Pentaho BI 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....5
Installation Checklist....6
Prerequisites....8
Hardware Requirements....8
Software Requirements....8
Important Note on Local User Accounts For Linux/Unix....9
How to Check Your Java Version....10
Setting the PENTAHO_JAVA_HOME Variable on Linux....11
Setting the PENTAHO_JAVA_HOME Variable on Windows....11
Setting the PENTAHO_INSTALLED_LICENSE_PATH Variable on Linux....11
Setting the PENTAHO_INSTALLED_LICENSE_PATH Variable on Windows....12
Preparing a Headless Linux or Solaris Server....12
Obtaining the Pentaho BI Server Build Materials....14
Building the BI Server....15
Establishing a context.xml File For Tomcat....15
Solution Repository JNDI Configuration Files for JBoss Deployments....16
Changing the Web Application Context and Default Port Number....16
Changing the Web Application Name on Tomcat....16
Changing the Web Application Name on JBoss....17
Creating a Pentaho Directory....17
Copying the Oracle Quartz JAR Pre-Build....18
How to Build BI Server WARs with Apache Ant....18
Ant Build Options....19
Deploying the BI Server....21
Copying Solution Database JDBC Drivers....21
Moving the Pentaho Solutions Directory....21
Initializing a PostgreSQL Database....21
PostgreSQL Solution Repository Configuration....22
Configuring Quartz For PostgreSQL....22
Initializing a MySQL Database....23
Initializing an Oracle Database....23
Oracle Solution Repository Configuration....23
Configuring Quartz For Oracle....24
Deploying to Tomcat 6.0....24
Modifying startup.sh on Linux and Solaris....25
Modifying startup.bat on Windows....25
Modifying server.xml To Work With Accented Characters....25
Deploying to JBoss 5.1....26
Modifying run.conf on Linux and Solaris....26
Modifying run.bat on Windows....26
Deploying a WAR to JBoss 5.1....27

Adding PDI Enterprise Repository Content Support to the BI Server....28
Installing Hadoop Hive Plugins....29
Installing Pentaho User Console Plugins....30
Installing the Pentaho Enterprise Console: Linux/OS X/Windows....31
Installing the Pentaho Enterprise Console: Solaris....32
Initial Startup of the BI Server and Pentaho Enterprise Console....33
Post-Install Configuration....34
Testing and Using Your Server....35
Preparing for Production....36
Switching to a Production Login Screen....36
Troubleshooting....37
Version Check....37
Examining Log Files....37
File Names and Paths....37
JDBC Driver Problems....37
Licenses Not Found After Installation....37
Cannot Create Hibernate Tables in MySQL....38
Unable to Use the Database Init Scripts for PostgreSQL....38
context.xml Changes Do Not Take Effect After Re-deploying a WAR....38
vfs-provider.xml Duplicates....38
Library Conflicts....39
Varying Context and Data Source Configuration Methods....39
Report Parameters That Include Accented Characters Fail to Validate....39
JBoss Fails to Start When the Pentaho HSQLDB Sample Database Is Running....39
JBoss Fails to Start After Manually Unpacking pentaho.war....40
Web-App Path doesn't validate in Enterprise Console....40
Action Sequences That Call PDI Content Won't Run....40
Adding PDI Enterprise Repository Content Support to the BI Server....40

Manual Deployment Overview

Pentaho provides several installation paths to meet a variety of customer needs. This guide contains advice and step-by-step instructions for performing a complete custom installation of the Pentaho BI Server Enterprise Edition 3.9 into a development or production environment that has an existing Java application server and relational database that you want to store Pentaho content in.

Note: This is the only installation procedure that Pentaho will support for single sign-on (CAS) configuration.

