# Docker: Windows host with Linux container

## Methodology

The basic setup is to run docker desktop on Windows (10) OS host and container with Linux OS (Ubuntu 16.04)

### UI Inputs

The default installed Ubuntu OS was expected to run mainly via commands only, to build capability for UI related operations, a light weight package (xfce) for UI is installed and ran through a vnc viewer for display and interaction.

## Prerequisite

-Download docker desktop from Dockerhub and install   
(<https://docs.docker.com/docker-for-windows/install/>)

-Enable Hyper-V

-Register an account with docker

## Docker Settings

TBA

## Creating and running a Container (specific OS)

Run command prompt

|  |
| --- |
| C:\Windows\system32> docker run -it ubuntu:16.04 |

Docker will search if there is a local container called ubuntu16.04, if no existing container with name found, it will search in database for the specific name and download if available. Search will fail is nothing is found.

Once the container is loaded/downloaded, you will enter into the container it as **root** like ssh and can command it like a regular linux terminal. At this point we have a bare minimum installation of Ubuntu where we can install the packages which we require. Note that you are **root** account.

|  |
| --- |
| root@a92f38e003c3:/# | |

### Update

|  |
| --- |
| root@a92f38e003c3:/# apt-get update && apt-get upgrade -y |

### Package installations

|  |
| --- |
| root@a92f38e003c3:/# apt-get install firefox  root@a92f38e003c3:/# apt-get install meshlab |

Installation of any packages are exactly the same with using the specific OS, example of installing firefox and meshlab.

### Exit and save the container instance into a named image (**without VNC** options enabled)

The container pauses in the very instance when you exit, we can save the very instance into an image so we can run it again and continue from the instance. We can check all the instances with the docker ps –a command and saved images with docker images command.

|  |
| --- |
| root@a92f38e003c3:/# exit  exit  C:\Windows\system32> docker ps –a  CONTAINER ID IMAGE COMMAND CREATED Status PORTS  343fd1b24cef ubuntu:16.04 “/bin/bash” ..ago Exited.. savvy\_skymatch  C:\Windows\system32> docker commit 343fd1b24cef my\_image:v1  sha256:d617d707d509b7980b3310fea8c50be  C:\Windows\system32> docker images  REPOSITORY TAG IMAGE ID CREATED SIZE  my\_image v1 d617d707d5 ..ago 280MB  C:\Windows\system32> |

### Running your save image (**without VNC** options enabled)

To run the saved image, we use the run command again but with the saved image name.

|  |
| --- |
| C:\Windows\system32> docker run –it my\_image:v1  root@a92f38e003c3:/# | |

### Specific options for visual display through VNC

|  |
| --- |
| ~~root@a92f38e003c3:/# apt-get install xfce4 xfce4-goodies~~  root@a92f38e003c3:/# DEBIAN\_FRONTEND=noninteractive apt-get install xfce4 -y --fix-missing  root@a92f38e003c3:/# apt-get install xfce4-goodies –y  root@a92f38e003c3:/# apt-get install wget  root@a92f38e003c3:/# wget https://bintray.com/tigervnc/stable/download\_file?file\_path=ubuntu-16.04LTS%2Famd64%2Ftigervncserver\_1.7.0-1ubuntu1\_amd64.deb  root@a92f38e003c3:/#  root@a92f38e003c3:/# dpkg -i download\_file\?file\_path\=ubuntu-16.04LTS%2Famd64%2Ftigervncserver\_1.7.0-1ubuntu1\_amd64.deb  root@a92f38e003c3:/# apt-get install –f -y |

To enable and visualise the OS with a gui, we install the UI package of xfce4 and xfce4-goodies for the UI and

We can load the vncserver with the following command, but since we had not forward the port previously, the vncserver does not display into a readable port for us now.

|  |
| --- |
| root@a92f38e003c3:/# vncserver :1 –geometry 1920x1080 –SecurityTypes None |

### Exit and save the container instance into a named image (**with VNC** options enabled)

The container pauses in the very instance when you exit, we can save the very instance into an image so we can run it again and continue from the instance. We can check all the instances with the docker ps –a command and saved images with docker images command.

