Micro Focus メインフレームソリューション

スターターズキット

7. Enterprise Developer for Visual Studio 2012 on Windows8

: データベース連携

7.1 目的

当チュートリアルではデータベース接続を行うための下記手順の習得を目的としています。

- 1) プリコンパイルについて
- 2) プロジェクトの作成と既存プログラムの取り込み方法
- 3) プロジェクトと個別ファイルプロパティの設定方法
- 4) XA スイッチモジュールの作成方法
- 5) プログラムのコンパイル方法
- 6) Enterprise Server 設定方法
- 7) XA リソース正常ロードの確認方法
- 8) JCLの実行と確認方法

7.2 チュートリアルの前提と準備

本チュートリアル実施に伴う前提条件と、例題プログラムに関連する資源を用意します。

- 1) 「5. Enterprise Developer JCL チュートリアル」を終了していることを前提とします。
- 2) 対象データベースクライアントがインストール済で接続確認済であることを前提とします・
- 3) キットに添付されている DBtutorial.zipをC:¥ 直下へ解凍し、C:¥DBtutorialを作成します。

本書作成時に使用したデータベース環境は下記の通りです。

Oracle

Oracle クライアント 11g 11.2.0.4.0 64bit DB: リモート環境 (Oracle 10g 10.2.0.1.0 64bit) (Oracle Pro*COBOL プリコンパイラ使用)

DB2

DB2 クライアント 9.7.900.250 64bit DB: ローカル環境 (DB2 9.7.900.250 64bit) (DB2 PREP プリコンパイラ使用)

SQL Server

ODBC ドライバー: SQL Server Native Client 11.0 DB: ローカル環境 (SQL Server 2012 R1 64bit) (ODBC 使用)

7.3 プリコンパイルについて

埋め込み SQL 付き COBOL ソースは下記 2 通りのプリコンパイル方法があります。

- 1) 予め Micro Focus 形式の COBOL ソースを作成してプリコンパイルする。
- 2) 製品機能のプリプロセッサ機能からプリコンパイラを呼び出し内部的にプリコンパイルする。

ここでは、オリジナルソースイメージでデバッグが可能かつ管理が容易な2)を紹介します。

JCL バッチからの使用方法として紹介していますが、CICS や IMS プログラムで Oracle, DB2, SQL Server などとの 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設定となります。

7.4 プロジェクトの作成とプログラム取り込み

1) スタートアイコンから Enterprise Developer for Visual Studio 2012 を起動します。



2) 新しいプロジェクトを作成します。

プロダウンメニューの [ファイル]>[新規作成]>[プロジェクト] を選択します。

	スタート ページ - Microsoft Visual S	Studio					クイック起	動 (Ctrl+Q)	P
771	(ル(F) 編集(E) 表示(V) デバッグ(D)	チ–Д(M)	SQL	(Q)	ツール(T)	テスト(S)	分析(N)	ウィンドウ(W)	へルプ(H)
	新規作成(N)		•	1	プロジェクト(P)		Ctrl+S	hift+N
	開<(O)		Þ		Web サイト	(W)		Shift+	Alt+N
	閉じる(C)				チーム プロジ	ኦェクト(丅)			
×	ソリューションを閉じる(T)			Ď	ファイル(F)	•		Ctrl+N	L
	選択されたファイルを上書き保存(S)	Ctrl+S			既存の⊐−ト	からプロジェク	トを作成(E)		
	選択したファイルに名前を付けて保存(A)			7					
ъ ⁶	すべてを保存(L)	Ctrl+Shift	+S	dov	vs 8				
	テンプレートのエクスポート(E)			dov	/S				
	ソース管理(R)		Þ	re					
₽	ページ設定(U)			l					
	印刷(P)	Ctrl+P		nov	/S				
	最近使ったプロジェクトとソリューション(J)		Þ	ie č					
×	終了(X)	Alt+F4		eP	oint				

- 3) プロジェクトの種類と名前、場所を指定します。
 - 1. ペインツリーの [テンプレート]>[COBOL プロジェクト]>[Native] 内から "メインフレーム サブシステム アプリ ケーション"を選択します。
 - 2. [名前] は "DBDEMO" を入力します・
 - 3. [場所] は 既存 C:¥work フォルダを指定します。
 - 4. [ソリューションのディレクトリを作成] のチェックを外して [OK] をクリックします。

▶ 最近使用したファイル		.NET F	ramework 4.5 🔹 並べ替	討基準: 既定	Ŧ	# 🗉 インス	トール済み 🔎 -
▲ インストール済み		CBL	Enterprise Server アプー	・COBOL プロジェクト	種類: COBC	DL プロジェクト	
▲ テンプレート ▲ COBOL プロジェ Cloud	クト	CBL	Windows アプリケーション	COBOL プロジェクト	Micro Focus CICS®、JCL 作成するための	Server 配下で または IMS®))プロジェクトです。	実行される アプリケーションを
Database Managod			メインフレーム サブシステム…	COBOL プロジェクト			
Native Web		C:\	コンソール アプリケーション	COBOL プロジェクト			
▷ Visual C# ▷ PL/I			リンク ライブラリ	COBOL プロジェクト			
LightSwitch ▷ 他の言語			空のプロジェクト	COBOL プロジェクト			
▶ Other Project サンプル	Types		Micro Focus INT/GNT	COBOL プロジェクト			
▶ オンライン							
名前(N):	DBDEMO						
場所(L):	C:¥work¥			*	参照(B)		
ソリューション(S):	新しいソリューション	を作成す	5	*			
ソリューション名(M):	DBDEMO				□ ソリューションの	ディレクトリを作成	(D)
					□ ソース官理に追	2JUL(U)	
						ОК	キャンセル

