Docker Workshop
Appendix 3
VS Code Docker related extensions

1 References

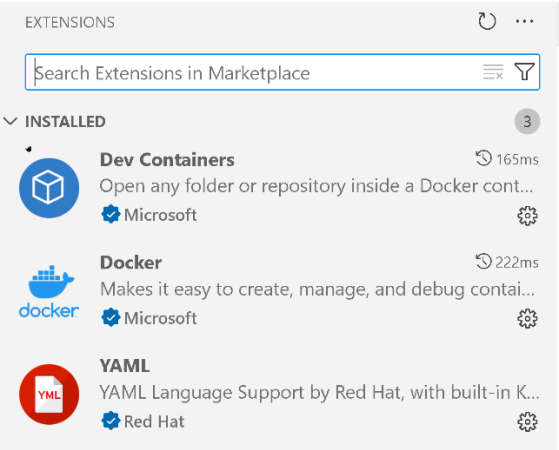
The main official reference for the Docker extension in VS Code

The official reference for working with Dev Containers

2 Setting up

You can install the following extensions from the marketplace for working with Docker

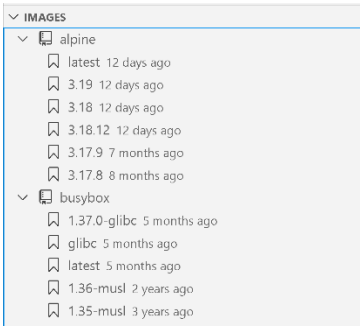
- Docker
- Dev Containers
- YAML



3 Working with the Docker extension

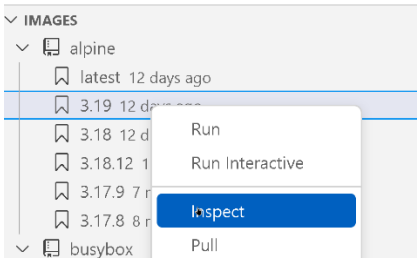
3.1 Working with images

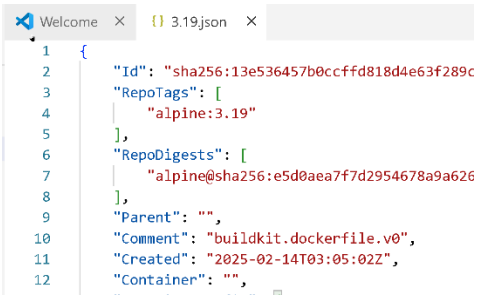
You should be able to see image listing grouped according to their repo (the front portion of the `repo:tag` format for the image name).



3.1.1 Inspecting images

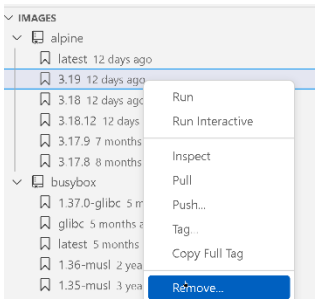
This performs the `docker inspect` command; the information is placed in JSON file which can be downloaded or searched through.

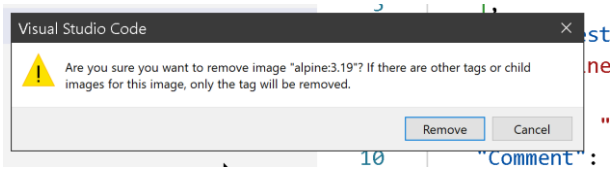




3.1.2 Deleting images

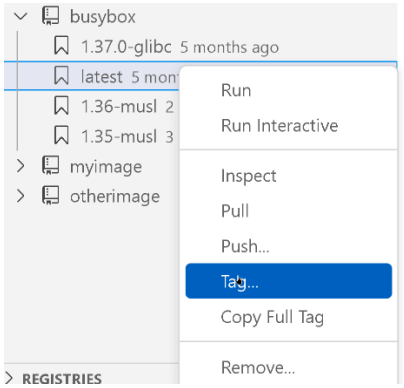
This performs the `docker rmi` command; which means only the tag will be removed for the referenced image in the event there are multiple aliases for that image.

