Rocket software

Visual COBOL チュートリアル

RESTful Web サービスによる COBOL 資産の再利用

Visual Studio 編

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 REST API を利用し Web サービスとして呼び出す機能を提供しています。RESTful の Web サービスとして呼び出しを行う場合、JSON 形式でやり取りが可能であれば呼び出し側のプログラムに依存することなく連携できるようになります。

このドキュメントでは COBOL のソースコードに一切手を加えることなくビジネスロジックとして Enterprise Server にディプロイし、それを Visual COBOL のクライアント生成機能を使って動作確認用のクライアントを作成し連携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。

- 開発クライアント ソフトウェア
  OS： Windows 11
  COBOL 開発環境製品： Visual COBOL 10.0 for Visual Studio 2022
  IDE： Microsoft Visual Studio Professional 2022 (17.7.4)?

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

3 チュートリアル手順

3.1 Windows クライアントでの開発準備作業

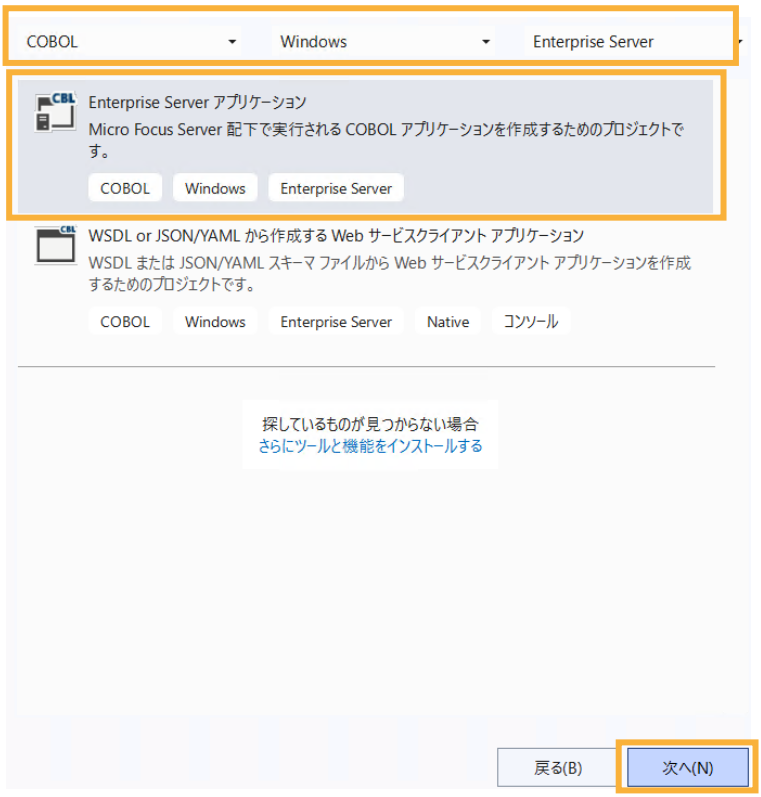
1) Visual COBOL for Visual Studio を起動

[スタート] メニュー > [すべてのアプリ] > [Visual Studio 2022] を選択します。

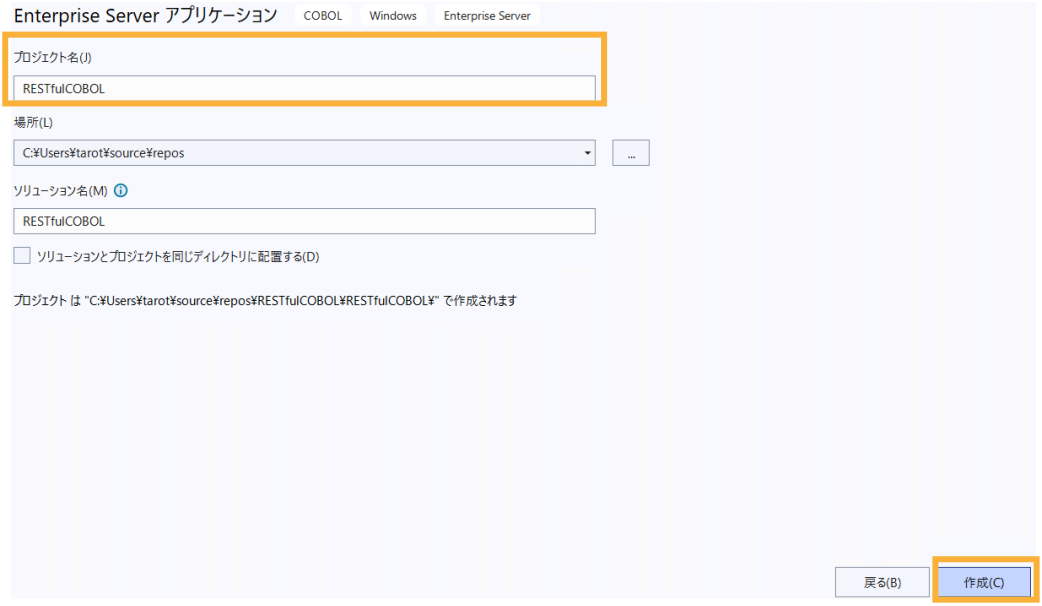
2) [Enterprise Server アプリケーション] プロジェクトの作成

① 「新しいプロジェクトの作成」を選択します。

② 「新しいプロジェクトの作成」 ウィザードが表示されるので ［言語］ を 「COBOL」、[プラットフォーム] を「Windows」、[プロジェクト タイプ] に「Enterprise Server」を選択し、[次へ(N)] をクリックします。



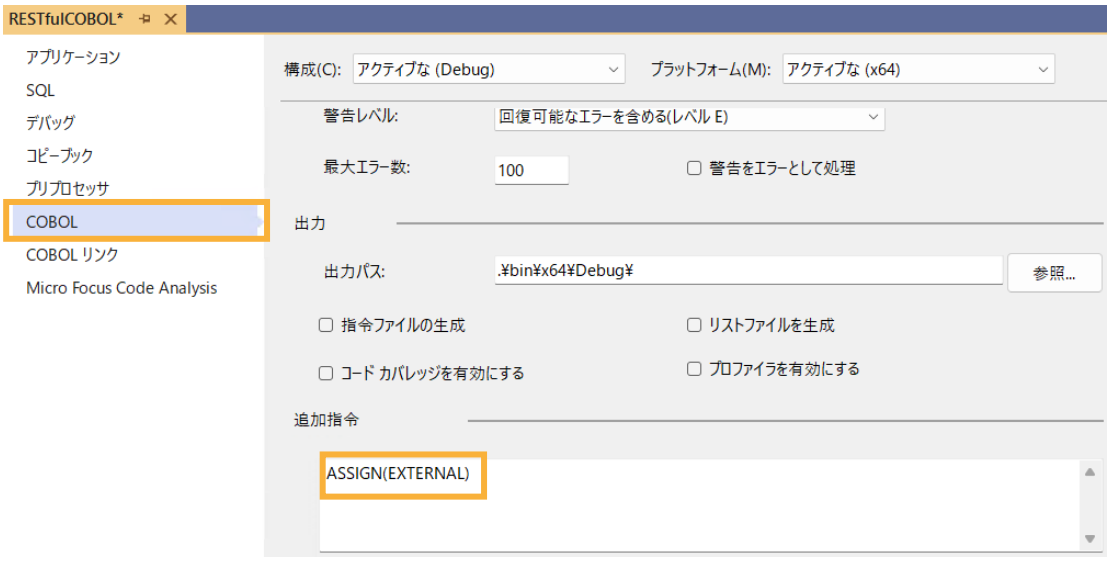
③ [名前(N)] フィールドに “RESTfulCOBOL” を入力して、[OK] をクリックします。[場所(L)] は任意のフォルダーを指定し、[ソリューション名] は、デフォルトのままとし、[作成(C)] をクリックします。



3) コンパイラオプションの指定とソースコードのインポート

① 作成されたプロジェクトの Properties をダブルクリックします。

② [COBOL] をクリックし、画面を下にスクロールして、[追加指令] に “ASSIGN(EXTERNAL)” を指定し、保存したうえで画面を閉じます。



③ エクスプローラーを起動し、サンプルのソースコードを解凍したフォルダーから ”BOOK-INFO.cpy” と ”BOOK.cbl” をプロジェクトフォルダーにドラッグアンドドロップします。

④ ソリューションエクスプローラーから、２つのファイルが正常にロードされていることを確認します。

3.2 Enterprise Server の設定変更

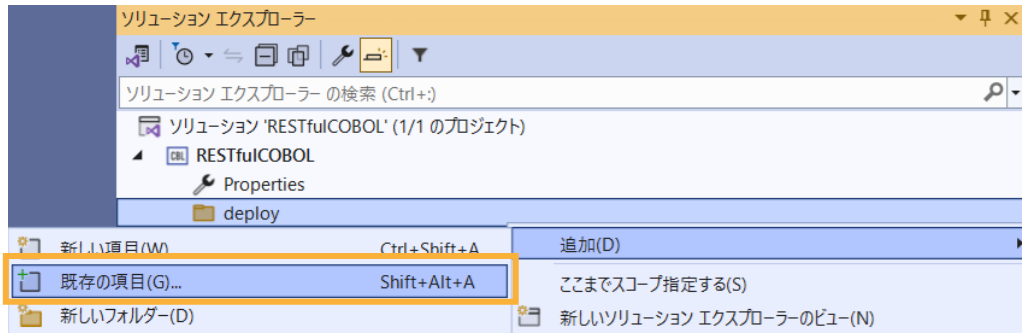
1) ディプロイ用フォルダーを作成します。

① 「RESTfulCOBOL」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 [追加(D)]→[新しいフォルダー(D)] を選択します。

