

Visual COBOL チュートリアル

1

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 10, Rocky Linux 9.4
- Windows 上に Visual COBOL 10.0 for Eclipse がインストール済みであること
- Rocky Linux 9.4 上に Visual COBOL 10.0 Development Hub がインストール済みであること

本チュートリアルの実施前に、「Visual COBOL for Eclipse チュートリアル」を実施してください。



内容

- 1 目的
- 2 前提
- 3 チュートリアル手順
 - 3.1 環境のセットアップ
 - 3.1.1 Visual COBOL for Eclipse のセットアップ
 - 3.1.2 Visual COBOL Development Hub のセットアップ
 - 3.2 Development Hub のインストール確認
 - 3.3 COBOL リモートプロジェクトの作成
 - 3.4 X サーバーの準備
 - 3.5 リモートデバッグ



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_10.0_rocky_x86_64

6) インストール処理を開始します2

インストール処理を用いています。
./setup_visualcobol_devhub_10.0_rocky_x86_64 -installlocation=/opt/mf/VC100
-=-====================================
Micro Focus Product - Product Extractor
www.microfocus.com
Please Wait.
Performing Product and Platform Checks
All Checks Passed.
Extracting Payload
Creating work area
===============
Micro Focus Visual COBOL Development Hub 10.0
www.microfocus.com

¹ インストーラのファイル名は、「setup_visualcobol_devhub_10.0_<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/microfocus/VisualCOBOL」です。本例では
- 「-installlocation=/opt/microfocus/VC100」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/mf/VC100] 製品の構成エリア [/opt/microfocus/config/00037] 製品 : Micro Focus Visual COBOL Development Hub 10.0 ESadminID : Not given ビルド ID : pkg_358807 機能 : 32/64 bit, SOA, Remote Dev, AutoPass Licensing Java 最小バージョン : 17.0.0.0 インストール済み Java バージョン: OpenJDK 11.0.23.0 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_10.0_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/mf/VC100/etc/EULA_VCED_v10_0_jp.htm Micro Focus Visual COBOL Development Hub 10.0 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。 Micro Focus Visual COBOL Development Hub 10.0 インストールが完了しました。

4



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/mf/VC100/etc/EULA_VCED_v10_0.htm
このバージョンの次の製品を使用するには : Micro Focus Visual COBOL Development Hub 10.0
環境を設定するため、"cobsetenv"を実行してください。
. /opt/mf/VC100/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/mf/VC100/bin/cobsetenv
COBDIR set to /opt/mf/VC100

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。

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。





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。

- 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。
- 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。

ファイ	イル(F)	編集(E)	リファクタリング	ナビゲート(N	検索	プロシ	バンクト	∽(P) 実	行(R)	ウィンドウ(W)	ヘルプ(H)
	新規(N)		A	lt+シフト	+N >	2	COBOL	L プロジ	ェクト	
	ファイノ	レを開く(.)					1	COBOL	ו דצ"–־	ファイル プロジェイ	7ト
۵,	 ファイル・システムからプロジェクトを開く< ③ リモート COBOL プロジェクト 										
	Recei	nt Files				>	电	リモート	CORO	Lコヒーノアイル・	ノロシェクト

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)"
ሀቺ-Ւ COBOL プロジェクト
ワークスペースまたは外部にリモート COBOL プロジェクトを作成
プロジェクト名: RemoteProject ファイル システム ファイル システムを選択: リモート ファイル システム (RSE)
セキュアシェル (SSH) ファイル システムを使用すると、SSH 接続サポートのみを使用してリモート プロジェクトを処 理できます。ローカル ファイル システム上の場所を指定する必要はありませんが、リモート マシン上の場所のみ指 定する必要があります。
リモート ファイル システムの場合、RSE サポートによりリモート プロジェクトで作業できます。ローカル ファイルシステムの場所の指定は不要で、リモートマシン上の場所の指定だけが必要です。
ネットワークファイル システムの場合は、ローカルマシン上のプロジェクトの場所(マップされたドライブ上のプロジェクトパス)とリモートマシン上のパスを指定する必要があります。
(P) (N) > 終了(F) キャンセル

③ プロジェクトテンプレートは [Micro Focus 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。



