Experiment 4
Docker Build and Push using GitHub Actions

Objective: Set up a GitHub Actions workflow to automatically build a Docker image from a Dockerfile in your GitHub repository and push it to a container registry (e.g., Docker Hub).

Prerequisites:

GitHub account

- Docker installed on your local machine
- A Dockerfile in your GitHub repository
- A Docker Hub account (or any other container registry)

Exercise Steps:

Step 1: Fork and Clone the Repository

- Fork a sample GitHub repository containing a Dockerfile or create a new repository and add a Dockerfile to it.
- Clone the forked repository to your local machine.

Step 2: Create Docker Hub Access Token

- Log in to your Docker Hub account.
- Go to your account settings and click on the "Security" tab.
- Under "Access Tokens," click "New Access Token." Give it a name, select the required permissions (e.g., "Write" for pushing Docker images), and click "Create."
- Copy the generated access token. You will need it to authenticate with Docker Hub in your GitHub Actions workflow.

Step 3: Create a GitHub Actions Workflow

- In your cloned repository, create a directory named .github/workflows if it doesn't exist.

- Inside the .github/workflows directory, create a YAML file (e.g., docker-build-and-push.yml) to define your GitHub Actions workflow. You can use any text editor to create the file.
- Edit docker-build-and-push.yml and add the following content:

```yaml
name: Docker Build and Push
on:
  push:
    branches:
      - main  # Change this to your main branch name


jobs:
  build-and-push:
    runs-on: ubuntu-latest


    steps:
    - name: Checkout code
      uses: actions/checkout@v2


    - name: Login to Docker Hub
      run: docker login -u ${{ secrets.DOCKER_USERNAME }} -p $
{{ secrets.DOCKER_PASSWORD }}
      env:
        DOCKER_USERNAME: ${{ secrets.DOCKER_USERNAME }}
        DOCKER_PASSWORD: ${{ secrets.DOCKER_PASSWORD }}


    - name: Build and Push Docker Image
      run: |
        docker build -t your-dockerhub-username/your-repo-name:latest .
        docker push your-dockerhub-username/your-repo-name:latest
```

Replace your-dockerhub-username and your-repo-name with your Docker Hub username and repository name.

Step 4: Add Docker Hub Credentials to GitHub Secrets

- Go to your GitHub repository on the GitHub website.
- Click on "Settings" and then "Secrets" in the left sidebar.
- Click on "New repository secret" and add two secrets:
- DOCKER_USERNAME: Set this to your Docker Hub username.
- DOCKER_PASSWORD: Set this to the Docker Hub access token you generated earlier.

Step 5: Commit and Push Changes

Save the docker-build-and-push.yml file.

Commit the changes to your local repository:

```
git add .

git commit -m "Add GitHub Actions workflow for Docker build and push"

git push origin main
```

Step 6: Check the Workflow Status

- Go to your GitHub repository on the GitHub website.
- Click on the "Actions" tab to see the workflow running. You should see a workflow named "Docker Build and Push" or the name you specified in the YAML file.
- Monitor the workflow's progress, and once it completes successfully, you should see a green checkmark indicating a successful build and push of the Docker image to Docker Hub.

