WebOTX アプリケーション開発ガイド

WebOTX アプリケーション開発ガイド

バージョン： 7.1

版数： 初版

リリース： 2007 年 7 月

目次

1.デバッグ

1.1.テスト用サーバを使用した一般的なデバッグ方法

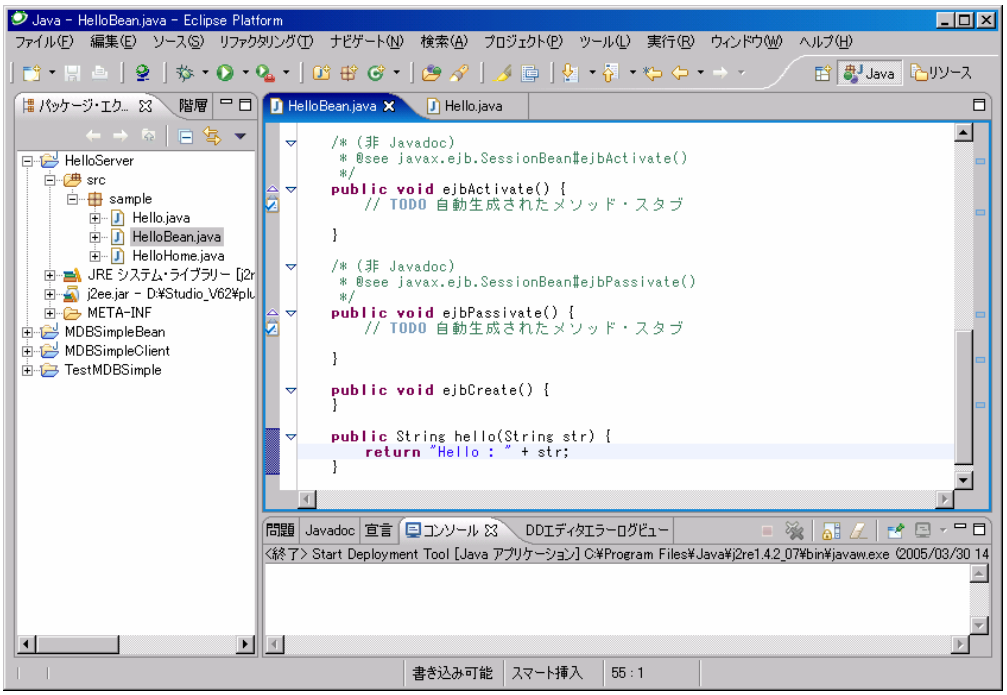
テスト用サーバを使用した一般的なデバッグ方法について記述します。EJB を例に説明します。
（「3.3.1.ステートレスセッション Bean」を使用します。）

1.1.1.設定について

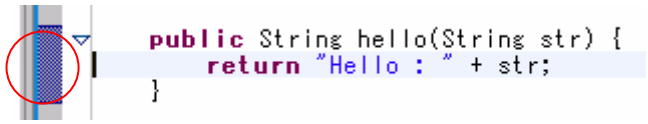
テスト用サーバは、インストールした時点でデバッグできるように設定されています。リモートデバッグのポートの番号として「4004」を設定しています。デバッグポート番号を変更する必要がある場合には、「2.2.1.デバッグ用ポートの設定」を参照して変更してください。

1.1.2.ブレークポイントの設定

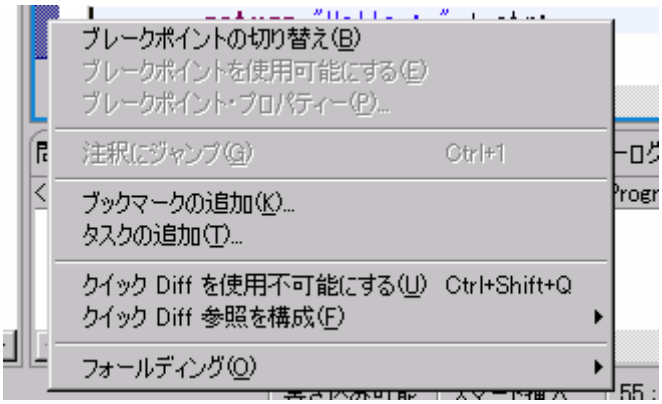
デバッグしたいソースを表示させます。ここでは、HelloBean.java のソースを表示します。



