

Dockerfiles

Adfinis**sy**Group

Be smart. Think open source.

Dockerfiles

What is a Dockerfile?

Build instructions for Docker images

Each command creates a layer in the image

```
FROM centos:7
LABEL maintainer="foo.bar@example.com"
LABEL maintainer.name="Foo Bar"
LABEL maintainer.email="foo.bar@example.com"
LABEL com.example.version="0.0.2"

RUN yum install -y \
    curl \
    gzip \
    tar \
    && yum clean all

COPY docker-entrypoint.sh /usr/local/bin/

ENTRYPOINT ["docker-entrypoint.sh"]

CMD ["bash"]
```

Docker build command

Build an image from a Dockerfile in a directory

```
docker build -t imagename:tag .
```

Best Practice: Always specify a name and tag when building an image

Docker build context

All files in the directory where the Dockerfile resides are sent to the Docker daemon as the **build context**

To exclude specific files create a **.dockerignore** file

```
# exclude README.md
README.md
# exclude *.pyc files
**/*.pyc
# exclude .git directory
.git/
```

Which commands are available?

FROM

Build upon this Docker image

```
FROM centos:7
```

Best Practice: Always specify a tag!

LABEL

Add metadata to your Docker image

```
LABEL maintainer="foo.bar@example.com"  
LABEL maintainer.name="Foo Bar"  
LABEL maintainer.email="foo.bar@example.com"  
LABEL com.example.version="0.9"  
LABEL desc="Even text spanning \  
multiple lines is possible"
```

MAINTAINER (deprecated)

Add maintainer information to your Docker image

```
MAINTAINER Foo Bar <foo.bar@example.com>
```

This command has been deprecated in favor of LABEL

RUN

Run a command in the build container

```
RUN yum install -y \  
curl \  
tar \  
gzip \  
&& yum clean all
```

Best Practice: Use as few `RUN` instructions as possible

Best Practice: Clean up temporary data at the end of `RUN`

RUN

Yes this is ugly, but good practice

```
RUN mkdir -p /usr/src/things \  
&& wget http://example.com/big.tar.xz \  
&& tar -xJf big.tar.xz -C /usr/src/things \  
&& make -C /usr/src/things all \  
&& rm -f big.tar.xz
```

RUN

Still ugly, but no `rm` required

```
RUN mkdir -p /usr/src/things \  
&& curl -SL http://example.com/big.tar.xz \  
| tar -xJC /usr/src/things \  
&& make -C /usr/src/things all
```

ENV

Set a environment variable used during build and in the running container

```
ENV MARIADB_MAJOR 10.1
ENV MARIADB_VERSION 10.1.17
RUN { echo '[mariadb]; \
      echo 'name = MariaDB; \
      echo "baseurl = http://yum.mariadb.org/$MARIADB_MAJOR/centos7-amd64"; \
      echo 'gpgkey = https://yum.mariadb.org/RPM-GPG-KEY-MariaDB; \
      echo 'gpgcheck = 1; \
    } > /etc/yum.repos.d/MariaDB.repo
RUN yum install -y \
    MariaDB-server-$MARIADB_VERSION
&& yum clean all
```

Best Practice: Use it to define versions of packages

Best Practice: Define app behaviour (e.g. PGDATA)

COPY

Copy a file/dir from the **build context** to the Docker image

```
COPY src /var/www/html/
```

ADD

Extended variant of **COPY**

Able to e.g. extract tar files and fetch remote sources

```
FROM scratch  
ADD rootfs.tar.xz /
```

Best Practice: Only use it if you really need it

See [ADD](#) or [COPY](#) for more information!

EXPOSE

Mark a local port in the container for external exposure

```
EXPOSE 80 443
```

Best Practice: Only expose required ports

VOLUME

Mark a directory that will contain persistent data

Data in the image will be copied to the volume

```
VOLUME /var/lib/mysql
```

USER

Execute all following commands as this user

```
USER tomcat
```

Best Practice: Avoid switching users multiple times

CMD

Command to be run when the container is being started

```
CMD ["apache2", "-DFOREGROUND"]
```

Best Practice: Start the service for service containers, specify a shell for all other purposes

ENTRYPOINT

Set the main command that the container runs

The command configured in **CMD** will be appended

```
COPY docker-entrypoint.sh /usr/local/bin/  
ENTRYPOINT ["docker-entrypoint.sh"]  
CMD ["mysql"]
```

Best Practices: Use for wrapper scripts preparing services (MariaDB, PostgreSQL, ...)

For more information see [ENTRYPOINT](#)

ENTRYPOINT

This is an example entrypoint wrapper script for PostgreSQL

```
#!/bin/bash
set -e

if [ "$1" = 'postgres' ]; then
  chown -R postgres "$PGDATA"

  if [ -z "$(ls -A "$PGDATA")" ]; then
    gosu postgres initdb
  fi

  exec gosu postgres "$@"
fi

exec "$@"
```

More questions?

See the full documentation at <https://docs.docker.com/engine/reference/builder/>

Feel Free to Contact Us

www.adfinis-sygroup.ch

[Tech Blog](#)

[GitHub](#)

info@adfinis-sygroup.ch

[Twitter](#)

