

Visual COBOL チュートリアル

JavaEE アプリケーションから

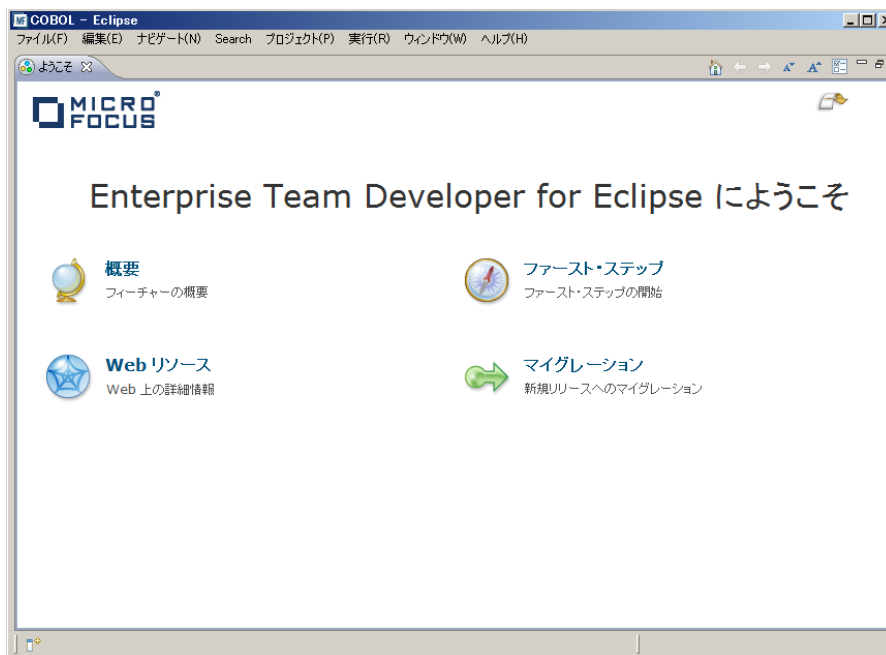
ネイティブ COBOL サービスの呼出し

2. 最も簡単な COBOL サービスのデプロイ

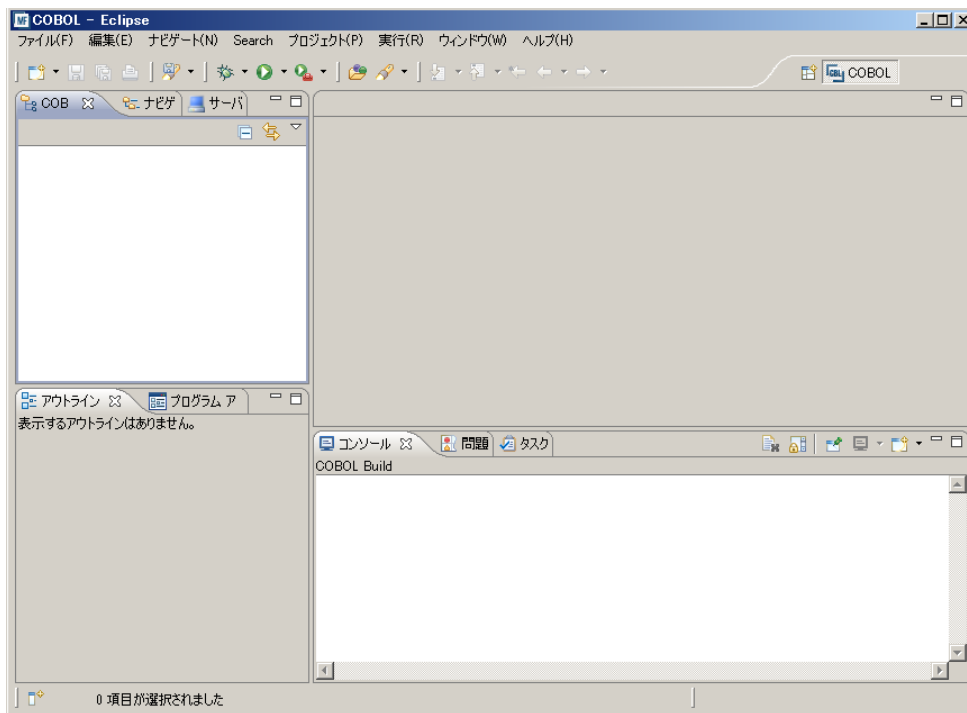
チュートリアル 1 ではすでに完成済みのパッケージを使用して COBOL, Java の双方にデプロイして動作を確認しました。ここで使用したパッケージは Visual COBOL 製品の IDE である Enterprise Developer が装備する Interface Mapping パースペクティブを使用して作成することができます。ビジネスロジックを提供する COBOL プログラムさえ用意されていれば、一行もプログラミングすることなく作成できます。

2.1 Eclipse プロジェクトの作成

- 1) Windows スタートメニューより [すべてのプログラム] > [Micro Focus Enterprise Developer] > [Enterprise Developer for Eclipse] を開きます。
- 2) ワークスペースを選択するダイアログでは C:\¥IMTKDEMO を指定します。
- 3) 以下の「ようこそ」バナーが表示されます。



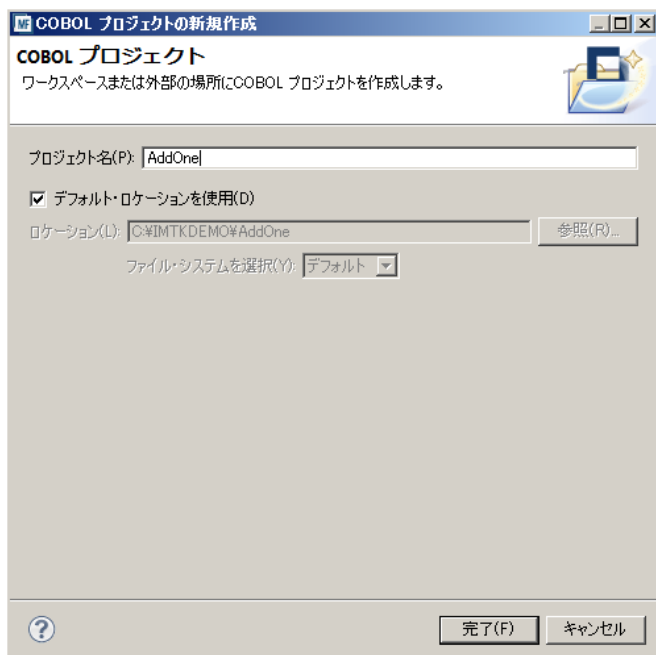
- 4) バナーを閉じると COBOL パースペクティブが開いています。そうでない場合は COBOL パースペクティブに変更してください。



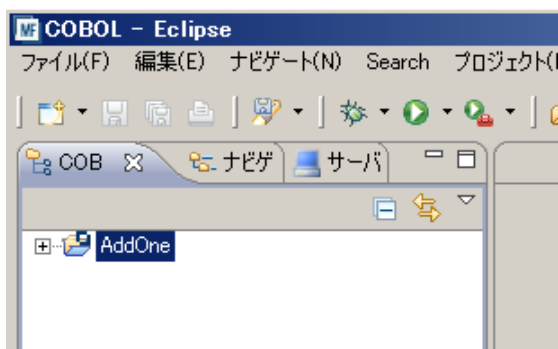
- 5) [ファイル] > [新規] > [COBOL プロジェクト] を選択します。



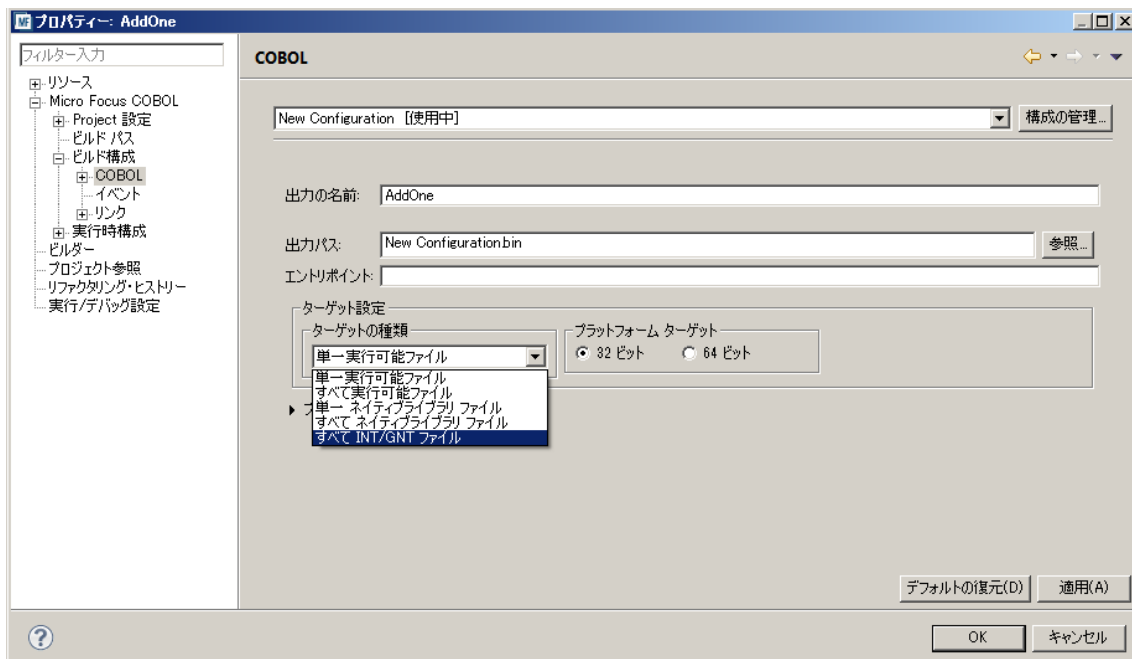
- 6) 以下のダイアログで作成するプロジェクトの名称を入力します。ここでは “AddOne” とします。



- 7) [完了]ボタンをクリックすると以下のように COBOL エクスプローラー内に AddOne プロジェクトが表示されます。

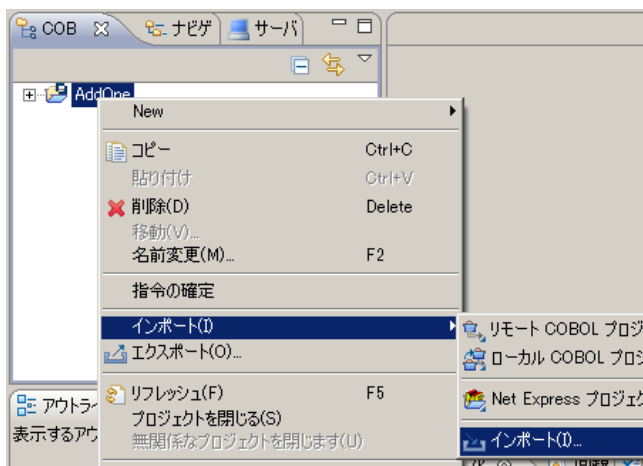


- 8) AddOne プロジェクトを右クリックし [プロパティ] を選択します。
- 9) 以下のプロジェクトプロパティのダイアログの左側ペインで [Micro Focus COBOL] > [ビルド構成] > [COBOL] を選択し、右側ペインの [ターゲットの種類] のプルダウンから [すべて INT/GNT ファイル] を選択します。

