANSI(x64)	16.00.00.00	Postgre
PostgreSQL Unicode	16.00.00.00	Postgre
<b>PostgreSQL Unicode(x64)</b>	16.00.00.00	Postgre
SQL Server	10.00.17763.6054	Microso

< >

< 戻る(B) **完了** キャンセル

- ③ 以下の入力を行い、[作成(C)] をクリックします。

データソース名: PostgreSQL

サーバー名: PostgreSQL サーバーのアドレス

Port: PostgreSQL サーバーのリッスンポート番号

データベース名: チュートリアルで使用するサンプルテーブルの格納データベース名

PostgreSQL Unicode ODBC セットアップ

×

データソース名: (N) PostgreSQL

説明: (D)

SSL Mode: (L) 無効

サーバー名: (S) PostgreSQL-Addr

データベース名: (b) vcdemo

Port: 5432

既定の認証

ユーザー名: (u) postgres

パスワード: (w) ●●●●●●●●

オプション (高度な設定)

データソース

全体設定

PostgreSQL Ver7.3 Copyright (C) 1998-2006; Insight Distribution Systems  
In the original form, Japanese patch Hiroshi-saito

テスト

保存

キャンセル

なお、[保存] 前に [テスト] で構成情報に誤りがないかを確認ください。

データソースが登録されたことを確認したのち、本画面を閉じます。

ODBC データソースアドミニストレーター (64 ビット)

>

ユーザー DSN システム DSN ファイル DSN ドライバー トレース 接続プール バージョン情報

システム データソース(S):

名前	プラットフォーム	ドライバー
<b>PostgreSQL</b>	64 ビット	PostgreSQL Unicode(x64)

追加(D)...

削除(R)

### 3.2 ODBC 用の XA スイッチモジュールの作成

#### 2) XA スイッチモジュールの作成

- ① [スタート] メニュー > [Rocket Visual COBOL] > [Visual COBOL Command Prompt (64-bit)] を選択します。
- ② <製品のインストールフォルダー>¥src¥enterpriseserver 配下の xa フォルダーを任意のフォルダーにコピーします。本手順では C ドライブ直下にコピーします。

補足)

デフォルトのインストールフォルダーは、C:¥Program Files (x86)¥Rocket Software¥Visual COBOL です。

- ③ ① のコマンドプロンプト上で、以下のコマンドを実行して XA スイッチモジュールをビルドします。

注意)

各箇条書きのコマンドはそれぞれ 1 行で入力してください。LIB 環境変数に、Windows SDK へのパスを設定していますが、このパスはデフォルトのインストールフォルダーです。異なるフォルダーを指定した場合はそちらに修正してください。

また、環境変数設定時にはパス内に含まれる半角スペースにご注意ください。

- cd ¥xa
- set LIB="c:¥Program Files (x86)¥Windows Kits¥10¥Lib¥10.0.26100.0¥um¥x64";"c:¥Program Files (x86)¥Windows Kits¥10¥Lib¥10.0.26100.0¥um¥x86";%LIB%
- build odbbc

以下のように、「ESODBCXA\_D.dll」と「ESODBCXA.dll」ファイルが作成されます。

```
C:¥xa>dir |sort /R
2025/11/14 10:07 <DIR> .
2025/11/14 10:07          95,744 ESODBCXA_D.dll
2025/11/14 10:07          87,552 ESODBCXA.dll
2025/05/27 11:33          26,136 ESXAEXTCFG.CBL
2025/05/27 11:33           2,505 writeSecrets.bat
2025/04/27 01:45         106,658 ESPGSQLXA.CBL
2025/04/26 00:05         124,655 ESORAXA.CBL
2025/04/26 00:05          11,065 build.bat
2025/02/06 20:35          49,160 xapd.cpy
(以下、省略)
```

#### 3) リソースアダプターファイルの構成

- ① <製品のインストールフォルダー>¥javaee フォルダーを任意のフォルダーにコピーします。本チュートリアルでは、C ドライブ直下にコピーします。
- ② さきほど使用したコマンドプロンプト画面で上記フォルダーに移動したのち、“ravaluesupdater.bat”を実行します。実行後は以下の入力を行い、Enter キーを押します。  
Please enter the application server you would like to update: “jboss80EAP”  
Please enter the resource adapter you would like to update: “mfcobol-xa.rar”  
Would you like to change the value of ServerHost? (y/n/reset to default x to exit): “n”

Would you like to change the value of ServerPort? (y/n/reset to default x to exit): "n"

Would you like to change the value of Trace? (y/n/reset to default x to exit): "x"

Any changes have already been saved. Are you sure you want to exit? Please enter y to confirm: "y"

```
C:\javaee>ravaluesupdater.bat
Your available application servers are:
ibmwebsphere855
ibmwebsphere90
ibmwebsphereliberty
jboss74EAP
jboss80EAP
oracleweblogic1221
oracleweblogic1411

Please enter the application server you would like to update: jboss80EAP
Your available resource adapters are:
mfcobol-localtx.rar
mfcobol-notx.rar
mfcobol-xa.rar

Please enter the resource adapter you would like to update: mfcobol-xa.rar
ServerHost is currently set to: localhost (Default: localhost)
Would you like to change the value of ServerHost? (y/n/reset to default x to exit)
n
ServerPort is currently set to: 9003 (Default: 9003)
Would you like to change the value of ServerPort? (y/n/reset to default x to exit)
n
Trace is currently set to: false (Default: false)
Would you like to change the value of Trace? (y/n/reset to default x to exit)
x
Any changes have already been saved. Are you sure you want to exit? Please enter
y to confirm:
y

C:\javaee>
```

javaee7¥jboss80EAP¥mfcobol-xa.rar が更新されます。

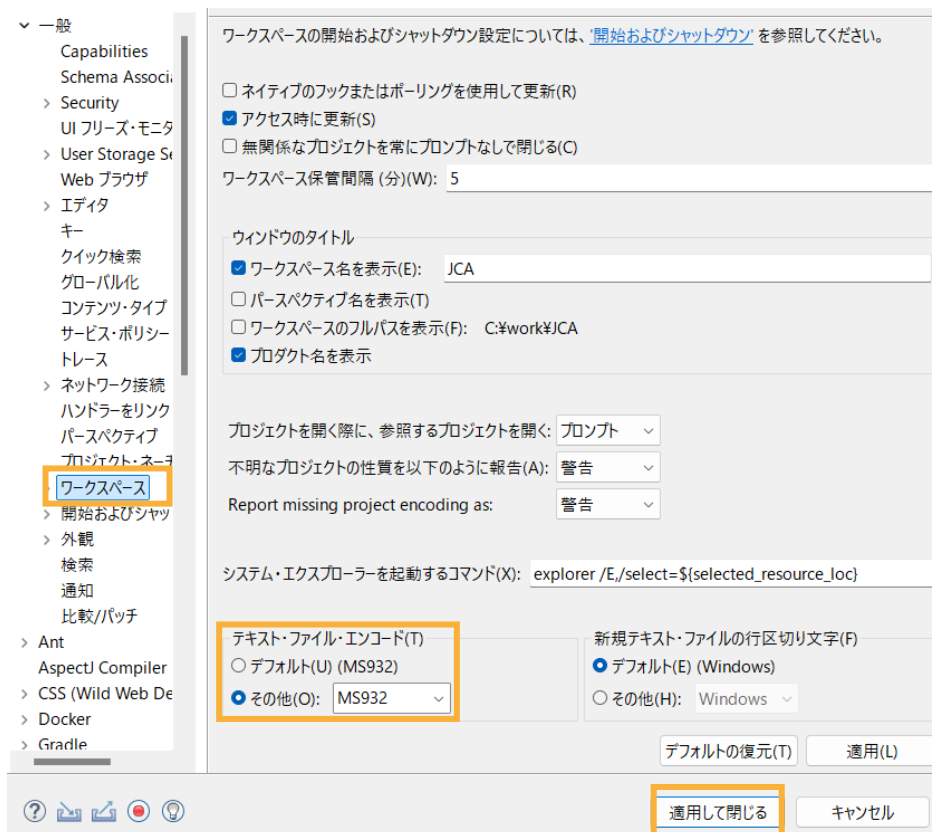
### 3.3 Visual COBOL for Eclipse を起動してソースコードのインポート

#### 1) Visual COBOL for Eclipse を起動

- ① [スタート] メニュー > [Rocket Visual COBOL] > [Visual COBOL for Eclipse] を選択します。
- ② ワークスペースの選択画面にて 任意の作業フォルダーを指定します。ここでは “C:\work\JCA” を指定します。

#### 2) 文字コードの指定

- ① Shift-JIS を指定して日本語を表示する場合、文字コードの指定を明確に行う必要があります。最初に、[Window(W)] メニュー > [設定(P)] より [一般] > [ワークスペース] とナビゲートし、[テキスト・ファイル・エンコード]に “MS932”が設定されているかを確認してください。異なる値が設定されている場合は、“MS932” を選択して [適用して閉じる] をクリックします。



補足)

環境によっては、Windows-31J と表示されることがあります。その場合は、MS932 を Windows-31J に読み替えてください。

※Preference Recorder のダイアログが表示されたら [キャンセル] を選択してください。

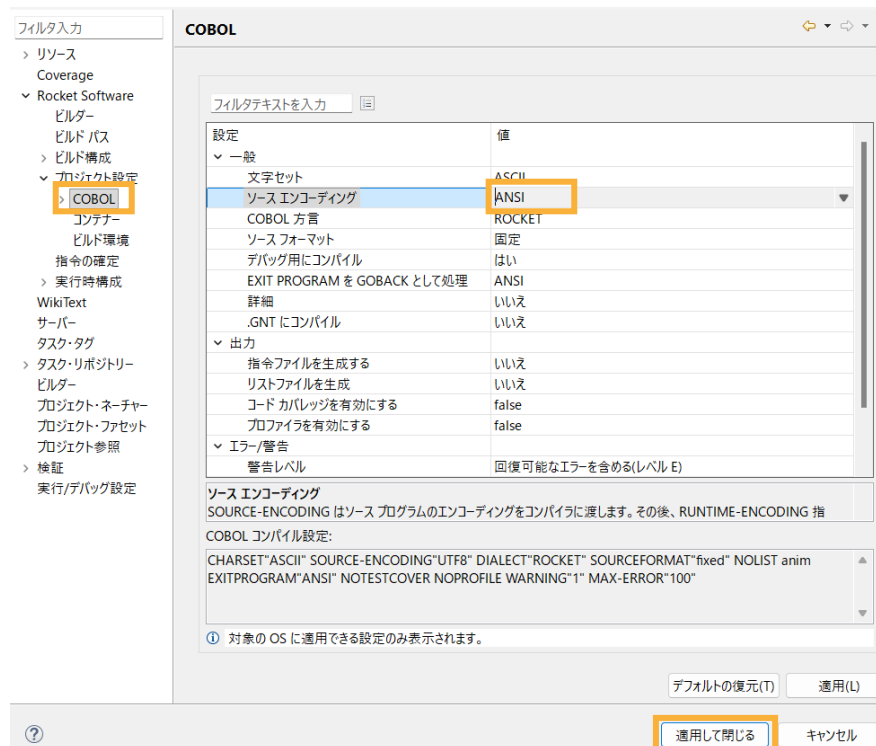


### 3) ネイティブ COBOL プロジェクトの作成

- ① [ファイル(F)] メニュー > [新規(N)] > [COBOL プロジェクト] を選択し、プロジェクト名に “NativeCOBOL” を入力し、プロジェクトテンプレートに「64 ビット」を指定して、[終了(F)]を押します。

