SWMM5+ Installation Guide

1 Quick Start Guide

Follow the workflow below to install SWMM5+ This Quick Start Guide is intended to provide users a step-by-step instruction for *installing* the SWMM5+ software. Details on how to run the SWMM5+ executable are provided in §4 of the main Beta Release Documentation.

System Requirements: This installation guide supports **Windows 10** (via Docker) and Ubuntu **Linux** 16+. Other up to date Linux distributions will likely work, but have not been tested. Currently, **MacOS** users can run SWMM5+ via Docker in serial (one processor) mode only.

Windows 10/MacOS users begin in §1.1.

Linux users begin in §1.2/

1.1 Windows 10/MacOS - Using Docker

1.1.1 Installing the SWMM5+ Docker container

Install the Docker Desktop application from `https://www.docker.com/get-started`

Note: If you need admin privileges to install the Docker Desktop application, you will likely need to run the following commands also as an administrator.

After installation of the Docker Desktop application, open a command terminal in a directory of your choosing and execute

```
$ docker pull cimmresearch/swmm5plus-docker
```

This command pulls a Docker image from the CIMM Research DockerHub repository. The Docker image contains the dependencies required to build and run the SWMM5+ executable.

Note: This takes awhile (approx. 30 minutes) and requires about 30GB of disk space.

The Docker image itself is immutable, and cannot save any output files once the image is closed. For output files and edits to be recoverable between Docker image launches, we need to attach a "volume" folder from Docker to a local directory.

```
$ docker volume create SWMM5plus
$ docker run --mount source=SWMM5plus,destination=/SWMM_5_Plus
    cimmresearch/swmm5plus-docker
```

At this point any changes made within the `cimmresearch/swmm5plus-docker` container will be saved and persisted in the SWMM5plus mounted volume. To test this, we can run a quick test within the Docker Desktop app. The test will be to create a textfile, and verify that the file is recoverable once the container is restarted.

Open the Docker Desktop app, there should be an entry in the Containers/Apps window with a funny two part name, autogenerated by Docker, with a "cimmresearch" tag.

