Создаём image с помощью Dockerfile

1)-FROM alpine

2)-RUN apk update && apk add nano && apk add openjdk17

3)-ADD Hello.java Hello.java

4)-ENTRYPOINT javac Hello.java && java Hello

1)-Create our image using alpine as a base\_image

2)-The commands that we want to be executed by alpine. Whatever you want to run in terminal

3)-Add the file to the alpine (it is copying the files from host to container, we are doing it to be able to use them, since alpine originally does not have these files and we need them).

4)-It is what is to be done after all the settings are finished

**docker build -t alpine\_nano\_java17\_hello\_using\_dockerfile C:\forExperements\alpine\_java\_nano** ->creating the image

-t ->option for to give the name to image we create.

alpine\_nano\_java17\_hello\_using\_dockerfile ->name of image we create.

C:\forExperements\alpine\_java\_nano ->directory where files for image are **located** (including Dockerfile)

**C:\Users\user>docker run alpine\_nano\_java17\_hello\_using\_dockerfile** -> to build the container,

I like to be a programmer

C:\Users\user> -> outcome.

**!!! IMPORTANT !!!** The alpine did not have been executed in intractable mode, because in a dockerfile indicated entrypoint which is doing something, I think it does not matter what it is exactly doing the point is that it is indicated in dockerfile, so the dockers container will do exactly what is indicated in it. Knowing this if we give the command:

**C:\Users\user>docker run -it alpine\_nano\_java17\_hello\_using\_dockerfile** (including options -it of interactive mode it will not work, the output will be the same as without -it. *In order to run container in interactive mode we need to override the entrypoint*:

**C:\Users\user>docker run -it --entrypoint=/bin/sh alpine\_nano\_java17\_hello\_using\_dockerfile**