Fayssal Elaazouzi

Mar 23 · 5 min read · Listen

Save

Day 7: Understanding package manager and systemctl 👍

This is #90DaysofDevops challenge under the guidance of Shubham Londhe sir.

Day 7 TASK

check this for task:

90DaysOfDevOps/tasks.md at master · LondheShubham153/90DaysOfDevOps

This repository is a Challenge for the DevOps Community to get stronger in DevOps. This challenge starts on the 1st…

github.com

1. What is a package manager in Linux?

In simpler words, a package manager is a tool that allows users to install, remove, upgrade, configure and manage software packages on an operating system. The package manager can be a graphical application like a software center or a command line tool like apt-get or pacman.

2. What is a package?

A package is usually referred to an application but it could be a GUI application, command line tool or a software l other software programs). A package is essentially an archive file containing the binary executable, configuration file and sometimes information about the dependencies.

3. Different Package Managers in Linux

There are several package managers available in Linux, each with its own strengths and weaknesses. Some of the most popular package managers include:

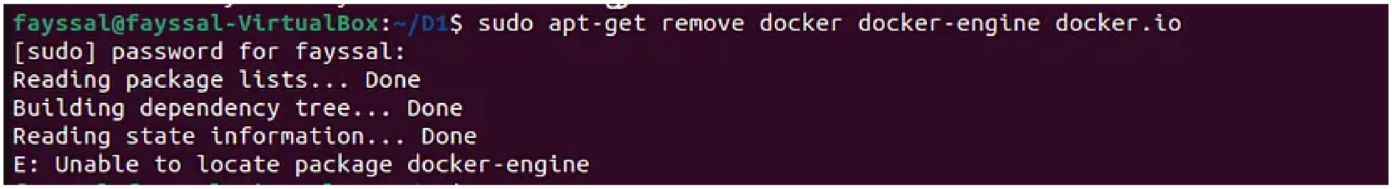
1. **APT:** Advanced Package Tool (APT) is a package manager used by Debian, Ubuntu, and other Debian-based distributions.
2. **YUM:** Yellowdog Updater, Modified (YUM) is a package manager used by Red Hat, CentOS, and other Red Hat-based distributions.
3. **Pacman**: Pacman is a package manager used by Arch Linux and its derivatives.
4. **Zypper**: Zypper is a package manager used by SUSE Linux and its derivatives.

Task 1 : You have to install docker in your system from your terminal using package managers

Docker and Jenkins are two popular tools that are widely used for building and deploying applications. In this tutorial, we will show you how to install these tools on Ubuntu using the package manager.

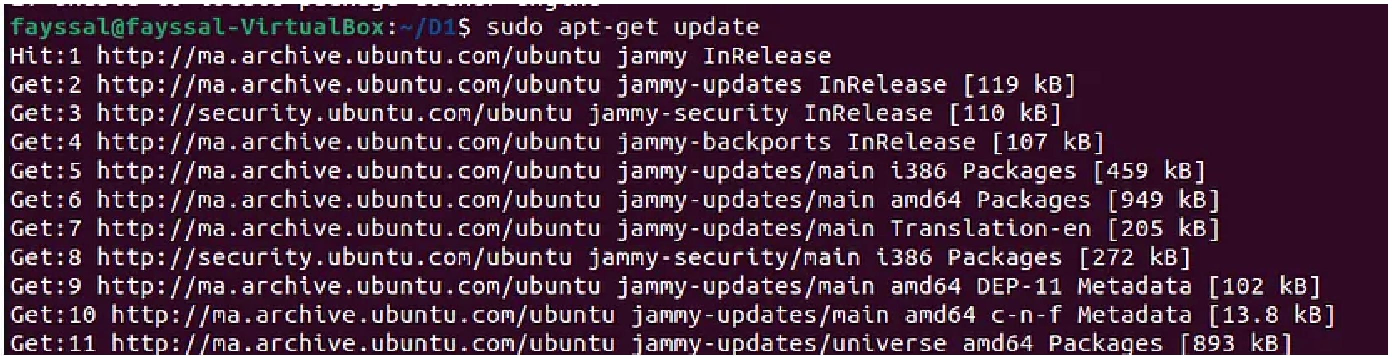
To install Docker on Ubuntu, follow these steps:

