**Docker Introduction**

**Problem that Docker Solves**

**Problem Overview**

We need to understand what problem Docker is going to solve.

Let's say we have an app that was developed with:

* **JDK 21**
* **SQL V2**
* **Redis V1**

This app was developed on a **Windows** machine.  
Now, suppose a team member is using a **MacOS** machine and has installed:

* **JDK 23**
* **SQL V4**
* **Redis V7** (some latest versions)

Now the code must work on the new machine, but version conflicts need to be resolved to run the code correctly.

**Challenges**

➡️ If there are **many developers** in the team and new people keep joining, it will become hectic to manage dependencies and configurations.

➡️ The problem will become even more complicated when moving the application to **production**.

**Common Problem:**

✅ Many developers in the team  
✅ Moves to production  
✅ All these things will lead to a common problem called **"It works in my machine."**

**What is Docker?**

Docker helps us to resolve this issue by building **containers**.

➡️ **Package the code and its dependent software into a single unit.**

**Why Do We Need Docker?**

This packaged unit can be shared with fellow developers.  
➡️ The beauty of these containers is they can run on **any machine**.

**Docker Containers**

Docker provides a platform that helps in:

* **Building containers**
* **Updating containers**
* **Destroying containers**

**Properties of Docker Containers:**

✅ **Portable** – This can be shared between machines  
✅ **Lightweight** – These can be easily created, updated, and destroyed

**Docker Image and Containers**

**Key Term – Docker Image**

1. Docker image is an **executable file**; it has instructions about how the container should be constructed.
2. Using an image, we can build **multiple containers**.
3. We don’t store containers; we store images, and this shared image can build containers.

**Analogy:**

➡️ **Class →** It’s a blueprint of an object  
➡️ **Object →** It’s an instance of a class

Similarly:  
➡️ **Image →** Instructions about dependencies in the container  
➡️ **Container →** Instance of an image

**Docker Installation and Setup**

**Setup Process:**

1. After setting up Docker:

**docker pull hello-world**

This command will pull the image from **DockerHub**.

1. After the image is pulled into our machine, we have to build a container:

**docker run hello-world**