



# Workshop pre-requisites

## iCross Reactive Transport Modelling Workshop I: Radionuclides Model-Chain

Jaime Garibay-Rodriguez

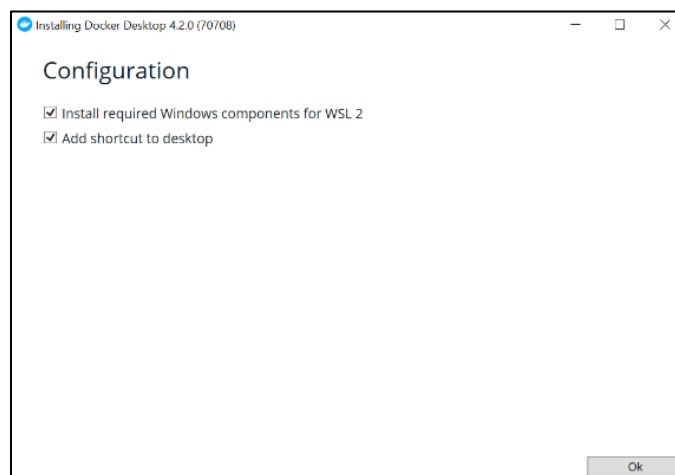
Even though OGS-6 is not [platform-specific](#), we will show how to work with Jupyter notebooks on Linux, whether native for Linux-users or via [Docker](#) for Mac plus the Windows Subsystem for Linux ([WSL2](#)) for Windows-users. An overview of Jupyter notebooks in OGS-6 can be found [here](#).

**Note:** please make sure to follow the below steps previous to the workshop. Be prepared to restart your computer during the installation. The complete process should take about 1 hour if installing Docker and WSL2 + Ubuntu (time may be longer for slower connection speeds).

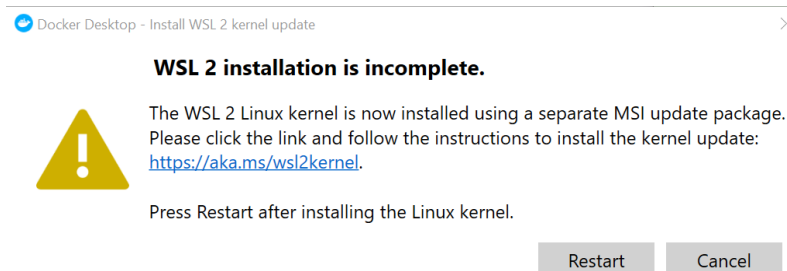
In case of any questions, do not hesitate to contact: [jaime.garibay-rodriguez@ufz.de](mailto:jaime.garibay-rodriguez@ufz.de)

## 1. Setup Docker and WSL2 (Windows-users only)

**Step 1.** Install Docker from <https://docs.docker.com/desktop/windows/install/> and ensure the *Enable Hyper-V Windows Features* or the *Install required Windows components for WSL 2* option is selected on the Configuration page:



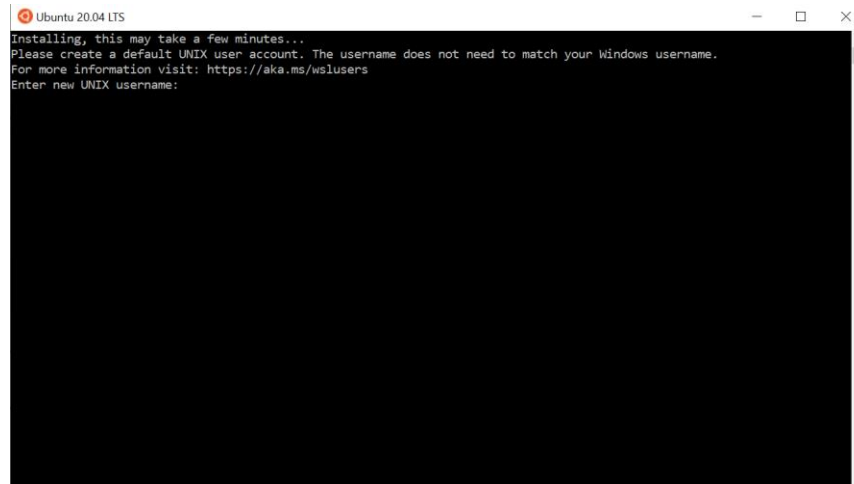
After the installation is completed, you may need to restart your computer. After the restart, you may be prompted to install a Linux kernel update:



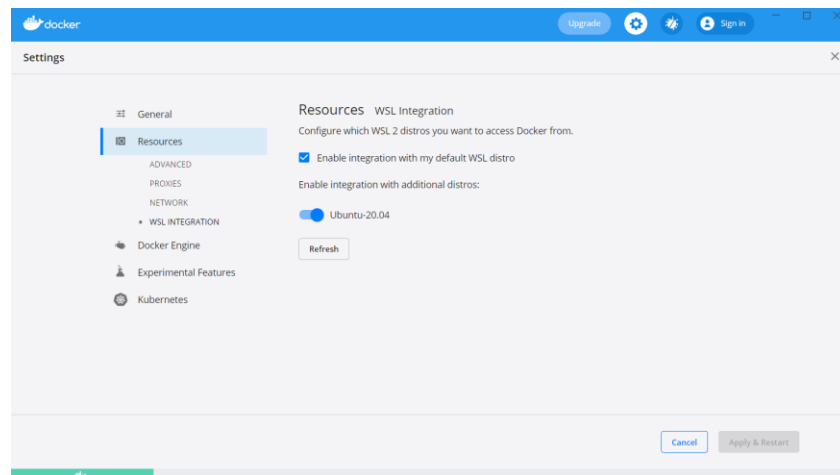
So, go ahead and install the [update](#). Next, restart Docker.

**Step 2.** Install [Ubuntu 20.04 LTS](#) from the Microsoft App Store.

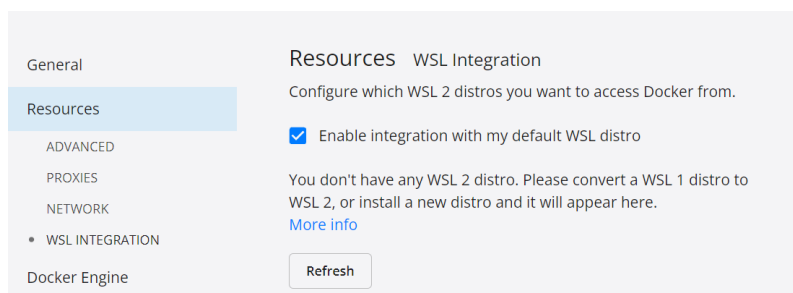
**Step 3.** Open the Ubuntu 20.04 LTS command prompt by searching “ubuntu” in the start menu. You should see a window similar to the one below. Ubuntu will ask for a username and a password (remember the password! It will be necessary when installing software in Ubuntu):



**Step 4.** Open the Docker Desktop application (restart if necessary) and make sure you have Ubuntu-20.04 enabled/added under *Settings / Resources / WSL integration*. **Note:** Docker is minimized in the tray in Windows.



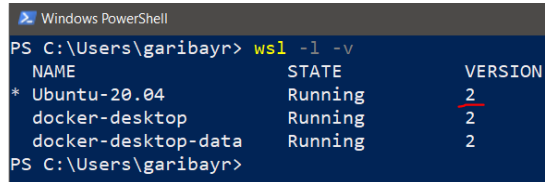
In case you can't see the Ubuntu-20.04 distro in the list, you need to make sure you have [WSL2 instead of WSL1](#) (Docker won't recognize WSL1):



First check the list and version of your WSL distros by running the following command in a command prompt or PowerShell:

```
wsl -l -v
```

You should see something like this:



```
Windows PowerShell
PS C:\Users\garibayr> wsl -l -v
NAME                STATE      VERSION
* Ubuntu-20.04      Running    2
docker-desktop      Running    2
docker-desktop-data Running    2
PS C:\Users\garibayr>
```