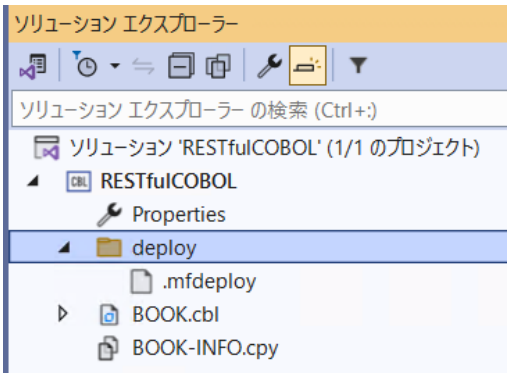
② フォルダー名に ”deploy” を指定します。

2) 「.mfdeploy」ファイルをインポートします。

① 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 [追加(D)]→[既存の項目(G)] を選択します。

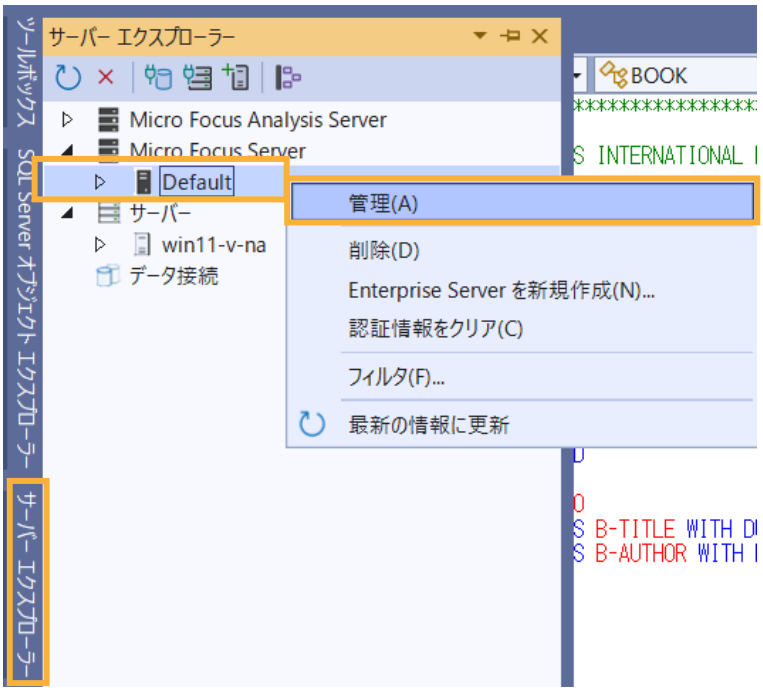


② [すべてのファイル(*.*)] に変更した上で、Visual COBOL インストールフォルダー¥deploy 配下にある [.mfdeploy] ファイルを指定します。



3) ESCWA 管理画面の設定

① サーバーエクスプローラーを表示し、[Micro Focus Server] > [Default] 上で右クリックし、コンテクストメニューから [管理(A)] を選択します。



認証画面が表示された場合は認証情報を入力します。

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

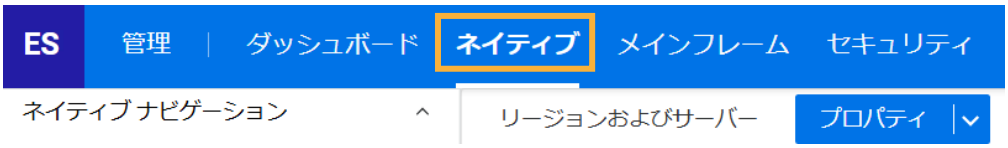
② ログイン画面が表示されますので、上記と同じ認証情報を入力して、[ログイン] をクリックします。

3.2.1 不要ログの抑止

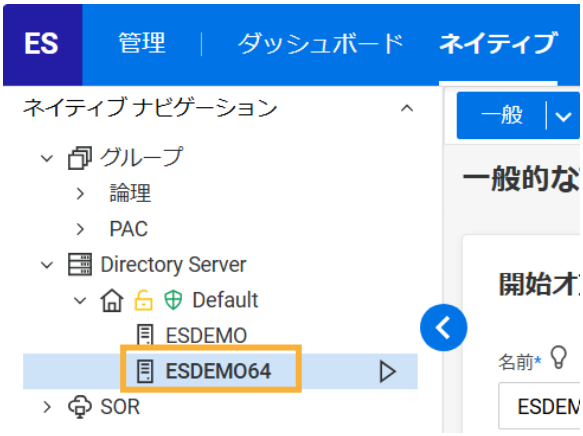
1) [プロパティ] タブをクリックし、画面をスクロールしたのち、モニターセクションの [有効] のチェックを外したうえで、画面上部にある [適用] をクリックします。

3.2.2 リスナー構成の変更

1) [ネイティブ] タブをクリックします。



2) 次に左側メニューの [Directory Server] をドリルダウンして、[Default] → [ESDEMO64]をクリックします。



3) [一般]メニューが表示されるので横にある下向き記号をクリックし、[リスナー] を選択します。

4) [通信プロセス 1] > [Web] をクリックします。

リスナープロパティが表示されます。横のスライドバーを下に下げていくとのカスタム構成情報が表示されています。デフォルトは「uploads=<ES>/deploy」となっています。この指定により、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しか書き込みできないため、さきほど作成した deploy フォルダーに変更します。

例：uploads=C:¥Users¥tarot¥source¥repos¥RESTfulCOBOL¥RESTfulCOBOL¥deploy

カスタム構成

```
[virtual paths]
cgi=<ES>/bin
uploads=C:\Users\tarot\source\repos\RESTfulCOBOL\RESTfulCOBOL\deploy
<default>=/dev/null
```

上記は、プロジェクトフォルダーが C:¥Users¥tarot¥source¥repos¥RESTfulCOBOL¥RESTfulCOBOL の場合です。

入力が終わったらスライドバーを上に移動して、[適用] をクリックします。

5) [Web Services and J2EE]のポート番号の変更を行います。

[通信プロセス 1] > [Web Services and J2EE] をクリックし、ポート番号を「*」から「9003」に変更します。

最後に、スライドバーを上に移動して、[適用] をクリックします。

3.2.3 Enterprise Server の環境設定

1) [一般]をクリックし、[追加設定]フィールドに下記の命令を追加したうえで、[適用] をクリックします。

※ サンプルファイルを c:¥vs-tutorial に解凍した場合です。環境に合わせてパスを修正してください。

```
[ES-Environment]
BOOKINFO=C:¥vc-tutorial¥DAT¥BOOKINFO.DAT
```

追加設定

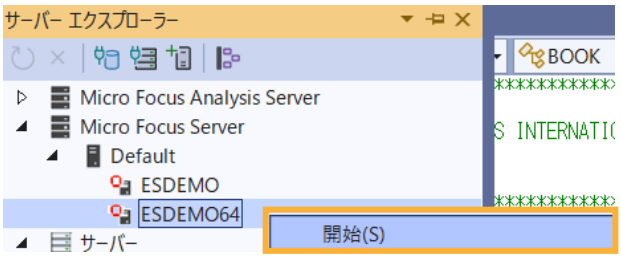
構成情報

```
[ES-Environment]
BOOKINFO=C:\vc-tutorial\DAT\BOOKINFO.DAT
```

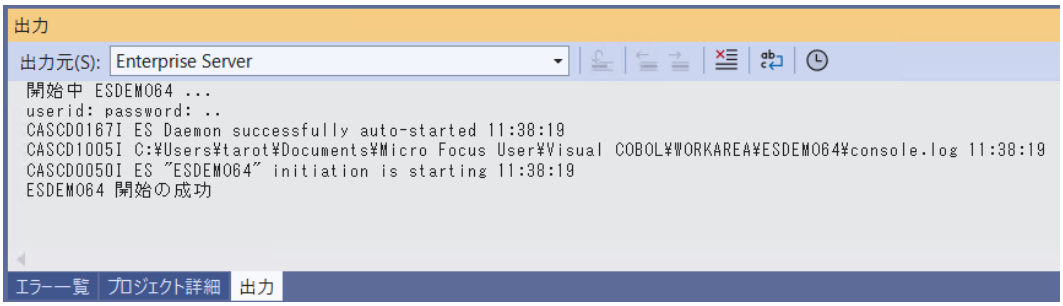
3.3 Enterprise Server の起動

1) Visual Studio に戻ります。[表示(V)] > [サーバーエクスプローラー(V)] を選択します。

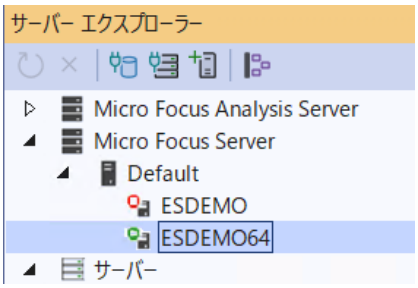
2) [Micro Focus Server] > [localhost] > [ESDEMO64] と展開します。「ESDEMO64」の上で右クリックし、コンテクストメニューから[開始(S)]を選択します。



