

CLIFv2 installation guide



<http://clif.ow2.org/>

Table of contents

1. How to get CLIF working?	3
1.1. Technical requirements	3
1.2. Ready-to-use distributions	3
1.3. Checking Clif version and execution environment	4
1.4. Upgrading from CLIF v1.x.x	5
2. Installing a CLIF server or a CLIF Swing console	6
2.1. Requirements	6
2.2. Configuration	6
2.3. Execution	6
3. Installation of Eclipse-based console	7
3.1. Introduction	7
3.2. Install the Eclipse-RCP based standalone CLIF distribution	7
3.3. Install the Eclipse plug-ins	7
3.3.1. Dependencies	7
3.3.2. Installation	8
3.3.3. Execution	8
4. Mandatory requirements	9
4.1. Download requirements	9
4.1.1. Sun/Oracle Java SE 6	9
4.1.2. Apache ant utility version 1.7.1 or greater	9
4.2. Environment variables setting	9
4.2.1. For Windows	9
4.2.2. For Linux	12
4.2.3. For Mac OS X	13

1. How to get CLIF working?

1.1. Technical requirements

CLIF framework and provided load injectors are 100% Java™. CLIF requires a Java development kit (JDK), and the Java-based ant utility from Apache.org. As of CLIF version 2.0.5, requirements are:

- Java SE 6 runtime, such as Sun/Oracle or Open JDK 6.0
Download Sun/Oracle from <http://www.oracle.com/technetwork/java/javase/downloads>
Note: while ready-to-use CLIF distributions require a Java SE 6 runtime, CLIF's source code is still Java 1.5 compatible. So, it is possible to recompile CLIF's source code with a Java 1.5 compiler to get a CLIF distribution running on 1.5 Java runtimes.
- Apache ant utility version 1.8.0 or greater
download from <http://ant.apache.org/bindownload.cgi>
Note: make sure ant is using the right JDK! (see, for example, environment variable JAVA_HOME)
- Any Linux distribution based on 2.6 kernels
- Apple MacOS.X™ tiger or later, PPC or Intel
- Microsoft Windows XP™ or later

System probes for Linux are also 100% Java, while system probes for Windows and MacOS.X are native (C-code embedded in Java code via the Java Native Interface).

1.2. Ready-to-use distributions

There are two ways of getting a CLIF runtime environment: either by getting and compiling the whole source code from the Subversion repository at OW2.org's Forge, or by getting a ready-to-use binary distribution. For the former method, please refer to the developer's guide. For the latter method, CLIF's site at OW2.org Forge offers several binary distributions, available as zip files (see http://forge.ow2.org/project/showfiles.php?group_id=57):

- `clif-2.0.5-swingGui.zip.zip`
"CLIF Swing console": simplified graphical console, system-independent (pure Java), supporting test plan definition, execution and monitoring, as well as final reporting. Does not support editing ISAC scenarios. Note: it also includes the CLIF server features.
- `clif-2.0.5-server.zip`
"CLIF server": provides the minimal runtime environment to be installed on a remote host for deploying probes and injectors. It is system-independent.
- `clif-2.0.5-eclipseconsole-windows-x86.zip`
"CLIF Eclipse console": full-featured, Eclipse RCP based standalone console for Windows/Intel 32 bits
- `clif-2.0.5-eclipseconsole-linux-i386.zip`
"CLIF Eclipse console": full-featured, Eclipse RCP based standalone console for Linux/Intel 32 bits
- `clif-2.0.5-eclipseconsole-linux-amd64.zip`
"CLIF Eclipse console": full-featured, Eclipse RCP based standalone console for Linux/Intel 64 bits
- `clif-2.0.0-eclipseconsole-macosx.zip`
"CLIF Eclipse console": full-featured, Eclipse RCP based standalone console for Mac OS X/Intel