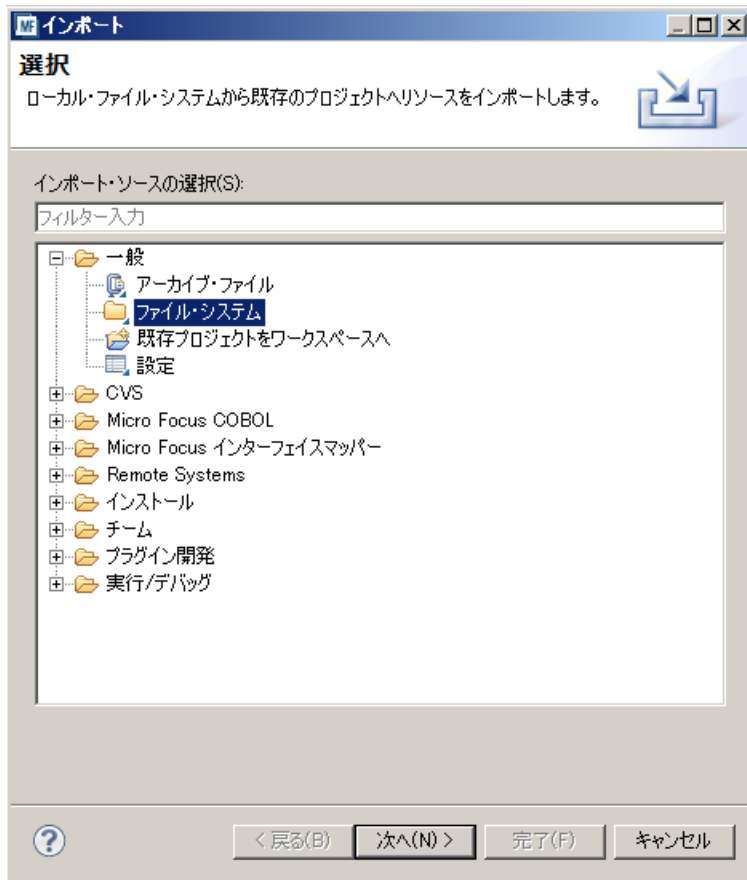


2.2 COBOL プログラムの追加

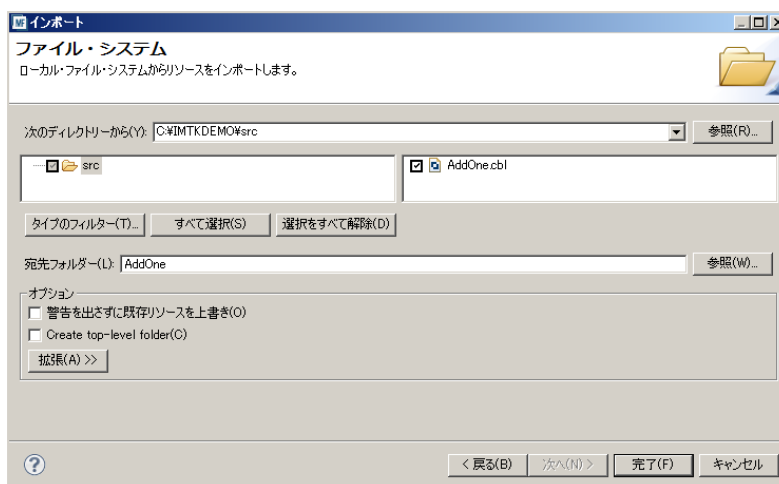
- 1) COBOL エクスプローラー内の AddOne プロジェクトを右クリックし、[インポート] > [インポート] を選択します。



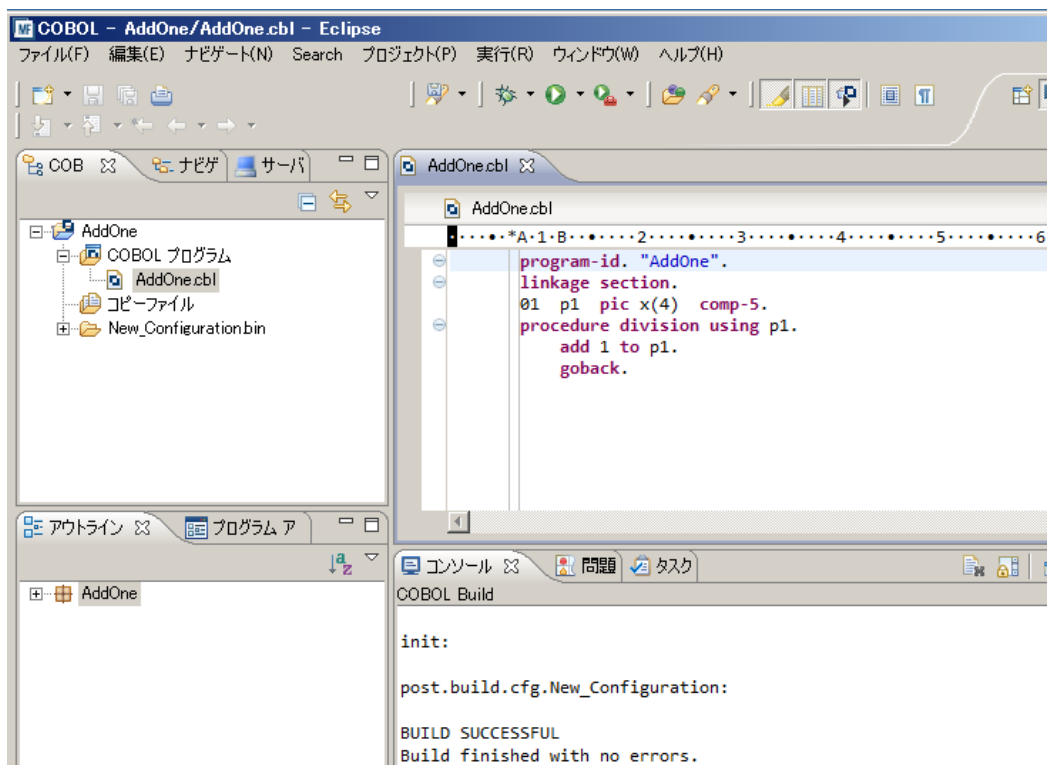
- 2) [インポート] ダイアログで [一般] > [ファイルシステム] を選択し、[次へ] をクリックします。



- 3) 以下の[インポート] ダイアログで、C:\¥IMTKDEMO¥src フォルダにナビゲートし、AddOne.cbl ファイルをチェックします。[完了]ボタンをクリックします。



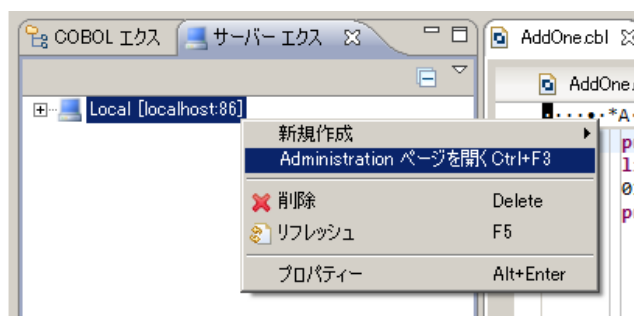
- 4) 以下のように COBOL プログラムがプロジェクトに追加されます。



2.3 開発者用 Enterprise Server の開始

Visual COBOL の IDE である Enterprise Developer には、開発者用のランタイムとしての Enterprise Server が組み込まれています。IDE の中からこれを起動しておきます。

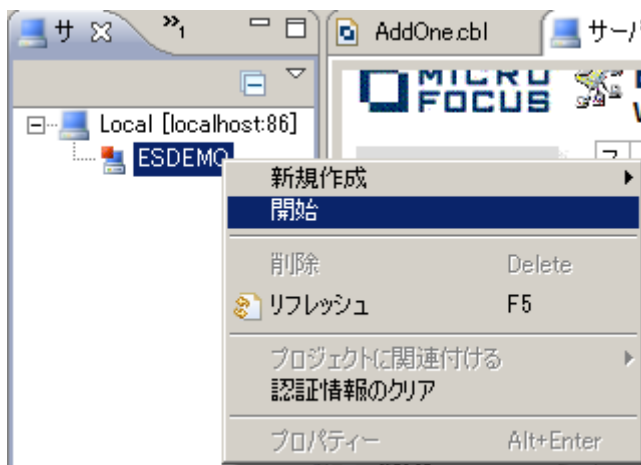
- 1) 以下のように [サーバーエクスプローラー] ビューを開き、[Local] を右クリックして [Administration ページを開く] を選択します。



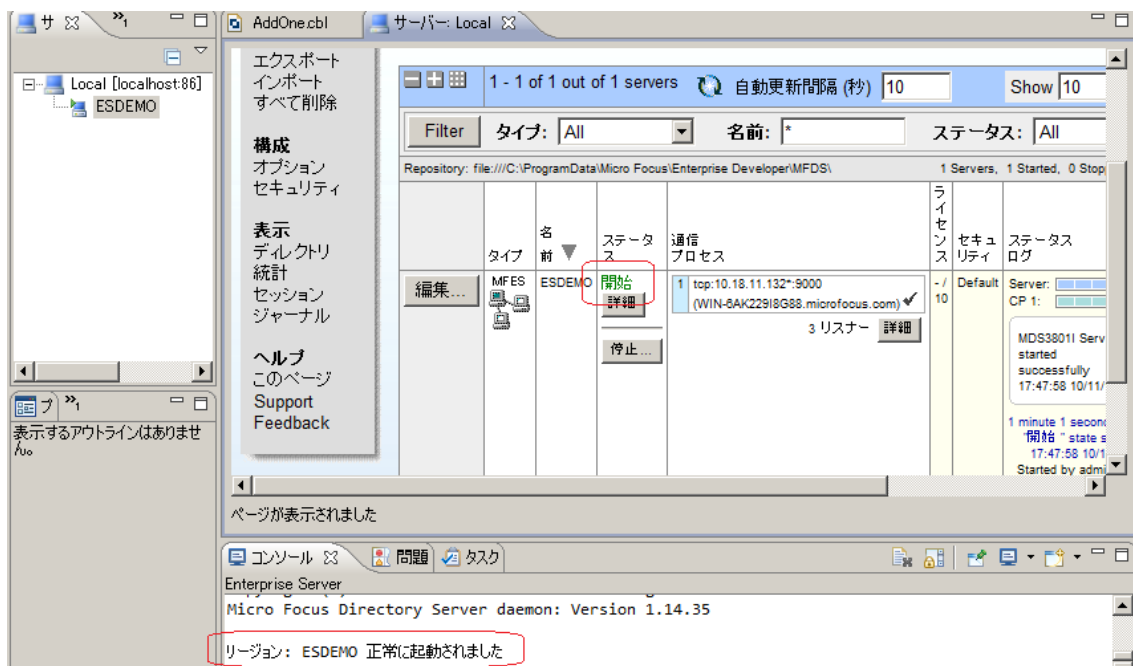
- 2) Enterprise Server 管理コンソールが Eclipse プロジェクト内に開きます。



