

Visual COBOL チュートリアル

1

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 に準拠しているので同じく 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。

OS	Windows 10
Java	Adoptium 17.0.11.9
AP サーバー製品	JBoss Enterprise Application Server 7.4.16

アプリケーションサーバー ソフトウェア

● 開発クライアント ソフトウェア

OS	Windows 10
COBOL 環境製品	Visual COBOL 10.0 for Eclipse
PostgreSQL ODBC Driver	psqlodbc 16.00.00.00
Windows SDK	10.0.26100.1

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 前提条件
- 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 ディプロイ先の変更
 - 3.6 ディプロイ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 [スタート] メニュー > [Windows 管理ツール] > [ODBC データ ソース (64 ビット)] 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。



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	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
	PostareSOL Unicode	16.00.00.00	Postgre
	PostgreSQL Unicode(x64	16.00.00.00	Postgre
	SQL Server	10.00.17763.6054	Microso
	<		>
	< 戻る(B)	完了	キャンセル

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

4



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 Unicod	de ODBC セットアップ	×		
データソース名: (N) 説明:(D) SSL Mode:(L) サーバー名: (S) データベース名: (b)	PostgreSQL 無効 ~ PostgreSQL-Addr vcdemo	テスト 保存 キャンセル Port: 5432		
既定の認証 ユーザー名: (U) パスワード: (w) PostgreSG	postgres ●●●●●●●● L Ver7.3 Copyright (C) 1998-2006; Insight Dist	マジョン(高度な設定) データソース 全体設定		



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	₩ 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 PostgreSQL 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 > [Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(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)¥Micro Focus¥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 odbc
- 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 s	ort /R		
2024/08/05	11:50	<dir></dir>	
2024/08/05	11:50	<dir> .</dir>	
2024/08/05	11:50	96,768 ESODBCXA_D.dll	
2024/08/05	11:50	88,064 ESODBCXA.dll	
2024/04/23	15:36	121,877 ESORAXA.CBL	
2024/04/23	15:36	10,852 build.bat	
2024/02/08	17:44	48,866 xapd.cpy	
(以下、省略)			

4

- 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: "jboss74EAP"

ServerHost に対する Please enter the resource adapter you would like to update: "mfcobolxa.rar"

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

Trace に対する Would you like to change the value of ServerHost? (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
oracleweblogic1221
oracleweblogic1411



Please enter the application server you would like to update: jboss74EAP Your available resource adapters are: mfcobol-localtx.rar mfcobol-notx.rar mfcobol-notx.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¥jboss74EAP¥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 > [Micro Focus 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
 - 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]を「デフォルト(windows-31j)」に変更し、[適用して閉じる] をクリックします。



フィルタ入力	ワークスペース	← → ⇒ 8
◆ 一般 へ Capabilities Schema Associ > Security UI フリーズ・モニタ > User Storage Se Web ブラウザ > エディタ	 ワークスペースの開始およびシャットダウン設定については、開始およびシャットダ □ ネイティブのフックまたはポーリングを使用して更新(R) ☑ アクセス時に更新(S) □ 無関係なブロジェクトを常にプロンプトなしで閉じる(C) ワークスペース保管間隔(分)(W): 5 	ウン'を参照してください。
キー クイック検索 グローバル化 コンテンツ・タイプ サービス・ポリシー トレース	ウィンドウのタイトル ✓ ワークスペース名を表示(E): JCA □ パースペクティブ名を表示(T) □ ワークスペースのフルパスを表示(F): C:¥Users¥Public¥Micro Focus¥Vis ✓ プロダクト名を表示	ual COBOL¥eclipse¥¥work¥JCA
 ネットワーク接続 ハンドラーをリンク パースペクティブ プロジェクト・ネーラ ワークスペース 開始およびシャッ 外観 絵索 	プロジェクトを開く際に、参照するプロジェクトを開く: プロンプト 〜 不明なプロジェクトの性質を以下のように報告(A): 警告 〜 Report missing project encoding as: 警告 〜	
通知 比較/パッチ	システム・エクスフローフーを起動するコマント(X): explorer /E,/select=\$[sele テキスト・ファイル・エンコード(T)	cted_resource_loc}
AspectJ Compiler > CSS (Wild Web De > Gradle	 ● デフォルト(U) (windows-31j) ● デフォルト(E) (Windows-31) ● その他(O): windows-31 ∨ ● その他(H): Windows-31 ∨ 	Vindows)
HTML (Wild Web I *	デフ <i>れ</i>	ルトの復元(T) 適用(L)
	四开	TY/200

※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
 - [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。
 - 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クストメニューを表示し、[プロパティ (R)]を選択します。[Micro Focus] > [ビルド構成] > [COBOL]とナビゲートし、「構成の固有な設定を可能に する(C)」にチェックを入れて[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。



フィルタ入力	COBOL		⇒ ▼
> リソース			
Coverage			
 Micro Focus 	フィルタテキフトを入力		
ビルダー			
ビルド パス	設定	值	^
> ビルド構成	▶ 一般		
✓ プロジェクト設定	文字セット	ASCII	
> COBOL	ソース エンコーディング	ANSI	
コンテナー	COBOL 方言	Micro Focus	
ビルド環境	ソース フォーマット	固定	
指令の確定	デバッグ用にコンパイル	はい	
> 実行時構成	EXIT PROGRAM を GOBACK として処理	ANSI	
WikiText	詳細	いいえ	
サーバー	.GNT にコンパイル	いいえ	
タスク・タグ	▼ 出力		
タスク・リポジトリー	指令ファイルを生成する	いいえ	
ビルダー	リストファイルを生成	いいえ	
プロジェクト・ネーチャー	コード カバレッジを有効にする	false	
プロジェクト・ファセット	プロファイラを有効にする	false	
プロジェクト参照	▼ Iラ-/警告		
検証	警告レベル	回復可能なエラーを含める(レベル E)	\sim
実行/デバッグ設定	ソース エソコーディング SOURCE-ENCODING はソース プログラムのエンコ	ーディングをコンパイラに渡します。その後、RUNTIME-ENCODING 指	\$
	COBOL J9//1ル設定:		
	CHARSET"ASCII" SOURCE-ENCODING"ANSI EXITPROGRAM"ANSI" NOTESTCOVER NOPR	" DIALECT"MF" SOURCEFORMAT"fixed" NOLIST anim OFILE WARNING"1" MAX-ERROR"100"	~
		デフォルトの復元(1) 適用	∃(<u>L</u>)
?		適用して閉じる キャンセ	ュル

