JBoss Enterprise SOA Platform 5

BPEL Tools リファレンスガイド

このガイドは開発者向けです。

Last Updated: 2023-03-28

JBoss Enterprise SOA Platform 5 BPEL Tools リファレンスガイド

このガイドは開発者向けです。

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

このガイドは、JBDS の BPEL プラグインの使用方法を開発者に説明します。

目次

はじめに

1. ドキュメント規則

本ガイドでは、いくつかの規則を使用して特定の単語やフレーズを強調表示し、特定の情報への注意を促しています。

1.1. 表記規則

特定の単語や句への注意を促すために 4 つの表記慣習を使用しています。これらの規則や、これらが適用される状況は以下のとおりです。

等幅ボールド

シェルコマンド、ファイル名、パスなど、システム入力を強調表示するために使用されます。キーとキーの組み合わせを強調表示するためにも使用されます。以下に例を示します。

> 現在の作業ディレクトリーのファイル **my_next_bestselling_novel** の内容を表示するには、シェルプロンプトで **cat my_next_bestselling_novel** コマンドを入力し、**Enter** を押してコマンドを実行します。

上記には、ファイル名、シェルコマンドおよびキーが含まれます。これはすべて等幅ボールドで表示され、コンテキストにより区別可能なものになります。

キーの組み合わせは、各部分をつなぐプラス記号によって、個別のキーと区別できます。以下に例を示します。

> **Enter** を押してコマンドを実行します。
>
> **Ctrl**+**Alt**+**F2** を押して、仮想ターミナルに切り替えます。

最初の例では、押す特定のキーを強調表示しています。2 つ目の例は、同時に押す 3 つのキーのセットというキーの組み合わせを強調表示しています。

ソースコードの場合、段落内で記述されるクラス名、メソッド、関数、変数名、および戻り値は、上記のように **等幅ボールド** で示されます。以下に例を示します。

> ファイル関連のクラスには、ファイルシステムの **filesystem**、ファイルの **file**、ディレクトリーの **dir** が含まれます。各クラスには、独自の関連付けられたパーミッションセットがあります。

プロポーショナルボールド

これは、アプリケーション名、ダイアログボックステキスト、ラベルが付いたボタン、チェックボックスおよびラジオボタン、メニュータイトルおよびサブメニュータイトルなど、システムで発生した単語またはフレーズを示します。以下に例を示します。

> メインメニューバーから System → Preferences → Mouse を選択し、Mouse Preferences を起動します。**Buttons** タブで、**Left-handed mouse** チェックボックスを選択し、**Close** をクリックしてメインのマウスボタンを左から右に切り替えます (マウスを左手で使い易くします)。
>
> 特殊文字を gedit ファイルに挿入するには、メインメニューバーから Applications → Accessories → Character Map を選択します。次に、Character Map メニューバーから Search → Find... を選択し、**Search** フィールドに文字の名前を入力して **Next** をクリックします。目的の文字が **Character Table** で強調表示されます。この強調表示し

た文字をダブルクリックして **Text to copy** フィールドに配置し、**Copy** ボタンをクリックします。ここでドキュメントに戻り、**gedit** メニューバーから **Edit → Paste** を選択します。

上記のテキストにはアプリケーション名、システム全体のメニュー名および項目、アプリケーション固有のメニュー名、GUI インターフェイス内のボタンおよびテキストなどがあります。すべては proportional bold で示され、コンテキストと区別できます。

等幅ボールドイタリック または ***プロポーショナルボールドイタリック***

等幅ボールドまたはプロポーショナルボールドのいずれでも、イタリックが追加されると、置換または変数テキストを意味します。イタリックは、状況に応じて変化するテキストや、文字を入力しないテキストを表します。以下に例を示します。

ssh を使用してリモートマシンに接続するには、シェルプロンプトで **ssh *username@domain.name*** を入力します。リモートマシンが **example.com** で、そのマシン上でのユーザー名が john の場合は、**ssh john@example.com** と入力します。

mount -o remount *file-system* コマンドにより、指定したファイルシステムが再マウントされます。たとえば、**/home** ファイルシステムを再マウントする場合、コマンドは **mount -o remount /home** となります。

現在インストールされているパッケージのバージョンを表示するには、**rpm -q *package*** コマンドを使用します。これにより、***package-version-release*** のような結果が返されます。

上記の太字のイタリック体の用語、username、domain.name、file-system、package、version、および release に注意してください。各単語はプレースホルダーで、コマンドの発行時に入力するテキストまたはシステムによって表示されるテキストのどちらかになります。

作業のタイトルを示す標準的な使用法のほかに、イタリックは新用語と重要な用語の最初の使用を示します。以下に例を示します。

Publican は *DocBook* 公開システムです。

1.2. 引用規則

端末の出力およびソースコードの一覧は、周りのテキストから視覚的に表示されます。

ターミナルに送信される出力は **mono-spaced roman** に設定され、以下のように表示されます。

```
books       Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

ソースコードの一覧も **mono-spaced roman** に設定されますが、以下のように構文の強調表示が追加されます。

```
static int kvm_vm_ioctl_deassign_device(struct kvm *kvm,
                 struct kvm_assigned_pci_dev *assigned_dev)
{
        int r = 0;
        struct kvm_assigned_dev_kernel *match;

        mutex_lock(&kvm->lock);

        match = kvm_find_assigned_dev(&kvm->arch.assigned_dev_head,
```

```
                                assigned_dev->assigned_dev_id);
        if (!match) {
                printk(KERN_INFO "%s: device hasn't been assigned before, "
                  "so cannot be deassigned\n", __func__);
                r = -EINVAL;
                goto out;
        }

        kvm_deassign_device(kvm, match);

        kvm_free_assigned_device(kvm, match);