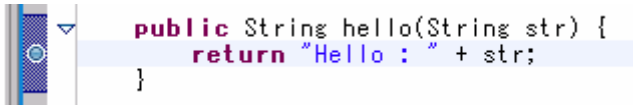
デバッグする行にブレークポイントを設定します。



左を右クリックすると、次の図のように「ブレークポイントの切り替え」のポップアップメニューが表示されます。

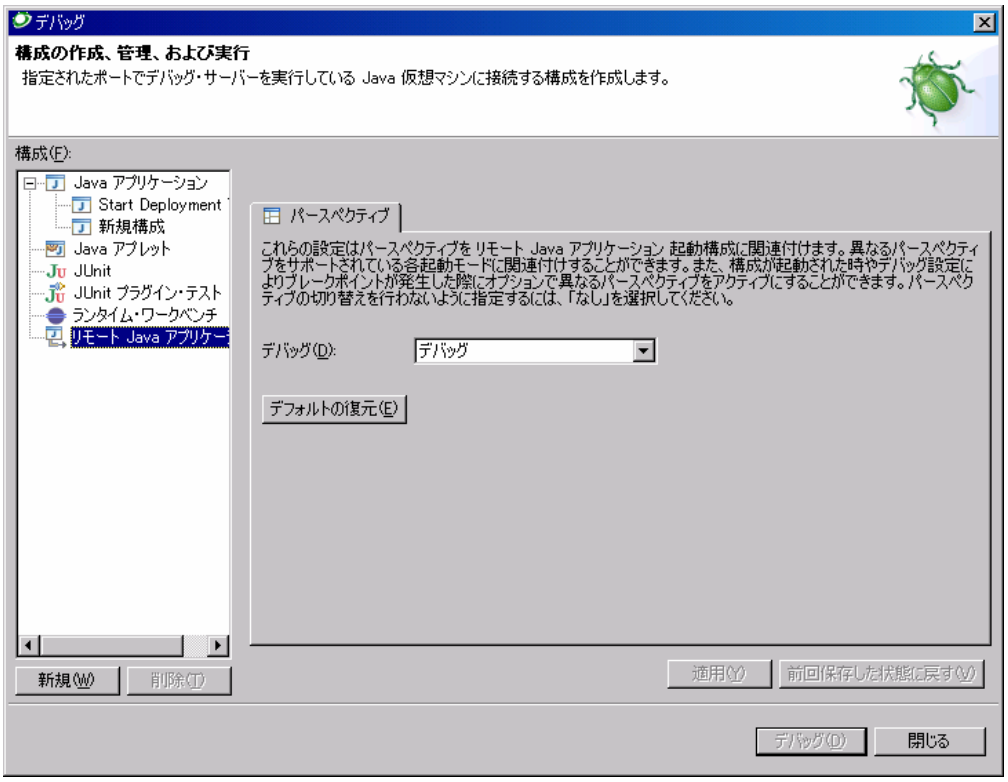


「ブレークポイントの切り替え」を選択すると、次の図のようにソースの左にブレークポイントマークが付きます。

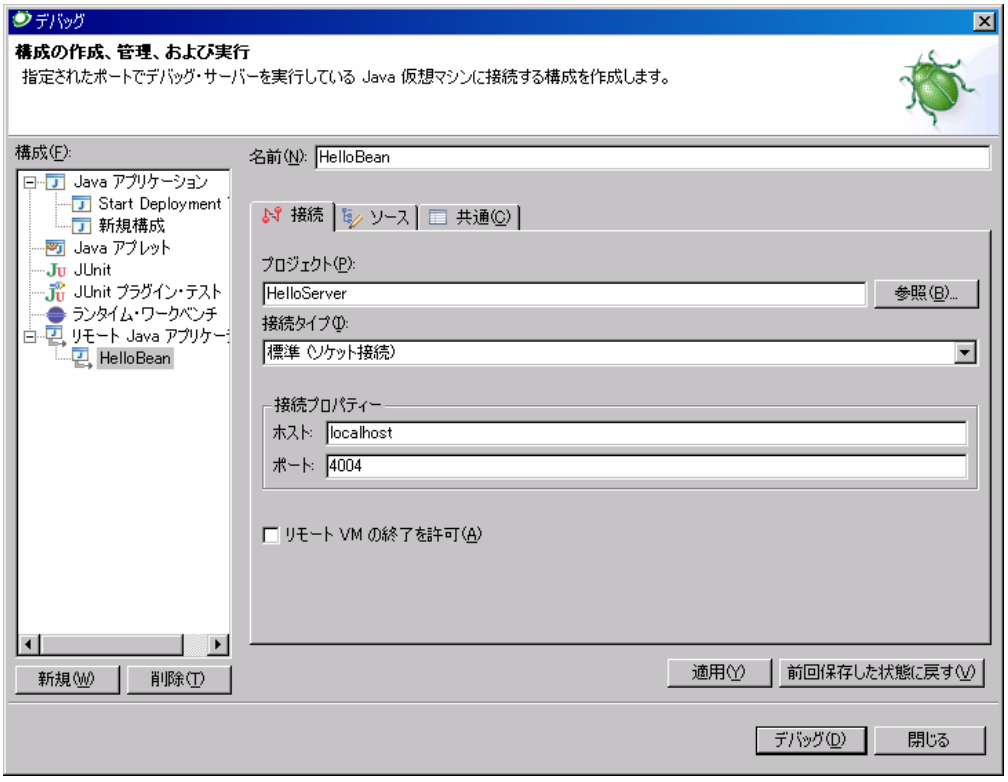


1.1.3.デバッグの実行

「実行」→「デバッグ」メニューを選択します。(EJB アプリケーションを配備して、実行している必要があります。)

