

Luntbuild - Installation Guide

Luntbuild - Installation Guide

Copyright © 2005-2006 Luntbuild

Table of Contents

1. Installing Luntbuild	1
Using Luntbuild installer (with GUI)	1
Installation using zip distribution (without GUI)	1
Building Luntbuild from source distribution	2
Upgrading from previous versions of Luntbuild	2
Running Luntbuild in standalone mode	3
Backing up Luntbuild data	4
2. Configuring Luntbuild to use database	5
HSQL database	5
MySql database	6
PostgreSQL database	6
Updating database during Luntbuild upgrade	7
3. Configuring Luntbuild to use LDAP	8

List of Tables

2.1. QuantumDB HSQL DB Eclipse Configuration	5
2.2. QuantumDB MySql Eclipse Configuration	6
2.3. QuantumDB PostgreSQL Eclipse Configuration	7
3.1. LDAP configuration properties	8

Chapter 1. Installing Luntbuild

Installation using Luntbuild installer is the easiest way to install Luntbuild. We recommend to use the installer for all Luntbuild standard installations including upgrades. See installer section and upgrade section for details.

In case you need to modify for some reason Luntbuild configuration files (`web.xml`, `applicationContext.xml`) with attributes/values beyond the attributes/values modified by Luntbuild installer, you might choose installation using Luntbuild distribution zip file. See zip section for details.

In case you need to modify and/or extend the source code implementation of Luntbuild, you might choose to build Luntbuild distribution from the source code. See source section for details.

If you are upgrading from previous releases, please refer to upgrade section.

Using Luntbuild installer (with GUI)

Note

This is clean installation of Luntbuild. For upgrades from previous versions of Luntbuild please refer to upgrade section.

1. Make sure you have `jdk1.4` or `jdk1.5` installed, and add the directory which contains the `java` and `jar` executable into your system path - **IMPORTANT!**. Go to <http://java.sun.com> [<http://java.sun.com/>] for JDK download, if you didn't install it yet.
2. Make sure you get one of Luntbuild supported servlet containers or application servers installed (`Servlet2.3` and `JSP1.2` support are required), and make sure it has been stopped. Alternatively you can run Luntbuild in standalone mode.
3. Download Luntbuild installer from Luntbuild Sourceforge site [<http://sourceforge.net/projects/luntbuild/>]. This file is normally named as *luntbuild-xxx-installer.jar*, where `xxx` denotes current version.
4. Run command `java -jar luntbuild-xxx-installer.jar`. A GUI will display to guide you through the installation, and Luntbuild will install into the selected directory, let's say `/opt/luntbuild`.
5. Deploy *luntbuild.war* (located in `/opt/luntbuild` directory) into your servlet container or application server. Note, that if you selected the deployment directory of your servlet container or application server during installation, the installer will deploy *luntbuild.war* for you. If you plan to run Luntbuild in standalone mode (without servlet container), just start Luntbuild as described in standalone section.
6. Access the Luntbuild web application and you should be able to start your Luntbuild adventure, :D

Installation using zip distribution (without GUI)

Note

This is clean installation of Luntbuild. For upgrades from previous versions, please refer to upgrade section.

1. Download the Luntbuild zip distribution from Luntbuild Sourceforge site [<http://sourceforge.net/projects/luntbuild/>]. This file is normally named *luntbuild-xxx.zip*, where `xxx` denotes the current version.
2. Extract the zip file into the directory where you want to install Luntbuild, say `/opt/luntbuild`. Edit the following files with your text editor:

Edit file `/opt/luntbuild/web/WEB-INF/web.xml`:

Replace `$INSTALL_PATH` with your Luntbuild installation path (`/opt/luntbuild` here).

Replace `#{sessionTimeout}` with your desired session timeout value (normally 30).

Edit file `/opt/luntbuild/web/WEB-INF/applicationContext.xml`:

Replace `#{luntbuildPassword}` with your desired site administrator password.

3. If you plan to run Luntbuild in standalone mode (without servlet container), just start Luntbuild as described in standalone section. Else copy all the contents under `/opt/luntbuild/web` directory, and deploy it as a web application to your application server, or servlet container. For example, if you are using Tomcat servlet container:

Make sure Tomcat has been stopped

Change to Tomcat install dir: `> cd <tomcat install dir>/webapps`

Make luntbuild directory: `> mkdir luntbuild`

Copy luntbuild/web to webapps: `> cp -r /opt/luntbuild/web/* <tomcat install dir>/webapps/luntbuild`

Start Tomcat

Note. Do not create `luntbuild.war` file, just copy the contents under `/opt/luntbuild/web` directory to the `luntbuild` directory in the appropriate web application directory of your application server, or servlet container.