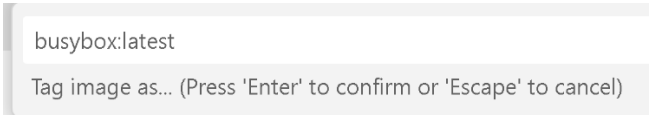




3.1.3 Tagging images

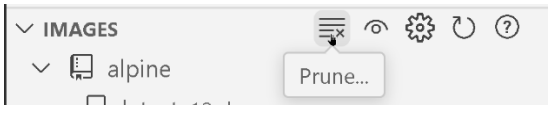
This performs the `docker tag` command; which creates multiple `repo:tag` references to identify a unique image. We can create as many alternative references (tags) for a given image as we want

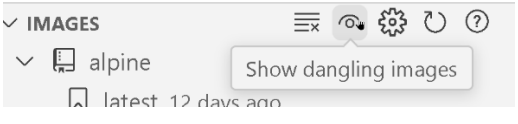




3.1.4 Viewing and deleting dangling images

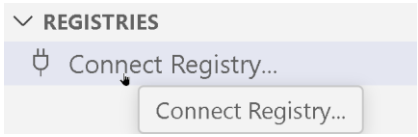
You can choose whether to view or hide dangling images and also delete all of them (equivalent to the `docker image prune` command).

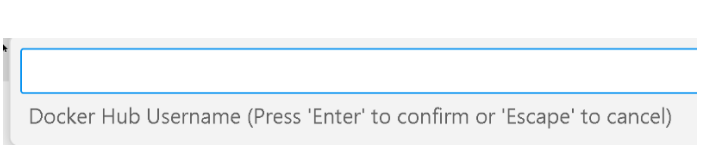




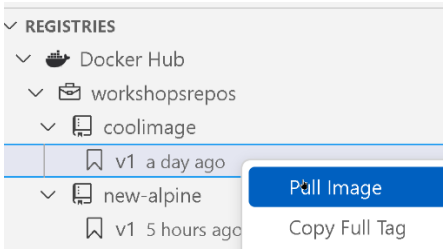
3.1.5 Accessing DockerHub (and other local / remote registries)

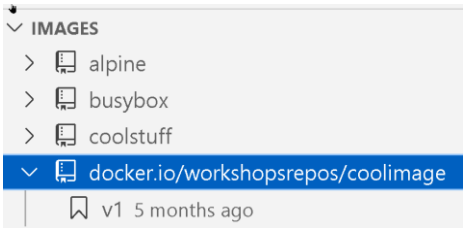
In the Registries section, you can establish connection to DockerHub by entering the appropriate username and password combination. You need to ensure you have logged into Docker Hub via the Docker CLI command (`docker login`) before attempting establishing this connection. If it does not work (due to authorization error), you may need to restart VS code and re-execute `docker login`



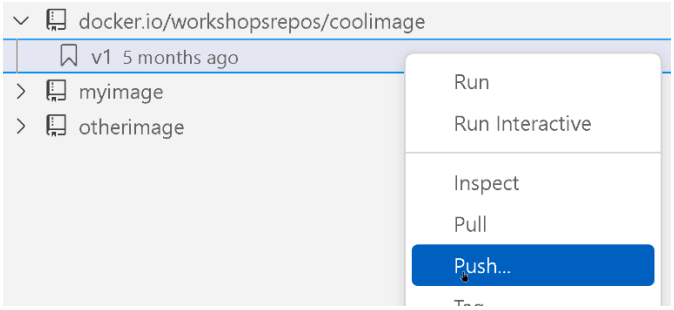


You should able to view all the repos in your DockerHub account and select any of them to pull to your local registry:





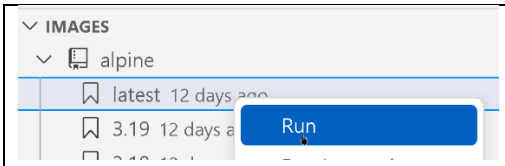
You can also push existing appropriately tagged images to your DockerHub account.

