Pentaho BA Server Manual Deployment Guide

Help and Support Resources

If you have questions that are not covered in this guide, or if you would like to report errors in the documentation, please contact your Pentaho technical support representative.

Support-related questions should be submitted through the Pentaho Customer Support Portal at http://support.pentaho.com.

For information about how to purchase support or enable an additional named support contact, please contact your sales representative, or send an email to sales@pentaho.com.

For information about instructor-led training on the topics covered in this guide, visit http://www.pentaho.com/training.

Limits of Liability and Disclaimer of Warranty

The author(s) of this document have used their best efforts in preparing the content and the programs contained in it. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, express or implied, with regard to these programs or the documentation contained in this book.

The author(s) and Pentaho shall not be liable in the event of incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of the programs, associated instructions, and/or claims.

Trademarks

Pentaho (TM) and the Pentaho logo are registered trademarks of Pentaho Corporation. All other trademarks are the property of their respective owners. Trademarked names may appear throughout this document. Rather than list the names and entities that own the trademarks or insert a trademark symbol with each mention of the trademarked name, Pentaho states that it is using the names for editorial purposes only and to the benefit of the trademark owner, with no intention of infringing upon that trademark.

Company Information

Pentaho Corporation
Citadel International, Suite 340
5950 Hazeltine National Drive
Orlando, FL 32822
Phone: +1 407 812-OPEN (6736)
Fax: +1 407 517-4575
http://www.pentaho.com

E-mail: communityconnection@pentaho.com

Sales Inquiries: sales@pentaho.com

Documentation Suggestions: documentation@pentaho.com

Sign-up for our newsletter: http://community.pentaho.com/newsletter/

Contents

Manual Deployment Overview

Pentaho provides several installation paths to address a variety of computing environment requirements. These instructions show you how to install each Pentaho product with the specific databases and software you already have or prefer to supply for yourself. Typically, the Business Analytics Server and Pentaho Enterprise Console will go on the server, and each workstation will have its own copy of the plugins.

Use this guide if you plan to...

- Use your own application server
- Implement single sign-on through a Central Authentication Service

- Have your own or are creating your own deployment scripts
- Are using a command line interface or are performing a headless installation
- Are performing a remote installation
- Plan to connect to a database that is not the default installation database in the installation wizard
- Are independently upgrading or migrating to a new version of a particular product or plugin
- Need more fine-grained control over configuration and deployment options for servers

Use the *Archive-based Installion Guide*.

Or, if you prefer to have all of this functionality ...

- Install using a graphical installer
- Install quickly and/or for evaluation purposes
- Use the default PostgreSQL database
- Use the Tomcat application server

Use the *Pentaho Business Analytics Graphical Installer Guide*.

Components

This table shows the components included in the manual deployment and the databases and software you must supply on your own.

Can Be Installed as Part of This Process	You Must Supply
A JBoss- or Tomcat-compatible J2EE application archive that contains the Pentaho BA Server **Archive packages containing BA Server plugins:** - Analyzer - Dashboard Designer - Interactive Reporting - Mobile **A Jetty application server running the Pentaho Enterprise Console**	**A supported operating system:** - Linux - Windows - Solaris - Mac OS **Build tools:** - A Sun Java Development Kit (JDK) - Apache Ant **A Sun JRE** on the server, and on each workstation that you will install data preparation or design tools to. **A supported application server:** - Tomcat - JBoss **A supported database to store solutions:** - MySQL - PostgreSQL - Oracle **One or more data sources:** - Any JDBC-compliant database - A spreadsheet

Can Be Installed as Part of This Process	You Must Supply
	• A flat file containing comma-separated values

Installation Checklist

The Installation Checklist is a concise list of instructions intended to show a high-level overview of the manual build and deployment process. It also serves as a quick reference for administrators and developers who have performed several installations in the past and only need a brief rehash of the necessary steps. **This is not the complete instruction set.** If you need more details than are provided in this checklist, consult the appropriate section in the verbose instruction set that comprises the rest of this guide.

Step	Procedure	Done
Step 1	On Linux and other Unix-like operating systems, create a **pentaho** user account with a home directory and a password.	
Step 2	Establish a **PENTAHO_JAVA_HOME** system variable that points to a Sun JRE or JDK version 1.6.0.	
Step 3	Establish a **PENTAHO_INSTALLED_LICENSE_PATH** system variable that points to `/home/pentaho/.installedLicenses.xml` or wherever you intend to install licenses to.	
Step 4	Download the Pentaho Enterprise Console, BA Server J2EE "manual deployment" build package, and any BA Server plugins (Analyzer, Dashboard Designer, Interactive Reporting, etc.) from the Pentaho Customer Support Portal. If you need to access PDI content stored in a PDI Enterprise Repository, you will also need to download a PDI client tool package.	
Step 5	Unpack the build materials to an appropriate temporary location; this does not have to be the same location that you plan to deploy the Web application to.	
Step 6	Prepare Hibernate and Quartz data source configuration files for your application server according to the details shown below. The default instructions for doing this are: Create either a **context.xml** file for Tomcat, or **PentahoHibernate-ds.xml** and **quartz-ds.xml** files for JBoss deployments, and change the data source details so that they match the solution repository and schedule database that you will initialize later.	
Step 7	If you intend to change the Web application context name from **pentaho** to something else, modify **web.xml** and **jboss-web.xml** to change the application name at build time.	
Step 8	Establish a `/pentaho/server/biserver-ee/` directory in an accessible location (your new **pentaho** user's home directory, if you are on Linux or Solaris, and anywhere that is secure and convenient if you are on Windows).	
Step 9	If you have a Pentaho Analysis Enterprise Edition license, unpack the pentaho-analysis-ee archive to a temporary location, then copy the requisite JARs to the `/biserver-manual-ee/build-resources/custom-pentaho-webapp/WEB-INF/lib/` directory, and the requisite configuration files to the `/biserver-manual-ee/build-resources/custom-pentaho-webapp/WEB-INF/classes/` directory.	
Step 10	If you intend to use Oracle for your solution repository, copy the Oracle Quartz JAR to `/biserver-manual-ee/build-resources/custom-pentaho-webapp/WEB-INF/lib/`.	
Step 11	Using Apache Ant, build the Pentaho WAR with the appropriate options for your environment. ```ant -Darchive.target=war-pentaho-jboss -Ddb=oracle10g```	
Step 12	Copy the solution database JDBC driver to the `/tomcat/lib/` directory for Tomcat, or the `/jboss/server/default/lib/` directory for JBoss.	
Step 13	Move the **pentaho-solutions** directory from `/biserver-manual-ee/build/` to `/pentaho/server/biserver-ee/`	

Step	Procedure	Done
Step 14	Using the SQL scripts for your database of choice in `/biserver-manual-ee/build-resources/pentaho-data/`, initialize your Pentaho solution repository (hibernate) and schedule database (quartz).	
Step 15	Configure your application server to connect to your solution repository by appropriately modifying the following files in the `/pentaho/server/biserver-ee/pentaho-solutions/system/` directory: • applicationContext-spring-security-hibernate.properties • /quartz/quartz.properties And one of the following: • /hibernate/postgresql.hibernate.cfg.xml • /hibernate/mysql5.hibernate.cfg.xml • /hibernate/oracle10g.hibernate.cfg.xml	
Step 16	Copy the Pentaho WAR to your application server's **deploy** or **webapps** directory.	
Step 17	Modify your application server's configuration files appropriately to accommodate Pentaho's Java settings requirements, and copy over any database driver JARs that you need for your data sources.	
Step 18	If you need to access PDI content stored in a Pentaho Enterprise Repository, edit the settings.xml file, copy over the repositories.xml file, unpack the PDI client tool package to a temporary location, then copy the contents of `/data-integration/plugins/` to the `/pentaho/server/biserver-ee/pentaho-solutions/system/kettle/plugins/` directory.	
Step 19	Unpack any BA Server plugin packages that you have licenses for to the `/pentaho-solutions/system/` directory.	
Step 20	For Linux only, configure dashboard printing.	
Step 21	Unpack the Pentaho Enterprise Console archive package to the `/pentaho/server/` directory.	
Step 22	Start the application server you deployed the BA Server WAR to, and the Pentaho Enterprise Console (you must be logged into your new **pentaho** user account to start the Pentaho Enterprise Console server).	
Step 23	Install license keys; establish users, roles, data sources; and perform any remaining configuration tasks. Use the *Pentaho Business Analytics Administrator's Guide* for assistance.	
Step 24	Access the BA Server from a workstation and verify that you can log in as a normal user, and create reports and analysis views.	

Prerequisites

In order to perform a manual deployment, you must be familiar with technical concepts.

- You must be familiar the web server, Java application server, database, and firewall configuration for your organization's network.
- You need to know the IP addresses of each relevant server, and which ports all of the necessary services use
- Building the BA Server from the command line involves using Apache Ant, so you must have a basic familiarity with using Ant from the command line, and with modifying XML-based build scripts.
- You or a system administrator must have the ability to install software, create and delete directories, open firewall ports, and start and stop system services on the machine you are installing on.
- If you need to access a remote database, make sure you have the level of permission necessary to do so from both your test machine and the production server.

Local User Accounts on Linux

If you create a new local user account called **pentaho** as described in *Creating a Pentaho System User (Linux/Solaris)* , you must put the pentaho-solutions directory and store license information recorded by the Pentaho Enterprise Console in that directory. You must also adjust your `init` scripts to start your Web application server (which will run the BA Server) and the Pentaho Enterprise Console as this user. If you are unable to do this, you must modify the instructions in the rest of this guide to support your own custom user and directory configuration.

If you do not run the Pentaho Enterprise Console and the application server that runs the BA Server as the same system user, the BA Server will not be able to find the appropriate license information and will not operate with full functionality. If you need to launch the Pentaho Enterprise Console and the BA Server as separate users or services, you will have to set a **-D** parameter in your web application server's service configuration or startup scripts to specify a static location for your Pentaho license file. The Java parameter is **-Dpentaho.installed.licenses.file=/pentaho/.installedLicenses.xml**, though you will need to change this example to match your configuration.

Creating a Pentaho System User on Linux

You may be able to use a different local user account, however you will have to modify all of the instructions to match that configuration.

Pentaho licenses are installed to an XML file in the home directory of the user that starts the Pentaho Enterprise Console. While you can use any system user to install licenses, it is easier to create a new user to start and stop the Pentaho Enterprise Console, and to install and update licenses with.

1. Open a local terminal on, or an OpenSSH session to the server you are hosting the BA Server on.

   ```
   ssh pgibbons@192.168.1.133
   ```

2. With root permissions, create a new user account called **pentaho**.

   Bash is not a requirement, but it is typically the shell that Linux users want to standardize on. On many Linux distributions, the default new user shell is /bin/sh or some equivalent (such as Dash) that may not use the ~/.bashrc configuration file that you will work with later. If you don't have or want to use Bash, adjust the instructions throughout this guide accordingly.

   ```
   sudo useradd -s /bin/bash -m pentaho
   ```

3. Set a password for the pentaho user (note that when using the **sudo** command, you must first supply the root password).

   ```
   sudo passwd pentaho
   ```

4. Verify that you can log in using the credentials you specified.

   ```
   su pentaho -
   ```

You now have a user account created specifically for running the BA Server and controlling Pentaho Enterprise Console start and stop scripts. You should stay logged into this new account to create the pentaho directory and perform all other installation tasks that do not explicitly require root access.

You must use this new user account for starting and stopping the Pentaho Enterprise Console, and for installing and updating licenses if you use the command line tool to manage them instead of the graphical interface in Pentaho Enterprise Console. If you create any RC or init scripts to start Pentaho Enterprise Console automatically at boot time, then you will have to write those scripts such that they start the service with the **pentaho** user credentials.

How to Check Your Java Version

Pentaho Business Analytics requires a Java Runtime Environment (JRE) or Java Development Kit (JDK). Follow this procedure to see which version of Java is installed on your system and configured to be the default Java executable. There may be multiple JREs or JDKs on your system, but only one can be set as the global default. If a particular JRE or JDK is not specified by an application on startup, the default is used. Pentaho establishes a specific system variable named **PENTAHO_JAVA_HOME** to declare which Java instance it will use.

1. Open a terminal or command prompt window.
2. Type this command in: **java -version** and press **Enter**.

   Along with the Java version, the bit-ness, 32-bit or 64-bit, and patch level also shows in the output. For example ...

```
java version "1.6.0_21"
Java(TM) SE Runtime Environment (build 1.6.0_21-b06)
Java HotSpot(TM) 64-Bit Server VM (build 17.0-b16, mixed mode)
```

Setting the PENTAHO_JAVA_HOME Variable on Linux

To ensure that Business Analytics will always use the correct Sun Java Runtime Environment, especially in software environments that contain multiple JREs, you must create a **PENTAHO_JAVA_HOME** system variable for your **pentaho** user account and point it to a supported JRE or JDK. If you do not set this variable, Business Analytics will attempt to use the JRE that the JAVA_HOME variable points to. We do not recommend that your BA Server installation rely on the **JAVA_HOME** variable, as this is the global default JRE for the entire system, and changing it could adversely affect other Java-based applications in the system. Therefore, Pentaho recommends that you set PENTAHO_JAVA_HOME as described in these steps.

1. Edit your **/etc/environment** file with a text editor.

   If you're using Solaris, you will have to set this environment variable through whatever means are available to you.
2. Add this line in a convenient place (replacing the path with the location of the JRE on your system): **export PENTAHO_JAVA_HOME=/usr/lib/jvm/java-6-sun**.
3. You must log out and log back into the operating system for the change to take effect.
4. Verify that the variable is properly set.

```
env | grep PENTAHO_JAVA_HOME
```

Setting the PENTAHO_JAVA_HOME Variable on Windows

To ensure that Business Analytics will always use the correct Sun Java Runtime Environment, especially in software environments that contain multiple JREs, you must create a **PENTAHO_JAVA_HOME** system variable for your **pentaho** user account and point it to a supported JRE or JDK. If you do not set this variable, Business Analytics will attempt to use the JRE that the JAVA_HOME variable points to. We do not recommend that your BA Server installation rely on the **JAVA_HOME** variable, as this is the global default JRE for the entire system, and changing it could adversely affect other Java-based applications in the system. Therefore, Pentaho recommends that you set PENTAHO_JAVA_HOME as described in these steps.

1. In **Windows**, right-click on **Computer**, then select **Properties** from context menu, then click **Advanced System Settings**.

   The **System Properties** window will come up.
2. In the System Properties window, click the **Advanced** tab, then click **Environment Variables**.
3. In the System Variable section, click New.
4. A popup dialog will ask for a variable name and value. Type **PENTAHO_JAVA_HOME** into the name field.

5. In the value field, enter the directory for the JRE. An example of a JRE directory is `C:\Program Files\Java\jre6`. Click **OK**.

6. In the parent window, click **Apply Changes**.

7. You must restart your computer for the change to take effect.

8. Verify that the variable is properly set.

```
echo %PENTAHO_JAVA_HOME%
```

Setting the PENTAHO_INSTALLED_LICENSE_PATH Variable on Linux

To ensure that the BA Server, DI Server, and Pentaho Enterprise Console all use the same location to store and retrieve Pentaho licenses, you must create a **PENTAHO_INSTALLED_LICENSE_PATH** system variable for your **pentaho** user account. If you do not set this variable, Enterprise Console and the command line license installation script will store license data in a place that the DI Server and BA Server will not look.

