Devis Dal Moro Università degli Studi di Trento

Setup ROS2 Foxy & Plansys2 on Ubuntu

Introduction

In this document, I'm going to briefly describe the requirements and the process to install ROS2 Foxy with the additional Plansys2 packages on Ubuntu.

Requirements

I assume Ubuntu is already installed on your machine or on a virtual machine of yours. I'd highly recommend to have **Ubuntu 20.04 LTS** installed and not other versions, since in the official ROS2 Foxy documentation for supported platforms that's the platform which is said to be supported. Note that you can have ROS2 Foxy on other platforms too (e.g. Windows 10 and MacOs), but the process won't be described here. Moreover, the end result wrt. the full-working of Plansys2 too needs to be properly tested.

(Optional): It's highly suggested to have also:
- proper code editors with support for C++, Python and CMake files (eg. VSCode with C++, Python extensions by Microsoft and CMake extension by twxs)
- a tool which allows you to have multiple terminals in one window (e.g. Terminator) for future usages
- python3-pip install (prompt into the terminal `sudo apt install python3-pip`)

Installing ROS2 Foxy Fitzroy

In this guide we're going to follow the official documentation[1] of ROS2 Foxy wrt. the installation process on Ubuntu via Debian Packages. You won't need to perform ALL those actions in the guide, so here we're going to focus exclusively on what you need to do.

1. **Set locale** to make sure you have a locale which supports UTF-8. In order to do that type separately in the following order these commands:

```
locale  # check for UTF-8

sudo apt update && sudo apt install locales
sudo locale-gen en_US en_US.UTF-8
sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
export LANG=en_US.UTF-8

locale  # verify settings
```

[1] Updated to the 06/21: if something does not work, check the official steps

2. Set up the **sources**:

```
sudo apt update && sudo apt install curl gnupg2 lsb-release

sudo curl -sSL
https://raw.githubusercontent.com/ros/rosdistro/master/ros.key  -o
/usr/share/keyrings/ros-archive-keyring.gpg
```

3. Add the repository to your sources list:

```
echo "deb [arch=$(dpkg --print-architecture)
signed-by=/usr/share/keyrings/ros-archive-keyring.gpg]
http://packages.ros.org/ros2/ubuntu $(lsb_release -cs) main" | sudo tee
/etc/apt/sources.list.d/ros2.list > /dev/null
```

4. Do an update and then install the desktop version[2]: it's going to take at least a few minutes since you're downloading 2.5GB of data, so go take a coffee ;-)

```
sudo apt update

sudo apt install ros-foxy-desktop
```

5. Install argcomplete

```
sudo apt install python3-pip
sudo apt install -y python3-argcomplete
pip3 install -U argcomplete
```

6. Now ROS2 should be successfully installed, however it's highly recommended to follow also this step. In order to use ROS2 in a terminal session, you would need to source every time its setup file, which has to be found in /opt/ros/foxy/setup.bash. To avoid to prompt this every time you open a new shell, put at the end of the **~/.bashrc** file the following command: *`source /opt/ros/foxy/setup.bash`*

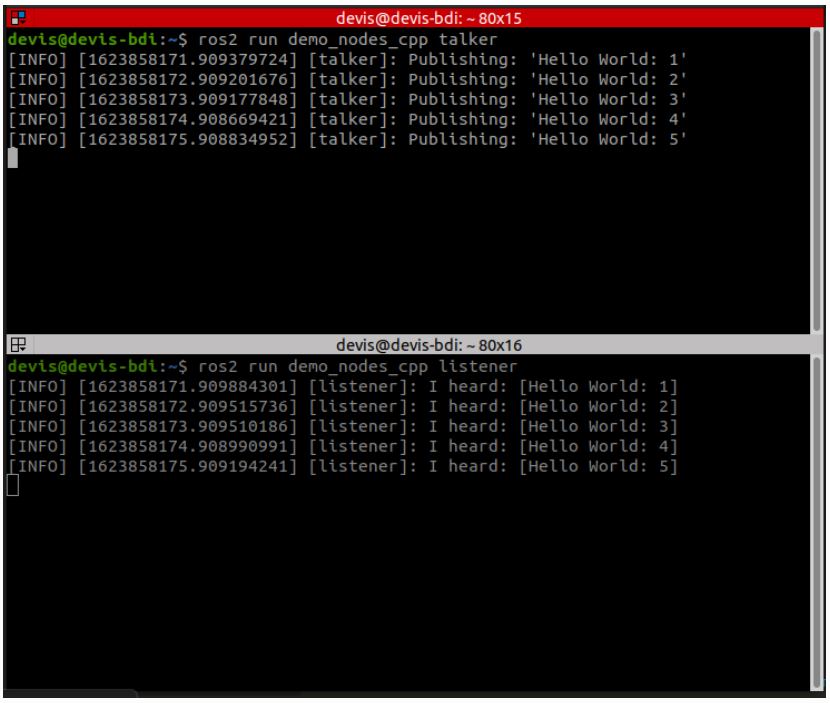
[2] If you look at the official documentation, you can see that it's possible to install also a more basic version without GUI tools, less pre-installed packages and so on. For the purpose of playing just with plansys2 actually the latter could be sufficient, but you're missing out on a lot of interesting packages that can come useful along your experience with ROS2. Searching for them and installing those packages later manually could be a struggle which is not worth wrt. an initial saving in memory space.

