Docker cheatsheet

Dockerfile Instructions

`FROM`
Specifies the base image to use.
`RUN`
Executes commands during image build.
`CMD`
Specifies the default command to run when the container starts.
`EXPOSE`
Informs Docker of the ports the container listens on.
`ENV`
Sets environment variables.
`ADD`
Copies files/directories from source to the container filesystem.
`COPY`
Copies files/directories from source to the container filesystem (similar to ADD, but without URL or tar extraction support).
`WORKDIR`
Sets the working directory for subsequent instructions.
`USER`
Sets the user to run subsequent commands as.
`VOLUME`
Creates a mount point with the specified name and marks it as holding externally mounted volumes from native host or other containers.

Basic Docker Commands

`docker run [OPTIONS] IMAGE [COMMAND] [ARG...]`
Create and run a new container from an image. **Example:** `docker run -it ubuntu bash` (Runs Ubuntu in interactive mode)
`docker ps`
List running containers.
`docker ps -a`
List all containers (running and stopped).
`docker stop CONTAINER_ID`
Stop a running container.
`docker start CONTAINER_ID`
Start a stopped container.
`docker restart CONTAINER_ID`
Restart a container.
`docker rm CONTAINER_ID`
Remove a stopped container.
`docker exec -it CONTAINER_ID [COMMAND]`
Execute a command inside a running container. **Example:** `docker exec -it my_container bash`
`docker kill CONTAINER_ID`
Forcefully stop a running container.

Docker Images

`docker images`
List all available images.
`docker pull IMAGE_NAME`
Download an image from Docker Hub.
`docker rmi IMAGE_ID`
Remove an image.
`docker build -t IMAGE_NAME .`
Build an image from a Dockerfile in the current directory.
`docker push IMAGE_NAME`
Push an image to Docker Hub.
`docker tag SOURCE_IMAGE[:TAG] TARGET_IMAGE[:TAG]`
Tag an image for pushing to a registry.
`docker history IMAGE_ID`
Show the history of an image.

Docker Networking

`docker network ls`
List Docker networks.
`docker network create NETWORK_NAME`
Create a new Docker network.
`docker network connect NETWORK_NAME CONTAINER_ID`
Connect a container to a network.
`docker network disconnect NETWORK_NAME CONTAINER_ID`
Disconnect a container from a network.
`docker network inspect NETWORK_NAME`
Inspect a Docker network.
`docker port CONTAINER_ID`
List port mappings for a specific container.

Docker Compose

`docker-compose up`
Builds, (re)creates, starts, and attaches to containers defined in a `docker-compose.yml` file.
`docker-compose up -d`
Runs the `docker-compose up` command in detached mode (background).
`docker-compose down`
Stops and removes containers, networks, volumes, and images created by `up`.
`docker-compose ps`
Lists the containers created by `docker-compose`.
`docker-compose logs`
View the logs of the containers.
`docker-compose stop`
Stop running services without removing them.
`docker-compose start`
Start previously stopped services.
`docker-compose restart`
Restart services.
`docker-compose build`
Build or rebuild services.

Docker Volumes

`docker volume ls`
List Docker volumes.
`docker volume create VOLUME_NAME`
Create a new Docker volume.
`docker run -v VOLUME_NAME:/path/in/container IMAGE_NAME`
Mount a volume to a container.
`docker volume inspect VOLUME_NAME`
Inspect a Docker volume.
`docker volume rm VOLUME_NAME`
Remove a Docker volume.
`docker run -v /host/path:/container/path IMAGE_NAME`
Mount a host directory as a volume.

Container Resource Limits

`docker run --memory 512m IMAGE_NAME`	Limit container's memory usage to 512MB.
`docker run --cpus 2 IMAGE_NAME`	Limit container's CPU usage to 2 CPUs.
`docker update --memory 1g CONTAINER_ID`	Update container's memory limit to 1GB.
`docker stats`	Display live resource usage statistics for containers.

Docker Registry

`docker login [SERVER]`	Log in to a Docker registry (e.g., Docker Hub).
`docker logout [SERVER]`	Log out from a Docker registry.
`docker search TERM`	Search for images on Docker Hub.
`docker tag IMAGE[:TAG] REGISTRY_HOST/USERNAME/IMAGE[:TAG]`	Tag an image for a private registry.
`docker push REGISTRY_HOST/USERNAME/IMAGE[:TAG]`	Push an image to a private registry.