- 4) プログラムソースのインポート
 - COBOL エクスプローラーのパースペクティブを開き、COBOL エクスプローラービューにて プロジェクトフォルダーを右クリ ックし、コンテクストメニューから [インポート(I)] > [インポート(I)] を選択します。

	インポート(i) >	Ê,	リモート プロジェクト
4	エクスポート(O)	8	ローカル Micro Focus プロジェクトのリモート プロジェクトへの変換
8	更新(F) F5	2	Net Express プロジェクトの変換
	プロジェクトを閉じる(S)	2	インポート(1)

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				
フォルターまたはアーカイン	ブ由来のプロジェクト	-		
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	スペースへ			
🔲 🛄 設定				
> 🗁 EJB				
> 🗁 Git				
> 🗁 Gradle				~
? < 房	ēる(B) ガ	ζ∧(N) >	終了(F)	キャンセル

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・シスラ	-ムからリソースをインポートしま	रंग.				
次のディレクトリーから(\	/): c:¥pgm				~	参照(R)
🔳 🗁 pgm			PSQLTES	STX.cbl STXE.cbl		
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					参照(W)
オプション	タリソースを上書き(O) レダーを作成(C)					
拡張 >>(A)						
?		< 戻る	(B) 次/	∿(N) >	終了(F)	キャンセル
① プロジェクトフォルタ	ダーを展開し、2つの	ファイルが正常(こロードされて	いることを確	認します。	
පි COB 🗙 🖻 プ	コ 😤 Appl 🔳 サ・	-/ť				
✓ I NativeCOBO	L					
V / COBOL 7	ログラム FSTX cbl					
> 🖻 PSQLTI	ESTXE.cbl					
> 🗁 New_Con	figuration.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。
- プロパティ設定ダイアログが表示されます。[Micro Focus] > [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 [Micro Focus] > [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。

 ✓ Micro Focus ビルダー ビルド パス ジルダー ビルド構成 ◇ COBOL イベント ディブロイ ビルド環境 ✓ ビルド環境 ✓ レinkage / ビカクを約 / ハativeCO 出力の名称 / ハativeCO 出力の名称 / レント ガイブロイ / レント ブイフロイ / レント ブイフロイ / レント ブイワイ / レント ブイワイ / レント ブイワイ / レント ブイワイ / レント ブイワイ / レント ブイワイ / レント ブイフレクラキストを入力 (値 / レント ガイント / レント ブイフレクラキストを入力 (値 / レント ガイント / レント ブイント ブイフレクラキストを入力 (値 / レント ブイント / レント ブイント ブイント ブイント / レンネ / レンネ<	80L guration.bin
ビルダー ビルドパス ン ビルド構成 ン COBOL イベント ディブロイ ビルド環境 ン Uinkage レルド環境 ン レクムシアを定 ン COBOL ロカの名前 バストリボイント クーグットの種類 ブストリボイント クーグットの種類 ブストリボイント クーグットの種類 ブストリボイント クーグットの種類 ブストリボイント シ COBOL ロンテナー ビルド環境 指令の確定 マルクテキストを入力 国 したなの にしいな ローク レルス マーグットの種類 フィルクテキストを入力 国 したなの たい レート レート レート レート レート マーグットの種類 フィレクテージ化 レールス マーグットの種類 フィントリボイント シーグットの種類 マーグットの種類 レート レート レート レート レート レート レート レート	<i>BOL</i> guration.bin
ビルドパス シ ビルド構成 シ COBOL イベント ディブロイ ビルド環境 シ COBOL レルド環境 シ COBOL レルド環境 シ COBOL ロンテナー ビルド環境 指令の確定 アレジスム レルズ レルズス シ COBOL ロンデナー ビルド環境 ビット数 COBOL ロンデナー ビルド環境 ロンデナー ビルド環境 ロンデナー ビルド環境 ロンデスム マレジスム レルズス ロンデナー ビルド環境 ロンデナー ビルド環境 ロンデスム マレジスム レルズス ロンデナー ビルド環境 ロンデスム ロンデントリズイント ロンデナー ビルド環境 ロンデスム ロンデントリズイント ロンデナー ビルド環境 ロンデスム ロンデントリズイント ロンデントリズイント ロンデントリズイント ロンデントリズイント ロンデントリズイント ロンデントリズイント レルズ ロンデント レルズス ロンデントリズイント ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント レルズ ロンデント ロンデント レルズ ロンデント ロンズ ロンデント ロンデント ロンズ ロンデント ロンズ ロンデント ロンズ ロンズ ロンズ ロンデント ロンズ ロンズ ロンデント ロンズ ロンズ ロンズ ロンズ ロンズ ロンズ ロンズ ロンズ	BOL guration.bin
 ◇ ビルド構成 > COBOL イベント ディブロイ ビルド環境 ◇ Linkage (2) レンク (3) レンク (4) レンク 	80L guration.bin
 > COBOL イベント ディブロイ ビルド環境 > リンク ンロンエンド設定 > COBOL ビルド環境 > リンク ンロンエンド設定 > COBOL コンテナー ビルド環境 当 ペマ INT/ クーゲットの種類 すべて INT/ ビット数 64 ビット LBR にパッケージ化 いいえ COBOL レメペのアプリケーションから呼び出 いいえ マルチスレッド ばい 	30L guration.bin
イベント ディブロイ ビルド環境 シリンク シリンク シリンク シリンク シリンク シリンク シリンク ジレンナー設定 シ COBOL コンテナー ビルド環境 指令の確定 マルチスレッド ない マルチスレッド ない ない	30L iguration.bin
ディブロイ ビルド環境 シリンク → リンク → ロンコンテナー ビルド環境 当つデナー ビルド環境 指令の確定 オペて INT/ レルズ なく INT/ レルズ ・ ロカパス クーゲットの種類 ・ マルチス レット数 ・ ロット数 ・ ロット ・ ロー ・ ロット ・ ロット ・ ロー ・ ロー ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・	guration.bin
ビルド環境 > リンク → ロンコント設定 > COBOL コンテナー ビルド環境 指令の確定 アルチスレッド ないいえ マルチスレッド レいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マルチスレッド ないいえ マーグットの種類 ・ いいえ ・ いいえ ・ いいえ マーグットの弾力 いいえ マーグットの種類 ・ いいえ ・ いいえ ・ いいえ ・ いいえ マーグットの弾力 ・ ジト数 ・ いいえ ・ いいえ ・ いいえ ・ ひいえ ・ いいえ ・ ひいえ ・ ひいえ ・ ひいえ ・ ひいえ ・ ひいえ ・ ひん ・ ひいえ ・ ひいえ ・ ひいえ ・ ひん ・ ひいえ ・ ひん ・ ひいえ ・ ひん ・ ひいえ ・ ひいえ ・ ひん ・ ひいえ ・ ひん ・ ひいえ ・ ひいえ ・ ひん ・ ひいえ ・ ひいえ ・ ひん ・ ひん ・ ひいえ ・ ひん ・ ひん ・ ひいえ ・ ひん ・ ひいえ ・ ひん ・ ひいえ ・ ひん ・ ひん ・ ひん ・ ひいえ ・ ひん ・ ひいえ ・ ひいえ ・ ひいえ ・ ひん ・ ひん	garation.bin
 > リンク > リンク > COBOL コンテナー ビルド環境 サービスを COBOL アーカイブ (.car) ファイル いいえ マルチスレッド 	
 ✓ プロプエクト設定 > COBOL コンテナー ビルド環境 ビルド環境 サービスを COBOL アーカイブ (.car) ファイル いいえ マルチスレッド 	GNT ファイル
> COBOL LBR にパッケージ化 いいえ コンデナー ビルド環境 レレスタのアブリケーションから呼び出 いいえ どルド環境 サービスを COBOL アーカイブ (.car) ファイル いいえ	
コンデナー ビルド環境 指令の確定 マルチスレッド はい ない ない レレネ レルネ ロルネ ロルネ ロルネ ロルネ ロルネ ロルネ ロルネ ロ	
ビルド環境 サービスを COBOL アーカイブ (.car) ファイル いいえ 指令の確定 マルチスレッド ばい	
指令の確定 マルチスレッド はい	
> 実行時構成 実行時モデル 共有	
WikiText 現在の実行時システムだけにバインドする いいえ	
サーバー 出力の種類 コンソール	
タスク・ダク ターゲット オペレーティング システム Windows	
ダスク・リホジトリー 詳細 いいえ	
ビルター cpp ライブラリを含める いいえ	
プロジェクレーフーセルレー 未定義シンボルでエラー いしえ	
コントリポイントアドレスを読み込む いいえ	
	•
プロジェクト参照	、ソースごとに 1 個の