Ultimately it does not matter what you set this location to. However, it needs to be available to the user account(s) that runs Enterprise Console and the BA and DI Server.

1. Edit your **/etc/environment** file with a text editor.

   If you're using Solaris, you will have to set this environment variable through whatever means are available to you.

2. Add this line in a convenient place (changing the path as explained above, if necessary): **export PENTAHO_INSTALLED_LICENSE_PATH=/home/pentaho/.installedLicenses.xml**.

3. You must log out and log back into the operating system for the change to take effect.

4. Verify that the variable is properly set.

```
env | grep PENTAHO_INSTALLED_LICENSE_PATH
```

The PENTAHO_INSTALLED_LICENSE_PATH variable is now set.

Setting the PENTAHO_INSTALLED_LICENSE_PATH Variable on Windows

To ensure that the BA Server, DI Server, and Pentaho Enterprise Console all use the same location to store and retrieve Pentaho licenses, you must create a **PENTAHO_INSTALLED_LICENSE_PATH** system variable for your **pentaho** user account. If you do not set this variable, Enterprise Console and the command line license installation script will store license data in a place that the DI Server and BA Server will not look.

Ultimately it does not matter what you set this location to. However, it needs to be available to the user account(s) that runs Enterprise Console and the BA and DI Server.

1. In **Windows XP**, right-click on **My Computer**, then select **Properties** in the context menu. In **Windows 7**, right-click on **Computer**, then select **Properties** from context menu, then click **Advanced System Settings**.

   The **System Properties** window will come up.

2. In the System Properties window, click the **Advanced** tab, then click **Environment Variables**.

3. In the System Variable section, click New.

4. A popup dialog will ask for a variable name and value. Type **PENTAHO_INSTALLED_LICENSE_PATH** into the name field, and the directory you intend to install licenses to plus **.installedLicenses.xml** in the value field, then click **OK**.

```
C:\pentaho\.installedLicenses.xml
```

5. In the parent window, click **Apply Changes**.

6. You must restart your computer for the change to take effect.

7. Verify that the variable is properly set.

```
echo %PENTAHO_INSTALLED_LICENSE_PATH%
```

The PENTAHO_INSTALLED_LICENSE_PATH variable is now set.

If you run Tomcat automatically as a Windows service, you must also configure it to run on the Windows user account where the **.installedLicenses.xml** file is located.

Preparing a Headless Linux or Solaris Server

There are two headless server scenarios that require special procedures on Linux and Solaris systems. One is for a system that has no video card; the other is for a system that has a video card, but does not have an X server installed. In some situations -- particularly if your server doesn't have a video card -- you will have to perform both procedures in order to properly generate reports with the BA Server.

Systems without video cards

The **java.awt.headless** option enables systems without video output and/or human input hardware to execute operations that require them. To set this application server option when the BA Server starts, you will need to modify the startup script for either the BA Server, or your Java application server. Add the following item to the list of CATALINA_OPTS parameters: **-Djava.awt.headless=true**.

The entire line should look something like this:

```
export CATALINA_OPTS="-Djava.awt.headless=true -Xms256m -Xmx768m -XX:MaxPermSize=256m
  -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000"
```

If you intend to create a BA Server service control script, you must add this parameter to that script's CATALINA_OPTS line.

Note: If you do not have an X server installed, you must also follow the below instructions.

Systems without X11

In order to generate charts, the Pentaho Reporting engine requires functionality found in X11. If you are unwilling or unable to install an X server, you can install the **xvfb** package instead. xvfb provides X11 framebuffer emulation, which performs all graphical operations in memory instead of sending them to the screen.

Use your operating system's package manager to properly install xvfb.

Obtaining the Pentaho BA Server Build Materials

Enterprise Edition customers can obtain the BA Server build package from the Pentaho Customer Support Portal by using their Pentaho account login credentials. If you are unfamiliar with these details, consult the Welcome Kit provided to you by Pentaho customer support as part of the Enterprise Edition enablement process.

The only strictly required package is `biserver-manual-ee-4.8.0-GA.zip`. It is platform-agnostic, so there is only one download for all operating systems, databases, and application servers. Download and unpack this file to a directory approved by your system administrator. The location does not matter to the BA Server, but it should be in a directory that is not over- or under-restricted by the operating system. This directory and the files in it are only used for building the BA Server. Once the software has been configured and built, it will be deployed directly to your application server as a WAR archive.

Packages

The packages you will need for a Windows, Linux, Solaris and Mac OS servers are:

- **BA Server:** biserver-manual-ee-4.8.0-GA.zip
- **Dashboard Designer:** pdd-plugin-ee-4.8.0-GA.zip (if licensed)
- **Interactive Reporting:** pir-plugin-ee-1.1.0-GA.zip (if licensed)
- **Analyzer:** paz-plugin-ee-4.8.0-GA.zip (if licensed)
- **Pentaho Enterprise Console:** pec-4.8.0-GA.zip
- **Pentaho Mobile:**pentaho-mobile-plugin-ee-4.8.0-GA.zip (if licensed)

Building the BA Server

Building the Pentaho BA Server requires running Apache Ant from the command line interface, using special build options that shape the application according to your specifications. All of the necessary build options are listed in this section, followed by Ant build instructions.

In addition to specifying a database- and application server-specific build target, you will also be adding custom configuration files into the build. This protects you from having to make a lot of surgical changes after the Pentaho WAR and solutions directory have been deployed, as is the case with an "unpack and modify" archive-based installation.

Establishing a context.xml File For Tomcat

This procedure is only for Tomcat users. If you are on any other Web application server, skip these instructions.

Web application configuration options in Tomcat, including data sources, should go into a **context.xml** file inside of the WAR archive. It is possible to add this file to the META-INF directory after the WAR is deployed, but it's easier to add it into the WAR during the build. Follow the directions below to accomplish this.

1. Consult your database documentation to determine the JDBC class name and connection string for your Pentaho solution database.
2. Navigate to the `/biserver-manual-ee/build-resources/custom-pentaho-webapp/META-INF/` directory.
3. Create a file in this directory called **context.xml**, and paste the following text into it, replacing the **Resource name**, **docbase**, **username**, **password**, **driverClassName**, and **url** parameters (or any relevant connection settings) to match your solution repository configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<Context path="/pentaho" docbase="webapps/pentaho/">
    <Resource name="jdbc/Hibernate" auth="Container" type="javax.sql.DataSource"
        factory="org.apache.commons.dbcp.BasicDataSourceFactory" maxActive="20"
 maxIdle="5"
        maxWait="10000" username="hibuser" password="password"
        driverClassName="com.mysql.jdbc.Driver" url="jdbc:postgresql://
localhost:5432/hibernate"
        validationQuery="/* ping */ select 1"/>

    <Resource name="jdbc/Quartz" auth="Container" type="javax.sql.DataSource"
        factory="org.apache.commons.dbcp.BasicDataSourceFactory" maxActive="20"
 maxIdle="5"
        maxWait="10000" username="pentaho_user" password="password"
        driverClassName="com.postgresql.jdbc.Driver" url="jdbc:postgresql://
localhost:5432/quartz"
        validationQuery="/* ping */ select 1"/>
</Context>
```

4. If you are using Oracle for your solution repository, replace the value of **validationQuery** in the above XML snippet with **"select 1 from dual"**.

```
validationQuery="select 1 from dual"
```

5. If you are using PostgreSQL or MySQL for your solution repository, replace the value of **validationQuery** in the above XML snippet with **"select 1"**.

```
validationQuery="select 1"
```

6. Save the file and close your text editor.

You can now continue with the standard WAR build process.

Solution Repository JNDI Configuration Files for JBoss Deployments

This procedure is only for JBoss users. Skip this step if you are building and deploying WAR files for Tomcat.

Web application configuration options in JBoss packages, including data sources, should go into data source XML configuration files that will later be pulled into the WAR archive. Follow the directions below to accomplish this.

1. Consult your database documentation to determine the JDBC class name and connection string for your Pentaho solution database.

2. Copy the database-specific **PentahoHibernate-ds.xml** and **quartz-ds.xml** files from `/biserver-manual-ee/build-resources/pentaho-res/jboss/datasources/` to the `/biserver-manual-ee/build-resources/custom-pentaho-webapp/META-INF/` directory.

3. Edit the two files you just copied and replace the **Resource name**, **driverClassName**, and **url** parameters (or any relevant connection settings) to match your solution repository configuration.

4. Copy the two modified data source config files to the `/server/default/deploy/` directory in your JBoss installation.

5. Make the same changes to the following files (replacing **database** in the hibernate configuration file with mysql5, postgresql, or oracle10g depending on which database you are using):

   - `/biserver-manual-ee/build-resources/pentaho-solutions/system/hibernate/ <database>.hibernate.cfg.xml`
   - `/biserver-manual-ee/build-resources/pentaho-solutions/system/applicationContext-spring-security-hibernate.properties`
   - `/biserver-manual-ee/build-resources/pentaho-solutions/system/applicationContext-spring-security-jdbc.xml`

You can now continue with the standard JBoss build process.

Changing the Web Application Context

The Pentaho BA Server's default port number is **8080**, and the default application name is **pentaho**, which is the name of the WAR archive, the name of the directory that your application server will create, and also part of the URL structure for all content in the Pentaho User Console. If you need to change the application name to something else, or if your application server is running on a port other than 8080, follow the below processes for either JBoss or Tomcat.

Changing the Web Application Name on Tomcat

These instructions will only work on Tomcat servers that are configured to accept context.xml overrides built into deployed WARs. Some Tomcat deployments may not have this feature turned on. You can change the Tomcat configuration on your own, or consult your Tomcat documentation to learn about other methods of changing a web application context; use the XML snippet below in whichever configuration file you end up creating or modifying.

Follow the below instructions to change the web application context for a Pentaho WAR that will be deployed to a Tomcat server.

1. Navigate to the `/biserver-manual-ee/build-resources/custom-pentaho-webapp/META-INF/` directory.

2. Edit the **context.xml** in this directory, and change the **pentaho** references in the **Context path** tag to your preferred context name:

```
<Context path="/pentaho" docbase="webapps/pentaho/">
```

3. Save and close the file, then change to the `/biserver-manual-ee/build-resources/pentaho-webapp/ WEB-INF/` directory.

4. Edit the **web.xml** file found there, and change the **fully-qualified-server-url** entry to match the context name you specified previously, and your Tomcat port number.

```
<context-param>
    <param-name>fully-qualified-server-url</param-name>
    <param-value>http://localhost:5150/example/</param-value>
</context-param>
```

You can now continue with the standard WAR build process. Once the WAR is built, you must rename it to whatever you just set the context name to.

Changing the Web Application Name on JBoss

Follow the below instructions to change the Web application context for a Pentaho WAR that will be deployed to a JBoss server.

1. Navigate to the `/biserver-manual-ee/build-resources/pentaho-webapp/WEB-INF/` directory.
2. Edit the **jboss-web.xml** file and add the following text inside of the **jboss-web** section, replacing **example** with your new context name:

```
<context-root>example</context-root>
```

3. Save and close the file, then edit **web.xml** and modify the **fully-qualified-server-url** entry to match your new context name and port number.

```
<context-param>
    <param-name>fully-qualified-server-url</param-name>
    <param-value>http://localhost:5150/example/</param-value>
</context-param>
```

4. Save and close the file

You can now continue with the standard WAR build process. Once the WAR is built, you must rename it to whatever you changed the context name to.

Customizing Port Numbers

If you want to customize port numbers, follow the below procedure.

1. Navigate to the `/pentaho/server/biserver-ee/tomcat/conf/` directory
2. Edit the **server.xml** file found there, and search for **Define a non-SSL HTTP/1.1 Connector**. Change the port number in the **Connector port** element below that from **8080** to your preferred port number.

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
    <Connector port="8080" maxHttpHeaderSize="8192"
               maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
               enableLookups="false" redirectPort="8443" acceptCount="100"
               connectionTimeout="20000" disableUploadTimeout="true" />
```

3. Save and close the file, then navigate to the `/pentaho/server/biserver-ee/tomcat/webapps/pentaho/WEB-INF/` directory and edit the **web.xml** file found there.
4. Change the **fully-qualified-server-url** entry to match the new port number you specified in server.xml.

```
<context-param>
    <param-name>fully-qualified-server-url</param-name>
    <param-value>http://localhost:8080/pentaho/</param-value>
</context-param>
```

5. Save and close the file, then navigate to the `/pentaho/server/enterprise-console/resource/config/` and edit the **console.properties** file found there.
6. Change the **console.start.port.number** and **console.stop.port.number** values to match the port numbers you want to run Enterprise Console on.

```
console.start.port.number=8088
console.stop.port.number=8033
```

Creating a Pentaho Directory

The Pentaho Web application expects a single, canonical location for your solutions directory. Though you can put the BA Server in any directory on your system -- access and write permissions aside -- it will be easier for you to follow all of Pentaho's installation and administration documentation if you establish the recommended directory structure below.

1. Navigate to the top-level directory where you want your solutions directory to reside. This must be accessible to the application server and writable by the user account that runs the application server.

   ```
   cd /home/pentaho/
   ```

2. Create a **pentaho/server/biserver-ee/** directory.

   ```
   mkdir -p pentaho/server/biserver-ee/
   ```

   This directory structure is identical to the one laid down by the Pentaho Business Analytics installation utility. Even though your application server, Pentaho client tools, and other files are not stored here, replicating the exact BA Server directory structure makes it easier to troubleshoot problems and follow instructions in documentation.

   You now have a directory in which to install Pentaho Enterprise Console and the Pentaho solutions directory, where all of your reports, analysis views, dashboards, and rendered content will be stored.

Installing the Analysis Enterprise Edition Plugin

Follow the instructions below to install the Pentaho Analysis Enterprise Edition package, which contains Analysis engine enhancements for large ROLAP deployments. This procedure does not cover Pentaho Analyzer installation.

☞ **Note:** If you are performing a manual WAR build and deployment, and you want to add the Pentaho Analysis Enterprise Edition JARs into the WAR pre-build, you can substitute `/biserver-manual-ee/build-resources/custom-pentaho-webapp/WEB-INF/` for `/WEB-INF/` references below.

1. If you have not already done so, retrieve the **pentaho-analysis-ee-4.8-GA** package from the Pentaho Customer Support Portal (login credentials were emailed to you with your Welcome Kit).
2. Unpack the archive to a temporary location.
3. If it is currently running, shut down the BA Server.
4. Copy only the following JARs from the `/pentaho-analysis-ee/lib/` directory to the `/tomcat/webapps/pentaho/WEB-INF/lib/` directory.

   - infinispan-core- 4.2.1.FINAL
   - jboss-transaction-api-1.0.1.GA
   - jcip-annotations-1.0
   - jgroups-2.12.0.CR5
   - marshalling-api-1.2.3.GA
   - memcached-0.0.1-PENTAHO
   - pentaho-analysis-ee-3.4.0-GA-obf
   - river-1.2.3.GA

5. Copy all of the configuration files from `/pentaho-analysis-ee/config/` to the `/tomcat/webapps/pentaho/WEB-INF/classes/ directory`.
6. Depending on the installation type; there would not be a **pentaho.war** (in archive based installations and executable based installations the **pentaho.war** is already deployed and the application will show as: `/tomcat/webapps/pentaho/`.
7. Remove the temporary **pentaho-analysis-ee** directory.

Pentaho Analysis Enterprise Edition is now installed with the default Infinispan configuration.

If you need to switch from Infinispan to Memcached, continue to the next section.

