21st January 2016

CLIF v2.3 Installation Manual

http://clif.ow2.org/

Table of contents

1 How to get CLIF working?

1.1 Technical requirements

The CLIF framework and provided load injectors are 100% Java™. CLIF requires a Java Runtime Environment (JRE) or a Java Development Kit (JDK), version 6 or greater. OpenJDK or Oracle (http://www.oracle.com/technetwork/java/javase/downloads) Java environments are supported.

Optionally, CLIF comes with a command line interface based on the Java-based Apache ant utility, version 1.8.0 or greater (http://ant.apache.org/bindownload.cgi). Note that this ant-based interface is now deprecated. It is recommended to use the shell script-based command line interface instead, for it provides a reliable configuration system with regard to network properties. Besides, it is simpler to use since it does not require additional dependencies. Should you go for the ant-based interface however, make sure ant is using the right JDK! Take care of the JAVA_HOME environment variable (refer to section 2.4).

The main CLIF graphical user interface (aka *console*) is based on the Eclipse environment. You may install CLIF's plug-ins in your Eclipse environment (versions from 3.3 "Europa" to 3.7 "Indigo"), or use ready-to-use Eclipse distributions including CLIF plug-ins. These distributions are available for download from the CLIF forge's download area, for the following operating systems: Linux (2.6 kernels or greater), MacOSX, Windows (XP or later), all for both 32 bits and 64 bits Intel architectures.

1.2 Ready-to-use distributions

To get a ready-to-use binary distribution, go to CLIF's web site at http://clif.ow2.org, and go to the download area: http://forge.ow2.org/project/showfiles.php?group_id=57. Here, you will find the following software packages:

- `clif-eclipse-linux-32`
  "CLIF Eclipse console": full-featured, Eclipse-based console for Linux/Intel 32 bits
- `clif-eclipse-linux-64`
  "CLIF Eclipse console": full-featured, Eclipse-based console for Linux/Intel 64 bits
- `clif-eclipse-macosx-32`
  "CLIF Eclipse console": full-featured, Eclipse-based console for MacOSX/Intel 32 bits
- `clif-eclipse-macosx-64`
  "CLIF Eclipse console": full-featured, Eclipse-based console for MacOSX/Intel 64 bits
- `clif-eclipse-windows-32`
  "CLIF Eclipse console": full-featured, Eclipse-based console for Windows/Intel 32 bits
- `clif-eclipse-windows-64`
  "CLIF Eclipse console": full-featured, Eclipse-based console for Windows/Intel 64 bits
- `clif-server`
  "CLIF server": provides the minimal runtime environment to be installed on a remote host for deploying probes and injectors. It is operating system-independent.
- `clif-selbench`
  "CLIF selfbench": CLIF lab for automatic test drive. Basically a "CLIF server" distribution enhanced with components for load injection regulation and saturation control (for advanced users).
- `clif-swing`
  "CLIF Swing console": a "CLIF server" enhanced with a simplified graphical console,

operating system-independent. This console supports test plan definition, execution and monitoring, as well as final analysis and reporting. Does not support editing scenarios ("ISAC").

- `eclipse-plugins`
  CLIF plugins for Eclipse. It allows to use the full-featured CLIF console from your Eclipse environment. Unzip to the `dropins` directory of your Eclipse environment.
- `jenkins-plugin`
  CLIF plugin for the Jenkins continuous integration server (http://jenkins-ci.org/). Supports importing CLIF test plans as Jenkins jobs. Load tests may be launched either on demand or automatically. Performance reports are automatically generated, including statistics and charts on response times and throughput. Performance trends through test runs are also reported.

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`" only gives the ant version.

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

1.4 Upgrading from CLIF v1.x.x

As of version 2.x.x of CLIF, some changes have occurred that introduce incompatibility issues with previous 1.x.x versions. The main change is about renaming `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` to `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

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 distribution zip file. The `clifcmd` script in the `bin` subdirectory gives all possible controls:

- for running a CLIF infrastructure, including configuration, registry and CLIF servers;
- for deploying test plans and controlling test executions;
- for getting valuable information about CLIF and its execution environment.