© Rocket Software, Inc. or its affiliates 1990–2024. All rights reserved. Rocket and the Rocket Software logos are registered trademarks of Rocket Software, Inc. Other product and service names might be trademarks of Rocket Software or its affiliates.



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



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、コンテクストメニューから [新規作成

(N)] > [Java インターフェイス] を選択します。

ファイル(F) 編集(E) リファクタ		新規作成(N)	>	2	COBOL JVM プロジェクト
□ • · • </td <td></td> <td>開く(O) 表示方法(W) アプリケーションから開く</td> <td>Alt+シフト+W > ></td> <td></td> <td>COBOL JVM ユニット テスト プロジェクト COBOL コピーファイル プロジェクト COBOL プロジェクト COBOL プロジェクト</td>		開く(O) 表示方法(W) アプリケーションから開く	Alt+シフト+W > >		COBOL JVM ユニット テスト プロジェクト COBOL コピーファイル プロジェクト COBOL プロジェクト COBOL プロジェクト
✓	D	Jピ-	Ctrl+C	2	COBOL/Java 相互運用機能のプロジェクト
✓ // COBOL プログラム	Ē	貼り付け	Ctrl+V	2	リモート COBOL JVM プロジェクト
> D PSQLTESTX.cbl	×	削除(D)	削除	(リモート COBOL コピーファイル プロジェクト
> 🖻 PSQLTESTXE.cbl	<u>.</u>	コンテキストから除去	Ctrl+Alt+シフト+下	(リモート COBOL プロジェクト
> Dew_Configuration		移動(V)		R	リモート COBOL ユニット テスト プロジェクト
		名前を変更(M)	F2		プロジェクト(R)…
		ビルド アクション タスクのスキャン 指令の確定 プログラムをコピーファイルに変換 ファイル指令の削除 ブログラムのフォーマット	> Alt+°/7ト+F		COBOL コピーファイル COBOL ブログラム COBOL ユニット テスト スタンドアロン ファイル リモート スタンドアロン ファイル Java インターフェイス
		コピーファイル グラフ		29	REST Web サービス

 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	-0
Java インターフェイス名: PSQLTESTXs マッピング: ① デフォルト 〇 無し	
マップするプログラム: NativeCOBOL/PSQLTESTX.cbl	参照
?	終了(F) キャンセル

© Rocket Software, Inc. or its affiliates 1990–2024. All rights reserved. Rocket and the Rocket Software logos are registered trademarks of Rocket Software, Inc. Other product and service names might be trademarks of Rocket Software or its affiliates.



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。

Separate Section:		PSQI TESTX ★パレーショ	1.1.4、1.4、-フ	ィンコィール	.K.
名前 PICTURE POP-CODE X PMOD-VAL 9(5) comp-3 LINK-EMPNO S9(9) comp-5 分目 LNK-EMPDEPT		名前 ➡ OP_CODE_io ➡ MOD_VAL_io ➡ LNK_EMPNO_io > 聞 LNK_EMPDEPT_ic	方向 方向 入出力 入出力 入出力 入出力 入出力 入出力	型 String int int	OCC
🔵 วา-ルド プロパティ — 🏾 เ	n x				
名前: OP_CODE_io 型: String ~ (方向	DCCURS: 0				
マッピング OP-CODE	編集				

[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。



P	PSQLTESTX オペレーション・インターフェイス フィールド:									
4	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_id		String							
	LNK_JOB_io		String							
	LNK_SAL_io		BigDeci							
	➡ LNK_DNAME_id	b	String							

3.5 ディプロイ先の変更

- 1) COBOL エクスプローラー表示設定の変更
 - 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。

웜 COB × 🖻 プロ 😤 Appl 🥌 サーバ 🖳 Anal 🗖 ▼ 🖻 🕏 📮 🚸	INKAGE SECTION:
 V 28 NativeCOBOL ゆ COBOL プログラム 	COBOL JVM プロジェクト表示(R) > トップレベル要素(T) >
 PSQLIESTX.cbl PSQLTESTXE.cbl G Java インターフェイス * PSQLTESTXs PSQLTESTXs New Configuration.bin 	 ワーキング・セットの選択(W) ワーキング・セットの選択解除(K) アクティブなワーキング・セットの編集(E) 1 ウィンドウ・ワーキング・セット
> 🖻 repos	⑦ フィルタとカスタマイズ(F)