Step 7: Verify the Docker Image on Docker Hub

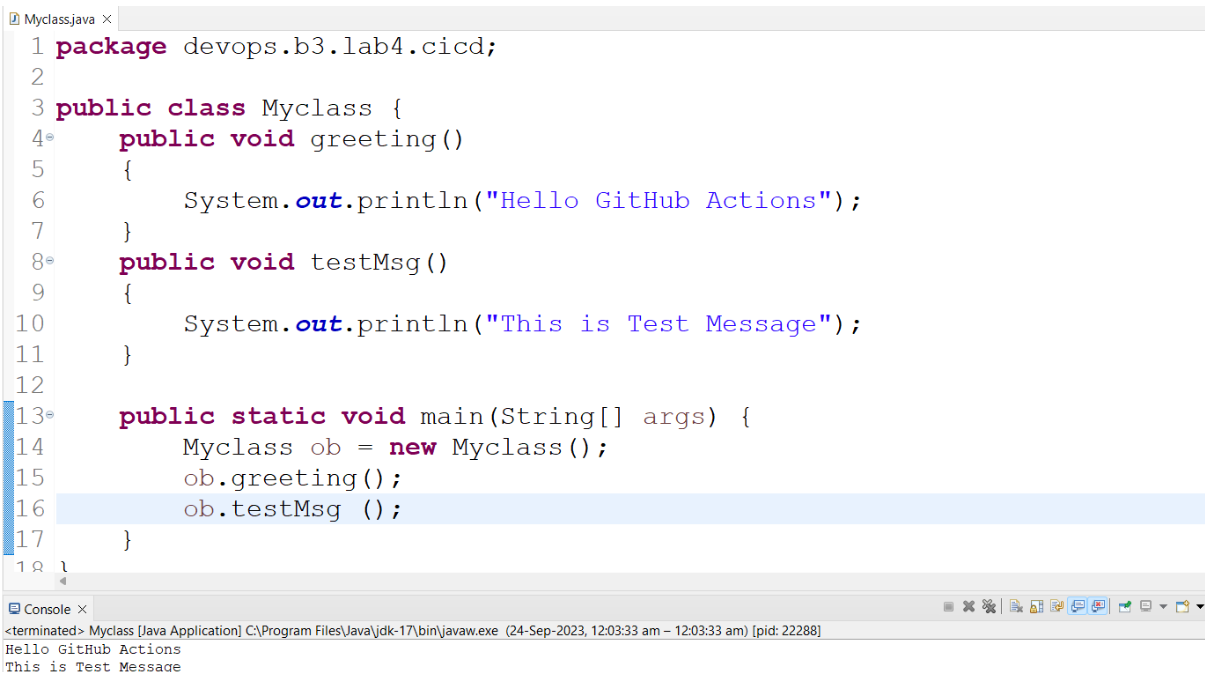
- Log in to your Docker Hub account.
- Navigate to your Docker Hub repository, and you should see the Docker image you pushed from the GitHub Actions workflow.

Step 8: Optional - Trigger a Build

To test the workflow, make changes to your Dockerfile or application code, commit, and push them to the repository. This should trigger the GitHub Actions workflow automatically.

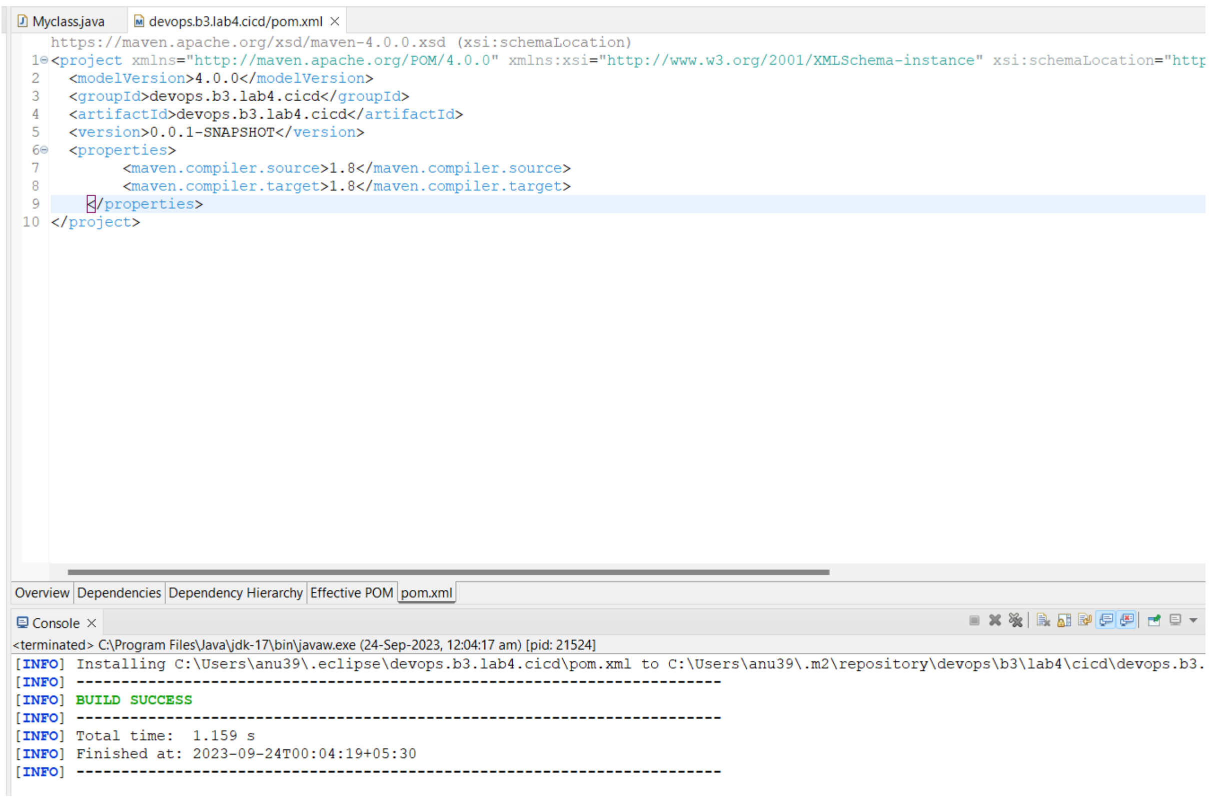
Conclusion:

In this lab exercise, you've set up a GitHub Actions workflow to build a Docker image from a Dockerfile and push it to Docker Hub. Participants should now have a basic understanding of how to automate Docker image creation and deployment using GitHub Actions. You can extend this exercise by exploring more advanced Docker features or integrating other container registries.



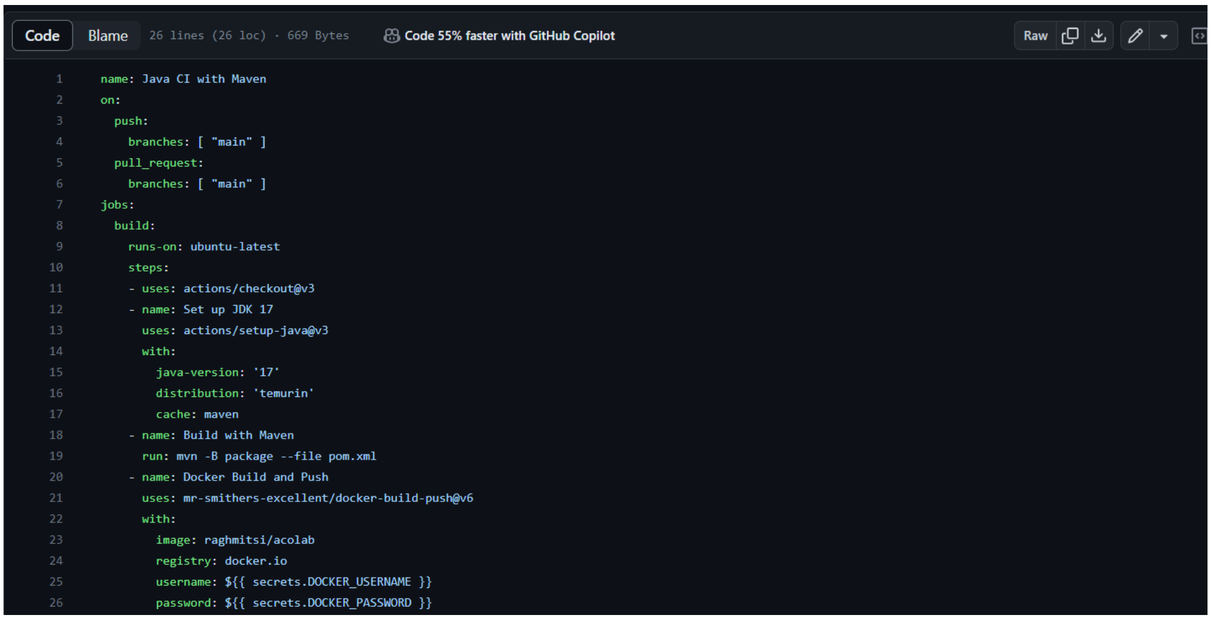
```java
package devops.b3.lab4.cicd;

public class Myclass {
	public void greeting()
	{
		System.out.println("Hello GitHub Actions");
	}
	public void testMsg()
	{
		System.out.println("This is Test Message");
	}

	public static void main(String[] args) {
		Myclass ob = new Myclass();
		ob.greeting();
		ob.testMsg ();
	}
}
```

```
<terminated> Myclass [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe  (24-Sep-2023, 12:03:33 am – 12:03:33 am) [pid: 22288]
Hello GitHub Actions
This is Test Message
```

