Micro Focus Enterprise Developer チュートリアル

メインフレーム COBOL 開発：データベース連携

Visual Studio 2017 編

1. 目的

本チュートリアルでは、Visual Studio 2017 を使用したメインフレーム COBOL プロジェクトの作成、コンパイル、データベース接続を使用した JCL の実行までを行い、その手順の習得を目的としています。

2. 前提

- 本チュートリアルで使用したマシン OS ： Windows 10 Enterprise
- 使用マシンに Microsoft Visual Studio 2017 がインストールされていること
- 使用マシンに Micro Focus Enterprise Developer 4.0 for Visual Studio 2017 がインストールされていること
- 使用マシンに対象のデータベースクライアントがインストール済みで、プリコンパイラを含め動作確認済みであること
- 使用したデータベースは以下の通り
  - Oracle サーバー：Oracle 10g 10.2.0.1.0 64bit
    クライアント：Oracle クライアント 11g 11.2.0.4.0 64bit（Pro*COBOL プリコンパイラ使用）
  - DB2 サーバー：DB2 11.1.0.1527 64bit
    クライアント：DB2 クライアント 9.7.900.250 64bit（DB2 PREP プリコンパイラ使用）
  - SQL Server サーバー：SQL Server 2012 R1 64bit
    クライアント：ODBC ドライバー SQL Server Native Client 11.0

3. チュートリアル手順の概要

1. チュートリアルの準備
2. Visual Studio の起動
3. メインフレーム COBOL プロジェクトの作成
4. プロジェクトプロパティの設定
5. ビルドの実行
6. XA スイッチモジュールの生成
7. Enterprise Server インスタンスの設定
8. Enterprise Server インスタンスの開始と確認
9. データベースアクセスを含む COBOL バッチプログラムの実行
10. Enterprise Server インスタンスの停止

3.1 チュートリアルの準備

例題プログラムに関連する資源を用意します。

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

2) Visual Studio のソリューションを保存する VS フォルダーを C:¥ 直下に作成します。

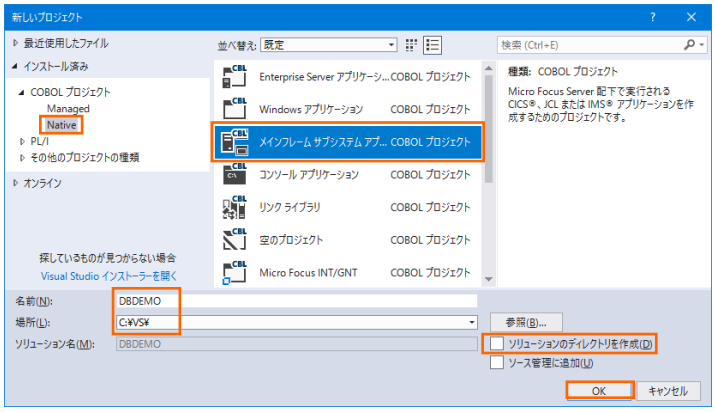
3.2 Visual Studio の起動

1) Visual Studio 2017 を起動します。

3.3 メインフレーム COBOL プロジェクトの作成

1) 新しいソリューションとプロジェクトを作成します。 [ファイル] プルダウンメニューから [新規作成] > [プロジェクト] を選択して [新しいプロジェクト] ウィンドウを表示し、下記項目を指定後 [OK] ボタンをクリックします。

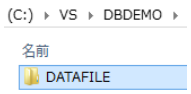
項目名	説明
左側ツリービュー	[インストール済] > [テンプレート] > [COBOL プロジェクト] > [Native] を選択します。
中央リスト	[メインフレーム サブシステム アプリケーション] を選択します。
名前	任意ですが、ここでは DBDEMO を入力します。
場所	前項で作成した C:¥VS を指定します。
ソリューションのディレクトリーを作成	ここではチェックをオフにします。