out:
        mutex_unlock(&kvm->lock);
        return r;
}
```

1.3. 注記および警告

最後に、3 つの視覚的スタイルを使用して、見落とす可能性のある情報に注意を促します。

注記

注記とは、タスクへのヒント、ショートカット、または代替アプローチです。注意を無視しても悪い結果を招くことはありませんが、便利なヒントを見逃してしまう可能性があります。

重要

見落としやすい詳細のある重要なボックス: 現行セッションにのみ適用される設定変更や、更新を適用する前に再起動が必要なサービスなどです。Important というラベルが付いたボックスを無視しても、データが失われることはありませんが、スムーズな操作が行えないことがあります。

警告

警告は無視すべきではありません。警告を無視すると、データが失われる可能性があります。

2. ヘルプの利用とフィードバック提供

2.1. ヘルプが必要ですか？

本ガイドで説明されている手順で問題が発生した場合は、Red Hat カスタマーポータル http://access.redhat.com にアクセスしてください。カスタマーポータルでは、以下を行うことができます。

- Red Hat 製品に関する技術サポート記事の知識ベースの検索または閲覧。

- Red Hat グローバルサポートサービス (GSS) へのサポートケースの送信。
- その他の製品ドキュメントへのアクセス。

Red Hat は、Red Hat のソフトウェアおよびテクノロジーについて、多くの電子メーリングリストも提供しています。一般に公開されているメーリングリストの一覧は、https://www.redhat.com/mailman/listinfoを参照してください。メーリングリストの名前をクリックして、その一覧をサブスクライブするか、またはメーリングリストのアーカイブにアクセスします。

2.2. ご意見をお寄せください

本ガイドで誤字脱字を発見されたり、このマニュアルを改善するための提案をお持ちの場合は、弊社までご連絡ください。JBoss Enterprise SOA Platform の製品に対して、Bugzilla: http://bugzilla.redhat.com/ レポートを送信してください。

バグレポートを送信する際には、マニュアル識別子である『BPEL_Tools_Reference_Guide』を指定してください。

本ガイドを改善するためのご意見やご提案をお寄せいただく場合は、できるだけ具体的にご説明ください。エラーが見つかった場合は、簡単に確認できるように、セクション番号と前後のテキストを含めてください。

第1章 はじめに

1.1. ビジネス統合

動的かつアジャイルなビジネスインフラストラクチャーを提供するためには、異なるアプリケーションとデータソースが最小限のオーバーヘッドで相互に通信できるように、サービス指向のアーキテクチャーを用意することが重要です。

JBoss Enterprise SOA Platform は、ビジネスプロセスの変更に対応するために、継続的に再利用することなく、ビジネスサービスをオーケストレーションできるフレームワークです。JBoss Enterprise SOA Platform では、ビジネスルールとメッセージの変換およびルーティング機能を使用することで、複数のソースからビジネスデータを操作できます。

バグの報告

1.2. サービス指向アーキテクチャーとは

はじめに

SOA (*Service Oriented Architecture*) は単一のプログラムまたはテクノロジーではありません。ソフトウェア設計パラダイムと見なします。

すでに分かっているように、*ハードウェアバス* は、複数のシステムとサブシステムを結び付ける物理コネクターです。これを使用する場合は、システムのペア間で多数のポイントツーポイントコネクターを使用する代わりに、各システムを中央バスに接続するだけです。*エンタープライズサービスバス* (ESB) は、ソフトウェアとまったく同じことを行います。

アーキテクトは、メッセージングシステムの上記のアーキテクチャー層にあります。このメッセージングシステムは、このメッセージングシステムを使用することでサービス間の *非同期通信* を容易にします。実際、アーキテクトを使用している場合、すべてを概念的に、*サービス*（このコンテキストではアプリケーションソフトウェア）またはサービス間で送信される *メッセージ* のいずれかです。サービスは接続アドレスとして一覧表示されます (*エンドポイント参照* と呼ばれています)。

このコンテキストでは、サービスは必ずしも Web サービスであるとは限らないことに注意することが重要です。File Transfer Protocol や Java Message Service などのトランスポートを使用する他のタイプのアプリケーションもサービスになる可能性があります。

注記

この時点で、エンタープライズサービスバスがサービス指向のアーキテクチャーと同じ場合は、妨げられる場合があります。回答は Not exactly です。 アーキテクトは、サービス指向のアーキテクチャーを形成しません。代わりに、ツールを多数使用して構築できます。特に、SOA が必要とする *loose-coupling* および *非同期メッセージを渡すこと* が容易になります。SOA は常にソフトウェア以外のものと考えてください。これは一連の原則、パターン、およびベストプラクティスです。

バグの報告

1.3. サービス指向アーキテクチャーの重要なポイント

以下は、サービス指向のアーキテクチャーの主要なコンポーネントです。

1. 交換される *メッセージ*
2. サービスリクエスターおよびプロバイダーとして動作する *エージェント*
3. メッセージを送受信できるようにする *共有トランスポートメカニズム*。

バグの報告

1.4. JBOSS ENTERPRISE SOA PLATFORM とは

JBoss Enterprise SOA Platform は、エンタープライズアプリケーションインテグレーション (EAI) およびサービス指向アーキテクチャー (SOA) ソリューションを開発するためのフレームワークです。これは、エンタープライズサービスバス (JBoss Warehouse) およびビジネスプロセス自動化インフラストラクチャーで設定されます。これにより、ビジネスサービスの構築、デプロイ、統合、オーケストレーションを行うことができます。

バグの報告

1.5. SERVICE-ORIENTED ARCHITECTURE PARADIGM

SOA (Service-ient Architecture)は、リクエスター、プロバイダー、およびブローカーの 3 つのロールで設定されます。

サービスプロバイダー

サービスプロバイダーはサービスへのアクセスを許可し、サービスの説明を作成し、サービスブローカーに公開します。

サービスリクエスター

サービスリクエスターは、サービスブローカーが提供するサービスの説明を検索して、サービスを検出します。リクエスターはサービスプロバイダーが提供するサービスにバインドするロールも果たします。

サービスブローカー

サービスブローカーは、サービスの説明のレジストリーをホストします。リクエスターをサービスプロバイダーにリンクします。

バグの報告

1.6. コアおよびコンポーネント

JBoss Enterprise SOA Platform は、データ統合のニーズに対応する包括的なサーバーを提供します。基本的なレベルでは、Enterprise Service Bus を介してビジネスルールを更新し、メッセージをルーティングすることができます。

JBoss Enterprise SOA Platform の中心となるのは、Enterprise Service Bus です。JBoss (ESB) はメッセージの送受信を行う環境を作成します。メッセージにアクションを適用して変換し、サービス間でルーティングすることができます。

JBoss Enterprise SOA Platform を設定するコンポーネントは複数あります。ESB とともに、レジストリ (jUDDI)、変換エンジン (Smooks)、メッセージキュー (HornetQ)、BPEL エンジン (Riftsaw) が用意されています。

バグの報告

1.7. JBOSS ENTERPRISE SOA PLATFORM のコンポーネント

- 完全な Java EE 準拠のアプリケーションサーバー (JBoss Enterprise Application Platform)
- Enterprise Service Bus (JBoss ESB)
- ビジネスプロセス管理システム (jBPM)
- ビジネスルールエンジン (JBoss ルール)
- オプションの JBoss Enterprise Data Services (EDS) 製品のサポート。

バグの報告

1.8. JBOSS ENTERPRISE SOA PLATFORM の機能

JBoss Enterprise Service Bus (ESB)

ドメインはサービス間でメッセージを送信し、それらを変換して、異なるタイプのシステムで処理できるようにします。

Business Process Execution Language (BPEL)

Web サービスを使用して、この言語を使用してビジネスルールをオーケストレーションできます。これは、ビジネスプロセス命令を簡単に実行するために SOA に含まれています。

Java Universal Description、Discovery and Integration (jUDDI)

これは SOA のデフォルトサービスレジストリーです。ここで説明する内容は、インストラクター上のサービスに関する情報をすべて保管する場所です。

Smooks

この変換エンジンは SOA と組み合わせて使用してメッセージを処理できます。また、メッセージを分割して正しい宛先に送信するためにも使用できます。

JBoss ルール

