Installing DSpace

For the Impatient

Since some users might want to get their test version up and running as fast as possible, offered below is an *unsupported* outline of getting DSpace to run quickly in a Unix-based environment using the DSpace source release.

> Only experienced unix admins should even attempt the following without going to the detailed Installation Instructions

```
useradd -m dspace
gzip xzf dspace-4.x-src-release.tar.gz
createuser --username=postgres --no-superuser --pwprompt dspace
createdb --username=postgres --owner=dspace --encoding=UNICODE dspace
cd [dspace-source]
vi build.properties
mkdir [dspace]
chown dspace [dspace]
su - dspace
cd [dspace-source]
mvn package
cd [dspace-source]/dspace/target/dspace-<version>-build
ant fresh_install
cp -r [dspace]/webapps/* [tomcat]/webapps
/etc/init.d/tomcat start
[dspace]/bin/dspace create-administrator
```

Hardware Recommendations

You can install and run DSpace on most modern PC, laptop or server hardware. However, if you intend to run DSpace for a large community of potential end users, carefully review the Hardware Recommendations.

Prerequisite Software

The list below describes the third-party components and tools you'll need to run a DSpace server. These are just guidelines. Since DSpace is built on open source, standards-based tools, there are numerous other possibilities and setups.

Also, please note that the configuration and installation guidelines relating to a particular tool below are here for convenience. You should refer to the documentation for each individual component for complete and up-to-date details. Many of the tools are updated on a frequent basis, and the guidelines below may become out of date.

UNIX-like OS or Microsoft Windows

- UNIX-like OS (Linux, HP/UX, Mac OSX, etc.) : Many distributions of Linux/Unix come with some of the dependencies below pre-installed or easily installed via updates. You should consult your particular distribution's documentation or local system administrators to determine what is already available.
- Microsoft Windows: After verifying all prerequisites below, see the Windows Installation section for Windows tailored instructions

Oracle Java JDK 7 (standard SDK is fine, you don't need J2EE) or OpenJDK 7

Oracle's Java can be downloaded from the following location: http://www.oracle.com/technetwork/java/javase/downloads/index.html. Again, you can just download the Java SE JDK version.

OpenJDK download and installation instructions can be found here http://openjdk.java.net/.

Apache Maven 3.x (Java build tool)

Maven is necessary in the first stage of the build process to assemble the installation package for your DSpace instance. It gives you the flexibility to customize DSpace using the existing Maven projects found in the *[dspace-source]/dspace/modules* directory or by adding in your own Maven project to build the installation package for DSpace, and apply any custom interface "overlay" changes.

Maven can be downloaded from the following location: http://maven.apache.org/download.html

Configuring a Proxy

You can configure a proxy to use for some or all of your HTTP requests in Maven. The username and password are only required if your proxy requires basic authentication (note that later releases may support storing your passwords in a secured keystore, in the mean time, please ensure your *settings.xml* file (usually *${user.home}/.m2/settings.xml*) is secured with permissions appropriate for your operating system).

Example:

```
<settings>
  .
  .
  <proxies>
   <proxy>
      <active>true</active>
      <protocol>http</protocol>
      <host>proxy.somewhere.com</host>
      <port>8080</port>
      <username>proxyuser</username>
      <password>somepassword</password>
      <nonProxyHosts>www.google.com|*.somewhere.com</nonProxyHosts>
    </proxy>
  </proxies>
  .
  .
</settings>
```

Apache Ant 1.8 or later (Java build tool)

Apache Ant is required for the second stage of the build process. It is used once the installation package has been constructed in *[dspace-source]/dspace /target/dspace-<version>-build* and still uses some of the familiar ant build targets found in the 1.4.x build process.

Ant can be downloaded from the following location: http://ant.apache.org

Relational Database: (PostgreSQL or Oracle)

- **PostgreSQL 8.4 to 9.1** PostgreSQL can be downloaded from http://www.postgresql.org/ . It is highly recommended that you try to work with Postgres 8.4 or greater, however 8.3 should still work. Unicode (specifically UTF-8) support must be enabled. This is enabled by default in 8.0+. Once installed, you need to enable TCP/IP connections (DSpace uses JDBC):
  - In `postgresql.conf`: uncomment the line starting: `listen_addresses = 'localhost'`. This is the default, in recent PostgreSQL releases, but you should at least check it.
  - Then tighten up security a bit by editing `pg_hba.conf` and adding this line: `host dspace dspace 127.0.0.1 255.255.255.255 md5`. This should appear *before* any lines matching `all` databases, because the first matching rule governs.
  - Then restart PostgreSQL.

- **Oracle 10g or greater** Details on acquiring Oracle can be downloaded from the following location: http://www.oracle.com/database/. You will need to create a database for DSpace. Make sure that the character set is one of the Unicode character sets. DSpace uses UTF-8 natively, and it is suggested that the Oracle database use the same character set. You will also need to create a user account for DSpace (e.g. *dspace*) and ensure that it has permissions to add and remove tables in the database. Refer to the Quick Installation for more details.
    - **NOTE:** If the database server is not on the same machine as DSpace, you must install the Oracle client to the DSpace server and point tnsnames.ora and listener.ora files to the database the Oracle server.
    - **NOTE:** DSpace uses sequences to generate unique object IDs — beware Oracle sequences, which are said to lose their values when doing a database export/import, say restoring from a backup. Be sure to run the script *etc/oracle/update-sequences.sql* after importing.
    - For people interested in switching from Postgres to Oracle, I know of no tools that would do this automatically. You will need to recreate the community, collection, and eperson structure in the Oracle system, and then use the item export and import tools to move your content over.

