
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 2019 編

1. 目的

本チュートリアルでは、Visual Studio 2019 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 2019 がインストールされていること
- 使用マシンに Micro Focus Enterprise Developer 7.0 for Visual Studio 2019 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 Database 19c Enterprise Edition Release 19.0.0.0.0
Version 19.3.0.0.0 64bit
クライアント : Oracle クライアント 19.0.0.0.0 Version 19.3.0.0.0 64bit
(Pro*COBOL プリコンパイラ使用)
 - Db2 サーバー : Db2 11.1.0.1527 64bit (Db2 PREP プリコンパイラ使用)
 - SQL Server サーバー : SQL Server 2016 Developer Edition 13.0340.1.0 64bit
クライアント : ODBC Driver 17 for SQL Server

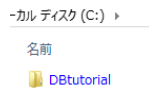
3. チュートリアル手順の概要

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. 文字エンコーディングの設定
8. Enterprise Server インスタンスの設定
9. Enterprise Server インスタンスの開始と確認
10. 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
11. Enterprise Server インスタンスの停止

3.1 チュートリアル準備

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 フォルダを C:¥ 直下に作成します。

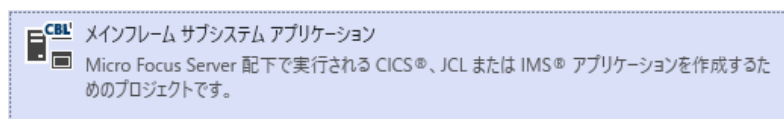
3.2 Visual Studio の起動

- 1) Visual Studio 2019 を起動します。



3.3 メインフレーム サブシステム アプリケーション 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、[メインフレーム サブシステム アプリケーション] を選択後、[次へ] ボタンをクリックします。

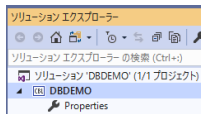


- 2) 下記項目を入力後、[作成] ボタンをクリックします。

項目名	説明
プロジェクト名	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。
ソリューションとプロジェクトを同じディレクトリに配置する	ここではチェックをオンにします。

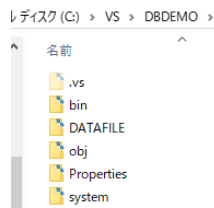


- 3) [ソリューション エクスプローラー] へ作成したプロジェクトが表示されます。



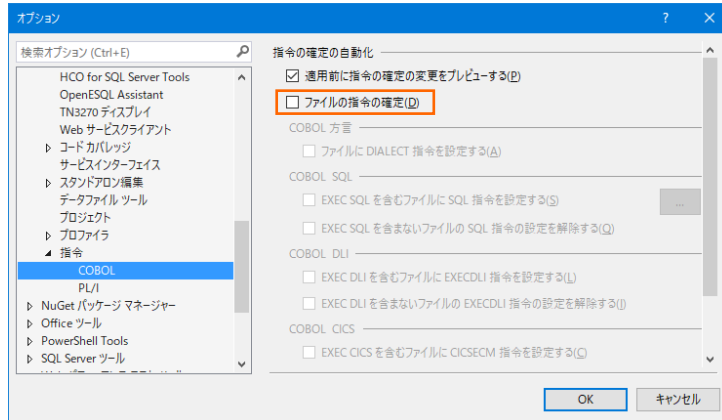
- 4) プロジェクトを作成したことにより C:\¥VS¥DBDEMO フォルダが作成されています。このフォルダ配下に JES 機能で使用するフォルダをあらかじめ用意しておきます。

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

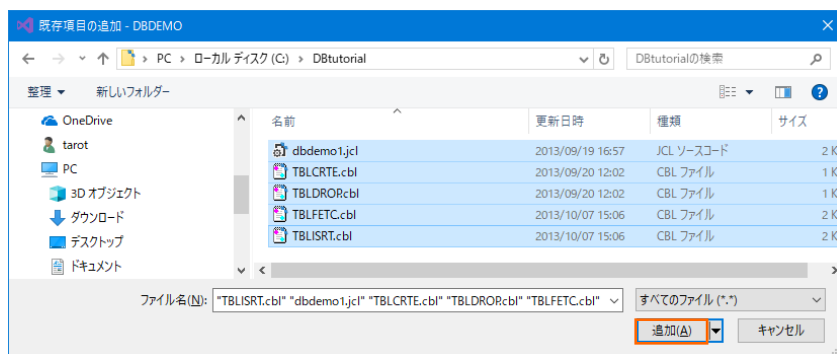


- 5) 既存ファイルのインポート時、自動的にコンパイル指令が指定される機能が用意されていますが、本チュートリアルではこれを解除します。[ツール] プログラムメニューの [オプション] を選択してオプションウィンドウを表示します。

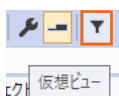
左側ツリービューの [Micro Focus Tools] > [指令] > [COBOL] > [ファイルの指令の確定] チェックボックスをオフにして [OK] ボタンをクリックします。



