EAPI Cheat Sheet

Christian Faulhammer
fauli@gentoo.org

Ulrich Müller
ulm@gentoo.org

Version 6.0
16th April 2017

Abstract

An overview of the main EAPI changes in Gentoo, for ebuild authors. For full details, consult the Package Manager Specification found on the project page;[1] this is an incomplete summary only.

Official Gentoo EAPIs are consecutively numbered integers (0, 1, 2, . . .). Except where otherwise noted, an EAPI is the same as the previous EAPI. All labels refer to the PMS document itself, built from the same checkout as this overview.

EAPIs 0, 1, and 2

Omitted for lack of space. See version 5.0 of this document for differences between these previous EAPIs.

EAPI 3 (2010-01-18)

Additions/Changes

Support for `.xz` Unpack of `.xz` and `.tar.xz` files is possible without any custom `src_unpack` functions. See UNPACK-EXTENSIONS on page 67.

Offset prefix Supporting installation on Prefix-enabled systems will be easier with this EAPI.

[1] https://wiki.gentoo.org/wiki/Project:Package_Manager_Specification

[2] http://creativecommons.org/licenses/by-sa/3.0/

EAPI 4 (2011-01-17)

Additions/Changes

`pkg_pretend` Some useful checks (kernel options for example) can be placed in this new phase to inform the user early (when just pretending to emerge the package). Most checks should usually be repeated in `pkg_setup`. See PKG-PRETEND on page 38.

`src_install` The `src_install` phase is no longer empty but has a default now. This comes along with an accompanying `default` function. See SRC-INSTALL-4 on page 42.

`pkg_info` on non-installed packages The `pkg_info` phase can be called even for non-installed packages. Be warned that dependencies might not have been installed at execution time. See PKG-INFO on page 43.

`econf` changes The helper function now always activates `--disable-dependency-tracking`. See ECONF-OPTIONS on page 59.

USE dependency defaults In addition to the features offered in EAPI 2 for USE dependencies, a `(+)` or `(-)` can be added after a USE flag (mind the parentheses). The former specifies that flags not in IUSE should be treated as enabled; the latter, disabled. Cannot be used with USE_EXPAND flags. This mimics parts of the behaviour of `--missing` in `built_with_use`. See USE-DEP-DEFAULTS on page 36.

Controllable compression All items in the `doc`, `info`, `man` subdirectories of `/usr/share/` may be compressed on-disk after `src_install`, except for `/usr/share/doc/${PF}/html`. `docompress path ...` adds paths to the inclusion list for compression. `docompress -x path ...` adds paths to the exclusion list. See DOCOMPRESS on page 64.

`nonfatal` for commands If you call `nonfatal` the command given as argument will not abort the build process in case of a failure (as is the default) but will return non-zero on failure. See NONFATAL on page 56.

`dodoc` recursion If the `-r` switch is given as first argument and followed by directories, files from there are installed recursively. See DODOC on page 61.

`doins` symlink support Symbolic links are now properly installed when using recursion (`-r` switch). See DOINS on page 62.

`PROPERTIES` Is mandatory for all package managers now to support interactive installs.

`REQUIRED_USE` This variable can be used similar to the `(R|P)DEPEND` variables and define sets of USE flag combinations that are not allowed. All elements can be further nested to achieve more functionality.

- **Illegal combination** To prevent activation of `flag1` if `flag2` is enabled use "`flag2? ( !flag1 )`".
- **OR** If at least one USE flag out of many must be activated on `flag1` use "`flag1? ( || ( flag2 flag3 ... ) )`".
- **XOR** To allow exactly one USE flag out of many use "`^^( flag1 flag2 ... )`".

See REQUIRED-USE on page 29.

`MERGE_TYPE` This variable contains one of three possible values to allow checks if it is normal merge with compilation and installation (`source`), installation of a binary package (`binary`), or a compilation without installation (`buildonly`). See MERGE-TYPE on page 51.

`REPLACING_VERSIONS`, `REPLACED_BY_VERSION` These variables, valid in `pkg_*`, contain a list of all versions (`PVR`) of this package that we are replacing, and the version that is replacing the current one, respectively. See REPLACE-VERSION-VARS on page 53.

Removals/Bans

`dohard`, `dosed` Both functions are not allowed any more. See BANNED-COMMANDS on page 56.

No `RDEPEND` fall-back The package manager will not fall back to `RDEPEND=DEPEND` if `RDEPEND` is undefined. See RDEPEND-DEPEND on page 30.

`S` fallback changes The value of the variable `S` will not automatically be changed to `WORKDIR`, if `S` is not a directory, but abort. Virtual packages are the only exception. See S-WORKDIR-FALLBACK on page 38.

`AA`, `KV` These variables are not defined any more. See AA on page 49 and KV on page 51.

