



## EasyBuild: building software with ease

Kenneth Hoste  
HPC-UGent, Ghent University, Belgium  
kenneth.hoste@ugent.be

[http://users.ugent.be/~kehoste/EasyBuild-intro-Basel\\_20150209.pdf](http://users.ugent.be/~kehoste/EasyBuild-intro-Basel_20150209.pdf)

EasyBuild hackathon @ sciCORE/SIB/UniBas, Switzerland  
20150209

# HPC-UGent in a nutshell



- HPC team at central IT dept. of Ghent University (Belgium)
- 9 team members: 1 manager, ~3 user support, ~5 sysadmin
- 6(+2) Tier2 clusters + one Tier1 (8.5k cores), >1k servers in total
- ~1.5k user accounts, across all scientific domains
- tasks: hardware, system administration, user support/training, ...
- member of Flemish Supercomputer Centre (VSC)
  - virtual centre, collaboration between Flemish university associations



# “Please install this on the HPC?”

In the context of high performance computing, *building from source* should be preferred, when possible (i.e., if sources are available).

This allows for controlling used compilers and libraries, optimizing the software for the specific system architecture (e.g., AVX, network), etc.

Installing (lots of) *scientific* software is typically:

- error-prone, trial-and-error
- tedious, hard to get right
- repetitive & boring (well...)
- time-consuming (hours, days, even weeks)
- frustrating (“*Pandora’s box*”)
- sometimes simply not worth the effort...



# Common issues with scientific software

Researchers focus on the *science* behind the software they implement, and care little about tools, build procedure, portability, . . .

Scientists are not software developers or sysadmins (nor should they be).

This results in:

- use of non-standard build tools (or broken ones)
- incomplete build procedure, e.g., no configure/install step
- interactive installation scripts
- hardcoded parameters (compilers, libraries, paths, . . .)
- poor/outdated/missing/incorrect documentation
- dependency (version) hell

*“If we would know what we are doing, it wouldn't be called ‘research’.”*

# Prime example I: WRF

Weather Research and Forecasting Model (<http://www.wrf-model.org>)

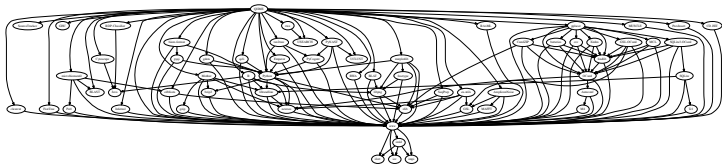
- dozen dependencies: netCDF (C, Fortran), HDF5, tcsh, JasPer, ...
- known issues in last release are (only) documented on website  
no patch file provided, infrequent bugfix releases
- interactive 'configure' script :(
- resulting `configure.wrf` needs work:  
fix hardcoded settings (compilers, libraries, ...), tweaking of options
- custom 'compile' script (wraps around 'make')  
building in parallel is broken without fixing the Makefile
- no actual installation step

**Wouldn't it be nice to build & install WRF with a single command?**

[http://easybuild.readthedocs.org/en/latest/Typical\\_workflow\\_example\\_with\\_WRF.html](http://easybuild.readthedocs.org/en/latest/Typical_workflow_example_with_WRF.html)

# Prime example II: QIIME

QIIME: Quantitative Insights Into Microbial Ecology (<http://qiime.org/>)



- scientific research domain: bioinformatics ...
- 59 dependencies in total (*without* compiler toolchain), some optional
  - depends on Haskell (GHC), Java, Python, R, Perl, OCaml, ...
  - several deps use a non-standard build procedure (in various degrees)
- very picky about dependency versions (e.g., *must* be Python v2.7.3)
- took us several weeks to get it installed (like we wanted)...
- ... **now we can (re)build/install it all with a single command!**

*(disclaimer: QIIME is not supported yet in the latest EasyBuild release)*

# What about existing tools?

Existing tools are not well suited to scientific software and HPC systems.