これは、SOA にパッケージ化されたルールエンジンです。受信するメッセージからデータを推測して、実行する必要があるアクションを判別できます。

バグの報告

1.9. JBOSS ENTERPRISE SOA PLATFORM の JBOSSESB コンポーネントの機能

JBoss Enterprise SOA Platform の JBossESB コンポーネントは以下をサポートします。

- 複数のトランスポートおよびプロトコル
- リスナーアクションモデル (これにより、サービスを相互に選択可能)
- コンテンツベースのルーティング (JBoss Rules エンジン、XPath、Regex、および Smooks 経由)
- サービスオーケストレーション機能を提供するために JBoss Business Process Manager (jBPM) との統合
- ビジネスルールの開発機能を提供するために、JBoss ルールとの統合
- BPEL エンジンとの統合。

さらに、新しいデプロイメントにレガシーシステムを統合し、同期または非同期で通信できるようにします。

さらに、エンタープライズサービスバスは、以下を可能にするインフラストラクチャーおよびツールセットを提供します。

- さまざまなトランスポートメカニズム (電子メールや JMS など) と連携するよう設定されます。
- 汎用オブジェクトリポジトリーとして使用します。
- プラグ可能なデータ変換メカニズムを実装できるようにします。
- 対話のロギングをサポートします。

重要

ソースコードには、**org.jboss.internal.soa.esb** と **org.jboss.soa.esb** の 2 つのツリーがあります。**org.jboss.internal.soa.esb** パッケージの内容をそのまま使用します。これは、通知なしに変更される可能性があるためです。これとは対照的に、**org.jboss.soa.esb** パッケージ内のすべての内容は、Red Hat の非推奨ポリシーでカバーされます。

バグの報告

1.10. タスク管理

JBoss SOA は、影響を受けるすべてのシステムで汎用的に実行するタスクを指定することにより、タスクを簡素化します。つまり、ユーザーは各ターミナルで個別に実行するようにタスクを設定する必要はありません。ユーザーは、Web サービスを使用してシステムを簡単に接続できます。

JBoss SOA を使用して各マシンに複数回ではなく、ネットワーク全体でトランザクションを 1 回委譲することで、時間を節約できます。これにより、エラーが発生する可能性も低くなります。

バグの報告

1.11. 統合のユースケース

ACME Equity は、大規模な経理サービスです。会社には多くのデータベースやシステムがあります。旧式の COBOL ベースのレガシーシステムや、近年で小規模な企業で取得されるデータベースもあります。ビジネスルールが頻繁に変化するため、これらのデータベースを統合することは困難でコストがかかります。会社は、クライアント向け e-commerce の Web サイトを新たに開発することを希望していますが、現在使用している既存のシステムとは同期しない場合があります。

会社は、安価なソリューションを希望していますが、企業セクターの厳密な規制およびセキュリティー要件に準拠するソリューションを検討しています。会社が望ましくないことは、レガシーデータベースおよびシステムを接続するために glob コードを作成および維持する必要があることです。

JBoss Enterprise SOA Platform は、これらのレガシーシステムを新しいお客様の Web サイトに統合するためにミドルウェアレイヤーとして選択されました。フロントエンドシステムとバックエンドシステムの間のブリッジを提供します。JBoss Enterprise SOA Platform で実装されたビジネスルールは、すぐに簡単に更新できます。

その結果、SOA の統合方法により、古いシステムは新しいシステムと同期できるようになりました。1 カ月あたり数万のトランザクションであっても、ボトルネックはありません。XML、JMS、FTP などのさまざまな統合タイプは、システム間でデータを移動するために使用されます。数多くのエンタープライズ標準のメッセージングシステムの 1 つを JBoss Enterprise SOA Platform にプラグインすることで、柔軟性を高めることができます。

さらに利点は、既存のインフラストラクチャーにより多くのサーバーやデータベースが追加されると、システムを簡単にスケールアップできることです。

バグの報告

1.12. ビジネス環境での JBOSS ENTERPRISE SOA PLATFORM の使用

エラーメッセージが発生する可能性が低いサービスの実装により、コストを削減できます。生産性とソーシングオプションの強化により、継続的なコストを削減できます。

情報およびビジネスプロセスは、接続が増加するため、迅速に共有できます。これは Web サービスによって強化され、クライアントを簡単に接続するために使用できます。

レガシーシステムは Web サービスと組み合わせて使用して、異なるシステムが同じ言語にピークにすることができます。これにより、システムの同期に必要なアップグレードおよびカスタムコードの量が減ります。

バグの報告

第2章 はじめに

2.1. 本ガイドの対象者

本書は、JBoss BPEL のツールの使用方法を学びたい開発者を対象としています。新しいプロジェクトの作成方法、プロジェクトのデバッグ方法、およびエディターの使用方法について説明します。

バグの報告

2.2. ガイドの目的

このガイドは、プロジェクトで BPEL を使用する方法の概要をユーザーに提供することを目的としています。ユーザーには、プロジェクトを作成し、それらのプロジェクトを編集するために従うべき手順が示されます。また、さまざまな種類のエディターとエラーメッセージの処理方法についても学びます。

バグの報告

2.3. インストール

手順2.1 タスク

1. **Help** -> **Install New Software...** -> **Add...** をクリックして、名前と場所 (例: https://devstudio.jboss.com/updates/5.0/staging/soa-tooling/) を挿入します。次に、**OK** をクリックします。
2. **BPEL Editor** を選択したら、**Next** をクリックし、**Next**クリックします。
3. ライセンス同意書に同意します。
4. **Finish** をクリックして JBDS を再起動します。(コンピューターを再起動する必要はありません。)
5. JBoss BPEL Editor は、再起動後に利用できるようになります。

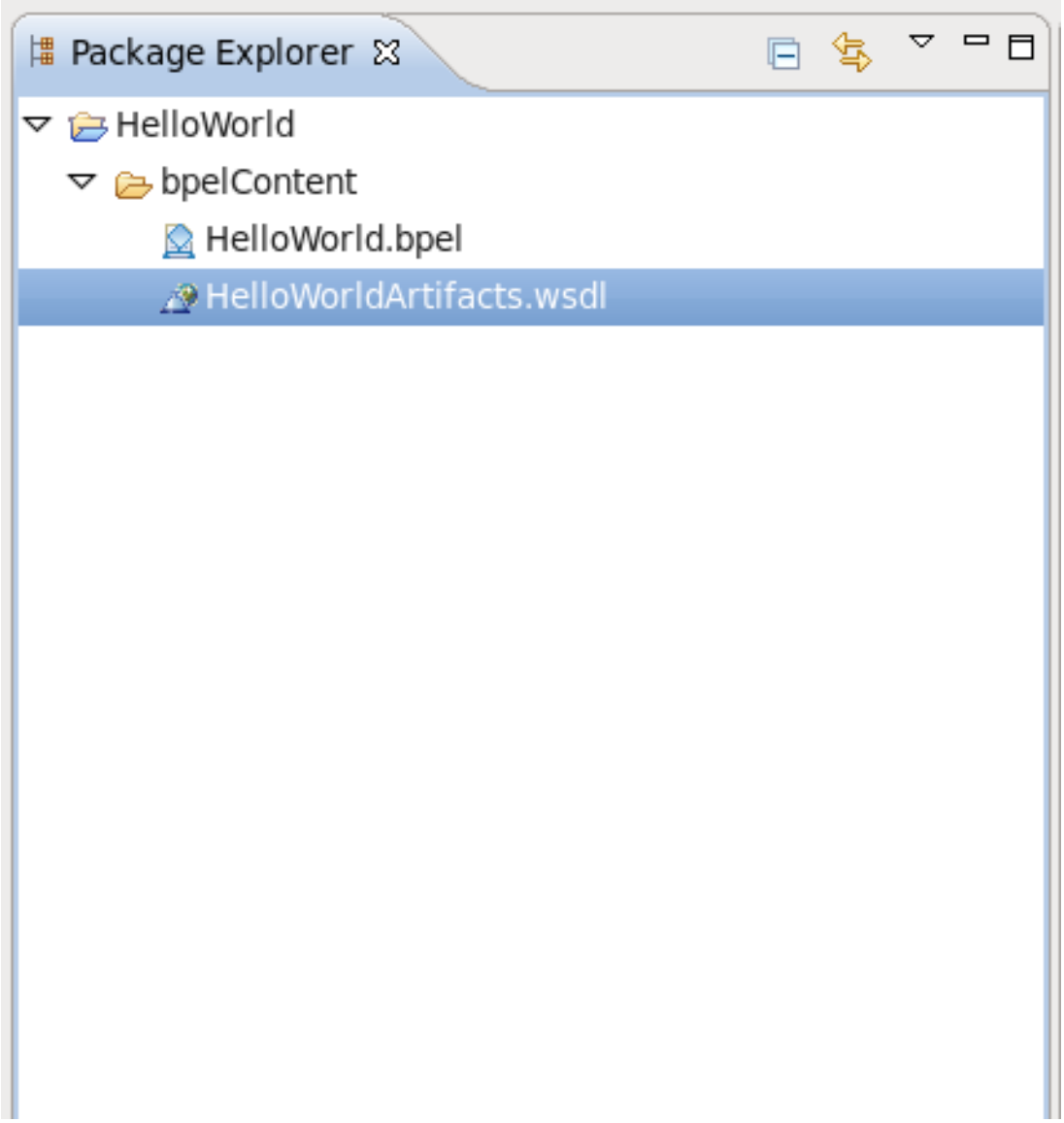
バグの報告

第3章 タスク

3.1. BPEL プロジェクトの作成

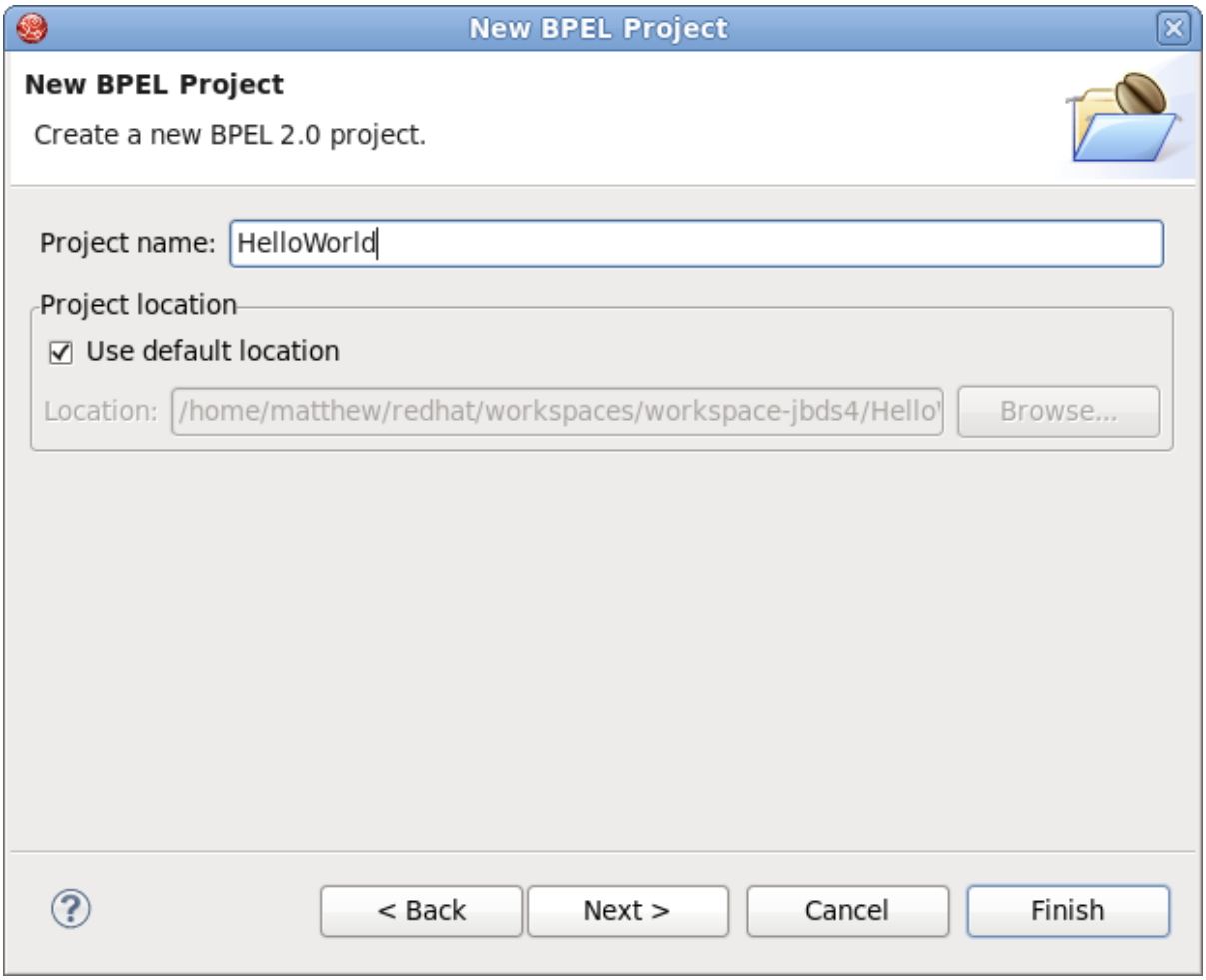
1. まず、メニューバーから **File** → **New** → **Project...** → **BPEL 2.0** → **BPEL Project** または **Legacy BPEL Project** を選択します。次に、**Next** ボタンをクリックします。

図3.1 図 1



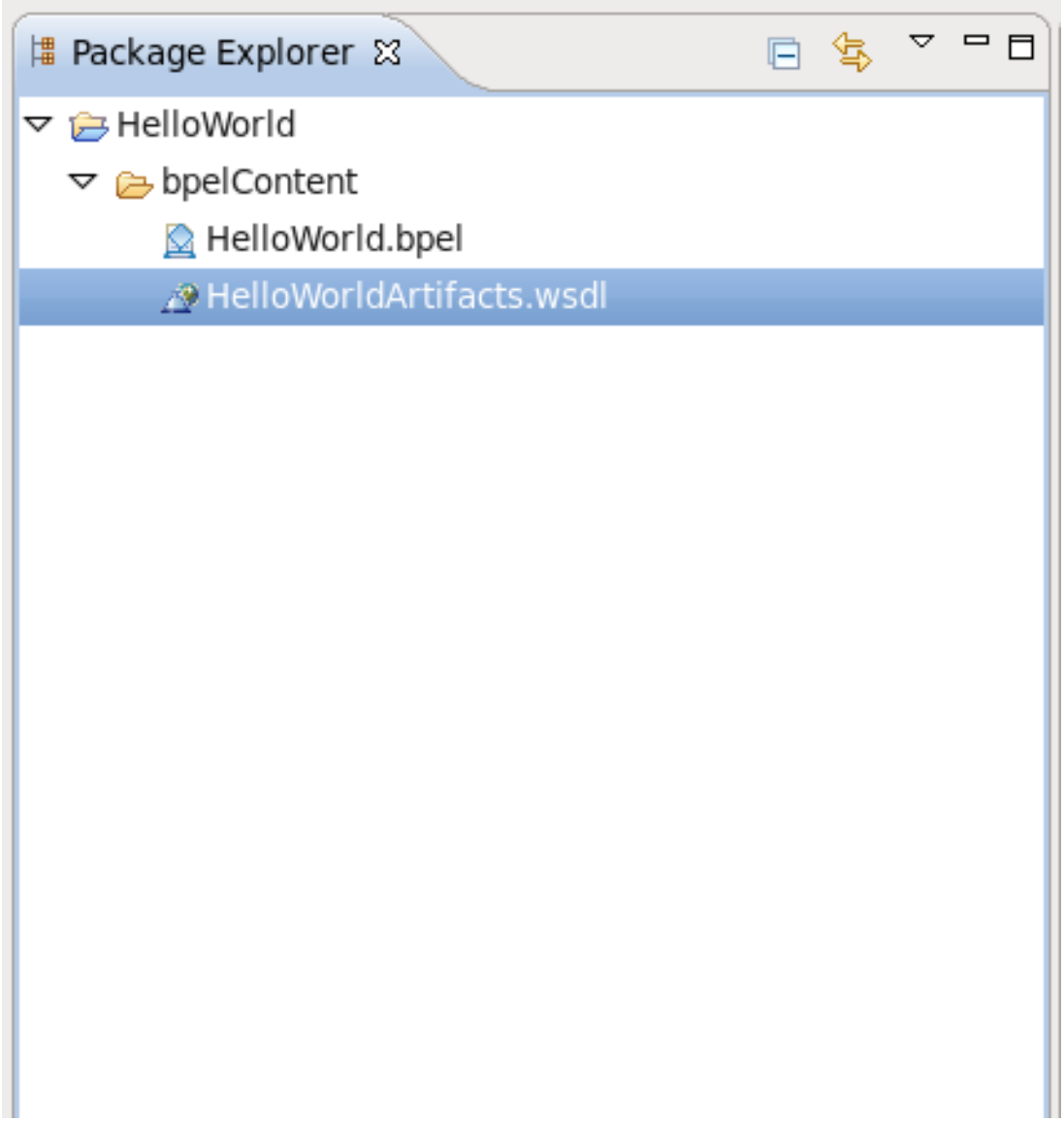
2. Project Name フィールドにプロジェクト名を入力します。

図3.2 図 2



3. **Finish** ボタンをクリックします。以下の画面が表示されます。

図3.3 図 3



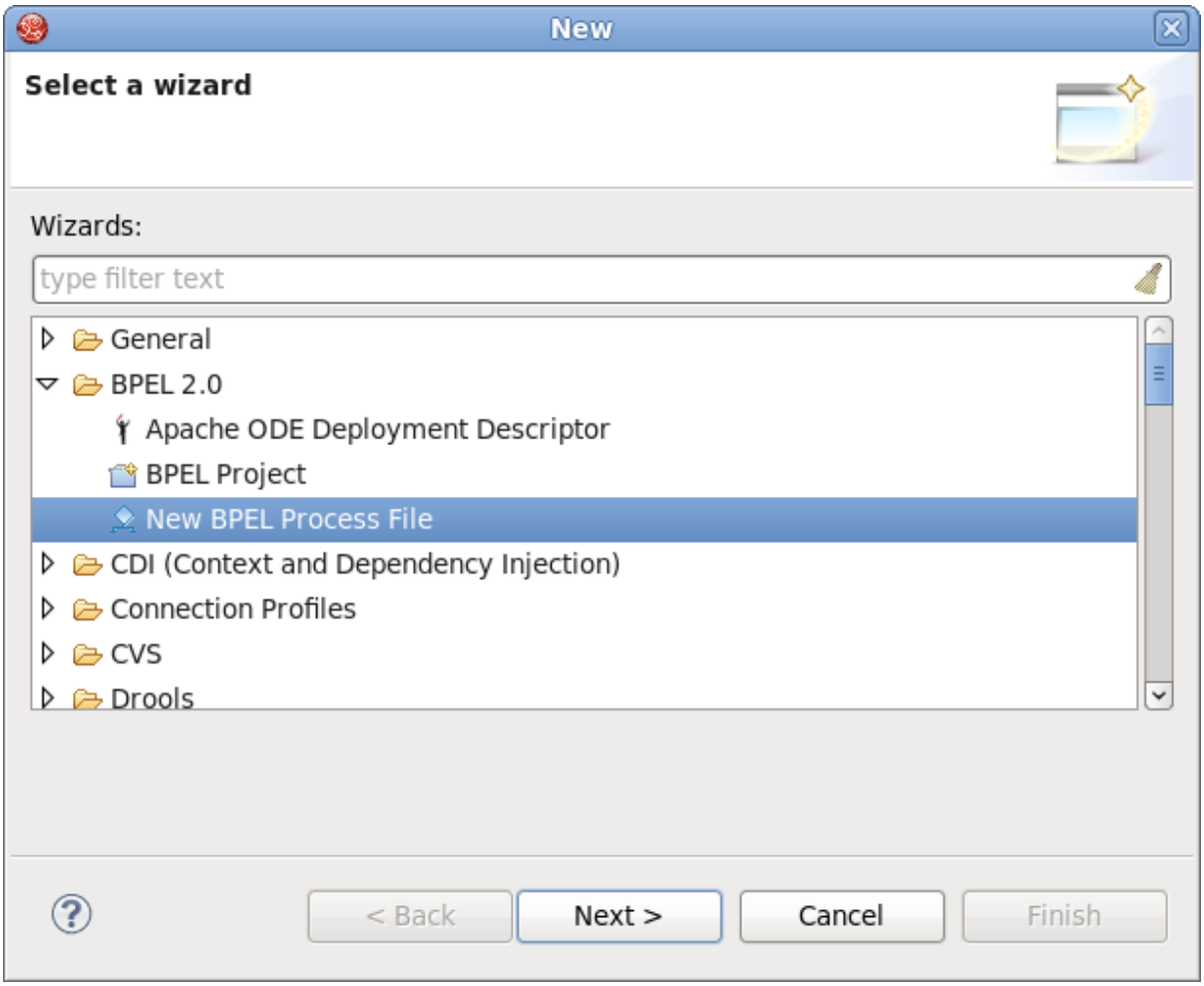
4. これで新規プロジェクトが作成されました。

バグの報告

3.2. BPEL プロセスの作成

1. まず、**File → New → Others... → BPEL 2.0 → BPEL Process File** を選択し、**Next** をクリックします。

図3.4 図 1



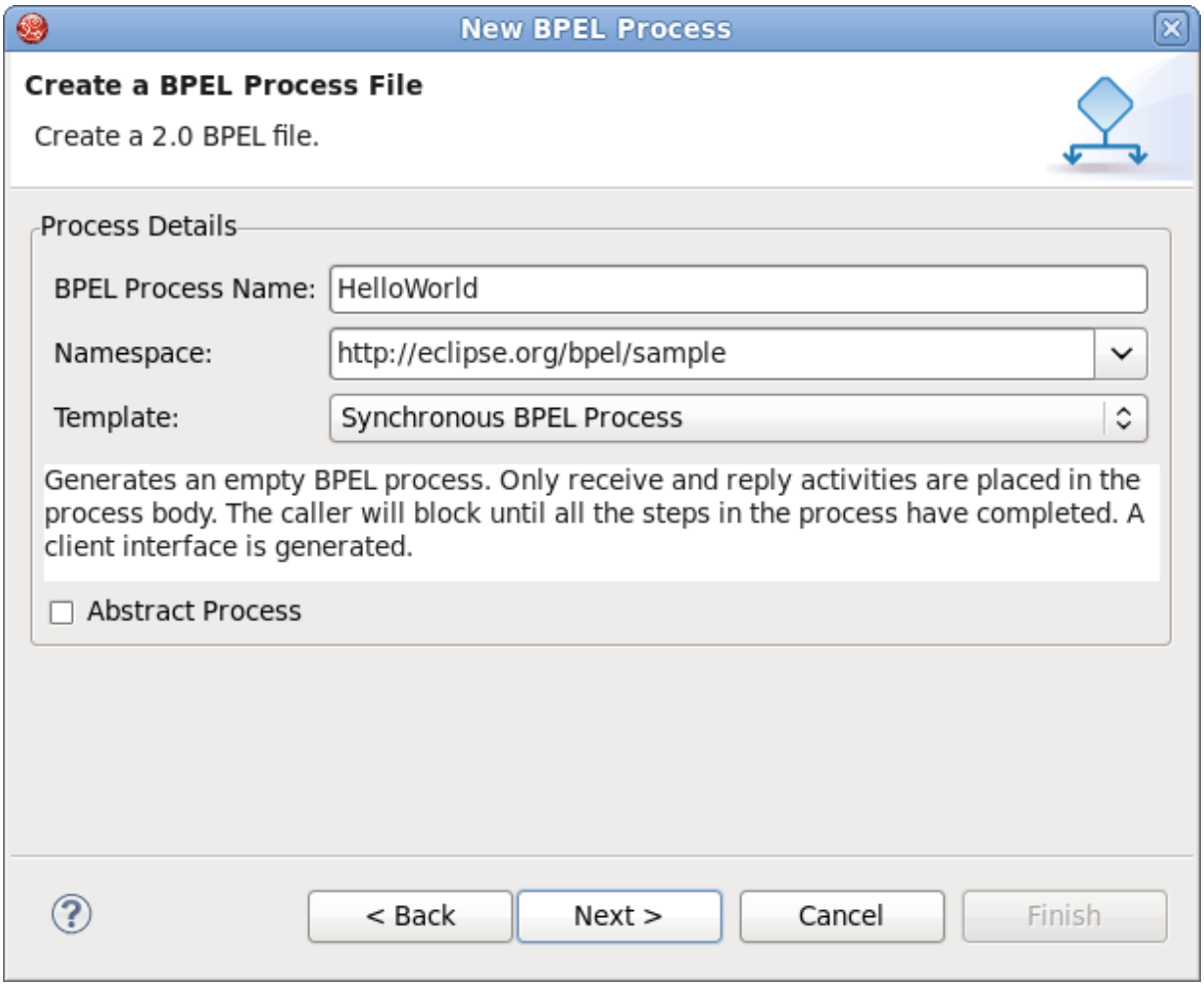
2. ここから、BPEL プロセスをテンプレートから作成するか、サービス記述から作成するかを選択できます。前者を推奨します。

3. 以下の情報を入力します。

表3.1 フィールドと値

フィールド	値
BPEL プロセス名	プロセス名を入力します。例: HelloWorld
Namespace	BPEL プロセスの namespace を入力または選択します。
Template	BPEL プロセスに適したテンプレートを選択します。テンプレートを選択すると、その情報が表示されます。**Synchronous BPEL Process** を選択します。

図3.5 図 2

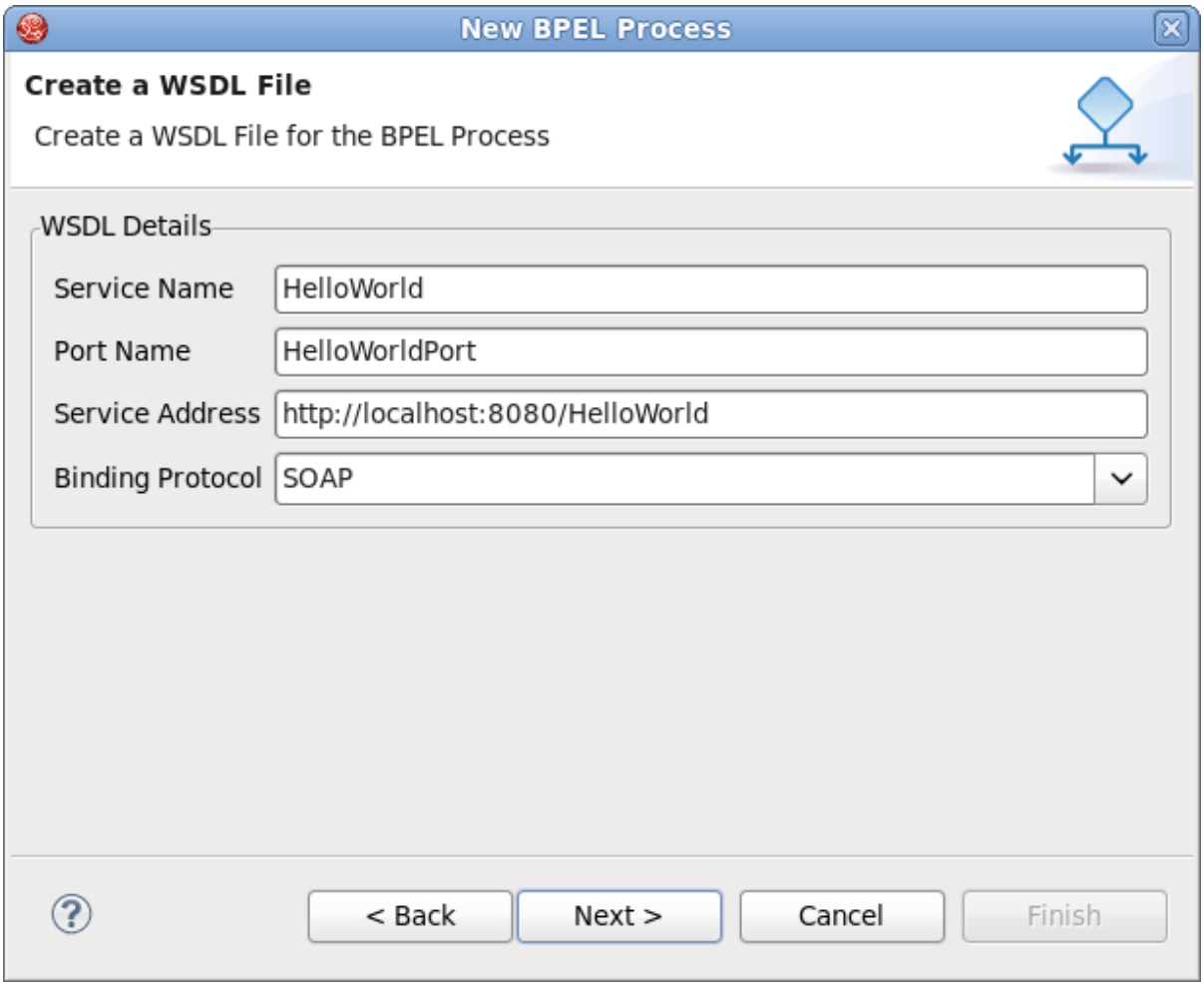


4. **Next** をクリックします。このページでは、テンプレートを使用して WSDL サービスの詳細をカスタマイズできます。以下の情報を入力します。

表3.2 フィールドと値

フィールド	値
サービス名	BPEL プロセスの WSDL サービス名。デフォルトの名前は **HelloWorld** です。
ポート名	BPEL プロセスの WSDL ポート名。デフォルトの名前は **HelloWorldPort** です。
サービスアドレス	BPEL プロセスの WSDL サービスのアドレス。デフォルト値は **http://localhost:8080/HelloWorld** です。
バインドプロトコル	WSDL で使用するバインドプロトコル。SOAP または HTTP を選択できます。デフォルト値は **SOAP** です。

図3.6 図 3



5. **Next** ボタンをクリックします。このページでは、ワークスペース内のプロジェクトからプロセスファイルのフォルダーを選択できます。フォルダーを選択しないと、デフォルトのフォルダー **HelloWorld/bpelContent** が使用されます。