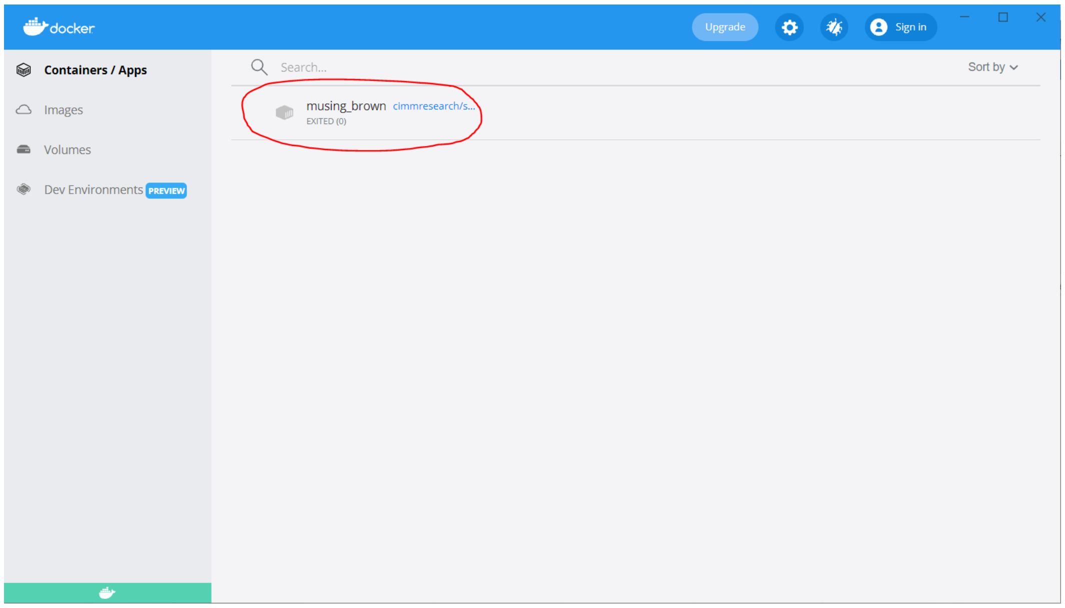


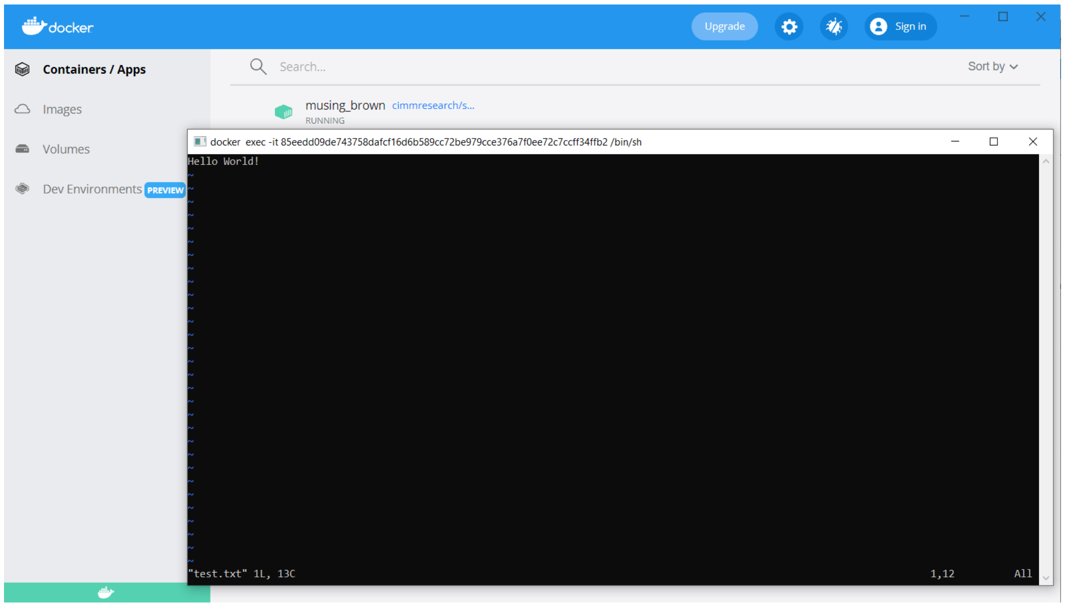
Figure 1: Docker Desktop GUI

Hold the cursor over the container, and click "Start" to activate the container. Then click the "CLI" button to launch a Docker terminal.

Note: This new terminal is running in the Docker container's Linux image, so Linux command syntax should be used.

Create a new file called "test.txt", write some text, then save and close the file.

```
$ vim test.txt
```



```
docker  exec -it 85eedd09de743758dafcf16d6b589cc72be979cce376a7f0ee72c7ccff34ffb2 /bin/sh
Hello World!
~
"test.txt" 1L, 13C                                                        1,12          All
```

Figure 2: Text file to test that the volume was mounted to the container correctly.

Use the command

```
$ exit
```

to close the container terminal, and click the "STOP" button to close the container itself. Then restart the container, and use the

```
$ ls
```

command to verify that the test.txt file is still there. If the test.txt file is not accessible, it means that the volume was not mounted correctly.

Note: This test can also be run in a Windows terminal. The container can be launched using the command

```
$ docker run -it -v [LOCAL_SWMM_FULLPATH]:/SWMM_5_Plus
    cimmresearch/swmm5plus-docker
```

where the [LOCAL_SWMM_FULLPATH] is the path to the directory where the SWMM5plus Volume is mounted. Unless you've navigated away from the working directory, this full path ought to be everything to the left of the commandline marker, ">".