Refer to the CLIF user manual for details about available commands.

Should you get into trouble with command `clifcmd`, please first report to the CLIF mailing list (clif@ow2.org), and then think about using the command line interface based on the Apache `ant` utility (see section 2.4 below). It is deprecated but still working.

The Swing console must be installed on the host you will be using to define and run your test plans, using a graphical user interface. The CLIF server distribution must be installed on every host on which you'll deploy injectors and probes, as defined in your test plans. It can be used also to deploy and run existing test plans, but without any graphical interface.

2.2 Configuration

The Swing console just runs without any configuration as long as your test plans don't use remote CLIF servers. Configuration is necessary anytime you need to run CLIF servers on a different host than the one where the CLIF registry is running. The `clifcmd` script provides a configuration helper. Refer to the User Manual for details.

In order to be able to run the `clifcmd` command, you need either to prefix the command name with its path, or to add its absolute path (including the bin directory of the CLIF distribution) to your system PATH environment variable. The `clifcmd` command will use the `clif.opts` configuration file in current directory. If it does not exist, a default `clif.opts` file is automatically created from the template available in the CLIF distribution (in directory `etc/`).

2.3 Execution

To run the Swing console, enter command line: `clifcmd gui`

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 to be running.

To run a CLIF server, enter command line: `clifcmd server`

Refer to the CLIF User Manual for more information about distributed test plans and configuration of networking parameters.

2.4 Getting Apache ant working

Note. The ant-based interface is deprecated. Consider using the command line interface based on the `clifcmd` utility, which does not require `ant` to be installed.