<complex-block></complex-block>		ሀቺ-ト COBOL プロジェクト
ТУЗИН ТУЛИ-14 ЖИТ Image: Strate Strate Strate Image: Strate Str		ワークスペースまたは外部にリモート COBOL プロジェクトを作成
# Mana force 7,771 1/6 # 291 * Mana force 7,771 1/6 # 291 * C71 1-0082 # 281 * C71 1-0082		プロジェクト テンプレートを選択
<form></form>		²⁸ Micro Focus テンプレート [32 ビット] ²⁸ Micro Focus テンプレート [64 ビット]
WFF アバルジスアム花进形 Earle アバルジスアム花进形 Earle (2) EARle ET(1) (3) EARLE ERRE ERRE (4) EARLE ERRE ERRE ERRE (5) EARLE ERRE ERRE ERRE ERRE (4) EARLE ERRE ERRE </th <th></th> <th><u>テンプレートの設定を構成</u> 「テンプレートの参照</th>		<u>テンプレートの設定を構成</u> 「テンプレートの参照
・ (医46) (水ハN) (村下) (村ヤセル ・ (医緑感の新規作成] をグリックします。 ・ (医体のの新規作成] をグリックします。 ・ ワート COBOL プロラエクト ・ ワースペースまたは外部にリモート COBOL プロジェクトを作成 プロジェクト名: RemoteProject リモート設定 ま読る: びってい DevHub SSH 使用] を選択し、[次へ(N)] をグリックします。 ・ ワート・システム・タイプの選択 Micro Focus DevHub SSH 使用] を選択し、[次へ(N)] をグリックします。 ・ フォニム・タイプの選択 Micro Focus DevHub SSH 使用] を選択し、[SSH) プロトコルによるファイルへのアクセス ・ フォニム・タイプ ・ フォニム・タイプ ・ ア・「ーの起動とゼキュアシェル (SSH) プロトコルによるファイルへのアクセス ・ フォニム・タイプ ・ フォニム・タイプ ・ フォニム・タイプ アレート ・ ア・の ・ ア・の <th></th> <th>場所: 参照 ファイルシステムを選択: default ~</th>		場所: 参照 ファイルシステムを選択: default ~
 (9) [接続の新規作成] をクリックします。 リモート COBOL プロジェクト っクスペースまたは外部にリモート COBOL プロジェクトを作成		(?) 終了(F) キャンセル
リモート COBOL プロジェクト ワークパイ-スまたは外部にリモート COBOL プロジェクトを作成 プロジェクト名: プロジェクト名: アート・設定 漫歌の新規作成 使きの新規作成 (1) Croc Focus DevHub SSH 使用]を選択し、[次へ(N)]を分リックします。 シート・クステム・タイプの選択 Micro Focus DevHub SSH 使用]を選択し、[次へ(N)]を分リックしたする システム・タイプの選択 パレクカフ クスロートの「のにの DevHub パク5 経生」アシェル (SSH) プロトコルによるファイルへのアクセス システム・タイプ: パレクカフ (1) Micro Focus DevHub SSH 使用) (1) Micro Focus DevHub SSH 使用) (2) (1) (1) (2) (2) (2) (2) (2) (2) (2) (2) (2) (2	4	[接続の新規作成] をクリックします。
プロジカや: RemoteProject 受信・設定 支続の 支続の新規作成… (*)		リモート COBOL プロジェクト ワークスペースまたは外部にリモート COBOL プロジェクトを作成
せ 株会:		プロジェクト名: RemoteProject リモート設定
 (Micro Focus DevHub SSH 使用)を選択し、[次へ(N)]をクリックします。 シニート・システム・タイプの選択 Micro Focus DevHub - サーバーの起動とセキュアシェル (SSH) プロトコルによるファイルへのアクセス ジステム・タイプ フステム・タイプ フステム・タイプの選択 マーの 「Micro Focus DevHub (RSE 経由) Micro Focus DevHub SSH 使用 (Micro Focus DevHub SSH 使用) マ ーの マーの (Micro Focus DevHub SSH 使用) マーの マーの マーの (Micro Focus DevHub SSH 使用) (Micro Focus DevHub SSH 使用) マーの マーの (Micro Focus DevHub SSH 使用) (Micro Focus DevHub SSH 使用) (Micro Focus DevHub SSH 使用) (Micro Focus DevHub SSH 使用) 		接続名: ど 接続の新規作成
Micro Focus DevHub - サーバーの起動さゼキュ ゲッゴル (SSH) プロドコルによる ガイ1 小のグウゼス システム・タイプ: フルタ入力	5	[Micro Focus DevHub SSH 使用] を選択し、[次へ(N)] をクリックします。 リモート・システム・タイプの選択
9254-947: フルタ入力 ● 一般 ● Micro Focus DevHub (RSE 経由) ● Micro Focus DevHub SSH 使用 ● Cous DevHub SSH 使用 ● Cous DevHub SSH 使用 ● Cous DevHub SSH 使用 ● Micro Focus DevHub SSH 使用		Micro Focus DevHub - サーバーの起動とゼキュアシェル (SSH) フロトコルによるファイルへのアクセス
ア ● 一般 ● Micro Focus DevHub (RSF 経由) ● Micro Focus DevHub SSH 使用 ● Micro Focus DevHub SSH 使用 ② < 戻る(B) 文へ(N) > 終了(f) キャンセル		システム・タイプ:
? < 戻る(B) 次へ(N) > 終了(F) キャンセル 補足)		▼ ➢ 一般 Micro Focus DevHub (RSF 经由) Micro Focus DevHub SSH 使用
 ? < 戻る(B) 次へ(N) > 終了(F) キャンセル 補足) 		
? < 戻る(B) 次へ(N) > 終了(F) キャンセル 補足)		
補足)		? < 戻る(B) 次へ(N) > 終了(F) キャンセル
	補足	<u>}</u>)

© Rocket Software, Inc. or its affiliates 1990–2024. All rights reserved. Rocket and the Rocket Software logos are registered trademarks of Rocket Software, Inc. Other product and service names might be trademarks of Rocket Software or its affiliates.

9



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/microfocus/VC100/bin/cobsetenv

COBDIR set to /opt/microfocus/VC100

3. デーモンを起動します。

\$COBDIR/remotedev/startrdodaemon

Starting RSE daemon...

Reading "/opt/mf/VC100/remotedev/rdo.cfg" ...

