Red Hat JBoss BPM Suite 6.0

スタートガイド

Red Hat JBoss BPMS のスタートガイド

Last Updated: 2017-11-23

Red Hat JBoss BPM Suite 6.0 スタートガイド

Red Hat JBoss BPMS のスタートガイド

Kanchan Desai
kadesai@redhat.com

Doug Hoffman

Eva Kopalova

Red Hat Content Services

法律上の通知

概要

本書は、初めて Red Hat JBoss BPMS をインストールおよび設定するユーザー向けのガイドです。

目次

第1章 はじめに

1.1. RED HAT JBOSS BPM SUITE について

Red Hat JBoss BPM Suite は、ビジネスプロセス管理とビジネスルール管理を組み合わせるオープンソースのビジネスプロセス管理スイートで、ビジネスおよび IT ユーザーによる、ビジネスプロセスとルールの作成、管理、検証、およびデプロイメントを実現します。

Red Hat JBoss BRMS および Red Hat JBoss BPM Suite は、すべてのリソースが保存される集中リポジトリーを使用します。これにより、ビジネス全体で一貫性や透明性を維持し、監査を行えます。ビジネスユーザーは、IT 担当者のサポートを受けなくてもビジネスロジックおよびビジネスプロセスを編集できます。

ビジネスルールコンポーネントに対応するため、Red Hat JBoss BPM Suite には統合された Red Hat JBoss BRMS が含まれています。

Business Resource Planner は、本リリースの技術プレビューとして含まれています。

バグを報告する

1.2. RED HAT JBOSS BPM SUITE のコンポーネント

Red Hat JBoss BPM Suite には以下のコンポーネントが含まれています。

実行エンジン

ビジネス資産 (プロセス、タスク、ルールなど) のランタイム環境。詳細は、『Red Hat JBoss BPM Suite 管理および設定ガイド』を参照してください。

アーティファクトリポジトリー (ナレッジストア)

改訂管理機能を提供するビジネス資産のストレージ (ビジネス資産で GIT リポジトリーへ接続)。

Business Central

資産の作成、管理、およびビジネス資産の監視向けの Web ベースのアプリケーションです。ルールおよびプロセスオーサリングツール、アーティファクトレポジトリー用のビジネス資産管理ツール、ランタイムデータ管理ツール、リソースエディター、BAM (ビジネスアクティビティ監視) ツール、タスク管理ツール、BRMS ツールなど、ツールとの統合環境を提供します。詳細は、『Red Hat JBoss BPM Suite ユーザーガイド』を参照してください。

バグを報告する

1.3. ユースケース: 銀行ローンのプロセスベースソリューション

Red Hat JBoss BPM Suite (BPMS) をデプロイして、銀行のローン承認プロセスなどのビジネスプロセスを自動化できます。これは、BPM を企業全体に採用するための最初のステップとなる、典型的な「特定プロセスベース」のデプロイメントです。この場合、BPM と BPMS のビジネスルール機能の両方を利用します。

銀行は、期間や資格要件が異なる複数のローン商品を提供します。ローンを必要とする顧客は、銀行へローンの申込書を提出します。申し込み後、銀行は複数の手順で申し込みを処理し、資格の検証、期間の判断、不正行為の確認、および適切なローン商品の決定を行います。ローンの承認後、銀行は申込者のローン口座を作成し、入金を行います。その後、申込者はその資金を使用できます。銀行は、プロセ

スの各手順が対象となるすべての銀行規則に適合しているか確認し、ローンのポートフォリオを管理して利益を最大化する必要があります。各手順でデシジョンメーキング (意思決定) を手助けするためにポリシーが使用され、これらのポリシーは銀行の業績を最大化するため管理されます。

銀行のビジネスアナリストは、BPM Suite の BPMN2 オーサリングツール (プロセスデザイナー) を使用して、ローン申し込みプロセスをモデル化します。

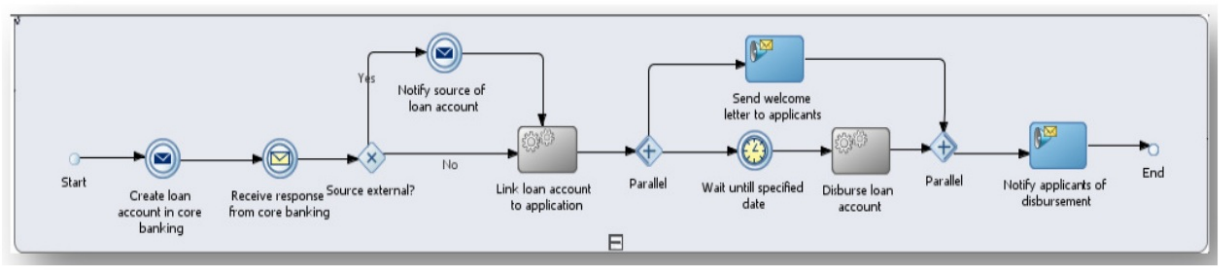


図1.1 ハイレベルのローン申し込みプロセスフロー

ポリシーを実行し、デシジョンメーキングを行うため、ビジネスルールは BPM Suite のルールオーサリングツールで開発されます。ルールはプロセスモデルへリンクされ、プロセスの各手順で正しいポリシーが実行されます。

銀行の IT 部門が BPM Suite をデプロイし、ローン申し込みのプロセス全体が自動化されるようにします。

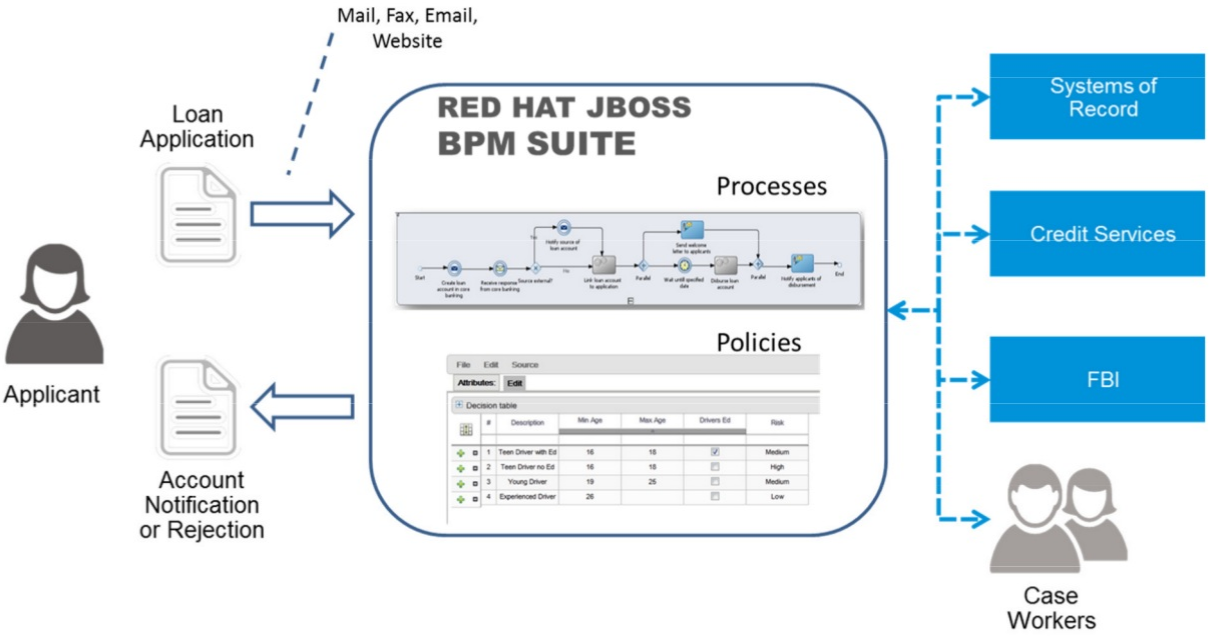


図1.2 ローン申し込みプロセスの自動化

銀行のビジネスアナリストは、ローンプロセスおよびルールのすべてをいつでも編集できます。銀行は、変化する規則に常時準拠できます。また、競争優位を保ち、収益を向上させるために、ローンの新商品をすばやく導入し、ローンのポリシーを改善できます。

バグを報告する

第2章 RED HAT JBOSS BPM SUITE のクイックスタート

本項では、最小手順で Red Hat JBoss BPM Suite をダウンロード、インストールおよび実行します。詳細な手順や、他のインストール方法については、本項以外の内容を参照してください。

この手順では、ご使用のシステムに最小限サポートされる Java バージョンがインストールされていることを前提とします。Red Hat JBoss EAP サーバーが検出されない場合はインストーラーがインストールするため、既存の Red Hat JBoss EAP サーバーは必須ではありません。

手順**2.1 BPM Suite** のクイックスタート

1. access.redhat.com より BPM Suite インストーラーをダウンロードします。
2. 次のコマンドを実行して、インストーラーを起動します。

   ```
   java -jar jboss-bpms-installer-VERSION.GA-redhat-MINOR.jar
   ```

3. GUI インストーラーの指示に従って進み、インストール後に BPM Suite アプリケーションにアクセスするために必要なユーザー名とパスワードを書き留めておきます。
4. インストールが終了したら、コマンドプロンプトを用いて BPM Suite がインストールされた EAP ホームフォルダーに移動します。
5. 以下のコマンドを実行して、BPM Suite サーバーを起動します。

   ```
   bin/standalone.sh
   ```

6. Web ブラウザーを開き、**`http://localhost:8080/business-central/`** にアクセスして Business Central にログインします。インストール時に作成した、BPM Suite アプリケーション用のユーザー名とパスワードの組み合わせを使用してログインします。

これで BPM Suite が正しくインストールされ、実行されます。以下が可能になります。

ロールに関する知識を深める:「ロールの定義」
追加ユーザーの作成:「ユーザーの作成」
Hello World プロジェクトの作成: 5章*Hello World プロジェクト*
BPM Example App Guide の対処:『Working with the BPM Suite Example App』
インストーラーを用いた詳細なインストール手順の確認:『インストールガイド』

バグを報告する

第3章 インストールオプション

Red Hat JBoss BPM Suite には、以下の 2 つのバージョンがあります。

- Red Hat JBoss Enterprise Application Platform (EAP) 6.1.1 にインストールするための実行可能 jar インストーラー。
- 以下の 2 つのバージョンがある zip ファイルインストール。
  - **`jboss-bpms-6.`*`MINOR_VERSION`*`-redhat-`*`x`*`-deployable-eap6.`*`x`*`.zip`**: Red Hat JBoss Enterprise Application Platform (EAP 6.1.1) でのデプロイメント用のバージョン。
  - **`jboss-bpms-6.`*`MINOR_VERSION`*`-redhat-`*`x`*`-deployable-generic.zip`**: Red Hat JBoss Web Server (WS)、Apache Tomcat 6 、および Apache Tomcat 7 上のデプロイメントに適した追加ライブラリーが含まれるデプロイ可能バージョン。

ご使用の環境に応じて、プロジェクトに最も適したインストールオプションを選択してください。

バグを報告する

3.1. EAP6 バンドルのインストール

本項では、Red Hat JBoss Enterprise Application Platform (EAP) にデプロイ可能な Red Hat JBoss BPM Suite パッケージのインストールについて説明します。

注記

Red Hat JBoss BPM Suite のインストールでは Red Hat JBoss EAP 6.1.1 以上がサポート対象となり、6.1.0 以下はサポート対象外になります。

バグを報告する

3.1.1. EAP6 パッケージのダウンロード

JBoss Enterprise Application Platform 向けのデプロイ可能な Red Hat JBoss BPM Suite パッケージをダウンロードするには、以下の手順に従います。

1. Red Hat Customer Portal にアクセスし、ログインします。
2. **Downloads** → **Red Hat JBoss Middleware** → **Download Software** の順に選択します。
3. **`Product`** ドロップダウンメニューより **`BPM Suite`** を選択します。
4. **`Version`** ドロップダウンメニューより製品のバージョンを選択します。
5. 表の **`Red Hat JBoss BPM Suite 6.0.2 Deployable for EAP 6.1.1`** の行に移動し、**`Download`** をクリックします。

バグを報告する

3.1.2. EAP6 パッケージのインストール

未設定の EAP 向けのデプロイ可能なパッケージをインストールするには、以下の手順に従います。

1. Red Hat カスタマーポータル からダウンロードした、EAP にデプロイ可能な zip パッケージを展開します。
2. EAP にデプロイ可能な zip パッケージを展開した後、EAP の *SERVER_HOME* ディレクトリーへマージします。