[出力] に起動メッセージが表示されます。



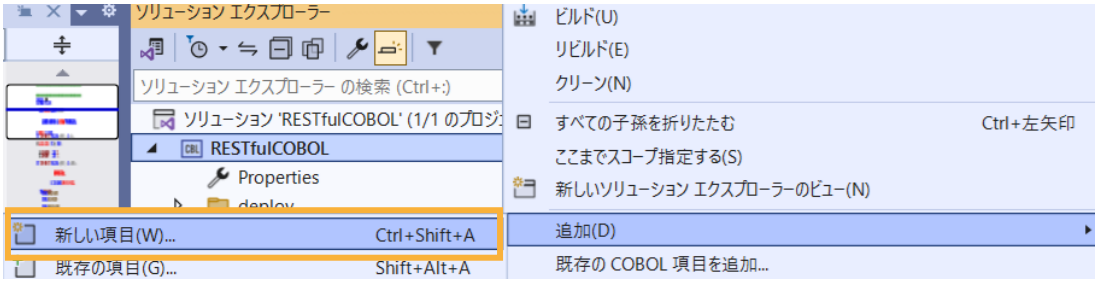
正常に開始されると [サーバーエクスプローラー] 上の ESDEMO64 アイコンが起動されたことを示す緑色のアイコンに切り替わります。



3.4 RESTful Web サービスの開発作業

1) RESTful Web サービスのプロファイル作成

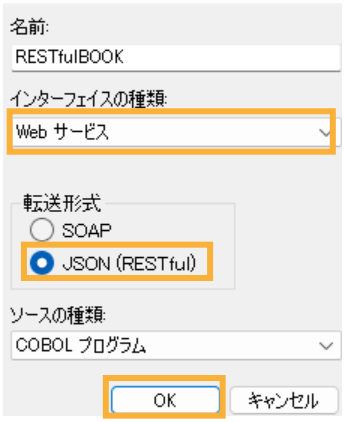
① 「RESTfulCOBOL」プロジェクトを右クリックし、コンテクストメニューから [追加(D)]> [新しい項目(W)] を選択します。



② [COBOL] > [Native] アイテムを選択し、 [Service Interface] を指定します。

③ [名前] に ”RESTfulBOOK.svi” を指定し、[追加(A)] をクリックします。

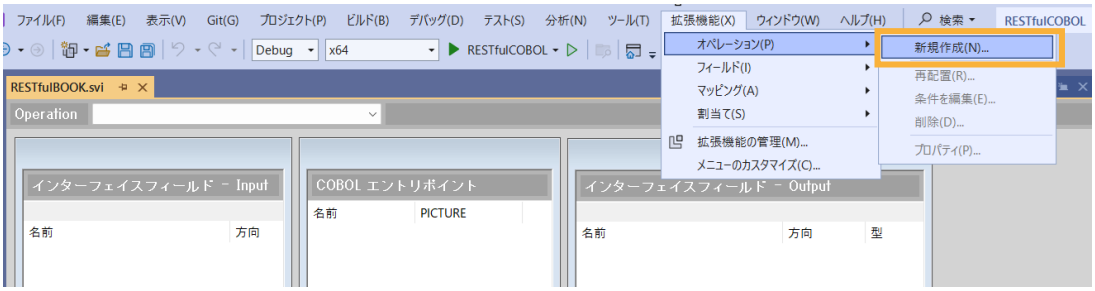
④ [サービスインターフェイス] 画面が表示されるので、[インターフェイスの種類] に「Web サービス」を選択し、[転送形式]は、「JSON (RESTful)」を選択し [OK] をクリックします。



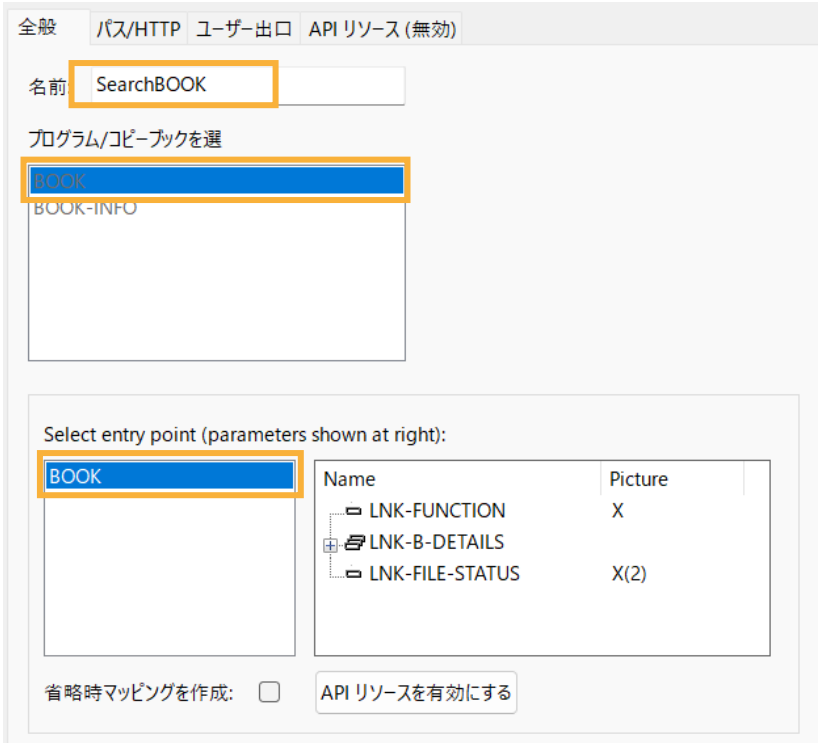
[RESTfulBOOK] のオペレーション初期画面が表示されます。

2) 書籍検索用のオペレーションを作成

① 下図のように Visual Studio 内で「RESTfulBOOK.svi」が開いている状態で [拡張機能] > [オペレーション(P)] メニュー > [新規作成(N)] を選択します。

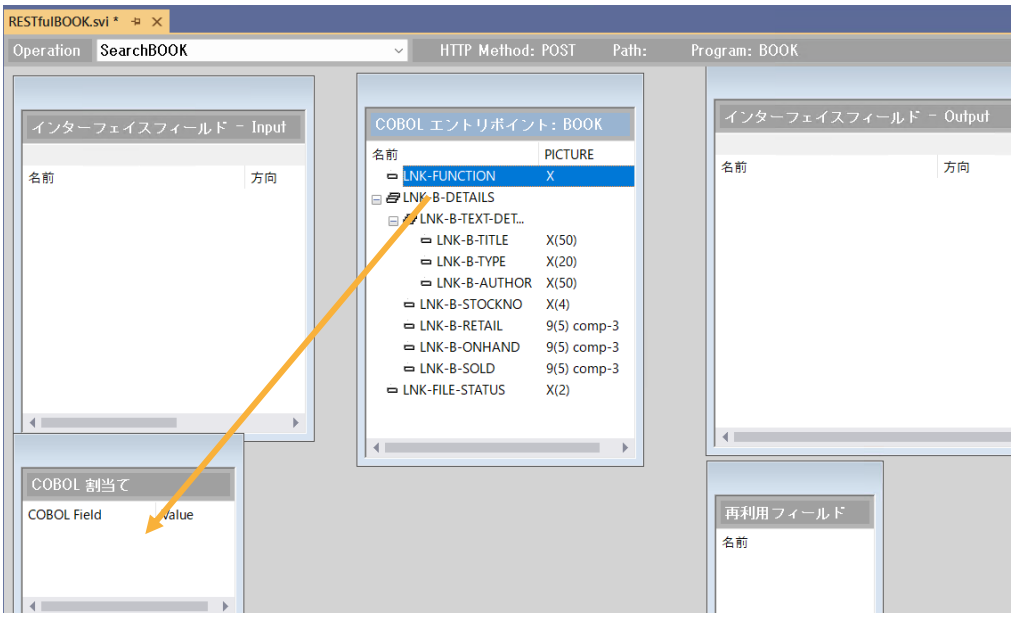


② [オペレーションプロパティ] ダイアログが表示されるので [名前] に ”SearchBOOK” を入力し、[プログラム/コピーブックを選択] では、「BOOK」を選択、[Select entry point] も「BOOK」を選択し、[OK] をクリックします。