2.4.1 For Windows

You must set the `JAVA_HOME` system variable, giving the path to your Java Runtime Environment or Development Kit directory (adapt this example accordingly to your specific Java set-up):
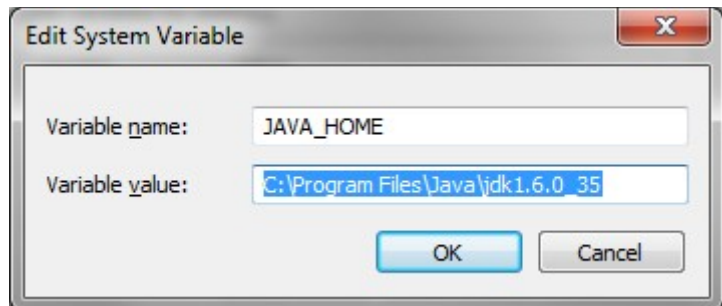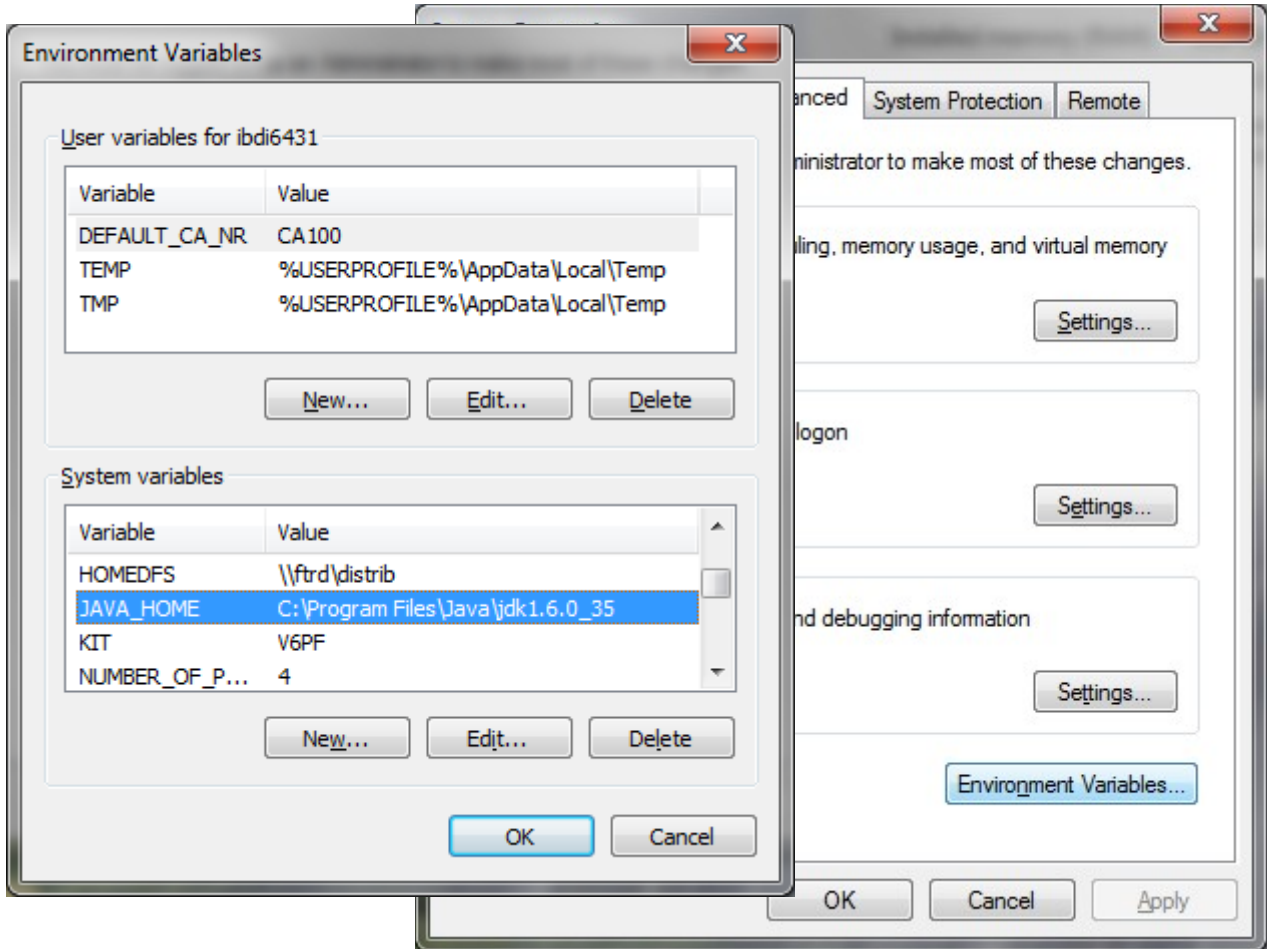
- `JAVA_HOME=C:\Program Files\Java\jdk1.6.0_35`

You may also add `Apache ant`'s binary directory to your `PATH` system variable, so that you can invoke the command without entering its full path:

- `PATH=`*`...preexisting path...;`*`\bin`

To set both system variables, open Windows' Control Panel, select the System panel, and click on the Advanced system settings link. Then, click on the "Environment Variables..." button.
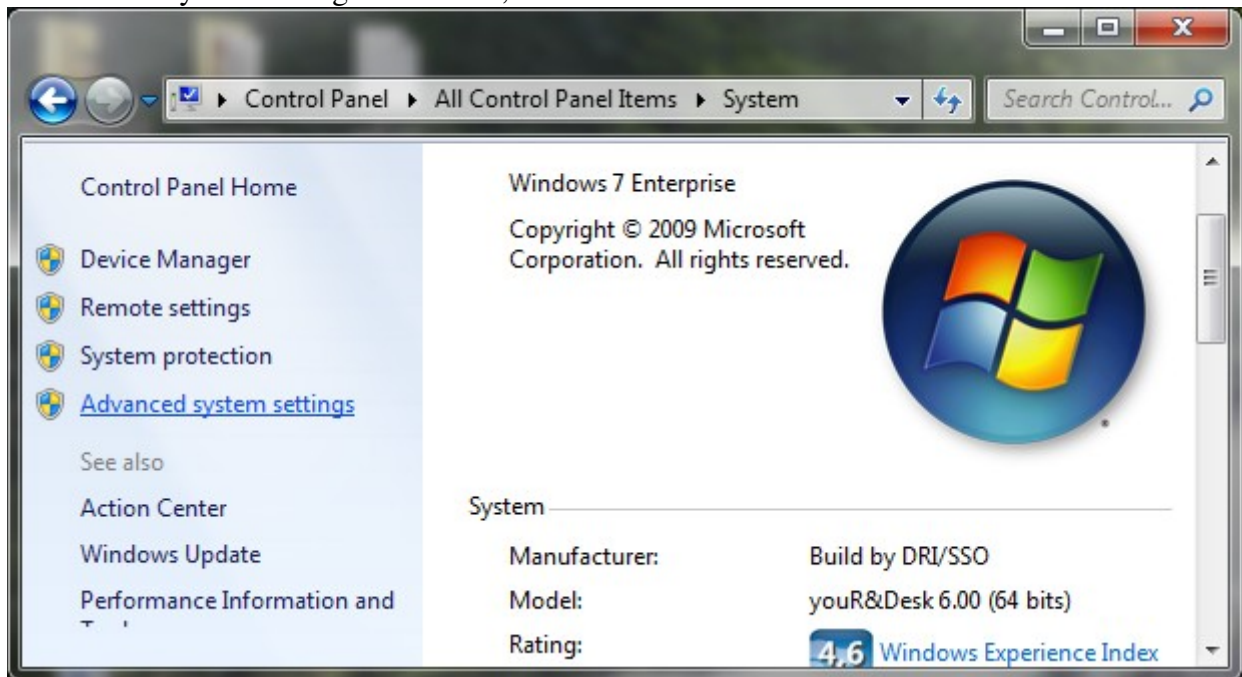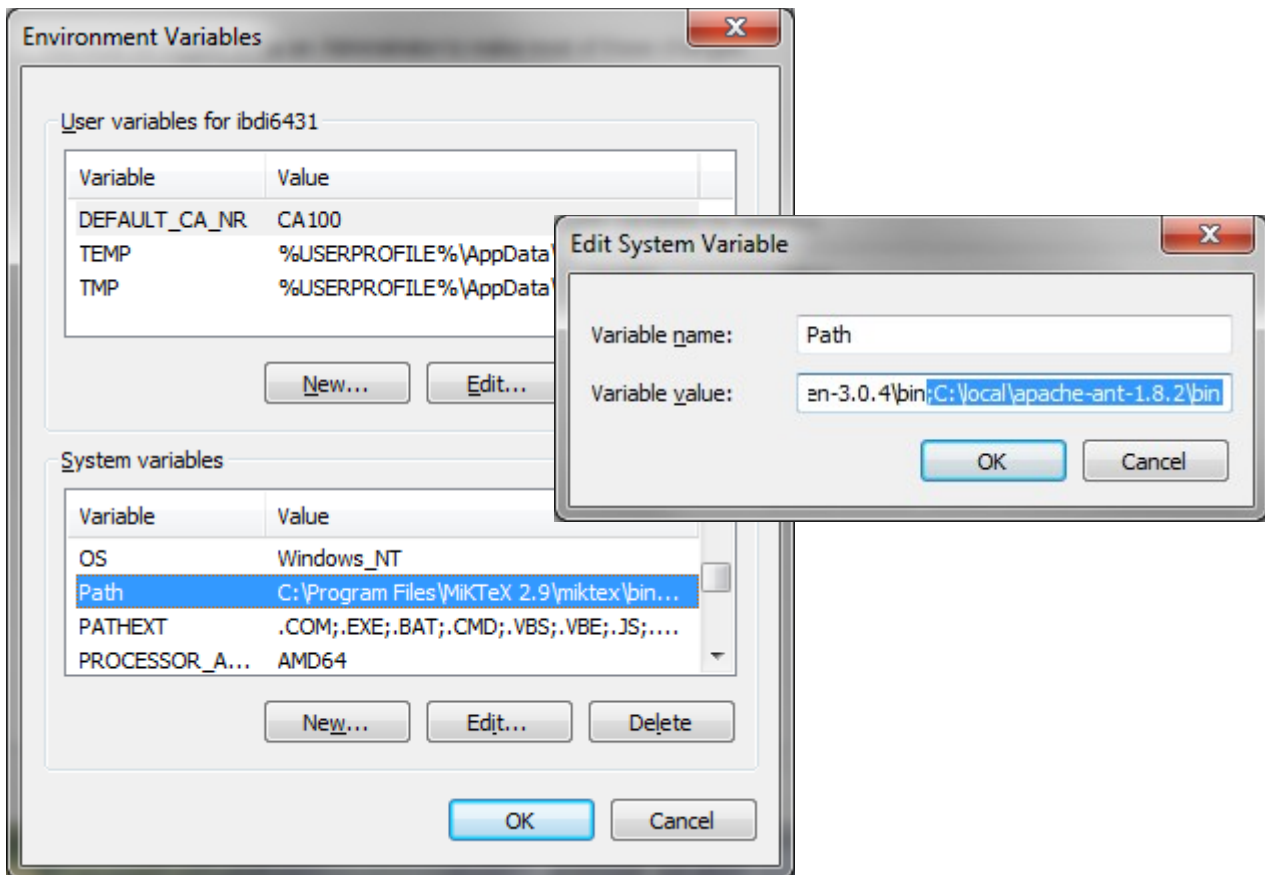

