iRobot™ Education

iRobot Create3 Setup Guide

Follow this document to install ROS2 and setup a ROS2 connection with the iRobot Create3. It is recommended that you follow the installation process outlined below, and refer to the documentation provided by ROS and iRobot for detailed instructions. Come back to this document for links and quick command reference as needed.

For ROS2 Rookies

If this is your first time using ROS2, it you will likely want read through/follow the beginner tutorials provided by ROS2 which introduce the `ros2` command line interface, package system, and `colcon` build system. These tutorials are linked in the documentation links below and are most helpful after installing ROS2. ROS2 is **not** an intuitive system at first glance with several quirks (cough *sourcing* cough *workspaces* cough) that may make it seem like black magic, but the tutorials and documentation go a long way towards pulling back to the curtain, or so to speak.

Table of Contents

Documentation Links

Link	Description
ROS2: Humble Hawksbill Documentation	Latest ROS2 version (as of Fall 2022)
ROS2: Humble CLI Tutorials	Introduction to command line tools provided by ROS2 for *debugging!*
ROS2: Humble Client Lib Tutorials	Introduction to ROS2 library creation and build systems.
ROS2: All Tutorials	Tutorials for all experience levels whereas the two above are introductory.
iRobot Create3: Getting Started Guide	Hardware documentation and setup guides for first steps.
iRobot Create3 Documentation	Detailed documentation for configuration and debugging.
iRobot Create3: APIs	ROS2 API

Setup Guide

ROS2 Installation

Install ROS2 ‘Humble’ or newer on a linux machine. For installation instructions and latest release go here. Make sure to install the latest *ROS2* version, not *ROS1*. They are both listed on the linked page. If you like, test your installation by running the `turtlesim` publish/subscribe example at the end of the ROS2 installation guide. A good follow-up to this example is following the CLI and Client Library tutorials that are linked in the documentation links above. These tutorials will continue to use `turtlesim` to illustrate the architecture used by ROS2 as well as introduce debugging tools.

If you have followed this guide before and simply want to install ROS2 for connecting to the Create3 as quickly as possible, see the install summary below.

Summary Installation summary for quick install:

1. Verify locale is UTF-8: `locale`
2. Verify Universe repository enabled on system
   - `apt-cache policy | grep universe`
   - enable Universe repository if not
     (a) `sudo apt install software-properties-common`
     (b) `sudo add-apt-repository universe`
3. Add ROS2 repository to system:
   (a) `sudo apt update && sudo apt install curl gnupg lsb-release`
   (b) `sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -o /usr/share/keyrings/ros-archive-keyring.gpg`
   (c) `echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/ros-archive-keyring http://packages.ros.org/ros2/ubuntu $(source /etc/os-release && echo $UBUNTU_CODENAME) main" | sudo tee /etc/apt/sources.list.d/ros2.list > /dev/null`
4. ***IMPORTANT:*** it is noted in ROS2 documentation that failure to update packages may cause catastrophic removal of vital system packages in Ubuntu 22.04 unless a package update and upgrade is performed.

   ```
   sudo apt update & sudo apt upgrade
   ```

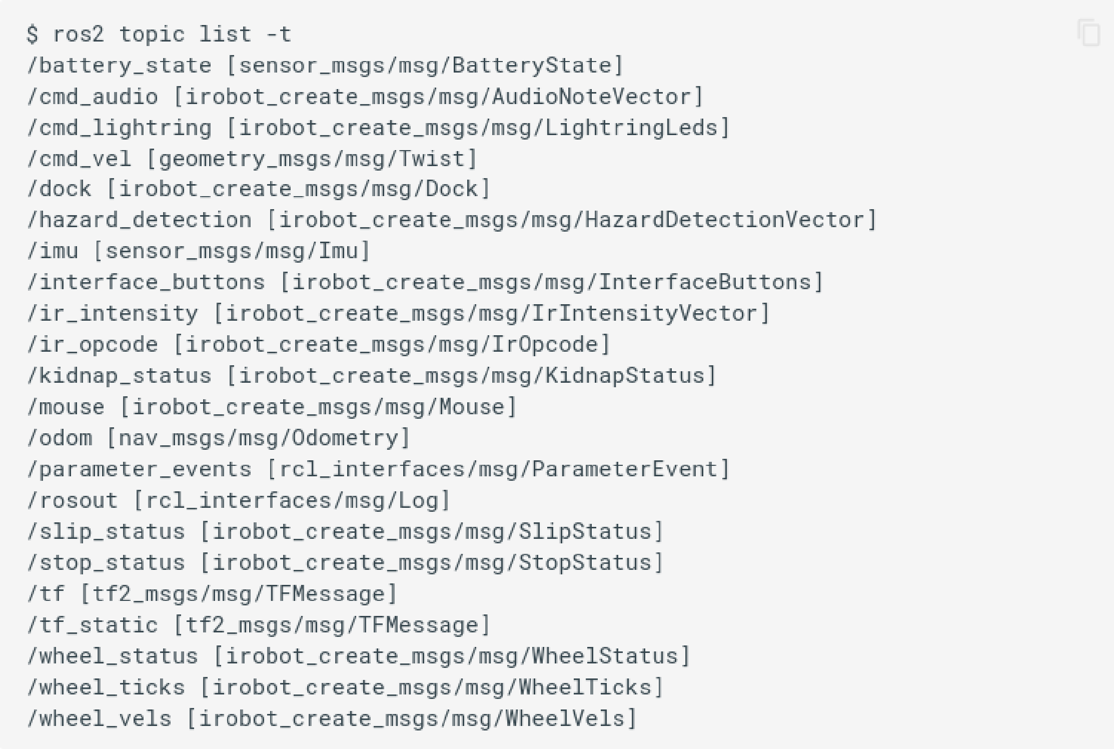
5. Install ROS2 with:

   ```
   sudo apt install ros-humble-desktop
   ```

6. Follow installation steps provided by iRobot’s documentation. *Make sure* you replace packages that include `-galactic-` with `-humble-`.
   - ***IMPORTANT DDS CONFIG:*** use CycloneDDS instead of FastRTPS, see step 4.