1. Remove any Docker files that are running in the system, using the following command:



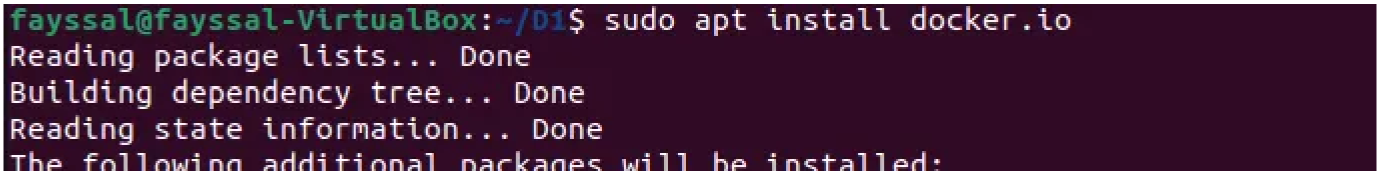
```
fayssal@fayssal-VirtualBox:~/D1$ sudo apt-get remove docker docker-engine docker.io
[sudo] password for fayssal:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package docker-engine
```

2. Check if the system is up-to-date using the following command:

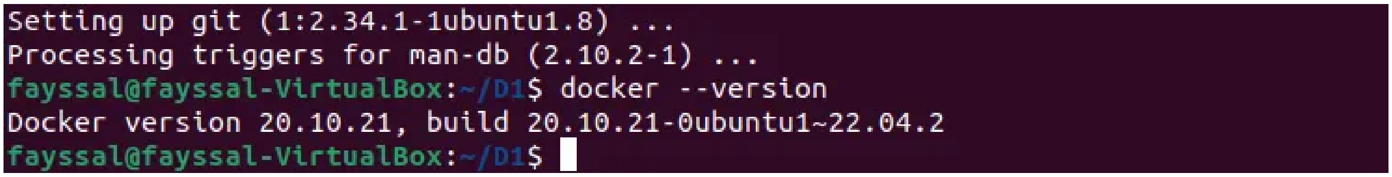


```
fayssal@fayssal-VirtualBox:~/D1$ sudo apt-get update
Hit:1 http://ma.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://ma.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:4 http://ma.archive.ubuntu.com/ubuntu jammy-backports InRelease [107 kB]
Get:5 http://ma.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [459 kB]
Get:6 http://ma.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [949 kB]
Get:7 http://ma.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [205 kB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [272 kB]
Get:9 http://ma.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [102 kB]
Get:10 http://ma.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [13.8 kB]
Get:11 http://ma.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [893 kB]
```

3. Install Docker using the following command:

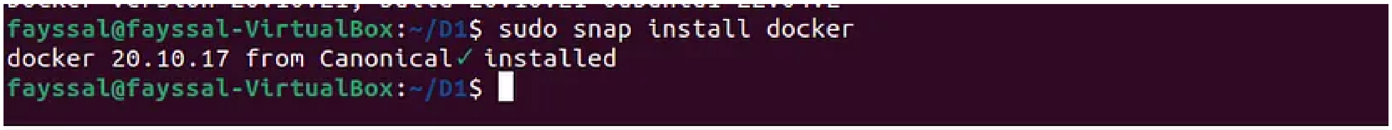


```
fayssal@fayssal-VirtualBox:~/D1$ sudo apt install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
```