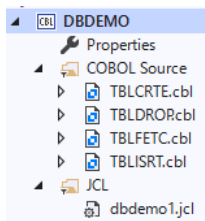
- 6) 用意した例題プログラム類をインポートします。DBDEMO プロジェクトを右クリックして [追加] > [既存の項目] を選択し、既存項目の追加ウィンドウにて C:\¥DBtutorial を指定すると内容が表示されますので、全ファイルを選択後 [追加] ボタンをクリックします。この実行により、プロジェクトフォルダへ例題プログラムが配置されます。



- 7) 種類別に表示するため、[ソリューション エクスプローラー] 内の [仮想ビュー] アイコンをクリックします。



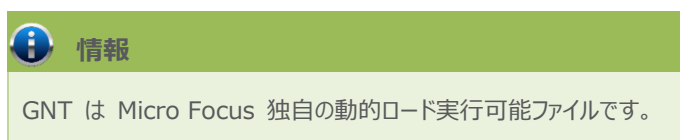
- 8) [ソリューション エクスプローラー] 内に表示されている DBDEMO プロジェクトにインポートしたファイルが表示されていることを確認します。



3.4 プロジェクトプロパティの設定

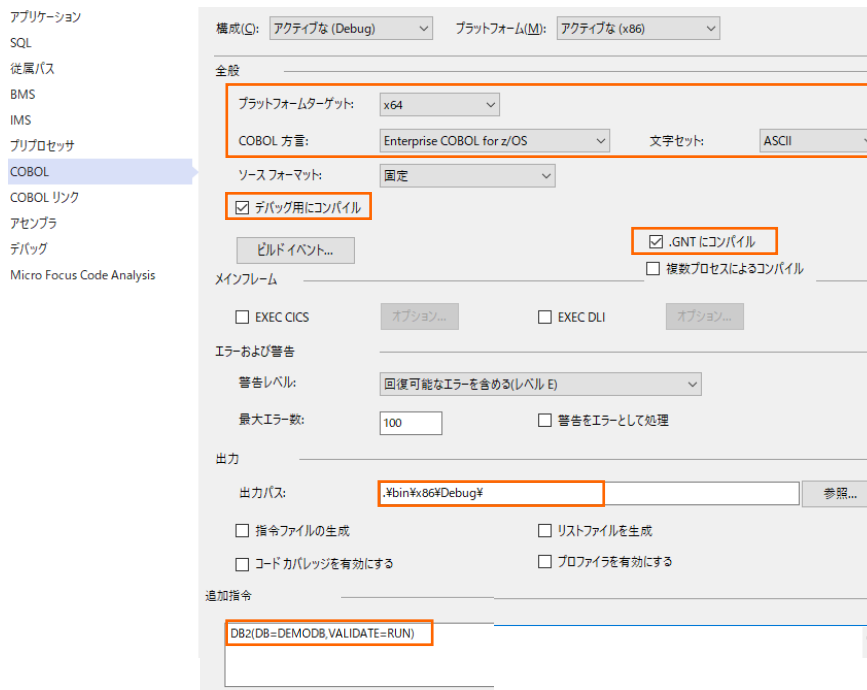
プログラム内容に沿ったプロジェクトのプロパティを設定します。埋め込み SQL 付き COBOL ソースは、予め Micro Focus 形式の COBOL ソースにプリコンパイルしてから使用することも可能ですが、製品にはプリプロセッサ機能からプリコンパイラを呼び出して内部的にプリコンパイルする機能があります。これを使用することにより、オリジナルソースイメージのままデバッグが可能となり管理も容易になります。ここでは後者の方法を紹介합니다。

- 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。



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。

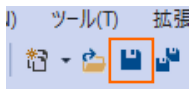
項目名	説明
プラットフォーム ターゲット	稼働ビット数を指定します。ここでは [x64] を指定します。
COBOL 方言	COBOL 言語方言を指定します。 例題プログラムは IBM Enterprise COBOL の方言を使用しているため、ここでは [Enterprise COBOL for z/OS] を指定します。
文字セット	EBCDIC または ASCII を指定します。ここでは [ASCII] を選択します。
デバッグ用にコンパイル	デバッグ実行時に使用するファイルを生成するように指定します。
.GNT にコンパイル	実行ファイル形式を GNT に指定するためにチェックをオンにします。
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=DEMODB,VALIDATE=RUN) 追加指令 DB2(DB=DEMODB,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、[プロファイル] には DBDEMO、[起動] には JCL を指定します。

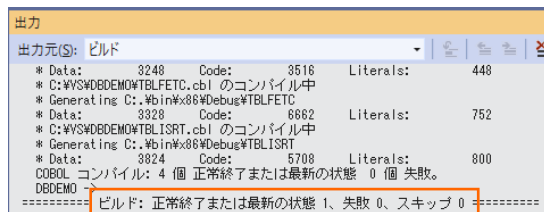


- 5) プロパティファイルを上書き保存します。

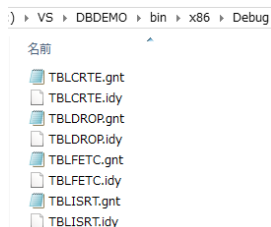


3.5 ビルドの実行

- 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) ビルドを行うため、製品フォルダに含まれている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。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】 製品関連フォルダの書き込み権限によるトラブルを避ける