6. **Finish** ボタンをクリックします。プロセスは完了です。

注記

BPEL プロジェクトで使用されるすべてのファイルは、BPEL プロジェクトの **bpelContent** フォルダー下にある必要があります。

バグの報告

3.3. サーバーランタイムの新規作成

手順3.1 タスク

1. **New Server** ウィザードに移動します。

2. **Add** をクリックします。

3. 必要に応じて名前を入力します (名前は事前に設定されているため、この手順はオプションです。

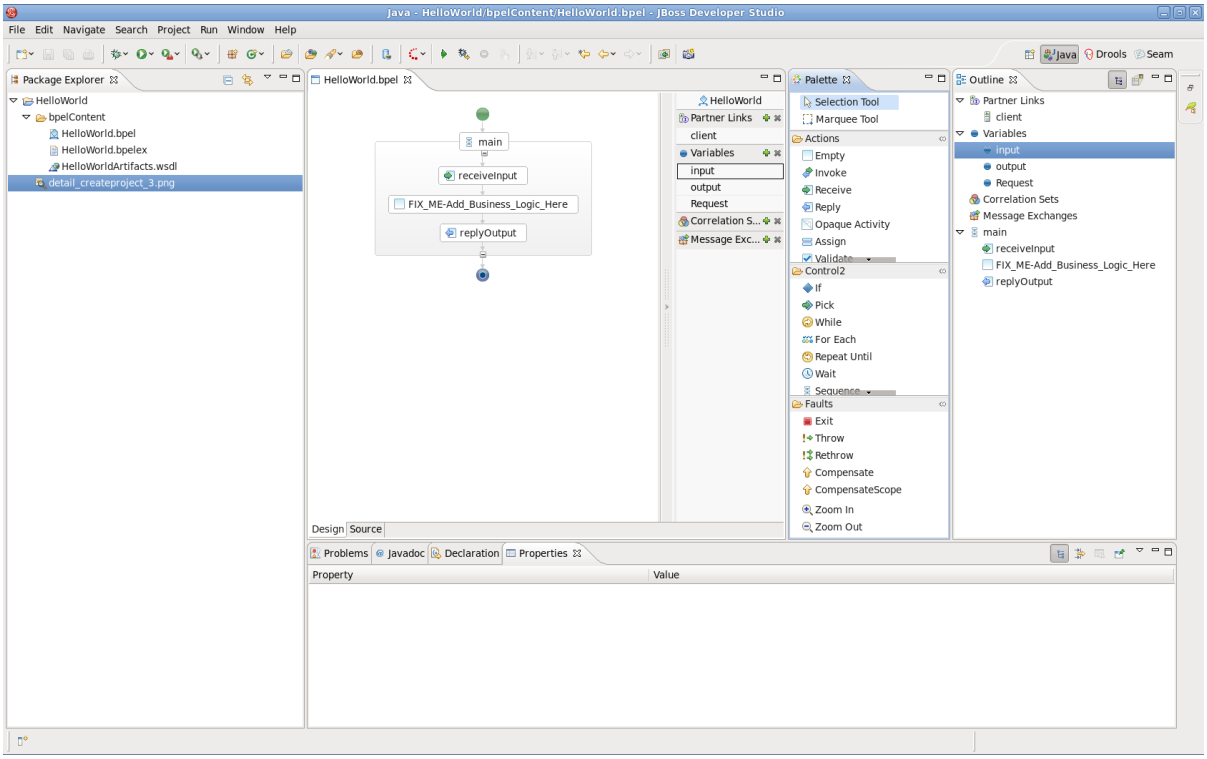
4. **Browse** をクリックし、ホームディレクトリーを選択します。
5. 使用可能な設定の 1 つを選択します (選択した設定では、BPEL エンジンが使用可能である必要があります)。
6. **Finish** をクリックします。

バグの報告

3.4. BPEL プロセスファイルの編集

1. BPEL エディターを右クリックし、**Show in Properties** または **Show Palette in Palette view** オプションを選択して、**Properties view** と **Palette view** を開きます。

図3.7 図 1



2. **Palette** ビューで、選択した BPEL 要素を BPEL エディターにドラッグアンドドロップします。
3. **Properties** ビューに切り替えて、BPEL プロセスに関する情報を表示します。
4. BPEL エディターで要素が選択されると、**Properties** ビューの内容が自動的に更新されます。

バグの報告

3.5. PROPERTIES ビューに表示されるタブ

表3.3 Properties ビューに表示されるタブ

タブ	説明
Description	名前、サイズなど、要素に関する情報を表示します。
Details	要素に関する詳細で重要な情報を表示します。要素のプロパティーのほとんどは、このセクションで設定されます。
Join Behavior	要素の Join Failure プロパティーを表示します。
Documentation	要素のドキュメントサブ要素を表示します。
Imports	BPEL プロセス定義にインポートするドキュメントを選択できます。
Namespaces	BPEL プロセスドキュメントで定義された namespace を編集できます。

バグの報告

3.6. BPEL プロセスの監視

1. 要素 Empty と receiveInput の間の replyOutput 要素を Assign に変更します。
2. Assign 要素と receiveInput 要素の間に replyOutput 要素を追加します。
3. BPEL エディターの Assign 要素をクリックして、Properties ビューのプロパティー情報を表示します。
4. Description タブで要素の名前を assignHelloMesg に設定します。
5. Properties ビューの Details セクションで New ボタンをクリックして、 copy サブ要素を要素に追加します。Variable to Variable を割り当てます (input:string から result:string)。 initializer ポップアップダイアログが表示されます。 Yes ボタンをクリックします。
6. 目的のコンポーネントに移動し、クリックします。コンポーネント名の左側にあるアイコンは、そのタイプを示します。青い点は BPEL 変数、エンベロープはメッセージ、e は XML 要素です。**New** ボタンをもう一度クリックし、**Expression to Variable** を選択します (**concat($input.payload/tns:input, ' World')** を **result:string** に割り当てます)。

バグの報告

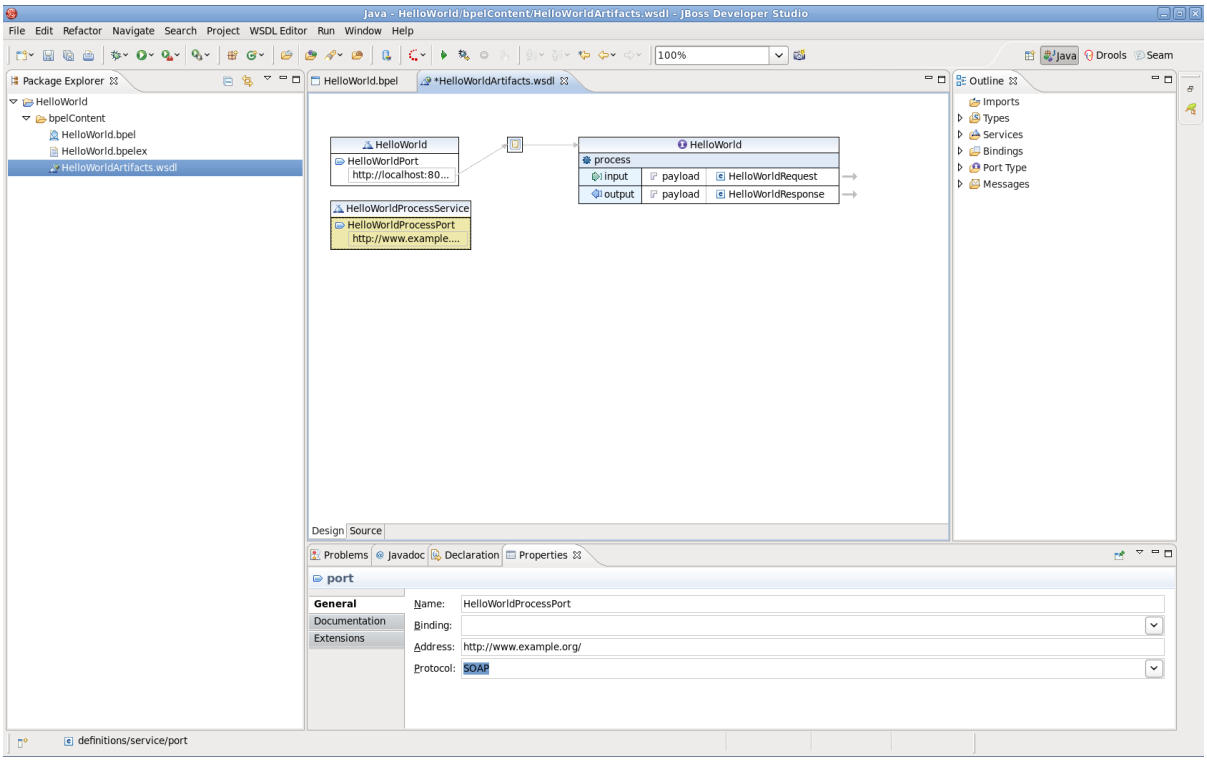
3.7. WSDL ファイルへのサービスの追加

BPEL プロセスファイルを作成すると、**HelloWorldArtifacts.wsdl** ファイルがサービスに追加されます。この WSDL ファイルには、デフォルトサービスがすでに定義されています。ただし、独自のサービスを追加する場合は、以下の手順に従います。

1. **HelloWorld** プロジェクトの **HelloWorldArtifacts.wsdl** ファイルを開きます。

2. WSDL エディターを右クリックし、**Add Service** オプションを選択します。新しいサービスがエディターに表示されます。**HelloWorldProcessService** という名前を付けます。**NewPort** という名前のポートがあります。選択および右クリックし、**Properties** ビューで名前を **HelloWorldProcessPort** に変更します。

   図3.8 図 1

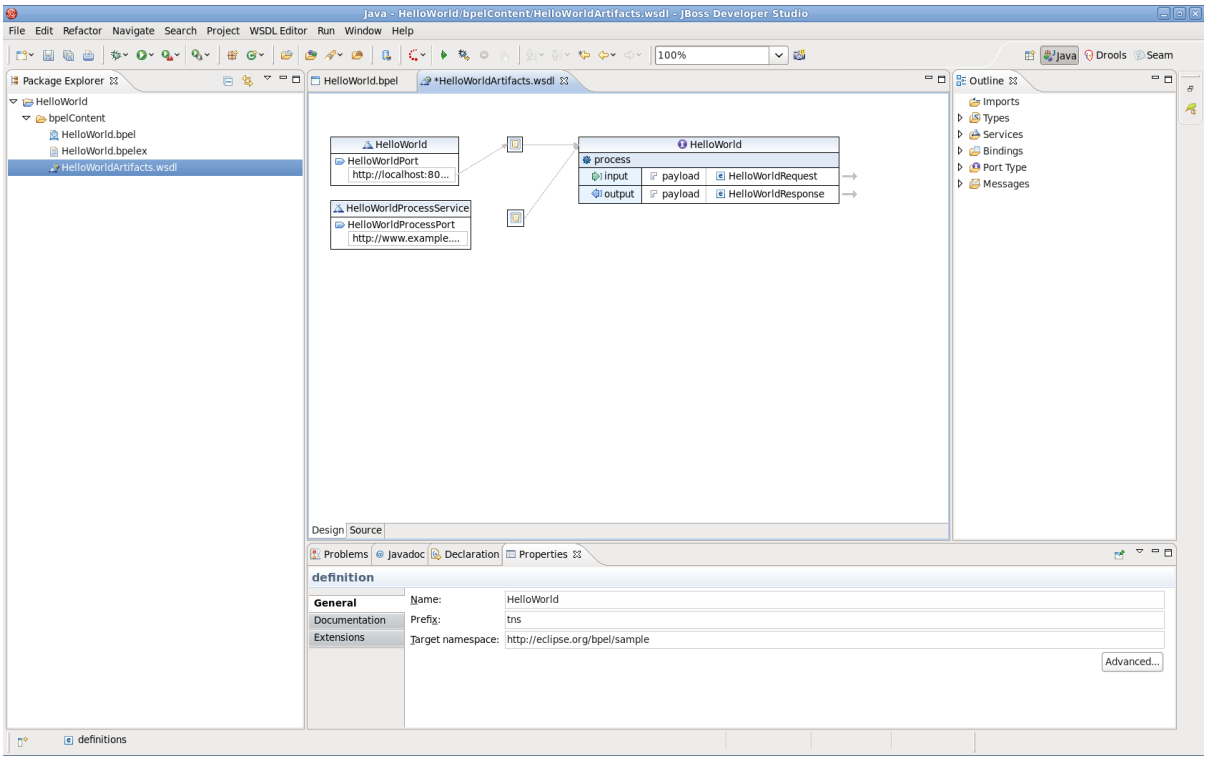


3. WSDL エディターの空白を右クリックし、**Add Binding** オプションを選択します。新規のバインドコンポーネントがエディターに表示されます。**HelloWorldSOAPBinding** という名前を付けます。これを選択し、**Properties** ビューの **General** タブで、**PortType** フィールドのポートタイプとして **HelloWorld** を選択します。

4. **Generate Binding Content...** ボタンをクリックして **Binding Wizard** を開きます。

5. ウィザードで **SOAP** を **Protocol** として選択します。**Finish** ボタンをクリックしてウィザードを閉じます。

図3.9 図 2

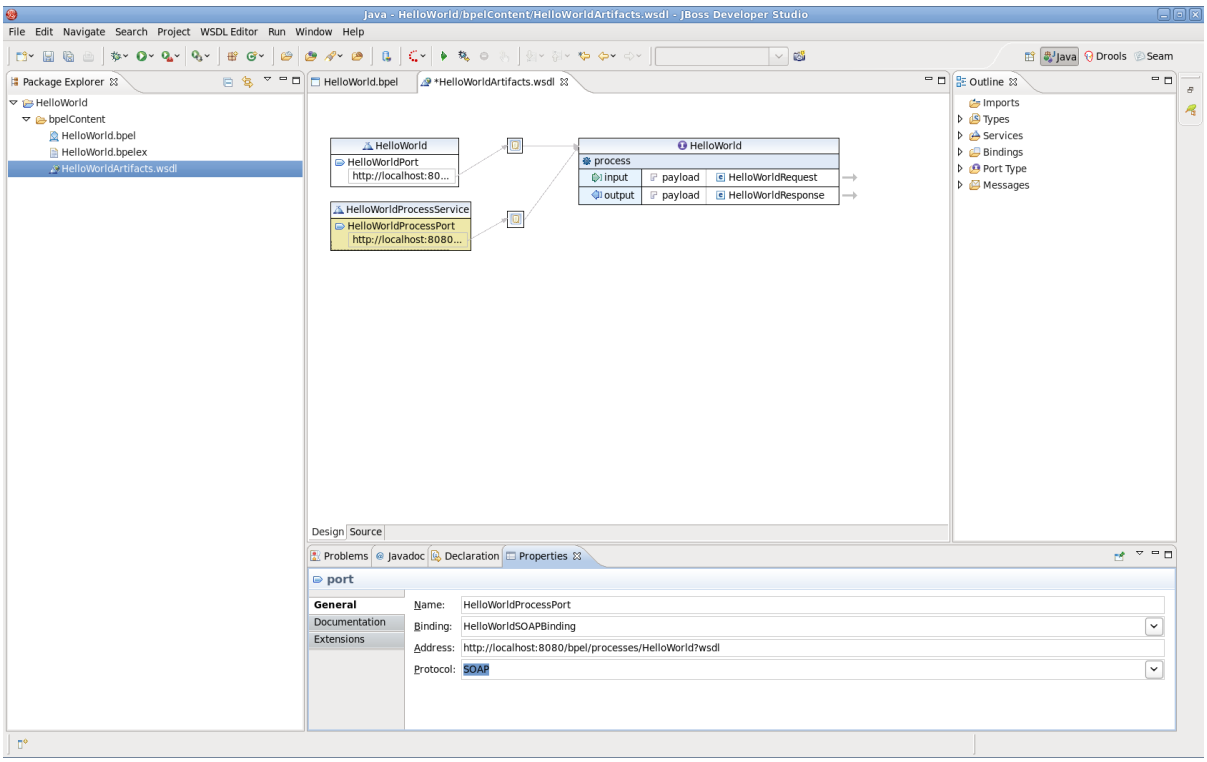


6. **Properties** ビューの **General** セクションで、**HelloWorldProcessPort** プロパティーをクリックします。

7. **Binding** コンボボックスで **HelloWorldSOAPBinding** を選択します。

8. **Address** フィールドに http://localhost:8080/bpel/processes/HelloWorld?wsdl と入力します。

図3.10 図 3

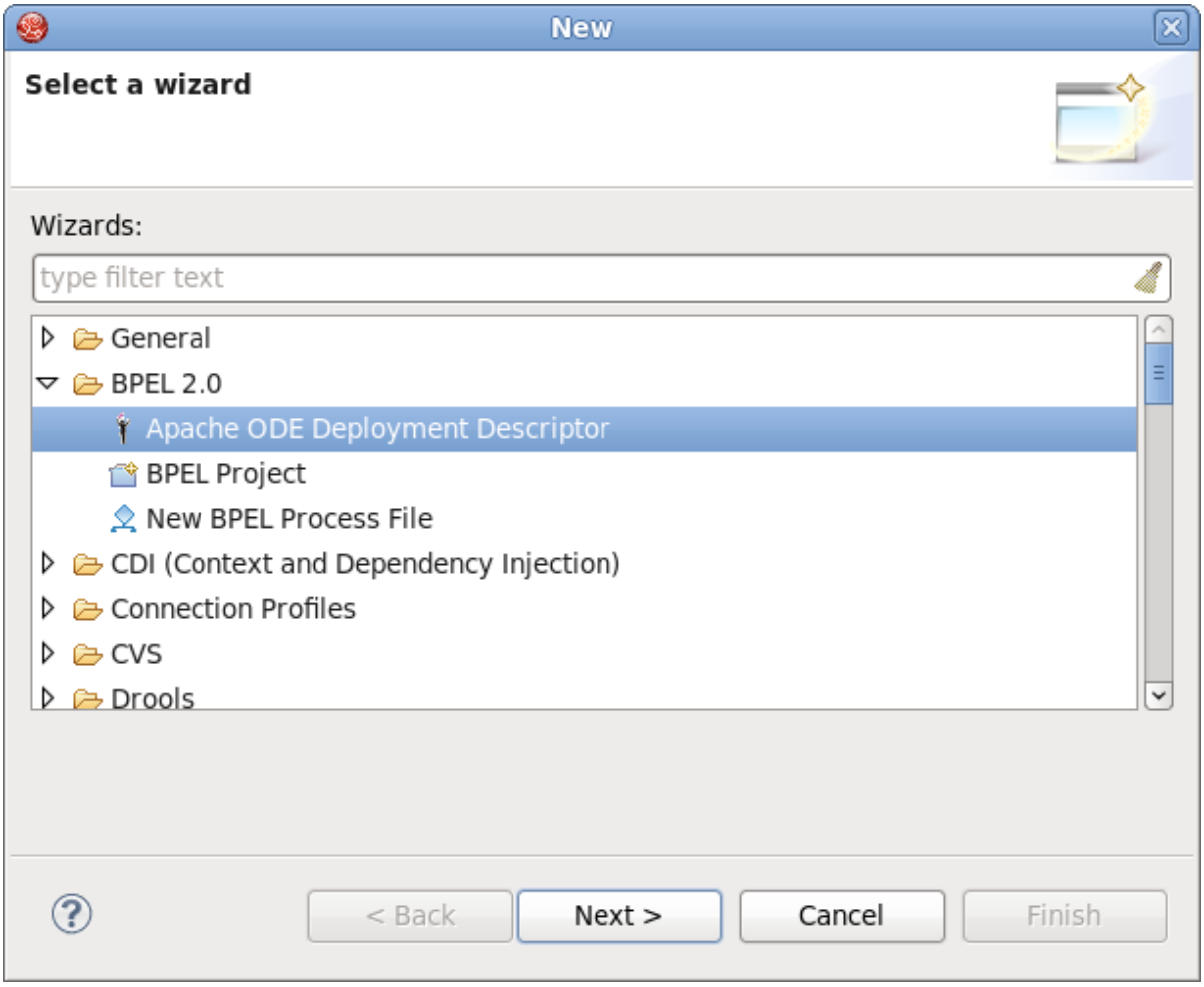


バグの報告

3.8. DEPLOY.XML ファイルの作成

1. BPEL プロジェクトをデプロイするための新しい **deploy.xml** ファイルを作成するには、**File → New → Other... → BPEL 2.0 → BPEL Deployment Descriptor** を選択します。**Next** ボタンをクリックします。

図3.11 図 1

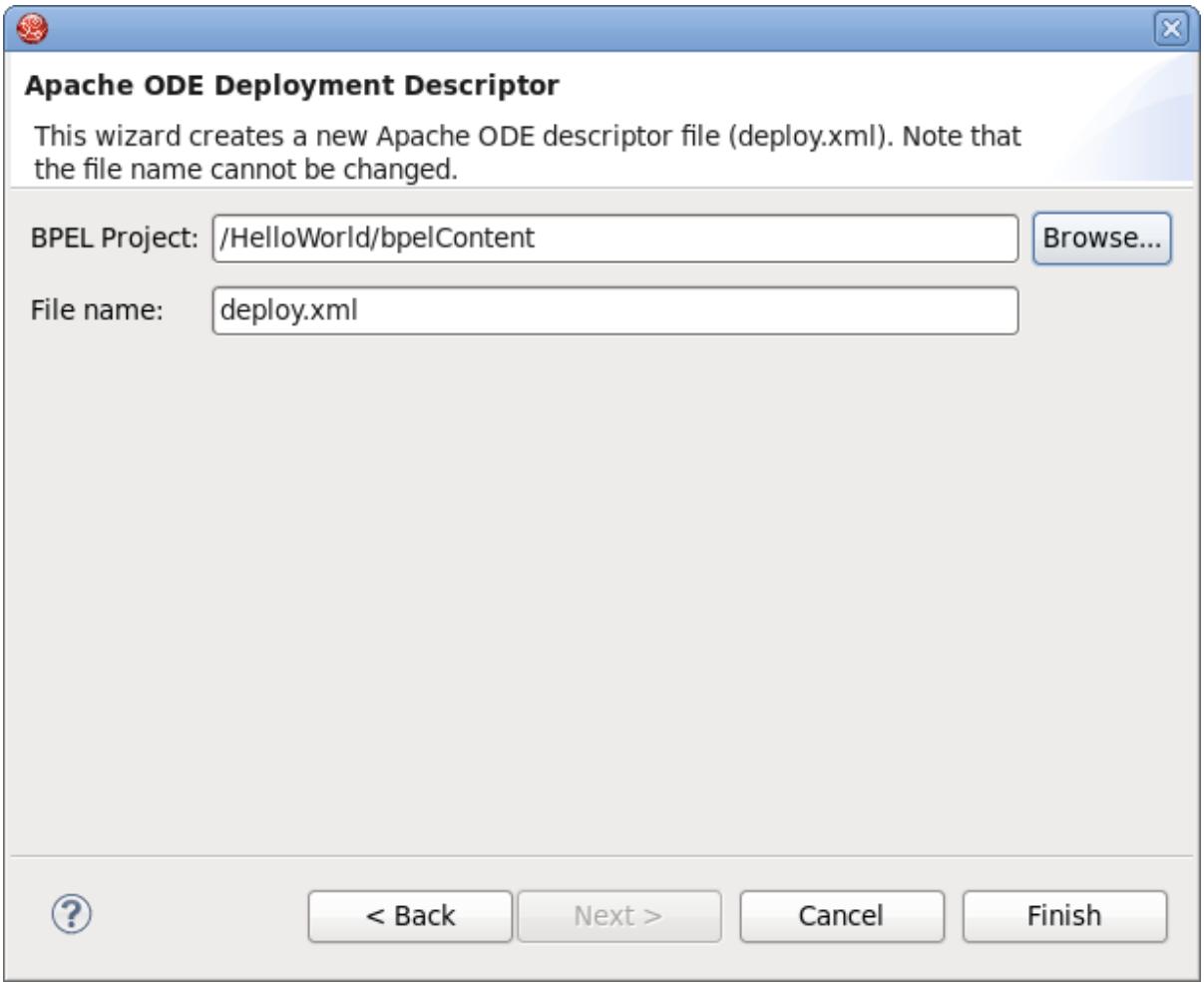


2. ウィザードのこのページで、 BPEL Project を入力します。**Browse...** ボタンをクリックして、ランタイムにデプロイするワークスペース内の BPEL プロジェクトを選択します。

3. BPEL Project フィールドには、新しい BPEL プロジェクト内の **bpelContent** フォルダーを選択します。**deploy.xml** というデフォルトのファイル名は、変更しないでください。

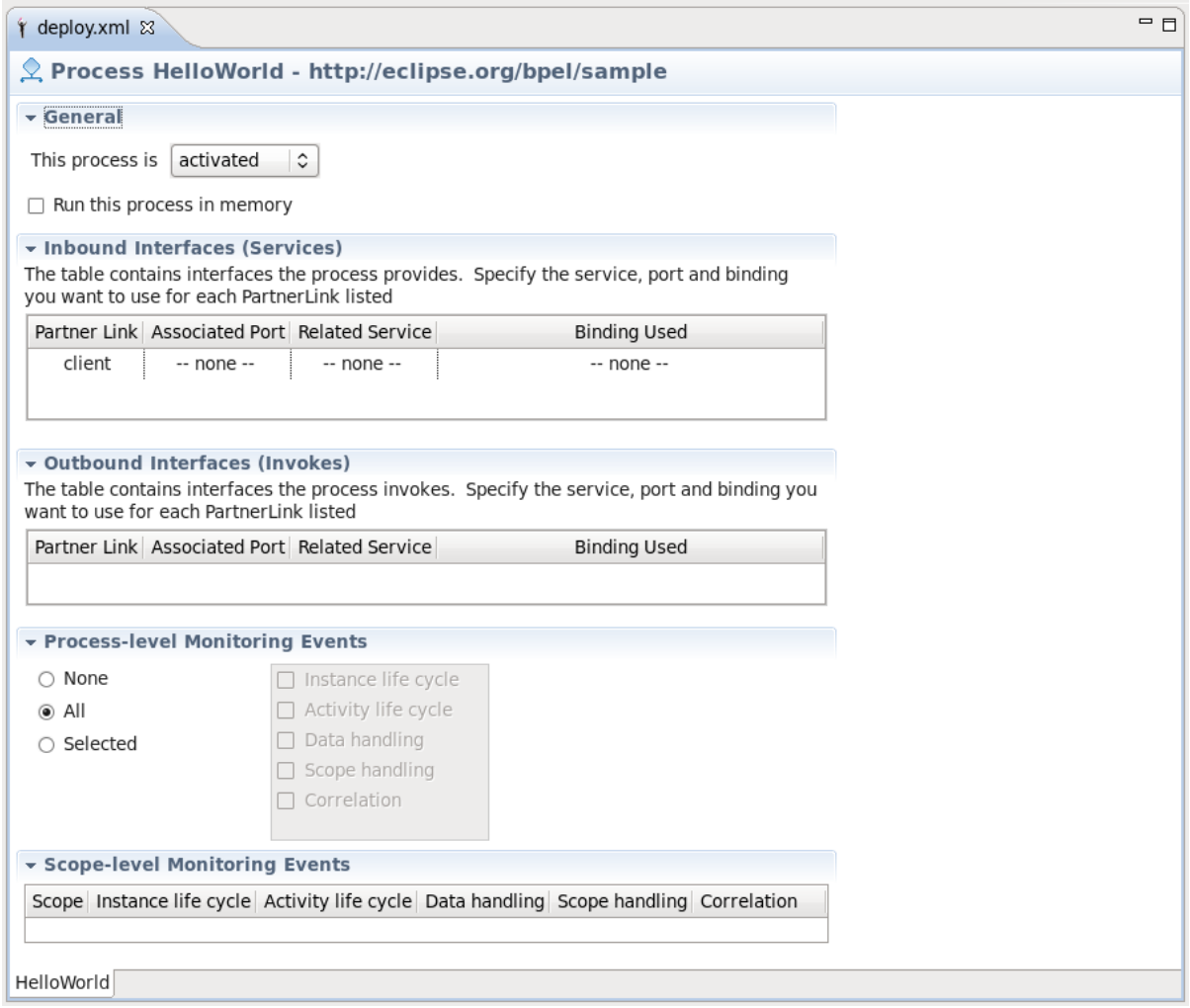
4. **Finish** ボタンをクリックしてウィザードを閉じると、新しい **deploy.xml** ファイルが作成されます。

図3.12 図 2



5. 最後に、**deploy.xml** ファイルをダブルクリックして **ODE Descriptor Deployment Editor** で開きます。**Inbound Interfaces** セクションで **Associated Port** 列をクリックし、コンボボックスで **HelloWorldProcessPort** を選択します。**Related Service** および **Binding Used** 列は自動的に入力されます。変更を **deploy.xml** ファイルに保存します。

図3.13 図 3

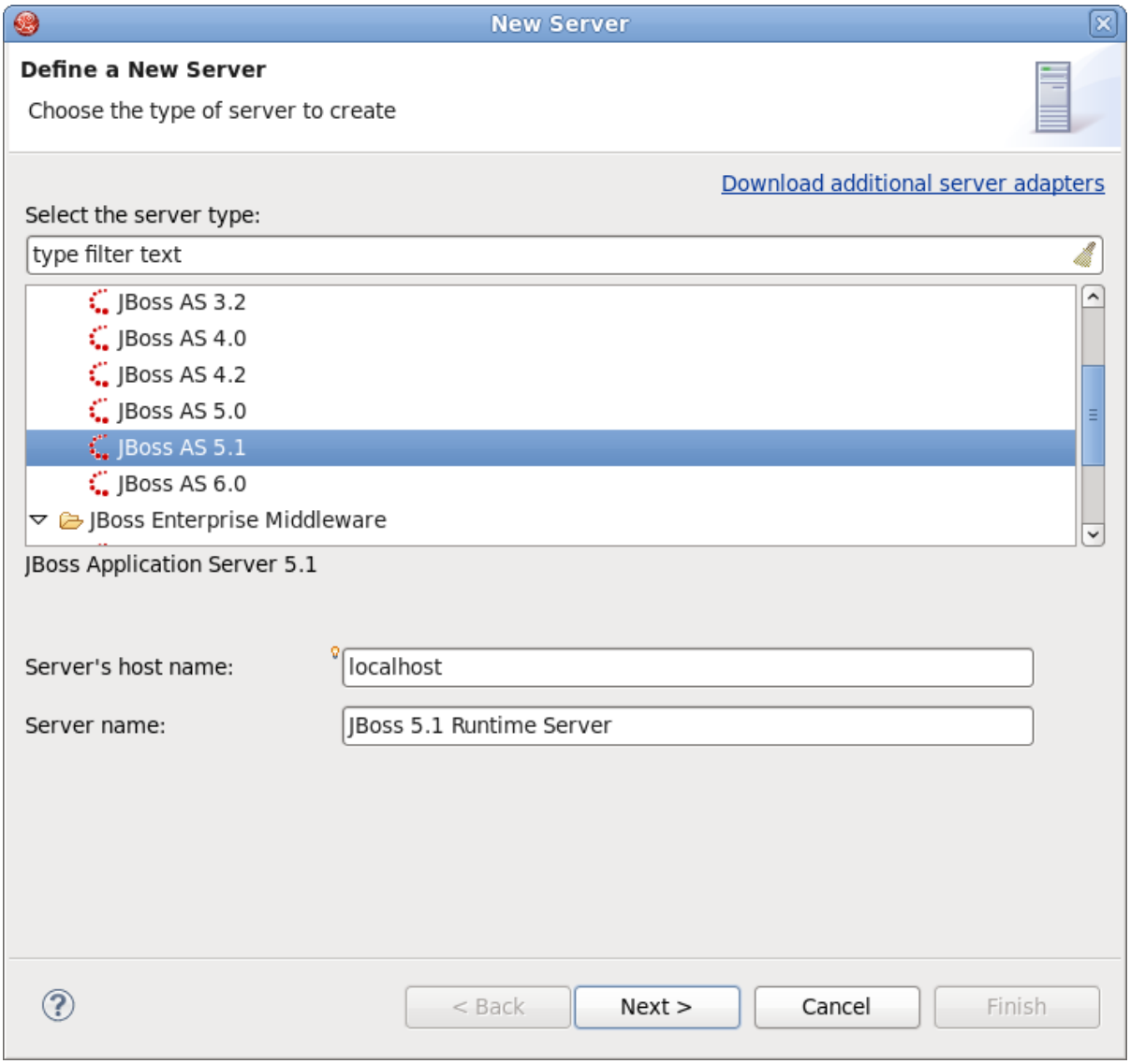


バグの報告

3.9. JBOSS BPEL SERVER の作成

1. **Windows → Show View → Other...** → **Server** → **Servers** を選択して、**Servers** ビューを開きます。
2. **Servers** ビューを右クリックし、**New** → **Server** を選択して **New Server** ウィザードを開きます。

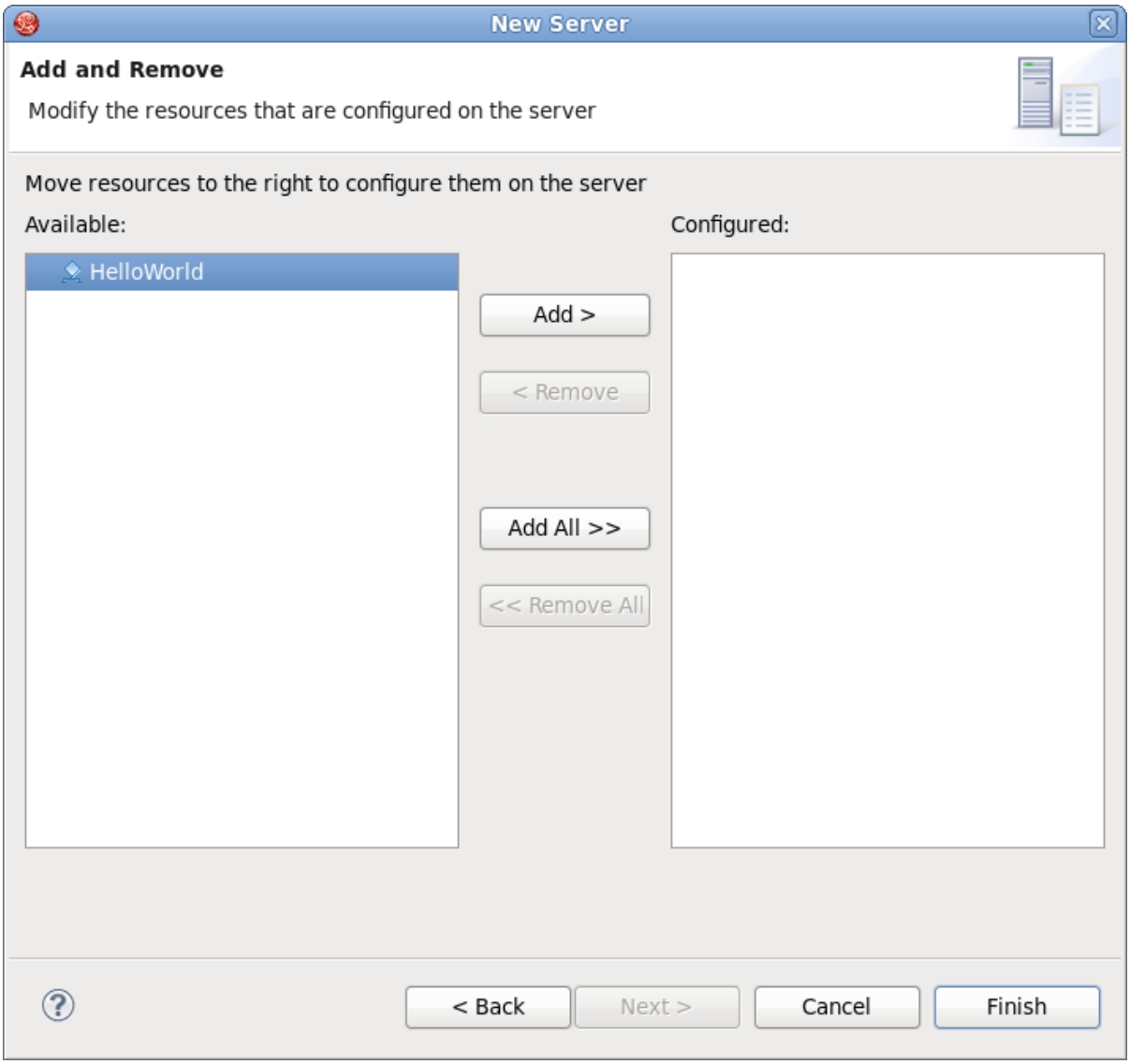
図3.14 図 1



3. サーバータイプとして **JBoss EAP 5.x** を選択します。

4. **Next** ボタンをクリックします。このページで、 JBoss EAP™ の場所を入力します。**Next** をクリックします。

図3.15 図 2



5. **HelloWorld** を選択し、**Add** ボタンをクリックしてプロジェクトをサーバーに追加します。最後に、**Finish** ボタンをクリックします。

6. サーバーを右クリックし、**Start** 項目を選択してサーバーを起動します。