警告

この手順は、EAP のインストールに使用した同じユーザーアカウントで実行する必要があります。このアカウントは、スーパーユーザーアカウント以外でなければなりません。

3. この処理では、EAP の *SERVER_HOME* ディレクトリーのファイルが複数上書きされますが、パッケージの展開処理によってこれらのファイルが上書きされる必要があります。上書きされるファイルの 1 つが ***SERVER_HOME*/bin/product.conf** ファイルです。マージに成功した後、このファイルに **slot=bpms** という文字列が含まれる必要があります。このファイルを開くと、ファイルが正常に上書きされたことを確認できます。

Red Hat Enterprise Linux では、zip ファイルをダウンロードしたディレクトリーで次のコマンドを実行すると、BPMS zip ファイルの展開とサーバーディレクトリーへのマージを一度に行えます。

```
unzip -u jboss-bpms-VERSION-TYPE.zip -d SERVER_HOME_PARENT_DIR
```

例**3.1 unzip** コマンド

```
unzip -u jboss-bpms-6.0.2-redhat-7-deployable-eap6.x.zip -d
/home/john/myServers/
```

サーバー起動時に、Red Hat JBoss BPM Suite がデプロイされます。

設定済みの EAP 向けのデプロイ可能なパッケージをインストールするには、以下の手順に従います。

1. Red Hat カスタマーポータル からダウンロードした、EAP にデプロイ可能な zip パッケージを展開します。
2. ダウンロードした zip アーカイブを展開しますが、ファイルを上書きしないでください。以下のファイルを手作業で *SERVER_HOME* ディレクトリーにマージします。
   - **jboss-eap-6.1/domain/configuration/*** - (BPMS には JMS が必要なため、BPMS ディストリビューションによって提供される **domain.xml** にあるすべてのプロファイルに JMS がデフォルトで追加されることに注意してください)
   - **jboss-eap-6.1/standalone/configuration/*** - (BPMS には JMS が必要なため、BPMS ディストリビューションによって提供されるすべてのプロファイル設定ファイル (特に **standalone.xml** および **standalone-ha.xml** 内) にデフォルトで JSM が追加されることに注意してください)
   - **jboss-eap-6.1/modules/layers.conf**

- **jboss-eap-6.1/bin/product.conf**

3. 目的の EAP のデプロイメントに、競合する名前が含まれないようにしてください。BPMS ディストリビューションから、フォルダー **jboss-eap-6.1/standalone/deployments** を *EAP_HOME* ディレクトリーにコピーします。

4. BPMS という名前の EAP モジュールレイヤーが存在しないことを確認し、フォルダー **jboss-eap-6.1/modules/system/layers/bpms** を EAP 6.1.1 フォルダーにコピーします。

バグを報告する

3.1.3. ロールの定義

サーバーを起動して Business Central へログインする前に、ユーザーアカウントを作成する必要があります。本項では、Red Hat JBoss BPM Suite で使用されるユーザーロールについて説明します。

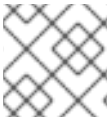
- admin: admin ロールを持つユーザーは、アプリケーションの管理者です。管理者は、ユーザーやリポジトリー (作成およびクローン) を管理でき、アプリケーションに必要な変更を加えるため完全アクセスを持ちます。管理者はシステム内のすべてのエリアにアクセスできます。

- developer: 開発者はほとんどの機能にアクセスでき、ルール、モデル、プロセスフロー、フォーム、およびダッシュボードを管理できます。資産リポジトリーを管理でき、プロジェクトの作成、構築、およびデプロイが可能です。また、Red Hat JBoss Developer Studio を使用してプロセスを表示できます。developer ロールは、新規リポジトリーの作成やクローンなど、管理機能の一部のみを使用できません。

- analyst: analyst ロールは、プロジェクトをモデル化および実行するため、すべての高レベル機能へアクセスできます。しかし、analyst ロールを持つユーザーは **Authoring** → **Administration** へアクセスできません。また、**Deployment** → **Artifact Repository** など、開発者を対象とする低レベル機能の一部にもアクセスできません。しかし、Project Editor を使用すると、**Build & Deploy** ボタンを使用できます。

- user: ユーザーまたはビジネスユーザーは、特定プロセスを操作するために使用されるビジネスタスクリストの作業を行います。このロールを持つユーザーは、ダッシュボードへのアクセスやプロセスの管理が可能です。

- manager: マネージャーはシステムを確認します。ビジネスプロセスに関する統計、ビジネスプロセスのパフォーマンス、ビジネスインディケーター、およびシステムのその他のレポート内容に関心があります。このロールを持つユーザーは BAM のみにアクセスできます。



注記

上記のロールは、ユーザー作成のプロセス中に入力します。

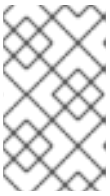
バグを報告する

3.1.4. ユーザーの作成

新規ユーザーを追加するには、EAP の bin ディレクトリーより add-user.sh スクリプト (Unix システムの場合) または add-user.bat ファイル (Windows システムの場合) を実行する必要があります。

1. Unix システムの場合は ./add-user.sh を、Windows システムの場合は add-user.bat を bin ディレクトリーより実行します。

2. ユーザータイプ入力のプロンプトが表示されたら「b」を入力して Application User を選択し、Enter キーを押します。
3. Enter を押してデフォルトのレルム (ApplicationRealm) を許可します。
4. ユーザー名入力のプロンプトが表示されたら、ユーザー名を入力し (例: **helloworlduser**)、確認します。
5. パスワード入力のプロンプトが表示されたらユーザーのパスワードを入力し (例: **Helloworld@123**)、再入力します。



注記

パスワードは 8 文字以上で、アルファベットの大文字と小文字 (A-Z、a-z)、1 文字以上の数字 (0-9)、および 1 文字以上の特殊文字 (~ ! @ # $ % ^ * ( ) - _ + =) が含まれる必要があります。

6. ロール入力のプロンプトが表示されたら、ユーザーが必要となるロールのコンマ区切りリストを入力します (「ロールの定義」 を参照してください)。

   Business Central のユーザーには最低でも **analyst** ロールが必要で、ダッシュビルダーのユーザーには **admin** ロールが必要となります。ロールはコンマ区切りリストで入力する必要があります。
7. ユーザーを追加することを確認します。
8. 次のプロンプトで「yes」を入力します (これにより、必要な場合にクラスタリングを有効にします)。

バグを報告する

3.2. 汎用デプロイ可能バンドルのインストール

Red Hat JBoss Web Server (WS) 上に Red Hat JBoss BPM Suite をインストールするには、製品の汎用デプロイ可能パッケージ (generic deployable package) を使用する必要があります。

WS 上のインストールでは、汎用デプロイ可能パッケージに Red Hat JBoss WS の一部でない追加のトランザクションマネージャーとセキュリティーライブラリーが含まれます。

汎用デプロイ可能パッケージには次の zip アーカイブが含まれています。

- **jboss-bpms-engine.zip**: エンジンをアプリケーションに組み込む場合、サポートされる実行エンジンライブラリーが必要です。
- **jboss-bpms-manager.zip**: **business-central.war** および **dashbuilder.war** Web アプリケーション。

バグを報告する

3.2.1. 汎用デプロイ可能パッケージのダウンロード

JBoss Web Server 用の Red Hat JBoss BPM Suite 汎用デプロイ可能パッケージをダウンロードするには、以下の手順に従います。

1. Red Hat Customer Portal にアクセスし、ログインします。

2. **Downloads** → **Red Hat JBoss Middleware** → **Download Software** の順に選択します。
3. **Product** ドロップダウンメニューより **BPM Suite** を選択します。
4. **Version** ドロップダウンメニューより製品のバージョンを選択します。
5. 表の **Red Hat JBoss BPM Suite 6.0.2 Deployable for all supported containers** の行に移動し、**Download** をクリックします。

バグを報告する

3.2.2. 汎用デプロイ可能パッケージのインストール

汎用デプロイ可能パッケージをインストールするには、基盤のプラットフォーム (Red Hat JBoss WS) をインストールした後に以下を設定する必要があります。

- データベースドライバーとトランザクションマネージャー (Bitronix) を設定します (「Red Hat JBoss Web Server 2.0 (Tomcat 7) へのトランザクションマネージャーの設定」 を参照してください)。
- Business Central アプリケーションを設定します。ユーザーとロールを設定し、永続性を設定します (「Red Hat JBoss Web Server 2.0 (Tomcat 7) への Business Central の設定」 を参照してください)。
- Dashbuilder アプリケーションを設定します。ユーザーとロールを設定し、永続性を設定します (「Red Hat JBoss Web Server 2.0 (Tomcat 7) への Dashbuilder の設定」 を参照してください)。

バグを報告する

3.2.2.1. Red Hat JBoss Web Server 2.0 (Tomcat 7) へのトランザクションマネージャーの設定

1. Red Hat Customer Portal からダウンロードした汎用デプロイ可能 zip パッケージを展開します。この zip パッケージには、**jboss-bpms-engine.zip** と **jboss-bpms-manager.zip** の 2 つの zip ファイルが含まれています。
2. **jboss-bpms-manager.zip** ファイルの内容を一時的な場所に展開します。この zip ファイルには、**business-central.war** と **dashbuilder.war** (展開後の形式) の 2 つの web アプリケーションが含まれ、これらのアプリケーションが一時的な場所に置かれます。これらのフォルダーの名前から **.war** 拡張子を削除し、名前を変更します。

   これらのフォルダーを、***$TOMCAT_DIR*/webapps** フォルダー直下にコピーします。

   これにより、***$TOMCAT_DIR*/webapps/business-central** と ***$TOMCAT_DIR*/webapps/dashbuilder** の 2 つの展開された形式のフォルダーが存在するようになります。

注記

$TOMCAT_DIR は、ご使用の web サーバーがあるホームディレクトリーを意味します。web サーバーのホームディレクトリーへの実際のパスに置き換えてください (例: **/home/john/jboss-ews-2.0/tomcat7/**)。

3. **jboss-bpms-engine.zip** アーカイブの **jboss-bpms-engine** フォルダーを、必要なライブラリーをコピーできる一時的な場所に展開します。展開後、展開されたフォルダーと **lib** フォルダー下にすべてのコア BPMS ライブラリーが含まれるようになります。

4. トランザクションマネージャーをインストールします。
jboss-bpms-engine ライブラリーを ***$TOMCAT_DIR*/lib/** ディレクトリーに展開した **lib** フォルダーより、以下のトランザクションマネージャー jar ライブラリーをコピーします。

   - btm-*VERSION*.jar
   - btm-tomcat55-lifecycle-*VERSION*.jar
   - jta-*VERSION*.jar
   - slf4j-api-*VERSION*.jar
   - slf4j-ext-*VERSION*.jar

   さらに、以下のライブラリーをダウンロードし、***$TOMCAT_DIR*/lib/** フォルダーにコピーします。

   - javax.security.jacc-api.jar

5. ドライバーをご使用のデータベースにインストールします。適切なデータベースドライバーを持つ jar ファイルを ***$TOMCAT_DIR*/lib/** にコピーします。

注記

組み込みの H2 データベースを使用している場合は、ドライバーは **business-central/WEB-INF/lib/** にあります。

6. トランザクションマネージャー設定ファイルを ***$TOMCAT_DIR*/conf/** に作成します。

   - btm-config.properties

```
bitronix.tm.serverId=tomcat-btm-node0
bitronix.tm.journal.disk.logPart1Filename=${btm.root}/work/btm1.tlog
bitronix.tm.journal.disk.logPart2Filename=${btm.root}/work/btm2.tlog
bitronix.tm.resource.configuration=${btm.root}/conf/resources.properties
```

   - resources.properties (resource.ds1.uniqueName defines は後に tomcat リソース定義で使用されるデータソース名を定義します - この値を覚えておいてください)

     ご使用の環境に合わせるため、以下の定義の値を変更してください。

例**3.2 H2** データソース定義

```
resource.ds1.className=bitronix.tm.resource.jdbc.lrc.LrcXADataSource
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=10
resource.ds1.maxPoolSize=20
```

```
resource.ds1.driverProperties.driverClassName=org.h2.Driver
resource.ds1.driverProperties.url=jdbc:h2:file:~/jbpm
resource.ds1.driverProperties.user=sa
resource.ds1.driverProperties.password=
resource.ds1.allowLocalTransactions=true
```

