Layman - Gentoo Wiki

Layman is a ebuild repository management tool. It offers a single command-line interface to repository management for end users.

Features

With app-portage/layman users can manage their ebuild repositories (overlays) in a simple, centralized manner. The layman provides an overview of available remote repositories and allows the user to select one or more for the system. Once selected, the user can update (similar to emerge --sync), add, remove, display and information about the overlays.

Versions greater than 2.1.0 are improved with a plug-in sync system.

Note
Although not all features of **plug-in sync system** of Portage are supported yet from layman the migration of Portage is a good step to be prepared for that.

Installation

USE flags

`bazaar`	Support dev-vcs/bzr based overlays
`cvs`	Support dev-vcs/cvs based overlays
`darcs`	Support dev-vcs/darcs based overlays
`g-sorcery`	Support app-portage/g-sorcery based overlays
`git`	Support dev-vcs/git based overlays

gpg	Support app-crypt/gnupg signed overlays lists and manifests
mercurial	Support dev-vcs/mercurial based overlays
sqlite	Add support for sqlite - embedded sql database
squashfs	Support mounting squashfs image overlays locally read-only
subversion	Support dev-vcs/subversion based overlays
sync-plugin-portage	Install the sys-apps/portage sync module
test	Enable dependencies and/or preparations necessary to run tests (usually controlled by FEATURES=test but can be toggled independently)

The `sync-plugin-portage` and `git` USE flags are especially important in newer versions of layman.

FILE **`/etc/portage/package.use/layman`Add important USE flags**

```
app-portage/layman sync-plugin-portage git
```

Please refer to portage projects page.

Emerge

Next install the layman package:

```
root #emerge --ask app-portage/layman
```

Configuration

There are two methods of integrating layman into Portage. Newer versions support both methods at the same time, so there is no need to configure Portage (except configurations mentioned in this section).

repos.conf method (version 2.1.0 or later)

Layman will create its configuration file in /etc/portage/repos.conf/.

Configure layman to use the repos.conf method in /etc/layman/layman.cfg. New installations of layman will probably have this already set correctly:

FILE **`/etc/layman/layman.cfg`**

```
# Repository config types used by layman
# (repos.conf, make.conf)
conf_type : repos.conf
```

Create the /etc/portage/repos.conf/ directory if it does not exist yet:

If you have layman version 2.3.0 or greater installed, you can force a rebuild of layman's repos.conf files:

make.conf method (versions prior to 2.1.0)

This is the older method, but newer versions of layman still support this.

Configure layman to use the make.conf method in /etc/layman/layman.cfg:

FILE **`/etc/layman/layman.cfg`**

```
# Repository config types used by layman
# (repos.conf, make.conf)
conf_type : make.conf
```

Insert a reference to layman in /etc/portage/make.conf:

```
root #echo "source /var/lib/layman/make.conf" >> /etc/portage/make.conf
```

Adding the source command to the /etc/portage/make.conf (performed in the command above) will ensure that Portage, when asked, will check the content of the various repositories managed by layman in the /var/lib/layman/make.conf file. In effect, it will update the *PORTDIR_OVERLAY* variable with directories layman uses. If a special directory has been previously defined for *PORTDIR_OVERLAY* in /etc/portage/make.conf, make sure its value is not overwritten with the value layman provides.

FILE **`/etc/portage/make.conf`Ensuring that layman's PORTDIR_OVERLAY is not overwritten**

```
source /var/lib/layman/make.conf
#for some local ebuilds to test, have to be after line for layman above!
PORTDIR_OVERLAY="/usr/local/portage/ ${PORTDIR_OVERLAY}"
```

Usage

Basic invocations

The layman man page (see External resources) provides a full overview of the available functions. However, for most users, the following commands suffice for repository management activities.

To fetch and display a list of all the repositories available through official references:

To add a repository in the list generated by the local list:

To add an unofficial repository:

```
root #layman -o <url of repository xml file> -f -a <name>
```

To remove a repository from the local list:

To update a specific repository:

To update all repositories:

Mountable repositories with layman-mounter

Since the release of layman version 2.2.0, support for squashfs repository types has been included. layman will interact with squashfs repository by mounting them as read-only on the filesystem. On the initial install of the squashfs repository, it will be mounted as read-only. However, after a reboot the repository will no longer be mounted and the ebuilds in that repository will not be accessible by the system.

