



EyeSeeTea
ICT for human beings

Malaria DHIS2 dockers

End-User guide

Created: Sep 20, 2019

Last update: May 26, 2020

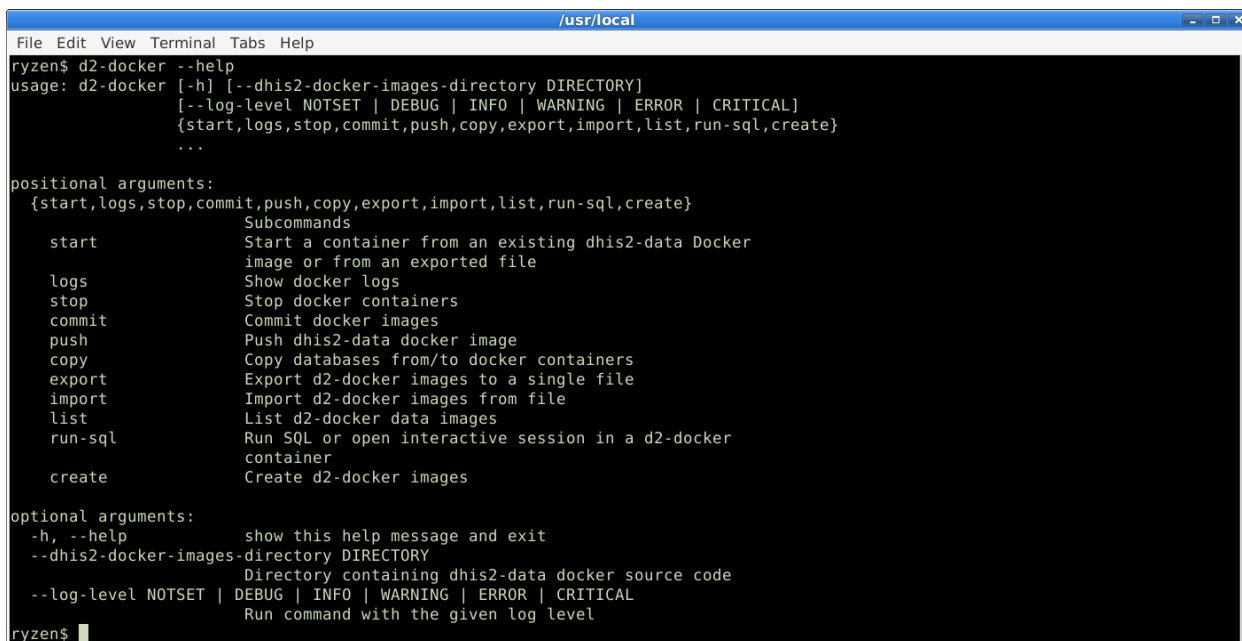
ABOUT d2-docker	3
TECHNICAL REQUIREMENTS	4
COMMANDS SYNTAX	5
TYPICAL EXAMPLES	6
Start a DHIS2 instance from Docker Hub (network needed)	6
Start a DHIS2 instance from a file (no network needed)	6
Cleaning-up	7

ABOUT d2-docker

D2-docker is a wrapper over *docker* and *docker-compose* to manage DHIS2 server instances from the command line.

The command-line interface resembles that of *docker-compose*, with a set of custom command needed to interact with DHIS2 instances. An advanced user which is familiar with the docker infrastructure may still use the *docker* command to interact with the base infrastructure (containers, images, volumes, and so on) should they need to.

License: GNU GPL v3.



```
File Edit View Terminal Tabs Help
/usr/local
ryzen$ d2-docker --help
usage: d2-docker [-h] [--dhis2-docker-images-directory DIRECTORY]
                [--log-level NOTSET | DEBUG | INFO | WARNING | ERROR | CRITICAL]
                {start,logs,stop,commit,push,copy,export,import,list,run-sql,create}
                ...

positional arguments:
  {start,logs,stop,commit,push,copy,export,import,list,run-sql,create}
    Subcommands
    start          Start a container from an existing dhis2-data Docker
                   image or from an exported file
    logs           Show docker logs
    stop           Stop docker containers
    commit         Commit docker images
    push           Push dhis2-data docker image
    copy           Copy databases from/to docker containers
    export         Export d2-docker images to a single file
    import         Import d2-docker images from file
    list           List d2-docker data images
    run-sql        Run SQL or open interactive session in a d2-docker
                   container
    create         Create d2-docker images

optional arguments:
  -h, --help            show this help message and exit
  --dhis2-docker-images-directory DIRECTORY
                        Directory containing dhis2-data docker source code
  --log-level NOTSET | DEBUG | INFO | WARNING | ERROR | CRITICAL
                        Run command with the given log level
ryzen$
```

For instructions on how to configure and install d2-docker, please refer to the [installation manual](#).