Daemon port loaded from "/opt/mf/VC100/remotedev/rdo.cfg": 4075

Server port range loaded from "/opt/mf/VC100/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。

欄の値か」ビーされます。指定が終わりましたら [次へ(N)] をクリックします。						
リモート 1 システム 接続情報の定義	接続(Micro Focu	s DevHub SSH 使)	用)			
親プロファイル:	win10-v-na			~		
ホスト名:	rocky9-v-na			~		
接続名:	rocky9-v-na					
記述/説明:						
□ ホスト名を検証 プロキシー設定を構成	2 4					
?	< 戻る(B)	次へ(N) >	終了(F)	キャンセル		
-						

- 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/mf/VC100/VisualCOBOL/remotedev/startrdoserver \${port} " & になります。



プロセス

サブシステム情報の定義

eclipse.devhub.pro	プロパティ	1+	
		但	
	SSH X11 転送を使用	true	
	SSH を介したトンネル	i true	
	サーバー ポート。 コマン	F 0	
>	サーバー起動コマンド	sh -c "/opt/mf/	VC100/remoted
	ランチャー	SSH	
ゲービス]		
tor Service			
一の起動			
プロパティー			
]		
, moneyabeyenvis		productyremoted	ievy start ruoserv
	F		
< 戻る(B)	次へ(N) >	終了(F)	キャンセル
inct			
iject			
ject		~ 接続の新規作	F成
ject		> 接続の新規作	■成
ject バのプロジェクト パスに1	没定しなければいけません。	✓ 接続の新規作	F成
ject ハンのブロジェクト パスに 発開 アイコンをクリ	设定しなければいけません。 ックレノます。	 接続の新規们 する 	F成
iject バンのプロジェクト パスに打 展開アイコンをクリ	設定しなければいけません。 ッ クします 。	✓ 接続の新規作 > ▲ 考	F成 ∲照
jject バのプロジェクト パスに 展開アイコンをクリ	設定しなければいけません。 ックします。	◇ 接続の新規1	F成 ▶照
ject バンのプロジェクト パスに 展開アイコンをクリ	设定しなければいけません。 ックします。	 接続の新規们 	F成
iject バンのプロジェクト パスに訂 展開アイコンをクリ	設定しなければいけません。 ックします。	- ✓ 接続の新規们 - ✓ ▲ 参	F成 }照
iject バッのブロジェクト パスに 軽開アイコンをクリ	设定しなければいけません。 ックします。	✓ 接続の新規们 > ▲ 考	F成 ▶照
	ービス tor Service ーの起動 プロパティー -を起動する方法を打 . /home/abc/env.s く 戻る(B) ます。 -や COBOL プロジェ	ビス tor Service の起動 プロパティー -を起動する方法を指定します。初期化スクリプ -を起動する方法を指定します。初期化スクリプ - /home/abc/env.sh && /opt/microfocus/ / /opt/microfocus/ ます。 クト モート COBOL プロジェクトを作成	ビス tor Service の起動 プロパティー -を起動する方法を指定します。初期化スクリプトを実行するには、 ・/home/abc/env.sh && /opt/microfocus/product/remoted < 戻る(B)

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。



システム・タイプ:	Micro Focus DevHub SSH 使用
ホスト名:	ROCKY9-V-NA
接続名:	rocky9-v-na
ユーザー ID:	tarot
パスワード(任意)(B):	*****
	✓ ユーザー ID の保管
	☑ パスワードを保管(C)
	OK キャンセル(A)

- ① 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。	
フォルダーの選択	
/home/tarot/RemoteProject	
▼ 巻 マイ・ホーム ▶ □ RemoteProject > □ work > ⇒ ルート	
Et (T)	+++++++(P)
OK 詳細(A) >>	キャンセル(B)

⑬ [終了(F)] をクリックします。



リモートの	ርዕBOL プロジェクト
ワークスペ	ースまたは外部にリモート COBOL プロジェクトを作成
プロミディク	k / RomotoDroject
	P治: RemoteProject
リモート訪	
按航名:	rocky9-v-na // 技術の初況TFA
リモートの	/home/tarot/RemoteProject ~ ▲ 参照
リモートの	場所はリモート マシンのプロジェクト パスに設定しなければいけません。
?	< 戻る(<u>B</u>) 次へ(<u>N</u>) > 終了(<u>F</u>) キャンセル
プロミジェノ	7.トが作成されます
/ 1/1/	
Score	3 × 🔁 プロ 😤 Appl 📇 サーバ 🛄 Anal 🗖 🗖
	V 🖹 🔁 📴 🍫
> 😕 F	RemoteProject [rocky9-v-na:/home/tarot/RemoteProject]

- 3) 文字コードの指定を行います。
 - 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。

フィルタ入力	ワークスペース	← ▼ ⇒ <
✓ 一般 Capabilities	ワークスペースの開始およびシャットダウン設定については、 <u>開始およびシャットダウン</u> を参考	照してください。
Schema Associ	□ ネイティブのフックまたはポーリングを使用して更新(R)	
> Security	✓ アクセス時に更新(S)	
UI JJ-X·T-9		
ン User Storage St Web ブラウザ		
、 Tディタ		
+-		
ウイック検索	77757009150	
グローバル化	✓ ワークスペース名を表示(E): workspace-rdev	
コンテンツ・タイプ	□ パースペクティブ名を表示(T)	
サービス・ポリシー	 ワークスペースのフルパスを表示(F): C:¥workspace-rdev 	
トレース	☑ プロダクト名を表示	
> ネットワーク接続		
ハンドラーをリンク		
パースペクティブ	ノロジェクトを開く際に、参照するノロジェクトを開く: ノロンノト 🗸	
プロジェクト・ネーラ	不明なプロジェクトの性質を以下のように報告(A): 警告 🗸	
> ワークスペース	Report missing project encoding as: 弊告	
> 開始およびシャッ		
〉外観		
使采	システム・エクスプローラーを起動するコマンド(X): explorer /E,/select=\${selected_reso	urce_loc}
迪和 比較 パッチ		
ΣL=x/////	テキスト・ファイル・エンコード(T) 新規テキスト・ファイルの行区切	り文字(F)
Aspect Compiler	● デフォルト(U) (windows-31i)	
> CSS (Wild Web De	$\bigcirc Z \oplus (t)$, windows 31 \lor	
> Gradle		1
> HTML (Wild Web I 🗡	デフォルトの復元(T)	適用(1)
< >	77777707875(1)	~(I)(E)
		キャンクル
	適用して閉しる	TYJUN

