Development Support:How To Perform a Release

From pitawiki

This document will help walk you through the entire release process including the initial set-up and the steps to perform a complete release. The details describe here reflect the standard configuration for all software projects at CDM; however, some variation of the configuration or overall process may exist.

Contents

Setting Up a Project for Release

Before performing a release for the first time, be sure the following criteria are met:

- Project includes SCM information ('scm' element included in the POM which points to the location of the source code in SVN).
  Note: The 'scm' element must point to the trunk or branch that you are releasing from.
- Project has a Jenkins job. (This is not truly a requirement to perform a release; however, the process described in this document requires it. If you need to release a project which does not have a Jenkins job, ask a member of the CM team for assistance).

Validating Before a Release

Each time before a release, make sure the following criteria are met:

- Project does not include any external dependencies on SNAPSHOT versions.
  Note: dependencies on other modules in the project being released are okay.
- Make sure you are using Maven 3 or above.
  Note: You can check the validity of a project before attempting a release by running 'clean verify' on the entire project. This is actually done as part of the release, but can be run separately if you want to double-check things before attempting a full release on Jenkins.
- Project is currently able to build on Jenkins successfully

Steps to Release

1. Determine your release version and your next development version.
   This is usually easy to do. Your release version is typically the current version of your project minus the "-SNAPSHOT", and your next development version is your next project release version plus "-SNAPSHOT". In other words, if your current version is 1.0.0-SNAPSHOT then your release version will be '1.0.0' and you next development will probably be 1.0.1-SNAPSHOT or 1.1.0-SNAPSHOT depending on what kinds of changes are planned for the next development cycle.
2. Update the Release notes (changes.xml)
   - To Update the release notes:
     - Open up a browser and navigate to the project in Jira.
     - Click on the "Release Notes" link.
     - Select the version that you want to release.
     - Select 'Changes-XML' as the style.
     - Click "Create". Jira will generate a 'release' element populated with Jira tasks which were attached to the project version.
     - Copy the release element into the body of the changes.xml document (located under the src/changes directory). Note: Do not overwrite the existing release element. There will be a separate element for each version released.
     - If desired, edit the contents of the generated release notes. You may manually add or remove items and/or edit the contents.
     - If the project has two Jira projects (one for the development issues and one for bugs) these steps will need to be done for each Jira project.
     - Commit the updated changes.xml
3. Navigate to the Jenkins build for your project.
4. The commit of the changes.xml will trigger a new build.
   - It is best to allow this build to complete before continuing on. There is typically a delay between the SVN commit and the build starting. You can click the "Build Now" to bypass the start delay. #: You can also choose to cancel this build this pending build; however this is more prone to errors.
5. You are now ready to start the release build.
   1. Login to Jenkins using the 'release' username. (Only authorized people may perform releases, so talk to your project lead, coordinator, or a Development Support person if you do not know the proper password.)
   2. Navigate to the Jenkins job for your project.
   3. Click on the "Perform Maven Release" link.
   4. If not already selected, select the "Specify one version for all modules" option. (At the time of this writing, we do not have any projects which use the other options).
   5. Check the box next to "Specify custom SCM login/password"
      Note: Make sure that "ENABLE AUTO REFRESH" appears in the upper-right corner of the browser. Otherwise, click the link to DISABLE refresh. If the auto refresh is running, you will have trouble retaining the values you enter for the SCM login/password.
   1. Fill in your SCM login and password. These are your SVN username and password for the SVN repository.
   2. Click the "Schedule Maven Release Build".

What Happens During a Release

The following steps occur automatically when you run "Perform Maven Release"...

- Update from SVN.
- Update the SNAPSHOT versions to the release version.
- Runs preparation goals (clean, verify, changes:announcement-generate).
- Check in modified POMs.
- Create SVN tag of release.
- Update versions from release version to next development version.
- Check in modified POMs.
- Check out tag.
- Run 'deploy' and 'deploy-site' goals.
- Clean up after build (remove checkout out tag from local workspace, clean up release property files, etc.)
  If configured properly, an email is sent out on the project mailing list which alerts people that a release has been made.

Problems You May Encounter During a Release

Problem
: Release email goes out, but the release did not complete correctly.

Solution
: The release email is sent after the parent pom is built and release, but before the entire release is complete, so it is possible that an alert email can go out without the entire build completing successfully. In a case like this, I recommend sending out an email to the mailing list to let them know to disregard the previous email.

Problem
: Release fails with error:

```
[ERROR] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Unable to commit files
Provider message:
The svn command failed.
Command output:
svn: Commit failed (details follow):
svn: Authentication error from server: Password incorrect
```

Solution
: When starting the build, you either mistyped or forgot to enter your SCM login and/or password (or the AUTO-REFRESH was enable and your password was cleared before you clicked the button to schedule the release). Before attempting to restart the release, wipe out the Jenkins workspace first. See below for instructions on wiping out the workspace.

