15 Scripts to Automate Docker Container Management

Link: https://blog.devops.dev/15-scripts-to-automate-docker-container-management-4bab4c3faf73

Each example comes with functioning code and detailed explanations.

1. Automatically Start All Containers

Sometimes after a system reboot or maintenance, you may want to start all stopped containers at once.

```
#!/bin/bash
# Start all stopped containers
docker start $(docker ps -aq)
```

- ‘docker ps -aq’ lists all container IDs (stopped and running).
- ‘docker start’ starts the containers by passing the IDs as arguments.

2. Stop All Running Containers

Quickly stop all currently running containers.

```
#!/bin/bash
# Stop all running containers
docker stop $(docker ps -q)
```

- ‘docker ps -q’ lists IDs of only running containers.
- ‘docker stop’ stops these containers.

3. Remove Stopped Containers

Free up space by cleaning up stopped containers.

```
#!/bin/bash
# Remove all stopped containers
docker rm $(docker ps -aq -f "status=exited")
```

- `docker ps -aq -f "status=exited"` filters stopped containers.
- ‘docker rm’ removes them.

4. Remove Dangling Images

Clear unused Docker images to save disk space.

```
#!/bin/bash
# Remove dangling images
docker rmi $(docker images -q -f "dangling=true")
```

- `docker images -q -f "dangling=true"` lists image IDs with no tags (dangling).
- ‘docker rmi’ removes these images.

5. Backup a Container’s Data

Export the filesystem of a running container to a tar file.

```
#!/bin/bash
# Backup a container's data
CONTAINER_ID=$1
BACKUP_FILE="${CONTAINER_ID}_backup_$(date +%F).tar"
docker export $CONTAINER_ID > $BACKUP_FILE
echo "Backup saved to $BACKUP_FILE"
```

- ‘docker export’ exports the filesystem of the container.
- Pass the container ID as an argument to the script.

6. Restore a Container from Backup

Recreate a container from a tar backup file.

```
#!/bin/bash
# Restore a container from a tar backup
BACKUP_FILE=$1
docker import $BACKUP_FILE restored_container:latest
```

```
echo "Container restored as 'restored_container:latest'"
```

- ‘docker import’ creates a new image from the tar file.
- The image can be used to start new containers.

7. Monitor Container Resource Usage

Display real-time stats for all running containers.

```
#!/bin/bash
# Monitor resource usage of all running containers
docker stats --all
```

- ‘docker stats’ shows real-time CPU, memory, and network stats.
- ‘--all’ includes stopped containers.

8. Restart a Container Automatically

Ensure critical containers restart after failure.

```
#!/bin/bash
# Restart a container with restart policy
CONTAINER_NAME=$1
docker update --restart always $CONTAINER_NAME
echo "$CONTAINER_NAME will now restart automatically on failure."
```

- ‘docker update --restart always’ configures the restart policy.
- Pass the container name as an argument.

9. Run a Container and Clean Up After Exit

Automatically remove a container after it stops.

```
#!/bin/bash
# Run a container and clean up
IMAGE_NAME=$1
docker run --rm $IMAGE_NAME
```

- ‘--rm’ removes the container when it stops.
- Useful for one-off tasks.

10. Check Logs of All Containers

Combine logs from multiple containers into one output.

```
#!/bin/bash
# Display logs of all containers
docker ps -q | xargs -I {} docker logs {}
```

- ‘docker ps -q’ lists running container IDs.
- ‘xargs’ passes these IDs to ‘docker logs’.

11. Auto-Prune Unused Resources

Schedule automated cleanup of unused Docker resources.

```
#!/bin/bash
# Prune unused resources
docker system prune -f --volumes
```

- ‘docker system prune’ removes unused containers, networks, and images.
- ‘--volumes’ also deletes unused volumes.

12. Update Running Containers

Recreate containers with the latest image version.

```
#!/bin/bash
# Update a running container
CONTAINER_NAME=$1
IMAGE_NAME=$(docker inspect --format='{{.Config.Image}}' $CONTAINER_NAME)
docker pull $IMAGE_NAME
docker stop $CONTAINER_NAME
docker rm $CONTAINER_NAME
docker run -d --name $CONTAINER_NAME $IMAGE_NAME
```

- ‘docker inspect’ fetches the image name of a container.
- The script pulls the latest image and recreates the container.

13. Copy Files from a Container

Extract files or directories from a container to the host.

```
#!/bin/bash
# Copy files from a container
CONTAINER_ID=$1
SOURCE_PATH=$2
DEST_PATH=$3
docker cp $CONTAINER_ID:$SOURCE_PATH $DEST_PATH
echo "Copied $SOURCE_PATH from $CONTAINER_ID to $DEST_PATH"
```

- ‘docker cp’ copies files between the container and the host.
- Pass container ID, source path, and destination path as arguments.

14. Restart All Containers

Restart all running containers quickly.

```
#!/bin/bash
# Restart all containers
docker restart $(docker ps -q)
```

- ‘docker restart’ restarts containers by their IDs.

15. List All Exposed Ports

Check the exposed ports of running containers.

```
#!/bin/bash
# List all exposed ports
docker ps --format '{{.ID}}: {{.Ports}}'
```

- ‘docker ps --format’ customizes the output to show container IDs and ports.

Feel free to tweak, experiment and customize them to your needs.

Docker

Bash

Bash Script

Programming

Revision #1
Created 7 January 2025 22:09:57 by Administrador

Updated 7 January 2025 22:11:46 by Administrador