Visual COBOL チュートリアル

COBOL 開発： Linux/UNIX 版 リモート開発編

1 目的

本チュートリアルは、Visual COBOL の Linux/UNIX が提供するリモート開発機能を学ぶためのチュートリアルです。リモート開発を利用することで、以下のメリットを享受できるようになります。

- 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 Linux/UNIX 環境をターゲットとしたアプリケーション開発が行えるため、Linux/UNIX 環境における開発効率の向上

2 前提

- 本チュートリアルで使用したマシン OS ： Windows Server 11, Rocky Linux 9.4
- Windows 上に Visual COBOL 11.0 Patch Update 01 for Eclipse がインストール済みであること
- Rocky Linux 9.4 上に Visual COBOL 11.0 Development Hub Patch Update 01 がインストール済みであること

本チュートリアルでは、「Visual COBOL for Eclipse チュートリアル」の内容を参照しています。Eclipse IDE を含めた製品操作の習熟を目的としていることから、本チュートリアル実施前には、「Visual COBOL for Eclipse チュートリアル」を実施してください。

実施しない場合は、こちらから本チュートリアルで参照するファイルをダウンロードしたうえで、任意のフォルダーに解凍してください。

内容

3 チュートリアル手順

3.1 環境のセットアップ

Visual COBOL の Linux/UNIX 版の開発ライセンスは Windows にインストールして利用する Visual COBOL for Eclipse と Linux/UNIX 環境にインストールする Visual COBOL Development Hub がセットになったライセンスです。リモート開発を行うためには、両製品ともにセットアップを行う必要があります。

3.1.1 Visual COBOL for Eclipse のセットアップ

未実施の場合、「Visual COBOL for Eclipse チュートリアル」の内容に従ってセットアップ、および、チュートリアル内容を先に行ってください。

3.1.2 Visual COBOL Development Hub のセットアップ

1) インストールプログラム[1]を Linux へ転送します。
2) リリースノートを確認し、インストール要件を満たしていることを確認します。
3) 転送したインストールプログラムを解凍します。
4) 管理者権限を持ったユーザーへ切り替えます。
5) 解凍したインストーラへ実行権限を与えます 。

```
# chmod u+x setup_visualcobol_devhub_11.0_patchupdate01_387838_rocky_x86_64
```

6) インストール処理を開始します[2] 。

```
# ./setup_visualcobol_devhub_11.0_patchupdate01_387838_rocky_x86_64 -installlocation=/opt/rs/VC110PU01
    -=-================================================================-=-
                    Rocket Software Product - Product Extractor
                                www.rocketsoftware.com

   Please Wait.
      Performing Product and Platform Checks...

          All Checks Passed.

      Extracting Payload...

      Creating work area...

    -=-================================================================-=-
            Rocket Visual COBOL Development Hub 11.0 - Patch Update 01
                                www.rocketsoftware.com
```

[1] インストーラのファイル名は、「setup_visualcobol_devhub_11.0_patchupdate01_<プラットフォーム名>」の形式で構成されています。お客様のインストール環境によって、ファイル名が異なりますのでご注意ください。

[2] デフォルトのインストールディレクトリは「/opt/rocketsoftware/VisualCOBOL」です。本例では「-installlocation=/opt/rs/VC110PU01」を指定し、インストールディレクトリを変更しています

```
   製品のインストール先
      [ /opt/rs/VC110PU01 ]

   製品の構成
      [ /opt/microfocus/config/55855 ]

   製品のライセンス機能
      CES          : [ /opt/microfocus/licensing/bin ]
      RocketPass : [ /opt/rocketsoftware/licensing/rocketpass ]

   製品の情報
      製品         : Rocket Visual COBOL Development Hub 11.0 - Patch Update 01
      ESadminID : Not given
      ビルド ID : pkg_387838

      機能         : 32/64 bit, SOA, Remote Dev, RocketPass Licensing

               Java 最小バージョン      : 21.0.0.0
   インストール済み Java バージョン : OpenJDK 17.0.17.0

   インストール時に、SOA サポートを構成するには、コマンドライン引数
   -ESadminID=[ User ID ] を指定してインストーラを実行してください。

                  -=----------------------------------------------=-

   このソフトウェア製品をインストールしてご使用になる前に、この
   製品に同封のエンドユーザ使用許諾契約（以下「使用許諾契約」
   という）の条項に拘束されることに同意する必要があります。
   使用許諾契約は必ずお読みください。
   使用許諾契約にご同意いただけない場合は、インストール処理を
   実行する前にご購入の窓口担当までご連絡ください。

   製品をインストールする前に「使用許諾契約」のコピーが必要な場合は、
   同意しないで、次のコマンドでインストーラを再度実行してください：

      ./setup_visualcobol_devhub_11.0_patchupdate01_387838_rocky_x86_64 -EULA

使用許諾契約の条件に同意しますか? (y/n): y

   使用許諾契約（EULA）は製品ディレクトリの次のファイルで確認できます:
      /opt/rs/VC110PU01/etc/EULA_VCED_v11_0_jp.htm


   Rocket Visual COBOL Development Hub 11.0 - Patch Update 01 の SOA サポートを構成するには、
```

```
      $COBDIR/bin/casperm.sh を実行してください。

   -=-=================================================
=================-=-
          Rocket Visual COBOL Development Hub 11.0 - Patch Update 01
                    インストールが完了しました。

   使用許諾契約（EULA）は製品ディレクトリの次のファイルで確認できます:
       /opt/rs/VC110PU01/etc/EULA_VCED_v11_0.htm

   -=-=================================================
=================-=-

        このバージョンの次の製品を使用するには :
           Rocket Visual COBOL Development Hub 11.0 - Patch Update 01

        環境を設定するため、"cobsetenv" を実行してください。

                . /opt/rs/VC110PU01/bin/cobsetenv

   -=-=================================================
=================-=-
#
```

3.2 Development Hub のインストール確認

Visual COBOL Development Hub は本書で紹介するリモート開発機能に加えて従来の COBOL 製品が提供するコマンドラインインターフェース機能も引き継いでいます。本章では前章でインストールした Visual COBOL Development Hub が正しくインストールされたことをこのコマンドラインインターフェースを使ったコンパイル及びテスト実行作業を通じて確認します。

1) ライセンスが未投入の場合は、インストールマニュアルに従い、ライセンスを適用します。
2) 一般ユーザーに戻ります。
3) Visual COBOL の利用に必要な環境変数を設定します。

   Visual COBOL Development Hub をインストールすると Visual COBOL の利用に最低限必要な環境変数をセットアップするスクリプトが

   <インストールディレクトリ> /bin/cobsetenv

   に用意されます。本ステップではこのセットアップスクリプトを実行して環境変数設定をします。

```
$ . /opt/rs/VC110PU01/bin/cobsetenv
COBDIR set to /opt/rs/VC110PU01
```

このスクリプトにより設定される主な環境変数を下記に記します。

環境変数名	設定内容
COBDIR	製品のベースディレクトリ(インストールディレクトリ)
PATH	$COBDIR/bin
ライブラリ探索パス[3]	$COBDIR/lib

4) 製品同梱サンプルをコピーします。

Visual COBOL Development Hub をインストールすると $COBDIR/demo ディレクトリ配下にサンプルプログラム及びビルドスクリプトがカテゴリ分けされて配置されます。ここでは、このサンプル中における簡単なコンソールアプリケーションプログラムをワークディレクトリにコピーします。

```
$ cp -p $COBDIR/demo/cobol/tictac/*.cbl ./
$ ls
tictac.cbl
```

5) コピーしたプログラムを実行形式にコンパイルします。

Visual COBOL は COBOL プログラムを実行形式、ライブラリファイル、呼び出し可能な共有オブジェクト、動的ロードモジュール等、目的に応じて適切な形式にビルドする機能を持っています。ここでは、コピーしたサンプルプログラムを実行形式ファイルへシングルステップでビルドします。下記のコマンド実行結果からもわかるように、この 1 つのコマンドにより、中間コード、オブジェクトコードの生成並びに実行形式へのリンクが処理されていることがわかります。

```
$ cob -x tictac.cbl
$ ls
tictac   tictac.cbl   tictac.idy   tictac.int   tictac.o
```

6) ビルドしたアプリケーションをテスト実行します。

Visual COBOL Development Hub にはテスト実行機能が装備されており、コンパイル・ビルドしたモジュールを同環境上でテスト実行することが可能です。ここではこの機能を使って、生成した実行形式のプログラムをテスト実行してみます。tictac は○×ゲームのロジックを COBOL で組み上げたものとなります。プロンプトに従って○×ゲームを進めてみてください。

【実行イメージ】

１．実行形式を実行

```
$ ./tictac
To select a square type a number between 1 and 9
Shall I start ?
```

２．先攻／後攻を選択、本例では player が先攻となるよう選択

```
Shall I start ? n
```

３．ゲーム画面に切り替わるので、フィールドを選択してゲームを実行

```
To select a square type a number between 1 and 9
Please select an empty square 0
                              7|      8|      9
                               |       |
                               |       |
                         -------+-------+-------
```

[3] LD_LIBARY_PATH, LIBPATH, SHLIB_PATH 等、プラットフォームによって環境変数名は異なります。

```
                                4|       5|       6
                                 |        |
                                 |        |
                          -------+--------+-------
                                1|       2|       3
                                 |        |
                                 |        |
```

7) ゲーム終了後、アプリケーションの終了を選択

```
To select a square type a number between 1 and 9
stalemate
Play again ? n
```

3.3 COBOL リモートプロジェクトの作成

ここからは、Windows 上にインストールされた Visual COBOL for Eclipse を使って Linux/UNIX 環境上に直接 COBOL アプリケーションをビルド生成してみます。アプリケーションのリソースは事前学習で利用した「Visual COBOL for Eclipse チュートリアル」で用意したものを利用します。

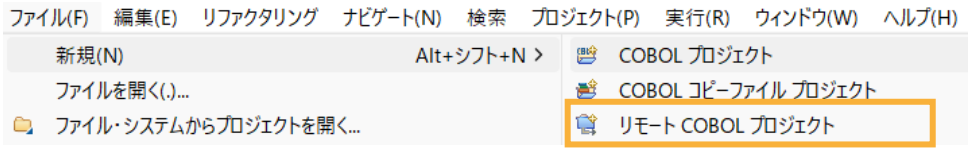
1) Windows 上にて、Visual COBOL for Eclipse を起動します。