7. In order to check that ROS2 Foxy is correctly installed you can launch your first two nodes in ROS2 taking the executables from the demo set which already come with the desktop version.
   So prompt in two different terminals the commands to launch them:

```
ros2 run demo_nodes_cpp talker
```

```
ros2 run demo_nodes_cpp listener
```

   The first node publishes “Hello world” messages with a concatenated incremental number and the other one basically hears and prints out the same message.
   You should get a result like the following:



```
devis@devis-bdi: ~ 80x15
devis@devis-bdi:~$ ros2 run demo_nodes_cpp talker
[INFO] [1623858171.909379724] [talker]: Publishing: 'Hello World: 1'
[INFO] [1623858172.909201676] [talker]: Publishing: 'Hello World: 2'
[INFO] [1623858173.909177848] [talker]: Publishing: 'Hello World: 3'
[INFO] [1623858174.908669421] [talker]: Publishing: 'Hello World: 4'
[INFO] [1623858175.908834952] [talker]: Publishing: 'Hello World: 5'
```

```
devis@devis-bdi: ~ 80x16
devis@devis-bdi:~$ ros2 run demo_nodes_cpp listener
[INFO] [1623858171.909884301] [listener]: I heard: [Hello World: 1]
[INFO] [1623858172.909515736] [listener]: I heard: [Hello World: 2]
[INFO] [1623858173.909510186] [listener]: I heard: [Hello World: 3]
[INFO] [1623858174.908990991] [listener]: I heard: [Hello World: 4]
[INFO] [1623858175.909194241] [listener]: I heard: [Hello World: 5]
```

Installing Colcon argcomplete (optional)

This step could be seen as optional if you do not intend to develop your own ROS2 nodes, but otherwise consider it as almost essential, since colcon is the main tool you’ll need to use for compiling and building your executables.
The following will get you the auto-completion feature for colcon.

Prompt into the terminal:

```
sudo apt install python3-colcon-common-extensions
```

Add the following line at the end of the ~/.bashrc file:
source /usr/share/colcon_argcomplete/hook/colcon-argcomplete.bash

Setup your ROS2 workspace

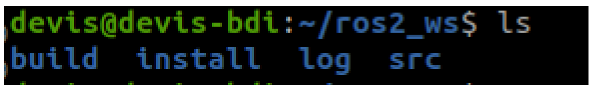
Decide where you'd like to set up your ROS2 workspace, where you're going to put the source and build files of your nodes, as well as ROS2 logs, bags,... Note that in the source folder we're going to put then the plansys2 packages (or other third-party ROS2 packages that we'll need to compile from source) which we'll need to compile from source using colcon, so it's highly recommended to not skip this passage.
Let's say that you create a ros2_ws folder on your home directory for the workspace:

```
cd
mkdir ros2_ws && cd ros2_ws
mkdir src
```

Now you should be inside the (just created) directory ros2_ws which contains the empty directory src.
Type "colcon build" to do a first local building of your packages (even if you still have none).

```
colcon build
```

The build should be successful and you should seen the following folder now in ros2_ws:



```
devis@devis-bdi:~/ros2_ws$ ls
build  install  log  src
```

Under ~/ros2_ws/install/setup.bash there is another script which should be sourced every time you want to run the executables in your workspace after opening a terminal or having recompiled your packages. As done above, to reduce the overhead, I'd suggest to put it as a command to execute in ~/.bashrc: *source ~/ros2_ws/install/setup.bash*

Install Plansys2

Before directly building plansys2 I suggest to follow this quick guide to create an empty ROS2 cpp package and compile it. This will allow us to setup some environment variables needed for the compilation of plansys2 packages as well.

```
sudo apt install build-essential

cd ~/ros2_ws/src
ros2 pkg create my_cpp_pkg --dependencies rclcpp --build-type ament_cmake
cd ..
colcon build
```

Furthermore, you'll probably need to install some missing packages which are mandatory for plansys2 building. Do it using the available debian package sources:

```
sudo apt install ros-foxy-rclcpp-cascade-lifecycle ros-foxy-behaviortree-cpp-v3
ros-foxy-test-msgs ros-foxy-nav2-msgs
```

Then go into the src directory of your ROS2 workspace and clone the two repositories for the core and the examples of PlanSys2.

```
sudo apt install git
cd ~/ros2_ws/src
git clone https://github.com/IntelligentRoboticsLabs/ros2_planning_system.git
git clone https://github.com/IntelligentRoboticsLabs/ros2_planning_system_examples.git
```

Since plansys2 is still under development[3], some other packages might be missing too. In that case come back here and run the following commands to install all of them.

```
cd ~/ros2_ws
sudo apt install python3-rosdep2
rosdep update

rosdep install -y -r -q --from-paths src --ignore-src --rosdistro foxy
```

If you're running on Mint 20 (still based on ubuntu), add the following option to the last command `--os="ubuntu:"`

Now before jumping to the compilation of plansys2 packages,I source all the installation files! It's important to do it every time you add (via apt install) or you (re-)compile one or more ROS2 package(s). Since we've added a lot of them, it's a good idea to source everything back again. If you precisely follow this guide, you just need to prompt:

```
source ~/.bashrc
```

Now, go back again to the root of your workspace and compile everything.
It should take some time.

```
cd ~/.ros2_ws
colcon build --symlink-install
```

The building process should be successful!
Great, now you have ROS2 and PlanSys2 correctly installed and setup.
To make sure of that, run the first two tutorial examples following the instructions in the PlanSys2 webpage.

[3] at least at moment I'm writing this