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。[Micro Focus] > [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)

フィルタ入力	COBOL	÷ ج	-
> リソース			
Coverage			
 Micro Focus 	フィルタテキストを入力		
ビルダー			
ビルド パス	設定	值	
> ビルド構成	∨ 一般		
✓ プロジェクト設定	文字セット	ASCII	
> COBOL	ソース エンコーディング	ANSI	
ビルド環境	COBOL 方言	Micro Focus	
指令の確定	ソース フォーマット	固定	
> 実行時構成	デバッグ用にコンパイル	はい	
WikiText	EXIT PROGRAM を GOBACK として処理	ANSI	
サーバー	詳細	いいえ	
タスク・タグ	.GNT にコンパイル	いいえ	
> タスク・リポジトリー	✓ 出力		
ビルダー	指令ファイルを生成する	いいえ	
プロジェクト・ネーチャー	リストファイルを生成	いいえ	
プロジェクト・ファセット	コード カバレッジを有効にする	false	
プロジェクト参照	プロファイラを有効にする	false	
> 検証	✓ Iラ-/警告		
実行/デバッグ設定	警告レベル	回復可能なエラーを含める(レベル E)	
	<mark>ソース エンコーディング</mark> SOURCE-ENCODING はソース プログラムのエンコ	コーディングをコンパイラに渡します。その後、RUNTIME-ENCODING 指	
	COBOL コンパイル設定:		
	CHARSET"ASCII" SOURCE-ENCODING"ANS EXITPROGRAM"ANSI" NOTESTCOVER NOP	I" DIALECT"MF" SOURCEFORMAT"fixed" NOLIST anim ROFILE WARNING"1" MAX-ERROR"100"	
		デフォルトの復元(I) 通	፤用(<u>L</u>
?		適用して閉じるキャ	ンセル
オ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。

 E 5 	2 %	8				
RemoteProject [rocky9-v-na:/home/tarot/RemotePr	oject		新規作成(N)	>	1	
			表示方法(W)	Alt+シフト+W >		
		Ð	כול ביו	Ctrl+C		
		Ē	貼り付け	Ctrl+V		
		×	削除(D)	削除		
		<u>.</u>	コンテキストから除去	Ctrl+Alt+シノト+ ト		
			移動(V)	53		
			石削を変更(IVI)	F2		
			タスクのスキャン			
			コード分析	>		
			インボート(i)	>	Ŷ,	。 リモート プロジェクト
			エクスポート(O)		8	↓ ローカル Micro Focus プロジェクトのリモート プロジェクトへの変
		8	更新(F)	F5	Ē	Net Express プロジェクトの変換
E アウトライン × ፼ プログラム アウト ፼ コピーファイル従	-		プロジェクトを閉じる(S)			インボート(I)
	69		無関係なプロジェクトを閉じる(U)			

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。



選択 ローカル・ファイル・シン	ステムから既存のプロジェク	フトヘリソースをインポート	します。	Ľ
インポート・ウィザード	の選択(S):			
フィルタ入力				
 ◇ >> 一般 ○ アーカイブ ○ ファイル・: ○ フォルター ☆ 既存プロ・ □ 設定 > >> EJB 	・ファイル システム またはアーカイブ由来のプ ジェクトをワークスペースへ	ΟΫΙクト		^
> 🗁 Git > 🗁 Gradle				~
?	< 戻る(B)	次へ(N) >	終了(F)	キャンセル

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。

ファイル ソースは	・システム 空ではいけません。			
次のディ	レクトリーから(Y):	7	~	参照(R)
	workspace_vc_tutorial > BATCHRPT >	BATCHRPTの 検	索 2	
いフォルら	1-		:== ▼ ?	
d	^ 名前 [^]	更新日時	種類	
BOL	.settings	2024/08/02 14:13	ファイル フォルダー	
に こ プ	New_Configuration.bin	2024/08/02 15:51	ファイル フォルター	
,				
イスク (C	× <			
フォル	Ø-:	フォルダーの選択	キャンセル	

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。



ファイル・システム

ファイル・システム ローカル・ファイル・システムからリソースをインポートします。			
次のディレクトリーから(Y): C:¥workspace_vc_tutorial¥BA	ATCHRPT	~	参照(R)
> 🔳 🗁 BATCHRPT	 		
タイプをフィルター(T) すべて選択(S) 選択を	すべて解除(D)		
インボート先フォルダ(L): RemoteProject			参照(W)
オプション ■ 警告を出さずに既存リソースを上書き(O) ■ トップ・レベルのフォルダーを作成(C) 拡張 >>(A)			
?	< 戻る(B) 次へ(N) >	終了(F)	キャンセル