2) [ソリューション エクスプローラー] へ作成したプロジェクトが表示されます。

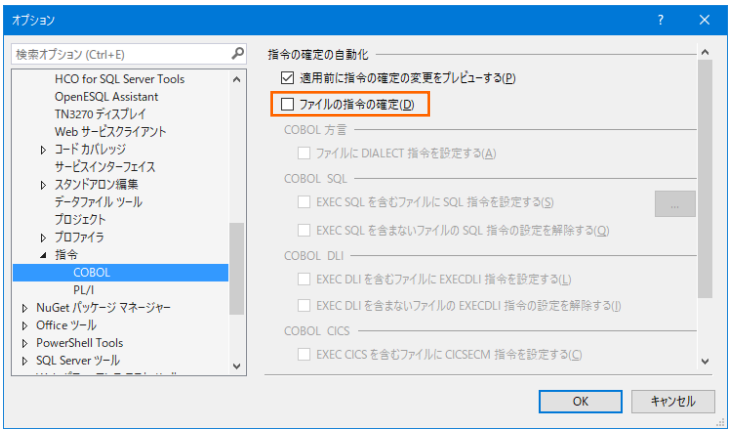
3) プロジェクトを作成したことにより C:¥VS¥DBDEMO フォルダーが作成されています。このフォルダー配下に JES 機能で使用するフォルダーをあらかじめ用意しておきます。

カタログファイルやスプールファイルを配置するため DATAFILE フォルダーを C:¥VS¥DBDEMO 配下へ作成します。

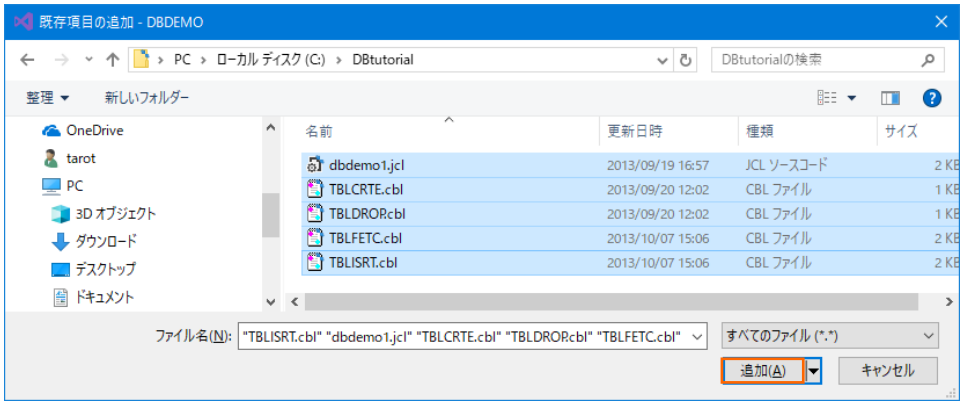


4) 既存ファイルのインポート時、自動的にコンパイル指令が指定される機能が用意されていますが、本チュートリアルではこれを解除します。 [ツール] プロダウンメニューの [オプション] を選択してオプションウィンドウを表示します。

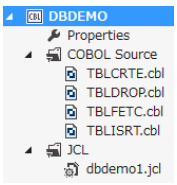
左側ツリービューの [Micro Focus] > [指令] > [COBOL] > [ファイルの指令の確定] チェックボックスをオフにして [OK] ボタンをクリックします。



5) 用意した例題プログラム類をインポートします。 DBDEMO プロジェクトを右クリックして [追加] > [既存の項目] を選択し、既存項目の追加ウィンドウにて C:¥DBtutorial を指定すると内容が表示されますので、全ファイルを選択後 [追加] ボタンをクリックします。この実行により、プロジェクトフォルダーへ例題プログラムが配置されます。



