

Tools of the trade II

Joachim Vandekerckhove

Winter 2025

- Introduction to containerization
- **Start the class Docker container**
- Connect with VSCode
- What are SSH keys?

Start the class Docker container

This guide will help you build, run, and test the class Docker container using Docker Desktop. The container is designed to provide a consistent Python environment for our classwork.

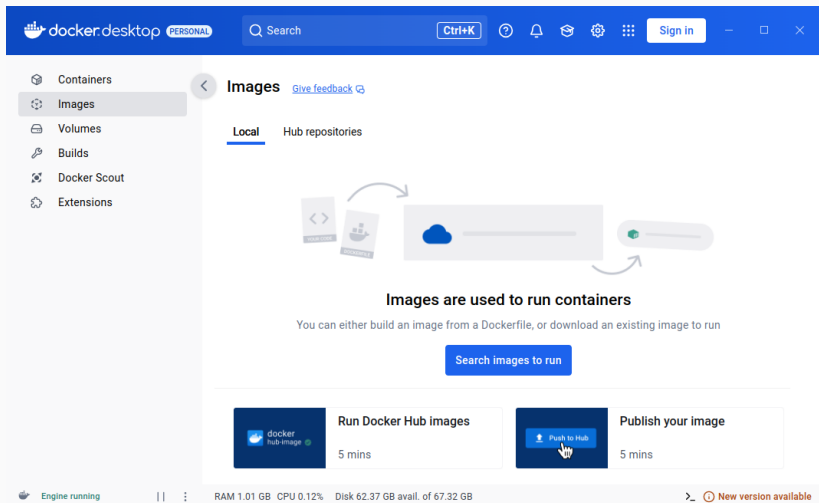
Prerequisites

Before you begin, make sure you have:

1. **Docker Desktop** installed on your computer.
 - Download it from Docker Desktop.
 - Follow the installation instructions for your operating system.

Prerequisites

Open Docker Desktop and navigate to the “Images” tab:



Prerequisites

2. The **Dockerfile** provided for the class.

Save this in a file named Dockerfile:

```
FROM ubuntu:24.04
ENV DEBIAN_FRONTEND=noninteractive
RUN apt-get update && apt-get install -y \
    --no-install-recommends \
    python3=3.12.3-0ubuntu2 \
    python3-pip=24.0+dfsg-1ubuntu1.1 \
    bash=5.2.21-2ubuntu4 \
    openssh-server=1:9.6p1-3ubuntu13.5
RUN mkdir /var/run/sshd && \
    echo 'root:rootpassword' | chpasswd
EXPOSE 22
CMD ["/usr/sbin/sshd", "-D"]
```

Step 1: Build the Docker image

1. **Open a terminal:**

- Navigate to the directory where your Dockerfile is saved.

2. **Run the build command:**

- Use the terminal to build the image:

```
docker build -t class-docker .
```

Step 1: Build the Docker image

3. Verify the image:

- In the “Images” tab, check that `class-docker` is listed.

The screenshot shows the Docker Desktop application window. The top bar is blue with the Docker logo, 'docker desktop PERSONAL', a search bar, and a 'Sign in' button. The left sidebar contains navigation options: Containers, Images (selected), Volumes, Builds, Docker Scout, and Extensions. The main panel is titled 'Images' and has tabs for 'Local' (selected) and 'Hub repositories'. It shows a progress bar for '0 Bytes / 211.99 MB in use' and '1 Images'. Below this is a search bar and a table of images. The table has columns: Name, Tag, Image ID, Created, Size, and Actions. One image is listed: 'class-dor' with tag 'latest' and Image ID 'a0c5fe6247b4', created '1 day ago' and sized '296.58 MB'. At the bottom, there's a 'Walkthroughs' section and a status bar showing 'Engine running', system resources (RAM 1.23 GB, CPU 0.12%, Disk 62.49 GB avail. of 67.32 GB), and a 'New version available' notification.

Images [Give feedback](#)

Local Hub repositories

0 Bytes / 211.99 MB in use 1 Images Last refresh: 3 hours ago

Search

	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	class-dor	latest	a0c5fe6247b4	1 day ago	296.58 MB	▶ ⋮ 🗑

Showing 1 item

Walkthroughs [×](#)

Engine running || : RAM 1.23 GB CPU 0.12% Disk 62.49 GB avail. of 67.32 GB [>](#) [New version available](#)