例**3.3 MySQL 5.5** データソース定義

```
resource.ds1.className=com.mysql.jdbc.jdbc2.optional.MysqlXADat
aSource
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=0
resource.ds1.maxPoolSize=10
resource.ds1.driverProperties.URL=jdbc:mysql://localhost:3306/s
ampledb
resource.ds1.driverProperties.user=dbuser
resource.ds1.driverProperties.password=dbpassword
resource.ds1.allowLocalTransactions=true
```

例**3.4 DB2** タイプ **4** のデータソース定義

```
resource.ds1.className=com.ibm.db2.jcc.DB2Driver
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=0
resource.ds1.maxPoolSize=10
resource.ds1.driverProperties.URL=jdbc:db2://localhost:50000/sa
mpledb
resource.ds1.driverProperties.user=dbuser
resource.ds1.driverProperties.password=dbpassword
resource.ds1.allowLocalTransactions=true
```

例**3.5 Oracle** のデータソース定義

```
resource.ds1.className=oracle.jdbc.xa.client.OracleXADataSource
resource.ds1.uniqueName=jdbc/jbpm
resource.ds1.minPoolSize=0
resource.ds1.maxPoolSize=10
resource.ds1.driverProperties.URL=jdbc:oracle:thin:@//localhost
:1521/bpms
resource.ds1.driverProperties.user=dbuser
resource.ds1.driverProperties.password=dbpassword
resource.ds1.allowLocalTransactions=true
```

例**3.6 Microsoft SQL** サーバーのデータソース定義

```
resource.ds1.className=com.microsoft.sqlserver.jdbc.SQLServerDr
iver
resource.ds1.uniqueName=jdbc/jbpm
```

```
resource.ds1.minPoolSize=0
resource.ds1.maxPoolSize=10
resource.ds1.driverProperties.URL=jdbc:sqlserver://localhost:14
33;databaseName=bpms;
resource.ds1.driverProperties.user=dbuser
resource.ds1.driverProperties.password=dbpassword
resource.ds1.allowLocalTransactions=true
```

7. トランザクションマネージャーリスナーを ***$TOMCAT_DIR*/conf/server.xml** で設定し、コンテナの起動およびシャットダウン時に Bitronix を起動および停止します。

   以下の要素を最後の **<Listener>** 要素として **<Server>** 要素に追加します。

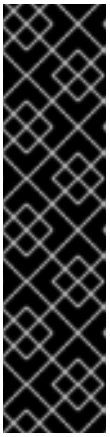
   ```
   <Listener
   className="bitronix.tm.integration.tomcat55.BTMLifecycleListener" />
   ```

8. btm.root システムプロパティーと、bitronix 設定ファイルが置かれる場所を定義します。

   ***$TOMCAT_DIR*/bin/** 内に、以下の内容が含まれる **setenv.sh** ファイルを作成します。

   ```
   CATALINA_OPTS="-Xmx512M -XX:MaxPermSize=512m -
   Dbtm.root=$CATALINA_HOME -
   Dbitronix.tm.configuration=$CATALINA_HOME/conf/btm-config.properties
   -Dorg.jbpm.designer.perspective=RuleFlow"
   ```

   必要な場合はファイルの実行パーミッションを付与します。デザイナーのデフォルトのパースペクティブを **Full** ではなく **RuleFlow** にするため、最後のプロパティー **org.jbpm.designer.perspective** は **RuleFlow** に設定されます。



重要

Microsoft Windows システムでは、ファイル内容の **$CATALINA_HOME** の値を同等の環境変数名に置き換えるか、以下の例のように絶対パスを使用して **setenv.bat** に値を追加します。

```
set "CATALINA_OPTS=-Xmx512M -XX:MaxPermSize=512m -
Dbtm.root=C:/Tomcat -
Dbitronix.tm.configuration=C:/Tomcat/conf/btm-
config.properties -
Dorg.jbpm.designer.perspective=RuleFlow"
```

バグを報告する

3.2.2.2. Red Hat JBoss Web Server 2.0 (Tomcat 7) への Business Central の設定

以下の手順に従って、Business Central を設定します。

1. Tomcat に設定されたユーザーが、Business Central の Web アプリケーションによってロードされるようにするため、**Valve** を設定します。

   a. ユーザーとロールを ***$TOMCAT_DIR*/conf/tomcat-users.xml** に定義します。Business Centralには、**admin** や **analyst** として指定されたロールを持つユーザーが必要であるこ

とに注意してください (ユーザーおよびロール定義の詳細は、Tomcat 7 のドキュメントを参照してください)。

以下のプログラムリストは、admin および analyst ロールを追加し、**bpmsadmin** というユーザーにこれらのロールを割り当てる方法の例を示しています。

```
  <role rolename="admin"/>
  <role rolename="analyst" />
  <user username="bpmsadmin" password="P@ssw0rd"
roles="admin,analyst"/>
```

b. ***$TOMCAT_DIR*/webapps/business-central/WEB-INF/lib/** にある **kie-tomcat-integration-*VERSION*.jar** を ***$TOMCAT_DIR*/lib/** にコピーします。

c. ***$TOMCAT_DIR*/webapps/business-central/WEB-INF/lib/** にある **jaxb-api-*VERSION*.jar** を ***$TOMCAT_DIR*/lib/** にコピーします。

d. ***$TOMCAT_DIR*/conf/server.xml** 内で、Tomcat Valve 宣言を関連する **<host>** 要素に追加します。

```
<Valve className="org.kie.integration.tomcat.JACCValve" />
```

e. ***$TOMCAT_DIR*/webapps/business-central/WEB-INF/web.xml** にて、**TOMCAT-JEE-SECURITY** コメントが付けられたエントリーをアンコメントします。

f. Tomcat 認証ソースを設定します。***$TOMCAT_DIR*/webapps/business-central/WEB-INF/classes/META-INF/services/** ディレクトリーにて、**org.uberfire.security.auth.AuthenticationSource** ファイルの名前を **org.uberfire.security.auth.AuthenticationSource-ORIGIN** に変更し、**org.uberfire.security.auth.AuthenticationSource-TOMCAT-JEE-SECURITY** ファイルの名前を **org.uberfire.security.auth.AuthenticationSource** に変更します。

```
# Example command if you run this from $TOMCAT_DIR/webapps
directory
$ mv business-central/WEB-INF/classes/META-
INF/services/org.uberfire.security.auth.AuthenticationSource
business-central/WEB-INF/classes/META-
INF/services/org.uberfire.security.auth.AuthenticationSource-
ORIGIN
$ mv business-central/WEB-INF/classes/META-
INF/services/org.uberfire.security.auth.AuthenticationSource-
TOMCAT-JEE-SECURITY business-central/WEB-INF/classes/META-
INF/services/org.uberfire.security.auth.AuthenticationSource
```

g. ***$TOMCAT_DIR*/webapps/business-central/WEB-INF/beans.xml** にて、**JAASUserGroupInfoProducer** をアンコメントし、**org.jbpm.kie.services.cdi.producer.DefaultUserGroupInfoProducer** をコメントアウトします (任意)。このファイルの **alternatives** 部分が以下のようになるはずです。

```
  <alternatives>
```

```
    <!--
<class>org.jbpm.kie.services.cdi.producer.DefaultUserGroupInfoPro
ducer</class> -->
    <!-- uncomment JAASUserGroupInfoProducer when using JEE
security on Tomcat -->

<class>org.jbpm.kie.services.cdi.producer.JAASUserGroupInfoProduc
er</class>
  </alternatives>
```

2. 基盤の H2 データベースが提供するデフォルトデータソースではないデータソースを使用している場合は、永続性を設定する必要があります。デフォルトの H2 データベースを使用している場合は、これ以降の手順を省略できます。

   この手順では、以前に MySQL オプション向けに bitronix の **resources.properties** ファイルの uniqueName=jdbc/jbpm に定義された JNDI 名 **jdbc/myDatasource** を用いてデータソースを設定します。

   a. **business-central/META-INF/context.xml** にて、<Resource> 要素のデータソース JNDI 名を置き換えます。uniqueName 属性は、**resources.properties** に設定された resource.ds1.uniqueName プロパティーを参照します。

   ```
   <Resource name="jdbc/myDatasource" uniqueName="jdbc/jbpm"
   auth="Container" removeAbandoned="true"
   factory="bitronix.tm.resource.ResourceObjectFactory"
   type="javax.sql.DataSource"/>
   ```

   b. **business-central/WEB-INF/web.xml** にて、<res-ref-name> 要素のデータソース JNDI 名を使用するデータソース名に置き換えます。

   ```
   <resource-ref>
           <description>Console DS</description>
           <res-ref-name>jdbc/myDatasource</res-ref-name>
           <res-type>javax.sql.DataSource</res-type>
           <res-auth>Container</res-auth>
   </resource-ref>
   ```

   c. **business-central/WEB-INF/classes/META-INF/persistence.xml** を変更します。

   H2 以外のデータベースを使用している場合、このファイルでデータベースの Hibernate ダイアレクトの名前を変更します。以下のコードは、**persistence.xml** の元のデータベース情報を示しています。

   ```
   <property name="hibernate.dialect"
   value="org.hibernate.dialect.H2Dialect"/>
   ```

   この情報は次のように更新できます (以下は MySQL データベースの場合)。

   ```
   <property name="hibernate.dialect"
   value="org.hibernate.dialect.MySQLDialect"/>
   ```

注記

DB2 のダイアレクトは **org.hibernate.dialect.DB2Dialect** になります。AS/400 上の DB2 の場合は **org.hibernate.dialect.DB2400Dialect**、Oracle は **org.hibernate.dialect.Oracle10gDialect**、Microsoft SQL Server は **org.hibernate.dialect.SQLServerDialect** になります。

d. BPMS プロセスエンジンが新しいデータベースを使用できるようにするため、**business-central/WEB-INF/classes/META-INF/persistence.xml** ファイルを変更します。

以下のコードは、**persistence.xml** の元のデータソース情報を示しています。

```
<jta-data-source>java:comp/env/jdbc/jbpm</jta-data-source>
```

この値を、前に定義したデータソースに変更します。

```
<jta-data-source>java:comp/env/jdbc/myDatasource</jta-data-source>
```

3. これで JBoss Web Server を起動し、Business Central へログインできるようになります。

a. *$TOMCAT_HOME*/**bin** ディレクトリーで、**startup.sh** を実行します。

```
./startup.sh
```

b. Web ブラウザで http://localhost:8080/business-central にアクセスします。

c. ユーザーロールを定義した **tomcat-users.xml** ファイルに指定されている正しいユーザー名とパスワードを使用して、ログインします。

バグを報告する

3.2.2.3. Red Hat JBoss Web Server 2.0 (Tomcat 7) への Dashbuilder の設定

Red Hat JBoss Web Server 上で Dashbuilder を設定するには、以下の手順に従います。

1. ユーザーとロールを ***$TOMCAT_DIR*/conf/tomcat-users.xml** に定義します。Dashbuilder には、**admin** や **analyst** として指定されたロールを持つユーザーが必要であることに注意してください。すでにこれらのユーザーが Business Central 向けに定義されている場合は、再度定義する必要はありません。

2. ***$TOMCAT_DIR*/conf/server.xml** ファイルの以下の行をアンコメントし、Dashbuilder と Business Central との間のシングルサインオンを有効にします。

```
<Valve className="org.apache.catalina.authenticator.SingleSignOn" />
```

3. Business Central の設定と同様に、デフォルトおよび統合された H2 データベース以外のデータベースを使用している場合は、永続性を設定する必要があります。

ここでは、bitronix の **resources.properties** ファイルの uniqueName=jdbc/jbpm に定義された JNDI 名 **jdbc/dashbuilderDS** でデータソースを設定します。

a. **dashbuilder/META-INF/context.xml** にて、**<Resource>** 要素のデータソース JNDI 名 を置き換えます。uniqueName 属性は **resources.properties** の resource.ds1.uniqueName プロパティーセットを参照します。

```
<Resource name="jdbc/dashbuilderDS" uniqueName="jdbc/jbpm"
auth="Container" removeAbandoned="true"
factory="bitronix.tm.resource.ResourceObjectFactory"
type="javax.sql.DataSource"/>
```