Servlet Engine (Apache Tomcat 7 or later, Jetty, Caucho Resin or equivalent)

Tomcat 7 Version

If you are using Tomcat 7, we recommend running Tomcat 7.0.30 or above. Tomcat 7.0.29 and lower versions suffer from a memory leak. As a result, those versions of tomcat require an unusual high amount of memory to run DSpace. This has been resolved as of Tomcat 7.0.30. More information can be found in DS-1553

- **Apache Tomcat 7 or higher.** Tomcat can be downloaded from the following location: http://tomcat.apache.org.
    - Note that DSpace will need to run as the same user as Tomcat, so you might want to install and run Tomcat as a user called '*dspace*'. Set the environment variable *TOMCAT_USER* appropriately.
    - You need to ensure that Tomcat has a) enough memory to run DSpace and b) uses UTF-8 as its default file encoding for international character support. So ensure in your startup scripts (etc) that the following environment variable is set: *JAVA_OPTS="-Xmx512M -Xms64M -Dfile.encoding=UTF-8"*
    - **Modifications in *[tomcat]/conf/server.xml*** : You also need to alter Tomcat's default configuration to support searching and browsing of multi-byte UTF-8 correctly. You need to add a configuration option to the *<Connector>* element in *[tomcat]/config/server.xml*: *URIEncoding="UTF-8"*e.g. if you're using the default Tomcat config, it should read:

      ```
      <!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
      <Connector port="8080"
                 maxThreads="150"
                 minSpareThreads="25"
                 maxSpareThreads="75"
                 enableLookups="false"
                 redirectPort="8443"
                 acceptCount="100"
                 connectionTimeout="20000"
                 disableUploadTimeout="true"
                 URIEncoding="UTF-8"/>
      ```

      You may change the port from 8080 by editing it in the file above, and by setting the variable *CONNECTOR_PORT* in *server.xml*.
- **Jetty or Caucho Resin** DSpace will also run on an equivalent servlet Engine, such as Jetty (http://www.mortbay.org/jetty/index.html) or Caucho Resin (http://www.caucho.com/). Jetty and Resin are configured for correct handling of UTF-8 by default.

Perl (only required for [dspace]/bin/dspace-info.pl)

Installation Instructions

Overview of Install Options

With the advent of a new Apache Maven 2 based build architecture (first introduced in DSpace 1.5.x), you now have two options in how you may wish to install and manage your local installation of DSpace. If you've used DSpace 1.4.x, please recognize that the initial build procedure has changed to allow for more customization. You will find the later 'Ant based' stages of the installation procedure familiar. Maven is used to resolve the dependencies of DSpace online from the 'Maven Central Repository' server.

It is important to note that the strategies are identical in terms of the list of procedures required to complete the build process, the only difference being that the Source Release includes "more modules" that will be built given their presence in the distribution package.

- Binary Release (dspace-<version>-release.zip)
    - This distribution will be adequate for most cases of running a DSpace instance. It is intended to be the quickest way to get DSpace installed and running while still allowing for customization of the themes and branding of your DSpace instance.
    - This method allows you to customize DSpace configurations (in dspace.cfg) or user interfaces, using basic pre-built interface "overlays".
    - It downloads "precompiled" libraries for the core dspace-api, supporting servlets, taglibraries, aspects and themes for the dspace-xmlui, dspace-xmlui and other webservice/applications.
    - This approach only exposes selected parts of the application for customization. All other modules are downloaded from the 'Maven Central Repository' The directory structure for this release is the following:
        - *[dspace-source]*
            - *dspace/* - DSpace 'build' and configuration module
- Source Release (dspace-<version>-src-release.zip)

- This method is recommended for those who wish to develop DSpace further or alter its underlying capabilities to a greater degree.
- It contains **all** dspace code for the core dspace-api, supporting servlets, taglibraries, aspects and themes for Manakin (dspace-xmlui), and other webservice/applications.
- Provides all the same capabilities as the binary release. The directory structure for this release is more detailed:
    - *[dspace-source]*
        - *dspace/* - DSpace 'build' and configuration module
        - *dspace-api/* - Java API source module
        - *dspace-jspui/* - JSP-UI source module
        - *dspace-lni* - Lightweight Network Interface source module
        - *dspace-oai* - OAI-PMH source module
        - *dspace-services* - Common Services module
        - *dspace-sword* - SWORD (Simple Web-serve Offering Repository Deposit) deposit service source module
        - *dspace-swordv2* - SWORDv2 source module
        - *dspace-xmlui* - XML-UI (Manakin) source module
        - *pom.xml* - DSpace Parent Project definition

Overview of DSpace Directories

Before beginning an installation, it is important to get a general understanding of the DSpace directories and the names by which they are generally referred. (Please attempt to use these below directory names when asking for help on the DSpace Mailing Lists, as it will help everyone better understand what directory you may be referring to.)

DSpace uses three separate directory trees. Although you don't need to know all the details of them in order to install DSpace, you do need to know they exist and also know how they're referred to in this document:

1. **The installation directory**, referred to as `[dspace]`. This is the location where DSpace is installed and running. It is the location that is defined in the `dspace.cfg` as "dspace.dir". It is where all the DSpace configuration files, command line scripts, documentation and webapps will be installed.
2. **The source directory**, referred to as `[dspace-source]` . This is the location where the DSpace release distribution has been unpacked. It usually has the name of the archive that you expanded such as `dspace-<version>-release` or `dspace-<version>-src-release`. Normally it is the directory where all of your "build" commands will be run.
3. **The web deployment directory**. This is the directory that contains your DSpace web application(s). In DSpace 1.5.x and above, this corresponds to `[dspace]/webapps` by default. However, if you are using Tomcat, you may decide to copy your DSpace web applications from `[dspace]/webapps/` to `[tomcat]/webapps/` (with `[tomcat]` being wherever you installed Tomcat, also known as `$CATALINA_HOME`).
For details on the contents of these separate directory trees, refer to directories.html. *Note that the* `[dspace-source]` *and* `[dspace]` *directories are always separate!*

If you ever notice that many files seems to have duplicates under `[dspace-source]/dspace/target` do not worry about it. This "target" directory will be used by Maven for the build process and you should not change any file in it unless you know exactly what you are doing.

