Rocket software

Visual COBOL チュートリアル

ファイルハンドラーを使ったデータファイルアクセス

1　目的

メインフレームやオフコンからオープンプラットフォームに COBOL 資産を移行した時に必要になる処理は、やはりデータファイルアクセスではないでしょうか。Visual COBOL を使用しないでこれらのデータファイルにアクセスする場合、OS が提供するローレベルな API を使ってカーネルディスクドライバーを経由しファイルにアクセスします。ただし、この API が提供するのは低水準のバイトストリームファイルアクセスであり、COBOL アプリケーションから発行されるレコード単位による I/O 処理には対応できません。Visual COBOL は、 COBOL アプリケーションと OS の API の間にファイルハンドラーというインターフェースを介して COBOL アプリからの各種 I/O のハンドリングを行います。これにより、オープンプラットフォームにおいても、これまで同様、レコード単位によるデータファイルへのアクセスが可能になります。データファイルの種類には順編成ファイル、索引編成ファイルなどの種類が存在します。

順編成ファイルは、バッチ処理で扱うトランザクションファイルの操作に優れています。多くのバッチ処理は、夜間などのコンピュータ資源を占有できる環境で実行されるため、排他制御などのオーバーヘッドを受けることなく効率的に実行できる点にメリットがあります。一方で、夜間のバッチ処理は翌朝の業務開始時刻までには確実に処理が完了していることが必須であり、オンライン処理のレスポンスタイムと同様に、あるいは、それ以上に厳しい性能要求にさらされる処理となっています。たとえ 0.01 秒しかかからない処理でも 100 万回繰り返すことで 3 時間以上になりますので、大量バッチ処理を定時までに完了させるためには慎重なプログラミングが必要となります。トランザクションファイル 1 件の読み込みでも最も高速な手段が要求されるため、これを COBOL の順編成ファイルとして扱うことは最適な選択となります。
索引編成ファイルは、キー値で指定されたファイル内の固有のレコードにランダムにアクセスできる機能を持つファイル編成です。索引編成ファイルはマスターファイルとして使用されます。たとえば従業員マスターファイルは全従業員に関する様々な情報を保持していますが、従業員番号や従業員名などでランダムアクセスできなければなりません。
索引編成ファイルの概念は SQL を使い慣れた方には理解しやすいものです。実際、歴史的には索引編成ファイルは、データベースの概念に先立って登場しており、その後複数の索引編成ファイルの有機的な統合を実現するためにデータベース管理システムが考案された経緯があります。

このチュートリアルでは、ファイルハンドラーを使用したファイルのアクセス方法、ソートユーティリティの概要、rebuild ユーティリティの使用方法を学びます。

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 製品：　Visual COBOL 10.0 for Visual Studio

- チュートリアル用プログラム

下記のリンクから事前にチュートリアル用のプログラムやデータ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。

プログラムおよびデータファイルのダウンロード

内容

1 目的

2 前提

3 チュートリアル手順

3.1 データファイルの準備

3.2 Visual COBOL for Visual Studio の起動とプロジェクト作成の準備

3.3 Native COBOL プログラムの作成と動作確認

3.4 .NET COBOL プログラムの作成と動作確認

3.5 クラシックデータファイルツール

3.6 ソートユーティリティ MFSOFT の確認

3.7 ファイル管理ユーティリティ REBUILD の確認

3 チュートリアル手順

3.1 データファイルの準備

1) チュートリアル用データファイルの用意

ダウンロードしたファイルを任意の場所に解凍します。work フォルダー、MFSORT フォルダー、REBUILD フォルダーが入っていることを確認します。本チュートリアルでは、これらのフォルダーは C ドライブ直下に解凍しています。以降の説明において、これらのフォルダーを参照する場合は、パスを適宜読み替えてください。

3.2 Visual COBOL for Visual Studio の起動とプロジェクト作成の準備

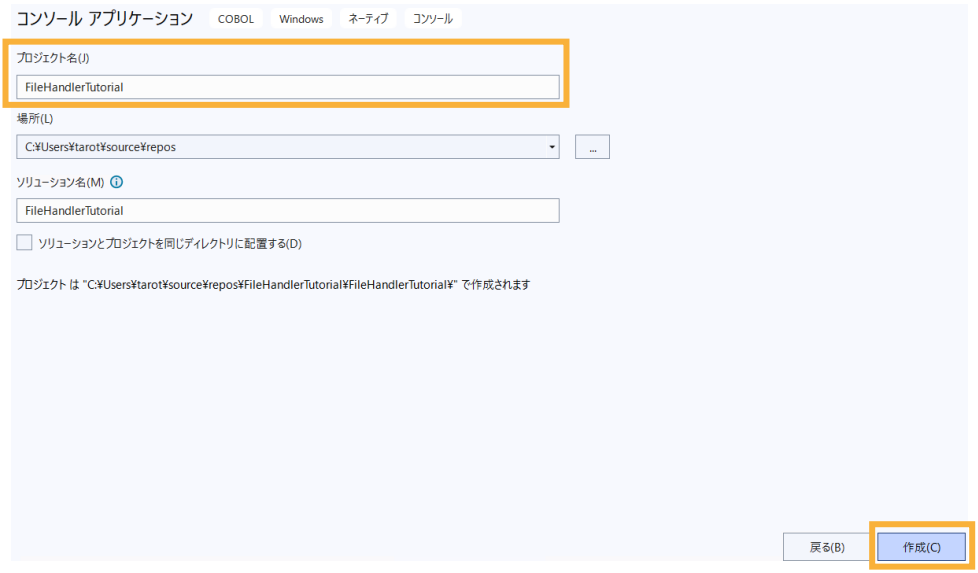
2) Visual COBOL for Visual Studio 2022 の起動とプロジェクトの作成

① スタートメニューより Visual Studio を起動します。

② 「新しいプロジェクトの作成」を選択します。

③ [言語] を「COBOL」、[プラットフォーム] を「Windows」、[プロジェクト タイプ] に「コンソール」を選択し、[コンソール アプリケーション] を選択し、[次へ(N)] をクリックします。