- 3) [サーバーエクスプローラー]内で [Local] を展開すると [ESDEMO] があります。これを右クリックして [開始] を選択します。

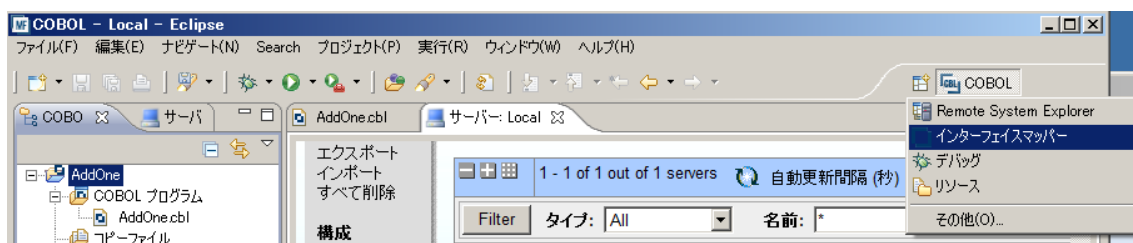


- 4) しばらく待つと以下のように ESDEMO が開始状態となります。

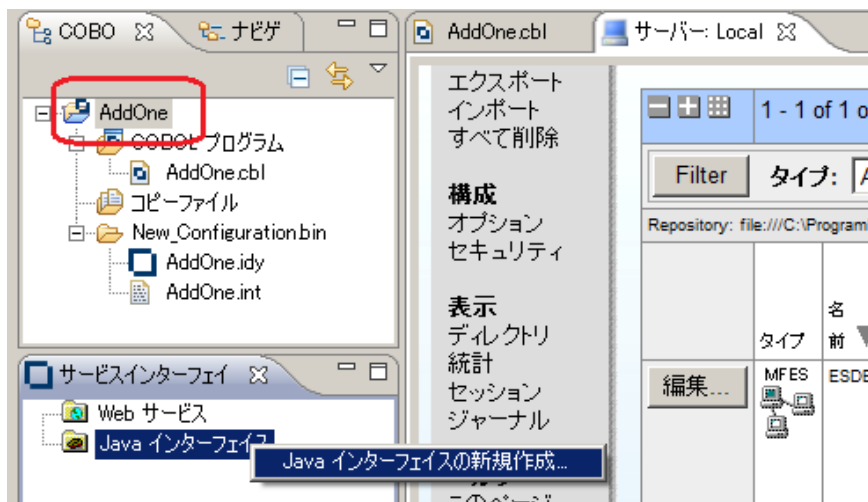


2.4 インターフェイスマッピングの作成

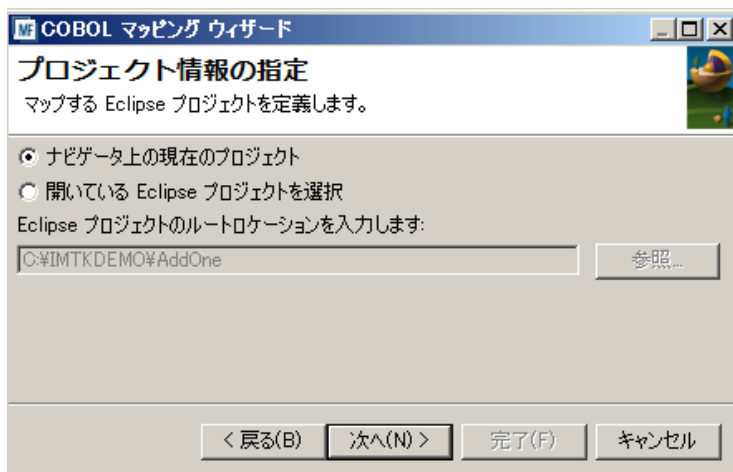
- 1) IDE の右上のパースペクティブ選択にて、[インターフェイスマッパー] パースペクティブを選択します。



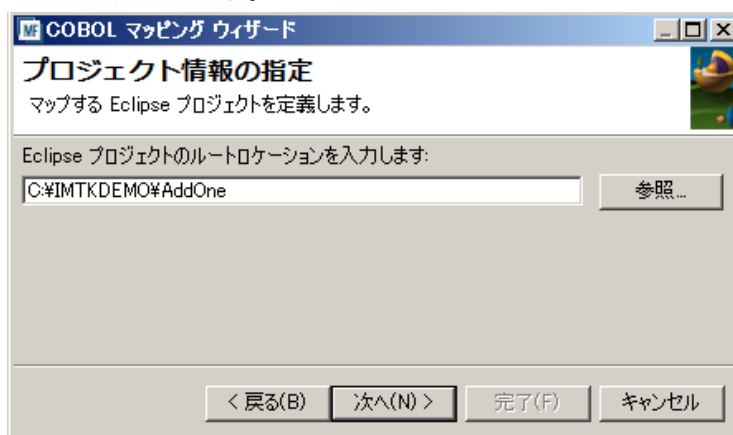
- 2) 以下のように COBOL エクスプローラ内で AddOne プロジェクトを選択した状態で、インターフェイスマッパービュー内で [Java インターフェイス] を右クリックし、[Java インターフェイスの新規作成...] を選択します。



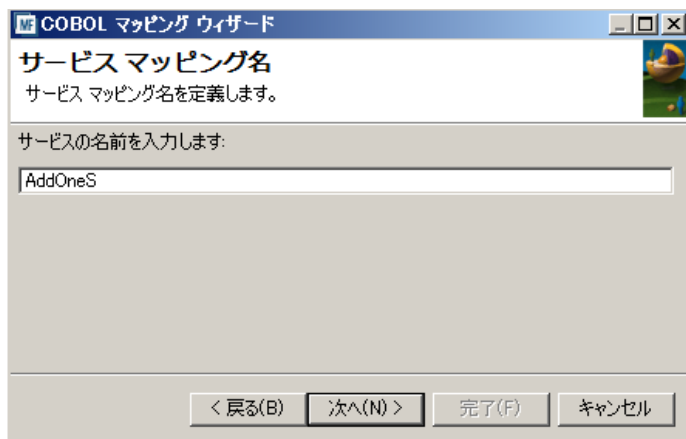
- 3) COBOL マッピングウィザード一度が開始します。以下のダイアログではデフォルトの [ナビゲータ上の現在のプロジェクト] をチェックし [次へ] をクリックします。



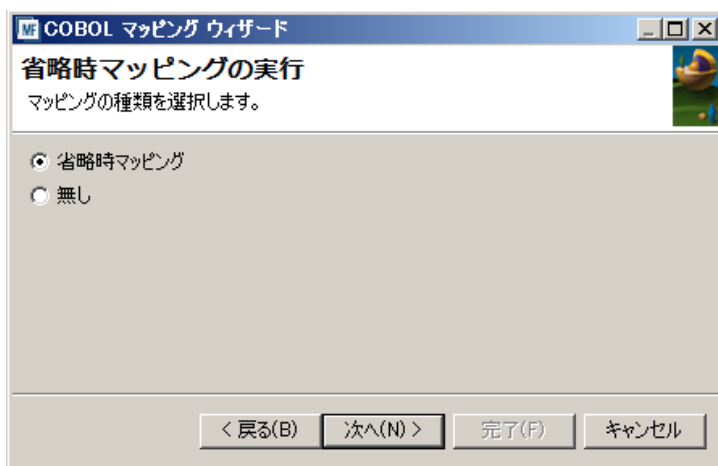
- 4) 以下のダイアログでは現在開いている AddOne プロジェクトのパスのまま [次へ] をクリックします。



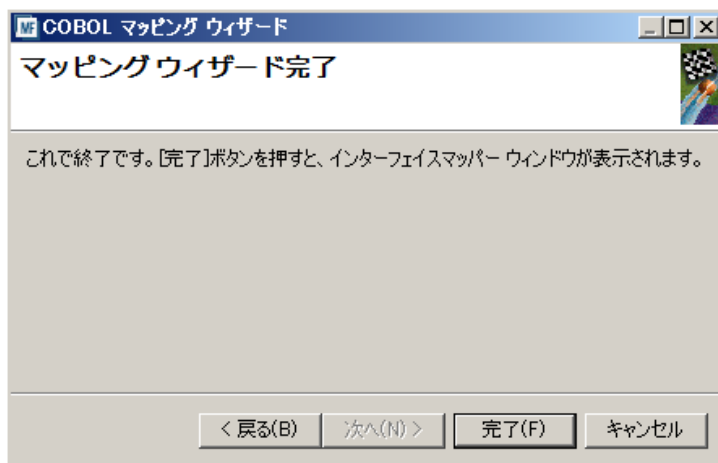
- 5) 以下のダイアログではサービス名として “AddOneS” を入力して [次へ] をクリックします。



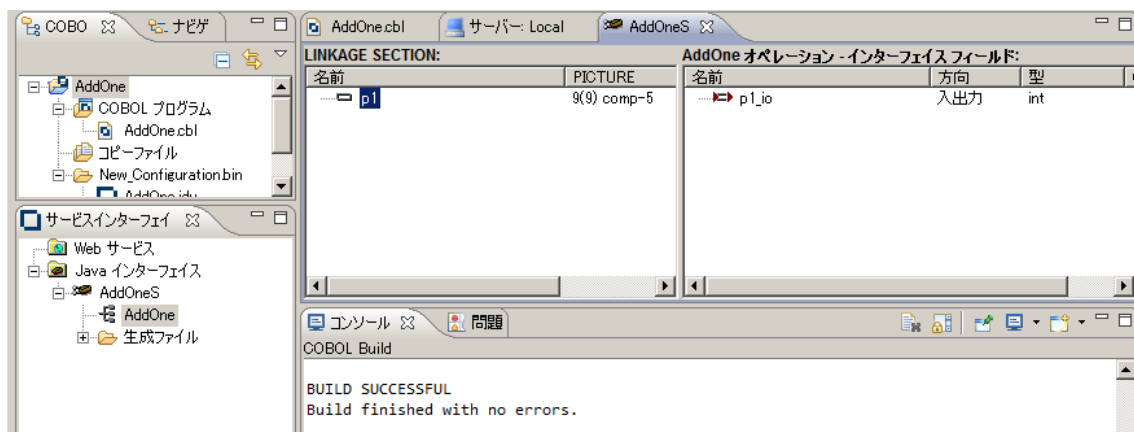
6) 以下のダイアログではデフォルト設定のまま [次へ] をクリックします。



7) 以下のダイアログではそのまま [完了] をクリックします。



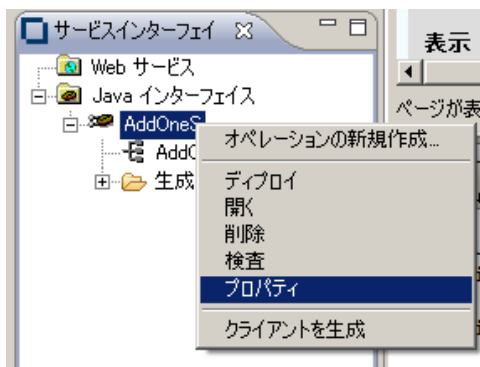
8) 以下のようにインターフェイスマッピングツールキットが開き、COBOL プログラム AddOne.cbl に対するデフォルトのマッピングが表示されています。