6) [ソリューション エクスプローラー] 内に表示されている DBDEMO プロジェクトにインポートしたファイルが表示されていることを確認します。

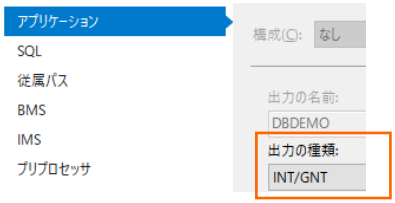


3.4 プロジェクトプロパティの設定

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

1) [ソリューション エクスプローラー] 内の [Properties] をダブルクリックしてプロパティウィンドウを表示します。

2) 左側ツリービューの [アプリケーション] を選択して、生成する実行ファイルを GNT にするため [出力の種類] へ [INT/GNT] を選択します。



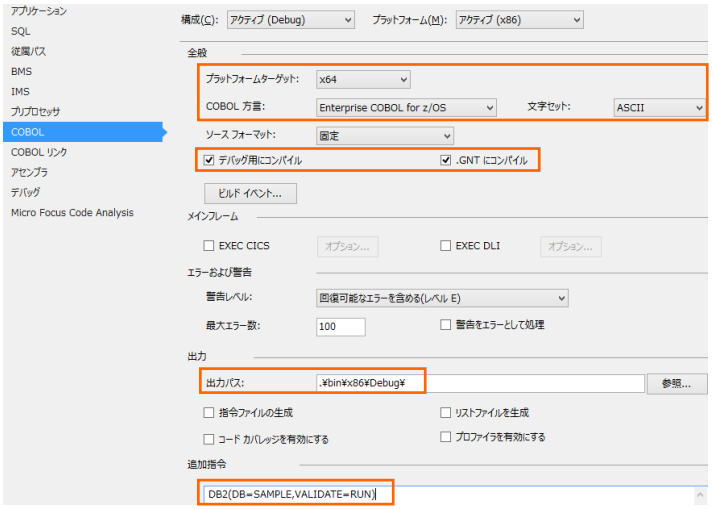
情報

GNT は Micro Focus 独自のオブジェクトで、Micro Focus COBOL ランタイム環境下で実行可能となります。

3) 左側ツリービューの [COBOL] を選択して、下記項目を入力します。

項目名	説明
プラットフォーム ターゲット	稼働ビット数を指定します。ここでは [x64] を指定します。
COBOL 方言	COBOL 言語方言を指定します。 例題プログラムは IBM Enterprise COBOL の方言を使用しているため、ここでは [Enterprise COBOL for z/OS] を指定します。
文字セット	EBCDIC または ASCII を指定します。ここでは [ASCII] を選択します。
デバッグ用にコンパイル	デバッグ実行時に使用するファイルを生成するように指定します。
.GNT にコンパイル	実行ファイル形式を GNT に指定するためにチェックをオンにします。
出力パス	実行ファイルが出力されるパスを指します。任意に指定可能です。

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



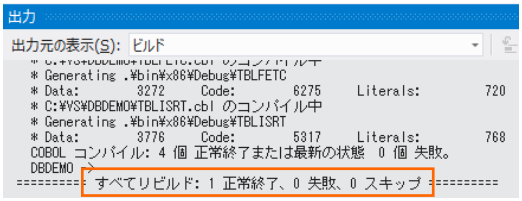
4) 左側ツリービューの [デバッグ] を選択して、[アクティブ設定を選択] へ [JCL] を選択します。

5) プロパティファイルを上書き保存します。

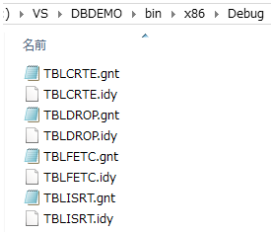
3.5 ビルドの実行

1) [ソリューション エクスプローラー] の DBDEMO ソリューションを右クリックして [ソリューションのビルド] を選択すると、コンパイル指定に沿ったビルドが実行されます。

2) [出力] ウィンドウで成功を確認します。



3) 前項で確認した出力パスへ実行ファイルに指定した gnt ファイルが作成されていることを確認します。



3.6 XA スイッチモジュールの生成

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

1) プリコンパイルを行うため、製品フォルダーに含まれている下記フォルダーを書き込み権限があるフォルダー配下へコピーします。本チュートリアルでは C:¥ 直下へコピーします。

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

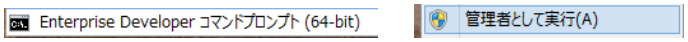
【理由 2】製品関連フォルダーの書き込み権限によるトラブルを避ける

【コピー元フォルダー例】

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

【コピー先フォルダーの例】C:¥xa

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



3) コマンドプロンプトで、コピーした C:¥xa パスへ移動します。

```
C:¥Users¥tarot¥Documents>cd c:¥xa

c:¥xa>
```

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

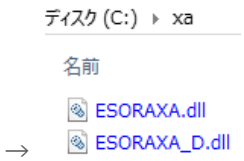
① Oracle

コマンド）build ora11（対象バージョンにより ora12）

```
c:¥xa>build ora11
Building 64-bit switch module...
Micro Focus COBOL
Version 4.0 Copyright (C) Micro Focus 1984-2018. All rights reserved.

* Cobsql Integrated Preprocessor
* CSQL-I-018: Oracle プリコンパイラトランスレータを起動します。
* CSQL-I-020: Oracle プリコンパイラの出力を処理中。
* CSQL-I-001: COBSQL：チェッカへの引き渡しを完了しました。
* チェック終了：エラーはありません- コード生成を開始します
* Generating ESORAXA
* Data:       15952     Code:       62590     Literals:        2976
Micro Focus COBOL - CBLLINK utility
Version 4.0.0.79 Copyright (C) Micro Focus 1984-2018. All rights reserved.
```