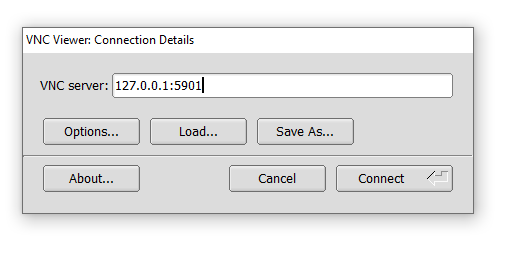
|  |
| --- |
| root@a92f38e003c3:/# vncserver –kill :1  root@a92f38e003c3:/# rm –rf /tmp/\*  root@a92f38e003c3:/# rm –rf /tmp/.\*  root@a92f38e003c3:/# exit  exit  C:\Windows\system32> docker ps –a  CONTAINER ID IMAGE COMMAND CREATED Status PORTS  546f44e3c4ef my\_image:v1 “/bin/bash” ..ago Exited.. happy\_groundhog  C:\Windows\system32> docker commit 546f44e3c4ef my\_image:v2  sha256:d62d70c5c504e3940b3310fea8c50be  C:\Windows\system32> docker images  REPOSITORY TAG IMAGE ID CREATED SIZE  my\_image v2 d62d70c5c5 ..ago 460MB  my\_image v1 d617d707d5 ..ago 280MB  C:\Windows\system32> |

### Running your save image (with VNC options enabled)

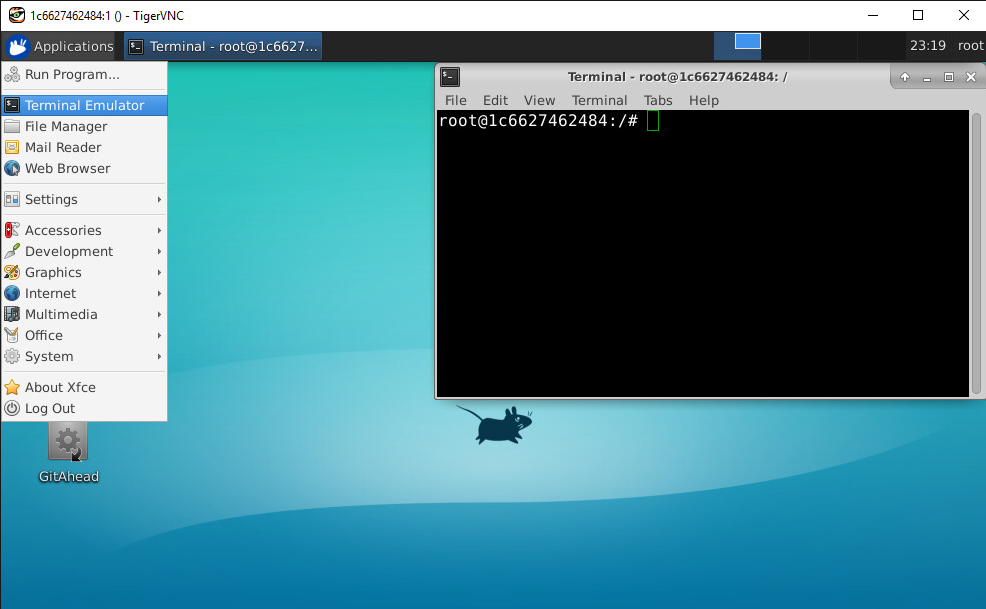
To run the saved image, we use the run command with saved image name and a few options enabled, namely, the memory fix expanding to 2g to fix loading issues and the port forward to 5901 to allow vnc to display into port.

|  |
| --- |
| C:\Windows\system32> docker run --shm-size 2g –p 5901:5901 –it my\_image:v2  root@a92f38e003c3:/# vncserver :1 –geometry 1920x1080 –SecurityTypes None  root@a92f38e003c3:/# | |

We can run vncviewer (Tiger VNC) with address 127.0.0.1:5901 and should be able to enter the UI through the vncviewer.



At this point, we can open Applications, install programs and operate the system just like a regular linux system with the UI.