4) プロジェクトへ例題プログラムを追加します。

ソリューションエクスプローラでプロジェクトを右クリックし、[追加]>[既存の項目] を選択すると、ファイル選択のためのエ クスプローラが表示されます。



5) [すべてのファイル]を選択して、展開した C:DBtutorial 直下に存在する全ファイルを選択し [追加] をクリックしま す。

长 - م 🕑 🔹 🛞	ビューター 🕨 ローカル ディスク (C:) 🕨 DBtutorial		∨ C DBtutor	ialの検索	Q
整理 ▼ 新しいフォルダー				•== •	
デスクトップ ^	名前	更新日時	種類	サイズ	
 過 最近表示した場所 	lbdemo1.jcl	2013/09/19 16:57	JCL ソースコード	2 KB	
<u>™</u> = / Ť=u	TBLCRTE.cbl	2013/09/20 12:02	COBOL ソースファ	1 KB	
(a) 71/79 ○ 51/79	D TBLDROP.cbl	2013/09/20 12:02	COBOL ソースファ	1 KB	
	TBLFETC.cbl	2013/10/07 15:06	COBOL ソースファ	2 KB	
	TBLISRT.cbl	2013/10/07 15:06	COBOL ソースファ	2 KB	
♪ ミュージック ペ ホームグループ					
🌉 コンピューター					
🊢 ローカル ディスク ((
👸 CD ドライブ (D:) ' 🗸					
ファイルな	名(N): 「TBLISRT.cbl" "dbdemo1.jcl" "TBLCR"	TE.cbl" "TBLDROP.cbl"	' "TBLF 〜 【すべてのス 追加(A	ゆイル (*.*) A) ▼ キ	マ ヤンセル

6) ソリューションエクスプローラへ追加されたファイルが表示されます。



- 7.5 プロジェクトと個別ファイルプロパティの設定
 - 1) プロジェクトのプロパティを設定します。
 - ① ソリューション エクスプローラ内のプロジェクト [Properties] をダブルクリックし、設定ウィンドウを開きます。



アプリケーション	構成(C): 該当なし	>フォー /、(M): 該当対 /
SQL		
従属パス		
BMS	出力の名前:	
IMS	MSSDEMO 出力の種類:	(1) 補足情報
COBOL	INT/GNT v	[リンクライブラリ] を指定すると".DLL"が作成されます。
COBOL リンク	エントリポイント:	
アセンブラ	(設定なし)	
デバッグ		環境(E)

② ペインツリーの [アプリケーション]>[出力の種類] は [INT/GNT] オブジェクトタイプを選択します。

- ③ ペインツリーの [COBOL] を選択し、以下の項目を設定します。
 - 1. [全般]>[プラットフォームターゲット] は "x64" を選択します。
 - 2. [全般]>[COBOL方言] は "Enterprise COBOL for z/OS" を選択します。
 - 3. [全般]>[文字セット] は "ASCII" を選択します。
 - 4. [全般]>[.GNT にコンパイル] ヘチェックをして有効にします。
 - 5. [全般]>[デバッグ用にコンパイル] ヘチェックをして有効にします。

全船
全船
±22
プラットフォームターゲット: ×64 ✓
COBOL 方言・ Enterprise COBOL for z/C × 文字セット・ ASCII ×
メインフレーム
□ EXEC CICS オブション □ EXEC DLI オブション
エラーおよび警告
警告レベル: 情報メッセージを含める(レベル I) >
最大エラー数: 100 営告をエラーとして処理
出力
出力パス: .¥bin¥x86¥Debug¥ 参照
□ 指令ファイルの生成 □ リストファイルを生成
追加指令
A
· · · · · · · · · · · · · · · · · · ·
ビルド設定 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一
dialect"ENTCOBOL" anim cobidy".¥bin¥x86¥Debug¥" warnings"3" max- error"100" charset"ASCII"

「① 作成するオブジェクト;ビット数 = ② XA リソース;ビット数 = ③ Enterprise Server;ビット数 = ④ DB;許容するビット数 」 が正常に稼働するために必要な条件です。

上記 [プラットフォームターゲット] 指定は①にあたり、今回は 64 ビットオブジェクトを作成するために指定します。

プロジェクト単位に指定可能です。

④ プロパティ設定ファイルを上書き保存します。

ファイル(F)	編集(E)	表示(V)
G - O	17 - 省	

- 2) 個別ファイルのプロパティを設定します。
- ① ソリューション エクスプローラ内の ".cbl" ファイルを選択して右クリックし [プロパティ] を選択すると、設定ウィンドウ が開きます。



 ペインツリーの [COBOL] を選択し、使用するデータベース製品に合わせ、埋め込み SQL 対応のプリプロセッサの 設定を追加します。

1. Oracleの例:

COBSQL プリプロセッサを使用するため、以下の指令を追加指令欄へ追加します。 ^{*}DIALECT(ENTCOBOL) P(COBSQL) ENDP"