注記

データベースによっては、ここに他のプロパティーを定義する必要があることがあります。たとえば、Oracle 環境ではこのエントリーは次のようになります。

```
<Resource name="jdbc/jbpm" uniqueName="jdbc/jbpm"
auth="Container"  removeAbandoned="true"
factory="bitronix.tm.resource.ResourceObjectFactory"
type="javax.sql.DataSource" username="username"
password="password"
driverClassName="oracle.jdbc.xa.client.OracleXADataSou
rce" url="jdbc:oracle:thin:YOUR-URL:1521:YOUR-DB"
maxActive="8" />
```

b. **dashbuilder/WEB-INF/web.xml** にて、データソース名で **<res-ref-name>** 要素にデータソース JNDI 名を追加します。

```
<resource-ref>
    <description>Dashboard Builder Datasource</description>
    <res-ref-name>jdbc/dashbuilderDS</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
    <res-auth>Container</res-auth>
</resource-ref>
```

c. **dashbuilder/META-INF/context.xml** でトランザクションファクトリーを定義します。

```
 <Transaction
factory="bitronix.tm.BitronixUserTransactionObjectFactory"/>
```

d. **<session-factory>** 要素の **dashbuilder/WEB-INF/etc/hibernate.cfg.xml** にあるデータソース JNDI 名を更新します。

```
<property
name="connection.datasource">java:/comp/env/jdbc/dashbuilderDS</p
roperty>
```

4. Java Web サーバーを再起動し、これらの変更を適応します。再起動したら、 Business Central 内から Dashbuilder にアクセスするか、直接 **http://localhost:8080/dashbuilder** にアクセスします。

バグを報告する

3.3. サーバーの起動

インストーラーまたは EAP 6 のバンドルインストールを使用して BPMS をインストールした場合は、2 つのモードの 1 つでサーバーを起動できます。

注記

Red Hat Java Web Server 上の汎用デプロイ可能バージョンを使用して BPMS をインストールした場合は、ダウンロードおよびインストールの手順にサーバー起動の手順が含まれているため、以下の説明に従う必要はありません。

Red Hat JBoss BPM Suite に含まれるデフォルトの起動スクリプトは、パフォーマンス重視で最適化されています。パフォーマンスモードでサーバーを実行する場合は、以下の手順に従います。

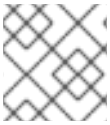
1. コマンドラインで、***$SERVER_HOME*/bin/** ディレクトリーへ移動します。
2. Unix 環境では以下を実行します。

```
./standalone.sh
```

Windows 環境では以下を実行します。

```
./standalone.bat
```

Red Hat JBoss BPM Suite には、セキュリティー重視で最適化されている **standalone-secure.sh** スクリプトも含まれています。 このスクリプトによって、既知の脆弱性を保護するセキュリティーポリシーがデフォルトで適用されます。



注記

本番環境では **standalone-secure.sh** スクリプトの使用が推奨されます。

警告

セキュリティーマネージャーを使用すると、パフォーマンスが著しく劣化することに注意してください。セキュリティーとパフォーマンスのバランスは、個々の状況を判断して決定する必要があります。「Java Security Manager とパフォーマンス管理」 を参照してください。

このスクリプトを用いてサーバーをセキュアモードで実行する場合は、以下の手順に従います。

1. コマンドラインで、***$SERVER_HOME*/bin/** ディレクトリーへ移動します。
2. Unix 環境では以下を実行します。

```
./standalone-secure.sh
```

Windows 環境では以下を実行します。

```
./standalone-secure.bat
```

バグを報告する

3.4. JAVA SECURITY MANAGER とパフォーマンス管理

前述のとおり、BPMS で MVEL スクリプトの評価をサンドボックス化できるように Java Security Manager (JSM) を有効にすると、高負荷の環境ではパフォーマンスが劣化します。BPMS をデプロイする場合、環境やパフォーマンスに注意する必要があります。以下のガイドラインを使用して、セキュアで高パフォーマンスな BPMS アプリケーションをデプロイしてください。

- パフォーマンスが重要な高負荷の環境では、他のシステムで開発され、適切に評価されたアプリケーションのみをデプロイすることが推奨されます。また、このようなシステムでは、Analyst ロールを持つユーザーを作成しないことが推奨されます。このような対策が取られた場合、パフォーマンスの劣化が発生しないため、このようなシステムで JSM を無効にしても安全です。

- 高負荷にならないテストおよび開発環境や、ルールやプロセスのオーサリングが外部ネットワークに公開される環境では、MVEL の評価を適切にサンドボックス化してセキュリティーを強化するため、JSM を有効にすることが推奨されます。

JSM が無効な状態で、Analyst ロールを持つユーザーに Business Central コンソールへのログインを許可するのはセキュアでないため、推奨されません。

バグを報告する

第4章 BUSINESS CENTRAL へのログイン

サーバーの起動後に、Business Central へログインします。

1. Web ブラウザーで http://localhost:8080/business-central にアクセスします。ドメイン名から実行するようユーザーインターフェースが設定されている場合は、http://www.example.com:8080/business-central のように **localhost** をドメイン名に置き換えます。

2. インストール中に作成されたユーザークレデンシャルを使用してログインします (例: ユーザー **helloworlduser** とパスワード **Helloworld@123**)。

バグを報告する

第5章 HELLO WORLD プロジェクト

Red Hat JBoss BPM Suite の基本機能を実証するため、本章では **Hello World** ビジネスルールを使用して **Hello World** プロジェクトを設定する方法を説明します。このビジネスプロセスは、**Hello World!** メッセージを表示し、実行を完了します。

手順では以下を行います。

1. Artifact リポジトリーに Hello World リポジトリーを作成します。
2. **HelloWorld** ビジネスプロセス定義で Hello World プロジェクトを作成します。
3. グラフィカルなプロセスデザイナーツールを使用して、Hello World プロセスロジックをビジネスプロセスにモデル化します。
4. プロジェクトをローカルで実行されている実行エンジンに構築およびデプロイします。
5. Hello World プロセスを実行します。
6. プロセスの実行を監視します。

バグを報告する

5.1. リポジトリー構造の作成

すべてのビジネス資産は、組織単位 (Organizational Unit) にあるリポジトリーに存在します。組織単位は Artifact リポジトリーのディレクトリーです。デフォルトでは、Artifact リポジトリーには組織単位が含まれていません。そのため、ビジネスプロセスなどの独自のビジネス資産を作成するには、Artifact リポジトリーに組織単位を作成し、リポジトリー (Git) を作成する必要があります。作成後、リポジトリーにプロジェクトを追加できます。プロジェクトには、内容を論理「ディレクトリー」構造にする任意のパッケージ構造を含むことができます。ビジネス資産には任意のパッケージを追加できます (Artifact リポジトリーの詳細は、『Red Hat JBoss BPMS ユーザーガイド』を参照してください)。

リポジトリー構造を作成するには、以下の手順に従います。

1. Web ブラウザーで Business Central を開き (ローカルで実行している場合は http://localhost:8080/business-central)、**admin** ロールを持つユーザーとしてログインします (**helloworlduser**)。
2. Artifact リポジトリーに組織単位を作成します。
   a. **Authoring** → **Administration** に移動します。
   b. パースペクティブメニューで **Organizational Units** → **Manage Organizational Units** の順に選択します。
   c. 表示された **Organizational Unit Manager** ビューで **Add** をクリックします。

      表示された **Add New Organizational Unit** ダイアログボックスでユニットプロパティーを定義し、**OK** をクリックします。

      - 名前: **helloworld**
      - 所有者: **helloworlduser**
3. helloworld 組織単位に新しいリポジトリーを作成します。

a. **Authoring** → **Administration** に移動します。

b. パースペクティブメニューで **Repositories** → **New repository** の順に選択します。

c. 表示された **Create Repository** ダイアログボックスでリポジトリープロパティーを定義します。

- リポジトリー名: **helloworldrepo**
- 組織単位: **helloworld**

4. **Authoring** → **Project Authoring** に移動します。

5. 組織単位ドロップダウンボックスの Project Explorer で **helloworld** を選択し、リポジトリードロップダウンボックスで **helloworldrepo** を選択します。

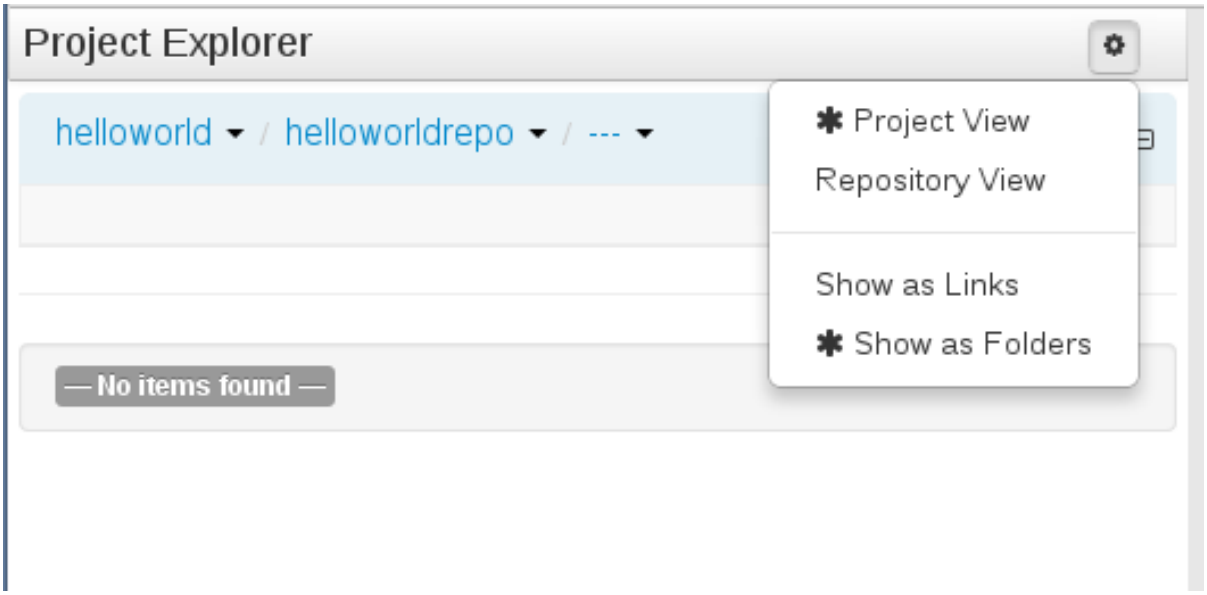


図5.1 **Project Explorer** の **helloworld** 組織単位で **helloworldrepo** リポジトリーを選択

6. **helloworld** リポジトリーでプロジェクトを作成します。

a. パースペクティブメニューで **New Item** → **Project** に移動します。

b. 表示された **Create new** ダイアログボックスでプロジェクトプロパティーを定義します。

- リソース名 (プロジェクト名): **HelloWorld**
- パス: **default://master@helloworldrepo/**

c. **New Project Wizard** ダイアログで、プロジェクトの maven プロパティーを定義します。各エントリーの入力後に Enter キーを押します。

- グループ ID: **org.brms**
- アーティファクト ID: **HelloWorld**
- バージョン ID: **1.0**

d. **Finish** をクリックします。

バグを報告する

第6章 HELLO WORLD プロセス

本章では、継続して Hello World の例を取り上げます。エンドツーエンドのビジネスプロセスを作成し、基本的な Hello World プロセスの作成について説明します。

バグを報告する

6.1. ビジネスプロセスの作成

新しいビジネスプロセス定義を作成するには、以下の手順に従います。

1. プロジェクトオーサリングパースペクティブを表示します (**Authoring** → **Project Authoring**)。
2. Project Explorer ビューの左側で **helloworld** 組織単位、**helloworldrepo** リポジトリー、**HelloWorld** プロジェクトおよび org.jbpm パッケージを選択します。これにより、ビジネスプロセス定義を作成する Artifact リポジトリーの場所が定義されます。

注記

必ず org.jbpm パッケージを選択してください。誤ったパッケージを選択するとデプロイメントに失敗します。

3. パースペクティブメニューで **New Item** → **Business Process** とクリックし、プロセス定義の詳細を定義します。
   - **HelloWorld** をリソース名として入力します。
4. **OK** をクリックします。

作成されたプロセス定義のキャンバスを持つプロセスデザイナーが開かれます。

バグを報告する

6.2. ビジネスプロセスのモデル化

ビジネスプロセス定義を作成した後、ビジネスプロセスデザイナーでビジネスプロセスを設計できます。ビジネスプロセスデザイナーは右側のタブに開かれている必要があります。プロジェクトエクスプローラーのタブを閉じてしまった場合は、**Business Processes** 下の **HelloWorld** をクリックして、ビジネスプロセスデザイナーを再度開きます。正しいパッケージ (org.jbpm.helloworld) を選択するようにしてください。プロセスの内容を設定するには、以下の手順に従います。