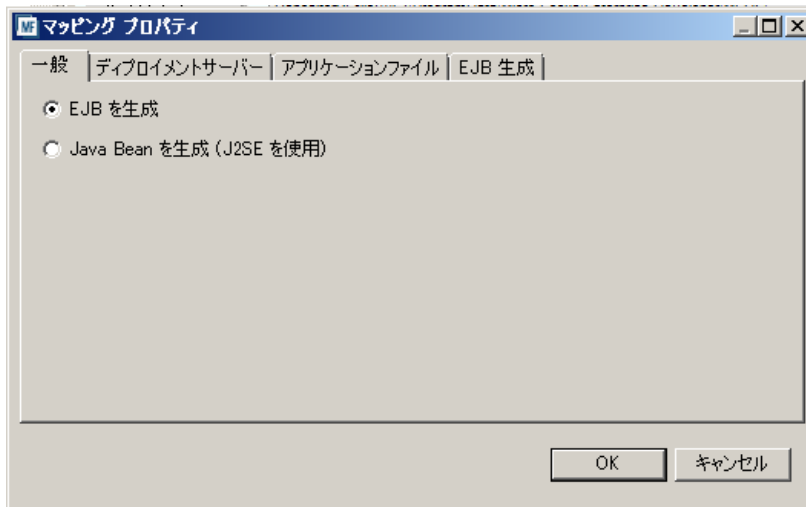
- 9) このプログラムはただ一つの整数型パラメータを持つだけです。ここではそれを `int` 型の入出力パラメータとしてマップしています。このチュートリアルではこのままで問題ありませんので、このままこのペインを閉じます。

2.4 サービスマッピングの属性設定

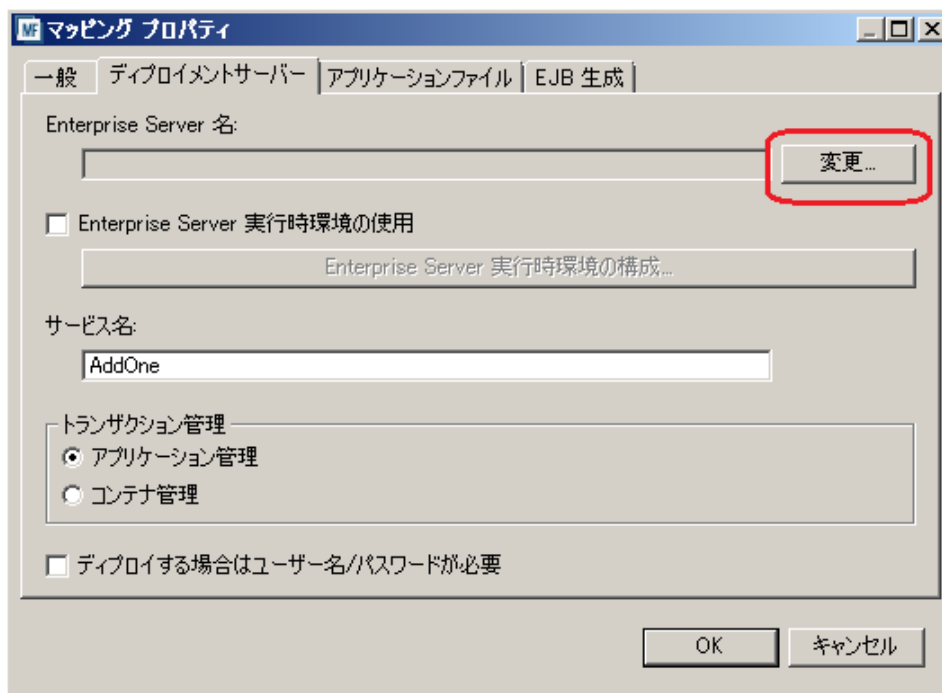
- 1) 作成されたインターフェースマッピング [AddOneS] を右クリックし [プロパティ] を選択します。



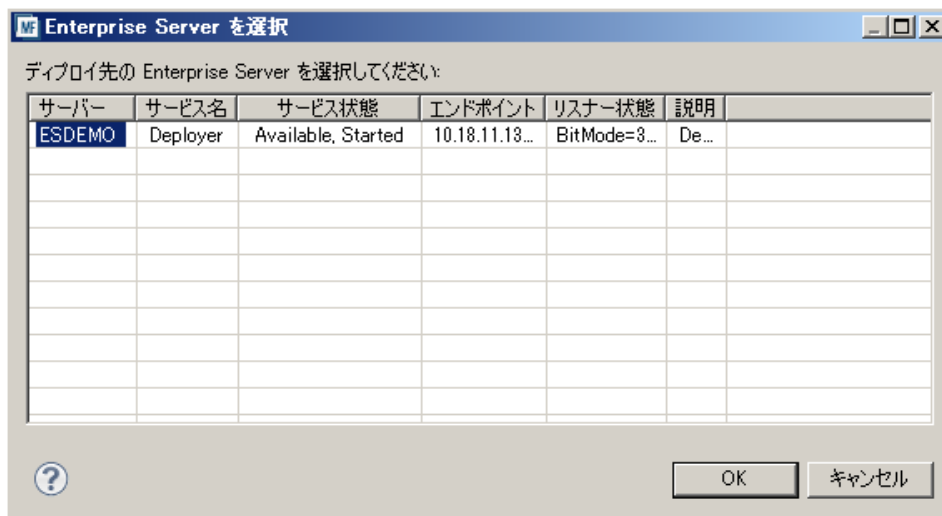
- 2) 以下の [一般] タブでは [EJB を生成] をチェックします。



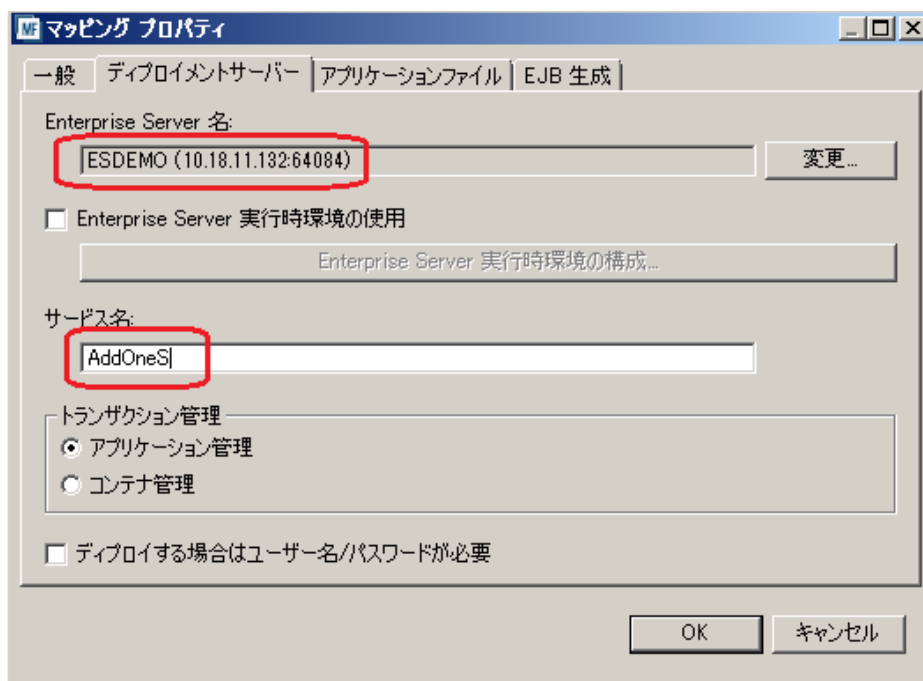
- 3) [Deployment Server] tab, to specify the Enterprise Server to deploy to, click the [Change...] button.



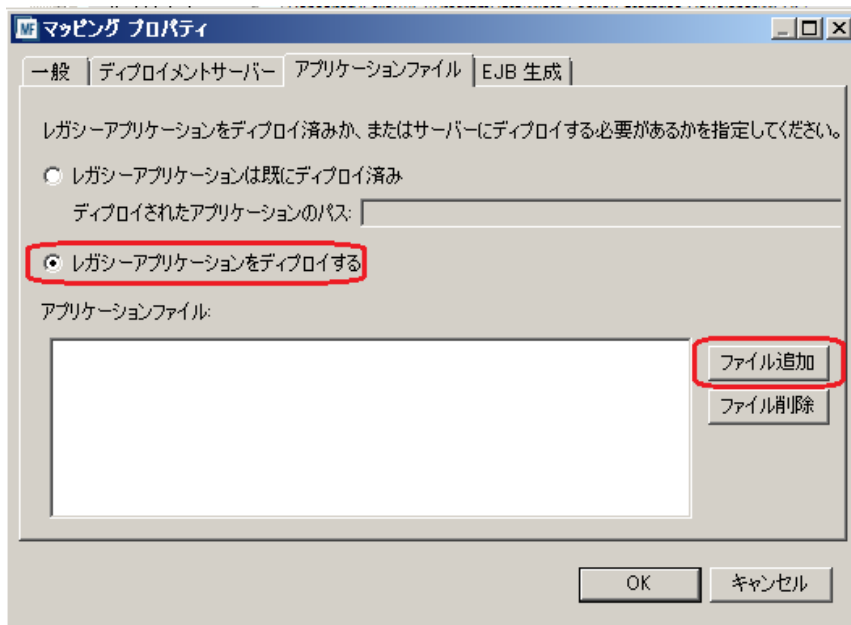
- 4) The ESDEMO is displayed, so select it and click [OK].