ワークスペースには任意のフォルダーを指定してください。

起動後、ようこそ画面は閉じてください。

2) COBOL リモート プロジェクトを作成します。

① [ファイル(F)] メニューから [新規(N)] > [リモート COBOL プロジェクト] を選択します。



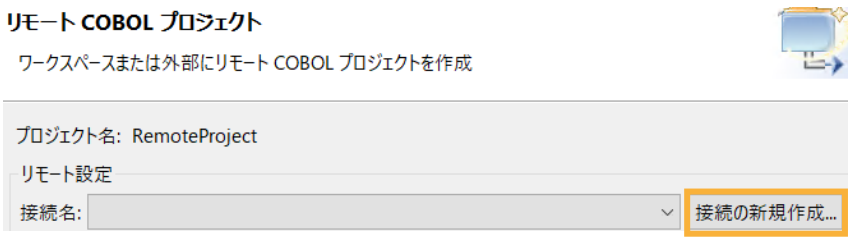
② 以下の設定を行い、[次へ(N)] をクリックします。

[プロジェクト名]： “RemoteProject”

[ファイルシステムを選択]： “リモートファイルシステム(RSE)”

③　プロジェクトテンプレートは［Rocket テンプレート[64 ビット]］を選択し［次へ(N)］をクリックします。

④　［接続の新規作成］をクリックします。



⑤ [Rocket DevHub SSH 使用] を選択し、[次へ(N)] をクリックします。

補足）

Linux 側にデーモンを起動して接続する構成も選択できます。この方式では、デーモン起動時に設定された環境変数が接続ユーザーに対して適用されます。一方、本書で使用する接続方式では、ログインユーザーの環境変数が接続時に有効になります。

リモートデーモンは、以下の手順で起動できます。

１．管理者権限を持つユーザーに切り替えます。

２．Visual COBOL の利用に必要な環境変数を設定します。

```
# . /opt/rs/VC110PU01/bin/cobsetenv
COBDIR set to /opt/rs/VC110PU01
```

３．デーモンを起動します 。

```
# $COBDIR/remotedev/startrdodaemon
Starting RSE daemon...
Reading "/opt/rs/VC110PU01/remotedev/rdo.cfg" ...
Daemon port loaded from "/opt/rs/VC110PU01/remotedev/rdo.cfg": 4075
Server port range loaded from "/opt/rs/VC110PU01/remotedev/rdo.cfg": 10000-10003
```

⑥ [ホスト名] 欄に、Linux 側のホスト名、もしくは、IP アドレスを指定します。[接続名] 欄は自動で [ホスト名] 欄の値がコピーされます。指定が終わりましたら [次へ(N)] をクリックします。

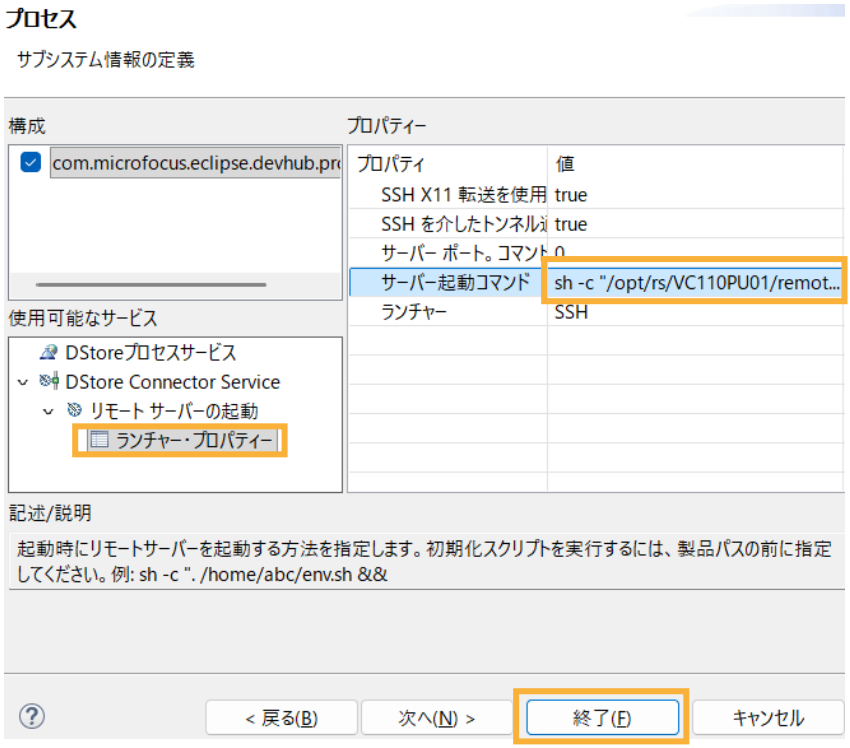
⑦ Development Hub のインストール先を変更した場合は、以下の設定を行ってください。

-installlocation オプションを設定せず、インストールした場合は変更する必要はありません。

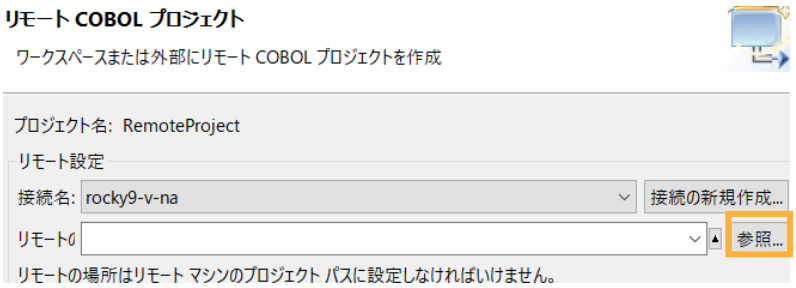
- [ランチャー・プロパティー] を選択したのちに表示される [サーバー起動コマンド]

  sh -c "<製品インストールディレクトリ>/remotedev/startrdoserver ${port} " &

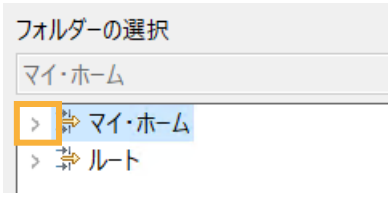
  本書の例では、sh -c "/opt/rs/VC110PU01/remotedev/startrdoserver ${port} " & になります。



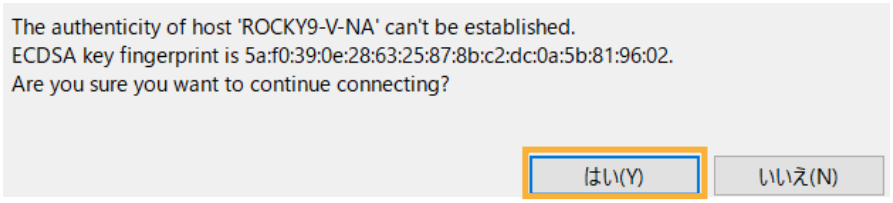
⑧ [参照...] をクリックします。



⑨ [マイ・ホーム] の左の展開アイコンをクリックします。



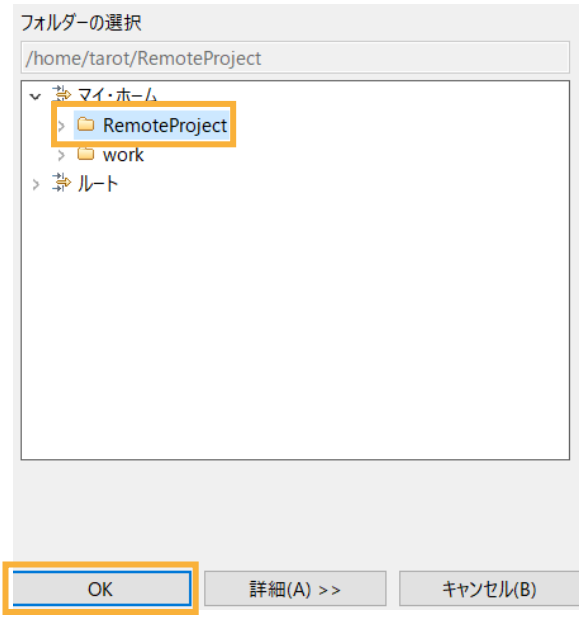
以下のようなダイアログには、[はい(Y)] をクリックします。



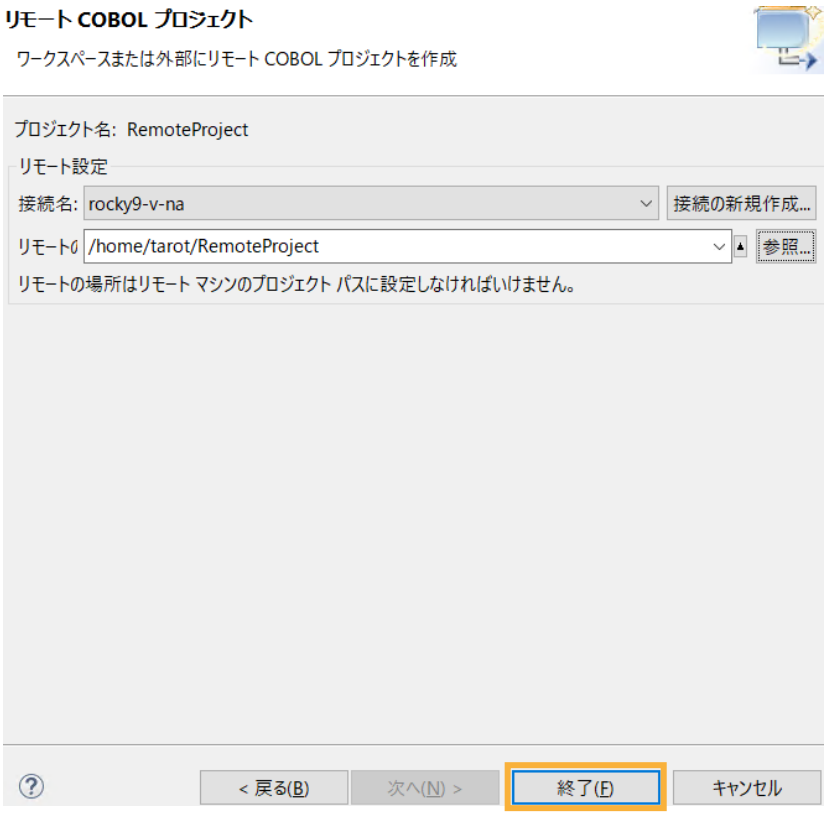
⑩ Linux/UNIX 側で利用する認証情報を入力、[パスワードを保管] を選択して [OK] をクリックします。