Problem
: Release fails with error:

```
[ERROR] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
```

```
[INFO] Unable to commit files
Provider message:
The svn command failed.
Command output:
svn: warning: Can't open file '/root/.subversion/servers': Permission denied
svn: Commit failed (details follow):
svn: Can't get username or password
```

Solution

When starting the build, you either forgot to enter your SCM login and/or password (or the AUTO-REFRESH was enable and your password was cleared before you clicked the button to schedule the release). Before attempting to restart the release, wipe out the Jenkins workspace first. See below for instructions on wiping out the workspace.

Problem

Release fails with error about non released dependencies, and the dependencies are NOT part of the project which I am attempting to release. Example of Contrib XML Parser project release error:

```
[INFO] Checking dependencies and plugins for snapshots ...
[HUDSON] Archiving /home/hudson/.hudson/jobs/contrib-xmlparser-deploy/workspace
/contrib-xmlparser/pom.xml to /home/hudson/.hudson/jobs/contrib-xmlparser-deploy
/modules/com.cdmtech.contrib$contrib-xmlparser/builds/2010-10-04_10-37-30/archive
/com.cdmtech.contrib/contrib-xmlparser/1.0.9-SNAPSHOT/pom.xml
[INFO] ------------------------------------------------------------------------
[ERROR] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Can't release project due to non released dependencies :
com.cdmtech.contrib:contrib-parent:pom:1.1.0-SNAPSHOT
com.cdmtech.core:core_version:jar:1.01.00-SNAPSHOT:compile
com.cdmtech.atlas.maven:cdm-version-plugin:maven-plugin:2.0.0-SNAPSHOT:runtime
in project 'Contrib XML Parser' (com.cdmtech.contrib:contrib-xmlparser:jar:1.0.9-
SNAPSHOT)
```

Solution

You cannot perform a release on a project which includes SNAPSHOT versions of plugins or dependencies. Update your POMs to use only release versions of dependencies, check in the changes, and make sure that the build is able to complete successfully before starting your release build.

Problem

Release fails with error about non released dependencies, but the dependencies are part of the project which I am attempting to release. Example of MARVEL project release error:

```
[INFO] ------------------------------------------------------------------------
[ERROR] BUILD FAILURE
[INFO] ------------------------------------------------------------------------
[INFO] Can't release project due to non released dependencies :
com.cdmtech.ares.marvel:marvel-common:jar:1.0.6-SNAPSHOT:compile
com.cdmtech.ares.marvel:marvel-parent:pom:1.0.6-SNAPSHOT
in project 'MARVEL Model' (com.cdmtech.ares.marvel:marvel-model:jar:1.0.6-SNAPSHOT)
```

Solution

This error occurs when using Maven 2.2.1 or ealier and the 'unpack' or 'copy' goals of the maven-

dependency-plugin. To fix, locate any executions using either of these goals and replace with 'unpack-dependencies' or 'copy-dependencies' instead. Check in the changes and make sure that the build is able to complete successfully before starting your release build.

Note: You cannot simply replace the goal used since the goals require changes to the overall configuration. See the maven-dependency-plugin documentation for more info about configuring the goals.

Problem

Release fails with error:

```
[INFO] [ERROR] maven-changes-plugin: None of the configured sortColumnNames 'null'
are correct.
[INFO] [INFO] Downloading from JIRA at: https://jira.cadrc.calpoly.edu/jira/secure
/IssueNavigator.jspa?view=rss&pid=10640&statusIds=6&resolutionIds=1&tempMax=25&
reset=true&decorator=none
[INFO] Nov 17, 2010 11:21:22 AM org.apache.commons.httpclient.HttpMethodBase
getResponseBody
[INFO] WARNING: Going to buffer response body of large or unknown size. Using
getResponseBodyAsStream instead is recommended.
[INFO] [INFO]
------------------------------------------------------------------------
[INFO] [ERROR] BUILD ERROR
[INFO] [INFO]
------------------------------------------------------------------------
[INFO] [INFO] Couldn't find the release '1.0.0' among the supplied releases.
[INFO] [INFO]
------------------------------------------------------------------------
[INFO] [INFO] For more information, run Maven with the -e switch
[INFO] [INFO]
------------------------------------------------------------------------
[INFO] [INFO] Total time: 5 seconds
[INFO] [INFO] Finished at: Wed Nov 17 11:21:22 PST 2010
[INFO] [INFO] Final Memory: 24M/231M
[INFO] [INFO]
------------------------------------------------------------------------
[HUDSON] Archiving /home/hudson/.hudson/jobs/atlas-cdm-maven-changes-plugin-templates
/workspace/cdm-maven-changes-plugin-templates/pom.xml to /home/hudson/.hudson
/jobs/atlas-cdm-maven-changes-plugin-templates/modules/com.cdmtech.atlas.maven$cdm-
maven-changes-plugin-templates/builds/2010-11-17_11-21-08/archive
/com.cdmtech.atlas.maven/cdm-maven-changes-plugin-templates/1.0.0-SNAPSHOT/pom.xml
[INFO] ------------------------------------------------------------------------
[ERROR] BUILD ERROR
[INFO] ------------------------------------------------------------------------
[INFO] Maven execution failed, exit code: '1'
```

Solution

The message "Couldn't find the release X among the supplied releases" indicates that you failed to update your changes.xml file (or it is missing). Update the changes.xml file with a 'release' element corresponding to the version you are trying to release, wipe out the Jenkins job's workspace, and restart the release build.

