Dockerfile Config Cheatsheet

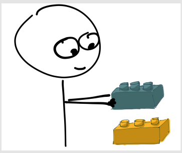
Common configuration options for building your own Docker images

FROM <image>

```
# Examples:

    FROM alpine
    FROM node:4.6.0
    FROM rabbitmq:management
```

Set the base image from which this Dockerfile will build. This may be a linux distribution, such as “alpine” or an application platform such as “node: 4.6.0”

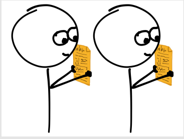


COPY <src> <dest>

```
# Examples:

    COPY /my/app /var/app/
    COPY app/package.json /var/app/
    COPY cmd.sh /cmd.sh
```

Copy any file or folder from the working directory (“build context”) of the local system, into the Docker image, during the build process.

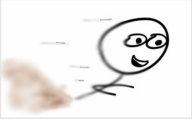


RUN <cmd>

```
# Examples:

    RUN mkdir -p /var/app
    RUN npm install -g nodemon
    RUN chmod +x /my-script.sh
```

Run any arbitrary command in the image, during the build process. Each “RUN” creates a new intermediate Docker image, used for cacheing builds and improving build time.

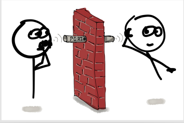


EXPOSE <port #>

```
# Examples:

    EXPOSE 3000
    EXPOSE 5672 15672
    EXPOSE 80 443
```

Tell Docker that the container will listen on the specified port numbers. Allows these ports to be mapped to the host system with the “-p” or “-P” flags of the “docker run” command.



WORKDIR <folder name>

```
# Example:

    WORKDIR /var/app
```

Set the working directory (pwd) for the command that is executed, when a container is run from this image.

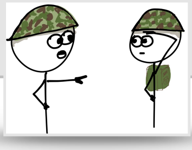
CMD [“<command>”]

```
# Examples:

    CMD [“node”, “index.js”]
    CMD [“grunt”, “watch”]
    CMD [“/my-script.sh”]
```

The command to execute when a container, made from this image, is run.

See **this WatchMeCode screencast** for a complete discussion of ENTRYPOINT and CMD.



For more configuration options and examples, see **the complete Dockerfile reference**.

WatchMeCode.net