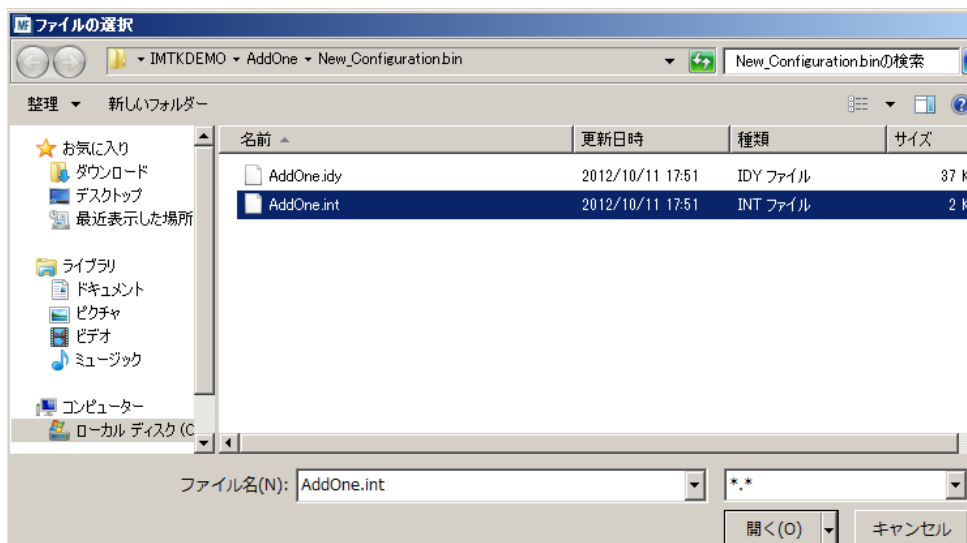
- 5) 以下のようにディプロイ先が設定されます。サービス名として“AddOneS”を入力します。



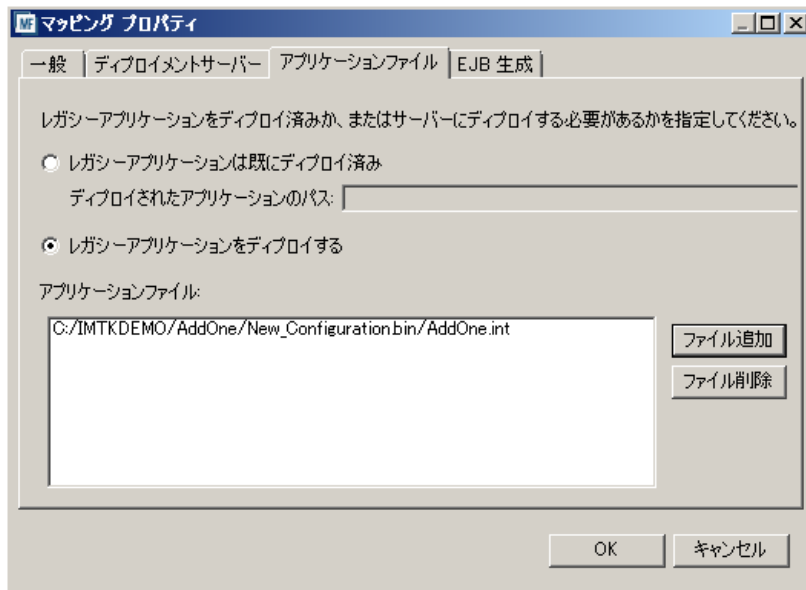
- 6) [アプリケーションファイル]タブでは、[レガシーアプリケーションをディプロイする]をチェックオンし、[ファイル追加]ボタンをクリックします。



- 7) Eclipse プロジェクトの New_Configuration.bin フォルダの下にある AddOne.int を選択して [開く] をクリックします。



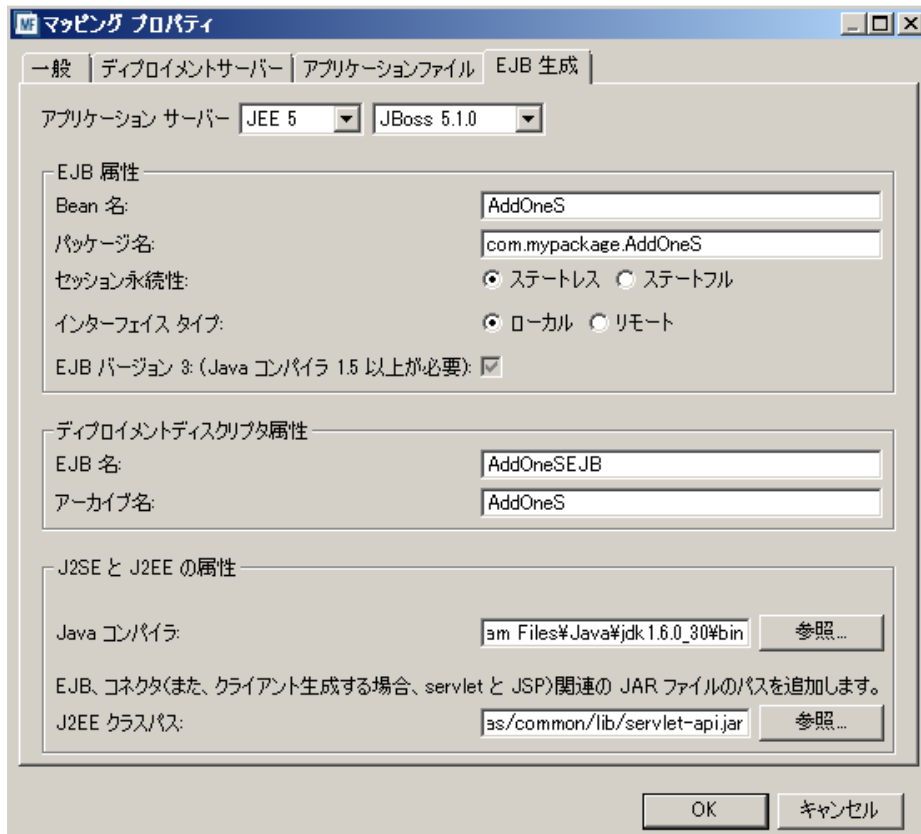
- 8) 以下のようにデプロイされるべきプログラムのコンパイル済みバイナリがパッケージに追加されました。



- 9) [EJB 生成] タブでは、[アプリケーションサーバー] として [JEE5] の [JBoss 5.1.0] を選択します。[Java コンパイラ] には、お使いの JDK の `%bin` フォルダのパス名を入力します。[J2EE クラスパス] には、JBoss が提供する以下の JAR ファイルをセミコロンで区切ってフルパスで指定します。

`%jboss-as%lib%jboss-javaee.jar`

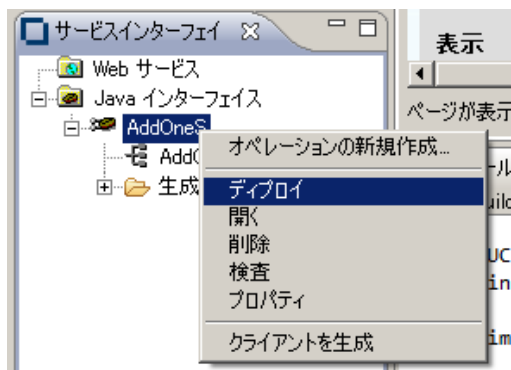
`%jboss-as%common%lib%servlet-api.jar`



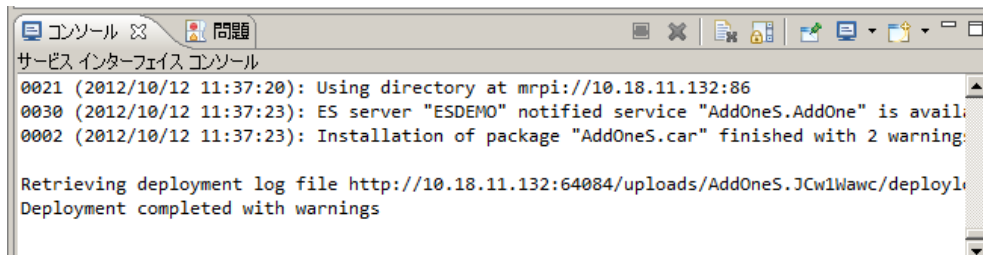
10) [OK] をクリックしてプロパティダイアログを閉じます。

2.5 サービスのデプロイ

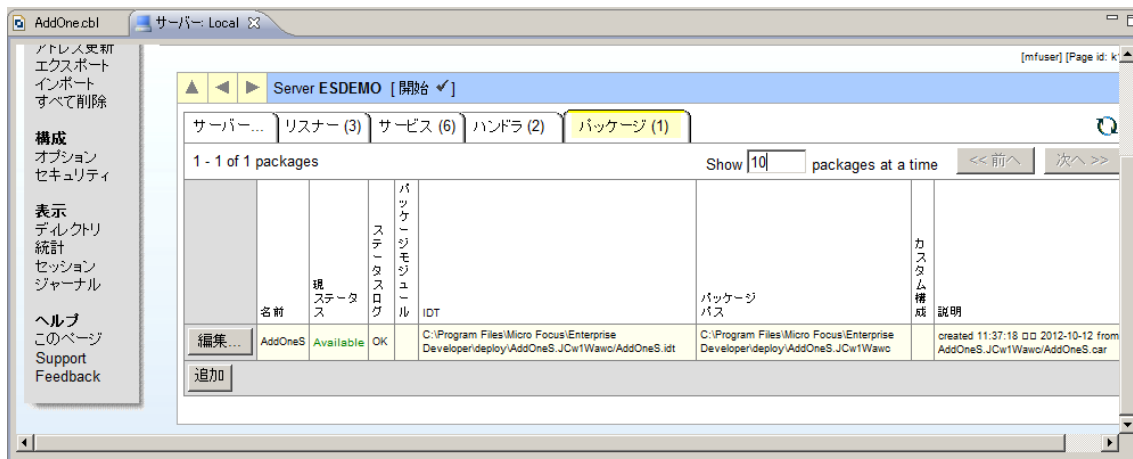
1) [AddOneS] を右クリックして [デプロイ] を選択します。



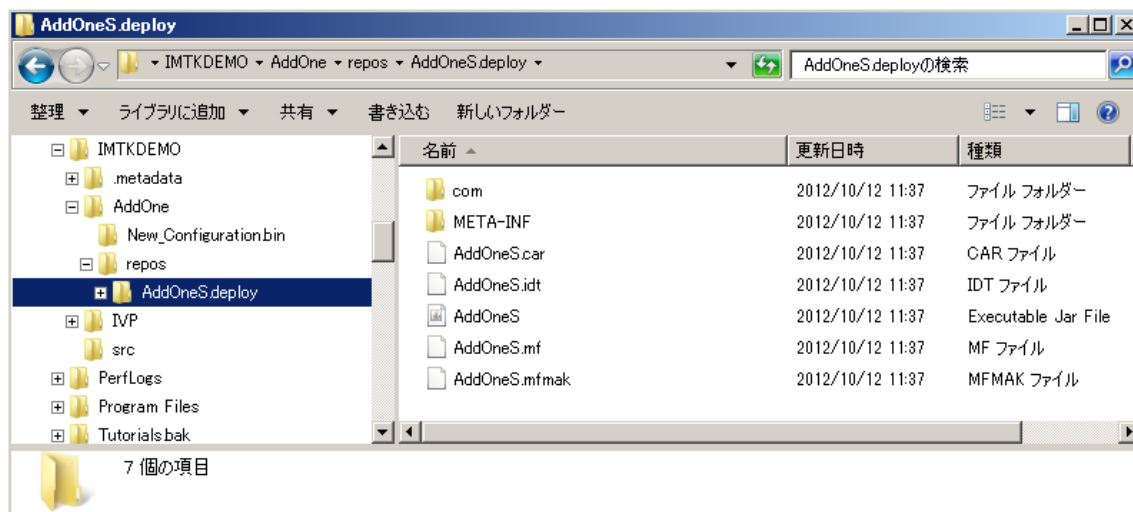
2) 以下のようにコンソールウィンドウにデプロイが完了した旨のメッセージが表示されます。