- package managers: **yum** (RPMs), **apt-get** (.deb), ...
- **Homebrew** (Mac OS X), <http://brew.sh/>
- **Linuxbrew**, <http://brew.sh/linuxbrew/>
- **Portage** (Gentoo), <http://wiki.gentoo.org/wiki/Project:Portage>
- **pkgsrc** (NetBSD & (a lot) more), <http://pkgsrc.org/>
- **Nix**, <http://nixos.org/>

Common problems:

- usually poor support for multiple versions/builds existing side-by-side
- not flexible enough to deal with idiosyncrasies of scientific software
- hard to maintain (bash, heavy copy-pasting, ...)
- little support for scientific software, other compilers (not GCC), ...

# Houston, we have a problem

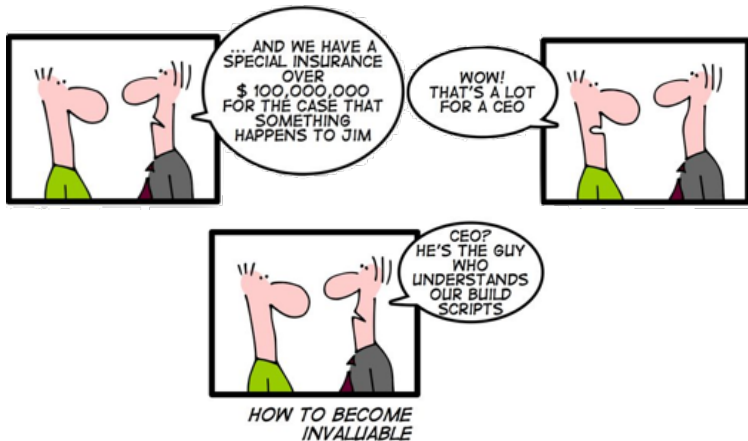
Installation of scientific software is a *tremendous* problem for HPC sites all around the world.

- huge burden on HPC user support teams
- researchers lose time just to get stuff installed
- every site deals with it in its own way (scripting, ...)
- very little collaboration among HPC sites :(





# How to become invaluable



<http://geekandpoke.typepad.com/geekandpoke/2010/05/how-to-become-invaluable.html>

# EasyBuild: building software with ease



<http://hpcugent.github.io/easybuild/>

- open source (GPLv2) framework for building and installing software
- collection of Python packages and modules
- original implementation by HPC-UGent, since 2009
- thriving community: actively contributing, driving development
- new release every 4–6 weeks (latest: EasyBuild v1.16.1, Dec'14)  
next release: EasyBuild v2.0.0!
- supports *over 550* different software packages  
including CP2K, NAMD, NWChem, OpenFOAM, PETSc,  
QuantumESPRESSO, WRF, WPS, ...
- well documented: <http://easybuild.readthedocs.org>

# Similar projects

- **Spack** (LLNL), <http://scalability-llnl.github.io/spack/>
- **iVEC Build System (iBS)**, <http://ivec.org/> (*not public (yet)*)
- **Smithy** (NICS, ORNL), <http://anthonydigirolamo.github.io/smithy/>

Major differences with EasyBuild:

- smaller community
- fewer supported software packages
- less flexibility

Most (all?) are interested in switching to/merging with EasyBuild.

# EasyBuild: requirements

- main target platform (for now) is Linux x86\_64 (HPC systems)
  - Red Hat Linux based (Scientific Linux, CentOS, RHEL, ...)
  - also other Linux distros: Debian, Ubuntu, OpenSUSE, SLES, ...
  - (kind of) works on OS X
  - *no* Windows support (and none planned)
  - support for Cray systems and Linux on POWER is being looked into
- Python v2.6.x or more recent v2.x (no Python 3 support yet)
- a modules tool:
  - latest release of Tcl(/C) environment modules;
  - or a recent version of *Lmod*
- (a system C/C++ compiler, to get started)

# EasyBuild: feature highlights

- fully **autonomously** building and installing (scientific) software
  - automatic dependency resolution
  - automatic generation of module files
- thorough **logging** of executed build/install procedure
- **archiving** of build specifications ('*easyconfig files*')
- highly **configurable**, via config files/environment/command line
- **dynamically extendable** with additional *easyblocks*, *toolchains*, etc.
- support for **custom module naming schemes** (incl. hierarchical)
- **comprehensively tested**: lots of unit tests, regression testing, ...
- active worldwide **community**