Docker Network Commands

`docker network create <network_name>`	Create a new Docker network. **Example:** `docker network create my_network`
`docker network ls`	List all Docker networks. **Example:** `docker network ls`
`docker network inspect <network_name>`	Inspect a Docker network to view its details. **Example:** `docker network inspect my_network`
`docker network connect <network_name> <container_name>`	Connect a container to a Docker network. **Example:** `docker network connect my_network my_container`
`docker network disconnect <network_name> <container_name>`	Disconnect a container from a Docker network. **Example:** `docker network disconnect my_network my_container`
`docker network rm <network_name>`	Remove a Docker network. **Example:** `docker network rm my_network`

Docker Volume Commands

`docker volume create <volume_name>`	Create a new Docker volume. **Example:** `docker volume create my_volume`
`docker volume ls`	List all Docker volumes. **Example:** `docker volume ls`
`docker volume inspect <volume_name>`	Inspect a Docker volume to view its details. **Example:** `docker volume inspect my_volume`
`docker run -v <volume_name>:<container_path> <image_name>`	Mount a Docker volume to a container. **Example:** `docker run -v my_volume:/data ubuntu`
`docker volume rm <volume_name>`	Remove a Docker volume. **Example:** `docker volume rm my_volume`
`docker volume prune`	Remove all unused Docker volumes. **Example:** `docker volume prune`

Docker Compose Commands

`docker-compose up`	Build, (re)create, start, and attach to containers based on docker-compose.yml. **Example:** `docker-compose up`
`docker-compose up -d`	Run containers in detached mode. **Example:** `docker-compose up -d`
`docker-compose down`	Stop and remove containers, networks, volumes, and images created by `up`. **Example:** `docker-compose down`
`docker-compose ps`	List containers. **Example:** `docker-compose ps`
`docker-compose logs <service_name>`	View output from containers. **Example:** `docker-compose logs web`
`docker-compose exec <service_name> <command>`	Execute a command in a running container. **Example:** `docker-compose exec web bash`

Docker Image Commands

`docker image history <image_name>`	Show the history of an image. **Example:** `docker image history ubuntu:latest`
`docker image tag <source_image> <target_image>`	Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE. **Example:** `docker image tag ubuntu:latest my_ubuntu:latest`
`docker image prune`	Remove unused images. **Example:** `docker image prune`
`docker image save -o <file_name>.tar <image_name>`	Save one or more images to a tar archive. **Example:** `docker image save -o ubuntu.tar ubuntu:latest`
`docker image load -i <file_name>.tar`	Load an image from a tar archive. **Example:** `docker image load -i ubuntu.tar`
`docker image import <file_name>.tar <image_name>`	Import the contents from a tarball to create an image. **Example:** `docker image import ubuntu.tar my_ubuntu:latest`

Docker Container Commands

`docker container stats <container_name>`	Display live stream of container(s) resource usage statistics. **Example:** `docker container stats my_container`
`docker container diff <container_name>`	Inspect changes to files or directories on a container's filesystem. **Example:** `docker container diff my_container`
`docker container cp <container_name>:<path> <host_path>`	Copy files/folders between a container and the local filesystem. **Example:** `docker container cp my_container:/app/data ./`
`docker container pause <container_name>`	Pause all processes within a container. **Example:** `docker container pause my_container`
`docker container unpause <container_name>`	Unpause all processes within a container. **Example:** `docker container unpause my_container`
`docker container update <container_name> --memory 512m`	Update configuration of one or more containers. **Example:** `docker container update my_container --memory 512m`

Docker System Commands

`docker system df`	Show docker disk usage. **Example:** `docker system df`
`docker system prune`	Remove all unused containers, networks, images (dangling and all untagged). **Example:** `docker system prune`
`docker system events`	Get real time events from the server. **Example:** `docker system events`
`docker info`	Display system-wide information. **Example:** `docker info`
`docker version`	Show the Docker version information. **Example:** `docker version`
`docker login [OPTIONS] [SERVER]`	Log in to a Docker registry. **Example:** `docker login docker.example.com`

ECR Authentication

Authenticate Docker CLI to AWS ECR using the AWS CLI. This command retrieves an authentication token and configures Docker to use it.

```
aws ecr get-login-password --region <region> | docker login --username AWS --password-stdin <aws_account_id>.dkr.ecr.<region>.amazonaws.com
```

- `<region>`: The AWS region where your ECR repository is located (e.g., `us-west-2`).
- `<aws_account_id>`: Your AWS account ID.