OOBOL 方言: Enterprise COBOL for z/OS デパッグ用にコンパイル: (はい CIOS 指令: なし DLI 指令: (以、え エラーおび習告 (以、え 最大工ラー説: 100 学告をエラーとして処理: (以、え レンス 警告をエラーとして処理: しいえ (以、え 加指令 DIALECT(ENTCOBOL) P(COBSOL) ENDP () 指令ファイルの生成: (以、え リストファイルを生成: (以、え GNT (ニコンパイル: (は、) ビレド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定	OOBOL 方言: Enterprise COBOL for z/OS デパッグ用にコンパイル ばな) GICS 指令: なし DLI 指令: (パルネ マラーカよび習苦 最大エラー数: 100 警告をエラーとして処理: (パルネ ビルンル: 「情報メッセージを含める(レベル D) 運動指令 DIALECT(ENTCOBOL) P(COBSQL) ENDP より 出か とハ ビレンパト (パルネ ジリストファイルの生成: (パルネ ジレトド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 通知指令 - P(COBSQL) ENDP レトド設定				
デバッグ用にコンパイル: (はい マ CICS 指令: なし マ DLI 指令: (い.)え マ 重ちてう一致: 100 マ 警告をエラーとして処理: (い.)え マ 警告をエラーとして処理: (い.)え マ 警告をレベル: 「指報スッセージを含める(レベル D) マ 道加指令 (い.)え マ DIALECT(ENTCOBOL) P(COBSOL) ENDP マ 生た/ (い.)え マ 生た/ (い.)え マ ビントファイルの生成: (い.)え マ ウストファイルの生成: (い.)え マ ウストファイルの生成: (い.)え マ ウストファイルの生成: (い.)え マ ウストファイルの生成: レントフシュクトのビルド設定 ● DIALECT(ENTCOBOL) プロジェクトのビルド設定 ● ●	デバッグ用にコンパイル・ (はい × OICS 指令: なし × DLI 指令: (×、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、	 COBOL 方言:	Enterprise COBOL for z/OS	*	
CIOS 指令: なし マ DLI 指令: (X.)え マ エラーおよび警告 マ 最大エラー数: 100 マ 警告をエラーとして処理: (X.)え マ 警告をエラーとして処理: (X.)え マ 警告レベル: 情報スッセージを含める(レベル D) マ 適加指令 DIALECT(ENTCOBOL) P(COBSOL) ENDP マ 法プ 指令ファイルの生成: (X.)え マ リストファイルを生成: GNT にコンパイル: はい、マ DIALECT(ENTCOBOL) プロジェクトのビルド設定	CICS 指令: なし マ DLI 指令: (10.1次 マ 正うーおび警告 最大エラー数: 100 マ 警告をエラーとして処理: (14.1次 マ 警告レベル: 情報メッセージを含める(レベル D) マ 追加指令 DIALECT(ENTCOBOL) P(COBSOL) ENDP マ 出力 出力 出力 ビレトジを生め: 「以注 (14.1) マ ビレト設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 」通加指令 - P(COBSOL) ENDP	デバッグ用にコンパイル	はい	¥	
DLI指令: (いいえ) v エラーおよび警告 最大エラー数: 100 v 警告をエラーとして処理: (いいえ) v 警告レベル: 情報メッセージを含める(レベル D) v 適加指令 DIALECT(ENTCOBOL) P(COBSOL) ENDP v 法 増合ファイルの生成: (い、え) v リストファイルを生成: リストファイルを生成: GNT (こコンパイル: はい) v ビルド設定	DLI指令: (いいえ) マ エラーおび警告 最大エラー数: 100 マ 警告をエラーとして処理: (いいえ) マ 警告レベル: 情報メッセージを含める(レベル D) マ 追加指令 DIALECT(ENTCOBOL) P(COBSQL) ENDP マ 出力 出力 出力 日ALTT(ニンパイル: (いいえ) マ 日本 日本 日本 日本 日本 日本 日本 日本 日本 日本 日本 <td< td=""><td>CICS 指令:</td><td>なし</td><td>¥</td><td></td></td<>	CICS 指令:	なし	¥	
エラーおよび警告 最大エラー説: 100 警告をエラーとして処理: (いいえ 警告レベル・ 100 100 100 100 100 100 100 100 100 100 100 100 1100	エラーおよび警告 100 マ 最大エラー数: 100 マ 警告をエラーとして処理: (¼、½、 マ 管告レベル: 情報メッセージを含める(レベル D) マ 適加指令 0 マ DIALECT(ENTCOBOL) P(COBSQL) ENDP マ 出方 マ 出方 マ 出方 マ 日AL5ファイルの生成: (¼、½ 以注 マ 以注 (¼、½ ロAL5ファイルを生成: (¼、½ QNT (こフンパイル: (¼、) ビレト設定 0 DIALECT(ENTCOBOL) プロジェクトのビルド設定 0 通知指令: P(COBSQL) ENDP ショ動加指令: マ	DLI 指令:	いいえ	~	
 最大工ラー数: 100 v 警告をエラーとして処理: (いいえ v 警告レベル: 情報3ッセージを含める(レベル D) v 道加指令 DIALECT(ENTCOBOL) P(COBSQL) ENDP v 出プ 出プ 指令ファイルの生成: (いいえ v リストファイルを生成: (いいえ v リストファイルを生成: (いいえ v ビいえ v ビいえ v ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 v 	最大エラー数: 100 ▼ 智告をエラーとして処理: (ヾいえ ▼ 智告レベル: 情報メッセージを含める(レベル D) ▼ 違加指令 DIALECT(ENTCOBOL) P(COBSOL) ENDP よ お 広/ 広/ 近いていたいたいたいたいたいたいたい 日本の していたいたいたいたいたいたいたいたいたいたいたいたいたいたいたいたいたいたいた	エラーおよび警告			
