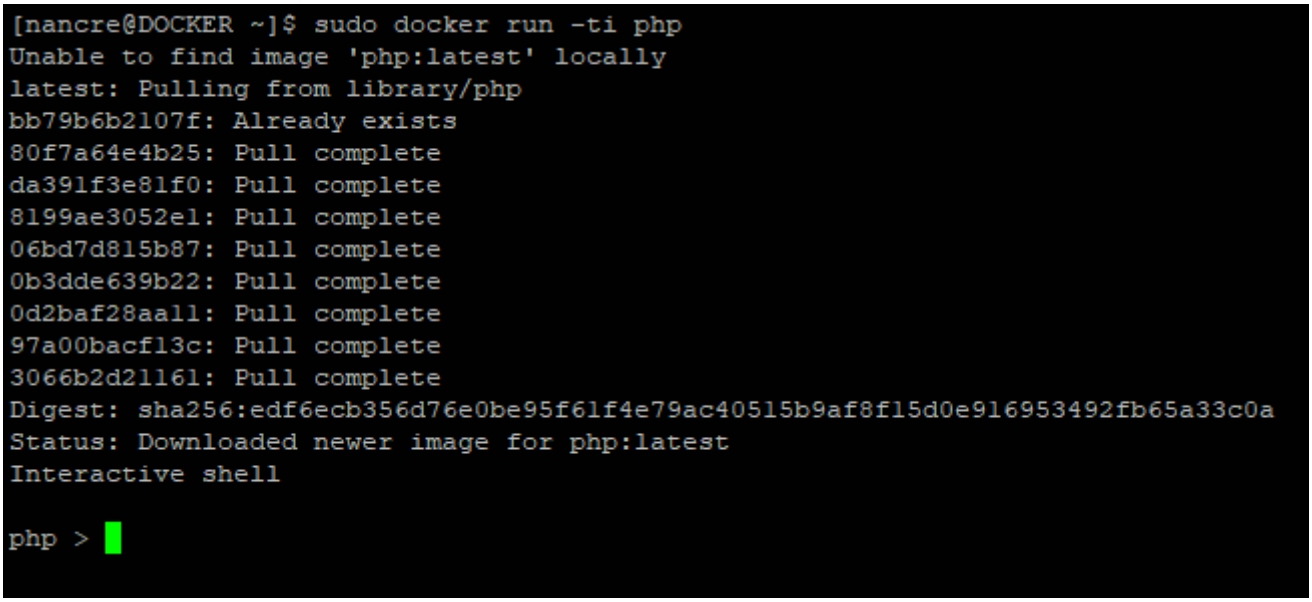
Labo 7 - Création d'images

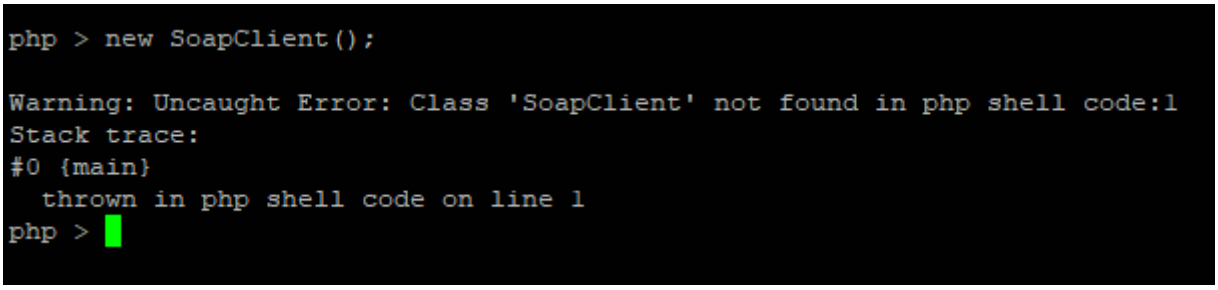
Nous allons voir dans cette exemple comment customiser une image officielle php.
D'abord créons le conteneur et on se retrouve dans la console php du conteneur

```
sudo docker run -ti php
```



