HACK REACTOR

Docker 101 Demo

1. Install Docker. Start Docker. Create a Dockerfile and .dockerignore file in your root directory. CD into that directory.

➔ Refer to the Docker official docs to get started.

2. Build a Docker image

➔
```
docker build -t kapolyak/dockerdemo .
```

-t sets the image tag. Remember to precede the tag with your docker username.
. this period is not a typo! It tells docker to include everything in this directory in the image.

3. Run the Docker image

➔
```
docker run -p 80:3030 -d kapolyak/dockerdemo
```

-p sets the relationship between a port on the host machine (where docker is installed) and a port on the docker container.
-d runs the image in detached head mode, so it will continue running when you exit the terminal window.

4. Push the Docker image to Docker Hub.

➔
```
docker push kapolyak/dockerdemo
```

We are pushing the image to dockerhub, not the container. However, we must build a container before pushing the image.

5. Pull the Docker image to the host machine.

➔
```
docker pull kapolyak/dockerdemo
```

6. View all Docker images

➔
```
docker images
```

7. Run a Docker image (create a container)

➜ `docker run -p 80:3030 -d kapolyak/dockerdemo`

8. Show running Docker containers

➜ `docker ps`

9. Show logs generated by code running in a container

➜ `docker logs [container id]`

Pro-tip: you only need to type the first three characters of a container ID to reference that container.

Some important files:

Dockerfile

The Dockerfile, placed in the root directory of your project, provides instructions for Docker to execute when building an image. Below is an example:

```
FROM node:8

# Create app directory
WORKDIR /usr/src/app

# Install app dependencies
# A wildcard is used to ensure both package.json AND
package-lock.json are copied
# where available (npm@5+)
COPY package*.json ./

RUN npm install
# If you are building your code for production
# RUN npm install --only=production

# Bundle app source
COPY . .
EXPOSE 3001
```

```
CMD [ "npm", "start" ]
```

.dockerignore
This file sounds like, and acts like, .gitignore - your image will not include any files listed in this file. For example:

```
node_modules
Npm-debug.log
```

Docker system prune --all

Docker is non-destructive and won't delete any containers you have previously built or images you have loaded. This can quickly hog your machine's storage as these containers tend to be large. There are a few different ways to clean the system, but this is the most useful one during development. Here is Docker's own warning regarding this command:

> WARNING! This will remove:
> - all stopped containers
> - all networks not used by at least one container
> - all dangling images
> - all build cache