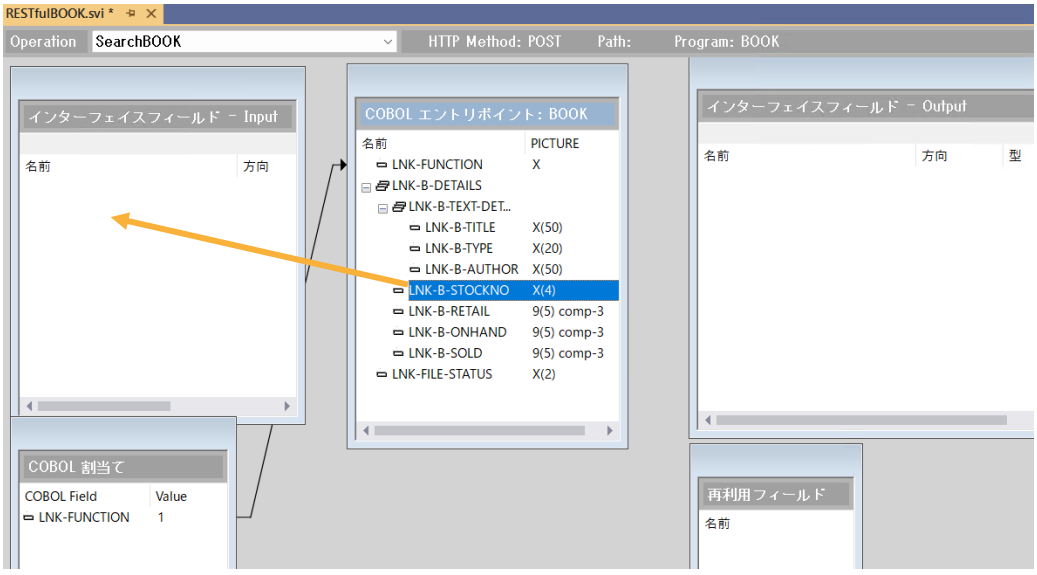
3) COBOL と RESTful Web サービス間の変数型変換マッピングを定義（書籍情報検索用オペレーション）

① COBOL エントリポイントが展開されますので「LNK-FUNCTION」を［COBOL 割当て］ペインにドラッグ＆ドロップします。

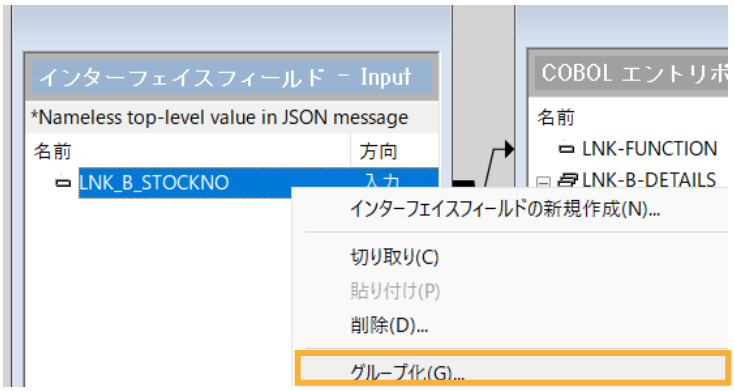


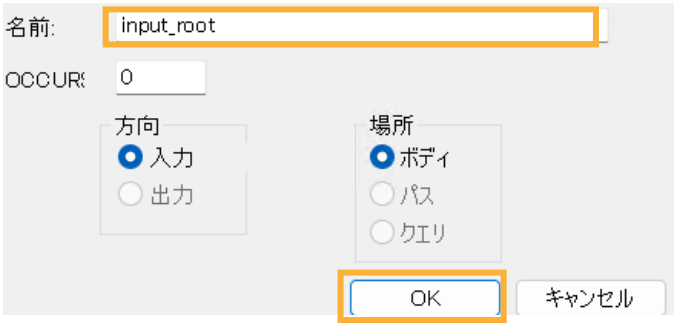
② ［値］フィールドには “1” を入力し、[OK] をクリックします。

③ 次に「LNK-B-STOCKNO」を［インターフェイスフィールド - Input］ペインにドラッグ＆ドロップします。

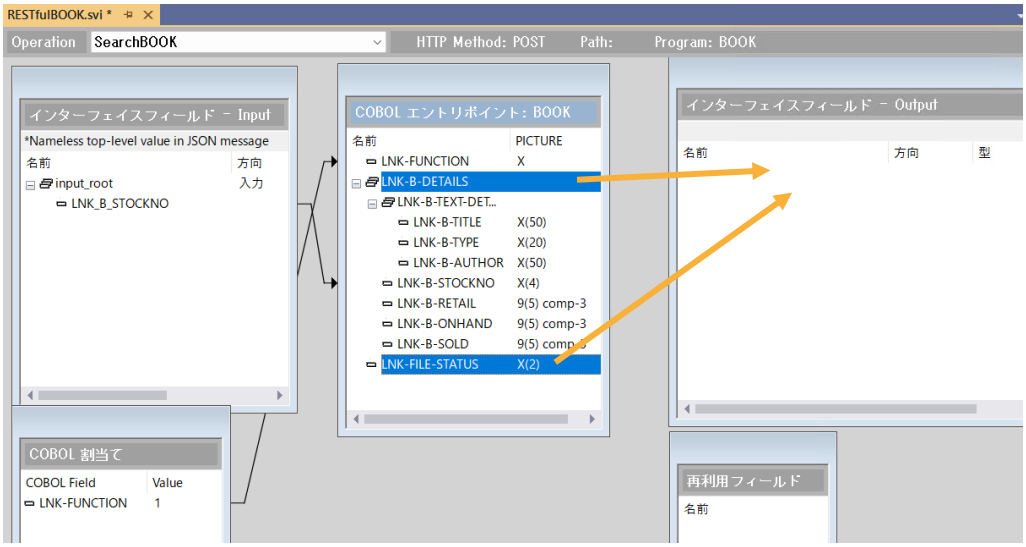


④ ［インターフェイスフィールド - Input］にある「LNK-B-STOCKNO」上で右クリックから[グループ化]を選択し、[グループプロパティ] ウィンドウにて ”input_root” と入力して、 [OK] をクリックします。

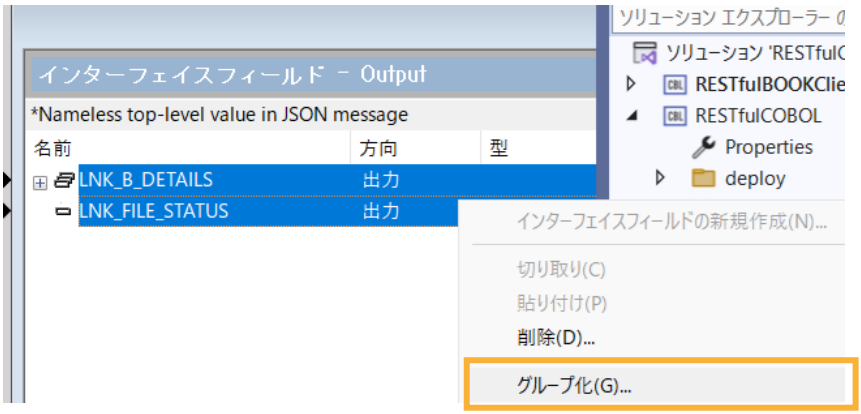


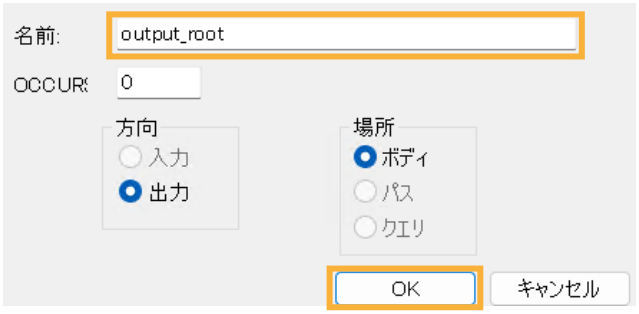


⑤ 「LNK-B-DETAILS」と「LNK-FILE-STATUS」を順に［インターフェイスフィールド - Output］ペインにドラッグアンドドロップします。

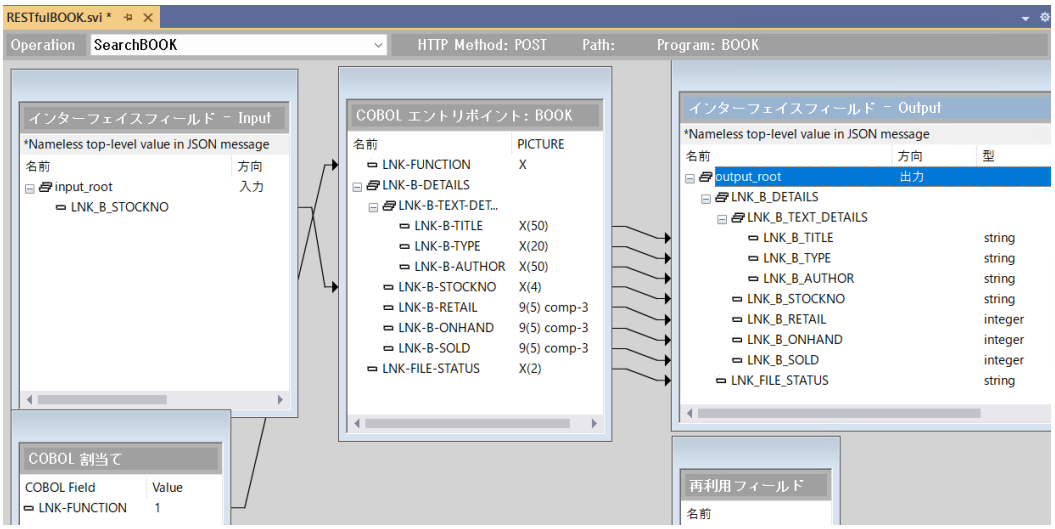


⑥ ［インターフェイスフィールド - Output］にある「LNK-B-DETAILS」及び「LNK-FILE-STATUS」を選択した状態で右クリックから［グループ化］を選択し、［グループプロパティ］ウィンドウにて ”output_root” と入力して、［OK］をクリックします。