Nous allons utiliser Soap (qui permet de créer des web services)

```
new SoapClient();
```



Nous voyons que le module Soap n'est pas présent sur l'image officielle de php téléchargée précédemment
Pour l'installer, il faut d'abord se connecter au conteneur php en bash depuis un terminal de la machine hôte

```
sudo docker exec -ti <idduconteneur> bash
```

```
[nancre@DOCKER ~]$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAMES
151d702372f6        php                 "docker-php-entrypoi…"   8 minutes ago       Up 8 minutes                            naughty_nash
[nancre@DOCKER ~]$ sudo docker exec -ti 151 bash
root@151d702372f6:/#
```

Pour trouver comment installer l'extension souhaitée, il faut aller sur le [hub de docker] dans explore, puis php

How to install more PHP extensions

Many extensions are already compiled into the image, so it's worth checking the output of `php -m` or `php -i` before going through the effort of compiling more.

We provide the helper scripts `docker-php-ext-configure`, `docker-php-ext-install`, and `docker-php-ext-enable` to more easily install PHP extensions.

In order to keep the images smaller, PHP's source is kept in a compressed tar file. To facilitate linking of PHP's source with any extension, we also provide the helper script `docker-php-source` to easily extract the tar or delete the extracted source. Note: if you do use `docker-php-source` to extract the source, be sure to delete it in the same layer of the docker image.

```
FROM php:7.4-cli
RUN docker-php-source extract \
    # do important things \
    && docker-php-source delete
```

PHP Core Extensions

For example, if you want to have a PHP-FPM image with the `gd` extension, you can inherit the base image that you like, and write your own `Dockerfile` like this:

```
FROM php:7.4-fpm
RUN apt-get update && apt-get install -y \
        libfreetype6-dev \
        libjpeg62-turbo-dev \
        libpng-dev \
    && docker-php-ext-configure gd --with-freetype --with-jpeg \
    && docker-php-ext-install -j$(nproc) gd
```

Remember, you must install dependencies for your extensions manually. If an extension needs custom `configure` arguments, you can use the `docker-php-ext-configure` script like this example. There is no need to run `docker-php-source` manually in this case, since that is handled by the `configure` and `install` scripts.

If you are having difficulty figuring out which Debian or Alpine packages need to be installed before `docker-php-ext-install`, then have a look at the `install-php-extensions` project. This script builds upon the `docker-php-ext-*` scripts and simplifies the installation of PHP extensions by automatically adding and removing Debian (apt) and Alpine (apk) packages. For example, to install the GD extension you simply have to run `install-php-extensions gd`. This tool is contributed by community members and is not included in the images, please refer to their Git repository for installation, usage, and issues.

See also "Dockerizing Compiled Software" for a description of the technique Tianon uses for determining the necessary build-time dependencies for any bit of software (which applies directly to compiling PHP extensions).

Donc dans le terminal bash du conteneur

```
docker-php-ext-install soap
```

```
configure: error: Package requirements (libxml-2.0 >= 2.7.6) were not met:

No package 'libxml-2.0' found

Consider adjusting the PKG_CONFIG_PATH environment variable if you
installed software in a non-standard prefix.

Alternatively, you may set the environment variables LIBXML_CFLAGS
and LIBXML_LIBS to avoid the need to call pkg-config.
See the pkg-config man page for more details.
root@151d702372f6:/#
```