Authenticate Docker CLI to AWS ECR (alternative method):

```
aws ecr get-login --region <region> --no-include-email | sh
```

- `--no-include-email`: Omits the email address from the generated login command.

Note: Ensure the AWS CLI is configured with appropriate credentials (IAM user/role with ECR permissions) before running these commands. If you encounter issues, double-check your AWS CLI configuration and IAM permissions.

Creating ECR Repositories

Create a private ECR repository using the AWS CLI:

```
aws ecr create-repository --repository-name <repository_name> --image-scanning-configuration scanOnPush=true --image-tag-mutability MUTABLE
```

- `--repository-name`: The name of your repository.
- `--image-scanning-configuration scanOnPush=true`: Enables image scanning on push.
- `--image-tag-mutability MUTABLE|IMMUTABLE`: Configures whether image tags can be overwritten.

Create a public ECR repository (ECR Public):

```
aws ecr-public create-repository --repository-name <repository_name>
```

Tagging Docker Images

Tag your Docker image with the ECR repository URI before pushing:

```
docker tag <image_name>:<tag> <aws_account_id>.dkr.ecr.<region>.amazonaws.com/<repository_name>:<tag>
```

- `<image_name>`: The name of your local Docker image.
- `<tag>`: The tag for your Docker image (e.g., `latest`, `1.0`).
- `<aws_account_id>`: Your AWS account ID.
- `<region>`: The AWS region.
- `<repository_name>`: The name of your ECR repository.

Example:

```
docker tag my-app:latest 123456789012.dkr.ecr.us-west-2.amazonaws.com/my-app:latest
```

Pushing Images to ECR

Push the tagged Docker image to your ECR repository:

```
docker push <aws_account_id>.dkr.ecr.<region>.amazonaws.com/<repository_name>:<tag>
```

Pushing to ECR Public:

```
docker push public.ecr.aws/<public_registry_alias>/<repository_name>:<tag>
```

Replace `<public_registry_alias>` with your public registry alias.

Pulling Images from ECR

Pull an image from ECR:

```
docker pull <aws_account_id>.dkr.ecr.<region>.amazonaws.com/<repository_name>:<tag>
```

Pulling from ECR Public:

```
docker pull public.ecr.aws/<public_registry_alias>/<repository_name>:<tag>
```

Deleting Images and Repositories

Delete a specific image in an ECR repository:

```
aws ecr batch-delete-image --repository-name <repository_name> --image-ids imageTag=<tag>
```

Delete an entire ECR repository (ensure it's empty first):

```
aws ecr delete-repository --repository-name <repository_name> --force
```

`--force`: Deletes the repository even if it contains images.

Deleting images by digest:

```
aws ecr batch-delete-image --repository-name <repository_name> --image-ids imageDigest=<image_digest>
```

ECR Permissions and IAM

Example IAM policy for allowing users to push and pull images from ECR:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPullPush",
            "Effect": "Allow",
            "Action": [

"ecr:GetAuthorizationToken",

"ecr:BatchCheckLayerAvailability",

"ecr:GetDownloadUrlForLayer",

"ecr:GetRepositoryPolicy",

"ecr:DescribeRepositories",
                "ecr:ListImages",
                "ecr:BatchGetImage",

"ecr:InitiateLayerUpload",
                "ecr:UploadLayerPart",

"ecr:CompleteLayerUpload",
                "ecr:PutImage"
            ],
            "Resource": "arn:aws:ecr:
<region>:
<aws_account_id>:repository/<repository_
name>"
        }
    ]
}
```

- Replace `<region>`, `<aws_account_id>`, and `<repository_name>` with your specific values.

Important: Carefully manage ECR permissions to control who can access and modify your container images. Use IAM roles for EC2 instances or other AWS services that need to interact with ECR.