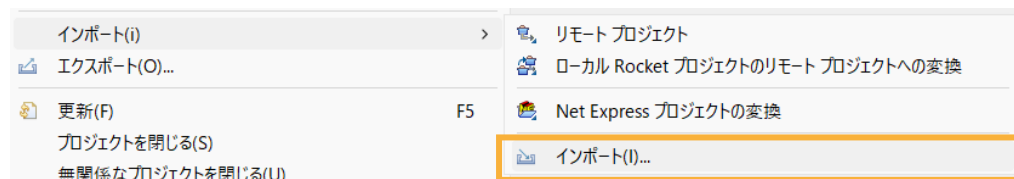
なお、“ようこそ” の画面が開いている場合は閉じてください。

- ② 次に作成した COBOL プロジェクトを選択した状態で、マウスの右クリックにてコンテキストメニューを表示し、[プロパティ(R)] を選択します。[Rocket Software] > [プロジェクト構成] > [COBOL] とナビゲートし、[一般] > [ソース エンコーディング]を“UTF-8”から“ANSI”に変更し、[適用して閉じる]ボタンをクリックします。



### 4) プログラムソースのインポート

- ① COBOL エクスプローラービューにて NativeCOBOL プロジェクトフォルダーを右クリックし、コンテキストメニューから [インポート(i)] > [インポート(I)] を選択します。



- ② サンプルファイルをインポートします。一般のフォルダーを展開し、[ファイル・システム] を選択し、[次へ(N)] ボタンを押します。

## 選択

ローカル・ファイル・システムから既存のプロジェクトヘリソースをインポートします。



インポート・ウィザードの選択(S) :

フィルタ入力

- 一般
  - アーカイブ・ファイル
  - ファイル・システム**
  - フォルダーまたはアーカイブ由来のプロジェクト
  - 既存プロジェクトをワークスペースへ
  - 設定
- EJB
- Git
- Gradle

- ③ インポートダイアログが表示されるので [参照(R)] をクリックしてソースコードを解凍したフォルダーを指定します。ここでは“C:¥pgm”を指定しています。配下の「PSQLTESTX.cbl」と「PSQLTESTXE.cbl」を指定し、[終了(F)]を押します。

## ファイル・システム

ローカル・ファイル・システムからリソースをインポートします。



次のディレクトリーから(Y): c:¥pgm

☒ PSQLTESTX.cbl  
☒ PSQLTESTXE.cbl  
☐ setup\_database.sql

インポート先フォルダ(L): NativeCOBOL

オプション

☐ 警告を出さずに既存リソースを上書き(O)  
☐ トップ・レベルのフォルダーを作成(C)

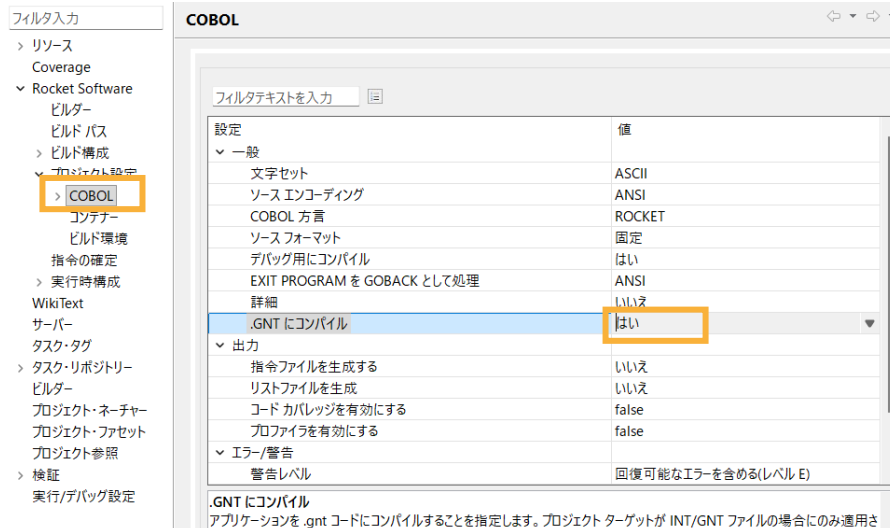
- ④ プロジェクトフォルダーを展開し、2つのファイルが正常にロードされていることを確認します。

COB... x プロジ... Appl...

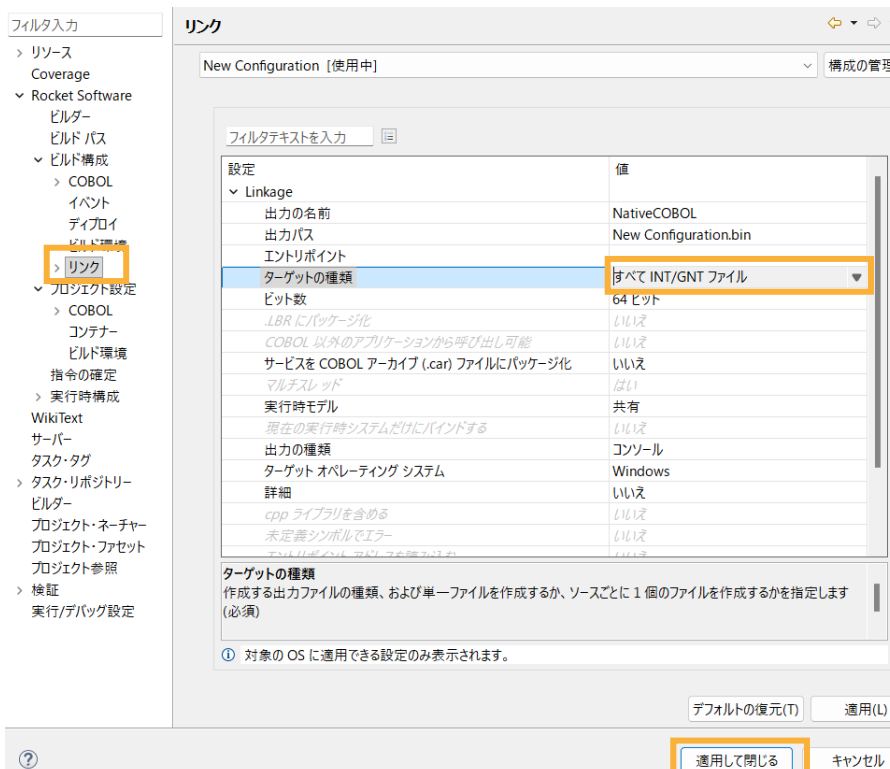
- NativeCOBOL
  - COBOL プログラム
    - PSQLTESTX.cbl
    - PSQLTESTXE.cbl
    - New\_Configuration.bin

## 5) ビルドオプションの変更

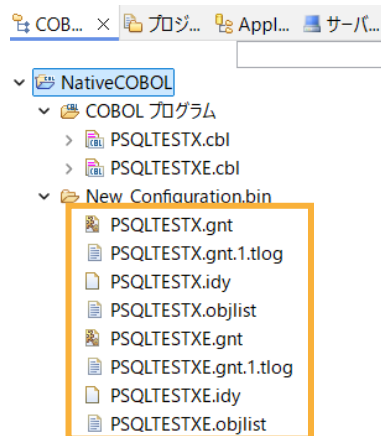
- ① COBOL エクスプローラーにて作成した NativeCOBOL プロジェクトを右クリックし、[プロパティ(R)] を選択します。
- ② プロパティ設定ダイアログが表示されます。[Rocket Software] > [プロジェクト設定] > [COBOL] を選択、[.GNT にコンパイル] を “いいえ” から “はい” に変更したうえで、[適用(L)] をクリックします。



- ③ [Rocket Software] > [ビルド構成] > [リンク] を選択し、[ターゲットの種類] を「すべて INT/GNT ファイル」に変更し、[適用して閉じる] をクリックします。



- ④ COBOL エクスプローラーにて“New\_Configuration.bin”を展開して下記のファイルが作成されていることを確認します。



### 3.4 JCA 用のアプリケーションプロファイルの作成

#### 1) EJB/JCA アプリケーションプロファイルの作成

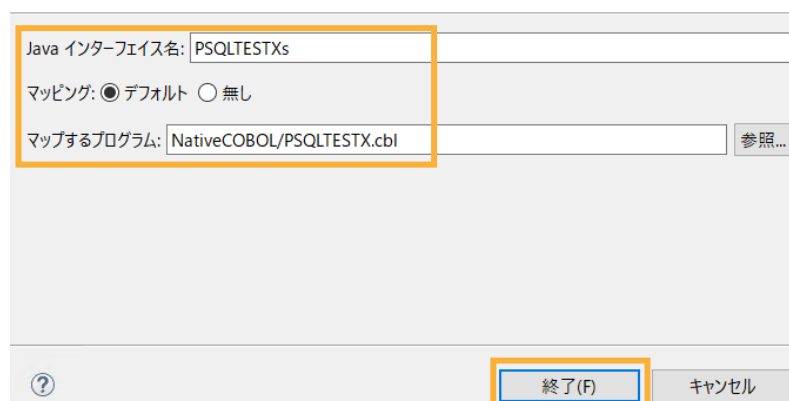
- ① EJB/JCA アプリケーションとして利用する「PSQLTESTX.cbl」を右クリックし、コンテキストメニューから [新規作成 (N)] > [Java インターフェイス] を選択します。



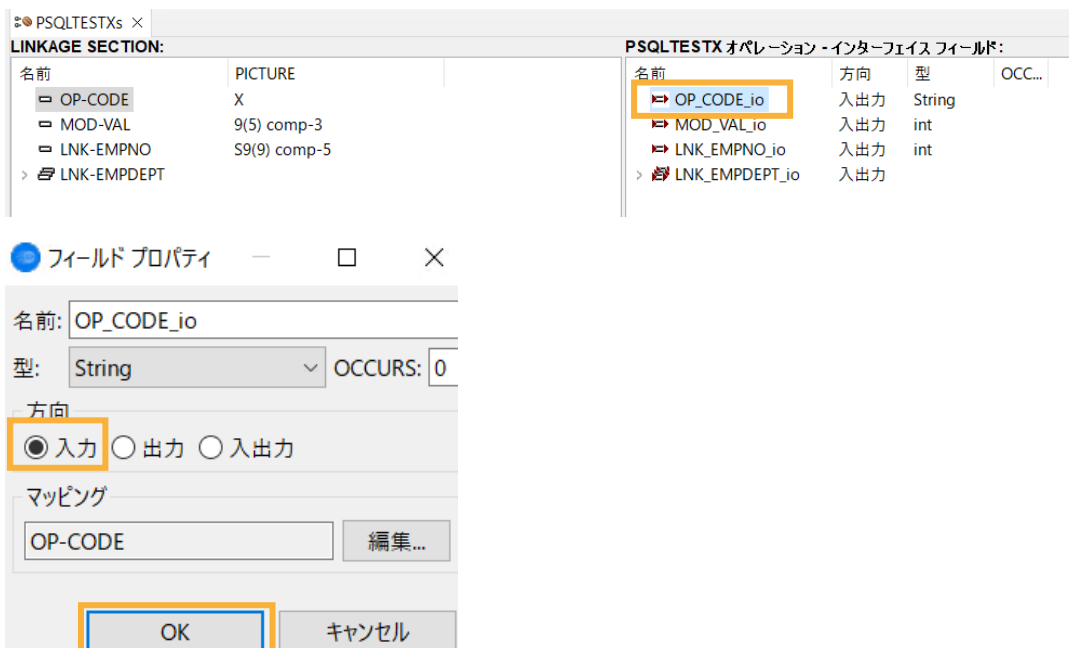
- ② [Java インターフェイス名] 欄には "PSQLTESTXs" を入力し、[マッピング] 欄は "デフォルト" を選択し、マッピングするプログラムは "PSQLTESTX.cbl" が選択されていることを確認し [終了(F)] を押します。

#### Java インターフェイスの新規作成

このページで Java インターフェイスを新規作成します



- ③ [PSQLTESTXs オペレーション・インターフェイス フィールド] 欄の [OP\_CODE\_io] をダブルクリックしたうえで、[方向] に “入力” を選択し [OK] をクリックします。



LINKAGE SECTION:

名前	PICTURE
OP-CODE	X
MOD-VAL	9(5) comp-3
LNK-EMPNO	S9(9) comp-5
LNK-EMPDEPT	

PSQLTESTX オペレーション・インターフェイス フィールド:

名前	方向	型	OCC...
OP_CODE_io	入出力	String	
MOD_VAL_io	入出力	int	
LNK_EMPNO_io	入出力	int	
LNK_EMPDEPT_io	入出力		

フィールド プロパティ

名前: OP\_CODE\_io

型: String OCCURS: 0

方向: ☒ 入力 ☐ 出力 ☐ 入出力

マッピング: OP-CODE 編集...

