

ILCInstall

ILC Software Installation Script

Jan Engels

DESY

02 May 2007

What is ILCInstall?

- Python script for installing the whole ILC Software framework + the needed external tools (e.g. GSL, CLHEP, CERNLIB, QT ...)
 - Automatic check of dependencies
 - Installation summary + preview
 - Maintain ILC software up-to-date with almost no effort
 - Written in Python 2.2.3 (www.python.org)
-

Different use-cases of ILCInstall

- **Every user:**
 - quickly install a piece or a set of ILC software tools without having to waste time with:
 - Downloading things by hand
 - Reading installation procedures
 - Manually deal with “dependency-hell”
 - Set environment variables
 - Edit environment shell scripts
 - **Administrators:**
 - Install and keep ILC Software repository up-to-date with almost no maintenance effort
 - **Private users:**
 - Quickly have a running, up-to-date version of the ILC Software
-

Some details of ILCInstall

- Different .cfg files for different purposes (e.g. *release.cfg*, *marlin_dev.cfg*, *lcio.cfg*)
 - Automatic generation of an environment shell script (*build_env.sh*) for each package
 - Automatic detection of required updates when installing a new package
 - Flexible definition of global or package dependent environment variables in the configuration file for building the software framework
 - Flexible definition of new Marlin packages + their dependencies
 - Installing core software takes about 1 hour (P4 3GHz)
 - External dependencies (e.g. GSL, CERNLIB, CLHEP, QT) can take up to 5 hours * Just leave the script running over night :)
 - If installation breaks for some reason, re-running the script will resume the installation at failure point (use generated *build_env.sh* scripts for fixing packages).
-

Install – Use - Link

- In a configuration file, packages can be set to 3 different modes (install, use and link):
- `Ilcsoft = ILCSOFT(“/scratch/ilcsoft”)`
- `Ilcsoft.install(LCIO(“v01-08”))`
 - Installs LCIO with the cvs tag “v01-08”
- `Ilcsoft.use(LCIO(“/scratch/my_lcio/v01-08”))`
 - This will look for an installed version of LCIO located at “/scratch/my_lcio/v01-08” and use it.
 - **ILCInstall** will automatically prompt for rebuilding packages in “use” mode that are not up-to-date with the configuration file!
- `Ilcsoft.link(LCIO(“/afs/desy.de/group/it/ilcsoft/lcio/v01-08”))`
 - Create a link to the directory given above in a subdirectory called “lcio” with the name “v01-08” and use it.
 - Linked packages are always safe!!
- “Use” can afterwards be switched to “version-only” : `Ilcsoft.use(LCIO(“v01-08”))`
 - Will search in “/scratch/ilcsoft/lcio/v01-08” for an already installed version of LCIO and use it.

+ /afs/desy.de/group/it/ilcsoft/

- CLHEP/
 - * 2.0.2.1
 - * 2.0.2.2
- GSL/
 - * 1.6/
 - * 1.8/
- RAIDA/
 - * v00-03
 - * v01-01
- CondDBMySQL/
 - * ILC-0-5-10
- MySQL/
 - * 5.0.26 -> link to /opt/mysql...
- lcio/
 - * v01-08
 - * v01-08-01
 - * v01-08-02
- lccd/
 - * v00-03-05`

Example of ILCSoft release

+ /afs/desy.de/group/it/ilcsoft/release_v01-00/

- CLHEP/
 - * 2.0.2.2 -> link to ../CLHEP/2.0.2.2
- GSL/
 - * 1.8 -> link to ../GSL/1.8
- lcio/
 - * v01-08-02 -> link to ../lcio/v01-08-02
- lccd/
 - * v00-03-05 -> link to ../lccd/v00-03-05
- Marlin/
 - * v00-09-06
- MarlinUtil/
 - * HEAD version
- MarlinReco/
 - * HEAD version

Packages without dependencies should be placed in a top-level ilcsoft directory

Packages with dependencies should remain in a subdirectory which contains links to the dependencies

