Micro Focus Visual COBOL for Eclipse

チュートリアル





はじめに

Micro Focus Visual COBOL for Eclipse は、高機能なオープンソースの IDE(統合開発環境)として広く普及する Eclipse 上で COBOL のアプリケーション開発を行うための製品です。
COBOL プログラマが既存の COBOL 資産を Windows、UNIX/Linux といったオープン環境で活用するだけでなく、COBOL プログラミング経験のないプログラマが初めて COBOL アプリケーション開発を行う場合に最適な製品です。

本書は、Micro Focus Visual COBOL for Eclipse(Windows)を学ぶためのチュートリアルです。 本書の読者は、プログラミングの基礎知識があり、かつ Windows の基本操作を理解しているものとします。

また、本書に掲載している画面イメージは Windows 10 Pro 64 bit 版でキャプチャしています。他の Windows OS では多少異なる場合がありますが、ご了承ください。



第1章 環境のセットアップ

Micro Focus Visual COBOL for Eclipse は、COBOL プログラミングの IDE として Eclipse を利用します。本製品には、Eclipse 4.7(64 bit)がバンドルされていますが、既に 32bit 版 Eclipse 4.6 もしくは 64 bit 版 Eclipse 4.6 をお使いの場合は、お使いの Eclipse にプラグイン形式で本製品をインストールすることも可能です。本章ではインストーラーを使った方法を紹介いたします。

- 1 入手したインストールプログラム vce_40.exe をダブルクリックします。
- 2 表示されるセットアップ画面で エンドユーザ使用許諾契約書 をクリックします。





3 エンドユーザー使用許諾契約書が表示されます。

ブラウザが起動し、エンドユーザー使用許諾契約書の内容が表示されますので内容を確認しま す。

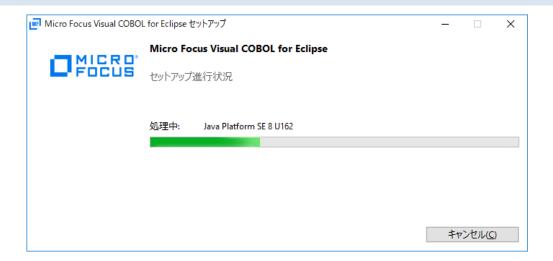
4 インストールを開始します。

表示された内容に問題なければ、**同意する(A)** にチェックを入れ [**インストール(I)**] ボタンを クリックしてインストールを開始します。





5 セットアップが開始されます。



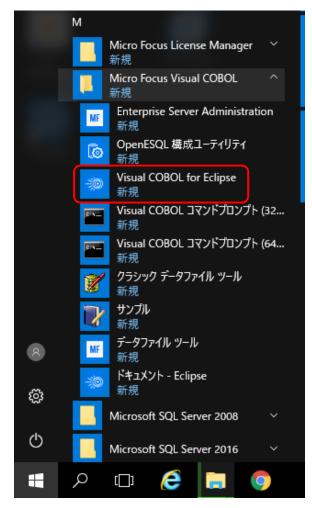
インストールの処理が完了したら、[閉じる(C)] ボタンをクリックします。





以上で、チュートリアルの準備は終了しました。 Windows のスタートメニューに「Micro Focus Visual COBOL」が登録され、その配下に **Visual COBOL for Eclipse** が追加されていることを確認

してください。





第2章 Eclipse IDE に慣れよう

Eclipse の IDE を初めて利用する COBOL プログラマのために、概要を簡単に説明します。 既に Eclipse に習熟されている方は、本章を読み飛ばしてください。

Eclipse の IDE を使う場合、ビュー、エディター、ツールバー、メニューバー、ステータスバー、パースペクティブから構成されるワークベンチというウィンドウ内で作業します。パースペクティブは、行いたい作業によって切り替えて利用するワークベンチのレイアウト(表示するビュー、メニュ

ー、ツールバーの種 類や場所)のことを 指します。パースペ クティブを切り替え ることによって目的 の作業に応じた構成 要素を揃えることが できます。各要素の 配置はカスタマイズ 可能です。



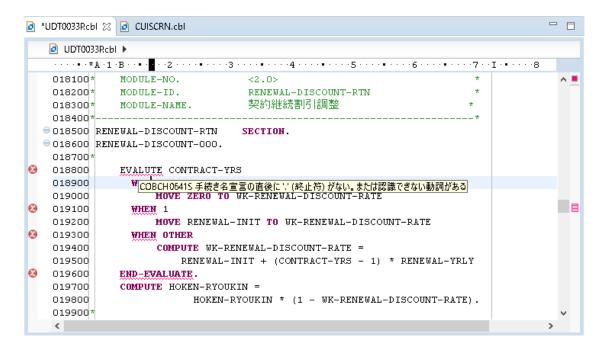
Eclipse のワークスペースとプロジェクトには、アプリケーションの作成に必要なビルドパス、データ接続、フォルダー、およびファイルを表す項目等が含まれています。 ワークスペースには複数のプロジェクトを含めることができ、プロジェクトには、通常、複数の項目が含まれます。COBOL エクスプローラーには、ワークスペースに紐づけられたプロジェクト、それらのプロジェクト内の項目が階層状に表示されます。COBOL エクスプローラー上で目的の要素を検索し、編集するファイルを開く、プロジェクトに新規ファイルを追加する、プロジェクトおよび項目のプロパティを表示するなどの操作を

実行できます。パースペクティブによっては異なる名称のビューが同等の用途のために用意されています。例えば Java パースペクティブであればパッケージエクスプローラーという Java 開発時に必要な要素をフィルター表示するビューが紐つけられています。

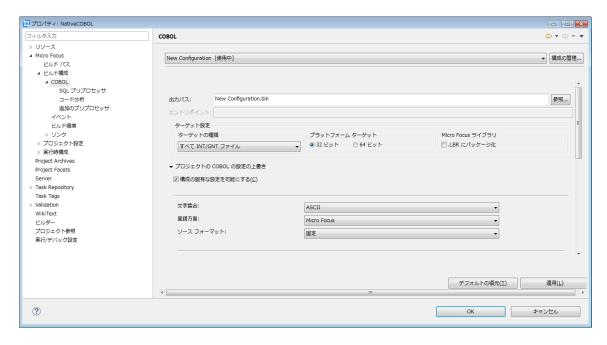




COBOL パースペクティブに紐つくエディターには、COBOL 予約語とデータ名や手続き名などの利用者語を色分け表示したり、COBOL スニペットなど COBOL 言語固有の機能拡張が含まれます。ソースコードを入力するとバックグラウンドチェックを実行して、赤の波線でエラー箇所を強調表示します。 そのエラー箇所にマウスポインタを移動すればエラー内容を確認したり、定義への移動、他の参照検索などの操作が可能です。



プロジェクトを右クリックし [プロパティー(R)] \rightarrow [Micro Focus COBOL] と遷移し表示される 各画面では COBOL アプリケーションのビルド方法等を構成します。





コンソールビューにはビルド時のメッセージやアプリケーションのコンソール出力等が表示されます。問題ビューには、不正な構文、キーワードのスペルミス、型の不一致などのコンパイルエラーが表示されます。

```
Micro Focus 以下: InsuEstimOnline

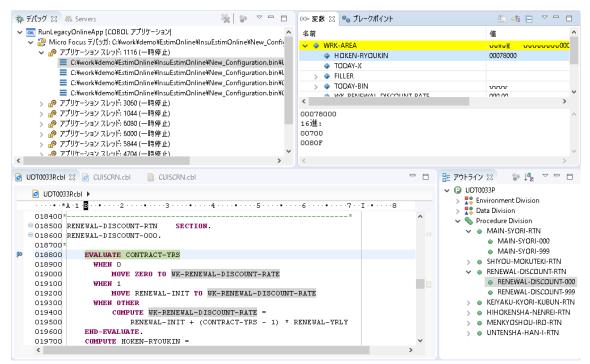
[echo] Project Location: 'pathVar.ECLIPSE_HOME'=C:/Users/Public/Micro Focus/Visual COBOL/eclipse
[echo] Project Location: 'pathVar.PARENT_LOC'=C:/work/demo/EstimOnline

pre.build.cfg.New_Configuration:

[cobol]
[cobol] Compiling UDT0033P.cbl...
[cobol] * 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
[cobol] Compilation complete with no errors

cobol.link.cfg.New_Configuration:
[cobol] Linking UDT0033P.dll...
[cobollink] Linking UDT0033P.dll...
[cobollink] Micro Focus COBOL - CBLLINK utility
```

ビルドしたアプリケーションは、実行時の論理エラーやセマンティックエラーなどの問題を検出して修正するために、デバッグ機能を利用します。 Eclipse のデバッガーは、コードをステップ実行したり様々な条件を設定したブレークポイントで実行を中断させ、変数ビューを使用してローカル変数やその他の関連データを調べることができます。



デバッグが完了したアプリケーションは、アプリケーションサーバ等との連携機能を利用して自動 配備するか、ファイルを手動でコピーして、本番環境に配置します。

なお、本番環境には COBOL Server が事前にインストールされている必要があります。



第3章 はじめての Visual COBOL

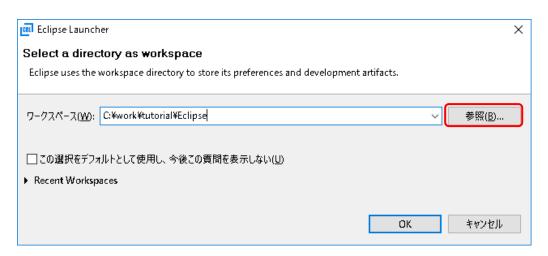
それでは、Visual COBOL for Eclipse を使って、Windows のコマンドプロンプト画面に「Hello World」を表示する COBOL アプリケーションを作成してみましょう。

1 Visual COBOL for Eclipse を起動します。

Windows のスタートメニューから、Visual COBOL for Eclipse をクリックします。

2 ワークスペースの保存先を選択します。

ワークスペースを保存する任意のフォルダを指定します。[**参照(B)**] ボタンをクリックしエクスプローラー経由で選択することも可能です。

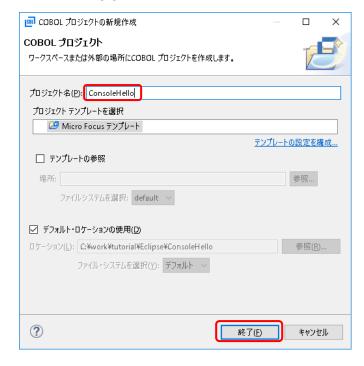


3 COBOL プロジェクトを作成します。

ファイル(F)メニューから 新規(N) \rightarrow COBOL プロジェクト を選択します。



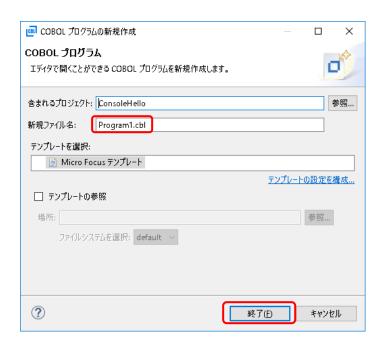
プロジェクト名欄に ConsoleHello と入力し [終了(F)] ボタンをクリックします。



4 COBOL プログラムを追加します。

COBOL エクスプローラービューにて プロジェクト フォルダを右クリックし 新規作成(N) → COBOL プログラムを選択します。

ファイル名はデフォルトの Program1.cbl を使用します。[終了(F)] ボタンをクリックします。

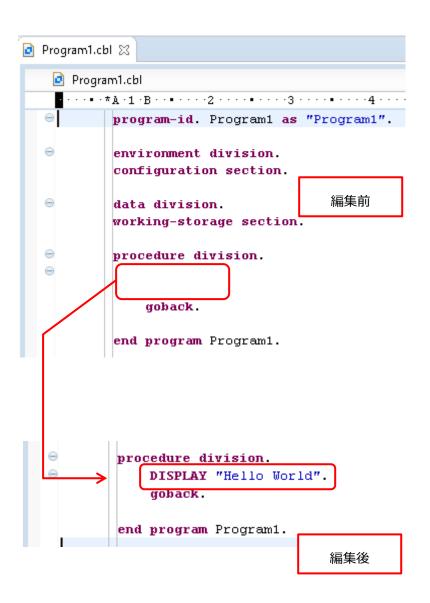




5 エディターで COBOL ソースコードを入力します。

COBOL プロジェクトにて COBOL プログラムを新規に作成するとプログラムを構成する見出し部 (identification division)、環境部(environment division)、データ部(data division)、手続き部 (procedure division) や program-id 段落が埋め込まれたかたちで生成されます。今回は「Hello World」を表示して終了するプログラムなので、手続き部 3 行目の goback 文の手前に以下のように display 文を加えます。

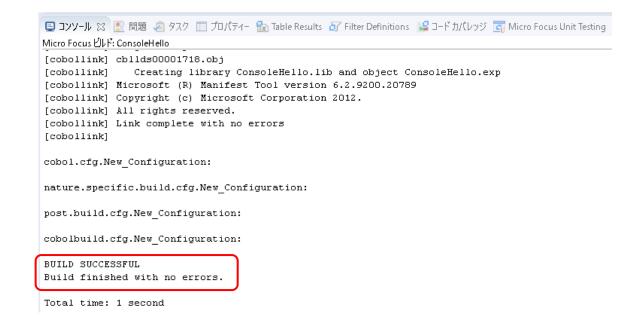
なお、COBOL 正書法ではエディタービュー左右にある線で区切られた領域を特別な領域として利用するので、通常のソースコードはこれを避けて入力します。