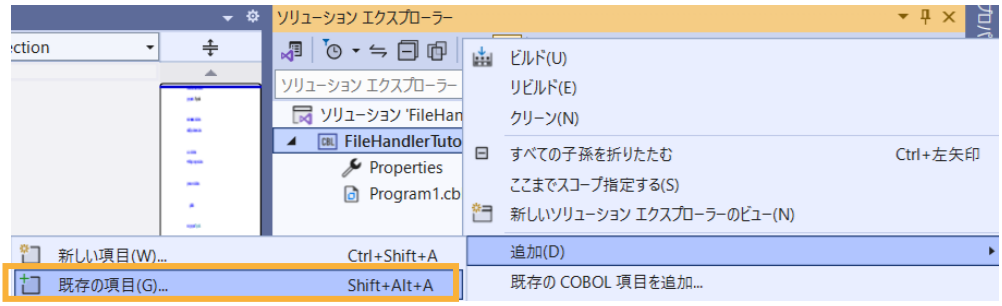
④ [プロジェクト名] に “FileHandlerTutorial” を入力し、[作成(C)] をクリックします。



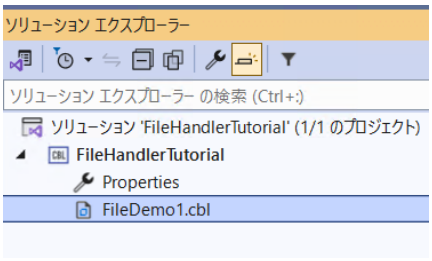
3.3 Native COBOL プログラムの作成と動作確認

1) Native COBOL プロジェクトの作成とプログラムのインポート

① 「FileHandlerTutorial」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)] > [既存の項目(G)] を選択します。



② 解凍した C:¥Work フォルダーから「FileDemo1.cbl」を選択します。また、Program1.cbl は削除します。

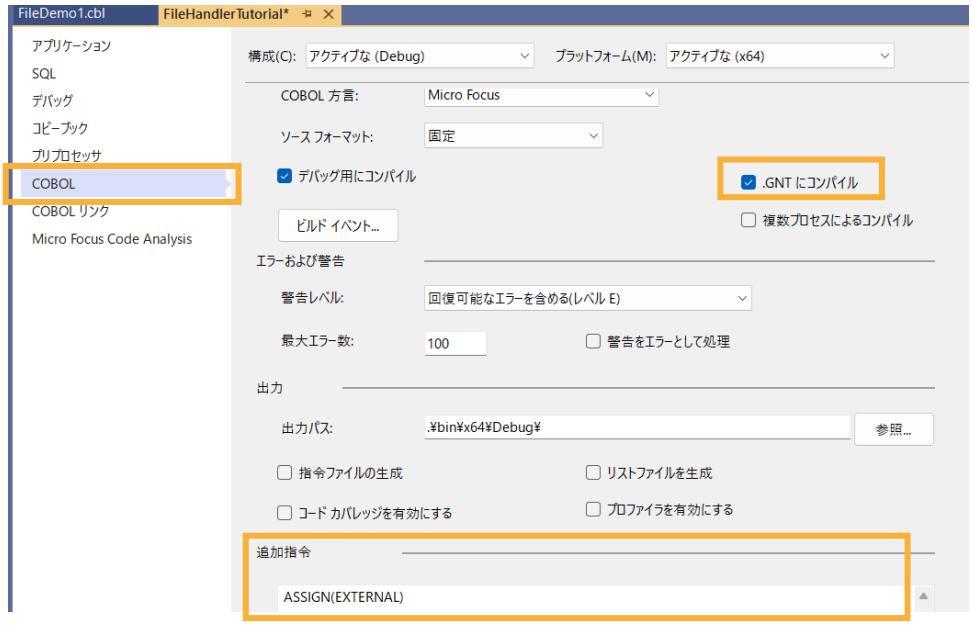


2) プロジェクトプロパティの指定

① プロジェクトの構成を変更します。ソリューションエクスプローラーにて、「FileHandlerTutorial」プロジェクトの配下にある「Properties」をダブルクリックします。

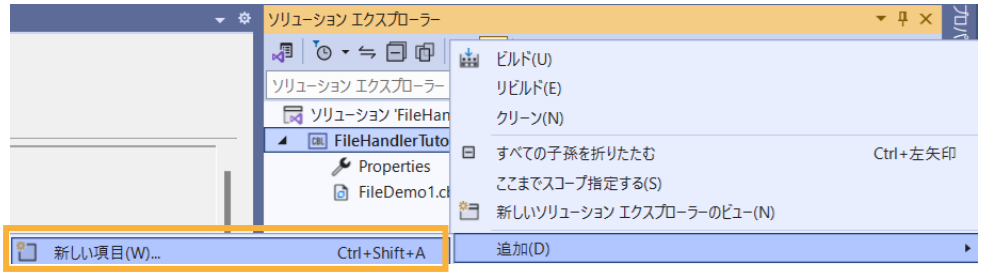
② 左側の [アプリケーション] をクリックし、[出力の種類] を「INT/GNT」に変更します。

③ 左側の [COBOL] をクリックし、[.GNT にコンパイル] を有効にして、[追加指令] に "ASSIGN(EXTERNAL)" を追加します。その後、設定を保存してください。