TECHNICAL REQUIREMENTS

D2-docker has been tested on **GNU/Linux, Windows 10** and **macOS**.

A typical DHIS2 instance will **take around 4-8GB of RAM**, so make sure you have that amount of free memory for each server you are planning to run simultaneously.

COMMANDS SYNTAX

D2-docker uses a single executable with subcommands (start, stop, ...) with this structure:

```
$ d2-docker [GLOBAL_OPTIONS] COMMAND [COMMAND_OPTIONS]
```

The complete interface:

```
$ d2-docker --help
usage: d2-docker [-h] [--dhis2-docker-images-directory DIRECTORY]
                [--log-level NOTSET | DEBUG | INFO | WARNING | ERROR | CRITICAL]
{start,logs,stop,rm,commit,push,pull,copy,export,import,list,run-sql,create,upgrade}
    ...

positional arguments:
  {start,logs,stop,rm,commit,push,pull,copy,export,import,list,run-sql,create,upgrade}
    Subcommands
    start      Start a container from an existing dhis2-data Docker
               image or from an exported file
    logs       Show docker logs
    stop       Stop docker containers
    rm         Remove dhis2-data docker images/containers
    commit     Commit d2-docker data image
    push       Push dhis2-data docker image
    pull       Pull dhis2-data docker image
    copy       Copy databases from/to docker containers
    export     Export d2-docker images to a single file
    import     Import d2-docker images from file
    list       List d2-docker data images
    run-sql    Run SQL or open interactive session in a d2-docker
               container
    create     Create d2-docker images
    upgrade    Upgrade DHIS2 version on core+data containers/images

optional arguments:
  -h, --help            show this help message and exit
  --dhis2-docker-images-directory DIRECTORY
                        Directory containing dhis2-data docker source code
  --log-level NOTSET | DEBUG | INFO | WARNING | ERROR | CRITICAL
                        Run command with the given log level
```

TYPICAL EXAMPLES

Start a DHIS2 instance from Docker Hub (network needed)

You can start a **dhis2-data** image existing on Docker Hub, an example:

```
$ d2-docker start -d eyeseetea/dhis2-data:2.30-ento -p 8080
```

This will pull some images from *hub.docker.com* (only the first time), and it will start a DHIS2 instance on the background (*-d / --detach*) on port 8080 (if the port option is not specified, it will use the first free port starting at 8080). If the detach option is not used, you may stop the server pressing *Control + C*.

You can check all the running instances with the **list** command.

```
$ d2-docker list
```

You can run multiple instances, but only one instance for a specific image.

WARNING: Any change you make in the server will be lost on the next run of the Docker image. To save the data of an active instance, you must commit it:

```
$ d2-docker commit
```

This change will affect only your local machine, the remote image in Docker Hub will remain untouched. Finally, stop the instance when finished:

```
$ d2-docker stop
```

Sometimes it might be useful to just download an image without starting it yet. In that case, you can use pull command:

```
$ d2-docker pull eyeseetea/dhis2-data:2.30-ento
```

Start a DHIS2 instance from a file (no network needed)

If you received a *.tgz* (or *.tar.gz*) bundle file containing all the docker images, import it:

```
$ d2-docker import dhis2-ento.tgz
[d2-docker:INFO] Load images from file: dhis2-ento.tgz
[d2-docker:DEBUG] Run: docker load -i dhis2-ento.tgz
Loaded image: eyesetea/dhis2-data:2.30-ento
Loaded image: mdillon/postgis:10-alpine
Loaded image: jwilder/nginx-proxy:alpine
Loaded image: eyesetea/dhis2-core:2.30
```

And now start the *dhis2-data* image:

```
$ d2-docker start -d eyesetea/dhis2-data:2.30-ento
```

Create a copy of a docker

If you want to copy a docker, with its DB and installed apps, just to generate a backup, to avoid modifying an existing image or because you want to extract the database, the copy command is the one you need to use. You can copy an image into a new one (beware, if you provide the name of an existing image it will replace it):

```
$ d2-docker copy eyesetea/dhis2-data:2.30-ento eyesetea/dhis2-data:2.30-ento-copy
```

