

Using one or more versions of Lenstool in WSL and Visual Studio Code

Kyde BELLON

This tutorial has been made for Lenstool users working with Visual Studio Code on Windows and willing to have a full Lenstool setup (one or more versions of Lenstool) via WSL. The installation of SAOImageDS9 is also explained.

1 Prerequisites

1.1 Install WSL

Not covered here, please follow the following page: <https://learn.microsoft.com/en-us/windows/wsl/install>

You need to have at least Windows 10 version 2004 and higher (Build 19041 and higher) or Windows 11 to be able to get WSL.

1.2 Install miniconda or anaconda in WSL

1/ Download either miniconda or anaconda from the following page: <https://docs.conda.io/projects/conda/en/stable/user-guide/install/linux.html>

2/ Copy-Paste in your home directory, which has the name of the session you created in WSL. Accessible via the file explorer as shown in Figure 1: LINUX > UBUNTU > HOME > [YOUR_SESSION_NAME]

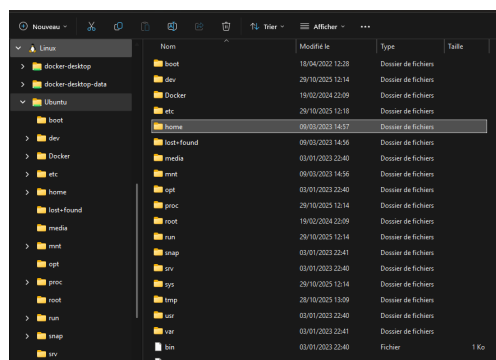


Figure 1: Your session name can be found in the HOME directory find as shown in the file explorer. To see the Linux directory, you might have to launch WSL first.

If you can't access this location with your explorer, you can make it appear using the following line in the WSL terminal:

```
explorer.exe
```

Then open a terminal at this location and run the bash file using the command:

```
bash <conda-installer-name>-latest-Linux-x86_64.sh
```

For example, with miniconda:

```
bash Miniconda3-latest-Linux-x86_64.sh
```

During the installation, you will be asked to:

- Press Enter;
- Accept the license terms by typing "yes";
- Press Enter to use the default location¹.
- **Restart your terminal**

To test if conda is working well, you may write:

```
conda --version
```

Conda should answer with its current version. If you get the error (or something similar):

```
command not found: conda
```

Something went wrong during the installation... You might try again from the "bash Miniconda3-latest-Linux-x86_64.sh" or seek documentation. Good Luck...

2 Install the latest version of Lenstool in WSL

Once conda is properly installed, the installation of Lenstool is pretty simple and consists of 2 lines to copy-paste directly in the terminal (in WSL or via Visual Studio Code, see section 3)[1]:

```
conda create -n lenstool -c conda-forge lenstool astropy matplotlib numpy  
conda activate lenstool
```

And you are done! Easy, isn't it ?

¹You might choose another location, but I personally didn't manage to have something running properly as I couldn't find how to change the environment variable related to WSL. Good luck.

Each time you restart your terminal, be sure to be in the right conda environment to use Lenstool:

```
conda activate lenstool
```

3 Open a WSL folder with Visual Studio Code

Download Visual Studio Code (VS Code) for Windows: <https://visualstudio.microsoft.com/fr/downloads>

To open a WSL directory, you have 2 options.

First Method: You need first to open VS Code in Windows, then click in the lower left corner and select "Connect to WSL".