6 COBOL アプリケーションをビルドします。

終止符(ピリオド)を含めてスペルミスがなければ、エディタービュー上の Program1.cbl をアクティブにしたまた**ファイル(F)** メニューの**保管(S)** 或いは **Ctrl + S** キーで保存します。これにより自動的にコンパイラがキックされビルド処理が始まります。コンソールビューに正常にビルドできた旨のメッセージが出力されていることを確認します。



√√メモ:

Visual COBOL for Eclipse ではデフォルトでこのような自動ビルド機能が有効となっています。自動ビルド機能を無効にし、任意のタイミングでビルドしたい場合は

プロジェクトメニュー > 自動的にビルド

についたチェックを外します。ビルドする際はプロジェクトメニュー配下の「すべてビルド」或いは 「プロジェクトをビルド」を目的に応じて選択します。



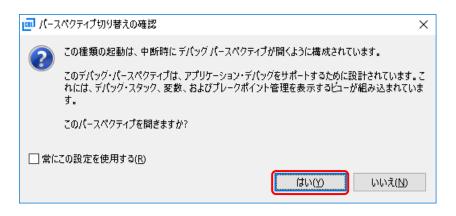
7 COBOL アプリケーションをデバッグ実行します。

COBOL エクスプローラー上の Program1.cbl を右クリックし

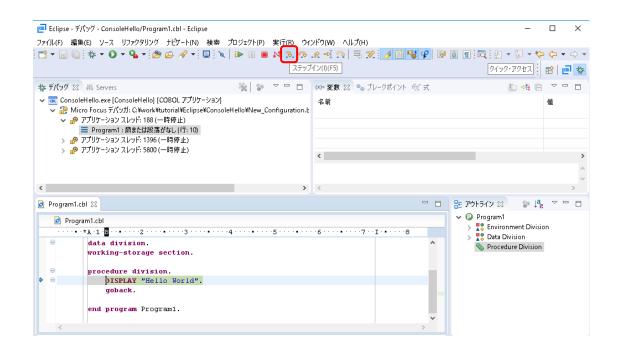
デバッグ(D) → COBOL アプリケーション

を選択します。

パースペクティブ切り替えに関する確認メッセージには [はい(Y)] を選択します。

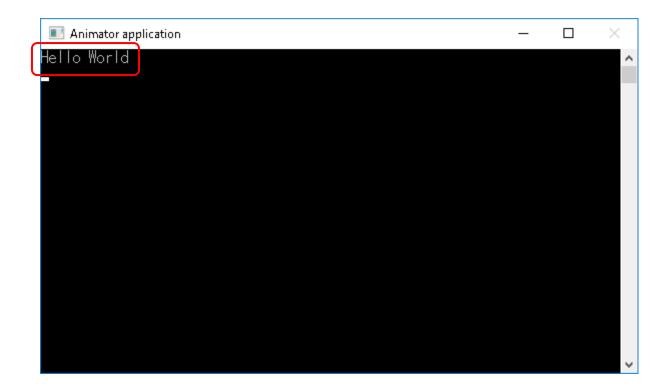


デバッグパースペクティブに切り替わりましたら、DISPLAY 文を実行する手前でステップが一時停止していますので、[ステップイン] アイコンを一回クリックし DISPLAY 文を実行します。





Animator application 画面に「Hello World」が表示されたことを確認できましたら、[**ステップイン**] アイコンを再度クリックしデバッグを終了します。





第4章 COBOL JVM アプリケーションの作成

本章では、Visual COBOL for Eclipse の COBOL JVM 機能を利用して Java バイトコードにコンパイルされた COBOL アプリケーションを作成する方法を紹介します。ここでは、Java プログラムから「Hello World」を表示する COBOL プログラム呼び出すアプリケーションを作成します。

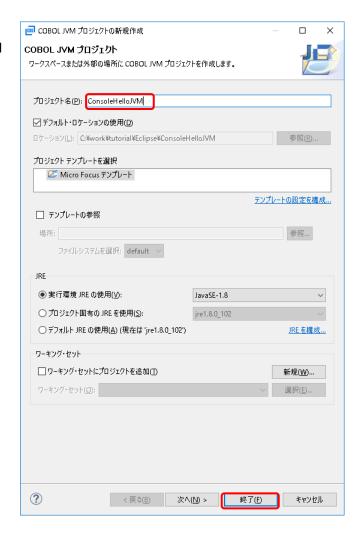
1 Visual COBOL for Eclipse を起動します。

Windows のスタートメニューから、**Visual COBOL for Eclipse** をクリックし、第3章で使用したワークスペースを選択します。第3章から続けて実施される場合は、このステップはスキップしワークスペースを継続して使用します。

2 COBOL JVM プロジェクトを作成します。

ファイル(F)メニューから 新規(N) \rightarrow COBOL JVM プロジェクト を選択します。

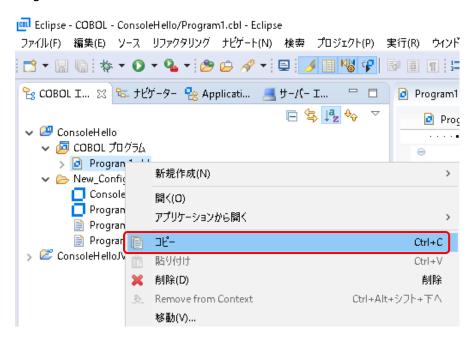
プロジェクト名欄に **ConsoleHelloJVM** と入力し [**終了(F)**] ボタンをクリックします。





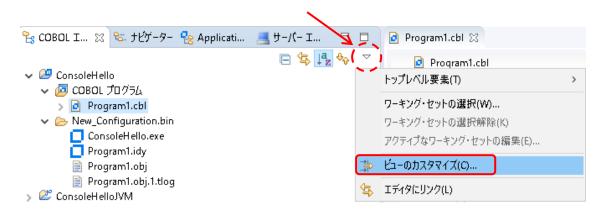
3 Native 用のプロジェクトで使用したプログラムをコピーします。

COBOL エクスプローラービューにて **ConsoleHello** プロジェクト配下の COBOL プログラムフォルダに前章で追加した Program1.cbl を右クリックし **コピー** を選択します。



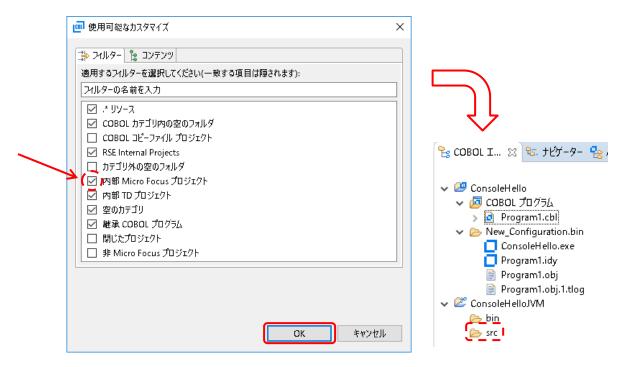
4 COBOL エクスプローラーのフィルターをカスタマイズします。

COBOL エクスプローラービュー中の右上のアイコンをクリックし、**ビューのカスタマイズ** を選択します。





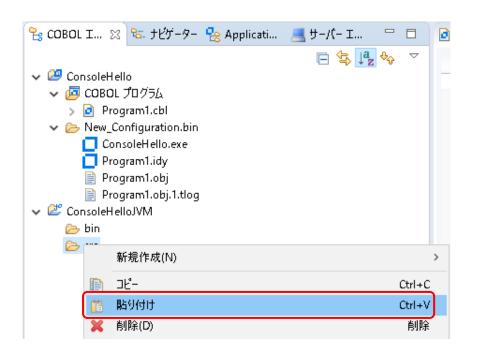
カテゴリ外の空のフォルダ のチェックを外し、OK ボタンをクリックします。



COBOL for JVM プロジェクト配下に src フォルダが表示されます。

5 COBOL for JVM のプロジェクトにコピーしたプログラムを追加します。

COBOL for JVM のプロジェクト配下にある src フォルダを右クリックし **貼り付け**を選択します。





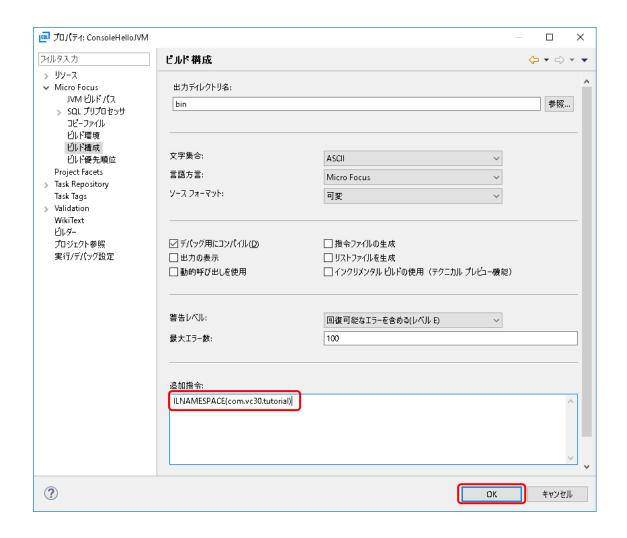
6 COBOL から生成された JVM クラスをパッケージ化します。

COBOL エクスプローラービューにて COBOL for JVM のプロジェクトを右クリックし、[プロパティ(R)] を選択します。

[Micro Focus] > [ビルド構成]

へとナビゲートし展開される画面中の [追加指令] 欄に

ILNAMESPACE(com.vc30.tutorial) を記入し、[OK] ボタンをクリックします。





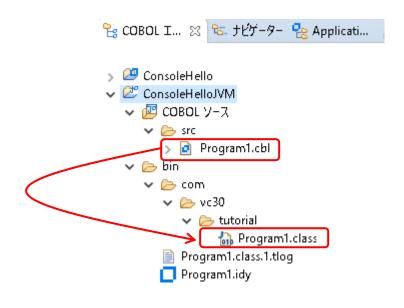
7 COBOL JVM Class ファイルをコンパイルします。

自動的にビルドが有効になっていれば、前ステップでプログラムを追加した時点でビルド処理が走ります。

コンソールビューで正常に処理できた旨のメッセージを確認できます。



また、COBOL エクスプローラービューでは、追加したプログラムが .class の拡張子を持った JVM クラスとしてビルドされていることを確認できます。プログラム中にはパッケージ化する旨の記述は加えていませんが先のステップで指定したコンパイラ指令により、パッケージ化された JVM クラスとして生成されています

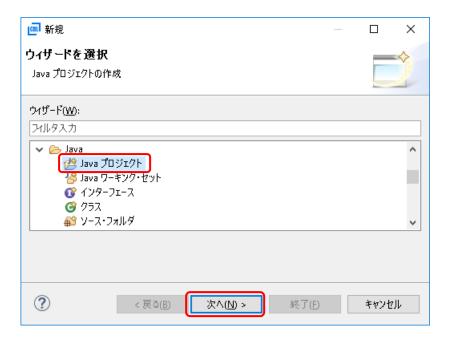




8 Java プロジェクトを作成します。

ファイル(F)メニューから 新規(N) → その他 を選択します。

Java 配下の Java プロジェクトを選択し [次へ(N)] ボタンをクリックします。



プロジェクト名欄に

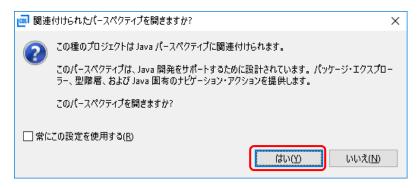
CallCobHello

と入力し [**終了(F)**] ボタンをクリックします。





Java パースペクティブに関連付けられる旨のメッセージには [はい(Y)] を選択します。



9 Java クラスファイルを追加します。

パッケージエクスプローラービューにて **CallCobHello** プロジェクト配下の src フォルダを右ク リックし

新規(W) → クラスを選択します。

新規 Java クラスウィンドウでは

パッケージ欄 com.vc30.tutorial

名前欄 CallCobHello

のように入力します。また、「public static void main(String[] args)」にチェックを入れ、[終**了**

(F)] ボタンをクリックします。





10 COBOL JVM プロジェクト及び Visual COBOL のランタイムをビルドパスに追加します。

パッケージエクスプローラービューにて CallCobHello プロジェクトを右クリックし [プロパティ(R)] を選択します。

[Java のビルド・パス]ページの [ライブラリー] タブを選択します。 [クラスフォルダの追加(C)] ボタンをクリックします。

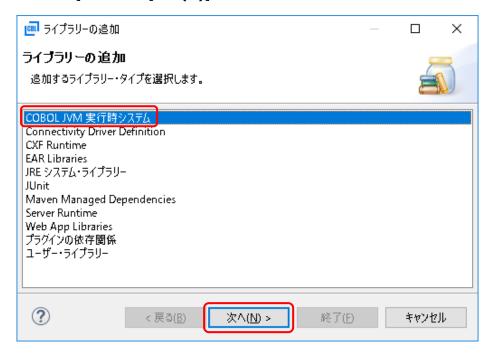
ConsoleHelloJVM プロジェクト配下の [bin] フォルダにチェックを入れ、[**OK**] ボタンをクリックします。





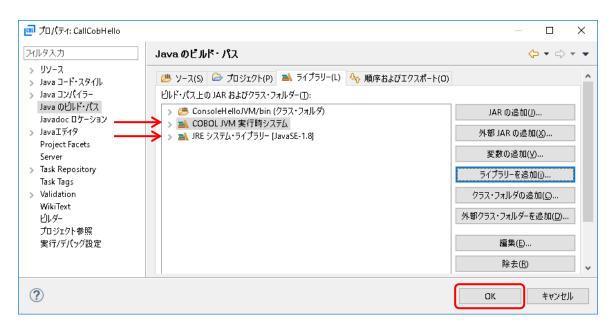
続いて、[**ライブラリーを追加(i)**] ボタンをクリックします。

[COBOL JVM 実行時システム] を選択し [次へ(N)] ボタンをクリックします。



[終了(F)] ボタンをクリックします。

上で追加したライブラリーが画面に反映されていることを確認し、[**OK**] ボタンをクリックしプロパティ画面を閉じます。





11 エディターで Java ソースコードを入力します。

main メソッドに以下のコードを追記します。

```
Program1 cobclass = new Program1();
long res = cobclass.Program1();
```

```
package com.vc30.tutorial;

public class CallCobHello {

public static void main(String[] args) {

Program1 cobclass = new Program1();

long res = cobclass.Program1();

}
```