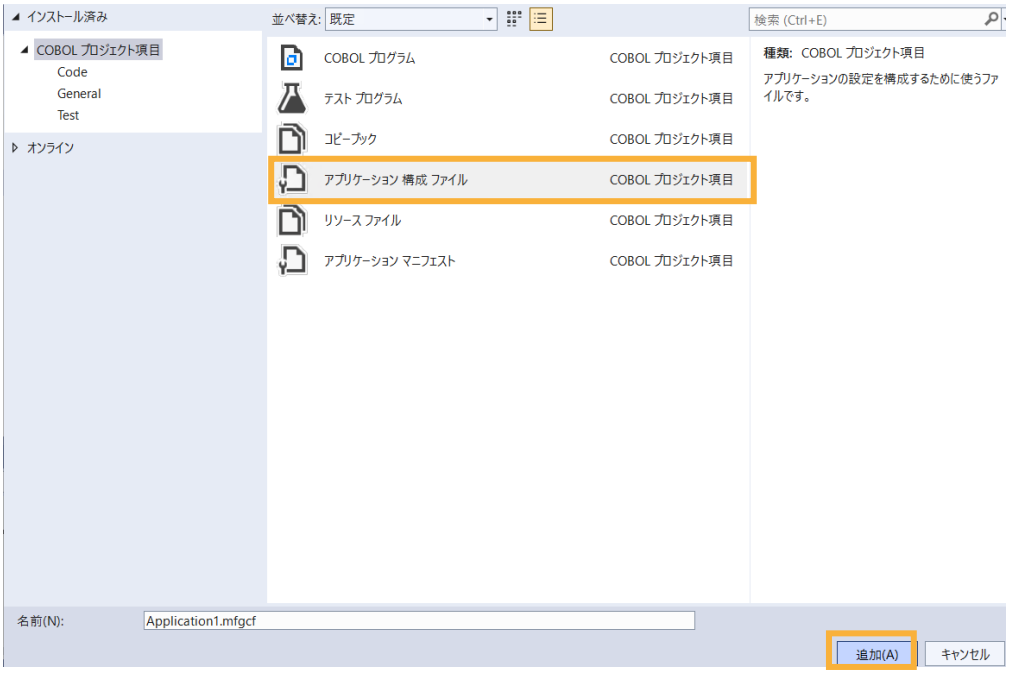


3) プログラムの実行

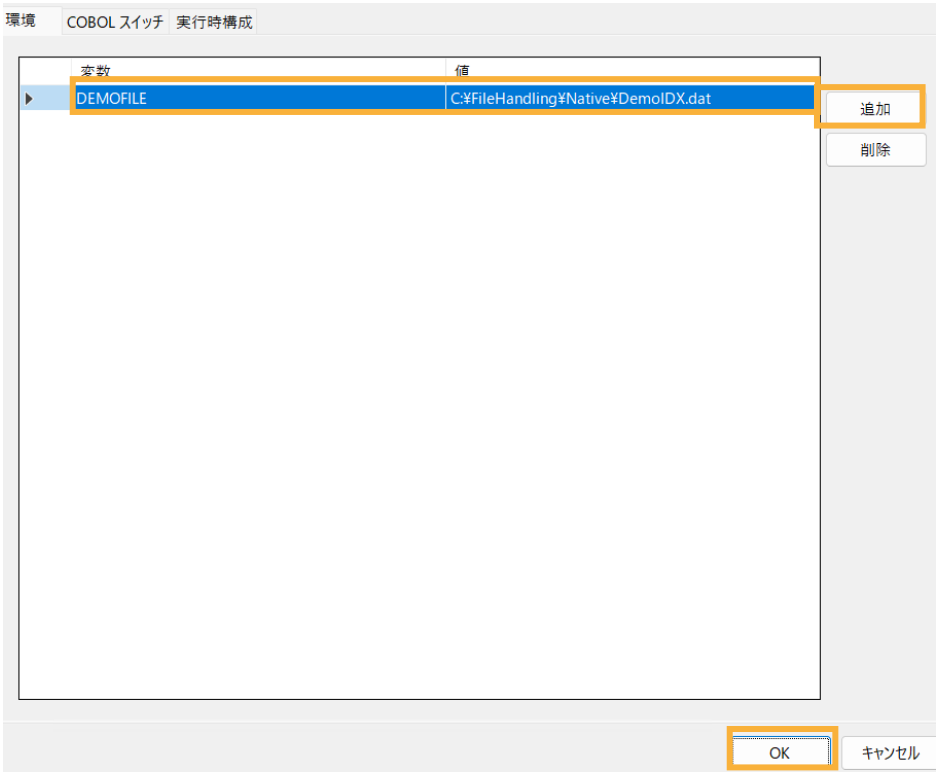
① 「FileHandlerTutorial」プロジェクトを右クリックし、コンテキストメニューから [追加(D)] > [新しい項目(W)] を選択します。



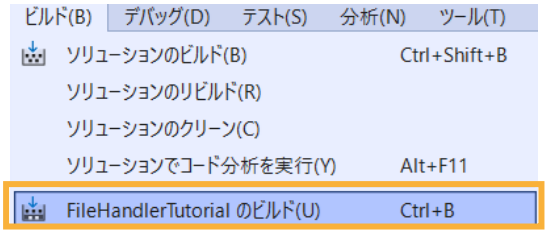
② [アプリケーション構成ファイル]を選択し、[追加(A)] をクリックします。



③ 「Application1.mfgcf」ファイルをダブルクリックします。[追加] をクリックし、[名前] に ”DEMOFILE” 、[値] に “C:¥FileHandling¥Native¥DemoIDX.dat” を入力したうえで、 [OK] をクリックし画面を閉じます。