Define `JAVA_HOME` as a System variable, and add the path to `Apache ant`'s root directory to the `Path` system variable.

2.4.2 For Linux

Packages of `Apache ant` are provided by most Linux distributions. They make `ant` work just out of the box.

Should you experience troubles while trying to run `ant`, or if you manually install Java and `Apache ant` without your distribution's packaging tools, please refer to the following recipe:

You have to set the following environment variables (adapt this example according to the specific paths of your Linux distribution and Java installation):

- `JAVA_HOME=/usr/lib/jvm/java-6-openjdk`
- `ANT_HOME=/usr/share/ant`

If you did not install ant through the standard package installation process of your Linux distribution, then add ant's bin directory to your `PATH` environment variable:

- `PATH=$PATH:$ANT_HOME/bin`

The same issue also applies to the installation of Java:

- `PATH=$PATH:$JAVA_HOME/bin`

Edit the `.bash_profile` file (or equivalent according to your default shell) in your home directory and insert the following lines:

```
# User specific environment and startup programs
```

```
JAVA_HOME=/usr/lib/jvm/java-6-openjdk
ANT_HOME=/usr/share/ant
PATH=$PATH:$ANT_HOME/bin:$JAVA_HOME/bin
export ANT_HOME
export JAVA_HOME
```

Then, you must log out and in again to reload the .bash_profile file.

Finally, check if the right Java version and ant version are used by your system.

```
$ java -version
java version "1.6.0_24"
OpenJDK Runtime Environment (IcedTea6 1.11.5) (6b24-1.11.5-
0ubuntu1~11.04.1)
OpenJDK 64-Bit Server VM (build 20.0-b12, mixed mode)
$ ant -version
Apache Ant version 1.8.1 compiled on October 13 2010
```

2.4.3 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`

Refer to section 2.4.2 above to set these variables, for the configuration process is quite similar.

3 Installation of the Eclipse-based console

3.1 Install the Eclipse-based standalone CLIF distribution

Eclipse-based binary distributions of CLIF are standalone programs that include a full Eclipse environment for a specific operating system and architecture. All they require is a JDK.

Unzip the distribution anywhere on your computer, and simply run the `clif-console` program.

As an Eclipse application, a number of common options may be added to the `clif-console` command, 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 workspace directory different from the default `CLIFspace` directory;
- `-vmargs ...JVM options...` to pass arguments to the JVM.

3.2 Alternative: install the Eclipse plug-ins

Unzip the `eclipse-plugin` distribution to the `dropins` directory of your Eclipse environment. The CLIF menu should appear in your Eclipse menu bar.

Otherwise, some troubleshooting must be done to get CLIF plug-ins working. You may check the availability of 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
```