警告をエラーとして処理: (いいえ v 警告レベル: 情報3メッセージを含める(レベル D v 追加指令 DIALECT(ENTCOBOL) P(COBSQL) ENDP レレビングロングロングを含める(レベル D v 追加指令 ロALECT(ENTCOBOL) P(COBSQL) ENDP レレビングロングロングを含める(レベル D v ロングロングレングロングを含める(レベル D v ロングロングレングロングを含める(レベル D v ロングロングレングロングを含める(レベル D v ロングロングレングロングレングロングレングロングレングロングロングレングロングロングレングロングロングレングロングロングロングロングロングロングロングロングレンジを含める(レベル D v ロングロングロングロングロングロングロングロングロングロングロングロングロングロ	警告をエラーとして処理: (いいえ v) 警告レベル: (接種はシャセージを含める(レベル D) v) 通知指令 DIALECT(ENTCOBOL) P(COBSQL) ENDP v) 法カ 指令ファイルの生成: (いいえ v) リストファイルを生成: (いいえ v) GNT (こフンパイル: (はい v) Eルド設定 DIALECTENTCOBOL) プロジェクトのビルド設定 v) 通知指令 - P(COBSQL) ENDP v)	最大エラー数:	100	~	
警告レベル: 情報扱ッセージを含める(レベル D 、 道加指令 DIALECT(ENTCOBOL) P(COBSQL) ENDP DIALECT(ENTCOBOL) P(COBSQL) ENDP レストファイルの生成: (、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、、	警告レベル 情報初ッセージを含める(レベル D 、 追加指令 DIALECT(ENTCOBOL) P(COBSQL) ENDP 、 エカ 指令ファイルの生成: (いいえ 、 リストファイルを生成: (いいえ 、 GNT (ニコンパイル: はい 、 ビレド該定 DIALECT(ENTCOBOL) プロジェクトのビルド設定) 追加指令 - P(COBSQL) ENDP 、	警告をエラーとして処理:	いいえ	~	
遠加指令 DIALECT(ENTCOBOL) P(COBSQL) ENDP カノ 指令ファイルの生成: (いいえ ∨ リストファイルを生成: (いいえ ∨ GNT (にコンパイル: はい ∨ ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定	追加指令 DIALECT(ENTCOBOL) P(COBSOL) ENDP	警告レベル:	情報メッセージを含める(レベル 1)	~	
DIALECT(ENTCOBOL) P(COBSQL) ENDP 造パ」 指令ファイルの生成: リストファイルを生成: 「いいえ マ GNT (ごつンパイル: ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定	DIALECT(ENTCOBOL) P(COBSQL) ENDP 法グ 指令ファイルの生成: (いいえ) リストファイルを生成: (いいえ) (いえ) (い	追加指令 ——			
出 指令ファイルの生成: りストファイルを生成: 「以いえ くのれてにコンパイル: (はい じルド設定 DIALECT(ENTCOBOL) クロジェクトのビルド設定	法/j 指令ファイルの生成: (いいえ ~ リストファイルを生成: (いいえ ~ GNT (ニコンパイル: (はい ~ ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 通知指令 - P(COBSOL) ENDP	DIALECT(ENTCOBOL)	P(COBSQL) ENDP	A	
出 指令ファイルの生成: りストファイルを生成: 気NT (ごユンパイル: じルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定				Ü.	
お命ファイルの生成 (い、 、 、 、 、	指令ファイルの生成: リストファイルを生成: GNT (ニコンパイル: ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 ショ動加指令 - P(COBSOL) ENDP				
18日ファイルのビルズ しれた マ リストファイルを生成 <u>「いいえ</u> マ GNT にコンパイル・ はい マ ビルド設定	18年ライルの主が、 リストファイルを生成: GNT にコンパイル: ビルで設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 ショ動加指令 - P(COBSOL) ENDP	地合う人口の生成。	1515		
9ストンアハルを主か。 GNT (ニコンパイル・ (はい	GNT (コンパイル: しいな GNT (コンパイル: はい ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 注助加指令 - P(COBSOL) ENDP	指示ファイルの生成。	600X	· · · ·	
GNT にコンパイル・ UU、 ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・	GNILコンパイル: [31/1 ~ ~ ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 3箇切指令 - P(COBSOL) ENDP	リストノアイルを主が	<u>1.1.1/2</u>	······································	
ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定	ビルド設定 DIALECT(ENTCOBOL) プロジェクトのビルド設定 3箇加指令 - P(COBSQL) ENDP	GNT (232/1917)	(20)	*	
DIALECT(ENTCOBOL) プロジェクトのビルド設定	DIALECT(ENTCOBOL) クロジェクトのビルド設定)意加指令 - P(COBSQL) ENDP	ビルド設定			
追加指令 - P(COBSQL) ENDP	~ · · · · · · · · · · · · · · · · · · ·	DIALECT(ENTCOBOL) 追加指令 - P(COBSQ	ブロジェクトのビルド設定 L) ENDP	^	
	· · · · · · · · · · · · · · · · · · ·				
				v.	