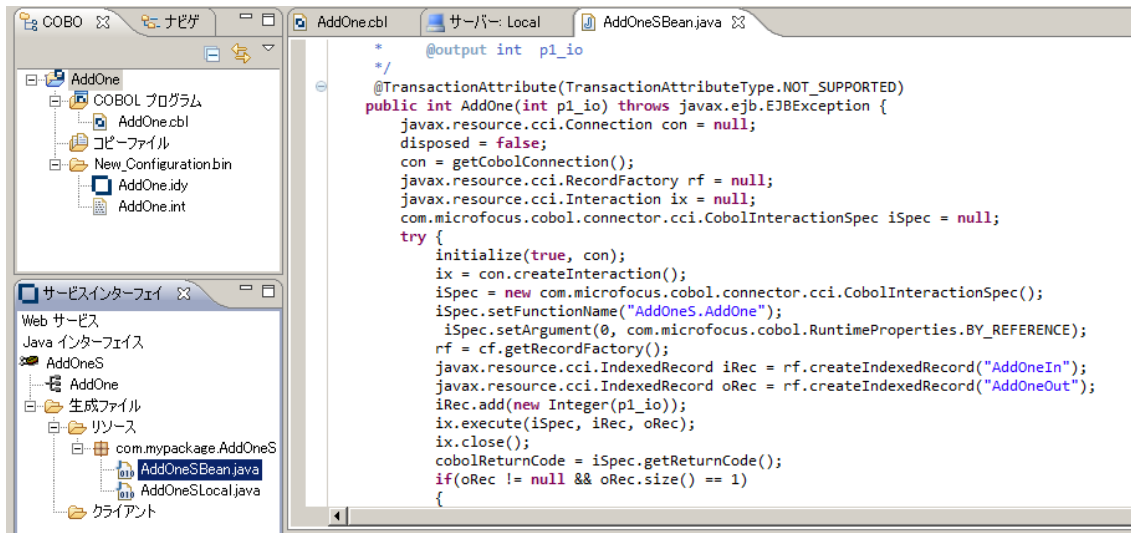
- 3) Enterprise Server 管理コンソール上に AddOneS サービスが追加されていることを確認します。



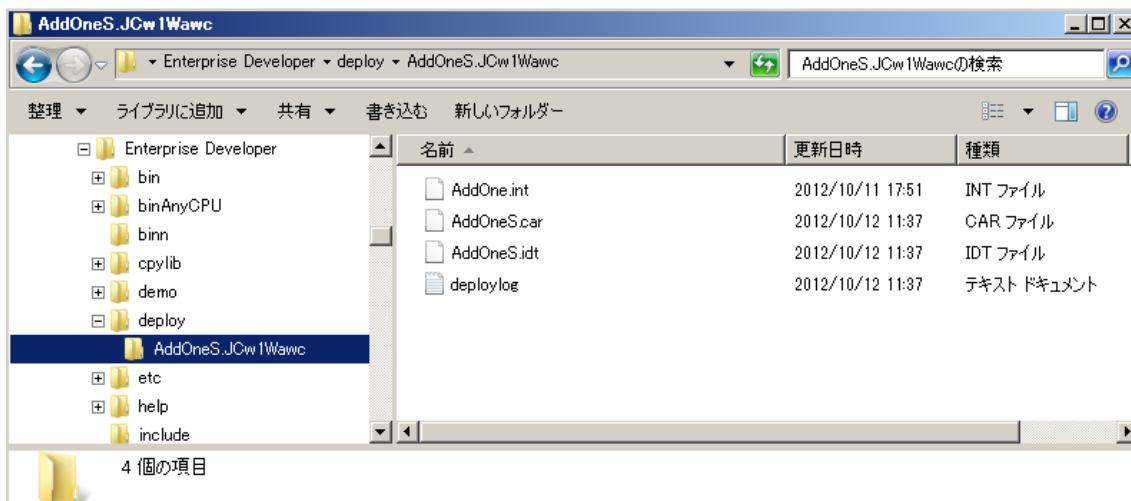
- 4) Eclipse プロジェクトフォルダの下に repos\AddOneS.deploy の下に COBOL サービスのディプロイメントパッケージである AddOneS.car と、EJB ラッパーである AddOneS.jar が生成されていることを確認します。



- 5) 自動生成された EJB ラッパーのソースは、以下のようにサービスインターフェイスの下に [生成ファイル] というフォルダの下から参照することができます。以下のように AddOne という EJB メソッドにラップされていることが確認できます。



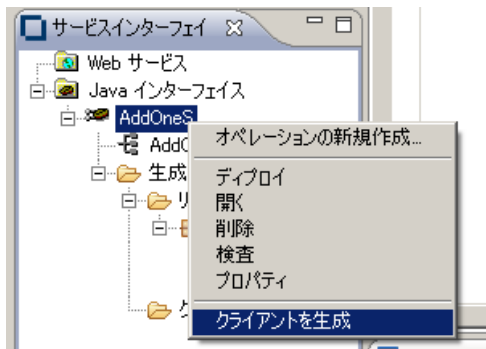
- 6) COBOL サービスはすでにデプロイされており、デフォルトのデプロイ先として C:\Program Files\Micro Focus\Enterprise Developer\deploy の下のサブフォルダに置かれていることが確認できます。



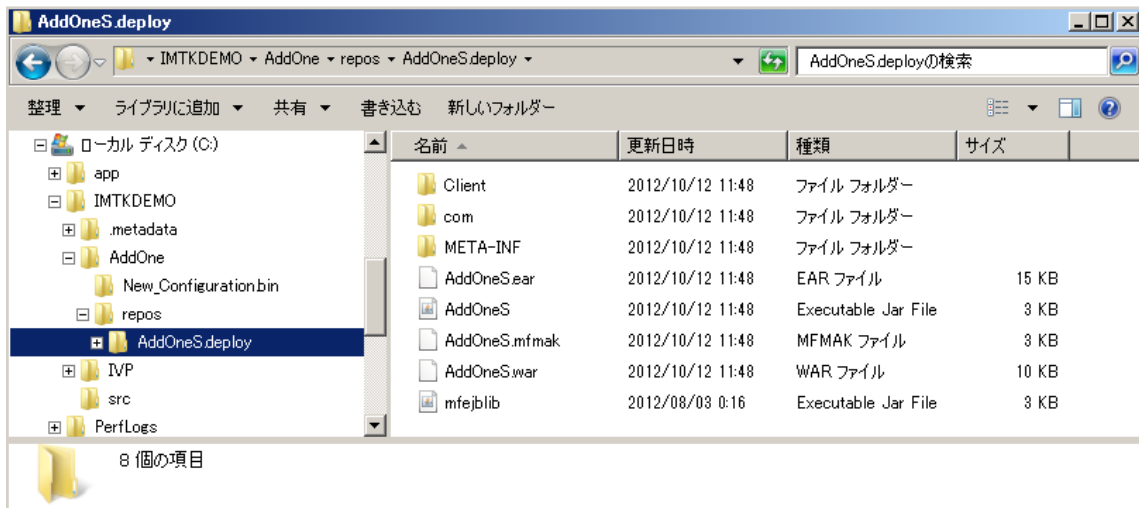
2.6 J2EE クライアントの自動生成

ここまでの手順で生成された EJB ラッパー AddOneS.jar を使用した J2EE アプリケーションを作成することによって COBOL サービスを簡単に利用することができます。本チュートリアルではさらにそのような J2EE アプリケーションのサンプルを自動生成する機能を使用して動作を確認します。

- 1) [AddOneS] を右クリックして [クライアントを生成] を選択します。



- 2) コンソールウィンドウに自動生成が完了した旨のメッセージが表示され、以下のように AddOneS.deploy フォルダの下に AddOneS.ear が生成されています。



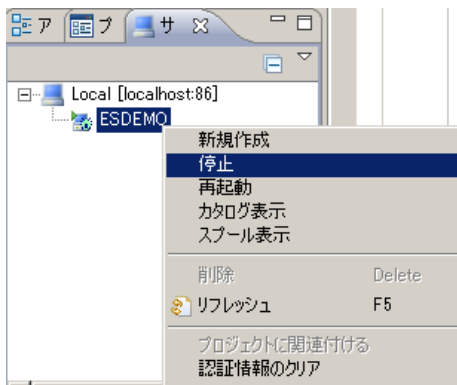
- 3) チュートリアル 1 で実施したように AddOneS.ear を JBoss にディプロイして、サンプルアプリケーションを実行します。

続いてデバッグのチュートリアルに進みますので、ディプロイしたサービスの後始末はしないでください。

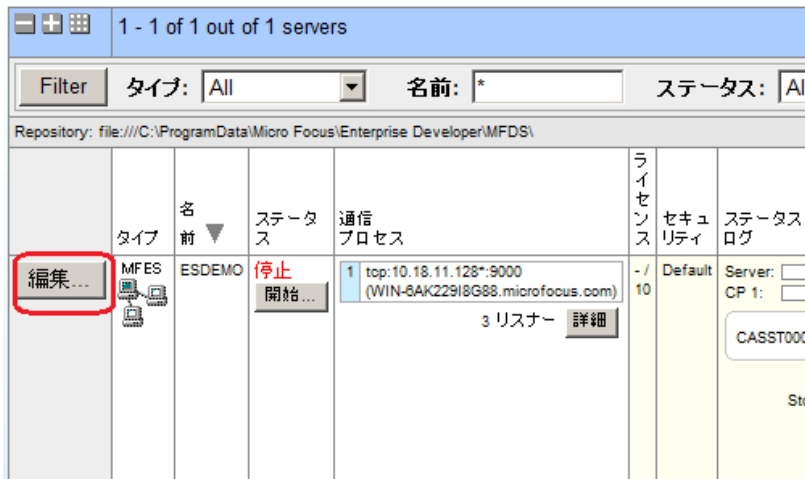
2.7 デバッグ用のサーバー再起動

続いて、ディプロイされた COBOL サービスの実行をデバッグする方法を学習します。

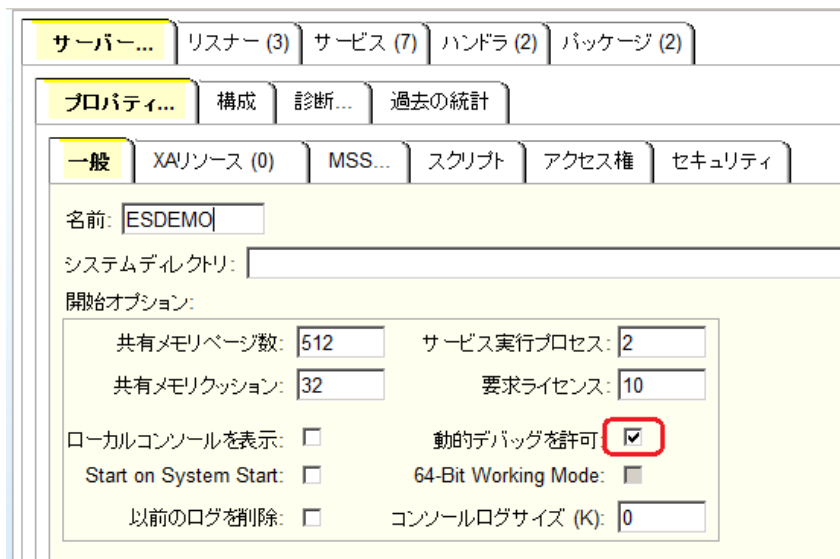
- 1) [サーバーエクスプローラー]内で [Local] を展開すると [ESDEMO] があります。これを右クリックして [停止] を選択します。