⑪ いくつかワーニングダイアログが表示されることがあります。その場合、すべての応答に [はい(Y)] をクリックします。

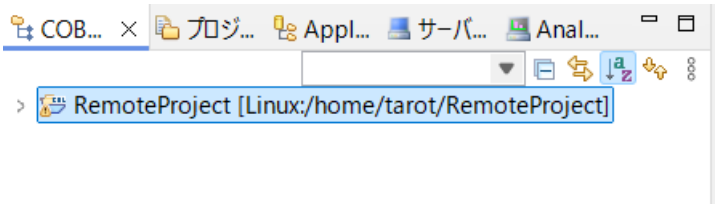
⑫ Linux/UNIX 側でソースや生成されるモジュール等を格納するプロジェクトディレクトリとして利用するディレクトリをツリーで選択し、[OK] をクリックします。



⑬ [終了(F)] をクリックします。

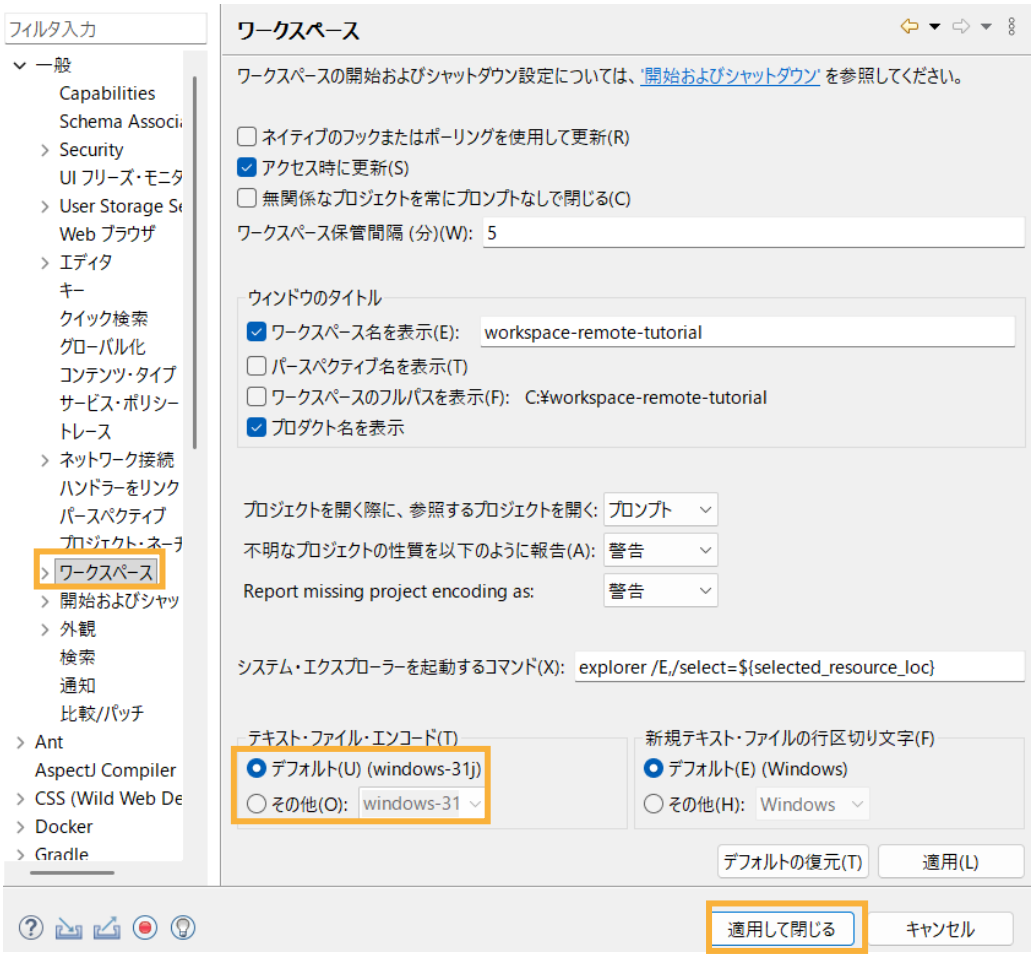


プロジェクトが作成されます。



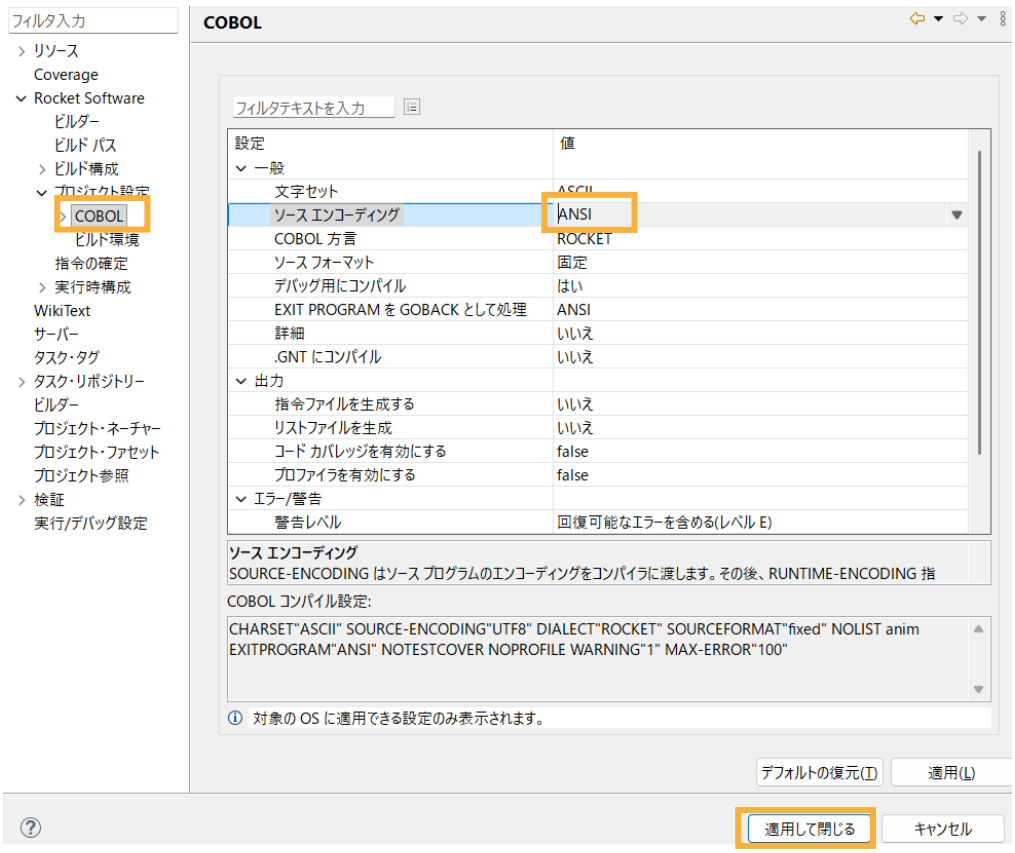
3) 文字コードの指定を行います。

① SJIS 資産を使用する場合、文字コードの指定を明確に行う必要があります。最初に、[Window(W)]メニュー > [設定(P)] より [一般] > [ワークスペース] とナビゲートし、テキストファイルエンコードを「デフォルト(windows-31j)」に変更し、[適用して閉じる] をクリックします。



Preference Recorder のダイアログが表示された場合は、[キャンセル] をクリックします。

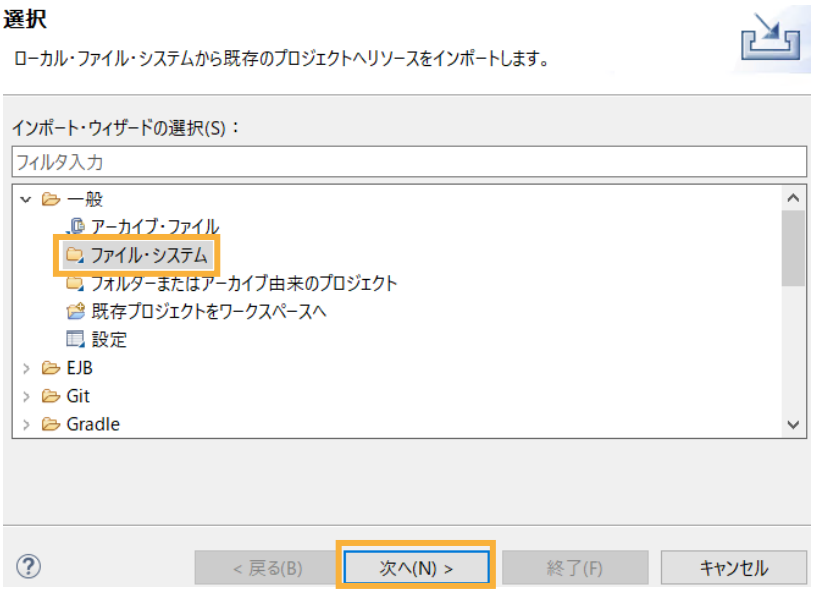
② 次に作成した COBOL プロジェクトを選択した状態で、マウスの右クリックにてコンテクストメニューを表示し、[プロパティ(R)] を選択します。[Rocket Software] > [プロジェクト設定] > [COBOL] とナビゲートし、 [一般] > [ソース エンコーディング]を”UTF-8”から”ANSI”に変更し、[適用して閉じる] をクリックします。