OK キャンセル

[OP\_CODE\_io] の方向が “入力” になったことを確認します。

PSQLTESTX オペレーション・インターフェイス フィールド:

名前	方向	型	OCC...
OP_CODE_io	入力	String	
MOD_VAL_io	入出力	int	
LNK_EMPNO_io	入出力	int	
LNK_EMPDEPT_io	入出力		

- ④ 上記と同じ手順で、以下の項目の方向を変更し、保存します。

名前	方向
MOD_VAL_IO	入力
LNK_EMPNO_io	入力
LNK_EMPDEPT_io	出力

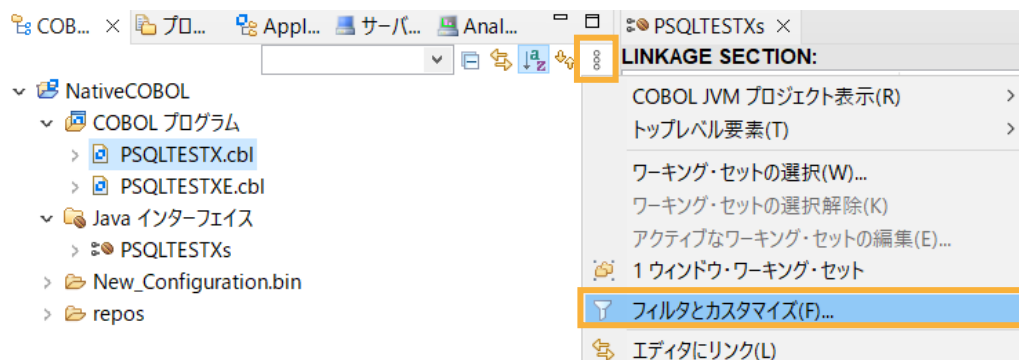
変更後は以下のようになります。

PSQLTESTX オペレーション - インターフェイス フィールド:			
名前	方向	型	OCC...
OP_CODE_io	入力	String	
MOD_VAL_io	入力	int	
LNK_EMPNO_io	入力	int	
LNK_EMPDEPT_io	出力		
LNK_ENAME_io		String	
LNK_JOB_io		String	
LNK_SAL_io		BigDeci...	
LNK_DNAME_io		String	

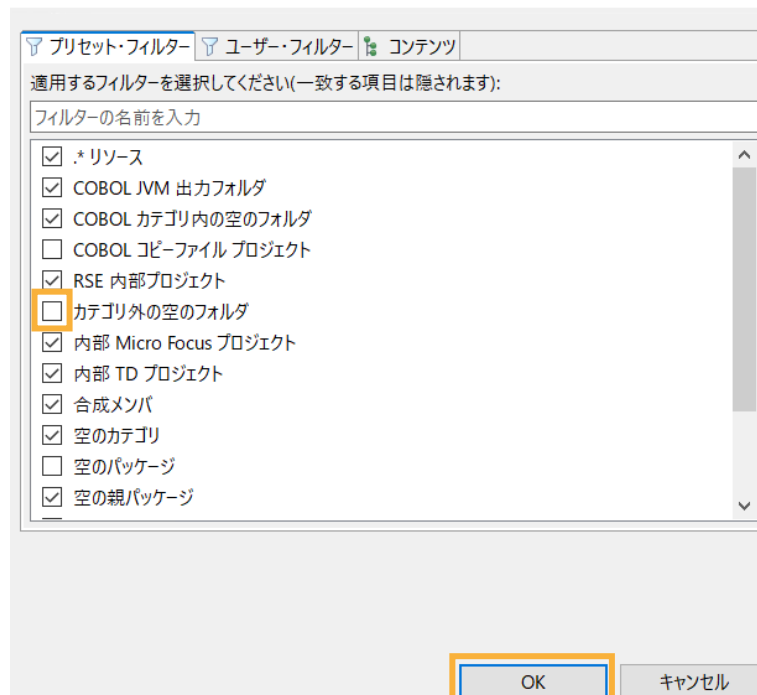
### 3.5 デイプロイ先の変更

#### 1) COBOL エクスプローラー表示設定の変更

- ① COBOL エクスプローラー右上の「↓ ↑」アイコンの右横にあるアイコンをクリックし、[フィルタとカスタマイズ(F)] を選択します。



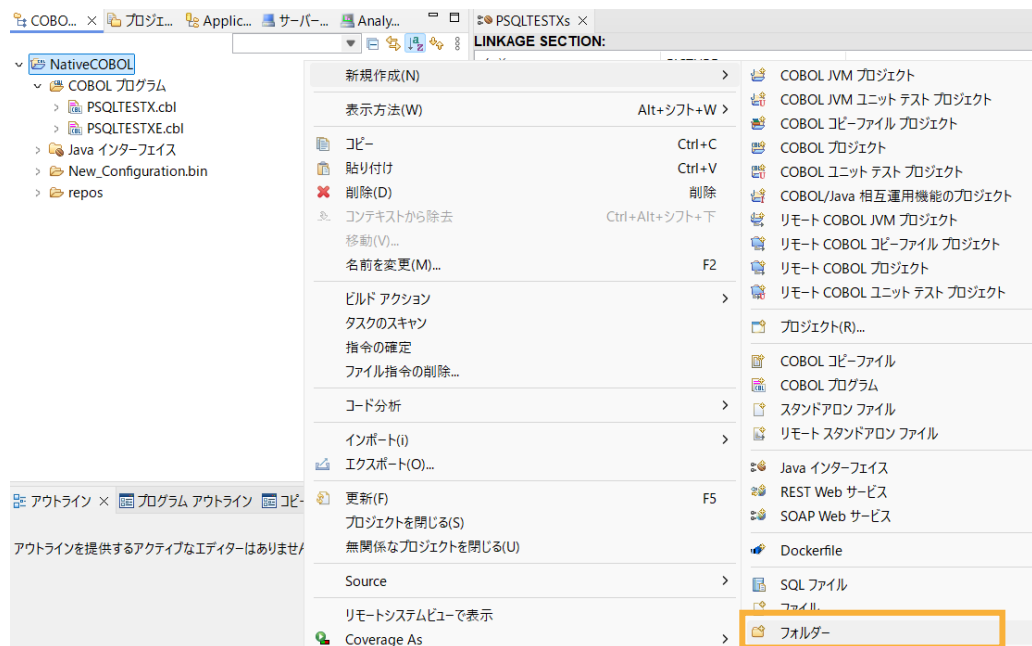
- ② [カテゴリ外の空のフォルダ] にチェックされている場合は、チェックを外したのち、[OK] ボタンをクリックします。



#### 2) デイプロイ用フォルダーの作成

- ① 「NativeCOBOL」プロジェクト上で右クリックし、コンテキストメニューから [新規作成(N)]→[フォルダー] を選択し

ます。



- ② 「NativeCOBOL」プロジェクトが選択されていることを確認の上、フォルダー名に “deploy” を指定し、[終了(F)] をクリックします。

### フォルダー

新規のフォルダ・リソースを作成します



親フォルダを入力または選択(E):

NativeCOBOL

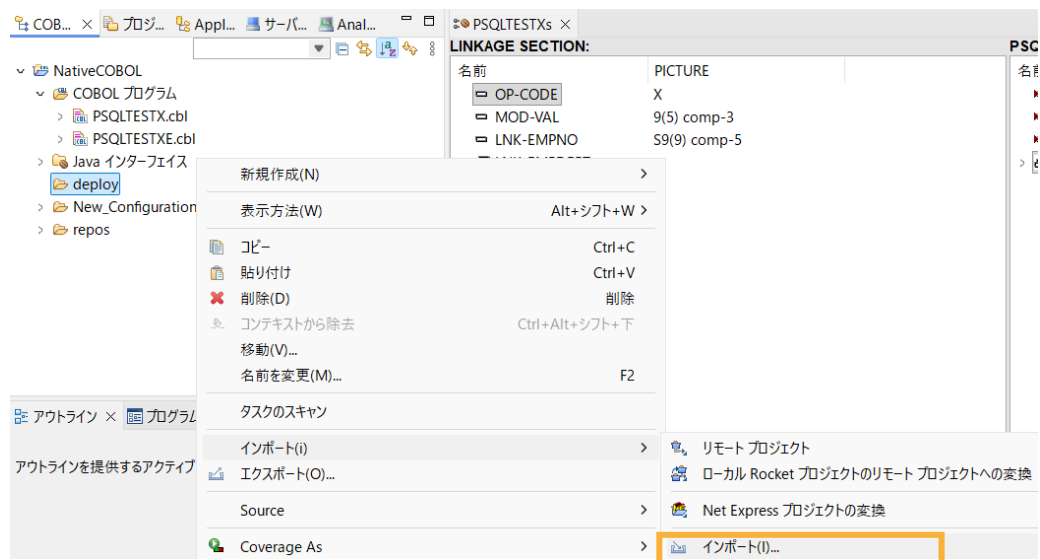
InternalTDProject  
NativeCOBOL  
RemoteSystemsTempFiles

フォルダ名(N):

拡張 >>(A)

終了(F) キャンセル

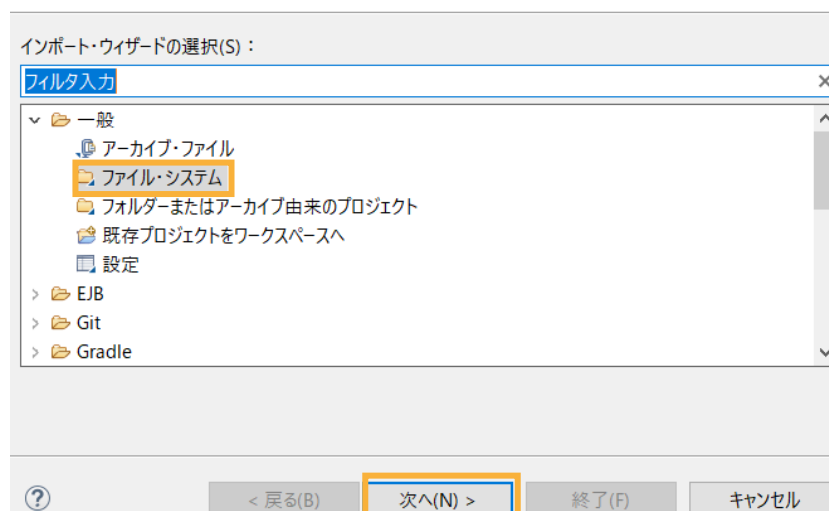
- ③ 作成した「deploy」フォルダー上で右クリックし、コンテキストメニューから [インポート] > [インポート] を選択します。



- ④ [一般] 配下の [ファイル・システム] を選択し、[次へ(N)] をクリックします。

#### 選択

ローカル・ファイル・システムから既存のプロジェクトヘリソースをインポートします。



- ⑤ [参照(R)] をクリックし、<製品のインストールフォルダー¥deploy> を選択し、[.mfdeploy] ファイルにチェックしたうえで、[終了(F)] をクリックします。



## ファイル・システム

ローカル・ファイル・システムからリソースをインポートします。



次のディレクトリーから(Y): C:\Program Files (x86)\Micro Focus\Visual COBOL\deploy

参照(R)...