1.1.2 Creating the SWMM5+ executable in Docker

Now that you have a Docker container with all necessary dependencies and a Volume mounted to store files, the next step is pulling the SWMM5+ code from the CIMM public github repository. From an active container (either in the Docker application GUI or in a Windows terminal), enter the command

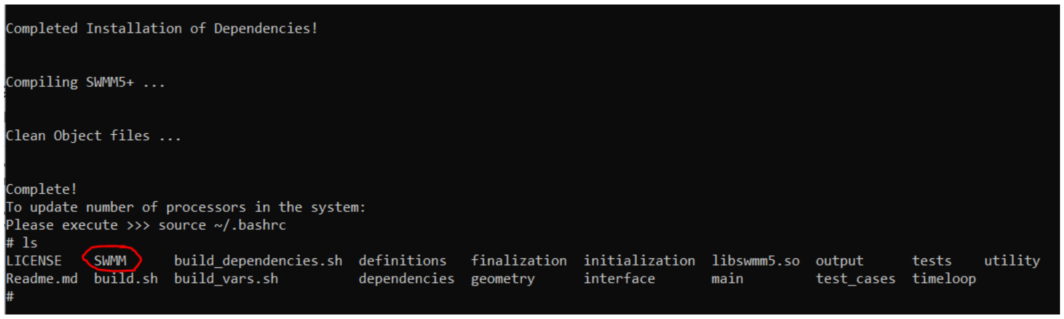
```
$ git clone https://github.com/CIMM-ORG/SWMM5plus.git
```

Navigate into the "SWMM5plus" directory, and build the SWMM5+ executable using

```
$ ./build.sh
```

Note: For more details pertaining to building the executable with optional flags, please see §3.2 of the main Beta Release Documentation.

Another $ ls command will reveal whether the executable (simply called "SWMM") has been created successfully.



```
Completed Installation of Dependencies!


Compiling SWMM5+ ...


Clean Object files ...


Complete!
To update number of processors in the system:
Please execute >>> source ~/.bashrc
# ls
LICENSE    SWMM      build_dependencies.sh  definitions   finalization  initialization  libswmm5.so  output      tests     utility
Readme.md  build.sh  build_vars.sh          dependencies  geometry      interface       main         test_cases  timeloop
#
```

Figure 3: Image of successfully compiled SWMM5+ executable. For more details about other files/folders in the SWMM5plus working directory, please see the main Beta Release Documentation.

1.2 Linux

Linux users have options for installing and running SWMM5+. The Docker Desktop app and workflow can be hosted on a Linux OS; the only difference would be selecting the appropriate Linux distribution from `https://www.docker.com/get-started`.

Alternatively, Linux users can install the requisite dependencies on their local OS to manage and run SWMM5+ without the "hassle" of dealing with containerization.