2. DB2の例:

DB2 ECM プリプロセッサを使用するため、以下の指令を追加指令欄へ追加します。 *DIALECT(ENTCOBOL) DB2(DB=LCLDB2,VALIDATE=RUN)" *DB="に続く名称は DB2エイリアス名です。

COBOL	全般				
SQL	COBOL 方言:	Enterprise COBOL for z/OS	~		
	デバッグ用にコンパイル・	はい	~		
	CICS 指令:	なし	~		
	DLI 指令:	いいえ	~		
	エラーおよび警告				
	最大エラー数:	100	¥		
	警告をエラーとして処理:	いいえ	~		
	警告レベル	情報メッセージを含める(レベル I)	~		
	追加指令 ——				
	DIALECT(ENTCOBOL)	DB2(DB=SAMPLE,VALIDATE=RUN)			
	871				
	指令ファイルの生成:	()()Ž	~		
	リストファイルを生成・	()()7			
	ONT (TO) JEAN	(#())	*		
	CINT CLU7 Y176	(6(6))	v		
	ビルド設定				
	DIALECT(ENTCOBOL) 追加指令 - DB2(DB=	プロジェクトのビルド設定 SAMPLE,VALIDATE=RUN)	^		
			~		
			ОК	キャンセル	適用(A)

3. SQL Server の例:

OPENESQL プリプロセッサを使用するため、以下の指令を追加指令欄へ追加します。 ODBCを指定するため、使用する ODBC データソースの追加が必要です。 *DIALECT(ENTCOBOL) SQL(DBMAN=ODBC,BEHAVIOR=JCL,TARGETDB=MSSQLSERVER)"

COBOL	全般				
SQL	COBOL 方言:	Enterprise COBOL for z/OS	~		
	デバッグ用にコンパイル・	(はい	~		
	CICS 指令:	なし	~		
	DLI 指令:	いいえ	~		
	エラーおよび警告				
		100	~		
	警告をTラーとして処理:	いいえ	~		
	警告レベル:	情報メッセージを含める(レベル D)	~		
	追加指令				
	DIALECT(ENTCOBOL)	SOL			
	(DBMAN=ODBC,BEHA	VIOR=JCL,TARGETDB=MSSQLSER	VER)		
	1277				
	指令ファイルの生成	いいえ	~		
	リストファイルを生成:	いいえ	~		
	GNT にコンパイル	はい	~		
	ビルド設定				
	DIALECT(ENTCOBOL)	プロジェクトのビルド設定	^		
	追加指令 - SQL (DBMAN=ODBC.BEHA	VIOR=JCLTARGETDB=MSSQLSER	VER		
)		~		
			OK	キャンセル	適田(A)

7.6 XA スイッチモジュールの生成

この環境で実行するプログラム内では、XA スイッチモジュール経由でデータベースと接続することになります。 ここでは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。作成手順は以下の通りです。

1) スタートアイコンから Enterprise Developer 2012 コマンドプロンプト(64-bit)を管理者として実行します。



2) プリコンパイルを行うため下記フォルダを書き込み権限があるフォルダ配下へコピーします。当チュートリアルでは C:¥直下 へコピーします。

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

【理由②】製品関連フォルダの書き込み権限によるトラブルを避ける

【コピー元フォルダ】

C: ¥Program Files (x86) ¥Micro Focus ¥Enterprise Developer for Visual Studio 2012 ¥src ¥enterprises erver ¥xa

【コピー先フォルダの例】

C:¥xa

鷆 « src 🕨 ente	erpriseserver >	v C er	nterpriseserverの検乳
^	名前	更新日時	種類
	🚹 exits	2014/02/19 15:01	ファイル フォルダー
	🔰 xa	2014/04/03 10:23	ファイル フォルダー
🏭 ト コンピューター	▶ ローカル ディスク (C:) ▶	-□ 5 v	カル ディスク (C:)の検索
^	名前	更新日時	種類
	🗟 msdia80.dll	2006/12/01 23:37	アプリケーション拡張
	\mu work	2014/04/03 10:56	ファイル フォルダー
た場所	🔰 xa	2014/04/03 10:47	ファイル フォルダー

3) コマンドプロンプトから C:¥xa へ移動します。



- 4) Oracle を使用する場合
- ① XA スイッチモジュールを build コマンドで作成します。

"build < ora10 または ora11 >" を実行します。ここでは 11g を使用するため下記コマンドを実行します。 "build ora11"



 JCL機能でIKJEFT01ユーティリティを使用する場合は、xxx1pc.dll サブモジュール生成も必要になるため、下記コ マンドを実行します。

"build ora1pc"



正常に作成されました。メッセージにあるように "ESORAOPC.DLL" ファイルを製品の bin フォルダヘコピーします。

例)C:¥Program Files (x86)¥Micro Focus¥Enterprise Developer¥bin64

- 5) DB2 を使用する場合
- ① XA スイッチモジュールを下記 build コマンドで作成します。

"build db2"

管理者: Enterprise Developer 2012 コマンドプロンプト (64-bit) - □
C:¥Users¥tarot¥Documents>cd c:¥xa
c:¥xa>build db2 Building 64-bit switch module Micro Focus COBOL - CBLLINK utility Version 2.2.0.93 Copyright (C) Micro Focus 1984-2013. All rights reserved.
Micro Focus COBOL Version 2.2.00151 Copyright (C) Micro Focus 1984-2013. All rights reserved. * チェック終了:エラーはありません- コード生成を開始します * Generating ESDB2XA * Data: 3784 Code: 7424 Literals: 992 Microsoft (R) Incremental Linker Version 11.00.50727.1 Copyright (C) Microsoft Corporation. All rights reserved.
ESDB2XA.obj cbllds000011B4.obj Creating library ESDB2XA.lib and object ESDB2XA.exp Microsoft (R) Manifest Tool version 6.2.9200.16384 Copyright (c) Microsoft Corporation 2012. All rights reserved.
c:¥xa>_

- ② build db21pc < DBのエイリアス > を実行します。ここでは下記コマンドを実行します。
 "build db21pc LCLDB2"
 - DB2 ヘアクセスするためのログイン画面が表示されますので、ユーザ ID とパスワードを入力します。

Micro Focus SQL Logon
Connect failed with SQL code -30082. SQL database SAMPLE
Enter user name and password:
User name: db2inst1
Password: *****
□ <u>S</u> ave
Save to INI file
OK Cancel