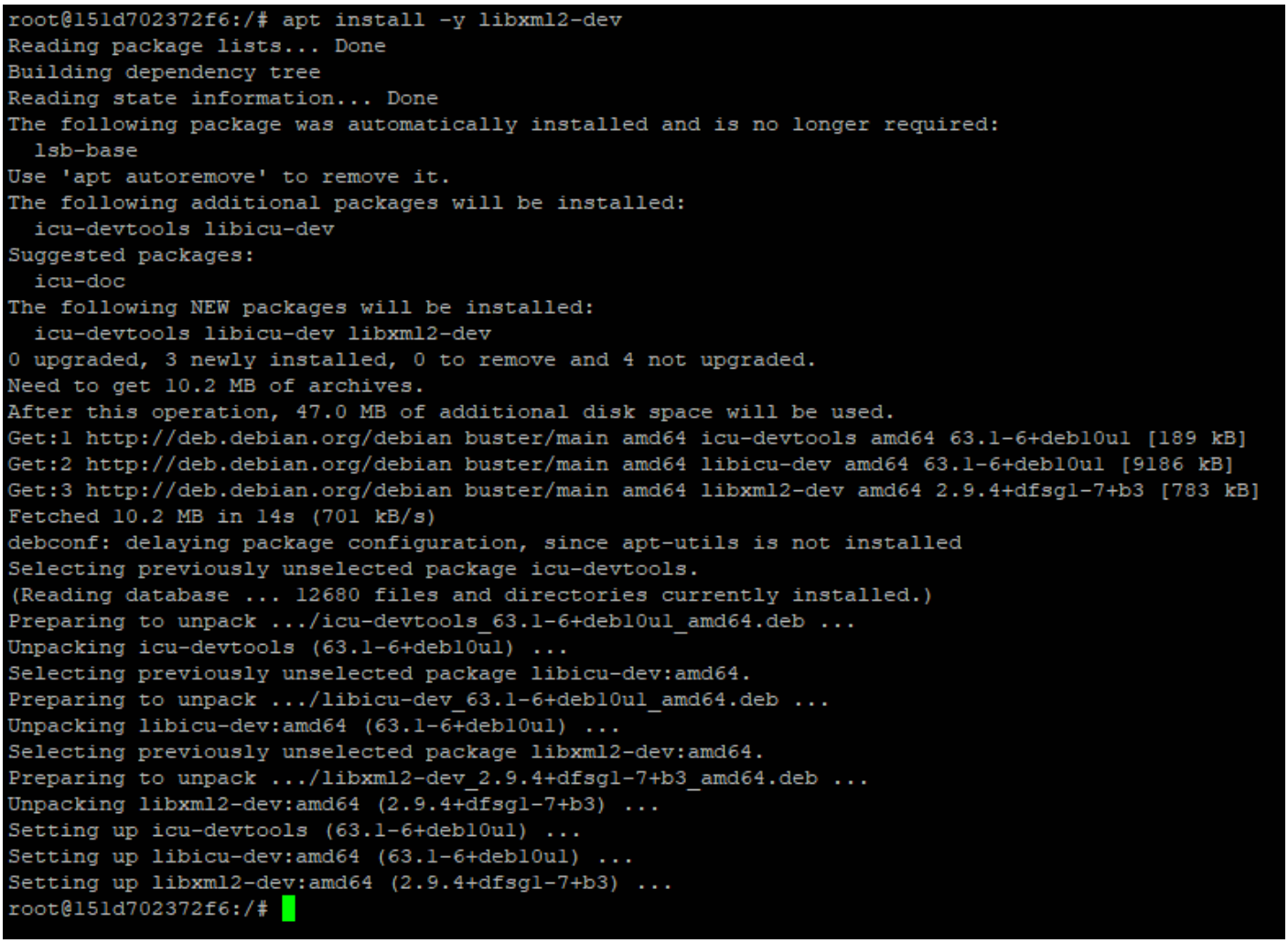
Visiblement il manque une librairie : libxml-2.0
Pour là trouver de retour dans le terminal bash du conteneur :

```
apt update && apt-cache search libxml2
```

```
root@151d702372f6:/# apt update && apt-cache search libxml2
Hit:1 http://security.debian.org/debian-security buster/updates InRelease
Hit:2 http://deb.debian.org/debian buster InRelease
Hit:3 http://deb.debian.org/debian buster-updates InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
4 packages can be upgraded. Run 'apt list --upgradable' to see them.
gir1.2-freedesktop - Introspection data for some FreeDesktop components
libghc-libxml-sax-dev - bindings for libXML2 SAX
libghc-libxml-sax-doc - bindings for libXML2 SAX; documentation
libghc-libxml-sax-prof - bindings for libXML2 SAX; profiling libraries
libxml++2.6-2v5 - C++ interface to the GNOME XML library (libxml2)
libxml++2.6-dev - C++ interface to the GNOME XML library (libxml2) - dev files
libxml-libxml-perl - Perl interface to the libxml2 library
libxml2 - GNOME XML library
libxml2-dbg - Debugging symbols for the GNOME XML library
libxml2-dev - Development files for the GNOME XML library
libxml2-doc - Documentation for the GNOME XML library
libxml2-utils - XML utilities
python-libxml2 - Python bindings for the GNOME XML library
python-libxml2-dbg - Python bindings for the GNOME XML library (debug extension)
python3-libxml2 - Python3 bindings for the GNOME XML library
python3-libxml2-dbg - Python3 bindings for the GNOME XML library (debug extension)
python-lxml - pythonic binding for the libxml2 and libxslt libraries
python-lxml-dbg - pythonic binding for the libxml2 and libxslt libraries (debug extension)
python-lxml-doc - pythonic binding for the libxml2 and libxslt libraries (documentation)
python3-lxml - pythonic binding for the libxml2 and libxslt libraries
python3-lxml-dbg - pythonic binding for the libxml2 and libxslt libraries (debug extension)
ruby-libxml - Ruby Bindings for LibXML2
libui-gxmlcpp-dev - High-level C++ wrapper library for libxml2/libxslt (development)
libui-gxmlcpp5v5 - High-level C++ wrapper library for libxml2/libxslt (run time)
root@151d702372f6:/#
```