図3.16 図 3

7. Web ブラウザーでリンク http://localhost:8080/bpel-console/app.html を入力して、デプロイしたプロセスにアクセスします。

バグの報告

3.10. 相関セットの作成

1. メッセージングアクティビティーの相関を作成するには、ダッシュボードタブの **Correlation Sets** に移動し、プラスボタンをクリックします。プロンプトが表示されたら、セットの名前を設定します。

2. **Properties** ビューで **Details** タブをクリックしてから **Add...** ボタンをクリックします。**Select a Property** ダイアログが表示されます。

3. 新しい WSDL プロパティーの名前とその型を入力します。(XSD 単純型または XML スキーマ要素のいずれかを入力します。)

4. **Browse** ボタンをクリックして、型を選択します。これにより、**Type Selection** ダイアログが表示されます。

5. **Aliases** セクションで **New** をクリックして、新しい WSDL プロパティーエイリアスを作成します。

6. **Message Type**、XSD **Simple Type** または XML スキーム **Element** ラジオボタンのいずれかを選択し、**Browse** をクリックしてその型を選択します。**OK** をクリックします。

7. 相関は、メッセージングアクティビティー (Invoke、Receive、Reply など) に割り当てることができます。アクティビティーを選択し、**Correlation** プロパティータブの **Add** をクリックして、適切な相関セットを選択します。

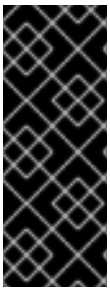
バグの報告

第4章 参照資料

4.1. ウィザード

表4.1 ウィザード

ウィザード名	Description
New BPEL Project ウィザード	JBoss Riftsaw ランタイムエンジンにデプロイできるファセットプロジェクトを作成します。**File → New → Other → BPEL 2.0 → BPEL Project** を選択すると、使用できます。**bpelContent** フォルダーには、プロジェクトに必要なすべてのファイルが含まれています。
New BPEL Process File ウィザード	ウィザードで定義された複数のテンプレートのいずれかに基づいて、BPEL プロセスを作成します。ウィザードは、**Project Explorer** または **Navigator** ビューの現在のプロジェクトに新しい BPEL プロセスを作成することを想定しています。同じ名前の BPEL プロセスがプロジェクト内にすでに存在する場合は、アクションが実行される前に警告メッセージが表示されます。
New BPEL Deployment Descriptor	このウィザードを使用して **Deployment Descriptor** ファイルを作成します。このファイルは Web サービスのマニフェストであり、BPEL プロセスをランタイムエンジンにデプロイする場合に必要です。このウィザードが完了すると、BPEL Deployment Descriptor Editor が開きます。



重要

プロセスをデプロイする場合、BPEL アーティファクトは **bpelContent** フォルダー階層内のどこかに含まれている必要があります。複雑なプロジェクトはフォルダー階層に編成できますが、これらのフォルダーは **bpelContent** 内に含まれている必要があります。

Deployment Descriptor ファイルは、**bpelContent** フォルダー内およびすべてのフォルダー階層のルートに含まれている必要があります。

バグの報告

4.2. ビュー

表4.2 ビュー

View	Description

View	Description
Outline	**Outline** ビューは、BPEL プロセスの構造レイアウトを提供します。関連するボタンを押すと、プロセスを階層ツリー構造の概要またはサムネイルビューとして表示できます。
Palette	BPEL Designer の主要な編集、作成、および表示ツールは、**Palette** からアクセスできます。**Palette** は、BPEL Designer のメインウィンドウの右端または左端にドッキングすることも、デタッチして独自のビューに表示することもできます。 ● **Selection Tool** は、エディターの描画キャンバスで個々のアクティビティーを選択するために使用されます。**CTRL** キーまたは **SHIFT** キーを押しながらマウスの左クリックを行うと、複数のアクティビティーを選択できます。**Marquee Tool** を使用すると、アクティビティーのグループのまわりを選択用の四角形でドラッグして選択することができます。 ● BPEL アクティビティーは、ラベルの付いた **Actions**、**Controls**、および **Faults** パレットセクション (または引き出し) からエディターの描画キャンバスにアイコンをドラッグすることによって作成されます。これらのセクションは、個々のパレットセクションタイトルをクリックすることで、折りたたんだり展開したりできます。別のセクションが展開された場合に折りたたまれないように **固定** することもできます。 ● **Palette** の下部にあるツールは、描画キャンバスを拡大または縮小するために使用されます。
Dashboard	このパネルは BPEL Designer キャンバスに埋め込まれており、現在選択されているアクティビティーまたは BPEL プロセスに対して定義されている BPEL 要素の概要を提供します。プロセス名はダッシュボードの上部に表示されます。メインのダッシュボードエリアには、プロセスに対して現在定義されているすべての **Partner Links**、**Variables**、**Correlation Sets**、および **Message Exchanges** が一覧表示されます。緑のプラス記号とグレーの x 記号を使用すると、これらの各要素を追加および削除できます。すべての要素名のインライン編集は、名前を選択してからもう一度クリックしてエディターを有効にすることで機能します。

バグの報告

4.3. プロパティーセクションのタブ

表4.3 プロパティーセクション

名前	Description
Description タブ	**Description** タブには、アクティビティー名が含まれています。名前は XML 要素の命名規則に従う必要があり、文字、数字、および特定の特殊文字のみに制限されます (スペースは使用できません)。
Join Behavior タブ	結合条件は、リンクのターゲットアクティビティーによって評価されます。ドロップダウン **Expression language** メニューを使用して、結合の条件を定義する XPath 式を入力します。プロセスまたはそれを含むスコープによって定義された **Suppress Join Failure** 動作は、下部のラジオボタンでオーバーライドできます。
Correlation タブ	**Correlation** タブには、現在選択されている **Receive**、**Reply**、または **Invoke** アクティビティーで使用されるすべての相関が一覧表示されます。このタブを使用して、相関をアクティビティーに追加したり、アクティビティーから削除したりできます。
Namespaces タブ	Namespaces は、インターネット上の一連のリソースを一意に識別する URI (Uniform Resource Identifier) です。接頭辞と呼ばれる省略形のエイリアスは通常、XML ファイルを読みやすくするために使用されます。**Namespaces** タブには、現在選択されているアクティビティーの範囲内にあるすべての namespace URI とその接頭辞が一覧表示されます。namespace にまだ接頭辞が割り当てられていない外部プロパティー (XSD で定義された要素) への参照を作成すると、BPEL Designer は接頭辞を作成するように要求します。これは、プロパティーの **Properties** シートの **Namespace** タブで、**Assign Prefix** ボタンをクリックして行うこともできます。
Message Exchange タブ	メッセージ交換は、 Reply アクティビティーを inbound message アクティビティーと関連付けるために使用され、 Receive, OnMessage または OnEvent のいずれかになります。これらは、2 つのパーティー間のリクエスト/レスポンスの会話に付けられる説明的な名前であり、XML 要素の命名規則に準拠する必要があります。

バグの報告

4.4. PROCESS PROPERTY SHEET タブ

表4.4 Process Property Sheet タブ

名前	Description
Description タブ	**Description** タブでは、プロセス名とその namespace URI を変更できます。
Details タブ	**Process Details** タブでは、デフォルトの **Expression** と **Query** 言語を選択できます。**Exit on Standard Fault** を **Yes** に設定すると、結合エラー以外の WS-BPEL 標準エラーが発生した場合にプロセスが終了します。現在、XPath 1.0 のみがサポートされています。
Join Behavior タブ	**Process Join Behavior** タブでは、プロセスが結合の失敗を処理する方法を決定します。**Yes** に設定すると、 **JoinFailure** フォールトは、プロセス内のすべてのアクティビティーで無視されます。アクティビティーは、この値をオーバーライドしたり、その親から値を継承したりできます。
