Docker based installation

- Precondition on host system
- License agreement
- Install translate5 and its services
  - Get docker-compose.yml
  - Get docker-compose.production.yml - change if needed
  - Create a .env file
  - Execute docker-compose up
  - Configure cron jobs (scheduled jobs)
- Additional instructions for users of paid or non-public translate5 plug-ins
  - Visual / VisualReview
  - General steps for paid/private plug-ins
- Additional hints for experts regarding optional configuration steps
  - Editing files in the docker container
  - E-Mailing
  - SSL Configuration in the delivered nginx proxy
  - SSL Offloading
- Common problems
  - DNS resolution in docker container to other docker containers fail
  - PDFConverter
  - Use a local setup only

This is installation is for new installations. If you upgrade from an existing installation see translate5 > 5.8.0 - needed visualreview to docker migration

Precondition on host system

On the host system docker and docker-compose must be installed.

A public IP and a hostname / domain pointing to that IP must be configured. If that is not given check the common problems below.

License agreement

The translate5 installation consists of several Docker images providing the needed software apart from translate5 itself.

In order to use the docker compose files you have to remove the whole LICENSE DISCLAIMER, from START to END.

By removing these lines you accept the several different licenses used by the main software brought contained in each of the docker images, that you install. Which license this is you can see in the readme of every of those images on dockerhub.

This also means, that you yourself have the responsibility, that you combine the different licenses and download the different software used by translate5 and use them together. You do this on your own risk and need to make sure, that this works for you from a legal perspective and warranty perspective.

As with all Docker images, the installed images likely also contain other software which may be under other licenses (such as Bash, etc from the base distribution, along with any direct or indirect dependencies of the primary software being contained).

As for any pre-built image usage, it is the image user's responsibility to ensure that any use of these images complies with any relevant licenses for all software contained within.

Install translate5 and its services

Get docker-compose.yml

Get the latest docker-compose.yml file.

Remove the license disclaimer from the file before using it!

Get docker-compose.production.yml - change if needed

Get the latest docker-compose.production.yml file - which is an addition to the base config file from above.

Changes regarding your local setup should be done in that file. This may be the configuration of volumes, host and port configurations.

The production file configures several volumes where your data is stored.

Create a .env file

Create a file with name ".env" (nothing before the "." in the file name) and the following content.

Change the values of MYSQL_USERNAME and MYSQL_PASSWORD, if needed/wanted.

Change the value of APP_HOST to the domain under which translate5 should run without leading protocol (so without https://).

The value of COMPOSE_FILE can stay as it is, except if you want also install translate5 Visual (see below in this case).

In case you want to install translate5 Visual, please jump to the instructions for the Visual.

```
MYSQL_USERNAME=translate5
MYSQL_PASSWORD=translate5
APP_HOST=t5docker.localdev
COMPOSE_FILE=docker-compose.yml:docker-compose.production.yml
```

⚠ When installing under Windows, the separator between the different yml files in COMPOSE_FILE must be ; (semicolon) instead of : (colon) in the above example.

Execute docker-compose up

Put all 2 (or in case of translate5 Visual usage 3) fetched docker-compose yml files and the generated ".env" file into the same directory and call:

```
docker compose up -d

# If the above command returns an error, try the following command
docker-compose up -d
```

Now everything needed for translate5 will be automatically downloaded, installed and configured for your needs.

Sit back, and watch!

Configure cron jobs (scheduled jobs)

In order that things are working properly in translate5 some regular jobs must be called. Since in the host system mostly a cron is installed, we suggest to add the specific calls to the hosts crontab:

Please adjust the docker container name - localadm-php-1 in the example below. To get the proper name call docker ps.
And adjust the path to the docker binary if needed.

The interval of 15 minutes for the periodical calls should be kept. The point of time for the daily call could be adjusted so that it does not interfer with backups or so.

```
*/15 * * * * /usr/bin/docker exec localadm-php-1 t5 cron
30 23 * * * /usr/bin/docker exec localadm-php-1 t5 cron --daily
```

Alternativ way: call via URL

Additional instructions for users of paid or non-public translate5 plug-ins

Visual / VisualReview

In order to use the Visual plug-in:

- Get a docker hub token from MittagQI
- Login to docker hub with by executing
  - docker login -u translate5
  - At the password prompt, enter the personal access token.
- Download the docker-compose.visual.yml
- Change in your ".env" file the value of COMPOSE_FILE to docker-compose.yml:docker-compose.visual.yml:docker-compose.production.yml
- Open the docker-compose.production.yml and search for the following line:
  - # ENABLE ME IF USING Visual plugin
  - Uncomment the 4 lines below this line (remove the starting # from them)
- Go to Execute docker-compose up and follow those intructions. After this paragraph is done, go to General steps for paid/private plug-ins and follow those steps. Then you are done.

General steps for paid/private plug-ins

In order to use paid or private customer specific plug-ins, follow the normal installation. If all is up and running request the installer package containing all private plugins from mittagqi. You will receive an URL pointing to the package.zip.

On your CMD execute the following:

```
docker-compose exec php bash
t5 status
# now you should be in /var/www/translate5
wget URL_TO_YOUR.zip
./install-and-update.sh YOUR.zip
t5 service:autodiscovery -a
```

This steps will be improved in the near future, so that private plug-ins are registered via the config and pulled automatically.