CLIF installation guide

- `clif-2.0.5-eclipseplugins-linux.zip`
CLIF plugins distribution for Eclipse Linux. It allows to use the full-featured CLIF console from your Eclipse environment.
- `clif-2.0.5-eclipseplugins-windows-x86.zip`
CLIF plugins distribution for Eclipse/Windows. It allows to use the full-featured CLIF console from your Eclipse environment.
- `isac-DnsInjector.zip`
Support for DNS traffic injection within ISAC scenarios.
- `isac-FtpInjector.zip`
Support for FTP traffic injection within ISAC scenarios.
- `isac-HttpInjector.zip`
Support for HTTP(S) traffic injection within ISAC scenarios.
- `isac-ImapInjector.zip`
Support for IMAP traffic injection on incoming e-mail servers, within ISAC scenarios.
- `isac-JdbcInjector.zip`
Support for traffic injection on databases using a JDBC driver, within ISAC scenarios.
- `isac-JmsInjector.zip`
Support for JMS traffic injection within ISAC scenarios.
- `isac-LdapInjector.zip`
Support for LDAP traffic injection within ISAC scenarios.
- `isac-SipInjector.zip`
Support for SIP traffic injection within ISAC scenarios.
- `isac-SocketInjector.zip`
Support for TCP traffic injection within ISAC scenarios.
- `isac-UdpInjector.zip`
Support for UDP traffic injection within ISAC scenarios.
- `isac-RtpInjector.zip`
Support for RTP traffic injection within ISAC scenarios.

You may unzip these files wherever you like, except Eclipse plug-ins that you may install in your Eclipse environment (see section 3.3) and `isac-xxxInjector` that you may install in your Eclipse RCP environment or Eclipse RCP standalone console.

1.3. Checking CLIF version and execution environment

From a CLIF Swing console or a CLIF server environment, use command “`ant version`” to get the version numbers of Java environment, operating system and CLIF. Caution: command “`ant -version`” gives the ant version.

From the Eclipse-based graphical CLIF console, open the information pop-up with option “CLIF>About CLIF...”.

1.4. Upgrading from CLIF v1.x.x

As of version 2.0 of CLIF, some changes have occurred that introduces incompatibility issues with previous 1.x.x versions. The main change is about renaming all reference to `org.objectweb` to `org.ow2`.

These incompatibilities affect:

- CLIF test plan files (`.ctp`) with regard to the probes' fully-qualified class names
- ISAC scenarios, with regard to
 - the renaming of some ISAC plug-ins for the sake of homogeneity
 - the change of XML DOCTYPE declaration
- ISAC plug-ins, with regard to

- Java source files, because of the renaming of package `org.objectweb` into `org.ow2`
- XML descriptors, because of a change of XML DOCTYPE declaration
- measures collected into the `report` directory, with regard to
 - the copied test plans (see above)
 - the events `.classname` files
- BIRT reports.

To easily cope with this, a translation tool is provided via the `ant` utility and the `build.xml` file provided in the `dist` module (see section Developer Manual):

- `ant -f /path/to/dist/build.xml 2clif2`
First, this command recursively copies the content of current directory to a new directory named `2clif2-output`. Then, it recursively looks for `.java`, `.xis`, `.ctp`, `.xml`, `.classname` and `.rptdesign` files in this copy directory, and proceeds with all necessary translations.

2. Installing a CLIF server or a CLIF Swing console

2.1. Requirements

To install a CLIF server or the Swing console, just unzip the corresponding zip file. Execution of all utilities (console, registry, analyzer, server and all batch commands) are supported through the Apache ant utility. So, ant must be installed too.

The Swing console must be installed on the host you will be using to define and run your test plans. The server environment must be installed on all the hosts on which you'll deploy injectors and probes, as defined in your test plans.

2.2. Configuration

The Swing console just runs without any configuration. Refer to the User Manual for advanced set-up. As far as the CLIF server is concerned, configuration is necessary anytime you run it on a different host than the one where the CLIF registry is running. A utility helps setting the basic parameters, invoked by command: `ant config`. Refer to the User Manual for details.

2.3. Execution

To run the Swing console: `ant console`

A CLIF registry must be running before running a CLIF server. The easiest way is just to run a CLIF console first, since it automatically starts a CLIF registry if none is found running.

To run a CLIF server:

- `ant server` (name in the registry defaults to current host name), or
- `ant server -Dserver.name=server1` (named server1 in the registry).

Refer to the CLIF User Manual for more information.

3. Installation of Eclipse-based console

3.1. Introduction

All parts of the Eclipse-based console are available as separate Eclipse plug-ins, or delivered as a single ready-to-use standalone program. Note that CLIF's runtime environment directory, as often referred to in this documentation, is located in the console plug-in path, i.e. something like `plugins/org.ow2.clif.console.plugin_x.x.x/`

Please refer to the on-line help embedded in the console for detailed documentation about these parts.