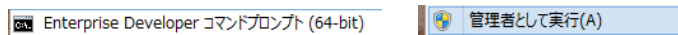
【コピー元フォルダ例】

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

【コピー先フォルダの例】 C:¥xa



- 2) Windows のプログラムメニューから [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) DLL を生成するために、Windows SDK が必要になります。リンクエラーを避けるために、これがインストールされているかご確認ください。また、複数の SDK や Microsoft Build Tools がインストールされている場合には、COBOL 環境が使用するバージョンを指定することもできます。

使用可能な SDK と Microsoft Build Tools の確認コマンド)

cblms -L

```
c:¥xa>cblms -L
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Windows SDK
Id バージョン 場所
0] 10.0.18362.0 c:¥Program Files (x86)¥Windows Kits¥10
1] 10.0.19041.0 c:¥Program Files (x86)¥Windows Kits¥10
2] 10.0.20348.0 c:¥Program Files (x86)¥Windows Kits¥10

Microsoft Build Tools
Id バージョン 場所
0] 14.29.30133 c:¥Program Files (x86)¥Microsoft Visual Studio¥2019¥Professional
```

最新バージョンに設定するコマンド)

cblms -U

```
c:\%xa>cblms -U
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Windows SDK
場所 = c:\Program Files (x86)\Windows Kits\10
バージョン = 10.0.20348.0

Microsoft Build Tools
場所 = c:\Program Files (x86)\Microsoft Visual Studio\2019\Professional
バージョン = 14.29.30133
```

特定のバージョンを指定するコマンド例)

cblms -US:1

```
c:\%xa>cblms -US:1
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Windows SDK
場所 = c:\Program Files (x86)\Windows Kits\10
バージョン = 10.0.19041.0

c:\%xa>cblms -UB:0
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Microsoft Build Tools
場所 = c:\Program Files (x86)\Microsoft Visual Studio\2019\Professional
バージョン = 14.29.30133
```

COBOL 環境が使用するバージョンを表示するコマンド)

cblms -Q

```
c:\%xa>cblms -Q
Micro Focus COBOL - Configuration Utility for the Microsoft Build Tools & SDK
7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

Windows SDK
場所 = c:\Program Files (x86)\Windows Kits\10
バージョン = 10.0.19041.0

Microsoft Build Tools
場所 = c:\Program Files (x86)\Microsoft Visual Studio\2019\Professional
バージョン = 14.29.30133
```

利用可能なオプションを表示するコマンド)

cblms -H

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

① Oracle



コマンド) build ora19 (対象バージョンにより ora18)

```
c:\%xa>build ora19
Building 64-bit switch module...
Micro Focus COBOL
Version 7.0 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.

* 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: チェック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.D
* Data:          34752   Code:          78794   Literals:          4000
Micro Focus COBOL - CBLLINK utility
Version 7.0.0.75 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.
```

ディスク (C:) > xa

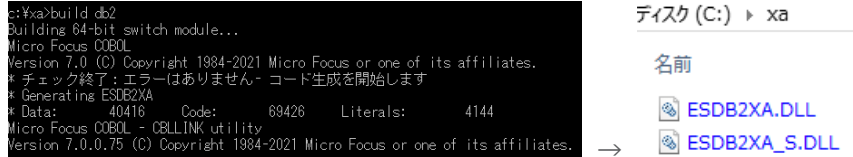
名前

-  ESORAXA.dll
-  ESORAXA_D.dll

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

② Db2

コマンド) build db2



```

c:\xa>build db2
Building 64-bit switch module...
Micro Focus COBOL
Version 7.0 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.
* 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: 40416 Code: 69426 Literals: 4144
Micro Focus COBOL - CBLLINK utility
Version 7.0.0.75 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.
  
```

ディスク (C:) > xa

名前

- ESDB2XA.DLL
- ESDB2XA_S.DLL

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

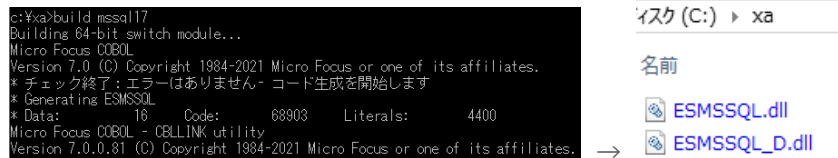
③ SQL Server

A) ビルドの実行

SQL Server に使用する ODBC 17.3 ドライバー以降、Microsoft 社は XA プロトコルを変更したため、ODBC 17.3 以降を使用する場合は、mssql17 ビルドオプションを使用します。また、ご利用の Windows に合わせた SDK をインストールしておいてください。

本チュートリアルでは ODBC 17.8 を利用するため mssql17 を使用してビルドします。

コマンド) build mssql17



```

c:\xa>build mssql17
Building 64-bit switch module...
Micro Focus COBOL
Version 7.0 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.
* 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
* Data: 16 Code: 68903 Literals: 4400
Micro Focus COBOL - CBLLINK utility
Version 7.0.0.81 (C) Copyright 1984-2021 Micro Focus or one of its affiliates.
  