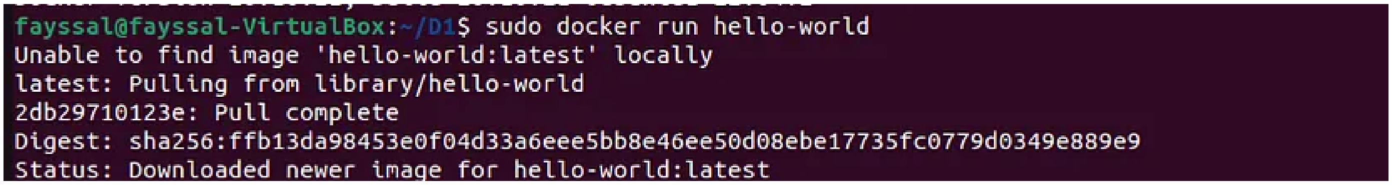
```
Setting up git (1:2.34.1-1ubuntu1.8) ...
Processing triggers for man-db (2.10.2-1) ...
fayssal@fayssal-VirtualBox:~/D1$ docker --version
Docker version 20.10.21, build 20.10.21-0ubuntu1~22.04.2
fayssal@fayssal-VirtualBox:~/D1$
```

4. Install all the dependency packages using the following command:



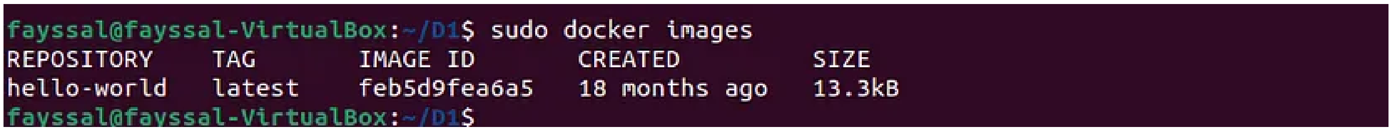
```
fayssal@fayssal-VirtualBox:~/D1$ sudo snap install docker
docker 20.10.17 from Canonical✓ installed
fayssal@fayssal-VirtualBox:~/D1$
```

5 . Pull an image from the Docker hub using the following command:



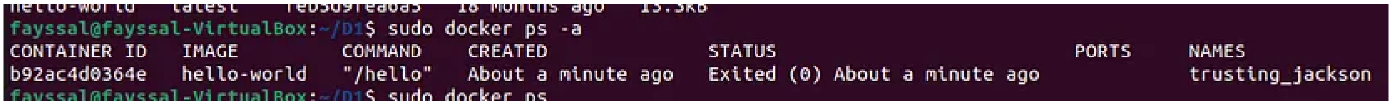
```
fayssal@fayssal-VirtualBox:~/D1$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:ffb13da98453e0f04d33a6eee5bb8e46ee50d08ebe17735fc0779d0349e889e9
Status: Downloaded newer image for hello-world:latest
```

6. Check if the docker image has been pulled and is present in your system using the following command:



```
fayssal@fayssal-VirtualBox:~/D1$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
hello-world   latest    feb5d9fea6a5   18 months ago    13.3kB
fayssal@fayssal-VirtualBox:~/D1$
```