⑦ 下記の図のようになっていることを確認して、保存します。

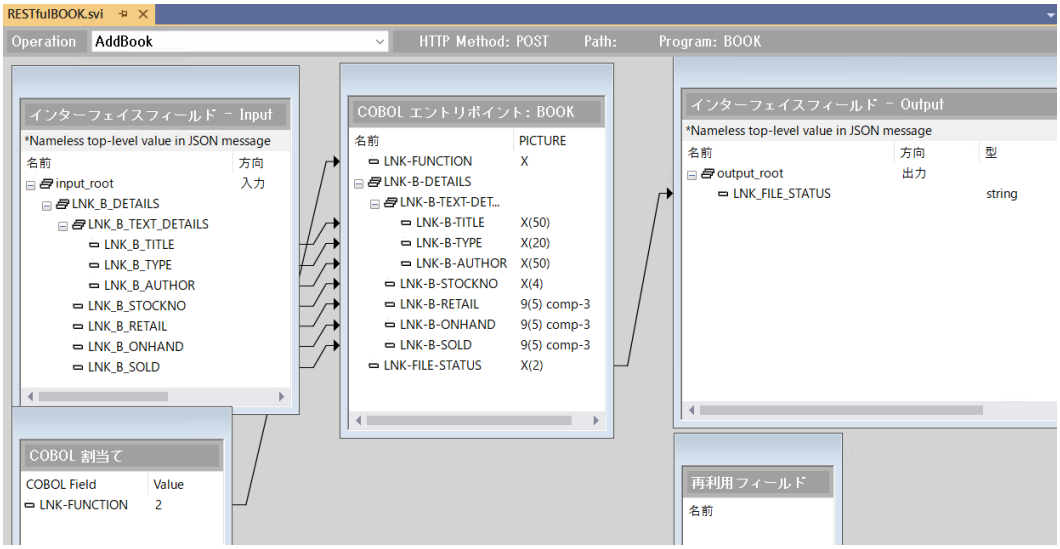


4) 書籍データ追加機能のオペレーションを追加

① さきほどと同じ手順で「AddBOOK」オペレーションを作成し、オペレーションの項目マッピングでは、以下の手順を実施し、保存します。

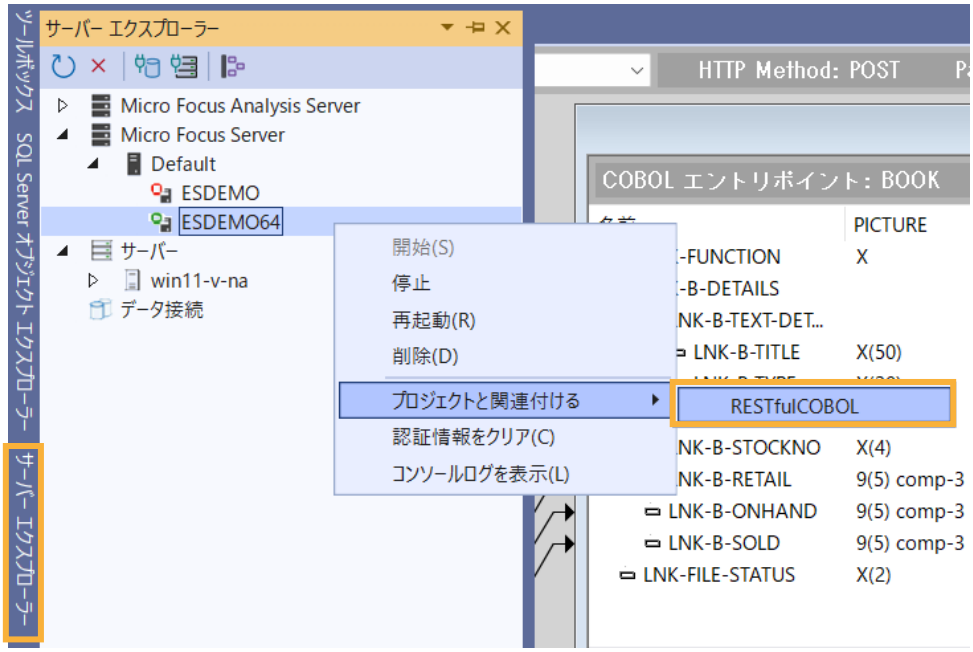
- 「LNK-FUNCTION」項目を［COBOL 割当て］にドラッグ、値に “2” を入力
- 「LNK-B-DETAILS」項目を［インターフェイスフィールド – Input］にドラッグ
  ドラッグした項目からコンテクストメニューを開き、[グループ化] を選択。”input_root” を作成
- 「LNK-FILE-STATUS」項目を［インターフェイスフィールド - Output］にドラッグ
  ドラッグした項目からコンテクストメニューを開き、[グループ化] を選択。”output_root” を作成

作業完了後、以下のようになります。



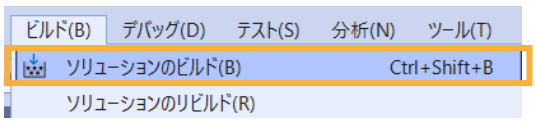
5) プロジェクトと Enterprise Server「ESDEMO64」を関連付ける

① サーバーエクスプローラーにて、[Micro Focus Servers] > [localhost] > [ESDEMO64] を右クリックし、コンテクストメニューから [プロジェクトと関連付ける] > [RESTfulCOBOL] を選択します。



3.5 Enterprise Server ソリューションのビルド作業

1) メニューより、[ビルド(B)] > [ソリューションのビルド(B)] を選択します。



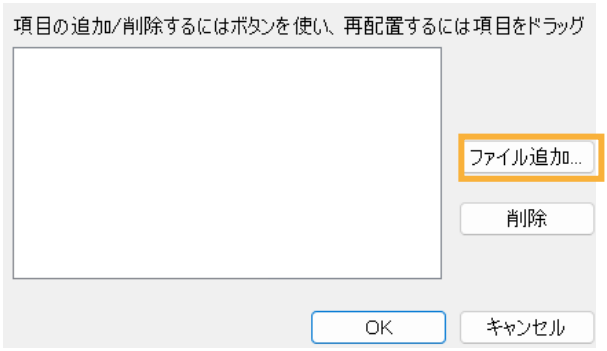
3.6 コンパイルした COBOL アプリケーションを Enterprise Server へディプロイ

1) ディプロイする COBOL プログラムの指定

① ソリューションエクスプローラーにて「RESTfulBOOK.svi」を右クリックし、コンテクストメニューから［プロパティ］を選択します。

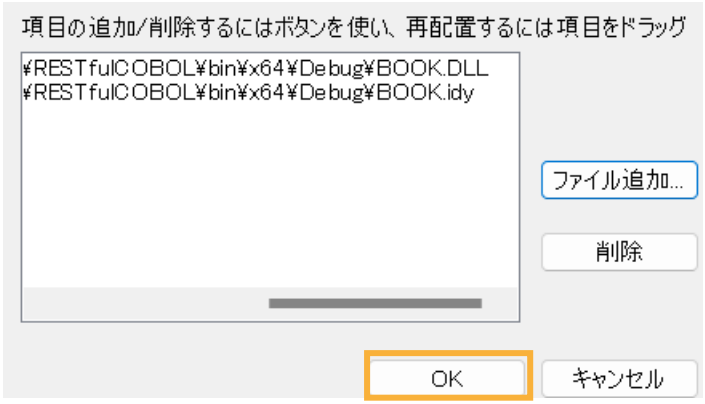
② プロパティペインの［ディプロイするアプリケーションファイル］右横の空欄をクリックしたうえで表示される［...］をクリックします。

③ ［項目の追加/削除］ウィンドウが表示されるので［ファイル追加］を押します。

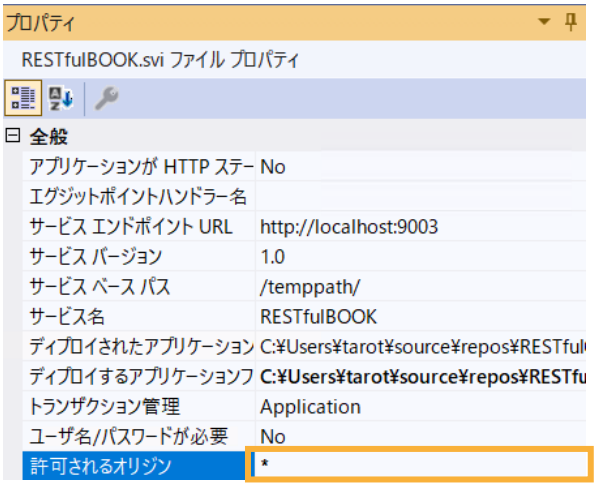


④ エクスプローラーから「VisualStudio プロジェクトフォルダー¥RESTfulCOBOL¥bin¥x64¥Debug」まで移動し、「BOOK.dll」及び「BOOK.idy」を指定します。ファイルが存在しない場合は、[ビルド(B)] > [ソリューションのリビルド(R)] を実行してください。