₩XE:

従来の手続き型の COBOL プログラムを JVM クラスにコンパイルすると、プログラム名がクラス名となり、PROCEDURE DIVISION 以下で記載された命令は、プログラムと同名のインスタンスメソッドとして実装されます。生成された JVM クラスは Java から生成される JVM クラスと同様に扱えるため、Java 中で特別なロジックを記述することなく COBOL の呼び出し命令を記述できます。Eclipse 上の Java のエディタも COBOL から生成されたクラスのシンボルを Java と同様に認識できるため、Java エディタ上で COBOL プログラムに対してコードアシストの機能等を利用することができます。コードアシストは Ctrl + Space で起動できます。

```
public static void main(String[] args) {
    Program1 cobclass = new Program1();
    long res = cobclass.P

    Program1():long -Program1
}

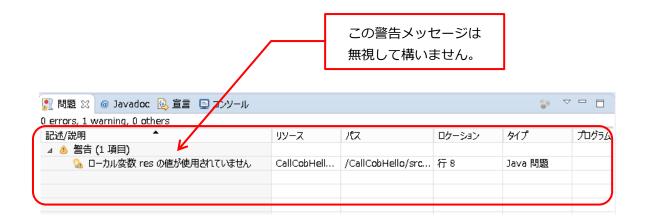
コードアシストの利用
イメージ

'Ctrl+スペース'の押下でテンプレート・プロポーザルを表示
```



12 COBOL JVM Class ファイルをコンパイルします。

スペルミスがなければ、エディタービュー上の CallCobHello.java をアクティブにしたまた**ファイル(F)** メニューの**保管(S)** 或いは Ctrl + S キーで保存します。問題ビューに何もエラーが出力されていないことを確認します。

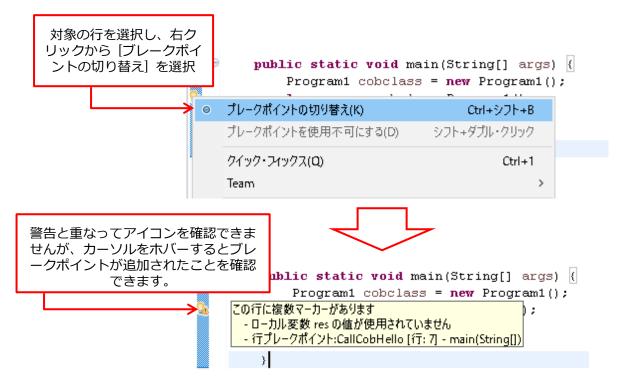


13 アプリケーションをデバッグ実行します。

COBOL プログラム中のメソッド呼び出す

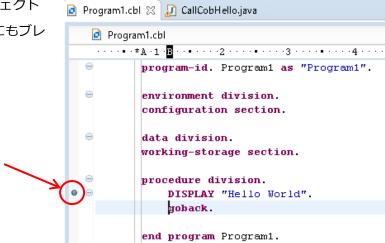
long res = cobclass.Program1();

にブレークポイントを設定します。





同様に ConsoleHelloJVM プロジェクト 中の Program1.cbl の DISPLAY 文にもブレ ークポイントを指定します。



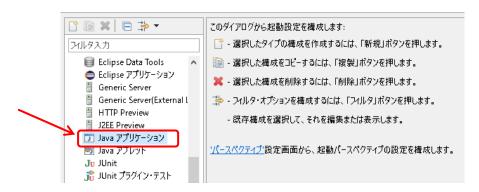
CallCobHello > src > com.vc30.tutorial 配下の **CallCobHello.java** を右クリックし デバッグ(**D**) → デバッグの構成(**B**)を選択します。

Java **アプリケーション** を ダブルクリックします。

■ デバッグ構成

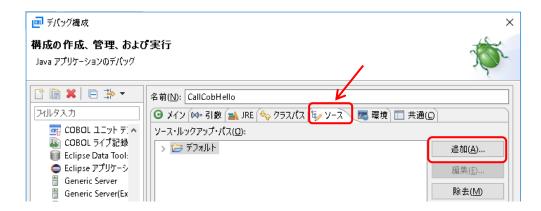
構成の作成、管理、および実行

Java アプリケーションのデバッグ



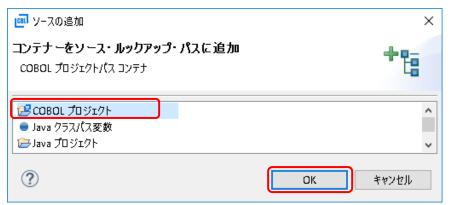
[**ソース**] タブをクリックします。

「追加(A)」ボタンをクリックします。





[COBOL プロジェクト] を選択の上、[OK] ボタンをクリックします。

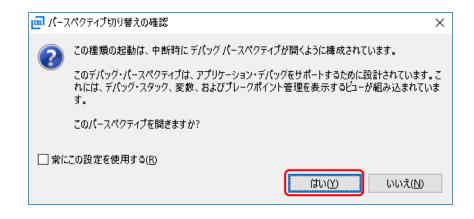


COBOL for JVM のプロジェクト
[ConsoleHelloJVM] にチェックを入れ、[OK] ボタンをクリックします。



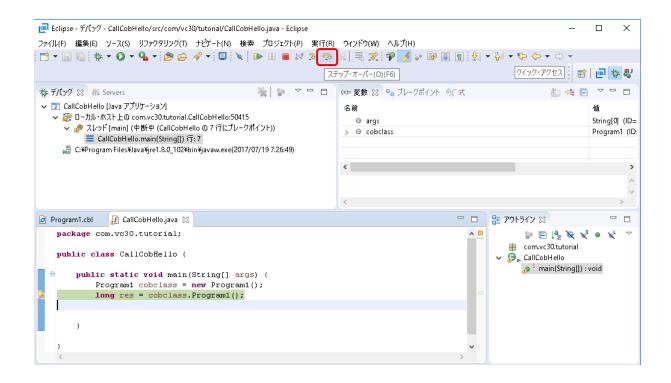
[デバッグ(D)] ボタンをクリックします。

パースペクティブ切り替えの確認メッセージには [はい(Y)] を選択します。

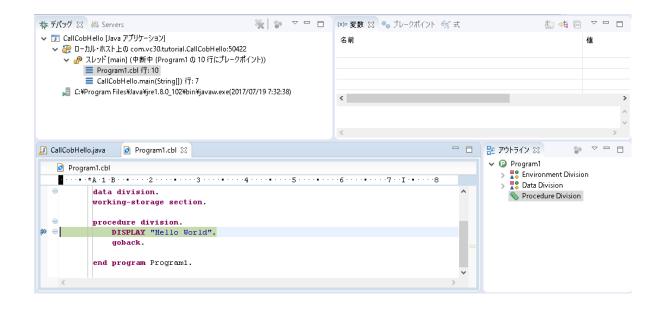




ステップインアイコンをクリックします。ステップ実行には、ステップイン、ステップオーバとメニューが用意されています。ステップインを選択した場合は Call 先のメソッドの中までステップ実行し、ステップオーバーを選択した場合は、メソッドの中にステップを進めずメソッドを実行します。ここでは、ステップオーバーを選択し COBOL プログラム中に指定したブレークポイントまでステップを進めます。

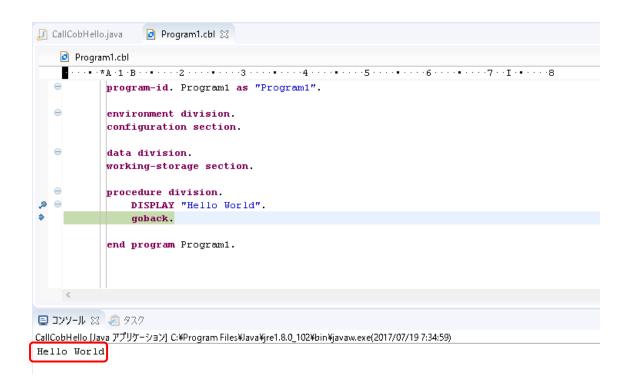


COBOL アプリケーションのソースの中にステップが進んでいることが確認できます。





再度、ステップインアイコンをクリックし、DISPLAY 文を実行するとコンソールに「Hello World」が表示されることが確認できます。



確認できましたら、アプリケーションが終了するまでステップを進めデバッグを終了させます。



第5章 Visual COBOL のファイル入出力

次に、エクセルやメモ帳で作成した CSV ファイルを読み込んで、固定長順編成ファイルを作成する COBOL アプリケーションを Visual COBOL for Eclipse で作成しましょう。

1 Visual COBOL for Eclipse を起動します。

Windows のスタートメニューから、Visual COBOL for Eclipse をクリックします。

ワークスペースは第3章で用意したものをそのままご使用いただいても構いません。

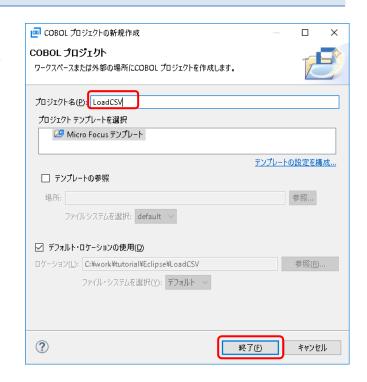
2 COBOL プロジェクトを作成します。

ファイル(F)メニューから 新規(N) \rightarrow COBOL プロジェクト を選択します。

√ メモ:

ファイル(F) メニュー \rightarrow 新規(N) にて候補として表示されるプロジェクトはワークスペース中でアクティブなパースペクティブにより切り替わります。例えば、Visual COBOL が提供するパースペクティブがアクティブな場合、COBOL プロジェクトは候補として表示されます。一方、他の Java のパースペクティブ等がアクティブな場合は候補に含まれないため、パースペクティブを COBOL に切り替えてから作業します。

プロジェクト名欄に **LoadCSV** と 入力し[**終了(F)**] ボタンをクリックしま す。

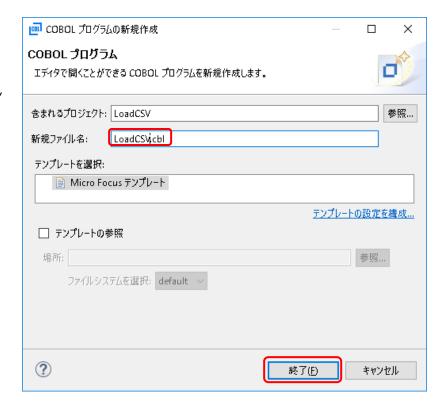




3 COBOL プログラムを追加します。

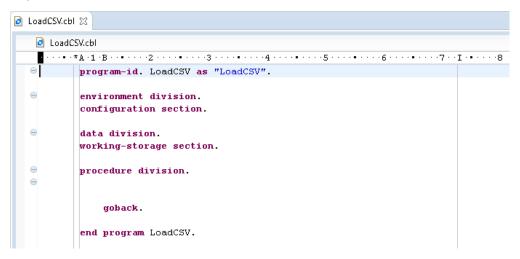
COBOL エクスプローラービューにて **LoadCSV プロジェクト** フォルダを右クリックし **新規(N) \rightarrow COBOL プログラム**を選択します。

新規ファイル名欄には **LoadCSV.cbl** を指定します。
[**終了(F)**] ボタンをクリックします。



4 コードエディターで COBOL ソースコードを入力します。

COBOL ソースファイルを新規に作成した直後の段階ではコンソールアプリケーションのひな形が埋め込まれています。ここでは、環境部(environment division)、データ部(data division)、手続き部 (procedure division) を書き換えます。





まず、環境部の構成節(configuration section)を削除し、以下の入出力節(input-output section)を追加します。 まだ、データ部のファイル定義が未入力なので IN-FILE と OUT-FILE がエラーとなりますが、ここでは無視して構いません。

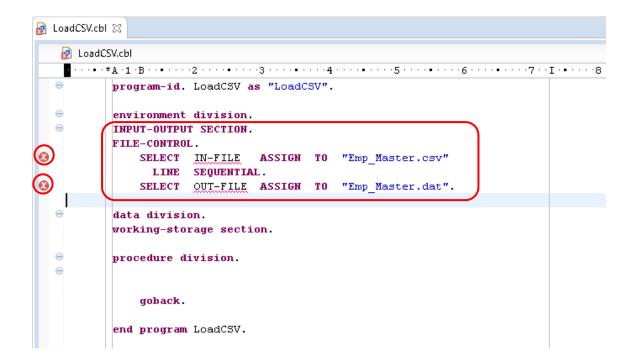
```
INPUT-OUTPUT SECTION.

FILE-CONTROL.

SELECT IN-FILE ASSIGN TO "Emp_Master.csv"

LINE SEQUENTIAL.

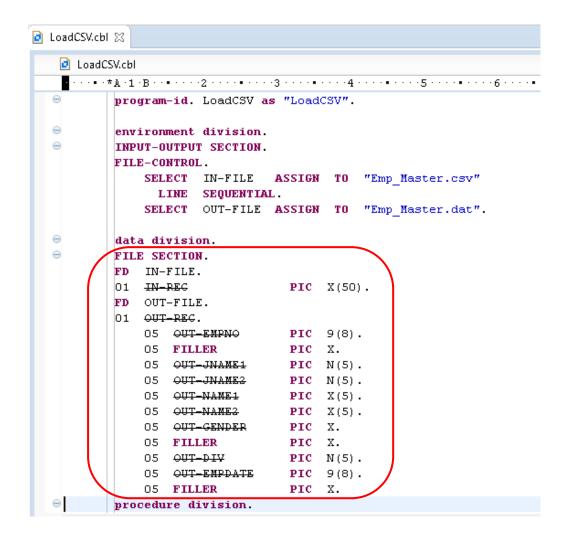
SELECT OUT-FILE ASSIGN TO "Emp_Master.dat".
```