<input checked="" type="checkbox"/> deploy	<input checked="" type="checkbox"/> .mfdeploy <input type="checkbox"/> web-deploy.html
--	---

タイプをフィルター(T)...    すべて選択(S)    選択をすべて解除(D)

インポート先フォルダ(L): NativeCOBOL/deploy    参照(W)...

オプション

☐ 警告を出さずに既存リソースを上書き(O)

☐ トップ・レベルのフォルダーを作成(C)

拡張 >>(A)

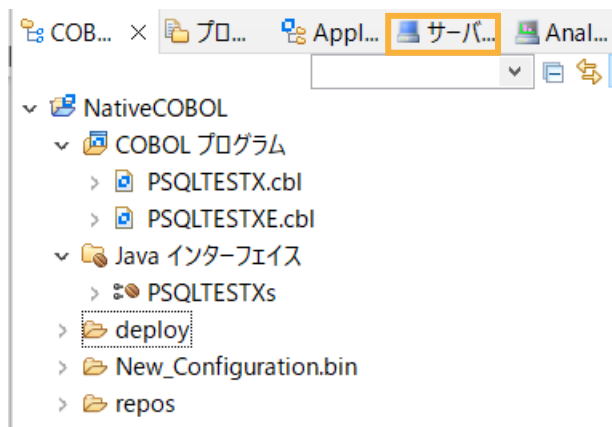
### 補足)

上記手順完了後も COBOL エクスプローラー上の NativeCOBOL\deploy フォルダ配下に .mfdeploy ファイルは表示されませんが、これはフィルタによるものです。

正しくインポートされたことを確認する場合は、さきほど同様、[フィルタとカスタマイズ] を選択し [.\* リソース] のチェックを外すことで表示されるようになります。しかし、リソースに関する設定情報が表示されるようになるため、通常はチェックを行い、非表示状態とすることを推奨します。

## 3) Enterprise Server インスタンスの設定

- ① Eclipse に戻り、[サーバーエクスプローラー] を開きます。



- ② [+] アイコンを押します。

### 補足)

すでに作成済みの場合は、②～③はスキップしてください。



- ③ 以下の入力を行い、[終了(F)] をクリックします。

名前： “ローカル”

サーバアドレス(IPv4/ホスト名)： “localhost”

### 接続の新規作成

既存の Enterprise Server Common Web Administration インスタンスへの接続を新規作成します



名前:

サーバアドレス (IPv4/ホスト名):

サーバポート:

☐ TLS 有効

TLS 設定

CA 証明書:

[参照...](#)

ESCWA のセキュリティが有効な場合は、以下のダイアログが表示されますので、以下の入力を行い、[OK] をクリックします。

ユーザー名 / パスワード： 設定した情報

[認証情報の保存] にチェック

ESCWA: ローカル の接続の詳細を入力してください

☒ サーバーに認証情報が必要

認証情報

ユーザー名:

パスワード:

☒ 認証情報の保存

☐ 認証情報がクリアされるまで、再度プロンプトを表示しない

補足)

インストール直後に有効となっている際の認証情報は以下の手順で確認できます。

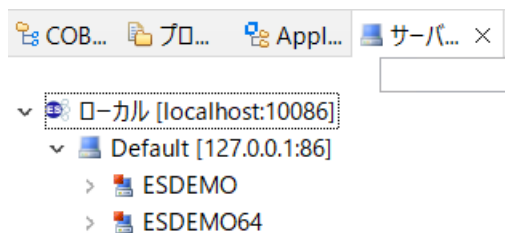
スタートメニューより、[Rocket Visual COBOL] > [Visual COBOL Command Prompt (64-bit)] を選択します。

コマンドライン上で “mfsecretsadmin read microfocus/temp/admin” のコマンドを実行します。

以下の場合、ユーザー名は “SYSAD”、パスワードは “LZQe4VS3” です。

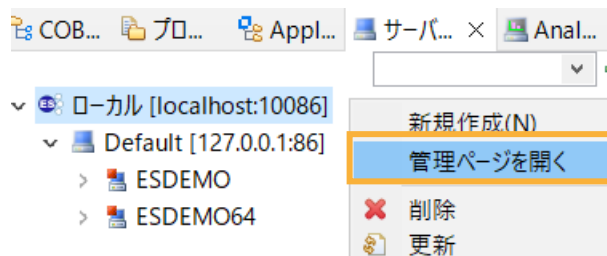
```
C:\xa>mfsecretsadmin read microfocus/temp/admin
{"mfUser":"SYSAD", "mfPassword":"LZQe4VS3"}
C:\xa>
```

Enterprise Server インスタンスが追加されます。

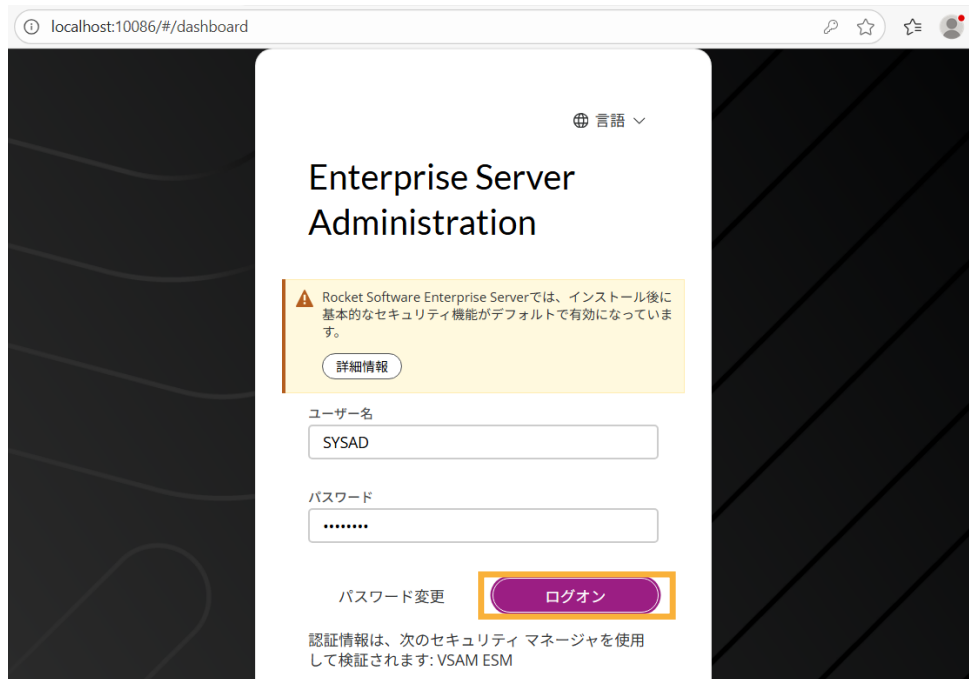


#### 4) ESCWA 管理画面の表示

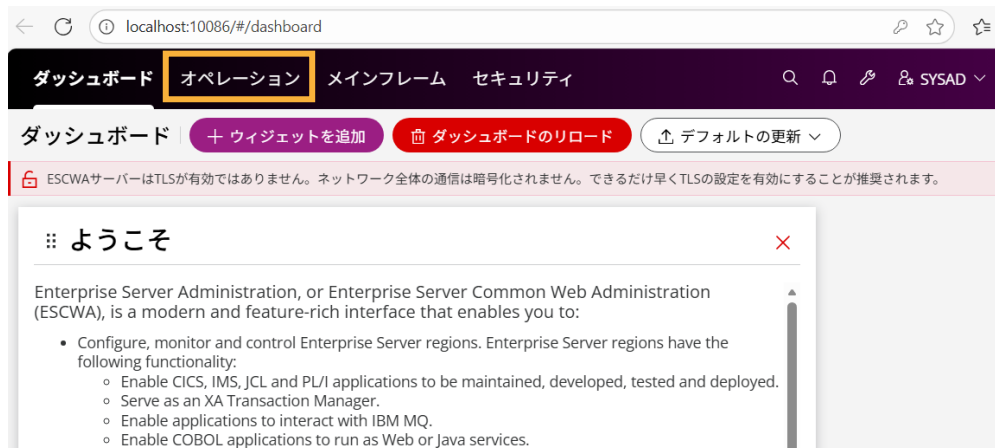
- ① [ローカル] > [Default] を選択したうえで、マウスの右クリックでコンテキストメニューを開き、[管理ページを開く] を選択します。



- ② ブラウザーが開いたら、さきほどの認証情報を入力して、[ログオン] をクリックします。



ログインに成功すると、ESCWA (Enterprise Server Common Web Administration) 画面にてダッシュボードが表示されますので、[オペレーション] をクリックします。



## 5) 不要ログの抑止

- ① ダッシュボード画面左側より、[Directory Server] > [Default] をクリックし、上部のメニューより [プロパティ] をクリックします。



- ② 以下の入力を行い、[適用] をクリックします。

モニター > 有効のチェックをはずす

## モニター

☐ 有効 ⓘ

キープアライブ間隔\* ⓘ  
 60 ^ 秒

許容応答時間\* ⓘ  
 5 ^ 秒

このDirectory ServerホストではTLSが有効ではありませんが、ループ

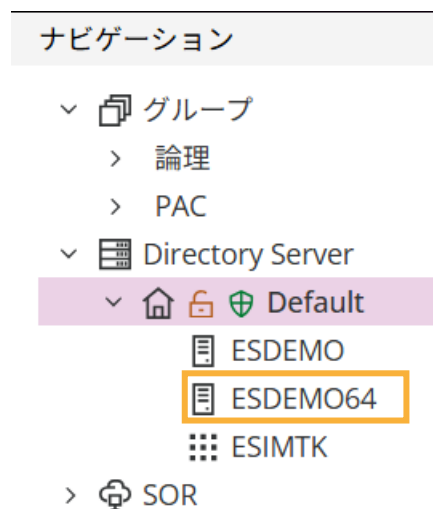
Directory ServerにMTO ライセンスがありません。ライセンス状態を

Directory Serverの構成

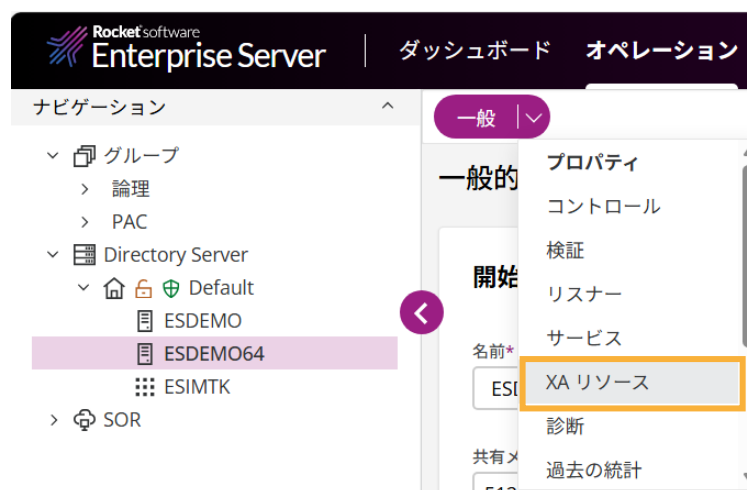
適用

## 6) XA スイッチモジュールの構成

- ① ESCWA 画面左のツリーより、[Directory Server] > [Default] > [ESDEMO64] をクリックします。



- ② [一般] をクリックすると表示される [XA リソース] をクリックします。



[新規作成] をクリックします。

一般

XA リソース

\* 新規作成

- ③ 以下の入力を行い、[OK] をクリックします。

ID: "PSQLXA"

名前: "PSQLXA"