7. To display all the containers pulled, use the following command:

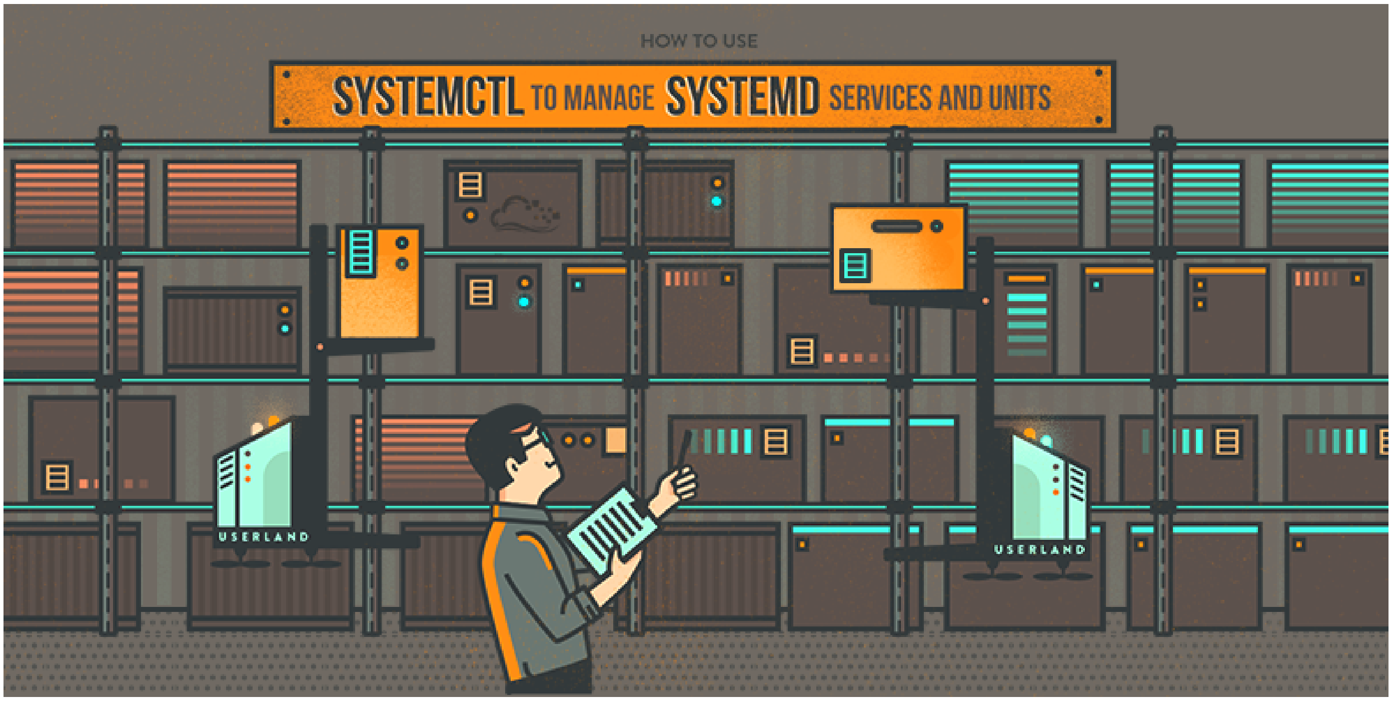


```
hello-world   latest    feb5d9fea6a5   18 months ago   13.3kB
fayssal@fayssal-VirtualBox:~/D1$ sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED              STATUS                          PORTS     NAMES
b92ac4d0364e   hello-world   "/hello"   About a minute ago   Exited (0) About a minute ago             trusting_jackson
fayssal@fayssal-VirtualBox:~/D1$ sudo docker ps
```

What is systemctl and systemd ?

`systemd` is a Linux init system and system manager that is widely used in modern Linux distributions as the default init system. It provides a way to manage and control the various services that run on a Linux system, as well as other system-level functionality.

`systemctl` is the command line tool used to control and manage the `systemd` system and service manager. It provides various commands to start, stop, restart, enable, and disable services, as well as other functionalities such as inspecting the status of services, displaying log messages, and managing system-level settings and configurations.