Will Be Installed as Part of This Process	You Must Supply On Your Own
A JBoss- or Tomcat-compatible J2EE application archive that contains the Pentaho BI Server **Archive packages containing BI Server plugins:** • Analyzer • Dashboard Designer • Interactive Reporting **A Jetty application server running the Pentaho Enterprise Console**	**A supported operating system:** • Linux (Red Hat Enterprise Linux 5, SUSE Linux Enterprise Linux 10) • Windows (XP, Vista, 2008, 7) • Solaris 10 • OS X (10.5 or newer) **Build tools:** • A Sun Java Development Kit (JDK) version 1.6.0 (6.0) • Apache Ant **A Sun JRE 1.6.0** on the server, and on each workstation that you will install data preparation or design tools to **A supported application server:** • Tomcat 6.0 • JBoss 5.1 **A supported database to store solutions:** • MySQL 5.0 • PostgreSQL 8.3 • Oracle 10g or 11g **One or more data sources:** • Any JDBC-compliant database • A spreadsheet • A flat file containing comma-separated values

This may not be the best installation path for you. This procedure expects you to provide a database to store your BI Suite solutions, content, and schedules; and a Java application server to run the BI Server Web application. If you would prefer not to build your own WAR file and install an application server, consult the *Pentaho BI Suite Archive-Based Installation Guide*.

Note: The example commands in this guide are relevant for Unix-like operating systems -- Linux, Solaris, OS X, etc. -- except in sections where there are separate instructions for Windows. If you are deploying to Windows, adjust the examples accordingly for your environment.

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, BI Server J2EE "manual deployment" build package, and any BI Server plugins (Analyzer, Dashboard Designer, Interactive Reporting, etc.) from the Enterprise Edition FTP site or Pentaho Knowledge Base. 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 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 10	Using Apache Ant, build the Pentaho WAR with the appropriate options for your environment. `ant -Darchive.target=war-pentaho-jboss -Ddb=oracle10g`	

Step	Procedure	Done
Step 11	Copy the solution database JDBC driver to the `/tomcat/lib/` directory for Tomcat, or the `/jboss/server/default/lib/` directory for JBoss.	
Step 12	Move the **pentaho-solutions** directory from `/biserver-manual-ee/build/` to `/pentaho/server/biserver-ee/`	
Step 13	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 14	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 15	Copy the Pentaho WAR to your application server's **deploy** or **webapps** directory.	
Step 16	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 17	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 18	If you have a Pentaho for Hadoop license, unpack the Hadoop Hive plugins to the `/pentaho-solutions/system/kettle/plugins/` directory.	
Step 19	Unpack any BI Server plugin packages that you have licenses for to the `/pentaho-solutions/system/` directory.	
Step 20	Unpack the Pentaho Enterprise Console archive package to the `/pentaho/server/` directory.	
Step 21	Start the application server you deployed the BI 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 22	Install license keys; establish users, roles, data sources; and perform any remaining configuration tasks. Use the *Pentaho BI Suite Administrator's Guide* for assistance.	
Step 23	Access the BI 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 the Web server, Java application server, database, and firewall configuration for your organization's network. Specifically you need to know the IP addresses of each relevant server, and which ports all of the necessary services use.

Building the BI 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 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. If you don't have the requisite access or permissions, having the cooperation of someone who does (the sysadmin or database administrator) is probably good enough.

Hardware Requirements

The Pentaho BI Suite does not have strict limits on computer or network hardware. As long as you meet the minimum software requirements (note that your operating system will have its own minimum hardware requirements), Pentaho is hardware agnostic. There is, however, a recommended set of minimum system specifications:

Server:

RAM	At least 2GB
Hard drive space	At least 2GB for the software, and more for solution and content files
Processor	Dual-core AMD64 or Intel EM64T

Workstation:

RAM	At least 1GB
Hard drive space	At least 1GB for the software, and more for solution and content files
Processor	Dual-core AMD64 or Intel EM64T

It's possible to use less capable machines, but in most realistic scenarios, the too-limited system resources will result in an undesirable level of performance.

Your environment does not have to be 64-bit, even if your processor architecture supports it; while all modern desktop, workstation, and server machines have 64-bit processors, they often ship by default with 32-bit operating systems. If you want to run the Pentaho BI Suite in a pure 64-bit environment, you will have to install a 64-bit operating system, ensure that your solution database and Java Runtime Environment are 64-bit, and install the BI Suite via the 64-bit graphical installer, or through the archive-based or manual deployment methods.

Note: A 32-bit JRE has a hard memory limit of 2GB (1.5GB on Windows), so if you have 2GB or more of RAM, you must use a 64-bit JRE on a 64-bit operating system to take full advantage of it. This means that, through architectural limitations, a 32-bit environment will likely be under-powered for a production BI Server deployment.