# 'Quick' demo for the impatient

```
eb HPL-2.0-goolf-1.4.10-no-OFED.eb --robot
```

- **downloads** all required sources (best effort)
- **builds/installs** *goolf* toolchain (be patient) + HPL on top of it  
*goolf*: GCC, OpenMPI, LAPACK, OpenBLAS, FFTW, ScaLAPACK
- **generates module file** for each installed software package
- default: source/build/install dir in `$HOME/.local/easybuild`  
can be easily changed by configuring EasyBuild differently

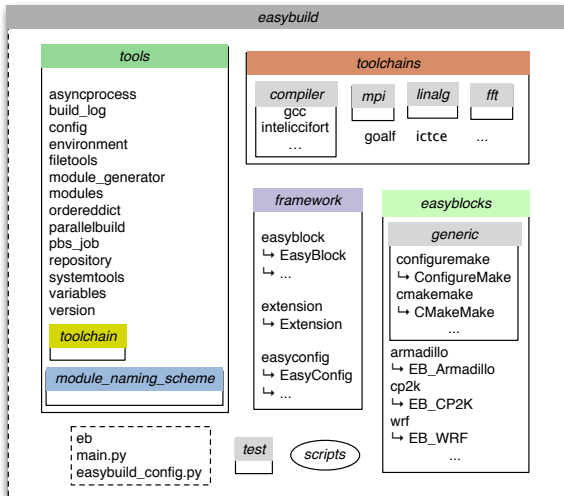
# EasyBuild: high-level design overview

- EasyBuild *framework*
  - core of EasyBuild
  - provides supporting functionality for building and installing software
- *easyblock*
  - a Python module
  - implements a (generic) software build/install procedure
- *easyconfig* file
  - build specification: software name/version, compiler toolchain, etc.
- compiler *toolchain*
  - compilers with accompanying libraries (MPI, BLAS/LAPACK, etc.)

## Putting it all together

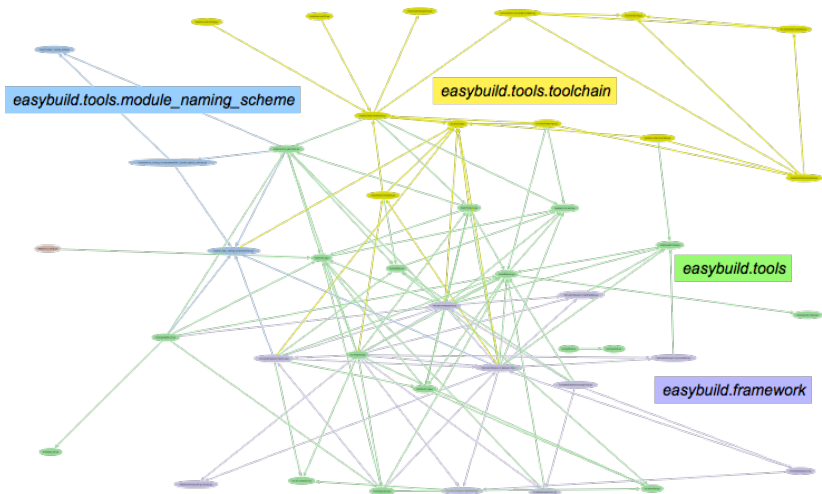
The EasyBuild *framework* leverages *easyblocks* to automatically build and install (scientific) software using a particular *compiler toolchain*, as specified by one or multiple *easyconfig files*.

# EasyBuild: high-level design overview



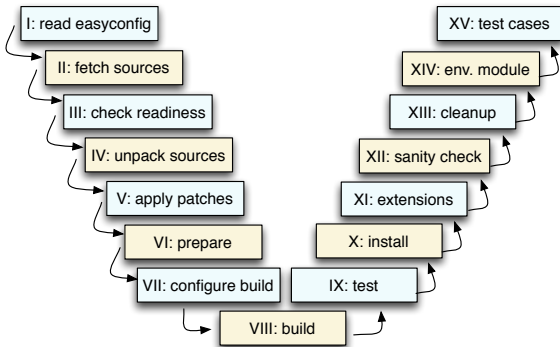


# EasyBuild: high-level design overview



# Step-wise install procedure

build and install procedure as implemented by EasyBuild



most of these steps can be customised if required,  
via *easyconfig* parameters or a *custom easyblock*