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) ディ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、コンテクストメニューから [新規作成(N)]→[フォルダー] を選択し



ます。					
≌ COB × 🛅 プロ.		😪 Appl 📕 サーバ 🔚 Anal 🗖 🗖	\$⊗ PSQLTESTXs ×		
		新規作成(N)	>	않	COBOL JVM プロジェクト
✓ I NativeCOBOL ✓ I COBOL JI		表示方法(W)	Alt+シフト+W >	20 20	COBOL JVM ユニット テスト プロジェクト COBOL コピーファイル プロジェクト
→ PSQLTE > PSQLTE		コピー 貼り付け	Ctrl+C Ctrl+V	2 2	COBOL プロジェクト COBOL ユニット テスト プロジェクト
〜 🕞 Java インタ → 📽 PSQLTE	2	削除(D) コンテキストから除去 移動のの	削除 Ctrl+Alt+シフト+下		COBOL/Java 相互運用機能のプロジェクト リモート COBOL JVM プロジェクト
> 🗁 New_Conf > 🗁 repos		√≥動(∨) 名前を変更(M). . .	F2	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	リモート COBOL Jビー Jアイル ブロジェクト リモート COBOL プロジェクト
		ビルド アクション	>	₩3 →	リモート COBOL ユニット ナスト ノロシェクト プロジェクト(R)
		シスクのスキャク 指令の確定 ファイル指令の削除		i i i i i i i i i i i i i i i i i i i	COBOL コピーファイル COBOL プログラム
		コード分析	>		スタンドアロン ファイル リモート スタンドアロン ファイル
) 5- アウトライン × 📻	イン - ゴン	インポート(i) エクスポート(O)	>	: :	Java インターフェイス REST Web サービス
	\$	更新(F)	F5	:2	SOAP Web サービス
アウトラインを提供する		フロジェクトを閉じる(S) 無関係なプロジェクトを閉じる(U)		ø	Dockerfile
		Source	>		SQL ファイル ファイル
		リモートシステムビューで表示		Ċ	フォルダー

 「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): deploy		
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、コンテクストメニューから [インポート] > [インポート] を選択します。



COB × CoB * Appl COB * Appl COB COB		サーバ 🖳 Anal 🗖 🗖 V 🖻 🔁 📴 🕹 🖇	© PSQLTESTXs × LINKAGE SECTION: 名前 □ OP-CODE	PIC X	TURE		
	新規作成(N)		> Alt+シフト+W > Ctrl+C Ctrl+V 削除 Ctrl+Alt+シフト+下 F2		om	1p-3	
> PSQLIESTXE.cbl x PSQLIESTXE.cbl x PSQLIESTXE.cbl		表示方法(W)			comp-5		
 G Java インターフェイス PSQLTESTXs deploy New_Configuration.bin Prepos 	 □ビー □Bり付け ■ 削除(D) ■ コンテキストから除去 移動(V) 名前を変更(M) タスクのスキャン 						
		インポート(i) エクスポート(O)		>	宅, 23	リモート プロジェクト ローカル Micro Focus プロジェクトの	
	8	更新(F)		F5	2	Net Express プロジェクトの変換	
語 アウトライン × 国フログラム アウト		Source		>	\geq	インポート(I)	

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へ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中来のプロジェクト 		- 1
 ◎ 既存プロジェクトをワークスペースへ □ 設定 		
> 🗁 EJB		
> 🗁 Git		
> 🗁 Gradle		~
? < 戻る(B) 次へ(N) >	終了(F)	キャンセル

 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 COB	OL¥deploy ~	参照(R)					
eploy	.mfdeploy }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)							
インボート先フォルダ(L): NativeCOBOL/deploy		参照(W)					
オブション 一 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)							
? < 戻る(B)	次へ(N) > 終了(F)	キャンセル					
補足)							
上記手順完了後も COBOL エクスプローラー上の Nativ	/eCOBOL¥deploy フォルダー配 ⁻	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							
エ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	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

2

① Eclipse に戻り、[サーバーエクスプローラー] を開きます。

🔓 COB	Х	🔁 プロ	e e	Appl	🔳 サーバ	📇 Anal
						Y 🖪 🔁 .
🗸 😕 Nat	ive	COBOL				
v 🗖 (COE	SOL プログラ	4			
>	D P	SQLTESTX	.cbl			
>	D P	SQLTESTX	E.cb			
🗸 🗸 🗸	ava	インターフェ	イス			
> 8	© F	SQLTESTX	s			
> 🗁 (dep	loy				
> 🗁 🛛	Vev	v_Configur	atio	n.bin		
> 🗁 i	еро	DS				
[+] アイコン	っを打	甲します。				
補兄)						

すでに作成済みの場合は、②~③はスキップしてください。



	ີ 🔓 COB 造 プロ 🤹	Appl 📕 サーバ ×	📇 Anal		
	i ESCWA インスタンス	た 定義	*	🕈 🖻 🖇	
)	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 Co	mmon Web Administra	tion インスタ	ンスへの接続を新規	作成します
	名前:	ローカル			
	サーバアドレス (IPv4/ホスト名):	localhost			
	サーバーポート:	10086			
	□ TLS 有効				
	TLS 設定				
	CA 証明書:				
		参照			
		27			
	?			終了(<u>F</u>)	キャンセル
		あわ埋今け いてん	シーンション	ミニナわますのべ	いてのこちを行

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	ック			
ESCWA: D-;	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:	SYSAD				
パスワード:	******				
	☑ 認証情報	の保存			
🗌 認証情報	尿がクリアされ	るまで、再度プロンプトを表示しな	1		
				OK	ナトンクリ
				UK	キャノビル

© Rocket Software, Inc. or its affiliates 1990–2024. All rights reserved. Rocket and the Rocket Software logos are registered trademarks of Rocket Software, Inc. Other product and service names might be trademarks of Rocket Software or its affiliates.



