



## Getting Started with EasyBuild

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

[http://users.ugent.be/~kehoste/EasyBuild\\_HPCAC\\_start\\_20160323.pdf](http://users.ugent.be/~kehoste/EasyBuild_HPCAC_start_20160323.pdf)

March 23rd 2016 – HPC Advisory Council conference – Lugano

# Topics

- installing and configuring EasyBuild
- basic usage of 'eb'
- typical workflow
- creating new or customizing existing easyconfig files
- easyconfigs vs easyblocks
- figuring out what EasyBuild is going to do
- dealing with failures/errors
- contributing back
- advanced features

# 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 # replace with your prefix!  
$ module use /tmp/$USER/modules/all  
$ module load EasyBuild
```

Standard Python install tools (`easy_install`, `pip`, ...) should also work.

Update EasyBuild with ... EasyBuild!

```
$ module load EasyBuild/2.6.0  
$ eb EasyBuild-2.7.0.eb  
$ module swap EasyBuild EasyBuild/2.7.0
```

# 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` overrides `$EASYBUILD_PREFIX`
- `$EASYBUILD_PREFIX` overrides `prefix` in configuration file

Use `eb --show-config` to get an overview of the current configuration.

# Basic usage

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

- specify software name/version and toolchain to 'eb' command
- commonly via name(s) of easyconfig file(s):

```
eb GCC-4.9.2.eb Clang-3.6.0-GCC-4.9.2.eb
```

- or directory name(s) providing set(s) of easyconfig files:

```
eb $HOME/myeasyconfigs
```

- or via command line options:

```
eb --software-name=GCC
```

- --robot/-r: dependency resolution, --debug/-d: debug logging

```
eb WRF-3.6.1-foss-2015a-dmpar.eb -dr
```

# Typical workflow

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

- searching for available easyconfigs (case-insensitive):

```
eb -S hpl
```

- pick an easyconfig file, based on software version, toolchain, etc.
- overview of required/available modules (dry run):

```
eb HPL-2.1-foss-2016a.eb -D
```

- enable debug logging & dependency resolution ('robot' mode):

```
eb HPL-2.1-foss-2016a.eb -dr
```

# Creating easyconfig files

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

- install another software version, use different compiler toolchain, tweak build options, ...  $\implies$  new easyconfig file
- for simple changes, use `--try-*` command line options
  - different software version: `--try-software-version`
  - different compiler toolchain: `--try-toolchain`
  - the `--try-*` options integrate with `--robot`
- or copy existing easyconfig file, adjust as needed
  - 'non-trivial' changes: update dependencies, change build options, ...
  - for new software packages
- 28 generic easyblocks support over 80% of supported software!

# Easyconfigs vs easyblocks

- thin line between 'fat' easyconfigs and (software-specific) easyblock
- easyblocks are "do once and forget"
- central solution for build peculiarities
- easyblocks allow to significantly simplify easyconfigs
- implemented in Python on top of framework API: very flexible
- reasons to consider an easyblock alongside a simple easyconfig:
  - 'critical' values for easyconfig parameters
  - configure/build/install options that are toolchain-dependent
  - custom (configure) options for included dependencies
  - hackish usage of parameters for existing (generic) easyblocks



# eb --extended-dry-run (or eb -x)

eb WRF-3.6.1-foss-2015a-dmpar.eb -x

- figure out how EasyBuild is going to perform an installation
- runs in seconds, reports install procedure that *would* be performed
- not 100% accurate, but close enough to be practical
- opens up 'black box' that EasyBuild is sometimes perceived to be
- useful for:
  - more quickly debugging easyconfigs/easyblocks
  - understanding the effect of tweaking parameters/options
  - giving (new) users more confidence about using EasyBuild
- implemented by popular request from (new) community members

# Dealing with errors

- EasyBuild is not perfect (and it will probably never be)
- most frequent issues:
  - failure to download source tarballs/installation files
    - required files can be provided manually in EasyBuild source path
  - build failures of old software versions on recent systems
    - reconsider upgrading to newer version(s)
    - see if issue can be side-stepped with a minor tweak (Google. . . )
  - missing dependency specification in EasyBuild
    - report problem (via GitHub issue tracker)
    - maybe try fixing it yourself, & contribute back?
  - (un)known bug in EasyBuild
- consult EasyBuild log file, enable `--debug` if needed  
<http://easybuild.readthedocs.org/en/latest/Logfiles.html>
- getting help: EasyBuild mailing list, IRC, bi-weekly conf calls, . . .  
<http://easybuild.readthedocs.org/en/latest/#getting-help>

# Contributing back

- contribute back your working easyconfigs/easyblocks!
- also: report bugs, share ideas for enhancements, patches, etc.
- share your expertise with the community, avoid duplicate work
- especially if:
  - software package is not supported yet
  - changes are required for a new version/toolchains
  - it is frequently used software package (compilers, MPI, etc.)
- requires a limited amount of knowledge of Git/GitHub
- contributions are reviewed & thoroughly tested prior to inclusion
- currently experimental: `eb --new-pr`, `eb --update-pr`
- see EasyBuild wiki for detailed walkthrough & guidelines:

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

<https://github.com/hpcugent/easybuild/wiki/Policy-for-easyconfig-pull-requests>

# Advanced features

- upload test reports for GitHub pull requests: `eb --from-pr <id> -u`
- support for custom module naming schemes
  - make EasyBuild spit out module files following your site policy
  - also supports *hierarchical* module naming schemes
- plug in your own easyblocks, compiler toolchain definitions, etc.
  - `--include-easyblocks`, `--include-toolchains`, ...
- stable support for installing software on Cray systems
  - on top of Cray-provided modules (or not)
  - currently supports `PrgEnv-gnu`, `PrgEnv-intel`, `PrgEnv-cray`
  - sensible way to align software stacks across Cray systems/sites
- the EasyBuild community ...