Puis faire

```
apt install -y libxml2-dev
```



Et maintenant relançons l'installation de soap

```
docker-php-ext-install soap
```

```
Libraries have been installed in:
   /usr/src/php/ext/soap/modules

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,--rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------

Build complete.
Don't forget to run 'make test'.

Installing shared extensions:     /usr/local/lib/php/extensions/no-debug-non-zts-20190902/
find . -name \*.gcno -o -name \*.gcda | xargs rm -f
find . -name \*.lo -o -name \*.o | xargs rm -f
find . -name \*.la -o -name \*.a | xargs rm -f
find . -name \*.so | xargs rm -f
find . -name .libs -a -type d|xargs rm -rf
rm -f libphp.la      modules/* libs/*
```

Si nous relançons la commande new SoapClient depuis le terminal php cela ne fonctionnera pas puisque php est déjà lancé et il n'a pas l'extension précédemment installée.

Depuis le terminal bash, lançons d'abord php -a (pour se remettre en mode php, puis

```
new SoapClient('http://webservices.oorsprong.org/webexamples.countryinfo/CountryInfoService.wso?WSDL');
```

```
root@151d702372f6:/# php -a
Interactive shell

php > new SoapClient('http://schemas.xmlsoap.org/soap/encoding/');

Warning: Uncaught SoapFault exception: [WSDL] SOAP-ERROR: Parsing WSDL: Couldn't find <definitions> in 'http://schemas.xmlsoap.org/soap/encoding/' in php shell code:1
Stack trace:
#0 php shell code(1): SoapClient->SoapClient('http://schemas....')
#1 {main}
  thrown in php shell code on line 1
php > new SoapClient('http://webservices.oorsprong.org/webexamples.countryinfo/CountryInfoService.wso?WSDL');

Warning: Uncaught SoapFault exception: [WSDL] SOAP-ERROR: Parsing WSDL: Couldn't load from 'http://webservices.oorsprong.org/webexamples.countryinfo/CountryInfoService.ws
ong.org/webexamples.countryinfo/CountryInfoService.wso?WSDL"
 in php shell code:1
Stack trace:
#0 php shell code(1): SoapClient->SoapClient('http://webservi...')
#1 {main}
  thrown in php shell code on line 1
php >
```

Le webservice a été déplacé... mais cela n'empêche en rien la création de l'image.