→



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

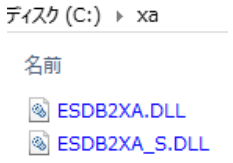
② DB2

コマンド）build db2

```
c:¥xa>build db2
Building 64-bit switch module...
Micro Focus COBOL
Version 4.0 Copyright (C) Micro Focus 1984-2018. 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:       21408     Code:       46154     Literals:        2768
Micro Focus COBOL - CBLLINK utility
Version 4.0.0.79 Copyright (C) Micro Focus 1984-2018. All rights reserved.

Microsoft (R) Incremental Linker Version 14.12.25830.2
Copyright (C) Microsoft Corporation.  All rights reserved.
```

→



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

③ SQL Server

A) ビルドの実行

コマンド）build mssql

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

B) リンクエラー時

環境変数「LIB」へ下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してください。

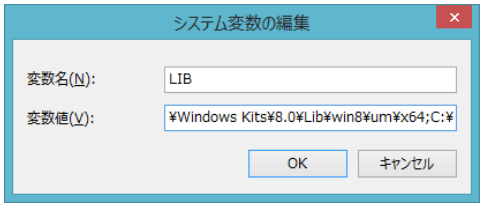
xaSwitch.Lib を利用するためには Windows SDK が必要になります。

32 ビット例：C:¥Program Files (x86)¥Windows Kits¥8.0¥Lib¥win8¥um¥x**86**

64 ビット例：C:¥Program Files (x86)¥Windows Kits¥8.0¥Lib¥win8¥um¥x**64**

```
ESMSSQL.obj
cbllds00015630.obj
LINK : fatal error LNK1181: cannot open input file 'xaSwitch.Lib'
```

↓

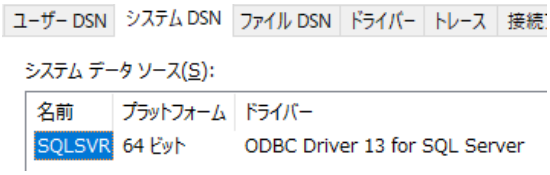


↓ コマンドプロンプト再起動後に再ビルド

```
ESMSSQL.obj
cbllds00015740.obj
   Creating library ESMSSQL.lib and object ESMSSQL.exp
Microsoft (R) Manifest Tool version 6.2.9200.20789
Copyright (c) Microsoft Corporation 2012.
All rights reserved.
        1 個のファイルをコピーしました。
Micro Focus COBOL
Version 2.3.00351 Copyright (C) Micro Focus 1984-2015. 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 ESMSSQL_D
* Data: 16 Code: 55176 Literals: 3536
Micro Focus COBOL - CBLLINK utility
Version 2.3.0.95 Copyright (C) Micro Focus 1984-2015. All rights reserved.
```

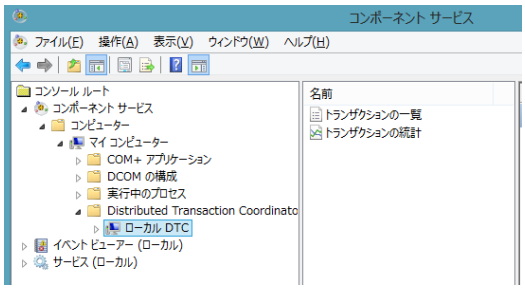
C) ODBC の追加

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

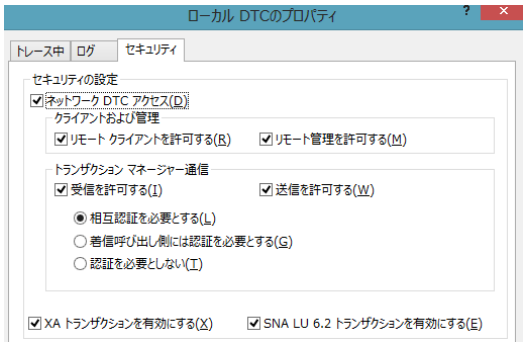


D) XA トランザクションの有効化

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



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

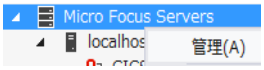


XA スイッチモジュールのビルド詳細に関しては製品ヘルプをご参照ください。

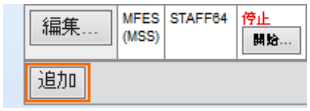
3.7 Enterprise Server インスタンスの設定

Enterprise Server インスタンスには JES をエミュレーションする機能が搭載されており、この開発用インスタンスを使用してメインフレームアプリケーションのテスト実行やデバッグを行います。本番環境には実行製品である Enterprise Server をインストールし、本番用インスタンス上でアプリケーションを稼働させます。