4. Access the Luntbuild web application and you should be able to start your Luntbuild adventure, :D

Building Luntbuild from source distribution

1. Make sure you have jdk1.4 or jdk1.5 installed, and add the directory which contains the java and jar executable into your system path - **IMPORTANT!**. Go to <http://java.sun.com> [<http://java.sun.com/>] for JDK download, if you didn't install it yet.
2. Make sure you have Apache ant 1.6.1 (or higher) installed. Goto <http://ant.apache.org> for Ant download.
3. Download the source distribution from Luntbuild Sourceforge site [<http://sourceforge.net/projects/luntbuild/>]. This file is normally named `luntbuild-xxx-src.zip`, where xxx denotes the current version.
4. Extract the source distribution into a directory, let's say `/yourhome/luntbuild-src`. Change to the directory `/yourhome/luntbuild-src/build`, and run command `ant clean installer zip`. Then Luntbuild installer and zip distribution will both be generated into directory `/yourhome/luntbuild-src/distribute`.
5. Follow the installation procedures in sections installer section, zip section, or upgrade section to install Luntbuild.

Upgrading from previous versions of Luntbuild

1. Assuming you've installed Luntbuild under `/opt/luntbuild`. Backup the directory `/opt/luntbuild/db` which contains your db files to another location. Alternatively, if you used external database to store Luntbuild data, backup the database. Backup all important build artifacts.
2. If you are upgrading from Luntbuild version 1.1.1, upgrade 1.1.1 web application with war file from here

. If you are upgrading from version 1.2 or higher just follow next steps.

Note. This only upgrades web application, and should not change anything under */opt/luntbuild*.

3. Access Luntbuild web application again, Select *Administration* tab, and export data into your specified file, let's say */yourhome/luntbuild-data.xml*. This file will be stored on the machine that runs your servlet container or application server hosting your Luntbuild application.
4. Follow the instructions in installer section, zip section, or source section of this file to install new release of Luntbuild into your previous Luntbuild installation directory, that is */opt/luntbuild* in this case.
5. Access the Luntbuild web application, click on *Administration* tab, and import from previously exported data file */yourhome/luntbuild-data.xml*. That will migrate data of previous Luntbuild installation to latest version.

However, if your previous version is 1.1.1, some settings needs to be re-configured:

The property *environment file* has been removed in 1.2. If you are using this property in 1.1.1, you need to extract contents of the environment file, and enter them as *Environment variables* property.

All build success condition has been reset for 1.2 format and above.

All build necessary condition has been reset for 1.2 format and above.

Format of property "Next build version" has been changed, please verify your version string.

Build properties passed to Ant build script has been changed, thus your Ant build script need to be changed to use new build properties.

Ant builder command has been reset to default value. You may need to change it based on your Ant installation directory.

Running Luntbuild in standalone mode

You do not need servlet container or application server to run Luntbuild. Luntbuild comes with *build-in* servlet container based on Jetty [<http://jetty.mortbay.org/jetty/>], a 100% Java HTTP Server and Servlet Container. Standalone Luntbuild launcher accepts three arguments:

hostname - name of the Luntbuild host machine

port - port number for servlet container

stop port - port number to use to stop launcher (optional, defaults to port+1)

Standalone Luntbuild stopper accepts two arguments:

hostname - name of the Luntbuild host machine

stop port - port number to use to stop launcher (optional, defaults to port+1)

You can run standalone Luntbuild launcher from the command line:

```
> cd <luntbuild-install-dir>
> bin/luntbuild.sh localhost 8888
```

or alternatively

```
> cd <luntbuild-install-dir>
> java -jar luntbuild-standalone.jar localhost 8888
```

To stop the standalone Luntbuild, you can use:

```
> cd <luntbuild-install-dir>
> bin/stop-luntbuild.sh localhost 8889
```

or alternatively

```
> cd <luntbuild-install-dir>
> java -cp luntbuild-standalone.jar com.luntsys.luntbuild.StandaloneStopper localhost 8889
```


While we sure you have several options, we suggest the excellent Java Service Wrapper [<http://wrapper.tanukisoftware.org/doc/english/introduction.html>] to run standalone Luntbuild. You'll get a clean, cross-platform way to start/stop/restart your services. On Unix, you'll get an init-style script, and on Windows you'll be able to integrate your app as a system service if you like.