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。

ー ファイル・システム		
ローカル・ファイル・システムからリソースをインポートします。		
次のディレクトリーから(Y): C:¥workspace_vc_tutorial¥BAT	CHRPT¥New_Configuration.bin ~	参照(R)
■ ➢ New_Configuration.bin Øイブをフィルター(T) すべて選択(S) 選択をすへ	□ BATCHAPT.objlist □ BATCHRPT.exe □ O Cntl_Card.dat □ Cntl_Card.pro □ B Emp_Master.dat □ O Hire_Report.dat □ New_Configuration.gcf	~
インポート先フォルダ(L): RemoteProject オブション		参照(W)
0	< 戻る(B) 次へ(N) > 終了(F)	キャンセル

プロジェクト配下は、以下のようになります。



- COB... ×
 プロ...
 2 Appl...
 サーバ...
 Anal...
 ロ

 Appl...
 サーバ...
 Anal...
 ロ

 <td
- 5) プロジェクト構成を設定します。
 - ① COBOL エクスプローラ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 [Micro Focus] > [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。

フィルタ入力	リンク		⇔ ◄ ⇔ ◄			
> リソース	New Carlingerting (/t====1)		# + om			
Coverage	New Configuration [使用中] V					
 Micro Focus 						
ビルダー						
ビルド パス	フィルタテキストを入力					
∨ ビルド構成	· 設定	値	^			
> COBOL	k Linkage					
イベント	・ 出力の名前	RemoteProject				
ディプロイ	出力パス	New Configuration bi	n			
ビルド環境	エントリポイント		· · · · · · · · · · · · · · · · · · ·			
> リンク	ターゲットの種類	単一実行可能ファイル				
> フロジェクト設定	ビット数	64 ビット				
指令の確定	.LBR にパッケージ化	ししえ				
> 実行時構成	COBOL 以外のアプリケーションから呼び出	ししえ				
WIKIIEXt	サービスを COBOL アーカイブ (.car) ファイル	いいえ				
	マルチスレ ッド	いいえ				
クスク・タク	実行時モデル	共有				
> ダスク・ワルントリー	現在の実行時システムだけにバインドする	いいえ				
Lルツー プロジェクト・ウーチャー	出力の種類	コンソール				
プロジェクト・ファセット	ターゲット オペレーティング システム	Unix/Linux				
プロジェンド・ファビット プロジェクト参照	詳細	いいえ				
ノロシェクト参照	cpp ライブラリを含める	いいえ				
✓ (突亜 宇行/デバッグ設定	未定義シンボルでエラー	いいえ				
天11/7/1971以進	エントリポイント アドレスを読み込む	いいえ	~			
			-			

 ④ [Micro Focus] > [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。



フィルタ入力	COBOL	← < ⇒
> リソース		
Coverage		
 Micro Focus 	フィルタテキフトを入力	
ビルダー		
ビルド パス	設定	值 /
✔ ビルド構成	COBOL 方言	Micro Focus
> COBOL	ソース フォーマット	固定
イベント	デバッグ用にコンパイル	はい
ディプロイ	EXIT PROGRAM を GOBACK として処理	ANSI
ビルド環境	詳細	いいえ
> リンク	.GNT にコンパイル	いいえ
✔ プロジェクト設定	▼ 出力	
> COBOL	指令ファイルを生成する	いいえ
ヒルド環境	リストファイルを生成	いいえ
指令の確定	コード カバレッジを有効にする	false
> 実行時構成	プロファイラを有効にする	false
WikiText	✓ Iラ-/警告	
サーバー	警告レベル	回復可能なエラーを含める(レベル E)
タスク・タグ	最大エラー数	100
タスク・リポジトリー	✔ 追加指令	
ビルダー	追加指令	ASSIGN(EXTERNAL)
プロジェクト・ネーチャー		
プロジェクト・ファセット	追加指令	1
プロジェクト参照	コンパイラに渡す追加のCOBOLコンパイラ指令です	· · · · · · · · · · · · · · · · · · ·
検証	COBOL コンパイル設定:	
実行/デバッグ設定	CHARSET"ASCII" SOURCE-ENCODING"ANSI"	DIALECT"MF"
	NOPROFILE WARNING"1" MAX-ERROR"100"	ASSIGN(EXTERNAL)
		デフォルトの復元(<u>I</u>) 適用(<u>L</u>)
3		

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。

🖳 איעב 🖳 X	🖹 問題	🔊 タスク	🗖 プロパティー	🗈 Table Results
Micro Focus ビル	ት: Remo	teProject		
cobolbuild.	cfg.New	_Config	guration:	
BUILD SUCCES Build finis	SSFUL hed wit	:h no er	rors.	
Total time:	0 seco	onds Iド完了—		

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。

Rocket software

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。そのため Reflection Desktop for X という X サーバーも併せて配備する必要があります。Windows 端末上に既に他の X サーバーソフトをインストールしていればそれを利用することも可能です。未インストールの場合はこの Reflection Desktop for X をインストールしてリモート開発時に利用してください。

Reflection Desktop for X のインストーラはリリースノートに記載されておりますのでそちらを参照してください。

※「VcXsrv」と呼ばれる無償のXサーバーソフトも市場にあるのでそれを使うことも可能です。

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0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。

「Visual COBOL for Eclipse 自習書」では最終的に該当する社員情報が見つからなくなるようメンテナンスしました。ここでは初期値に戻し検索条件を有効にします。

