Building and deploying cdrlite

The CDR-Lite is a Grails application. To get this standalone version up and running, please follow the steps below. Prerequisites:

Knowlege of Java and Git is assumed.

Install the latest JDK 7 per instructions for your environment. JDK 8 is not supported at this time.

Install PostgreSQL 9.3

Create a user and password. By default, the cdrlite will try to connect to the database as “cdr/admin.” The cdr user will not need any special privileges. Through hibernate, it will create tables and sequences only.

Install Grails 2.4.4 per instructions for your environment. Later versions of Grails are not supported at this time. Earlier versions may not be compatible. This is very important.

See here for more Grails installation information: <http://grails.org/doc/latest/guide/gettingStarted.html>

Clone the project form GitHub:

<https://github.com/NCIP/CDR-Lite>

After cloning from GitHub, open a console window, navigate to the cdrlite folder and run the following commands:

0) set JAVA\_OPTS=-XX:MaxPermSize=128m -XX:PermSize=512m -Xms1024m -Xmx2048m (This is a generous estimate. Default memory settings are not enough.)

In your local environment, open the file

[cdrlite/grails-app/conf/DataSource.groovy](https://github.com/NCIP/CDR-Lite/blob/master/cdrlite/grails-app/conf/DataSource.groovy)

Update the entries at

|  |
| --- |
| environments { |
|  | development { |
|  | dataSource { |

To match the username and password you created in postgreSQL

1) grails RunApp

This will download all project plug-ins, resolve dependencies and determine if your environment is set up properly. If all goes well, you should see the following message: Server running. Browse to http://localhost:8080/cdrlite

2) Open a browser if your choice and enter http://localhost:8080/cdrlite

3) You may log in with the following test accounts:

admin/admin

If you run into errors, try the following:

1) Grails clean

2) Grails upgrade (say yes)

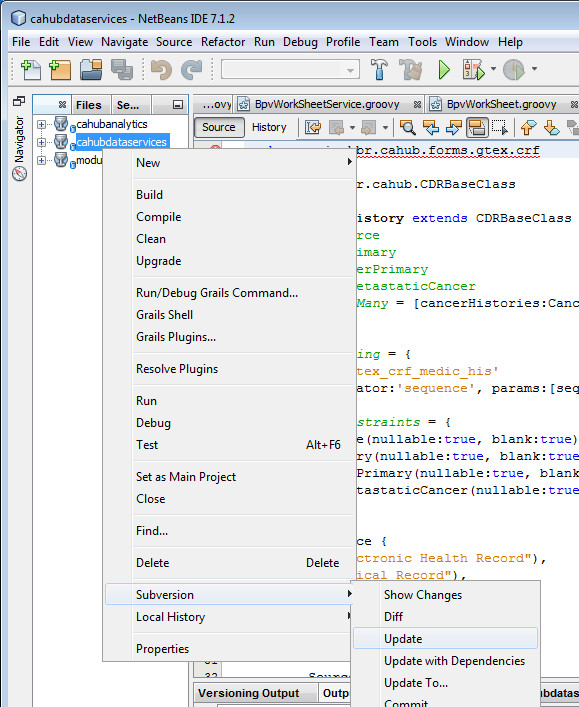
3) Grails RunApp

The CDR was originally developed in the NetBeans IDE against Oracle XE. This version is using PostgreSQL 9.3.

Following are the basic steps involved in building the CDR Lite application .WAR file and deploying the same on a server.

Updating latest code from SVN:

Right click on the project in the Netbeans IDE and get the latest updates to the existing code by doing a subversion update. If unsure about the location of the source code trunk for SVN, please check with your project lead.



Currently the SVN trunk to access source code for application cdrlite is set to:

<https://cahubrepo.ncifcrf.gov/svn/grails/cdrlite/trunk> . Please check with your project lead if there are any changes or problems accessing this link.

BUILD WAR FILE:

Make sure tomcat is not running within the Netbeans IDE before you start.

The **wars** batch script is found in the main project folder and is run from the MS-DOS command line prompt. The script executes a Grails command and builds the war file for a requested environment. An environment could be any one of the following: Production, Test, QA, DEV. Depending on the script options, the war file can be built for **all** or for **any single** environment. The environment definitions are found in cdrlite/conf/DataSource.groovy.

