19.ROS2 common command tools

1. Package management tool ros2 pkg

1.1.ros2 pkg create

Function: Create a function package. When creating, you need to specify the package name, compilation method, dependencies, etc.

Command format: ros2 pkg create --build-type ament_python pkg_name rclpy std_msgs sensor_msgs

ros2 pkg create: Instructions for creating packages

--build-type: If the newly created function package uses C++ or C, then write ament_cmake here. If it uses Python, write ament_python.

pkg_name: the name of the created function package

rclpy std_msgs sensor_msgs: These are some compilation dependencies

1.2.ros2 pkg list

Function: View the list of function packages in the system

Command format: ros2 pkg list

```
yahboom@yahboom-virtual-machine:~$ ros2 pkg list
action_msgs
action_tutorials_cpp
action_tutorials_interfaces
action_tutorials_py
actionlib_msgs
ament_cmake
ament_cmake_auto
ament_cmake_copyright
ament_cmake_core
ament_cmake_cppcheck
ament_cmake_cpplint
ament_cmake_export_definitions
ament_cmake_export_dependencies
ament_cmake_export_include_directories
ament_cmake_export_interfaces
ament_cmake_export_libraries
ament_cmake_export_link_flags
ament_cmake_export_targets
ament_cmake_flake8
ament_cmake_gmock
ament_cmake_gtest
ament_cmake_include_directories
ament_cmake_libraries
ament_cmake_lint_cmake
ament_cmake_pep257
ament_cmake_pytest
ament_cmake_python
ament_cmake_ros
ament_cmake_target_dependencies
```

1.3. ros2 pkg executeables

Command function: View the list of executable files in the package

Command format: ros2 pkg executables pkg_name

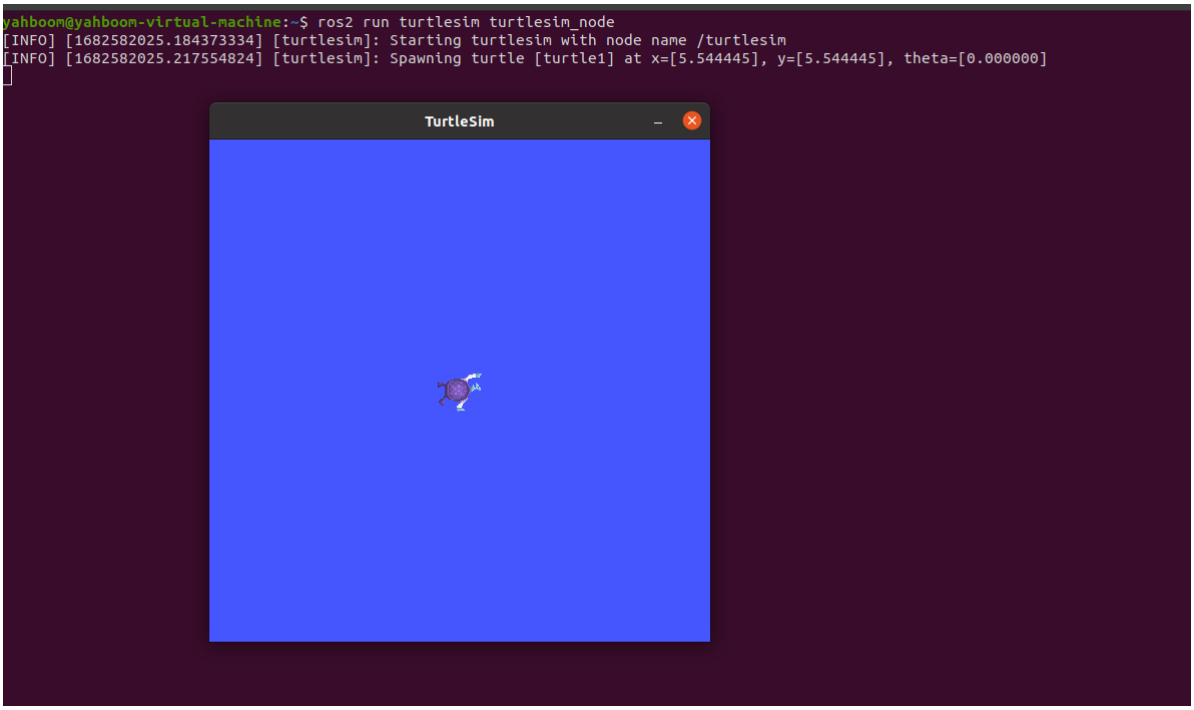
```
yahboom@yahboom-virtual-machine:~$ ros2 pkg executables turtlesim
turtlesim draw_square
turtlesim mimic
turtlesim turtle_teleop_key
turtlesim turtlesim_node
```