モジュール: "c:\xa\ESODBCXA\_D.dll"

OPEN 文字列: "DSN=PostgreSQL"

### XA リソースの構成

ID\*

名前\*

PSQLXA

PSQLXA

モジュール\*

c:\xa\ESODBCXA\_D.dll

☒ 有効

OPEN 文字列

DSN=PostgreSQL

CLOSE 文字列

説明

\* 入力必須の項目です

保存 戻る

補足)

モジュールには、3.2 で作成した ESODBCXA\_D.dll ファイルを絶対パスで指定します。

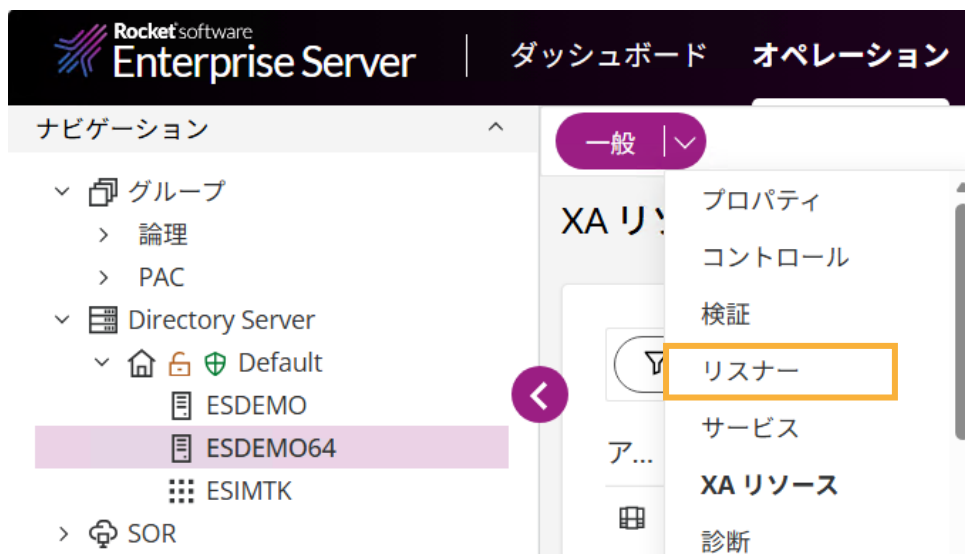
OPEN 文字列の DSN= の後は、3.1 で作成したデータソース名を入力します。

XA リソースが登録されます。



## 7) リスナーの構成

- ESCWA 画面の [一般] をクリックしたのちに表示される [リスナー] を選択します。



- 画面左下より、[通信プロセス 1] > [Web] をクリックします。




- リスナープロパティのカスタム構成情報が表示されます。アプリケーションのディプロイ先は [高度] をクリックして開き、[カスタム設定] 内の uploads 項目で定義されますが、現在は「uploads=<ES>/deploy」となっています。この場合、Visual COBOL インストールディレクトリ配下の deploy フォルダがディプロイ用フォルダとして使用されます。通常、Program Files (x86)等のフォルダは管理者権限を持つユーザーフォルダ書き込みできフォルダ変更を行います。下記のように変更し、画面上部の [適用] をクリックします。

uploads=C:/work/JCA/NativeCOBOL/deploy

補足)

ここで指定しているフォルダパスは、2) で作成した deploy フォルダに対応します。

カスタム構成 

```
[virtual paths]
cgi=<ES>/bin
uploads=C:/work/JCA/NativeCOBOL/deploy
<default>=/dev/null
```

- ④ 画面左下部より、[通信プロセス 1] > [Web Services and J2EE] を選択します。



- ⑤ [ポート] に “9003” を入力して、[適用] をクリックします。

リスナー プロパティ 適用 削除

\* 入力必須の項目です

名前\*   
 ☐ レガシー Micro Focus アプリケーション形式 

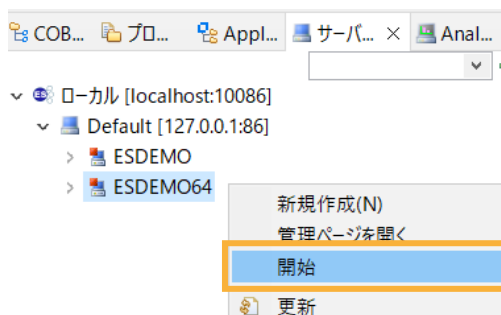
説明 

☐ このエンドポイントはネットワーク経由でアクセス可能になり、TLSが無効になります。

プロトコル\*   ホスト名またはIP アドレス\*   ポート 

## 8) Enterprise サーバーインスタンスの起動

- ① Eclipse に戻り、[サーバーエクスプローラー] を開きます。
- ② [ESDEMO64] を選択したうえでマウスの右クリックでコンテキストメニューを開き、[開始] を選択します。



認証ダイアログが表示された場合は、ESCWA の認証情報を入力します。



サーバー ESDEMO64 の ESMAC 認証情報を入力してください

☒ サーバーに認証情報が必要

認証情報

ユーザー名: SYSAD

パスワード: \*\*\*\*\*

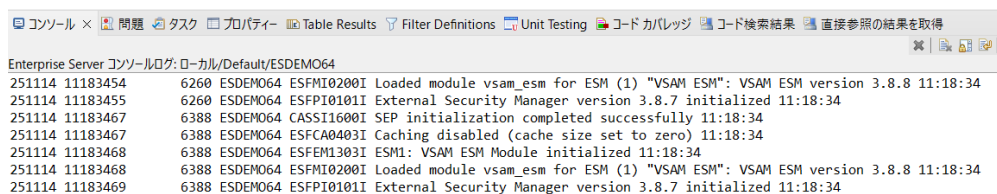
グループ:  デフォルト グループは空白

☒ 認証情報の保存

☐ 認証情報がクリアされるまで、再度プロンプトを表示しない

OK キャンセル

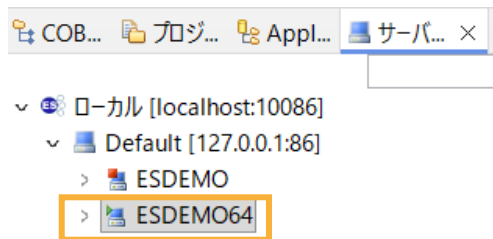
Eclipse の Secure Storage に関するダイアログが表示された場合、[いいえ] を選択してください。開始処理の状況は、[コンソール] ビューでモニターできます。



```

Enterprise Server コンソールログ: ローカル/Default/ESDEMO64
251114 11183454 6260 ESDEMO64 ESFMI0200I Loaded module vsam_esm for ESM (1) "VSAM ESM": VSAM ESM version 3.8.8 11:18:34
251114 11183455 6260 ESDEMO64 ESFPI0101I External Security Manager version 3.8.7 initialized 11:18:34
251114 11183467 6388 ESDEMO64 CASSI1600I SEP initialization completed successfully 11:18:34
251114 11183467 6388 ESDEMO64 ESFCA0403I Caching disabled (cache size set to zero) 11:18:34
251114 11183468 6388 ESDEMO64 ESFEM1303I ESM1: VSAM ESM Module initialized 11:18:34
251114 11183468 6388 ESDEMO64 ESFMI0200I Loaded module vsam_esm for ESM (1) "VSAM ESM": VSAM ESM version 3.8.8 11:18:34
251114 11183469 6388 ESDEMO64 ESFPI0101I External Security Manager version 3.8.7 initialized 11:18:34
  
```

正常に起動しますと、緑のアイコンになります。



## 9) JBoss ヘリソースアダプタのデプロイ

- ① 構成ファイル「standalone.xml」の変更を行います。

<JBoss インストールフォルダー>%standalone%configuration%standalone.xml をエディターで開き、下記手順に従って、mfcobol-xa.rar の設定を行います。

<https://www.amc.rocketsoftware.co.jp/manuals/VC110/Eclipse/vc110indx.html>

[デプロイ] > [Enterprise Server へのデプロイ] > [モダナイズ済みのアプリケーションのデプロイおよび構成]  
> [EJB およびリソース アダプターのデプロイ] > [EJB のデプロイ - 概要] > [JBoss へのデプロイ]

- ② JBoss アプリケーションサーバーを Standalone モードで起動します。

コンソールログからエラーがないことを確認してください。

- ③ <JBoss インストールフォルダー>%standalone%deployments 配下に「リソースアダプターファイルの構成」で作成された mfcobol-xa.rar をコピーします。

コンソールログから mfcobol-xa.rar のデプロイがエラーとなっていないことを確認します。

```

11:37:43,507 INFO [org.jboss.as.repository] (ServerService Thread Pool -- 74) WFLY
DR0001: ロケーション C:%EAP81%standalone%data%content%76%272d06103a2827f65f7b781
328f27954431c53%content にコンテンツが追加されました。
  
```

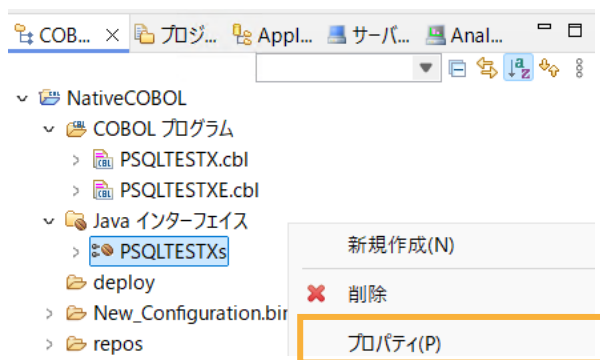
```

11:37:43,536 INFO [org.jboss.as.server.deployment] (MSC service thread 1-8) WFLY
SRV0027: "mfcobol-xa.rar" (runtime-name: "mfcobol-xa.rar") のデプロイメントを開始しました。
11:37:43,951 INFO [org.jboss.as.connector.deployment] (MSC service thread 1-8) W
FLYJCA0007: 接続ファクトリ java:/eis/MFCobol_v1.5 を登録しました
11:37:43,953 WARN [org.jboss.as.connector.deployers.RaXmlDeployer] (MSC service t
hread 1-8) IJ020016: Missing <recovery> element. XA recovery disabled for: java:/ei
s/MFCobol_v1.5
11:37:43,961 INFO [org.jboss.as.connector.deployers.RaXmlDeployer] (MSC service th
read 1-8) IJ020002: Deployed: file:/C:/EAP81/standalone/tmp/vfs/temp/temp41ab2a5
7230ac14e/content-e0d38a95185794f6/contents/
11:37:43,962 INFO [org.jboss.as.connector.deployment] (MSC service thread 1-3) W
FLYJCA0002: Jakarta Connectors ConnectionFactory [java:/eis/MFCobol_v1.5] をバインド
しました
11:37:44,029 INFO [org.jboss.as.server] (ServerService Thread Pool -- 74) WFLYSRV
0010: "mfcobol-xa.rar" (runtime-name : "mfcobol-xa.rar") をデプロイしました。

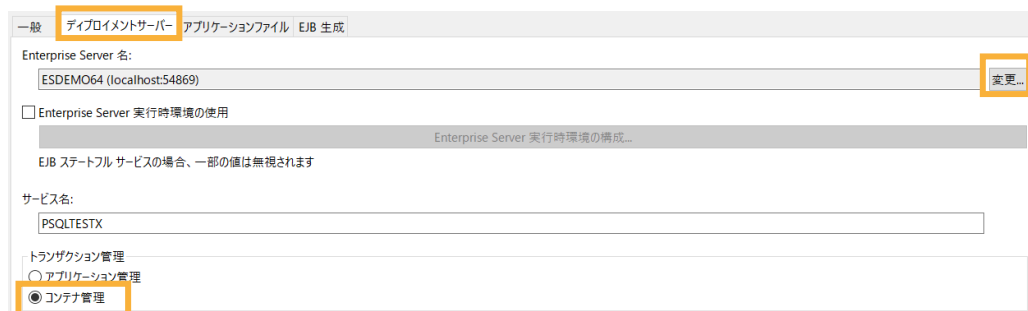
```