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、[Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(64 ビット)] 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
 - [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クストメニューを開き、[管理ページを開く] を選択します。

诸 COB 陷 プロ 🧏 Appl	📕 サ−バ × 📕 Anal
	× =
~ ◎ □−カル [localhost:10086]	新規作成(N)
✓	管理ページを開く
> 🐁 ESDEMO > 🐁 ESDEMO64	✗ 削除 ⋒ 田田田田田田田田田田田田田田田田田田田田田田田田田田田田田田田田田田田田

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。



🐙 E:	S Administration	× +									1
(j) localhost:10086/#/nati	ve/ds/127.0	0.0.1/86			A»		띠	₹_≡	Ē	q
		ES			⊕ 言語	~					
	1.112	Ente	erprise	Serv	er						
		Con	nmon V	Veb							
		Adn	ninistra	ation							
		A Micro Fo 的なセキ	ocus Enterprise Se キュリティ機能がう 青報	erverでは、イ デフォルトで	'ンストール後 有効になってい	に基本	- >				
		ユーザー名	3						1		
		SYSAD)								
		パスワート	<i>a</i>								
								No.			
States a		パス	ワード変更		ログオン			Binn			

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、「ネイティブ] 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

ES Administration	:	× +				
\leftarrow C (i) localhost:10086/#/n	ative/	ds/127.0.0.1/	86/regio	ns		
ES 管理 / ダッシュボード ネイ	ティブ	メインフレ	-7 -	セキュリティ		
ネイティブナビゲーション ^ 」	ージョ	ンおよびサーバ	- 7	プロパティ 🗸 セキ	ュリティ 🛛	ジャーナル
 「」 「」 	このDirec	tory Serverホスト	ではTLSが有	i効ではありませんが、ルー:	プバック トラフィ	、ックの受信のみに
> 論理	Directory	ServerにMT0 ライ	センスがあ	りません。ライセンス状態を	を確認してください	N。一部の機能は [、]
→ PAC → III Directory Server	ージョ	ョンおよびț	ナーバー	- リスト * 新	見作成 📋	すべて削除
✓ ☆ ⊕ Default						
E ESDEMO						
	74	ろ前		♡ 説明	Υ P	AC
> ф SOR						
	ア	名前	タイプ	ステ	64ビ	MSS
	Ξ.	ESDEMO	Region	Stopped		
	=	ESDEM064	Region	Stopped	\checkmark	
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						



- 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。

ES	管理 ダッシュボード	ネイティブ	メインフレーム	セキュリティ
ネイティ	ブナビゲーション ^	一般		
~ D?	グループ ======	一般的	プロパティ	▲ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □
>	論理 PAC		コントロール	
~ 📰 [Directory Server	8844	検証	
~ 1	☐ ⊕ Default	IFI XC	リスナー	
	E ESDEMO	名前*	サービス	クトリ 8
> ф s	SOR	ESI	XA リソース	
			診断	

[新規作成]をクリックします。

ES 管理 ダッシュ7	ボード ネイティブ メインフレー	-ム セキュリティ
ネイティブナビゲーション	^ ──般 🗸	
 ・	XA リソース * 新	規作成
✓		
🕤 🏠 🔂 Default	T ID	▼ 名前
ESDEMO		
ESDEMO64	Ⅲ 表示列 ✔	

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* 8	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	保存反	3
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 DS	SN= 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。

XA UV	/ース *	新規作成			
7	D	Y	名前	Y	OPEN 文字列
回 君 ア	長示列 ✔ ID	名前	有効	OPEN	アクション
₿	PSQLXA	PSQLXA	\checkmark	DSN=Postg	Ø Ū

- 7) リスナーの構成
 - ① 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。



ES 管理 ダッシュボード	ネイティブ メインフレーム t	
ネイティブナビゲーション ^	一般 🗸	
◇ ⑦ グループ	אא שי ליםולדי	
> im理 > PAC	コントロール	
✓ I Directory Server 検証		
✓ ☐ ☐ ⊕ Default		
E ESDEMU64 > ∲ SOR	ХА IJУ—ス	

② 画面左下より、[通信プロセス1] > [Web] をクリックします。

ネイテ	ィブ リスナー ナビゲーション	· ^
通信	ヨサーバーの新規作成	С
~ E] 通信プロセス1	
	🛿 🖁 HTTP Echo	
	∎ * Web	
	🛿 🖁 Web Services and J2EE	
	s¦ [®] RFA	

③ 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 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 フォルダーがディプロイ用フォルダーとして使用されます。通常、Program Files (x86)等のフォルダ ーは管理者権限を持つユーザーフォルダー書き込みできフォルダー変更を行います。下記のように変更し、[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ーパスは、2) で作成した deploy フォルダーに対応します。

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。
 ネイティブリスナー ナビゲーション ^

通	言サーバーの新規作成	С
~ [] 通信プロセス 1 ff HTTP Echo	
	ਡ਼ੀ¥ੇ Web	
	art Web Services and J2EE	
	af¥ RFA	



⑤ [ポート] に "9003" を入力して、[適用] をクリックします。

一般 ~		ESDEM064	(Default)	& ユ−ザ− ∨	С
リスナー プロパティ	適用 茴 削除				С
* 入力必須の項目です 名前* ♀					
Web Services and J2EE		🔲 レガシー Micro Focus アプリケーミ	ション形式 🖓		
6 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	こなります。			
プロトコル* 💡	ホスト名またはIP アドレス* 💡		#− F 8		
tcp ~	*		9003		

- 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クストメニューを開き、[開始] を選択します。

🔓 COB 🛯 陷 プロ	😤 Appl	黒 サーバ ×	📇 Anal
			× =
~ 🗐 🛛 ーカル [localho	ost:10086]		
🗸 📕 Default [12]	7.0.0.1:86]		
> 📩 ESDEMC)		
> 🚼 ESDEMC	64		
	矛	沂規作成(N)	
	1	き理パージを聞く	_
	ß	绢 始	
	8 5	巨新	

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:	ー力ル/Default/ESDEMO64
230817	13424619	CASCD0127I SEP 00002 created for ES ESDEM064, process-id = 5128 13:42:46
230817	13424628	5128 ESDEMO64 CASSII500I SEP initialization started 13:42:46
230817	13424633	144 ESDEMO64 CASSI1600I SEP initialization completed successfully 13:42:46
230817	13424635	144 ESDEMO64 CASSI4100I PLTPI Phase 1 Processing Starting 13:42:46
230817	13424636	144 ESDEMO64 CASSI4106I PLTPI Phase 1 starting File initialization program 13:42:46
230817	13424637	144 ESDEMO64 CASSI5040I Active SEP memory strategy set to x'00000001', retain count 100 13:42:46
30817	13424681	4896 ESDEMO64 CASCS5100I Communications Process instance 01 is ready to accept requests 13:42:46
30817	13424721	CASCD1071I Administration SEP created for Server ESDEM064, process-id = 5712 13:42:47
230817	13424726	5712 ESDEMO64 CASSI1500I SEP initialization started 13:42:47
230817	13424735	5712 ESDEMO64 CASSI1600I SEP initialization completed successfully 13:42:47
230817	13424739	5128 ESDEMO64 CASSI1600I SEP initialization completed successfully 13:42:47