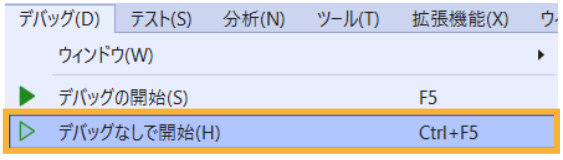


④ [ビルド(B)] > [FileHandlerTutorial のビルド(U)] を選択し、ビルドを行います。



⑤ エクスプローラーなどを用いて、「C:¥FileHandling¥Native」ディレクトリを作成します。

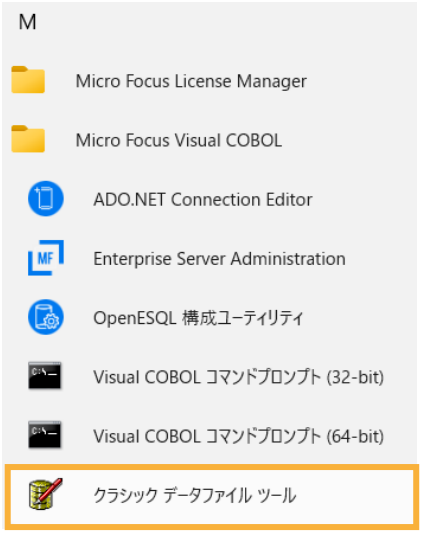
⑥ [デバッグ(D)] > [デバッグなしで開始(H)] をクリックします。



⑦ プログラムが実行され、ファイル作成が行われます。

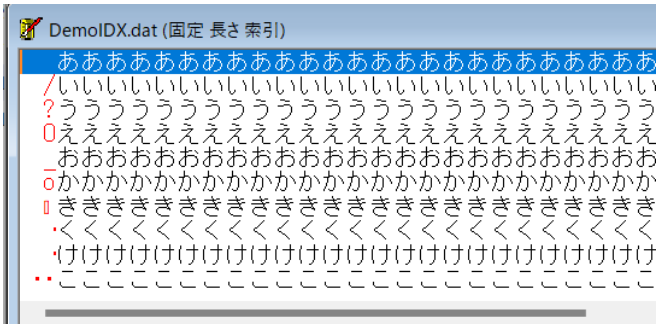
4) 実行結果の確認

① Windows のスタートメニューから [Micro Focus Visual COBOL] > [クラシックデータファイルツール] を選択します。



② [ファイル(F)] メニュー から [開く(O)] を選択し、「C:¥FileHandling¥Native¥DemoIDX.dat」をオープンします。

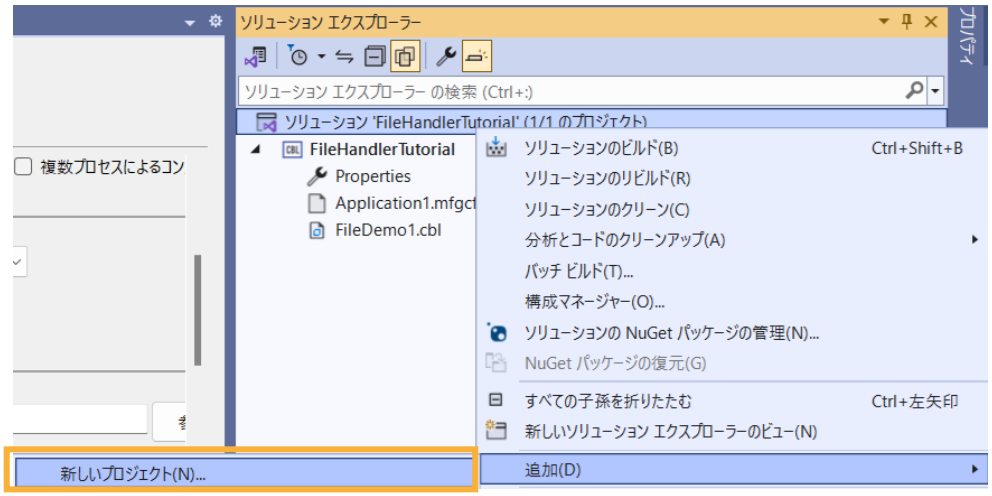
③ プログラムで指定したファイルが作成されていることを確認します。



3.4 .NET COBOL プログラムの作成と動作確認

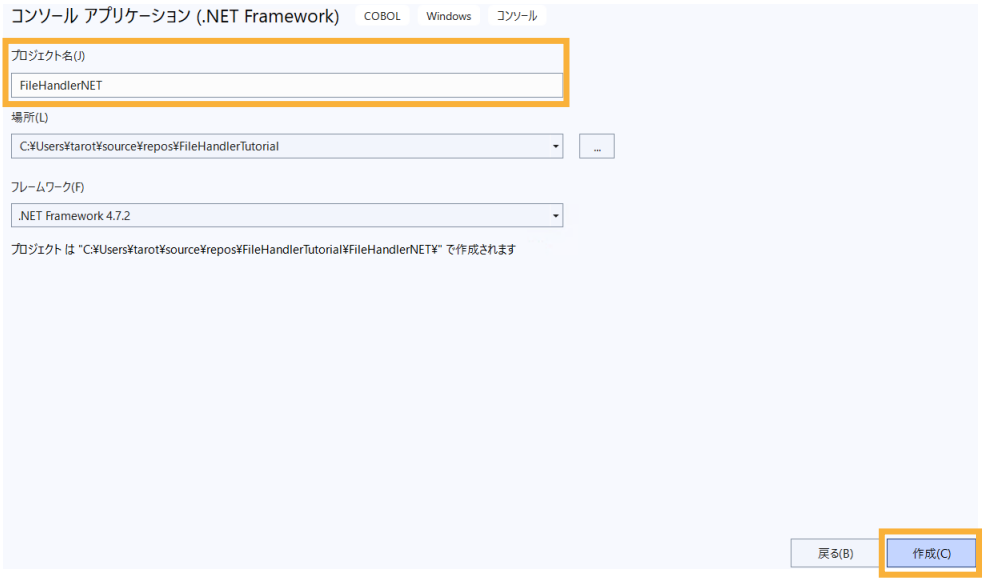
1) .NET COBOL プロジェクトの作成とプログラムのインポート

① 「FileHandlerTutorial」ソリューションの上で右クリックし、コンテクストメニューから [追加(D)] > [新しいプロジェクト(N)] を選択します。

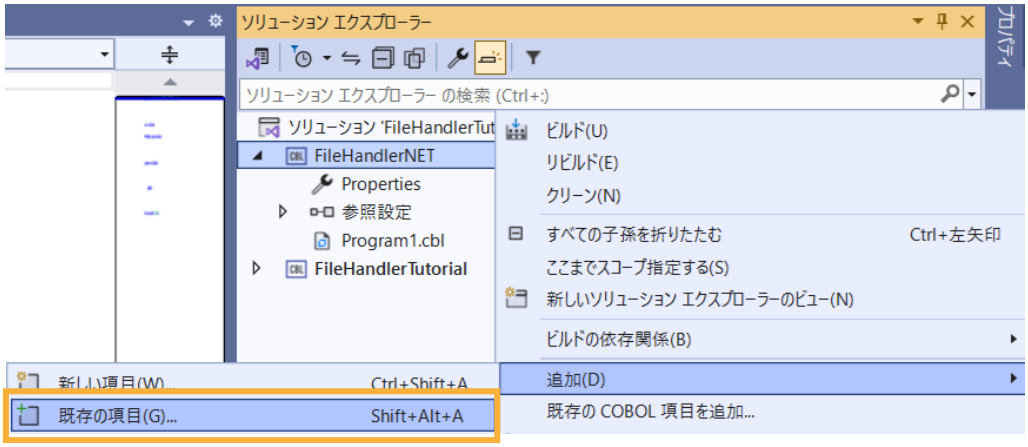


② [言語] を「COBOL」、[プラットフォーム] を「Windows」、[プロジェクト タイプ] に「コンソール」を選択し、[コンソールアプリケーション（.NET Framework）] を選択したうえで、[次へ(N)] をクリックします。