# EasyBuild: statistics

## EasyBuild v1.16.1

- ~ 19,500 LoC in framework (14 Python packages, 132 Python modules)
  - + ~ 4,500 LoC in vsc-base library it requires
  - + ~ 7,000 LoC more in unit tests
  - ⇒ ~ 31,000 LoC in total
- 161 easyblocks in total
  - 141 software-specific easyblocks
  - 20 generic easyblocks
  - ~ 15,000 LoC
- 597 different software packages supported (incl. toolchains)
  - bio: 113, tools: 73, lib: 62, devel: 55, vis: 51, data: 40,
  - toolchain: 39, math: 39, numlib: 22, mpi: 21, chem: 24,
  - lang: 24, system: 17
- 3,066 easyconfig files: different versions/variants, toolchains, ...

# List of supported software packages

a2ps ABAQUS ABINIT ABySS ACML **ALADIN** Allinea ALLPATHS-LG AMOS AnalyzeFMRI ANSYS ant ANTs APBS ARB argtable aria2 Armadillo arpack-ng ASE ATLAS Autoconf Automake bam2fastq BamTools Bash BayesTraits bbcp bbFTP bbftpPRO bc beagle-lib Beast BEDOPS BEDTools BFASt binutils BioPerl Biopython BiSearch Bison BitSeq BLACS BLAST BLAT BOINC Bonnie++ Boost Bowtie Bowtie2 BWA byacc bzip2 cairo CAP3 CBLAS ccache Ccfits CD-HIT CDO CEM CFITSIO cflow CGAL cgdg Chapel CHARMM Chimera Circos Clang CLHEP CLoog Clustal-Omega ClustalW2 CMake Coreutils Corkscrew **CP2K** CPLEX CRF++ ctfindin Cube CUDA Cufflinks cURL cutadapt CVS CVXOPT Cython DB DBD-mysql DBD-SQLite DB\_File DIALIGN-TX Diffutils DL\_POLY\_Classic Docutils **DOLFIN** Doxygen **EasyBuild** ECore ed Eigen ELinks ELPA ELPH Emacs EMOSSO EFD ErlangOTP ESMF ESPResSo evmix expat eXpress FASTA fasthack FastTree FASTX-Toolkit FCM FDS FDTD\_Solutions Ferret FFC FPM FFF FFW FIAT file findutils fixesproto flex FLTK FLUENT fmri FoldX fontconfig FRC\_align freeglut FreeSurfer freetype FSL g2clib g2lib GAMESS-US GATE GATK gawk GCC GD GDAL GDB Geant4 GEM-library GEMSTAT GenomeAnalysisTK GEOS gettext GHC Ghostscript GIMPS gif GLib GLIMMER GLPK glproto GMAP-GSNAP GMP GMT gnuplot gnutls Go GObject-Introspection google-sparsehash GPAW gperftools grace Graphviz GraphViz Greenlet grep grib\_api GROMACS GSL gsl GSSAPI GTI GTS guile gzip h4toh5 h5py h5utils HarfBuzz Harmin HDF HDF5 HH-suite HMMER horton HPCG HPL HTSeq HTSLib hwloc Hypr icc ifort imake imkl impi Infernal inputproto Inspector Instant intltool iompi IOR Iperf ipy IPython Isoliner ispc itac JAGS Jambon JasPer Java Jellyfish Jinja2 JUnit kbproto Kerberos.V5 LAPACK less lftp libcap-ng libcircle libclt libdrm libevent libffi libgd libgtextutils libharu libibmad libibumad libibverbs libICE libidn Libint libint2 libjpeg-turbo libmatheval libpciaccess libpng libpthread-stubs libreadline libSMM libsvm LibTIFF libtool libudev libungif libunistring libunwind libX11 libXau libXaw libxc libxcb libXdmcp libXext libXfixes libXft libXi libXinerama libxml2 libXmu libXp libXpm libXrender libxslt libXt libyaml likwid Lmod Lua LWM2 lxml lynx LZO M4 MAFFT make makedefend Maple MariaDB Mathematica MATLAB matplotlib Maven mc MCL mcpp MDP mdtest Meep MEME Mercurial Mesa Mesquite MetaVelvet MethPipe METIS MMSEQ Modeller Molden Molekel molmod Mothur motif MPFR mpi4py mpiBLAST MPICH MPICH2 MRBayes MTL4 MUMMER MUSCLE MUST MUSTANG MVAPICH2 MySQL NAMD nano NASM NCBI-Toolkit ncd4 **NCL** ncurses ncview Nedit netaddr netCDF netCDF-C++ netCDF-C++4 netCDF-Fortran netcdf4-python netifaces NetLibIDN netloc nettle NEURON nodesjs nnumactl numexpr numpy NWChem O2scl Oases OCaml Oger OPARI2 OpenBabel OpenBLAS OpenCV **OpenFOAM** **OpenFOAM-Extend** OpenIFS OpenMD OpenMPI OpenPGM OpenSees OpenSSL ORCA orthoncol otcl OTF OTF2 packmol PAML pandas PANDASEQ Pango PAPI parallel Paraview PkgFlow ParMETIS ParmGridGen Pasha patch paycheck PCC PCRE PDT Perl **PETSc** petsc4py phonopy PhyML picard pixman pkg-config PLINK PnMPI popt PP PRANK Primer3 printproto problog protobuf pscom PSI psmpi psmpi2 PyQuante pysqlite pyTables **Python** python-dateutil python-meep PyYAML PyZMQ Qhull QLOGICMPI Qt qtop QuadProg++ **QuantumESPRESSO** R RAxML RCS RDP-Classifier RELION renderproto rjags RNAz ROOT Rosetta rSeq RSEQttools Ruby runjags Sabletron SAMtools ScalAPACK Scalasca ScientificPython scikit-learn scipy SCons SCOOP Score-P SCOTCH SDCC SDPA sed segemehl setuptools Shapely SHRIMP SIBELia sickle Silo slalib-c SLEPC SOAPaligner SOAPdenovo SOAPdenovo2 SOAPec SPAdes Sphinx SPRNG SQLite SRA-Toolkit Stacks stemming Stow Stride SuiteSparse SURF SWIG sympy systemd Szip TAMkin Tar tbb TCC Tcl tccl tcsh Tesla-Deployment-Kit texinfo Theano TiCCutlits TiMBL TinySVM Tk TopHat TopHat-TopHat TotalView TREE-Puzzle2 Trilinos Trinity UDUNITS UFC UFL util-linux Valgrind VCfTools Velvet ViennaRNA Vim Viper vsc-base vsc-mypirun vsc-mypirun-scoop vsc-processcontrol VSC-tools VTK VTune WHAM **WIEN2k** wiki2beamer **WPS** **WRF** xbitmaps xcb-proto XCrySDen xextproto xineramaproto XML XML-Dumper XML-LibXML XML-Parser XML-Simple XML-Twig xorg-macros xproto xtrans XZ yaff YamCha YAML-Syck Yasm YAXT ZeroMQ zlib zsh zsync