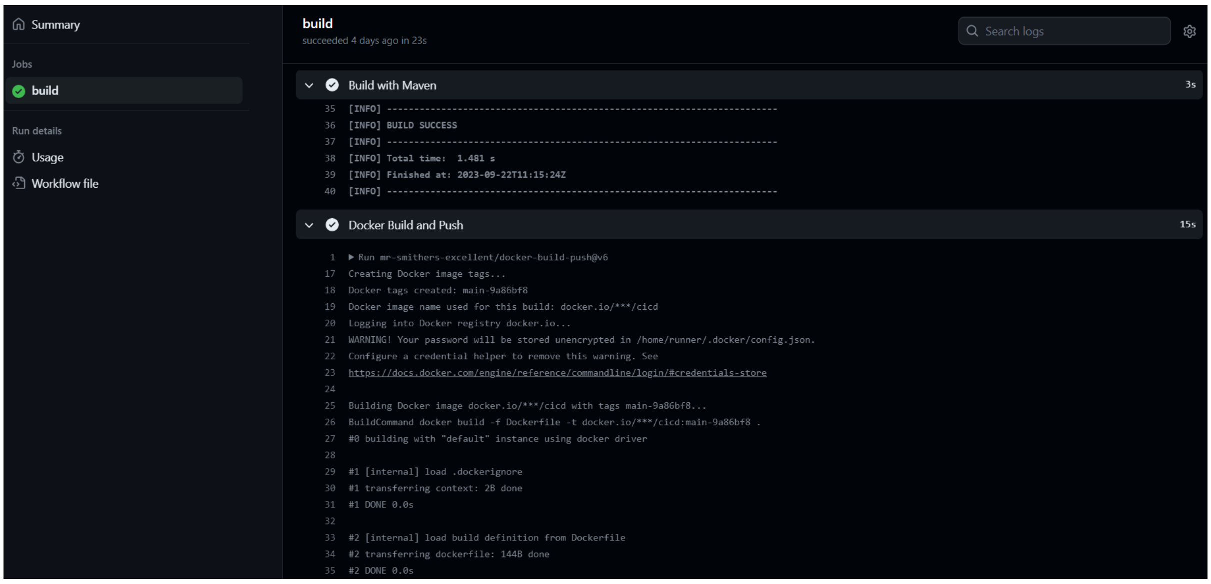


```xml
https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http
  <modelVersion>4.0.0</modelVersion>
  <groupId>devops.b3.lab4.cicd</groupId>
  <artifactId>devops.b3.lab4.cicd</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <properties>
  	<maven.compiler.source>1.8</maven.compiler.source>
  	<maven.compiler.target>1.8</maven.compiler.target>
  </properties>
</project>
```

```
<terminated> C:\Program Files\Java\jdk-17\bin\javaw.exe (24-Sep-2023, 12:04:17 am) [pid: 21524]
[INFO] Installing C:\Users\anu39\.eclipse\devops.b3.lab4.cicd\pom.xml to C:\Users\anu39\.m2\repository\devops\b3\lab4\cicd\devops.b3.
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.159 s
[INFO] Finished at: 2023-09-24T00:04:19+05:30
[INFO] ------------------------------------------------------------------------
```