1. プロセス要素を持つ **Object Library** パレットを展開します。ビジネスプロセスデザイナータブの左上隅にある二重矢印のボタン () をクリックします。
2. Start Event 要素がキャンバスに表示されます。
3. Start Event 要素をクリックします。クイックリンク項目がノードの近くに表示されます。Task アイコン () をクリックし、外向きのシーケンスフローおよび Start Event へ紐付けされた Task 要素を作成します。
4. Timer Event を作成します。パレットから **Timer** (**Catching Intermediate Events** の下にある) をドラッグアンドドロップし、クイックリンクメニューを使用して Task を Timer 要素に紐付けします。

5. Timer Event 要素へ紐付けされる End Event 要素を作成します (クイックリンク機能を使用するか、パレットよりドラッグします)。

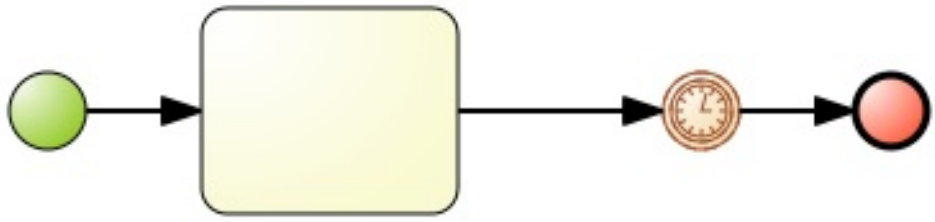


図6.1 HelloWorld プロセスのドラフト

バグを報告する

6.3. 要素プロパティーの定義

ここで、要素プロパティーを定義する必要があります。

1. 右上隅にある二重矢印 ( ) をクリックし、**Properties** タブを展開します。
2. キャンバスのどこかをクリックします。**Properties** タブに **HelloWorld** プロセスのプロパティーが表示されます。必要なプロパティーは事前定義されていますが、必要な場合は値を変更できます。
3. Start 要素と End 要素は任意のプロパティーのみを持ちますが、Task 要素にはタイプとタイプ固有のプロパティーを定義する必要があります。Task 要素をクリックし、必要なプロパティーを定義します。
   - 名前: **ScriptTask**
   - タスクタイプ: **Script**
   - スクリプト言語: **java**
   - スクリプト: **System.out.println("Hello World!");**

     OK をクリックします。

   これで実行時にスクリプトを実行するタスクが定義されました。スクリプトは Java で定義され、**System.out.println("Hello World!");** メソッドを実行します。このメソッドは、サーバーの標準出力に **Hello World!** を書き込みます (デフォルトでは、サーバーの標準出力はサーバーが起動したコンソールになります)。
4. 実行を検証できるようにするため、プロセスが起動時に待機するよう、タイマーイベントのプロパティーを定義する必要があります。タイマー要素をクリックし、必要なプロパティーを定義します。
   - タイムサイクル言語: **Cron**
   - 期間: **1m** (定義後、Enter を押します)

     よって、プロセスはスクリプトタスクの実行後、タイマーイベントを 1 分間待ちます。

5. プロパティーよりタイマーイベントに名前を付け、プロセスを保存します。**Save** メニュー ( ) を開き、**Save** をクリックします。

6. プロセスデザイナーのツールバーにあるボタン ( ) をクリックし、定義されたプロセスが有効であることを確認します。1 つ以上の検証エラーがある要素は外形がオレンジ色になります。検証機能については、『Red Hat JBoss BPMS ユーザーガイド』を参照してください。

プロセス要素やそれらのプロパティーに関する詳細は、『JBoss BPMS ユーザーガイド』を参照してください。

バグを報告する

6.4. 構築とデプロイ

Hello World プロセスをインスタンス化 (実行) する前に、プロジェクト全体を構築し、実行サーバーへデプロイする必要があります。

1. Business Central のメインメニューで **Authoring** → **Project Authoring** に移動します。

2. Project Explorer で Hello World プロジェクトを見つけます。

3. プロジェクトエディターでプロジェクトを開きます。**Tools** → **Project Editor** とクリックします。

4. **Project Screen** に正しいプロジェクト詳細が表示されていることを確認し、Project Screen ビューの右上隅にある **Build & Deploy** ボタンをクリックします。

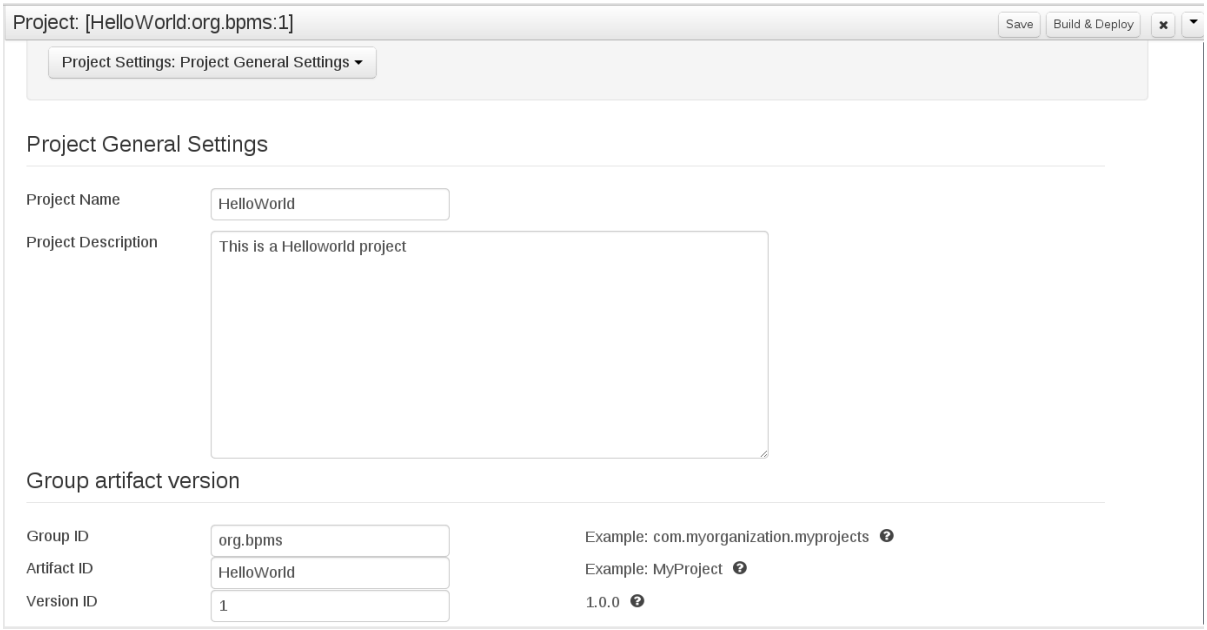


図6.2 プロジェクトエディターと **helloWorld** プロジェクトプロパティー

プロジェクトの構築および実行サーバーへのデプロイメントが完了し、インスタンス化が可能になったことを伝える緑色の通知が画面上部に表示されます。

バグを報告する

6.5. ビジネスプロセスのインスタンス化

Hello World プロセスのインスタンスを作成する (ビジネスプロセスを実行する) には、以下の手順に従います。

1. メインメニューの **Process Management** → **Process Definitions** をクリックします。
2. 表示された **Process Definitions** タブで **Hello World** を見つけます。**Refresh** をクリックしてリストにデプロイメントを表示する必要があることがあります。
3. プロセス定義エントリーの横にある **Start** ボタン () をクリックし、**Form** ダイアログ内の Play ボタン () をクリックしてプロセスをインスタンス化することを確認します。

   現在ログインしているユーザーがプロセス所有者となりプロセスがインスタンス化され、プロセスフォームが表示されます (定義されている場合、プロセスのインスタンス化で、フォームによってユーザーからの入力を要求できます。詳細は『Red Hat JBoss BPMS ユーザーガイド』を参照してください)。

起動されたプロセスインスタンスの詳細を示す **Process Instance Details** ビューが表示されます。**Hello World!** メッセージが標準出力に出力されます。通常、標準出力はサーバーが起動したターミナルエミュレーターになります。その後、プロセスインスタンスがタイマーイベントを待ちます。**Views - Process Model** をクリックして、現在の実行状態を確認します。

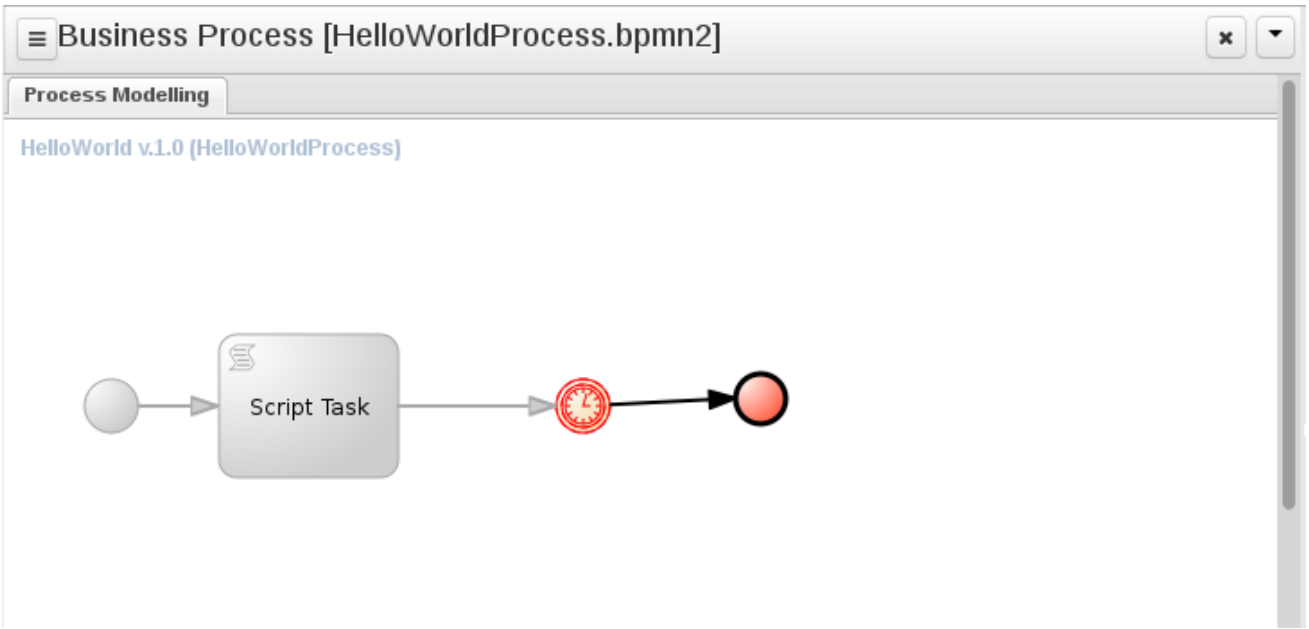


図**6.3 HelloWorld** のリアルタイム実行図: タイマーイベントの実行

バグを報告する

6.6. ビジネスプロセスのアボート

この時点で、**HelloWorld** プロセスのインスタンスは実行サーバー上で実行され、**Process Instances** ビューで確認できます。このビューを表示するには、**Process Management** → **Process Instances** に移動します。

このビューでは、Details ボタン () を使用してプロセスインスタンスの詳細を表示できます。また、プロセスインスタンスへのシグナル送信 () や、そのアボート () など、基本的な管理アクションを実行できます。

HelloWorld プロセスのタイマーイベント待ちをアボートします。**HelloWorld** プロセスインスタンスの情報がある行の ( ) をクリックします。アクティブなプロセスインスタンスのリストから削除され、**Aborted** インスタンスのリストに表示されます。

図6.4 HelloWorld プロセスインスタンスが含まれる Aborted プロセスインスタンス

バグを報告する

第7章 HELLO WORLD ビジネスルール

BPMS には統合された BRMS が含まれるため、本章ではビジネスルールの仕組みと BPMS への統合方法について初心者向けに説明します。

特定のビジネスルールグループのビジネスルールを発火 (チェック) する、新しいタスク (ビジネスルールタスク) を HelloWorld プロセスに追加します。

ここでは大変簡単な統合ケースを取り上げますが、本番環境ではデシジョンテーブルなどの高度な概念や技術が必要になることがあります。ビジネスルールや BRMS の詳細は、『Red Hat JBoss BRMS ユーザーガイド』を参照してください。

バグを報告する

7.1. ビジネスルールの作成

ビジネスルールは when-then ステートメントで定義され、true の場合に何かが発生します (例: 18 歳以上の場合に特定のコンテンツにアクセスできる)。ビジネスルールは DRL ファイルに保存されます。

HelloWorld プロジェクトの **helloworldrule** ビジネスルールを定義するには、以下の手順に従います。

1. **Project Explorer** にて **helloworld** 組織単位、**helloworldrepo** リポジトリー、**HelloWorld** プロジェクトおよび default パッケージを選択します。