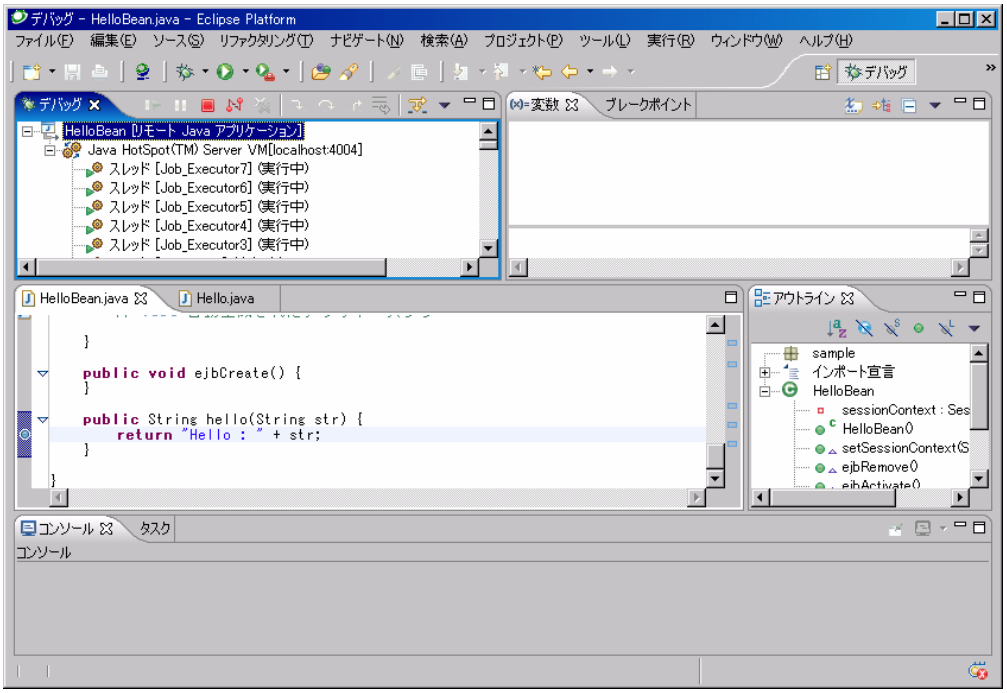


「リモート Java アプリケーション」を選択し、「新規」ボタンを押します。

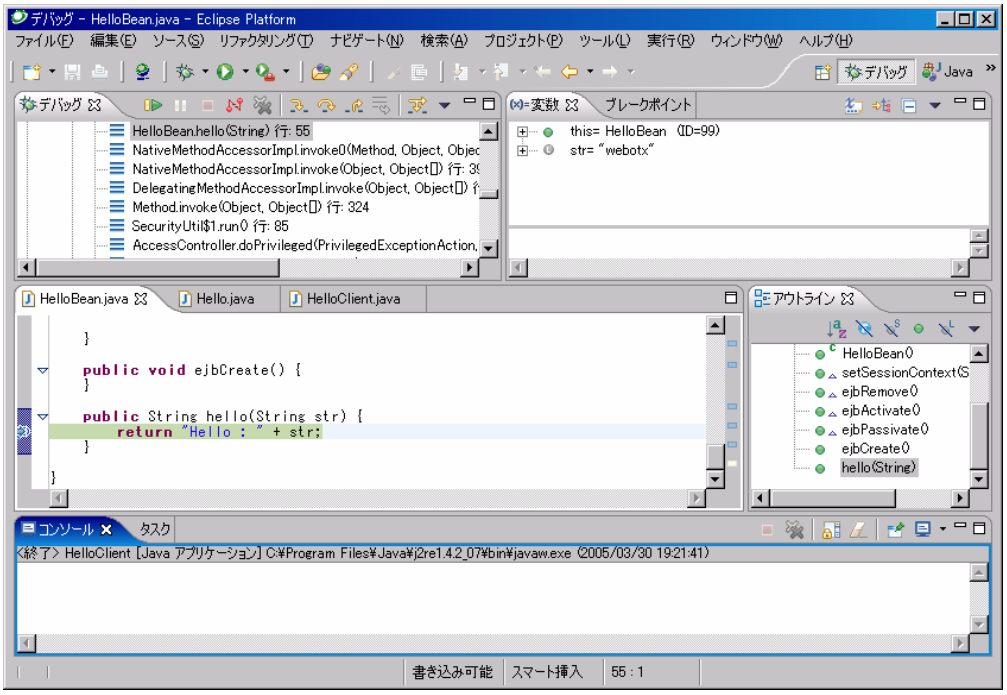


「ポート」に「4004」を設定し、「デバッグ」ボタンを押します。

パースペクティブをデバッグにすると、次の画面が表示されます。

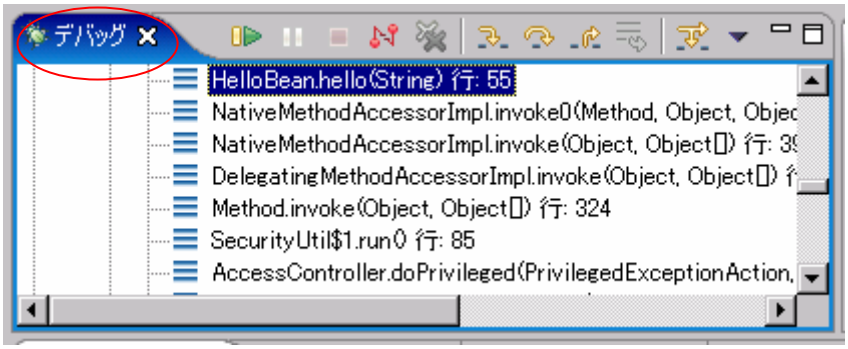


ここで、クライアントプログラムを実行すると、ブレークポイントを設定した箇所で実行が停止します。



1.1.4.デバッグの操作について

以下の図の赤丸部分をクリックし、「F1」キーを押します。



以下のようにポップアップウィンドウが表示されますので、該当のヘルプを参照してください。

1.2.Web サービスアプリケーション

1.2.1.設定と実行手順

設定と実行手順については『**7.1.テスト用サーバを使用した一般的なデバッグ方法**』を参照してください。デバッグを行うにあたって、Web サービスアプリケーション固有の設定や実行手順はありません。