La création de l'image passe par un Dockerfile

De retour dans le terminal hôte nous allons éditer un Dockerfile

```
sudo vi Dockerfile
```

Puis copier coller le texte dedans en sauvegardant

```
FROM php:7.0.31-cli
RUN apt update
RUN apt install -y libxml2-dev
RUN docker-php-ext-install soap
```

Pour construire l'image, il faut

```
sudo docker build -t php_soap:7.0.31 .
```

build -> construction de l'image à l'aide du Dockerfile
-t -> permet de donner un tag à notre image
. -> chemin courant où se trouve le Dockerfile (pour un Dockerfile avec un nom personnalisé, il faut rajouter -f MonDockerFile .)

J'ai choisi de personnaliser le nom de mon Dockerfile donc je lance la commande

```
sudo docker build -t php_soap:7.0.31 -f MonDockerFile .
```

```
(cd .libs && rm -f soap.la && ln -s ../soap.la soap.la)
/bin/bash /usr/src/php/ext/soap/libtool --mode=install cp ./soap.la /usr/src/php/ext/soap/modules
cp ./.libs/soap.so /usr/src/php/ext/soap/modules/soap.so
cp ./.libs/soap.lai /usr/src/php/ext/soap/modules/soap.la
PATH="$PATH:/sbin" ldconfig -n /usr/src/php/ext/soap/modules
----------------------------------------------------------------------
Libraries have been installed in:
   /usr/src/php/ext/soap/modules

If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,--rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------

Build complete.
Don't forget to run 'make test'.

Installing shared extensions:     /usr/local/lib/php/extensions/no-debug-non-zts-20151012/
find . -name \*.gcno -o -name \*.gcda | xargs rm -f
find . -name \*.lo -o -name \*.o | xargs rm -f
find . -name \*.la -o -name \*.a | xargs rm -f
find . -name \*.so | xargs rm -f
find . -name .libs -a -type d|xargs rm -rf
rm -f libphp.la       modules/* libs/*
Removing intermediate container a37d4a4fd8cb
 ---> 3e13e0a62e8d
Successfully built 3e13e0a62e8d
Successfully tagged php_soap:7.0.31
[nancre@DOCKER ~]$
```

On vérifie que l'image soit bien dans le cache :

```
sudo docker images
```

```
[nancre@DOCKER ~]$ sudo docker images
[sudo] password for nancre:
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
php_soap            7.0.31              3e13e0a62e8d        2 minutes ago       504MB
php                 latest              f4f453029716        6 days ago          405MB
nginx               latest              c39a868aad02        6 days ago          133MB
ubuntu              latest              d70eaf7277ea        2 weeks ago         72.9MB
bash                3.2                 c01d62f285d1        3 weeks ago         10.1MB
bash                5                   39a95ac32011        3 weeks ago         13.1MB
bash                latest              39a95ac32011        3 weeks ago         13.1MB
hello-world         latest              bf756fb1ae65        10 months ago       13.3kB
php                 7.0.31-cli          9ce6144lc9b4        2 years ago         357MB
[nancre@DOCKER ~]$
```

Testons notre image pour voir si le module soap a bien été installé :

```
sudo docker run -ti php_soap:7.0.31
```

```
[nancre@DOCKER ~]$ sudo docker run -ti php_soap:7.0.31
Interactive shell

php > new SoapClient();

Warning: Uncaught SoapFault exception: [Client] SoapClient::SoapClient(): Invalid parameters in php shell code:1
Stack trace:
#0 php shell code(1): SoapClient->SoapClient()
#1 {main}
  thrown in php shell code on line 1
php >
```

L'image customisée avec soap est bien présente !

