PORT(1) BSD General Commands Manual PORT(1)

NAME

port -- operate on individual or multiple Mac ports

SYNOPSIS

port [-vdqfonRusbckixpt] [-D portdir] [-F cmdfile] [action] [actionflags]
[[portname | pseudo-portname | port-url]]
[[@version] [+/-variant ...] ... [option=value ...]]

DESCRIPTION

port is designed to operate on individual or multiple Mac ports, optionally within a single call, based on the requested action. If no portdir or portname is specified, the current working directory is assumed; if no action is specified the port command enters interactive mode, in which commands are read via stdin. Batch commands may be passed via a cmdfile. Port options are passed as key=value pairs and take precedence over individual portname options as specified in its Portfile and system-wide settings.

Port variants can specified as +name, which indicates the variant is desired, or -name, indicating the contrary. In case of ambiguities, a port can be fully specified with the @version_revision+variants format.

Installed ports can be activated or deactivated without being uninstalled. A port can be installed if all other version/variant(s) combinations installed at any given moment are deactivated.

The port command knows various pseudo-portnames that will expand to the specified set of ports from the available ports tree(s). These may be used in place of a portname. Common options are:

- all: all the ports in each ports tree listed in sources.conf
- current: the port in the current working directory.
- active: set of installed and active ports.
- inactive: set of installed but inactive ports.
- installed: set of all installed ports.
- uninstalled: ports in the ports tree(s) that aren't installed.
- outdated: installed ports that are out of date with respect to their current version/revision in the ports tree(s)

Other options, also known as pseudo-portname selectors, matching the most common keys used in a Portfile are also accepted, in both singular and plural form where applicable. These are:

- name
- version
- revision
- epoch
- variant
- variants
- category
- categories
- maintainer
- maintainers
- platform
- platforms
- description
- long_description
- homepage
- portdir

Search strings that will expand to a set of matching ports can be constructed based on the "pseudo-portname selector":regex combination used. portnames containing valid

UNIX glob patterns will also expand to the set of matching ports. Any action passed to port will be invoked on each of them. For example:

```
port list variant:no_ssl
port uninstall name:sql
port echo apache*
```

Logical operators "and", "or", "not", "!", "(" and ")" may be used to combine individual portnames, port glob patterns and/or pseudo-portnames to construct complex port expressions that expand to the set of matching ports. For example:

```
port upgrade installed and apache*
port echo maintainer:jberry and uninstalled and \( category:java and not
commons* \)
```

The port command also recognizes several command line flags and targets:

OPTIONS

-v verbose mode (generate verbose messages)

-d debug mode (generate debugging messages, implies -v)

-q quiet mode (suppress messages)

-n don't follow dependencies in upgrade (only for upgrading)

-R also upgrade dependents (only for upgrading)

-u uninstall non-active ports when upgrading and uninstalling

-f force mode (ignore state file)

-o honor state files older than Portfile

-s source-only mode (build and install from source, ignore all binary archives, do not create/recreate binary archives) (only applies when archive mode is enabled)

-b binary-only mode (build and install from binary archives, ignore source, abort if no archive present; do not create/recreate binary archives from source) (only applies when archive mode is enabled)

-c autoclean mode (execute clean after install)

-k keep mode (don't autoclean after install)

-D specify portdir

-F Read and process the file of commands specified by the argument. If the argument is '-', then read commands from stdin. If the option is given multiple times, then multiple files will be read.

-i Read commands from stdin. Short for -F -

-x In batch and interactive mode, exit on the first error encountered. Otherwise, errors during batch execution are simply reported.

-p Despite any errors encountered, proceed to process multiple ports and commands.

-t enable trace mode debug facilities on platforms that support it (Mac OS X). This feature is two-folded. It consists in automatically detecting and reporting undeclared dependencies based on what files the port reads or what programs the port executes. In verbose mode, it will also report unused dependencies for each stage of the port installation. It also consists in forbidding and reporting file creation and file writes outside allowed directories (temporary directories and ${workpath}).

USER TARGETS

Targets most commonly used by regular MacPorts users are:

search
Search for an available port whose name matches a regular expression. For example:

```
port search vim
```

info
Displays all the meta-information available for portname. Specific meta-information may be requested through an option such as --maintainer or --category (recognized field names are those from the PortIndex). If the global option -q is in effect, the meta-info fields will not be labeled. If the option --line is provided, all such data will be consolidated into a single line per port, suitable for processing in a pipe of commands. If the option --index is provided, the information will be pulled from the PortIndex rather than from the Portfile (in this case variant information, such as dependencies, will not affect the output).

For example:

```
port info vim +ruby
port info --category --name apache*
port -q info --category --name --version category:java
port info --line --category --name all
port info --index python24
```

variants
Lists the build variants available for portname.

deps
Lists the other ports that are required to build and run portname.

dependents
Lists the installed ports that depend on the port portname.

install
Install and activate portname.

uninstall
Deactivate and uninstall portname. To uninstall all installed but inactive ports, use -u. To recursively uninstall all dependents of this port, use --follow-dependents.

For example:

```
port uninstall vim
port -u uninstall
port uninstall --follow-dependents python24
```

activate
Activate the installed portname.

deactivate
Deactivate the installed portname.

installed
List all installed ports.

location
Print the install location of a given port.

contents
Lists the files installed by portname.

provides
Determines which port owns a given file and can take either a relative or absolute path. For example:

```
port provides /opt/local/etc/irssi.conf
port provides include/tiff.h
```

sync

Performs a sync operation only on the ports tree of a MacPorts installation, pulling in the latest revision available of the Portfiles from the MacPorts rsync server. To update you would normally do:

```
sudo port -d sync
```

If any of the ports tree(s) uses a file: URL that points to a local subversion working copy, sync will perform an svn update on the working copy with the user set to the owner of the working copy.

outdated
List the installed ports that need upgrading.

upgrade
The upgrade target works on a port and its dependencies. If you want to change this behaviour, look at the switches for n (no dependencies) and R (dependents) below.

