

How To Get Triana From CVS And Build From Source

Matthew Shields*

Abstract

This document describes the steps in correctly checking out the source code for the core Triana and Triana toolboxes from the cvs repository and building from the source code.

If you don't know what Triana is, don't know how to use a CVS client or don't have a CVS account, then it is recommended that you download the latest packaged version of Triana from the website <http://www.trianacode.org/projects/triana>.

1 Conventions

Note: These instructions assume you have the necessary user accounts and passwords. It is aimed at command line cvs users. WinCVS or similar users should be able to follow the instructions by using the repository and module names within their particular CVS Client.

- **Text written this font** should be typed as command line input. Commands should be entered without line breaks unless explicitly instructed.
- *Text written in this font* and surrounded by < ... > should be replaced by the users appropriate details.
- **Text in this font** signifies the unix command line prompt, the text following it will be the command to type.

*matthew.shields@astro.cf.ac.uk

2 CVSRoot and Passwords

The “CVSROOT” you should you will depend on whether you have read only “pserver” access or write “ext” access. This document assumes “pserver”, if you don’t know then ask the person who gave you your user name and password.

For the Core Triana and default Toolboxes repositories there is anonymous “pserver” access, use the username *<anonymous>* with no password, just hit return at the password prompt.

3 A Word About Directory Structure

For administration reasons Triana is split into a number of different packages which are stored in various CVS repositories. Some of these are project specific and not public so don’t assume that because something is listed in this document that you will be able to get the source code from the CVS server.

The public packages are:

1. Core Triana. The Triana Environment itself.
2. Toolboxes. The default toolboxes that come with Triana.
3. Toolboxes-dev. The unstable toolboxes that developers are currently working with. Use at you own risk.

The project specific packages are:

1. GEO. The Gravitational Waves tools.
2. Gravity. Example tools from the book “Gravity From The Ground Up” by Bernard Schutz.
3. GriPhyN. Tools from collaboration work with the GriPhyN project.

There are two standard ways to structure a Triana distribution from CVS, dependant on: whether you expect to be making regular changes to the source code under your control and/or you want to do frequent updates for the latest version from CVS; Or you just want to check out the latest version and build it once. CVS allows checking modules out inside other modules which will give the same structure as the packaged version of Triana but it will complain if an `update` or `commit` is attempted in the source tree where there is a foreign module lower down the tree.

4 Easy Install

These instructions will checkout and install Triana to the same structure as the packaged release version. It is fine for most users however if you expect to use CVS for more than updating small numbers of files it is suggested you use the other set of instructions.

This install will put all of the various toolboxes inside the Triana source tree and they will need to be removed to perform a `cv`s update on the core Triana code and then checked back out again.

From the command line:

```
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/triana login
(Logging in to username@trianacode.org)
CVS password: <userpassword>
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/triana checkout
-P triana
prompt$ cd triana
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/trianatools
login
(Logging in to username@trianacode.org)
CVS password: <userpassword>
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/trianatools
checkout -P toolboxes

Note: The rest of the Triana modules from CVS are optional and
depend on project permissions to access.

prompt$ cd toolboxes
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/trianatools
checkout -P toolboxes-dev
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/geotools
login
(Logging in to username@trianacode.org)
CVS password: <userpassword>
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/geotools
checkout -P GEO
```

```
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/gravity
login
```

(Logging in to username@trianacode.org)

```
CVS password: <userpassword>
```

```
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/gravity
checkout -P GravityFromTheGroundUp
```

```
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/griphyntools
login
```

(Logging in to username@trianacode.org)

```
CVS password: <userpassword>
```

```
prompt$ cvs -d :pserver:<username>@trianacode.org:/home/cvsroot/griphyntools
checkout -P GriPhyN
```

After those CVS commands you should have a directory structure like this:

```
triana/
  /bin/
  /toolboxes/
    /Audio
    /... other standard toolboxes
    /toolboxes-dev
    /GEO
    /GravityFromTheGroundUp/
    /GriPhyN
```

The toolbox structure is the import thing, there will be extra directories under `triana/` not mentioned here.

4.1 Build and Run

To build set an environment variable for your system, `$TRIANA` for Unix like systems or `%TRIANA%` for Windows, to point the top level `triana` directory. Run the build script, `buildTriana` for Unix or `buildTriana.bat` for Windows:

```
prompt$ $TRIANA/bin/buildTriana
```

Run the start script, `triana` or `triana.bat`:

```
prompt$ $TRIANA/bin/triana
```

5 Developer Install

If you are intending to write or modify any code within the various CVS modules or you just want to regularly update the modules from CVS. Then we suggest that you keep all the CVS modules in their own separate directory structures. The *Ant*¹ build file is written to be able to build from default either the previous structure or a directory structure where all the cvs modules are at the same level in the file system.

For instance my directory structure looks like this:

```
project/triana/  
project/toolboxes/  
project/toolboxes-dev/  
project/toolboxes_other  
project/toolboxes_other/GEO  
project/toolboxes_other/GravityFromTheGroundUp  
project/toolboxes_other/GriPhyN
```

So from a sensible starting directory run the CVS commands from section 4 with the `cd` commands so that the Triana core and default toolboxes modules from cvs reside in your current directory.

Note: The GEO, Gravity and Griphyn toolboxes should really reside in a separate subdirectory. Here I've created a directory called "toolboxes_other", this is because Triana assumes that a toolbox contains packages, so GEO for instance is the top level package for the GEO tools. When Triana attempts to locate tools it uses the following path :-

[toolbox][tool package][toolname]*

e.g. [/project/toolboxes/] [SignalProc/] [Input/] [Wave]
or [/project/toolboxes_other/] [GEO/] [Algorithms/] [SaturationMon]

The build and run instructions are almost the same as for the easy install.

- Set the TRIANA environment variable to point to the `triana` directory.
- Either edit the build file (`build.xml` in the main Triana home directory) and set the appropriate toolbox location variables in the

¹<http://ant.apache.org/>

“User Editable” section, or the preferred method add a new file called “build.properties” to the Triana home directory with the properties and their values. the contents of my “build.properties” file is listed below

```
javac.flag.debug=on
javac.flag.deprecation=on
 triana.geo.tools=../toolboxes_other/GEO
 triana.gravity.tools=../toolboxes_other/GravityFromTheGroundUp
 triana.griphyn.tools=../toolboxes_other/GriPhyN
```

- Run the buildTriana script.

Note: Unlike the standard installation, Triana will not automatically pick up the toolboxes when it is started so you will have to add those within Triana from the menu *Tools, Edit Tool Box Paths*. In my case, I select the directories - /project/toolboxes/, /project/toolboxes-dev/, /project/toolboxes_other/.

6 Other Install

You can actually have your toolbox modules anywhere you like in your file system and still have the build pick them up and compile them. In the file `build.xml` there is a commented section titled “USER EDITABLE PROPERTIES” within this section there are a number of commented out properties for the various toolbox modules. Uncomment any of these and set them to the appropriate location to override the default locations. Alternatively add a file called “build.properties” with the appropriate lines containing - `propertyname=value`.