注記

必ず default パッケージを選択してください。誤ったパッケージを選択するとデプロイメントに失敗します。

2. DRL ファイルを作成します。
   a. パースペクティブメニューで **New Item** → **DRL file** と選択します。
   b. **Create new** ダイアログボックスで、リソース名を **helloworldrule** として定義し、宛先パスが **default://master@helloworldrepo/HelloWorld/src/main/resources** であることを確認します。
   c. **OK** をクリックします。
3. 表示された DRL エディターと **helloworldrule.drl** ファイルでルールを定義します。

```
rule "helloworldrule"
ruleflow-group "helloworldgroup"
when
then
 System.out.println("Hello World!");
end
```

このルールは when 条件を定義しません。そのため、発火される (実行するため呼び出される) と常に true になり、**Hello World!** フレーズが出力されます。

4. **Save** をクリックします。

5. **Save this item** プロンプトが表示されます。**Check-in comment** を入力し、Save をクリックします。

注記

check-in comment は変更内容の簡単な説明などで、資産を保存するたびに入力する必要があります。

バグを報告する

7.2. ビジネスルールタスクの追加

ビジネスルールタスクは、特定のルールフローグループに属するルールを発火するタスクのことです。

ビジネスルールタスクをプロセスに追加するには、以下の手順に従います。

1. ビジネスプロセスをプロセスデザイナーで開きます。**Project Explorer** で **HelloWorld** プロジェクトと org.jbpm パッケージを選択します。**BUSINESS PROCESSES** をクリックし、**HelloWorld** プロセスをクリックします。

注記

必ず org.jbpm パッケージを選択してください。誤ったパッケージを選択するとデプロイメントに失敗します。

2. helloworld プロセスがある表示されたビジネスプロセスデザイナーで **Object Library** パレットを展開します。ビジネスプロセスデザイナータブの左上隅にある二重矢印のボタン () をクリックします。

3. **Tasks** メニューを展開し、ビジネスルールタスクを右側のキャンバスへドラッグアンドドロップします。

   タスクをワークフローへ統合するために、フローの紐付けを調整します。

4. ビジネスルールタスクを選択し、そのプロパティーを **Properties** パネルに定義します。

   - 名前: **BusinessRule**
   - ルールフローグループ: **helloworldgroup**

     Ruleflow Group プロパティーは、タスクが実行された時に発火されるルールのグループを定義します。この例では、**helloworldrule.drl** に定義された **helloworldrule** ルールのみが、**helloworldgroup** グループに存在します。

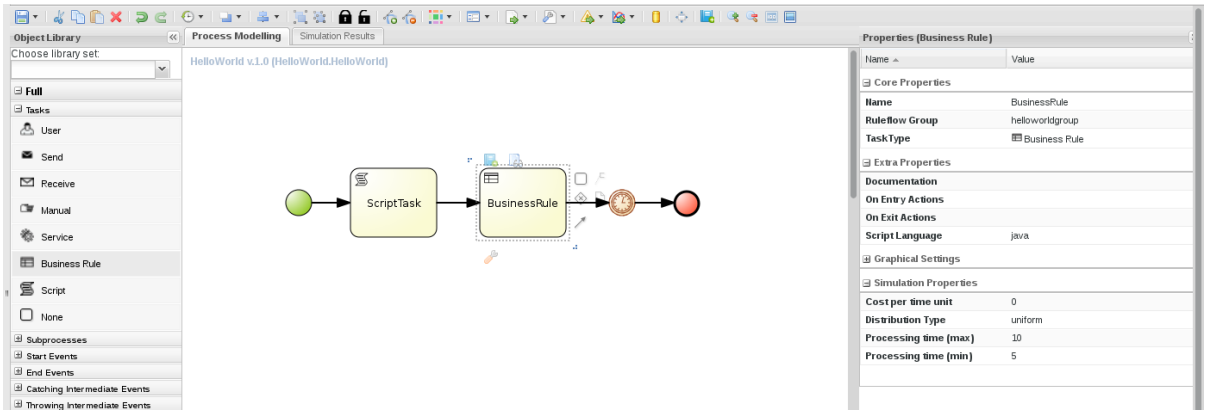


図7.1 **helloworld** プロセスのビジネスルールタスクおよびそのプロパティー

5. Save ボタンをクリックし、変更を保存します。

バグを報告する

7.3. 構築とデプロイ

プロジェクト全体を構築し、実行サーバーへデプロイします。

1. Business Central のメインメニューで **Authoring** → **Project Authoring** に移動します。
2. Project Explorer で Hello World プロジェクトを見つけます。
3. プロジェクトエディターでプロジェクトを開きます。**Tools** → **Project Editor** とクリックします。
4. **`Project Screen`** に正しいプロジェクト詳細が表示されていることを確認し、Project Screen ビューの右上隅にある **`Build & Deploy`** ボタンをクリックします。

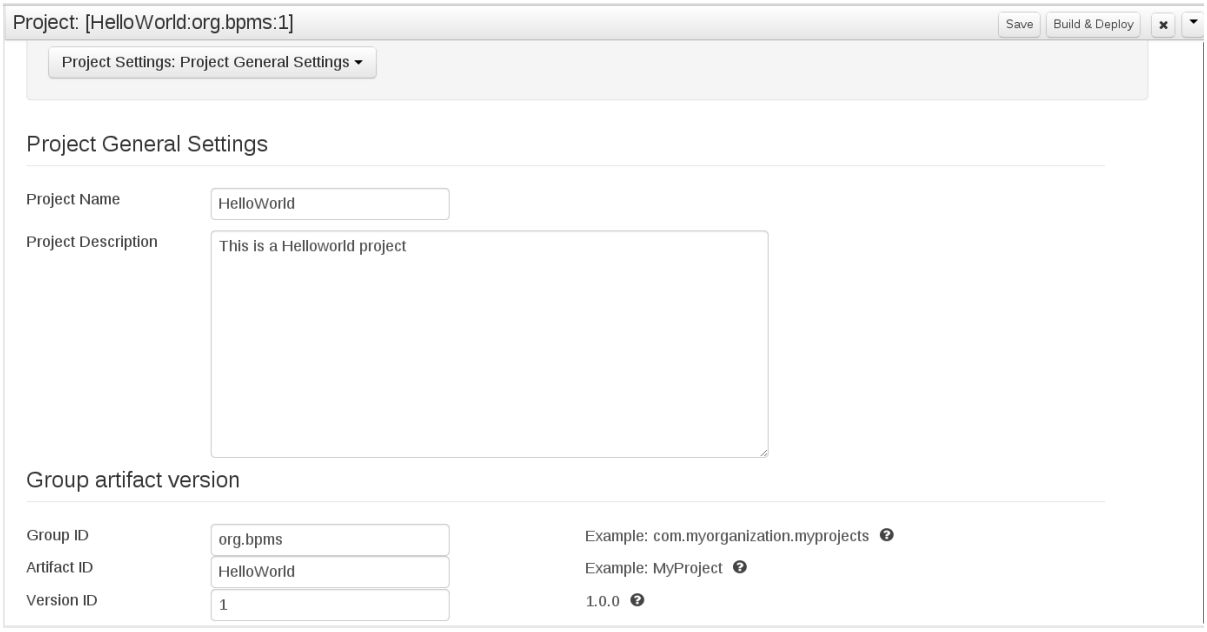


図7.2 プロジェクトエディターと **helloWorld** プロジェクトプロパティー

プロジェクトの構築および実行サーバーへのデプロイメントが完了し、インスタンス化が可能になったことを伝える緑色の通知が画面上部に表示されます。以前のバージョンの **`helloWorld`** デプロイメントが、Business Rule Task がある新バージョンに置き換えられることに注意してください。新旧両方の

デプロイメントを保持したい場合は、プロジェクトエディターにあるプロジェクトのバージョン番号を変更します。

バグを報告する

7.4. ビジネスプロセスのインスタンス化

Hello World プロセスのインスタンスを作成する (ビジネスプロセスを実行する) には、以下の手順に従います。

1. メインメニューの **Process Management** → **Process Definitions** をクリックします。
2. 表示された **Process Definitions** タブで **Hello World** を見つけます。**Refresh** をクリックしてリストにデプロイメントを表示する必要があることがあります。
3. プロセス定義エントリーの横にある **Start** ボタン ( ) をクリックし、**Form** ダイアログ内の Play ボタン ( ) をクリックしてプロセスをインスタンス化することを確認します。

   現在ログインしているユーザーがプロセス所有者となりプロセスがインスタンス化され、プロセスフォームが表示されます (定義されている場合、プロセスのインスタンス化で、フォームによってユーザーからの入力を要求できます。詳細は『Red Hat JBoss BPMS ユーザーガイド』を参照してください)。

起動されたプロセスインスタンスの詳細を示す **Process Instance Details** ビューが表示されます。**Hello World!** メッセージが標準出力に 2 回出力されます。通常、標準出力はサーバーが起動したターミナルエミュレーターになります。その後、プロセスインスタンスがタイマーイベントを待ちます。**Views - Process Model** をクリックして、現在の実行状態を確認します。

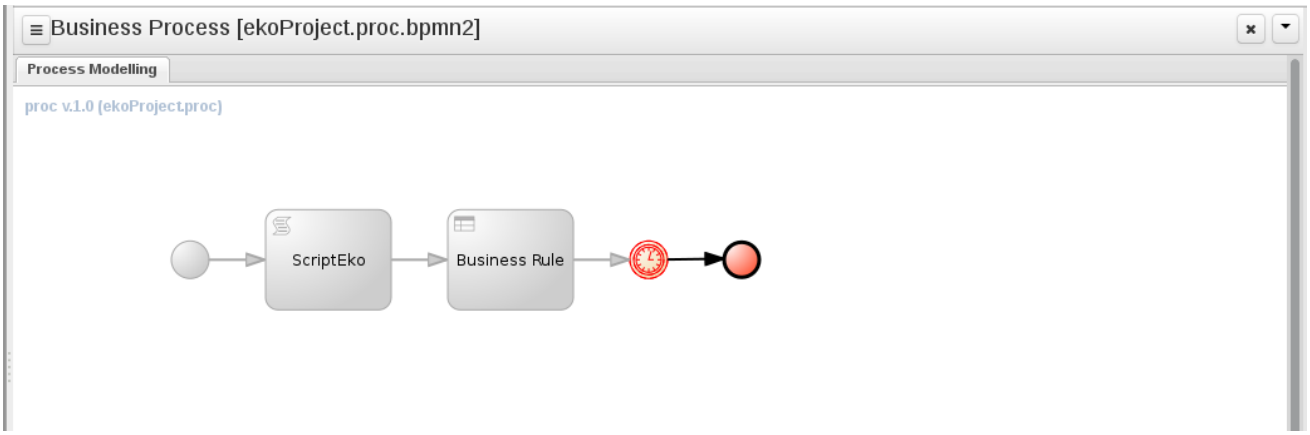


図7.3 HelloWorld のリアルタイム実行図: タイマーイベントの実行

バグを報告する

第8章 BAM

8.1. RED HAT JBOSS BPM SUITE DASHBUILDER へのアクセス

Dashbuilder は、Red Hat JBoss BPM Suite に含まれるビジネスアクティビティー監視用の Web ベースユーザーインターフェースです。Business Central から Dashbuilder にアクセスするには、**Dashboards → Process & Task Dashboards** と選択します。

表示されたダッシュボードには、左側で選択されたランタイムデーターの統計が表示されます。Dashbuilder で独自のダッシュボードを作成するには、**Dashboards → Business Dashboards** とクリックして Dashbuilder を表示します。