EAPI 5 (2012-09-20)

Additions/Changes

Sub-slots The `SLOT` variable and slot dependencies may contain an optional sub-slot part that follows the regular slot, delimited by a `/` character; for example `2/2.30`. The sub-slot is used to represent cases in which an upgrade to a new version of a package with a different sub-slot may require dependent packages to be rebuilt. If the sub-slot is not specified in `SLOT`, it defaults to the regular slot. See SUB-SLOT on page 36.

Slot operator dependencies Package dependencies can specify one of the following operators as a suffix, which will affect updates of runtime dependencies:

`:*` Any slot value is acceptable. The package will not break when its dependency is updated.

`:=` Any slot value is acceptable, but the package can break when its dependency is updated to a different slot (or sub-slot).

See SLOT-OPERATOR-DEPS on page 36.

Profile `IUSE` injection Apart from the USE flags explicitly listed in `IUSE`, additional flags can be implicitly provided by profiles. See PROFILE-IUSE-INJECT on page 53.

At-most-one-of groups In `REQUIRED_USE` you can use "`?? ( flag1 flag2 ... )`" to allow zero or one USE flag out of many. See AT-MOST-ONE-OF on page 33.

Parallel tests The default for `src_test` runs `emake` without `-j1` now. See PARALLEL-TESTS on page 41.

`econf` changes The `econf` function now always passes `--disable-silent-rules` to `configure`. See ECONF-OPTIONS on page 59.

`has_version` and `best_version` changes The two helpers support a `--host-root` option that causes the query to apply to the host root instead of `ROOT`. See HOST-ROOT-OPTION on page 57.

`usex` Usage for this helper function is `usex` *<USE flag> [true1] [false1] [true2] [false2]*. If the USE flag is set, outputs *[true1][true2]* (defaults to `yes`), otherwise outputs *[false1][false2]* (defaults to `no`). See USEX on page 66.

doheader and newheader These new helper functions install the given header file(s) into /usr/include. The -r option enables recursion for doheader, similar to doins. See DOHEADER on page 61.

new* standard input The newins etc. commands read from standard input if the first argument is - (a hyphen). See NEWFOO-STDIN on page 63.

EBUILD_PHASE_FUNC This variable is very similar to EBUILD_PHASE, but contains the name of the current ebuild function. See EBUILD-PHASE-FUNC on page 51.

Stable use masking/forcing New files use.stable.{mask,force} and package.use.stable.{mask,force} are supported in profile directories. They are similar to their non-stable counterparts, but act only on packages that would be merged due to a stable keyword. See STABLEMASK on page 23.

EAPI 6 (2015-11-13)

Additions/Changes

Bash version Ebuilds can use features of Bash version 4.2 (was 3.2 before). See BASH-VERSION on page 27.

failglob The failglob option of Bash is set in global scope, so that unintentional pattern expansion will be caught as an error. See FAILGLOB on page 55.

Locale settings It is ensured that the behaviour of case modification and collation order for ASCII characters (LC_CTYPE and LC_COLLATE) are the same as in the POSIX locale. See LOCALE-SETTINGS on page 52.

src_prepare This phase function has a default now, which applies patches from the PATCHES variable with the new eapply command, and user-provided patches with eapply_user. See SRC-PREPARE-6 on page 40.

src_install The default implementation uses the new einstalldocs function for installing documentation. See SRC-INSTALL-6 on page 42.

nonfatal die When die or assert are called under the nonfatal command and with the -n option, they will not abort the build process but return with an error. See NONFATAL-DIE on page 57.

`unpack` changes `unpack` has been extended:

- **Pathnames** Both absolute paths and paths relative to the working directory are accepted as arguments. See UNPACK-ABSOLUTE on page 67.
- **`.txz` files** Suffix `.txz` for xz compressed tarballs is recognised. See UNPACK-EXTENSIONS on page 67.
- **Filename case** Character case of filename extensions is ignored. See UNPACK-IGNORE-CASE on page 67.

`econf` changes Options `--docdir` and `--htmldir` are passed to `configure`, in addition to the existing options. See ECONF-OPTIONS on page 59.

`eapply` The `eapply` command is a simplified substitute for `epatch`, implemented in the package manager. The patches from its file or directory arguments are applied using `patch -p1`. See EAPPLY on page 58.

`eapply_user` The `eapply_user` command permits the package manager to apply user-provided patches. It must be called from every `src_prepare` function. See EAPPLY-USER on page 58.

`einstalldocs` The `einstalldocs` function will install the files specified by the `DOCS` variable (or a default set of files if `DOCS` is unset) and by the `HTML_DOCS` variable. See EINSTALLDOCS on page 68.

`in_iuse` The `in_iuse` function returns true if the USE flag given as its argument is available in the ebuild for USE queries. See IN-IUSE on page 66.

`get_libdir` The `get_libdir` command outputs the `lib*` directory basename suitable for the current ABI. See GET-LIBDIR on page 68.

Removals/Bans

`einstall` No longer allowed. Use `emake install` as replacement. See BANNED-COMMANDS on page 56.