Lesson 2. Build your own image

Dockerfile

A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image.

Here is the format of the Dockerfile:

```
# Comment
INSTRUCTION arguments
```

> Visit https://docs.docker.com/engine/reference/builder/ to read more.

Executing the instructions

https://github.com/extsoft/jcat - a repository with a Dockerfile

```
docker build --help
docker build git@github.com:extsoft/jcat.git
```

> Read more on https://docs.docker.com/engine/reference/commandline/build/.

A name of an image

Sample image https://store.docker.com/community/images/extsoft/jcat with several tags.

Terminology

```
extsoft/jcat:0.2.0
^^^^^^^^^^^^ ^^^^^
 repository : tag
^^^^^^^^^^^^^^^^^
       name
```

Register	Sample repository
Docker HUB	`extsoft/jcat`
Private	`docker.my-company.com/extsoft/jcat`

Add name to existing image

```
docker tag <current name or ID> extsoft/jcat:latest
```

Build with a name

```
docker build --tag extsoft/jcat:0.3.1 git@github.com:extsoft/jcat.git
```

Instructions

`FROM`

Almost always the first instruction in the Dockerfile, define a base image.

```
FROM docker/whalesay:latest
```

How to select a base image? By minimal size!

Name	Size, ~ MB	Link
`scratch`	`0`	https://docs.docker.com/engine/userguide/eng-image/baseimages
`busybox`	`1.13`	https://store.docker.com/images/busybox
`alpine`	`3.97`	https://store.docker.com/images/alpine
`debian`	`100`	https://store.docker.com/images/debian
`ubuntu`	`120`	https://store.docker.com/images/ubuntu
`centos`	`193`	https://store.docker.com/images/centos
`fedora`	`231`	https://store.docker.com/images/fedora

> Read more on https://docs.docker.com/engine/reference/builder/#from.

`RUN`

Execute any commands in a new layer on top of the current image and commit the results.

```
FROM docker/whalesay:latest
RUN apt-get -y update && apt-get install -y fortunes
```

> Read more on https://docs.docker.com/engine/reference/builder/#run.

`CMD`

Provide defaults for an executing container.

```
FROM docker/whalesay:latest
RUN apt-get -y update && apt-get install -y fortunes
CMD /usr/games/fortune -a | cowsay
```

```
docker build -t magic .
docker run magic
docker run magic echo "My message"
```

> Read more on https://docs.docker.com/engine/reference/builder/#cmd.

`SHELL`

Set the default shell used for the shell form of commands to be overridden.

The default shell on Linux is `["/bin/sh", "-c"]`, and on Windows is `["cmd", "/S", "/C"]`.

```
SHELL ["/bin/zsh","-c"]
RUN echo "I am a shell form"  # == /bin/bash -c "echo \"I am a shell form\""
RUN ["echo", "-e", "I am an exec form"]
```

`RUN`, `ENTRYPOINT`, `CMD`, `COPY`, `ADD` support `shell` and `exec` form of a command.

> Read more on https://docs.docker.com/engine/reference/builder/#shell.

`ENTRYPOINT`

Configure a container that will run as an executable.

Just `ENTRYPOINT` usage:

```
FROM docker/whalesay:latest
ENTRYPOINT ["echo", "Go to"]
```

```
docker build -t magic .
docker run -it magic home
docker run -it magic
docker run -it --entrypoint bash magic
```

`ENTRYPOINT` & `CMD` together:

```
FROM docker/whalesay:latest
ENTRYPOINT ["echo", "Go to"]
CMD ["home!"]
```

```
docker build -t magic .
docker run -it magic
docker run -it magic party
```

> Read more on https://docs.docker.com/engine/reference/builder/#entrypoint.

`WORKDIR`

Set the working directory for any of `RUN`, `CMD`, `ENTRYPOINT`, `COPY` and `ADD` instructions that follow it in the Dockerfile.

```
FROM docker/whalesay:latest
RUN pwd
WORKDIR /my/app/dir
RUN pwd
```

```
docker build -t magic .
```

`LABEL`

Add metadata to an image.

```
FROM docker/whalesay:latest
LABEL author="Dmytro Serdiuk"
LABEL version=2 \
      commit=fetgeti84
```

```
docker build -t magic .
docker inspect magic | less
```

> Read more on https://docs.docker.com/engine/reference/builder/#label.

`EXPOSE`

Inform Docker that the container listens on the specified network ports at runtime.

> `EXPOSE` does not make the ports of the container accessible to the host. Use `-p` option in docker run command.

```
FROM docker/whalesay:latest
EXPOSE 80/tcp
EXPOSE 81/udp
EXPOSE 8080
```

```
docker build -t magic .
docker inspect magic | less
```

> Read more on https://docs.docker.com/engine/reference/builder/#expose.

`COPY` & `ADD`

Copy new files or directories from `<src>` and adds them to the filesystem of the container at the path `<dest>`.

```
# file to file
COPY conf.txt conf
# file to directory
COPY conf.txt conf/
# files to directory
COPY a b c d ./
# URL to directory
ADD https://down.my.com/app.tag.gz conf/
```

> Read more on https://docs.docker.com/engine/reference/builder/#copy and https://docs.docker.com/engine/reference/builder/#add.

ENV

Set the environment variable `<key>` to the value `<value>`.

```
FROM docker/whalesay:latest
ENV foo 3
ENV bar1=1 bar2=2
CMD echo "foo=$foo bar1=$bar1 bar2=$bar2"
```

```
docker build -t magic .
docker inspect magic | less
docker run -it magic
docker run -it --env foo=42 --env bar1=42 magic
```

> Read more on https://docs.docker.com/engine/reference/builder/#env.

