

Developing the Mauve GUI in Eclipse

The Mauve graphical user interface (GUI) is a Java program that provides a front-end for constructing alignments with the command-line mauveAligner program in addition to offering interactive browsing of the alignment results. Mauve currently requires Java versions 1.4 or later.

The Mauve GUI can be developed in nearly any Java development environment, although we use and recommend [Eclipse](#) version 3 or later. Mauve source code is stored in a subversion repository, which can be accessed via authenticated https using any subversion client, but especially the [Subclipse plug-in](#) from within Eclipse. The brave developer may elect to use a text editor and the included build.xml script for [Apache Ant](#).

What follows is a step-by-step recipe for configuring the standard Mauve build environment:

1. Download and run eclipse

Download from the [eclipse web site](#). Eclipse does not come with a windows installer, so simply run the executable after unpacking it from the compressed archive.

2. Install the subclipse plug-in

Tigris.org maintains [detailed instructions](#) on installing subclipse.

3. Check out the Mauve source code

From within eclipse, go to the "Window->Open perspective->Other..." menu item. A dialog box will appear asking for a perspective selection. Select the "SVN Repository Exploring" perspective. The SVN repository perspective will load. Click the "Add SVN repository" button that appears (it has tiny lettering that says SVN and a + symbol). The svn URL for mauve is <https://mauve.svn.sourceforge.net/svnroot/mauve/mauve>. Upon entering this repository URL, a dialog box appears cautioning that a hostname mismatch exists between the certificate and the server. This is normal and the certificate should be permanently accepted. Assuming all has gone well, the mauve repository should now be registered with Subclipse. At this point the development trunk of mauve can be checked out by expanding the repository (click on the plus), and then right-clicking on "trunk" and checking it out into the workspace. Use the "Check out as project into..." menu item to select the destination directory for the source tree.

4. Build Mauve

Switch back to the Java perspective by again going to the "Window->Open perspective->Other..." menu item and selecting "Java (default)" in the dialog box that appears. If a blue Eclipse welcome screen appears, close it to reveal the Mauve source tree. The code should have already compiled and any warnings or errors will appear in the Problems tab near the bottom of the Eclipse workspace.

5. Install platform-specific aligner binaries for debugging

The Mauve GUI depends on a set of C++ programs that implement the alignment algorithms. These platform specific binaries are mauveAligner, progressiveMauve, and muscle_aed. Copies of these binaries compiled for each of Windows, Linux, and Mac OS X reside in the win32, linux-x86, and osx directories of the Mauve GUI source tree. When launched from within eclipse, Mauve needs a copy of the binaries for the appropriate platform in the top-level repository directory. To install the binaries, copy the appropriate set for your platform to the top level directory (one level up).

6. Run Mauve in the debugger

A debug profile must be created in order to run Mauve in the debugger. Select the "Run->Debug..." menu item and create a new Java Application debug configuration. Name it something like "Mauve debug" and set the main class to org.gel.mauve.gui.Mauve. In the arguments tab, set the Java heap size to something larger than the default. For example, the command -Xmx512m would request 512mb of heap space. Mauve is very memory hungry and 384Mb should probably be considered an absolute minimum

heap allocation. 1024Mb is preferred. Optionally, an alignment or a url of an alignment may be specified as a command-line argument. If specified, the alignment will be loaded upon program launch.