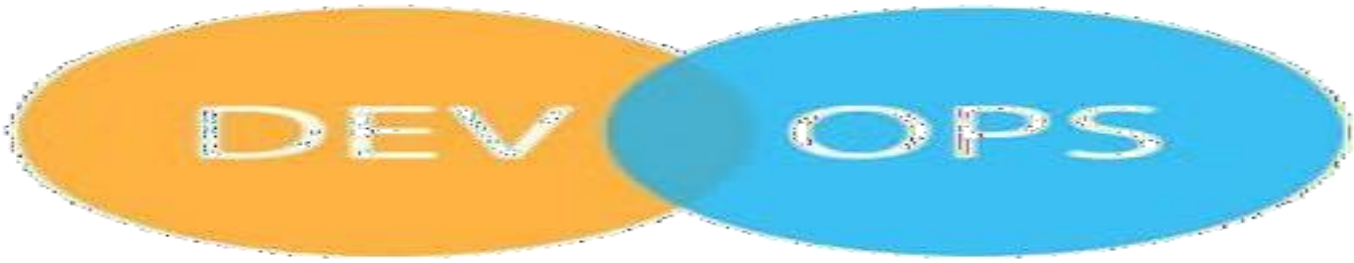

DEV
OPS

Task-1:-

Separating build and runtime environment using multi-stage builds
Create HelloWorld.java file as shown below

```
cat >HelloWorld.java <<EOF
class HelloWorld {
    public static void main(String[] a) {
        System.out.println("Hello world!");
    }
}
EOF
```

Then, create Dockerfile

```
cat >Dockerfile <<EOF
FROM openjdk:11-jdk
COPY HelloWorld.java .
RUN javac HelloWorld.java
CMD java HelloWorld
EOF
```

```
docker image build --tag helloworld:huge .
docker container run helloworld:huge
```

Try to understand below command:

```
docker container run --volume $PWD:/src --workdir /src openjdk:11-jdk javac  *.java
```

After the above command, simplified version of Dockerfile

```
cat >Dockerfile <<EOF
FROM openjdk:11-jre
COPY HelloWorld.class .
CMD java HelloWorld
EOF
```

This time, the resulting image will not contain the source code…

```
docker image build --tag helloworld:run .
```

…still produce the same output…

```
docker container run helloworld:run
```

…and stay below 300MB in size:

```
docker image ls
```

Enter multi-stage builds

Whatever we did above, we can simplify using Multi-Stage Docker build.

```
cat >Dockerfile <<EOF
FROM openjdk:11-jdk AS build
COPY HelloWorld.java .
```

```
RUN javac HelloWorld.java

FROM openjdk:11-jre AS run
COPY --from=build HelloWorld.class .
CMD java HelloWorld
EOF
```

Building the image looks very similar to the well-known process:

```
docker image build --tag helloworld:small .
```

The resulting image will work as expected…

```
docker container run helloworld:small
```

…but have the same small size as above:

```
docker image ls
```

Task-2: Execute below Dockerfile and try to understands Multi-Stage in Dockerfile.

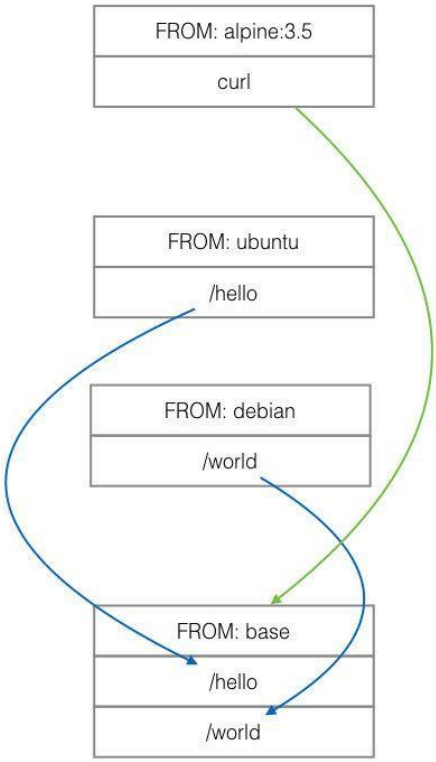
Dockerfile

```
# Base Image
FROM alpine:3.5 AS base
RUN apk add --no-cache curl

# Second Image
FROM debian AS second
RUN echo hello > /hello
LABEL image=second

# Third Image
FROM ubuntu AS third
RUN echo world > /world
LABEL image=third

# FINAL Image
FROM base
# Copy files from other images
COPY --from=second /hello /hello
COPY --from=third /world /world
```



Task-3: Build, Multi-Stage Build and build.sh file.

Try to execute and learn from the attached code.

multi-stage-build-code.zip

Note: To run docker container refer README.md file.