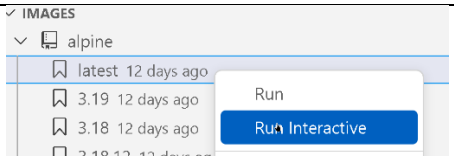


3.1.6 Running containers from existing images

You can choose either to run a container from an image in detached mode (equivalent to `docker run -d`) or with an interactive shell (equivalent to `docker run -it`).
Running it in an interactive mode opens shell terminal in the terminal pane at the bottom of VS Code.

When running in either detached or interactive mode, the `--rm` option is also additionally used which means that the container is immediately deleted once it stops.

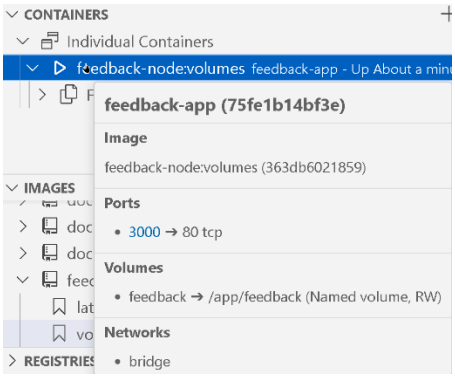




3.2 Working with containers

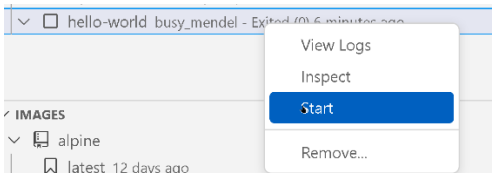
3.2.1 Getting basic info on a container

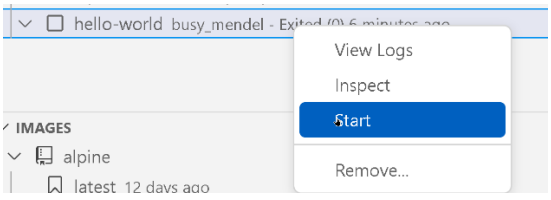
Hovering the mouse over a running container allows you to view info such as the image it was created from, port mappings associated with it, volumes associated with it and so on.



3.2.2 Starting and stopping containers

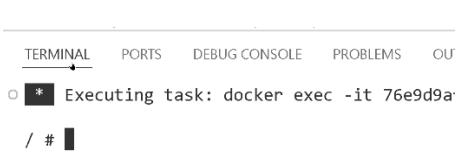
This provides the functionality equivalent to `docker start` and `docker stop`

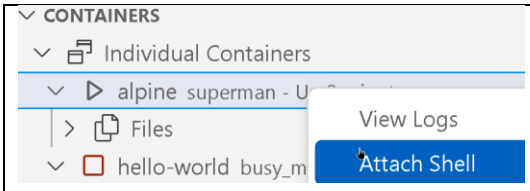




3.2.3 Attaching to a running container in a shell

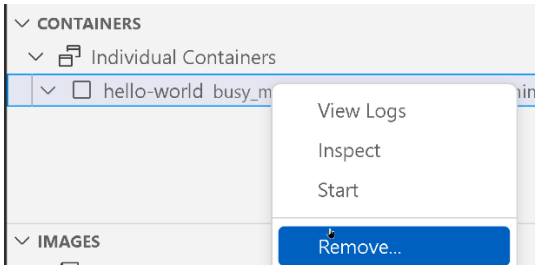
This performs the `docker exec -it` command. The shell is opened in the Terminal pane of VS Code.

