eMerger Documentation

Roberto Antoniello & Edoardo Ferrari

November 24, 2023

In this file we will put a complete documentation of eMerger. So you can read simply how it really works without waste too much time reading only the code.

1 Introduction

We started this project for fun without knowing a bit of Bash, but then we have continued to release new features and improve the main script, also learning new stuffs during the development.
The project started with the name of *Updater*, then we changed to *eMerger*(*blink Gentoo dudes).
Probably you already know if you're reading this but let's repeat what is eMerger.
eMerger is a script that allows users to do a clean update for Linux based system with a single command.

2 Installation

The only thing you need to do to install eMerger is to launch *setup.sh*.
When you launch it for the first time, it just put an alias on your /.bashrc named *up* pointed on the source file *emerger.sh*.
After the alias is added correctly or the alias already exists, it will launch an integrity test by calling the source file *integrity-check.sh*.
During the installation a little cache is also created so eMerger will always remember which system you're using without fetching this data every time.

2.1 Integrity test

In this script there's a check of the existence and stability of the content of source files needed for eMerger correct execution. If it's all ok it continues, otherwise the current operation is aborted.

2.2 Uninstall

The script *uninstall.sh* just removes the alias created during the installation and delete every file related to eMerger.

3 Functioning

At this moment, the main source file is *emerger.sh*.
First of all, eMerger starts checking for subcommand(you can view the list by typing *up -help*). if it's a subcommand that offer an extra feature, eMerger executes it and return to shell. Otherwise it starts with the real execution by do these steps:

- It generates the cache of system information or it fetch the exising one.

  Essentially if it's the first time, it creates a cache to remember the system the next time. Otherwise it fetch the data from the existing cache and it already start with the correct package manager based on the system we are launching eMerger on.

- update packages repository.
- upgrade packages.
- autoremove not necessary packages.
- update/upgrade of external packages manager such as snap and flatpak if they are used by the user on his system.
- clean the trashed.

3.1 Updating eMerger

Obviously there's a way to update the Updater(*yes, we like to joke...)*. Our basic but functional way is to execute a git pull from the main branch of the project, you can do that by typing *up -up*.

4 OS supported

The list of OS supported is always growing to make it to offer the power of eMerger to as many systems as possible and many people as possible. Our goal is to give an alternative to update the system with a single command without remember every time all the commands are needed for a clean update.

- Arch Linux
- Debian
- EndeavourOS
- Fedora Project
- Kali Linux
- Manjaro
- Raspbian
- Termux
- Ubuntu

5 Package manager supported

And now we list the package managers, not only those which are connected to a os but also those which are dedicated to no-free app external to the main pkg manager of the os.

- rpm
- apt-get
- apt
- pacman
- emerge(*I know you already got it about our project name...*)
- flatpak
- nixos
- zypper
- snap
- pkg

6 Source files organization

The organization and how the single source files works is not really clearly. In this chapter we try to describe the most useful one to help understand better the functioning of eMerger.

6.1 utils folder

Inside the **utils** folder we can find some source Bash files which helps not to repeat lines of code in the project. The most useful is *global.sh* which saves in variables the ascii code of emojis for every package manager and the definition of *puts* function which is a custom of the printf with different colours based of which type of string we have to show on terminal.

Another source file we can describe a little bit is *cache-gen.sh* which is the one that generates the cache file the first time we try to launch eMerger.

- It saves the terminal the user is using.(such as xfce terminal or terminator).
- It saves the package manager the system is using.
- It saves all of these information inside a file.
- In emerger.sh this file will be hashed with md5.

6.2 package folder

Inside this folder there are all the source files for every package manager with their respective commands and flow of actions.

7 use cases

Here we list the amount of use cases eMerger have in this project.

- The user install eMerger by launching *setup.sh*
- The user uninstall eMerger by launching *uninstall.sh*
- The user launches a normal update by typing *up*
- The user launches an update but he would like to knoe the weather by typing *up -w*
- The user updates eMerger by typing *up -up*
- There's an error, then the user can revel it by typing *up -err*

There are a few more cases but here we shared the most commons one.
If you still have some questions, you can explore our code on the repository or ask directly to us by creating a new issue!