Introduction

In this lesson, we will discuss briefly about the currently used technologies and will peek a bit into the past.

WE'LL COVER THE FOLLOWING

A Brief About Technologies

Technology is changing so fast that it is very hard, if not impossible to follow. The moment we learn about a new technology, it is already obsolete and replaced with something else.

Take **containers** as an example. Docker appeared only a few years ago, and everyone is already using it for a myriad of scenarios. Still, even though it is a very young product, it changed many times over. Just when we learned how to use `docker run`, we were told that it is obsolete and should be replaced with `docker-compose up`. We started converting all our `docker run` commands into Compose YAML format.

The moment we finished the conversion, we learned that containers should not be run directly. We should use a container scheduler instead. To make things more complicated, we had to make a selection between **Mesos and Marathon, Docker Swarm, or Kubernetes**.

We can choose to ignore the trends but that would mean that we would fall behind the competition. There is no alternative to a constant struggle to be competitive. Once we drop our guard and stop learning and improving, the competition will take over our business. Everyone is under pressure to improve, even highly regulated industries. Innovation is impossible until we manage to get to the present tense. Only once we master what others are doing today, we can move forward and come up with something new.

Today, container schedulers are a norm. They are not the thing of the future. They are the present. They are here to stay, even though it is likely that they will change a lot in the coming months and years. Understanding container schedulers is extremely important.

> **Kubernetes** is the most widely used container scheduler that has a massive community behind it.

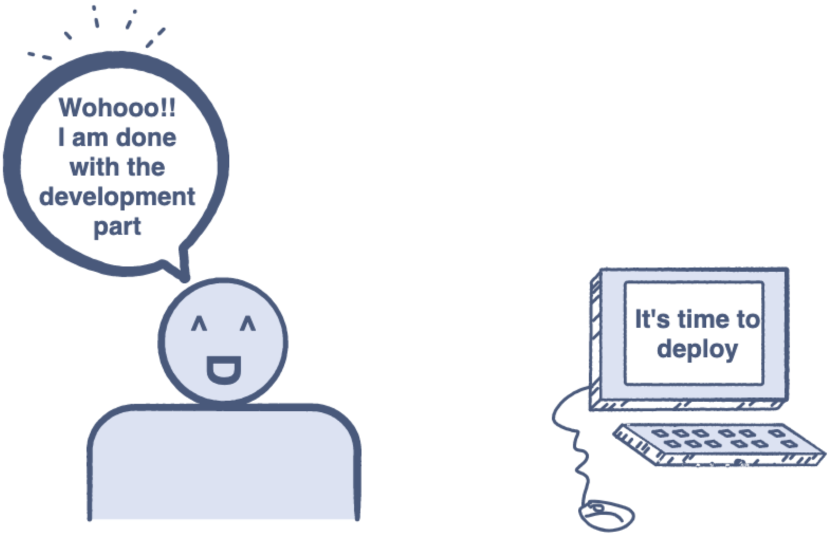
Before we dive into Kubernetes, it might be worthwhile going through some history in an attempt to understand some of the problems we were trying to solve, as well as some of the challenges we were facing.

A Glimpse From The Past

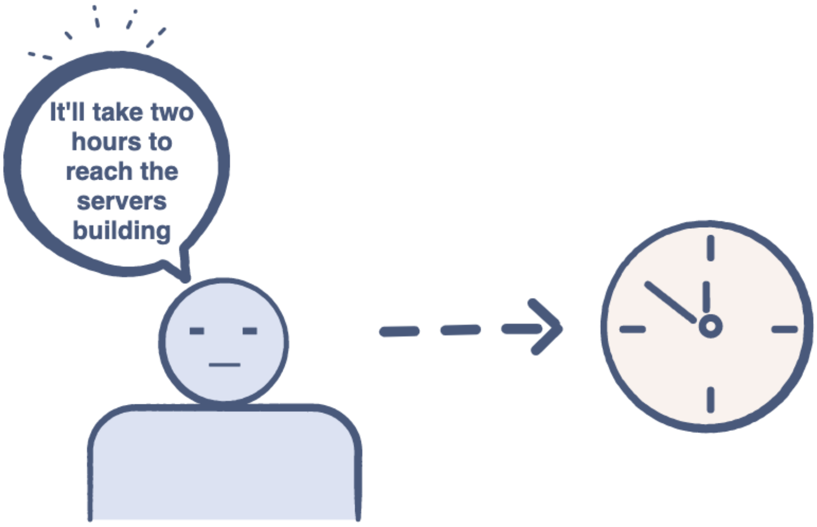
Picture a young boy. He just finished a few months worth of work. He's proud of what he accomplished yet fearful whether it will work. He did not yet try it out on a "real" server. This will be the first time he'll deliver the fruits of his work.