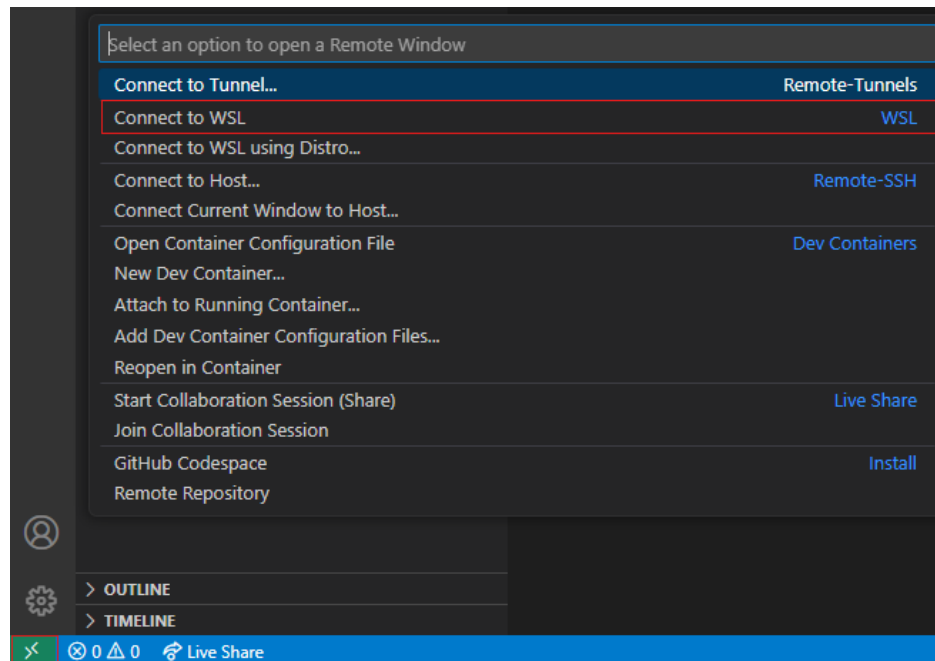


Figure 2: Caption

Second Method: Launch WSL and type in the terminal:

```
code .
```

Then choose the folder you want to open as you would do classically: FILE > OPEN FOLDER.

Pay attention that some of your extensions might need to be totally/partially reinstall in WSL to be used (Python debugger...) [2].

4 Install SAOImageDS9 in WSL

Any version of SAOImageDS9 installed on Windows will not be able to communicate with the WSL terminal that we will use. It is require to install SAODs9 in the WSL environment.

First Method: Download SAOImageDS9 from: <https://sites.google.com/cfa.harvard.edu/saoimageds9/download>

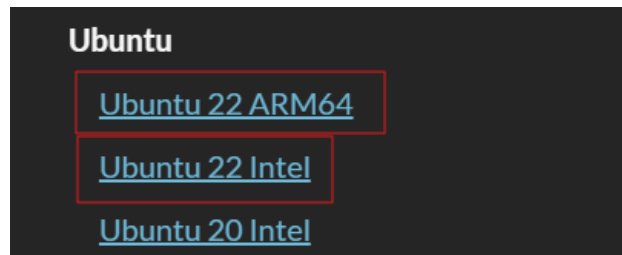


Figure 3: WSL will install the latest version of Ubuntu. Depending on your machine hardware, choose the correct installer.

Copy and paste the installer into your directory in WSL. Accessible via the file explorer as shown in Figure 1: LINUX > UBUNTU > HOME > [YOUR_SESSION_NAME]

Then, uncompress the file using either the adapted command line or via the file explorer.

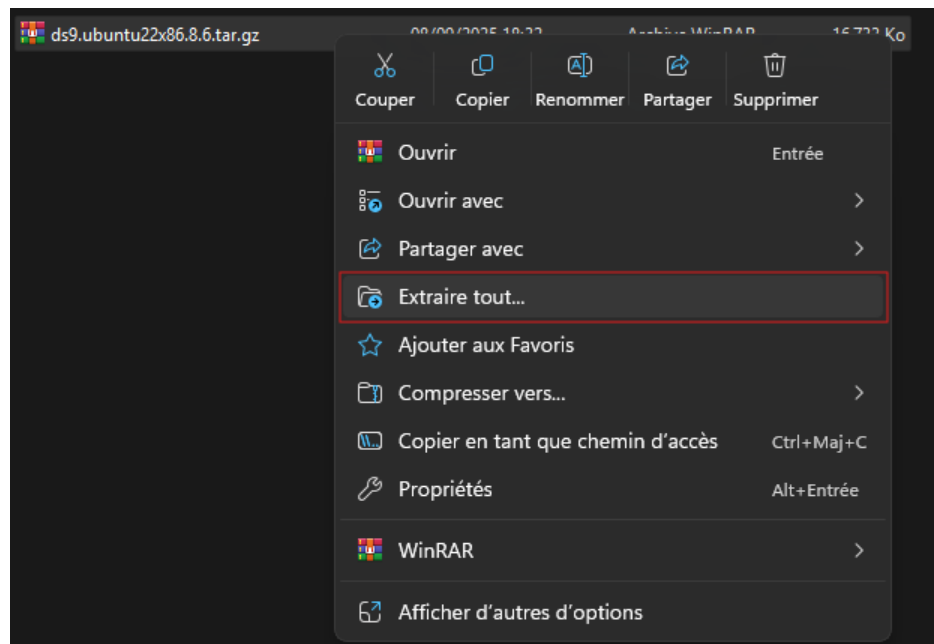


Figure 4: Uncompress the ds9 file.

Second Method: Use the command line:

```
sudo apt install saods9
```

Then launch, with the ability to still use the terminal, by writing

```
ds9 &
```

If you get the following error:

```
Error in startup script: can't find package base64
while executing"package require base64"
(file "/usr/share/saods9/library/ds9.tcl" line 219)
```

Solve the case by following the recommendation of this page: <https://askubuntu.com/questions/1451534/saods9-package-require-base64>.