If Ubuntu-20.04 shows VERSION = 1, you need to convert from WSL1 to WSL2. It's very simple, just run the following command:

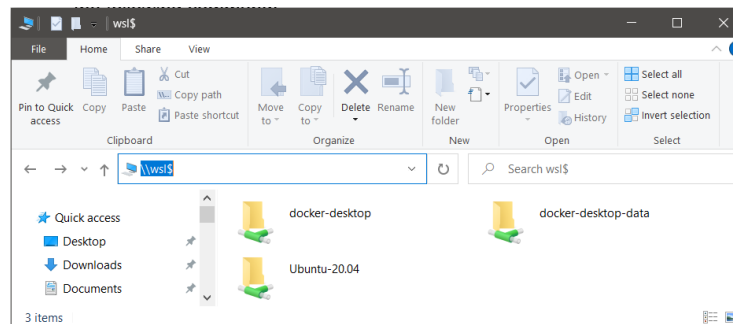
```
wsl --set-version Ubuntu-20.04 2
```

The conversion will take a few minutes. After that, go back to Docker *Settings / Resources / WSL integration* and refresh the list of WSL 2 distros. Ubuntu-20.04 should now appear there (if not, please restart Docker from the system tray). Then Apply & Restart and everything is set to start using OGS via a remote container with Jupyter.

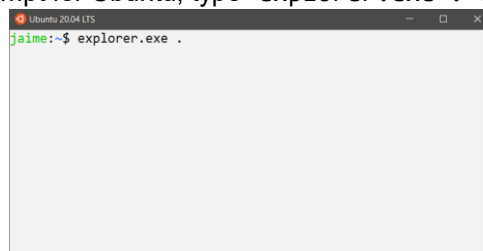
### Notes about the file system of WSL2 in Windows:

Windows mounts the default folder of the Linux environment in a “network” location (i.e., cannot be accessed directly from the file explorer). To access the “/home/<username>” location, there are a few options:

1. Open the file explorer and type `\\wsl$` and press enter. You should see a folder name “Ubuntu-20.04”; this is the location where the Linux subsystem stores data (navigate to “home/<username>” to see the default start folder).



2. In the default command prompt for Ubuntu, type `explorer.exe .` and press enter:



**Final remark:** the Linux Subsystem is much more efficient when all file operations happen inside the above location (at the root folder of the Linux file system, you can access your windows drives with, for example,

/mnt/c/Users/<Windows\_username>, where c is the Windows drive letter). Therefore, it is recommended that all work (in this case, all OGS files and results) is kept inside the Linux file system during computation. You can move those later into a “normal” Windows folder for more convenient access, if needed.

## 1a. Install Docker or Singularity for Linux/Mac

There are several ways of installing [Docker](#) or [Singularity](#) for Linux or Mac (both work for using OGS containers). If you are a Linux or Mac-native user, please make sure you have a running installation of Docker or Singularity before the workshop:

Docker for Ubuntu: <https://docs.docker.com/engine/install/ubuntu/>

Docker for Mac: <https://docs.docker.com/desktop/mac/install/>

Singularity for Linux: <https://sylabs.io/guides/3.0/user-guide/installation.html>