Installation

This method gets you up and running with DSpace quickly and easily. It is identical in both the Default Release and Source Release distributions.

1. **Create the DSpace user**. This needs to be the same user that Tomcat (or Jetty etc.) will run as. e.g. as ***root*** run:

    ```
    useradd -m dspace
    ```

2. **Download** the latest DSpace release. There are two version available with each release of DSpace: (*dspace-n.x-release.* and *dspace-n.x-src-release.zzz*); you only need to choose one. If you want a copy of all underlying Java source code, you should download the *dspace-n.x-src-release.xxx* Within each version, you have a choice of compressed file format. Choose the one that best fits your environment.
    a. Alternatively, you may choose to check out the latest release from the DSpace GitHub Repository. In this case, you'd be checking out the full Java source code. You'd also want to be sure to checkout the appropriate tag (e.g. dspace-4.0) or branch. For more information on using / developing from the GitHub Repository, see: Development with Git
3. **Unpack the DSpace software**. After downloading the software, based on the compression file format, choose one of the following methods to unpack your software:
    a. **Zip file**. If you downloaded *dspace-4.x-release.zip* do the following:

        ```
        unzip dspace-4.x-release.zip
        ```

    b. **.gz file**. If you downloaded *dspace-4.x-release.tar.gz* do the following:

        ```
        gunzip -c dspace-4.x-release.tar.gz | tar -xf -
        ```

    c. **.bz2 file**. If you downloaded _dspace-4.x-release.tar.bz do the following:

        ```
        bunzip2 dspace-4.x-release.tar.bz | tar -xf -
        ```

For ease of reference, we will refer to the location of this unzipped version of the DSpace release as *[dspace-source]* in the remainder of these instructions. After unpacking the file, the user may wish to change the ownership of the *dspace-4.x-release* to the "dspace" user. (And you may need to change the group).

4. **Database Setup**
   - *PostgreSQL:*
     - Create a `dspace` database user. This is entirely separate from the `dspace` operating-system user created above *(you are still logged in as "root")*:

       ```
       createuser --username=postgres --no-superuser --pwprompt dspace
       ```

       You will be prompted (twice) for a password for the new `dspace` user. Then you'll be prompted for the password of the PostgreSQL superuser (`postgres`).

     - Create a `dspace` database, owned by the `dspace` PostgreSQL user *(you are still logged in as 'root')*:

       ```
       createdb --username=postgres --owner=dspace --encoding=UNICODE dspace
       ```

       You will be prompted for the password of the PostgreSQL superuser (`postgres`).
   - *Oracle:*
     - Setting up DSpace to use Oracle is a bit different now. You will need still need to get a copy of the Oracle JDBC driver, but instead of copying it into a lib directory you will need to install it into your local Maven repository. (You'll need to download it first from this location: http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-112010-090769.html.) Run the following command (all on one line):

       ```
       mvn install:install-file
           -Dfile=ojdbc6.jar
           -DgroupId=com.oracle
           -DartifactId=ojdbc6
           -Dversion=11.2.0.3.0
           -Dpackaging=jar
           -DgeneratePom=true
       ```

     - You need to compile DSpace with an Oracle driver (ojdbc6.jar) corresponding to your Oracle version - update the version in *[dspace-source]/pom.xml* E.g.:

       ```
       <dependency>
         <groupId>com.oracle</groupId>
         <artifactId>ojdbc6</artifactId>
         <version>11.2.0.3.0</version>
       </dependency>
       ```

     - Create a database for DSpace. Make sure that the character set is one of the Unicode character sets. DSpace uses UTF-8 natively, and it is required that the Oracle database use the same character set. Create a user account for DSpace (e.g. *dspace*) and ensure that it has permissions to add and remove tables in the database.
     - Uncomment and edit the Oracle database settings in *[dspace-source]/build.properties* (see below for more information on the build.properties file):

       ```
       db.name   = oracle
       db.driver = oracle.jdbc.OracleDriver
       db.url = jdbc:oracle:thin:@host:port/SID
       ```

       Where SID is the SID of your database defined in tnsnames.ora, default Oracle port is 1521.
       Alternatively, you can use a full SID definition, e.g.:

       ```
       db.url = jdbc:oracle:thin:@(description=(address_list=(address=(protocol=TCP)
       (host=localhost)(port=1521)))(connect_data=(service_name=DSPACE)))
       ```

     - Later, during the Maven build step, don't forget to specify `mvn -Ddb.name=oracle package`
5. **Initial Configuration:** Edit `[dspace-source]/build.properties`. This properties file contains the basic settings necessary to actually build /install DSpace for the first time (see build.properties Configuration for more detail). In particular you'll need to set these properties -- examples or defaults are provided in the file:
   - `dspace.install.dir` - must be set to the *[dspace]* (installation) directory (*On Windows be sure to use forward slashes for the directory path!* For example: "C:/dspace" is a valid path for Windows.)
   - `dspace.hostname` - fully-qualified domain name of web server.
   - `dspace.baseUrl` - complete URL of this server's DSpace home page but without any context eg. /xmlui, /oai, etc.

- `dspace.name` - "Proper" name of your server, e.g. "My Digital Library".
- `solr.server` - complete URL of the Solr server. DSpace makes use of Solr for indexing purposes.
- `default.language`
- `db.name - postgres or oracle`
- `db.driver`
- `db.url`
- `db.username` - the database username used in the previous step.
- `db.password` - the database password used in the previous step.
- `mail.server` - fully-qualified domain name of your outgoing mail server.
- `mail.from.address` - the "From:" address to put on email sent by DSpace.
- `mail.feedback.recipient` - mailbox for feedback mail.
- `mail.admin` - mailbox for DSpace site administrator.
- `mail.alert.recipient` - mailbox for server errors/alerts (not essential but very useful!)
- `mail.registration.notify`- mailbox for emails when new users register (optional)

> The "build.properties" file is provided as a convenient method of setting only those configurations necessary to install/upgrade DSpace. Any settings changed in this file, will be automatically copied over to the full "dspace.cfg" file (which is held in `[dspace-source]/dspace/config/dspace.cfg`). Refer to the General Configuration section for a fuller explanation.
>
> It is also worth noting that you may choose to copy/rename the "build.properties" under a different name for different environments (e.g. "development.properties", "test.properties", and "production.properties"). You can choose which properties file you want to build DSpace with by passing a "-Denv" (environment) flag to the "mvn package" command (e.g. "mvn package -Denv=test" would build using "test.properties"). See General Configuration section for more details.