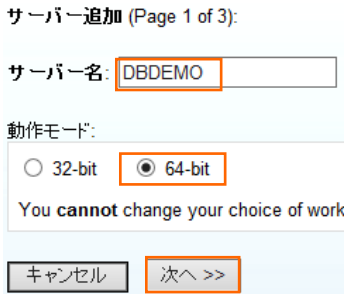
1) Enterprise Server インスタンスを作成します。[サーバー エクスプローラー] タブの [Micro Focus Server] を右クリックして [管理] を選択します。



2) Enterprise Server Administration 画面に Enterprise Server インスタンス一覧が表示されますので、画面の左下にある [追加] ボタンをクリックします。



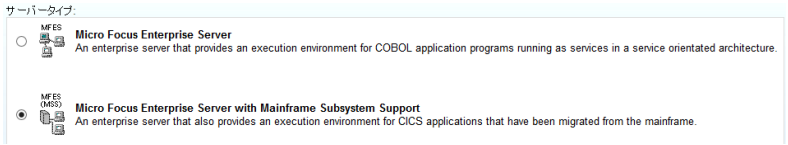
3) サーバー名には [DBDEMO] を入力、動作モードは 64-bit を指定して [次へ] ボタンをクリックします。



重要

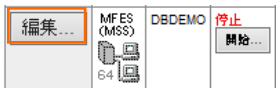
実行ファイル生成に指定した稼働ビット数 = Enterprise Server インスタンス稼働ビット数 = XA リソースビット数 = データベースクライアント対応ビット数 である必要があります。

4) 画面の Page 2/3 では、CICS や JCL を実行可能な機能を持つ [Micro Focus Enterprise Server with Mainframe Subsystem Support] が選択されていることを確認後、[次へ] ボタンをクリックします。



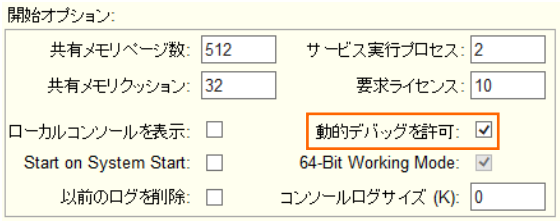
5) Page 3/3 では [TN3270 リスナーの作成] のチェックをオフにして [追加] ボタンをクリックすると、[DBDEMO] という名前の 64 ビットアプリケーション稼働用 Enterprise Server インスタンスが追加されます。

6) 左にある [編集] ボタンをクリックします。



7) [サーバー] > [プロパティ] > [一般] タブ内の下記項目を設定します。

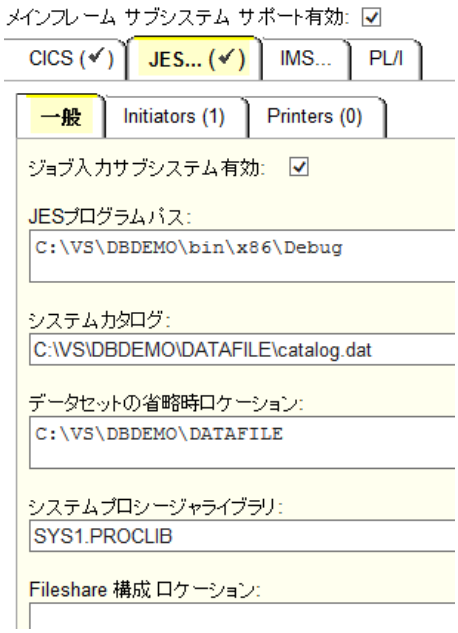
① [動的デバッグを許可] チェックボックスをオンにします。この指定により、 Visual Studio からの動的デバッグが可能になります。



② [適用] ボタンをクリックします。

8) [サーバー] > [プロパティ] > [MSS] > [JES] タブで表示される画面の各項目を設定します。入力後は [Apply] ボタンをクリックします。

項目名	説明
メインフレーム サブシステム サポート有効	[MSS] タブ配下の設定をオン、オフ指定します。ここではオンに指定します。
ジョブ入力サブシステム 有効	[JES] タブ配下の設定をオン、オフ指定します。ここではオンに指定します。
JES プログラム パス	COBOL アプリケーション実行ファイルが存在するパスを指定します。
システムカタログ	カタログファイルが存在するパスと、そのファイル名称を指定します。
データセットの省略時ロケーション	ジョブ実行時に生成されるスプールデータやカタログされるデータセットのデフォルトパスを指定します。
システムプロシージャライブラリ	プロシージャライブラリの名前を指定します。 ここでは SYS1.PROCLIB を入力します。