Code
Blame
26 lines (26 loc) · 669 Bytes
Code 55% faster with GitHub Copilot
Raw

```yaml
name: Java CI with Maven
on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v3
    - name: Set up JDK 17
      uses: actions/setup-java@v3
      with:
        java-version: '17'
        distribution: 'temurin'
        cache: maven
    - name: Build with Maven
      run: mvn -B package --file pom.xml
    - name: Docker Build and Push
      uses: mr-smithers-excellent/docker-build-push@v6
      with:
        image: raghmitsi/acolab
        registry: docker.io
        username: ${{ secrets.DOCKER_USERNAME }}
        password: ${{ secrets.DOCKER_PASSWORD }}
```



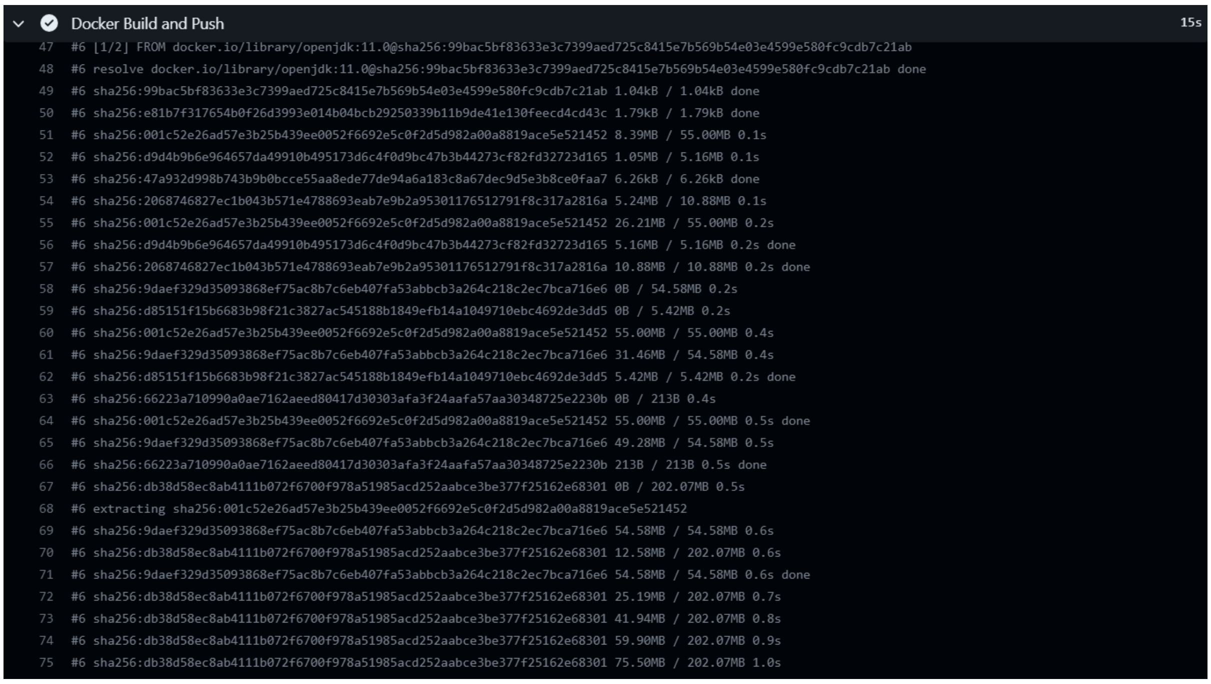
build
succeeded 4 days ago in 23s

Build with Maven 3s

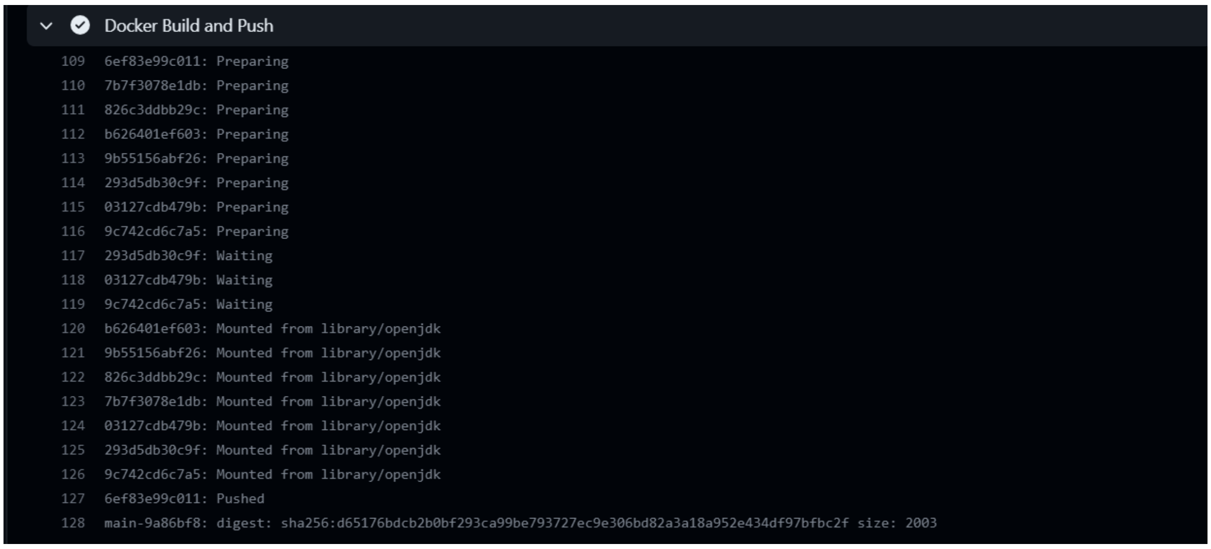
```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.481 s
[INFO] Finished at: 2023-09-22T11:15:24Z
[INFO] ------------------------------------------------------------------------
```

Docker Build and Push 15s

```
▶ Run mr-smithers-excellent/docker-build-push@v6
Creating Docker image tags...
Docker tags created: main-9a86bf8
Docker image name used for this build: docker.io/***/cicd
Logging into Docker registry docker.io...
WARNING! Your password will be stored unencrypted in /home/runner/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Building Docker image docker.io/***/cicd with tags main-9a86bf8...
BuildCommand docker build -f Dockerfile -t docker.io/***/cicd:main-9a86bf8 .
#0 building with "default" instance using docker driver

#1 [internal] load .dockerignore
#1 transferring context: 2B done
#1 DONE 0.0s

#2 [internal] load build definition from Dockerfile
#2 transferring dockerfile: 144B done
#2 DONE 0.0s
```