```

ディスク (C:) > xa

名前

- ESMSSQL.dll
- ESMSSQL_D.dll

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

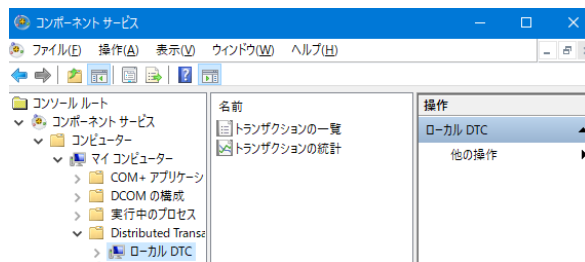
B) ODBC の追加

使用ビット数に合わせた ODBC データソースを Windows の [コントロールパネル] > [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 インスタンスへ登録する XA リソース定義の OPEN 文字列で使用する DSN 名となります。

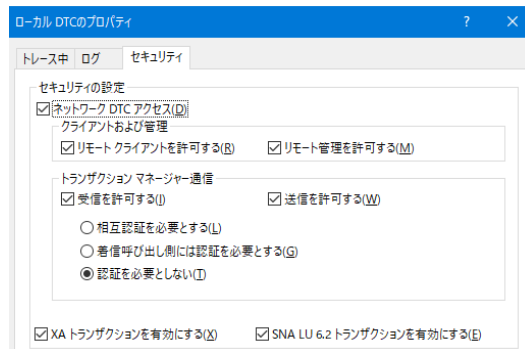
ユーザー DSN	システム DSN	ファイル DSN	ドライバー	トレース	接続プール	バージョン
システム データソース(S):						
名前	プラットフォーム	ドライバー				
SQLSVR	64 ビット	ODBC Driver 17 for SQL Server				

C) XA トランザクションの有効化

Windows の [コントロールパネル] > [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マニュアルをご参照ください。

3.7 文字エンコーディングの設定

Enterprise Server インスタンスを運用、管理する Enterprise Server Common Web Administration (以降 ESCWA) 機能では、スプールやデータ内容などに含まれる日本語を正しく表示させるために、事前に文字セットを所定のフォルダへ展開します。製品マニュアルの「リファレンス > コードセットの変換 > CCSID 変換テーブルのインストール > CCSID 変換テーブルをインストールするには」を参照しながら進めてください。

- 1) CCSID 変換テーブルをインストールします。

製品マニュアルにリンクされている下記の IBM CCSID 変換テーブルを、Web ブラウザから任意のフォルダへダウンロードします。アドレスは変更される可能性がありますので、製品マニュアルにてご確認ください。

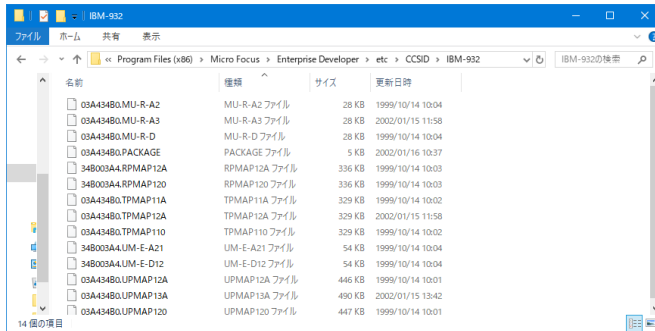
<http://www.microfocus.com/docs/links.asp?vc=cdctables>

- 2) 製品インストールフォルダ配下の etc フォルダに CCSID フォルダがない場合はこれを作成します。

例) C:\Program Files (x86)\Micro Focus\Enterprise Developer\etc\CCSID

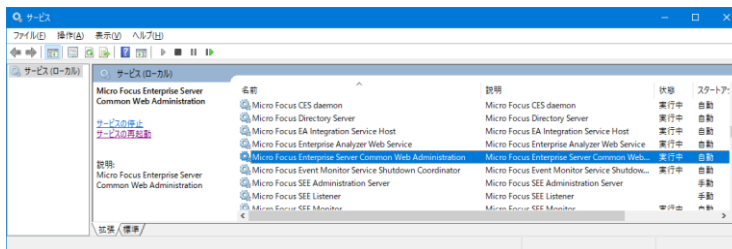
- 3) ダウンロードファイルに含まれている Package2.zip を展開します。
- 4) 展開した Package2 フォルダに含まれる IBM-932.zip を展開します。