2.Node runs ros2 run

Command function: Run function package node program

Command format: ros2 run pkg_name node_name

- pkg_name: function package name
- node_name: name of the executable program



3. Node related tools ros2 node

3.1.ros2 node list

Command function: List all node names in the current domain

Command format: ros2 node list

```
yahboom@yahboom-virtual-machine:~$ ros2 node list
/turtlesim
```

3.2. ros2 node info

Command function: View node details, including subscriptions, published messages, enabled services and actions, etc.

Command format: ros2 node info node_name

- node_name： The node name to be viewed

```
yahboom@yahboom-virtual-machine:~$ 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:
```

4. Topic-related tools ros2 topic

4.1.ros2 topic list

Command function: List all topics in the current domain

Command format: ros2 topic list

```
yahboom@yahboom-virtual-machine:~$ ros2 topic list
/parameter_events
/rosout
/turtle1/cmd_vel
/turtle1/color_sensor
/turtle1/pose
```

4.2. ros2 topic info

Command function: Display topic message type, number of subscribers/publishers

Command format: ros2 topic info topic_name

- topic_name：The name of the topic to be queried

```
yahboom@yahboom-virtual-machine:~$ ros2 topic info /turtle1/cmd_vel
Type: geometry_msgs/msg/Twist
Publisher count: 0
Subscription count: 1
```

4.3. ros2 topic type

Command function: View the message type of the topic

Command format: ros2 topic type topic_name

- topic_name：Need to query the name of the topic type

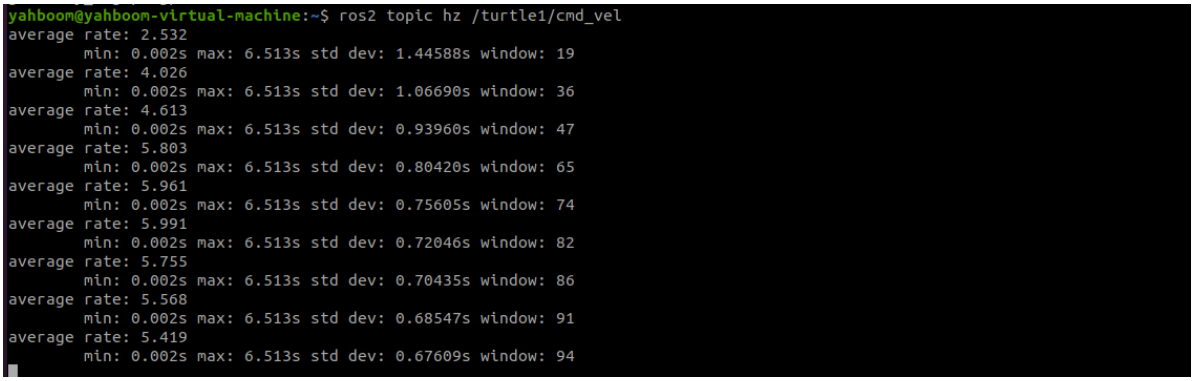
```
yahboom@yahboom-virtual-machine:~$ ros2 topic type /turtle1/cmd_vel
geometry_msgs/msg/Twist
```