Docker Build and Push 15s

```
#6 [1/2] FROM docker.io/library/openjdk:11.0@sha256:99bac5bf83633e3c7399aed725c8415e7b569b54e03e4599e580fc9cdb7c21ab
#6 resolve docker.io/library/openjdk:11.0@sha256:99bac5bf83633e3c7399aed725c8415e7b569b54e03e4599e580fc9cdb7c21ab done
#6 sha256:99bac5bf83633e3c7399aed725c8415e7b569b54e03e4599e580fc9cdb7c21ab 1.04kB / 1.04kB done
#6 sha256:e81b7f317654b0f26d3993e014b04bcb29250339b11b9de41e130feecd4cd43c 1.79kB / 1.79kB done
#6 sha256:001c52e26ad57e3b25b439ee0052f6692e5c0f2d5d982a00a8819ace5e521452 8.39MB / 55.00MB 0.1s
#6 sha256:d9d4b9b6e964657da49910b495173d6c4f0d9bc47b3b44273cf82fd32723d165 1.05MB / 5.16MB 0.1s
#6 sha256:47a932d998b743b9b0bcce55aa8ede77de94a6a183c8a67dec9d5e3b8ce0faa7 6.26kB / 6.26kB done
#6 sha256:2068746827ec1b043b571e4788693eab7e9b2a95301176512791f8c317a2816a 5.24MB / 10.88MB 0.1s
#6 sha256:001c52e26ad57e3b25b439ee0052f6692e5c0f2d5d982a00a8819ace5e521452 26.21MB / 55.00MB 0.2s
#6 sha256:d9d4b9b6e964657da49910b495173d6c4f0d9bc47b3b44273cf82fd32723d165 5.16MB / 5.16MB 0.2s done
#6 sha256:2068746827ec1b043b571e4788693eab7e9b2a95301176512791f8c317a2816a 10.88MB / 10.88MB 0.2s done
#6 sha256:9daef329d35093868ef75ac8b7c6eb407fa53abbcb3a264c218c2ec7bca716e6 0B / 54.58MB 0.2s
#6 sha256:d85151f15b6683b98f21c3827ac545188b1849efb14a1049710ebc4692de3dd5 0B / 5.42MB 0.2s
#6 sha256:001c52e26ad57e3b25b439ee0052f6692e5c0f2d5d982a00a8819ace5e521452 55.00MB / 55.00MB 0.4s
#6 sha256:9daef329d35093868ef75ac8b7c6eb407fa53abbcb3a264c218c2ec7bca716e6 31.46MB / 54.58MB 0.4s
#6 sha256:d85151f15b6683b98f21c3827ac545188b1849efb14a1049710ebc4692de3dd5 5.42MB / 5.42MB 0.2s done
#6 sha256:66223a710990a0ae7162aeed80417d30303afa3f24aafa57aa30348725e2230b 0B / 213B 0.4s
#6 sha256:001c52e26ad57e3b25b439ee0052f6692e5c0f2d5d982a00a8819ace5e521452 55.00MB / 55.00MB 0.5s done
#6 sha256:9daef329d35093868ef75ac8b7c6eb407fa53abbcb3a264c218c2ec7bca716e6 49.28MB / 54.58MB 0.5s
#6 sha256:66223a710990a0ae7162aeed80417d30303afa3f24aafa57aa30348725e2230b 213B / 213B 0.5s done
#6 sha256:db38d58ec8ab4111b072f6700f978a51985acd252aabce3be377f25162e68301 0B / 202.07MB 0.5s
#6 extracting sha256:001c52e26ad57e3b25b439ee0052f6692e5c0f2d5d982a00a8819ace5e521452
#6 sha256:9daef329d35093868ef75ac8b7c6eb407fa53abbcb3a264c218c2ec7bca716e6 54.58MB / 54.58MB 0.6s
#6 sha256:db38d58ec8ab4111b072f6700f978a51985acd252aabce3be377f25162e68301 12.58MB / 202.07MB 0.6s
#6 sha256:9daef329d35093868ef75ac8b7c6eb407fa53abbcb3a264c218c2ec7bca716e6 54.58MB / 54.58MB 0.6s done
#6 sha256:db38d58ec8ab4111b072f6700f978a51985acd252aabce3be377f25162e68301 25.19MB / 202.07MB 0.7s
#6 sha256:db38d58ec8ab4111b072f6700f978a51985acd252aabce3be377f25162e68301 41.94MB / 202.07MB 0.8s
#6 sha256:db38d58ec8ab4111b072f6700f978a51985acd252aabce3be377f25162e68301 59.90MB / 202.07MB 0.9s
#6 sha256:db38d58ec8ab4111b072f6700f978a51985acd252aabce3be377f25162e68301 75.50MB / 202.07MB 1.0s
```


Docker Build and Push

```
6ef83e99c011: Preparing
7b7f3078e1db: Preparing
826c3ddbb29c: Preparing
b626401ef603: Preparing
9b55156abf26: Preparing
293d5db30c9f: Preparing
03127cdb479b: Preparing
9c742cd6c7a5: Preparing
293d5db30c9f: Waiting
03127cdb479b: Waiting
9c742cd6c7a5: Waiting
b626401ef603: Mounted from library/openjdk
9b55156abf26: Mounted from library/openjdk
826c3ddbb29c: Mounted from library/openjdk
7b7f3078e1db: Mounted from library/openjdk
03127cdb479b: Mounted from library/openjdk
293d5db30c9f: Mounted from library/openjdk
9c742cd6c7a5: Mounted from library/openjdk
6ef83e99c011: Pushed
main-9a86bf8: digest: sha256:d65176bdcb2b0bf293ca99be793727ec9e306bd82a3a18a952e434df97bfbc2f size: 2003
```