正常に作成されました。メッセージにあるように "ESDB2OPC.DLL" ファイルを製品の bin フォルダヘコピーします。

例)C:¥Program Files (x86)¥Micro Focus¥Enterprise Developer¥bin64

🔤 管理者: Enterprise Developer 2012 コマンドプロンプト (64-bit) 🛛 - 🔍 🎫
c:¥xa>build db21pc LCLDB2 Building 64-bit switch module Micro Focus COBOL Version 2.2.00151 Copyright (C) Micro Focus 1984-2013. All rights reserved. * チェック終了:エラーはありません。コード生成を開始します * Generating ESDB2OPC * Data: 5856 Code: 10834 Literals: 976 Micro Focus COBOL - CBLLINK utility Version 2.2.0.93 Copyright (C) Micro Focus 1984-2013. All rights reserved.
Microsoft (R) Incremental Linker Version 11.00.50727.1 Copyright (C) Microsoft Corporation. All rights reserved.
ESDB20PC.obj cbllds00000804.obj Creating library ESDB20PC.lib and object ESDB20PC.exp Microsoft (R) Manifest Tool version 6.2.9200.16384 Copyright (c) Microsoft Corporation 2012. All rights reserved.
If you have upgraded your installation from an earlier version of Net Express or Studio, then you may need to additionally rebuild the main switch module, that is, execute :
build db2
Unable to locate the COBOL bin64 directory.
If you intend to execute JES-initiated transactions under Enterprise Server you must copy ESDB20PC.DLL to a directory on your PATH, such as your COBOL bin64 directory.
If you do not do so, then such transactions will not be able to communicate with the database server.
c:¥xa>

- 6) SQL SERVER を使用する場合
- ① XA スイッチモジュールを下記 build コマンドで作成します。

"build mssql"



xaswitch.lib の読み込み失敗でリンクエラーとなる場合は、環境変数「LIB」へ左記ファイルパスを

追加し、コマンドプロンプトを再起動後に再ビルド成してください。

例:C:¥Program Files (x86)¥Windows Kits¥8.0¥Lib¥win8¥um¥x64

⊕ ∋ - ↑ 퉬 « V	Vindows Kits → 8.0 → Lib → win8 → um → x64	~ Č	x64の検索
☆ お気に入り ■ ダウンロード	▲ 名前 聞 Xāudio2.iib	更新日時 2012/0//25 16:0/	種類 Object File Libr
■ デフカトップ		2012/07/25 10:11	Object File Libi
	システム変数の編集		
変数名(<u>N</u>):	LIB		
変数値(<u>∨</u>):	x86)¥Windows Kits¥8.0¥Lib¥win8¥um¥x64		
	OK キャンセル		

② 使用するビット数に合わせた ODBC データソースを追加します。

1. [コントロールパネル]>[管理ツール]>[ODBC データソース(64 ビット)]を選択します。

| 💮 ▶ コントロール パネル ▶ すべてのコントロール パネル項目 ▶ 管理ツール

	^	名前	
		🔝 iSCSI イニシェーター	
		📷 ODBC データ ソース (32 ビット)	
た場所		📷 ODBC データ ソース (64 ビット)	

2. [システム DSN]タブを選択して[追加]ボタンをクリックします。

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 追加(D)	

3. [SQL Server Native Client 11.0]を選択して[完了]ボタンをクリックします。

SQL Server	6.02.9200.16384
SQL Server Native Client 11.0	2011.110.3000.0
<	>
< 戻る(<u>B</u>) 完了	キャンセル

4. [名前]、[サーバー]を入力して[次へ]ボタンをクリックします。

・名前 ・・・ Enterprise Server の XA リソース定義で使用する DSN 名称です。

・サーバー ・・・ DB が存在する場所を選択します。このチュートリアルではローカルに DB が存在しています。

新しいデータソースの名前を入力してください。
名前(M): SQLSVR
データ ソースについての説明を入力してください。 説明(<u>D</u>):
接続する SQL Server を選択してください。
サーバー(S): (local)
完了 次へ(N) > キャンセル ヘルプ

5. [認証方法]、[対象データベース]などを入力後、最終ウィンドウで[データソースのテスト]ボタンをクリックして接続を 確認してください。

Micro Focus メインフレームソリューション スターターズキット データベース連携編

③ XA スイッチモジュールを有効にするため下記設定を確認します。

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



2. [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]にチェックされていることを確認、もしくはチェックして有効にします。

ローカル DTCのプロパティ ?	×
- セキュリティの設定 □ ネットワーク DTC アクセス(D) - クライアントおよび管理	
□ リモート クライアントを許可する(R) □ リモート管理を許可する(M)	
トランザクション マネージャー通信 受信を許可する(I) 送信を許可する(W)	
 相互認証を必要とする(L) 着信呼び出し側には認証を必要とする(G) 認証を必要としない(T) 	
 ✓ XA トランザクションを有効にする(X) ✓ SNA LU 6.2 トランザクションを有効にする(E) 	

7.7 プログラムのコンパイル

1) ソリューションエクスプローラで4つのコボルファイルを選択後、右クリックで[コンパイル]を選択します。



2) 出力ウィンドウで正常に完了したことを確認します。

出力								
出力元の表示(S):	ビルド				-	: E ·	<u>*</u>	ab c 🔁
* Data: 133	3208	Code:	2779	Literals:	416			
* C:¥work¥DBDEMO	D¥TBLFETC	licbl のコンパ	イル中					
* Generating .¥b	bin¥x86¥D	ebug¥TBLFETC						
* Data: 133	3280	Code:	6119	Literals:	848			
* C:¥work¥DBDEMO	D¥TBLISRT	icbl のコンパ	「イル中					
* Generating .¥b	bin¥x86¥D	ebug¥TBLISRT						
* Data: 133	3784	Code:	4427	Literals:	848			
	COBOL 🗆	ンパイル:41	固 正常終	了または最新の状態	0 個	失敗。		