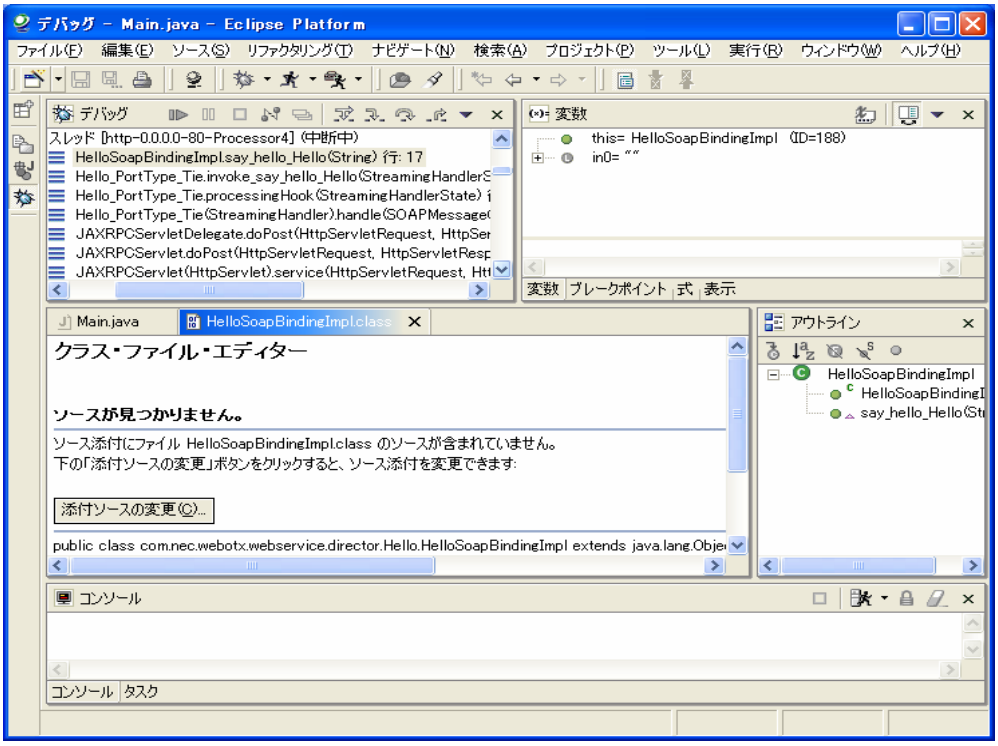
1.2.2.Web サービスをデバッグする

ここでは、Web サービス作成ウィザードを使用して作成した Web サービスアプリケーションのデバッグについて説明します。

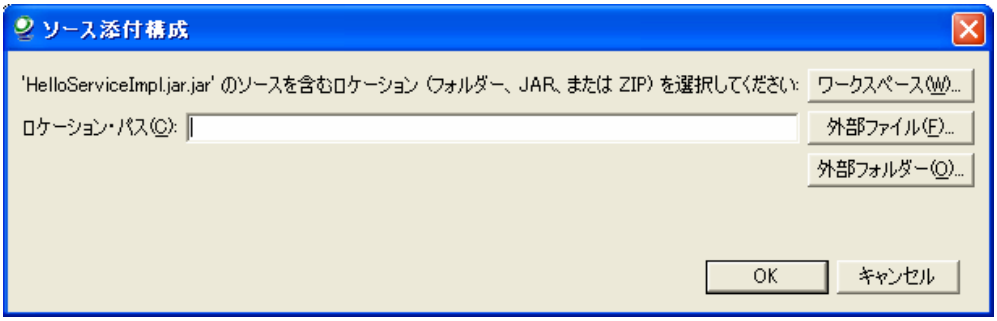
配備済みの Web サービスが正しい処理をして行っているか確認するためには、Web サービスの実装ロジックをデバッグします。Web サービスプロジェクトでは、サーバサイドのソースコードが生成されているので、そのソースコードに対してブレークポイントを追加します。Web サービスの実装形式が Web アプリケーションの場合は<Web サービス名>SoapBindingImpl クラス、EJB の場合は EJB そのものが実装クラスです。ビジネスロジックをデバッグする必要がある場合は、<Web サービス名>SoapBindingImpl クラスかタイクラスにブレークポイントを追加します。<Web サービス名>SoapBindingImpl クラスでは、ウィザードで指定したビジネスロジックのクラス・メソッドを呼び出しているところがありますので、そこにブレークポイントを追加するとよいでしょう。EJB の場合はタイクラスにブレークポイントを追加します。その場合、invoke_ビジネスロジックのメソッド名という名前のメソッド内で設定するとよいでしょう。ブレークポイントの追加を行い、リモートデバッグの設定が終了したら、クライアントを実行してください。デバッグが開始されます。

既存のアプリケーションなどで、ソースコードがない場合は、ソースコードではなくクラス・ファイル・エディターが表示されます。[**添付ソースの変更**]ボタンを押すと、ソース添付構成ダイアログが表示されます。ワークスペース、外部ファイル、外部フォルダのうちから 1 つを選択し、ソースコードが含まれるフォルダまたはファイルを選択するとソースコードが表示されるようになります。