次に、データ部の作業場所節(working-storage section)を削除し、以下のファイル節(FILE SECTION)を追加します。 なお、データ部のファイル定義を入力したので、環境部のエラーは無くなります。

```
FILE SECTION.
FD IN-FILE.
                     PIC X(50).
01 IN-REC
FD OUT-FILE.
01 OUT-REC.
   05 OUT-EMPNO
                      PIC 9(8).
   05 FILLER
                      PIC X.
   05 OUT-JNAME1
                      PIC N(5).
   05 OUT-JNAME2
                      PIC N(5).
   05 OUT-NAME1
                      PIC X(5).
   05 OUT-NAME2
                      PIC X(5).
                      PIC X.
   05 OUT-GENDER
   05 FILLER
                      PIC X.
   05 OUT-DIV
                      PIC N(5).
   05 OUT-EMPDATE
                      PIC 9(8).
   05 FILLER
                      PIC X.
```





最後に、手続き部の goback 文を削除し、以下の手続き文を追加します。

```
PROC1.
   OPEN INPUT IN-FILE.
    OPEN OUTPUT OUT-FILE.
PROC2.
   READ IN-FILE AT END GO TO PROC9.
    INITIALIZE OUT-REC.
   UNSTRING IN-REC DELIMITED BY "."
      INTO OUT-EMPNO
           OUT-JNAME1
           OUT-JNAME2
           OUT-NAME1
           OUT-NAME2
           OUT-GENDER
           OUT-DIV
           OUT-EMPDATE
   END-UNSTRING.
   WRITE OUT-REC.
   GO TO PROC2.
PROC9.
   CLOSE IN-FILE OUT-FILE.
   STOP RUN.
```

```
IcadCSV.cbl ⋈
   LoadCSV.cbl
   •···•*A·1·B··•···2···•3···•···4···•···5···•6···•7·:
          procedure division.
          PROC1.
              OPEN INPUT
                           IN-FILE.
              OPEN OUTPUT OUT-FILE.
          PROC2.
              READ IN-FILE AT END
                                   GO TO PROC9.
              INITIALIZE OUT-REC.
              UNSTRING IN-REC DELIMITED BY ","
                INTO OUT-EMPNO
                     OUT-JNAME1
                     OUT-JNAME2
                     OUT-NAME1
                     OUT-NAME2
                     OUT-GENDER
                     OUT-DIV
                     OUT-EMPDATE
              END-UNSTRING.
              WRITE OUT-REC.
              GO TO PROC2.
          PROC9.
              CLOSE IN-FILE OUT-FILE.
              STOP RUN.
          end program LoadCSV.
```



5 COBOL アプリケーションをビルドします。

終止符(ピリオド)を含めてスペルミスがなければ、エディタービュー上の LoadCSV.cbl をアクティブにしたまた**ファイル(F)** メニューの**保管(S)** 或いは Ctrl + S キーで保存します。コンソールビューに正常にビルドできた旨のメッセージが出力されていることを確認します。

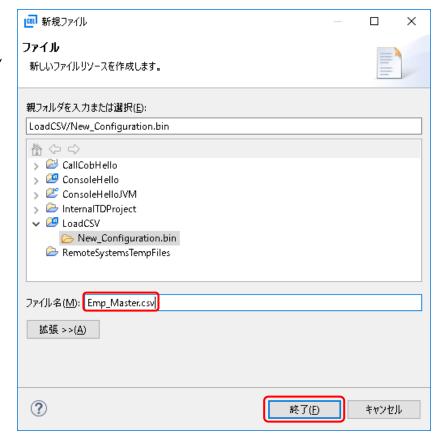


6 CSV ファイルを作成します。

LoadCSV プロジェクト配下の **New_Configuration.bin** フォルダを右クリックし **新規(N) → ファイル**を選択します。

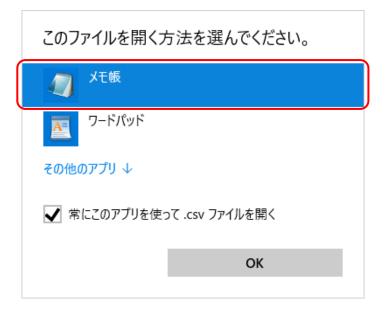
ファイル名欄に

Emp_Master.csv を指定し [**終了(F)**] ボタンをクリック します。





「この種類のファイル(.csv)を開くには、どのアプリを使いますか?」の問いがポップアップされた場合は「メモ帳」を選択します。



以下のデータを入力し、ファイルを保存の上、閉じます。

11111113,佐藤,隆,サトウ,タカシ,M,営業部,19980401,0 22222226,鈴木,尚之,スズキ,ナオュキ,M,技術部,19981015,0 33333339,田中,直美,タナカ,ナオミ,F,総務部,19990401,0 4444442,山田,洋一,ヤマダ,ヨウイチ,M,営業部,20000701,0 55555555,伊藤,弘子,イトウ,ヒロコ,F,技術部,20010401,0 66666668,木村,貴弘,キムラ,タカヒロ,M,営業部,20021220,0 77777771,中村,慎司,ナカムラ,シンジ,M,技術部,20030401,0 88888884,橋本,悦子,ハシモト,エツコ,F,総務部,20040805,0 99999997,三井,薫,ミツイ,カオル,F,営業部,20050401,0

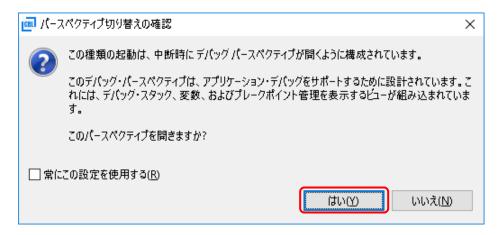


7 COBOL アプリケーションをデバッグ実行します。

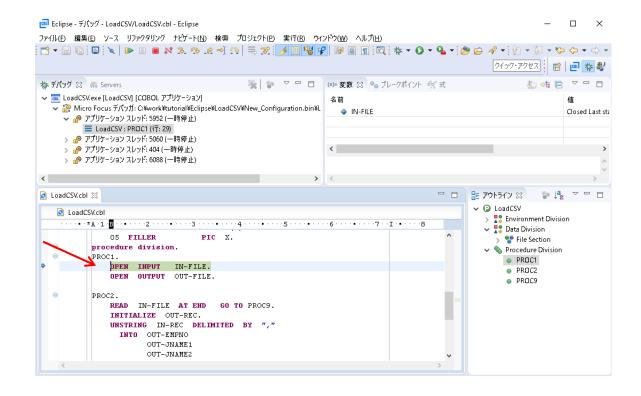
New_Configuration.bin フォルダ配下の LoadCSV.exe を右クリックし

デバッグ(D) → COBOL アプリケーションを選択します。

パースペクティブ切り替えの確認メッセージには [はい(Y)] を選択します。

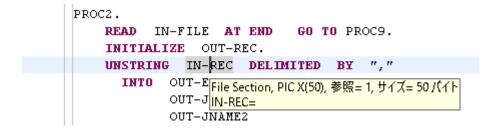


デバッガーは手続き部の最初の COBOL 文である open 文の処理前に一時停止している状態となっています。





入力ファイルから読み込んだレコードの内容を確認するため、IN-REC に格納される値の変遷を追います。UNSTRING文の IN-REC を選択します。

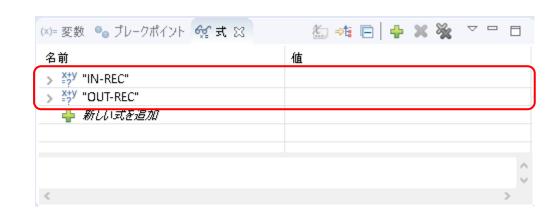


この状態で右クリックし、「項目を検査」を選択します。

「式ビューに追加」をクリックします。

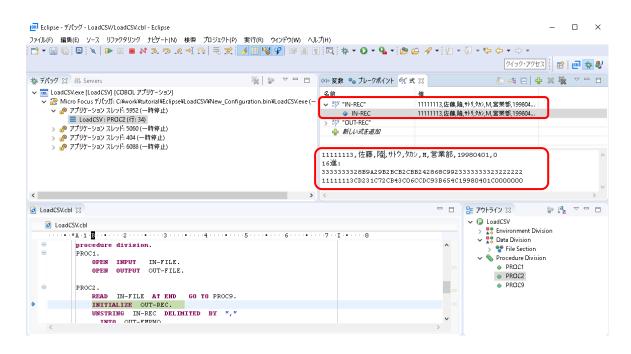


同様に出力ファイルに書き出すレコードの内容もトレースするため、INITIALIZE 文の **OUT-REC** を同様に式ビューに追加します。

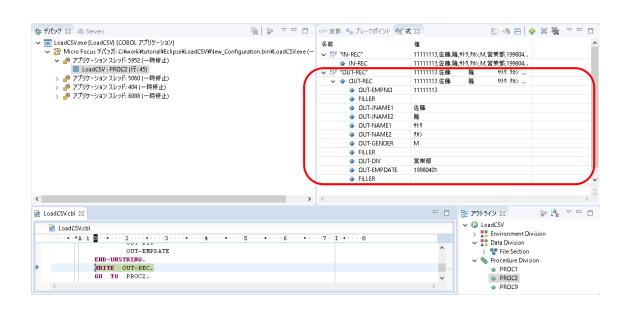




F5 キー(ステップイン)を 3 回押すと、デバッガーは READ 文実行後、処理を中断します。 式ビューの IN-REC の値には CSV ファイルから読み込んだ最初のレコードが表示されます。



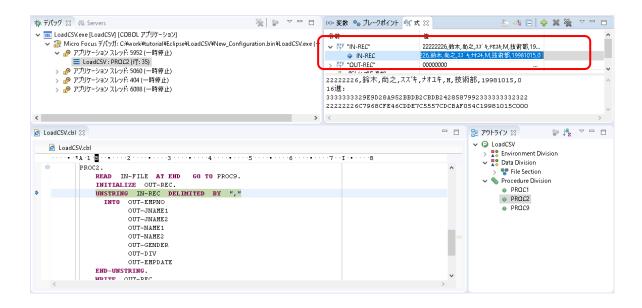
さらに **F5** キーを 2回押すと、デバッガーは UNSTRING 文を実行後、処理を中断します。 式ビューの OUT-REC の値には出力ファイルへ書き出す最初のレコードが表示されます。



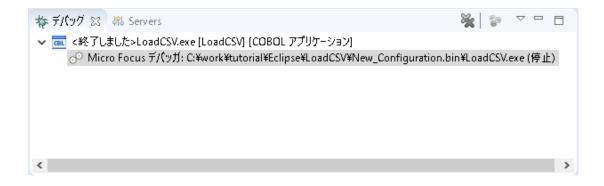
さらに F5 キーを 4回押すと、デバッガーは INITIALIZE 文を実行後、処理を中断します。



ウォッチ式の IN-REC の値には CSV ファイルから読み込んだ 2番目のレコードが表示され、OUT-REC の値は INITIALIZE 文で初期化されています。



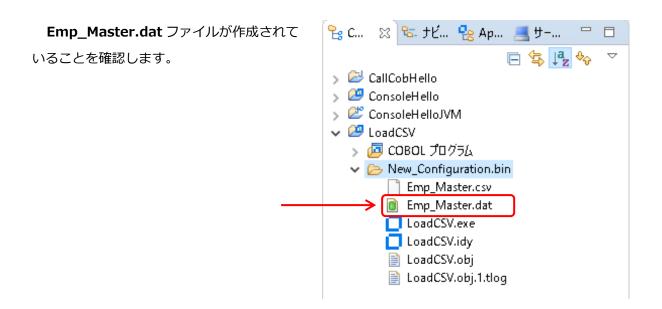
F8(再開)キーを打鍵するか CSV ファイルからすべてのレコードを読み込むまで **F5** キーを押すと、 デバッガは終了します。





ワークスペース右上コーナにあるパースペクティブ切り替えボタンエリアにて「**COBOL**」を選択し、COBOL パースペクティブに戻ります。

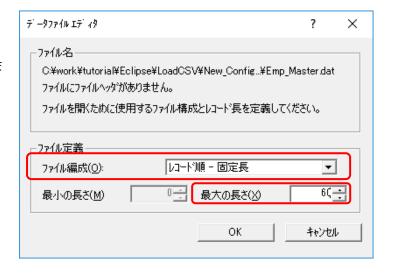
COBOL エクスプローラービューにて **LoadCSV** プロジェクト配下の **New_Configuration.bin** フォルダを右クリックし「**更新(F)**」を選択します。



Emp Master.dat を右クリックし、

アプリケーションから開く → クラシックデータファイルツールを選択します。Visual COBOL に

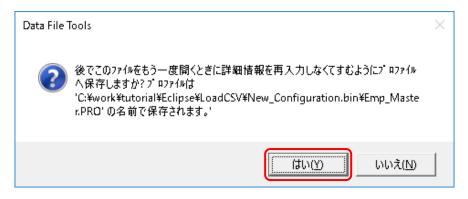
付属する データファイルエディタが起動します。[ファイル編成] 欄はデフォルトの「レコード順 - 固定長」のままにしておきます。[最大の長さ] 欄には「60」を指定します。