> **Do not remove or comment out settings in build.properties**
>
> When you edit the "build.properties" file (or a custom *.properties file), take care not to remove or comment out any settings. Doing so, may cause your final "dspace.cfg" file to be misconfigured with regards to that particular setting. Instead, if you wish to remove/disable a particular setting, just clear out its value. For example, if you don't want to be notified of new user registrations, ensure the "mail.registration.notify" setting has no value, e.g.
>
> `mail.registration.notify=`

6. **DSpace Directory:** Create the directory for the DSpace installation (i.e. `[dspace]`). As *root* (or a user with appropriate permissions), run:

```
mkdir [dspace]
chown dspace [dspace]
```

(Assuming the *dspace* UNIX username.)

7. **Build the Installation Package:** As the *dspace* UNIX user, generate the DSpace installation package.

```
cd [dspace-source]/dspace/
mvn package
```

> **In DSpace 4.0, the above command must be run from [dspace-source]**
>
> In the DSpace 4.0 release, the above "mvn package" command **must** be run from the root source directory (i.e. `[dspace-source]`), otherwise you will receive build errors. This was a small (but annoying) bug in our Maven build process, which is fixed in the 4.1 release (see DS-1867)

> **Defaults to PostgreSQL settings**
>
> Without any extra arguments, the DSpace installation package is initialized for PostgreSQL. If you want to use Oracle instead, you should build the DSpace installation package as follows:
> `mvn -Ddb.name=oracle package`

> **Defaults to building installation package with settings from "build.properties"**
>
> Without any extra arguments, the DSpace installation package will be initialized using the settings in the `[dspace-source]/build.properties` file. However, if you want it to build using a custom properties file, you may specify the "-Denv" (environment) flag as follows:
>
> `mvn -Denv=test package` (would build the installation package using a custom `[dspace-source]/test.properties` file)
>
> `mvn -Denv=local package` (would build the installation package using a custom `[dspace-source]/local.properties` file)
>
> See General Configuration section for more details.

8. **Install DSpace and Initialize Database:** As the *dspace* UNIX user, initialize the DSpace database and install DSpace to `[dspace]`:

```
cd [dspace-source]/dspace/target/dspace-[version]-build
ant fresh_install
```

To see a complete list of build targets, run: `ant help` *The most likely thing to go wrong here is the database connection. See the Common Problems Section.*

9. **Deploy Web Applications:** Please note that in the first instance you should refer to the appropriate documentation for your Web Server of choice. The following instructions are meant as a handy guide. You have two choices or techniques for having Tomcat/Jetty/Resin serve up your web applications:

    - *Technique A.* Tell your Tomcat/Jetty/Resin installation where to find your DSpace web application(s). As an example, in the directory `[tomcat]/conf/Catalina/localhost` you could add files similar to the following (but replace `[dspace]`with your installation location):

DEFINE A CONTEXT FOR DSpace XML User Interface: xmlui.xml

```
<?xml version='1.0'?>
<Context
        docBase="[dspace]/webapps/xmlui"
        reloadable="true"
        cachingAllowed="false"/>
```

DEFINE A CONTEXT PATH FOR DSpace JSP User Interface: jspui.xml

```
<?xml version='1.0'?>
<Context
        docBase="[dspace]/webapps/jspui"
        reloadable="true"
        cachingAllowed="false"/>
```

DEFINE A CONTEXT PATH FOR DSpace OAI User Interface: oai.xml

```
<?xml version='1.0'?>
<Context
        docBase="[dspace]/webapps/oai"
        reloadable="true"
        cachingAllowed="false"/>
```

DEFINE ADDITIONAL CONTEXT PATHS FOR OTHER DSPACE WEB APPLICATIONS (SOLR, SWORD, LNI, etc.): [app].xml

```
<?xml version='1.0'?>
<!-- CHANGE THE VALUE OF "[app]" FOR EACH APPLICATION YOU WISH TO ADD -->
<Context
        docBase="[dspace]/webapps/[app]"
        reloadable="true"
        cachingAllowed="false"/>
```

The name of the file (not including the suffix ".xml") will be the name of the context, so for example `xmlui.xml` defines the context at `http://host:8080/xmlui`. To define the *root context* (`http://host:8080/`), name that context's file `ROOT.xml`.

Tomcat Context Settings in Production

The above Tomcat Context Settings show adding the following to each `<Context>` element:

`reloadable="true" cachingAllowed="false"`

These settings are extremely useful to have when you are first getting started with DSpace, as they let you tweak the DSpace XMLUI (XSLTs or CSS) or JSPUI (JSPs) and see your changes get automatically reloaded by Tomcat (without having to restart Tomcat). However, it is worth noting that the Apache Tomcat documentation recommends Production sites leave the

default values in place (`reloadable="false" cachingAllowed="true"`), as allowing Tomcat to automatically reload all changes may result in "significant runtime overhead".

It is entirely up to you whether to keep these Tomcat settings in place. We just recommend beginning with them, so that you can more easily customize your site without having to require a Tomcat restart. Smaller DSpace sites may not notice any performance issues with keeping these settings in place in Production. Larger DSpace sites may wish to ensure that Tomcat performance is more streamlined.

- *Technique B.* Simple and complete. You copy only (or all) of the DSpace Web application(s) you wish to use from the [dspace]/webapps directory to the appropriate directory in your Tomcat/Jetty/Resin installation. For example:
  `cp -R [dspace]/webapps/* [tomcat]/webapps*` (This will copy all the web applications to Tomcat).
  `cp -R [dspace]/webapps/jspui [tomcat]/webapps*` (This will copy only the jspui web application to Tomcat.)