How to run the SCRIPT?

At the command line prompt change directory to the project folder:

**C:\dev\svn\cdrlite>wars test (build war file for the test environment)**

**C:\dev\svn\cdrlite>wars (builds multiple war files for every environment)**

**\* run ‘grails upgrade’ at DOS prompt if applicationContext.xml not found error is seen**

\*Please note: The above command may take a long time to complete (average time is about 3 -7 minutes per environment)

Naming convention for the war file:

The war file is built with the environment name and the current version appended to it. The war file(s) built from above step can be found in the **target** directory within the main project folder. The version number is read from the application.properties file found in the main project directory.

The war file in this example is built as version 3.5 and targeted for the ec2cdr environment:

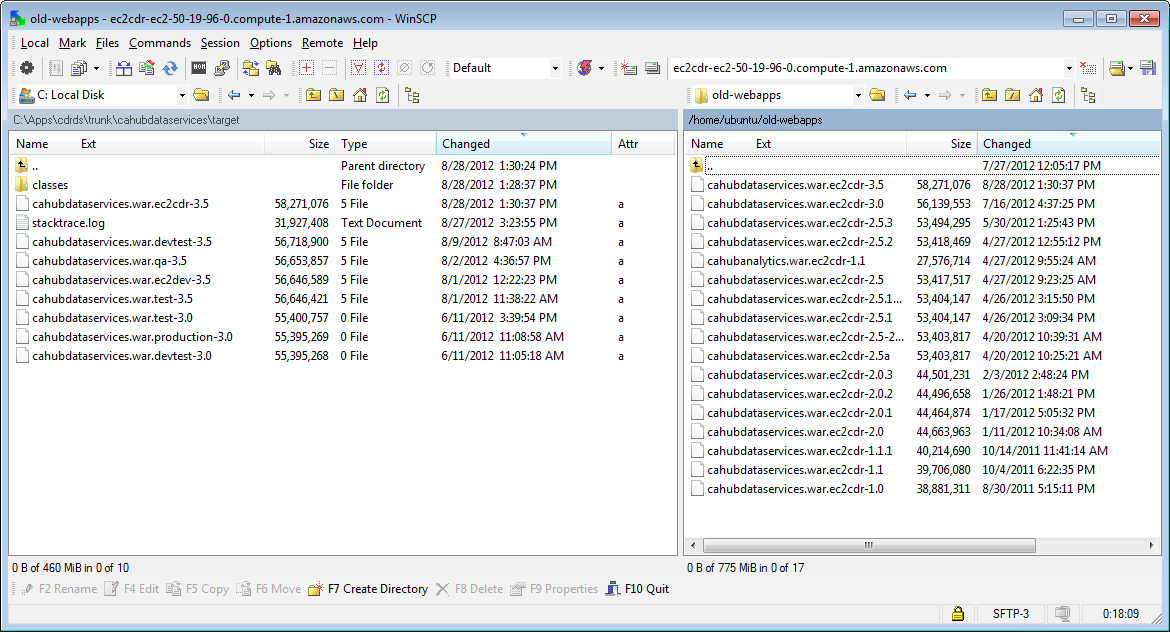
cdrlite.war.ec2cdr-3.5

Update WAR file to the server:

Using WinSCP, remote copy the War file to the server for which it is built[[1]](#footnote-1).

Example: In the case of test, login as ‘jenkins’ and copy the war file from the target directory to the **/var/storage/old-webapps** directory.

The Config.groovy and DataSource.groovy files for the correct environment will be embedded in the war file.



Steps to update the application on the server with the latest version

Using Putty, login via ssh to the server for which the War file has been built.

Open two sessions of putty so that the logs can be viewed on one, while the Tomcat server is being stopped and restarted on the other

On the first session (A), become root (sudo su -) and change directory upon logging in to: **/var/log/tomcat7** and tail the log file: **tail -200f catalina.out**

On the second session (B), become root (sudo su -) and change directory to **/var/lib/tomcat7/webapps**. Set your session to root by typing **sudo su –** and entering the appropriate password. You need to be root to stop and start tomcat.