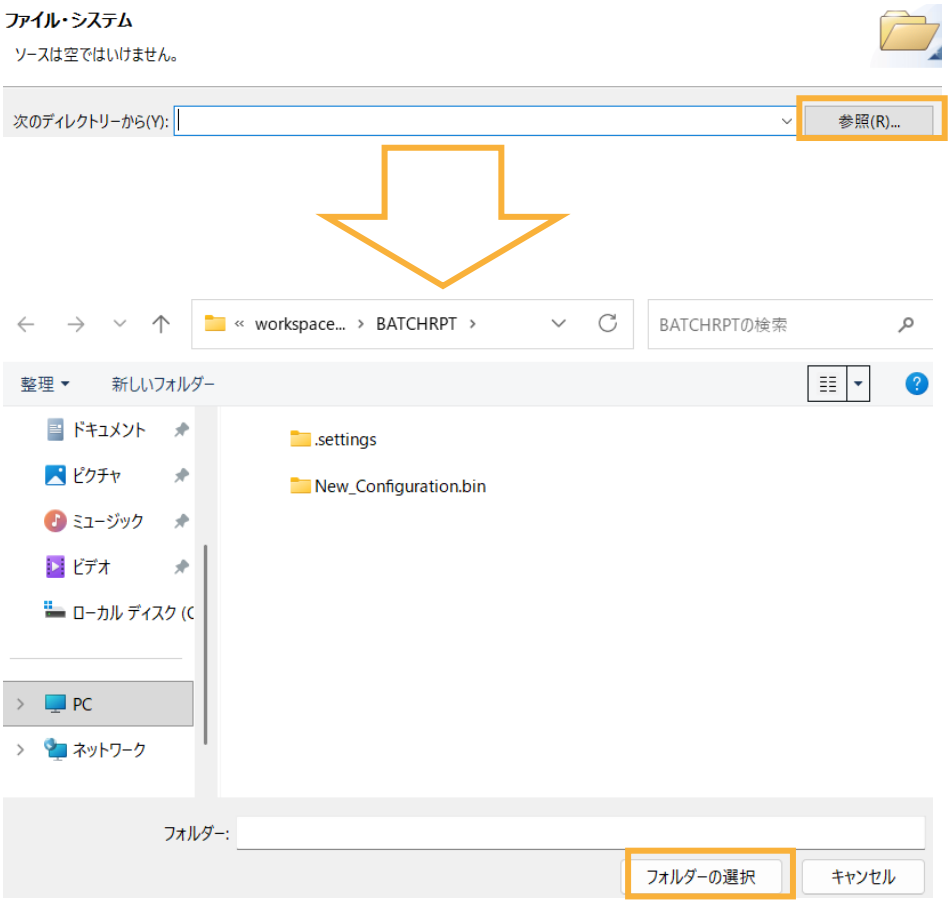
4) プロジェクトにリソースを追加します。

① プロジェクトを右クリックし、[インポート(I)] > [インポート(I)] を選択します。

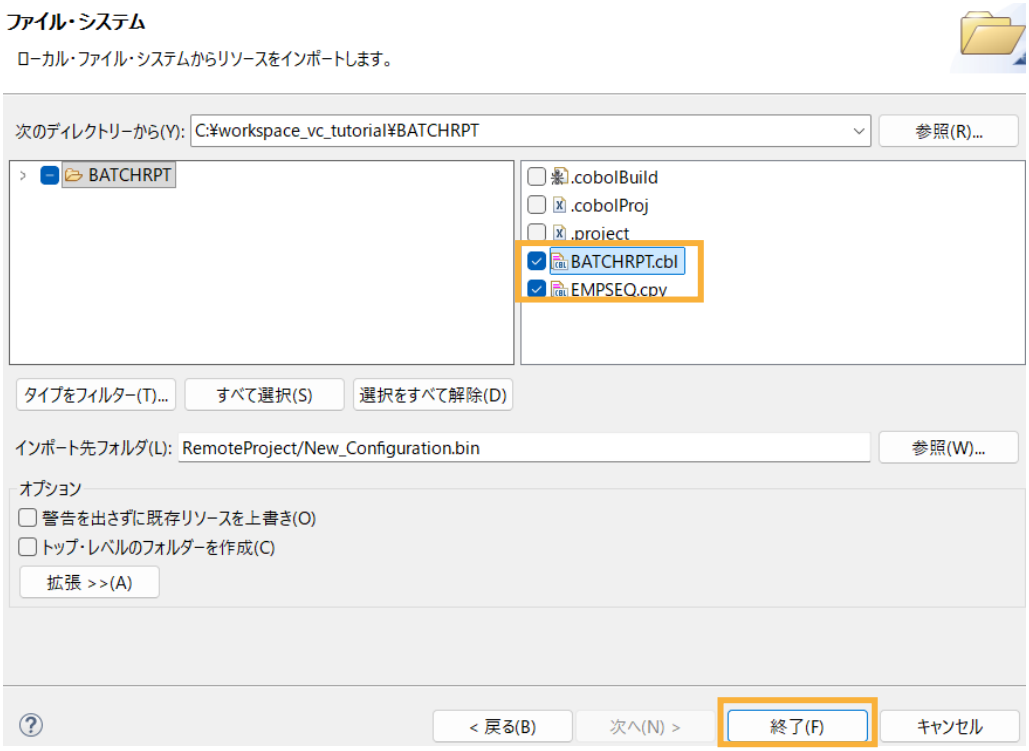
② [一般] > [ファイル・システム] を選択し [次へ(N)] をクリックします。



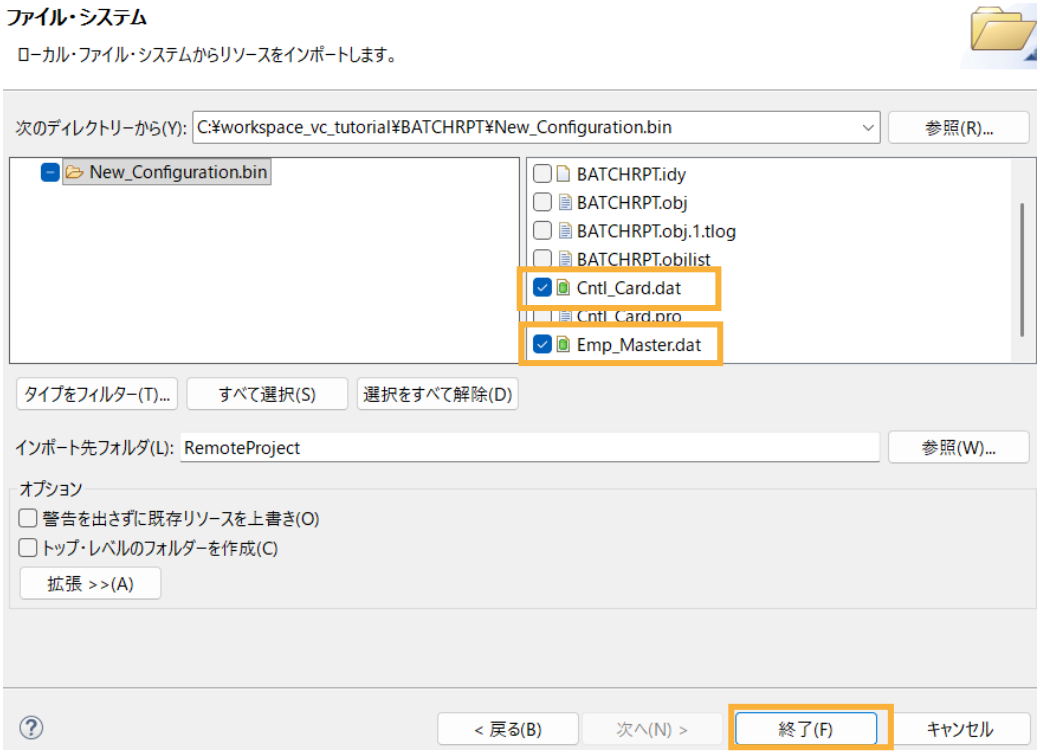
③ [参照(R)] をクリックし、ポップアップするエクスプローラーにて「Visual COBOL for Eclipse 自習書」で作成した [BATCHRPT] プロジェクトフォルダーを選択し [フォルダーの選択] をクリックします。



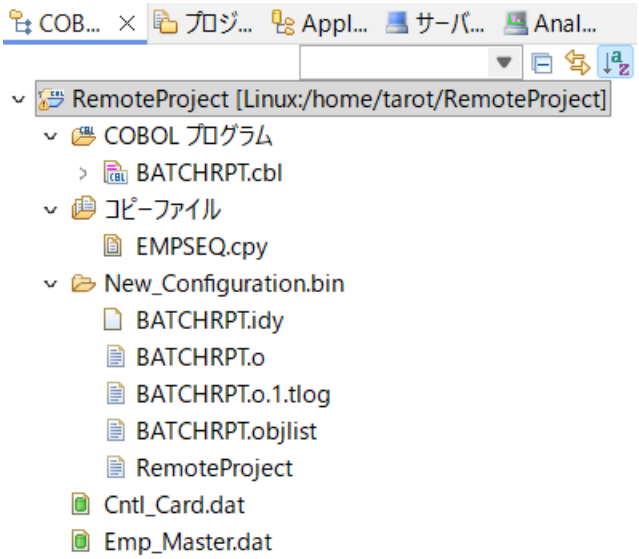
④ [BATCHRPT.cbl] 及び [EMPSEQ.cpy] にチェックを入れ、[終了(F)] をクリックします。



⑤ ③、④の要領で [Cntl_Card.dat] 及び [Emp_Master.dat] も BATCHRPT のプロジェクトフォルダー配下の New_Configuration.bin フォルダー下から COBOL リモートプロジェクトに追加します。