- 5) 展開した IBM-932 フォルダを切り取り、作成した CCSID フォルダ配下へ貼り付け、14 ファイルが含まれていることを確認します。



詳細については、製品マニュアルの「デプロイ > 構成および管理 > Enterprise Server の構成および管理 > Enterprise Server Common Web Administration > [Native] > [Directory Servers] > リージョンとサーバー > リージョン > エンタープライズ サーバー リージョンの文字エンコーディングのサポート」をご参照ください。

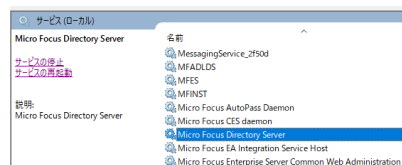
- 6) Windows サービスとして起動している Micro Focus Enterprise Server Common Web Administration を再起動し、インストールした CCSID をロードさせます。



3.8 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。

- 7) 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。Windows の Micro Focus Directory Server サービスが開始されているか確認し、停止している場合は開始してください。

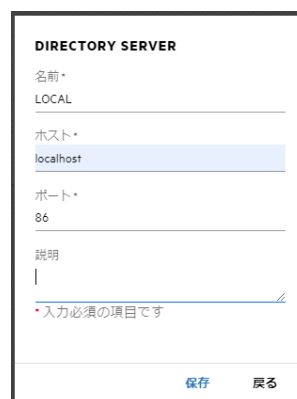


- 8) ブラウザが立ち上がり、Enterprise Server インスタンスを運用、管理する ESCWA が表示されます。ESCWA では Micro Focus Directory Server サービスのポートへ接続して登録された Enterprise Server インスタンスを管理するため、まず接続を作成します。既に存在している場合には、インスタンス作成に進んでください。

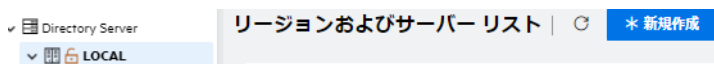
左側ペインで [Directory Server] をクリックし、右側ペインの [追加] ボタンをクリックします。



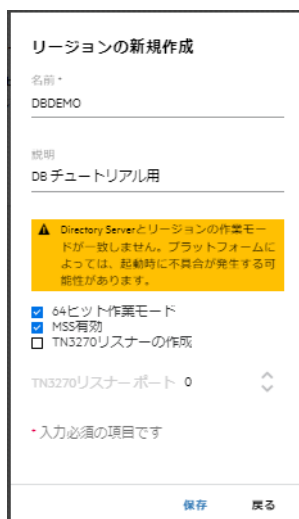
追加ウィンドウでは [名前] は任意ですが、ここでは LOCAL を、
[ホスト] は localhost またはマシンの IP アドレスを、
[ポート] は Micro Focus Directory Server の
デフォルトポートである 86 を入力して [保存] ボタンをクリックします。



- 9) 作成した [LOCAL] 接続をクリックし、右側ペインの [新規作成] ボタンをクリックします。



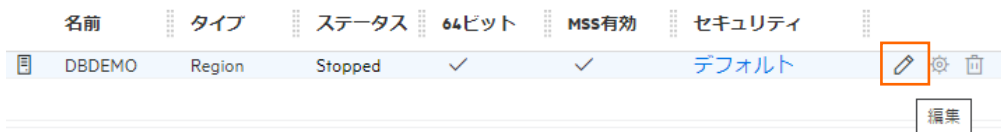
- 10) [リージョンの新規作成] 項目の [名前]、[説明] は任意ですが、ここでは名前に DBDEMO、説明に DB チュートリアル用と入力します。実行可能ファイルは 64 ビットを指定してコンパイルしたため、稼働させる Enterprise Server インスタンスも同様に [64 ビット作業モード] にチェックを入れます。これにより警告が表示されますが無視して先に進んでください。[MSS有効] にチェックが入っていることを確認し、[TN3270リスナーの作成] のチェックを外して [保存] ボタンをクリックします。



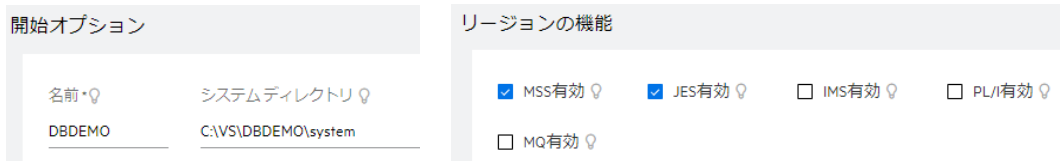
重要

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である必要があります。

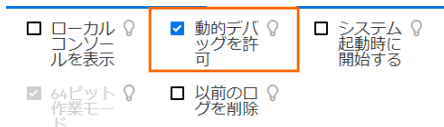
- 11) 64 ビットアプリケーション稼働用の DBDEMO インスタンスが作成され、一覧に表示されます。DBDEMO インスタンスにカーソルを合わせ、[編集] アイコンをクリックします。



- 12) DBDEMO インスタンスのログなどが出力される [システムディレクトリ] には前項で作成した system フォルダを指定して、[リージョンの機能] の [JES 有効] をチェックします。