- 2) しばらく待つと以下のように ESDEMO が停止状態となります。ここで [編集...] ボタンをクリックします。



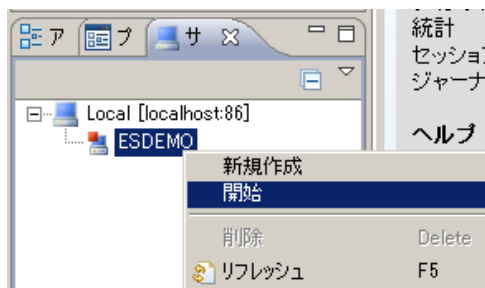
- 3) 以下の [プロパティ...] > [一般] タブで「動的デバッグを許可」をチェックします。



- 4) 画面下の [OK] ボタンをクリックします。



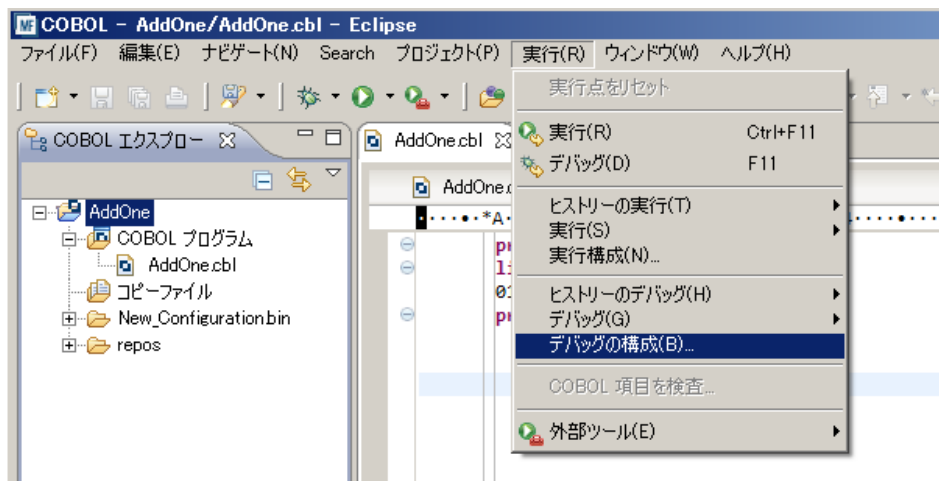
- 5) 再び、[サーバーエクスプローラー]内で[ESDEMO]を右クリックして [開始] を選択します。



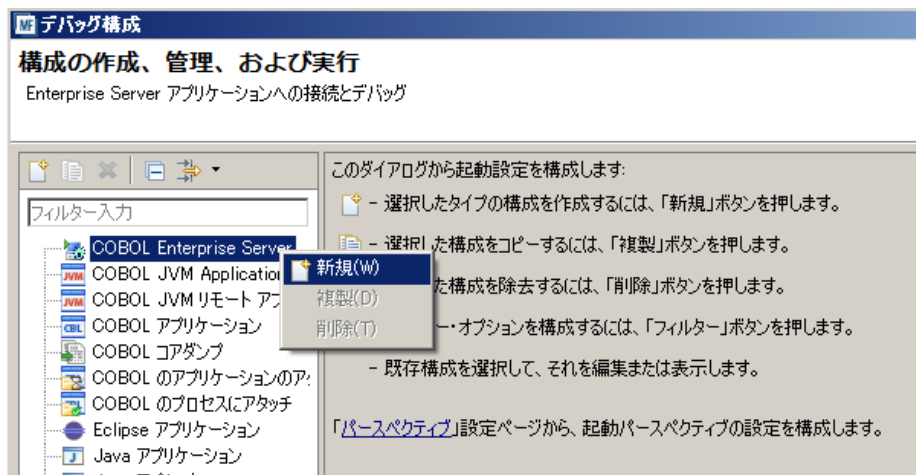
- 6) しばらく待つと以下のように ESDEMO が再度開始状態となります。

	タイプ	名前	ステータス	通信プロセス	ライセンス	セキュリティ	ステータスログ
編集...	MFES	ESDEMO	開始 詳細 停止...	1 tcp:10.18.11.128*:9000 (WIN-8AK229I8G88.microfocus.com) ✓ 3 リスナー 詳細	10 / 10	Default	Server CP 1: MDS

- 7) COBOL パースペクティブに移ります。以下のように AddOne プロジェクトを選択して [実行] > [デバッグの構成...] を選択します。



- 8) 以下のデバッグ構成ダイアログが現れます。左側ペイン内の [COBOL Enterprise Server] を右クリックして [新規] を選択します。



- 9) 以下の [新規構成] ダイアログが現れます。[COBOL プロジェクト]は AddOne をそのまま選択します。

名前(N): 新規構成

一般 ソース 共通(C) デバッグシンボル

COBOL プログラムの起動を待機しながら Enterprise Server 上でデバッグセッションを開始します。

▼ COBOL プロジェクト(P)

AddOne 参照...

▼ Enterprise Server

接続: サーバー: 参照...

▼ デバッグの種類

CICS JCL IMS Web サービス Java

CICS 設定(空白の場合はすべての CICS トランザクションをデバッグ)

ユーザー:

ターミナル:

トランザクション:

リンク・レベル プログラム:

▼ デバッグ オプション

☐ デバッグとコンパイルのソースファイル パスが一致する場合にのみブレークポイントをバインドする

- 10) [Enterprise Server] の [参照] ボタンをクリックして以下のように Local の下の ESDEMO を選択し [OK] をクリックします。

▼ Enterprise Server

接続: サーバー: 参照...

▼ デバッグの種類

CICS JCL IMS Web サービス Java

CICS 設定(空白の場合はすべての CICS トランザクションをデバッグ)

ユーザー:

ターミナル:

トランザクション:

リンク・レベル プログラム:

▼ デバッグ オプション

☐ デバッグとコンパイルのソースファイル パスが一致する場合にのみブレークポイントをバインドする

Enterprise Server を選択

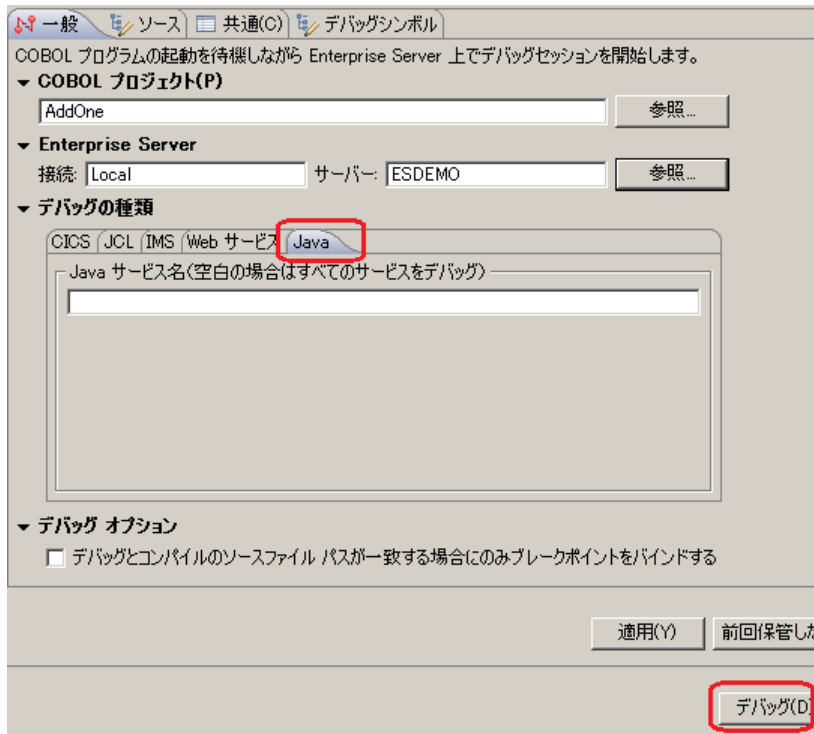
サーバーを選択してください。

Local [localhost:86]

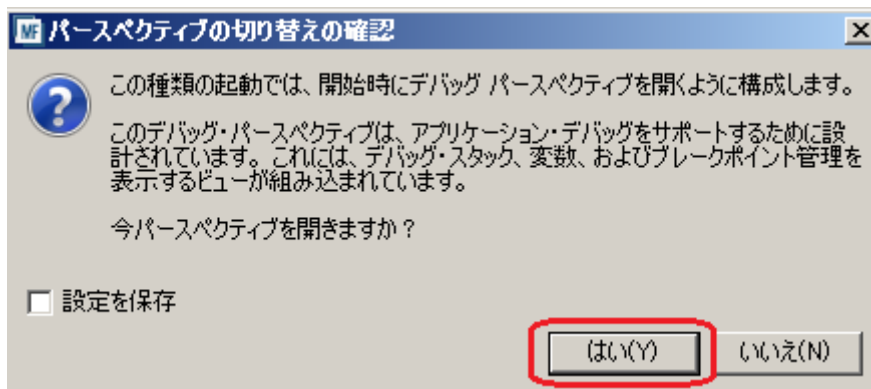
ESDEMO

OK キャンセル

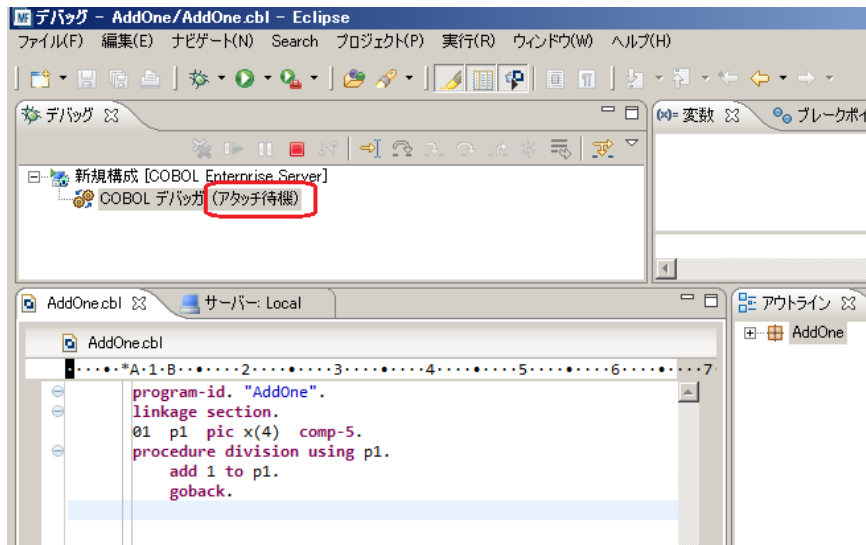
- 11) [デバッグの種類] は以下のように [Java] を選択します。[Java サービス名]は空欄のままにし、[デバッグ] ボタンをクリックします。