クラス・ファイル・エディター



ソース添付構成ダイアログ



1.3.Web アプリケーション

1.3.1.Servlet のデバッグ

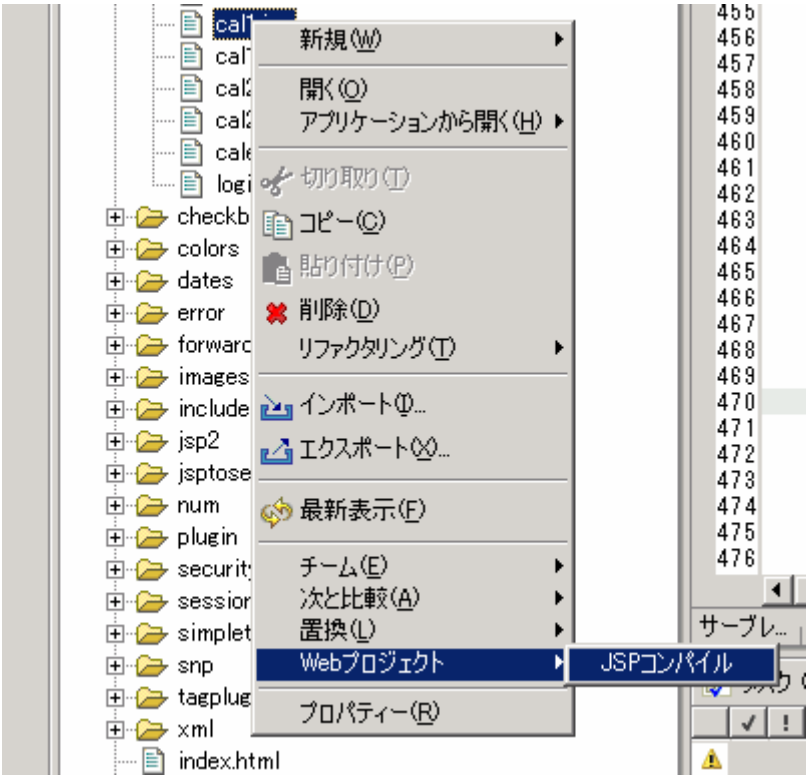
Web アプリケーション中の Servlet のデバッグについては、通常の Java クラスのデバッグと同様ですので、「7.1.テスト用サーバを使用した一般的なデバッグ方法」を参照してください。

1.3.2.JSP のデバッグ

JSP のデバッグは JSP の事前コンパイル機能を利用することで行うことができます。

JSP の事前コンパイルは、パッケージ･エクスプローラーで JSP ファイルを選択して、右クリックで表示されるコンテキストメニューの **Web プロジェクト** | **JSP コンパイル**を選択することで行うことができます。

また、Web プロジェクトを右クリックして、**Web プロジェクト** | **一括 JSP コンパイル**を行うことで Web プロジェクト中のすべての JSP ファイルをコンパイルすることができます。



JSP の事前コンパイルを行うと、コンパイルされた JSP ファイルに対応する Servlet ソースがソースフォルダの org.apache.jsp パッケージ配下に自動的に作成されます。さらに、この Servlet ソースの定義を web.xml ファイルに自動的に追加します。

JSP コンパイルにより作成される Servlet ソースにブレークポイントを設定してデバッグをすることで JSP のデバッグを行います。

1.4.EJB アプリケーション

「7.1.テスト用サーバを使用した一般的なデバッグ方法」を参照してください。

1.5.コネクタアプリケーション

1.5.1.設定と実行手順

設定と実行手順については『**7.1.テスト用サーバを使用した一般的なデバッグ方法**』を参照してください。デバッグを行うにあたって、コネクタアプリケーション固有の設定や実行手順はありません。

1.5.2.コネクタアプリケーションをデバッグする

RAR ファイルに梱包された EIS にアクセスするためのライブラリや Java クラスのデバッグは、デバッグ対象のソースにブレークポイントを設定してデバッグ実行するだけです。ブレークポイントの設定については、『**7.1.2. ブレークポイントの設定**』を参照してください。また、デバッグ実行については、『**7.1.3. デバックの実行**』を参照してください。