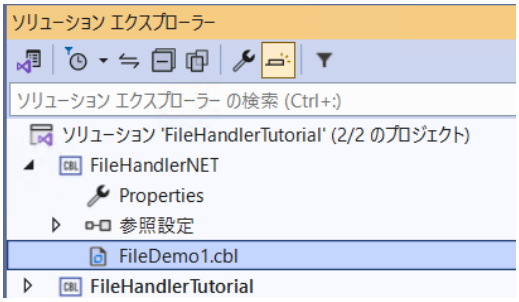
③ [プロジェクト名] に ”FileHandlerNET” を入力し、[作成(C)] をクリックします。



④ 「FileHandlerNET」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)］＞［既存の項目(G)］を選択します。



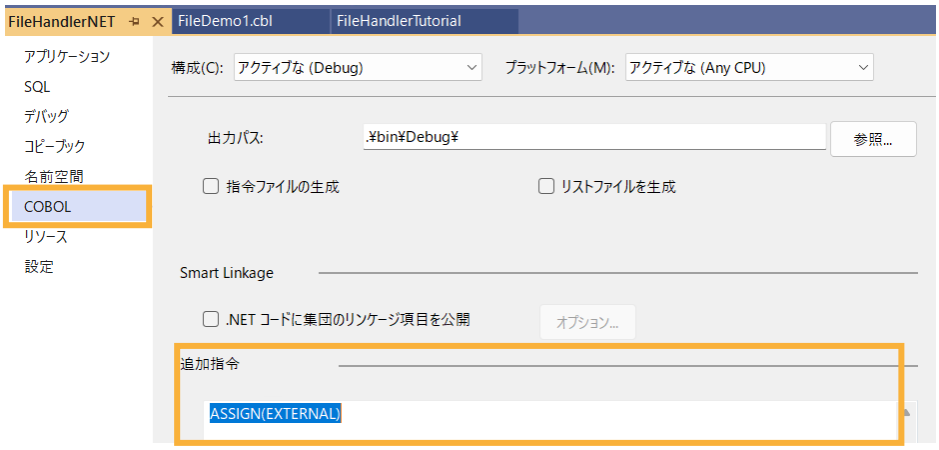
⑤ 解凍した C:¥Work フォルダーから「FileDemo1.cbl」を選択します。また、Program1.cbl は削除します。



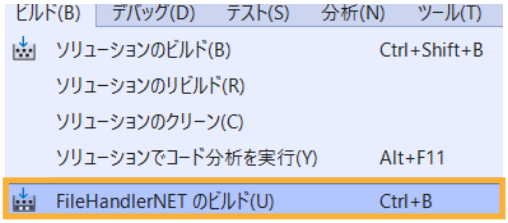
2) プロジェクトプロパティの指定

① プロジェクトの構成を変更します。ソリューションエクスプローラーにて、「FileHandlerNET」プロジェクトの配下にある「Properties」をダブルクリックします。

② 左側の［COBOL］をクリックし、[追加指令］に ”ASSIGN(EXTERNAL)” を追加して、変更を保存します。



③ ［ビルド(B)］＞［FileHandlerNET のビルド(H)］を選択し、ビルドを行います。



3) プログラムの実行

さきほど同様、Visual Studio 上からも実行できますが、今回はコマンドライン上で実行します。

① Windows のスタートメニューから [Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(64-bit)] を選択します。

② DOS プロンプト上で環境変数をセットします。

```
C:¥Users¥tarot¥Documents>SET DEMOFILE=C:¥FileHandling¥NET¥DemoIDX.dat
```

③ CD コマンドで「C:¥FileHandling」に移動します。MKDIR コマンドで NET フォルダーを作成します。

```
C:¥Users¥tarot¥Documents>cd ¥FileHandling
C:¥FileHandling>mkdir NET
C:¥FileHandling>
```

④ 次に実行形式のファイルがあるディレクトリに移動します。

```
C:¥FileHandling>cd "¥Users¥tarot¥source¥repos¥FileHandlerTutorial¥FileHandlerNET¥bin¥Debug"
C:¥Users¥tarot¥source¥repos¥FileHandlerTutorial¥FileHandlerNET¥bin¥Debug>
```

注意）

環境に合わせてパスを変更してください。指定するパスは、FileHandlerNET プロジェクトフォルダー¥bin¥Debug です。

⑤ 「FileHandlerNET.exe」を実行します。プログラムが起動し、ファイル作成が行われます。「C:¥FileHandling¥NET¥DemoIDX.dat」が作成されます。

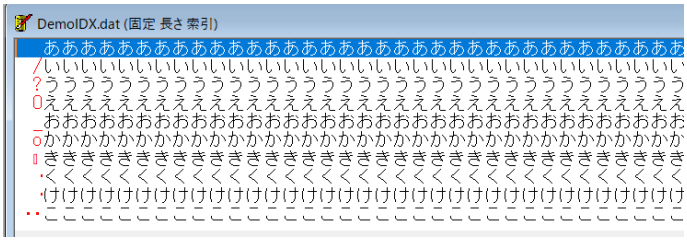
```
C:¥Users¥tarot¥source¥repos¥FileHandlerTutorial¥FileHandlerNET¥bin¥Debug>FileHandlerNET.exe

C:¥Users¥tarot¥source¥repos¥FileHandlerTutorial¥FileHandlerNET¥bin¥Debug>dir c:¥FileHandling¥NET
 ドライブ C のボリューム ラベルがありません。

 c:¥FileHandling¥NET のディレクトリ

2024/08/30  11:08    <DIR>          .
2024/08/30  11:06    <DIR>          ..
2024/08/30  11:09             5,712 DemoIDX.dat
               1 個のファイル                 5,712 バイト

C:¥Users¥tarot¥source¥repos¥FileHandlerTutorial¥FileHandlerNET¥bin¥Debug>
```

