

CORE : Auto Deploy Project Installation

This page last changed on Oct 15, 2008 by Hebell.

Contents

- [Introduction](#)
- [Infrastructure](#)
- [Environment](#)
- [Properties](#)

Links

- [Build and Deployment Automation Handbook](#)
- [Auto Deploy Project Configuration](#)

Introduction

This describes the installation for all Auto Deploy Project. A project is labelled "Auto Deploy" once it conforms to the guidelines described in the [Build and Deployment Automation Handbook](#). Some example projects include the caDSR Sentinel Tool and CDE Curation Tool. To successfully install such a project requires creating an environment by first installing infrastructure software and second creating a directory structure to hold the project deployed files. This is described to the process using a set of Properties.

The build and deployment process starts after the source extraction, e.g. checkout, download, etc. These Ant tasks do not care where the source is stored or if it has been changed. There are no targets in the build.xml which assume a specific SCM environment or tool.

Infrastructure

Each project will have an Installation Document detailing dependent software and their versions. Dependent software must be installed on either the build machine, target deployment machine or both. Common packages includes:

- Build Machine
 - Ant
 - JDK
 - Oracle SQL/Plus
- Deployment Machine
 - JBoss
 - BASH
 - SSH

The software products listed above may also have dependencies. The installation for each must be complete and working prior to the build and deployment of an Auto Deploy project. Once JBoss installation is complete the following additional steps are required.

JBoss conf/log4j.xml

Each project wants to record log messages in separate files. However there is only one (1) log4j.xml file in a JBoss configuration. For an Auto Deploy project to successfully add itself into the log4j.xml,

the default file included in the JBoss installation must be broken apart. From the server/default/conf directory...

1. Create a directory named "log4j"
2. Copy log4j.xml to ./log4j/.
3. Create ./log4j/log4j-header.xml using the <log4j> opening element tag and all appenders from log4j.xml.
4. Create ./log4j/log4j-footer.xml using all categories and the </log4j> closing element tag from log4j.xml.

During the project deployment a specific appender and category file will be copied into the ./log4j/ directory and all files will be merged into a single log4j.xml and moved to the ./conf/ directory.

Warning any hand edits made to the ./conf/log4j.xml will be lost. To preserve hand edits and mix with Auto Deploy projects, create two (2) files in the ./log4j/ directory named, handedits-log4j-appender.xml and handedits-log4j-category.xml, respectively. Add the appenders and categories to the appropriate files. If changes are needed to original default settings, edit the log4j-header.xml and log4j-footer.xml.

JBoss conf/login-config.xml

Although the auto deploy project will not change the content of the login-config.xml file it may add additional files in the conf/ directory with the name format "project-login-config.xml", where "project" is the name of the owning project, e.g. formbuilder-login-config.xml is owned and managed by the deployment of the caDSR Form Builder Tool. The content of these files should not be altered.

SSHEXEC

The Ant build.xml uses SSHEXEC to perform remote deployments when the target deployment machine is not the same as the build machine. Perform standard installation using a valid certificate.

Database, etc

Whatever database dependency is documented for the project, e.g. Oracle for the caDSR, access is provided via JAR files which must be included on the class path. At NCICB for JBoss Web applications these are copied into the server/default/lib directory and consequently the project build will not include these in WAR and EAR files. For projects which have non-web applications the JAR needed to run on the NCICB servers is included in the <project>/lib directory. To avoid potential problems, check the project dependencies mentioned earlier. When using versions other than that specifically documented, copy the JAR files installed on the target deployment machine into the JBoss class path and/or <project>/lib directory as appropriate.

Environment

As mentioned above, the installation process expects the source to already be extracted and available to build and deploy. This may be accomplished via the NCICB Download Site, checkout from the SCM repository, browsed via GForge or copied from another media. The scripts require the project organization and structure remain intact. The specific content however may or may not be edited as desired.

The build.xml in all Auto Deploy projects contain two (2) targets for a complete build and deployment, "build-all" and "deploy". As the names imply there are two steps to completing a project installation. To execute the process from a command line, go into the project root directory and enter

```
ant -f build.xml -DPROP.FILE=build.properties build-all deploy
```

The content of build.properties is described below in the **Properties** section.

build-all

This target builds all project deliverables and places them in a directory named "deployment-artifacts" under the project root. Subdirectories are created to group the artifacts based on the intended destination, e.g. deployment-artifacts/jboss are files which will be place under JBoss.

deploy

This target distributes all artifacts to the appropriate locations. At NCICB the "remote" option (see below) is used because the build machine is not the same as the deployment machine. Although the process supports a local build it is not often used and may not be reliable. If errors occur during a local deployment refer to the remote deployment targets in the build.xml.

Anthill, etc.

A number of build control utilities exist to assist with managing Auto Deploy projects. These utilities are not required for the project installation to complete successfully.