10. **Administrator Account:** Create an initial administrator account:

```
[dspace]/bin/dspace create-administrator
```

11. **Initial Startup!** Now the moment of truth! Start up (or restart) Tomcat/Jetty/Resin. Visit the base URL(s) of your server, depending on which DSpace web applications you want to use. You should see the DSpace home page. Congratulations! Base URLs of DSpace Web Applications:
    - *JSP User Interface* - (e.g.) `http://dspace.myu.edu:8080/jspui`
    - *XML User Interface* (aka. Manakin) - (e.g.) `http://dspace.myu.edu:8080/xmlui`
    - *OAI-PMH Interface* - (e.g.) `http://dspace.myu.edu:8080/oai/request?verb=Identify` (Should return an XML-based response)

In order to set up some communities and collections, you'll need to login as your DSpace Administrator (which you created with `create-administrator` above) and access the administration UI in either the JSP or XML user interface.

Advanced Installation

The above installation steps are sufficient to set up a test server to play around with, but there are a few other steps and options you should probably consider before deploying a DSpace production site.

'cron' jobs / scheduled tasks

A few DSpace features **require** that a script is run regularly (via cron, or similar):

- the e-mail subscription feature that alerts users of new items being deposited;
- the 'media filter' tool, that generates thumbnails of images and extracts the full-text of documents for indexing;
- the 'checksum checker' that tests the bitstreams in your repository for corruption;
- the sitemap generator, which enhances the ability of major search engines to index your content and make it findable;
- the curation system queueing feature, which allows administrators to "queue" tasks (to run at a later time) from the Admin UI;
- and Discovery (search & browse), OAI-PMH and Usage Statistics all receive performance benefits from regular re-optimization

For much more information on recommended scheduled tasks, please see Scheduled Tasks via Cron.

Multilingual Installation

In order to deploy a multilingual version of DSpace you have to configure two parameters in *[dspace-source]/config/dspace.cfg:*

- *default.locale*, e.g. `default.locale = en`
- *webui.supported locales*, e.g. `webui.supported.locales = en, de`

The Locales might have the form country, country_language, country_language_variant.

According to the languages you wish to support, you have to make sure, that all the i18n related files are available see the Multilingual User Interface Configuring MultiLingual Support section for the JSPUI or the Multilingual Support for XMLUI in the configuration documentation.

DSpace over HTTPS

If your DSpace is configured to have users login with a username and password (as opposed to, say, client Web certificates), then you should consider using HTTPS. Whenever a user logs in with the Web form (e.g. *dspace.myuni.edu/dspace/password-login*) their DSpace password is exposed in plain text on the network. This is a very serious security risk since network traffic monitoring is very common, especially at universities. If the risk seems minor, then consider that your DSpace administrators also login this way and they have ultimate control over the archive.

The solution is to use *HTTPS* (HTTP over SSL, i.e. Secure Socket Layer, an encrypted transport), which protects your passwords against being captured. You can configure DSpace to require SSL on all "authenticated" transactions so it only accepts passwords on SSL connections.

The following sections show how to set up the most commonly-used Java Servlet containers to support HTTP over SSL.

Enabling the HTTPS support in Tomcat 7.0

Loosely based on http://tomcat.apache.org/tomcat-7.0-doc/ssl-howto.html.

1. **For Production use:** Follow this procedure to set up SSL on your server. Using a "real" server certificate ensures your users' browsers will accept it without complaints. In the examples below, *$CATALINA_BASE* is the directory under which your Tomcat is installed.
    a. Create a Java keystore for your server with the password *changeit*, and install your server certificate under the alias *"tomcat"*. This assumes the certificate was put in the file *server.pem*:

```
$JAVA_HOME/bin/keytool -import -noprompt -v -storepass changeit
        -keystore $CATALINA_BASE/conf/keystore -alias tomcat -file
        myserver.pem
```

    b. Install the CA (Certifying Authority) certificate for the CA that granted your server cert, if necessary. This assumes the server CA certificate is in *ca.pem*:

```
$JAVA_HOME/bin/keytool -import -noprompt -storepass changeit
        -trustcacerts -keystore $CATALINA_BASE/conf/keystore -alias ServerCA
        -file ca.pem
```

    c. Optional – ONLY if you need to accept client certificates for the X.509 certificate stackable authentication module See the configuration section for instructions on enabling the X.509 authentication method. Load the keystore with the CA (certifying authority) certificates for the authorities of any clients whose certificates you wish to accept. For example, assuming the client CA certificate is in *client1.pem*:

```
$JAVA_HOME/bin/keytool -import -noprompt -storepass changeit
        -trustcacerts -keystore $CATALINA_BASE/conf/keystore  -alias client1
        -file client1.pem
```

    d. Now add another Connector tag to your *server.xml*/Tomcat configuration file, like the example below. The parts affecting or specific to SSL are shown in bold. (You may wish to change some details such as the port, pathnames, and keystore password)

```
<Connector port="8443"
              maxThreads="150" minSpareThreads="25"
              maxSpareThreads="75"
              enableLookups="false"
              disableUploadTimeout="true"
              acceptCount="100" debug="0"
              scheme="https" secure="true" sslProtocol="TLS"
              keystoreFile="conf/keystore" keystorePass="changeit" clientAuth="true" - ONLY if
using client X.509 certs for authentication!
              truststoreFile="conf/keystore" trustedstorePass="changeit" />
```

    Also, check that the default Connector is set up to redirect "secure" requests to the same port as your SSL connector, e.g.:

```
<Connector port="8080"
              maxThreads="150" minSpareThreads="25"
              maxSpareThreads="75"
              enableLookups="false"
              redirectPort="8443"
              acceptCount="100" debug="0" />
```