Software Requirements

Pentaho supports a variety of software platform configurations. You must have one of the following operating systems, application servers, and repository databases installed, configured, remotely accessible if necessary, and its port and IP address settings readily available to you:

Operating systems

- Windows XP SP2, 2003, Vista, 7, and Server 2008 (for Pentaho servers only; there is no client tool support for Windows Server 2008)
- Modern Linux distributions (SUSE Linux Enterprise Server 10 and Red Hat Enterprise Linux 5 are officially supported, but most others should work)
- Solaris 10
- Mac OS X 10.5

Note: Some other operating systems, such as Windows Vista and FreeBSD, may work, but are not officially supported.

Application servers

- JBoss 5.1
- Tomcat 6.0

Note: Newer versions of these application servers may work, but were not part of Pentaho's quality assurance process for this version of the BI Suite.

Repository databases

- MySQL version 5.0 or higher
- PostgreSQL version 8.3 or higher
- Oracle 10g

Operating environment

The Sun Java Runtime Environment (J2SE) version 1.6 (sometimes referenced as version 6.0) is required for running the BI Server and the Pentaho client tools. Earlier versions, and alternative JREs from IBM, Microsoft, Blackdown, or other companies and projects are not supported and may not work properly.

In order to build a BI Server WAR package, you will need the Sun Java Development Kit (JDK) on your server. The JDK includes a JRE, so if you have the JDK 1.6 installed on any client machines, you will not need to install a separate JRE.

A recent version of Apache Ant is required. **Your Ant installation must be in your execution path as well**. If you install Ant on Linux through a package manager, you should be all set. If you're installing on Windows, you may have to edit your **PATH** variable to include the new Ant directory. Consult your IT manager or system administrator if you do not know how to do this.

Important Note on Local User Accounts For Linux/Unix

The section below advises you to create a local user account called **pentaho**. In this new user's home directory you will put the pentaho-solutions directory and store license information recorded by the Pentaho Enterprise Console. You will have to adjust your init scripts to start your Web application server (which will run the BI 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.

Caution: If you do not run the Pentaho Enterprise Console and the application server that runs the BI Server as the same system user, the BI 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 BI 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.

Caution: If you choose to put the pentaho-solutions directory in a parent directory that the pentaho user (or whatever user account you are running your Pentaho services as) does not have write access to, you will not be able to save any reports, analysis views, or dashboards, and you will not be able to set user permissions on any existing content.

Creating a Pentaho System User (Linux/Solaris)

This is for Linux, Solaris, and other Unix-like operating systems only. Windows users do not need to follow this procedure. Theoretically, you can use a different local user account than the one suggested here. However, you will have to modify all of the instructions from here onward 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 BI 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 BI Server and controlling Pentaho Enterprise Console start and stop scripts. You should stay logged into this new account to create the pentaho directory in the next section, and 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 fashion those scripts such that they start the service with the **pentaho** user credentials.

How to Check Your Java Version

The Pentaho BI Suite requires a Java Runtime Environment (JRE) or Java Development Kit (JDK) version 1.6 (sometimes referred to as 6.0). Follow the procedure below to see which version of Java is installed on your system and configured to be the default Java executable.

Note: There may be multiple JREs or JDKs on your system, but only one can be set as the global default. Any of these Java instances can be explicitly used for any Java program; if no specific Java executable is called, 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 will also show in the output.

```
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 the BI Suite 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 JRE or JDK version 1.6. If you do not set this variable, the BI Suite will use the JRE that the JAVA_HOME variable, which is generally established on most systems that run Java programs, points to. However, this variable may currently be pointing to a newer JRE that is required by a non-Pentaho program and cannot be changed, or it may change automatically in the future due to a Java upgrade and cause unexpected operation in the BI Suite. Therefore, Pentaho recommends that you set PENTAHO_JAVA_HOME as explained below.

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
   ```

The PENTAHO_JAVA_HOME variable is now set, so all Java applications will be able to access the JRE binaries and other resources.

