

# Packaging and deploying Mauve releases

The Mauve source tree includes scripts to support packing Mauve releases on Windows, Linux, and Mac OS X. In general, a computer running each OS is required to package the corresponding release. It is not possible, for example, to cross-package a Mac OS X release from within Linux.

## Packaging Linux releases

1. Download the Mauve GUI source code from subversion or the Mauve web site
2. Ensure that the correct pre-compiled aligner binaries exist in the linux-x86 folder
3. Set the release version in the top-level build.xml file
4. Execute the command ``ant dist`` from the top-level mauve source directory
5. A release file called `mauve_linux_${version}.tar.gz` will be created in the dist directory

## Packaging Mac OS X releases

1. Download the Mauve GUI source code from subversion or the Mauve web site
2. Ensure that the correct pre-compiled aligner binaries exist in the osx folder
3. Set the release version in the top-level build.xml file
4. Execute the command ``ant macdist`` from the top-level mauve source directory
5. A release file called `Mauve-${version}.dmg` will be created in the dist directory

## Packaging Windows releases

Windows release packaging requires the freely available [Nullsoft Scriptable Installation System](#) . Please download and install this software prior to packaging Mauve installers for Windows.

1. Download the Mauve GUI source code from subversion or the Mauve web site
2. Ensure that the correct pre-compiled aligner binaries exist in the win32 folder
3. Set the release version in the top-level build.xml file
4. Download the latest win32 Java online installer from java.com to the top-level mauve directory. Update mauve.nsi to contain the path to latest installer (currently `jre-1_5_0_09-windows-i586-p-iftw.exe`)
- 5a. From within Eclipse, run the 'nsicompile' target of build.xml

OR

- 5b. From the windows command-line, execute ``ant nsicompile``
6. A release file called `mauve_installer_${version}.dmg` will be created in the dist directory

## Deploying releases

Additional ant targets exist in the build.xml file called `deployLinux`, `deployWin32`, and `deployMacOSX`. These targets assume that your workstation is directly connected to the Genome Evolution Lab network and that ssh private keys have been installed to permit password-free authentication. Unless you are me or you hack our network, you will not be able to automatically deploy releases.

## Deploying ASAP-integrated Mauve releases

Mauve includes an applet that allows the Mauve Java Web Start application to communicate directly with the ASAP annotation database. To build an ASAP-integrated Mauve release, it will be necessary to either create or obtain Java code signing certificates. Once a certificate has been obtained, the `key.keystore` variable in `build.xml` should be modified to reflect the location of the keystore file. The ASAP deployment code in `build.xml` assumes that the deployment takes place from a machine running Windows, deployed to a web server running Linux. The variables `asap.dir` and `asap.codebase` in `build.xml` must be configured appropriately. `asap.dir` is the location of the web server directory where the production copy of the Mauve applet and application reside. It must be accessible as a UNC path within windows. This directory can be a symlink on the web server from the user's home directory to the production directory. `asap.codebase` is the base URL where the newly installed Mauve application and applet will reside. Once these variables have been configured correctly and the keystore is in place, the `deployASAP` target of `build.xml` can be executed. Note that the password for the keystore must be provided when the target is executed. This can be done either on the `java -jar ant.jar` command line, or from within eclipse by providing a new VM Argument in the JRE tab of the "Run as...->Ant build..." dialog box. The vm argument to add is `-Dkey.password=<your_keystore_password>`