- 13) 表示画面の下にある [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。



- 14) [追加設定] の [構成情報] 欄に、文字エンコーディングを指定する MFACCCGI_CHARSET 環境変数に IBM-932 を認識させるための値である Shift_JIS と、プロジェクトのパスを指定する環境変数を設定し、最後に [適用] ボタンをクリックします。

入力値)

[ES-Environment]

proj=C:¥VS¥DBDEMO

MFACCCGI_CHARSET=Shift_JIS


追加設定

構成情報


[ES-Environment]
proj=C:\VS\DBDEMO
MFACCCGI_CHARSET=Shift_JIS

- 15) 画面上部の [JES] プルダウンメニューから [構成] を選択し、表示される画面の各項目を設定します。構成情報に指定した proj 環境変数を使用して値を入力後、[適用] ボタンをクリックします。

項目名	説明
JES プログラム パス	COBOL アプリケーションの実行ファイルが存在するパスを指定します。
システムカタログ	カタログファイルを出力するパスと、そのファイル名称を指定します。
データセットの省略時刻ケーション	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。


JESの構成 |  **適用**

JES プログラムパス <input type="text"/>	システムカタログ <input type="text"/>
<code>\$proj\bin\x86\Debug</code>	<code>\$proj\DATAFILE\catalog.dat</code>
データセットの省略時ロケーション <input type="text"/>	システムプロシージャ ライブラリ <input type="text"/>
<code>\$proj\DATAFILE</code>	<code>SYS1.PROCLIB</code>

 **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。

16) [イニシエータ] の [新規作成] ボタンをクリックします。

イニシエータ |  *** 新規作成**

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

JESイニシエータ

名前

INITABC

クラス

ABC

説明

A,B,C クラスのイニシエータ

*入力必須の項目です


保存 **戻る**


18) セキュリティ観点から、Web リスナーのデフォルトステータスは [Disabled] になっています。安全を確認したうえで、[一般] プルダウンメニューから [リスナー] を選択し、表示された Web リスナーのステータスを [Stopped] へ変更後、[適用] ボタンをクリックします。

TLS設定

ステータス	ステータスの設定 <input type="text"/>	実際のアドレス
Disabled	Stopped	tcp:0.0.0.0

19) 画面左側ペインの [LOCAL] をクリックして一覧画面に戻ります。

▼  Directory Server

▼  **LOCAL**

重要

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

対象ファイルの特定には MFSYSCAT 環境変数を利用して、カタログファイルを指定します。

例)

```
set MFSYSCAT=C:¥VS¥DBDEMO¥DATAFILE¥catalog.dat
```

```
mfsplcnv -2
```

詳しくは製品マニュアルをご参照ください。

- 20) 前項で作成した XA リソースを登録します。画面上部の [一般] プルダウンメニューから [XA リソース] を選択し、表示される画面で [新規作成] ボタンをクリックします。

項目名	説明
ID	プログラムや JCL の IKJEFT ユーティリティに渡す DSN TSO コマンドの SYSTEM パラメタへ指定する ID を指定します。ここでは XADB を指定します。 ID <input type="text" value="XADB"/>
名前	XA リソース名として任意の名前を指定します。Oracle の場合は Oracle_XA 固定です。 名前 <input type="text" value="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 <input type="text" value="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 <input type="text" value="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 <input type="text" value="c:\xa\ESMSSQL_D.dll"/>
再接続試行	再接続の試行回数を指定します。デフォルトは 1 で 0 は指定できません。-1 は継続的に試行します。ここではデフォルトの 1 を使用します。

OPEN 文字列	<p>対象データベースのオープン文字列を指定します。</p> <p>【Oracle 使用時の例】 ORACLE_XA+SesTm=100+SqlNet=orcl-19c+Acc=P/scott/tiger を入力します。</p> <p>【Db2 使用時の例】 DB=DEMODB,uid=tarot,pwd=password,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。</p> <p>【SQL Server 使用時の例】 DSN=SQLSVR を入力します。(=ODBC 名)</p>
有効	有効、無効切り替えチェックを指定します。ここではオンを指定します。

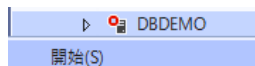
XA リソースの構成		XA リソースの構成		XA リソースの構成	
ID	名前	ID	名前	ID	名前
XADB	ORACLE_XA	XADB	DB2_XA	XADB	SQLSVR_XA
モジュール	clvaESORAXA.dll	モジュール	clvaESDB2XA.DLL	モジュール	clvaESM5SQL.Dll
<input checked="" type="checkbox"/> 有効	再接続試行 1	<input checked="" type="checkbox"/> 有効	再接続試行 1	<input checked="" type="checkbox"/> 有効	再接続試行 1
OPEN 文字列	ORACLE_XA+SesTm=100+SqlNet=orcl-19c+Acc=P/oracle/oracle	OPEN 文字列	DB=DEMODB,uid=tarot,pwd=password,AXLIB=casaxlib	OPEN 文字列	DSN=SQLSVR