7.8 Enterprise Server の設定

本チュートリアルでは例題 JCL を Enterprise Server にサブミットして実行します。64 ビットオブジェクトを作成しているため、64 ビットの Server を準備します。

- 1) Enterprise Server Administration ウィンドウで [追加] ボタンをクリックし下記を設定します。(1/3 ページ)
- ① [サーバー名] には "DBDEMO64" を入力します。
- ② [動作モード] では "64-bit" を選択します。
- ③ [次へ] ボタンをクリックします。

	※Enterprise Server Administration > サーバー追加 TOK-kt-W8v1.microfocus.com (10.18.11.153:86)	Version 1.18.42 Enterprise Server (MSS LICENSED)
Home アクション アドレフ 更新	ス テ MDS0000I OK タス	Wed Mar 19 11:22:53 2014
エクスポート インポート すべて削除	サ <i>ーバー追加</i> (Page 1 of 3):	[mfuser] [Page id: a001]
構成 オプション セキュリティ	サーバー名: DBDEM064	
表示 ディレクトリ 統計 セッション	動作モード: ③ 32-bit ④ 64-bit You cannot change your choice of working mode once a server is creat change it when conving or importing a server	ted, although you can
ヘルブ このページ Support Feedback	thange it when copying of impolling a server. キャンセル 次へ >>	

2) [サーバータイプ] では [Micro Focus Enterprise Server with Mainframe Subsystem Support]を選択 し、[次へ] ボタンをクリックします。(2/3 ページ)

Home アクション アドレス更新	ス MDS0000I OK タ ス	Wed Mar 19 11:25:40 2014
エンスホート インポート すべて削除	サ <i>ーバー追加</i> (Page 2 of 3):	turne 1 to -2
構成 オプション セキュリティ	サーバー名: DBDEMO6	
*=	サーバータイプ:	
衣示 ディレクトリ 統計 セッション ジャーナル	MFES Micro Focus Enterprise Server An enterprise server that provides an programs running as services in a ser	execution environment for COBOL application vice orientated architecture.
ヘルブ このページ Support Feedback	Micro Focus Enterprise Server with An enterprise server that also provides applications that have been migrated	h Mainframe Subsystem Support s an execution environment for CICS from the mainframe.
	You can change your choice of server type later.	
	<<戻る 次へ>>	

[64-Bit Working Mode] ヘチェックされていることを確認し、[TN3270 リスナーの作成] のチェックをはずします。
 [追加] ボタンをクリックします。(3/3 ページ)

共有メモリページ数: 512	サービス実行プロセス: 2
共有メモリクッション: 32	トレーステーブルサイズ: 341
ローカルトレースサイズ:341	診断ファイル最大サイズ: 0
要求ライセンス: 10	
コールドスタート診断ファイル: 🔽	システムアベンド時ダンブ: 🔽
補助トレースアクティブ: 🗌	ローカルコンソールを表示: 🗌
Mainframe Subsystem Support: 🗹	64-Bit Working Mode: 🗹
トレースフラグ:	
タスク管理 🗌 ストレージ管理	里口 テーブル管理 🗌
アプリケーションコンテナ 🗌 🛛 要求ハンド	ラ 🗌 RMインタフェース 🗌
通信 🗌 アプリケーション	ンロ 終了日
生成オプション:	

4) 管理ウィンドウから [編集] ボタンをクリックし、[XA リソース] タブを開き、[追加] ボタンをクリックします。



🔺 🚽 🕨 Server MSSDEMO [停止]	
サーバー… リスナー (3) サービス (4) ハンドラ (3) パッケージ (0)	
プロパティ 構成 診断 過去の統計	
一般 XAリソース (0) MSS (✔) スクリプト アクセス権 セキュリティ	
(追加)	

- 5) 使用するデータベース製品に合わせた XA リソースの下記項目を設定します。
 - ① [ID]: IKJEFT ユーティリティに渡す DSN TSO コマンドの SYSTEM パラメタで渡す DB2 サブシステム名を指定 します。(このチュートリアルでは XADB とします)
 - ② [名前]: Oracle の場合は "ORACLE_XA" を指定し、他はリソース名として任意の名前を指定します。
 - ③ [モジュール]:前項で作成した XA スイッチメインモジュールのパスと DLL ファイル名を指定します。(xxx1pc.dll は指定無し)

<<DB に依存した当チュートリアルで指定するべき値の例>>

Oracle
"C:¥xa¥ESORA11XA.DLL"
DB 2

"C:¥xa¥ESDB2XA.DLL"

- SQL Server
 "C:¥xa¥ESMSSQL.DLL"
- ④ [OPEN 文字列]: DB オープンのために必要な値を指定します。(設定済みの DB 環境設定に依存)
 - 1. Oracle の例

例えば以下のように設定し、[追加] ボタンをクリックします。名前と OPEN 文字の先頭は必ず "ORACLE_XA" を指定してください。

一般	XAリソース (3)	MSS (🖌)	MQ	スクリプト	アク
ID:					
XAI	DB				
名前	i:				
OR	ACLE_XA				
モジ	ュール:		-		
C:\x	a\ESORA11XA.DL	.L			
OPE	N文字列:				
OR	ACLE_XA+SesTm	=100+SqlNet=t	ok-par+A	cc=P/scott/tig	ger
CLO)SE文字列:				
説明	1:				
有効	:				
追	加 キャンセル				