⑤ ［項目の追加/削除］ウィンドウにファイルがセットされるので［OK］をクリックします。

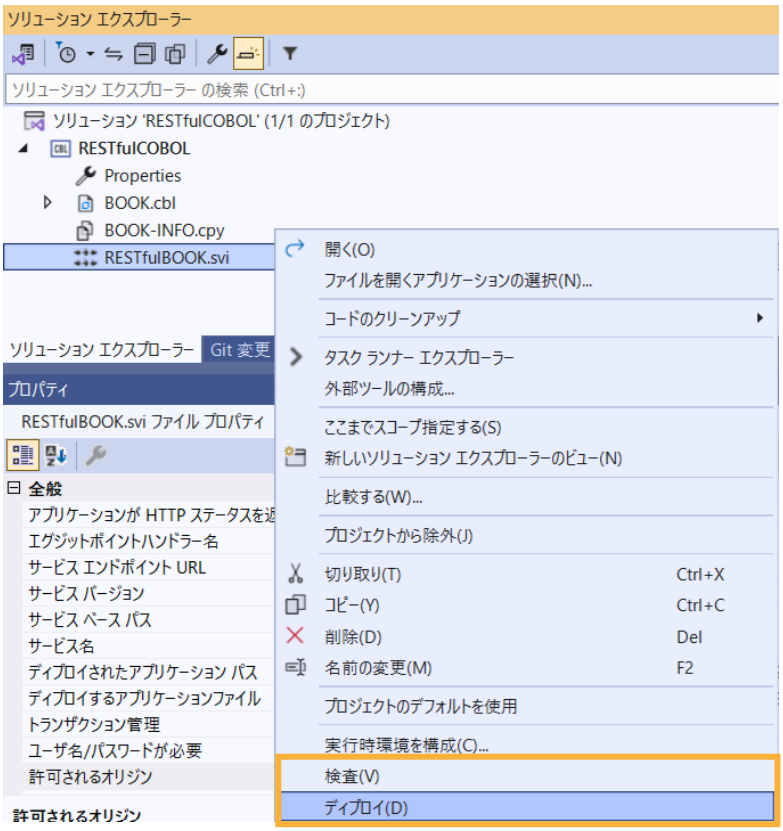


⑥ デフォルトではオリジン間リソース共有は許可されていません。もしこれに関するエラーが発生する場合、許可設定を行います。ここでは[許可されるオリジン]に "*" を入力します。

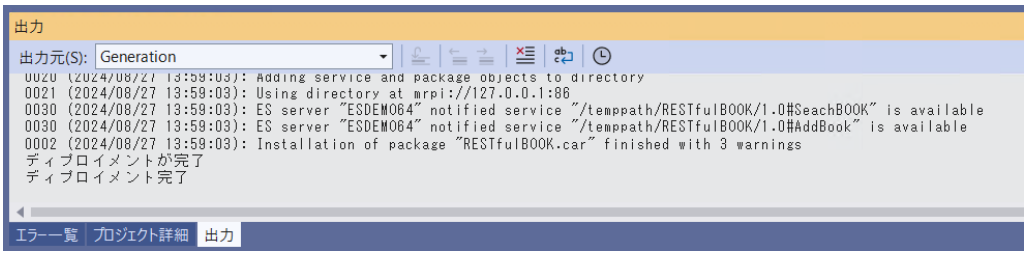


2) RESTful Web サービスのコンポーネント一式を Enterprise Server へディプロイ

① 「RESTfulBOOK.svi」を右クリックし、コンテクストメニューから [検査(V)]を実行し、問題ないことを確認したうえで、[ディプロイ(D)] を選択します。



② ディプロイが完了すると下図のようなメッセージが出力されます（ここでは警告は無視して構いません）。

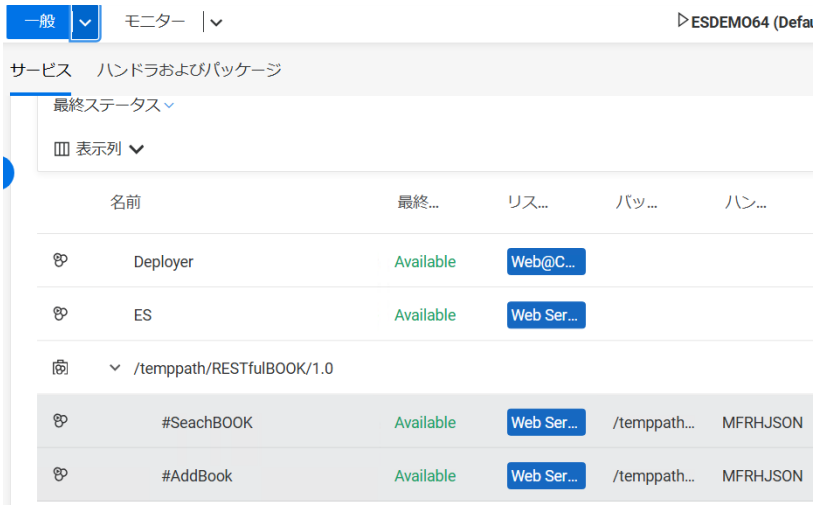


3) ESCWA(Enterprise Server Common Web Administration) からディプロイされたことを確認

① ブラウザー上の ESCWA に切り替えます。

② [一般メニュー] から [サービス] をクリックします。

③ 画面下にスクロールしていくと最下行にディプロイした RESTful Web サービスが追加されていることを確認します。

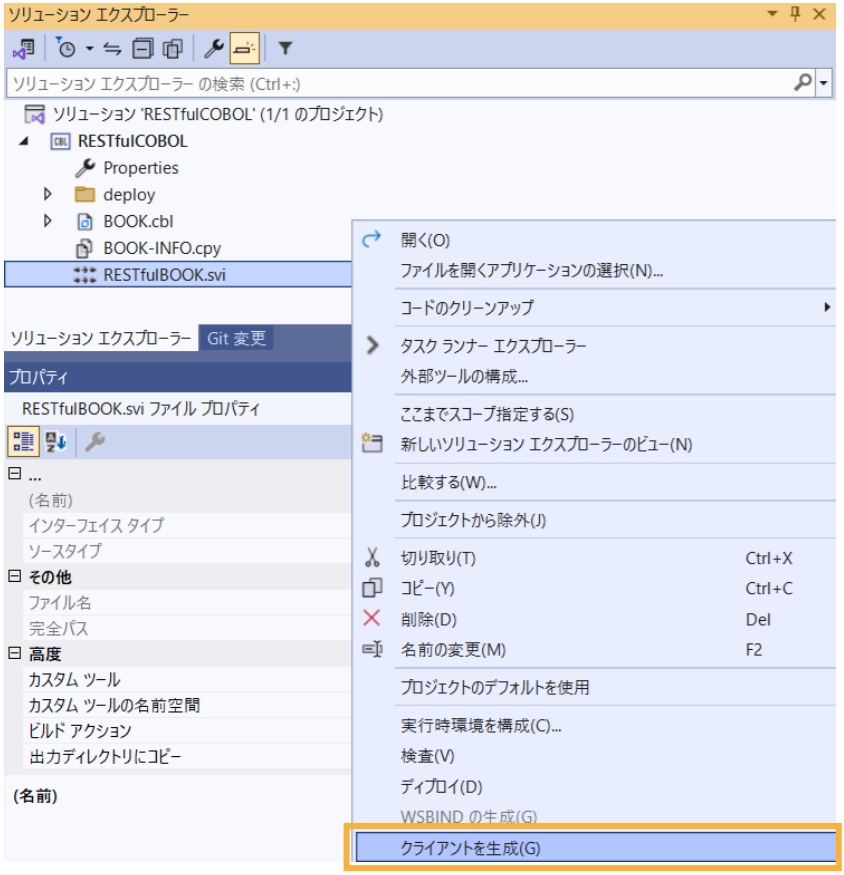


3.7 RESTful Web サービスのテスト

3.7.1 RESTful Web サービスクライアントプログラムの利用

1) RESTful Web サービステスト用のクライアント生成

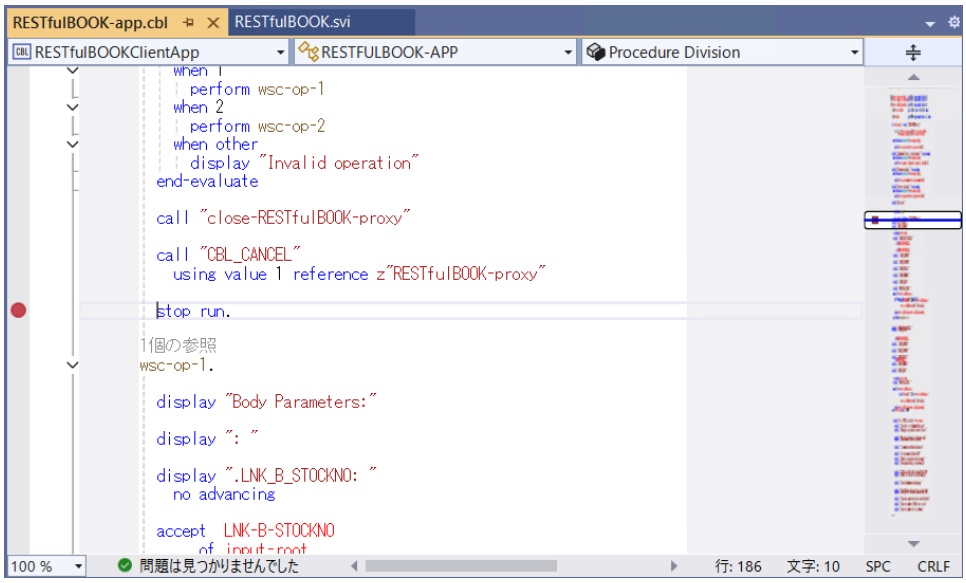
① 「RESTfulBOOK.svi」を右クリックし、コンテクストメニューから [クライアントの生成(G)] を選択します。