Imports タブ	**Imports Detail** タブには、インポートされたすべてのサービスインターフェイス (WSDL) と、プロセスで使用される XML スキーマ (XSD) が一覧表示されます。このページのインポートに追加の WSDL および XSD ファイルを追加することができます。新しいリソースがインポートされた後、**Namespaces** タブから namespace URI に接頭辞を割り当てることができます。インポートされたリソースは、プロジェクトの root フォルダー (デフォルトでは **bpelContent**) またはサブフォルダーに配置する必要があります。
Namespace タブ	ここで、プロジェクトの名前を入力できます。
Documentation タブ	このタブをクリックして、プロセスに関連する関連ドキュメントを表示します。

バグの報告

4.5. DETAILS タブのオプション

表4.5 Details タブのオプション

名前	Description

名前	Description
Partner Links	Partner Links は、2 つのサービス間の会話を定義する際に役立ちます。会話で各パートナーが果たすロールと、パートナー間で交換できるメッセージのタイプを定義します。**Details** タブでは、 Partner Link の要素を選択するための **Expression language** と **Query language** を選択できます。
Variables	Variables は、ビジネスロジックによる検査と操作向けに、インバウンドおよびアウトバウンドメッセージを格納するために BPEL で使用されます。また、中間結果とプロセス状態を保存するためにも使用できます。3 種類の変数宣言は、メッセージ型、XML スキーマ型、および XML スキーマ要素になります。**Details** タブでは、変数の宣言された型とその構造を、既知の型から選択して定義できます。変数の型が定義されると、変数の構造が表示されます。ハイパーリンクをクリックすると、選択したタイプまたは要素の WSDL または XML スキーマエディターが開きます。
Empty	Empty アクティビティーは未定義の Basic Activity のプレースホルダーで、プロセスが実際に実行される前に、最終的には実際のアクティビティーに置き換えられることを意図しています。BPEL エンジンが Empty アクティビティーに直面すると、それは無視されます。**Details** タブでは、 Invoke、 Receive、 Reply、および Assign の 4 つの基本アクションのいずれかを選択できます。マウスを選択ボタンの 1 つに合わせると、そのアクティビティーの簡単な説明が表示されます。
Invoke	Invoke アクティビティーには、そのサービスの WSDL で定義されているように、 Partner Link 名と Operation が必要です。右側の **Quick Pick** ツリーコントロールを使用して、 Partner Link と Operation を選択できます。一方向の呼び出しの場合は、 Input Variable のみを指定します。リクエスト/レスポンス呼び出しの場合は、 Output Variable も指定する必要があります。**Use WSDL Message Parts Mapping** というラベルの付いたチェックボックスは、要求メッセージに変数を使用する代わりの方法を提供します。

名前	Description
Receive	Receive アクティビティーには、このサービスの WSDL で定義されているように Partner Link 名と操作が必要です。右側の **Quick Pick** ツリーコントロールを使用して、 Partner Link と Operation を選択できます。以前に定義した変数を使用してメッセージデータを保持したり、**Use WSDL Message Parts Mapping** チェックボックスを設定して、着信メッセージを匿名の WSDL メッセージ変数に格納したりできます。**Create a new Process Instance** チェックボックスを有効にすると、BPEL エンジンが新しいプロセスを開始します。これにより、クライアントとの新しい会話が開始されます。
Reply	Reply アクティビティーには、WSDL で定義されているように Partner Link 名と Operation が必要です。右側の **Quick Pick** ツリーコントロールを使用して、 Partner Link と Operation を選択できます。以前に定義した変数を使用して応答メッセージデータを提供するか、**Use WSDL Message Parts Mapping** チェックボックスを設定して、匿名の WSDL メッセージ変数からのデータを使用できます。
Opaque	Opaque アクティビティーは抽象的なプロセスでのみ使用され、まだ決定されていない他のアクティビティーのプレースホルダーとして使用されます。Opaque アクティビティーを描画キャンバスにドラッグ＆ドロップすると、プロセスは実行不可能な抽象的なプロセスに変換されます。
Assign	Assign セクションには、メッセージオプションや管理ボタンなどの変数の配列が含まれています。追加のタイプ選択またはデータ入力ウィジェットが、コンボボックスで選択されたソースおよびターゲット項目カテゴリーに応じて、**From** および **To** コンボボックスの下に表示されます。デフォルトのコンボボックスの選択が Variable であるため、これらは最初はプロセス変数の選択の制御になります。
Validate	Validate 詳細タブには、検証する変数のリストが含まれています。
While および RepeatUntil	これらのアクティビティーには同じ詳細タブがあり、条件付きアクティビティーに対して評価される XPath 式を指定できます。
Link	Link 詳細タブでは、 Flow 同期が満たされ、ターゲットアクティビティーを続行できるようにする条件を指定することができます。これは、他の条件付きアクティビティーの詳細タブに似ています。

名前	Description
Pick	Pick タブでは、イベントが新しいプロセスインスタンスを作成するかどうかを指定できます。
OnMessage	OnMessage アクティビティーは Pickおよびイベントハンドラーで使用されます。**Details** タブでは、アクティビティーによって期待される Partner Link、Operation および Message Type、そして受信したメッセージデータを含むプロセス変数を指定できます。
OnAlarm	OnAlarm アクティビティーは、メッセージの到着を待っている間にタイムアウトを処理する Pick またはイベントハンドラーで使用されます。このアクティビティーは、特定の期間または特定の日時まで待機するように設定できます。**Details** タブでは、アクティビティーによって期待される Partner Link、Operation および Message Type、そして受信したメッセージデータを含むプロセス変数を指定できます。**Repeat** 条件は、イベントハンドラーの OnAlarm にのみ許可されます。これにより、アクティビティーに含まれるアクティビティーを繰り返し実行できます。**Repeat** 期間は、各繰り返しの前にプロセスが待機する時間です。
ForEach	ForEach は、ループの反復を追跡するために使用するカウンター変数を指定できます。**Parallel execution** チェックボックスは、すべての反復を並行して実行します。**Counter Values** タブでは、開始カウンター値と終了カウンター値を指定します。オプションの **Completion** タブでは、ループの早期終了条件を指定できます。
Wait	Wait アクティビティーの詳細タブを使用すると、遅延 (Duration) を設定したり、日付と時刻を指定してプロセスの実行を続行したりできます。
Scope	Scope アクティビティーの詳細タブを使用すると、スコープが **isolated** かどうかを定義できます。
Throw	Throw アクティビティーは、囲んでいる Scope アクティビティーのフォールトハンドラーを呼び出します。 Throw には、標準の BPEL フォールトの名前か、ユーザー定義のフォールトメッセージの名前が必要です。フォールトデータの値を保持するために変数が使用されます。
CompensateScope	CompensateScope アクティビティーは、**Target Activity** の名前で指定された Scope または Invoke アクティビティーのコンペンセーションハンドラーを呼び出します。

バグの報告

4.6. BPEL DESIGNER の機能

表4.6 BPEL Designer の機能

名前	Description
Drawing Canvas	BPEL プロセスのグラフィカルな表現が含まれており、エディターウィンドウの下部にある **Design** タブが選択されている場合に表示されます。アクティビティー名のいずれかをクリックすると、インラインエディターがアクティブになり、プロセス名を編集できます。編集を終了するには、**ENTER** キーを押すか、別のウィンドウコントロールをクリックしてフォーカスを変更します。
Source	このタブには、プロセスの XML (テキスト) 表現が表示されます。一方のビューで行われた変更は、他方のビューに表示されます。アクティビティーのデフォルトのレイアウトは上から下ですが、コンテキストメニューから水平レイアウトに変更できます。
Palette	BPEL Designer の主要な編集、作成、および表示ツールには、このツールからアクセスします。