2. **Quick-and-dirty Procedure for Testing:** If you are just setting up a DSpace server for testing, or to experiment with HTTPS, then you don't need to get a real server certificate. You can create a "self-signed" certificate for testing; web browsers will issue warnings before accepting it but they will function exactly the same after that as with a "real" certificate. In the examples below, *$CATALINA_BASE*is the directory under which your Tomcat is installed.
    a. Optional – ONLY if you don't already have a server certificate. Follow this sub-procedure to request a new, signed server certificate from your Certifying Authority (CA):
        - Create a new key pair under the alias name *"tomcat"*. When generating your key, give the Distinguished Name fields the appropriate values for your server and institution. CN should be the fully-qualified domain name of your server host. Here is an example:

```
$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA -keysize
        1024 \
  -keystore $CATALINA_BASE/conf/keystore -storepass changeit
        -validity 365 \
  -dname 'CN=dspace.myuni.edu, OU=MIT Libraries, O=Massachusetts
        Institute of Technology, L=Cambridge, S=MA, C=US'
```

- Then, create a *CSR* (Certificate Signing Request) and send it to your Certifying Authority. They will send you back a signed Server Certificate. This example command creates a CSR in the file *tomcat.csr*

```
$JAVA_HOME/bin/keytool -keystore $CATALINA_BASE/conf/keystore \
  -storepass changeit \
  -certreq -alias tomcat -v -file tomcat.csr
```

- Before importing the signed certificate, you must have the CA's certificate in your keystore as a *trusted certificate*. Get their certificate, and import it with a command like this (for the example *mitCA.pem*):

```
$JAVA_HOME/bin/keytool -keystore $CATALINA_BASE/conf/keystore \
-storepass changeit -import -alias mitCA -trustcacerts -file mitCA.pem
```

- Finally, when you get the signed certificate from your CA, import it into the keystore with a command like the following example: (cert is in the file *signed-cert.pem*)

```
$JAVA_HOME/bin/keytool -keystore $CATALINA_BASE/conf/keystore \
  -storepass changeit \
  -import -alias tomcat -trustcacerts -file signed-cert.pem
```

Since you now have a signed server certificate in your keystore, you can, obviously, skip the next steps of installing a signed server certificate and the server CA's certificate.

b. Create a Java keystore for your server with the password *changeit*, and install your server certificate under the alias *"tomcat"*. This assumes the certificate was put in the file *server.pem*:

```
$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA -keystore \
  $CATALINA_BASE/conf/keystore -storepass changeit
```

When answering the questions to identify the certificate, be sure to respond to "First and last name" with the fully-qualified domain name of your server (e.g. *test-dspace.myuni.edu*). The other questions are not important.

c. Optional – ONLY if you need to accept client certificates for the X.509 certificate stackable authentication module See the configuration section for instructions on enabling the X.509 authentication method. Load the keystore with the CA (certifying authority) certificates for the authorities of any clients whose certificates you wish to accept. For example, assuming the client CA certificate is in *client1.pem*:

```
$JAVA_HOME/bin/keytool -import -noprompt -storepass changeit \
  -trustcacerts -keystore $CATALINA_BASE/conf/keystore -alias client1 \
  -file client1.pem
```

d. Follow the procedure in the section above to add another Connector tag, for the HTTPS port, to your *server.xml* file.

Using SSL on Apache HTTPD with mod_jk

When using Apache 2.4.2 (and lower) in front of a DSpace webapp deployed in Tomcat, mod_proxy_ajp and possibly mod_proxy_http breaks the connection to the back end (Tomcat) prematurely leading to response mixups. This is reported as bug CVE-2012-3502 (http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-3502) of Apache and fixed in Apache 2.4.3 (see http://www.apache.org/dist/httpd/CHANGES_2.4) . The 2.2.x branch hasn't shown this problem only the 2.4.x branch has.

If you choose Apache HTTPD as your primary HTTP server, you can have it forward requests to the Tomcat servlet container via Apache Jakarta Tomcat Connector. This can be configured to work over SSL as well. First, you must configure Apache for SSL; for Apache 2.0 see Apache SSL/TLS Encryption for information about using mod_ssl.

If you are using X.509 Client Certificates for authentication: add these configuration options to the appropriate *httpd* configuration file, e.g. *ssl.conf*, and be sure they are in force for the virtual host and namespace locations dedicated to DSpace:

```
##  SSLVerifyClient can be "optional" or
        "require"
        SSLVerifyClient optional
        SSLVerifyDepth  10
        SSLCACertificateFile
        path-to-your-client-CA-certificate
        SSLOptions StdEnvVars ExportCertData
```

Now consult the Apache Jakarta Tomcat Connector documentation to configure the *mod_jk* (note: **NOT** *mod_jk2*) module. Select the AJP 1.3 connector protocol. Also follow the instructions there to configure your Tomcat server to respond to AJP.

To use SSL on Apache HTTPD with mod_webapp consult the DSpace 1.3.2 documentation. Apache have deprecated the *mod_webapp* connector and recommend using *mod_jk*.

To use Jetty's HTTPS support consult the documentation for the relevant tool.

The Handle Server

First a few facts to clear up some common misconceptions:

- You don't **have** to use CNRI's Handle system. At the moment, you need to change the code a little to use something else (e.g PURLs) but that should change soon.
- You'll notice that while you've been playing around with a test server, DSpace has apparently been creating handles for you looking like *hdl:123456789/24* and so forth. These aren't really Handles, since the global Handle system doesn't actually know about them, and lots of other DSpace test installs will have created the same IDs. They're only really Handles once you've registered a prefix with CNRI (see below) and have correctly set up the Handle server included in the DSpace distribution. This Handle server communicates with the rest of the global Handle infrastructure so that anyone that understands Handles can find the Handles your DSpace has created.
  If you want to use the Handle system, you'll need to set up a Handle server. This is included with DSpace. Note that this is not required in order to evaluate DSpace; you only need one if you are running a production service. You'll need to obtain a Handle prefix from the central CNRI Handle site.