4.4.ros2 topic hz

Command function: Display the average publishing frequency of the topic

Command format: ros2 topic hz topic_name

- topic_name：Need to query the name of topic frequency

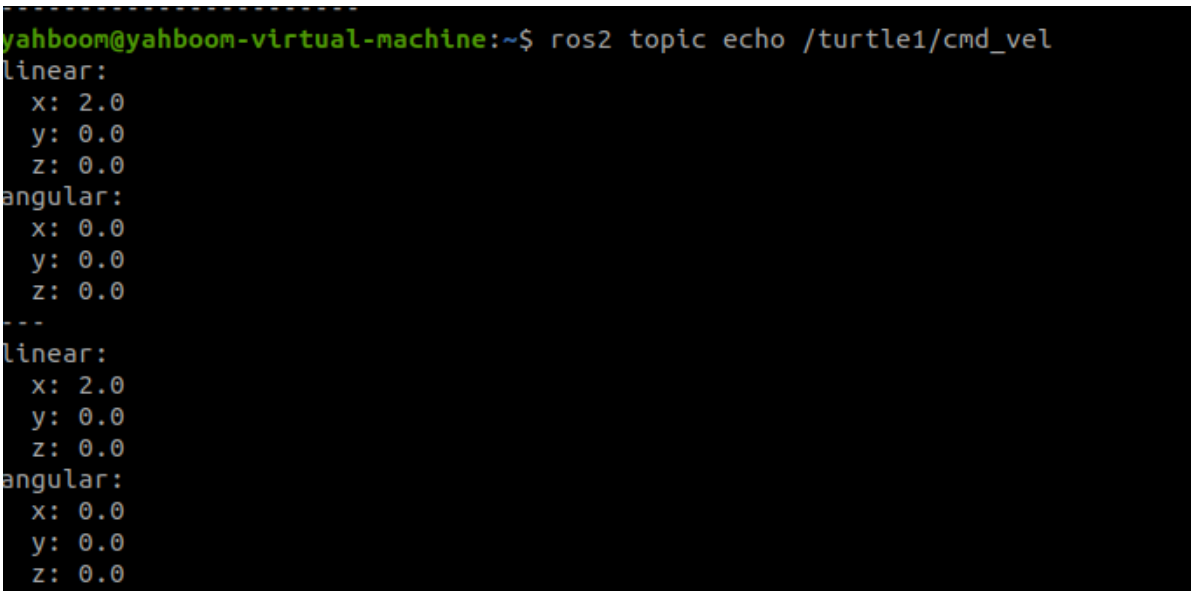


4.5. ros2 topic echo

Command function: print topic messages in the terminal, similar to a subscriber

Command format: ros2 topic echo topic_name

- topic_name：The name of the topic where the message needs to be printed



4.5. ros2 topic pub

Command function: Publish specified topic messages in the terminal

Command format:ros2 topic pub topic_name message_type message_content

- topic_name: the name of the topic where topic messages need to be published
- message_type: the data type of the topic
- message_content: message content

The default is to publish at a frequency of 1Hz. The following parameters can be set,

- Parameter -1 is published only once, ros2 topic pub -1 topic_name message_type message_content

- Parameter -t count loop publishing count times ends, ros2 topic pub -t count topic_name message_type message_content
- Parameter -r count is published cyclically at a frequency of count Hz, ros2 topic pub -r count topic_name message_type message_content

```
ros2 topic pub turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.5, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.2}}"
```