Properties

The installation and environment are described to the Ant build.xml using properties. Each property has a specific use and format. Although quotes and apostrophies appear in the descriptions below, do **not** include them in the property value, e.g. JDEBUG=on is correct, JDEBUG='on' is not correct. All property names are case sensitive and must be upper case.

Property	Description	Restrictions
JDEBUG	Controls the debug information included in the .class files.	'on' (recommended) includes line number information, 'off' removes all debug information
CADSR.DS.USER	The caDSR user id required by the project to manage the connection pool.	Must be created as a valid user in the caDSR database prior to project installation.
CADSR.DS.PSWD	The password for the CADSR.DS.USER account.	It is recommended to always create a password for the account.
CADSR.DS.URL	The caDSR database connection URL.	Must be in the form <server>:<port>:<SID>
CADSR.DS.TNS.ENTRY	The Oracle tnsnames.ora entry for the caDSR database.	Typically the <SID>
TIER	The tier code. At NCICB development occurs on one machine and is promoted to others during the project release life cycle, e.g. the DEV tier used for development is indicated as '-dev'.	This may be empty, e.g. TIER=
TIER.UPPER	The upper case notation for the tier, e.g. Stage is STAGE.	This may be empty, e.g. TIER.UPPER=
TIER.NAME	The display name of the tier, e.g. DEV is Development.	This may not be empty, e.g. TIER.NAME=Production indicates the Production deployment tier at NCICB along with TIER.UPPER= and TIER=.
JBOSS.HOME.DIR	The deployment machine root relative directory for the JBoss home directory, e.g. /usr/local/jboss405	Must not terminate with a '/' (end of path character).

JBOSS.SERVER.DIR	The deployment machine root relative directory for the JBoss server, e.g. <code>\${JBOSS.HOME.DIR}/server/default</code>	May reference previous properties, e.g. <code>\${JBOSS.HOME.DIR}</code> . Must not terminate with a '/'
JBOSS.CONF.DIR	The deployment machine root relative directory for the JBoss configuration files, e.g. <code>\${JBOSS.SERVER.DIR}/conf</code>	May reference previous properties, e.g. <code>\${JBOSS.SERVER.DIR}</code> . Must not terminate with a '/'
JBOSS.DEPLOY.DIR	The deployment machine root relative directory for the JBoss deployed application files, e.g. <code>\${JBOSS.SERVER.DIR}/deploy</code>	May reference previous properties, e.g. <code>\${JBOSS.SERVER.DIR}</code> . Must not terminate with a '/'
JBOSS.LOG.DIR	The deployment machine root relative directory for the JBoss log files, e.g. <code>\${JBOSS.SERVER.DIR}/log</code>	May reference previous properties, e.g. <code>\${JBOSS.SERVER.DIR}</code> . Must not terminate with a '/'
TOOL.ROOT.DIR	The deployment machine root relative project deployment application root, e.g. <code>/local/content/cadsrsentinel/</code>	Must terminate with a '/'.
TOOL.LOG.DIR	The deployment machine root relative project log files, e.g. <code>/local/content/autorun_log</code>	Must not terminate with a '/'.
TEST	Indicates automated Junit based tests should be run as part of the deployment	Planned for future use, always use 'false'.
TEST.VALID.USER	A valid caDSR user id used for testing.	Must be created prior to project deployment. Ignored when <code>TEST=false</code> .
TEST.VALID.PSWD	The password for <code>TEST.VALID.USER</code> .	Must be created prior to project deployment. Ignored when <code>TEST=false</code> .
TEST.BAD.USER	An invalid caDSR user id.	Must not be a valid user id. Ignored when <code>TEST=false</code> .
TEST.BAD.PSWD	An invalid password.	Must not be a valid password for any test account. Ignored when <code>TEST=false</code> .
DEPLOY.LOCATION	The relative location of the target deployment machine.	'remote' when the build and target deployment machine are physically different, 'local' when the build and target deployment machine are the same.
SCP.USER	The SSH user id to connect to the remote target deployment machine.	Ignored when <code>DEPLOY.LOCATION=local</code> .
SCP.HOST	The target deployment host machine name, e.g. <code>server.nci.nih.gov</code>	Ignored when <code>DEPLOY.LOCATION=local</code> .
SCP.PATH		

	The target deployment directory root for the artifact files, e.g. /local/home/build	Must be owned by the SCP.USER account. Ignored when DEPLOY.LOCATION=local.
SCP.KEYFILE	The keyfile name for the security certificate to allow using SCP.USER, e.g. \${user.home}/.ssh/keyfile	Avoids exposing the SCP.USER password and restricts access to the target deployment machine. Ignored when DEPLOY.LOCATION=local.
SCP.PASSPHRASE	The security pass phrase.	Ignored when DEPLOY.LOCATION=local.
SCP.PORT	The SSEXEC port number on the target deployment machine.	Ignored when DEPLOY.LOCATION=local.
SQL.EXE	A client which can execute PL/SQL scripts via command line, i.e. unattended, e.g /app/oracle/product/10gClient/bin/sqlplus	Must be installed on the build machine.