He takes a floppy disk out from a drawer, inserts it into his computer, and copies the files he compiled previously. He feels fortunate that perforated cards are a thing of the past.

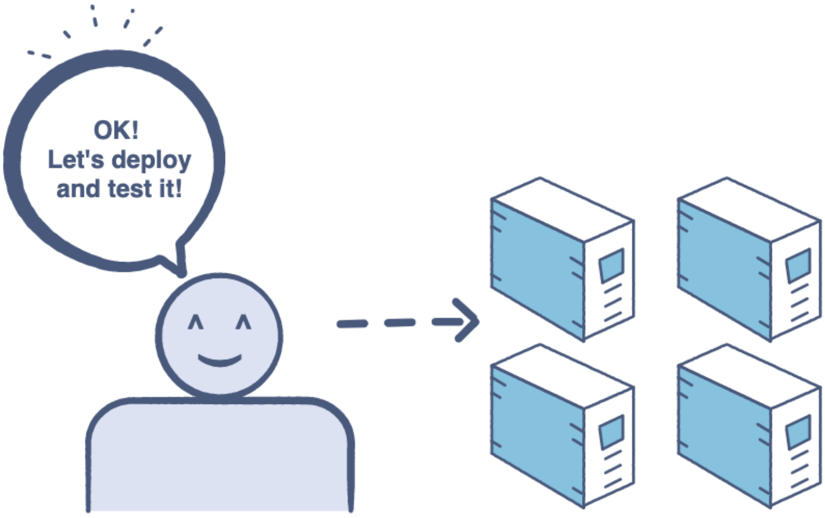
He gets up from his desk, exits the office, and walks towards his car. It will take him over two hours to get to the building with servers. He's not happy with the prospect of having to drive for two hours, but there is no better alternative. He could have sent the floppy with a messenger, but that would do no good since he wants to install the software himself. He needs to be there. There is no remote option.



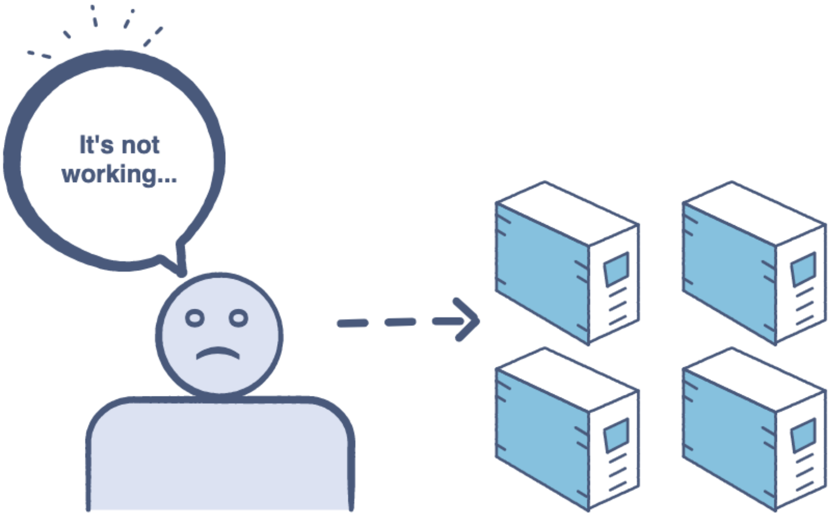
1 of 4


It'll take two hours to reach the servers building

2 of 4


OK! Let's deploy and test it!

3 of 4



4 of 4

A while later, he enters the room with the servers, inserts the floppy disk, and copies and installs the software. Fifteen minutes later, his face shows signs of stress. Something is not working as expected. There is an unforeseen problem. He's collecting outputs and writing notes. He's doing his best to stay calm and gather as much info as he can. He's dreading a long ride back to his computer and days, maybe even weeks, until he figures out what caused the problem and fixes it. He'll be back and install the fix. Perhaps it will work the second time. More likely it won't.

So, this was a glimpse from the past. We can imagine the uncertainty and the effort one needed to put for getting a simple deployment task done.

In the next lesson, we will look into the history of infrastructure management.