A Handle server runs as a separate process that receives TCP requests from other Handle servers, and issues resolution requests to a global server or servers if a Handle entered locally does not correspond to some local content. The Handle protocol is based on TCP, so it will need to be installed on a server that can send and receive TCP on port 2641.

1. To configure your DSpace installation to run the handle server, run the following command:

   ```
   [dspace]/bin/dspace make-handle-config [dspace]/handle-server
   ```

   Ensure that *[dspace]/handle-server* matches whatever you have in *dspace.cfg* for the *handle.dir* property.

   a. If you are using Windows, the proper command is:

      ```
      [dspace]/bin/dspace dsrun net.handle.server.SimpleSetup [dspace]/handle-server
      ```

      Ensure that *[dspace]/handle-server* matches whatever you have in *dspace.cfg* for the *handle.dir* property.
2. Edit the resulting *[dspace]/handle-server/config.dct* file to include the following lines in the *"server_config"* clause:

   ```
   "storage_type" = "CUSTOM"
   "storage_class" = "org.dspace.handle.HandlePlugin"
   ```

   This tells the Handle server to get information about individual Handles from the DSpace code.
3. Once the configuration file has been generated, you will need to go to http://hdl.handle.net/4263537/5014 to upload the generated sitebndl.zip file. The upload page will ask you for your contact information. An administrator will then create the naming authority/prefix on the root service (known as the Global Handle Registry), and notify you when this has been completed. You will not be able to continue the handle server installation until you receive further information concerning your naming authority.
4. When CNRI has sent you your naming authority prefix, you will need to edit the *config.dct* file. The file will be found in */[dspace]/handle-server*. Look for *"300:0.NA/YOUR_NAMING_AUTHORITY"*. Replace *YOUR_NAMING_AUTHORITY* with the assigned naming authority prefix sent to you.
5. Now start your handle server (as the dspace user):

   ```
   [dspace]/bin/start-handle-server
   ```

   a. If you are using Windows, the proper command is (please replace "[dspace]\handle-server" with the full path of the handle-server directory):

      ```
      [dspace]/bin/dspace dsrun net.handle.server.Main [dspace]/handle-server
      ```

      Ensure that *[dspace]/handle-server* matches whatever you have in *dspace.cfg* for the *handle.dir* property.

Note that since the DSpace code manages individual Handles, administrative operations such as Handle creation and modification aren't supported by DSpace's Handle server.

Updating Existing Handle Prefixes

If you need to update the handle prefix on items created before the CNRI registration process you can run the *[dspace]/bin/dspace update-handle-prefix script*. You may need to do this if you loaded items prior to CNRI registration (e.g. setting up a demonstration system prior to migrating it to production). The script takes the current and new prefix as parameters. For example:

```
[dspace]/bin/dspace update-handle-prefix 123456789 1303
```

This script will change any handles currently assigned prefix 123456789 to prefix 1303, so for example handle 123456789/23 will be updated to 1303/23 in the database.

Google and HTML sitemaps

To aid web crawlers index the content within your repository, you can make use of sitemaps. There are currently two forms of sitemaps included in DSpace: Google sitemaps and HTML sitemaps.

Sitemaps allow DSpace to expose its content without the crawlers having to index every page. HTML sitemaps provide a list of all items, collections and communities in HTML format, whilst Google sitemaps provide the same information in gzipped XML format.

To generate the sitemaps, you need to run *[dspace]/bin/dspace generate-sitemaps* This creates the sitemaps in *[dspace]/sitemaps/*

The sitemaps can be accessed from the following URLs:

- http://dspace.example.com/dspace/sitemap - Index sitemap
- http://dspace.example.com/dspace/sitemap?map=0 - First list of items (up to 50,000)
- http://dspace.example.com/dspace/sitemap?map=n - Subsequent lists of items (e.g. 50,0001 to 100,000) etc...
  HTML sitemaps follow the same procedure:
- http://dspace.example.com/dspace/htmlmap - Index HTML based sitemap
- etc...

When running *[dspace]/bin/dspace generate-sitemaps* the script informs Google that the sitemaps have been updated. For this update to register correctly, you must first register your Google sitemap index page (*/dspace/sitemap*) with Google at http://www.google.com/webmasters/sitemaps/. If your DSpace server requires the use of a HTTP proxy to connect to the Internet, ensure that you have set *http.proxy.host* and *http.proxy.port* in *[dspace]/config/dspace.cfg*

The URL for pinging Google, and in future, other search engines, is configured in *[dspace]/config/dspace.cfg* using the *sitemap.engineurls* setting where you can provide a comma-separated list of URLs to 'ping'.

You can generate the sitemaps automatically every day using an additional cron job:

```
# Generate sitemaps at 6:00 am local time each day
0 6 * * * [dspace]/bin/dspace generate-sitemaps
```


More information on why we **highly recommend** enabling sitemaps can be found at Search Engine Optimization (SEO).

Statistics

DSpace uses the Apache Solr application underlaying the statistics. There is no need to download any separate software. All the necessary software is included. To understand all of the configuration property keys, the user should refer to DSpace Statistic Configuration for detailed information.

Windows Installation

Essentially installing on Windows is the same as installing on Unix so please refer back to the main Installation Instructions section.

- Download the DSpace source from SourceForge and unzip it (WinZip will do this)
- If you install PostgreSQL, it's recommended to select to install the pgAdmin III tool. It provides a nice User Interface for interacting with PostgreSQL databases.
- For all path separators use forward slashes (e.g. "/"). For example: "C:/dspace" is a valid Windows path. But, be warned that "C:\dspace" IS INVALID and will cause errors.

Checking Your Installation

The administrator needs to check the installation to make sure all components are working. Here is list of checks to be performed. In brackets after each item, it the associated component or components that might be the issue needing resolution.

