Running MoveIt 2 with LXD

For many ROS users one of the biggest hurdles to testing ROS 2 is that it requires a lot of effort to set everything up. The MoveIt 2 Foxy release requires Ubuntu 20.04 with `ros-foxy-desktop` installed which many users just don't have the time or capacity next to their usual work. The LXD image `moveit2-foxy` provides a precompiled MoveIt 2 Foxy workspace in a Ubuntu 20.04 environment that allows you to start testing and debugging right away. For that, all you need to do is to pull the LXD image and launch it with a `gui` profile as described in this step-by-step tutorial.

1. Install and setup LXD

Following the LXD Getting Started tutorial, install and initialize LXD on your host system (here Ubuntu 18.04, see tutorial for alternative platforms). Please make sure to use the latest snap version and not the Ubuntu debian.

```
snap install lxd
lxd init
```

The second command will allow you to interactively configure features like network access and storage usage for your lxd setup. For starters, defaults should work just fine.

Import the MoveIt 2 LXD Image

Now you can download the MoveIt 2 Image from here and import it into your lxd register using the LXD's command line client **lxc**.

```
lxc image import moveit2-foxy.tar.gz --alias moveit2-foxy
```

You can verify all your imported images using:

```
lxc image list
```

2. Apply LXD GUI Image

In order to run RViz or any other GUI applications from inside the LXD container the hosts X11 session needs to be made accessible by applying an LXD launch profile (from https://blog.simos.info/how-to-easily-run-graphics-accelerated-gui-apps-in-lxd-containers-on-your-ubuntu-desktop/)

Create a new file gui.txt with the following content:

```
config:
  environment.DISPLAY: :0
  raw.idmap: both 1000 1000
  user.user-data: |
    #cloud-config
    runcmd:
      - 'sed -i "s/; enable-shm = yes/enable-shm = no/g" /etc/pulse/client.conf'
      - 'echo export PULSE_SERVER=unix:/tmp/.pulse-native | tee --append /home/ubuntu/.profile'
    packages:
      - x11-apps
      - mesa-utils
      - pulseaudio
description: GUI LXD profile
devices:
  PASocket:
    path: /tmp/.pulse-native
    source: /run/user/1000/pulse/native
    type: disk
  X0:
    path: /tmp/.X11-unix/X0
    source: /tmp/.X11-unix/X0
    type: disk
  mygpu:
    type: gpu
name: gui
used_by:
```

Create a new LXD profile named `gui`:

lxc profile create gui
cat gui.txt | lxc profile edit gui

3. Launch the LXD container

Launch the image `moveit2-foxy` as container `moveit2-foxy` with profiles `default` and `gui`:

lxc launch --profile default --profile gui moveit2-foxy moveit2-foxy

and open a bash shell in the running container instance as user `ubuntu` with sudo permissions:

lxc exec moveit2-foxy -- sudo --user ubuntu --login

Verify GUI is working by running **glxgears.** In case **glxgears** is not available or the command fails, see the GUI Troubleshooting section below.

Once you are logged in, you will find the precompiled and source ROS2 workspace directory inside `~/ws_ros2`. Now you are ready to start running the demos (MoveItCpp, MoveGroup, MoveIt Servo) as for instance:

ros2 launch run_moveit_cpp run_moveit_cpp.launch.py

GUI Troubleshooting

Below are the most likely reasons why GUI support doesn't work. For all fixes it might be necessary to restart the container instance or even to reapply the `gui` profile.

Wrong Display Device

Verify that $DISPLAY is set to the same value for host and container. The default value is ":0", if the host uses a different display, you need to apply the same value to your gui profile.

Missing Packages
In some cases cloud-init silently fails to install the necessary packages 'mesa-utils', 'x11-apps' and 'pulseaudio'. You can simply add them by hand from inside your container shell:

sudo apt install mesa-utils x11-apps pulseaudio

Nvidia Troubleshooting

On some Nvidia systems it can happen that OpenGL is not accessible in the container. In that case the easiest solution is to simply install the libraries with **the same** NVidia version as installed on the host system, for example like below:

In your host system, check the currently installed driver version:

apt list --installed nvidia-driver-*

-> **nvidia-driver-440/...**

Install GL library in your container instance:

sudo apt install libnvidia-gl-440

You can also add the gl library to your lxc gui profile (under config/user.user-data/packages), you just need to make sure to always use a version that matches your host system.