Docker

dasturlash.uz

Docker

- Docker is a container management service.
- The keywords of Docker are **develop, ship** and **run** anywhere.
- The whole idea of Docker is for developers to easily develop applications, ship them into containers which can then be deployed anywhere.

What is Docker?

- *an open-source project that automates the deployment of software applications inside **containers** by providing an additional layer of abstraction and automation of **OS-level virtualization** on Linux.*
- Docker is a tool that allows developers, sys-admins etc. to easily deploy their applications in a sandbox (called *containers*) to run on the host operating system i.e. Linux.
- The key benefit of Docker is that it allows users to **package an application with all of its dependencies into a standardized unit** for software development.
- Unlike virtual machines, containers do not have high overhead and hence enable more efficient usage of the underlying system and resources.

Docket install

- https://docs.docker.com/desktop/install/windows-install/

Docker for Windows error:

- "Hardware assisted virtualization and data execution protection must be enabled in the BIOS“

- 1. Restart PC
- 2. While you are on the 'restart' screen press any of these keys and you enter the bios settings in windows: **esc, f1, f2, f3, f4, f8 or delete**
- 3. For intel based systems:
  - press f7 (advanced mode)
  - go to advanced
  - cpa configuration
  - enable virtualization
- https://stackoverflow.com/questions/39684974/docker-for-windows-error-hardware-assisted-virtualization-and-data-execution-p

WSL 2 installation is incomplete.

- https://docs.microsoft.com/en-us/windows/wsl/install-manual#step-4---download-the-linux-kernel-update-package

Docker - Hub

- Docker Hub is a registry service on the cloud that allows you to download Docker images that are built by other communities.
- You can also upload your own Docker built images to Docker hub

Hello world

- This command will download the **hello-world** image, if it is not already present, and run the **hello-world** as a container.
  - docker run hello-world

Docker - Images

- In Docker, everything is based on Images. An image is a combination of a file system and parameters.
- docker run hello-world
- The Docker command is specific and tells the Docker program on the Operating System that something needs to be done.
- The **run** command is used to mention that we want to create an instance of an image, which is then called a **container**.
- Finally, "hello-world" represents the image from which the container is made.

Displaying Docker Images

- To see the list of Docker images on the system, you can issue the following command.
  - docker images
-

Downloading Docker Images

- Images can be downloaded from Docker Hub using the Docker **run** command. Let's see in detail how we can do this.
  - docker run image
-

Removing Docker Images

- The Docker images on the system can be removed via the **docker rmi** command. Let's look at this command in more detail.
  - docker rmi ImageID

docker images -q

- This command is used to return only the Image ID’s of the images.
- docker images

- **q** – It tells the Docker command to return the Image ID’s only
  - docker images -q
-

Docker Images: docker inspect

- This command is used see the details of an image or container.
  - docker inspect Repository
-

Docker - Containers

- Containers are instances of Docker images that can be run using the Docker run command.
- The basic purpose of Docker is to run containers.
- Running a Container
  - Running of containers is managed with the **Docker run** command. To run a container in an interactive mode, first launch the Docker container.
- Listing of Containers
  - One can list all of the containers on the machine via the **docker ps** command. This command is used to return the currently running containers.
- This command is used to list all of the containers on the system
  - docker ps -a
- With this command, you can see all the commands that were run with an image via a container.
  - docker history
- das

Spring Boot + Docker

dasturlash.uz

Spring Boot

```
@RestController
@RequestMapping("")
public class TestController {

    @GetMapping("")
    public String test() {
        return "<h1>Salom</h1>";
    }
}
```

Docker File

- A Dockerfile is a text file, contains all the commands to assemble the docker image.

- 1. It creates a docker image base on adoptopenjdk/openjdk11:alpine-jre, an alpine linux with openjdk11 installed. This base image adoptopenjdk/openjdk11:alpine-jre is just an example, we can find more base images from the official Docker Hub

```
# For Java 8, try this
# FROM openjdk:8-jdk-alpine

# For Java 11, try this
FROM adoptopenjdk/openjdk11:alpine-jre

#For Java 17, try this
FROM openjdk:17-jdk-slim-buster
```

Docker File

- 2. Changed the working directory to /opt/app

```
WORKDIR /opt/app
```

- 3. Copy **spring-boot-docker.jar** to /opt/app/app.jar

```
ARG JAR_FILE=target/spring-boot-web.jar

# cp spring-boot-web.jar /opt/app/app.jar
COPY ${JAR_FILE} app.jar
```

- 4 Run the jar file with ENTRYPOINT.

```
# java -jar /opt/app/app.jar
ENTRYPOINT ["java","-jar","app.jar"]
```

A complete Dockerfile example.

```
# For Java 17, try this
FROM openjdk:17-jdk-slim-buster

# Refer to Maven build -> finalName
ARG JAR_FILE=target/spring-boot-web.jar

# cd /opt/app
WORKDIR /opt/app

# cp target/spring-boot-web.jar /opt/app/app.jar
COPY ${JAR_FILE} app.jar

# java -jar /opt/app/app.jar
ENTRYPOINT ["java","-jar","app.jar"]
```

Some useful Command 1

- The COPY layer will copy the local jar previously built by Maven into our image

```
# copy the packaged jar file into our docker image
COPY target/demo-0.0.1-SNAPSHOT.jar /demo.jar
```

- The CMD layer tells Docker the command to run inside the container once the previous steps have been executed

```
# set the startup command to execute the jar
CMD ["java", "-jar", "/demo.jar"]
```

Some useful Command 2

```
COPY target/docker-example-*.jar app.jar
```

Build docker image from .jar

- docker build –t some-name .
- OR
- docker build –t my-app:1.0.1 .

Start the docker container

```
docker run -d -p 8080:8080 image-name
```

- -d to start the container in detach mode – run the container in the background.
- -p to map ports.

Common Dockerfile

- Common Dockerfile instructions start with RUN, ENV, FROM, MAINTAINER, ADD, and CMD, among others.
- FROM - Specifies the base image that the Dockerfile will use to build a new image. For this post, we are using phusion/baseimage as our base image because it is a minimal Ubuntu-based image modified for Docker friendliness.
- MAINTAINER - Specifies the Dockerfile Author Name and his/her email.
- RUN - Runs any UNIX command to build the image.
- ENV - Sets the environment variables. For this post, JAVA_HOME is the variable that is set.
- CMD - Provides the facility to run commands at the start of container. This can be overridden upon executing the docker run command.
- ADD - This instruction copies the new files, directories into the Docker container file system at specified destination.
- EXPOSE - This instruction exposes specified port to the host machine.

Save Docker image

- docker save -o app_image.tar your_image_name
- OR
- docker save image_name > outpuyFileName.tar
- OR
- docker save --output busybox.tar busybox

- https://docs.docker.com/engine/reference/commandline/save/

Load Docker image

- docker load -i tar_tile_name.tar
- OR
- docker load --input fedora.tar
- OR
- docker load < busybox.tar.gz

- https://docs.docker.com/engine/reference/commandline/load/

Docker Compose

- Docker Compose is **a tool that was developed to help define and share multi-container applications**. With Compose, we can create a YAML file to define the services and with a single command, can spin everything up or tear it all down.

- It allows to deploy, combine, and configure multiple docker containers at the same time. The Docker "rule" is to outsource every single process to its own Docker container.