- System is up and running. *User can see the DSpace home page. [Tomcat/Jetty, firewall, IP assignment, DNS]*
- Database is running and working correctly. *Attempt to create a user, community or collection [PostgreSQL, Oracle] Run the test database command to see if other issues are being report: [dspace]/bin/dspace test-database*
- Email subsystem is running. The user can issue the following command to test the email system. t attempts to send a test email to the email address that is set in dspace.cfg (mail.admin). If it fails, you will get messages informing you as to why, will refer you to the DSpace documentation. *[dspace]/bin/dspace test-email*

Known Bugs

In any software project of the scale of DSpace, there will be bugs. Sometimes, a stable version of DSpace includes known bugs. We do not always wait until every known bug is fixed before a release. If the software is sufficiently stable and an improvement on the previous release, and the bugs are minor and have known workarounds, we release it to enable the community to take advantage of those improvements.

The known bugs in a release are documented in the *KNOWN_BUGS* file in the source package.

Please see the DSpace bug tracker for further information on current bugs, and to find out if the bug has subsequently been fixed. This is also where you can report any further bugs you find.

Common Problems

In an ideal world everyone would follow the above steps and have a fully functioning DSpace. Of course, in the real world it doesn't always seem to work out that way. This section lists common problems that people encounter when installing DSpace, and likely causes and fixes. This is likely to grow over time as we learn about users' experiences.

Common Installation Issues

- **Database errors occur when you run** `ant fresh_install`: There are two common errors that occur.
  - If your error looks like this:

```
[java] 2004-03-25 15:17:07,730 INFO
            org.dspace.storage.rdbms.InitializeDatabase @ Initializing Database
[java] 2004-03-25 15:17:08,816 FATAL
            org.dspace.storage.rdbms.InitializeDatabase @ Caught exception:
[java] org.postgresql.util.PSQLException: Connection refused. Check
            that the hostname and port are correct and that the postmaster is
            accepting TCP/IP connections.
[java]     at
            org.postgresql.jdbc1.AbstractJdbc1Connection.openConnection(AbstractJd
bc1Connection.java:204)
[java]     at org.postgresql.Driver.connect(Driver.java:139)
```

    it usually means you haven't yet added the relevant configuration parameter to your PostgreSQL configuration (see above), or perhaps you haven't restarted PostgreSQL after making the change. Also, make sure that the *db.username* and *db.password* properties are correctly set in *[dspace]/config/dspace.cfg*. An easy way to check that your DB is working OK over TCP/IP is to try this on the command line:

```
psql -U dspace -W -h localhost
```

    Enter the *dspace* database password, and you should be dropped into the psql tool with a *dspace=>* prompt.
  - Another common error looks like this:

```
[java] 2004-03-25 16:37:16,757 INFO
            org.dspace.storage.rdbms.InitializeDatabase @ Initializing Database
[java] 2004-03-25 16:37:17,139 WARN
            org.dspace.storage.rdbms.DatabaseManager @ Exception initializing DB
            pool
[java] java.lang.ClassNotFoundException: org.postgresql.Driver
[java]     at java.net.URLClassLoader$1.run(URLClassLoader.java:198)
[java]     at java.security.AccessController.doPrivileged(Native
           Method)
[java]     at
           java.net.URLClassLoader.findClass(URLClassLoader.java:186)
```

    This means that the PostgreSQL JDBC driver is not present in *[dspace]/lib*. See above.

- **GeoLiteCity Database file fails to download or install, when you run** `ant fresh_install`: There are two common errors that may occur:
  - If your error looks like this:

```
[get] Error getting http://geolite.maxmind.com/download/geoip/database/GeoLiteCity.dat.gz to /usr
/local/dspace/config/GeoLiteCity.dat.gz

BUILD FAILED
/dspace-release/dspace/target/dspace-1.8.0-build/build.xml:931: java.net.ConnectException:
Connection timed out
```

it means that you likely either (a) don't have an internet connection to download the necessary GeoLite Database file (used for DSpace Statistics), or (b) the GeoLite Database file's URL is no longer valid. You should be able to resolve this issue by following the "Manually Installing/Updating GeoLite Database File" instructions above.

- Another common message looks like this:

```
[echo] WARNING : FAILED TO DOWNLOAD GEOLITE DATABASE FILE
[echo]          (Used for DSpace Solr Usage Statistics)
```

Again, this means the GeoLite Database file cannot be downloaded or is unavailable for some reason. You should be able to resolve this issue by following the "Manually Installing/Updating GeoLite Database File" instructions above.

General DSpace Issues

- **Tomcat doesn't shut down**: If you're trying to tweak Tomcat's configuration but nothing seems to make a difference to the error you're seeing, you might find that Tomcat hasn't been shutting down properly, perhaps because it's waiting for a stale connection to close gracefully which won't happen.
  - To see if this is the case, try running: `ps -ef | grep java` and look for Tomcat's Java processes. If they stay around after running Tomcat's *shutdown.sh* script, trying running `kill` on them (or `kill -9` if necessary), then starting Tomcat again.
- **Database connections don't work, or accessing DSpace takes forever**: If you find that when you try to access a DSpace Web page and your browser sits there connecting, or if the database connections fail, you might find that a 'zombie' database connection is hanging around preventing normal operation.
  - To see if this is the case, try running: `ps -ef | grep postgres`
  - You might see some processes like this:

```
dspace 16325  1997  0  Feb 14  ?        0:00 postgres: dspace dspace    127.0.0.1 idle in
transaction
```

This is normal. DSpace maintains a 'pool' of open database connections, which are re-used to avoid the overhead of constantly opening and closing connections. If they're 'idle' it's OK; they're waiting to be used.

- However sometimes, if something went wrong, they might be stuck in the middle of a query, which seems to prevent other connections from operating, e.g.:

```
dspace 16325  1997  0  Feb 14  ?        0:00 postgres: dspace dspace    127.0.0.1 SELECT
```

This means the connection is in the middle of a *SELECT* operation, and if you're not using DSpace right that instant, it's probably a 'zombie' connection. If this is the case, try running `kill` on the process, and stopping and restarting Tomcat.