Switching to Memcached

In order to complete this procedure, you must have your own pre-configured Memcached instance. You should have also installed the Analysis Enterprise Edition package to your BA Server or standalone Mondrian engine.

If you already use the Memcached cache framework in your organization and would like to hook it up to the Pentaho Analysis ROLAP engine, follow the directions below to switch from the default Infinispan cache framework configuration.

⚠ **Caution:** Pentaho and Mondrian developers recommend against using Memcached. You are almost certain to have better performance with Infinispan.

1. If the BA Server or standalone Mondrian engine are running, shut them down now.

2. If you performed a default install of the Pentaho Analysis Enterprise Edition package, then you should have all of the required JARs installed to the BA or Mondrian server. If you aren't sure, verify now that the following JARs are present in the `/WEB-INF/lib/` directory inside of your deployed pentaho.war or Mondrian engine:

   - pentaho-analysis-ee
   - commons-lang
   - commons-io
   - commons-codec
   - pentaho-ee-dsc-core
   - memcached

3. Edit the **pentaho-analysis-config.xml** in the `/WEB-INF/classes/` directory inside the deployed pentaho.war or Mondrian engine, and change the value of **SEGMENT_CACHE_IMPL** to match the class name referenced below:

   ```
   <entry key="SEGMENT_CACHE_IMPL">com.pentaho.analysis.segmentcache.impl.memcached.
   MemcachedSegmentCache</entry>
   ```

4. Edit the **memcached-config.xml** in the `/WEB-INF/classes/` directory inside the deployed pentaho.war or Mondrian engine, and change the values of **SALT**, **SERVERS**, and **WEIGHT** to match your preference:

   ```
   <entry key="SALT">YOUR SECRET SALT VALUE HERE</entry>
   <entry key="SERVERS">192.168.0.1:1642,192.168.0.2:1642</entry>
   <entry key="WEIGHTS">1,1</entry>
   ```

Your Pentaho Analysis Enterprise Edition instance is now configured to use Memcached for ROLAP segment caching.

Memcached Configuration Options

These properties control Memcached settings, and are set in the **memcached-config.xml** file in the `/WEB-INF/classes/` directory inside of your deployed pentaho.war or Mondrian engine.

Note: This is not a comprehensive list of the potential Memcached settings; the options explained below are the ones most critical to Memcached configuration for Pentaho Analysis.

Property	Purpose
SERVERS	A comma-separated list of servers and port numbers representing the Memcached nodes usable by the plugin.
WEIGHTS	A comma-separated list of numbers representing the relative caching capacity of the servers defined in the SERVERS property. There must be exactly as many values of WEIGHTS as there are values of SERVERS. As an example, if the first server has a capacity of 128 megabytes, and the second has a capacity of 256 megabytes, the correct values for the WEIGHTS property should be "1,2", indicating that the first server has a relative size of half of the second one.
SALT	A secret key prefix to be used when saving and loading segment data from the Memcached nodes. This property must be the same for all Mondrian nodes that share their caches. If the SALT value is different from one node to the next, the nodes will not be able to share their cache data.

Copying the Oracle Quartz JAR Pre-Build

This section is only for administrators who are using an Oracle database for the Pentaho solution repository. You do not need to do this for Oracle data sources; only for Oracle solution repositories.

1. Create a `/biserver-manual-ee/build-resources/custom-pentaho-webapp/WEB-INF/lib/` directory.

2. Copy the `/biserver-manual-ee/build-resources/pentaho-third-party/quartz-oracle-1.7.2.jar` to the newly created `/biserver-manual-ee/build-resources/custom-pentaho-webapp/WEB-INF/lib/` directory.

The WAR build process includes the defined file. You are ready to proceed to the WAR build instructions.

Removing the Xalan JAR for JBoss Deployments

JBoss 5.1 includes a Xalan JAR file that interferes with one included in the BA Server; they are different versions, so there is a class-cast exception whenever Xalan is called by BA Server content.

To fix this problem, remove the Xalan JAR from the BA Server build materials before building the WAR:

```
rm ./biserver-manual-ee/build-resources/pentaho-third-party/xalan-2.6.0.jar
```

This will force the BA Server to use the newer JBoss-supplied Xalan instance.

> **Note:** If you've already built a WAR and don't want to rebuild it, you can unpack the WAR and remove this JAR from the lib directory by hand.

How to Build BA Server WARs with Apache Ant

Review the build options in the next section and determine the exact command line options to pass to Ant. You must have Ant installed in order to proceed, and you should be logged in on a user account that has permission to use Ant and write to the appropriate directories.

This procedure covers a custom Pentaho BA Server build, using Apache Ant with the Enterprise Edition manual deployment package.

1. Open the `/biserver-manual-ee/build-resources/pentaho-webapp/WEB-INF/web.xml` file with a text editor.
2. Change the **solution-path** parameter to match the directory you created in *Creating a Pentaho Directory*, plus **pentaho-solutions**, which you will copy there after the WARs are built.
   Linux and Solaris

```
<context-param>
    <param-name>solution-path</param-name>
    <param-value>/home/pentaho/pentaho/server/biserver-ee/pentaho-solutions</param-value>
</context-param>
```

   Windows

```
<context-param>
    <param-name>solution-path</param-name>
    <param-value>C:/Users/Pentaho/pentaho/server/biserver-ee/pentaho-solutions</param-value>
</context-param>
```

> **Note:** Windows paths must use forward slashes instead of the traditional backslashes.

3. Optionally, copy any specially-customized application server configuration files to the `/build-resources/custom-pentaho-webapp/META-INF/` directory.

   The files you put into this directory will be copied to the META-INF directory in the WAR, and will override the defaults if your application server is configured to accept them.
4. Open a terminal or command prompt window and navigate to the directory you unpacked the BA Server package to. By default, this is `/biserver-manual-ee/`
5. Run Ant with the appropriate build options, as listed in *Ant Build Options*.

The **pentaho-style.war**, which contains style information for Pentaho content, will be created in the `/biserver-manual-ee/build/pentaho-wars/` directory: The application package, **pentaho.war**, will be stored in the `/`

`biserver-manual-ee/build/pentaho-wars/appserver/` directory, where *appserver* is the name of the Java application server (JBoss or Tomcat) that you specified in the build options. JBoss WARs are one level deeper in the **no-portal** subdirectory.

The solution directory will be created and installed into the `biserver-manual-ee/build/pentaho-solutions/` directory.

Note: If you receive any Ant-related errors, check your Ant command syntax and the spelling of your parameters.

Ant Build Options

The build requires you to pass a few property/value pairs to Ant from the command line. The Ant option for adding parameters is **-D**, so that must preface every one of the below-listed parameters, with no space between the switch and the parameter.

Property	Value
archive.target	The application archive name, including the name of your application server, database, and package format. Possible values are: **war-pentaho-jboss** and **war-pentaho-tomcat**
db	The database that you will be using for the Pentaho solution repository. Possible values are: **mysql5**, **oracle10g**, **postgresql**.
logfile	Specifies a file name to output a detailed build log to. **This option does not require a -D operator.** See the example below for details. Note: The Ant output is not typically verbose enough to require a log file for parsing.

Here is an example Ant command that builds a WAR configured for the Tomcat application server and a MySQL repository database, and dumps the build log to a text file:

```
ant -Darchive.target=war-pentaho-tomcat -Ddb=mysql5 -logfile pentaho-
build.txt
```

Here is an example Ant command that builds a WAR configured for the JBoss application server and a PostgreSQL repository database:

```
ant -Darchive.target=war-pentaho-jboss -Ddb=postgresql
```

Deploying the BA Server Web Application

Once the BA Server WAR is built, it must be deployed to a Java application server. This involves initializing your database, adding the appropriate database drivers for your application server, and copying over the newly built WAR files. Below are instructions for JBoss and Tomcat on both the Windows and Linux or Solaris platforms. Only follow the sections that apply to your application server, database, and operating system.

In order to simplify troubleshooting, you should test the newly installed BA Server with a basic configuration before you add data sources and perform other complex configuration tasks.

Moving the Pentaho Solutions Directory

The pentaho-solutions directory contains all of your BA Server settings, preferences, solutions, and content, and is mirrored in the solution repository database. Follow the directions below to copy over the sample pentaho-solutions directory to a standard Pentaho directory hierarchy.

Copy the `/biserver-manual-ee/build/pentaho-solutions/` directory to the `/pentaho/server/biserver-ee/` directory that you created earlier in the guide.

```
cp -r /biserver-manual-ee/build/pentaho-solutions/ /home/pentaho/pentaho/server/
biserver-ee/
```

Your solutions directory has been properly created in a standard and application server-accessible location.

Installing Pentaho User Console Plugins

In order to proceed with this task, you must have already purchased an Enterprise Edition license for the products you want to install. Skip this process if you do not have the requisite license.

The Pentaho User Console is built with a plugin architecture that enables you to expand its functionality with new client tools and functions. Follow the directions below to install any of the following plugins:

* Pentaho Analyzer
* Pentaho Dashboard Designer
* Pentaho Interactive Reporting
* Pentaho Mobile

> **Note:** The Pentaho Analysis Enterprise Edition plugin is separate from Analyzer, and requires a different installation process. That process is found in *Installing the Analysis Enterprise Edition Plugin*.

1. If you have not already done so, download the plugin packages from the Pentaho Customer Support Portal. The installation materials list are defined in *Obtaining the Installation Matrials*.
2. Unpack the plugin packages to the `/pentaho/server/biserver-ee/pentaho-solutions/system/` directory.

   This will create a subdirectory for each plugin that you unpack.

The plugins you downloaded are now installed, though you will still need to merge your solution files and register your licenses.

Removing the Sample Pentaho Analysis Data Source

BA Server 4.8 ships with a "Steel Wheels" sample data source pre-configured. This is useful for some evaluation and test deployments, but since the Steel Wheels database is not included in the manual deployment package, starting the BA Server without removing the sample data source will cause an error. Follow the directions below.

1. Edit the `/pentaho-solutions/system/olap/datasources.xml` file.
2. Remove the following block of XML:

```
<Catalog name="SteelWheels">
    <DataSourceInfo>Provider=mondrian;DataSource=SampleData</DataSourceInfo>
    <Definition>solution:steel-wheels/analysis/steelwheels.mondrian.xml</Definition>
</Catalog>
Catalog name="SampleData">
```

```
        <DatasourceInfo>Provider=mondrian;DataSource=SampleData</DataSourceInfo>
        <Definition>solution:steel-wheels/analysis/sampledata.mondrian.xml</Definition>
    </Catalog>
```

3. Save and close the file.

Your Pentaho Analysis data source definition file is cleansed of sample data sources.

Adding Linux and OS X Support for Printing Analyzer Geo Maps

There is an experimental function in Analyzer that enables you to print Geo Map visualizations. Follow the directions below to enable support on Linux and OS X. Windows configuration is covered in *Adding Windows Support for Printing Analyzer Geo Maps* on page 22.

☞ **Note:** Pentaho requires a specific version of **wkhtmltoimage** in order for this print function to work correctly. Even if you have wkhtmltoimage installed on your machine, install the Pentaho-supplied version as explained below. It will be explicitly called by Analyzer, and will not interfere with your existing wkhtmltoimage instance.

☞ **Note:** If you used the Business Analytics graphical installer, **skip all steps except the last two.**

1. Navigate to the **wkhtmltoimage** directory in your Pentaho installation. For archive installations, this is `/pentaho/biserver-ee/third-party-tools/wkhtmltoimage/`, and for manual deployments it is `/biserver-manual-ee/build-resources/third-party-tools/wkhtmltoimage/`.

2. For OS X, unpack the **wkhtmltoimage_mac.tar.bz2** package; for 32-bit Linux, unpack the **wkhtmltoimage_linux_i386.tar.bz2** package; for 64-bit Linux, unpack the **wkhtmltoimage_linux_amd64.tar.bz2** package.

3. There is only one file in this archive: the executable **wkhtmltoimage**. If the executable has an extension, such as **i386** or **amd64**, rename the file to **wkhtmltoimage**.

4. Move the executable to: `/home/pentaho/pentaho/server/biserver-ee/wkhtmltoimage.` directory.

5. Edit the `/pentaho-solutions/system/pentaho.xml` file and set the value of **<html-to-image-tool>** to the full path to the **wkhtmltoimage** executable that you just unpacked.

```
<html-to-image-tool>/home/pentaho/pentaho/server/biserver-ee/third-party-tools/
wkhtmltoimage/wkhtmltoimage</html-to-image-tool>
```

6. Save and close the file, then edit the `/pentaho-solutions/system/analyzer/settings.xml` file and uncomment the following line:

```
<!--open_layers>geomapexport</open_layers-->
```

7. Save and close the settings.xml file.

You have enabled the experimental print feature for Analyzer Geo Map visualizations.

Adding Windows Support for Printing Analyzer Geo Maps

There is an experimental function in Analyzer that enables you to print Geo Map visualizations. Follow the below directions to enable support on Windows. Linux and OS X configuration are covered in *Adding Linux and OS X Support for Printing Analyzer Geo Maps* on page 22.

☞ **Note:** Pentaho requires a specific version of **wkhtmltoimage** in order for this print function to work correctly. Even if you have wkhtmltoimage installed on your machine, install the Pentaho-supplied version as explained below. It will be explicitly called by Analyzer, and will not interfere with your existing wkhtmltoimage instance.

☞ **Note:** If you used the Business Analytics graphical installer, **skip all steps except the last two.**

1. Navigate to the **wkhtmltoimage** directory in your Pentaho installation. For archive installations, this is `\pentaho\biserver-ee\third-party-tools\wkhtmltoimage\`, and for manual deployments it is `\biserver-manual-ee\build-resources\third-party-tools\wkhtmltoimage\`.

2. Run the **wkhtmltoimage_windows_installer.exe** executable.

3. Accept the license agreement to continue.
4. Uncheck the **wkhtmltopdf** option, then click **Next**.
5. For the **Destination folder**, type in or navigate to the following path, creating the directories if they do not exist: `C:\pentaho\biserver-ee\third-party-tools\wkhtmltoimage\`. Once you have navigated to the path, click **Install**.
6. When the installation is complete, click **Close**, then edit the `\pentaho-solutions\system\pentaho.xml` file and set the value of **<html-to-image-tool>** to the full path to the **wkhtmltoimage** executable that you specified in the installer, including the executable name of the package, which is **wkhtmltoimage**.

    ```
    <html-to-image-tool>C:\pentaho\biserver-ee\third-party-tools\wkhtmltoimage
    \wkhtmltoimage</html-to-image-tool>
    ```

7. Close that file, then edit the `\pentaho-solutions\system\analyzer\settings.xml` file and uncomment the following line:

    ```
    <!--open_layers>geomapexport</open_layers-->
    ```

8. Save and close the settings.xml file.

You have enabled the experimental print feature for Analyzer Geo Map visualizations.

Initializing a PostgreSQL Database

In order to complete this process, you need to have built a Pentaho WAR file using Apache Ant. You'll also need to have the PostgreSQL native client tools installed, and the database server running as a service. Your PostgreSQL configuration must also support logins from all users; this is not the default configuration in all cases, so you may have to edit your **pg_hba.conf** file to support this option.

> **Note:** If you do need to make changes to pg_hba.conf, you must restart the PostgreSQL server before proceeding.

This procedure will initialize Pentaho-related databases on your PostgreSQL server.

1. Navigate to the `/biserver-manual-ee/build-resources/pentaho-data/postgresql/` directory.

    There are several scripts in this directory. You should only run the ones mentioned below.
2. Edit the **create_repository_postgresql.sql** and **create_quartz_postgresql.sql** scripts and set the password references to the passwords you want to use for the Pentaho administrator and user accounts.

