

# Release procedure

Releases are now performed using Maven 2 or Maven 3 so you need to have one of these installed (see the [building](#) page). This page describes how the release process is handled.

## Organization

CARGO is made of four main modules (i.e., trunks):

- **pom**: That is the parent of all CARGO modules.
- **resources**: Contains resources, most notably the CARGO ping component as well as the test WARs.
- **core**: CARGO Java implementation.
- **extensions**: CARGO extensions, i.e. the ANT tasks and the Maven2/Maven3 plugin.

Of these modules/trunks, the **pom** is released separately from the other three. However, **resources**, **core**, and **extensions** are always released together.

CARGO uses the Codehaus Nexus instances for release, which means that we release in two phases:

1. The maven-release-plugin will actually release to a staging area, i.e. the generated artifacts are only available on the staging repository
2. Once the staged version is OK (to confirm by opening a vote on the CARGO developers list, for example), it can be promoted (therefore, deployed to the Codehaus Maven repository and also synced to Maven Central)

As a consequence of this, the release process is as follows:

1. If the parent (**pom**) has changed, release the parent first without using the "staging" phase. To do so, log in to the Codehaus Nexus interface directly (without waiting for a vote) and release.
2. For **resources**, **core** and **extensions**:
  - Release all three
  - This will indeed create a big "staging" repository
  - Open the vote so the whole team can decide whether the artifacts are OK
  - Once the vote is finished, promote the release (or drop if not OK)

## Prerequisites

In order to release, you will need:

- An access to the Codehaus Nexus instance as a manager for the CARGO project. Read more about it on [Codehaus Maven Repository Usage Guide](#)
- [The GNU privacy guard \(GPG\)](#) installed on your machine and at least one secret key.
  - The secret key's public part must have been uploaded to the [MIT PGP Public Key Server](#).
- To provide your Codehaus Xircles credentials using your `settings.xml` file (in the `$(user.home)/.m2/` directory). Here's an example `settings.xml`:

```

<settings>
  [...]
  <servers>
    <server>
      <id>codehaus-nexus-snapshots</id>
      <username>USERNAME</username>
      <password>PASSWORD</password>
    </server>
    <server>
      <id>codehaus-nexus-staging</id>
      <username>USERNAME</username>
      <password>PASSWORD</password>
    </server>
    <server>
      <id>cargo-website</id>
      <username>USERNAME</username>
      <password>PASSWORD</password>
      <filePermissions>664</filePermissions>
      <directoryPermissions>775</directoryPermissions>
    </server>
  </servers>
  [...]
</settings>

```

## Before you start

Before releasing, make sure what is going to be released is fully functional; in particular check that all containers pass all tests.

- The [Continous Integration for CARGO](#) actually passes tests for all downloadable containers. Green ones are doing well 😊
- For non-publicly-downloadable containers ([Glassfish 2.x](#), [JRun 4.x](#) as well as all versions of Orion/OC4J, WebLogic and WebSphere) tests need to be done on your computer.

## Tagging and deploying on to the staging area

1. Start by erasing your local Maven repository (`${user.home}/.m2/repository/`, unless you've reconfigured this in your `settings.xml`). This will ensure that you don't have any artifacts cached locally that can't be found in public repositories.
2. Check if there have been SVN commits on that artifact since the latest stable release. If there have been, you'll need to release.
  - The parent (**pom**) typically don't change between versions.
3. First release the parent if required. Promote the staged artifact and wait for it to be synced to Maven Central (typically takes an hour or two).
4. Then release each of the code trunks **resources**, **core** and **extensions**.
5. In each code trunk's topmost pom.xml, change the links to the other trunks' versions to `${project.version}` (the maven-release-plugin updates this to the actual version during release).

The typical Maven release procedure is to execute these commands for each trunk (i.e. repeat this for each trunk):

1. A dry-run before doing the actual release to ensure there aren't any problems:

```
mvn clean release:prepare -DdryRun=true
```

2. `mvn release:prepare`: the actual release preparation, with the actual tagging process
  - If the Subversion commit fails, try executing the same command but add `-Dusername=your_username -Dpassword=your_password` (of course, replace with your own user name and password on the CARGO SCM)
3. `mvn release:perform`: deploys the tagged artifacts on the Codehaus Nexus staging repository
  - If you're running on Maven 2.2.0 (note: that's the version that ships with some flavors of MacOS X), you might run into SHA1 and MD5 signature verification issues. Please upgrade to Maven 2.2.1.
  - If you get a message saying: `Error deploying artifact: Failed to transfer file. Return code is: 401 Unauthorized, please make sure your credentials in the settings.xml file are correct.`

 Release may take up to 6 hours. Make sure you have time before you start 😊

 Sometimes, you might get a message saying `Error deploying artifact: Failed to create destination WebDAV collection (directory): unable to find valid certification path to requested target`, that's because the Codehaus HTTPS WebDAV servers' certificate is not installed on your computer. In this case:

1. Download and compile the program from [InstallCert.java](#)
2. Execute the program `java InstallCert dav.codehaus.org` and make it install one of the certificates. If you run the program again, it should be happy and tell you that all certificates are installed.
3. Add the generated `jssecacerts` file to your `$JAVA_HOME/jre/lib/security` directory.

