

Installing DSpace on OSX (2014)

The process below is an efficient way of setting this all up on OSX. This guide was compiled on May 7, 2014, using a new MacBook Pro, running OSX 10.9 (Maverick), and grabbing the latest DSpace, something like DSpace 4.x or DSpace 5.x. There's no one-click installer, so some command-line proficiency is required.

The primary things needed for DSpace are: Java JDK, Tomcat, Postgres, and then git, maven, and ant for compiling/installing.

Installation Steps

- [Install Brew](#)
- [Install Java 7 JDK](#)
- [Install Git, Maven, Ant, Tomcat](#)
- [Install Postgres.App](#)
- [Configure Tomcat](#)
- [PRO-Tips:](#)
 - [Mail Catcher to trap and view emails locally](#)
 - [IntelliJ IDEA for a great coding environment](#)
 - [Sublime Text for a great light-weight config editing environment](#)
 - [pgAdmin3 for better PostgreSQL query environment](#)
 - [respace.sh - Single-Step Redeploy Script](#)

Install Brew

Brew is a package installer/manager for OSX. Its great as it allows you to properly install a number of programs (i.e. dependencies) from the command line.

Install brew, and follow instructions from <http://brew.sh>

```
brew doctor
```

```
brew update
```

Install Java 7 JDK

Download and Install Java 7 JDK, this step should be done before installing tomcat, and IntelliJ: <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Alternative Text Editors

 If vi/vim is too technical for you, you can use nano or sublime text (pro-tip at bottom) instead. Then, instead of `vi ~/.bash_profile`, it would be `nano ~/.bash_profile` or `subl ~/.bash_profile`

vi is just common among systems people

Configure JAVA_HOME for Java 7

```
vi ~/.bash_profile
```

Add:

```
export JAVA_HOME=$(/usr/libexec/java_home -v 1.7)
```

Install Git, Maven, Ant, Tomcat

```
brew install git maven ant tomcat tomcat-native
```

Install Postgres.App

Install postgres.app from: <http://postgresapp.com/>

Once installed, click the taskbar elephant, and "open psql", psql is the command-line-interface for Postgres. You can create database user, create database, or perhaps for future tasks, run direct SQL queries against DSpace.

[blocked URL](#)

The SQL to make the Postgres database-user is:

```
CREATE ROLE dspace with SUPERUSER LOGIN PASSWORD 'dspace';
```

The SQL to make the Postgres database is:

```
create database "dspace" with owner = dspace encoding='utf8' tablespace=pg_default lc_collate = 'C' lc_ctype='en_US.UTF-8' template template0;
```

"\q" gets you out of PSQL.

Configure Tomcat

Tell tomcat that tomcat-native is installed.

```
vi /usr/local/Cellar/tomcat/7.0.53/libexec/bin/setenv.sh
```

Add:

```
CATALINA_OPTS="$CATALINA_OPTS -Djava.library.path=/usr/local/Cellar/tomcat-native/1.1.30/lib"
```

Further, in order to reduce some memory errors that your bound to run into, bump up the memory settings (-Xms512m -Xmx512m -XX:MaxPermSize=256m), so combined it becomes:

```
CATALINA_OPTS="$CATALINA_OPTS -Xms512m -Xmx512m -XX:MaxPermSize=256m -Djava.library.path=/usr/local/Cellar/tomcat-native/1.1.30/lib"
```

Edit your tomcat server.xml config, to let it know about the DSpace webapps that will be there.

```
vi /usr/local/Cellar/tomcat/7.0.53/libexec/conf/server.xml
```

After the line:

```
<Host name="localhost" appBase="webapps"
    unpackWARs="true" autoDeploy="true">
```

Add the following lines:

```
<!-- Define a new context path for all DSpace web apps -->
<Context path="/xmlui" docBase="/dspace/webapps/xmlui" allowLinking="true"/>
<Context path="/sword" docBase="/dspace/webapps/sword" allowLinking="true"/>
<Context path="/oai" docBase="/dspace/webapps/oai" allowLinking="true"/>
<Context path="/jspui" docBase="/dspace/webapps/jspui" allowLinking="true"/>
<Context path="/lri" docBase="/dspace/webapps/lri" allowLinking="true"/>
<Context path="/solr" docBase="/dspace/webapps/solr" allowLinking="true"/>
<Context path="/rest" docBase="/dspace/webapps/rest" allowLinking="true"/>
```

Create the /dSPACE directory, below where I have "myUserName", customize that for your local setup, i.e. "peter".

```
sudo mkdir /dSPACE
sudo chown -R myUserName /dSPACE
```

Clone DSpace from GitHub, to your Projects directory, and begin the compile process.

```
cd ~
mkdir Projects
cd ~/Projects
git clone https://github.com/DSpace/DSpace.git
cd DSpace
git checkout dSPACE-4_x
mvn package
cd dSPACE/target/dSPACE-installer/
ant fresh_install
```

Note



Be cautious about running fresh_install. This is only for a new computer setup. Do not run fresh install if you are doing anything other than a fresh install of DSpace, i.e. you are upgrading, or already have existing data in your DSpace database, it will ruin that existing data.

Make an admin account for your DSpace:

```
/dSPACE/bin/dSPACE create-administrator
```

Start tomcat, and visit a DSpace webapp in your browser.

```
catalina start
open http://localhost:8080/xmlui
```

From here, you should have a ready to go system. You can always check the official installation documentation too: [Installing DSpace](#). It would be best to keep this Install on OSX guide up-to-date, so if you run into issues, please leave a comment.