At this point, email to the team that the server is being stopped for deploying a new version.

Run the following commands on session B:

**Make a copy of the current war file for fallback and recovery purposes:**

**/var/lib/tomcat7/webapps$ cp –p cdrlite.war cdrlite.war.backup.PEH.28-Aug-2012**

**Copy the war file from the {upload} dir (in this case /home/ubuntu/old-webapps) to the tomcat webapps dir**.

**Eg: /var/lib/tomcat7/webapps$ cp –p /home/ubuntu/old-webapps/cdrlite.war.ec2cdr-3.5 .**

**Check if tomcat is running**

**ps –aef | grep java**

**You should see something like:**

tomcat7 3059 1 0 Jul08 ? 00:10:23 /usr/lib/jvm/java-7-oracle/bin/java -Djava.util.logging.config.file=/var/lib/tomcat7/conf/logging.properties -Xms2048m -Xmx2048m -XX:PermSize=512m -XX:MaxPermSize=512m -Djavax.sql.DataSource.Factory=org.apache.commons.dbcp.BasicDataSourceFactory -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djava.endorsed.dirs=/usr/share/tomcat7/endorsed -classpath /usr/share/tomcat7/bin/bootstrap.jar:/usr/share/tomcat7/bin/tomcat-juli.jar -Dcatalina.base=/var/lib/tomcat7 -Dcatalina.home=/usr/share/tomcat7 -Djava.io.tmpdir=/tmp/tomcat7-tomcat7-tmp org.apache.catalina.startup.Bootstrap start

**Stop Tomcat**

**/etc/init.d/tomcat7 stop**

(View the logs scrolling on the screen on the other opened session (A).)

Remove the old war file and the (exploded) project dir..

**/var/lib/tomcat7/webapps$ rm –rf** cdrlite.war

**/var/lib/tomcat7/webapps$ rm –rf** cdrlite

Copy the uploaded war file from its environment and version-specific name to something tomcat will recognize and deploy upon startup:

**/var/lib/tomcat7/webapps$ cp –p cdrlite.war.ec2cdr-3.5 cdrlite.war**

Start Tomcat again. This will deploy the war file and re-create the project directory in tomcat7/webapps/ .

**/etc/init.d/tomcat7 start**

Watch the scrolling logfile in session (A) and the output from “top” in session B. Scan for any stacktraces in the logs and keep an eye on the java process in top until the application server has completed startup. You should see the following message at the bottom of the log file in session A:

Configuring Spring Security ...  
Configuring Spring Security LDAP ...  
Configuring Spring Security UI ...  
Aug 28, 2012 5:58:52 PM org.apache.coyote.http11.Http11Protocol start  
INFO: Starting Coyote HTTP/1.1 on http-8080  
Aug 28, 2012 5:58:52 PM org.apache.jk.common.ChannelSocket init  
INFO: JK: ajp13 listening on /0.0.0.0:8009  
Aug 28, 2012 5:58:52 PM org.apache.jk.server.JkMain start  
INFO: Jk running ID=0 time=0/21 config=null  
Aug 28, 2012 5:58:52 PM org.apache.catalina.startup.Catalina start  
INFO: Server startup in 136410 ms

CHECK TO SEE IF APPLICATION IS RUNNING AND CHANGES HAVE BEEN IMPLEMENTED

Open your browser and navigate to the application URL on the server where the war file has been deployed. Check if application is running as expected and all the changes for this version are intact.

*Noto Bene:* In our environment, an apache proxy, a redirect and an SSL certificate are in place, so that <http://cdrlite.ncifcrf.gov/> automatically takes the user to <https://cdrlite.ncifcrf.gov/cdrlite/> . The scope of this document does not cover server configuration and administration, but you may wish to consider something similar in your installation environment. Such configuration is highly dependent on the environment where the application will be running.

1. This manual process has been replaced by an automated deployment tool, Jenkins, but the manual process will work, and documentation of the deployment tool is out of scope for this document. [↑](#footnote-ref-1)