`VOLUME`

Create a mount point with the specified name and marks it as holding externally mounted volumes from native host or other containers.

```
FROM docker/whalesay:latest
VOLUME /data
VOLUME /secrets
```

```
docker build -t magic .
docker inspect magic | less
```

> Reade more on https://docs.docker.com/engine/reference/builder/#volume.

`ARG`

Define a variable that users can pass at build-time to the builder with the docker build command using the `--build-arg <varname>=<value>` flag.

```
ARG version=1
FROM mybase:${version}
ARG fooid
ENV foo ${fooid}
```

`FROM` instructions support variables that are declared by any `ARG` instructions that occur before the first `FROM`.

> Read more on https://docs.docker.com/engine/reference/builder/#arg and on https://docs.docker.com/engine/reference/builder/#understand-how-arg-and-from-interact.

`USER`

Set a user name or UID to use when running the image and for any `RUN`, `CMD` and `ENTRYPOINT` instructions that follow it in the Dockerfile.

```
USER <user>[:<group>]
USER <UID>[:<GID>]
```

> Read more on https://docs.docker.com/engine/reference/builder/#user.

`ONBUILD`

Add to the image a trigger instruction to be executed at a later time, when the image is used as the base for another build. The trigger will be executed in the context of the downstream build, as if it had been inserted immediately after the FROM instruction in the downstream Dockerfile.

```
ONBUILD [INSTRUCTION]
```

> Read more on https://docs.docker.com/engine/reference/builder/#onbuild.

`HEALTHCHECK`

Tell Docker how to test a container to check that it is still working.

```
HEALTHCHECK [OPTIONS] CMD command
```

The options that can appear before `CMD` are:

- `--interval=DURATION`
- `--timeout=DURATION`
- `--start-period=DURATION`
- `--retries=N`

> Read more on https://docs.docker.com/engine/reference/builder/#healthcheck.

`STOPSIGNAL`

Set the system call signal that will be sent to the container to exit.

```
STOPSIGNAL signal
```

> Read more on https://docs.docker.com/engine/reference/builder/#stopsignal.

Build cache

```
FROM ubuntu
RUN apt-get update
WORKDIR /app
RUN touch app && pwd && ls -lh
```

```
docker build .
docker build .
docker build --no-cache .
```

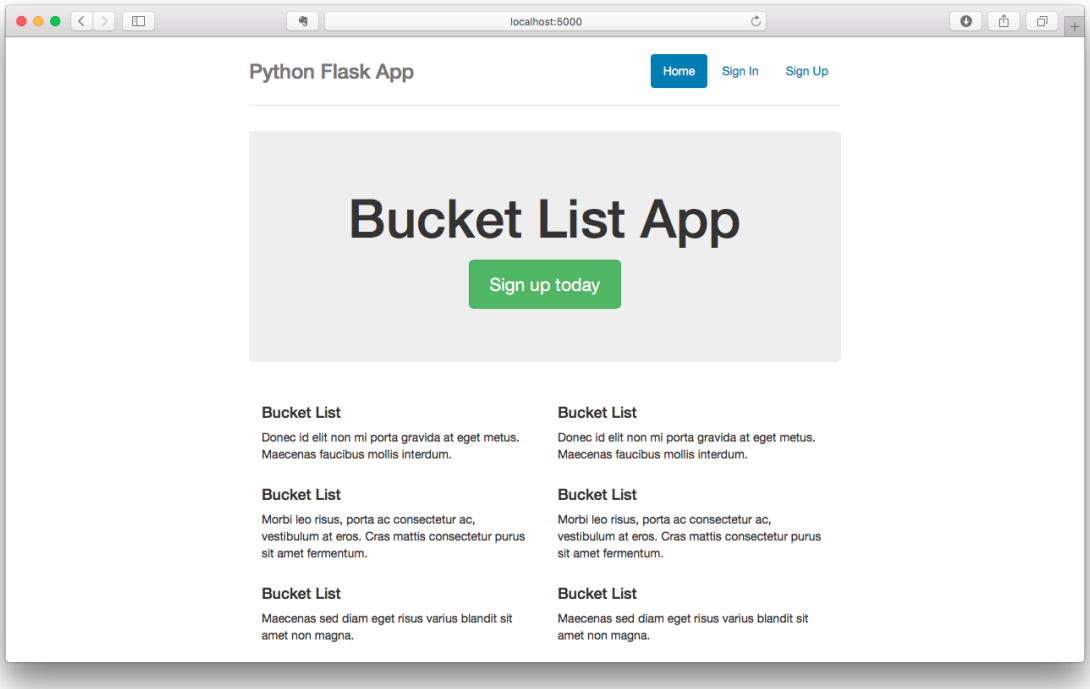
Homeworks

Please send the results of homeworks as an email.
Please use the following template:

- *Subject:* `[Docker] Homework 2`
- *To:* `trainer's email`
- *Body:* `your homework as a plain text - NO ATTACHMENTS!!!`

Homework 2.1 (mandatory)

Using sources and instruction from `homework-2.zip`, you need to create `Dockerfile` which will run an app in the container. If you build an image in a correct way, you will see the following picture in the browser:



Finally, push your image to your repository called `flask-example` with `2.1` tag.

Send for review:

1. a `Dockerfile`
2. a command to run your image

Homework 2.2 (optional)

Prepare a `Dockerfile` which uses `ADD` instruction to download some archive (or binary file) from the URL and send it for review.

Homework 2.3 (optional)

You need to create an image based on `docker/whalesay:latest`. The container has to have configured executable which reads a message from a file and pass it to `cowsay` program. Please push this image to repository called `whalesay` with `2.1` tag.

Send for review:

1. a `Dockerfile`
2. a command to run your image