3.9 Enterprise Server インスタンスの開始と確認

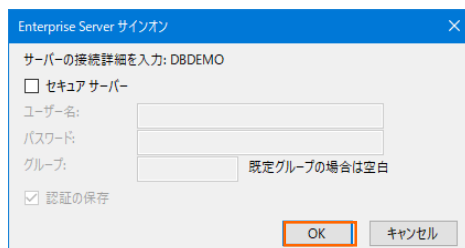
- 1) Visual Studio の [サーバー エクスプローラー] 内に DBDEMO インスタンスが表示されていることを確認します。表示されていない場合は [Micro Focus Server] を右クリックし、[最新の情報に更新] を選択してリフレッシュしてください。
- 2) [サーバー エクスプローラー] 内の DBDEMO インスタンスを右クリックし、[プロジェクトに関連付ける] > [DBDEMO] を選択します。これにより DBDEMO プロジェクトから実行されるアプリケーションは DBDEMO インスタンスで処理されることとなります。






- 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) ESCWA 画面へ移動して開始状態であることを確認後、[編集] アイコンをクリックします。

名前	タイプ	ステータス	64ビット	MSS有効	セキュリティ	
DBDEMO	Region	Started	✓	✓	デフォルト	  



- 6) 画面上部の [モニター] プルダウンメニューから [ログ] > [コンソールログ] を選択し、正常に開始されたことを確認します。
ログレベルが S や E の場合はエラー表示されます。

コンソール |    現行

プロセスID	メッセージID	ログレベル	メッセージ
	CASCD1038I	I	ES Communications Server created, ES
10812	CASKC1000I	I	ES concurrent request limit: 00000000
10812	CASSI1000I	I	Server Manager initialization complet
10812	JES000051I	I	Job Entry Subsystem (JES) services
6476	CASTS0007I	I	ES TSC Service Process initializat
10812	JES000059I	I	JES 5 digit job numbering support
3852	CASC55001I	I	Communications interface 01 initializati
	CASCD1060I	I	JES Initiator created for Server DBDEMO, process-id = 7744
10812	CASBJ0023I	I	Batch initiator INITABC: class(es) "ABC"
	CASCD0127I	I	SEP 00001 created for ES DBDEMO, process-id = 8584
6476	CASTS5115I	I	ES TSC Log facility file: C:\work\DBDEMO\system\LOGTSTDQ.dat
	CASCD0127I	I	SEP 00002 created for ES DBDEMO, process-id = 10868

【JES 機能の正常開始ログ抜粋】

JES000051I	I	Job Entry Subsystem (JES) services initialized
CASTS0007I	I	ES TSC Service Process initialization started
JES000059I	I	JES 5 digit job numbering support enabled
CASC55001I	I	Communications interface 01 initialization started
CASCD1060I	I	JES Initiator created for Server DBDEMO, process-id = 77
CASBJ0023I	I	Batch initiator INITABC: class(es) "ABC"

注意

いくつかのサービス開始が失敗してもインスタンスは開始されますので、ログ内容を必ず確認してください。

- 7) XA モジュールが正常にロードされ、オープンされると以下のようなログが出力されます。下記は Db2 の例ですが、動的登録を指示しているため Dynamic と出力されています。静的登録の場合は Static が出力されます。

CASX00020I	I	XADB XA interface loaded. Name(DB2 for WINDOWS), Registration Mode(Dynamic)
CASX00021I	I	XADB XA interface local transaction support enabled
CASX00015I	I	XADB XA interface initialized successfully

- 8) 画面左側ペインの [LOCAL] をクリックして一覧画面に戻ります。

3.10 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) Visual Studio の [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。

```

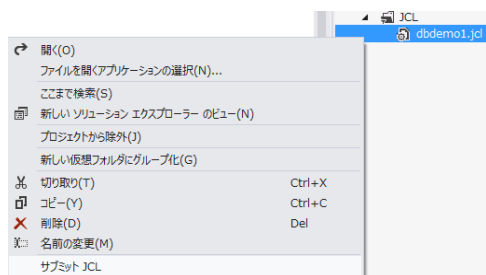
//DBDEMO1 JOB CLASS=A,MSGCLASS=A
//*****
//STEP01 EXEC PGM=JKEFT01
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DSN SYSTEM(VADB)
RUN PROGRAM(TBLCRTE) PLAN(SAMPLES)
END
/*
//SYSTSPRT DD SYSOUT=*
//*****
//STEP02 EXEC PGM=JKEFT01
//SYSOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DSN SYSTEM(VADB)
RUN PROGRAM(TBLISRT) PLAN(SAMPLES)
END
/*
//SYSIN DD *
0001Soseki Natsume 1-1,Koishikawa,Bunkyo-ku,Tokyo-to 1886
0002Ryota Shiba 2-3,Sonezaki,Kita-ku,Osaka-shi,Osaka-fu 1800
0003Hideo Noguchi 5-1,Inawashiro,Aizu-shi,Fukushima-ken 1811
0004Osamu Dazai 2-6,Tsugaru,Tsugaru-gun,Aomori-ken 1811
0005Eiji Yoshikawa 9-3,Miyamotoura,Miyasaka-gun,Okayama-ken 1820
0006Jirocho Shimizu 8-6,Jiro-cho,Shimizu-shi,Shizuoka-ken 1800
0007Osai Mori 3-1,Rintaro-cho,Tsuwano-shi,Shimane-ken 1886
0008Ryoma Sakamoto 1-1,Hariayabashi,Kochi-shi,Kochi-ken 1820
0009Shiki Masaoka 5-5,Dogo Onsen,Matsuyama-shi,Ehime-ken 1870
0010Yukichi Fukuzawa 8-8,Keio-cho,Nakatsu-shi,Oita-ken 1855
/*