Setting the PENTAHO_JAVA_HOME Variable on Windows

To ensure that the BI Suite 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 and point it to a JRE or JDK version 1.6. If you do not set this variable, the BI Suite will use the JRE that the JAVA_HOME variable, which is generally established on most systems that run Java programs, points to. However, this variable may currently be pointing to a newer JRE that is required by a non-Pentaho program and cannot be changed, or it may change in the future due to a Java upgrade and cause unexpected operation in the BI Suite. Therefore, Pentaho recommends that you set PENTAHO_JAVA_HOME as explained below.

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_JAVA_HOME** into the name field, and the directory you installed the JRE to into the value field, then click **OK**.
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.

   ```
   set | %windir%\system32\find "PENTAHO_JAVA_HOME"
   ```

The PENTAHO_JAVA_HOME variable is now set, so all Java applications will be able to access the JRE binaries and other resources.

Setting the PENTAHO_INSTALLED_LICENSE_PATH Variable on Linux

To ensure that the BI 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 BI 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 BI 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 BI 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 BI 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 BI 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:\Users\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.

   ```
   set | %windir%\system32\find "PENTAHO_INSTALLED_LICENSE_PATH"
   ```

The PENTAHO_INSTALLED_LICENSE_PATH variable is now set.

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 BI 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 BI Server starts, you

will need to modify the startup script for either the BI 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 BI Server service control script (instead of starting it automatically when Tomcat or JBoss starts), you will have to remember to 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 BI Server Build Materials

Enterprise Edition customers can obtain the BI Server build package from the Pentaho Enterprise Edition FTP site 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 `/4.0.0-GA/developers/biserver-manual-ee-3.9.0-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 BI 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 BI Server. Once the software has been configured and built, it will be deployed directly to your application server as a WAR archive.

The **Pentaho User Console plugins** items below refer to add-ons for the Pentaho User Console, such as Dashboard Designer and Pentaho Analyzer.

Windows

The packages you will need for a Windows server are:

- **BI Server:** /4.0.0-GA/developers/biserver-manual-ee-3.9.0-GA.zip
- **Pentaho User Console plugins:** /4.0.0-GA/server/plugins/
- **Pentaho Enterprise Console:** /4.0.0-GA/server/windows/pec-ee-3.9.0-GA.zip

Linux and Solaris

The packages you will need for a Linux or Solaris server are:

- **BI Server:** /4.0.0-GA/developers/biserver-manual-ee-3.9.0-GA.zip
- **Pentaho User Console plugins:** /4.0.0-GA/server/plugins/
- **Pentaho Enterprise Console:** /4.0.0-GA/server/nix/pec-ee-3.9.0-GA.tar.gz

Building the BI Server

Building the Pentaho BI 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 slipstreaming 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 slipstream 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:mysql://
localhost:3306/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.mysql.jdbc.Driver" url="jdbc:mysql://
localhost:3306/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 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 and Default Port Number

The Pentaho BI 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:

```
<?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:mysql://
localhost:3306/hibernate"
            validationQuery="select 1 from dual"/>

        <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.mysql.jdbc.Driver" url="jdbc:mysql://
localhost:3306/quartz"
            validationQuery="select 1 from dual"/>
</Context>
```

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.

Creating a Pentaho Directory

The Pentaho Web application expects a single, canonical location for your solutions directory. Though you can put the BI 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 use 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 BI Suite installation utility. Even though your application server, Pentaho client tools, and other files are not stored here, replicating the exact BI 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.

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-1.5.2.jar` to the newly created `/biserver-manual-ee/build-resources/custom-pentaho-webapp/WEB-INF/lib/` directory.

You are now ready to proceed to the WAR build instructions.

How to Build BI 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 BI Server build, using Apache Ant with the Enterprise Edition J2EE 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 just created, 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 BI Server package to. By default, this is `/biserver-manual-ee/`
5. Run Ant with the appropriate build options (listed in the next section).

The **pentaho-style.war**, which contains style information for Pentaho content, has been created in the `/biserver-manual-ee/build/pentaho-wars/` directory: The application package, **pentaho.war**, is 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 earlier. JBoss WARs are one level deeper in the **no-portal** subdirectory.

The solution directory has also been 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. Any property not specified in the Ant build command is assigned a default value, most of which are shown in the table below. If you do not need to change a default value, then there is no reason to specify a parameter in the build command. 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 BI Server

Once the BI Server 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 5.1 and Tomcat 6.0 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 BI Server with a basic configuration before you add data sources and perform other complex configuration tasks.