Dashboard	BPEL プロセスの概要を示します。
Property Sheet	描画キャンバスでアクティビティーを選択すると、アクティビティーのプロパティーが表示されます。
Outline	このパネルは、BPEL プロセスの構造ビューを提供します。

バグの報告

4.7. BPEL DESIGNER の概念

表4.7 BPEL Designer の概念

名前	Description
Assign エラー	このアイコンの上にマウスを置くと、エラーメッセージが表示されます。

名前	Description
Basic アクティビティー	Basic アクティビティーは、アイコンとアクティビティーのユーザー定義名を含む角の丸い四角形として描画キャンバスに表示されます。**Palette** の **Actions** セクションには、すべての基本的なアクティビティーが含まれています。たとえば、 Assign, Invoke や Receive などがあります。
Start および End	すべてのプロセスには、プロセスフローの開始と終了を視覚化するためのプレースホルダーとして機能する Start アクティビティーと End アクティビティーがあります。
Assign アクティビティー	Assign アクティビティーを使用すると、プロセスで定義されている変数とメッセージの内容を操作できます。
Invoke	Invoke アクティビティーは、外部サービスにメッセージを送信し (一方向の呼び出し)、オプションで応答 (リクエストとレスポンス) を待つために使用されます。 Invoke は、例外条件を処理するためのコンペンセーションハンドラーとフォールトハンドラーを定義することもできます。
Receive	Receive アクティビティーは、サービスクライアントからの特定のメッセージタイプを待ちます。
Reply	Reply アクティビティーは、特定のメッセージタイプまたはフォールトメッセージでクライアントに応答するために使用されます。
Validate	Validateアクティビティーは、XML スキーマおよび WSDL データ定義に対して変数の値を検証するために使用されます。これには、変数のデータ型と構造が含まれます。検証が失敗すると、BPEL 標準フォールトの invalidVariables が出力されます。検証は通常、パートナーまたはクライアントにメッセージを送信する直前、またはメッセージを受信した後に実行され、メッセージに必要なデータがすべて含まれていることを確認します。
Wait	Wait アクティビティーは、特定の時間、または特定の日時までプロセスの実行を遅らせます。これは通常、特定の時間に操作を呼び出すために使用されます。たとえば、プロセスの状態を毎時または毎日更新したり、特定の時間に別のサービスから情報を収集したりします。

バグの報告

4.8. STRUCTURED アクティビティー

Structured アクティビティーは、1つ以上のアクティビティーを保持できるコンテナーです。**Palette** コンテナーの **Controls** セクションには、 **structured activities** がすべて含まれています。これらのいずれかを描画キャンバスにドラッグアンドドロップすると、BPEL Designer はアクティビティーの基本的なスケルトンを作成し、デフォルトのプロパティーを割り当てます。

structured アクティビティーはすべて、有効と見なされる前に追加の設定が必要です。たとえば、BPEL では空の Sequence アクティビティーを許可しません。無効な structured アクティビティーには、basic アクティビティーと同様のエラーアイコンが表示されます。

図の下部にあるプラスボタンとマイナスボタンをクリックすると、描画キャンバス上で structured アクティビティーを展開したり折りたたんだりできます。下記のアクティビティー一覧をご覧ください。

表4.8 Structured アクティビティー

名前	Description
If、ElseIf、Else	If アクティビティーを使用すると、アクティビティーの1つ以上のシーケンスを条件付きで実行できます。これは、 If およびオプションの ElseIf 要素によって定義される1つまたは複数の条件分岐のシーケンスで構成されます。要素は左から右の順序で評価されます (水平レイアウトを選択した場合は上から下)。オプションの Else 分岐は、他の条件がいずれも true でない場合に実行されます。 If アクティビティーは、条件 (XPath として表現) と、条件が true と評価された場合に実行されるアクティビティーを定義する必要があります。追加の ElseIf と Else 要素を挿入するには、 If 図を右クリックし、コンテキストメニューから目的の要素を選択します。上の図は、オプションの ElseIf および Else 要素を持つ完全な If アクティビティーを表示しています。
Pick	Pick アクティビティーにより、プロセスは任意の数のメッセージのいずれかが受信されるまで待機します。オプションのタイマーを設定して、これらのメッセージの受信を待つ時間を制限できます。メッセージの受信とタイマーの期限切れを処理するアクティビティーは、 Pick で定義されています。メッセージの受信は OnMessage アクティビティーによって処理され、タイマーの期限切れは OnAlarm アクティビティーによって処理されます。
While	While アクティビティーは、各反復の開始時に条件が true と評価される限り、含まれているアクティビティーを繰り返し実行します。 While アクティビティーは条件を定義する必要があり、アクティビティーが含まれている必要があります。

名前	Description
ForEach	ForEach は、 Scope に含まれるアクティビティーを指定された回数実行するループアクティビティーです。 ForEach プロパティーの詳細タブで定義されたカウンター変数は、反復を追跡するために使用されます。 ForEach プロパティーは、このカウンター変数の開始値式と終了値式で設定する必要があります。カウンターは、最初は開始値に設定され、 Scope のアクティビティーはカウンターが終了値を超えるまで実行されます。 このアクティビティーは、すべての反復を並行して実行するように設定することもできます。つまり、囲まれている Scope アクティビティーは、複数の Scopes が Flow アクティビティーに囲まれているかのような動作をします。 オプションの早期終了値を定義できます。これにより、カウンターが終了値に達する前にループが完了します。 ForEach は、順次実行の場合、カウンターがこの早期終了値と等しくなると完了します。並列実行の場合、早期終了値は完了した反復の数になります。たとえば、 ForEach は、**一部のアクション** の少なくとも **一部の数** が終了したときに完了します。
RepeatUntil	RepeatUntil アクティビティーは、各反復の最後に条件が true と評価される限り、含まれているアクティビティーを繰り返し実行します。 RepeatUntil に対して条件を定義する必要があります。これには、アクティビティーが含まれている必要があります。
Sequence	Sequence は、1 つ以上の他のアクティビティーのコンテナーです。それらは順番に実行され、(Scope と Flow アクティビティーとは異なり)、その他の特別な特徴はありません。条件付きアクティビティー (If、While、 RepeatUntil と ForEach) は、実行のターゲットとして 1 つのアクティビティーのみを持つことができます。 Sequence は通常、複数のアクティビティーを実行するために使用されます。BPEL Designer は、2 番目のアクティビティーを条件付きアクティビティーのいずれかにドラッグアンドドロップした場合、自動的に Sequence を作成します。

名前	Description
Scope	Scope は、囲まれたアクティビティーのコンテキストを提供します。これは、区分化されたサブプロセスと考えることができます。 Scope が **分離** されていると宣言されている場合、プロセスと共有されている変数とパートナーリンクはロックされ、 Scope が実行されている間に、他の同時実行されている Scopes がアラートを出すことを阻止します。 Scope は任意の深さにネストすることもでき、 Scope で定義されたすべての変数、パートナーリンクなどは、その子に継承されます。 有効であるためには、 Scope にアクティビティーが 1 つ必要です。通常、サービスを呼び出し、応答メッセージまたはタイムアウトを待つために使用されます。
Flow	Flow アクティビティーを使用すると、複数のアクティビティーを並行して実行できます。すべてのアクティビティーまたは Flow に含まれる Sequences は、BPEL エンジンによって同時に実行されます。 Flow は、囲まれているすべてのアクティビティーが完了すると完了します。
Link	Links は同期に使用されます。 Link を作成するには、完了したアクティビティーを右クリックし、**Add Link** を選択します。次に、マウスをこれ (**target**) に依存する Flow のアクティビティーに移動し、マウスの左ボタンをクリックしてリンクを作成します。 Link は、 Flow 内で一意でなければならない名前によって識別されます。BPEL Designer はデフォルト名を生成しますが、これはプロパティーで変更できます。アクティビティーが完了したと見なすための条件を定義する Link にテストを追加することもできます。