Backing up Luntbuild data

It is very good idea to backup Luntbuild data so they can be restored in case they get corrupted. The backup strategy might depend on the type of the database you are using.

The database independent way to backup Luntbuild data is to use Export function in Administration tab. Unfortunately there is currently not an automated way to create a backup, it has to be created by login as Luntbuild administrator and perform Export.

In case your Luntbuild installation uses HSQLDB database (default) the database data is stored in <luntbuild install>/db directory. It is highly recommended to backup this directory on regular basis, if you do not want to use Export method mentioned above.

If you are using an external database (MySQL, PostgreSQL), please use the database backup/restore procedure, if you do not want to use Export method above.

Chapter 2. Configuring Luntbuild to use database

Luntbuild uses a database to make important Luntbuild data persistent. It uses ORM package Hibernate [<http://hibernate.org>] to access the persistent data and as a framework to access a database.

HSQL database

Luntbuild uses HSQL DB [<http://hsqldb.org/>] as default database running inside Luntbuild (in client mode). Luntbuild also supports following external (server-mode) databases:

MySql
ProgreSql

Database definitions and data are located in the db directory of the Luntbuild installation. HSQL DB [<http://hsqldb.org/>] uses db/luntbuild.script to define DB layout. Following files are also used, see HSQL DB documentation [<http://hsqldb.org/web/hsqldbDocsFrame.html>]:

db/luntbuild.data
db/luntbuild.bakup

Sometimes, it is useful to look at the data in the DB to find out issues. Following are the steps to run a SQL client to examine data in the HSQL DB:

Download HSQLDB 1.7.3.3 from [www.hsqldb.org](http://hsqldb.org) [<http://hsqldb.org/>]. (Latest version 1.8.0 should also work)
Extract the downloaded package.

Change into the demo folder, edit the file "runServer.bat", replace the original contents with this line:

```
@java -classpath ../lib/hsqldb.jar org.hsqldb.Server -database <db path>
```

<db path> should be replaced with your path to Luntbuild DB. For example, if you are installing Luntbuild in D:\luntbuild, you should specify <db path> as D:\luntbuild\db\luntbuild.

Run "runserver.bat" to start up the database server.

Run "runManager.bat", from the popup dialog, select "HSQL Database Engine Server" as "Type" option, then you'll able to run SQL commands to examine contents of the DB.

If you are using Eclipse [<http://eclipse.org>] you can use DB plugin Quantum DB [<http://quantum.sourceforge.net/>] and the set the connection:

Table 2.1. QuantumDB HSQL DB Eclipse Configuration

Parameter	Value
Jdbc Url	jdbc:hsqldb:file:D:\luntbuild-13alpha\db\luntbuild
User	sa
Password	
Driver Path	<app-server>/webapps/luntbuild/WEB-INF/lib/hsqldb.jar
Driver Class	org.hsqldb.jdbcDriver
Driver Type	HSQL (Hypersonic)

Note

Many Luntbuild tables contain binary data which can not be displayed through the client, for example, if you

run command "select * from LB_BUILD", nothing will be displayed. You should only specify non-binary columns here, for example, "select LB_ID, LB_NAME from LB_BUILD", etc.

MySQL database

MySQL [<http://www.mysql.com/>] database can be used as persistent storage for Luntbuild. The best way to configure Luntbuild to use MySQL is to use Luntbuild installer, select MySQL from Database install page, and fill appropriate database attributes. Alternatively you can configure <app-server>/webapps/WEB-INF/jdbc.properties for MySQL (see jdbc.mysql.properties in the same directory):

```
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:postgresql://localhost:5432/luntbuild
jdbc.username=luntbuild
jdbc.password=luntbuild
hibernate.dialect=org.hibernate.dialect.MySQLDialect
```

Then you have to create Luntbuild database as follows:

```
mysql> drop database luntbuild;
mysql> create database luntbuild;
mysql> use luntbuild;
mysql> source db\mysql\luntbuild.sql;
```

To manage MySQL consider using free version of SQLyog [http://www.webyog.com/sqlyog/download_sqlyogfree.html]. If you are using Eclipse [<http://eclipse.org>] you can use DB plugin Quantum DB [<http://quantum.sourceforge.net/>] and set the connection:

Table 2.2. QuantumDB MySQL Eclipse Configuration

Parameter	Value
Jdbc Url	jdbc:mysql://localhost:3306/luntbuild
User	
Password	
Driver Path	<app-server>/webapps/luntbuild/WEB-INF/lib/mysql-connector-java-3.1.7-bin.jar
Driver Class	com.mysql.jdbc.Driver
Driver Type	MySQL

PostgreSQL database

PostgreSQL [<http://www.postgresql.org/>] database can be used as persistent storage for Luntbuild. The best way to configure Luntbuild to use PostgreSQL is to use Luntbuild installer, select PostgreSQL from Database install page, and fill appropriate database attributes. Alternatively you can configure <app-server>/webapps/WEB-INF/jdbc.properties for PostgreSQL (see jdbc.postgresql.properties in the same directory):

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

Then you have to create Luntbuild database as follows:

```
> psql -h localhost -p 5432 postgres "admin"
postgres=# CREATE USER luntbuild;
postgres=# ALTER USER luntbuild PASSWORD 'luntbuild';
postgres=# CREATE DATABASE luntbuild WITH OWNER luntbuild;
postgres=# source luntbuild/db/postgresql/luntbuild.sql;
```

I use free pgAdminIII that comes with Postgres installation to manage Postgres database. If you are using Eclipse [<http://eclipse.org>] you can use DB plugin Quantum DB [<http://quantum.sourceforge.net/>] and set the connection:

Table 2.3. QuantumDB PostgreSQL Eclipse Configuration

Parameter	Value
Jdbc Url	jdbc:postgresql://localhost:5432/luntbuild
User	
Password	
Driver Path	<app-server>/webapps/luntbuild/WEB-INF/lib/postgresql-8.1-404.jdbc3.jar
Driver Class	org.postgresql.Driver
Driver Type	Postgres

Updating database during Luntbuild upgrade

The only supported way of updating Luntbuild database is to:

1. Export Luntbuild data from currently installed Luntuild.
2. Backup the Luntbuild database.
3. Delete the Luntbuild database.
4. Follow appropriate database installation to create database.
5. Follow appropriate Luntbuild installation.
6. Import previously exported data.

Chapter 3. Configuring Luntbuild to use LDAP

Luntbuild offers basic LDAP support for users authentication. LDAP can be configured in the installer, or, if you do not use installer to install Luntbuild, LDAP can be configured in the file `<servlet-webapps>/luntbuild/WEB-INF/applicationContext.xml`. The bean *luntbuildAuthenticationDAO* needs to be edited, for example:

```
<!-- luntbuild internal RDBMS based authentication DAO -->
<bean id="luntbuildAuthenticationDAO"
      class="com.luntsys.luntbuild.security.ApplicationInternalDAO">
  <property name="ldapHost" value="localhost"/>
  <property name="ldapPort" value="389"/>
  <property name="ldapUserDn" value="ou=People,dc=guessant,dc=org"/>
  <property name="ldapAuthentication" value="simple"/>
  <property name="ldapUserId" value="uid"/>
  <property name="ldapUseLuntbuildOnFail" value="true"/>
  <property name="ldapCreateLuntbuildUser" value="true"/>
  <property name="ldapCanCreateProject" value="true"/>
  <property name="ldapCanViewProject" value="true"/>
  <property name="ldapCanBuildProject" value="true"/>
  <property name="ldapEmailAttrName" value="mail"/>
</bean>
```

Following table explains the meaning of the LDAP configuration properties:

Table 3.1. LDAP configuration properties

Property	Meaning
ldapHost	LDAP server host
ldapPort	LDAP port (default is 389)
ldapUserDn	User DN address to access users database
ldapAuthentication	Authentication type (default is <i>simple</i>)
ldapUserId	User id LDAP identifier (default is <i>uid</i>)
ldapUseLuntbuildOnFail	<i>boolean</i> ; if true uses Luntbuild authentication if LDAP fails
ldapCreateLuntbuildUser	<i>boolean</i> ; if LDAP user is not a Luntbuild user, create Luntbuild user with the same name and password
ldapCanCreateProject	<i>boolean</i> ; if true and if Luntbuild user is created, the user can create project and can administer existing projects
ldapCanBuildProject	<i>boolean</i> ; if true and if Luntbuild user is created, the user can build existing projects
ldapCanViewProject	<i>boolean</i> ; if true and if Luntbuild user is created, the user can view existing projects
ldapEmailAttrName	User email LDAP identifier (default is <i>mail</i>)