Table of Contents

Alias foxy or noetic
harman@harman-VirtualBox:~$ foxy

harman@harman-VirtualBox:~$ ros2

usage: ros2 [-h] Call `ros2 <command> -h` for more detailed usage. ...

ros2 is an extensible command-line tool for ROS 2.

optional arguments:
  -h, --help   show this help message and exit

Commands:
  action   Various action related sub-commands
  bag   Various rosbag related sub-commands
  component Various component related sub-commands
  daemon   Various daemon related sub-commands
  doctor   Check ROS setup and other potential issues
  interface Show information about ROS interfaces
  launch   Run a launch file
  lifecycle Various lifecycle related sub-commands
  multicast Various multicast related sub-commands
  node   Various node related sub-commands
  param   Various param related sub-commands
  pkg   Various package related sub-commands
  run   Run a package specific executable
  security   Various security related sub-commands
  service   Various service related sub-commands
  topic   Various topic related sub-commands
  wtf   Use `wtf` as alias to `doctor`

  Call `ros2 <command> -h` for more detailed usage.

PACKAGES

harman@harman-VirtualBox:~$ ros2 pkg -h

usage: ros2 pkg [-h] Call `ros2 pkg <command> -h` for more detailed usage. ...
Various package related sub-commands
optional arguments:
  -h, --help   show this help message and exit
Commands:
  create   Create a new ROS2 package
  executables Output a list of package specific executables
  list   Output a list of available packages
  prefix   Output the prefix path of a package
  xml   Output the XML of the package manifest or a specific tag

ros2 pkg list | less

(Use Space Bar to page)

harman@harman-VirtualBox:~$ ros2 pkg list | grep turtle

turtlesim

NODES

harman@harman-VirtualBox:~$ ros2 node -h

usage: ros2 node [-h]
Call `ros2 node <command> -h` for more detailed usage. ...
Various node related sub-commands
optional arguments:
-h, --help show this help message and exit
Commands:
info Output information about a node
list Output a list of available nodes

harman@harman-VirtualBox:~$ ros2 node list

harman@harman-VirtualBox:~$ ros2 node list -a

/_ros2cli_daemon_0

ROS2 RUN

harman@harman-VirtualBox:~$ ros2 run -h

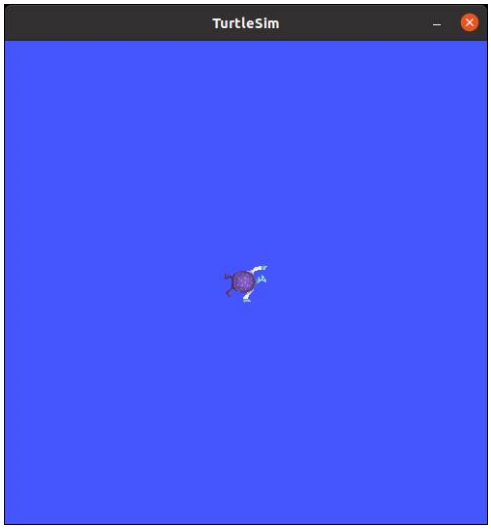
usage: ros2 run [-h] [--prefix PREFIX] package_name executable_name ...

Run a package specific executable

positional arguments:
package_name Name of the ROS package
executable_name Name of the executable
argv Pass arbitrary arguments to the executable

optional arguments:
-h, --help show this help message and exit
--prefix PREFIX Prefix command, which should go before the executable.
Command must be wrapped in quotes if it contains spaces
(e.g. --prefix 'gdb -ex run –args').

harman@harman-VirtualBox:~$ ros2 run turtlesim turtlesim_node



harman@harman-VirtualBox:~$ foxy (New Terminal)