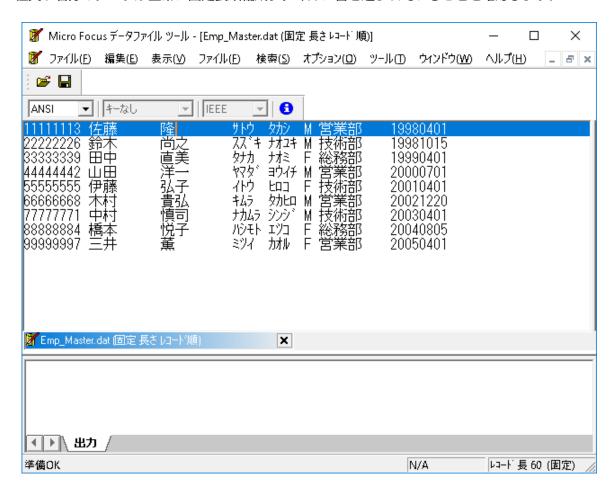


[**OK**] ボタンをクリックします。

プロファイルファイルを作成するかどうかの問いには [はい(Y)] を選択します。



社員9名分のデータが正常に固定長順編成ファイルに書き込まれていることを確認します。





第6章 Visual COBOL のバッチアプリケーション

本章では、第5章で作成した固定長順編成ファイルを読み込んでレポートファイルを作成するバッチアプリケーションを Visual COBOL for COBOL で作成します。

1 Visual COBOL for Eclipse を起動します。

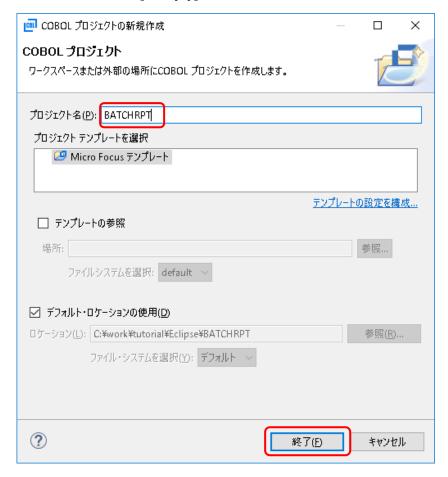
Windows のスタートメニューから、Visual COBOL for Eclipse をクリックします。

ワークスペースは前章までで使用されたものをそのまま使用します。

2 COBOL プロジェクトを作成します。

ファイル(F)メニューから 新規(N) \rightarrow COBOL プロジェクト を選択します。

プロジェクト名欄に BATCHRPT と入力し[終了(F)] ボタンをクリックします。

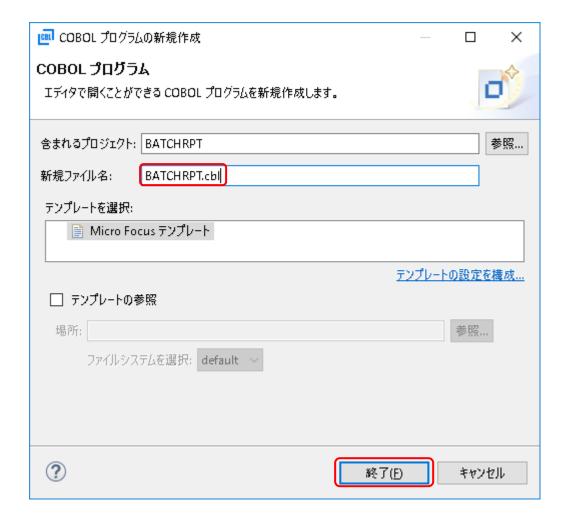




3 COBOL プログラムを追加します。

COBOL エクスプローラービューにて プロジェクト フォルダを右クリックし 新規(N) \rightarrow COBOL プログラムを選択します。

新規ファイル名欄には BATCHRPT.cbl を指定します。[終了(F)] ボタンをクリックします。





4 コードエディターで COBOL ソースコードを入力します。

本章では既存資産の流用を想定して COBOL 正書法に従った伝統的スタイルのソースコードを入力しますので、アスタリスクで始まるコメント行が 7 列目(エディタービュー左側のグレー領域の右端)から始まるよう注意して、以下の見出し部と環境部を入力します。 まだ、データ部のファイル定義が未入力なので 3 件のエラーとなりますが、ここでは無視して構いません。

IDENTIFICATION DIVISION. PROGRAM-ID. BATCHRPT.

ENVIRONMENT DIVISION. INPUT-OUTPUT SECTION. FILE-CONTROL.

* INPUT FILE: EMPLOYEE RECORDS

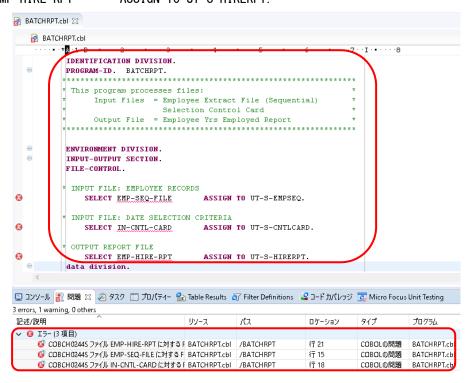
SELECT EMP-SEQ-FILE ASSIGN TO UT-S-EMPSEQ.

* INPUT FILE: DATE SELECTION CRITERIA

SELECT IN-CNTL-CARD ASSIGN TO UT-S-CNTLCARD.

* OUTPUT REPORT FILE SELECT EMP-HIRE-RPT

ASSIGN TO UT-S-HIRERPT.

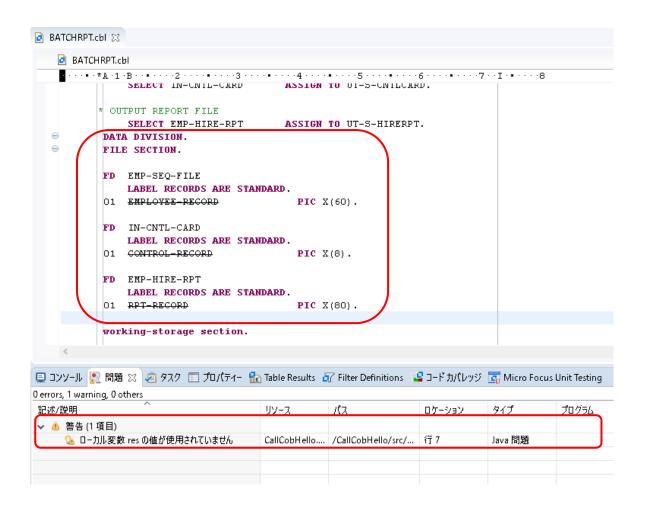




データ部のファイル節を入力します。 なお、データ部のファイル定義を入力したので、環境部のエラーは無くなります。

DATA DIVISION. FILE SECTION.

- FD EMP-SEQ-FILE LABEL RECORDS ARE STANDARD.
- 01 EMPLOYEE-RECORD PIC X (60).
- FD IN-CNTL-CARD LABEL RECORDS ARE STANDARD.
- 01 CONTROL-RECORD PIC X(8).
- FD EMP-HIRE-RPT LABEL RECORDS ARE STANDARD.
- 01 RPT-RECORD PIC X (80).





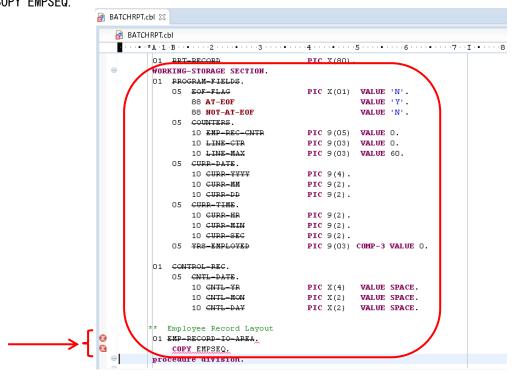
データ部の作業場所節で PROGRAM-FIELDS、CONTROL-REC データ項目を入力します。 COPY 文で外部参照する EMP-RECORD-IO-AREA データ項目はエラーとなりますが、無視して構いません。

WORKING-STORAGE SECTION.

)1	PR00	GRAM-FIELDS.						
	05	EOF-FLAG	PIC	X (01)	VALUE	' N' .		
		88 AT-E0F			VALUE	' Y' .		
		88 NOT-AT-EOF			VALUE	' N' .		
	05	COUNTERS.						
		10 EMP-REC-CNTR	PIC	9 (05)	VALUE	0.		
		10 LINE-CTR	PIC	9 (03)	VALUE	0.		
		10 LINE-MAX	PIC	9 (03)	VALUE	60.		
	05	CURR-DATE.						
		10 CURR-YYYY	PIC	9(4).				
		10 CURR-MM	PIC	9(2).				
		10 CURR-DD	PIC	9(2).				
	05	CURR-TIME.						
		10 CURR-HR	PIC	9(2).				
		10 CURR-MIN	PIC	9 (2).				
		10 CURR-SEC	PIC	9(2).				
	05	YRS-EMPLOYED	PIC	9 (03)	COMP-3	VALUE	0.	
)1	CONTROL-REC.							
		CNTL-DATE.						
		10 CNTL-YR	PIC	X (4)	VALUE	SPACE.		
					VALUE			
		10 CNTL-DAY			VALUE			

- ** Employee Record Layout
- 01 EMP-RECORD-IO-AREA. COPY EMPSEQ.

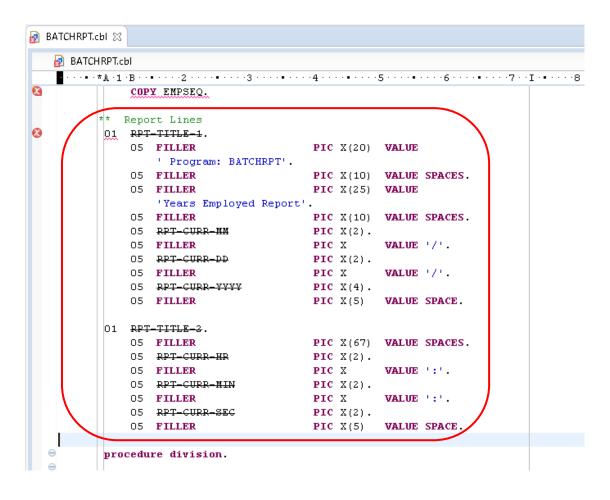
0





データ部の作業場所節で RPT-TITLE-1 と RPT-TITLE-2 データ項目を入力します。

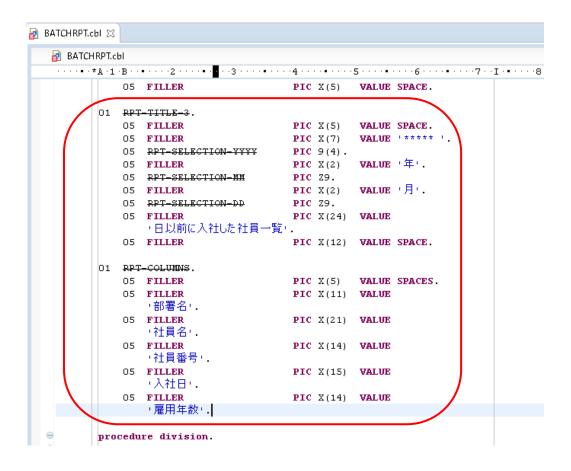
```
** Report Lines
 01 RPT-TITLE-1.
                                 PIC X(20) VALUE
     05 FILLER
         Program: BATCHRPT'.
     05 FILLER
                                 PIC X(10) VALUE SPACES.
     05 FILLER
                                 PIC X (25)
                                            VALUE
         'Years Employed Report'.
                                 PIC X(10)
                                           VALUE SPACES.
     05 FILLER
     05 RPT-CURR-MM
                                 PIC X(2).
                                            VALUE '/'.
     05 FILLER
                                 PIC X
     05 RPT-CURR-DD
                                 PIC X(2).
     05 FILLER
                                 PIC X
                                            VALUE '/'.
                                 PIC X(4).
     05 RPT-CURR-YYYY
     05 FILLER
                                 PIC X(5)
                                            VALUE SPACE.
 01 RPT-TITLE-2.
     05 FILLER
                                 PIC X (67) VALUE SPACES.
     05 RPT-CURR-HR
                                 PIC X(2).
                                            VALUE ':'.
     05 FILLER
                                 PIC X
     05 RPT-CURR-MIN
                                 PIC X(2).
     05 FILLER
                                            VALUE ':'.
                                 PIC X
     05 RPT-CURR-SEC
                                 PIC X(2).
                                            VALUE SPACE.
     05 FILLER
                                 PIC X (5)
```





作業場所節で RPT-TITLE-3 と RPT-COLUMNS データ項目を入力します。

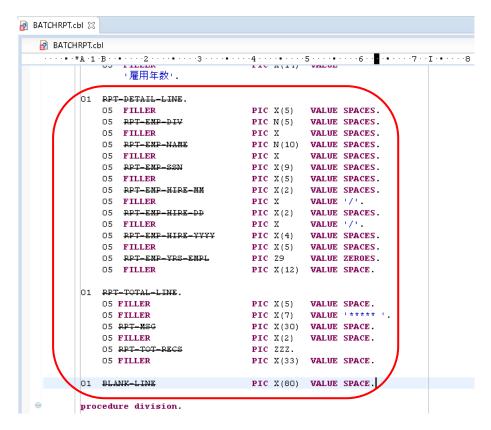
```
01 RPT-TITLE-3.
                              PIC X (5)
                                         VALUE SPACE.
   05 FILLER
                                         VALUE '***** '.
    05 FILLER
                              PIC X (7)
    05 RPT-SELECTION-YYYY
                              PIC 9(4).
    05 FILLER
                              PIC X(2)
                                         VALUE '年'.
    05 RPT-SELECTION-MM
                              PIC Z9.
                                         VALUE '月'.
    05 FILLER
                              PIC X (2)
    05 RPT-SELECTION-DD
                              PIC Z9.
                              PIC X (24)
                                         VALUE
    05 FILLER
        日以前に入社した社員一覧'.
    05 FILLER
                              PIC X(12) VALUE SPACE.
01 RPT-COLUMNS.
    05 FILLER
                              PIC X (5)
                                         VALUE SPACES.
    05 FILLER
                              PIC X(11)
                                         VALUE
       '部署名'.
    05 FILLER
                              PIC X(21) VALUE
       '社員名'.
                              PIC X(14) VALUE
    05 FILLER
        社員番号'.
    05 FILLER
                              PIC X (15) VALUE
       '入社日'.
   05 FILLER
                              PIC X (14) VALUE
        雇用年数'.
```





