Contents

[Dockerfile 1](#_Toc118402982)

[Commands: 1](#_Toc118402983)

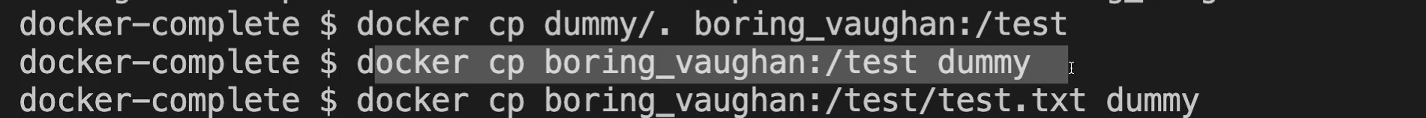
[Volumes = Storages 2](#_Toc118402984)

### Dockerfile

1. VOLUME [“/app/feedback”]
2. images, containers.
3. Docker Hub, Pre-built, Custom Images.
4. Image by Official Team (Node, Java), Community or Colleague.
5. We run Node image to create Node Container.
6. Dockerfile. Dockerfile extension in Visual Code IDE.

### Commands:

* 1. docker build . (dot)
     1. -t :
        1. 
  2. docker <any-command> --help
  3. docker ps
     1. -a : To list stopped containers too.
  4. docker start <container\_name\_or\_id>
     1. By Default, detached mode.
     2. Options:
        1. -a: To run in attached mode.
        2. -i : To run in interactive mode. (Pre-Condition: attached mode)
  5. docker run:
     1. By default, attached mode
     2. Options:
        1. -d : to run in detached mode.
        2. -i : interactive mode (To provide input to the container process)
        3. -t: terminal.
        4. --rm: To remove container automatically when process done.
           1. This will also remove associated anonymous volumes too.
        5. --name:  
           Graphical user interface

           Description automatically generated
        6. -v :
           1. **For named Volume: -v** **<volume-name>**:<path-to-folder-inside-container>  
              **Example:** -v feedback-volume:/app/feedback
           2. **For anonymous Volume:** Same as named volume except skip volume-name.
           3. **For Bind Mount: -v <absolute-path-on-host>:** <path-to-folder-inside-container>  
              **NOTE**: If absolute path contains some special character apart from / then put in double quotes.  
              **NOTE**: The only difference in both syntaxes is in the first part of the argument.
  6. docker attach <container\_id\_name>
     1. No matter a container was run with run or start command.
  7. docker log <container\_id\_name>
     1. Options:
        1. -f: To follow the future logs.
  8. docker stop <container\_name>
  9. docker rm <container\_name>
     1. It removes the stopped container otherwise warning.
     2. -f: To remove running container with SIGKILL.
     3. -v: To remove associated anonymous vol when removing a container.   
         As we know if we start a container with anonymous vol without –rm, anonymous volumes are not automatically removed when the container is removed.
  10. docker rmi <image\_id> : Will remove image along with layers.
      1. -f: There are two cases:
         1. If container is running and image is tagged.  
            Then it will just untag the image.  
            If container is running but image is not tagged,  
            Then it will throw exception.
         2. If container is not running, it will rm.
  11. docker images
  12. docker image
      1. prune : To remove dangling images  
         **dangling image**: Images without any name.  
         **Unused Image**: Images with a name (tagged image)
         1. -a : To remove dangling & unused images.
      2. inspect <image\_id>
  13. docker cp
      1. docker cp <local\_location> container\_id:<container\_ location>
      2. docker cp container\_id:<container\_ location> <local\_location>  
         
  14. docker volume
      1. **ls**: To list all volumes
      2. **rm** <volume-name>
      3. **prune**:

**Total ways to remove anonymous volume**

|  |  |
| --- | --- |
| Start container with --rm | docker run --rm |
| To remove all particular volumes | docker volume rm <vol-name> [<vol-name>] |
| To remove all unused volumes | docker volume prune |
| When removing a container | docker rm -v <container-id> |

### Volumes = Storages

1. Volumes, Bind Mounts.
2. Anonymous Volumes, Named Volumes.
3. Named Volumes.
4. Arguments, Environment Variables.
5. **External Data Storage Mechanisms:**
   1. **Volume**: Changes in either folder (on host machine or inside container) will be reflected in the other one.
   2. **Bind Mounts**:
6. attached, detached, interactive modes.
7. Running process in a container, Running Container.
8. Process in the container = Containerized App.
9. Docker Hub, Private Registry, Container Registry.
10. d
11. Hello
12. d

### Q&A

1. **Question**: Why the following command doesn’t work on Git Bash but works fine on CMD in case of highlighted part (anonymous volume)?
   1. docker run --name feedbackbind --rm -p 8080:80 -v named:/app/feedback -v "C:/jatin/Practice/Docker/data-volumes-03-adj-node-code/data-volumes-03-adj-node-code:/app" -v **/app/node\_modules feedback:bind**
2. **Answer**: See the difference when we run the above command in Git Bash & CMD?
3. 🡺 **Git Bash**:   
   In Git Bash, **/app/node\_modules** wasreplaced with   
   A picture containing chart

   Description automatically generated  
   Solution:  
   Graphical user interface, application, Word

   Description automatically generated  
   🡺 **CMD**:   
   **Already working fine**  
   Chart

   Description automatically generated with low confidence

1. ddfd sfdsfsfsds sfdsfsfsds sfdsfsfsds