2) テスト用アプリケーションの実行準備

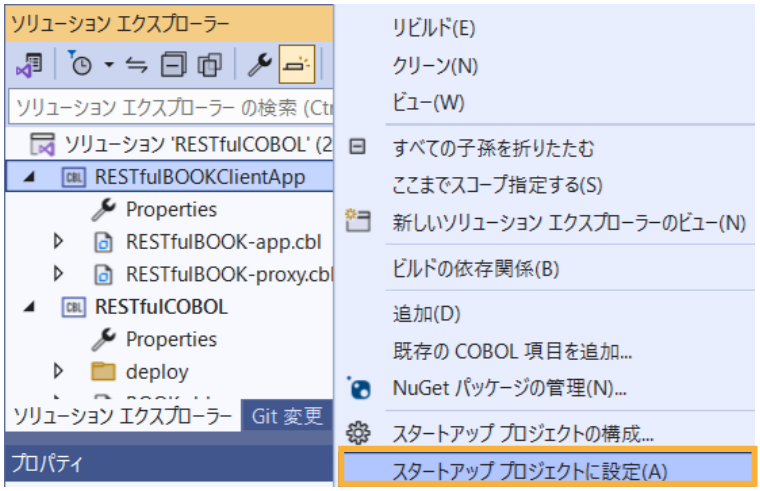
① RESTfulBOOK-app.cbl をダブルクリックして開きます。

② テスト用クライアントは実行を終了すると DOS 画面から消えてしまうため、186 行目にブレークポイントを設定します。



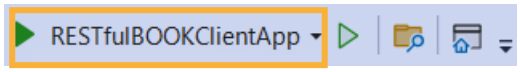
③ メニューより、[ビルド(B)] > [ソリューションのリビルド(R)] を選択します。

④ 「RESTfulBOOKClientApp]プロジェクトを右クリックし、コンテクストメニューから[スタートアッププロジェクトに設定(A)]を選択します。



3) 生成したテスト用 COBOL クライアントの実行（登録処理）

① ツールバーにて RESTfulBOOKClientApp の [開始] アイコンをクリックし、アプリケーションを起動します。DOS プロンプトでアプリケーションが起動します。



プロンプト画面では以下の入力を行います。

Service Address： そのまま Enter キーを押す

Supplemental Query String： そのまま Enter キーを押す

Username： そのまま Enter キーを押す

Password： そのまま Enter キーを押す

Operation： “2” を入力し、Enter キーを押す

LNK_B_TITLE： “PLANET OF THE APES” を入力し、Enter キーを押す

LNK_B_TYPE： “SCIENCE FICTION” を入力し、Enter キーを押す

LNK_B_AUTHOR： “PIERRE BOULLE” を入力し、Enter キーを押す

LNK_B_STOCKNO： “5555” を入力し、Enter キーを押す

LNK_B_RETAIL： “1000” を入力し、Enter キーを押す

LNK_B_ONHAND： “3000” を入力し、Enter キーを押す

LNK_B_SOLD： “2333” を入力し、Enter キーを押す

② RESTful の Web サービスが実行されます。「lnk_FILE_STATUS」に “00” が返ってきたら登録成功です。

```
Service Address (Enter = http://localhost:9003):
Supplemental Query String (optional):
Username (optional):
Password (optional):
Operation (1 = SearchBOOK, 2 = AddBook): 2
Body Parameters:
:
.LNK_B_DETAILS:
..LNK_B_TEXT_DETAILS:
...LNK_B_TITLE: PLANET OF THE APES
...LNK_B_TYPE: SCIENCE FICTION
...LNK_B_AUTHOR: PIERRE BOULLE
..LNK_B_STOCKNO: 5555
..LNK_B_RETAIL: 1000
..LNK_B_ONHAND: 3000
..LNK_B_SOLD: 2333
:
.LNK_FILE_STATUS: 00
```

③ ブレークポイントで止まっているのでツールバーから [続行:] をクリックして処理を終了させます。

4) 生成したテスト用 COBOL クライアント実行（検索処理）

① 再度、ツールバーにて [RESTfulBOOKClientApp] アイコンをクリックし、アプリケーションを起動します。

DOS プロンプトでアプリケーションが起動します。

プロンプト画面では以下の入力を行います。

Service Address： そのまま Enter キーを押す

Supplemental Query String： そのまま Enter キーを押す

Username： そのまま Enter キーを押す

Password： そのまま Enter キーを押す

Operation： “2” を入力し、Enter キーを押す

LNK_B_STOCKNO： “5555” を入力し、Enter キーを押す

さきほど登録した情報が表示されます。

```
Service Address (Enter = http://localhost:9003):
Supplemental Query String (optional):
Username (optional):
Password (optional):
Operation (1 = SearchBOOK, 2 = AddBook): 1
Body Parameters:
:
.LNK_B_STOCKNO: 5555
:
.LNK_B_DETAILS:
..LNK_B_TEXT_DETAILS:
...LNK_B_TITLE: PLANET OF THE APES
...LNK_B_TYPE: SCIENCE FICTION
...LNK_B_AUTHOR: PIERRE BOULLE
..LNK_B_STOCKNO: 5555
..LNK_B_RETAIL: 01000
..LNK_B_ONHAND: 03000
..LNK_B_SOLD: 02333
.LNK_FILE_STATUS: 00
```

② さきほど同様、[続行] アイコンをクリックして、デバッグを終了します。

3.7.2 Curl コマンドによるテスト

作成した RESTful Web サービスは、一般的な仕様であるため、テスト用の COBOL アプリケーションからだけではなく、Java などの他の開発言語などからもアクセスできます。

ここでは、curl コマンドを用いてリクエストが行えることを確認します。

1) Windows のコマンドプロンプト画面を開き、チュートリアル用のファイルを解凍したフォルダーに移動します。ここまでの手順では、C:¥vc-tutorial に解凍しています。

2) 以下のコマンドを実行し、さきほど登録した書籍情報を検索します。

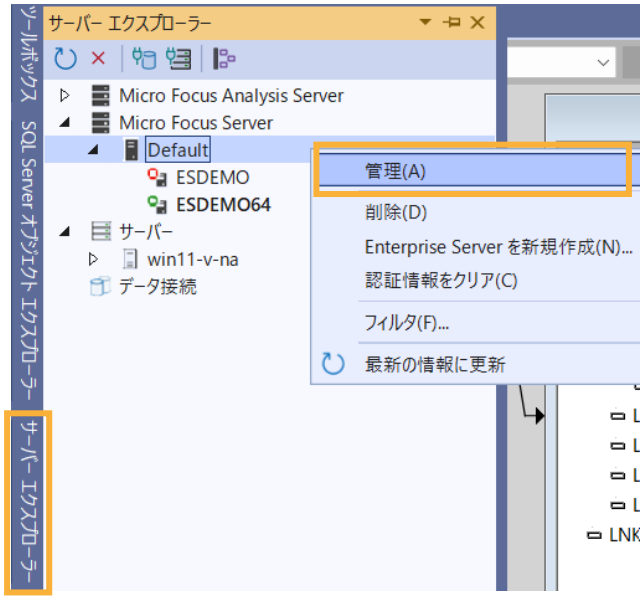
- curl http://localhost:9003/temppath/RESTfulBOOK/1.0/SearchBOOK -d @json¥search5555.txt

```
C:¥vc-tutorial>curl http://localhost:9003/temppath/RESTfulBOOK/1.0/SearchBOOK -d
@json¥search5555.txt
{
  "LNK_B_DETAILS" :
  {
    "LNK_B_TEXT_DETAILS" :
    {
      "LNK_B_TITLE" : "PLANET OF THE APES",
      "LNK_B_TYPE" : "SCIENCE FICTION",
      "LNK_B_AUTHOR" : "PIERRE BOULLE"
    },
    "LNK_B_STOCKNO" : "5555",
    "LNK_B_RETAIL" : 1000,
    "LNK_B_ONHAND" : 3000,
    "LNK_B_SOLD" : 2333
  },
  "LNK_FILE_STATUS" : "00"
}
C:¥vc-tutorial>
```