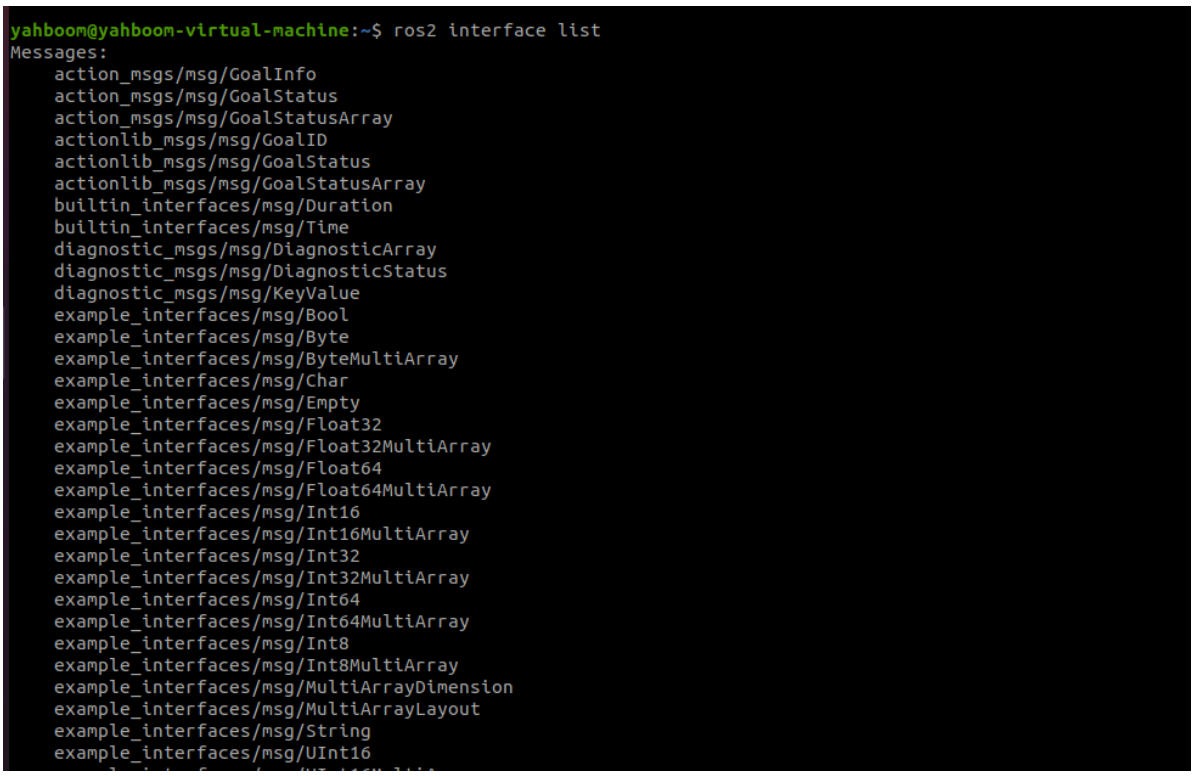
What needs to be noted here is that there is a space after the colon.

5. Interface related tools ros2 interface

5.1. ros2 interface list

Command function: List all interfaces of the current system, including topics, services, and actions.

Command format: ros2 interface list



5.2. ros2 interface show

Command function: display the details of the specified interface

Command format: ros2 interface show interface_name

- interface_name：the name of the interface content that needs to be displayed

```
yahboom@yahboom-virtual-machine:~$ ros2 interface show sensor_msgs/msg/LaserScan
# Single scan from a planar laser range-finder
#
# If you have another ranging device with different behavior (e.g. a sonar
# array), please find or create a different message, since applications
# will make fairly laser-specific assumptions about this data

std_msgs/Header header # timestamp in the header is the acquisition time of
                      # the first ray in the scan.
                      #
                      # in frame frame_id, angles are measured around
                      # the positive Z axis (counterclockwise, if Z is up)
                      # with zero angle being forward along the x axis

float32 angle_min     # start angle of the scan [rad]
float32 angle_max     # end angle of the scan [rad]
float32 angle_increment  # angular distance between measurements [rad]

float32 time_increment   # time between measurements [seconds] - if your scanner
                         # is moving, this will be used in interpolating position
                         # of 3d points
float32 scan_time        # time between scans [seconds]

float32 range_min        # minimum range value [m]
float32 range_max        # maximum range value [m]

float32[] ranges         # range data [m]
                         # (Note: values < range_min or > range_max should be discarded)
float32[] intensities    # intensity data [device-specific units].  If your
                         # device does not provide intensities, please leave
                         # the array empty
```

