Tools of the trade I

Joachim Vandekerckhove

today

Contents

Introduction to containerization

The goal of containerization is to ensure consistent and portable computational environments. That is, its goal is to create a computational system that reliably works the same on every computer. This is especially useful in computational research, where analyses can be complex and depend on a lot of interacting pieces of software and code that may come in many versions.

We will use Docker Desktop to understand the basic ideas and practical applications.

Learning objectives

By the end of this lecture, you should:

1. Understand what containerization means.
2. Know why containers are useful in everyday computing tasks.
3. Explore how Docker Desktop makes it easy to run programs consistently on any computer.

What is containerization?

Containerization is a way to package software and everything it needs to run into a single "box," called a **container**. This box can be moved to any computer, and the software will run exactly the same way. Containerization is a lot like virtualization but tends to be more efficient.

You can think of a container a little bit like a parasite that lives inside another computer. That has a negative connotation, so we will simply call it the **container** and the computer will be the **host**. A single host can have many containers.

Below, we will also talk about the **image**, which something like a template to make a kind of container. We can have many images, and each image can spawn many containers. Often, it's the image we will share to others.

Everything's in there

When you buy a piece of furniture from IKEA, the package includes everything you need: screws, tools, instructions, and the wooden pieces. Even if you don't have your own tools, you can still assemble the furniture because everything is in the box. Similarly, the **image** provides all the "tools" and "pieces" a program needs to run, so it works the same way on any computer. The **container** is like the finished piece of furniture.

Why use containers?

- **Consistency**: Containers ensure the same program works the same way, no matter whose computer it's on.
- **Portability**: You can move containers between your laptop, a friend's computer, or even powerful online servers.
- **Simplicity**: Containers bundle everything needed, so you don't have to install things one by one.

Key concepts and terminology

1. Images

- An **image** is like a recipe for making a container.
- It includes the program, its tools, and instructions on how to set everything up.

2. Containers

- A **container** is a running version of an image.
- Think of it as the assembled IKEA furniture created from the box's contents.

Key concepts and terminology

3. Docker and Docker Desktop

Docker is a set of services to create containers. It is a widely used industry standard. "A docker" is a container made this way.

Docker Desktop is the specific application we will use to work with Docker containers using a graphical interface and minimal technical setup.

Key concepts and terminology

Steps to get started:

1. Download and install Docker Desktop.
2. Open the application, and follow the setup instructions.
3. Verify the installation by running the "Hello World" container (from Docker Desktop).

Key concepts and terminology

4. Registry

- A **registry** is a library for storing and sharing images.
- Docker Hub is a public registry where you can find and share containers.

Key concepts and terminology

5. Port mapping

- A **port** is an entry point to a program running on a computer. Every computer has many ports, some dedicated for particular services, and they are usually closed.
- Port mapping assigns a port on the *host* computer to a port on the *container* so you can use its services.

Key concepts and terminology

Docker Desktop

Docker Desktop is a tool that makes it easy to create, share, and run containers without needing to write complex commands. It simplifies the process with a visual interface.

- **Graphical interface**: No need to memorize commands; most operations are clickable.
- **Cross-platform**: Works on Windows, Mac, and Linux.
- **Prebuilt containers**: Access ready-to-use containers for popular programs with just a few clicks.

Containers are important

Without containers

- Programs often need specific versions of tools or software to work.
- What works on one computer might not work on another.
- Sharing programs can be tricky because everyone's computer is set up differently.
- Changing your local setup for one program might break others.

Containers are important

Container solutions

- **Encapsulation**: Everything the program needs is packaged together.
- **Portability**: Containers run the same way everywhere.
- **Reproducibility**: You can guarantee the same results, no matter where or when the program is run.

Containers are important

Everyday benefits

- **For this class**: You can run the same programming environment as I do so my examples will work on your computer and your assignments will work on mine.
- **For students**: You can simulate a computer and if you mess it up you can just start over.
- **For teams**: Everyone works with the same tools, avoiding conflicts.
- **For sharing**: You can send a image to someone, and it will just work.
- **For researchers**: Reviewers and readers can download your Docker image and repeat your analyses.

Recap

- Containers are like portable "boxes" for programs, ensuring they run the same way everywhere.
- Docker Desktop makes creating and using containers simple and efficient.
- Containers solve problems of consistency, portability, and reproducibility.

Next steps

1. Download and install Docker Desktop on your computer.
2. Run the "Hello World" container to verify your setup.
3. Go to the next chapter