4) 実行結果の確認

① Windows のスタートメニューから[クラシックデータファイルツール] を選択します。

② [ファイル(F)] メニュー から [開く(O)] を選択し、「C:¥FileHandling¥NET¥DemoIDX.dat」をオープンします。

③ プログラムで指定したファイルが作成されていることを確認します。



3.5 クラシックデータファイルツール

データファイルツールは COBOL ファイル編成の変換、ファイルのブラウズ、レコードの編集等の機能を持った GUI ツールです。

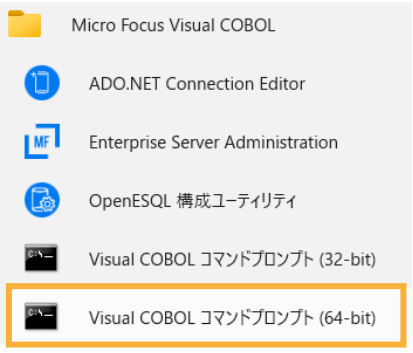
1) クラシックデータファイルツールの起動とファイルの読み込み

① 前の節で使用したクラシックデータファイルツールを終了した場合は、再度開き、以下のファイルをオープンします。

C:¥FileHandling¥NET¥DemoIDX.dat

2) デバッグ情報ファイルからレコードレイアウトを作成

① Windows のスタートメニューから [Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(64-bit)] を選択します。



② 「Data File Structure Command Line ユーティリティ」を使用してデバッグ情報ファイルからレコードレイアウトを作成します。下記のコマンドを実行してレコードレイアウトを生成します。

- dfstrcl "C:¥Users¥tarot¥source¥repos¥FileHandlerTutorial¥FileHandlerNET¥bin¥Debug¥FileHandlerNET.FILEDEMO1.idy" /d F-REC

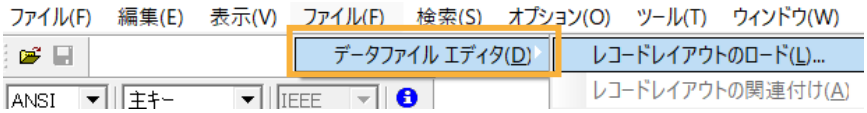
注意）
FILEDEMO1.idy へのパスは環境に合わせて変更してください。Visual Studio の FileHandlerNET プロジェクト¥bin¥Debug 配下にあります。

```
C:¥Users¥tarot¥Documents>dfstrcl "C:¥Users¥tarot¥source¥repos¥FileHandlerTutorial¥FileHandlerNET¥bin¥Debug¥FileHandlerNET.FILEDEMO1.idy" /d F-REC

Layout File Creation - DFSTRCL V20240423
Loading file - FileHandlerNET.FILEDEMO1.idy
Added default structure - F-REC
Structure file created - FileHandlerNET.FILEDEMO1.str
Processing complete

C:¥Users¥tarot¥Documents>
```

レコードレイアウトファイルは、FILEDEMO1.idy と同じフォルダーに作成されます。

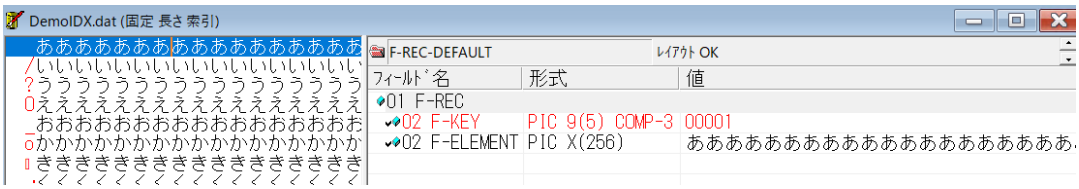
3) レコードレイアウトを適用し可視化できないデータを確認

① クラシックデータファイルツールに戻り、[ファイル(F)] メニュー > [データファイルエディタ(D)] > [レコードレイアウトのロード(L)…]を選択します。