#### 10) JCA アプリケーションのデプロイ

- ① Eclipse にもどり、COBOL エクスプローラー上から NativeCOBOL プロジェクト配下の [Java インターフェイス] > [PSQLTESTXs] を選択、マウスの右クリックでコンテキストメニューを開き、[プロパティ(P)] を選択します。



- ② [デプロイメントサーバー] タブを選択し、以下の作業を行います。  
Enterprise Server 名 : [変更] をクリックして、[ESDEMO64] を選択  
トランザクション管理 : “コンテナ管理” を選択



- ③ [アプリケーションファイル] を選択し、以下の作業を行います。  
[レガシーアプリケーションをデプロイします] を選択  
[ファイル追加] をクリックし、C:\¥work¥JCA¥NativeCOBOL¥New\_Configuration.bin 配下の「PSQLTESTX.gnt」と「PSQLTESTX.idy」を選択

一般 デプロイメントサーバー アプリケーションファイル EJB 生成

レガシーアプリケーションをデプロイ済みか、またはサーバーにデプロイする必要があるかを指定してください。

☐ レガシーアプリケーションは既にデプロイ済み

デプロイされたアプリケーションのパス:

☒ レガシーアプリケーションをデプロイする

アプリケーションファイル:

New\_Configuration.bin/PSQLTESTX.gnt  
New\_Configuration.bin/PSQLTESTX.idy

ファイル追加  
ファイル削除

④ [EJB 生成] を選択し、以下の作業を行ったうえで [OK] をクリックします。

アプリケーションサーバー: “JEE 7” / “JBoss EAP 8.0” を選択

Java コンパイラ: 使用している Java コンパイラへのパス

Java EE クラスパス: 以下の jar ファイルを指定

いずれも <JBoss EAP インストールフォルダー>%modules%system%layers%base%jakarta

- annotation/api/main/jakarta.annotation-api-2.1.1.redhat-00003.jar
- ejb/api/main/jakarta.ejb-api-4.0.1.redhat-00001.jar
- resource/api/main/jakarta.resource-api-2.1.0.redhat-00001.jar
- servlet/api/main/jakarta.servlet-api-6.0.0.redhat-00006.jar

一般 デプロイメントサーバー アプリケーションファイル EJB 生成

アプリケーションサーバー: JEE 7 JBoss EAP 8.0

トランザクション可能

EJB 属性

Bean 名: PSQLTESTXs  
パッケージ名: com.mypackage.PSQLTESTXs  
セッション永続性: ☐ スタートレス ☒ スタートフル  
インターフェイスタイプ: ☒ ローカル ☐ リモート

SEP 属性

SEP: ☐ スタートレス ☒ スタートフル

デプロイメントディスクリプタ属性

EJB 名: PSQLTESTXsEJB  
アーカイブ名: PSQLTESTXs

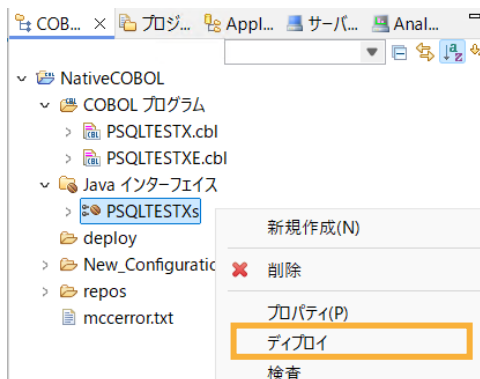
Java SE と Java EE の属性

Java コンパイラ: C:\Program Files (x86)\Rocket Software\Visual COBOL\AdoptOpenDKWbin 参照...

EJB, コネクタ (また、クラスパスに追加されるファイルのパスを指定してください): C:\EAP81\modules\system\layers\base\jakarta\annotation\api\main\jakarta.annotation-api-2.1.1.redhat-00003.jar;C:\EAP81\modules\system\layers\base\jakarta\ejb\api\main\jakarta.ejb-api-4.0.1.redhat-00001.jar;C:\EAP81\modules\system\layers\base\jakarta\resource\api\main\jakarta.resource-api-2.1.0.redhat-00001.jar;C:\EAP81\modules\system\layers\base\jakarta\servlet\api\main\jakarta.servlet-api-6.0.0.redhat-00006.jar 参照...

OK キャンセル

⑤ COBOL エクスプローラー上の NativeCOBOL プロジェクト配下の [Java インターフェイス] > [PSQLTESTXs] を選択、マウスの右クリックでコンテキストメニューを開き、[デプロイ] を選択します。



成功すると、[サービス インターフェイス コンソール] ビューでは、以下のログが出力されます。

```

サービス インターフェイス コンソール
0021 (2025/11/14 11:47:39): Using directory at mrpi://127.0.0.1:86

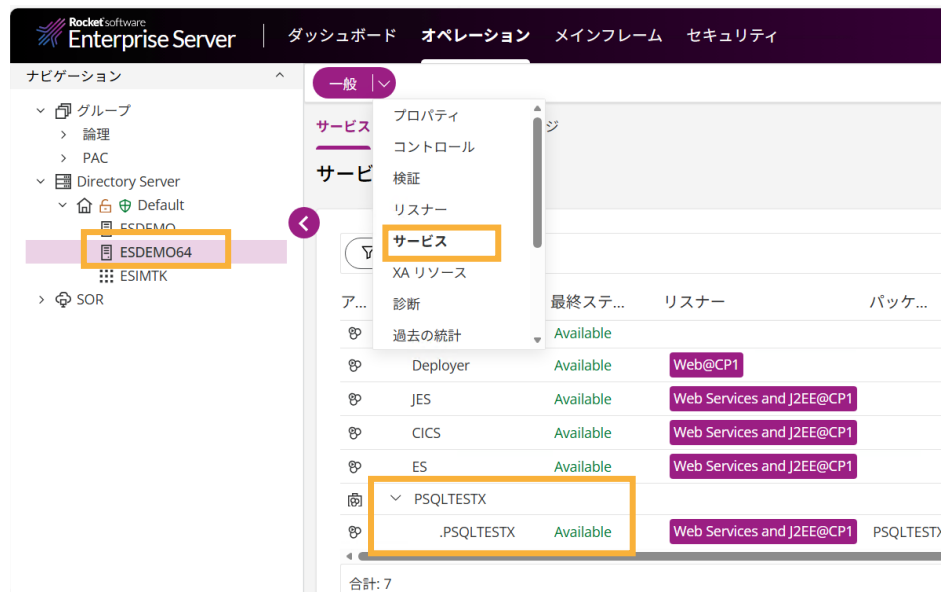
0030 (2025/11/14 11:47:39): ES server "ESDEMO64" notified service "PSQLTESTX.PSQLTESTX" is available

0002 (2025/11/14 11:47:39): Installation of package "PSQLTESTXs.car" finished with 3 warnings

```

また、ESCWA 画面からも以下の手順で確認できます。

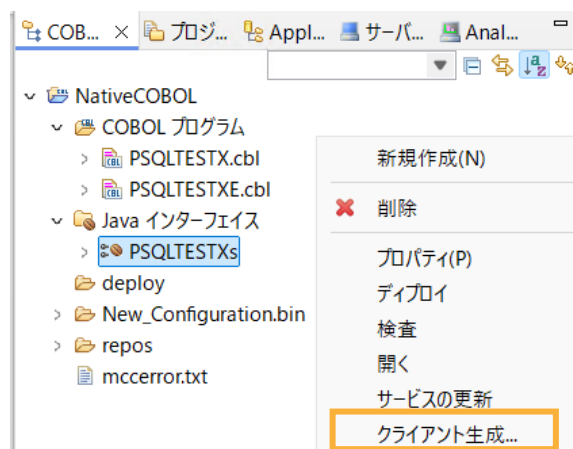
- 左側より [ESDEMO64] を選択
  - [一般] > [サービス] を選択
- PSQLTESTX がサービスとして登録されています。



## 11) テストクライアントアプリケーションの生成

Visual COBOL 製品にはテストクライアントを生成する機能があり、それを用いてテストを行います。

- ① COBOL エクスプローラー上の NativeCOBOL プロジェクト配下の [Java インターフェイス] > [PSQLTESTXs] を選択、マウスの右クリックでコンテキストメニューを開き、[クライアント生成] を選択します。



[サービス インターフェイス コンソール] ビューに結果が出力されます。

サービス インターフェイス コンソール

PSQLTESTXs.jarを追加中です(入=4334)(出=3887)(10%収縮されました)

PSQLTESTXs.warを追加中です(入=21560)(出=20265)(6%収縮されました)

mfejbllib.jarを追加中です(入=11259)(出=10059)(10%収縮されました)

META-INF/application.xmlを追加中です(入=539)(出=263)(51%収縮されました)

META-INF/jboss-deployment-structure.xmlを追加中です(入=519)(出=206)(60%収縮されました)

C:\Program Files (x86)\Rocket Software\Visual COBOL\bin64

クライアント生成が正常終了しました。

- ② COBOL エクスプローラー上の NativeCOBOL プロジェクト配下の repos¥を選択し、<JBoss EAP インストール フォルダー>¥standalone¥deployments にコピーします。

COB... x プロジ... Appl... サーバ... Anal...

NativeCOBOL

- COBOL プログラム
  - PSQLTESTX.cbl
  - PSQLTESTXE.cbl
- Java インターフェイス
  - PSQLTESTXs
- deploy
  - PSQLTESTXs.9fck32f6
  - New\_Configuration.bin
- repos
  - PSQLTESTXs.deploy
    - Client
    - com
    - META-INF
    - mfejbllib.jar
    - PSQLTESTXs.ear
    - PSQLTESTXs.jar
    - PSQLTESTXs.mfmak

JBoss EAP のコンソールに、以下のようなログが出力されます。

```
(省略)
12:00:42,214 INFO [org.jboss.as.server.deployment] (MSC service thread 1-1) WFLY
SRV0207: サブデプロイメントを開始します (runtime-name: "PSQLTESTXs.war")
12:00:42,296 WARN [org.jboss.as.ejb3] (MSC service thread 1-5) WFLYEJB0525: Jak
arta Enterprise Beans アノテーションの 'mappedName' はサポートされていません。Jakarta Enterpri
se Beans 'ejb/PSQLTESTXsEJB' の 'PSQLTESTXsBean' の値は無視されま す。
12:00:42,392 INFO [org.jboss.weld.deployer] (MSC service thread 1-5) WFLYWELD00
03: Weld デプロイメント PSQLTESTXs.ear を処理しています
12:00:42,459 INFO [org.hibernate.validator.internal.util.Version] (MSC service thread
1-5) HV000001: Hibernate Validator 8.0.2.Final-redhat-00001
12:00:42,591 INFO [org.jboss.weld.deployer] (MSC service thread 1-1) WFLYWELD00
03: Weld デプロイメント PSQLTESTXs.war を処理しています
12:00:42,602 INFO [org.jboss.weld.deployer] (MSC service thread 1-2) WFLYWELD00
03: Weld デプロイメント PSQLTESTXs.jar を処理しています
12:00:42,605 INFO [org.jboss.as.ejb3.deployment] (MSC service thread 1-2) WFLYEJ
B0473: デプロイメントユニット 'subdeployment "PSQLTESTXs.jar" of deployment "PSQLTESTX
s.ear"'の 'PSQLTESTXsEJB' という名前のセッション Bean の JNDI バインディ ングは次のとおりです:

    java:global/PSQLTESTXs/PSQLTESTXs.jar/PSQLTESTXsEJB!com.mypackage.PSQL
TESTXs.PSQLTESTXsLocal
    java:app/PSQLTESTXs.jar/PSQLTESTXsEJB!com.mypackage.PSQLTESTXs.PSQLTE
```