    The user account specified here is not for a specific Pentaho BA Server user; rather it is a generic account that the software uses to update the database when any regular user is logged in.
3. Execute the schedule database initialization script as the PostgreSQL root user by running this command from a terminal (change "postgres" to the name of your PostgreSQL root user): **psql -U postgres < create_quartz_postgresql.sql**.
4. Finally, initialize your repository database: **psql -U postgres < create_repository_postgresql.sql**.

PostgreSQL is now configured to act as a Pentaho BA Server solution repository.

PostgreSQL Solution Repository Configuration

Follow the below directions to configure the BA Server to use PostgreSQL as a solution repository with the default Pentaho security data access object. Even if you will end up using some other authentication method, follow all of these directions anyway so that you can verify that the BA Server works with a basic configuration.

1. Open the `/pentaho/server/biserver-ee/pentaho-solutions/system/applicationContext-spring-security-hibernate.properties` file in a text editor.
2. Edit the file and change the following values to those in this example, modifying the details for your specific PostgreSQL configuration:

    ```
    jdbc.driver=org.postgresql.Driver
    jdbc.url=jdbc:postgresql://localhost:5432/hibernate
    jdbc.username=hibuser
    jdbc.password=password
    hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
    ```

3. Save and close the file, then edit `/pentaho/server/biserver-ee/pentaho-solutions/system/hibernate/postgresql.hibernate.cfg.xml`

4. Change the default hostname, port number, and user credentials to match your database configuration, then save and close the file.

   Even if you plan to use the default settings, open the file to verify that the defaults are what you expect them to be.

The BA Server is now fully configured to connect to a PostgreSQL solution repository.

Configuring Quartz For PostgreSQL

Follow this procedure to replace the generic JDBC JobStore delegate with one specific to PostgreSQL.

1. Open the `/pentaho/server/biserver-ee/pentaho-solutions/system/quartz/quartz.properties` file in a text editor.

2. Find the following line in the properties file: `org.quartz.jobStore.driverDelegateClass =`

3. In the aforementioned line, replace the existing delegate with this:
   org.quartz.impl.jdbcjobstore.PostgreSQLDelegate

4. Save the file and close the text editor.

The BA Server's scheduling (quartz) database is now configured to work with PostgreSQL.

Configuring Audit Logging For PostgreSQL

Follow the below procedure to replace the generic JDBC driver for the Quartz JDBC JobStore.

1. Copy all of the files from `/pentaho/server/biserver-ee/pentaho-solutions/admin/audit/dialects/postgresql/` to the `/pentaho/server/biserver-ee/pentaho-solutions/admin/audit/` directory, overwriting the existing files there.

2. Copy the **audit_sql.xml** file from `/pentaho/server/biserver-ee/pentaho-solutions/system/dialects/postgresql/` to `/pentaho/server/biserver-ee/pentaho-solutions/system/`, overwriting the existing version there.

Audit logging is now configured for PostgreSQL.

Initializing a MySQL Database

In order to complete this process, you need to have built the Pentaho WAR file using Apache Ant. You'll also need to have the MySQL client installed on your machine, and the MySQL database server must be running as a service.

⚠️ **Caution:** You must use the **ASCII** character set in order for these scripts to execute properly. UTF-8 is known not to work because of text string length limitations.

This procedure will initialize Pentaho-related databases on your MySQL server.

1. Navigate to the `/biserver-manual-ee/build-resources/pentaho-data/mysql5/` directory.

   There are several scripts in this directory. You should only run the ones mentioned below. There are a variety of ways to execute an SQL script. The simplest way is by using the MySQL client from the terminal.

2. Edit the **create_repository_postgresql.sql** and **create_quartz_postgresql.sql** scripts and set the password references to the passwords you want to use for the Pentaho administrator and user accounts.

   The user account specified here is not for a specific Pentaho BA Server user; rather it is a generic account that the software uses to update the database when any regular user is logged in.

3. Execute the **create_quartz_mysql.sql** script via your preferred utility or process.

   ```
   mysql -u root -p <create_quartz_mysql.sql
   ```

4. Execute the **create_repository_mysql.sql**.

   ```
   mysql -u root -p <create_repository_mysql.sql
   ```

MySQL is now configured to act as a Pentaho BA Server solution repository.

MySQL Solution Repository Configuration

Follow the below directions to configure the BA Server to use MySQL as a solution repository with the default Pentaho security data access object. Even if you will end up using some other authentication method, follow all of these directions anyway so that you can verify that the BA Server works with a basic configuration.

1. Open the `/pentaho/server/biserver-ee/pentaho-solutions/system/applicationContext-spring-security-hibernate.properties` file in a text editor.

2. Edit the file and change the following values to those in this example, modifying the details for your specific MySQL configuration:

   ```
   jdbc.driver=com.mysql.jdbc.Driver
   jdbc.url=jdbc:mysql://localhost:3306/hibernate
   jdbc.username=hibuser
   jdbc.password=password
   hibernate.dialect=org.hibernate.dialect.MySQL5Dialect
   ```

3. Save and close the file, then edit `/biserver-manual-ee/build-resources/pentaho-solutions/system/hibernate/mysql.hibernate.cfg.xml`.

4. Change the default hostname, port number, and user credentials to match your database configuration, then save and close the file.

   Even if you plan to use the default settings, open the file to verify that the defaults are what you expect them to be.

The BA Server is now fully configured to connect to a MySQL solution repository.

Configuring Quartz For MySQL

Follow the below procedure to replace the generic JDBC driver for the Quartz JDBC JobStore.

1. Open the `/pentaho/server/biserver-ee/pentaho-solutions/system/quartz/quartz.properties` file in a text editor.

2. Find the following line in the properties file: `org.quartz.jobStore.driverDelegateClass =`

3. In the aforementioned line, replace the existing delegate portion with `org.quartz.impl.jdbcjobstore.StdJDBCDelegate.`

4. Save the file and close the text editor.

The BA Server's scheduling (quartz) database is now configured to work with MySQL.

Configuring Audit Logging For MySQL

Audit logging is enabled by default in the BA Server. To configure it for MySQL, follow the below process.

1. Copy all of the files from `/pentaho/server/biserver-ee/pentaho-solutions/admin/audit/dialects/mysql5/` to the `/pentaho/server/biserver-ee/pentaho-solutions/admin/audit/` directory, overwriting the existing files there.

2. Copy the audit_sql.xml file from `/pentaho/server/biserver-ee/pentaho-solutions/system/dialects/mysql5/` to `/pentaho/server/biserver-ee/pentaho-solutions/system/`, overwriting the existing version there.

Audit logging is now configured for MySQL.

Initializing an Oracle Database

In order to complete this process, you need to have built the Pentaho application file using Apache Ant. Oracle should be installed and running, and you should have a basic familiarity with Oracle database management.

This procedure will initialize a Pentaho table space on your Oracle server.

1. Navigate to the `/biserver-manual-ee/build-resources/pentaho-data/oracle10g/` directory.

   There are several scripts in this directory. You should only run the ones mentioned below.

2. Edit the **create_quartz_ora.sql** and **create_repository_ora.sql** scripts and replace the **admin/password** portion of the **conn** lines with the username and password you use to manage your Oracle database.

The user credentials you type in here should have permission to create tables in the **pentaho** database, create users to access that database, and create and manage permissions for tables in it as well.

3. Edit the **datafile** path with the path to your Oracle installation, then save and close your text editor.
4. Run the SQL scripts with your preferred utility or process, in the order shown above.

   SQL*Plus is probably the easiest tool to use for this purpose.

Oracle is now configured to act as a Pentaho BA Server data repository.

Oracle Solution Repository Configuration

Follow the below directions to configure the BA Server to use Oracle as a solution repository with the default Pentaho security data access object. Even if you will end up using some other authentication method, follow all of these directions anyway so that you can verify that the BA Server works with a basic configuration.

1. Open the `/pentaho/server/biserver-ee/pentaho-solutions/system/applicationContext-spring-security-hibernate.properties` file in a text editor.
2. Edit the file and change the following values to those in this example, modifying the details for your specific Oracle configuration:

   ```
   jdbc.driver=oracle.jdbc.driver.OracleDriver
   jdbc.url=jdbc:oracle:thin:@localhost:1521:pentaho
   jdbc.username=hibuser
   jdbc.password=password
   hibernate.dialect=org.hibernate.dialect.Oracle10gDialect
   ```

3. Save and close the file, then edit `/biserver-manual-ee/build-resources/pentaho-solutions/system/hibernate/oracle10g.hibernate.cfg.xml`.
4. Change the default hostname, port number, and user credentials to match your database configuration, then save and close the file.

   Even if you plan to use the default settings, open the file to verify that the defaults are what you expect them to be.

The BA Server is now fully configured to connect to an Oracle solution repository.

Configuring Quartz For Oracle

Follow the below procedure to replace the generic JDBC driver for the Quartz JDBC JobStore.

1. Open the `/pentaho/server/biserver-ee/pentaho-solutions/system/quartz/quartz.properties` file in a text editor.
2. Find the following line in the properties file: `org.quartz.jobStore.driverDelegateClass = org.quartz.impl.jdbcjobstore.StdJDBCDelegate`
3. In the aforementioned line, replace the **org.quartz.impl.jdbcjobstore.StdJDBCDelegate** portion with this: **org.quartz.impl.jdbcjobstore.oracle.OracleDelegate**
4. Save the file and close the text editor.

The BA Server's scheduling (quartz) database is now configured to work with Oracle.

Configuring Audit Logging For Oracle

Audit logging is enabled by default in the BA Server. To configure it for Oracle, follow the below process.

1. Copy all of the files from `/pentaho/server/biserver-ee/pentaho-solutions/admin/audit/dialects/oracle10g/` to the `/pentaho/server/biserver-ee/pentaho-solutions/admin/audit/` directory, overwriting the existing files there.
2. Copy the audit_sql.xml file from `/pentaho/server/biserver-ee/pentaho-solutions/system/dialects/oracle10g/` to `/pentaho/server/biserver-ee/pentaho-solutions/system/`, overwriting the existing version there.

Audit logging is configured for Oracle.

Deploying to Tomcat

The Pentaho Web application deployment process differs between Windows and Unix-like platforms. Refer to the appropriate section for your situation.

Modifying startup.bat on Windows

Before you proceed to deploying Business Analytics, you must modify the Tomcat startup script so that the BA Server has sufficient memory allocated to it, and can find your Pentaho licenses.

1. Stop the Tomcat server if it is running.
2. Open a terminal window and navigate to the `\bin\` subdirectory of your Tomcat home directory.
3. Using a text editor, open the `startup.bat` file.
4. Add this line directly before the **call "%EXECUTABLE%" start %CMD_LINE_ARGS%** line near the end of the file:

```
set CATALINA_OPTS=-Xms768m -Xmx1024m -XX:MaxPermSize=256m -
Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000 -
Dpentaho.installed.licenses.file=%PENTAHO_INSTALLED_LICENSE_PATH%
```

5. Save and close your text editor.

The Tomcat startup script is now properly modified to work with the Pentaho BA Server.

Modifying startup.sh on Linux and Solaris

Before you proceed to deploying Business Analytics, you must modify the Tomcat startup script so that the BA Server has enough memory and can find your Pentaho licenses.

1. Stop the Tomcat server if it is running.
2. Open a terminal window and navigate to the `/bin/` subdirectory of your Tomcat home directory.
3. Using a text editor, open the `startup.sh` file.
4. Add this line directly before the **exec "$PRGDIR"/"$EXECUTABLE" start "$@"** line near the end of the file:

```
export CATALINA_OPTS="-Xms768m -Xmx1024m -XX:MaxPermSize=256m -
Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000 -
Dpentaho.installed.licenses.file=$PENTAHO_INSTALLED_LICENSE_PATH"
```

5. Save and close your text editor.

The Tomcat startup script is now properly modified to work with the Pentaho BA Server.

Modifying server.xml To Work With Accented Characters

This procedure is only necessary if you plan to use character sets that include accents, such as Spanish and French.

Follow the instructions below to implement accented character set support. Change the paths to match your configuration.

1. Stop the Tomcat service.

```
sudo /etc/init.d/tomcat stop
```

2. Open the `/tomcat/server/conf/server.xml` file with a text editor.
3. Locate each **Connector** node (typically there are four in a default Tomcat configuration) and add a **URIEncoding="UTF-8"** parameter to it, as shown below:

```
<Connector URIEncoding="UTF-8" port="8080" protocol="HTTP/1.1"
              connectionTimeout="20000"
              redirectPort="8443" />
```

4. Save and close the file, then restart the Tomcat service.

```
sudo /etc/init.d/tomcat start
```

Tomcat is now properly configured to handle accented characters.

Deploying a WAR to Tomcat

You must previously complete the BA Server build process.

To deploy the BA Server, you must copy the WAR files to the proper location, and ensure that your application server has the right JDBC drivers to access the solution and repository as well as any data sources you want to connect to. In the below instructions, adjust the directory paths to match your situation.

1. Ensure that a **CATALINA_HOME** system variable is set, and that its value is correct. If it is not, consult the Apache Tomcat documentation to learn how to establish it.
2. Stop the Tomcat server.
3. Copy the `/biserver-manual-ee/build/pentaho-wars/tomcat/pentaho.war` file to the `webapps` subdirectory of your Tomcat home directory.
4. Copy the **pentaho-style.war** file from the `/biserver-manual-ee/build/pentaho-wars/` directory to the Tomcat webapps directory.
5. Copy the appropriate JDBC driver JAR files for your solution repository and your data source databases to the `/lib/` subdirectory in the Tomcat root.

   You are expected to provide your own JDBC driver JARs. You can find the most recent versions of the drivers from your database vendor or driver developer's Web site.
6. Optional: If you changed the Web application name earlier, you must now rename the exploded pentaho WAR directory to whatever you specified earlier.

   ```
   mv pentaho.war example.war
   ```

All of the appropriate Pentaho files are now installed into your application server. If you are not going to implement single sign-on (CAS) support in the BA Server, you can safely delete the `biserver-manual-ee` directory.

Deploying to JBoss

The Pentaho Web application deployment process differs between Windows and Unix-like platforms. Refer to the appropriate section for your situation.

Modifying run.bat on Windows

Before you proceed to deploying Business Analytics, you must modify the JBoss startup script to match Pentaho's resource requirements and point to the Pentaho license file. If you have an existing JAVA_OPTS setting on your server, it may already meet the minimum memory requirements, but you will still have to add the path to the installed license file.

1. Stop the JBoss server.
2. Open a terminal window and navigate to the `\bin\` subdirectory of your JBoss home directory.
3. Using a text editor, open the `run.bat` file.
4. Add this line below the JAVA_OPTS if statement (outside of the brackets, not part of the if statement):

   ```
   set JAVA_OPTS=%JAVA_OPTS% -Xms768m -Xmx1024m -XX:MaxPermSize=256m -
   Dpentaho.installed.licenses.file=%PENTAHO_INSTALLED_LICENSE_PATH%
   ```

   You may need to adjust the memory settings for your environment.
5. Save and close your text editor.

The JBoss startup script is now properly modified to work with the Pentaho BA Server.

Modifying run.conf on Linux and Solaris

Before you proceed to deploying Business Analytics, you must modify the JBoss startup script so that the BA Server has enough memory and can find your Pentaho licenses.