バグを報告する

8.2. インスタンスの監視

Dashbuilder は、実行エンジン上のランタイムデータ (プロセスインスタンスおよびタスク) の状態を監視できる特別なダッシュボードを提供します。

このデーターを表示するには、**Dashboards → Process & Task Dashboard** と選択します。

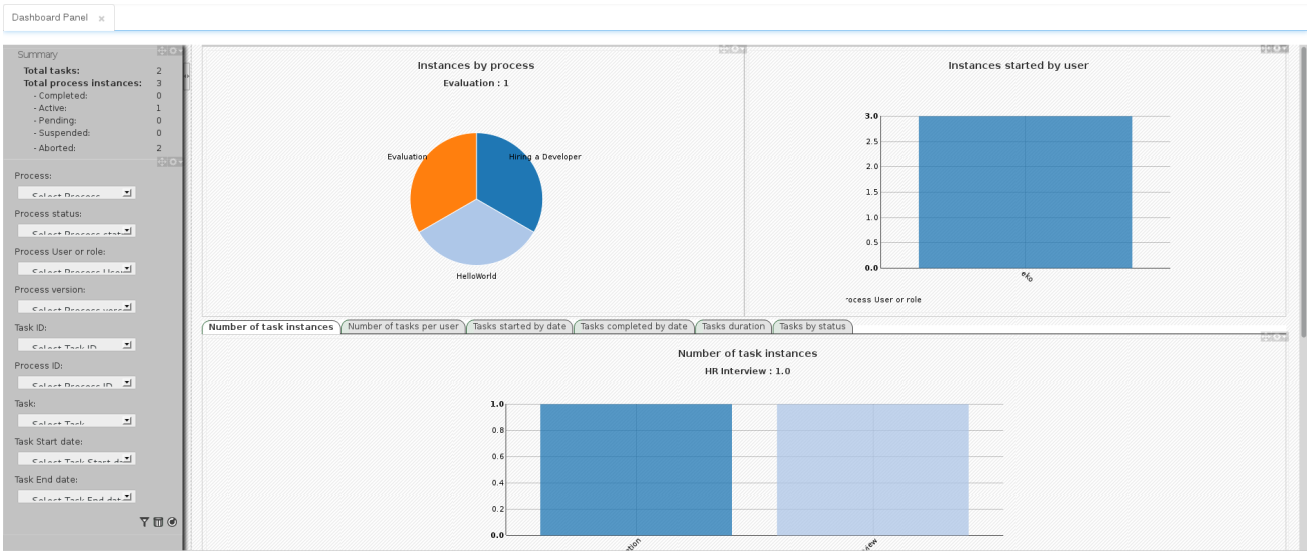


図8.1 プロセスおよびタスクダッシュボード

左側のパネルで、統計を表示するエンティティーを選択すると、右側の表とデーターが更新されます。

バグを報告する

第9章 RED HAT JBOSS DEVELOPER STUDIO

Red Hat JBoss Developer Studio は Eclipse を基にした JBoss 統合開発環境 (IDE) で、Red Hat カスタマーポータルの https://access.redhat.com から入手できます。JBoss Developer Studio は Red Hat JBoss BRMS および Red Hat JBoss BPM Suite 用のツールとインターフェースを持つプラグインを提供します。これらのプラグインはコミュニティーバージョンの製品が基になっています。そのため、BRMS プラグインは Drools プラグインと呼ばれ、BPM Suite プラグインは jBPM プラグインと呼ばれます。

インストールまたは設定の手順は、『Red Hat JBoss Developer Studio』ドキュメントを参照してください。

バグを報告する

9.1. JBOSS CENTRAL

JBoss Developer Studio 7.0 が先に起動している場合、JBoss Central はワークベンチのメインウィンドウに表示されます。**`Start from scratch`** 下にあるメニューオプションを選択すると、JBoss Central より新しいプロジェクトを作成できます。サンプルプロジェクトを起動するには、**`Start from a sample`** 下のリンクを選択します。

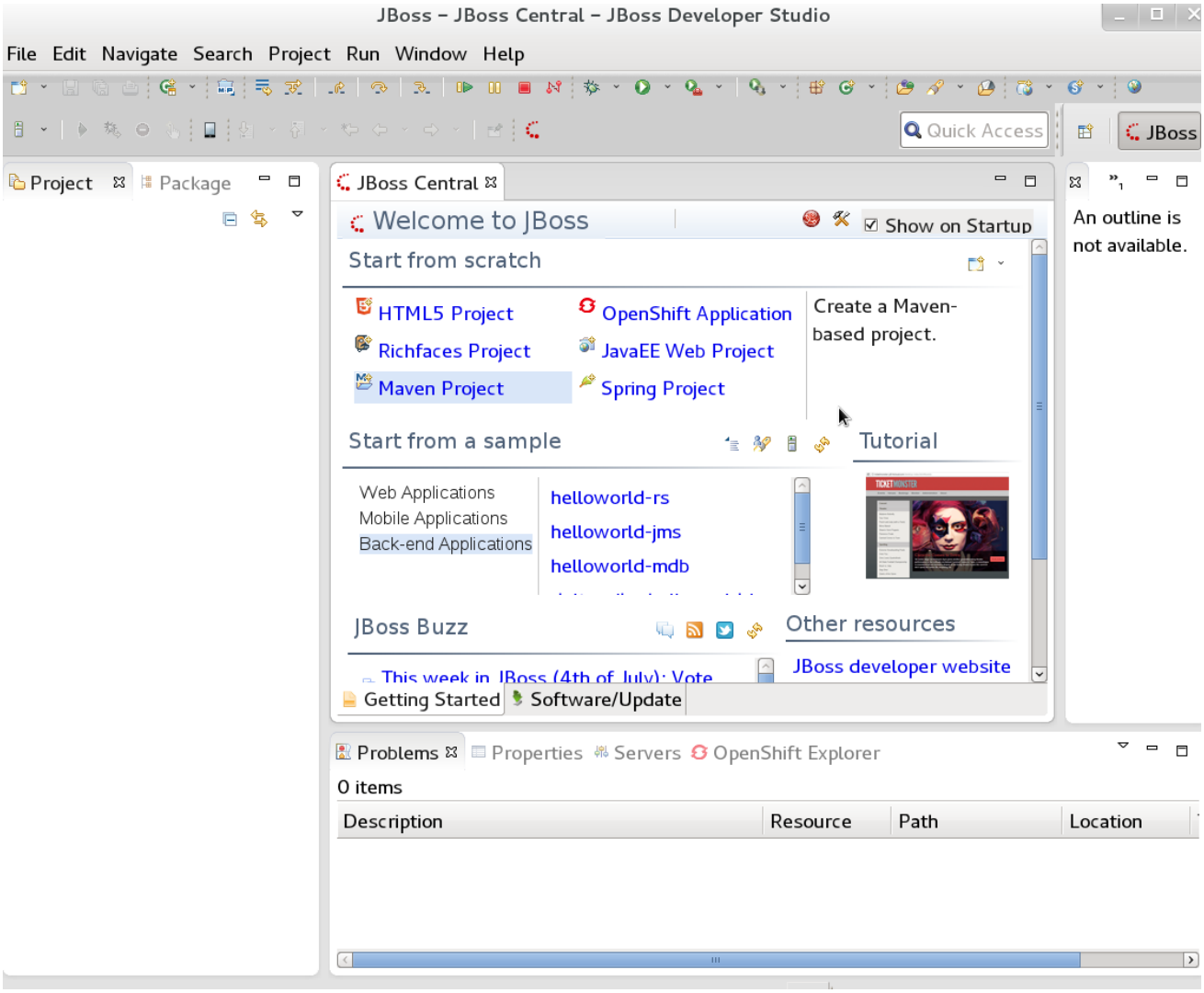


図9.1 JBoss Central

バグを報告する

9.2. JBOSS DEVELOPER STUDIO プラグインのインストール

JBoss Developer Studio の Drools および jBPM プラグインは更新サイトより入手できます。

手順**9.1 Drools** および **jBPM JBoss Developer Studio** プラグインのインストール

1. JBoss Developer Studio を起動します。
2. **Help** → **Install New Software** と選択します。
3. **Add** をクリックして **Add Repository** メニューを入力します。
4. **Name** フィールドのソフトウェアサイトの名前を指定し、**Location** フィールドに URL https://devstudio.jboss.com/updates/7.0/integration-stack/ を追加します。
5. **OK** をクリックします。
6. 利用可能なオプションより **JBoss Business Process and Rule Development** 機能を選択して **Next** をクリックし、再度 **Next** をクリックします。
7. ライセンスの内容を確認し、該当のラジオボタンを押して内容に同意し、**Finish** をクリックします。
8. プラグインのインストールが完了してから、JBoss Developer Studio を再起動します。

バグを報告する

9.3. DROOLS ランタイムの設定

Red Hat JBoss Developer Studio で JBoss BRMS プラグインを使用するには、ランタイムを設定する必要があります。

ランタイムは、このソフトウェアの特定のリリースを示す jar ファイルを集めたもので、ビジネス資産のコンパイルおよび実行に必要なライブラリを提供します。

JBoss BRMS の汎用デプロイ可能 zip アーカイブ (EAP 6 のデプロイ可能 zip アーカイブではありません) の jboss-brms-engine.zip アーカイブにあるランタイム jar ファイルを展開します (Red Hat カスタマーポータル より入手可能です)。

手順**9.2 BRMS** ランタイムの設定

1. JBoss Developer Studio メニューより **Window** を選択し、**Preferences** をクリックします。
2. **Drools** → **Installed Drools Runtimes** と選択します。
3. **Add...** をクリックし、新しいランタイムの名前を指定します。**Browse** をクリックしてランタイムが存在するディレクトリーを選択します。OK をクリックし、選択したランタイムを JBDS に登録します。
4. 横にあるチェックボックスをクリックして、作成したランタイムをデフォルトの Drools ランタイムとして指定します。
5. **OK** をクリックします。既存のプロジェクトがある場合は、JBDS を再起動してランタイムの更新を行う必要があると、ダイアログボックスに表示されます。

バグを報告する

9.4. JBPM ランタイムの設定

Red Hat JBoss Developer Studio で jBPM プラグインを使用するには、ランタイムを設定する必要があります。

ランタイムは、このソフトウェアの特定のリリースを示す jar ファイルを集めたものです。

JBoss BPM Suite の汎用デプロイ可能 zip アーカイブを Red Hat カスタマーポータル からダウンロードした場合、ランタイムを構成する jar ファイルは **jboss-bpms-engine.zip** アーカイブにあります。

手順**9.3 jBPM** ランタイムの設定

1. JBoss Developer Studio メニューより **Window** を選択し、**Preferences** をクリックします。
2. **jBPM** → **Installed jBPM Runtimes** と選択します。
3. **Add...** をクリックし、新しいランタイムの名前を指定します。**Browse** をクリックしてランタイムが存在するディレクトリーを選択します。
4. **OK** をクリックし、新しいランタイムを選択してから再度**OK** をクリックします。既存のプロジェクトがある場合は、JBDS を再起動してランタイムの更新を行う必要があると、ダイアログボックスに表示されます。

バグを報告する

9.5. JBOSS BPM SUITE サーバーの設定

Red Hat JBoss BPM Suite サーバーを実行するよう、JBoss Developer Studio を設定できます。

手順**9.4** サーバーの設定

1. jBPM ビューを開くには **Window** → **Open Perspective** → **Other** の順に選択して、**jBPM** を選択し、**OK** をクリックします。
2. **Window** → **Show View** → **Other...** の順に選択して **Server** → **Servers** と選択し、サーバービューを追加します。
3. サーバーパネルを右クリックしサーバーメニューを開き、**New** → **Server** を選択します。
4. **JBoss Enterprise Middleware** → **JBoss Enterprise Application Platform 6.1+** と選択してサーバーを定義し、**Next** をクリックします。
5. **Browse** をクリックし、ホームディレクトリーを設定します。JBoss BPM Suite がインストールされた JBoss EAP 6.1.1 のインストールディレクトリーを選択します。
6. **Name** フィールドにサーバー名を指定します。設定ファイルが設定されていることを確認してから **Finish** をクリックします。

バグを報告する

9.6. GIT リポジトリーから JBOSS DEVELOPER STUDIO へのプロジェクトのインポート

中心の Git 資産リポジトリーへ接続するよう、JBoss Developer Studio を設定できます。リポジトリーにはルール、モデル、関数、およびプロセスのバージョンが保存されます。この Git リポジトリーはすでに KIE Workbench によって定義されている必要があります。

ローカルの Git リポジトリーをインポートするか、リモート Git リポジトリーをクローンする必要があります。

手順**9.5** ローカル **Git** リポジトリーのインポート

1. Red Hat JBoss BPM Suite サーバーが稼働していない場合は、サーバータブよりサーバーを選択し、Start アイコンをクリックして起動します。
2. **File** → **Import...** と選択し、Git フォルダーを選択します。Git フォルダーを開き、**Projects from Git** を選択して Next をクリックします。
3. リポジトリーソースとして **Existing local repository** を選択し、Next をクリックします。