作業場所節で RPT-DETAIL-LINE、RPT-TOTAL-LINE と BLANK-LINE データ項目を入力します。

```
01 RPT-DETAIL-LINE.
                                 PIC X (5)
                                            VALUE SPACES.
    05 FILLER
    05 RPT-EMP-DIV
                                 PIC N(5)
                                            VALUE SPACES.
    05
       FILLER
                                 PIC X
                                            VALUE SPACES.
    05 RPT-EMP-NAME
                                 PIC N(10)
                                            VALUE SPACES.
    05 FILLER
                                 PIC X
                                            VALUE SPACES.
    05 RPT-EMP-SSN
                                 PIC X(9)
                                            VALUE SPACES.
    05 FILLER
                                 PIC X (5)
                                            VALUE SPACES.
                                            VALUE SPACES.
    05 RPT-EMP-HIRE-MM
                                 PIC X(2)
                                            VALUE '/'.
                                 PIC X
    05
       FILLER
    05 RPT-EMP-HIRE-DD
                                 PIC X(2)
                                            VALUE SPACES.
    05 FILLER
                                 PIC X
                                            VALUE '/'.
                                 PIC X (4)
                                            VALUE SPACES.
    05 RPT-EMP-HIRE-YYYY
    05 FILLER
                                 PIC X (5)
                                            VALUE SPACES.
    05 RPT-EMP-YRS-EMPL
                                 PIC Z9
                                            VALUE ZEROES.
    05 FILLER
                                 PIC X (12)
                                            VALUE SPACE.
01 RPT-TOTAL-LINE.
    05 FILLER
                                 PIC X (5)
                                            VALUE SPACE.
    05 FILLER
                                 PIC X (7)
                                            VALUE '***** '.
    05 RPT-MSG
                                 PIC X (30)
                                            VALUE SPACE.
    05 FILLER
                                 PIC X(2)
                                            VALUE SPACE.
    05 RPT-TOT-RECS
                                 PIC ZZZ.
    05 FILLER
                                 PIC X (33) VALUE SPACE.
01 BLANK-LINE
                                 PIC X(80) VALUE SPACE.
```





最後に、手続き部の 1000-START 節の前半部分を入力します。PERFORM 文で参照する手続き名が 未定義なのでエラーが 5 件増えますが、気にせず先に進んでください。

```
PROCEDURE DIVISION.

PERFORM 1000-START

PERFORM 2000-MAIN-PROCESSING

PERFORM 9000-CLOSE-AND-CLEANUP

STOP RUN.

THRU 1000-EXIT.

THRU 2000-EXIT UNTIL AT-EOF.

THRU 9000-EXIT.

THRU 1000-EXIT.

THRU 1000-EXIT.
```

* SET UP AND WRITE REPORT TITLE AND COLUMN HEADINGS

ACCEPT CURR-DATE FROM DATE YYYYMMDD.

MOVE CURR-MM TO RPT-CURR-MM.

MOVE CURR-DD TO RPT-CURR-DD.

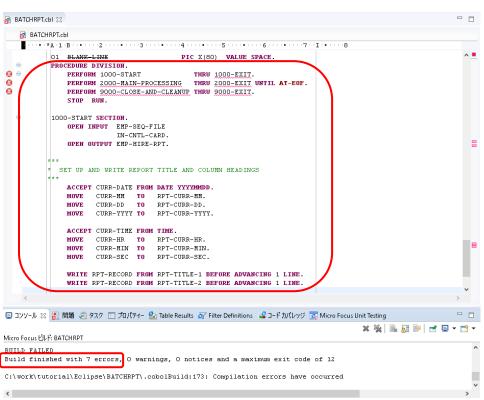
MOVE CURR-YYYY TO RPT-CURR-YYYY.

ACCEPT CURR-TIME FROM TIME.

OPEN OUTPUT EMP-HIRE-RPT.

MOVE CURR-HR TO RPT-CURR-HR.
MOVE CURR-MIN TO RPT-CURR-MIN.
MOVE CURR-SEC TO RPT-CURR-SEC.

WRITE RPT-RECORD FROM RPT-TITLE-1 BEFORE ADVANCING 1 LINE. WRITE RPT-RECORD FROM RPT-TITLE-2 BEFORE ADVANCING 1 LINE.





手続き部の 1000-START 節の後半部分を入力します。

- * READ CONTROL CARD FILE TO GET DATE FOR SELECTION CRITERIA.
- * IF FILE IS EMPTY, DEFAULT CNTL-DATE TO CURRENT DATE.

READ IN-CNTL-CARD INTO CONTROL-REC.

IF CNTL-DATE = SPACES

MOVE CURR-DATE TO CNTL-DATE
END-IF.

* ACCEPT CNTL-DATE FROM SYSIN.

MOVE CNTL-MON TO RPT-SELECTION-MM.

MOVE CNTL-DAY TO RPT-SELECTION-DD.

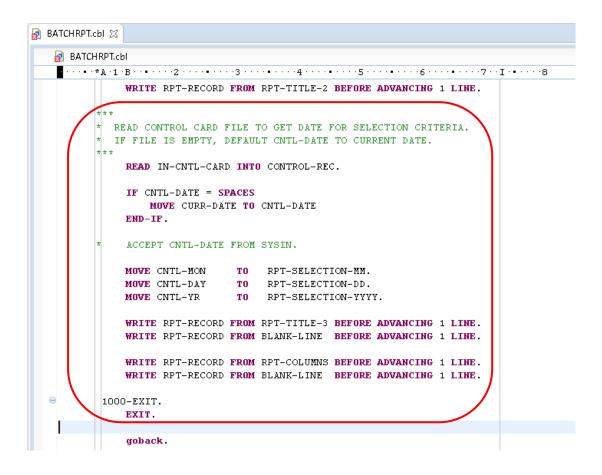
MOVE CNTL-YR TO RPT-SELECTION-YYYY.

WRITE RPT-RECORD FROM RPT-TITLE-3 BEFORE ADVANCING 1 LINE. WRITE RPT-RECORD FROM BLANK-LINE BEFORE ADVANCING 1 LINE.

WRITE RPT-RECORD FROM RPT-COLUMNS BEFORE ADVANCING 1 LINE. WRITE RPT-RECORD FROM BLANK-LINE BEFORE ADVANCING 1 LINE.

1000-EXIT.

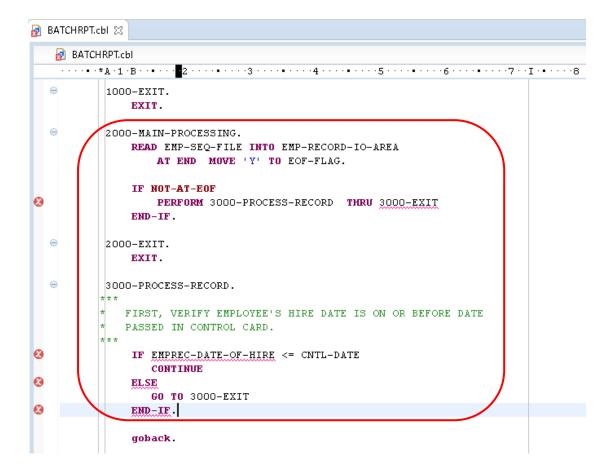
EXIT.





手続き部の 2000-MAIN-PROCESSING 段落と 3000-PROCESS-RECORD 段落の前半部分を入力します。

```
2000-MAIN-PROCESSING.
     READ EMP-SEQ-FILE INTO EMP-RECORD-IO-AREA
         AT END MOVE 'Y' TO EOF-FLAG.
     IF NOT-AT-EOF
         PERFORM 3000-PROCESS-RECORD THRU 3000-EXIT
     END-IF.
 2000-EXIT.
    EXIT.
 3000-PROCESS-RECORD.
***
   FIRST, VERIFY EMPLOYEE'S HIRE DATE IS ON OR BEFORE DATE
   PASSED IN CONTROL CARD.
***
     IF EMPREC-DATE-OF-HIRE <= CNTL-DATE
        CONTINUE
     ELSE
        GO TO 3000-EXIT
     END-IF.
```





手続き部の 3000-PROCESS-RECORD 段落の後半部分を入力します。

* FORMAT REPORT DETAIL LINES FROM EMPLOYEE RECORD.

MOVE EMPREC-DIV TO RPT-EMP-DIV.

MOVE SPACE TO RPT-EMP-NAME.

STRING EMPREC-JNAME1 DELIMITED BY SPACE

SPACE DELIMITED BY SIZE

EMPREC-JNAME2 DELIMITED BY SPACE

INTO RPT-EMP-NAME.

STRING EMPREC-SSN(1:7) DELIMITED BY SIZE
'-' DELIMITED BY SIZE

EMPREC-SSN (8:1) DELIMITED BY SIZE INTO RPT-EMP-SSN.

MOVE EMPREC-DOH-MM TO RPT-EMP-HIRE-MM.

MOVE EMPREC-DOH-DD TO RPT-EMP-HIRE-DD.

MOVE EMPREC-DOH-YYYY TO RPT-EMP-HIRE-YYYY.

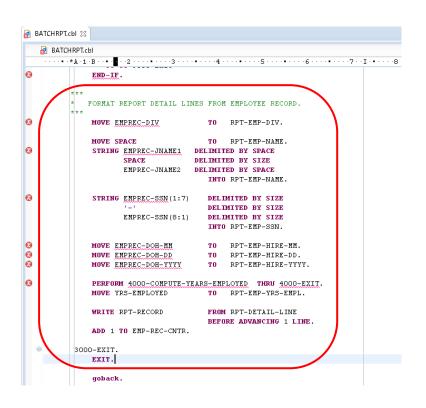
PERFORM 4000-COMPUTE-YEARS-EMPLOYED THRU 4000-EXIT. MOVE YRS-EMPLOYED TO RPT-EMP-YRS-EMPL.

WRITE RPT-RECORD FROM RPT-DETAIL-LINE
BEFORE ADVANCING 1 LINE.

ADD 1 TO EMP-REC-CNTR.

3000-EXIT.

EXIT.





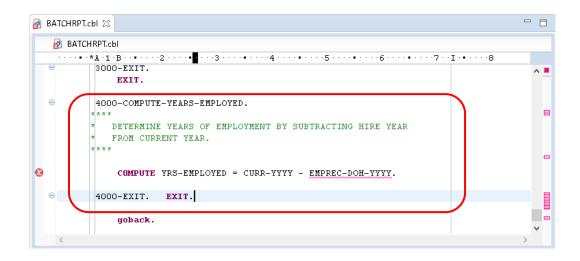
手続き部の 4000-COMPUTE-YEARS-EMPLOYED 段落を入力します。

4000-COMPUTE-YEARS-EMPLOYED.

- * DETERMINE YEARS OF EMPLOYMENT BY SUBTRACTING HIRE YEAR
- * FROM CURRENT YEAR.

COMPUTE YRS-EMPLOYED = CURR-YYYY - EMPREC-DOH-YYYY.

4000-EXIT. EXIT.





手続き部の 9000-CLOSE-AND-CLEANUP 段落を入力します。

9000-CLOSE-AND-CLEANUP.

IF EMP-REC-CNTR > 0

MOVE '処理レコード件数:' TO RPT-MSG

MOVE EMP-REC-CNTR TO RPT-TOT-RECS

ELSE

MOVE '処理レコードなし' TO RPT-MSG

END-IF.

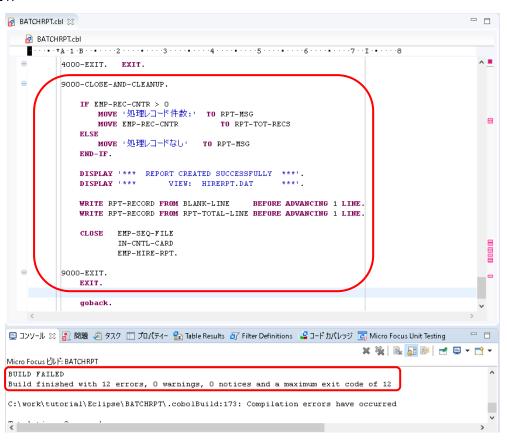
DISPLAY '*** REPORT CREATED SUCCESSFULLY ***'.
DISPLAY '*** VIEW: HIRERPT. DAT ***'.

WRITE RPT-RECORD FROM BLANK-LINE BEFORE ADVANCING 1 LINE. WRITE RPT-RECORD FROM RPT-TOTAL-LINE BEFORE ADVANCING 1 LINE.

CLOSE EMP-SEQ-FILE IN-CNTL-CARD EMP-HIRE-RPT.

9000-EXIT.

EXIT.