1. Stop the JBoss server.
2. Open a terminal window and navigate to the `/bin/` subdirectory of your JBoss home directory.

3. Using a text editor, open the `run.conf` file.
4. Modify the **Xms** memory settings in the **JAVA_OPTS** line to be at least 768MB (more, if you have the resources and are concerned with performance, or if you'll be running very large reports), and the **Xmx** to at least 1GB (this number must be higher than the value of Xms).
5. Add the following options to the **JAVA_OPTS** line:

   - -Djava.awt.headless=true
   - -Djava.io.tmpdir=/tmp/
   - -Dpentaho.installed.licenses.file=$PENTAHO_INSTALLED_LICENSE_PATH

```
# Specify options to pass to the Java VM.
    if [ "x$JAVA_OPTS" = "x" ]; then
        JAVA_OPTS="-Xms768m \
        -Xmx1024m \
        -XX:MaxPermSize=256m \
        -Dsun.rmi.dgc.client.gcInterval=3600000 \
        -Dsun.rmi.dgc.server.gcInterval=3600000 \
        -Djava.awt.headless=true \
        -Djava.io.tmpdir=/tmp/ \
        -Dpentaho.installed.licenses.file=$PENTAHO_INSTALLED_LICENSE_PATH
    fi
```

   You may need to adjust these settings for your environment. For instance, if you do not have a `/tmp/` directory, you may want to change that setting to `/var/tmp/` or some other location.
6. Save and close your text editor.

The JBoss startup script is now properly modified to work with Pentaho Business Analytics.

Deploying a WAR to JBoss

The JBoss server must be stopped in order to execute these steps.

To deploy the Pentaho WARs on JBoss, you must either copy them by hand to the JBoss deploy directory, or launch the JBoss Administration Console and install them as resources. This procedure assumes a default JBoss instance; feel free to modify the instructions, and the directory paths, to match your configuration.

> **Note:** If you use the JBoss Administration Console to install the Pentaho WARs, you may see a nullPointerException error message. This is related to the JBoss deployer; it will not prevent the pentaho.war and pentaho-style.war files from being deployed, and it will not show up in the JBoss logs again in the future. **Be sure to select the Deploy Exploded option.**

1. Copy **pentaho.war** from the `/biserver-manual-ee/build/pentaho-wars/jboss/no-portal/` directory to `/jboss/server/default/deploy/`.
2. Copy the **pentaho-style.war** from the `/biserver-manual-ee/build/pentaho-wars/` directory to `/jboss/server/default/deploy/`.
3. Optional: If you changed the Web application context earlier, you must now rename pentaho.war to whatever you specified earlier.

```
mv pentaho.war example.war
```

4. If you didn't use a deployer utility to unpack the pentaho.war, then you must now manually unpack it to a directory of the same name in the `/server/default/deploy/` directory. This means your directory name must be pentaho.war (including the .war extension).

> **Warning:** If you skip this step and JBoss does not unpack the WAR before you use the Pentaho Enterprise Console, then the Pentaho Enterprise Console will not be able to save your BA Server settings.

5. Copy the appropriate JDBC driver JAR files for your solution repository and your data source databases to the `/server/default/lib/` subdirectory in the JBoss root.

   You are expected to provide your own JDBC driver JARs. You can find the most recent versions of the drivers from your database vendor or driver developer's Web site.

All of the appropriate Pentaho files are now installed into your application server. If you are not going to implement single sign-on (CAS) support in the BA Server, you can safely delete the `biserver-manual-ee` directory.

Adding PDI Enterprise Repository Content Support to the BA Server

If you are using a Pentaho Data Integration (PDI) enterprise repository (through a Data Integration Server) to store PDI jobs and transformations, and you plan on using those jobs and transformations in action sequences that will be run on the BA Server, you must install some BA Server plugins from the PDI client tool package. This is not a typical scenario, but there is no harm in performing it if you aren't sure of the details.

1. Download a PDI Enterprise Edition 4.4 client tool archive package from the Pentaho Customer Support Portal.

   The package name (available in both tar.gz and zip formats) is: **pdi-ee-client-4.4.0-GA**

2. Unpack the archive to a temporary location.

3. Edit the `/pentaho/server/biserver-ee/pentaho-solutions/system/kettle/settings.xml` file.

4. Change the value of the **<repository.type>** node from **files** to **rdbms**.

5. Enter your enterprise repository connection information in the proper nodes.

6. Enter the location of your local **repositories.xml** file in the **<repositories.xml.file>** node.

   > **Note:** This file is created on your PDI client workstation when you establish a connection to an enterprise repository. Once you have made all of your repository connections on a workstation, copy the **repositories.xml** file to the `~/.kettle/` directory on the BA Server and DI Server machines. If the client tool and servers are all on the same machine, you do not have to copy the file. If you have not yet established any repositories, you will have to revisit this procedure later when your PDI environment is fully configured.

7. Copy the contents of `/data-integration/plugins/` to the `/pentaho/server/biserver-ee/pentaho-solutions/system/kettle/plugins/` directory.

   ```
   cp -r /tmp/data-integration/plugins/* /home/pentaho/pentaho/server/biserver-ee/
   pentaho-solutions/system/kettle/plugins/
   ```

8. Remove the unpacked archive.

   ```
   rm -rf /tmp/data-integration/
   ```

Your BA Server is now configured to run content stored in the DI Server.

Dashboard Printing in a Headless Linux Environment

To enable dashboard printing for headless Linux operating systems, you must implement an Xvnc server. Implementing an Xvnc server requires additional installation steps.

For step-by-step instructions, see *Enabling Dashboard Printing on Headless CentOS and Red Hat* or *Enabling Dashboard Printing on Headless Ubuntu*.

Enabling Dashboard Printing on Headless CentOS and Red Hat

These instructions apply to both CentOS and Red Hat, and use CentOS for exemplary purposes.

To implement headless dashboard printing, you need access to the SWT library that is shipped with the dashboards plugin. The WebKit library depends on this library. The relevant packages on CentOS are `webkitgtik-devel-1.2.6-2.el6_0` and `webkitgtk-1.2.6-2.el6_0`.

1. Install the relevant CentOS packages by entering `$ yum install webkitgtik-devel-1.2.6-2.el6_0 webkitgtk-1.2.6-2.el6_0`.
2. If the VNC server is not installed, run `$ yum install tigervnc-server`.
3. To install a desktop environment, run `$ yum groupinstall Desktop`. If you need more information, see the CentOS website and search for *virtual network computing (VNC)*.
4. Start the VNC server with this command: `$ vncserver :1 -geometry 1600x1200 -localhost $ export DISPLAY=localhost:1`.
5. Start the BA Server with this command: `$./start-pentaho.sh`. The VNC server and the BA Server must be started by the same user.

Set the `-geometry` option to the width and height of the largest supported client resolution.

You can add the commands to run and configure the VNC server to the Pentaho startup script, or you can start the VNC server as a regular service, whichever you prefer.

Enabling Dashboard Printing on Headless Ubuntu

These instructions show how to enable headless dashboard printing on Ubuntu Linux.

1. Install the relevant Ubuntu packages.

```
$ sudo apt-get install libwebkit-dev
$ sudo apt-get install libwebkit-1.0-2
$ sudo apt-get install libswt-gtk-3.5-jni
```

2. If the VNC server is not installed, run `$ sudo apt-get install vnc4server` to install it.
3. To install a desktop environment, run `$ sudo apt-get install ubuntu-desktop`.
4. Start the VNC server with this command: `$ vncserver :1 -geometry 1600x1200 -localhost $ export DISPLAY=localhost:1`.
5. Start the BA Server with this command: `$./start-pentaho.sh`. The VNC server and the BA Server must be started by the same user.

Set the `-geometry` option to the width and height of the largest supported client resolution.

You can add the commands to run and configure the VNC server to the Pentaho startup script, or you can start the VNC server as a regular service, whichever you prefer.

Installing the Pentaho Enterprise Console: Linux/OS X/Windows

The Pentaho Enterprise Console is a necessary BA Server administration tool, but it is not included with the manual deployment package. Follow the below instructions to install it.

> **Note: Solaris users:** The default behavior of the tar utility in Solaris 10 is to truncate long file names when unpacking an archive created with GNU tar, as the Pentaho archives are. Therefore, you must either install a GNU tar-compatible utility, or unpack the archives on a Linux or BSD machine before deploying them.

1. If you have not already done so, download the `pec-ee-4.8.0-GA` (zip or tar.gz) archive file.

   The only difference between the Windows zip and Linux/Solaris tar.gz files is the archive technology; tar.gz better preserves file and directory permissions on Unix-like operating systems.

2. Unzip the file to the `/pentaho/server/` directory.

   This will create an `/enterprise-console/` subdirectory there.

3. **For Oracle users:** You must copy the Oracle JDBC driver to the `/pentaho/server/enterprise-console/jdbc/` directory.

   Typically this file is named **ojdbc14.jar**. This is the same file that you copied to your application server when you deployed the WAR earlier.

4. Optional: The default Pentaho Enterprise Console start and stop port numbers are **8088** and **8033**, respectively. If you want to change them, navigate to the `/pentaho/server/enterprise-console/resource/config/` directory and edit the **console.properties** file. Change the **console.start.port.number** and **console.stop.port.number** values to match the port numbers you want to run Enterprise Console on.

```
console.start.port.number=18088
console.stop.port.number=18033
```

Pentaho Enterprise Console is now installed.

Installing the Pentaho Enterprise Console: Solaris

You must have the GNU tar utility installed in order to proceed. The default behavior of the tar utility in Solaris 10 is to truncate long file names when unpacking an archive created with GNU tar, as the Pentaho archives are. Therefore, you must either install a GNU tar-compatible utility, or unpack the archives on a Linux or BSD machine before deploying them.

The Pentaho Enterprise Console is a necessary BA Server administration tool, but it is not included with the manual deployment package. Follow the below instructions to install it.

1. If you have not already done so, download the `pec-ee-4.5.0-GA` (zip or tar.gz) archive file.

    The only difference between the Windows zip and Linux/Solaris tar.gz files is the archive technology; tar.gz better preserves file and directory permissions on Unix-like operating systems.

2. Unzip the file to the `/pentaho/server/` directory.

    This will create an `/enterprise-console/` subdirectory there.

3. **For Oracle users:** You must copy the Oracle JDBC driver to the `/pentaho/server/enterprise-console/jdbc/` directory.

    Typically this file is named **ojdbc14.jar**. This is the same file that you copied to your application server when you deployed the WAR earlier.

4. Edit the `/enterprise-console/start-pec.sh` script and replace the **#!/bin/sh** line at the top of the file with **#!/bin/bash**.

5. Edit the `/enterprise-console/stop-pec.sh` script and replace the **#!/bin/sh** line at the top of the file with **#!/bin/bash**.

6. Optional: The default Pentaho Enterprise Console start and stop port numbers are **8088** and **8033**, respectively. If you want to change them, navigate to the `/pentaho/server/enterprise-console/resource/config/` directory and edit the **console.properties** file. Change the **console.start.port.number** and **console.stop.port.number** values to match the port numbers you want to run Enterprise Console on.

    ```
    console.start.port.number=18088
    console.stop.port.number=18033
    ```

Pentaho Enterprise Console is now installed.

Initial Startup of the BA Server and Pentaho Enterprise Console

Linux, Solaris, and other Unix-like operating system users must perform the Pentaho Enterprise Console step below while logged in as or using the **pentaho** user credentials established earlier in this guide.

The basic installation procedures are now complete, so it's time to start the BA Server and Pentaho Enterprise Console servers so that you can proceed with low-level system configuration.

1. Start your application server normally.

    The BA Server should be running as soon as Tomcat or JBoss is fully operational. This process may take a few moments.

2. Execute the `/pentaho/server/enterprise-console/start-pec` script.

    This script has a **.bat** extension on Windows, and a **.sh** extension on Linux, Solaris, and OS X. It may take a few moments to compete the startup sequence.

Both the Pentaho BA Server and the Pentaho Enterprise Console server are now running and should be accessible via a Web browser at `http://localhost:8080` and `http://localhost:8088,` respectively.

Post-Install Configuration

After you've successfully installed the BA Server, you must set up your configuration information and other details. All of this is done through the Pentaho Enterprise Console.

> **Note:** In a manual deployment scenario, all paths in the Enterprise Console configuration screen -- Web-App, Solution Directory, Backup Directory -- must be absolute (full paths). Relative paths will not validate.

1. Open a Web browser and navigate to **http://localhost:8088** (change **localhost** to the IP address, hostname, or domain name of your BA Server).
2. Log in as **admin**, using **password** as the password.
3. Click the **+** (plus) button in the upper right corner of the Subscriptions section.

   An **Install License** dialogue will appear.
4. Click **Browse**, then navigate to the location you saved your LIC files to, click on one of the LIC files, then click **Open**.

   LIC files for each of your supported Pentaho products were emailed to you along with your Pentaho Welcome Kit. If you did not receive this email, or if you have lost these files, contact your Pentaho support representative. If you do not yet have a support representative, contact the Pentaho salesperson you were working with.
5. Click **OK**.

   The Setup screen will change according to the LIC file you installed. **Repeat the LIC file installation process for each license you have purchased.**

   > **Note:** After installing license files, Linux and Solaris users must restart the application server before they can use any functions in the Pentaho User Console. The Pentaho Enterprise Console does not have to be restarted, so you can continue configuring the system, but you must restart JBoss or Tomcat when you are finished.

6. In the **Solution Directory** field, enter `home/pentaho/server/biserver-ee/pentaho-solutions/`, or whatever the path to your pentaho-solutions directory is.
7. In the **Backup Directory** field, type in the location that you'd like to save Pentaho Enterprise Console backup data to.

   This can be any local directory that your application server has permissions to write to. You cannot, however, use relative paths in this or any other configuration field in this screen -- **all paths must be absolute**.
8. In the **Pentaho Web-App Path** field, enter `/tomcat/webapps/pentaho/`, or whatever the path to the unpacked pentaho.war directory is for your application server.
9. In the **Platform Administrator User Name** field, type in **admin**, or if you already have Pentaho User Console accounts established, type in the account name that you will use to manage reports and schedules.

   This user account does not have to exist yet, and you can change this value later.
10. Click **OK**.

    The rest of the settings in this screen do not need to change right now.
11. Click **Configuration** in the menu on the left side of Pentaho Enterprise Console.
12. Click the **Web Settings** tab at the top of the screen.
13. Change the **Fully Qualified Server URL** setting to match your server's hostname, domain name, or IP address. Do not change the directory or port number.

    If you need to change the port number, consult the *Pentaho Business Analytics Administrator's Guide*.

Your Pentaho BA Server has a minimal configuration. If you need to return to this screen, click on the wrench/screwdriver icon in the upper right corner of the screen.

Your administration work is only beginning; you should now consult the *Pentaho Business Analytics Administrator's Guide* to learn your way around the Pentaho Enterprise Console.

Testing and Using Your Server

Open a Web browser and navigate to the Pentaho BA Server Web address. If you installed the application onto your local machine, then the URL to reach the reporting and analysis samples is **http://localhost:8080/pentaho**. If you are trying to reach the application from a remote machine, the URL will be **http://example.com:8080/pentaho**, substituting **example.com** with the remote machine's hostname, IP address, or domain name. When the page loads, you will see a login screen.

You can log in as a sample user, but you must establish user accounts and roles in the Pentaho Enterprise Console if before you can log in with a unique user name and password. The sample users are:

Table 1: Sample Users

User Name	Access Level
joe	Administrator, Chief Executive Officer
pat	Developer
suzy	Information Services, Chief Technical Officer
tiffany	Developer, Development Manager

All sample users have a password of **password** as the default.