OLF/TP アダプタを使用して ACOS（あるいは TP-BASE）に接続するためには、接続ライセンスが必要になりますので、OLF/TP アダプタ実行環境をご購入いただく必要があります。

1.6.テスト用サーバ以外のデバッグ方法

テスト用サーバ以外のサーバ製品でデバッグを行う場合、Edition とデバッグ対象のコンポーネント種類に応じて手順が異なります。Edition とコンポーネント種類に応じて以下の表の通りに『**7.7 WebOTX ドメインで動作するアプリケーションのデバッグ**』または『**7.8 プロセスグループで動作するアプリケーションのデバッグ**』を参照してください。

	Web コンポーネント	EJB コンポーネント
Web Edition	**1.7 WebOTXドメインで動作するアプリケーションのデバッグ**	-
Standard-J Edition	**1.7 WebOTXドメインで動作するアプリケーションのデバッグ**	**1.7 WebOTXドメインで動作するアプリケーションのデバッグ**
Standard、Enterprise Edition(Web コンテナマルチプロセスモード)	**1.8 プロセスグループで動作するアプリケーションのデバッグ**	**1.8 プロセスグループで動作するアプリケーションのデバッグ**
Standard、Enterprise Edition(Web コンテナシングルプロセスモード)	**1.7 WebOTXドメインで動作するアプリケーションのデバッグ**	**1.8 プロセスグループで動作するアプリケーションのデバッグ**

Web サービスアプリケーションの場合は実装形式のコンポーネント種類と同じデバッグ方法になります。

コネクタアプリケーションの場合はそれが使用されるコンポーネントと同じデバッグ方法になります。

1.7.WebOTX ドメインで動作するアプリケーションのデバッグ

デバッグの手順はテスト用サーバと同じですが、WebOTX をデバッグモードで起動する必要があります。以下の otxadmin コマンドを実行の後、ドメインを再起動してください。

```
otxadmin> set -user admin -password adminadmin server.java-config.debug-enabled="true"
```

デバッグ終了後は以下の otxadmin コマンドでデバッグモードを解除し、ドメインを再起動してください。

```
otxadmin> set -user admin -password adminadmin server.java-config.debug-enabled="false"
```

1.8.プロセスグループで動作するアプリケーションのデバッグ

配備されたアプリケーションはドメインと別のプロセス（プロセスグループ）で動作するので、テスト用サーバと同じ方法ではデバッグできません。

1.8.1.設定手順

以下の手順でプロセスグループの JavaVM 起動オプションにデバッグオプションを追加し、プロセスグループを再起動します。デバッグ用ポート番号に 4005 を使用しています。

1. otxadmin> get -user admin -password adminadmin tpsystem.applicationGroups.(アプリケーショングループ名).processGroups.(プロセスグループ名).otherArguments
   を実行し、現在のプロセスグループの JavaVM オプションを取得します。
2. otxadmin> set -user admin -password adminadmin tpsystem.applicationGroups.(アプリケーショングループ名).processGroups.(プロセスグループ名).otherArguments="(get コマンドで得た値) -Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=4005"
   を実行します。

1.8.2.デバッグ

ブレークポイントの設定については、『**7.1.2. ブレークポイントの設定**』を参照してください。

デバッグ実行については、『**7.1.3. デバックの実行**』を参照してください。ただし、上記の手順においてリモートデバッグで使用するポート番号を「4005」と設定しているので、「接続プロパティー」の「ポート」は 4005 にしてください。

1.8.3.注意事項

- リモートデバッグ実行時はデバッグ対象プロセスグループのプロセス数を 1 にしてください。(複数プロセス動作の場合、同じ JavaVM オプションでそれぞれのプロセスが起動するので、使用するポートがバッティングし、最初に起動したプロセスのみがデバッグ対象となってしまいます。)
- 複数のプロセスグループをデバッグする際は、プロセスグループごとにデバッグ用ポート番号を変えてください。
- デバッグ終了後は、以下の otxadmin コマンドで 7.8.1.の 1.で得た JavaVM オプションに戻し、プロセスグループを再起動してください。
  otxadmin> set -user admin -password adminadmin tpsystem.applicationGroups.(アプリケーショングループ名).processGroups.(プロセスグループ名).otherArguments="(get コマンドで得た値) "