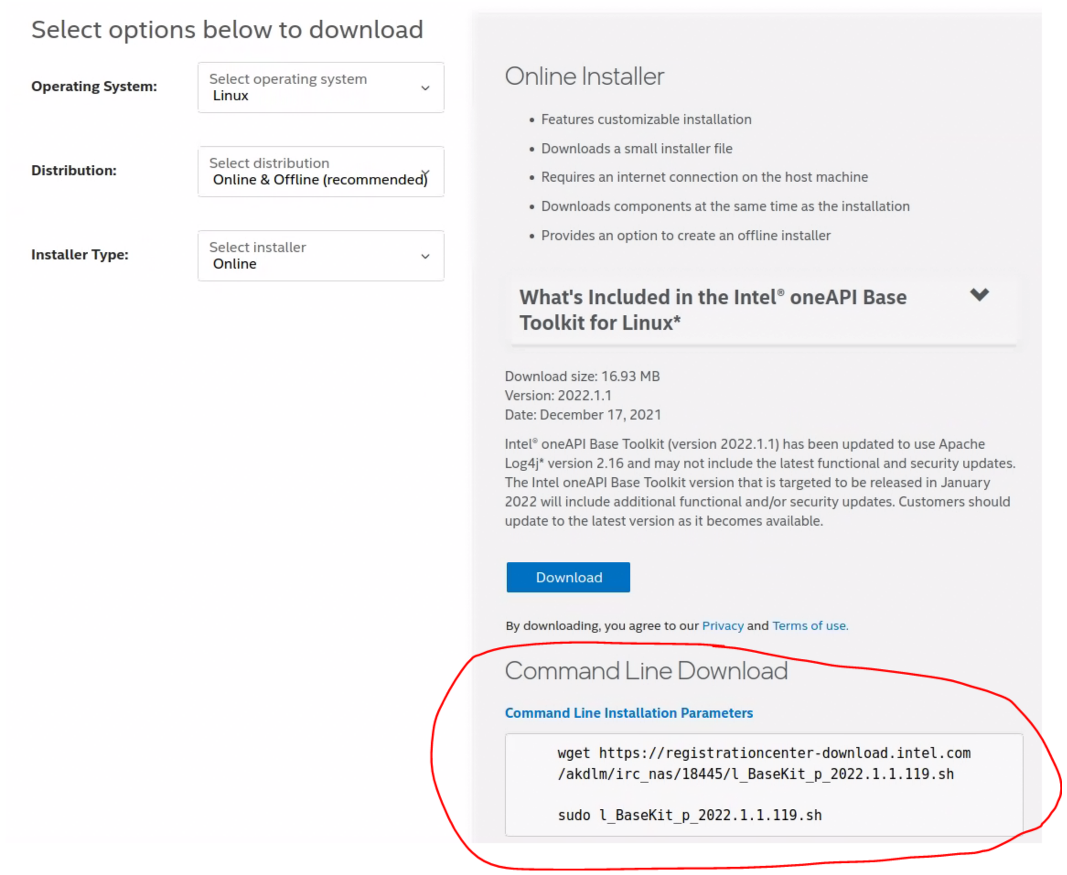
1.2.1 Installing Dependencies

The value of using the CIMM Docker container is that it comes preloaded with the dependencies needed to run SWMM5+. However, it is relatively straightforward to install these yourself on your native Linux OS.

Create a folder named "oneapi_install" where the dependencies will be installed and kept. Open a terminal in this new folder.

Open a web browser to `https://www.intel.com/content/www/us/en/developer/tools/oneapi/base-toolkit-download.html` and download the Intel oneAPI Base Toolkit for Linux. Select the Online & Offline distribution, and Online installer type.

The easier method (i.e. without requiring creating an Intel account) for downloading this toolkit is to use the Command Line Download instructions on the right side of the webpage.



Select options below to download

Operating System: Select operating system — Linux

Distribution: Select distribution — Online & Offline (recommended)

Installer Type: Select installer — Online

Online Installer

- Features customizable installation
- Downloads a small installer file
- Requires an internet connection on the host machine
- Downloads components at the same time as the installation
- Provides an option to create an offline installer

What's Included in the Intel® oneAPI Base Toolkit for Linux*

Download size: 16.93 MB
Version: 2022.1.1
Date: December 17, 2021

Intel® oneAPI Base Toolkit (version 2022.1.1) has been updated to use Apache Log4j* version 2.16 and may not include the latest functional and security updates. The Intel oneAPI Base Toolkit version that is targeted to be released in January 2022 will include additional functional and/or security updates. Customers should update to the latest version as it becomes available.

By downloading, you agree to our Privacy and Terms of use.

Command Line Download

Command Line Installation Parameters

```
wget https://registrationcenter-download.intel.com
/akdlm/irc_nas/18445/l_BaseKit_p_2022.1.1.119.sh

sudo l_BaseKit_p_2022.1.1.119.sh
```

Figure 4: Copy and paste the commands from this section to download the Intel oneAPI Base Toolkit. **Note:** this requires nearly 30GB of disk space.