When your system is properly configured, log in with your administrator account, then verify that you can do the following, assuming your data has been prepared for reporting and analysis:

- Create a new analysis view.
- Create a new ad hoc report.
- If you are a Dashboards Enterprise Edition customer, create a new dashboard with the Dashboard Designer plugin.
- If you are a Pentaho Analysis Enterprise Edition customer, create a new analysis report with the Pentaho Analyzer plugin.
- Schedule a report to run at a regular interval.
- Using Pentaho Report Designer, publish a report to the BA Server.

Preparing for Production

The default condition of the BA Server is designed to support quick and easy evaluation by new customers. Once you move from evaluation to production, you may want to remove some of the evaluation-specific features in the BA Server. The instructions in this section explain how to remove them.

Switching to a Production Login Screen

The default Pentaho User Console login screen contains information and instructions for evaluators. Pentaho allows this extra information to be hidden via configuration; follow the directions below to hide the extra information. If you have chosen to perform a manual deployment, this information is hidden by default.

1. Shut down the BA Server and Enterprise Console if they are currently running.
2. Open a terminal or file browser window and navigate to the `/pentaho-solutions/system/` directory.

   If you installed via the archive package or the graphical installer, the full path is `/pentaho/server/biserver-ee/pentaho-solutions/system/`.
3. Edit the **pentaho.xml** file.
4. Find the **login-show-sample-users-hint** node and change its value to **false**.
5. Start the BA Server and Pentaho Enterprise Console.

You now have a login screen that has been scrubbed of evaluation content.

Compatibility Matrix: Supported Components

Pentaho aims to accommodate our clients' diverse computing environments. This list provides details about the environment components and versions we support. If you have questions about your particular computing environment, please contact Pentaho support.

Client

Pentaho client software is hardware-independent and runs on client-class computers that comply with these specifications for minimum hardware and required operation systems.

Pentaho Software
Pentaho Aggregation Designer
Pentaho Data Integration
Pentaho Design Studio
Pentaho Metadata Editor
Pentaho Report Designer
Pentaho Schema Workbench

Hardware—32 or 64 bit	Operating System—32 or 64 bit
Processors: • Apple Macintosh Dual-Core • Intel EM64T or AMD64 Dual-Core **RAM:** 2 GB RAM **Disk Space:** 2 GB free after installation	• Apple Macintosh OS 10.7 & 10.8 • Microsoft Windows 7 • Ubuntu Server 10.X and 12.X

Server

Pentaho server software is hardware-independent and runs on server-class computers that comply with these specifications for minimum hardware and required operation systems.

Pentaho Software
Pentaho Business Analysis Server
Pentaho Data Integration Server
Pentaho Enterprise Console

Hardware—64 bit	Operating System—64 bit
• Apple Macintosh Pro Quad-Core or Macintosh Mini Quad-Core • Intel EM64T or AMD64 Dual-Core **RAM:** 8 GB with 4 GB dedicated to Pentaho servers, 1 GB to Pentaho Enterprise Console **Disk Space:** 20 GB free after installation	• Apple Macintosh OS X Server 10.6 & 10.7 • CentOS Linux 5 & 6 • Microsoft Windows 2008 Server R1 & R2 • Red Hat Enterprise Linux 5 & 6 • Solaris 10 • Ubuntu Server 10.X & 12.X

Embedded Software

When embedding Pentaho software into other applications, the computing environment should comply with these specifications for minimum hardware and required operation systems.

Pentaho Software
Embedded Pentaho Reporting
Embedded Pentaho Analysis
Embedded Pentaho Data Integration

Hardware—32 or 64 bit	Operating System—32 or 64 bit
Processors: • Apple Macintosh Pro Quad-Core or Macintosh Mini Quad-Core • Intel EM64T or AMD64 Dual-Core **RAM:** 8 GB with 4 GB dedicated to Pentaho servers **Disk Space:** 20 GB free after installation	• Apple Macintosh OS X Server 10.6 & 10.7 • CentOS Linux 5 & 6 • Microsoft Windows 2008 Server R1 & R2 • Microsoft Windows 7 • Red Hat Enterprise Linux 5 & 6 • Solaris 10 • Ubuntu Server 10.X & 12.X

Application Servers

Servers to which you deploy Pentaho software must run one of these application servers.

Pentaho Software	Application Server
Pentaho Business Analysis Server	• Jboss 5.1.x • Tomcat 6.0.x
Pentaho Data Integration Server	Tomcat 6.0.x

Solution Database Repositories

Pentaho software stores processing artifacts in these solution database repositories.

Pentaho Software	Database Repository
Pentaho Business Analysis Server	• MySQL 5.x • Oracle 10g/11i • PostgreSQL 8.x & 9.1.x*
Pentaho Data Integration Server	Integrated Pentaho-specific, H2 1.2.131

*Default installed solution database

Data Sources

Pentaho software connects to these relational and non-relational data sources.

Pentaho Software	Data Source
Pentaho Reporting	• JDBC 3** • ODBC • OLAP4J • XML • Pentaho Analysis • Pentaho Data Integration • Pentaho Metadata
Pentaho Business Analysis Server, Action Sequences	• Relational (JDBC) • Hibernate • Javascript

Pentaho Software	Data Source
	• Metadata (MQL) • Mondrian (MDX) • XML (XQuery) • Security User/Role List Provider • Data Integration Steps (PDI) • Other Action Sequences • Web Services • XMLA

Pentaho Data Integration	• JDBC 3**
	• OLAP4J
	• Salesforce
	• XML
	• CSV
	• Microsoft Excel
	• Pentaho Analysis
	• Apache Hadoop 0.20.2 & 0.20.203.0***
	• Cloudera CDH3u4***
	• CDH4***
	• MapR 1.1.3 & 1.2.0
	
	• Cassandra distributions
	
	   • Apache 1.1.2
	   • DataStax 1.1.2
	• MongoDB 2.0.4

**Use a JDBC 3.x compliant driver that is compatible with SQL-92 standards when communicating with relational data sources. For your convenience, we provide a list of drivers used to get data from relational JDBC databases.

***From one of these distributions: HBase 0.90.5 and Hive 0.7.1

SQL Dialect-Specific

Pentaho software generates dialect-specific SQL when communicating with these data sources.

Pentaho Software	Data Source
Pentaho Analysis	• Access • DB2 • Derby • Firebird • Greenplum • Hive • Hsqldb • Infobright • Informix • Ingres • Interbase • LucidDb • MicrosoftSqlServer • MySql • Neoview • Netezza • Oracle • PostgreSQL

Pentaho Software	Data Source
	• SqlStream • Sybase • Teradata • Vectorwise • Vertica • Other SQL-92 compliant****
Pentaho Metadata	• DB2 • Firebird • H2 • Hypersonic • Ingres • MS Access • ASSQL • MSSQLNative • MySQL • Netezza • Oracle • PostgresSQL • Sybase • Other SQL-92 compliant****
Pentaho Data Integration	• Apache Derby • AS/400 • InfiniDB • Exasol 4 • Firebird SQL • Greenplum • H2 • Hypersonic • IBM DB2 • Infobright • Informix • Ingres • Ingres VectorWise • LucidDB • MaxDB (SAP DB) • MonetDB • MySQL • MS SQL Server • Neoview • Netezza • Oracle • Oracle RDB • PostgreSQL • SQLite • Teradata • UniVerse database • Vertica • Other SQL-92 compliant****

****If your data source is not in this list and is compatible with SQL-92, Pentaho software uses a generic SQL dialect.

Security

Pentaho software integrates with these third-party security authentication systems.

Pentaho Software	Authentication System
Pentaho Business Analysis Server Pentaho Enterprise Console	• Active Directory • CAS • Integrated Microsoft Windows Authentication • LDAP • RDBMS
Pentaho Data Integration Server	• Active Directory • LDAP • RDBMS

Java Virtual Machine

All Pentaho software, except the Pentaho Mobile App, requires the Sun/Oracle version 1.6 (6.0) distribution of the Java Runtime Environment (JRE) or Java Development Kit (JDK).

Web Browsers

Pentaho supports these major versions of Web browsers that are publicly available six weeks prior to when Pentaho begins to finalize a release. We also support the preceding major version.

Pentaho Software	Web Browser
Pentaho User Console Pentaho Enterprise Console Pentaho Report Designer*****	• Apple Safari 5.x • Google Chrome 19 • Microsoft Internet Explorer 8 & 9 • Mozilla Firefox 13 & 14

*****Requires a web browser to preview the exported HTML reports.

Mobile Apps

Pentaho mobile apps run on the Apple iPad 2 and 3 using iOS 5.x and 6.

JDBC Drivers

JDBC Drivers

This reference is a continuous work in progress. If you are viewing it in the Pentaho InfoCenter and see something that is not correct, know of a driver that is not listed here, or have a tip you want to share, please let us know by using the comments fields found in the bottom right corner.

Database	Vendor	URL
Apache Derby	IBM	*http://db.apache.org/derby/ derby_downloads.html*
Cache'	InterSystems	*http://www.cachemonitor.de/ intersystems-documentation/cache-jdbc-driver*
CUBRID	CUBRID	*http://www.cubrid.org/? mid=downloadsitem=jdbc_driver*
Daffodil DB	Daffodil Software	*http://sourceforge.net/projects/ daffodildb/*

Database	Vendor	URL
DB2 AS/400	IBM	*http://www-03.ibm.com/systems/i/ software/toolbox/*
DB2 Universal Database	IBM	*http://www-306.ibm.com/software/ data/db2/java*
Firebird	Firebird Foundation	*http://www.firebirdsql.org/en/jdbc-driver/*
FrontBase	FrontBase	*http://www.frontbase.com/cgi-bin/ WebObjects/FBWebSite*
Greenplum	EMC2	*http://jdbc.postgresql.org/ download.html*
H2 Database	H2	*http://www.h2database.com*
Hive	Apache	*http://hive.apache.org/*
HSQLDB	HyperSQL	*http://sourceforge.net/projects/hsqldb/*
Informix	IBM	*http://www-01.ibm.com/software/*
Ingres	Actian	*http://esd.actian.com/product/drivers/ JDBC/java*
InterBase	Embarcadero	*http://edn.embarcadero.com*
jTDS Free MS Sybase	jTDS	*http://jtds.sourceforge.net/*
LucidDB	DynamoDB	*http://www.dynamobi.com/c/ downloads/stable/*
MaxDB	SAP	*http://maxdb.sap.com*
Mckoi	Mckoi SQL Database	*http://www.mckoi.com/ originalmckoisql/index.html*
Mimer	Mimer Information Technology	*http://www.mimer.com*
MonetDB	MonetDB	*http://www.monetdb.org/*
MySQL	Oracle	*http://dev.mysql.com/downloads/ connector/j/*
Neoview	HP	*https://h20392.www2.hp.com/portal/ swdepot/displayProductInfo.do? productNumber=NEO10*
Netezza	IBM	*http://www.netezza.com*
OpenBase SQL	OpenBase International	*http://www.openbase.com/index.php/ products/downloads*
Oracle	Oracle	*http://www.oracle.com/technetwork/ database/features/jdbc/index.html*
Pervasive	Pervasive	*http://www.pervasivedb.com/ download/Pages/PDBDownloads. aspx*
PostgreSQL	PostgreSQL Global Development Group	*http://jdbc.postgresql.org/*
SAP DB	SAP DB	*http://www.sapdb.org/ sap_db_jdbc.htm*
SQLite	Xerial	*http://www.xerial.org/trac/Xerial/wiki/ SQLiteJDBC*

Database	Vendor	URL
SQL Server	Microsoft	*http://msdn.microsoft.com/en-us/ sqlserver/aa937724.aspx*
Sybase ASE	SAP	*http://www.sybase.com/products/ allproductsa-z/softwaredeveloperkit/ jconnect*
Sybase SQL Anywhere	SAP	*http://www.sybase.com/products/ allproductsa-z/softwaredeveloperkit/ jconnect*
SmallSQL	SmallSQL	*http://www.smallsql.de/download.html*
Teradata	Teradata	*http://downloads.teradata.com/ download/connectivity/jdbc-driver*
Vertica	HP	*http://www.vertica.com*

Apache Derby

Vendor Name	Details	
Recommended Native Driver		
IBM	**Company URL** *http://www.ibm.com*	
	Driver URL *http://db.apache.org/derby/derby_downloads.html*	
	JDBC URL Syntax by Type Server—jdbc:derby://<server>[:<port>]/ <databaseName>[;<URL attribute>=>value] Embedded—jdbc:derby:<databaseName>[;create=true]	**Default Port** 1527
	JDBC Class org.apache.derby.jdbc.ClientDriver org.apache.derby.jdbc.EmbeddedDriver	**JDBC JAR File Name** derby.jar
	Shipped with Pentaho Products Pentaho Data Integration	
	Comments Open source database	

Caché

Vendor Name	Details
Recommended Native Driver	
InterSystems	**Company URL** *http://www.cachemonitor.de*
	Driver URL *http://www.cachemonitor.de/intersystems-documentation/cache-jdbc-driver*

Vendor Name	Details	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:Cache:// <server>[:<port>]/<namespace>	1972
	JDBC Class	**JDBC JAR File Name**
	com.intersys.jdbc.CacheDriver	cachedb.jar

CUBRID

Vendor Name	Details	
Recommended Native Driver		
CUBRID	**Company URL**	
	http://www.cubrid.org	
	Driver URL	
	http://www.cubrid.org/?mid=downloads&item=jdbc_driver	
	JDBC URL Syntax by Type	**Default Port**
	Server— jdbc:cubrid:<server>:<port>:<databaseName>:<username>:<password> : [?<URL attribute>=<value>[&<URL attribute>=<value>] …]	33000
	JDBC Class	**JDBC JAR File Name**
	cubrid.jdbc.driver.CUBRIDDriver	N/A
	Comments	
	Open source database highly optimized for Web applications.	