2. DB2の例

例えば以下のように設定し、[追加] ボタンをクリックします。

一般)	XAリソース (1)	MSS (🗸)	MQ	スクリプト `	アク
ID: XAD 名前: DB2	іВ XA]		
C:\x	a\ESDB2XA.DLL				_
OPE	N文字列:				
DB=	SAMPLE,uid=db2	inst1,pwd=ibm	db2		
CLO	SE文字列:				
有効:					
追加	10 キャンセル				

3. SQL Server の例

例えば以下のように設定し、[追加] ボタンをクリックします。 前項で作成した ODBC データソース名称を OPEN 文字列に指定します。

· DSN=SQLSVR
一般 XAリソース (3) MSS (✓) MQ スクリプト アク
ID:
XADB
名前:
SQLSVR_XA
C:Xa\ESMSSQL.DLL OPEN计字列:
DSN=SQLSVB
CLOSE文字列:
説明:
有効: 🗹
追加キャンセル

6) JCL チュートリアルで実行したように、JCL を実行するために [JES] の設定を行い [Apply] ボタンをクリックします。

<mark>ロバティ…</mark> 構成 診断… 過去の統計	
般 🛛 XAリソース (3) 🕺 MSS (イ) MQ 】 スクリプト 🗋 アクセス	権 セキュリティ
インフレーム サブシステム サポート有効: 🗹	
CICS (🗸) JES (✓) IMS PL/I	
General Initiators (1) Printers (0)	
ジョブ入力サブシステム有効: 🗹	
JESプログラムパス:	
C:\work\DBDEMO\bin\x86\Debug	
	v
システムカタログ:	
C:\work\DBDEMO\DATAFILE\CATALOG.DAT	\bigcirc
データセットの省略時ロケーション:	
C:\work\DBDEMO\DATAFILE	\bigcirc
シュフテトプロシュージャライブラート	
SYS1.PROCLIB	
Fileshere Oreferenties Leasting	
	~
	\sim

7) [JES]>[Initiators] を作成し、実行する JCL のクラスが含まれるよう入力後、[追加] ボタンをクリックします。

CICS (✔) JES (✔) IMS PL/I
一般 Initiators (0) Printers (0)
Add Initiator
名前: JCLABC
Class:
ABC
說明:
257 ABC X
キャンセル 追加

- 8) サーバーにプロジェクトを関連付けます。
- 9) サーバー エクスプローラから [DBDEMO64] を開始します。



7.9 XA リソース正常ロードの確認

XA スイッチモジュールが Enterprise Server に正しくロードできると、Enterprise Server Administration のコン ソールログには以下のメッセージが出力されます。下記は SQL Server の例です。

A 🖪 I	► Server DBDEMO64 [開始 ✓]		
サーバ	<mark></mark> リスナー (2) サービス (4) ハンドラ (3) バッケージ (0)		
プロパラ	ティ 構成 <mark>診断</mark> 過去の統計		
トレー	ス ダンプ <mark>ESコンソール CSコンソール</mark>		
● Show last 10 lines from 1 to 10 of 39 total entries			
Entry	y Event Show Entire Log		
30	140319 13200222 17892 DBDEMO64 CASSI1426I TD non-recoverable cold started 13:20:02		
31	140319 13200243 17692 DBDEM084 CASSI1426I TD recoverable warm started 13:20:02		
32	140319 13200284 17892 DBDEMO84 CASSI1800I SEP initialization completed successfully 13:20:02		
33	140319 13200285 17892 DBDEMO84 CASXO00201 XADB XA interface loaded. Name(Micro Focus SQL Server XA struct), Registration Mode(Static) 13:20:02		
34	140319 13200488 17692 DBDEMO64 CASXO0015I XADB XA interface initialized successfully 13:20:03		
35	140319 13200509 17692 DBDEMO64 CASSI5001I PLTPI Phase 1 - No PLT Specified 13:20:04		
36	140319 13200530 17692 DBDEMO64 CASSI5040I Active SEP memory strategy set to x'00000001', retain count 100 13:20:05		
37	140319 13200571 5184 DBDEMO84 CASSI1600I SEP initialization completed successfully 13:20:05		
38	140319 13200572 15118 DBDEMO84 CASSI18001 CER 101 Certain and an and a second state 13/200/05		
39	140319 13200875 5184 DBDEMO84 CASSI50211 注意事項		
	ロードに失敗してもサーバーは開始されますので、ログ内容を確認してください。		

7.10 JCL の実行と確認

1) ソリューション エクスプローラから dbdemo1.jcl をダブルクリックしてエディタを開き、内容を確認します。



Micro Focus メインフレームソリューション スターターズキット データベース連携編

このジョブは4つのステップから構成され下記の様に連携されています。

- STEP01:
 DB にテーブルを新規作成
 STEP02:
 - JCL に記述された SYSIN データをテーブルに追加
- STEP03:
 テーブルからデータを全件読み出し SYSOUT へ出力
- STEP04:
 テーブルを削除
- 2) ソリューション エクスプローラから dbdemo1.jcl を右クリックして [サブミット JCL] を選択します。



3) 以下のように DBDEMO1 ジョブが実行された旨のメッセージが出力ウィンドウへ表示されます。



 以降は JCL チュートリアルを参考にして、スプールビューの Out Hold キューおよび Output キューを開き、実行された DBDEMO1 ジョブの結果を確認したり、バッチプログラムのデバッグも同様に実行してみてください。また、各 DB 付属の 管理ツールを使用すると、各ステップにおける DB テーブルの状況が確認できます。

以上

Micro Focus メインフレームソリューション スターターズキット データベース連携編