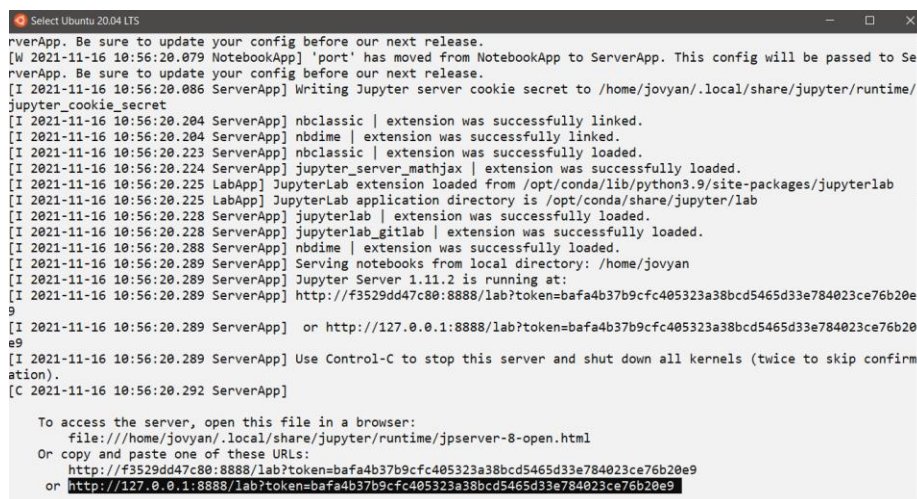
## 1b. Test installation with an OGS container

From this point forward, the experience should be quite similar independent of the platform used. To test your Docker installation beforehand, copy and paste the following in an Ubuntu command prompt in Windows or a terminal in Linux/Mac to get the latest development OGS container (**note: make sure Docker is running! In Windows, open the Docker Desktop application**):

```
docker run --rm -p 8888:8888 -v $PWD:/home/jovyan/work --user `id -u $USER`  
--group-add users registry.openegeosys.org/ogs/ogs/ogs-serial-jupyter:latest
```

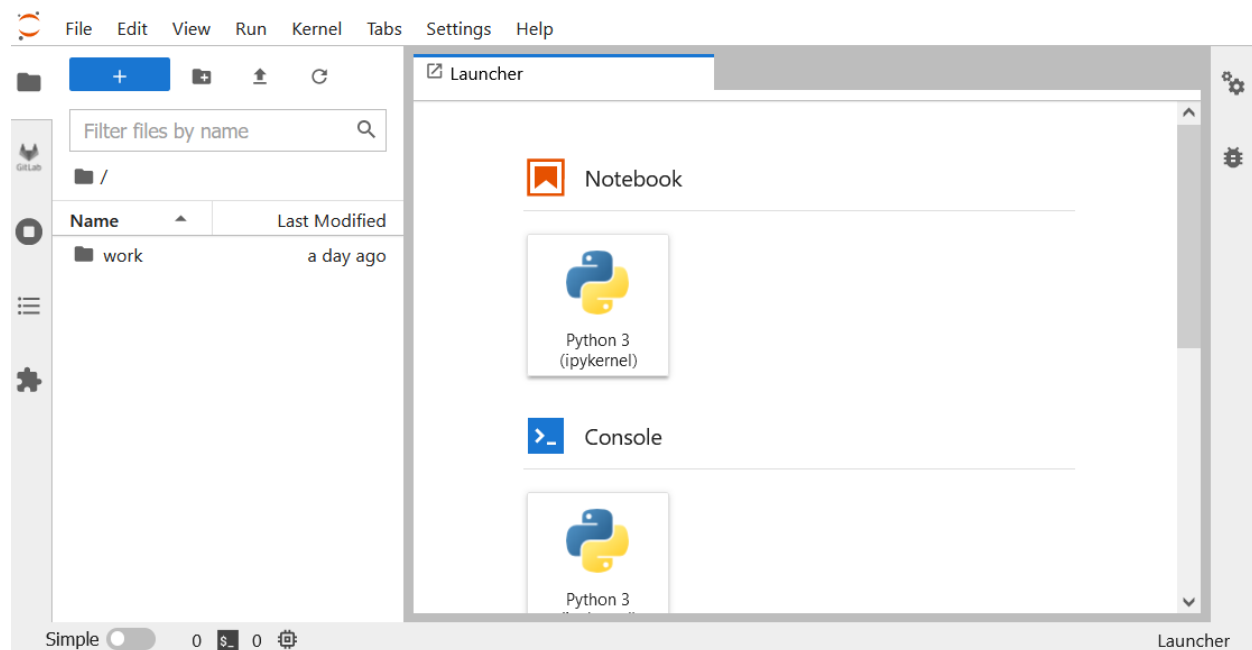
There's no need to change anything (user) if you run the command from the home folder (e.g., the default folder when opening the Ubuntu prompt in Windows, /home/<username>). The above command will mount your current directory into the `work` folder. The first time you run the command, a few packages will have to be downloaded (subsequent instances will be much faster). By adding `:latest` at the end of the command, you are sure to get the latest development features of OGS (which we'll use in the workshop). More information can be found in the [OGS webpage](#).