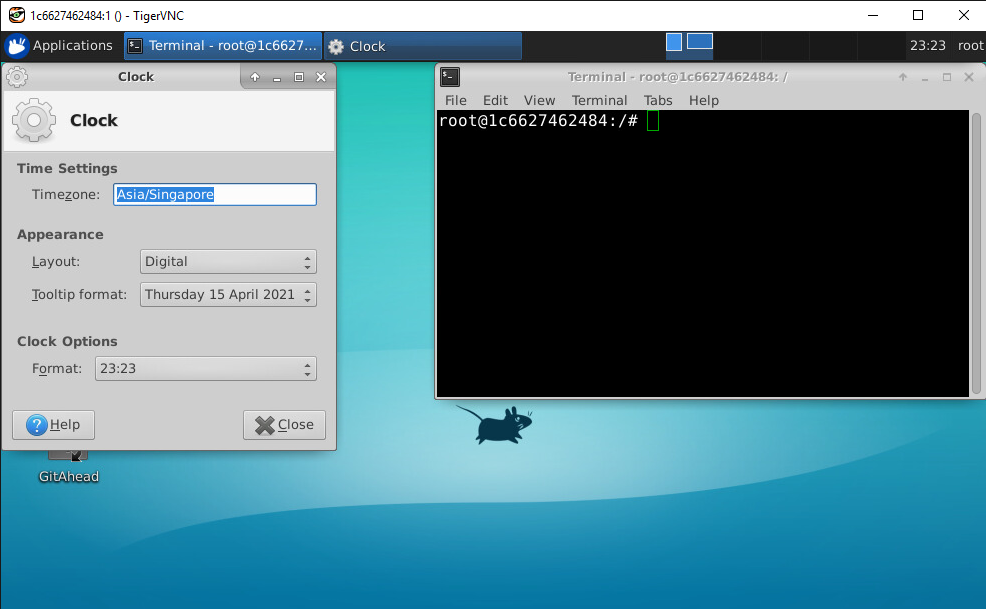


### Share directory between host and container, mounting an external volume

d

### Troubleshoot for time error

We can first set the time zone to the correct timing by right clicking the Clock -> Properties, set time zone to specific zone (e.g. Asia/Singapore)



If the timing is not in sync with your host clock, exit the container and run the hardware clock sync command.

|  |
| --- |
| C:\Windows\system32> docker run --rm --privileged my\_image:v2 hwclock -s |

### Housekeeping for data usage

We can run an image in a “temp” mode where no data or image instance is saved upon exit using the --rm option, this is useful when you have a working version of the container and do not need to develop or save the container instance further.

|  |
| --- |
| C:\Windows\system32> docker run --rm --shm-size 2g –p 5901:5901 –it my\_image:v2  root@a92f38e003c3:/# vncserver :1 –geometry 1920x1080 –SecurityTypes None  root@a92f38e003c3:/# | |

Periodically, check for floating containers (containers which are not committed) and clear them to save space.

|  |
| --- |
| C:\Windows\system32> docker ps –a  CONTAINER ID IMAGE COMMAND CREATED Status PORTS  546f44e3c4ef my\_image:v1 “/bin/bash” ..ago Exited.. happy\_groundhog  343fd1b24cef ubuntu:16.04 “/bin/bash” ..ago Exited.. savvy\_skymatch  C:\Windows\system32> docker system prune  WARNING! This will remove:  - all stopped containers  - all networks not used by at least one container  - all dangling images  - all dangling build cache  Are you sure you want to continue? [y/N] y  Deleted Containers:  d62d70c5c504e3940b3310fea8c50be  d617d707d509b7980b3310fea8c50be  Total reclaimed space: 514MB  C:\Windows\system32> |

## Using dockerfile for running or launching application instantly

We can use script and preset commands for launching docker straight into application

## Save and load container to be used on other PC

To save the current image into a tar file and load it on another PC. To save (or output) the image, run the save -o command, to load, run the load -i command.

|  |
| --- |
| (on pc with existing image)  C:\Windows\system32> docker save -o C:/myfolder/myfile.tar my\_image:v2  (on new PC)  C:\Windows\system32> docker load -i C:/myfolder/myfile.tar |

## Alternative and observations

VcXsrv X Server, display has high latency and operation has a significant lag.

Code and compiling workspace in external volume seems to load much slower compared to workspace within the container directory. Source code build time was 8:30 min vs 2.58 min for ATG source. Region growing segmentation also runs much faster when src is inside container.