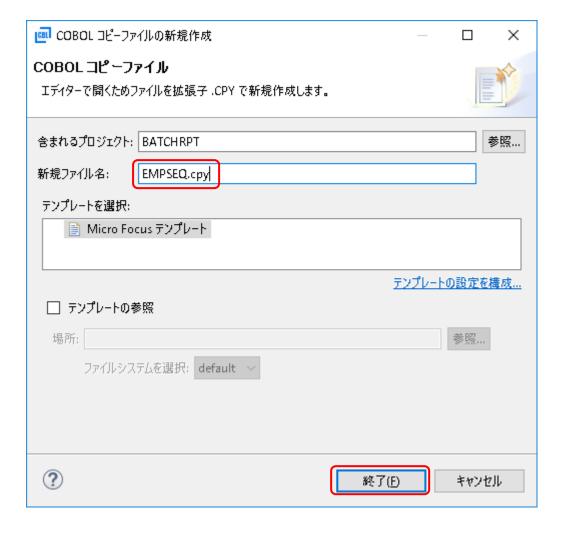
以上で BATCHRPT.cbl ソースプログラムの入力は終了です。 この時点でエラー件数が 12 であれば、問題ありません。先に進んでください。



5 コードエディターで COBOL コピーファイルを入力します。

COBOL エクスプローラービューにて **COBOL プログラム** フォルダを右クリックし **新規(N) \rightarrow COBOL コピーファイル**を選択します。

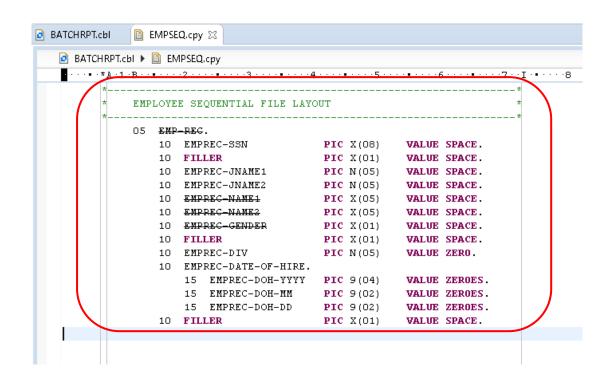
新規ファイル名欄には EMPSEQ.cpy を指定します。[終了(F)] ボタンをクリックします。





EMP-RECORD-IO-AREA データ項目のレコード記述を入力します。

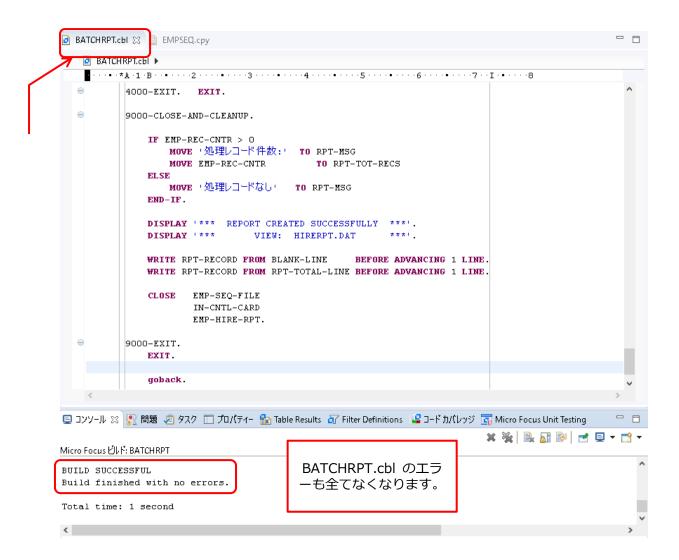
05	EMP	-REC.		
	10	EMPREC-SSN	PIC X(08)	VALUE SPACE.
	10	FILLER	PIC X(01)	VALUE SPACE.
	10	EMPREC-JNAME1	PIC N(05)	VALUE SPACE.
	10	EMPREC-JNAME2	PIC N(05)	VALUE SPACE.
	10	EMPREC-NAME1	PIC X(05)	VALUE SPACE.
	10	EMPREC-NAME2	PIC X(05)	VALUE SPACE.
	10	EMPREC-GENDER	PIC X(01)	VALUE SPACE.
	10	FILLER	PIC X(01)	VALUE SPACE.
	10	EMPREC-DIV	PIC N(05)	VALUE ZERO.
	10	EMPREC-DATE-OF-HIRE		
		15 EMPREC-DOH-YYYY	PIC 9(04)	VALUE ZEROES
		15 EMPREC-DOH-MM	PIC 9(02)	VALUE ZEROES
		15 EMPREC-DOH-DD	PIC 9(02)	VALUE ZEROES
	10	FILLER	PIC X(01)	VALUE SPACE.





エディタービュー上の EMPSEQ.cpy をアクティブにしたまま [ファイル(F)] メニューの [保管 (S)] 或いは Ctrl + S キーで保存します。これによりアプリケーションがビルドされます。

コンソールビューにエラーなくビルドできた旨が出力されていることを確認して、次に進んでください。





6 COBOL コンパイル指令を追加します。

本項ではファイル名の割り当てを EXTERNAL(外部割り当て)に変更するためのコンパイラ指令を指定します。

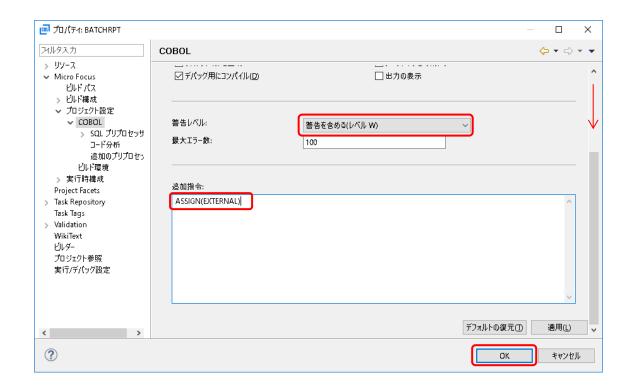
COBOL エクスプローラービューにて **BATCHRPT** プロジェクトを右クリックし [プロパティ(R)]を選択します。

Micro Focus COBOL > Project 設定 > COBOL とナビゲートし表示される画面では

警告レベル 警告を含める(レベル W)

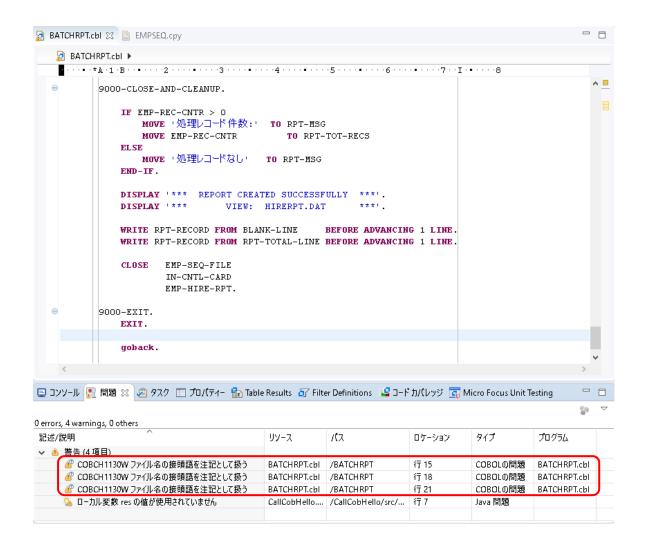
追加指令欄 ASSIGN(EXTERNAL)

のように選択/入力し、[**OK**] ボタンをクリックします。





問題ビューを選択し「ファイル名の接頭語を注記として扱う」警告が3件表示されることを確認して、先に進んでください。



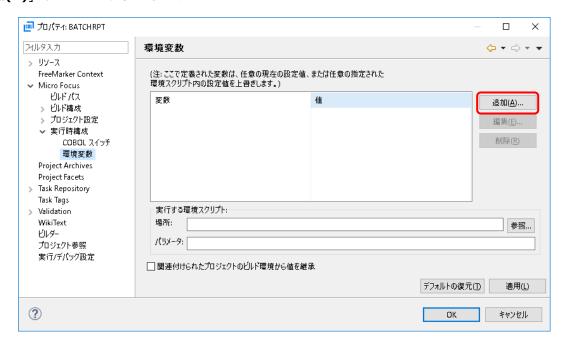


7 環境変数を構成します。

COBOL エクスプローラービューにて **BATCHRPT** プロジェクトを右クリックし[プロパティ(R)] を選択します。

Micro Focus COBOL > 実行時構成 > 環境変数とナビゲートします。

[追加(A)] ボタンをクリックします。



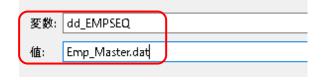
変数欄 dd_EMPSEQ

值欄 Emp Master.dat

💷 変数を追加

環境変数を追加または変更します

のように入力し [**OK**] ボタンをクリックします。





[**追加(A)**] ボタンをクリックし更に追加します。

変数欄 dd_CNTLCARD

值欄 Cntl_Card.dat

のように入力し [**OK**] ボタンをクリックします。



[**追加(A)**] ボタンをクリックし更に追加します。

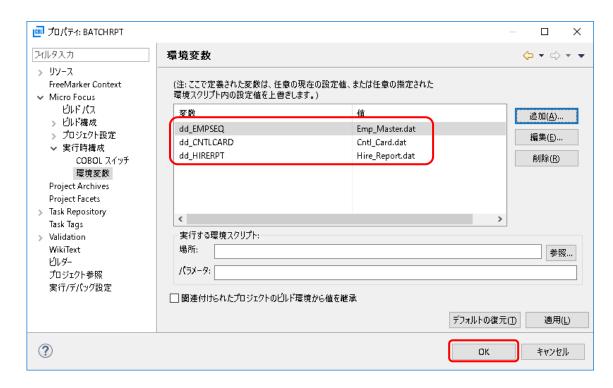
変数欄 dd_HIRERPT

值欄 Hire_Report.dat

のように入力し [**OK**] ボタンをクリックします。



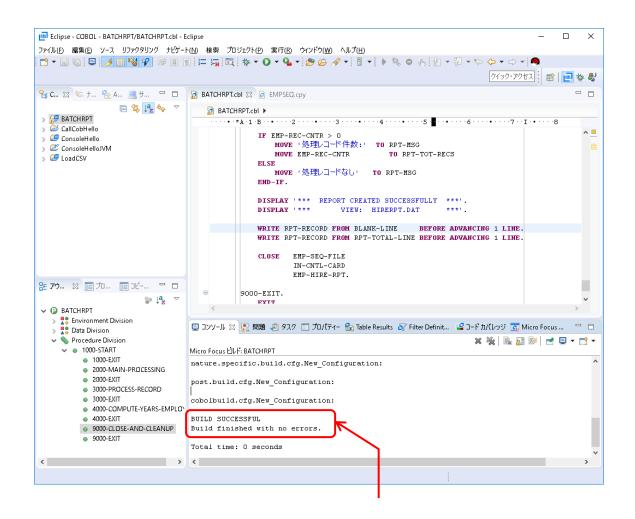
下図のように指定が反映されていることを確認して、[OK] ボタンをクリックします。





8 COBOL アプリケーションをビルドします。

プロパティー画面で [**OK**] ボタンをクリックしますと、自動ビルド機能によりビルド処理がキックされます。正常にビルドされることを確認してください。





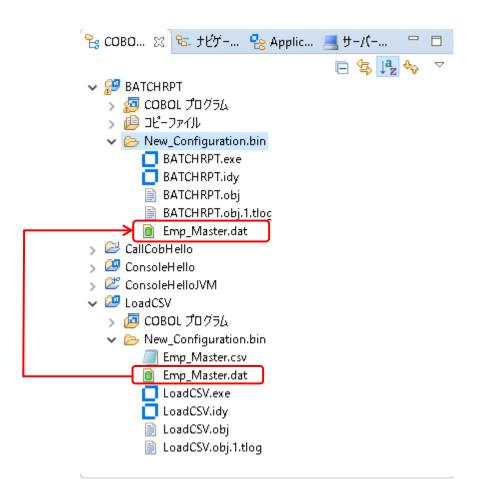
9 入力ファイルをコピーします。

前章で作成した **Emp_Master.dat** ファイルを、COBOL エクスプローラービューより選択します。

右クリックより「コピー」を選択します。

BATCHRPT プロジェクト配下の **New_Configuration.bin** フォルダを選択し右クリックから 「**貼り付け**」を選択します。

COBOL エクスプローラー上でデータファイルコピーが反映されたことを確認します。



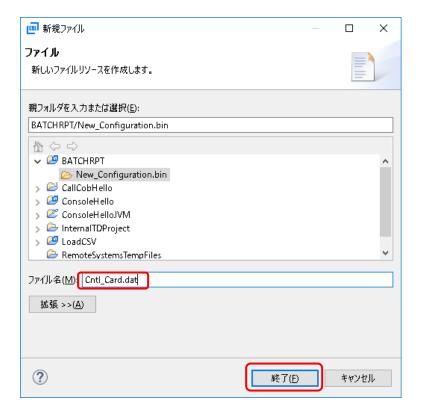


10 制御ファイルを作成します。

New_Configuration.bin フォルダを右クリックし

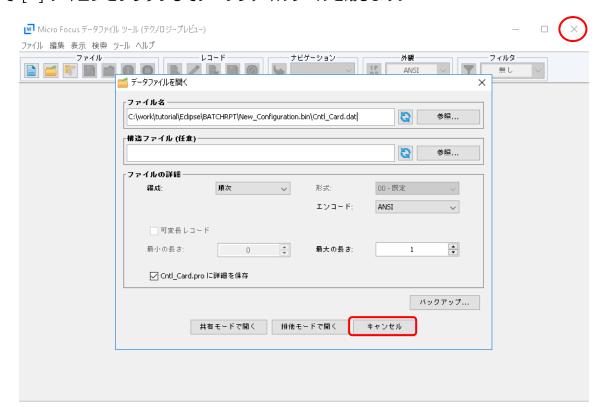
新規(N) → ファイルを選択します。

[ファイル名] 欄にて Cntl_Card.dat を指定し [終了(F)] ボタンをクリックします。





データファイルツールが起動します。本チュートリアルではこのテクノロジープレビュー版ではなく従来より提供しているクラシック版を利用しますのでここでは [**キャンセル**] ボタンをクリックし、続けて [×] アイコンをクリックしてデータファイルツールを閉じます。