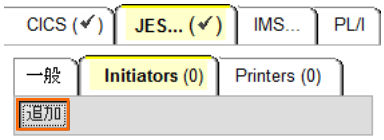


重要

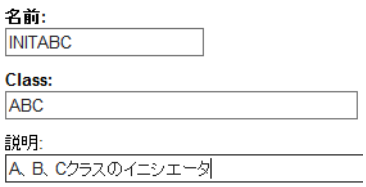
入力値は全て半角英数字で指定してください。

これらのフィールドでは改行を入れないように注意してください。

9) [サーバー] > [プロパティ] > [MSS] > [JES] > [Initiators] タブを表示し、左下の [追加] ボタンをクリックします。



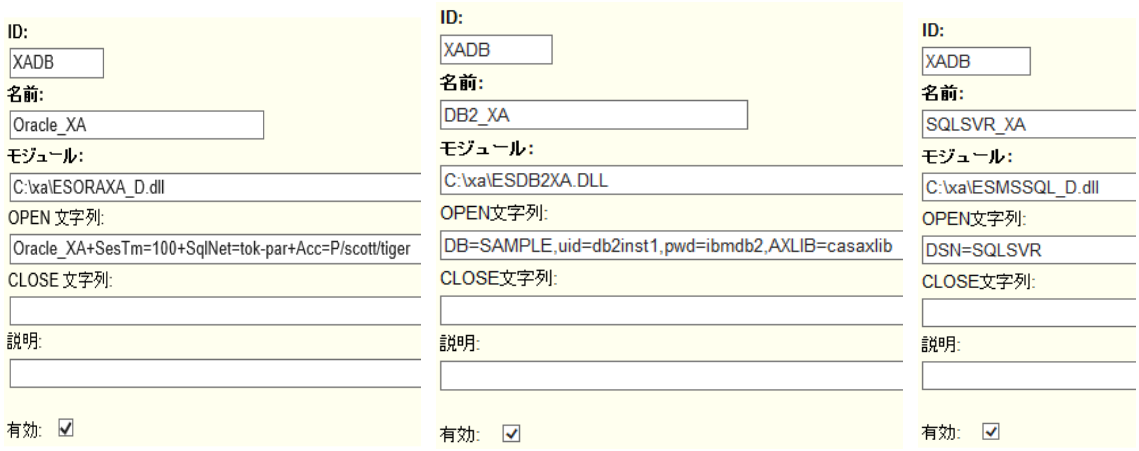
10) 下記画面のように入力して [追加] ボタンをクリックします。この指定により [DBDEMO] インスタンスが開始時にイニシエータが稼働し、ジョブクラス A,B,C のジョブが実行可能になります。



11) 前項で作成した XA リソースを定義します。[サーバー] > [プロパティ] > [XA リソース] タブを表示して、左下の[追加] ボタンをクリックします。

12) 前項で作成した、使用するデータベース製品の XA リソースを設定します。下記項目を入力後 [追加] ボタンをクリックします。

項目名	説明
ID	プログラムや JCL の IKJEFT ユーティリティに渡す DSN TSO コマンドの SYSTEM パラメタへ指定する ID を指定します。 ここでは XADB を指定します。 **ID:** XADB
名前	XA リソース名として任意の名前を指定します。 Oracle は Oracle_XA 固定です。 **名前:** Oracle_XA
モジュール	前項で作成した XA スイッチモジュールのパスとファイル名を指定します。 【Oracle 使用時の例】 動的登録の C:¥xa¥ESORAXA_D.dll を入力します。 **モジュール:** C:\xa\ESORAXA_D.dll 【DB2 使用時の例】 動的登録の C:¥xa¥ESDB2XA.dll を入力します。 **モジュール:** C:\xa\ESDB2XA.DLL 【SQL Server 使用時の例】 動的登録の C:¥xa¥ESMSSQL_D.dll を入力します。 **モジュール:** C:\xa\ESMSSQL_D.dll
OPEN 文字列	対象データベースのオープン文字列を指定します。 【Oracle 使用時の例】 Oracle_XA+SesTm=100+SqlNet=tok-par+Acc=P/scott/tiger を入力します。 OPEN文字列: Oracle_XA+SesTm=100+SqlNet=tok-par+Acc=P/scott/tiger 【DB2 使用時の例】 DB=SAMPLE,uid=db2inst1,pwd=ibmdb2,AXLIB=casaxlib を入力します。 静的登録の場合は末尾に SREG=T を指定します。デフォルトは動的です。 OPEN文字列: DB=SAMPLE,uid=db2inst1,pwd=ibmdb2,AXLIB=casaxlib 【SQL Server 使用時の例】 DSN=SQLSVR を入力します。（=ODBC 名） OPEN文字列: DSN=SQLSVR
有効	有効、無効切り替えチェックを指定します。ここではオンを指定します。