PRO-Tips:

Mail Catcher to trap and view emails locally

For development, send all emails to "mailcatcher", a dummy email account, that runs on localhost, so that real emails don't get sent out to real people.

edit build.properties (then recompile maven+ant), or edit /dSPACE/config/dSPACE.cfg:

```
# SMTP mail server
mail.server = localhost
# SMTP mail server alternate port (defaults to 25)
mail.server.port = 1025
```

If your doing anything ruby/gem/rails on your computer, you'll likely want to use RVM anyways.

```
\curl -sSL https://get.rvm.io | bash -s stable --ruby
```

Then install mailcatcher gem

```
gem install mailcatcher
```

Then to start mailcatcher:

```
mailcatcher
```

Starting MailCatcher

```
==> smtp://127.0.0.1:1025
```

```
==> http://127.0.0.1:1080
```

*** MailCatcher runs as a daemon by default. Go to the web interface to quit.

Once configured, you can view your "inbox", for outgoing messages from DSpace in your browser at <http://localhost:1080>

IntelliJ IDEA for a great coding environment

If you plan on doing significant Java programming, then it is recommended to use IntelliJ. The Ultimate version is best suited for this work, either start the trial, or use/buy a license. DSpace Committers have access to an "Open Source Project License", hooray JetBrains! IntelliJ gives you code-completion i.e. Conf -> ConfigurationManager, and detects syntax errors before you spend time compiling. There are other alternatives, such as Eclipse and NetBeans, but many DSpace developers stick with IntelliJ IDEA.

Sublime Text for a great light-weight config editing environment

If I have to quickly view some config files, its preferable to open that with Sublime. Sublime Text 3 is their latest-and-greatest version.

I like to have command-line integration, so you should run:

```
ln -s "/Applications/Sublime Text.app/Contents/SharedSupport/bin/subl" ~/bin/subl
```

If you get a strange error that "bin/subl: No such file or directory", then run the following, and then the "ln -s" command again:

```
mkdir ~/bin
```

Once you are all set, then to open a single file (dspace.cfg) for editing its:

```
subl /dspace/config/dspace.cfg
```

Or to open a directory of config files, you can pass it the directory:

```
subl /dspace/config
```

pgAdmin3 for better PostgreSQL query environment

I would recommend pgadmin3, for a better query-environment than PSQL. Google, download, install it.

To get pgadmin3 setup, after it is installed:

File -> Add Server

Name: localhost

Host: localhost

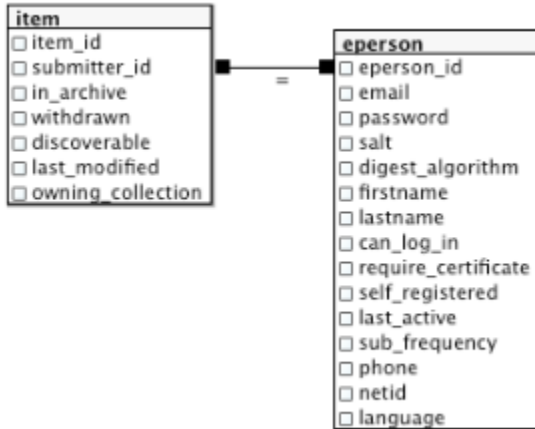
Username: dspace

Password: dspace

Then, navigate the "Server Groups" tree: Server Groups > Server > localhost > Databases > dspace > Schemas > public > Tables (41)

To make a simple query, you could right click on table such as "eperson" > View Data > View Top 100 Rows, and that will give you back a matrix /spreadsheet.

To make an arbitrary query, click Tools > Query Tool (Command + E). There is SQL Editor (write and execute raw SQL). Then there is Graphical Query Builder, this is a fun tool, as you can drag a table from the left-hand-side list, onto the query board. Its similar to MS Access, if you've ever constructed visual database queries there. An easy query would be to drag "item" to the board, and "eperson" to the board. Then hover over eperson_id (within eperson table), and click and drag that to item table submitter_id. This lets you easily construct a query that joins a table. When ready to see the data, click the green Execute Query button at the top. When it executes, it converts the graphic to SQL text, and runs that. To edit your query, you can either go back to graphical mode, or edit the generated SQL.



That then outputs a "spreadsheet" of data.

Data Output	Explain	Messages	History						
	item_id integer	submitter_id integer	in_archive boolean	withdrawn boolean	discoverable boolean	last_modified timestamp with time zone	owning_collection integer	eperson_id integer	email character varying(64)
1	44	1	t	f	t	2014-05-07 19:49:50.9		1	peter@longsight.com
2	38	1	t	f	t	2014-05-07 19:49:50.6		1	peter@longsight.com

Note: A classically trained database admin may cringe when they see you use the Graphical Query Builder to make joins, but its only because they have too much time on their hands.

respace.sh - Single-Step Redeploy Script

If you are doing development, and find yourself having to make changes to .java files, then cd to your DSpace-source directory, then maven, then cd dspace/target/..., then ant, then stop/start tomcat, and you find that tedious, why not automate that process?

Introducing, respace.sh. <https://gist.github.com/peterdietz/7893272>

To add it:

```
curl https://gist.githubusercontent.com/peterdietz/7893272/raw/2f8549cadd42a5f938abac173af1b046fc57cf2a/respace.sh
> ~/Projects/DSpace/respace.sh
```

```
chmod +x ~/Projects/DSpace/respace.sh
```

Then to run it, from your ~/Projects/DSpace directory:

```
./respace.sh
```

I've gotten fancy with this respace script, it sends a message to OSX's Notification Center, letting you know when it finishes.