Maintenant il est également possible de faire évoluer la version de notre image par exemple en 7.2.8

Modifions notre fichier customisé MonDockerFile

```
sudo vi MonDockerFile
```

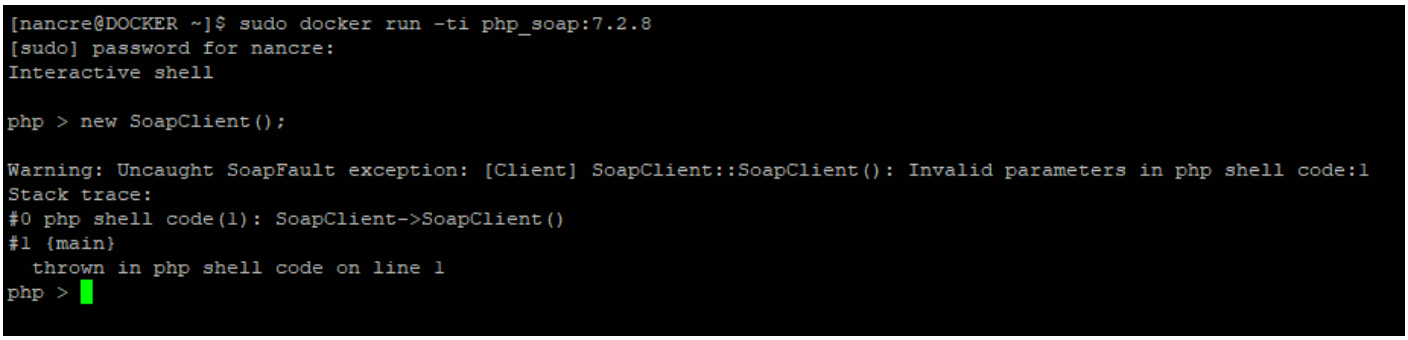
```
FROM php:7.2.8-cli
RUN apt update
RUN apt install -y libxml2-dev
RUN docker-php-ext-install soap
~
```

```
sudo docker build -t php_soap:7.2.8 -f MonDockerFile .
```

```
If you ever happen to want to link against installed libraries
in a given directory, LIBDIR, you must either use libtool, and
specify the full pathname of the library, or use the `-LLIBDIR'
flag during linking and do at least one of the following:
   - add LIBDIR to the `LD_LIBRARY_PATH' environment variable
     during execution
   - add LIBDIR to the `LD_RUN_PATH' environment variable
     during linking
   - use the `-Wl,--rpath -Wl,LIBDIR' linker flag
   - have your system administrator add LIBDIR to `/etc/ld.so.conf'

See any operating system documentation about shared libraries for
more information, such as the ld(1) and ld.so(8) manual pages.
----------------------------------------------------------------------

Build complete.
Don't forget to run 'make test'.

Installing shared extensions:     /usr/local/lib/php/extensions/no-debug-non-zts-20170718/
find . -name \*.gcno -o -name \*.gcda | xargs rm -f
find . -name \*.lo -o -name \*.o | xargs rm -f
find . -name \*.la -o -name \*.a | xargs rm -f
find . -name \*.so | xargs rm -f
find . -name .libs -a -type d|xargs rm -rf
rm -f libphp.la       modules/* libs/*
Removing intermediate container d5031f8bfa0e
 ---> fb7f1e33e2f1
Successfully built fb7f1e33e2f1
Successfully tagged php_soap:7.2.8
[nancre@DOCKER ~]$
```

Même chose que plus haut, vérifions

```
sudo docker run -ti php_soap:7.2.8
```



> Cela a fonctionné. Il ne nous reste plus qu'à arrêter l'ancien conteneur et mettre en marche le nouveau

---

Revision #1
Created 2026-04-24 13:17:14 UTC by Nico là
Updated 2026-04-24 13:17:24 UTC by Nico là