正常に起動しますと、緑のアイコンになります。

🔓 COB	🔁 プロ	😤 Appl	. 🔳 サーバ >
✓ ⁽¹⁾ (1) (1) (1) (1) (1) (1) (1) (1) (1) (1)	カル [localh	ost:10086]	
v 📕 [Default [12	7.0.0.1:86]	
>	🛓 ESDEMO	0	
> 1	🗏 ESDEMO	064	

- 9) JBoss ヘリソースアダプタのディプロイ
 - 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/VC100/Eclipse/vc100indx.html

[ディプロイ] > [Enterprise Server へのディプロイ] > [モダナイズ済みのアプリケーションのディプロイおよび構成]

- > [EJB およびリソース アダプターのディプロイ] > [EJB のディプロイ 概要] > [JBoss へのディプロイ]
- ② Windows メニューより [JBoss プラットフォーム] > [サーバーの起動(スタンドアローン] を選択します。



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。

16:44:36,690 INFO [org.jboss.as.repository] (DeploymentScanner-threads - 1) WFLY DR0001: ロケーション C:¥EAP-7.4¥standalone¥data¥content¥7d¥8b6b53fa83a874cea0279a e2ea9d945c1e82cf¥content にコンテンツが追加されました。 16:44:36,709 INFO [org.jboss.as.server.deployment] (MSC service thread 1-1) 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。 16:44:37,814 INFO [org.jboss.as.connector.deployment] (MSC service thread 1-3) W FLYJCA0007: 接続ファクトリ java:/eis/MFCobol_v1.5 を登録しました (中略) 16:44:38,358 INFO [org.jboss.as.clustering.infinispan] (ServerService Thread Pool --72) WFLYCLINF0002: ejb コンテナー から http-remoting-connector キャッシュを開始しました。 16:44:38,443 INFO [org.jboss.as.server] (DeploymentScanner-threads - 1) 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ィプロイ

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クストメニューを開き、[プロパティ(P)] を選択します。



🕆 COB 🗡 陷 プロ 🤮	g App	ol 📕 サーバ 🛄 Anal	
		× 🖻 🗧	¢ ↓ª
🗸 🛃 NativeCOBOL			
🗸 垣 COBOL プログラム			
> 🖻 PSQLTESTX.cb	I		
> Description	bl		
✓ 💫 Java インターフェイス	L .		
> SQLTESTXs		亲担作成(N)	
> 🗁 deploy		初125611-125(11)	
> 🗁 New_Configuration	, X	削除	
> 🗁 repos		プロパティ(P)	
		ディプロイ	
		検査	

② [ディ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

一般 ディブロイメントサーバー アブリケーションファイル EJB 生成	
Enterprise Server 소:	
ESDEMO64 (localhost:54869)	変更
□ Enterprise Server 実行結環境の使用	
Enterprise Server 実行時環境の構成	
EJB ステートフル サービスの場合、一部の値は無視されます	
<u> </u>	
PSQLTESTX]
トランザクション管理	
○ アプリケーション管理	
 ③ コンテナ管理 	

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ィプロイします] 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 ディブロイメントサーバー アブリケーションファイル EJB 生成	
	レガシーアブリケーションをディブロイ済みか、またはサーバーにディブロイする必要があるかを指定してください。 〇 レガシーアブリケーションは既にディブロイ済み ディブロイされたアブリケーションのパス:	
ľ	● レガシーアプリケーションをディプロイする	
Ī	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
		ファイル海川除

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 7.4"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¥javax

- annotation/api/main/jboss-annotations-api_1.3_spec-2.0.1.Final-redhat-00001.jar
- ejb/api/main/jboss-ejb-api_3.2_spec-2.0.0.Final-redhat-00001.jar
- resource/api/main/jboss-connector-api_1.7_spec-2.0.0.Final-redhat-00001.jar



servlet/api/main/jboss-servlet-api_4.0_spec-2.0.0.Final-redhat-00001.jar

一般 ディプロイメン	トサーバー アブ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 V JBoss EAP 7.4 V
✓ 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:	○ ステートレス
インターフェイス タイプ	:● ローカル ○ リモート
SEP 届性	
SEP:	○ ステートレス ◉ ステートフル
ディプロイメントディス	クリプク属性
EJB 名:	PSQLTESTXsEJB
アーカイブ名:	PSQLTESTXs
lava SEと lava EE (20篇件
	97 mil sa
Java コンパイラ:	C:¥Program Files¥Eclipse Adoptium¥jdk-17.0.11.9-hotspot¥bin 参照
EJB、コネクタ(また、	クライアント生成する場合、servlet と JSP)関連の JAR ファイルのパスを追加します。
Java EE クラスパス:	C:/EAP-7.4/modules/system/layers/base/javax/annotation/api/main/jboss-annotations-api_1.3_spec-2.0.1.Final-redhat-00001.jar;C:/EAP-7.4/modules/s
	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クストメニューを開き、[ディプロイ]を選択します。

😪 COB × 陷 プロ 😤 Appl 🧏 サーバ 🖳 Anal 「
\[\] \[\[\] \[\[\] \[\] \[\[\] \[\] \[\] \[\] \[\[\] \[\[\] \[\[\] \[\[\] \[\] \[\] \[\[\] \[\[\] \
✓ I NativeCOBOL
✓ 個 COBOL プログラム
> 🖻 PSQLTESTX.cbl
> 🖻 PSQLTESTXE.cbl
~ 词 Java インターフェイス
> 😂 PSQLTESTXs
> 🗁 deploy 新規作成(N)
> 🗁 New_Configurat 💢 削除
> 🗁 repos プロパティ(P)
iii mccerror.txt ディプロイ
検査

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。

 □ コンソール × 区間題 @ タスク □ プロパティー 喩 Table Results ☞ Filter Definitions 属 Micro Focus Unit Testing ピュードカバレッジ 当: サービス インターフェイス コンソール
 0021 (2024/08/05 17:06:54): Using directory at mrpi://127.0.0.1:86
 0030 (2024/08/05 17:06:54): ES server "ESDEM064" notified service "PSQLTESTX.PSQLTESTX" is available
 0002 (2024/08/05 17:06:54): Installation of package "PSQLTESTX.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。