① COBOL エクスプローラにて [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。

COB ×	& Appl 📕 サーバ 🖳 Anal 📮 🗖		
> 🗁 New_Configurat	新規作成(N)	>	
Cntl_Card.dat Emp_Master.dat 	開く(O) 表示方法(W)	Alt+シフト+W >	
	アプリケーションから開く	>	デキスト・エディター
		Ctrl+C	 ・汎用テキストエディタ ・ハステナ・エディター(S)

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。

ファイ	イル(F)	編集(E)	リファクタリング	ナビゲート(N)	検索	プロジ	ェクト(F))	実行(R)	ウィンド)ל
۵,	新規(ファイ) ファイ) Recer	N) レを開く(.) レ・システム: nt Files	からプロジェクトを	Ali 開く	t+シフト	+N > >	• : ♀ □ > 8 t]	•		rd.dat > 01	< <
	閉じる すべて	(C) 閉じる(L)		Ctrl	Ctrl +シフト	+W +W					
	保存(S)			Ctr	l+S					
8	名前を	付けて保ィ	存(A)								

Rocket software

- 4) デバッグの構成の各種設定項目を指定します。
 - [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。

🖼 СОВ 🗙 🍋 ЛЦ 😤 Аррі		サーハ 🛄 Anal 🗆 📄	Cntl_Card.dat ×			
		✓ E < ↓ Z	20110101			
✓ I RemoteProject [rocky9-v-n	a:/h	ome/tarot/RemoteProject]				
> 🖉 COBOL ブログラム		新規作成(N)	>	1		
 > ₩ JE-774/k > New_Configuration.bin ■ BATCHAPT.idy ■ BATCHAPT.o ■ BATCHAPT.o 		開く(O) 表示方法(W) リモート データ ファイルエディタで開く アプリケーションから開く	Alt+シフト+W > >			
BATCHAPT.objlist		วษํ-	Ctrl+C	L .		
RemoteProject	Ē	貼り付け	Ctrl+V			
Cntl_Card.dat	×	削除(D)	削除			
Emp_Master.dat	<u>.</u>	コンテキストから除去	Ctrl+Alt+シフト+下			
		移動(V)				
		名前を変更(M)	F2			
		タスクのスキャン				
		インポート(i)	>			
🏗 アウトライン 🗙 🔤 プログラム アウ	4	エクスポート(O)				
	8	更新(F)	F5			
アワトラインを提供するアクティフなエテ	Q.	Coverage As	>			
	0	実行(R)	>	L.,		
	蓉	デバッグ(D)	>	CBL	1 COBOL アプリケーション	Alt+シフト+D,M
	8	プロファイル(P)	>		デバッグ の構成(B)	
	\checkmark	榆証				

② [COBOL アプリケーション] をダブルクリックします。

構成の作成、管理、および実行

COBOL プログラムをデバッグします

	このダイアログから起動設定を構
フィルタ入力	🗋 - 選択した種類の構成を作用
🗄 Apache Tomcat 🔨	- 選択した種類の起動構成プロ
AspectJ/Java Application	 🔊 - 選択した構成をエクスポート
AspectJ Load-Time Weaving	▶ - 選択した構成をつピーするに
● Chrome Debug ■ COBOL/Java 相互運用機能の	■ 送入した時次を当とうした
COBOL Enterprise Server	
🚾 COBOL JVM アプリケーション) - ノイルタ・オノションを構成す
COBOL JVM ユニット テスト	- プロトタイプをリンクするには、起
	- プロトタイプのリンクを解除する
■ COBOL アブリケーション ■ COBOL フアダンプ	- プロトタイプ値でリセットするには

- ③ 以下の入力を行い、[適用(Y)]をクリックします。
 - 名前

ワークスペース内で実行時/デバッグ構成として識別可能な適当な名前を指定します。

● 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。



名前(N): RemoteProject				
🗟 一般 🦻 ソース 🚾 環境	🔲 共通 👂 実行時 🦆 デバッグシ	/ンボル 👌 動的分	所 🔮 CTF 🥑 コンテナー	
▼ COBOL プロジェクト(P)				^
RemoteProject		参照		
▼ 接続プロパティ				
リモートホスト(H):	ROCKY9-V-NA			
☑ リモート ホストで cobde	ebugremote プロセスを起動する			
🗌 リモートホストで cobde	bugremote プロセスが受信するポー	-トを指定する		
cobdebugremote ポート	: 8000			
✓ X サーバーを使用 (U)		-		
X サーバー (DISPLAY):	\${hostname}:0.0			
▼ 主プログラム				
↓ プログラ人 けプロミジェクト	「ILド構成の—部・New Configu	ration ~]	~
		Ī	前回保管した状態に戻す(V)	適用(Y)

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
 - 環境変数を追加または変更します

L

変数: dd_EMPSEQ		
值: /home/tarot/RemoteProject/Emp_Master.dat		
?	ОК	キャンセル

⑥ 同じ手順で以下の変数を追加します。

変数	值
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



名前(N): Re	moteProject		
🗟 一般 🍹	ソース 🌄 環境 🔲 共通 🎤	実行時 🧤 デバッグシンボル 🗳 動的分析 🥸 CTF 🥒 コンテナー	
(注: ここで知 環境スクリン	E義された変数は、任意の現れ プト内の設定値を上書きします	の設定値、または任意の指定された)	^
変数 dd_EMPS dd_CNTL dd_HIREI	ieq CARD RPT	値 /home/tarot/RemoteProject/Emp_Master.d /home/tarot/RemoteProject/Cntl_Card.dat /home/tarot/RemoteProject/Hire_Report.dat 創除(R)	
実行する 場所: パラメータ:	景境スクリプト: ファイルがプロジェクト内にある	参照…	
		前回保管した状態に戻す(V) 適用(Y)	~
		デバッグ(D) 閉じる	