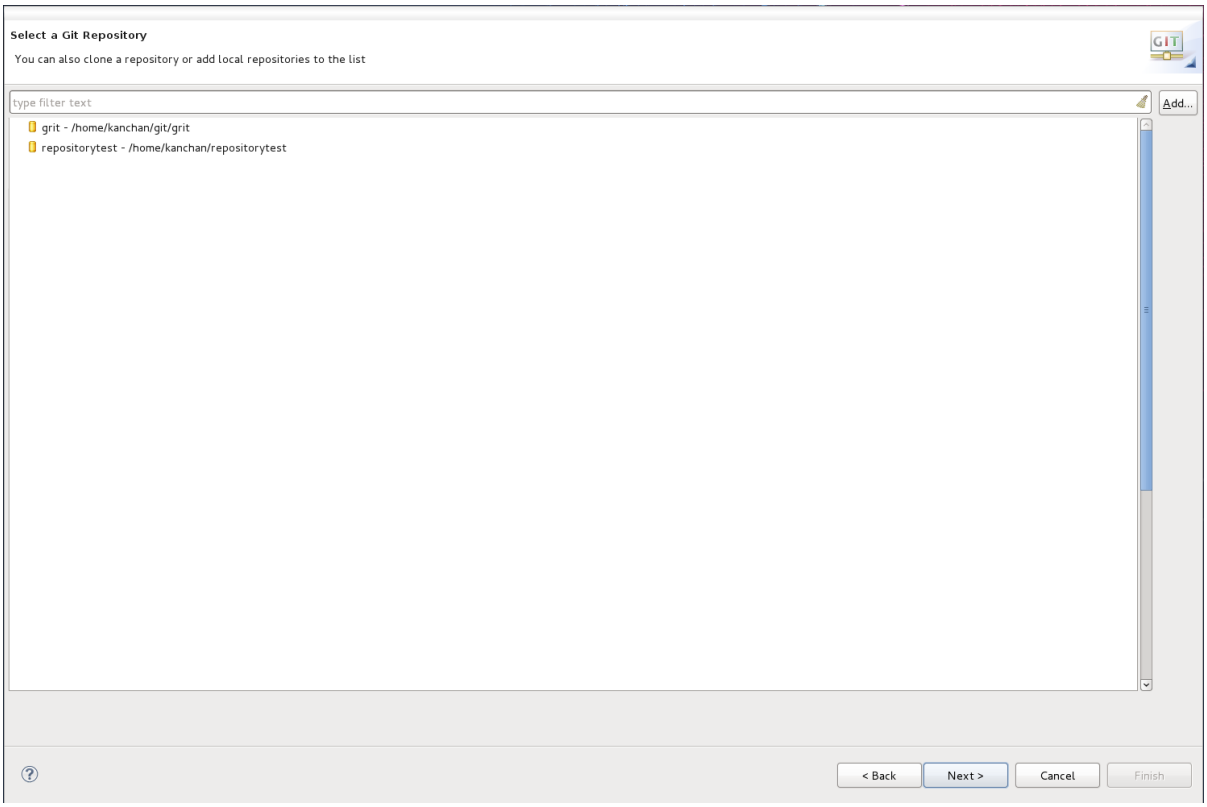


図**9.2 Git** リポジトリーの詳細

4. 使用できるリポジトリーのリストから設定するリポジトリーを選択します。
5. 次のウィンドウでプロジェクトを一般プロジェクトとしてインポートし、Next をクリックします。このプロジェクトに名前を付け、Finish をクリックします。

手順**9.6** リモート **Git** リポジトリーのクローン

1. Red Hat JBoss BPM Suite サーバーが稼働していない場合は、サーバータブよりサーバーを選択し、Start アイコンをクリックして起動します。
2. セキュアシェルサーバーが稼働していない場合は、以下のコマンドを使用して同時に起動します。コマンドは、Linux および Mac 固有のみになります。これらのプラットフォームで sshd がすでに起動していると、このコマンドに失敗しますが、無視しても問題ありません。

```
/sbin/service sshd start
```

3. **File → Import...** と選択し、Git フォルダーを選択します。Git フォルダーを開き、**Projects from Git** を選択して **Next** をクリックします。

4. **Clone URI** としてリポジトリーソースを選択し、**Next** をクリックします。

5. 次のウインドウに Git リポジトリーの詳細を入力し、**Next** をクリックします。

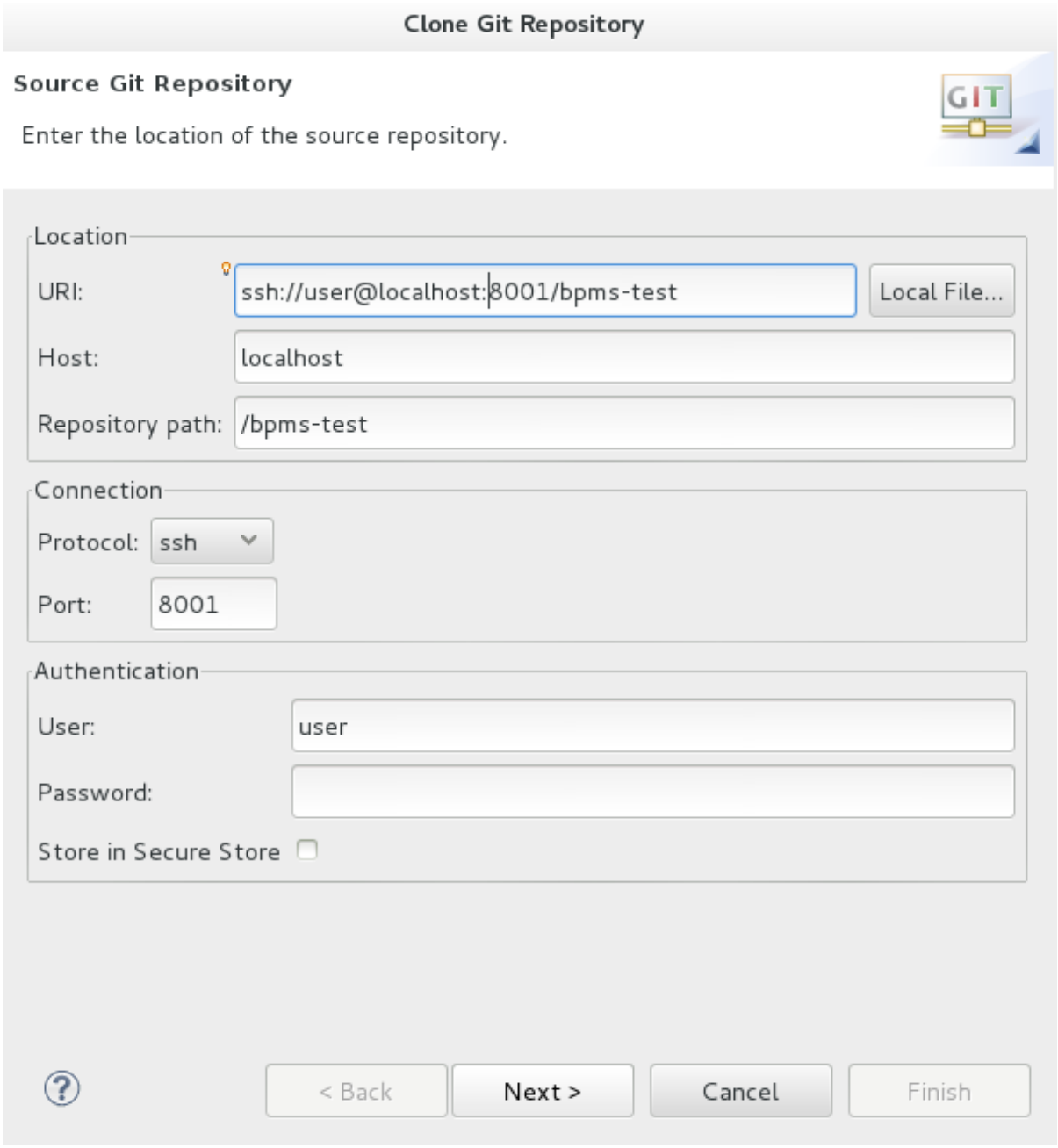


図9.3 Git リポジトリーの詳細

6. 次のウインドウでインポートするブランチを選択し、**Next** をクリックします。

7. このプロジェクトのローカルストレージを定義するオプションが表示されます。空でないディレクトリーを入力 (または選択) して設定の変更を行い、**Next** をクリックします。

8. 次のウィンドウでプロジェクトを一般プロジェクトとしてインポートし、**Next** をクリックします。このプロジェクトに名前を付け、**Finish** をクリックします。

バグを報告する

9.7. DROOLS プロジェクトの作成

手順9.7 新しい **Red Hat JBoss Developer Studio** プロジェクトの作成

1. メインメニューから **File** → **New** → **Project** を選択します。

   Drools → **Drools Project** と選択し、**Next** をクリックします。

2. プロジェクト名を **Project name:** テキストボックスに入力し、**Next** をクリックします。

   注記

   JBoss Developer Studio は、サンプル HelloWorld ルールファイルをプロジェクトに追加するオプションを提供します。**Next** をクリックしてこのデフォルトを許可し、以下の手順でサンプルプロジェクトをテストします。

3. Drools ランタイムを選択します (またはデフォルトを使用します)。

4. **Drools 6.0.x** と互換性のあるコードを選択します。**GroupID**、**ArtifactID**、および **Version** を入力し、**Finish** をクリックします。

5. プロジェクトをテストするには、主なメソッドを含む Java ファイルを右クリックして、**Run** → **run as** → **Java Application** の順に選択してください。

   Console タブに出力が表示されます。

バグを報告する

9.8. JBPM プロジェクトの作成

手順9.8 新しい **Red Hat JBoss Developer Studio** プロジェクトの作成

1. メインメニューから **File** → **New** → **Project** を選択します。

   jBPM → **jBPM Project** と選択し、**Next** をクリックします。

2. プロジェクト名を **Project name:** テキストボックスに入力し、**Next** をクリックします。

   注記

   JBoss Developer Studio は、サンプル HelloWorld ルールファイルをプロジェクトに追加するオプションを提供します。**Next** をクリックしてこのデフォルトを許可し、以下の手順でサンプルプロジェクトをテストします。

3. jBPM ランタイムを選択します (またはデフォルトを使用します)。

4. **Generate code compatible with jBPM 6 or above** を選択し、**Finish** をクリックします。

5. プロジェクトをテストするには、主なメソッドを含む Java ファイルを右クリックして、**Run → run as → Java Application** の順に選択してください。

   Console タブに出力が表示されます。

バグを報告する

第10章 BUSINESS RESOURCE PLANNER

Business Resource Planner は、Red Hat JBoss BPM Suite では技術プレビューとして使用できます。Business Resource Planner はライトウェイトの組み込み可能なプランニングエンジンで、プランニングの問題を最適化します。通常の Java ™ プログラマーがプランニングの問題を効率的に解決できるようにし、最適化ヒューリスティックおよびメタヒューリスティックと、大変効率的なスコア計算を組み合わせます。

プランナーは、次のようなさまざまなユースケースの解決に便利です。

- 従業員/患者の勤務表: プランナーを使用して看護婦の勤務シフトを作成でき、患者のベッド管理を追跡できます。
- 学校の時間割: プランナーは、授業、コース、試験、および会議のプレゼンテーションの計画を容易にします。
- 工場の計画: プランナーは、自動車の組み立てライン、機械の待機計画、および作業計画を追跡します。
- 在庫の削減: プランナーを使用すると、紙や金属などの資源の消費を削減し、無駄を最小限にすることができます。

バグを報告する

10.1. BUSINESS RESOURCE PLANNER のインストール

1. Red Hat カスタマーポータル にアクセスし、ユーザークレデンシャルを使用してログインします。
2. **Downloads** → **Red Hat JBoss Middleware** → **Download Software** の順に選択します。
3. **Products** ドロップダウンメニューより **BPM Suite** を選択します。
4. **Version** ドロップダウンメニューより、製品バージョン **6.0.2** を選択します。
5. **Red Hat JBoss BPM Suite 6.0.2 Business Resource Planner** を選択し、**Download** をクリックします。

バグを報告する

10.2. BUSINESS RESOURCE PLANNER サンプルの実行

1. コマンドラインで **examples/** ディレクトリーに移動します。
2. Unix 環境では、以下のコマンドを実行します。

```
./runExamples.sh
```

Windows 環境では、以下のコマンドを実行します。

```
./runExamples.bat
```

3. 開かれた GUI アプリケーションサンプルよりサンプルを 1 つ選択し、任意の IDE で実行します。

バグを報告する

付録A 改訂履歴

改訂 1.0.0-1	**Tue Aug 05 2014**	**CS Builder Robot**

Built from Content Specification: 22690, Revision: 684299