In order to assist users in handling these mountable repositories, a utility was added that goes by the name of layman-mounter.

To find all repositories that are currently mounted, type:

To find all repositories that are installed by layman that can be mounted, type:

To mount the mountable repositories, type:

```
root #layman-mounter -m <name>
```

To unmount the repositories, type:

```
root #layman-mounter -u <name>
```

Setting repository priorities with Layman

The information in this section has been **deprecated**. It

may or may not be relevant

for contemporary usage. Handle with care!

As each ebuild repository is assigned a unique priority, layman provides a simple way of defining priorities for repositories it manages. For more information about repository priorities see the ebuild repository priorities.

The file /var/lib/layman/installed.xml contains some information about the repositories, among which is the priority attribute in the repo tag. The number there determines only the priority relative to the other repository entries, 50 is the default value. Larger numbers take priority over smaller numbers. Layman then analyses this file and sets the order of the repository entries in the *PORTDIR_OVERLAY* variable defined in /var/lib/layman/make.conf.

As the file /var/lib/layman/make.conf is automatically generated by layman based on the settings in /var/lib/layman/installed.xml, it is strongly recommended that only /var/lib/layman/installed.xml is used to set the priorities.

To add a personal repository, and to ensure that the repository has a higher priority, add the repository *before* /var/lib/layman /make.conf is sourced.

FILE **`/var/lib/layman/make.conf`Example layman overlays setting**

```
PORTDIR_OVERLAY="
/home/jdoe/gamerlay
/var/lib/layman/lisp
/var/lib/layman/Spring
${PORTDIR_OVERLAY}" #the variable defined in /etc/portage/make.conf is now expanded
                    #when /var/lib/layman/make.conf is sourced in /etc/portage/make.conf
```

However, this can be also "fooled" by defining the *PORTDIR_OVERLAY* in /etc/portage/make.conf *after* /var/lib/layman/make.conf has been sourced.

FILE **`/etc/portage/make.conf`Custom repository setting**

```
source /var/lib/layman/make.conf #this sources the PORTDIR_OVERLAY variable defined by layman.
                                 #however, the variable expanded by layman was empty
PORTDIR_OVERLAY="/home/user/overlay ${PORTDIR} ${PORTDIR_OVERLAY}" #now the layman defined repositories take precedence,
                                                                   #but the user defined repository still has the lowest priorit
```

This "trick" is merely an opportunity offered by shell variable expansion.

Adding custom repositories

To add repositories which are not listed when layman -L is ran, find their repository XML files and add them using the `-o` option under a name specified by the `-a` option.

Example: repositories.xml in brother-overlay

Missing repository.xml file

In some cases the custom repository does not provide a repository XML file.

Creating repository XML file manually

The XML file can be created manually in the /etc/layman/overlays folder.

For example, if Larry the cow were to create his repository:

FILE **`/etc/layman/overlays/larry.xml`** **Larry the cow's nginx overlay**

```
<?xml version="1.0" ?>

<repositories version="1.0">
        <repo priority="50" quality="experimental" status="unofficial">
                <name>larry</name>
                <description>nginx server for the barn computer from Larry the cow.</description>
                <homepage>https://github.com/gentoo/nginx-overlay</homepage>
                <owner>
                        <email>larry@gentoo.org</email>
                </owner>
                <source type="git">https://github.com/gentoo/nginx-overlay.git</source>
        </repo>
</repositories>
```

Using layman-overlay-maker utility

With the addition of layman version 2.2.0 a new utility was added to assist users in this process that goes by the name of layman-overlay-maker. As long as the overlay information has been properly added via the prompts, layman-overlay-maker will create a XML defined overlay and save into /etc/layman/overlays or the specified in the layman configuration file for *overlay_defs*.

layman-overlay-maker can become a useful tool in assisting users who would like to submit a patch to have their overlays added to the official repositories.xml file.

To use the utility simply invoke it by name:

```
root #layman-overlay-maker
```

and go through its prompts until completion.

Enabling the repository

When finished rebuild the repos.conf using layman-updater:

Now you can add the custom repository by:

where `name` is the name of the repository that was created.

See also

- Eselect/Repository — an eselect module for manipulating /etc/portage/repos.conf entries.
- Overlay user guide
- Ebuild repository — a structure of directories and files used to add and extend packages for a Gentoo-based system's package manager.

External resources

- The Layman man page locally (man layman) or online at Sourceforge.net