Upgrade the installed portname. For example:

```
port upgrade vim
```

To upgrade all installed ports:

```
port upgrade installed
```

To upgrade portname and the ports that depend on it:

```
port -R upgrade libiconv
```

To force an upgrade (rebuild) use:

```
port -f upgrade vim
```

To upgrade portname without following its dependencies, use -n. For example:

```
port -n upgrade wireshark
```

clean
Clean the files used for building portname. To just remove the work files, use the --work actionflag. To remove the distribution files (tarballs, etc), specify --dist. To remove the archive(s) for the current version of a port, pass --archive. To remove the work files, distribution files and archives, pass --all. For example:

```
port clean --dist vim
port clean --archive vim
```

To remove only certain version(s) of a port's archives (version is any valid UNIX glob pattern), you can use:

```
port clean --archive vim 6.2.114
```

or:

```
port clean --archive vim '6.*'
```

echo
Writes to stdout the arguments passed to port. This follows the expansion of pseudo-portnames, portname glob patterns, pseudo-portname selectors and the evaluation of port expressions. echo may be used to determine the exact set of ports to which a given string of arguments will expand, without performing any further operations on them. For example:

```
port echo category:net
port echo maintainer:jmpp and name:netw
port echo maintainer:jmpp and \( net* or category:text \)
```

list
If no argument is given, display a list of the latest version of all available ports. If portname(s) are given as arguments, display a list of the latest version of each

port.

version
Display the release number of the installed MacPorts infrastructure.

platform
Display the platform information for the current system.

selfupdate
Updates the MacPorts system, ports tree(s) and base tools if needed, from the MacPorts rsync server, installing the newest infrastructure available. To update you would typically do:

```
sudo port -d selfupdate
```

See sync for more information about updating ports tree(s).

help
Displays a summary of all available actions and port command syntax on stdout.

DEVELOPER TARGETS

The targets that are often used by Port developers are intended to provide access to the different phases of a Port's build process:

dir
Displays the path to the directory containing portname.

file
Displays the path to the Portfile for portname.

cat
Concatenates and prints the contents of Portfile on stdout.

edit
Opens Portfile with your default editor specified in your shell's environment variable. Alias ed also invokes this command.

You can also use the --editor flag on the command line to specify an alternative editor. For example:

```
port edit --editor nano apache2
```

unarchive
Unpack the port from a pre-built binary archive. When archive mode is enabled, this command is called automatically, prior to fetch, to check for an existing binary archive to unpack. If found, it is unpacked and all stages up to install are then skipped.

fetch
Fetches the distribution files required to build portname.

checksum
Compute the checksums of the distribution files for portname, and compare them to the checksums listed in Portfile.

extract
Extracts the distribution files for portname.

patch
Applies any required patches to portname's extracted distribution files.

configure
Runs any configure process for portname.

build
Build portname.

destroot
Installs portname to a temporary directory.

test
Tests portname.

lint
Verifies Portfile for portname. To nitpick about whitespace and patchfile names, use --nitpick.

archive
Archive the port for a later unarchive. When archive mode is enabled, binary archives will be created automatically whenever an install is performed, or when the archive target is called explicitly.

distcheck
Check if the distfiles haven't changed and can be fetched.

distfiles
Display each distfile, its checksums, and the URLs used to fetch it.

livecheck
Check if the software hasn't been updated since the Portfile was last modified.

PACKAGING TARGETS

There are also targets for producing installable packages of ports:

pkg
Creates an OS X installer package of portname.

mpkg
Creates an OS X installer metapackage of portname and its dependencies.

dmg
Creates an internet-enabled disk image containing an OS X package of portname.

mdmg
Creates an internet-enabled disk image containing an OS X metapackage of portname and its dependencies.

rpm
Creates an RPM binary package of portname, similar to a tgz "archive".

srpm
Creates a SRPM source package of portname, similar to a xar "portpkg".

dpkg
Creates a DEB binary package of portname.

EXAMPLES

The following demonstrates invoking port with the extract target on portdir "textproc/figlet" and extract.suffix set to ".tgz":

```
port extract -D textproc/figlet extract.suffix=.tgz
```

FILES

${prefix}/etc/macports/macports.conf
Global configuration file for the MacPorts system.

${prefix}/etc/macports/sources.conf
Global listing of the ports trees used by MacPorts. This file also enables rsync synchronization.

${prefix}/etc/macports/variants.conf
Global variants used when a port is installed.

~/.macports/macports.conf
User configuration file for the MacPorts system. It overrides the global macports.conf file.

DIAGNOSTICS

The port utility exits 0 on success, and >0 if an error occurs.

SEE ALSO

macports.conf(5), portfile(7), portgroup(7), portstyle(7), porthier(7)

AUTHORS

Landon Fuller <landonf@macports.org>
James Berry <jberry@macports.org>
Jordan K. Hubbard <jkh@macports.org>
Juan Manuel Palacios <jmpp@macports.org>
Kevin Van Vechten <kevin@opendarwin.org>
Ole Guldberg Jensen <olegb@opendarwin.org>
Robert Shaw <rshaw@opendarwin.org>
Chris Ridd <cjr@opendarwin.org>
Matt Anton <matt@opendarwin.org>
Joe Auty <joe@opendarwin.org>