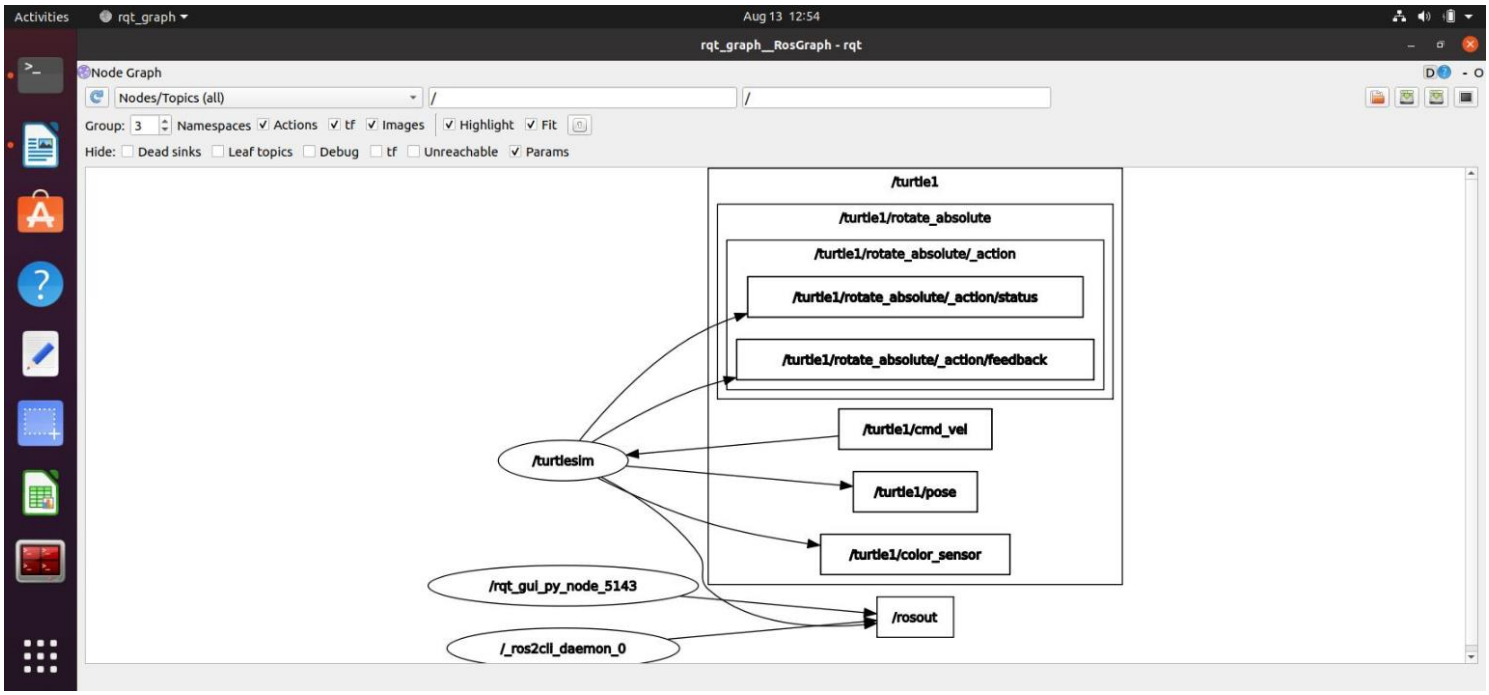
harman@harman-VirtualBox:~$ ros2 node info /turtlesim

/turtlesim
  Subscribers:
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /turtle1/cmd_vel: geometry_msgs/msg/Twist
  Publishers:
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /rosout: rcl_interfaces/msg/Log
    /turtle1/color_sensor: turtlesim/msg/Color
    /turtle1/pose: turtlesim/msg/Pose
  Service Servers:
    /clear: std_srvs/srv/Empty
    /kill: turtlesim/srv/Kill
    /reset: std_srvs/srv/Empty
    /spawn: turtlesim/srv/Spawn
    /turtle1/set_pen: turtlesim/srv/SetPen
    /turtle1/teleport_absolute: turtlesim/srv/TeleportAbsolute
    /turtle1/teleport_relative: turtlesim/srv/TeleportRelative
    /turtlesim/describe_parameters: rcl_interfaces/srv/DescribeParameters
    /turtlesim/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
    /turtlesim/get_parameters: rcl_interfaces/srv/GetParameters
    /turtlesim/list_parameters: rcl_interfaces/srv/ListParameters
    /turtlesim/set_parameters: rcl_interfaces/srv/SetParameters
    /turtlesim/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
  Service Clients:

  Action Servers:

/turtle1/rotate_absolute: turtlesim/action/RotateAbsolute
Action Clients:

rqt_graph



ROS2 TOPIC

harman@harman-VirtualBox:~$ ros2 topic -h

usage: ros2 topic [-h] [--include-hidden-topics]

Call `ros2 topic <command> -h` for more detailed usage. ...