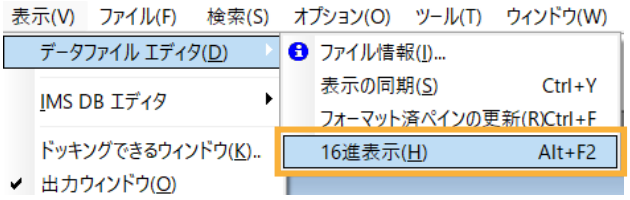


② 前のステップで作成した「FileDemo1.str」ファイルを指定します。

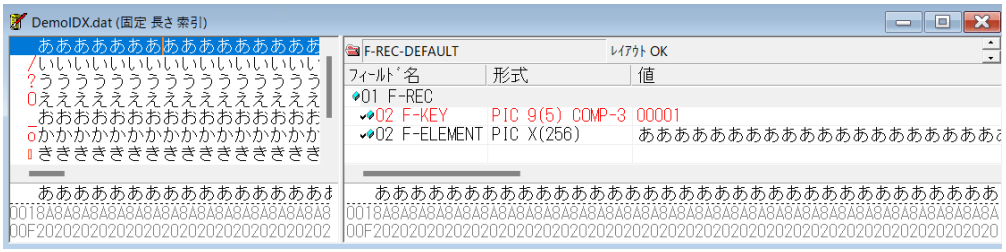
ファイルの中身がレコード単位で確認できるようになります。



③ [表示(V)] メニュー > [データファイルエディタ(D)] > [16 進表示(H)] を選択します。



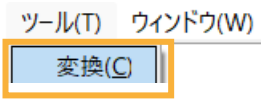
データの内容が 16 進でも表示されます。



4) 他のファイル編成に変換

① DemoIDX.dat ァイルを開いているウィンドウを一旦閉じます。

② [ツール(T)] メニュー > [変換(C)] を選択します。



③ 以下の入力を行い、[変換(C)] をクリックします。

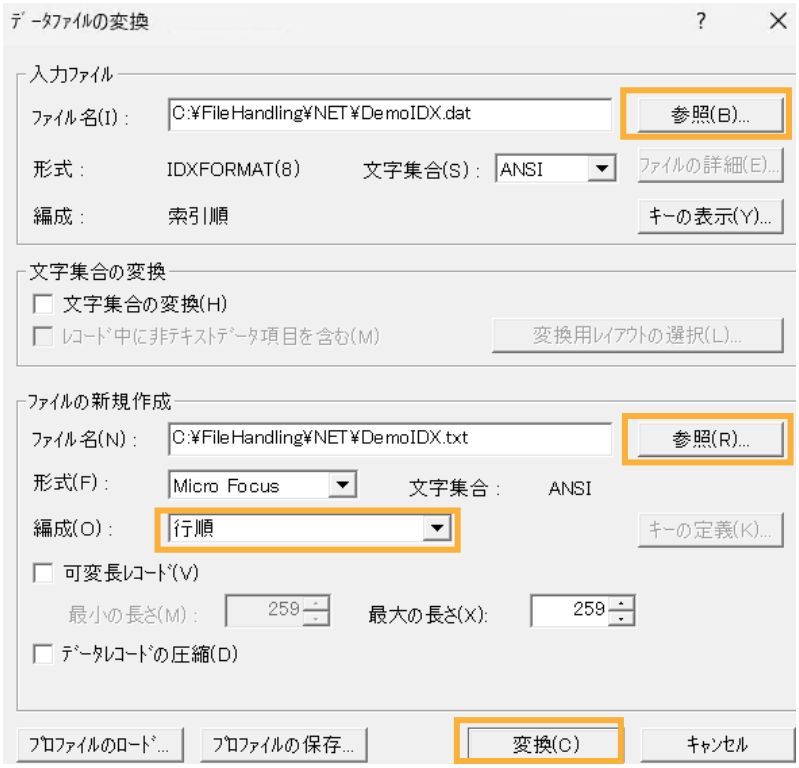
- [入力ファイル] > [ファイル名]

  ファイル名： [参照(B)] をクリックし、「C:¥FileHandling¥NET¥DemoIDX.dat」を選択

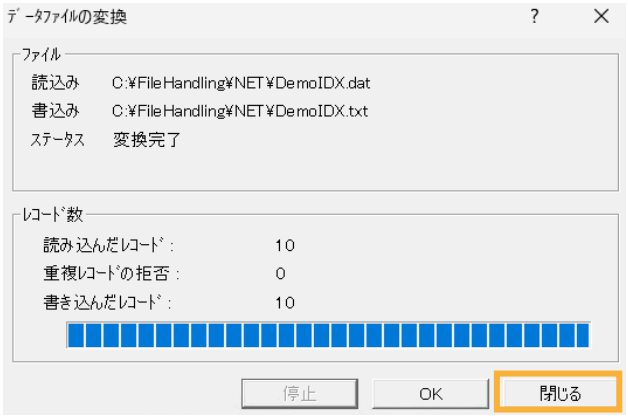
- [ファイルの新規作成]

  ファイル名： [参照(B)] をクリックし、「C:¥FileHandling¥NET¥DemoIDX.txt」を選択

  編成：「行順」を選択



④ 「データファイルの変換」ウィンドウが表示され、変換処理が完了した後［閉じる］をクリックします。



⑤ メモ帳で作成したファイルを開きます。各レコードが改行で区切られた行順ファイルになっていることを確認します。

補足）

メモ帳で開く場合は、エンコードに “ANSI” を選択してください。

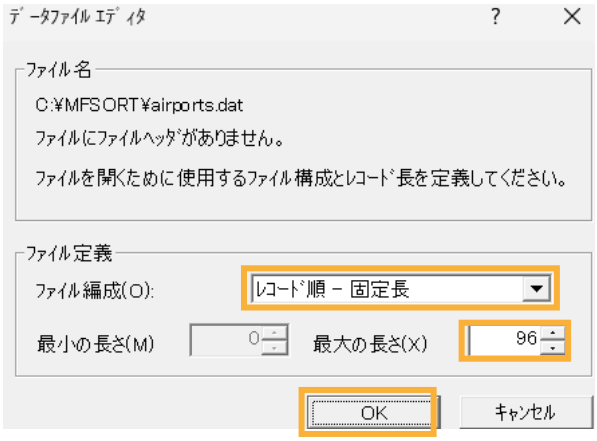
3.6 ソートユーティリティ MFSOFT の確認

ソートユーティリティ MFSORT は、IBM メインフレームの DFSORT とコマンド互換のあるソート・マージユーティリティです。

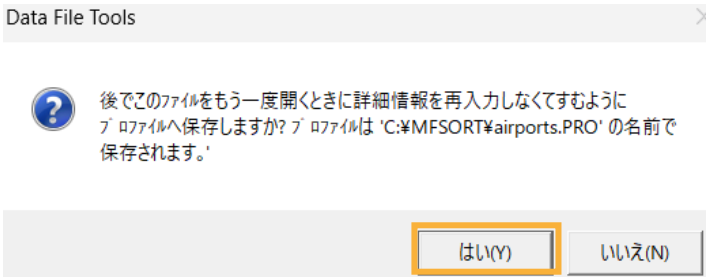
1) クラシックデータファイルツールの起動と読み込み

① クラシックデータファイルツールを起動して C:¥MFSORT フォルダー配下の「airports.dat」を開きます。

② [ファイル編成(O)] に “レコード順 – 固定長” が選択されていることを確認し、[最大の長さ(X)] に “96” を入力したうえで、[OK] をクリックします。

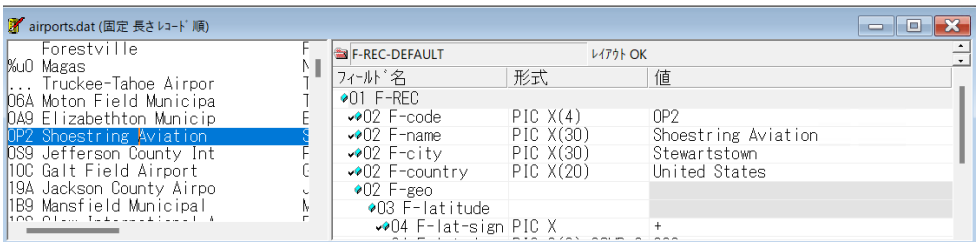


以下のようなダイアログが表示された場合は、[OK] をクリックします。



③ [ファイル(F)] > [データツールエディタ(D)] > [レコードレイアウトのロード(L)…] を選択し、C:¥MFSORT 内にある「AIRPORTSEQ.STR」ファイルを指定します。