As an Eclipse application, a number a useful common options may be set on the command line, such as:

- `-consolelog` to see messages printed out to your terminal;
- `-vm /path/to/the/jvm` to set the right Java Virtual Machine to be used;
- `-data /path/to/my/workspace` to use a different workspace directory from the default one;
- `-vmargs ...JVM options...` to pass arguments to the JVM.

3.2. Install the Eclipse-RCP based standalone CLIF distribution

Eclipse RCP based binary distributions of CLIF are standalone executables that include Eclipse 3.3 (aka Europa) RCP environment for a specific operating system. All they require is a JDK.

You just have to get the right binary distribution for your operating system, or generate it from the source (see the Developer Manual) and unzip it wherever you want on your computer.

Finally, run the `clif-console` program with any useful argument (as detailed above).

The set of available features is minimal, and, starting from CLIF version 2.0.5, the recommended way to practically use CLIF, is to install CLIF plug-ins in a common Eclipse IDE distribution (see the next section) to get a full-featured environment.

3.3. Install the Eclipse plug-ins

CLIF provides 4 Eclipse plug-ins, namely the `clif.console` plug-in, the `clif.isac` plug-in and two `clif.oda` plug-ins. The `clif.isac` Eclipse plug-in contains the Isac scenario editor and the Isac plug-in creation wizard. The `clif.oda` plug-ins provide BIRT reporting tools with access to measures generated by CLIF. The `clif.console` Eclipse plug-in contains all the remaining parts of CLIF (the CLIF core, the test plan editor, the supervision console, the custom reporting tool).

3.3.1. Dependencies

CLIF Eclipse plug-ins depends on a number of Eclipse plug-ins that may not be present by default in your Eclipse installation.

- mandatory plug-ins: `org.ow2.clif.console.plugin_2.0.x` and `org.ow2.clif.isac_2.0.x`.
- optional plug-ins: `org.ow2.clif.analyze.oda_2.0.x` and `org.ow2.clif.analyze.oda_2.0.x` to enable reporting with Eclipse BIRT.

3.3.2. Installation

As of version 2.0.5, CLIF plug-ins rely on Eclipse 3.5 (also known as Galileo). Starting from your Eclipse IDE for Java, you may just unzip the binary package `clif-2.0.5-eclipseplugins-<OS>.zip` in the `dropins` directory of your Eclipse installation to make CLIF work. This package contains CLIF's plug-ins and the necessary dependencies. It is recommended to start from the “Eclipse RCP/Plug-in Developers” packages available from:

<http://www.eclipse.org/downloads/packages/release/galileo/sr2>.

If you are interested in using Eclipse BIRT, start from the “Eclipse IDE for Java and Report Developers” package.

3.3.3. Execution

Remember to run Eclipse with the right JVM, as mentioned in section 1.1. Use option `-vm` to make sure Eclipse uses the right JVM. See section 3.1 for the most important options you may set when launching your Eclipse workbench.

If you can't see the CLIF menu, the CLIF perspective and the CLIF wizards, it means at least one dependency is missing. In this case, refer to the next section to check what is missing in your Eclipse set-up.

3.3.4. Troubleshooting

The two mandatory CLIF plug-ins (`clif-console` and `clif-isac`) require the following bundles:

- `org.apache.log4j,`
- `org.apache.lucene,`
- `org.eclipse.core.resources,`
- `org.eclipse.core.runtime,`
- `org.eclipse.help.appserver,`
- `org.eclipse.help.base,`
- `org.eclipse.help.ui,`
- `org.eclipse.help.webapp,`
- `org.eclipse.jdt.core,`
- `org.eclipse.jdt.launching,`
- `org.eclipse.jdt.ui,`
- `org.eclipse.jface,`
- `org.eclipse.jface.text,`
- `org.eclipse.ui,`
- `org.eclipse.ui.console,`
- `org.eclipse.ui.forms,`
- `org.eclipse.ui.ide,`
- `org.eclipse.ui.cheatsheets,`
- `org.eclipse.wst.sse.core,`
- `org.eclipse.wst.xml.core,`
- `org.eclipse.wst.sse.ui,`
- `org.eclipse.wst.xml.ui`

4. Requirements

4.1. Download requirements

4.1.1. Sun/Oracle Java SE 6

Click on the following link:

<http://www.oracle.com/technetwork/java/javase/downloads/jdk6-jsp-136632.html>

Click on the download button "Download Java SE Development Kit 6uNN and follow the instructions. Once the download is done, you can proceed with JDK installation.