     ```
     echo "export RMW_IMPLEMENTATION=rmw_cyclonedds_cpp" >> ~/.bashrc‘
     ```

   - ***TO VERIFY:*** run `ros2 topic list`, should see something like:



```
$ ros2 topic list -t
/battery_state [sensor_msgs/msg/BatteryState]
/cmd_audio [irobot_create_msgs/msg/AudioNoteVector]
/cmd_lightring [irobot_create_msgs/msg/LightringLeds]
/cmd_vel [geometry_msgs/msg/Twist]
/dock [irobot_create_msgs/msg/Dock]
/hazard_detection [irobot_create_msgs/msg/HazardDetectionVector]
/imu [sensor_msgs/msg/Imu]
/interface_buttons [irobot_create_msgs/msg/InterfaceButtons]
/ir_intensity [irobot_create_msgs/msg/IrIntensityVector]
/ir_opcode [irobot_create_msgs/msg/IrOpcode]
/kidnap_status [irobot_create_msgs/msg/KidnapStatus]
/mouse [irobot_create_msgs/msg/Mouse]
/odom [nav_msgs/msg/Odometry]
/parameter_events [rcl_interfaces/msg/ParameterEvent]
/rosout [rcl_interfaces/msg/Log]
/slip_status [irobot_create_msgs/msg/SlipStatus]
/stop_status [irobot_create_msgs/msg/StopStatus]
/tf [tf2_msgs/msg/TFMessage]
/tf_static [tf2_msgs/msg/TFMessage]
/wheel_status [irobot_create_msgs/msg/WheelStatus]
/wheel_ticks [irobot_create_msgs/msg/WheelTicks]
/wheel_vels [irobot_create_msgs/msg/WheelVels]
```

If there fewer outputs, you may have a network problem blocking connection to the robot. Possible solutions may also include sourcing a ROS2 setup script after building the project (see `install/setup.sh` or `install/local_setup.sh`).

iRobot Create3 Setup

ROS2 communicates with the iRobot Create3 via a 2.4GHz network so we need to configure it to connect to the network we will be using. It will also be important to configure the robot with a unique namespace so that it can operate on the same network as other robots. To do so, follow the written instructions in this guide and give it a namespace of the pattern `/create3-AP_NAME` (`create3-05F8`) where `AP_NAME` is the last four digits of the SSID broadcast by the robot. If this doesn't make sense, this should happen in section two, step two of the guide above (or see step 5 below).

IMPORTANT:

If you have assigned a namespace to your robot, all actions that you send to the robot **must start with the namespace**. For example, if your robot has the namespace `/create3_05F8` sending the dock command below will **not** work:

```
ros2 action send_goal /dock irobot_create_msgs/action/DockServo "{}"
```

But this will:

```
ros2 action send_goal /create3_05F8/dock irobot_create_msgs/action/DockServo "{}"‘
```

Summary A summary of the configuration in that guide is included below (no firmware update):

1. With robot docked, press and hold buttons on either side of the power button until light turns blue and you hear a beep.
2. Connnect to `Create-XXXX` network from laptop/desktop.
3. Open browser and navigate to access point(AP) `192.168.10.1`. This access point can later be reached if you know the IP address of the robot which can be found by scanning the network with Angry IP

and enabling the MAC address and MAC vendor fetchers.

4. Once reached, navigate to the 'Connect' page and use settings to connect to your desired network.
5. Finally, navigate to the Application configuration page and change namespace field to `/create3-XXXX` where `XXXX` matches the last four digits of the network SSID broadcast by the robot. In reality, this namespace can be anything you want, the convention is just for easy of use by others.

NOTE:

The steps above don't include firmware update, so make sure to do that first. Detailed configuration instructions and description can be found in the iRobot Create3 Documentation in the table above.

Verification

The final step is to verify connection between your laptop and the robot. To do this we will install and run the coverage example from iRobot's Create3 examples repository. Follow the instructions in the readme of that repository to create and install a new workspace, then follow the directions in `/create3_coverage` to run the example. If successful, your robot will undock and begin driving across the floor until it hits an obstacle.

Summary

1. ***Installation***, do once.

```
mkdir -p create3_examples_ws/src
cd create3_examples_ws/src
git clone https://github.com/iRobotEducation/create3_examples.git
cd ..
rosdep install --from-path src --ignore-src -yi
colcon build
```

NOTE:

ROS2 documentation recommends using a separate terminal for building a project and running/modifying the project to prevent strange errors that may arise from the build process. Simply put, open a new terminal window before proceeding.

1. ***Initialization***, must be done every time a terminal/session is opened for working on this project.

```
source ~/create3_examples_ws/install/local_setup.sh
```

1. In first terminal run:

```
ros2 run create3_coverage create3_coverage
```

1. In a second terminal:

```
 ros2 action send_goal /coverage create3_examples_msgs/action/Coverage "{explore_duration:{sec: 500
```