To use pelli/pcl/... commands, you need to enable the connection XPA with FILE > XPA > CONNECT as shown in Figure 5.

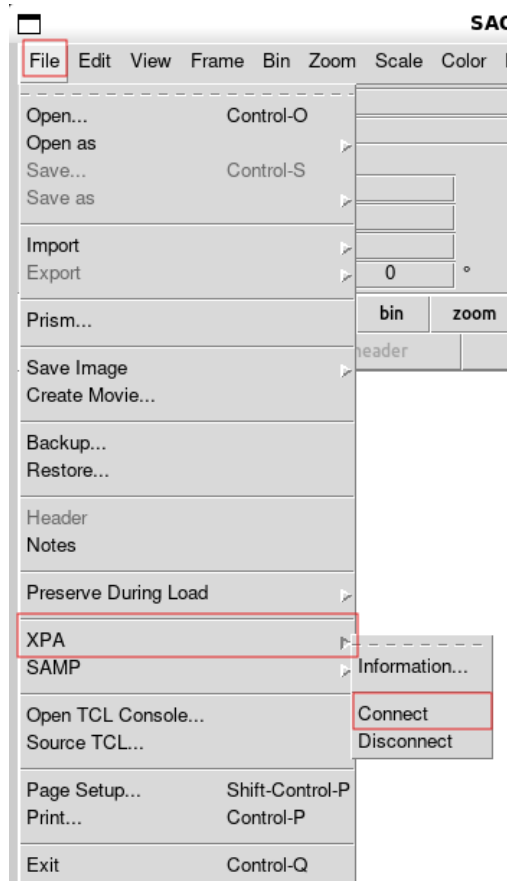


Figure 5: Enable the communication between the terminal and DS9.

5 Install an additional version of Lenstool in WSL

This is where the fun begins. For any reason (fun ?), you may want to install a version of Lenstool which is not the latest one and be able to switch easily between versions.

The first step is to get the version of Lenstool you are looking for, either through the Git <https://git-cral.univ-lyon1.fr/lenstool/lenstool> or received from someone.

To install this new version, I let here a series of command to by copy-paste in the terminal.

For example, with a version called "lenstool-8.0-dsrcplan", from the location where you put your archive:

```
mkdir lenstool_dsrcplan # to create a new sub directory
cd lenstool_dsrcplan # to go to the new sub directory
cp ../lenstool-8.0-dsrcplan.tar.gz .
# to copy the compressed files where I am
tar -xvzf lenstool-8.0-dsrcplan.tar.gz # uncrompress the files
cd lenstool-8.0/ # go into the new Lenstool directory

conda create --name lenstool_dsp
# create a new environment, name it as you want
conda activate lenstool_dsp
# activate the created environment

# Copy the 2 following lines without any line breaks
conda install -c conda-forge xpa wcslib cfitsio gsl pkg-config pip numpy
astropy gcc_linux-64 anaconda::sysroot_linux-64

./configure --prefix=$CONDA_PREFIX

#if never done it before
sudo apt install make

make -j
make clean #optional
make install
```

Once every step is done, you might check that everything went well. This can be done by typing:

```
which lenstool
```

and

```
lenstool
```

To which you should have an answer similar to the one presented in Figure 6.

```
(lenstool_dsp) kyde@Kyde:~/GravLensing$ which lenstool
/home/kyde/miniconda3/envs/lenstool_dsp/bin/lenstool
(lenstool_dsp) kyde@Kyde:~/GravLensing$ lenstool

Unexpected number of arguments

USAGE:
lenstool input_file [-n]

(lenstool_dsp) kyde@Kyde:~/GravLensing$
```

Figure 6: How to check that Lenstool is properly installed.

Once Lenstool is properly installed, you might want to launch a model or an optimisation with your new version of Lenstool. Each time you launch a terminal, be sure to activate the right conda environment. This can be done by following:

```
conda activate [your environment name]
```

If you want to modify the number of threads used by Lenstool:

```
export OMP_NUM_THREADS=4
```

If you want to use pelli/pcl/... commands of Lenstool and find out that the commands are not found, typing the following line might help you:

```
export LD_LIBRARY_PATH=$CONDA_PREFIX/lib
```

If many of those lines need to be copied, I recommend creating a bash file in your home directory or a dedicated subdirectory with these last lines to be run directly with:

```
source ~/[your_filename.sh]
```

Such that you would be ready to use the wanted version of Lenstool and able to switch from a version to another easily.

Bibliography

- [1] Jean-Paul Kneib et al. URL: <https://projets.lam.fr/projects/lenstool/wiki>.
- [2] Microsoft. URL: <https://learn.microsoft.com/en-us/windows/wsl/tutorials/wsl-vscode>.