# Installing EasyBuild

<http://easybuild.readthedocs.org/en/latest/Installation.html>

Install EasyBuild using the bootstrap script (*highly recommended*):

```
$ curl -O https://raw.githubusercontent.com/hpcugent/easybuild-framework/develop/easybuild/scripts/bootstrap_eb.py
```

```
$ python bootstrap_eb.py /tmp/$USER # specify your install prefix
```

```
$ export MODULEPATH=/tmp/$USER/modules/all:$MODULEPATH
```

```
$ module load EasyBuild
```

Update EasyBuild with ... EasyBuild!

```
$ module load EasyBuild/1.15.2
```

```
$ eb EasyBuild-1.15.1.eb --try-software-version=1.16.1
```

```
$ module swap EasyBuild EasyBuild/1.16.1
```

# Configuring EasyBuild

<http://easybuild.readthedocs.org/en/latest/Configuration.html>

By default, EasyBuild will (ab)use `$HOME/.local/easybuild`.

You should configure EasyBuild to your preferences, via:

- *configuration file(s)*: key-value lines, text files (e.g., `prefix=/tmp`)
- *environment variables* (e.g., `$EASYBUILD_PREFIX` set to `/tmp`)
- *command line parameters* (e.g., `--prefix=/tmp`)

Consistency across these options is guaranteed (see `eb --help | tail`).

Priority among different options: cmdline, env vars, config file.