Note: For me, the second command did not work, but the same effect could be achieved with

```
$ sudo sh ./l_Basekit_p_2022.1.1.119.sh
```

Follow the installation wizard, ignore warnings about GUI packages or IDE's not being installed. This will take several minutes.

The next package required is the Intel oneAPI HPC Toolkit, which can be downloaded from `https://www.intel.com/content/www/us/en/developer/tools/oneapi/hpc-toolkit-download.html`. Install the HPC Toolkit using the same parameters as the Base Toolkit.

Once both Intel oneAPI Toolkits have been installed, we need to update the system variables using the command

```
    $ source [PATH_to_oneAPI]/setvars.sh
```

Where the [PATH_to_oneAPI] is the location where the oneAPI Toolkits were installed. If you used the autoinstallation wizard, it should tell you the installation location at the bottom left of the wizard. If the installation wizard has already closed, you can retrieve the path by repeating the command

```
    $ sudo sh ./l_Basekit_p_2022.1.1.119.sh
```

which will direct you to your installed Intel oneAPI packages.

To check that the installation of the Intel oneAPI Toolkits and the resetting of the system variables was completed correctly, execute the command

```
    $ ifort -v
```

The return should be ifort version 2021.5 or higher. If the command return says ifort version 18.0.2 (or something similar) that means there was an issue with the dependency installation that needs to be rectified before moving on.

1.2.2 Creating the SWMM5+ executable in Linux

Open a command terminal in the working directory where you want to manage the SWMM5+ code. Then pull the code from the CIMM public github repository.

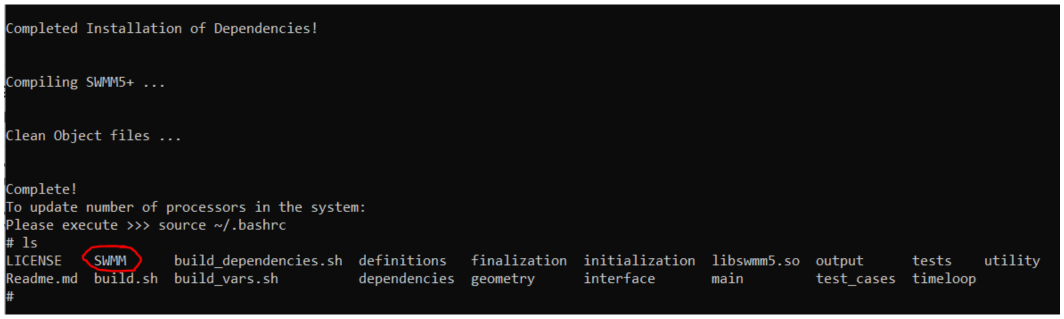
```
    git clone https://github.com/CIMM-ORG/SWMM5plus.git
```

Navigate into the "SWMM5plus" directory, and build the SWMM5+ executable using

```
    $ ./build.sh
```

Note: For more details pertaining to building the executable with optional flags, please see §3.2 of the main Beta Release Documentation.

An $ ls command will reveal whether the executable (simply called "SWMM") has been created successfully.



```
Completed Installation of Dependencies!


Compiling SWMM5+ ...


Clean Object files ...


Complete!
To update number of processors in the system:
Please execute >>> source ~/.bashrc
# ls
LICENSE    SWMM      build_dependencies.sh  definitions   finalization  initialization  libswmm5.so  output      tests     utility
Readme.md  build.sh  build_vars.sh          dependencies  geometry      interface       main         test_cases  timeloop
#
```

Figure 5: Image of successfully compiled SWMM5+ executable. For more details about other files/folders in the SWMM5plus working directory, please see the main Beta Release Documentation.

1.3 Additional Resources

Questions or problems pertaining to the installation of SWMM5+ on Windows or Linux computers can be directed towards PhD students/SWMM5+ Developers, Edward Tiernan and Gerardo Riaño-Briceño.

Edward Tiernan: etiernan@utexas.edu
Gerardo Riaño-Briceño: griano@utexas.edu