4.1.2. Apache ant utility version 1.8.0 or greater

Click on the following link: <http://ant.apache.org/bindownload.cgi> and follow the instructions. Once the download is done, just unzip the file.

4.2. Environment variables setting

4.2.1. For Windows

Now you have to set the following environment variable:

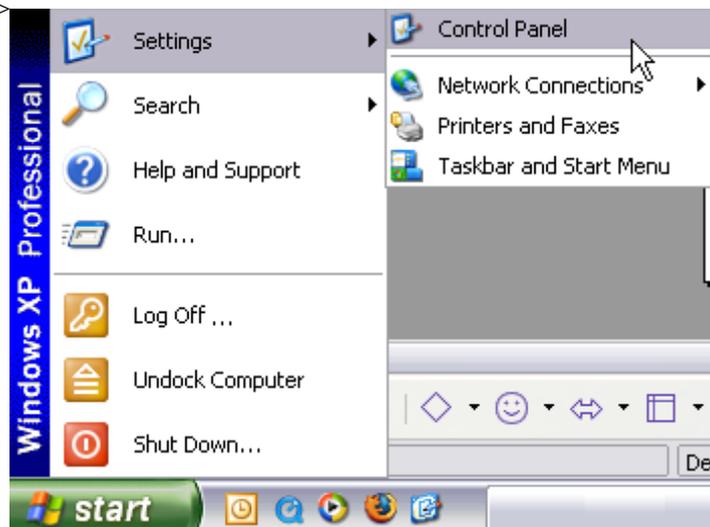
- JAVA_HOME=C:\Program Files\Java\jdk1.6.0_nn
- ANT_HOME=C:\Program Files\apache-ant-1.n.n

And to modify:

- PATH=<value already in path>;%JAVA_HOME%\bin;%ANT_HOME%\bin

Go to:

Start > Settings >
Control Panel

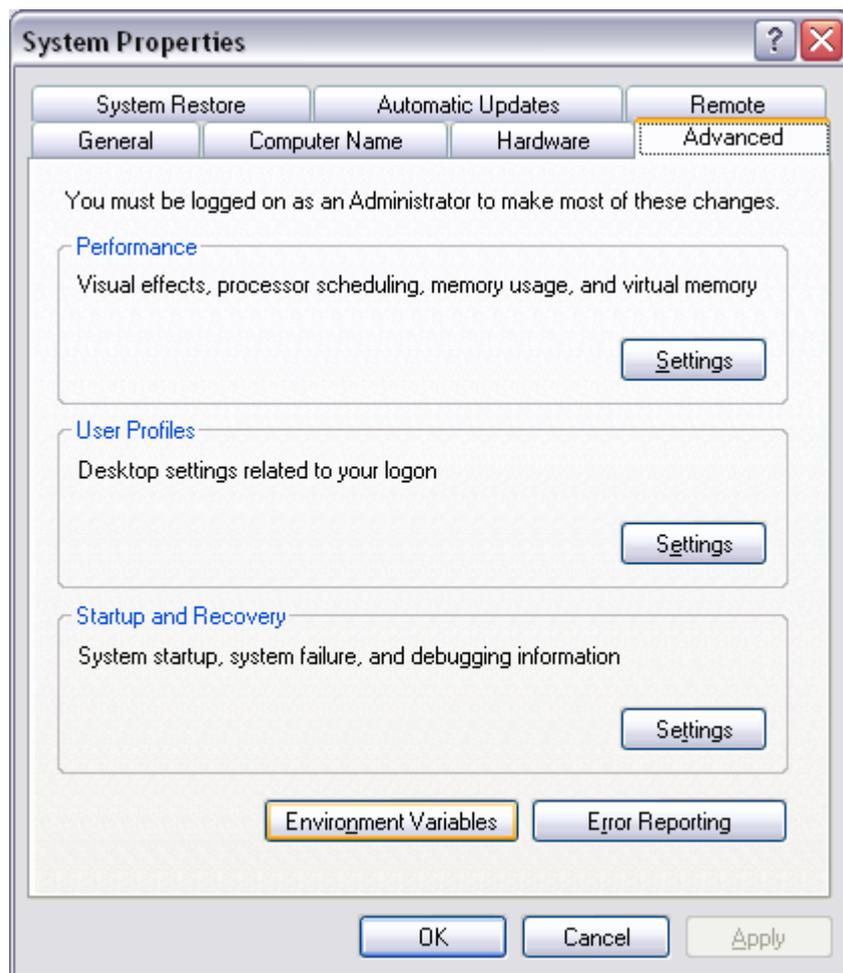


Double click on the system icon:

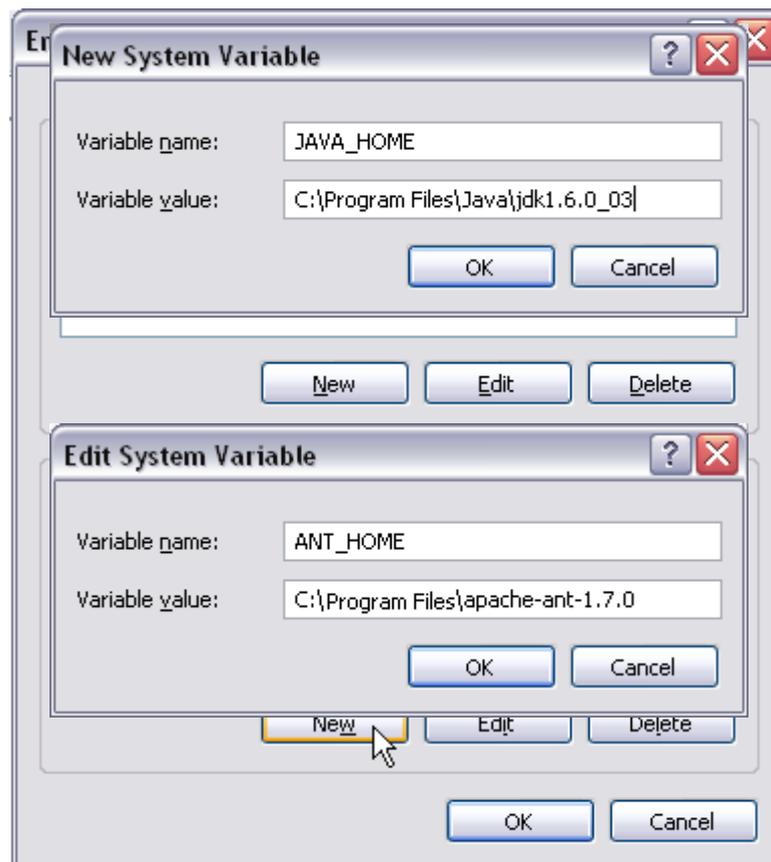


Choose advanced tab and click on the environment Variables button:

CLIF installation guide

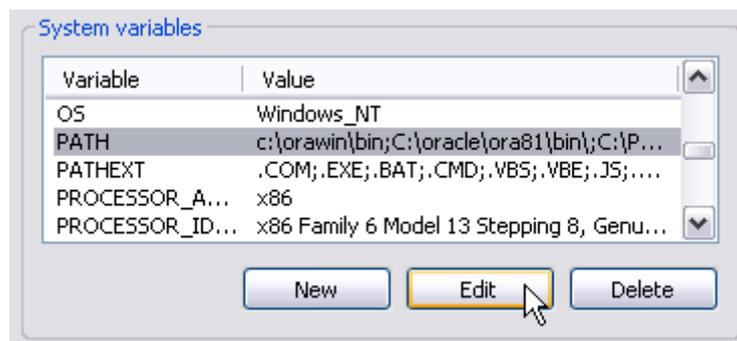


Now click on the New button of the System variables parameters group:
Then enter the variable name and its value:



Do the same thing for ANT_HOME variable:

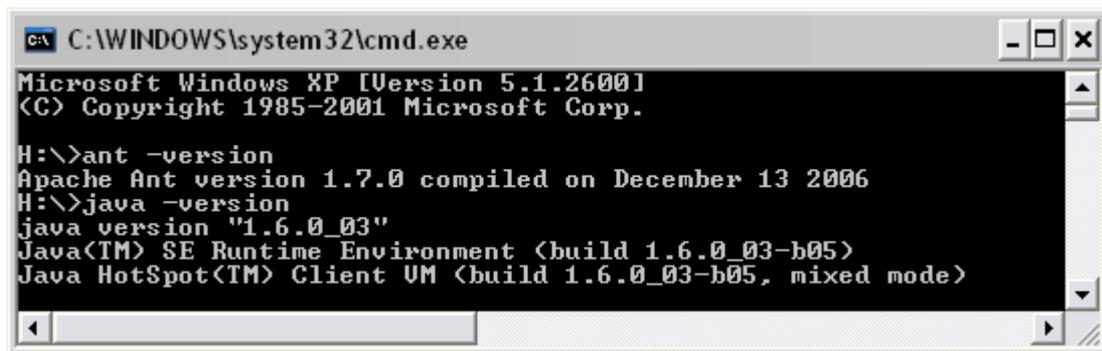
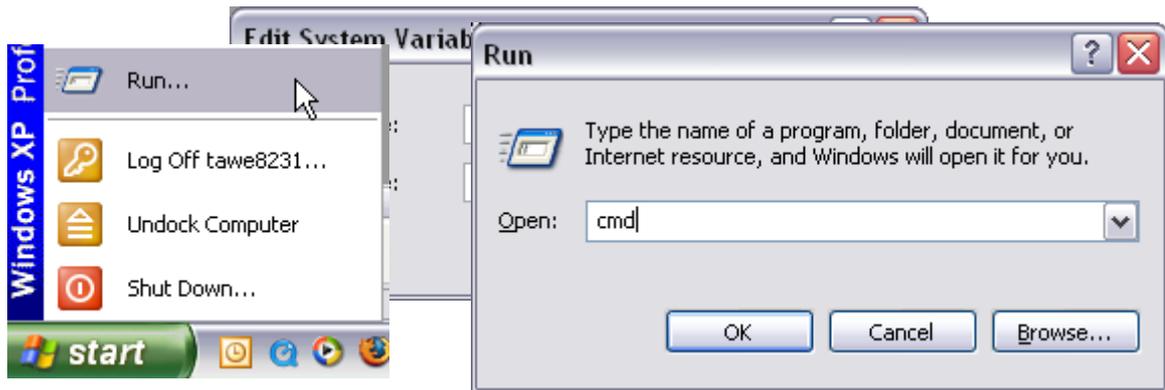
Now modify the PATH variable. Select the PATH variable and click on the Edit button:



Now add the reference to the ANT_HOME\bin and JAVA_HOME\bin repertory at the end of the PATH line:

Now you just have to check if the good java version and ant version are used by your system.

CLIF installation guide



4.2.2. For Linux

Now you have to set the following environment variable:

- JAVA_HOME=/usr/lib/jdk1.6.0_*nn*
- ANT_HOME=/usr/lib/apache-ant-1.*n.n*

And to modify:

- PATH=\$PATH:\$HOME/bin:\$ANT_HOME/bin:\$JAVA_HOME/bin

Go to the root directory of your user account.

Modify the .bash_profile file with the following command line: vi .bash_profile

If this file doesn't exist you can create it.

Put in it the following line:

```
# User specific environment and startup programs
ANT_HOME=/usr/lib/apache-ant-1.n.n
JAVA_HOME=/usr/lib/jdk1.6.0_nn
PATH=$PATH:$ANT_HOME/bin:$JAVA_HOME/bin
export ANT_HOME
export JAVA_HOME
```

Now you have to log off from your platform and then log in to reload the .bash_profile file.

Now you just have to check if the right Java version and ant version are used by your system.

```
[clif@ ~]$ ant -version
Apache Ant version 1.7.0 compiled on December 13 2006
[clif@ ~]$ java -version
java version "1.6.0_02"
Java(TM) SE Runtime Environment (build 1.6.0_02-b05)
Java HotSpot(TM) Server VM (build 1.6.0_02-b05, mixed mode)
[clif@ ~]$ █
```

4.2.3. For Mac OS X

Now you have to set the following environment variable:

- JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Home
- ANT_HOME=/usr/share/ant

And to modify:

- PATH=\$PATH:/usr/bin

Go to the root directory of your user account.

Modify the .bash_profile file with the following command line: vi .bash_profile

If this file doesn't exist you can create it.

Put in it the following line:

```
# User specific environment and startup programs
ANT_HOME=/usr/share/ant
JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Home
PATH=$PATH:/usr/bin
export ANT_HOME
export JAVA_HOME
```

Now you have to log off from your platform and then log in to reload the .bash_profile file.

Now you just have to check if the right Java version and ant version are used by your system.

```
[ host :~]    clif% ant -version
Apache Ant version 1.8.1 compiled on September 21 2010
[ host :~]    clif% java -version
java version "1.6.0_22"
Java(TM) SE Runtime Environment (build 1.6.0_22-b04-307-10M3261)
Java HotSpot(TM) 64-Bit Server VM (build 17.1-b03-307, mixed mode)
[ host :~]    clif% █
```