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。

This kind of launch is configured to open the デバッグ perspective when it suspends.			
このデバッグ・パースペクティブは、アプリケーションのデバッグをサポートするように設計されています。 これに は、デバッグ・スタック、変数、およびプレークポイント管理を表示するビューが組み込まれています。			
Switch to this perspective?			
□ 常にこの設定を使用する(R)			
	切り替え(S)	いいえ(N)	

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。

Cntl_Card.dat BATCHAPT.cbl ×			クイック アウトライン	Ctrl+O
BATCHAPT.cbl >			アプリクーションから開く(I) 表示方法	> Alt+シフト+W(W)>
	*A·1·B··•···2····3····•4····5··	-k	47111Em11/T)	Chul - V
	MOVE CNTL-DAY TO RPT-SELECTION-DD. MOVE CNTL-YR TO RPT-SELECTION-YYYY.	93 10 10	いり取り(1) コピー(C) 貼り付け(P)	Ctrl+C Ctrl+V
	WRITE RPT-RECORD FROM RPT-TITLE-3 BEFORE ADV WRITE RPT-RECORD FROM BLANK-LINE BEFORE ADVA		クイック・フィックス(Q) 右へシフト(S)	Ctrl+1
	WRITE RPT-RECORD FROM RPT-COLUMNS BEFORE ADV WRITE RPT-RECORD FROM BLANK-LINE BEFORE ADVA		左ヘシフト(H) リファクタリング ソース	>
Θ	1000-EXIT.	0	Coverage As	``````````````````````````````````````
	EXIT.	0	et(R)	>
Θ	2000-MAIN-PROCESSING. READ EMP-SEQ-FILE INTO EMP-RECORD-IO-AREA	*	デバッグ(D)	>
		8	プロファイル(P)	>
	MOVE 'Y' TO EOF-FLAG.		太昭	Ctrl+S/7b+G
			シューマにある	
Θ	IF NOT-AT-EOF PERFORM 3000-PROCESS-RECORD THRU 3000-EX END-IF.		完美位署∧	ES
			ファイル設定	>
	2000-EXIT		データフロー分析	>
-	EXIT.		プログラムフロー グラフ	
			コード分析	>
Θ	3000-PROCESS-RECORD.	۲	GitHub	>
<			監視ポイントの切り替え	
	< ■ 問題 □ デバッグ・シェル		検証	
			チーム(E)	>
Loaded: [User Module]BATCHAPT- Symbols loaded from: /home/t			比較対象(A)	>
			置換対象(L)	>
		⇒ï	指定行まで実行(L)	Ctrl+R
	•	R	実行点をリセット	

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。

(X)= 変数 × ● ブレークポイント ■	プログラム アウトラ 🖋 式 🏾 🗖 మ 🍻 🖻 🖇	変数ビューを確認すると先ほどは初期値が入っ ていた EMP-RECORD-IO-AREA にファイル
名前	值	
EMP-SEQ-FILE	Open Input Last status:0/0	
🖌 🌳 EMP-RECORD-IO-AREA	11111113 佐藤 隆 サトウ	
✓ ◆ EMP-REC	1111113 佐藤 隆 サトウ	
EMPREC-SSN	1111113	
FILLER		
EMPREC-JNAME1	佐藤	
EMPREC-JNAME2	隆	
EMPREC-NAME1	<u> </u>	
EMPREC-NAME2	<i>У</i> ЛЭ	
EMPREC-GENDER	М	
FILLER		
EMPREC-DIV	営業部	
> EMPREC-DATE-OF-H	19980401	
FILLER		
NOT-AT-EOF	真	
佐藤 16進: 8B9∆222222	^	
D231000000		

④ 条件付きブレークポイント機能を確認します。



	Cnt	l_Card.dat	BATCHAPT.cbl ×	
	ø	BATCHAPT.cbl	•	
		···•·*A·1·B	•••••••	•••3••••••4•••••5•••••6•••
(1000-EXIT. EXIT. 2000-MAIN-PROCESSING. READ EMP-SEQ-FILE INTO EMP-RECORD-IO-AREA AT END MOVE 'Y' TO EOF-FLAG. 			
\$	•	ľ	F NOT-AT-EOF PERFORM 3000	-PROCESS-RECORD THRU 3000-EXIT
		E	ND-IF.	
	Cr	tl_Card.dat	BATCHAPT.	cbl ×
		BATCHAPT.	cbl 🕨	
		····*A·1	•B•••••2•••	••••••
	Θ	100	0-EXIT.	
		200	EXIT.	SETUC
	, Ŭ	200	DEAD END CE	A ETIE TNTA END DECADA T
		ブレークポイン	/トの切り替え(K)	ダブル・クリック
		ブレークポイン	小を使用不可にする	る(D) シフト+ダブル・クリック
		ブックマークの	追加(K)	
1		タスクの追加	(T)	
		監視ポイント	の切り替え	
		検証		
	~	クイック Diff	の表示(O)	ブレークポイントにマウスカーソルを合わせ、右上
		行番号を表	示(N)	のアイコンをクリックして、[ブレークポイント プロパ
		設守(5)		ティ(R)] を選択します。
		取入上(「)…		
		ブレークポイン	ハ フロバティ(R)	