プロジェクト配下は、以下のようになります。



5) プロジェクト構成を設定します。

① COBOL エクスプローラーにて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。

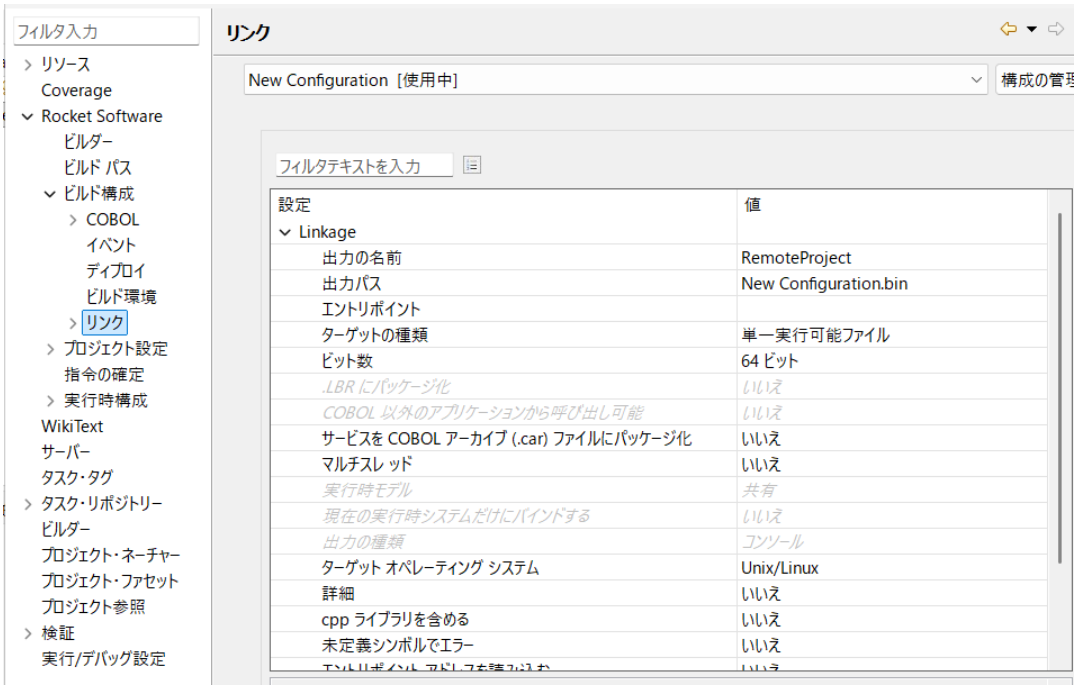
② [Rocket Software] > [ビルド構成] > [リンク] へとナビゲートします。

③ リンク設定を確認します。

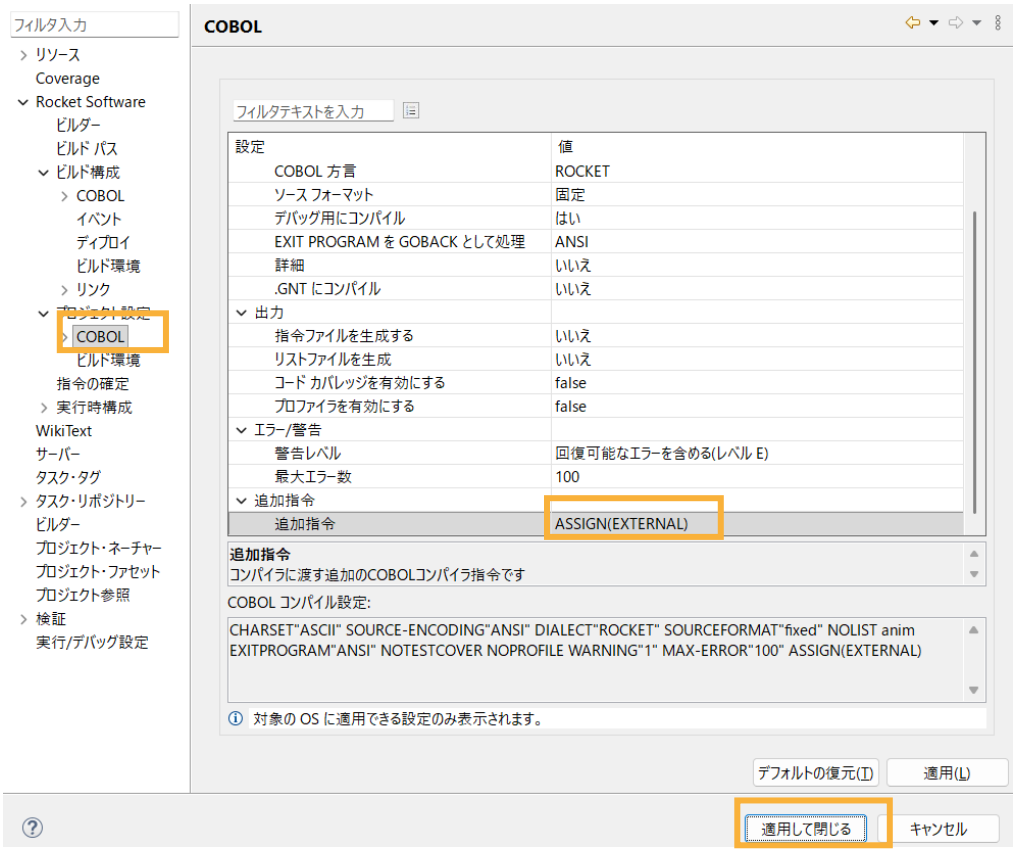
出力の名前： プロジェクト名と同名の RemoteProject としてモジュールが作成されます。

ターゲットの種類： プロジェクト中の COBOL プログラムを 1 つの実行形式に固めたモジュールが生成されます。

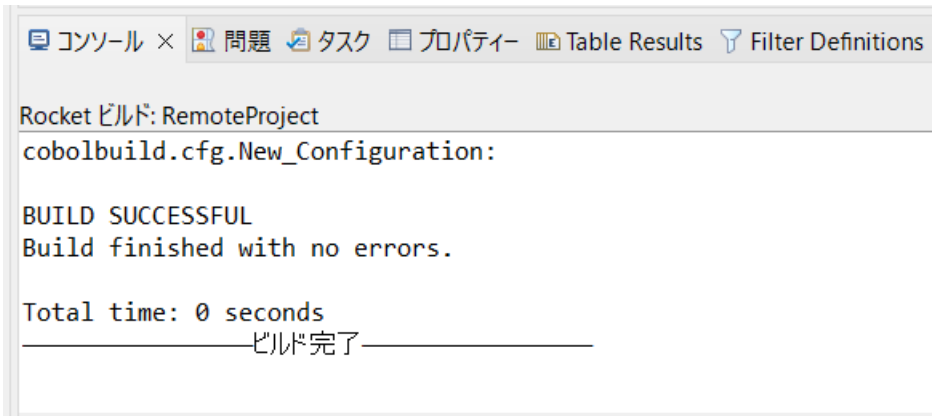
ビット数： 64-bit モジュールが生成されます。



④ [Rocket Software] > [プロジェクト設定] > [COBOL] を選択し、[追加指令] 欄に “ASSIGN(EXTERNAL)” を入力し [適用して閉じる] をクリックします。



コンソールビューでは、自動で行われたビルド結果が確認できます。



ターミナルソフトなどを用いて、Linux 側にファイルが作成されていることを確認してください。

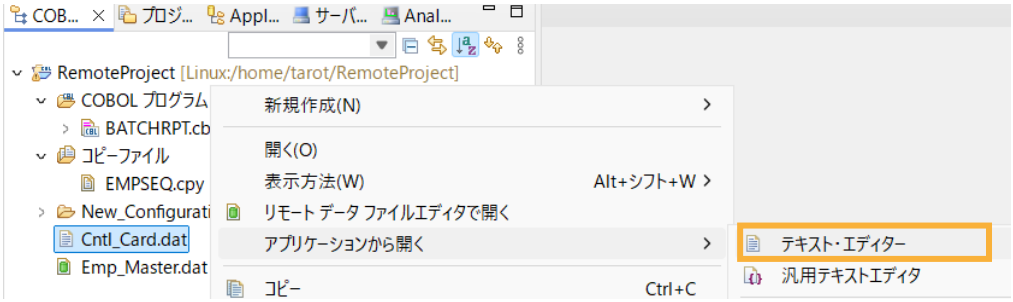
3.4 X サーバーの準備

リモート開発でデバッグする際、ACCEPT 文や DISPLAY 文によるコンソール入出力は X Window の技術を用いて、Windows 側に表示させます。そのため、リモート開発にてデバッグ／テスト実行する際は、Windows 側で X サーバーを起動する必要があります。商用製品から無償製品まで様々なツールを使用できますので、環境に合わせて製品を導入してください。

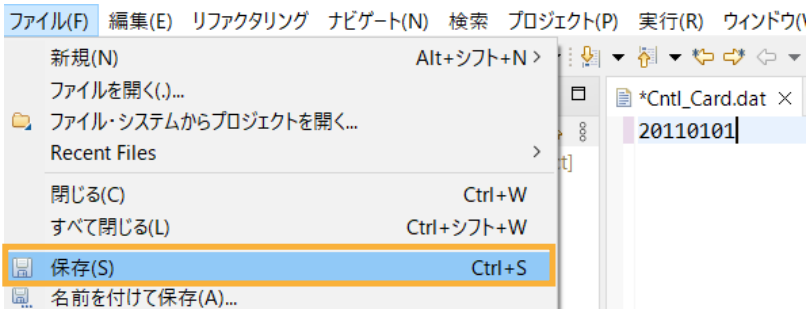
3.5 リモートデバッグ

ここまでの作業にて、Windows 上の Eclipse プロジェクトから直接 Linux/UNIX 側に実行形式を生成させました。本章ではこの生成されたモジュールを Linux/UNIX 上で実行させつつも Windows 上のデバッガでその処理を操作してみます。

1) Visual COBOL for Eclipse が閉じている場合は、起動し 3.3 で使用した Eclipse ワークスペースを開きます。
2) ご使用になる X サーバーを起動します。
3) 制御ファイルのメンテナンスをします。
   ① 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] > [テキスト・エディター] を選択します。