Various topic related sub-commands

optional arguments:

-h, --help show this help message and exit

--include-hidden-topics

Consider hidden topics as well

Commands:

bw Display bandwidth used by topic

delay Display delay of topic from timestamp in header

echo Output messages from a topic

find Output a list of available topics of a given type

hz Print the average publishing rate to screen

info Print information about a topic

list Output a list of available topics

pub Publish a message to a topic

type Print a topic's type

harman@harman-VirtualBox:~$ ros2 topic list

/parameter_events

/rosout

/turtle1/cmd_vel

/turtle1/color_sensor

/turtle1/pose

harman@harman-VirtualBox:~$ ros2 topic list -t (Include Type)

/parameter_events [rcl_interfaces/msg/ParameterEvent]

/rosout [rcl_interfaces/msg/Log]

/turtle1/cmd_vel [geometry_msgs/msg/Twist] (NOTE Spelling)

/turtle1/color_sensor [turtlesim/msg/Color]

/turtle1/pose [turtlesim/msg/Pose]

harman@harman-VirtualBox:~$ ros2 topic type -h

usage: ros2 topic type [-h] topic_name

Print a topic's type

positional arguments:

topic_name Name of the ROS topic to get type (e.g. '/chatter')

harman@harman-VirtualBox:~$ ros2 topic type /turtle1/cmd_vel

geometry_msgs/msg/Twist (ros2 topic list -t Does this for all topics)

harman@harman-VirtualBox:~$ ros2 topic type /turtle1/color_sensor

turtlesim/msg/Color (NOTE spelling)

harman@harman-VirtualBox:~$ ros2 topic info /turtle1/cmd_vel

Type: geometry_msgs/msg/Twist

Publisher count: 0

Subscription count: 1

ROS2 Messages Turtlesim

$ rosmsg show std_msgs/Bool (bool data)

ROS 2 equivalent

$ ros2 interface show std_msgs/msg/Bool (bool data)

bool data

harman@harman-VirtualBox:~$ ros2 interface -h

usage: ros2 interface [-h]

Call `ros2 interface <command> -h` for more detailed

usage. ...

Show information about ROS interfaces

optional arguments:

-h, --help show this help message and exit

Commands:

list List all interface types available

package **Output a list of available interface types within one package**

packages Output a list of packages that provide interfaces

proto **Output an interface prototype**

show Output the interface definition

harman@harman-VirtualBox:~$ ros2 interface package turtlesim

(Notice action and service messages)

turtlesim/action/RotateAbsolute

turtlesim/srv/Spawn

turtlesim/srv/TeleportAbsolute

turtlesim/msg/Color

turtlesim/srv/SetPen

turtlesim/srv/TeleportRelative

turtlesim/msg/Pose

turtlesim/srv/Kill

harman@harman-VirtualBox:~$ ros2 interface list -h

usage: ros2 interface list [-h] [-m] [-s] [-a]

List all interface types available

optional arguments:

-h, --help show this help message and exit

-m, --only-msgs Print out only the message types

-s, --only-srvs Print out only the service types

-a, --only-actions Print out only the action types

harman@harman-VirtualBox:~$ ros2 interface list -m | grep turtlesim

turtlesim/msg/Color

turtlesim/msg/Pose

harman@harman-VirtualBox:~$ ros2 interface proto -h

usage: ros2 interface proto [-h] [--no-quotes] type

Output an interface prototype

positional arguments:

  type   Show an interface definition (e.g.   'example_interfaces/msg/String')

optional arguments:

  -h, --help   show this help message and exit

  --no-quotes if true output has no outer quotes.

harman@harman-VirtualBox:~$ ros2 topic list -t

/parameter_events [rcl_interfaces/msg/ParameterEvent]

/rosout [rcl_interfaces/msg/Log]

/turtle1/cmd_vel [geometry_msgs/msg/Twist]

/turtle1/color_sensor [turtlesim/msg/Color]

/turtle1/pose [turtlesim/msg/Pose]

harman@harman-VirtualBox:~$ ros2 interface show --help

usage: ros2 interface show [-h] type

Output the interface definition

positional arguments:

  type   Show an interface definition (e.g.

              'example_interfaces/msg/String'). Passing '-' reads the argument

              from stdin (e.g. 'ros2 topic type /chatter | ros2 interface show

              -').

harman@harman-VirtualBox:~$ ros2 interface show geometry_msgs/msg/Twist

This expresses velocity in free space broken into its linear and angular parts.

Vector3 linear

Vector3 angular

harman@harman-VirtualBox:~$ ros2 interface proto geometry_msgs/msg/Twist

"linear:

  x: 0.0

  y: 0.0

  z: 0.0

angular:

  x: 0.0

  y: 0.0

  z: 0.0

"

https://design.ros2.org/articles/ros_parameters.html#parameter-events

Parameter Events

Each node will provide a topic on which parameter events will be published. This topic is to support monitoring parameters for change. It is expected that client libraries will implement the ability to register callbacks for specific parameter changes using this topic.