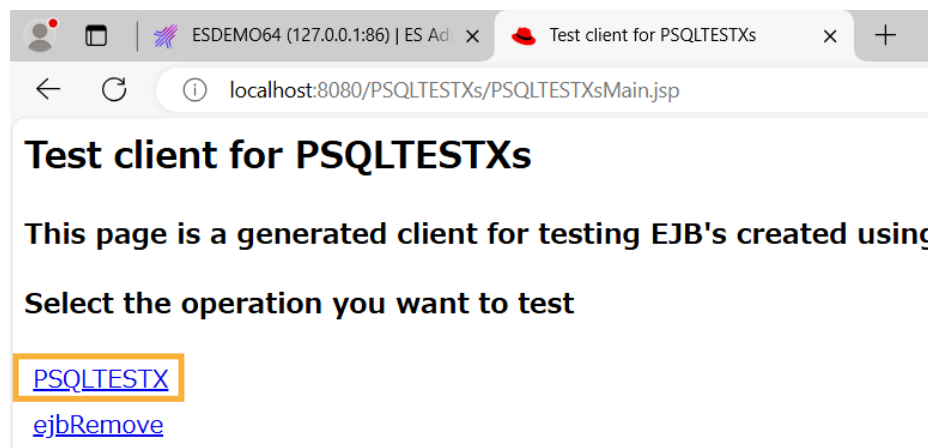
```
STXsLocal
  java:module/PSQLTESTXsEJB!com.mypackage.PSQLTESTXs.PSQLTESTXsLocal
  java:global/PSQLTESTXs/PSQLTESTXs.jar/PSQLTESTXsEJB
  java:app/PSQLTESTXs.jar/PSQLTESTXsEJB
  java:module/PSQLTESTXsEJB

12:00:42,645 INFO  [org.jboss.weld.Version] (MSC service thread 1-8) WELD-000900: 5.1.6 (redhat)
12:00:43,342 INFO  [org.wildfly.extension.undertow] (ServerService Thread Pool -- 14) WFLYUT0021: 登録された web コンテキスト: '/PSQLTESTXs' (サーバー 'default-server' 用)
12:00:43,391 INFO  [org.jboss.as.server] (ServerService Thread Pool -- 74) WFLYSRV0010: "PSQLTESTXs.ear" (runtime-name : "PSQLTESTXs.ear") をデプロイしました。
```

## 12) テストアプリケーションの実行

- ① ブラウザーを開き、以下のアドレスにアクセスし、[PSQLTESTX] リンクをクリックします。

<http://localhost:8080/PSQLTESTXs/PSQLTESTXsMain.jsp>



- ② データを更新します。

サンプルレコードの EMPNO: 7934 の情報は以下の通りです。

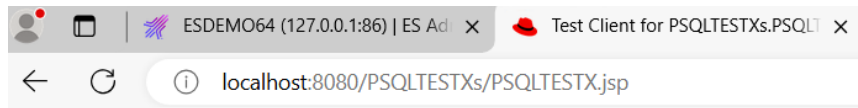
EMPNO	7934
ENAME	CMILLER
JOB	CLEARAK
SAL	1300.00
DEPTNO	3

Web 画面で以下の項目を入力して、[Go!] をクリックします。

PSQLTESTX\_OP\_CODE\_io: "I"

PSQLTESTX\_MOD\_VAL\_io: "150"

PSQLTESTX\_LNK\_EMPNO\_io: "7934"



## Test client for PSQLTESTXs.PSQLTESTX

[Back](#)

Perform the test by entering values:

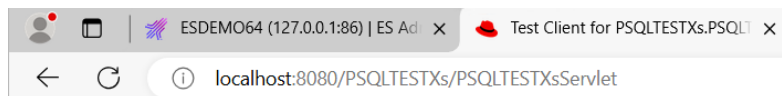
PSQLTESTX\_OP\_CODE\_io :

PSQLTESTX\_MOD\_VAL\_io :

PSQLTESTX\_LNK\_EMPNO\_io :

[Back](#)

結果が表示され、1450.00 になっていることがわかります。



## Test client for PSQLTESTXs.PSQLTESTX

[Back](#)

Perform the test by entering values:

PSQLTESTX\_OP\_CODE\_io :

PSQLTESTX\_MOD\_VAL\_io :

PSQLTESTX\_LNK\_EMPNO\_io :

Result:

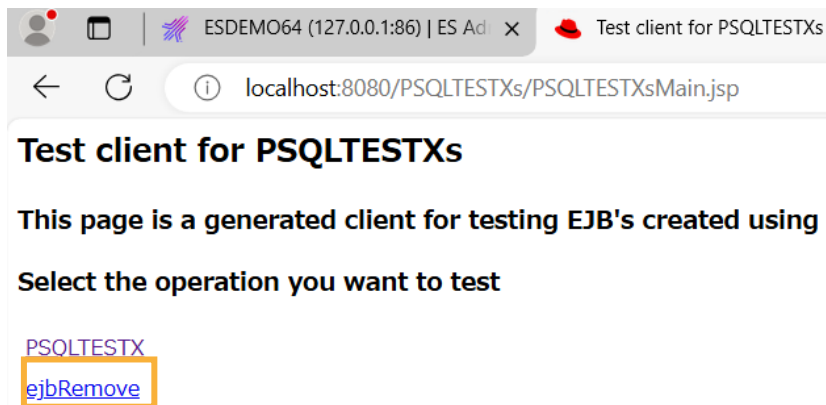
Variable	Value
LNK_EMPDEPT_io.LNK_ENAME_io	MILLER
LNK_EMPDEPT_io.LNK_JOB_io	CLEARK
LNK_EMPDEPT_io.LNK_SAL_io	1450
LNK_EMPDEPT_io.LNK_DNAME_io	ACCOUNTING

[Back](#)

実際のデータベースのレコードも 1450.00 に更新されています。

- ③ EJB の削除処理を行います。

画面下部に表示されている [Back] リンクをクリックし、[ejbRemove] リンクをクリックします。

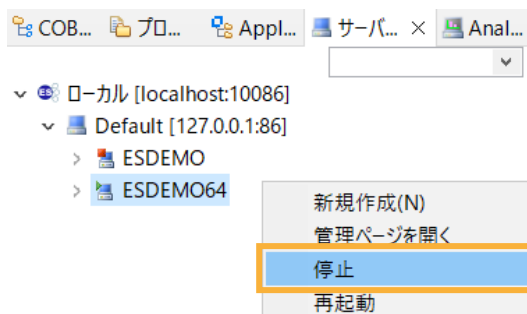


[Go!] をクリックし、ブラウザを閉じます。



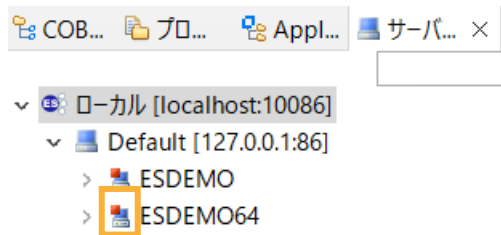
### 13) JBoss EAP, Enterprise Server インスタンスの停止

- ① JBoss アプリケーションサーバーを停止します。
- ② Eclipse に戻り、[サーバー エクスプローラー] を開き、[ESDEMO64] を選択、マウスの右クリックでコンテキストメニューを開き、[停止] をクリックします。



停止すると、赤いアイコンが表示されます。





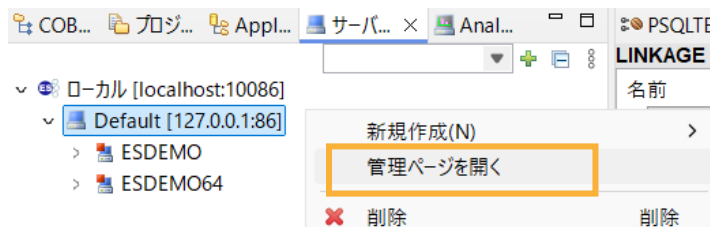
### 3.6 ディプロイしたアプリケーションのデバッグ

#### 1) JBoss EAP の起動

- ① JBoss アプリケーションサーバーを起動します。  
コンソールログからエラーがないことを確認してください。

#### 2) Enterprise Server インスタンスのデバッグ設定

- ① Eclipse に戻り、[サーバーエクスプローラー] を開きます。
- ② [Default [127.0.0.1:86]] を選択したうえで、マウスの右クリックでコンテキストメニューを開き、[管理ページを開く] を選択します。



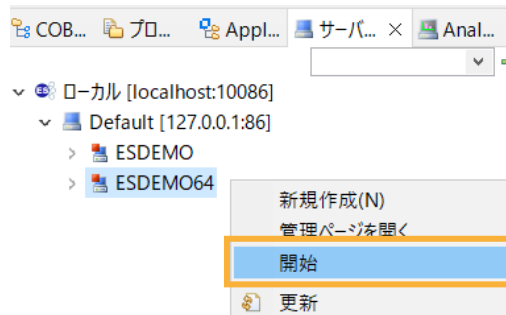
ブラウザーが開いたら、さきほどの認証情報を入力して、[ログオン] をクリックします。

- ③ 左側より、[Directory Server] > [Default] > [ESDEMO64] を選択し、[動的デバッグを許可] にチェックを行ったうえで、[適用] をクリックします。



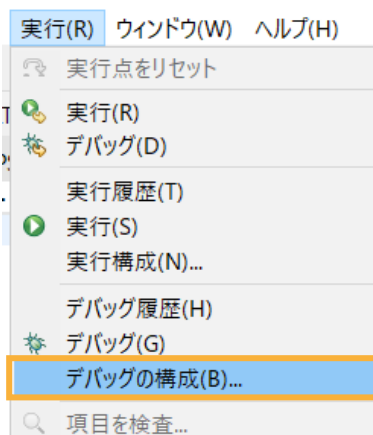
#### 3) Enterprise Server インスタンスの起動

- ① [ESDEMO64] を選択したうえでマウスの右クリックでコンテキストメニューを開き、[開始] を選択します。



#### 4) アプリケーションのデバッグ

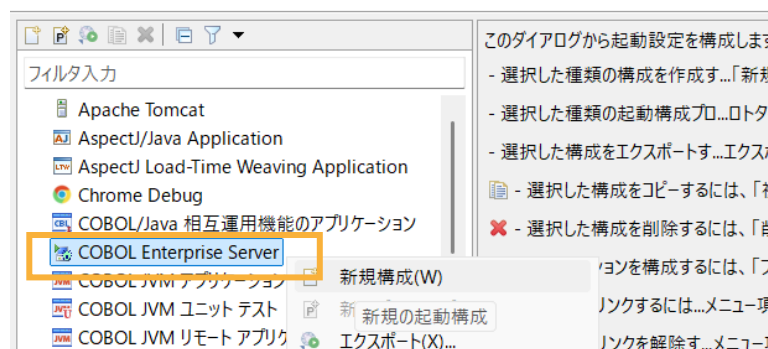
- ① Eclipse に戻り、[実行(R)] > [デバッグの構成(B)] を選択します。