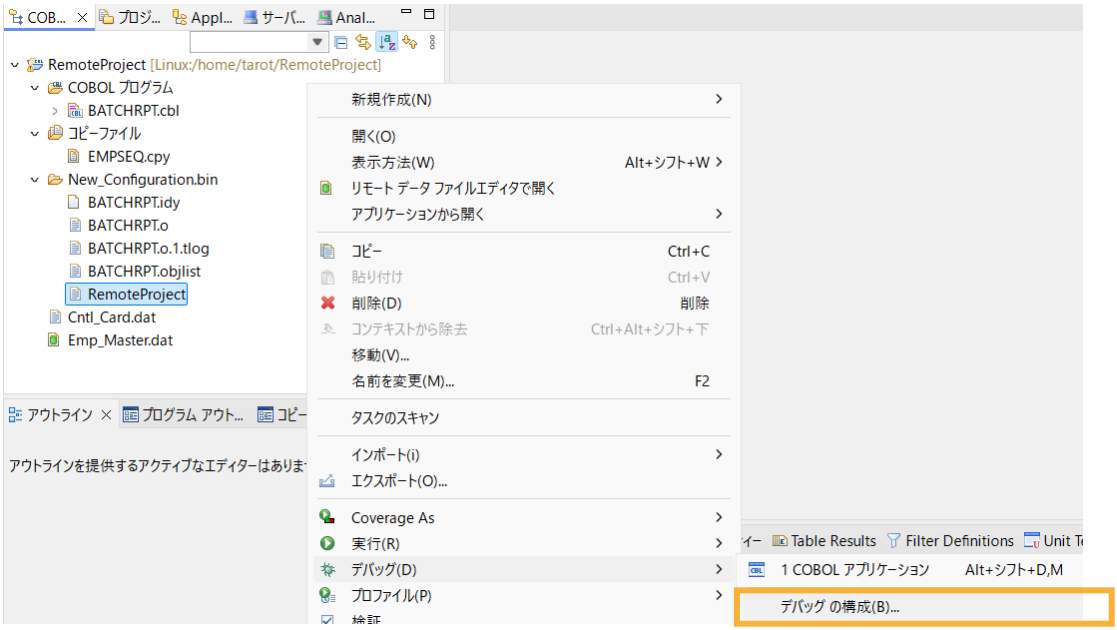


   ② “20110101” に変更し、[ファイル(F)] メニューから [保存(S)] を選択します。

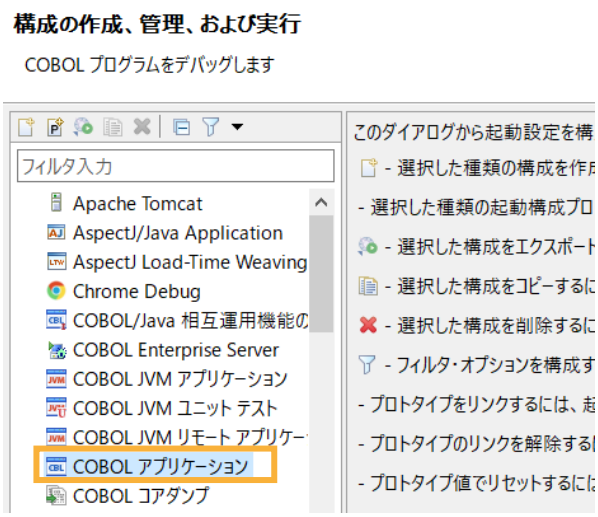


4) デバッグの構成の各種設定項目を指定します。

① [New_Configuration] 配下に生成されているプロジェクトと同名の実行形式を右クリックし [デバッグ(D)] > [デバッグの構成(B)] を選択します。



② [COBOL アプリケーション] をダブルクリックします。



③ 以下の入力を行い、[適用(Y)] をクリックします。

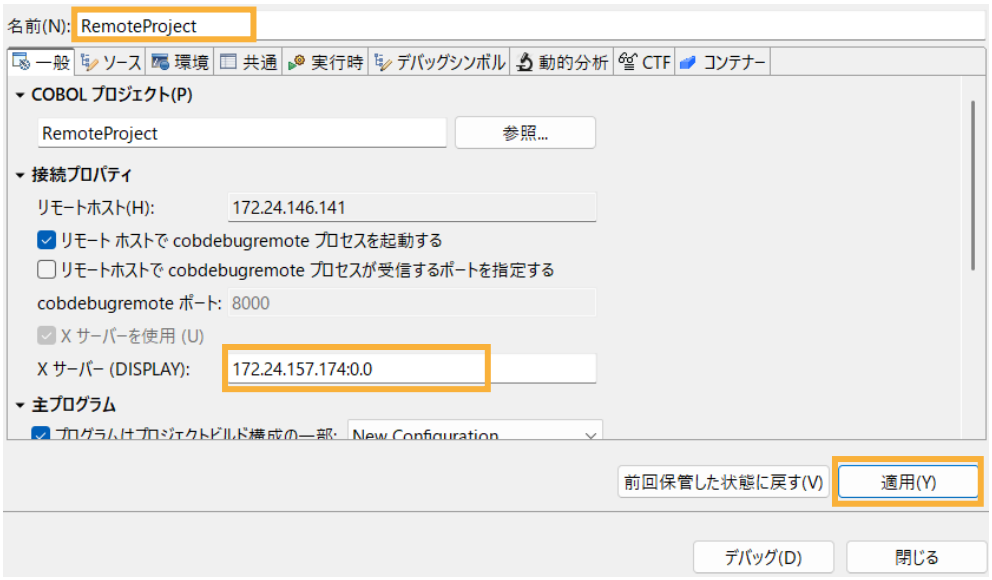
- 名前

  RemoteProject

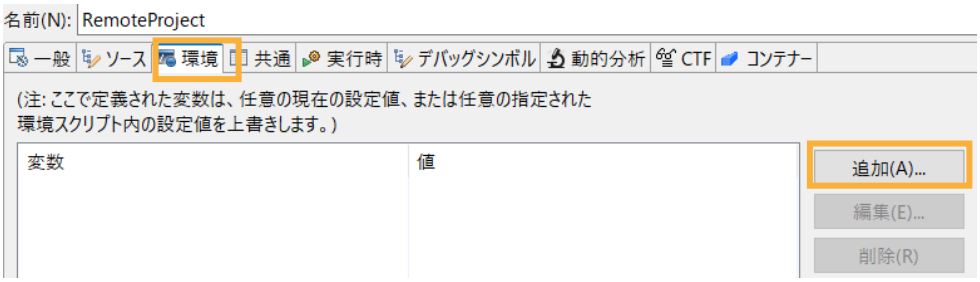
- X サーバー(DISPLAY)

  <Windows 側の IP> : <X サーバーのポート番号> の形式で入力します。

  Linux 側から Windows へ名前解決できる場合は、IP の部分をデフォルト値のホスト名にしても構いません。



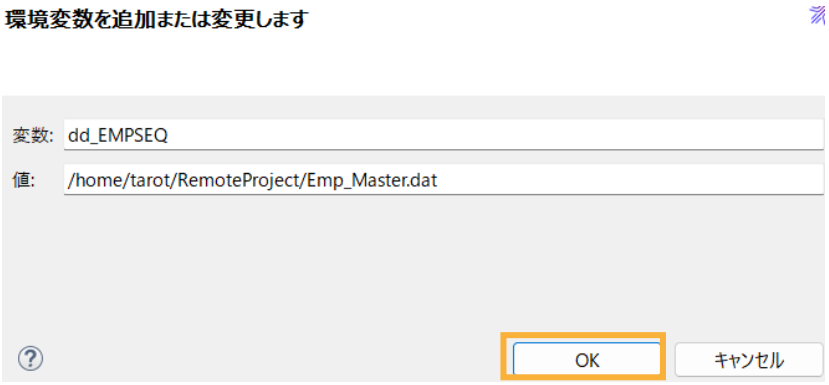
④ [環境] タブをクリックして、[追加(A)] をクリックします。



⑤ 下記のように入力し [OK] をクリックします。

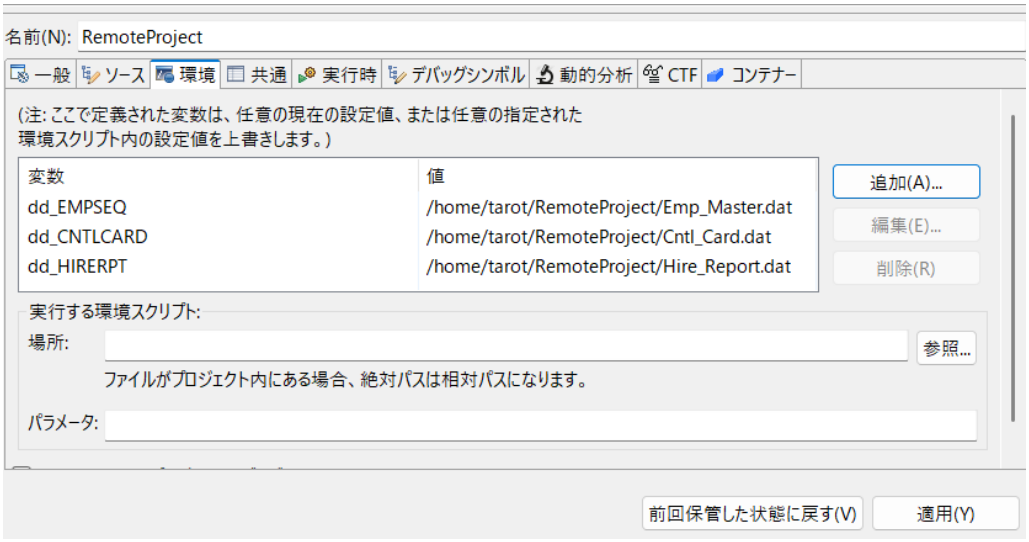
変数： dd_EMPSEQ

値： Emp_Master.dat までのフルパス



⑥ 同じ手順で以下の変数を追加します。

変数	値
dd_CNTLCARD	Cntl_Card.dat までのフルパス
dd_HIRERPT	<プロジェクトディレクトリ> /Hire_Report.dat

