One Tool to Rule Them All

In this lesson, you will learn how Docker acts as a single tool for managing multiple different technologies.

In the next chapter, I'll show you examples of *Dockerfile* files for several development technologies. What I'd like you to note beforehand is that the processes we use don't depend on the development technology. Whether Java, Python, .NET Core, Node.JS, PHP, or any other, the same instructions are used to build, ship, and deploy the applications.

This is a major benefit in many environments. If you're part of a team, the chances are that many technologies are being used; however, the way to build, ship, and deploy the applications depends on the applications' development tools. Having a single tool to manage them all yields several benefits:

- Members of the team that use one development stack can help those using others.
- A single CI/CD process and standard tools can be used across most projects, and improvements benefit all of those projects.

Practically, whatever the technology, the same instructions are used to build, ship, and deploy the applications. Something around those lines:

Build, deliver on the CI/CD server

```
docker build -t learnbook/my-server .
docker push learnbook/my-server
```

Deploy on the target machine

```
docker run --rm -it -p 80:80 learnbook/my-server
```

Let’s continue our discussion on the benefits of Docker in the next lesson.