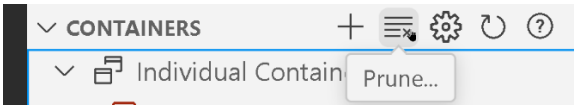




3.2.4 Deleting containers

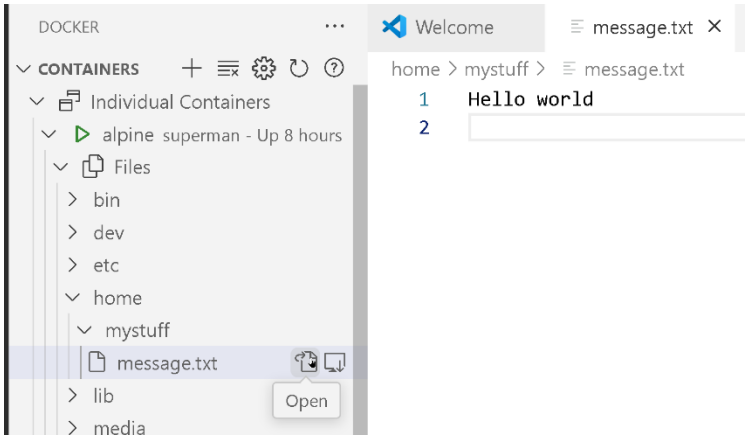
You can either select and delete a single container (`docker rm`) or all stopped containers (`docker container prune`).



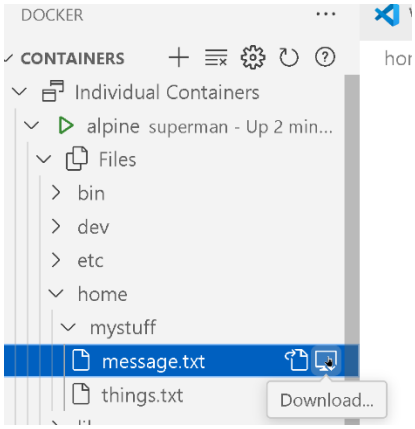


3.2.5 Viewing contents of container file system and downloading files

You should be able to drill down into the directory structure of the local file system of running containers and view the content of files within them. However, you will not be able to manipulate the directories or file contents.

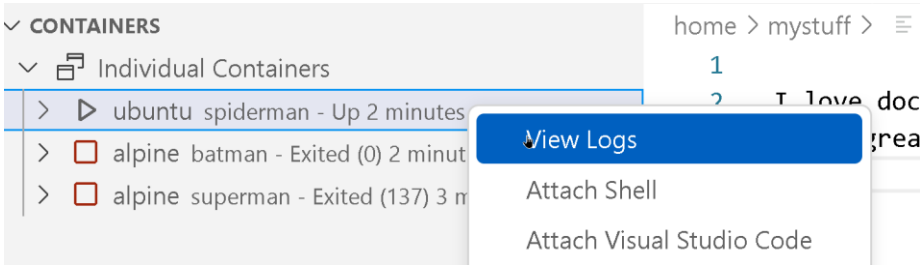


You should also be able to download files from the local file system of a running container to your host machine file system.