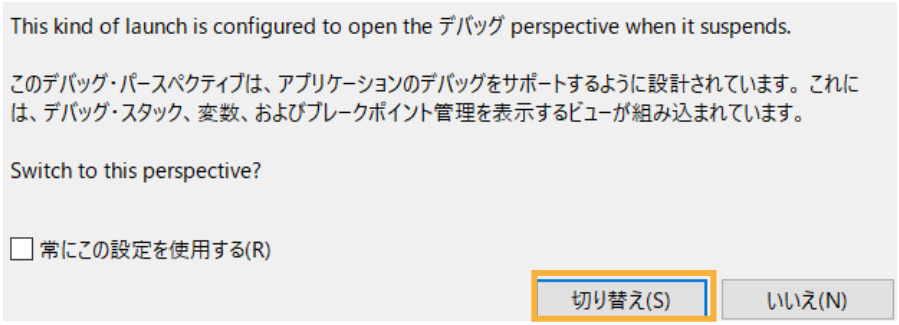


上記のようになっていることを確認のうえ、[適用(Y)] をクリックします。

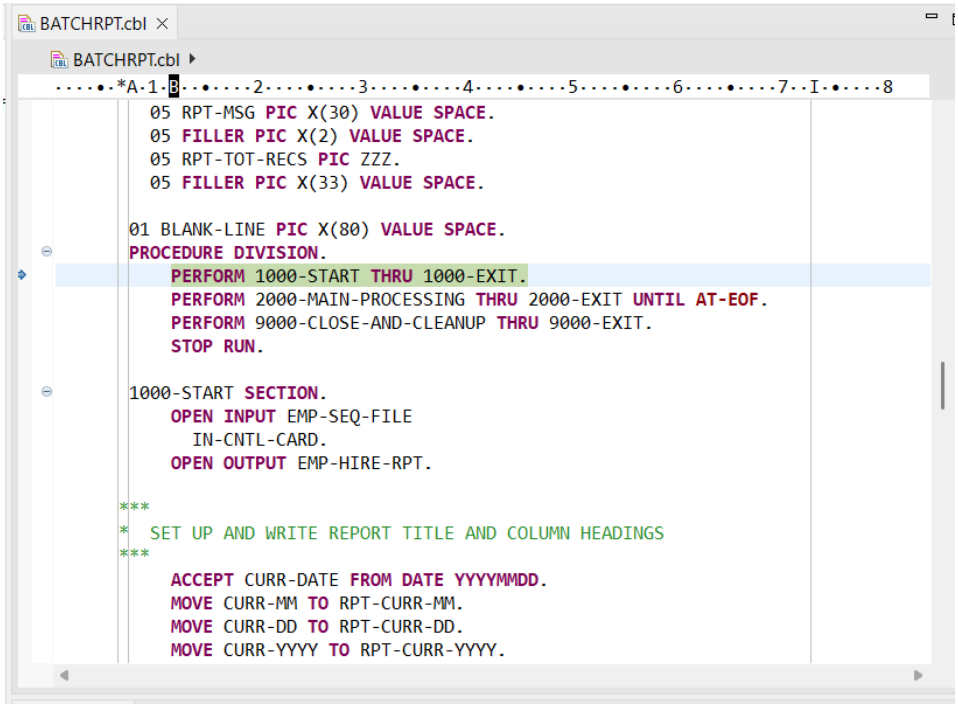
5) デバッグ実行を開始します。

前のステップで指定したデバッグ構成ウィンドウにて [デバッグ(D)] をクリックしデバッグ実行を開始します。

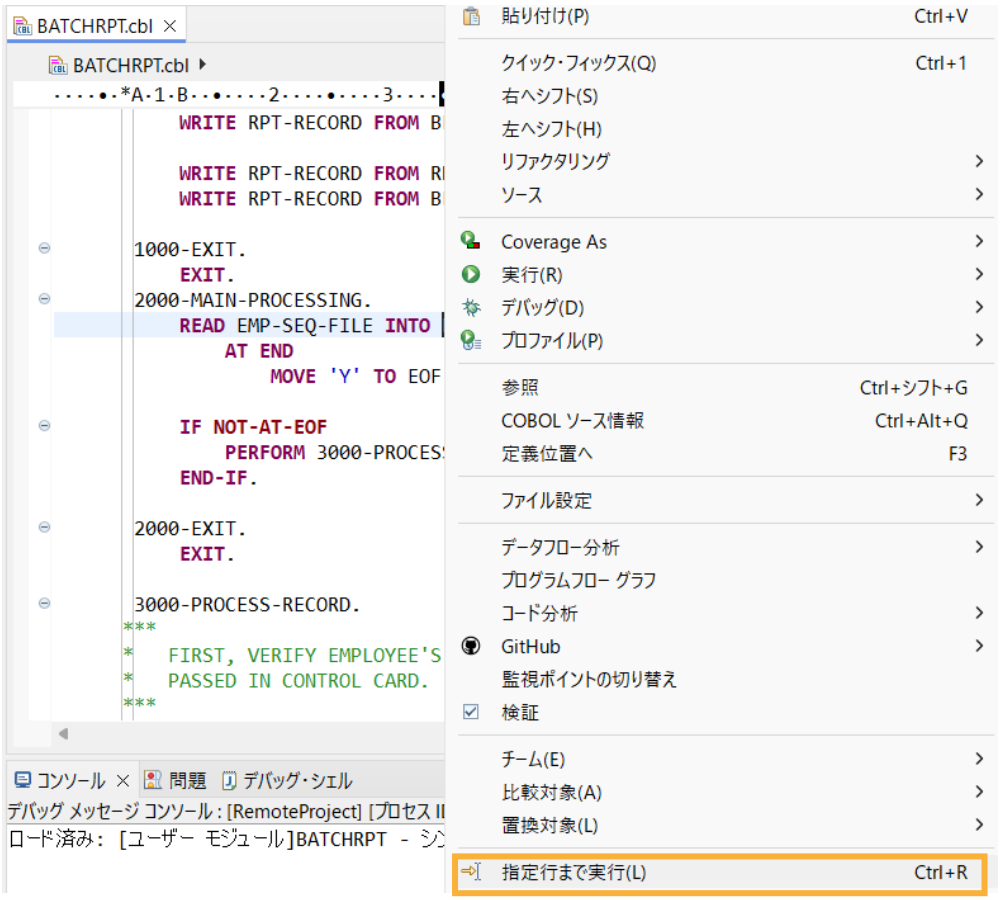
① パースペクティブの切り替えに関するメッセージに関しては [切り替え(S)] をクリックしてデバッグパースペクティブへ切り替えます。