For example:

- `--prefix` overrules `$EASYBUILD_PREFIX`
- `$EASYBUILD_PREFIX` overrules `prefix` in configuration file

# First steps with eb

- installing bzip2 v1.0.6 with system compiler (discouraged):

```
$ eb bzip2-1.0.6.eb  
$ module load bzip2/1.0.6
```

- or (equivalent), install latest version (that EasyBuild knows about):

```
$ eb --software-name=bzip2 --toolchain-name=dummy
```

- install goolf compiler toolchain (be patient):

```
$ eb goolf-1.4.10-no-OFED.eb --robot # no-OFED: no IB support
```

- install gzip v1.6 on top of goolf toolchain, debug log to stdout:

```
$ eb gzip-1.6-golf-1.5.14-no-OFED.eb -ldr
```

# EasyBuild command line cheat sheet

- getting help, overview of options: `eb --help`
- list of available easyconfig parameters: `eb -a`
- robot build, debug log to stdout: `eb bzip2-1.0.6.eb -ldr`
- overview of required/available modules: `eb --dry-run` or `eb -D`
- list of known toolchains and their definition: `eb --list-toolchains`
- generating other easyconfigs for the same software package:  
`eb bzip2-1.0.6.eb --try-toolchain=ictce,6.2.5 -r`
- searching for easyconfigs: `eb -S` or `eb --search`
- use easyconfigs available in an pull request: `eb --from-pr <PR#>`
- test pull request and upload test report:  
`eb --from-pr <PR#> --upload-test-report --github-user=boegel`



# Writing easyconfig files, contributing back

For software packages that follow some type of standard build procedure, just writing an easyconfig file is likely sufficient.

[http://easybuild.readthedocs.org/en/latest/Writing\\_easyconfig\\_files.html](http://easybuild.readthedocs.org/en/latest/Writing_easyconfig_files.html)

If you have working easyconfigs for new software packages, version updates, or using a different toolchain, consider *contributing back* to the central easybuild-easyconfigs repository.

<https://github.com/hpcugent/easybuild/wiki/Contributing-back>

Over 25% of the builds currently supported were defined by *external contributors* (non-HPC-UGent), and that ratio is increasing every day.

# Synergy between EasyBuild and Lmod

- EasyBuild can easily build and install hundreds of packages
  - ⇒ lots of modules, overwhelming for users
- Lmod's support for hierarchical modules trees can help
  - ⇒ support for using Lmod and hierarchical module naming schemes was added to EasyBuild
- EasyBuild has helped uncover multiple performance issues in Lmod
  - ⇒ Lmod has significantly improved the speed of certain operations
    - `module --terse avail` (doesn't parse module files)
    - faster `module avail` with Lmod system cache in place (v5.8.6)
- feature requests from the EasyBuild community
  - ⇒ Lmod has added new functionality, for example stack-based definition of environment variables using 'pushenv'

The synergy between both tools has made them significantly better!

# EasyBuild community

- almost 100 subscribers to the EasyBuild mailing list
- 15-20 active members on the #easybuild IRC channel
- users/contributors at HPC sites and companies around the world  
incl. sciCORE/UniBas, Flemish Supercomputer Centre sites,  
Jülich Supercomputer Centre, Stanford Univ., Univ. of Auckland,  
Bayer AG, Texas A&M, IMB (Austria), Univ. of Luxembourg,  
Cyprus Institute, ...
- *"Getting Scientific Software Installed"* BoF sessions at ISC/SC
- 8 'hackathon' workshops (2-3 days) since Aug'12
  - various European HPC sites: Ghent (Belgium), Luxembourg, Nicosia (Cyprus), Jülich (Germany), Vienna (Austria), Basel (Switzerland)
  - **next(?) hackathon at TACC (Austin, Texas), week before SC15**

# EasyBuild: recent developments

- EasyBuild v1.15.0, v1.15.1, v1.15.2 (Sept-Oct'14)
  - support for installing hidden modules
  - various fixes related to using a hierarchical module naming scheme
  - support for 13 new software packages (incl. IOR, NAMD)
- revamped documentation (Nov'14)
  - ported existing wiki pages to Read the Docs platform (reStructuredText format)
  - <http://easybuild.readthedocs.org/>
  - easy to contribute via <https://github.com/hpcugent/easybuild>
- EasyBuild v1.16.0, v1.16.1 (Dec'14)
  - support for `--robot-paths` configure option
  - deprecate automagic fallback to ConfigureMake easyblock
  - prepare to drop support for deprecated behaviour in EasyBuild v2.0
  - support for 39 new software packages (incl. OpenCV, OpenMD)