Additional hints for experts regarding optional configuration steps

Editing files in the docker container

It is dangerous to edit files directly in the docker container, since some are overwritten on updating the docker containers (with docker pull) or just updating translate5 itself.

Possible scenarios where this may be needed are described above.

If files must be edited in the docker container (like the installation.ini mentioned below) then there are basically two possibilities:

1. go into the desired container and edit the file directly with the CLI editor vi :

   ```
   docker-compose exec php bash
   cd /var/www/translate5
   vi application/config/installation.ini
   ```

2. copy the file out, edit it outside and copy it in again:

   ```
   # get the file from the container:
   docker-compose cp php:/var/www/translate5/application/config/installation.ini .

   # edit installation.ini with the tool you want

   # put the file back into the container:
   docker-compose cp installation.ini php:/var/www/translate5/application/config/installation.ini
   ```

E-Mailing

By default sending of E-Mails is disabled in translate5 directly in /var/www/translate5/application/config/installation.ini in the php docker container.

1. set runtimeOptions.sendMailDisabled = 0 in the installation.ini

2. Add the SMTP configuration as described in Mail server as SMTP
3. Check the e-maling with

```
t5 system:mailtest your(AT)example.com
```

SSL Configuration in the delivered nginx proxy

- In the proxy container by default SSL is disabled.
- In order to enable SSL several steps are needed:
    1. expose port 443 in docker-compose.production.yml

```
        ports:
            # EXPOSE PORT 80 and 443
            - "80:80"
            - "443:443"
```

    2. Provide the certificate and key files in PEM format. The certificate file should also contain the intermediate files. See https://nginx.org/en/docs/http/ngx_http_ssl_module.html#ssl_certificate
    3. Place that files on the host server
    4. Get the the ssl-server.conf template file from inside the docker container and place it beneath your docker-compose.yml files: docker-compose cp proxy:/etc/nginx/conf.d/ssl-server.conf nginx-server.conf
    5. Edit the nginx-server.conf locally, if needed.
    6. Mount that nginx-server.conf file and the certificate files into the docker container. So the default /etc/nginx/conf.d/server.conf is overwritten with the ssl one, and also the certificates are mounted into the container.

```
        volumes:
            - /home/translate5/docker/nginx-server.conf:/etc/nginx/conf.d/server.conf
            - /home/translate5/docker/certificate.cert:/etc/nginx/certificate.chain
            - /home/translate5/docker/private.key:/etc/nginx/private.key
```

    7. Recreate the proxy container:

```
docker-compose stop proxy
docker-compose up -d proxy
```

SSL Offloading

if you don't know what SSL offloading is, just do not read ahead.

SSL Offloading to an external load balancer is possible, the load balancer must support / enable websockets.

1. **Do not follow the above "SSL Configuration in the delivered nginx proxy" instructions! Keep the ordinary HTTP setup!**
2. Add in your docker-compose.production.yml file a new environment variable HTTPS=on to the php container

```
    php:
        restart: unless-stopped
        volumes:
            - translate5-data:/var/www/translate5:cached
        # just start the apache, if no updates on restart are desired
        # DO THAT AFTER NOT BEFORE THE DEFAULT ENTRYPOINT WAS USED ONCE!
        # entrypoint: apache2-foreground
                environment:
                        -HTTPS=on
```

This is needed in order that translate5 recognizes that the application is running under https.

3. Configure Translate5 as it would have SSL (runtimeOptions.server.protocol = https://) and run autodiscovery to configure messagebus correctly. Jump into the php container and call the autoconfiguration:

```
docker-compose exec php bash
t5 config runtimeOptions.server.protocol "https://"
t5 service:auto -a -s frontendmessagebus
```

Common problems

DNS resolution in docker container to other docker containers fail

If you experience problems, that the used service names (like termtagger) are not correctly resolving to the correct IP, then follow

https://github.com/moby/moby/issues/41766#issuecomment-884111508

PDFConverter

If the pdfconverter container is always dying immediately after start-up with message "[WARNING] The local web server is already running".

This happens mostly after stopping and starting the container.

The easiest solution is to stop, remove and recreate the container:

```
docker stop docker-pdfconverter-1
docker rm docker-pdfconverter-1
docker compose up -d pdfconverter
```

The whole error message of the pdfconverter container

```
docker-pdfconverter-1  |  [WARNING] The local web server is already running
docker-pdfconverter-1  |
docker-pdfconverter-1  |
docker-pdfconverter-1  | Local Web Server
docker-pdfconverter-1  |     Listening on http://127.0.0.1:8086
docker-pdfconverter-1  |     The Web server is using PHP FPM 7.4.33 (from default version in $PATH)
docker-pdfconverter-1  |
docker-pdfconverter-1  | Local Domains
docker-pdfconverter-1  |
docker-pdfconverter-1  | Workers
docker-pdfconverter-1  |     No Workers
docker-pdfconverter-1  |
docker-pdfconverter-1  | Environment Variables
docker-pdfconverter-1  |     None
```

Use a local setup only

If you do not plan to make the installation public and you just want to evaluate translate5 locally, you have to do the following additional steps:

Add to the proxy service section in your docker-compose.yml

```
services:
    proxy:
        ports:
            - "127.0.0.1:80:80" # map
        extra_hosts:
            - ${APP_HOST}:127.0.0.1
```

Set APP_HOST in your .env file to localhost.

Or if you wanna use a different hostname pointing to 127.0.0.1, configure that in your /etc/hosts file first.