- ② 左側より、「COBOL Enterprise Server」上で右クリックし、コンテキストメニューから [新規構成(W)] を選択します。

#### 構成の作成、管理、および実行

Enterprise Server アプリケーションへの接続とデバッグ



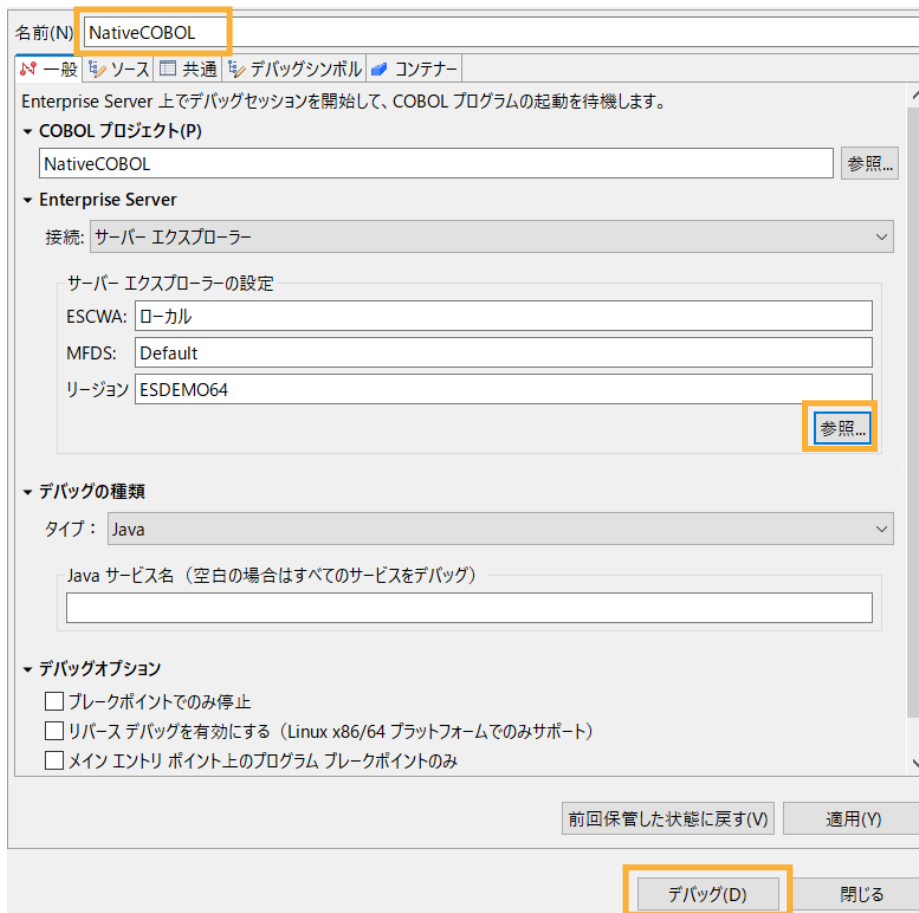
- ③ 以下の入力を行い、[デバッグ(D)] をクリックします。

名前 : "NativeCOBOL"

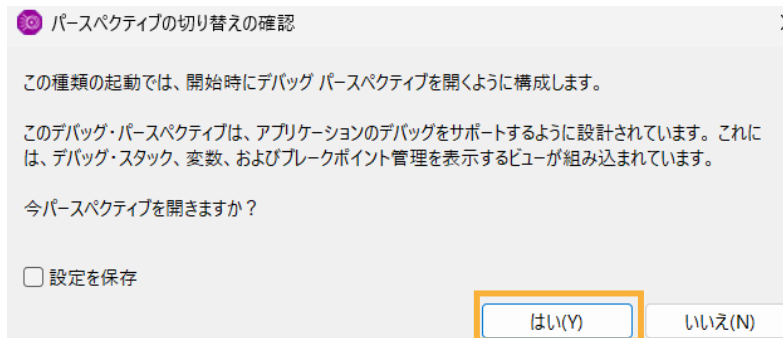
COBOL プロジェクト : NativeCOBOL

Enterprise Server : [参照] をクリックし、[ESDEMO64] を選択

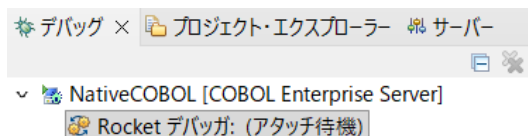
デバッグの種類 > タイプ : "Java"



以下のダイアログが表示された場合は、[はい(Y)] をクリックします。

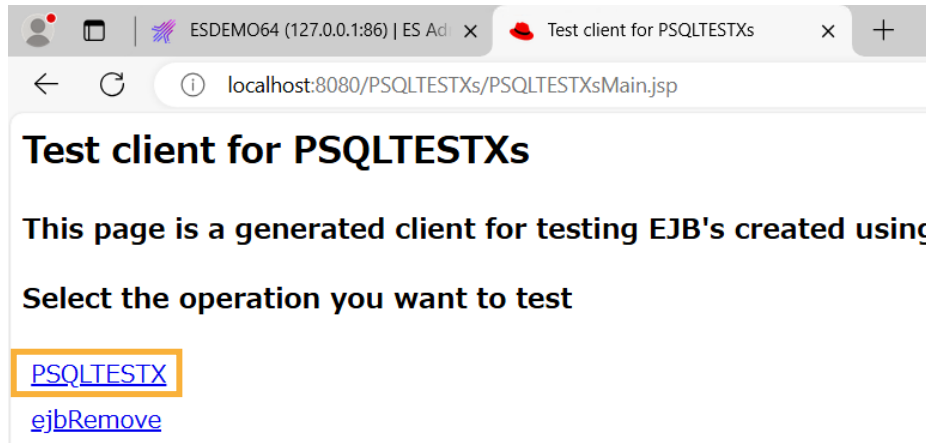


デバッガが以下のようにアタッチ待機になります。



- ④ ブラウザーを開き、以下のアドレスにアクセスし、[PSQLTESTX] リンクをクリックします。

<http://localhost:8080/PSQLTESTXs/PSQLTESTXsMain.jsp>



- ⑤ データを更新します。

現在、サンプルレコードの EMPNO: 7934 の情報は以下の通りです。

EMPNO	7934
ENAME	CMILLER
JOB	CLEAR
SAL	1450.00
DEPTNO	3

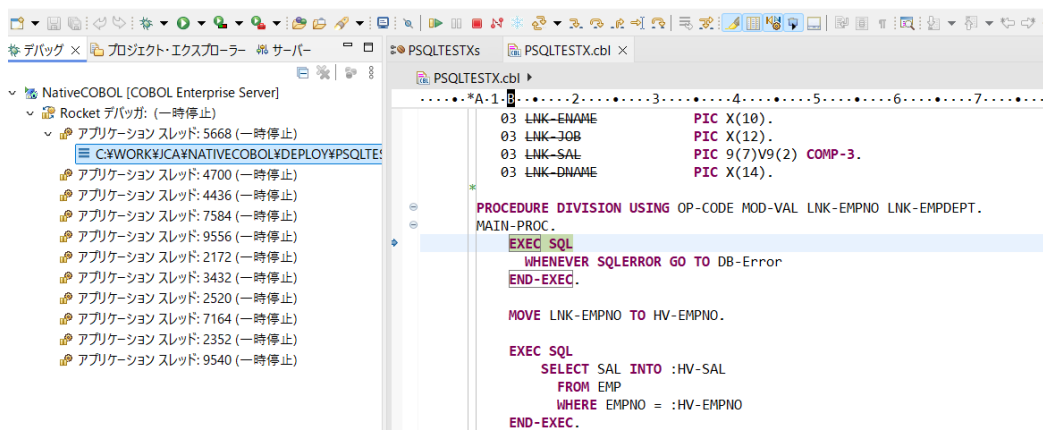
Web 画面で以下の項目を入力して、[Go!] をクリックします。

PSQLTESTX\_OP\_CODE\_io: "I"

PSQLTESTX\_MOD\_VAL\_io: "150"

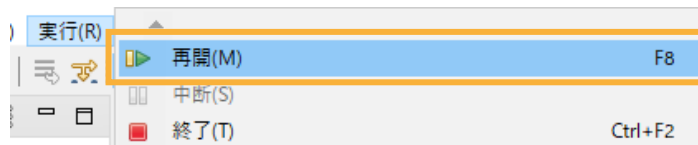
PSQLTESTX\_LNK\_EMPNO\_io: "7934"

- ⑥ Eclipse IDE を見ると、デバッグパースペクティブ上で、PSQLTESTX.cbl で停止していることが分かります。

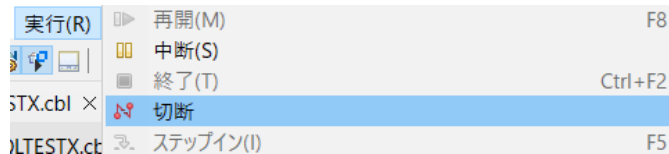


通常のコンソールアプリケーションと同様、項目値の確認や、ステップ実行などが行えます。

デバッグを終え、画面に応答結果を戻すためには、[実行(R)] > [再開(M)] を選択してください。

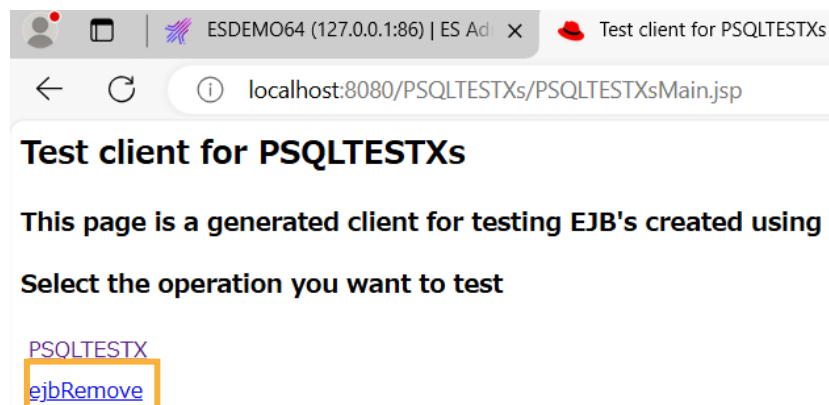


- ⑦ デバッグ作業を終了する際は、[実行(R)] メニューより [中断] を選択してください。



- ⑧ EJB の削除処理を行います。

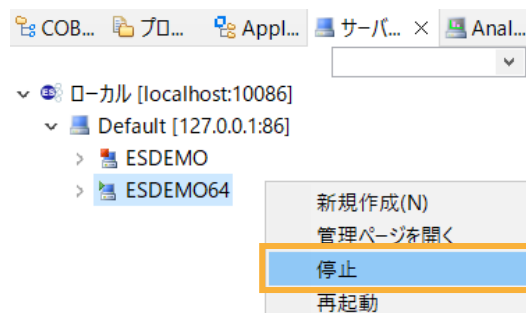
画面下部に表示されている [Back] リンクをクリックし、[ejbRemove] リンクをクリックします。



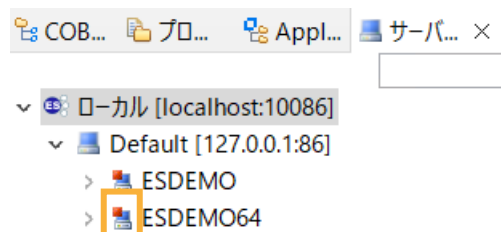
[Go!] をクリックし、ブラウザを閉じます。

## 5) JBoss EAP, Enterprise Server インスタンスの停止

- ① JBoss アプリケーションサーバーを停止します。
- ② Eclipse に戻り、[サーバー エクスプローラー] を開き、[ESDEMO64] を選択、マウスの右クリックでコンテキストメニューを開き、[停止] をクリックします。



停止すると、赤いアイコンが表示されます。



## 免責事項

ここで紹介したソースコードは、機能説明のためのサンプルであり、製品の一部ではございません。ソースコードが実際に動作するか、御社業務に適合するかなどに関しまして、一切の保証はございません。ソースコード、説明、その他すべてについて、無謬性は保障されません。

ここで紹介するソースコードの一部、もしくは全部について、弊社に断りなく、御社の内部に組み込み、そのままご利用頂いても構いません。

本ソースコードの一部もしくは全部を二次的著作物に対して引用する場合、著作権法の精神に基づき、適切な扱いを行ってください。