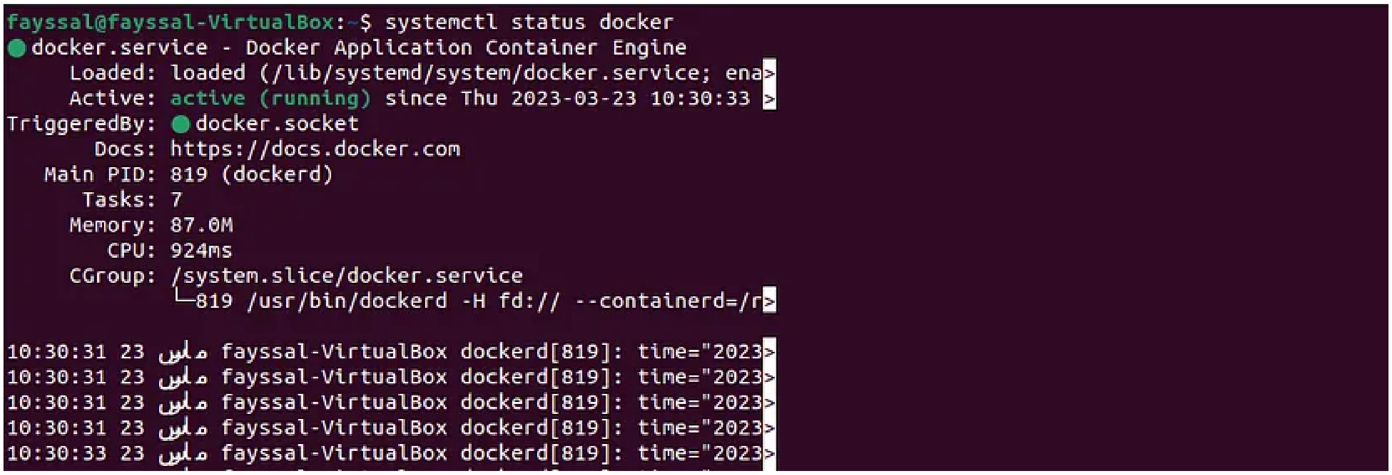


Here are a few examples of common tasks that can be performed using `systemctl`:

- Start a service: `systemctl start <service-name>`
- Stop a service: `systemctl stop <service-name>`

- Restart a service: `systemctl restart <service-name>`
- Enable a service to start automatically at boot: `systemctl enable <service-name>`
- Disable a service from starting automatically at boot: `systemctl disable <service-name>`
- Check the status of a service: `systemctl status <service-name>`

1. **check the status of docker service in your system (make sure you completed above tasks, else docker won't be installed)**



```
fayssal@fayssal-VirtualBox:~$ systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; ena>
     Active: active (running) since Thu 2023-03-23 10:30:33 >
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 819 (dockerd)
      Tasks: 7
     Memory: 87.0M
        CPU: 924ms
     CGroup: /system.slice/docker.service
             └─819 /usr/bin/dockerd -H fd:// --containerd=/r>

10:30:31 23 مارس fayssal-VirtualBox dockerd[819]: time="2023>
10:30:31 23 مارس fayssal-VirtualBox dockerd[819]: time="2023>
10:30:31 23 مارس fayssal-VirtualBox dockerd[819]: time="2023>
10:30:31 23 مارس fayssal-VirtualBox dockerd[819]: time="2023>
10:30:33 23 مارس fayssal-VirtualBox dockerd[819]: time="2023>
```

2. Read about the commands systemctl vs service

`systemctl` and `service` are both tools used to manage and control services on a Linux system. However, they have some differences:

1. `systemctl` is the newer tool and is used on systems that use the Systemd init system, which is now widely adopted as the default init system for many popular Linux distributions, including Fedora, Red Hat Enterprise Linux, and Ubuntu.
2. `service` is the older tool and is used on systems that use the System V init system, which was the previous standard init system used in many popular Linux distributions.
3. `systemctl` provides more advanced features compared to `service`, such as the ability to manage units, which are the basic building blocks of Systemd. This allows

you to manage not just services, but also other system components, such as sockets, devices, and mount points, with a unified interface.

4. `service` is limited to managing services only, and its syntax and options are not as advanced as those of `systemctl` .

If this post was helpful, please do follow and click the clap 👏 button below to show your support 😄

_ Thank you for reading💚

_Fayssal 👍

Ubuntu Linux Package Manager AWS DevOps