ES 管理 ダッシュボード	ネイティブ	メインフレーム	セキュリティ	
ネイティブナピゲーション ^ > 節 グループ > 論理 > PAC > 面 ごectory Server > 面 合 ⊕ Default ■ ESDEMO ■ ESDEMO64 > ♀ SOR	一般 サービス ♥	 モニター ・ プロパティ コントロール 検証 リスナー サービス XA リソース 診断 過去の統計 CICS 	最終… Available Available Available Available	リス HTTP Ec Web@C Web Ser
	8	ES	Available	Web Ser
	鹵	✓ PSQLTESTX		
	ଡ	.PSQLTEST>	(Available	Web Ser

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。

 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クストメニューを開き、[クライアント生成] を選択します。

ີ 🗄 COB 🗡 陷 プロ 🧯	🔁 Ap	ppl 🗏 サーバ 🖳 Anal 🖓	
 ✓ 認 NativeCOBOL ✓ 認 COBOL プログラム > 図 PSQLTESTX.cl > 図 PSQLTESTXE.l マ G Java インターフェイ? 	ol cbl Z	▲ E 5 18 40	
> 🔊 PSQLTESTXs > 🗁 deploy > 🗁 New Configurat	×	新規作成(N) 削除	
> 🍃 repos		プロパティ(P) ディプロイ 検査 開く サードフの再発	
[クライアント生成	