Config-File

```
# ILCSOFT("install path for ILC software")
ilcsoft = ILCSOFT("/scratch/ilcsoft/test")
ilc_home = "/afs/desy.de/group/it/ilcsoft/" # define a python variable called ilc_home
ilcsoft.debug = True # global debug flag
ilcsoft.buildDoc = True # global documentation flag

# Marlin
ilcsoft.install( Marlin( "HEAD" ))
ilcsoft.module( "Marlin" ).env["MARLIN_USE_DLL"] = 1 # define environmet
# variable for Marlin installation

# Marlin Packages
ilcsoft.install( MarlinReco( "HEAD" ))
ilcsoft.install( MarlinUtil( "HEAD" ))
ilcsoft.install( CEDViewer( "HEAD" ))

# LCIO
ilcsoft.install( LCIO( "v01-08-01" ))

# GEAR
ilcsoft.install( GEAR( "v00-04-01" ))

# CERNLIB
ilcsoft.link( CERNLIB( ilc_home + "cernlib/2006" ))

# CLHEP
ilcsoft.link( CLHEP( ilc_home + "CLHEP/2.0.2.2" ))

# GSL
ilcsoft.link( GSL( ilc_home + "gsl/1.8" ))
```

Marlin User Packages

Marlin User defined Package

```
MyPackage = MarlinPKG( "MyPackage", "v00-03" )  
ilcsoft.install( MyPackage )
```

If your package can be downloaded with wget from an URL and is inside a tar.gz file:

```
MyPackage.download.url = "http://www.url-to-my-package.de/marlinpackages/mypackage.tar.gz"
```

If your package is in a cvs repository define required cvs settings, for example:

```
MyPackage.download.env["CVSROOT"] = ":pserver:anonymous@cvs.freehep.org:/marlinpkgs/mypackage"  
MyPackage.download.type = "cvs"  
#MyPackage.download.type = "ccvssh"
```

MyPackage dependencies

```
MyPackage.addDependency( [ "Marlin", "LCIO" ] )    # MyPackage cannot be built without Marlin and LCIO  
MyPackage.buildWith( [ "GEAR" ] )                # build MyPackage with GEAR (if available)
```

Marlin

```
ilcsoft.install( Marlin( "HEAD" ) )  
ilcsoft.module( "Marlin" ).buildWith( [ "MyPackage" ] ) # tell Marlin to build and link itself with this package
```

LCIO

```
ilcsoft.use( LCIO( "v01-08-01" ) )
```

GEAR

```
ilcsoft.use( GEAR( "v00-04-01" ) )
```


Marlin WORKDIR

After Install:

```
$ cd /scratch/myworkdir
$ cp ILCSoftInstallPath/Marlin/version/userlib_workdir.gmk ./userlib.gmk
$ cp ILCSoftInstallPath/Marlin/version/build_env.sh .
$ emacs build_env.sh (set MARLINWORKDIR=$PWD!!)
$ source build_env.sh
$ mkdir packages
$ cd packages
$ ln -s /scratch/mypackages/PandoraPFA PandoraPFA
$ cd $MARLIN
$ gmake binonly
```

Observations

- Choose other compiler...
 - Define `ilcsoft.env["MY_CXX"] = g++4` (limited support!!)
 - CMake (www.cmake.org) solves this issue. (Work in progress!!)
 - QT asks for user input (license issues) in the build process...
 - Move QT close to the top in your config file
 - Create separate configuration file for external dependencies
 - Install script works best when installing everything from scratch!!
 - Support for other operating systems (Mac, Windows, ...)
 - Python code is portable (still some hacks that need to be fixed)
 - Makefiles have to be changed (CMake)
 - Work in progress!!
 - No “install” support for ROOT, MySQL, JAIDA, AIDAJNI
 - Mokka, Geant4 not supported!!
-

Summary & Outlook

- Summary:
 - The ILCInstall script helps you to automatically install the ILCSoftware, maintain it, and keep track of the dependencies used in your builds!
 - Available at:
 - <http://www-zeuthen.desy.de/lc-cgi-bin/cvsweb.cgi/ilcinstall/?cvsroot=marlin>
- Outlook:
 - Support for Mokka, Geant4
 - Move over to new build tool: Cmake (<http://www.cmake.org>)
 - Support for other platforms (e.g. Mac, Windows, ...) (Cmake)
 - Get rid of dependency Marlin -> MarlinPKGs at install/compile time (DLL Support)
 - Work in Progress!!

Thank you for your attention!
Your Feedback is welcome!

Thank you for your attention!

Questions, Suggestions?