Copying Solution Database JDBC Drivers

This is only for Pentaho solution databases. You will also need to install JDBC drivers for your data sources at a later time, but the instructions in this section are focused solely on solution database configuration and connectivity with the BI Server.

Follow the below process to enable the BI Server to connect to an Oracle, MySQL, or PostgreSQL solution database.

1. Find or retrieve a JDBC driver JAR from your database vendor or third-party driver developer.

   Due to licensing restrictions, Pentaho is unable to provide the necessary JDBC driver JARs. You can retrieve a JDBC driver from your database vendor. To that end, you may find these links helpful:

   - **Oracle:** *http://www.oracle.com/technology/tech/java/sqlj_jdbc/index.html*
   - **MySQL:** *http://www.mysql.com/downloads/connector/j/*
   - **PostgreSQL:** *http://jdbc.postgresql.org/download.html*
2. For the BI Server, copy the appropriate JDBC driver JAR file to the `/tomcat/lib/` directory for Tomcat, or the `/jboss/server/default/lib/` directory for JBoss.
3. For Enterprise Console, copy the driver JAR to the `/pentaho/server/enterprise-console/lib/` directory.

The BI Server and Enterprise Console now have the necessary driver to communicate with your solution repository database.

Moving the Pentaho Solutions Directory

The pentaho-solutions directory contains all of your BI 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.

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 BI 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 BI Server solution repository.

PostgreSQL Solution Repository Configuration

Follow the below directions to configure the BI 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 BI 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 BI Server is now fully configured to connect to a PostgreSQL solution repository.

Configuring Quartz For PostgreSQL

The default JobStore JDBC database driver does not work properly with PostgreSQL. Follow the below procedure to replace the generic JDBC driver 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 = org.quartz.impl.jdbcjobstore.StdJDBCDelegate`

3. In the aforementioned line, replace the **org.quartz.impl.jdbcjobstore.StdJDBCDelegate** portion with this: **org.quartz.impl.jdbcjobstore.PostgreSQLDelegate**
4. Save the file and close the text editor.

The BI Server's scheduling (quartz) database is now configured to work with 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. Execute the **create_quartz_mysql.sql** script via your preferred utility or process.

   ```
   mysql -u root -p <create_quartz_mysql.sql
   ```
3. Execute the **create_repository_mysql.sql**.

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

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

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 BI Server data repository.

Oracle Solution Repository Configuration

Follow the below directions to configure the BI Server to use Oracle as a solution repository with the default Pentaho authentication backend. Even if you will end up using some other authentication

method, follow all of these directions anyway so that you can verify that the BI 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 BI Server is now fully configured to connect to an Oracle solution repository.

Configuring Quartz For Oracle

The default JobStore JDBC database driver does not work properly with Oracle. Follow the below procedure to replace the generic JDBC driver with one specific to Oracle.

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

Deploying to Tomcat 6.0

At this point, your Tomcat server should be running and configured for normal operation, and you should have completed the BI Server build process.

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

1. Stop the Tomcat server.
2. Copy the `/biserver-manual-ee/build/pentaho-wars/tomcat/pentaho.war` file to the `webapps` subdirectory of your Tomcat home directory.
3. Copy the **pentaho-style.war** file from the `/biserver-manual-ee/build/pentaho-wars/` directory to the Tomcat webapps directory:
4. Copy the appropriate JDBC driver JAR files for 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.

5. 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 BI Server, you can safely delete the `biserver-manual-ee` directory.

Modifying startup.sh on Linux and Solaris

Before you proceed to deploying the BI Suite, you must modify the Tomcat startup script so that the BI 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 after the initial comments in 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 BI Server.

Modifying startup.bat on Windows

Before you proceed to deploying the BI Suite, you must modify the Tomcat startup script so that the BI 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 after the initial comments in 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 BI 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 to JBoss 5.1

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

Modifying run.conf on Linux and Solaris

Before you proceed to deploying the BI Suite, 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 requirements, and therefore you will not need to perform the below customizations.

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).
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 the Pentaho BI Suite.

Modifying run.bat on Windows

Before you proceed to deploying the BI Suite, 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 BI Server.