```

この JCL は 4 ステップから構成されています。

- ① STEP01
データベーステーブルを新規作成します。
- ② STEP02
SYSIN データを作成したテーブルへ挿入します。
- ③ STEP03
挿入したデータをテーブルから全件読み込み、SYSOUT へ出力します。
- ④ STEP04
作成したテーブルをデータベースから削除します。

- 2) [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) [DBDEMO スプール] タブが表示されますので、[メッセージ] で COND CODE を確認します。

DBDEMO スプール dbdemo1.jcl DBDEMO*

名前	ユーザー	状態	COND	クラス	優先度
DBDEMO1	JESUSER	Complete	0000	A	0

ファイル STXRFRDIR(0000000011.t

メッセージ

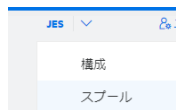
```

JCLCM0188I J0001000 DBDEMO1 JOB STARTED 16:49:00
JCLCM0182I J0001000 DBDEMO1 JOB ENDED - COND CODE 0000 16:49:07

```

状態	クラス	DD名	ステップ	ステップ番号	PROCステッ	レコード数
Hold	A	JESVSMMSG		0		96
Ready	A	SYSOUT	STEP01	1		1
Ready	A	SYSTSPRT	STEP01	1		5
Ready	A	SYSOUT	STEP02	2		11
Ready	A	SYSTSPRT	STEP02	2		5
Ready	A	SYSOUT	STEP03	3		11
Ready	A	SYSTSPRT	STEP03	3		5
Ready	A	SYSOUT	STEP04	4		1
Ready	A	SYSTSPRT	STEP04	4		5

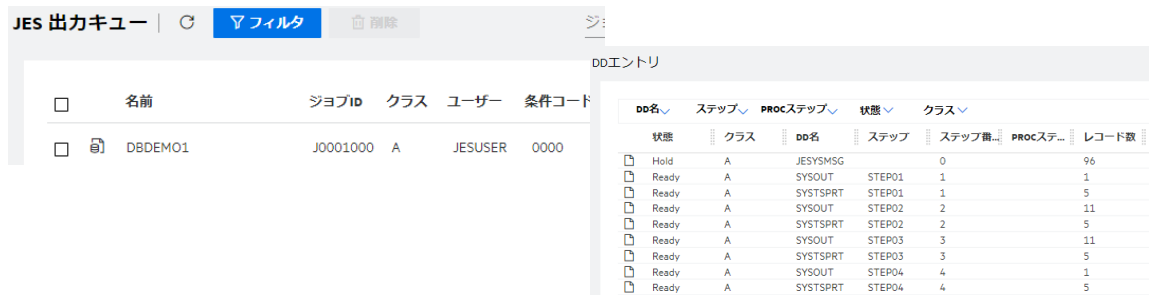
- 4) ESCWA で 詳細を確認してみます。DBDEMO インスタンスを選択後、[JES] プロダクションメニューから [スプール] を選択します。



- 5) スプルー一覧が表示されます。フィルタ機能で [完了] が指定されていることを確認します。



- 6) 実行した JOB 番号のスプルーをダブルクリックして内容を表示します。



- 7) 先頭の [JESYSMEG] をダブルクリックしてジョブログを確認すると、正常に終了していることが確認できます。

```
---> 16:49:07 JCLCM0182I JOB ENDED - COND CODE 0000
```

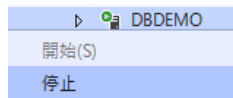
- 8) 右上にある [戻る] ボタンをクリックしてスプルー一覧に戻り、たとえば、STEP03 の SYSOUT を表示すると、作成したテーブルからインサート済データが正常に FETCH できていることが確認できます。



9) 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。

3.11 Enterprise Server インスタンスの停止

1) Visual Studio の [サーバー エクスプローラー] から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。

4. 免責事項

本チュートリアルの例題ソースコードは機能説明を目的としたサンプルであり、無謬性を保証するものではありません。例題ソースコードは弊社に断りなくご利用いただけますが、本チュートリアルに関わる全てを対象として、二次的著作物に引用する場合は著作権法に基づき適切な扱いを行ってください。

WHAT'S NEXT

- メインフレーム COBOL 開発 : JCL Visual Studio 2019 編
- メインフレーム COBOL 開発 : CICS Visual Studio 2019 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。