Daffodil DB

Vendor Name	Details	
Recommended Native Driver		
Daffodil Software	**Company URL**	
	http://db.daffodilsw.com	
	Driver URL	
	http://sourceforge.net/projects/daffodildb/	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:daffodilDB:// <server>[:<port>]/<databaseName> Embedded— jdbc:daffodilDB_embedded:<databaseName>	3456 N/A
	JDBC Class	**JDBC JAR File Name**
	in.co.daffodil.db.rmi.RmiDaffodilDBDriver	DaffodilDB_client.jar DaffodilDB_Embedded.jar,

Vendor Name	Details
	in.co.daffodil.db.jdbc.DaffodilDBDriver	DaffodilDB_Common.jar
	Comments
	Open source database

DB2 AS/400

Vendor Name	Details	
Recommended Native Driver		
IBM	**Company URL** *http://www.ibm.com*	
	Driver URL *http://www-03.ibm.com/systems/i/software/toolbox/*	
	JDBC URL Syntax by Type Server—jdbc:as400:// <server>naming=sql;errors=full	**Default Port** N/A
	JDBC Class com.ibm.as400.access.AS400JDBCDriver	**JDBC JAR File Name** jt400.jar
	Shipped with Pentaho Products Pentaho Data Integration	

DB2 Universal Database

Vendor Name	Details	
Recommended Native Driver		
IBM	**Company URL** *http://www.ibm.com*	
	Driver URL *http://www-306.ibm.com/software/data/db2/java*	
	JDBC URL Syntax by Type Server—jdbc:db2://<server>[:<port>]/ <databaseName>[;<URL attribute>=<value>]	**Default Port** 50000
	JDBC Class com.frontbase.jdbc.FBJDriver	**JDBC JAR File Name** frontbasejdbc.jar

Firebird

Vendor Name	Details
Recommended Native Driver	
Firebird Foundation	**Company URL**

Vendor Name	Details	
	http://www.firebirdsql.org	
	Driver URL *http://www.firebirdsql.org/en/jdbc-driver/*	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:firebirdsql:<server>[/<port>]:/<database-file>	3050
	(JDBC Type 4, official format)	3050
	Server—jdbc:firebirdsql://<server>[:<port>]/<database-file>	3050
		3050
		N/A
	(JDBC Type 4, compatibility format)	
	Server—jdbc:firebirdsql:native//<server>[/<port>]:/<database-file>	
	(JDBC Type 2, compatibility format)	
	Server—jdbc:firebirdsql:native://<server>[:<port>]/<database-file>	
	JDBC Type 2, compatibility format. Requires libraries)	
	Embedded—jdbc:firebirdsql:embedded:/<local-database-file>	
	(JDBC Type 2, compatibility format. Requires libraries)	
	JDBC Class org.firebirdsql.jdbc.FBDriver	**JDBC JAR File Name** jaybird-full-xxx.jar
	Shipped with Pentaho Products Pentaho Data Integration	

FrontBase

Vendor Name	Details	
Recommended Native Driver		
FrontBase	**Company URL** *http://www.frontbase.com*	
	Driver URL *http://www.frontbase.com/cgi-bin/WebObjects/FBWebSite*	
	JDBC URL Syntax by Type Server—jdbc:FrontBase://<host>[:<port>]/<databaseName>	**Default Port** N/A
	JDBC Class	**JDBC JAR File Name**

Vendor Name	Details
	com.frontbase.jdbc.FBJDriver	frontbasejdbc.jar

Greenplum

Vendor Name	Details	
Recommended Native Driver		
Greenplum	**Company URL**	
	http://www.greenplum.com	
	Driver URL	
	http://jdbc.postgresql.org/download.html	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:postgresql://<server>[:<port>]/<databaseName>	5342
	JDBC Class	**JDBC JAR File Name**
	org.postgresql.Driver	postgresql-8.x-xxx.jdbc4.jar
	Comments	
	Greenplum uses the Postgresql JDBC driver	

H2 Database

Vendor Name	Details	
Recommended Native Driver		
H2	**Company URL**	
	http://www.h2database.com	
	Driver URL	
	http://www.h2database.com	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:h2:tcp://server[:port]/file-path	9092
	Embedded—jdbc:h2:file-name	N/A
	JDBC Class	**JDBC JAR File Name**
	jdbc:h2:tcp://server[:port]/file-path	h2-x.x.xxx.jar
	org.h2.Driver	
	Shipped with Pentaho Products	
	• Pentaho Business Analysis Server • Pentaho Data Integration • Pentaho Metadata-Editor Pentaho Report-Designer	
	Comments	

Vendor Name	Details
	Open source Java SQL database

Hive

Vendor Name	Details	
Recommended Native Driver		
Apache	**Company URL**	
	http://hive.apache.org/	
	Driver URL	
	N/A	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:hive://<server>[:<port>]/default	10000
	JDBC Class	**JDBC JAR File Name**
	org.apache.hadoop.hive.jdbc.HiveDriver	hive-jdbc-x.x.x-pentaho-y.y.y.jar
		Example: hive-jdbc-0.7.0-pentaho-1.0.1.jar*
	Shipped with Pentaho Products	
	• Pentaho Business Analysis Server • Pentaho Data Integration • Pentaho Metadata-Editor Pentaho Report-Designer	
	Comments	
	Data warehouse infrastructure that provides data summarization and ad hoc querying	
	*x.x.x is the Hive version, y.y.y is the Pentaho version. Pentaho has enhanced the standard Hive JDBC driver	

HSQLDB

Vendor Name	Details	
Recommended Native Driver		
HyperSQL	**Company URL**	
	http://www.hsqldb.org	
	Driver URL	
	http://sourceforge.net/projects/hsqldb/	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:hsqldb:hsql://<server>[:<port>]/<databaseName>	9001
		N/A
	Embedded Memory—jdbc:hsqldb:mem:<databaseName>	N/A

Vendor Name	Details	
	Embedded File — jdbc:hsqldb:file:<database-file>	
	JDBC Class	**JDBC JAR File Name**
	org.hsqldb.jdbcDriver	hsqldb.jar
	Shipped with Pentaho Products	
	• Pentaho Enterprise-Console • Pentaho Business Analysis Server • Pentaho Aggregation-Designer • Pentaho Data Integration • Pentaho Metadata-Editor • Pentaho Report-Designer	

Informix

Vendor Name	Details	
Recommended Native Driver		
IBM	**Company URL**	
	http://www.ibm.com	
	Driver URL	
	=	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:informix-sqli://<server>[:<port>]/<databaseName>:informixserver=<dbservername>	1533
	JDBC Class	**JDBC JAR File Name**
	com.informix.jdbc.IfxDriver	ifxjdbc.jar
	Shipped with Pentaho Products	
	Pentaho Data Integration	

Ingres

Vendor Name	Details	
Recommended Native Driver		
Actian	**Company URL**	
	http://www.actian.com/	
	Driver URL	
	http://esd.actian.com/product/drivers/JDBC/java	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:ingres://<server>[:<port>]/<databaseName>	21071
	JDBC Class	**JDBC JAR File Name**

Vendor Name	Details	
	com.ingres.jdbc.IngresDriver	iijdbc.jar
	Comments	
	Open source relational database management system	

InterBase

Vendor Name	Details	
Recommended Native Driver		
Embarcadero	**Company URL**	
	http://edn.embarcadero.com	
	Driver URL	
	N/A	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:interbase://<server>/<full_db_path>	N/A
	JDBC Class	JDBC JAR File Name
	interbase.interclient.Driver	interclient.jar
	Shipped with Pentaho Products	
	Pentaho Data Integration	

jTDS Free MS SQL Sybase

Vendor Name	Details	
Recommended Native Driver		
jTDS	**Company URL**	
	http://jtds.sourceforge.net/	
	Driver URL	
	N/A	
	JDBC URL Syntax by Type	**Default Port**
	SQL Server—jdbc:jtds:<server_type>//<server>[:<port>][/<database>][;<property>=<value>[;...]]]	1433 7100
	Sybase—jdbc:jtds:<server_type>://<server>[:<port>][/<database>]	
	JDBC Class	**JDBC JAR File Name**
	interbase.interclient.Driver	jtds-x.x.x.jar
	Shipped with Pentaho Products	
	• Pentaho Enterprise-Console • Pentaho Business Analysis Server	

Vendor Name	Details
	• Pentaho Aggregation-Designer • Pentaho Data Integration • Pentaho Metadata-Editor • Pentaho Report-Designer

LucidDB

Vendor Name	Details
Recommended Native Driver	
DynamoDB	**Company URL** *http://www.dynamobi.com*
	Driver URL *http://www.dynamobi.com/c/downloads/stable/*
	JDBC URL Syntax by Type Server—jdbc:luciddb:http:// <server>[:<port>]
	Default Port 8034
	JDBC Class org.luciddb.jdbc.LucidDbClientDriver
	JDBC JAR File Name LucidDbClient-x.x.x.jar
	Shipped with Pentaho Products Pentaho Data Integration
	Comments **Open source BI solution for Big Data**

Vendor Name	Details	
Recommended Native Driver		
DynamoDB	**Company URL** *http://www.dynamobi.com*	
	Driver URL *http://www.dynamobi.com/c/downloads/stable/*	
	JDBC URL Syntax by Type Server—jdbc:luciddb:http:// <server>[:<port>]	**Default Port** 8034
	JDBC Class org.luciddb.jdbc.LucidDbClientDriver	**JDBC JAR File Name** LucidDbClient-x.x.x.jar
	Shipped with Pentaho Products Pentaho Data Integration	
	Comments **Open source BI solution for Big Data**	

MaxDB

Vendor Name	Details	
Recommended Native Driver		
SAP	**Company URL** *http://www.sap.com*	
	Driver URL *http://maxdb.sap.com*	
	JDBC URL Syntax by Type Server—jdbc:sapdb:// <server>[:<port>]/<databaseName>	**Default Port** 7210
	JDBC Class com.sap.dbtech.jdbc.DriverSapDB	**JDBC JAR File Name** sapdbc.jar
	Comments Database management system developed and supported by SAP AG	

Mckoi SQL Database

Vendor Name	Details	
Recommended Native Driver		
Mckoi	**Company URL**	
	http://www.mckoi.com	
	Driver URL	
	http://www.mckoi.com/originalmckoisql/index.html	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:mckoi:// <server>[:<port>][/<schema>]/	9157
	JDBC Class	**JDBC JAR File Name**
	com.mckoi.JDBCDriver	mckoidb.jar
	Comments	
	Open source SQL database written in Java	

Mimer

Vendor Name	Details	
Recommended Native Driver		
Mimer Information Technology	**Company URL**	
	http://www.mimer.com	
	Driver URL	
	N/A	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:mimer:<protocol>:// <server>[:<port>]/<database>	1360
	JDBC Class	**JDBC JAR File Name**
	com.mimer.jdbc.Driver	mimer.jar

MonetDB

Vendor Name	Details	
Recommended Native Driver		
MonetDB	**Company URL**	
	http://www.monetdb.org	
	Driver URL	
	N/A	
	JDBC URL Syntax by Type	**Default Port**
		50000

Vendor Name	Details	
	Server—jdbc:monetdb:// <server>[:<port>]/<database>	
	JDBC Class nl.cwi.monetdb.jdbc.MonetDriver	**JDBC JAR File Name** monetdb-jdbc-x.x.jar
	Shipped with Pentaho Products Pentaho Data Integration	
	Comments An open source database system	

MY SQL

Vendor Name	Details	
Recommended Native Driver		
Oracle	**Company URL** *http://www.mysql.com*	
	Driver URL *http://dev.mysql.com/downloads/connector/j/*	
	JDBC URL Syntax by Type Server—jdbc:mysql:// <hostname>[,<failoverhost>] [:<port>] /<dbname>[?<URL attribute>=<value>[&<URL attribute>=<value>] ...]	**Default Port** 3306
	JDBC Class com.mysql.jdbc.Driver (official class name) org.gjt.mm.mysql.Driver (older class name)	**JDBC JAR File Name** mysql-connector-java-5.x.xx-bin.jar
	Shipped with Pentaho Product • Pentaho Enterprise-Console • Pentaho Business Analysis Server • Pentaho Aggregation-Designer • Pentaho Metadata-Editor • Pentaho Report-Designer	

Neoview

Vendor Name	Details
Recommended Native Driver	
HP	**Company URL** *http://www.hp.com*

Vendor Name	Details	
	Driver URL	
	https://h20392.www2.hp.com/portal/swdepot/displayProductInfo.do?productNumber=NEO10	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:hpt4jdbc://<system>[:<port>]/[:][<URL attribute>=<value>[;<URL attribute>=<value> …]	18650
	JDBC Class	**JDBC JAR File Name**
	com.hp.t4jdbc.HPT4Driver	N/A

Netezza

Vendor Name	Details	
Recommended Native Driver		
IBM	**Company URL**	
	http://www.netezza.com	
	Driver URL	
	N/A	
	JDBC URL Syntax by Type	**Default Port**
	jdbc:netezza://<server>[:<port>]/<database>	5480
	JDBC Class	**JDBC JAR File Name**
	org.netezza.Driver	N/A

OpenBase SQL

Vendor Name	Details	
Recommended Native Driver		
OpenBase International	**Company URL**	
	http://www.openbase.com	
	Driver URL	
	http://www.openbase.com/index.php/products/downloads	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:openbase://<server>/<databaseName>	N/A
	JDBC Class	**JDBC JAR File Name**
	com.openbase.jdbc.ObDriver	OpenBaseJDBC.jar

Oracle

Vendor Name	Details	
Recommended Native Driver		
Oracle	**Company URL**	
	http://www.oracle.com	
	Driver URL	
	http://www.oracle.com/technetwork/database/features/jdbc/index.html	
	JDBC URL Syntax by Type	**Default Port**
	Thin Server—jdbc:oracle:thin:@<server>[:<port>]:<sid>	1521
	OCI Server—jdbc:oracle:oci:@<server>[:<port>]:<sid>	
	JDBC Class	**JDBC JAR File Name**
	oracle.jdbc.driver.OracleDriver	ojdbcx.jar, ora18n.jar
	oracle.jdbc.OracleDriver	
	Comments	
	The OCI server requires OCI libraries	

Pervasive

Vendor Name	Details	
Recommended Native Driver		
Pervasive	**Company URL**	
	http://www.pervasivedb.com/Pages/default.aspx	
	Driver URL	
	http://www.pervasivedb.com/download/Pages/PDBDownloads.aspx	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:pervasive://<server>[:<port>]/<datasource>	1583
	JDBC Class	**JDBC JAR File Name**
	com.pervasive.jdbc.v2.Driver	N/A
	Shipped with Pentaho Products	
	• Pentaho Data Integration • Pentaho Report-Designer	
	Comments	
	The data source is the ODBC DSN	

PostgreSQL

Vendor Name	Details	
Recommended Native Driver		
PostgreSQL Global Development Group	**Company URL** *http://www.postgresql.org/*	
	Driver URL *http://jdbc.postgresql.org/*	
	JDBC URL Syntax by Type Server—jdbc:postgresql:// <server>[:<port>]/<databaseName>	**Default Port** 5342
	JDBC Class org.postgresql.Driver	**JDBC JAR File Name** postgresql-8.x-xxx.jdbc4.jar
	Shipped with Pentaho Products • Pentaho Data Integration • Pentaho Report-Designer	

SAP DB

Vendor Name	Details	
Recommended Native Driver		
SAP DB	**Company URL** N/A	
	Driver URL *http://www.sapdb.org/sap_db_jdbc.htm*	
	JDBC URL Syntax by Type Server—jdbc:sapdb://<server>/ <database_name>	**Default Port** N/A
	JDBC Class com.sap.dbtech.jdbc.DriverSapDB	**JDBC JAR File Name** sapdbc-x.x.x.jar
	Shipped with Pentaho Products Pentaho Data Integration	
	Comments FREE Enterprise Open Source Database	

SQLite

Vendor Name	Details
Recommended Native Driver	
Xerial	**Company URL**

Vendor Name	Details	
	N/A	
	Driver URL	
	http://www.xerial.org/trac/Xerial/wiki/SQLiteJDBC	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:sqlite:<filename.db>	N/A
	JDBC Class	**JDBC JAR File Name**
	org.sqlite.JDBC	sqlite-jdbc-x.x.x.jar
	Shipped with Pentaho Products	
	Pentaho Data Integration	

SQL Server

Vendor Name	Details	
Recommended Native Driver		
Microsoft	**Company URL**	
	http://www.microsoft.com	
	Driver URL	
	http://msdn.microsoft.com/en-us/sqlserver/aa937724.aspx	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:sqlserver://<server>[:<port>];DatabaseName=<databaseName>	1433
	JDBC Class	**JDBC JAR File Name**
	com.microsoft.sqlserver.jdbc.SQLServerDriver	sqljdbc4.jar
	Comments	
	The open source jtds driver also works with MSSQL	