Faults	フォールトアクティビティーにより、通常のプロセス実行フローが特殊なハンドラーにジャンプします。これは、最新のプログラミング言語の例外と類似します。 Exit フォールトにより、プロセスはただちに終了します。 Throw フォールトは、指定されたフォールトをその先祖 Scope またはプロセス自体に伝播します。 Rethrow フォールトは、フォールトハンドラー内で使用され、元のフォールトデータを使用してフォールトを伝播します。 Compensate フォールトは、コンペンセーションハンドラーを呼び出すために使用されます。最後に、 CompensateScope フォールトは、囲んでいる Scope のコンペンセーションハンドラーを呼び出すために使用されます。

バグの報告

4.9. フォールト、コンペンセーション、終了、およびイベントハンドラー

表4.9 フォールト、コンペンセーション、終了、およびイベントハンドラー

名前	Description
フォールトハンドラー	アクティビティーによってエラーが出力されたときに実行されます。
コンペンセーションハンドラー	BPEL プロセスが Compensate または CompensateScope アクティビティーに直面すると実行されます。
終了ハンドラー	Scope が早期終了を余儀なくされる場合に実行されます。
イベントハンドラー	メッセージの受信やタイマーの期限切れなどのイベントに対して実行されます。
スコープレベルのハンドラー	スコープには任意のハンドラーを含めることができます。スコープはネストできるため、各レベルで独自のハンドラーセットを定義できます。内部スコープのハンドラーによってキャッチおよび処理されないイベントは、その先祖に伝播されます。
アクティビティーレベルのハンドラー	Invoke アクティビティーのみがハンドラーを定義できます。使用可能なハンドラーは、フォルトハンドラーとコンペンセーションハンドラーです。

バグの報告

4.10. BPEL DEPLOYMENT DESCRIPTOR EDITOR のプロパティー

表4.10 BPEL Deployment Descriptor Editor のプロパティー

名前	Description
プロセス選択タブ	これらのタブをクリックして、各プロセスの設定ページを表示します。
初期のプロセス状態	プロセスは、 **active**、 **inactive**、または **retired** のいずれかの状態でデプロイできます。
インバウンドインターフェイスの選択	クライアントがこのサービスを呼び出すために使用する WSDL ポートタイプを選択します。
出力インターフェイスの選択:	呼び出されたサービス (存在する場合) ごとに、そのポートタイプを選択する必要があります。

名前	Description
スコープレベルの監視イベント	BPEL エンジンは、プロセスで定義されたスコープごとに監視イベントを生成するように設定できます。

注記

BPEL プロジェクトをランタイムエンジンにデプロイする前に、**デプロイメント記述子**と呼ばれるものを作成する必要があります。これは単なるマニフェストファイルであり、XML としてシリアル化され、すべての BPEL プロセスとその BPEL エンジンへのインターフェイスを記述します。**デプロイメント記述子** ファイルは、プロジェクトのルートフォルダーに作成する必要があります。

バグの報告

4.11. ダイアログ

表4.11 ダイアログ

名前	Description
XPath 式エディター (埋め込みコントロール)	XPath 式エディターは、ポップアップ提案の形で状況依存の支援を提供します。電球アイコンは、**CTRL** キーと **SPACE** キーを同時に押すと、コンテンツアシストが利用できることを示しています。BPEL 2.0 仕様では、プロセスレベルおよびアクティビティーレベル (XPath を使用するアクティビティー) での XPath 言語バージョンの定義が規定されています。ただし、BPEL Designer および JBoss Riftsaw ランタイムエンジンでサポートされるのは XPath 1.0 のみです。
クイックピック (埋め込みコントロール)	ツリーコントロールは、メッセージ部分、パートナーリンク、および操作を選択するために、多くのプロパティーページで使用されます。

名前	Description
タイプの選択	このダイアログは、BPEL Designer がメッセージ、メッセージ部分、XML スキーマタイプ、または XML 要素の選択を要求するたびに表示されます。 1. **Type Name**: **Matches** (4) リストに表示される項目を制限するために使用されます。このフィルターのテキストで始まる項目のみが表示されます。 2. **Show XSD Types** : エディターが XSD ファイルを検索する場所を制限するために使用できます。 3. **Filter**: タイプに応じて一致の数をさらに減らします。 4. **Matches**: 選択したフィルターに一致する項目を表示します。このリストの項目を選択すると、**Type Structure** (5) ツリービューが更新されます。 5. **Type Structure**: **Matches** (4) リストで選択した項目の構造を表示します。要求された項目のタイプによっては、このツリーコントロールからも項目を選択する必要がある場合があります。**OK** ボタンが有効になっていることは、ここで選択が必要であることを示しています。 6. **Add Schema**: 必要な XML スキーマが解決されていない場合は、このボタンをクリックしてプロセスのインポートに追加できます。

バグの報告

4.12. 型の選択

表4.12 型の選択

名前	Description
Type Name	**Matches** (4) リストに表示される項目を制限するために使用されます。このフィルターのテキストで始まる項目のみが表示されます。
Show XSD Types	エディターが XSD ファイルを検索する場所を制限するために使用できます。
Filter	型に応じて、一致する数をさらに減らします。

名前	Description
Matches	選択したフィルターに一致する項目を表示します。このリストの項目を選択すると、**Type Structure** (5) ツリービューが更新されます。
Type Structure	**Matches** リストで選択した項目の構造を表示します。このツリーコントロールから項目を選択する必要がある場合があります。必要な場合は、**OK** ボタンが有効になります。
Add Schema	必要な XML スキーマが解決されていない場合は、このボタンをクリックしてプロセスのインポートに追加できます。

バグの報告

4.13. WSDL プロパティーの選択

このダイアログでは、既存の WSDL プロパティーを選択するか、新しいプロパティーを作成することができます。

表4.13 WSDL プロパティーの選択

名前	Description
Property Name	**Matches** リストに表示される項目を制限するために使用されます。このフィルターのテキストで始まる項目のみが表示されます。
New	このボタンをクリックして、新しい WSDL プロパティーを作成します。
Matches	**Property Name** に一致する項目、またはフィルターが空白の場合はすべての項目を表示します。
Property Type	**Matches** リストで選択された項目の型を表示します。

バグの報告

4.14. WSDL プロパティーの作成

このダイアログは、新しい WSDL プロパティーを作成するために使用され、 Select WSDL property ダイアログで **New** ボタンをクリックすると表示されます。

表4.14 WSDL プロパティーの作成

名前	Description
Name	新しいプロパティーの名前を入力します。
Defined As	プロパティータイプの定義方法を選択します。
Browse	このボタンをクリックして、プロパティータイプを選択します。
New	このボタンをクリックして、新しいプロパティーエイリアスを作成します。
Aliases	このリストには、プロパティーに定義されているすべてのプロパティーエイリアスが表示されます。

バグの報告

4.15. WSDL プロパティーエイリアスの作成

このダイアログでは、選択したプロパティーの WSDL プロパティーエイリアスを作成できます。

表4.15 WSDL プロパティーエイリアスの作成

名前	Description
Defined As	プロパティーエイリアスタイプの定義方法を選択します。 **
Browse	このボタンをクリックして、プロパティーエイリアスタイプを選択します。
Query	このエディターを使用すると、XPath 式エディターを使用してプロパティーエイリアスのクエリーを定義できます。

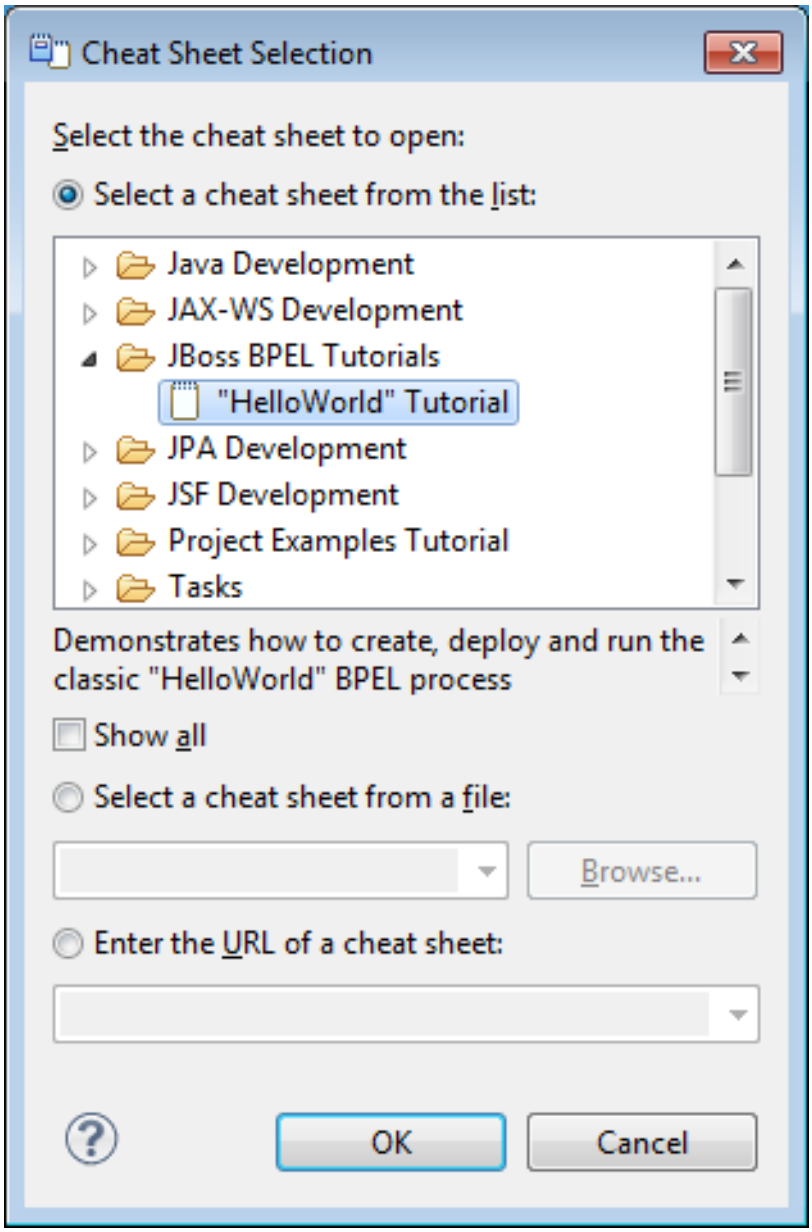
バグの報告

4.16. チートシート

手順4.1 タスク

1. **Help → Cheat Sheets** をクリックしてチートシートにアクセスします。
2. 以下に示すように、チートシートが別のビューで開きます。**Click to begin** リンクをクリックして開始します。

図4.1 図 1



3. プラグインツールのチュートリアルを表示できるようになりました。

バグの報告

4.17. コンテキストメニュー

描画キャンバス上のアクティビティー図を右クリックして、コンテキストメニューにアクセスします。サブメニューが表示されます。**Add** サブメニュー内の項目は現在のアクティビティーの後に新しいアクティビティーを追加し、**Insert Before** サブメニュー内の項目は現在のアクティビティーの前に新しいアクティビティーを挿入します。

バグの報告

付録A 更新履歴

改訂 5.3.1-31.400	**2013-10-31**	**Rüdiger Landmann**
Rebuild with publican 4.0.0		
改訂 5.3.1-31	**Wed Feb 20 2013**	**CS Builder Robot**
コンテンツ仕様からビルド: 7172、リビジョン: 375275		