Deploying a WAR to JBoss 5.1

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. 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. Copy the appropriate JDBC driver JAR files for 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 BI Server, you can safely delete the `biserver-manual-ee` directory.

Adding PDI Enterprise Repository Content Support to the BI 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 BI Server, you must install some BI 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 4.2.0 client tool archive package from the Pentaho Knowledge Base or Enterprise Edition FTP Site.

   The package name (available in both tar.gz and zip formats) is: **pdi-ee-client-4.2.0-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 BI 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 BI Server is now configured to

Installing Hadoop Hive Plugins

In order to proceed with this task, you must have already purchased a Pentaho BI Suite For Hadoop support entitlement. Skip this process if you do not have one.

The BI Server requires special Pentaho Data Integration plugins in order to use Hive metadata, and to use PDI transformations in action sequences. Follow the instructions below to install these plugins in the BI Server.

1. If you have not already done so, download the plugin packages from the Pentaho Enterprise Edition FTP site, or the Pentaho Knowledge Base.

   The package names are:

   - pdi-hadoop-plugin
   - pdi-hadoop-plugin-ee
2. Unpack the plugin packages to the `/pentaho/server/biserver-ee/pentaho-solutions/system/kettle/plugins/` directory. If this directory does not exist, create it.

The plugins you downloaded are now installed, though you will still need to register your license file in the Pentaho Enterprise Console, or with the command line license management tool. There are instructions for that process later in this guide. For now, continue on to the next section.

Installing Pentaho User Console Plugins

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

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

1. If you have not already done so, download the plugin packages from the Pentaho Enterprise Edition FTP site, or the Pentaho Knowledge Base.

   The Dashboard Designer package name always begins with **pdd**; the Pentaho Analyzer package always begins with **paz**; the Interactive Reporting plugin always starts with **pir**.
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 register your license files in the Pentaho Enterprise Console, or with the command line license management tool. There are instructions for that process later in this guide. For now, continue on to the next section.

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

The Pentaho Enterprise Console is a necessary BI 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-3.9.0-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. 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 BI 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-3.9.0-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. 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 BI 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 BI Platform and Pentaho Enterprise Console servers so that you can proceed with low-level system configuration.

1. Start your application server normally.

   The BI 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` 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 BI 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 built and deployed the BI Server, you must set up your configuration information and other details. All of this is done through the Pentaho Enterprise Console.

1. Open a Web browser and navigate to **http://localhost:8088** (change **localhost** to the IP address, hostname, or domain name of your BI Platform).
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 support entitlement 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 `/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 BI Suite Administrator's Guide*.

Your Pentaho BI Server now 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 BI Suite 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 BI 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 should see a Web page similar to the below picture:

You won't be able to log into the Pentaho User Console until you've established user accounts and roles in the Pentaho Enterprise Console.

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 BI Server.

Preparing for Production

The default condition of the BI 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 BI 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.

1. Shut down the BI 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 BI Server and Pentaho Enterprise Console.

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

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 BI Server Enterprise Edition version 4.0.0-GA. The installation process can change significantly between BI 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 BI Server version, including the open source BI Server Community Edition version 4.0-stable.

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 BI 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.

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. The BI Server installation instructions explain how to add driver JARs to the correct locations; there is also a section titled **Adding a JDBC Driver** in the *Pentaho BI Suite Administrator's Guide* that explains driver locations for all parts of the BI Suite.

Note: Some database vendors (and driver developers) require using JDBC version 4 drivers with a Java 6 environment. 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 BI Server and Pentaho Enterprise Console, then installed licenses, and are later unable to access Dashboard Designer, Pentaho Analyzer, or ad hoc 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 BI 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 BI 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** will, in some cases, 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 BI Server relies on many third-party libraries that provide everything from database connectivity to specific Java classes that add necessary features to the BI 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 BI 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 25.

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 BI 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 BI Server, the action sequence will not execute. This is because the BI 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 BI 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 BI Server, you must install some BI 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 4.2.0 client tool archive package from the Pentaho Knowledge Base or Enterprise Edition FTP Site.

   The package name (available in both tar.gz and zip formats) is: **pdi-ee-client-4.2.0-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 BI 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 BI Server is now configured to