Sybase ASE

Vendor Name	Details	
Recommended Native Driver		
SAP	**Company URL**	
	http://www.sybase.com	
	Driver URL	
	http://www.sybase.com/products/allproductsa-z/softwaredeveloperkit/jconnect	
	JDBC URL Syntax by Type	**Default Port**
	Server—jdbc:sybase:Tds:<server>[:<port>]/<databaseName>	5000

Vendor Name	Details
	JDBC Class com.sybase.jdbc4.jdbc.SybDriver	**JDBC JAR File Name** N/A
	Comments The open source jTDS driver works with Sybase as well	

Sybase SQL Anywhere

Vendor Name	Details
Recommended Native Driver	
SAP	**Company URL** *http://www.sybase.com*
	Driver URL *http://www.sybase.com/products/allproductsa-z/softwaredeveloperkit/jconnect*
	JDBC URL Syntax by Type Server— jdbc:sybase:Tds:<server>[:<port>]/ <databaseName>	**Default Port** 2638
	JDBC Class com.sybase.jdbc4.jdbc.SybDriver	**JDBC JAR File Name** N/A
	Comments This open source jTDS driver works with Sybase as well	

SmallSQL

Vendor Name	Details
Recommended Native Driver	
SmallSQL	**Company URL** *http://www.smallsql.de/*
	Driver URL *http://www.smallsql.de/download.html*
	JDBC URL Syntax by Type Embedded— jdbc:smallsql:databaseName[?URL attribute=value[URLattribute=value] …]	**Default Port** N/A
	JDBC Class smallsql.database.SSDriver	**JDBC JAR File Name** smallsql.jar
	Comments	

Vendor Name	Details
	Java desktop SQL database engine

Teradata

Vendor Name		Details	
Recommended Native Driver			
Teradata	**Company URL**		
	http://www.teradata.com		
	Driver URL		
	http://downloads.teradata.com/download/connectivity/jdbc-driver		
	JDBC URL Syntax by Type	**Default Port**	
	Server—jdbc:teradata://<dbshost>[/ <URL attribute>[;<URL attribute>]…]	N/A	
	JDBC Class	**JDBC JAR File Name**	
	com.teradata.jdbc.TeraDriver	terajdbc4.jar	

Vertica

Vendor Name		Details	
Recommended Native Driver			
HP	**Company URL**		
	http://www.vertica.com		
	Driver URL		
	TBD		
	JDBC URL Syntax by Type	**Default Port**	
	Server—jdbc:vertica:// <server>[:<port>]/<databaseName>	5433	
	JDBC Class	**JDBC JAR File Name**	
	com.vertica.Driver	N/A	

Troubleshooting

This section contains known problems and solutions relating to the procedures covered in this guide.

Version Check

The instructions in this guide are specific to the Pentaho BA Server Enterprise Edition version 4.8-GA. The installation process can change significantly between BA Server releases to address new features, updated requirements, and bug workarounds, so the instructions in this guide should be assumed not to work with any other BA Server version, including the open source BA Server Community Edition.

Examining Log Files

First of all, you should check the build log (if you opted to create one) to verify that all of the build options you specified were properly executed, and that the WAR package was created without any problems.

If the BA Server fails to start or work properly, the log file you should consult is **pentaho.log** in the `/pentaho/server/biserver-ee/tomcat/bin/` directory. The contents of this file will assist you in tracking down the problem. For JBoss, the pentaho.log resides in the JBoss folder structure under the /logs/ directory.

File Names and Paths

> **Note:** This is the most common installation problem.

Many of the configuration files and paths in this guide are similar, and it is easy to confuse them, which could result in modifying the wrong files or copying to the wrong locations. Double-check your file names and paths and ensure that you've copied all of the right files to all of the correct directories.

Trailing slashes are important; both their inclusion and their absence, depending on the file and parameter or element you are modifying. Follow the examples in this guide exactly unless otherwise directed.

JDBC Driver Problems

First, ensure that the correct JDBC driver JARs are installed to the correct locations, then check to make sure that there aren't conflicting driver versions. See the *Pentaho Business Analytics Administrator's Guide* for explanations of how to create JDBC Database connections and driver locations for all parts of Pentaho Business Analytics. Check with your database or driver vendor if you suspect you have having JDBC driver compatibility issues.

Licenses Not Found After Installation

If you've successfully installed the BA Server and Pentaho Enterprise Console, then installed licenses, and are later unable to access Dashboard Designer, Pentaho Analyzer, or Pentaho Interactive Reporting through the Pentaho User Console due to missing licenses, then you must either reinstall those licenses under the user account that will always start the Pentaho Enterprise Console and BA Server services, or you must set a Java virtual machine parameter to point to a static license path. This will override the default license path, which changes depending on which user started the Pentaho Enterprise Console and BA Server services or servers. The parameter is appended to the Tomcat or JBoss service execution commands, and to the Pentaho Enterprise Console **start-pec** script. Examples are shown below:

Windows Tomcat service adjustment command

```
tomcat5 //US//pentahobiserver ++JvmOptions "-Dpentaho.installed.licenses.file=C:
\Pentaho\.installedLicenses.xml"
```

Linux start-pec.sh script snippet

```
"$_PENTAHO_JAVA" -Dpentaho.installed.licenses.file="/home/
pentaho/.pentaho/.installedLicenses.xml" -Xmx512M -XX:PermSize=64M -
XX:MaxPermSize=128M -Djava.awt.headless=true -DCONSOLE_HOME=$DIR_REL
 -Dlog4j.configuration=resource/config/log4j.xml -cp $CLASSPATH
 com.pentaho.pac.server.ProJettyServer
```

Cannot Create Hibernate Tables in MySQL

The Pentaho solution repository uses long text strings that require a longer maximum character limit than the default UTF-8 configuration allows. Therefore, if your MySQL character set is configured to use UTF-8, you must change it to **ASCII** instead in order to use it as a Pentaho solution database. Using UTF-8 will prevent the MySQL initialization scripts from running during installation.

Unable to Use the Database Init Scripts for PostgreSQL

The **pg_hba.conf** file contains host-based authentication information. If you can't run the SQL scripts that generate the Hibernate and Quartz databases, it's probably because the default user accounts for each database don't have the right permissions. To change this, edit the file to ensure that connections from local users created by the Pentaho sql scripts (**hibuser** and **pentaho_user**) will be able to connect. The default on Debian-based systems is for local connections yo use **ident** authentication, which means that database users must have local user accounts. In other words, to continue using **ident**, you would have to create local **hibuser** and **pentaho_user** accounts. It's easier to just change the authentication method to something less restrictive, if your IT manager permits you to do so.

context.xml Changes Do Not Take Effect After Re-deploying a WAR

Re-deployment of a WAR or EAR with a custom **context.xml** may cause the original context.xml that you deployed with the original WAR or EAR to become permanently cached. Tomcat in particular will generate a WAR-specific context configuration file, and keep it in place even after the WAR is deleted. The location and naming convention for this file are: **$CATALINA_HOME/conf/Catalina/<host>/<war name>.xml**. Typically this will be something like: **/tomcat/conf/Catalina/localhost/pentaho.xml**. If this file exists, you will have to delete it prior to re-deploying pentaho.war if you have made any changes to context.xml.

vfs-provider.xml Duplicates

The above-referenced configuration file may be present in a number of JARs in other applications that you've deployed to your Java application server. Having multiple instances of this file will cause classpath exceptions. You must merge the multiple files into one canonical edition in order to solve the problem.

Library Conflicts

The BA Server relies on many third-party libraries that provide everything from database connectivity to specific Java classes that add necessary features to the BA Server. If you have incompatible versions of any of these third-party libraries in your application server's global lib directory, they can cause a variety of problems related to starting and running the BA Server. You will have to discover and individually canonicalize these files according to your needs.

Some known-problematic JARs are:

- commons-collections-3.2.jar (from Pentaho)
- commons-collections.jar (from JBoss in `/jboss/server/default/lib/`)
- jettison-1.01.jar (from Pentaho)
- jettison.jar (from JBoss in `/jboss/default/deploy/jbossws.sar`)

Varying Context and Data Source Configuration Methods

The instructions in this guide direct you to edit a **context.xml** override for Tomcat and build it into the WAR. Your application server may not support this method, or may not accept these files by default. This will be especially true if you are using unsupported application servers or versions, or if you are on Linux and your distribution provider alters the default package configuration for Tomcat.

There are other methods of changing context settings, especially through other kinds of configuration files such as **server.xml** or application-specific XML files that your application server creates for each deployed WAR. If the context and data source instructions in this guide do not work for you, consult your application server's documentation to learn the preferred method of configuration, and adapt the Pentaho-supplied process to accommodate it.

Report Parameters That Include Accented Characters Fail to Validate

If you run a report that has parameters that include accented characters and you see an error message that says "This parameter value is of an invalid value," then you must make the Tomcat server modification explained in this task: *Modifying server.xml To Work With Accented Characters* on page 27.

JBoss Fails to Start When the Pentaho HSQLDB Sample Database Is Running

Note: This problem can also manifest as the Pentaho sample database refusing to start when the BA Server is deployed to JBoss.

The Pentaho-supplied HSQLDB sample database operates on the default HSQLDB port of 9001. JBoss has its own HSQLDB instance running on the same port; therefore, the port collision will prevent the JBoss version from starting, and cause the startup process to halt. You can change the Pentaho sample database port by editing the **start_hypersonic** script and adding the **-port 9002** switch to the last line:

```
"$_PENTAHO_JAVA" -cp $THE_CLASSPATH org.hsqldb.Server -port 9002 -database.0 $DIR_REL/
hsqldb/sampledata -dbname.0 sampledata -database.1 $DIR_REL/hsqldb/hibernate -dbname.1
 hibernate -database.2 $DIR_REL/hsqldb/quartz -dbname.2 quartz
```

JBoss Fails to Start After Manually Unpacking pentaho.war

If you unpack the pentaho.war file by hand, you must name the resultant directory **pentaho.war** as well. If you unpack it to any other directory name, including "pentaho" without the .war extension, JBoss will fail to deploy the WAR without any meaningful warnings.

Web-App Path doesn't validate in Enterprise Console

If you are deploying to JBoss and don't unpack the pentaho.war into a directory, the Pentaho Enterprise Console will not be able to find the correct Web-App Path value. To fix the problem, unpack the pentaho.war to a directory of the same name.

Action Sequences That Call PDI Content Won't Run

If you've established an enterprise repository in PDI to store your jobs and transformations, and you attempt to use that stored PDI content in an action sequence on the BA Server, the action sequence will not execute. This is because the BA Server needs specific connection information for the Data Integration (DI) Server in order to retrieve the job or transformation.

Adding PDI Enterprise Repository Content Support to the BA Server

If you are using a Pentaho Data Integration (PDI) enterprise repository (through a Data Integration Server) to store PDI jobs and transformations, and you plan on using those jobs and transformations in action sequences that will be run on

the BA Server, you must install some BA Server plugins from the PDI client tool package. This is not a typical scenario, but there is no harm in performing it if you aren't sure of the details.

1. Download a PDI Enterprise Edition 4.4 client tool archive package from the Pentaho Customer Support Portal.

   The package name (available in both tar.gz and zip formats) is: **pdi-ee-client-4.4.0-GA**

2. Unpack the archive to a temporary location.

3. Edit the `/pentaho/server/biserver-ee/pentaho-solutions/system/kettle/settings.xml` file.

4. Change the value of the **<repository.type>** node from **files** to **rdbms**.

5. Enter your enterprise repository connection information in the proper nodes.

6. Enter the location of your local **repositories.xml** file in the **<repositories.xml.file>** node.

   > **Note:** This file is created on your PDI client workstation when you establish a connection to an enterprise repository. Once you have made all of your repository connections on a workstation, copy the **repositories.xml** file to the `~/.kettle/` directory on the BA Server and DI Server machines. If the client tool and servers are all on the same machine, you do not have to copy the file. If you have not yet established any repositories, you will have to revisit this procedure later when your PDI environment is fully configured.

7. Copy the contents of `/data-integration/plugins/` to the `/pentaho/server/biserver-ee/pentaho-solutions/system/kettle/plugins/` directory.

   ```
   cp -r /tmp/data-integration/plugins/* /home/pentaho/pentaho/server/biserver-ee/
   pentaho-solutions/system/kettle/plugins/
   ```

8. Remove the unpacked archive.

   ```
   rm -rf /tmp/data-integration/
   ```

Your BA Server is now configured to run content stored in the DI Server.

Mac OSX Error When Running Apps

When you run a Pentaho App file on OSX Mountain Lion, you may get a warning message from Gatekeeper. Temporarily changing security settings enables you to open apps successfully. For more information, see *OS X: About Gatekeeper*.

1. **Go to System Preferences** > **Security** > **Privacy** > **General** > **Allow applications downloaded from**.

2. Change the setting to **Anywhere**.

3. Double-click the app.

   A message box appears indicating this app is not signed.

4. Click **Open** to run the app.

5. Reset the security settings to what they originally were.
   The next time you run this app, the message box does not appear and the app runs normally.

Appendix: Installation Notes For FreeBSD

It is possible to install Pentaho Business Analytics onto a FreeBSD machine. However, you should take into account the following notes:

- **Linux binary compatibility is required for running the Pentaho client tools.** Install the Linux binary compatibility portion of the base system through sysinstall; load the Linux binary compatibility kernel module by adding this line to `/etc/loader.conf` and then restarting the computer:

```
linux_load="YES"
```

  You must also install the **linux_base-fc4** package or Port.
- **You must set the JAVA_HOME environment variable in your shell configuration file.** For the root account, edit `~/.cshrc` and add this line to the end of the file:

```
setenv JAVA_HOME /usr/local/diablo-jre1.6.0/
```

  For user accounts, you would typically edit `~/.profile` instead, and add these lines:

```
JAVA_HOME=/usr/local/diablo-jre1.6.0/
export JAVA_HOME
```

  Re-log for the environment changes to take effect.
- **Pentaho does not perform validation testing on FreeBSD.** While there is no reason to believe that any part of Business Analytics will not work on FreeBSD, Pentaho does not currently test on this platform, and therefore will not provide the same level of support that it does for Linux, Solaris, and Windows. Only basic, unofficial testing has been done on FreeBSD. Even these FreeBSD installation tips are not guaranteed to work.
- **Pentaho has only verified basic operation on FreeBSD.** No testing was done on versions of FreeBSD that are old or in development.
- **The Linux installer will not work.** The Business Analytics installation utility contains Linux-specific commands that will not work properly on FreeBSD, even with all of the standard binary compatibility options in place. The most reliable installation method is to install Tomcat 6 and either MySQL or PostgreSQL through Ports (or pkg_add), then use the manual deployment installation method to build and deploy a Pentaho WAR. The archive-based installation, which includes a version of Tomcat compiled for Linux, seems to work reasonably well, but if you want to run the BA Server as a service you will have to design your own RC scripts to start and stop it at boot time.
- **You must start Pentaho Enterprise Console manually.** While Tomcat and MySQL or PostgreSQL will start through the RC scripts set up by Ports, Pentaho Enterprise Console runs on a standalone Jetty server. The PEC startup script is `/pentaho/server/enterprise-console/start.sh`.
- **diablo-jre-16 should work.** Pentaho tests and validates its software with the Sun Java Runtime Environment version 1.6.0, 32-bit and 64-bit. The FreeBSD Diablo Latte JRE and Caffe JDK packages are produced from Oracle (Sun) source code and have passed Oracle's validation testing, however Pentaho does not do any of its own testing on this platform and cannot guarantee compatibility.