# Proper documentation covering the basics

EasyBuild installation:

<http://easybuild.readthedocs.org/en/latest/Installation.html>

EasyBuild configuration:

<http://easybuild.readthedocs.org/en/latest/Configuration.html>

Usage of eb command line:

[http://easybuild.readthedocs.org/en/latest/Using\\_the\\_EasyBuild\\_command\\_line.html](http://easybuild.readthedocs.org/en/latest/Using_the_EasyBuild_command_line.html)

Writing easyconfig files:

[http://easybuild.readthedocs.org/en/latest/Writing\\_easyconfig\\_files.html](http://easybuild.readthedocs.org/en/latest/Writing_easyconfig_files.html)

Navigating log files:

<http://easybuild.readthedocs.org/en/latest/Logfiles.html>

# HUST-14 paper

## Modern Scientific Software Management Using EasyBuild and Lmod

*Markus Geimer (JSC)*

*Kenneth Hoste (HPC-UGent)*

*Robert McLay (TACC)*

[http://hpcugent.github.io/easybuild/files/hust14\\_paper.pdf](http://hpcugent.github.io/easybuild/files/hust14_paper.pdf)

- paper at HUST-14 workshop (during SC14)
- explains basics of module tools, EasyBuild and Lmod
- highlights issues with current approaches in software installation
- advocates use of a hierarchical module naming scheme
- presents EasyBuild and Lmod as adequate tools for software management

# EasyBuild v2.0

<https://github.com/hpcugent/easybuild-framework/issues/1000>

- **drop Python v2.4.x support, move on to Python v2.6.x**  
Py2.6 is necessary to start working towards Py3.x support
- **drop support for deprecated behaviour** (significant code cleanup)  
<http://easybuild.readthedocs.org/en/latest/Deprecated-functionality.html>  
concerns both framework and easyblocks
- **add support for specifying particular easyconfigs with `--from-pr`**
- *stop including vsc-base, use it as a proper dependency* (WIP)
- more robust unit tests, skip those relying on optional deps
- provide a way to fix broken easyconfig files (WIP)
- add support for `--review-pr` (WIP)

*ETA: (1st week of) February 2015*

# EasyBuild: future work

- command line support for contributing easyconfigs: `eb --new-pr`
- support for Lmod-specific features in EasyBuild
  - module files in Lua syntax (*WIP*)
  - module properties
  - module families
- make the dependency resolution mechanism aware of 'subtoolchains'
- supporting multiple module naming schemes concurrently (e.g., to gradually move from one layout to another) (*wip*)
- better support for `eb --job` (*WIP*)
- support for packaging software installed using EasyBuild (via FPM?)
- support for rpath-style linking of libraries
- Cray support (*WIP*)



# Do you want to know more?

- EasyBuild website: <http://hpcugent.github.com/easybuild>
- EasyBuild documentation: <http://easybuild.readthedocs.org>
- stable EasyBuild releases: <http://pypi.python.org/pypi/easybuild>
  - EasyBuild framework: <http://pypi.python.org/pypi/easybuild-framework>
  - easyblocks: <http://pypi.python.org/pypi/easybuild-easyblocks>
  - easyconfigs <http://pypi.python.org/pypi/easybuild-easyconfigs>
- source repositories on GitHub
  - EasyBuild meta package + docs: <https://github.com/hpcugent/easybuild>
  - EasyBuild framework: <https://github.com/hpcugent/easybuild-framework>
  - easyblocks: <https://github.com/hpcugent/easybuild-easyblocks>
  - easyconfigs: <https://github.com/hpcugent/easybuild-easyconfigs>
- EasyBuild mailing list: [easybuild@lists.ugent.be](mailto:easybuild@lists.ugent.be)  
<https://lists.ugent.be/wvs/subscribe/easybuild>
- Twitter: @easy\_build, IRC: #easybuild on [chat.freenode.net](http://chat.freenode.net)