[適用して閉じる] をクリックします。



Cntl_Ca	rd.dat 🙋 BATCHAPT.cbl ×	- 0	(X)= 変数 × 🗣 ブレークポイント 🧱	プログラム アウトラ	🛠 式	- 6
🖻 BA	TCHAPT.cbl 🕨				£. 🗢	ti 🖂
	• .*A · 1 · 2 · · • · · · 2 · · · • · · · 3 · · · • · · · 4 · · · • · · · 5 · · · • 6 · · · • 6 · · · • 7 · · I · • · · · 8		名前	値		
Θ	1000-EXIT.	^	EMP-SEQ-FILE	Open Input Last	status:0/	/0
	EXIT.		✓ ◆ EMP-RECORD-IO-AREA	55555555 伊藤	弘子	1ኑታ
Θ	2000-MAIN-PROCESSING.		✓	55555555 伊藤	弘子	<u> </u>
20	READ EMP-SEQ-FILE INTO EMP-RECORD-IO-AREA		EMPREC-SSN	55555555		
	AT END		FILLER			
	MOVE Y TO EUF-FLAG.		EMPREC-JNAME1	伊藤		
	TE NOT-AT-FOF		EMPREC-JNAME2	弘子		
	PERFORM 3000-PROCESS-RECORD THRU 3000-EXIT		EMPREC-NAME1	<u> </u>		
	END-IF.		EMPREC-NAME2	ŁDO		
			EMPREC-GENDER	F		
•	2000-EXIT.		FILLER			
	EXIT.		EMPREC-DIV	技術部		
			> < EMPREC-DATE-OF-	H 20010401		
Ŭ	***		FILLER			
	* FIRST, VERIFY EMPLOYEE'S HIRE DATE IS ON OR BEFORE DATE	-				
	* PASSED IN CONTROL CARD.		55555555 伊藤 弘子	- - イトウ トロコ	F 技術	部。
	***		16進:		. 1811.	100
Θ	<pre>IF EMPREC-DATE-OF-HIRE <= CNTL-DATE</pre>		3333333328C9A22222284872	222222BCB22CDE	224285	8799
	CONTINUE		555555556893100000DFE1	00000024300BBA	0060BA	F054
	ELSE					

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。

🎋 デバッグ × 陷 プロジェクト・エクスプロ−ラ− 🕷	サーバー 🗖 🗖
	🖻 💥 🗊 🖇
✓ ■ <終了しました>RemoteProject [COBOL ア	プリケーション]
^{oo} Micro Focus デバッガ: /home/tarot/Re	noteProject/New

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プロジェ クトを右クリックの上、[更新(F)]を選択し、ビューをリフレッシュします。

- 0 🔓 COB... × 陷 プロ... 😤 Appl... 📕 サーバ... 📇 Anal... ✓ □ ♀ ↓ ↓ ♥ 8 RemoteProject [rocky9-v-na:/home/tarot/RemoteProject] > 垣 COBOL プログラム > 냳 コピーファイル ✓ ➢ New_Configuration.bin BATCHAPT.idv BATCHAPT.o BATCHAPT.o.1.tlog BATCHAPT.objlist RemoteProject Cntl_Card.dat Emp_Master.dat Hire_Report.dat

② COBOL エクスプローラ中の 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。



🕆 COB 🗵 陷 ರೆದಿ 🛛 😤 App	l 🗏 サーバ 🖳 Anal 🗖 🗖	Cntl_Card.dat	BATCHAPT.cbl ×
	✓ □ 4 ↓ 2	BATCHAPT.	cbl 🕨
 RemoteProject [rocky9-v- 	····*A·1	·B·••···2···•3····•	
> / COBOL プログラム		⊖ 100	0-EXIT.
> 🔑 コピーファイル			EXIT.
🗸 🗁 New_Configuration.bin		⊖ 200	0-MAIN-PROCESSING.
BATCHAPT.idy		•	READ EMP-SEQ-FILE INTO EMP-
BATCHAPT.o			AT END
BATCHAPTo.1.t			MOVE 'Y' TO EOF-FLA
	新規作成(N)		>
	開く(O)		NOT-AT-EOF
	表示方法(W)	Alt+シフト+	W> PERFORM 3000-PROCESS-RE
	リモート データ ファイルエディタで聞く)-IF.
Emp_Master.dat	アプリケーションから聞く		> 📄 テキスト・エディター
Hire_Report.dat			

「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。

Cntl_Card.dat	rd.dat 🖻 BATCHAPT.cbl		Hire_Report.dat ×			
Program: BATCHRPT		Ye	Years Employed Report		08/22/2024	ŧ.
****	2011年 1月 1	日以前に入	社した社員一覧		17:23:31	L
部署名	社員名		社員番号	入社日	雇用年数	
営業部	佐藤	隆	1111111-3	04/01/1998	26	
技術部	鈴木	尚之	222222-6	10/15/1998	26	
総務部	田中	直美	3333333-9	04/01/1999	25	
営業部	山田	洋一	444444-2	07/01/2000	24	
技術部	伊藤	弘子	5555555-5	04/01/2001	23	
営業部	木村	貴弘	6666666-8	12/20/2002	22	
技術部	中村	慎司	777777-1	04/01/2003	21	
総務部	橋本	悦子	8888888-4	08/05/2004	20	
営業部	三井	薫	9999999-7	04/01/2005	19	
****	処理レコード件数	Þ:	9			

以上で本チュートリアルは終了となります。

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。