You can also copy to a folder (just providing a name that cannot be recognized as a docker image, so without xxx/yyy:zzz format):

```
$ d2-docker copy eyesetea/dhis2-data:2.30-ento dhis2-data-2.30-ento-copy
```

That will generate a folder called “dhis2-data-2.30-ento-copy” with the apps and the database inside

You can copy back from a folder to an image with

```
$ d2-docker copy dhis2-data-2.30-ento-copy eyesetea/dhis2-data:2.30-ento-copy
```

That will generate an image called “eyesetea/dhis2-data:2.30-ento-copy” using the database and apps contained in the folder.

Upgrading the DHIS2 version

There's an advanced use of d2-docker that is, however, very useful for certain common situations. In some scenarios, an organization plans to migrate its DHIS2 server to a new version. As you know, DHIS2 is, since 2.30 version, using [Flyway](#), which controls the DB migrations between one DHIS2 version and the following one. To do so, DHIS2 core developers embed inside the war file, the scripts that need to be run against the DB, to make the proper upgrade between each version.

In d2-docker we have added a command called upgrade, that can be used in combination with some scripts to prepare some upgrades that might be delicate due to some specs of your organization. A typical example is the 2.31 → 2.32 upgrade, where DHIS2 started using geometry data type from postgresql databases, producing some deletions of coordinates if the DB is not previously prepared for the Flyway script upgrade.

```
usage: d2-docker upgrade [-h] [--core-image-suffix IMAGE]
                        [--from-version VERSION] [--to-version VERSION]
                        --from IMAGE --to IMAGE [-r] [--migrations DIRECTORY]
                        [-p N]
```

optional arguments:

-h, --help	show this help message and exit
--core-image-suffix IMAGE	Image core name suffix
--from-version VERSION	Source DHIS2 version
--to-version VERSION	Destination DHIS2 version
--from IMAGE	Source image
--to IMAGE	Destination image data name
-r, --keep-running	Keep last image running
--migrations DIRECTORY	Directory with migration scripts
-p N, --port N	DHIS2 instance port

The most basic upgrade (without applying any script and just creating all intermediate images) would be performed by executing:

```
$ d2-docker upgrade --from eyesetea/dhis2-data:2.30-ento --to eyesetea/dhis2-data:2.31-ento
```

That will create 2.31 version of the image, downloading the latest 2.31 war file from UiO CI server

But if you want to do something more complex, you can create a folder structure as follows:

```
→ upgrade
  → 2.31
    → dhis.war
    → drop-views.sql
  → 2.32
    → dhis.war
    → drop-views.sql
    → orgunits-geometry-fix.sql
    → script_to_execute_before_tomcat.sh
    → post_script_to_execute_after_tomcat.sh
```

--migrations parameter will recognize:

- Any "version" folder and will consider that it is what has to be applied for the upgrade to that version
- Inside each folder, if a dhis.war file is found, instead of downloading it from CI server, it will use that one
- Inside each folder, if .sql files are found, they will be executed against the DB before the tomcat is run
- Inside each folder, if .sh files are found, they will be executed after the sql files, but before running the tomcat, unless the script name starts by "post" in which case it will be executed after tomcat is running

With that folder, a typical execution would be:

```
$ d2-docker upgrade --from eyesetea/dhis2-data:2.30-ento --to eyesetea/dhis2-data:2.31-ento
--migrations upgrade
```

Cleaning-up

After some time working with d2-docker images, it's possible that you have plenty of downloaded images you will very unlikely use in the future. When that happens, deleting old images becomes handy. That can be done with the rm command:

```
$ d2-docker rm eyesetea/dhis2-data:2.30-ento-todelete
```

(remember this will only delete the image locally, if you want to delete it in dockerhub you will have to use its web interface)

Docker infrastructure (images, networks, containers, volumes) takes up a lot of hard-disk space. Those are some commands to do some cleaning-up:

To remove all local volumes not used by at least one container:

```
$ docker volume prune
```

Remove all stopped containers:

```
$ docker container prune
```

Remove all dangling images (the temporal images that have <none> on its name/tag):

```
$ docker image prune
```

[WARNING: Dangerous operation] To delete all stopped containers, networks, volumes, images and cache:

```
$ docker system prune -a --volumes
```