■ Docker Commands Cheat Sheet ■

Complete Reference Guide for Docker Commands

Container Management

`docker run -d nginx`	Run container in detached mode
`docker run -it ubuntu bash`	Interactive container with terminal
`docker run -p 8080:80 nginx`	Map port 8080 to container port 80
`docker run --name myapp nginx`	Run with custom container name
`docker run -v /host:/container app`	Mount volume from host to container
`docker ps`	List running containers
`docker ps -a`	List all containers (including stopped)
`docker start CONTAINER`	Start a stopped container
`docker stop CONTAINER`	Stop a running container
`docker restart CONTAINER`	Restart a container
`docker kill CONTAINER`	Force stop a container
`docker rm CONTAINER`	Remove a container
`docker rm -f CONTAINER`	Force remove a running container

Container Information & Logs

`docker logs CONTAINER`	View container logs
`docker logs -f CONTAINER`	Follow logs in real-time
`docker logs --tail 100 CONTAINER`	Show last 100 log lines

`docker inspect CONTAINER`	Detailed container information
`docker stats`	Live resource usage statistics
`docker top CONTAINER`	Running processes in container
`docker exec CONTAINER COMMAND`	Execute command in container
`docker exec -it CONTAINER bash`	Interactive bash session

Image Management

`docker images`	List local images
`docker images -a`	List all images (including intermediate)
`docker pull IMAGE[:TAG]`	Download image from registry
`docker push IMAGE[:TAG]`	Upload image to registry
`docker build .`	Build image from current directory
`docker build -t myapp:v1.0 .`	Build image with tag
`docker build --no-cache .`	Build without using cache
`docker rmi IMAGE`	Remove an image
`docker rmi -f IMAGE`	Force remove an image
`docker tag SOURCE TARGET`	Tag an image
`docker history IMAGE`	Show image layer history
`docker image prune`	Remove unused images
`docker image prune -a`	Remove all unused images

Network & Volume Management

`docker network ls`	List networks
`docker network create NETWORK`	Create custom network
`docker network rm NETWORK`	Remove network
`docker network inspect NETWORK`	Network detailed information
`docker volume ls`	List volumes
`docker volume create VOLUME`	Create named volume
`docker volume rm VOLUME`	Remove volume
`docker volume inspect VOLUME`	Volume detailed information
`docker volume prune`	Remove unused volumes

Docker Compose

`docker-compose up`	Start all services
`docker-compose up -d`	Start services in detached mode
`docker-compose up --build`	Rebuild images and start services
`docker-compose down`	Stop and remove containers/networks
`docker-compose down -v`	Stop and remove volumes too
`docker-compose ps`	List running services
`docker-compose logs`	View logs from all services
`docker-compose logs SERVICE`	View logs from specific service
`docker-compose exec SERVICE bash`	Execute bash in service container
`docker-compose restart SERVICE`	Restart specific service
`docker-compose scale SERVICE=3`	Scale service to 3 instances

System Management & Cleanup

`docker version`	Show Docker version information
`docker info`	Display system-wide information
`docker system df`	Show Docker disk usage
`docker system prune`	Remove unused data
`docker system prune -a`	Remove all unused data
`docker system prune -a --volumes`	Remove everything unused
`docker container prune`	Remove stopped containers
`docker login`	Login to Docker registry
`docker logout`	Logout from Docker registry
`docker search TERM`	Search Docker Hub for images

Common Docker Run Options

`-d, --detach`	Run container in background
`-it`	Interactive mode with TTY
`-p, --publish HOST:CONTAINER`	Publish container port to host
`-v, --volume HOST:CONTAINER`	Bind mount a volume
`--name NAME`	Assign name to container
`-e, --env KEY=VALUE`	Set environment variables
`--rm`	Remove container when it exits
`-m, --memory LIMIT`	Memory limit (e.g., 512m, 2g)
`--cpus NUMBER`	CPU limit (e.g., 0.5, 2.0)
`--restart POLICY`	Restart policy (no/always/unless-stopped)
`--network NETWORK`	Connect to specific network

`-w, --workdir PATH`	Set working directory

Common Dockerfile Instructions

`FROM image:tag`	Specify base image
`WORKDIR /path`	Set working directory
`COPY src dest`	Copy files from host to image
`ADD src dest`	Copy files (supports URLs & archives)
`RUN command`	Execute command during build
`ENV KEY=VALUE`	Set environment variable
`EXPOSE port`	Document port usage
`USER user:group`	Set user for subsequent commands
`CMD ["cmd", "arg1"]`	Default command to run
`ENTRYPOINT ["cmd"]`	Configure container executable
`VOLUME ["/data"]`	Create mount point
`LABEL key=value`	Add metadata to image

■ Pro Tips & Best Practices:

• Use specific image tags instead of 'latest' in production
• Use .dockerignore to exclude unnecessary files from build context
• Multi-stage builds help reduce final image size
• Use 'docker run --rm' for temporary containers
• Regularly clean up with 'docker system prune'
• Name your containers and volumes for easier management
• Use volumes for persistent data, not container filesystem
• Always check container logs when troubleshooting
• Use healthchecks in production deployments
• Keep containers stateless and configuration external