6. Service related tools ros2 service

6.1. ros2 service list

Command function: List all services in the current domain

Command format：ros2 interface show interface_name

```
yahboom@yahboom-virtual-machine:~$ ros2 service list
/clear
/kill
/reset
/spawn
/teleop_turtle/describe_parameters
/teleop_turtle/get_parameter_types
/teleop_turtle/get_parameters
/teleop_turtle/list_parameters
/teleop_turtle/set_parameters
/teleop_turtle/set_parameters_atomically
/turtle1/set_pen
/turtle1/teleport_absolute
/turtle1/teleport_relative
/turtlesim/describe_parameters
/turtlesim/get_parameter_types
/turtlesim/get_parameters
/turtlesim/list_parameters
/turtlesim/set_parameters
/turtlesim/set_parameters_atomically
yahboom@yahboom-virtual-machine:~$
```

6.2. ros2 service call

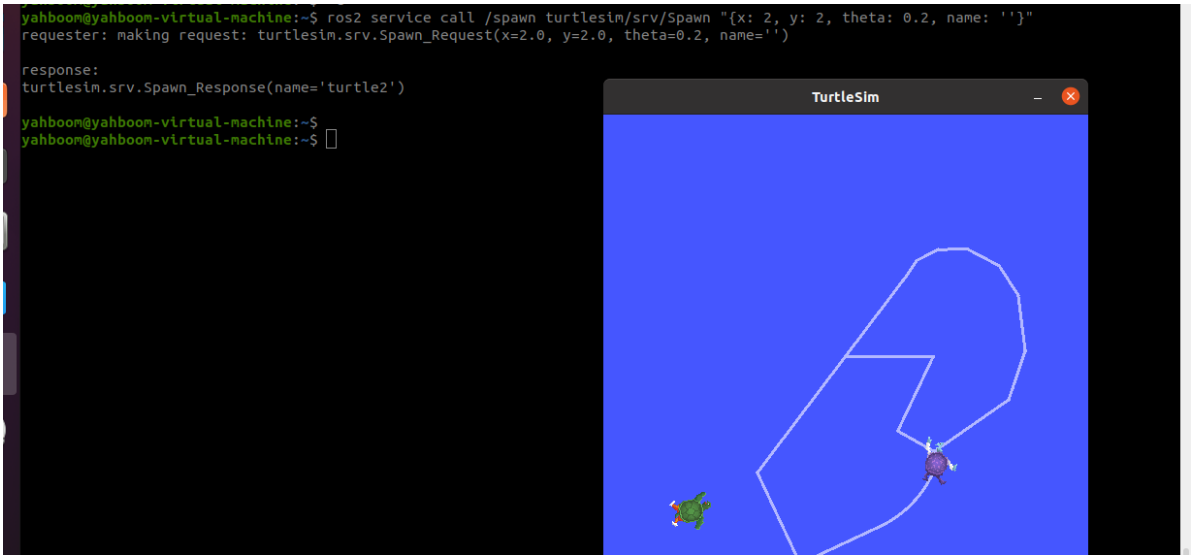
Command function: Call specified service

Command format: ros2 interface call service_name service_Type arguments

- service_name: the service that needs to be called
- service_Type: service data type
- arguments: parameters required to provide the service

For example, calling the spawn turtle service

```
ros2 service call /spawn turtlesim/srv/Spawn "{x: 2, y: 2, theta: 0.2, name:
''}"
requester: making request: turtlesim.srv.Spawn_Request(x=2.0, y=2.0, theta=0.2,
name='turtle2')
```


yahboom@yahboom-virtual-machine:~$ ros2 service call /spawn turtlesim/srv/Spawn "{x: 2, y: 2, theta: 0.2, name: ''}"
requester: making request: turtlesim.srv.Spawn_Request(x=2.0, y=2.0, theta=0.2, name='')
response:
turtlesim.srv.Spawn_Response(name='turtle2')
yahboom@yahboom-virtual-machine:~$
yahboom@yahboom-virtual-machine:~$
TurtleSim