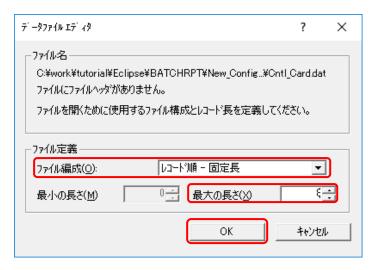
COBOL エクスプローラーにて Cntl_Card.dat を右クリックし、

アプリケーションから開く → **クラシックデータファイルツール** へとナビゲートします。





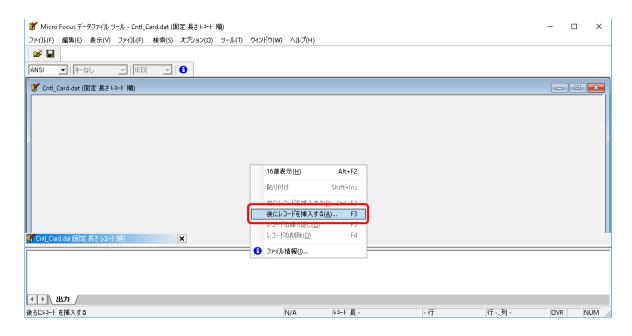
[ファイル編成] 欄はデフォルトの 「レコード順 - 固定長」のままにし て、[最大の長さ] 欄には「8」を指定し ます。[OK] ボタンをクリックします。



プロファイルファイルを作成するかどうかの問いには [はい(Y)] を選択します。



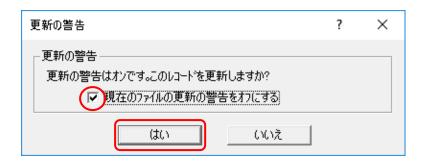
右クリックから [後にレコードを挿入する] を選択します。

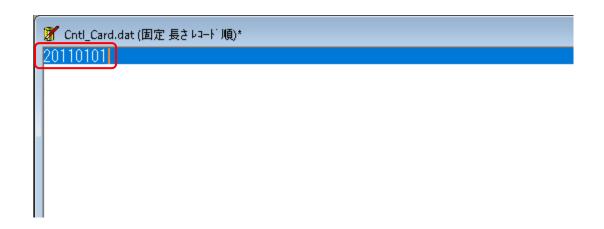




「20110101」を入力します。

更新の警告がポップアップされましたら、[**現在のファイルの更新の警告をオフにする**] にチェックを入れ、[**はい**] ボタンをクリックします。





[ファイル] メニューから [保存] を選択してファイルを保存します。

Data File Editor を終了します。

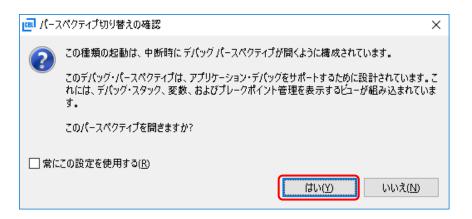


11 COBOL アプリケーションをデバッグ実行します。

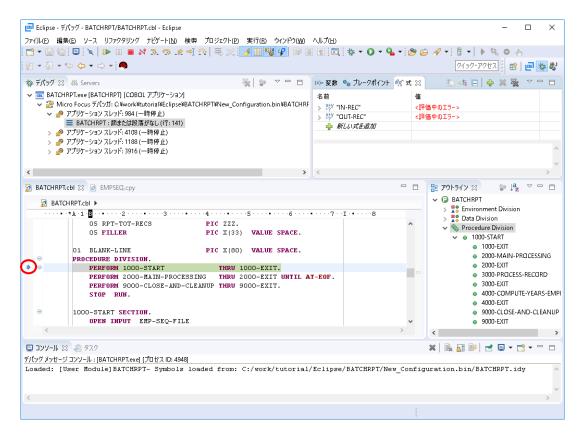
New_Configuration.bin フォルダ配下の BATCHRPT.exe を右クリックし

デバッグ(D) → COBOL アプリケーションを選択します。

パースペクティブ切り替えの確認メッセージには [**はい(Y)**] を選択します。

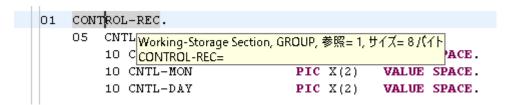


デバッガーがステップ実行を開始します。 デバッガーは手続き部の最初の COBOL 文である PERFORM 文の処理前に一時停止している状態となっています。





制御ファイルから読み込んだレコードの内容を確認するため、CONTROL-REC に格納される値の 変遷を追います。データ部の CONTROL-REC を選択します。

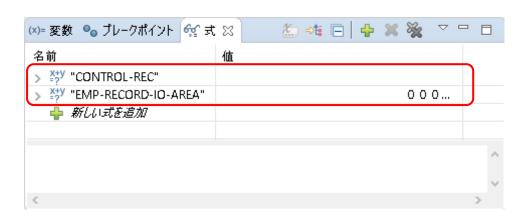


この状態で右クリックし、「**項目を検査**」を選択します。

「式ビューに追加」をクリックします。

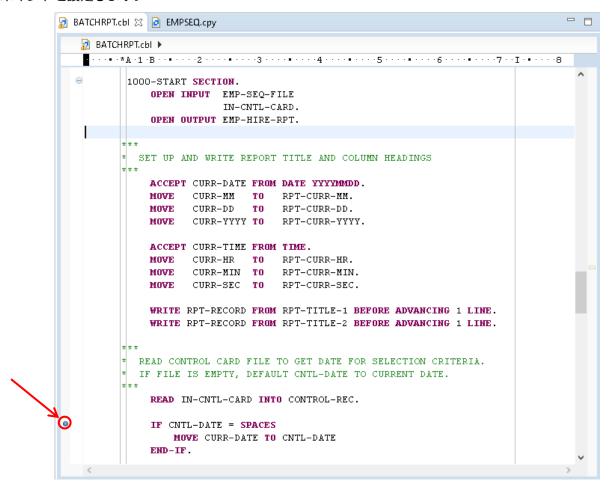


入力ファイルから読み込んだレコードの内容を確認するため、データ部の **EMP-RECORD-IO-AREA** を同じ要領で式ビューに追加します。(前章で追加した IN-REC 及び OUT-REC は式ビューから削除しても構いません。)

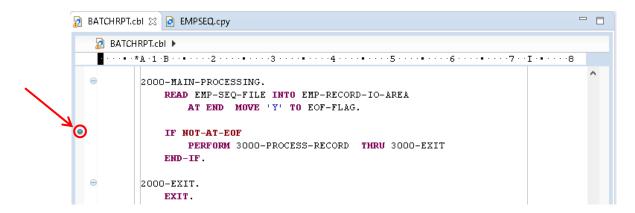




手続き部 **1000-START 節**の READ 文に続く IF 文でエディタービューの左端をクリックし、ブレークポイントを設定します。



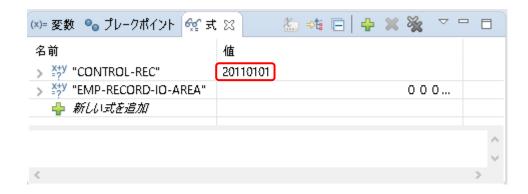
同様に手続き部 **2000-MAIN-PROCESSING 段落**の READ 文に続く IF 文でエディタービューの左端をクリックし、ブレークポイントを設定します。





実行(R)メニューから 再開(M) を選択するか F8 キーを打鍵すると、デバッガーは最初のブレークポイントで実行を中断します。

式ビュー中の CONTROL-REC の値に制御ファイルから読み込んだレコードが表示されます。



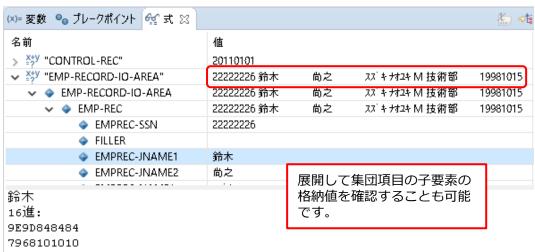
実行(R)メニューから **再開(M)** を選択するか **F8** キーを打鍵すると、デバッガーは 2 番目のブレークポイントで実行を中断します。

式ビューの EMP-RECORD-IO-AREA の値に入力ファイルから読み込んだ 1 番目のレコードが表示されます。



同様に**実行(R)**メニューから **再開(M)** を選択するか **F8** キーを打鍵すると、デバッガーは 2 番目のブレークポイントで実行を中断します。

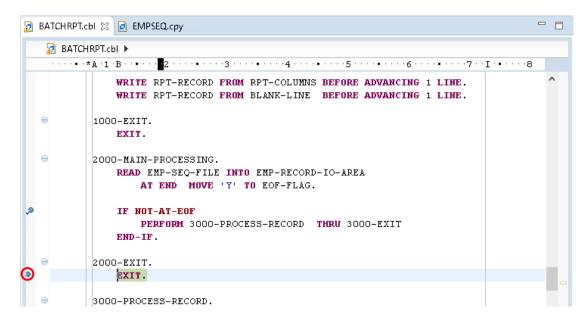
ウォッチ式の EMP-RECORD-IO-AREA の値に入力ファイルから読み込んだ 2 番目のレコードが表示されます。





さらに **F8** キーを 8 回、 **F5(ステップイン)**キーを 1 回打鍵すると、デバッガーは 2 番目のブレークポイントに続く EXIT 文で実行を中断します。

IF文の条件式は、入力ファイルがファイル終了状態であることを示しています。



実行(R)メニューから **再開(M)** を選択するか STOP 文を実行するまで **F5** キーを押すと、デバッガーは終了します。



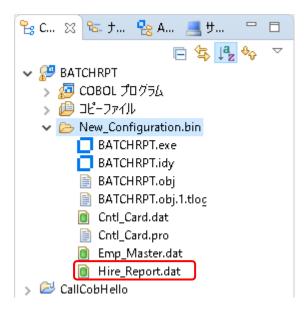
ワークスペース右上コーナにあるパースペクティブ切り替えボタンエリアにて「**COBOL**」を選択し、COBOL パースペクティブに戻ります。





COBOL エクスプローラービューにて **BATCHRPT** プロジェクト配下の **New_Configuration.bin** フォルダを右クリックし [**更新(F)**] を選択します。

New_Configuration.bin 配下にに Hire_Report.dat が生成されていることを確認します。



Hire_Report.dat を右クリックし

アプリケーションから開く → テキスト・エディターを選択します。

社員9名分のデータが表示されることを確認します1。



1 Eclipse におけるデフォルトのテキストエディタフォントがプロポーショナルになっている場合は多少見た目が異なる可能性があります。この場合、テキストエディタ上で右クリックから [設定] を選択し [一般] > [外観] > [色とフォント] で表示されるページにてフォントを変更できます。

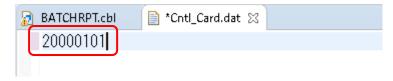


続いて、制御ファイルの値を変更してアプリケーションを実行します。

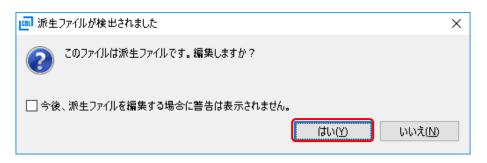
COBOL エクスプローラー上で Cntl_Card.dat ファイルを右クリックし

アプリケーションから開く → テキスト・エディターを選択します。

Cntl_Card.dat ファイル中のデータを「20000101」に更新します。



下図のような警告が出た場合は、はい を選択し、編集を進めてください。



[ファイル] メニュー > [保管] もしくは Ctrl + S を打鍵して、ファイルを保存します。

New_Configuration.bin フォルダ配下の BATCHRPT.exe を右クリックし 実行 → COBOL アプリケーション

を選択しアプリケーションを実行します。

コンソール画面がポップアップされたら、メッセージに従い、何等かのキーを打鍵しアプリケーションを終了します。





Hire_Report.dat を右クリックし

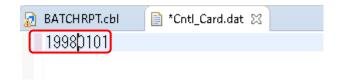
アプリケーションから開く → テキスト・エディターを選択します。

2000年1月1日以前に入社した社員3名分のデータだけが表示されることを確認します2。



続いて、制御ファイルの値を変更してアプリケーションを実行します。
COBOL エクスプローラー上で Cntl_Card.dat ファイルを右クリックし
アプリケーションから開く → テキスト・エディターを選択します。

Cntl_Card.dat ファイル中のデータを「19980101」に更新します。



[**ファイル**] メニュー > [**保管**] もしくは**Ctrl + S**を打鍵して、ファイルを保存します。

_

² 正しく表示されない場合は、COBOL エクスプローラー上で一度ファイルを右クリックの上 **更新** を選択しリフレッシュさせてください。



New_Configuration.bin フォルダ配下の BATCHRPT.exe を右クリックし 実行 → COBOL アプリケーションを選択しアプリケーションを実行します。

コンソール画面がポップアップされたら、メッセージに従い、何等かのキーを打鍵しアプリケーションを終了します。



Hire_Report.dat を右クリックし

アプリケーションから開く → テキスト・エディターを選択します。

処理されたレコードない旨の出力が表示されることを確認します。





2018 年 9 月 21 日

第5版

https://www.microfocus.co.jp/