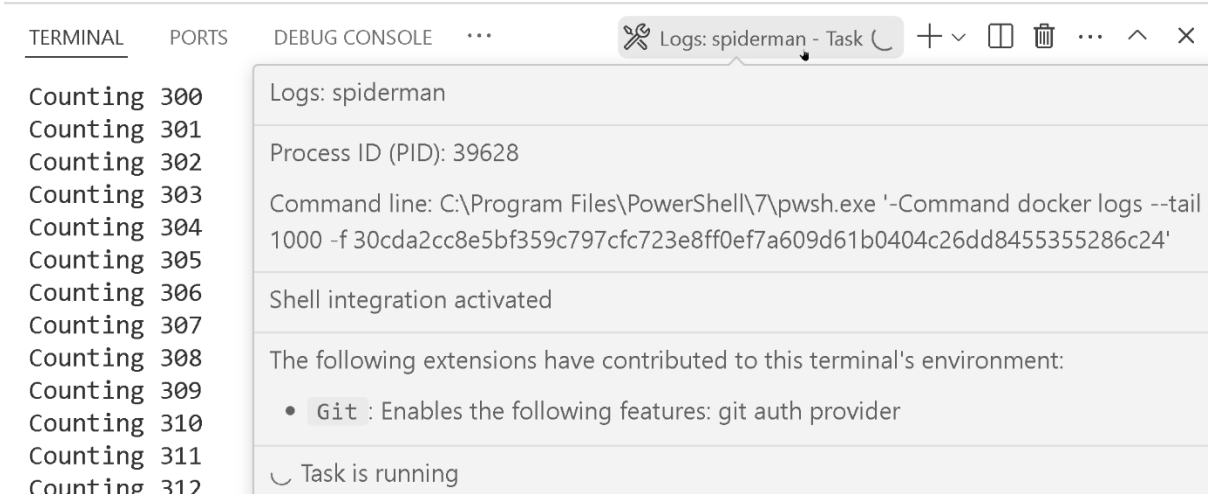


3.2.6 Viewing logs

You can view logs related to console output from a process (such as a script or a web server application) running within a live container.

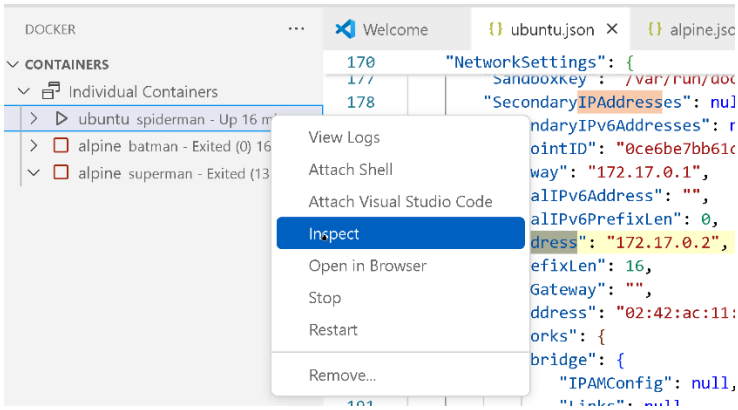


The log output is dynamically updated via a corresponding Docker CLI command issued in Powershell terminal, which you can subsequently terminate with a Ctrl-C.



3.2.7 Inspecting container to obtain configuration details

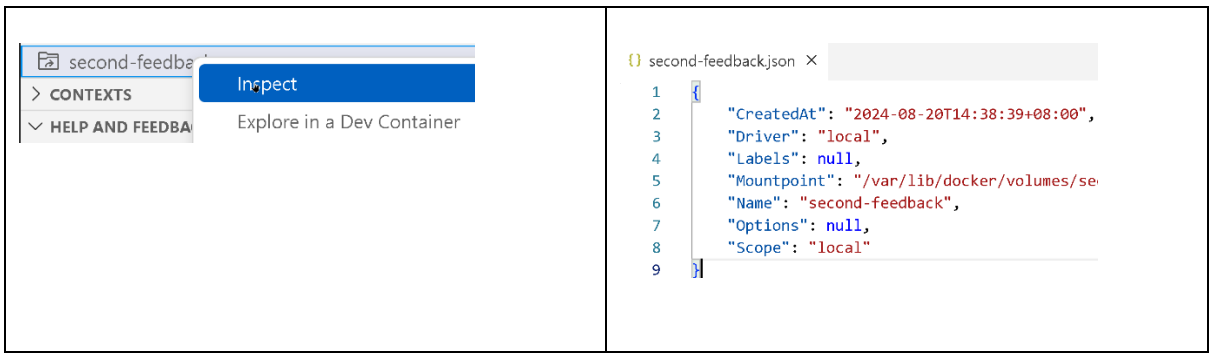
This provides the same information obtained via the `docker inspect container` command in JSON file, which you can then search for specific information, such as the IP address.



3.3 Working with volumes

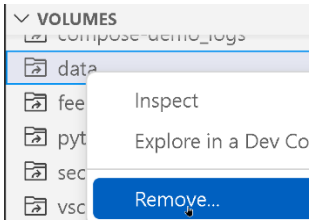
3.3.1 Inspecting images

This is equivalent to the command `docker inspect`; the information is placed in a JSON file which can be downloaded or searched through.



3.3.2 Removing a specific volume

This is equivalent to the command `docker volume rm`



3.3.3 Removing all dangling (unused) volumes

This is equivalent to the command `docker volume prune`