Problem

Release fails with error about "Return code is: 503":

```
[INFO] Uploading: https://hudson149.cdm.cdmtech.com/nexus/content/repositories
/releases/com/cdmtech/sol/sol-support/1.0.1/sol-support-1.0.1.jar
[INFO] [INFO]
------------------------------------------------------------------------
[INFO] [ERROR] BUILD ERROR
[INFO] [INFO]
------------------------------------------------------------------------
[INFO] [INFO] Error deploying artifact: Failed to transfer file:
https://hudson149.cdm.cdmtech.com/nexus/content/repositories/releases/com/cdmtech
/sol/sol-support/1.0.1/sol-support-1.0.1.jar. Return code is: 503
[INFO]
[INFO] [INFO]
------------------------------------------------------------------------
[INFO] [INFO] For more information, run Maven with the -e switch
[INFO] [INFO]
------------------------------------------------------------------------
[INFO] [INFO] Total time: 2 minutes 21 seconds
[INFO] [INFO] Finished at: Wed Oct 27 09:20:12 PDT 2010
[INFO] [INFO] Final Memory: 31M/344M
[INFO] [INFO]
------------------------------------------------------------------------
[HUDSON] Archiving /home/hudson/.hudson/jobs/mars-sol-support-deploy/workspace
/sol-support/pom.xml to /home/hudson/.hudson/jobs/mars-sol-support-deploy/modules
/com.cdmtech.sol$sol-support/builds/2010-10-27_09-16-33/archive/com.cdmtech.sol
/sol-support/1.0.1-SNAPSHOT/pom.xml
[INFO] ------------------------------------------------------------------------
[ERROR] BUILD ERROR
[INFO] ------------------------------------------------------------------------
[INFO] Maven execution failed, exit code: '1'
```

Solution

This is actually a problem with Nexus (the application hosting our internal Maven repositories). Alert someone in Configuration Management. Once the Nexus problem is fixed, you will need to wipe out your Hudson job's workspace, remove the tag from SVN, and revert the code in SVN to a revision before the release began. Someone in Configuration Management can help you do this.

Problem

I started a release build, but it just did a regular build.

Solution

If you attempt to start a release build while a regular build of the same job is in queue or already running, Jenkins will ignore your request to schedule a release. Wait until the job has finished building before scheduling your release. See below for information about starting a release without waiting for the previous build to finish.

How-tos

How to start a release without waiting for previous build to finish successfully

Often times you will try to perform a release immediately after making changes to the project. Most times, if you do not allow the previous build to run, your release will fail. It is recommended that you allow running or pending build to complete before attempting to release; however, if you are in a hurry, there are some things you can do to speed up the process. Please note,

If you are waiting on a build in the queue...

> When a build is waiting in the queue in a "quiet period" (i.e. it is observing a set delay time before executing and not simply waiting for an available executor), you can log in and hit "Build Now" to skip the waiting period.

If you are waiting on a build which is already running...

> When waiting for a build to complete because you had to make a few tweaks or last minute additions to your build, but you are **sure** that everything will build fine, you can wait for the SVN update to finish and then cancel the build. You can find out whether the SVN update is done, by watching the "Console Output." One of the first output lines should show "Updating svn://cvs/svnroot/some-project-location" followed by zero or more and lines showing the files being updated. Wait for the line "At revision XXXXX" to appear. This marks the end of the updates, and you can now safely cancel the build and follow the steps above to start your release.
>
> > Be Aware: If you don't know what you are doing, attempting this can lead for more problems, and can end up taking longer to clean up than just waiting it out in the first place, so I don't recommend this if you don't really know what you are doing. Also, if you didn't understand this step, you are too much of a newbie to attempt this, so ask Development Support for help instead.

```
; Why do you have to wait for the SVN updates to complete?
: Our Jenkins jobs are pretty much all configured to only build the parts of a maven project which were char
```

How To Wipe Out the Jenkins Workspace

It is sometimes necessary to wipe out a Jenkins workspace in order to clean out local changes and clear the way for a clean build. To wipe out a workspace:

1. Click on the Jenkins job.
2. Click on the "Workspace" link.
3. Click on the "Wipe Out Workspace" link.

Retrieved from "https://pitawiki/wiki/index.php/Development_Support:How_To_Perform_a_Release"
Categories: Development Support | How-To Guides

- This page was last modified on 8 July 2013, at 22:14.