After the execution of the previous command, you should see something similar to:



```
Select Ubuntu 20.04 LTS  
ServerApp. Be sure to update your config before our next release.  
[W 2021-11-16 10:56:20.079 NotebookApp] 'port' has moved from NotebookApp to ServerApp. This config will be passed to Se  
rverApp. Be sure to update your config before our next release.  
[I 2021-11-16 10:56:20.086 ServerApp] Writing Jupyter server cookie secret to /home/jovyan/.local/share/jupyter/runtime/  
jupyter_cookie_secret  
[I 2021-11-16 10:56:20.204 ServerApp] nbclassic | extension was successfully linked.  
[I 2021-11-16 10:56:20.204 ServerApp] nbdtm | extension was successfully linked.  
[I 2021-11-16 10:56:20.223 ServerApp] nbclassic | extension was successfully loaded.  
[I 2021-11-16 10:56:20.224 ServerApp] jupyter_server_mathjax | extension was successfully loaded.  
[I 2021-11-16 10:56:20.225 LabApp] JupyterLab extension loaded from /opt/conda/lib/python3.9/site-packages/jupyterlab  
[I 2021-11-16 10:56:20.225 LabApp] JupyterLab application directory is /opt/conda/share/jupyter/lab  
[I 2021-11-16 10:56:20.228 ServerApp] jupyterlab | extension was successfully loaded.  
[I 2021-11-16 10:56:20.228 ServerApp] jupyterlab_gitlab | extension was successfully loaded.  
[I 2021-11-16 10:56:20.288 ServerApp] nbdtm | extension was successfully loaded.  
[I 2021-11-16 10:56:20.289 ServerApp] Serving notebooks from local directory: /home/jovyan  
[I 2021-11-16 10:56:20.289 ServerApp] Jupyter Server 1.11.2 is running at:  
[I 2021-11-16 10:56:20.289 ServerApp] http://f3529dd47c80:8888/lab?token=bafa4b37b9cfc405323a38bcd5465d33e784023ce76b20e  
9  
[I 2021-11-16 10:56:20.289 ServerApp] or http://127.0.0.1:8888/lab?token=bafa4b37b9cfc405323a38bcd5465d33e784023ce76b20  
e9  
[I 2021-11-16 10:56:20.289 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirm  
ation).  
[C 2021-11-16 10:56:20.292 ServerApp]  
  
To access the server, open this file in a browser:  
file:///home/jovyan/.local/share/jupyter/runtime/jpserver-8-open.html  
Or copy and paste one of these URLs:  
http://f3529dd47c80:8888/lab?token=bafa4b37b9cfc405323a38bcd5465d33e784023ce76b20e9  
or http://127.0.0.1:8888/lab?token=bafa4b37b9cfc405323a38bcd5465d33e784023ce76b20e9
```

Just copy and paste the highlighted address in a web browser and Jupyter-lab should open a few seconds later:



Double-click on the `work` folder and you should see your home folder. You can create/modify/delete files and folders from this window. To shutdown the session, go to File / Shutdown and close the tab on the browser.

**Note: make sure you work inside the `work` directory, files outside may not be saved after shutting down the session!**

If using Singularity, the command is as follows:

```
singularity run docker://registry.openeosys.org/ogs/ogs/ogs-serial-jupyter:latest
```

Singularity works slightly different from Docker, there's no `work` folder that mounts your current folder, you have access to your entire file system from the start.