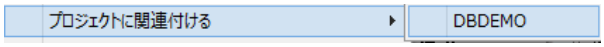


13) 画面左上の［Home］をクリックして一覧画面に戻ります。

3.8 Enterprise Server インスタンスの開始と確認

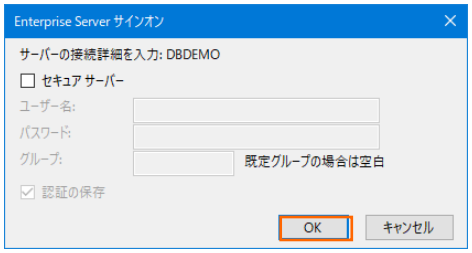
1) ［サーバー エクスプローラー］内に DBDEMO インスタンスが表示されていることを確認します。表示されていない場合は［Micro Focus Server］を右クリックし、［最新の情報に更新］を選択してリフレッシュしてください。

2) ［サーバー エクスプローラー］内の DBDEMO インスタンスを右クリックし、［プロジェクトに関連付ける］>［DBDEMO］を選択します。これにより DBDEMO プロジェクトから実行されるアプリケーションは DBDEMO インスタンスで処理されることになります。

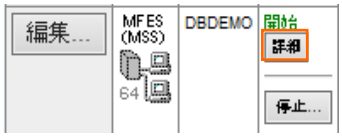


3) ［DBDEMO］インスタンスを右クリックして［開始］を選択します。

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

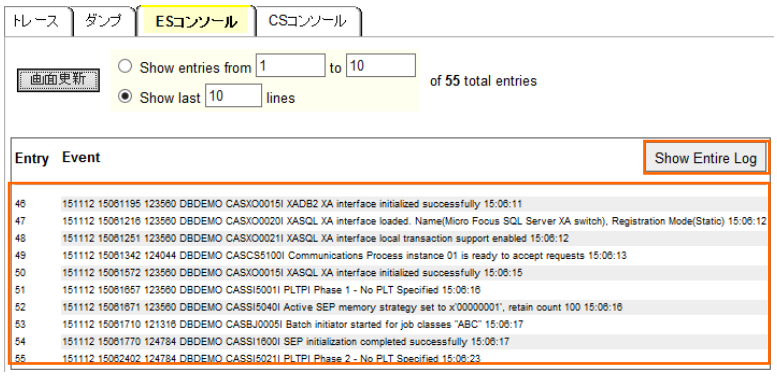


5) Enterprise Server Administration 画面へ移動して開始状態であることを確認後、[詳細] ボタンをクリックします。



6) [サーバー] > [診断] > [ES コンソール] で [DBDEMO] インスタンスのコンソールログをリアルタイムにチェックすることができます。また [Show Entire Log] をクリックしてログ全体を表示させることも可能です。

正常に開始されたことを確認します。



> **注意**
>
> いくつかのサービス開始や XA モジュールのロード、オープンが失敗してもインスタンスは開始されますので、ログ内容を必ず確認してください。

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

```
123560 DBDEMO   CASXO0020I XADB XA interface loaded. Name(Oracle_XA), Registration Mode(Dynamic) 15:06:07
123560 DBDEMO   CASXO0021I XADB XA interface local transaction support enabled 15:06:08
123560 DBDEMO   CASXO0015I XADB XA interface initialized successfully 15:06:09
```