[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。

```
    ■ コンソール × 図問題 タタク □ プロパティー 配 Table Results び Filter Definitions ज Micro Focus Unit Testing
    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=4362)(出=3903)(10%収縮されました)
    PSQLTESTXs.warを追加中です(入=21384)(出=20087)(6%収縮されました)
    mfejblib.jarを追加中です(入=11106)(出=9910)(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)/Micro Focus\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。



웜 COB × 陷 プロ 哈 Appl 易 サーバ
✓ I NativeCOBOL
✓ // COBOL プログラム
> PSQLTESTX.cbl
> PSQLTESTXE.cbl
🗸 🔓 Java インターフェイス
> 📽 PSQLTESTXs
> 🗁 deploy
> 🗁 New_Configuration.bin
🗸 🗁 repos
✓ ▷ PSQLTESTXs.deploy
> 🗁 Client
> 🗁 com
> 🗁 META-INF
🗎 mfejblib jar
PSQLTESTXs.ear
PSQLTESTXs.jar

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)
         17:21:39,796 INFO [org.jboss.weld.deployer] (MSC service thread 1-8) WFLYWELD00
         03: Weld デプロイメント PSQLTESTXs.ear を処理しています
         17:21:39,994 INFO [org.hibernate.validator.internal.util.Version] (MSC service thread
         1-8) HV000001: Hibernate Validator 6.0.22.Final-redhat-00002
         17:21:40,122 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
         17:21:40,130 INFO [org.jboss.weld.deployer] (MSC service thread 1-4) WFLYWELD00
         03: Weld デプロイメント PSQLTESTXs.jar を処理しています
         17:21:40,133 INFO [org.jboss.as.ejb3.deployment] (MSC service thread 1-4) 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
         17:21:40,191 INFO [org.jboss.weld.Version] (MSC service thread 1-5) WELD-00090
         0: 3.1.6 (redhat)
         17:21:40,856 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 7
         2) WFLYUT0021: 登録された web コンテ キスト: '/PSQLTESTXs' (サーバー 'default-server' 用)
         17:21:40,877 INFO [org.jboss.as.server] (DeploymentScanner-threads - 1) WFLYSRV
         0010: "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



	🗈 🛛 🎢 ESDEMO64 (127.0.0.1:86) ES Ad 🗙 📥 Test client for PSQLTESTXs	×	+
\leftarrow	C (1) localhost:8080/PSQLTESTXs/PSQLTESTXsMain.jsp		
Te	st client for PSQLTESTXs		
Thi	s page is a generated client for testing EJB's crea	ated	using
Sel	ect the operation you want to test		
<u>PSC</u>	<u>LTESTX</u>		
<u>ejb</u>	Remove		

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	CLEARK
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 : [

	[
	150
o :	7934
	Go!

Back



結果が表示され、1450.00 になって	いることがわた	かります。			
ESDEMO64 (127.0.0.1:86) ES Ad 🗙 🖣	🐣 Test Cl	lient for PSQLTE	STXs.PSQLT 🗙	1
\leftarrow C (i) localhost:8080/P	SQLTESTXs/PS	QLTESTXs	Servlet		
Test client for PSQLTES	TXs.PSQL	TEST	x		
Back					
Perform the test by entering value	5:				
PSQLTESTX_OP_CODE_io :					
PSQLTESTX_MOD_VAL_io: 150					
PSQLTESTX_LNK_EMPNO_io : 7934					
		Go!			
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 145	0.00 に更新	fされてい	はす。		
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、[ej	jbRemove] リンクをクリ	ノックします。
SDEMO64 (127.0).0.1:86) ES Ad	di 🗙	🐣 Test clier	nt for PSQLT	ESTXs
← C (i) localhost:8	080/PSQLTES	STXs/PS	QLTESTXsM	lain.jsp	
Test client for PSQLT	ESTXs				

This page is a generated client for testing EJB's created using t

Select the operation you want to test

PSQLTESTX <u>ejbRemove</u>

3

[Go!] をクリックし、ブラウザーを閉じます。



SDEMO64 (127.0.0.1:86) ES Ad 🗙 📥 Test Client for PSQLTESTXs.remov 🗙
← C (i) localhost:8080/PSQLTESTXs/removeSF.jsp
Test client for PSQLTESTXs.removeSF
Back
Perform the test by entering values:
Go!
Back

- 13) JBoss EAP, Enterprise Server インスタンスの停止
 - ① Windows スタートメニューより、[JBoss プラットフォーム] > [サーバーの終了 (スタンドアロン)] をクリックします。

	J
	JBoss ブラットフォーム ^ 新規
	JBoss 管理コンソール (スタンドアロン) 新規
	🍌 サーバー の起動 (スタンドアローン)
8	→ サーバー の起動 (ドメイン)
Ľ	😠 サーバーの終了 (スタンドアロン)
	😠 サーバーの終了 (ドメイン)
	プラットフォームのアンインストール 新規
ŝ	M
Φ	Micro Focus License Manager 🛛 🗸
	📼 🎯 📄 🤇

"続行するには何かキーを押してください . . ."の指示に従い、何かキーを押してプロンプト画面を終了します。 本作業により、JBoss EAP の起動時に使用していたプロンプト画面にも同様のメッセージが表示されますので、そち らも同様に終了します。

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クストメニューを開き、[停止]をクリックします。

ີະເດດອ 🚡 ວຳມ	😤 Appl	📕 サ−バ × 🛄 Anal
	. 40000	~
✓ ¹ ● □一刀ル [locali	nost:10086]	
🗸 📕 Default [1]	27.0.0.1:86]	
> 🐁 ESDEM	0	
> 🔚 ESDEM	064	新規作成(N)
	_	官理ページを開く
		停止
		再起動



停止すると、赤いアイコンが表示されます。
🔓 COB 陷 プロ 🧏 Appl 🔳 サ−バ ×
~ ☜ ローカル [localhost:10086]
🗸 📕 Default [127.0.0.1:86]
> 🛃 ESDEMO
> 🐁 ESDEMO64

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 Windows メニューより [JBoss プラットフォーム] > [サーバーの起動(スタンドアローン]を選択します。



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 [ローカル [localhost:10086] を選択したうえで、マウスの右クリックでコンテク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。 ダッシュボード画面が表示されたら、[ネイティブ] をクリックします。



	ES Administration x +					
\leftarrow	C (i) localhost:10086/#/dashboard					
ES	管理 ダッシュボード ネイティブ メインフレーム セキュリティ					
ダッう	シュボード + ウィジェットを追加 @ ダッシュボードのリロード ① デフォルトの更新 マ					
E ESCV	WAサーバーはTLSが有効ではありません。ネットワーク全体の通信は暗号化されません。できるだけ早くTLSの設定を有効にすること					
	# ようこそ					
	~ - 7					
L P	つこそ					
Enterprise Server Common Web Administration (ESCWA) は、最新の機能豊富なインターフェイスを備えたツールです。						
:	 サーバーとそのステータスや性能を表示および監視するためのカスタムダッシュボードを構成します。 JCL ジョブの実行および監視を含む、サービス実行プロセス、サービス、パッケージなど、Enterprise Server イン、 「複数のホストトのDirectory Serverを管理します。これにより、各ホストトのEnterprise Server インスタンスの停止 					

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クストメニューを開き、[開始] を選択します。

🔓 COB	🔁 プロ	😤 App	I 🔳 サーバ × 💻 Ai	nal			
~ -	+ 11 cr			¥ =			
⊻ 🧐 🛛 –	·カル [localho	ost:10086	b]				
v 📕	Default [12]	7.0.0.1:86	5]				
>	> 📇 ESDEMO						
>	🛃 ESDEMC	64					
			新規作成(N)				
			管理ページを聞く				
			開始				
			тали				
		\$	更新				

© Rocket Software, Inc. or its affiliates 1990–2024. All rights reserved. Rocket and the Rocket Software logos are registered trademarks of Rocket Software, Inc. Other product and service names might be trademarks of Rocket Software or its affiliates.



- 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。

	実行	, (R)	ウィンドウ(W)	ヘルプ(H)	
	 ・ ・ ・ ・ 				
T					
実行履歴(T) ● 実行(S) 実行構成(N)					
	蓉	デバ デバ	ッグ履歴(H) ッグ(G)		
		デバ	ッグの構成(B)		
	Q,	項目	を検査		

 を倒より、「COBOL Enterprise Server」上で右クリックし、コンテクストメニューから [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"

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"



名前(N) Native	COBOL						
い 一般 リソーン	ス 🔲 共通	☞ デバッグシンボル	🥒 コンテナー				
Enterprise Server 上でデバッグセッションを開始して、COBOL プログラムの起動を待機します。							^
▼ COBOL プロジ	ェクト(P)						
NativeCOBO	۱L					参照	
	erver						
接続: サ−バ-	- エクスプロー	5-				\sim	
サーバー エ	クスプローラー(の設定					
ESCWA:	ローカル						
MFDS:	Default						
リージョン	ESDEMO64						
						参照	
▼ デバッグの種類	Ę.						
タイプ: Java	а					~	
- Java サービ	ス名(空白	の場合はすべてのサ	ービスをデバッグ)				
▼ デバッグオプシ	ヨン						
□ ブレークポイ	イントでのみ停	L					
	バッグを有効	にする(Linux x86/ ∟のプロガニノ ブレ	′64 ブラットフォー♪ タポイントのフ	ムでのみサボート)			
	トリ 小1 ノト-		シホイントのみ				~
				前回保	管した状態に戻す(V)	適用(Y)	
					デバッグ(D)	閉じる	

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。

● パースペクティブの切り替えの確認	×			
この種類の起動では、開始時にデバッグ パースペクティブを開くように構成します。				
このデバッグ・パースペクティブは、アプリケーションのデバッグをサポートするように設計されています。 これに は、デバッグ・スタック、変数、およびブレークボイント管理を表示するビューが組み込まれています。				
今パースペクティブを開きますか?				
□ 設定を保存 はい(Y) いいえ(N)				
デバッガが以下のようにアタッチ待機になります。				
な デバッグ × № プロジェクト・エクスプローラー 綿 サーバー □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □				
 MativeCOBOL [COBOL Enterprise Server] Whicro Focus デバッガ: (アタッチ待機) 				

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



		🌾 ESDEMO64 (127.0.0.1:86) ES Ad 🗙 🍓 Test client for PSQLTESTXs	× +			
\leftarrow	C	(i) localhost:8080/PSQLTESTXs/PSQLTESTXsMain.jsp				

Test client for PSQLTESTXs

This page is a generated client for testing EJB's created using

Select the operation you want to test

<u>PSQLTESTX</u> <u>ejbRemove</u>

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	CLEARK
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。

	実行(R)		
		再開(M)	F8
1		中断(S)	
		終了(T) Ctrl	+F2

Rocket software

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

実行(R)		再開(M)	F8
		中断(S)	
		終了(T)	Ctrl+F2
STX.cbl ×	14	切断	
LTESTX.c	3.	ステップイン(I)	F5

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、ブラウザーを閉じます。

- 5) JBoss EAP, Enterprise Server インスタンスの停止
 - ① Windows スタートメニューより、[JBoss プラットフォーム] > [サーバーの終了 (スタンドアロン)] をクリックします。



"続行するには何かキーを押してください . . ." の指示に従い、何かキーを押してプロンプト画面を終了します。 本作業により、JBoss EAP の起動時に使用していたプロンプト画面にも同様のメッセージが表示されますので、そち らも同様に終了します。

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クストメニューを開き、[停止]をクリックします。





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。