最初の COBOL 行の実行前で処理が一時停止しています。

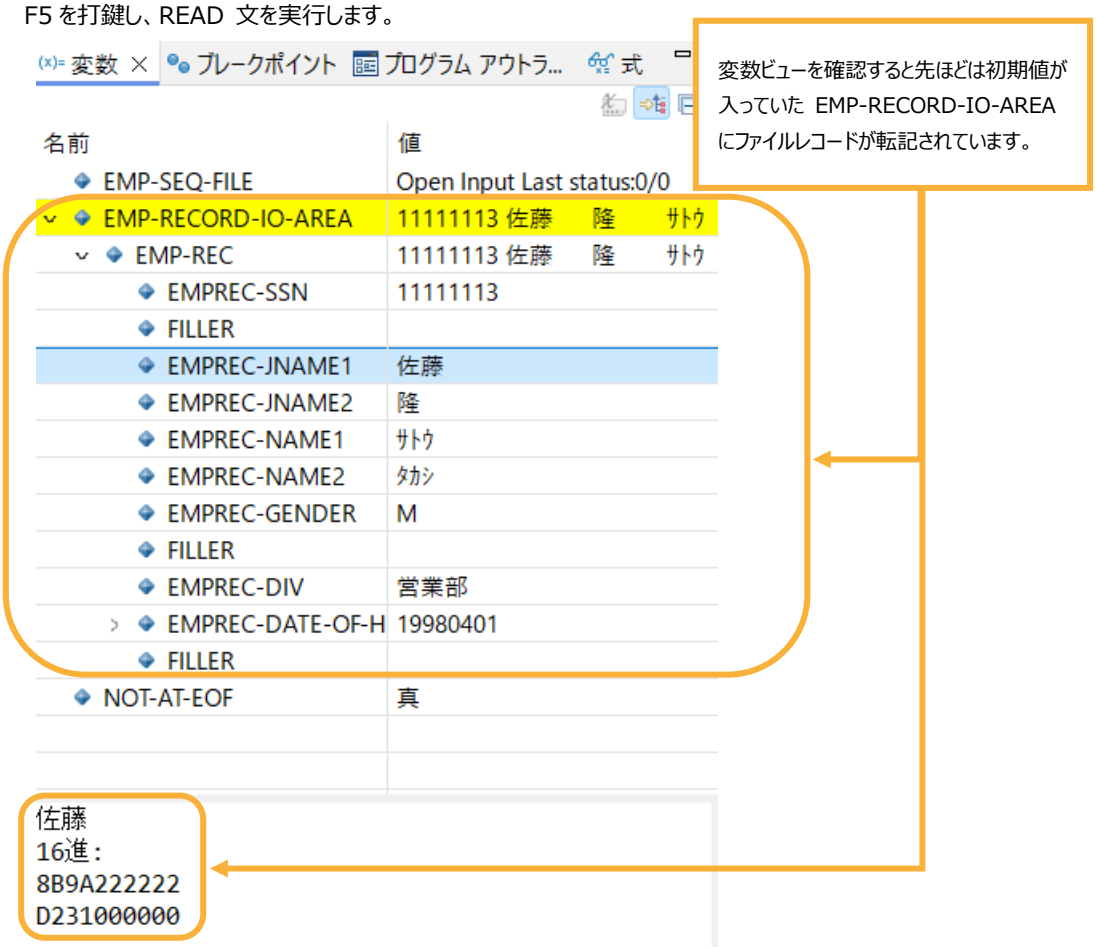


② [2000-MAIN-PROCESSING] 段落の最初の READ 文にカーソルを合わせ、右クリックから [指定行まで実行(L)] を選択します。

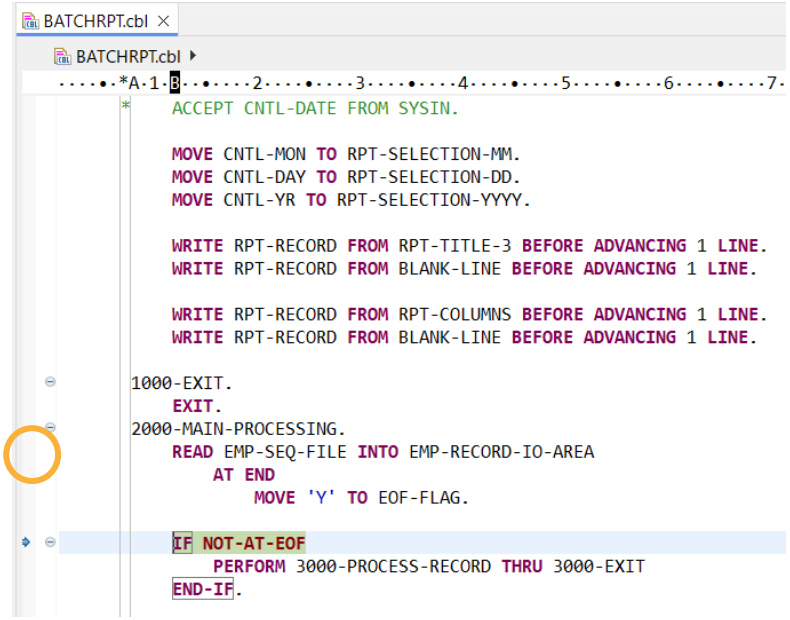


カーソル位置まで処理が進みます。

③ F5 を打鍵し、READ 文を実行します。



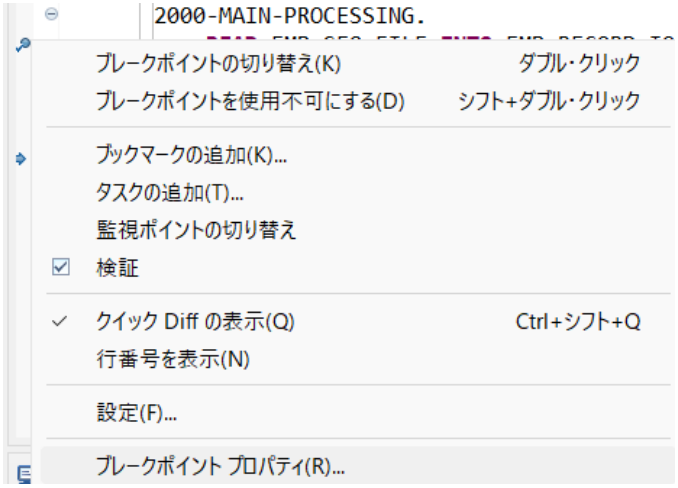
④ 条件付きブレークポイント機能を確認します。さきほどの READ 文に対して、〇で指定した左枠位置をダブルクリックして、ブレークポイントを設定します。

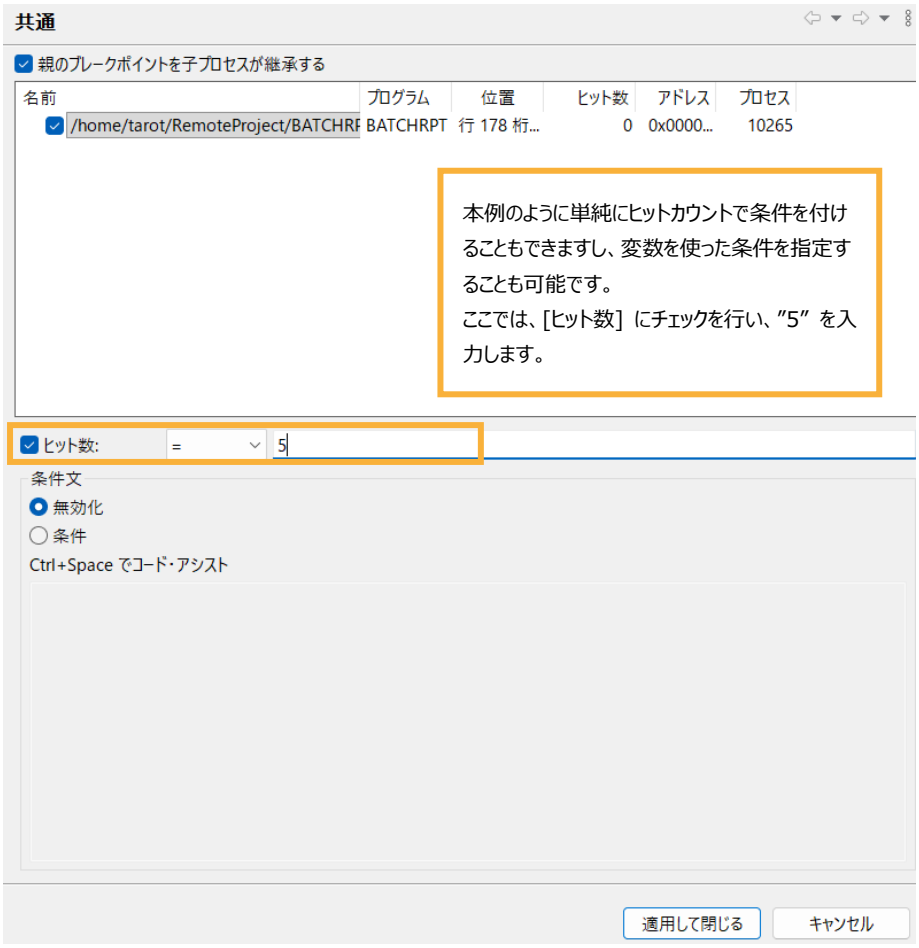


設定されると、アイコンが表示されます。

```
2000-MAIN-PROCESSING.
    READ EMP-SEQ-FILE INTO EMP-RECORD-IO-AREA
        AT END
            MOVE 'Y' TO EOF-FLAG.
```

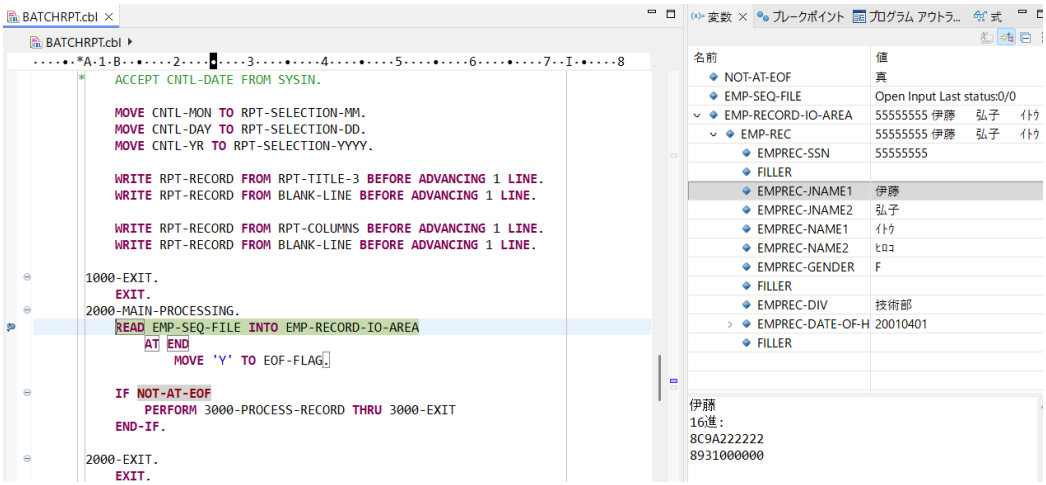
このアイコンにカーソルと合わせ、マウスの右クリックでコンテクストメニューをひらき、[ブレークポイント プロパティ(R)] を選択します。





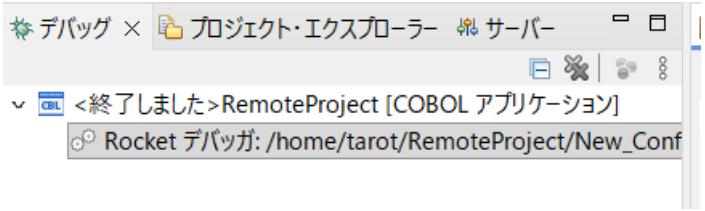
[適用して閉じる] をクリックします。

設定後、F8 を打鍵しますと、設定した直後から５回目の READ 文のヒットでデバッガが一時停止します。



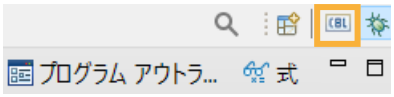
⑤ デバッガの動作が確認できましたら、F8 を打鍵しアプリケーションを最後まで実行します。

デバッガが終了した旨を [デバッグ] ビューより確認できます。



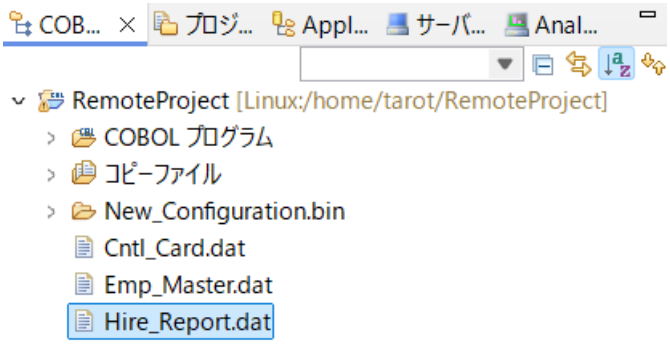
6) COBOL パースペクティブに戻します。

画面右上の COBOL パースペクティブアイコンを選択します。

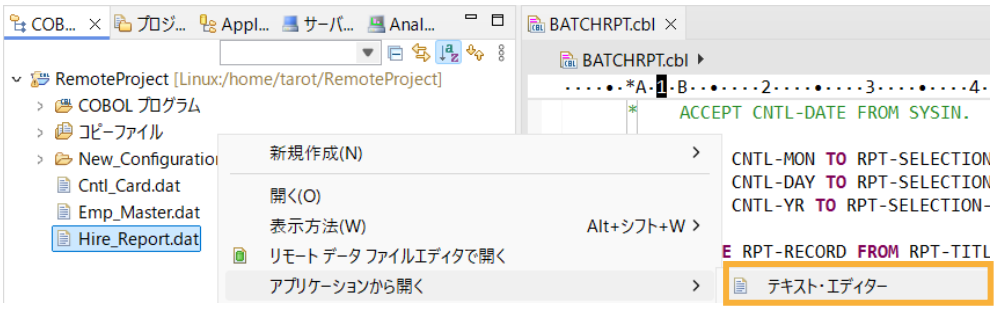


7) 生成された帳票を確認します。

① COBOL エクスプローラーにて Hire_Report.dat が生成されていることを確認します。確認できない場合は、RemoteProject プロジェクトを選択したうえで、F5 キーを押すか、[ファイル(F)] > [更新(F)] を選択します。



② COBOL エクスプローラー中の 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。



「Visual COBOL for Eclipse 自習書」で確認したのと同じ帳票が生成されていることが確認できます 。

以上で本チュートリアルは終了です。

チュートリルを終了する際、Eclipse はそのまま閉じていただいて構いません。

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。