**Note:** be aware that OGS container image is the latest dev version at the moment when the command is executed (it will not be automatically updated). To update the OGS container image version, you need to remove the latest image in Docker *Images / Clean up* and execute the above command again.

## 2. Download GMSH for creating FEM meshes

As a pre-processing step, we need to be able to create a mesh to input to the OGS model. We will use GMSH v3.0.6.

For Windows, download here: <https://gmsh.info/bin/Windows/gmsh-3.0.6-Windows64.zip>

Linux: <https://gmsh.info/bin/Linux/gmsh-3.0.6-Linux64.tgz>

Mac: <https://gmsh.info/bin/MacOSX/gmsh-3.0.6-MacOSX.dmg>

GMSH is a standalone executable (similar to OGS) but with a graphical interface. Unzip and open the downloaded folder. Then, you should be able to run GMSH by double-clicking on gmsh.exe (in Windows). GMSH also works from the command line interface, just add the location of the gmsh executable to your

system PATH. For Ubuntu (native or WSL2) you can add this line at the end of your `~.bashrc` file (in the home folder):

```
export PATH="/home/<username>/gms-3.0.6-Linux64/bin:$PATH"
```

This will enable you to run GMSH from the command line.

**Note:** Newer versions of GMSH may not be fully compatible with OGS.

### 3. Install ParaView GUI for post-processing and visualization

Although [VTUInterface](#) has many functions for output visualization (already included in the OGS container), ParaView is a powerful open-source software with a GUI for post-, pre-processing and visualization. If you want to learn hands-on about ParaView during the workshop, download and install [version 5.9.0](#).

### 4. Install PHREEQC v3 (and Python)

PHREEQC v3 is a powerful geochemical solver. It has been coupled to OGS-6#iPHREEQC to perform reactive transport calculations, but the standalone version is needed for performing some batch geochemical calculations when using look-up tables in OGS. Download and install PHREEQC for Windows [here](#).

We will use a simple Python script to quickly perform a series of PHREEQC batch calculations. In order to use it, you will need to install [Python](#) with the following options:

- Check Add Python 3.X to PATH
- Customize installation
- Make sure to have pip enabled (you may uncheck Documentation, tcl/tk, Python test suite)
- Check Add Python to environment variables!

After installing Python, you may add numpy, pandas and json packages beforehand (e.g., in a command prompt, `pip install numpy`)

**Note:** For Linux, only the PHREEQC source code is provided, so it is necessary to compile it.

### 5. [Optional] Install Notepad++ for XML file editing

To create and edit OGS project files, you need a text editor. There are several choices, but Notepad++ is a good option (you can download it [here](#)). There are many other alternatives, of course; the important feature is that your text editor needs to support XML. If possible, make sure you have the “Edit with Notepad++” option when you right-click on a file. Typical installations should have this (Windows).

**Note:** you can also edit OGS project files in a text editor inside a Jupyter notebook.

### 6. [Optional] Download OGS Windows executables

If you are not able to setup Docker and WSL2, you can download the latest OGS Windows executables (which include pre-processing utilities) from:

[https://gitlab.opengeosys.org/ogs/ogs/-/jobs/artifacts/master/browse/build/release?job=build+win%3A+%5BUSE\\_PYTHON%3DOFF%5D](https://gitlab.opengeosys.org/ogs/ogs/-/jobs/artifacts/master/browse/build/release?job=build+win%3A+%5BUSE_PYTHON%3DOFF%5D)

Just download and unzip the file named `ogs-X.X.X-XXX-xxXXXxXXxfx-Windows-XX.X.XXXXX-  
utils.zip`.

You can execute the OGS binary and utilities inside the `/bin` folder in a command prompt. For convenience, add the location of the `/bin` directory to your system PATH (in Windows, you can search “Edit the environment variables for you account”, double click on “Path” and add a new address pointing to, for instance, `C:\Users\\ogs6-directory\bin`).

**Note about Windows executables:** in some cases, OGS simulations may take more time to complete when running natively on Windows, therefore, we recommend the usage of Docker and WSL2 for Windows-users!