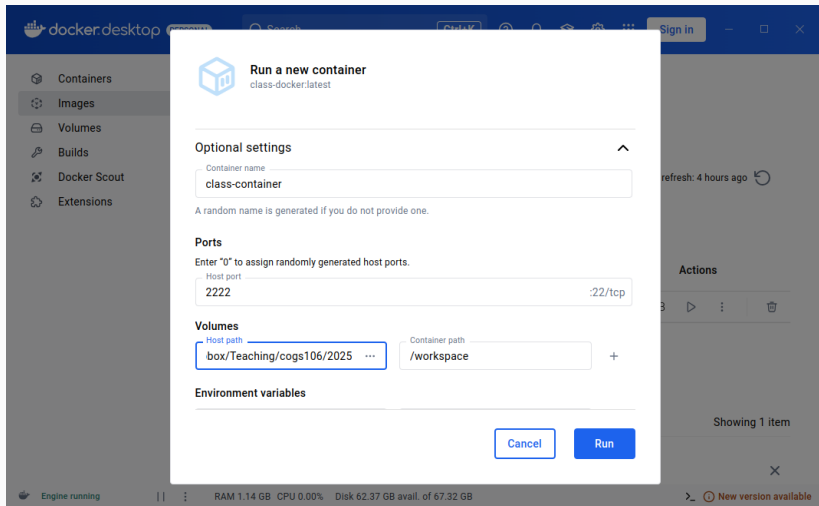
Step 2: Run the Docker Container

1. **Launch the container** from Docker Desktop:

- Go to the “Images” tab.
- Click the play icon next to the class-docker image.
- Set the following options:
 - **Name:** class-container
 - **Ports:** Map “Container port” 22 to “Host port” 2222.
 - **Volumes:** In “Host path,” enter the directory that you want to share with the container. In “Container path,” enter `/workspace`

Setting a Volume is important because only files stored in that directory will survive if the container is destroyed!

Step 2: Run the Docker Container



Step 2: Run the Docker Container

2. Alternatively, you can **use the terminal** to run the container:


```
docker run -dit \
  --name class-container \
  --publish 2222:22 \
  --volume /home/user/cogs106:/workspace \
  class-docker
```

(But don't do this now, use Docker Desktop.)

3. **Verify the container is running:**






- In Docker Desktop, check that the class-container status shows as “Running.”
- Or, use the terminal:

```
docker ps
```




 **docker:desktop** PERSONAL


Q Search


Ctrl+K


    


Sign in


  


 Containers

 Images

 Volumes








 Builds

 Docker Scout

 Extensions

Containers / class-container

class-container

  363730a5e735  [class-docker:latest](#) **STATUS**
Running (0 seconds ago)    

Logs





Inspect


Bind mounts


Exec

Files


Stats



 Engine running

RAM 1.14 GB CPU 0.00% Disk 62.37 GB avail. of 67.32 GB

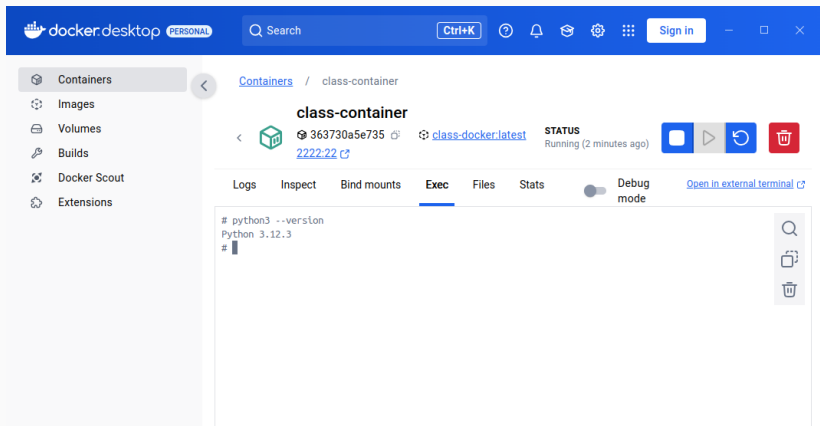
 New version available

Step 3: Test the container

Access the container

1. Use Docker Desktop's integrated shell, and test the Python installation:

```
python3 --version
```



Step 3: Test the container

The integrated shell is running on the Ubuntu/Linux operating system inside the container. You can learn the basic commands of the command line interface from the cheat sheet on GitHub.

2. Alternatively, open your system's terminal to connect to the container:

```
docker exec -it class-container bash
```

When you're done: Stop (and later restart) the container

In Docker Desktop, click the stop or play button in the Containers tab.

Or, in the terminal:

- To stop the container:

```
docker stop class-container
```

- To restart the container:

```
docker start class-container
```

Summary

You've now set up the class Docker container, confirmed it works, and learned how to manage it using Docker Desktop. This container will be your consistent Python environment for all class activities.

In the future, you could make new containers for new projects and manage them in Docker Desktop or submit them to Docker Hub.

Tools of the trade II

Joachim Vandekerckhove

Winter 2025