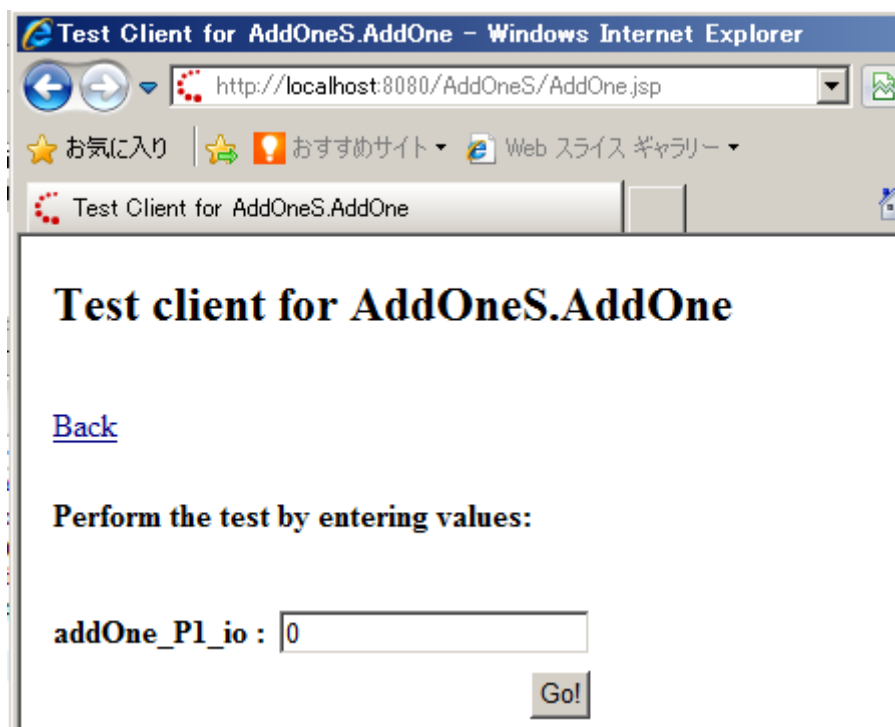
12) 以下のダイアログが現れたら [はい] をクリックします。



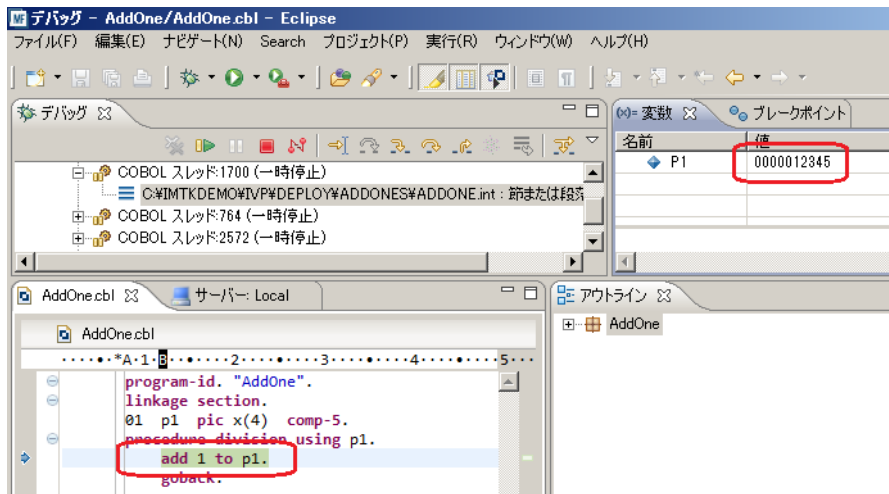
13) 以下の通りデバッグパースペクティブに移動し、「アタッチ待機」の状態になります。



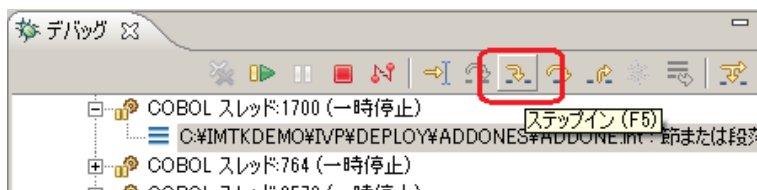
- 14) ここで J2EE アプリケーションを起動して COBOL サービスを呼び出します。Internet Explorer を開き、 <http://localhost:8080/AddOneS/AddOne.jsp> を開きます。以下の Web アプリケーションで適当な数値を入力して [Go!] をクリックします。



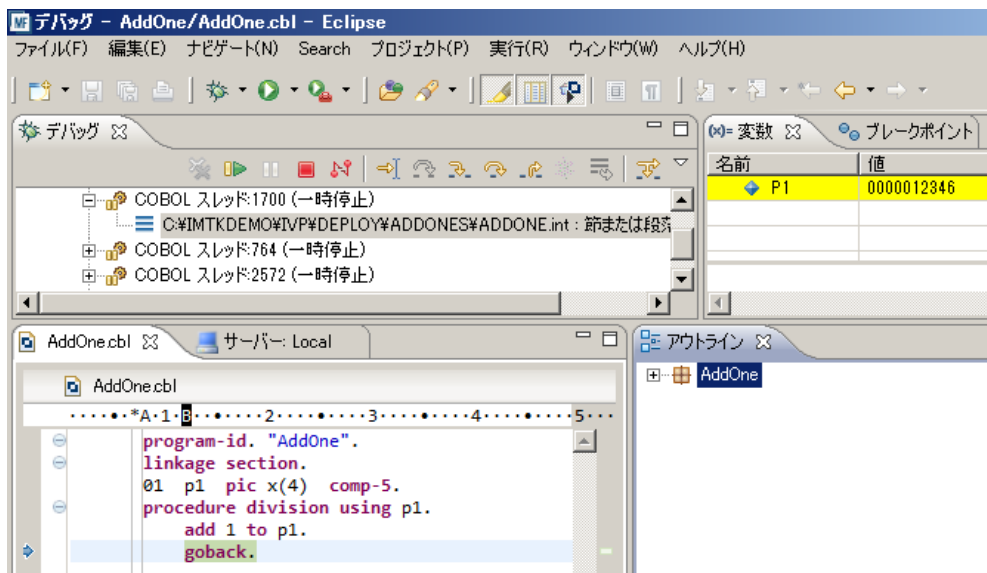
- 15) 待機中のデバッグパースペクティブに制御が移り、以下のように COBOL サービスプログラムの手続き部の先頭がハイライトされます。[変数ビュー] 内では Web アプリケーションで入力した数値がパラメタとして COBOL に渡されていることがわかります。



16) 以下の [ステップイン] ボタンをクリックすると COBOL の実行文がステップ実行されます。



17) ADD 文が実行されて P1 の数値が増加します。



18) もう一度 [ステップイン] ボタンをクリックすると GOBACK 文が実行され COBOL サービスの処理が終了します。このとき Web アプリケーション側に COBOL サービスの実行結果が返ります。

Test client for AddOneS.AddOne

[Back](#)

Perform the test by entering values:

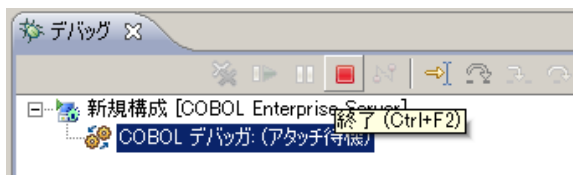
addOne_P1_io :

Go!

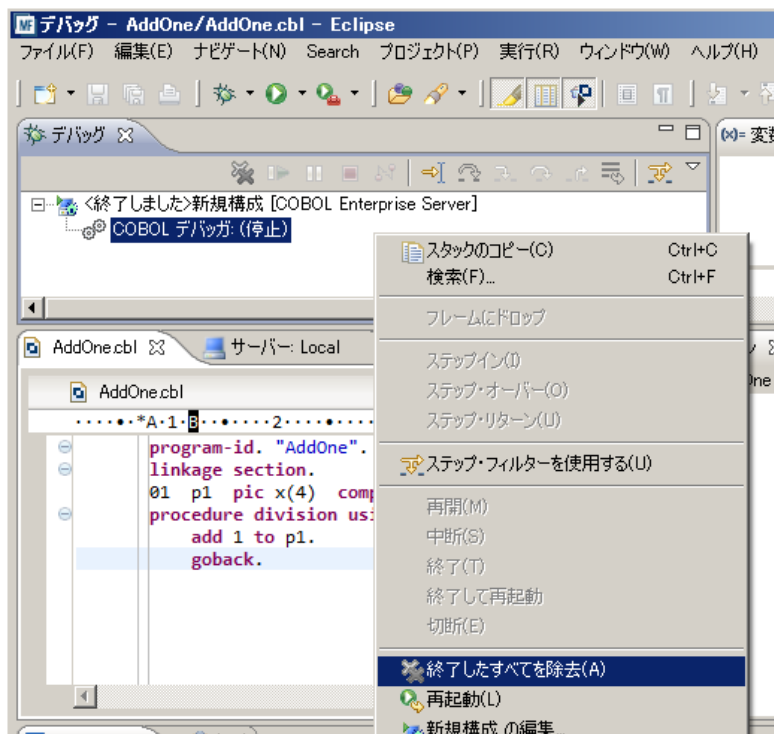
Result:

Variable	Value
Result	12346

- 19) デバッグパースペクティブで以下の [終了] ボタンをクリックしてデバッグセッションを終了します。



- 20) [デバッグ] ビューを右クリックし [終了したすべてを除去] を選択すると、デバッグパースペクティブがクリアされます。ここで COBOL パースペクティブに戻ります。



以上で、Java EE のチュートリアルを終了します。

以上、