ファイルの中身がレコード単位で確認できるようになります。



2) ソート順の変更

このデータファイルは「F-code」と呼ばれる空港コードの昇順でソートされています。これを「F-name」という空港名でソートします。

① ソート用のコマンドファイルを用意します。

１行目はインプットするデータファイルの指定、２行目はアウトプットするデータファイルの指定、３行目がソートするフィールド名の指定です。ここでは 「F-name」 を昇順で指定するので 5 バイト目から 30 バイト、昇順（ascending）に並べ替えを指定しています。

```
type airport.srt
use airports.dat record (f, 96)
give sorted.dat ORG SQ
sort fields (5, 30 CH, a)
```

補足）

上記ファイルは、MFSORT フォルダー配下内の airport.srt ファイルとして保存されています。

② Windows のスタートメニューから [Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(64-bit)] を選択します。

③ 「CD C:¥MFSORT」を実行し、C:¥MFSORT フォルダーまで移動します。

④ 以下のように、mfsort コマンドを実行します。

mfsort take airport.srt

```
C:¥MFSORT>mfsort take airport.srt

C:¥MFSORT>
```

注意）

クラシックデータファイルツールなどで airports.dat を開いている場合、上記コマンドを実行する前に閉じてください。

前の手順で airports.dat をクラシックデータファイルツールで開いている場合は、以下のようなエラーが報告されます。

C:¥MFSORT>mfsort take airport.srt

SORT013U: IO エラーのデータセット: 'airports.dat'

SORT014U: ステータス = 9/065

SORT020U: SORT(EXTSM) に失敗しました - ソートエンジンステータス = 9/065

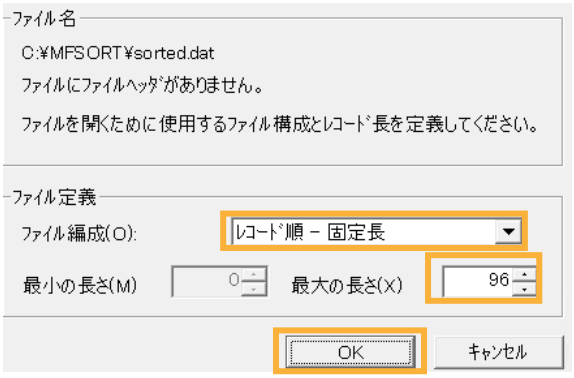
C:¥MFSORT>

3) 実行結果の確認

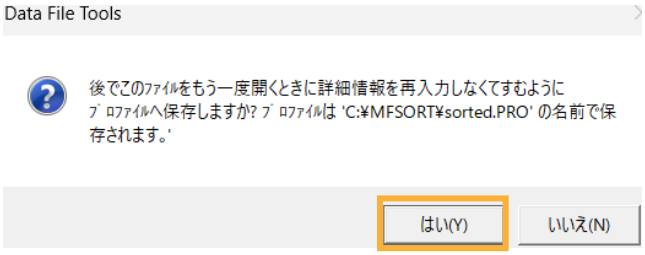
① 「SYSOUT」というファイルに実行結果ログが入っているのでメモ帳を起動して内容を確認します。

② クラシックデータファイルツールを起動して「sorted.dat」を開きます。

さきほど同様、[ファイル編成(O)] に “レコード順 – 固定長” を選択、[最大の長さ(X)] に “96” を指定して [OK] をクリックします。

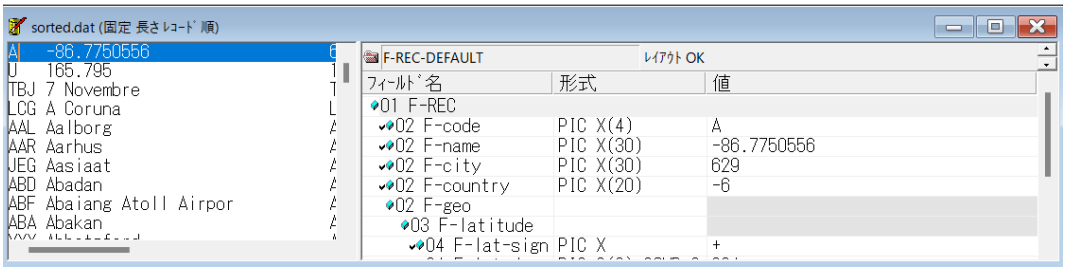


以下のダイアログが表示された場合は、[はい(Y)] をクリックします。



③ [ファイル(F)]メニュー ＞ [データファイルエディタ(D)] ＞ [レコードレイアウトのロード(L)…] を選択し、C:¥MFSORT 内にある「AIRPORTSEQ.STR」ファイルを指定します。

④ 空港名でソートされていることを確認します。



3.7　ファイル管理ユーティリティ REBUILD の確認

ファイル管理ユーティリティ　REBUILD は、COBOL ファイル編成の変換、索引ファイルの索引再編成、破損した索引ファイルのリビルド機能を提供するコマンドラインのユーティリティです。本チュートリアルでは索引ファイルの再編成を行います。

1)　rebuild コマンドによるインデックスの再編

①　Windows のスタートメニューから [Micro Focus Visual COBOL] > [Visual COBOL コマンドプロンプト(64-bit)] を選択します。

②　「CD C:¥REBUILD」を実行し、C:¥REBUILD フォルダーまで移動します。

③　rebuild コマンドを実行します。

- rebuild airportsIdxOld.dat,airportsIdxNew.dat

```
C:¥REBUILD>rebuild  airportsIdxOld.dat,airportsIdxNew.dat

airportsIdxOld.dat,airportsIdxNew.dat
再構成が完了しました - レコード読み込み =       5383

C:¥REBUILD>dir
 ドライブ C のボリューム ラベルがありません。
```

airportsIdxOld.dat は、レコードの追加、更新、削除を繰り返し実施したことによるファイルサイズが肥大化していました。しかし、rebuild コマンドによって再編された airportsIdxNew.dat はこれらの不要領域が削除され、ファイルサイズが小さくなっています。

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。