## Promoting the staged version

Once the new CARGO version is in the Codehaus Nexus staging repository, send an e-mail to the CARGO list so users can try out the new version. We typically leave the vote open for 72 hours.

Once the developers/users also validate the version as being stable, we need to do the actual release:

1. Follow the guide on [closing a staging repository on Codehaus Nexus](#) and [releasing a closed staging repository on Codehaus Nexus](#) in order to release the CARGO artifacts to Maven Central.
  - Tip: If the staged version was not good, read the instructions for [dropping a staging repository on Codehaus Nexus](#).
  - Important: The artifacts cannot be taken back once they've reached Maven Central.
2. Log onto Cargo JIRA, [release the current version and add the next version](#).
3. Check that the Cargo wiki is up to date. Specifically, perform the following updates:
  - a. Modify the status on the home page about the delivery:
    - Make sure the [Containers list](#) is complete
    - Make sure the [Navigation page](#) is complete
    - Make sure the [Javadocs for the Core Containers](#) are complete
    - Make sure the documentation for each container is up to date.
      - The source files (that you need to import using the **Insert** -> **Wiki markup** option in each container's page manually) are generated in `core/documentation/target/[container's name].log`.
      - If the web page in <http://cargo.codehaus.org/> is not updated directly, add the `?nc=1` option to the end of the URL; for example <http://cargo.codehaus.org/Jetty+7.x?nc=1>
    - Make sure the [Containers with DataSource and Resource support](#) is up to date.
      - The source file (that you need to import using the **Insert** -> **Wiki markup** option

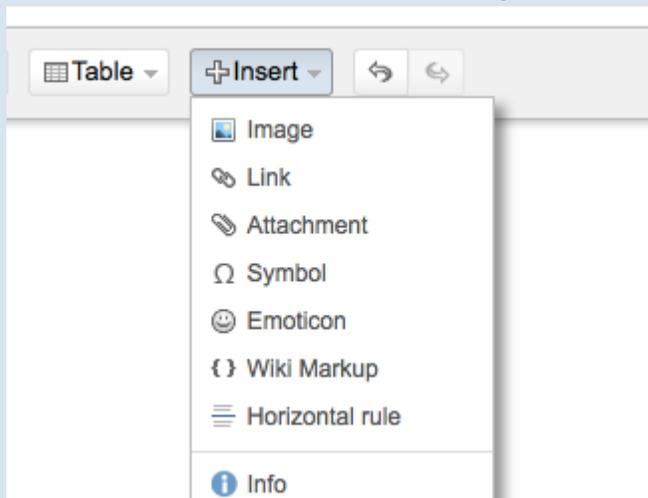
manually) is generated in `core/documentation/target/datasource.log`.

- If the web page in <http://cargo.codehaus.org/> is not updated directly, add the `?nc=1` option to the end of the URL; i.e. <http://cargo.codehaus.org/DataSource+and+Resource+Support>
  - Make sure the [Project Structure](#) is up to date.
    - The source file (that you need to import using the **Insert -> Wiki markup** option manually) is generated in `core/documentation/target/project-structure.log`.
    - If the web page in <http://cargo.codehaus.org/> is not updated directly, add the `?nc=1` option to the end of the URL; i.e. <http://cargo.codehaus.org/Project+Structure?nc=1>
- Modify the [Home](#) page to update the available version number and release date
  - Modify the [Maven2 Plugin Installation](#) page to have the latest SNAPSHOT version in there
  - Create a [blog post](#)
  - [Export the wiki to a zipped HTML file](#) (select all pages except for the *Downloads* (including all subsections) and *News* pages) and add it the [Downloads](#) page
  - Move the old version to the [Archived Downloads](#) page
    - Remember to move the old documentation archive from the [Downloads](#) page to that page as well
  - Modify the [Downloads](#) page to update the download links, available version number, documentation and release note links
- Send an announcement email to [Cargo mailing lists](#) ... and to other relevant sites you know about. Mailing lists of some servers can also be interesting.

### **i** How to insert using the Wiki Markup

To insert content using **Wiki Markup**:

1. Edit the document in the Codehaus Confluence
2. Clear the existing content by selecting all content and pressing delete on your keyboard
3. On the toolbar, click **Insert -> Wiki markup**:



4. Paste the wiki markup content