8) 画面左上の [Home] をクリックして一覧画面に戻ります。

3.9 データベースアクセスを含む COBOL バッチプログラムの実行

現在 DBDEMO インスタンスが稼働していますので、例題プログラムを実行することができます。まずは簡単な JCL を実行してみます。

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

```
//DBDEMO1 JOB CLASS=A,MSGCLASS=A
//*****************************************
//STEP01 EXEC PGM=IKJEFT01
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN  DD *
  DSN SYSTEM(XADB)
      RUN PROGRAM(TBLCRTE) PLAN(SAMPLES)
  END
/*
//SYSTSPRT  DD SYSOUT=*
//*****************************************
//STEP02 EXEC PGM=IKJEFT01
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN  DD *
  DSN SYSTEM(XADB)
      RUN PROGRAM(TBLISRT) PLAN(SAMPLES)
  END
/*
//SYSIN DD *
00001Soseki Natsume      1-1,Koishikawa,Bunkyo-ku,Tokyo-to         1886
00002Ryotaro Shiba       2-3,Sonezaki,Kita-ku,Osaka-shi,Osaka-fu   1900
00003Hideyo Noguchi      5-1,Inawashiro,Aizu-shi,Fukushima-ken     1911
00004Osamu Dazai         2-6,Tsugaru,Tsugaru-gun,Aomori-ken        1911
00005Eiji Yoshikawa      9-3,Miyamotomura,Mimasaka-gun,Okayama-ken 1920
00006Jirocho Shimizu     6-6,Jiro-cho,Shimizu-shi,Shizuoka-ken     1800
00007Ogai Mori           3-1,Rintaro-cho,Tsuwano-shi,Shimane-ken   1886
00008Ryoma Sakamoto      1-1,Harimayabashi,Kochi-shi,Kochi-ken     1820
00009Shiki Masaoka       5-5,Dogo Onsen,Matsuyama-shi,Ehime-ken    1870
00010Yukichi Fukuzawa    8-8,Keio-cho,Nakatsu-shi,Oita-ken         1835
/*
```

この JCL は 4 ステップから構成されています。

① STEP01

データベースへテーブルを新規作成します。

② STEP02

SYSIN データを作成したテーブルへ挿入します。

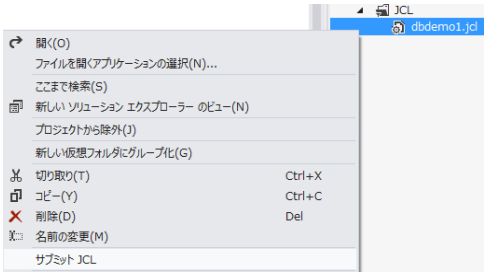
③ STEP03

挿入したデータをテーブルから全件読み込み、SYSOUT へ出力します。

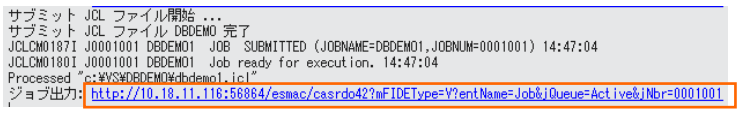
④ STEP04

作成したテーブルをデータベースから削除します。

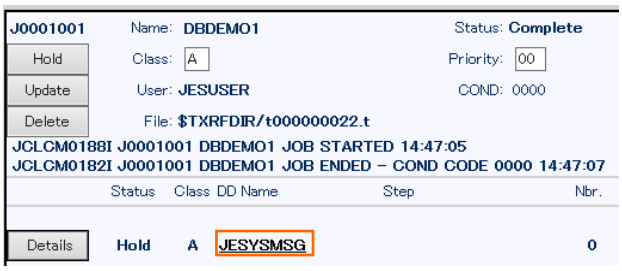
2) [ソリューション エクスプローラー] 内の dbdemo1.jcl を右クリックして [サブミット JCL] を選択し、この JCL を実行します。



3) [出力] タブの [出力元] に [Enterprise Server] を選択すると JOB 実行ログと JOB 番号が表示されますので、コントロールキーを押しながらリンクをクリックします。



4) この JOB 番号にかかわるスプール一覧が表示されます。先頭の [JESYSMEG] をクリックしてジョブログを確認します。



5) ジョブログの内容を確認すると、この JOB が正常に終了していることが確認できます。

6) [web ブラウザー 戻る] アイコンをクリックし、スプール一覧から各ステップの出力内容を確認することができます。

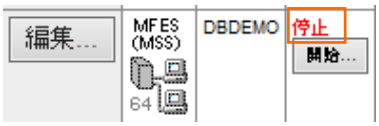
たとえば、STEP03 の SYSOUT を表示すると、作成したテーブルからインサート済データが正常に FETCH できていることが確認できます。

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

3.10 Enterprise Server インスタンスの停止

1) DBDEMO インスタンスを停止します。

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



WHAT'S NEXT

- メインフレーム COBOL 開発 ： JCL Visual Studio 2017 編
- メインフレーム COBOL 開発 ： CICS Visual Studio 2017 編
- 本チュートリアルで学習した技術の詳細については製品マニュアルをご参照ください。