3.8 RESTful Web サービスのデバッグ

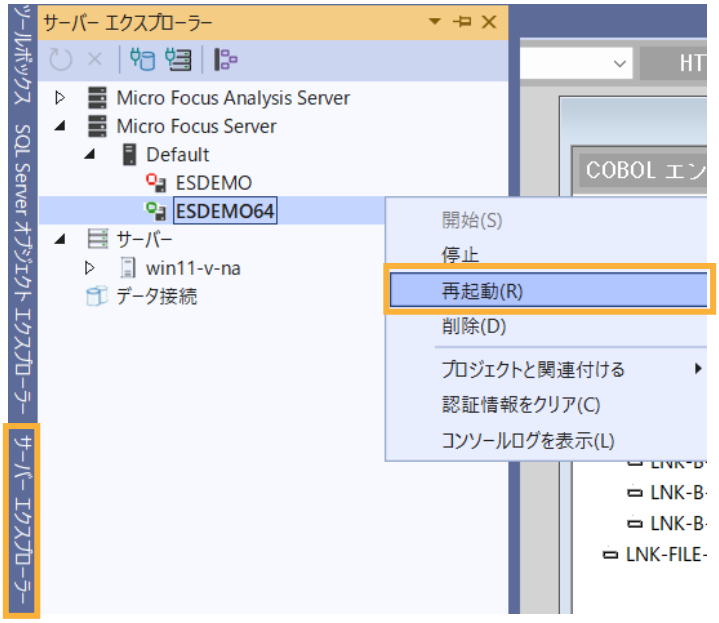
1) Visual Studio に戻り、[サーバーエクスプローラー] タブを選択します。

2) [Micro Focus Server] > [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(A)] を選択し、ESCWA 管理画面にログインします。

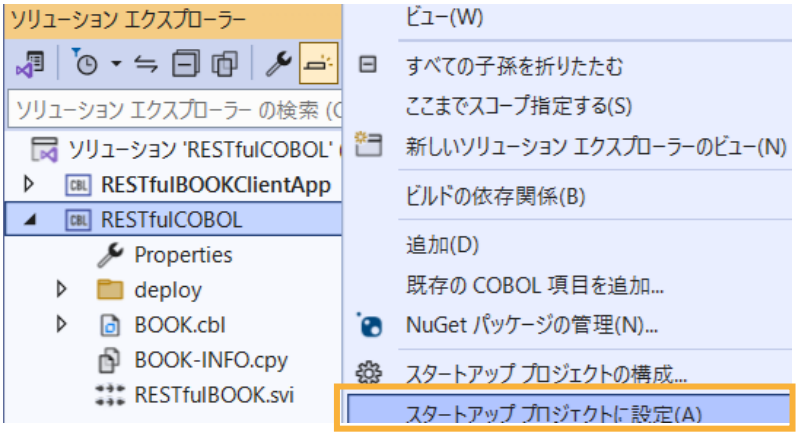


3) [Directory Server] > [Default] > [ESDEMO64] を選択し、[動的デバッグを許可] にチェックしたうえで、[適用] をクリックします。

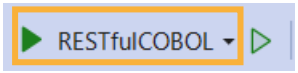
4) Visual Studio に戻り、サーバーエクスプローラーの [Micro Focus Server] > [Default] > [ESDEMO64] を選択し、マウスの右クリックでコンテクストメニューを開き、[再起動(R)] を選択します。



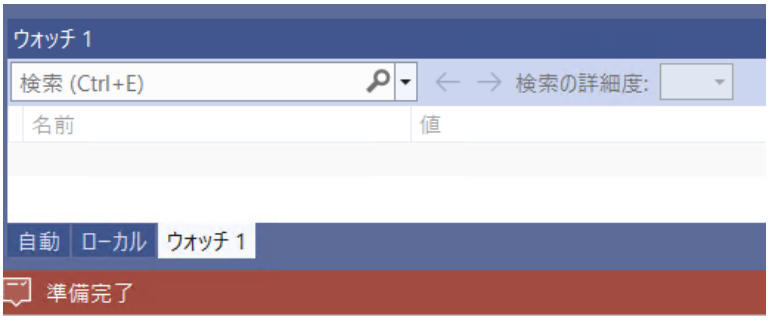
5) ソリューションエクスプローラーより、[RESTfulCOBOL] プロジェクトを選択し、コンテクストメニューを開き、[スタートアッププロジェクトに設定(A)] を選択します。



6) [RESTfulCOBOL] のデバッグアイコンをクリックして、デバッグを開始します。



画面左下に、”準備完了” という文字が表示され、デバッグ待機状態になります。

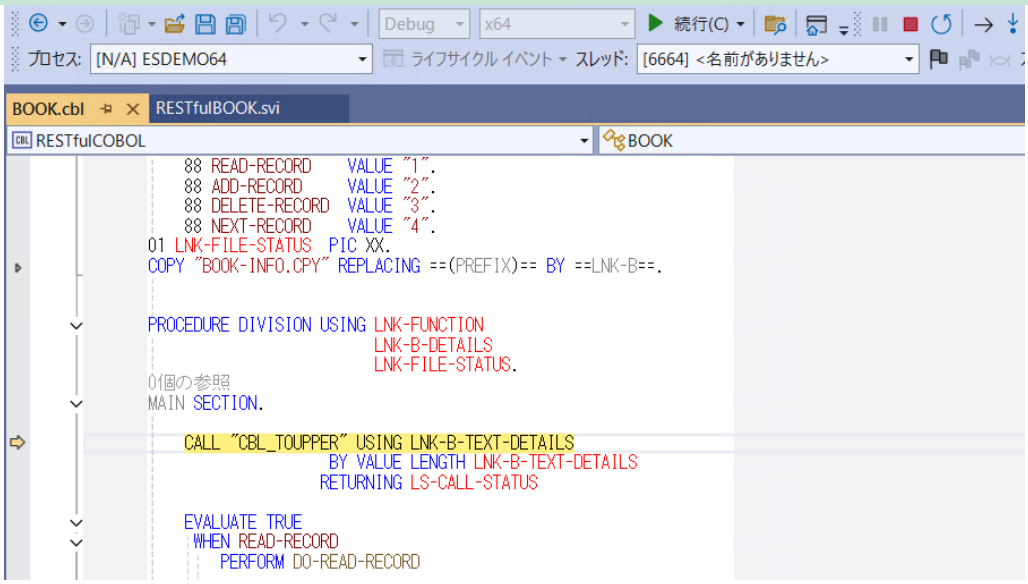


7) 0 と同様のリクエストを curl コマンドで送信します。

プロンプト画面ではリクエストがすぐに戻らず、Visual Studio 側では最初の処理で停止しています。

補足）

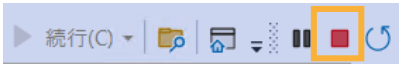
localhost がデバッグ対象にならないことがあります。この場合は、localhost を 127.0.0.1 に変更してください。



この状態では、通常のプログラム同様、ステップインやブレークポイントなどが利用できます。

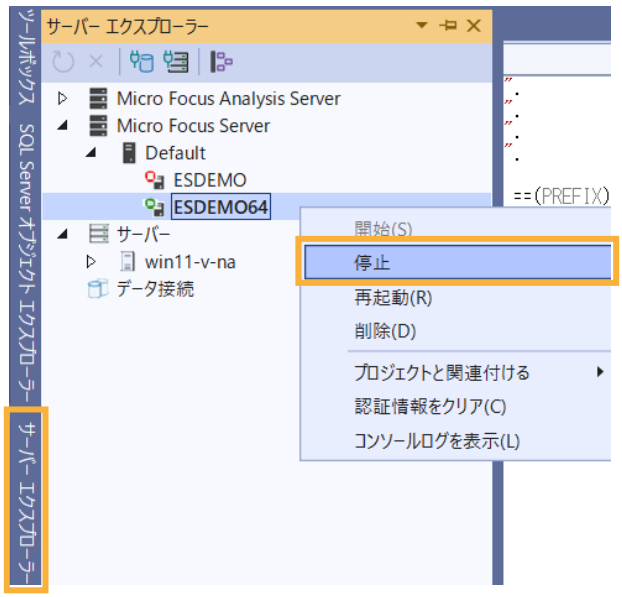
[続行(C)] を押して、処理を完了させると、プロンプト画面に結果が戻されます。

8) デバッグ終了のアイコンをクリックして、デバッグを終了します。

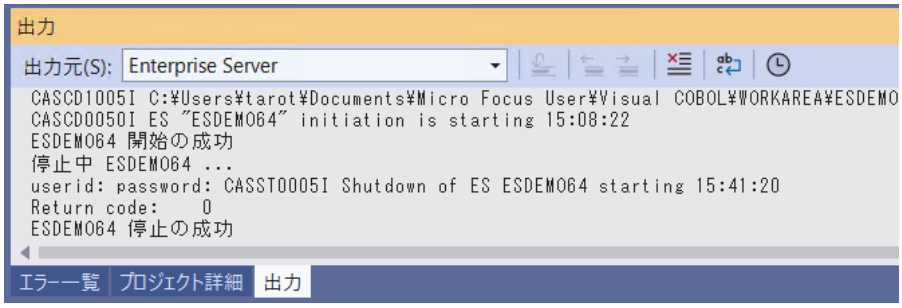


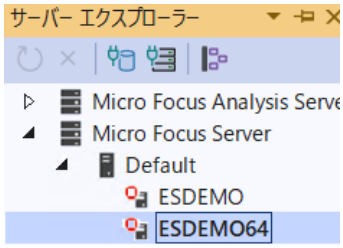
3.9 Enterprise Server の停止

1) サーバーエクスプローラーを選択し、[Micro Focus Server] > [Default] > [ESDEMO64] を選択し、コンテクストメニューから [停止] を選択します。



停止ログが出力され、アイコンが赤色になります。





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。