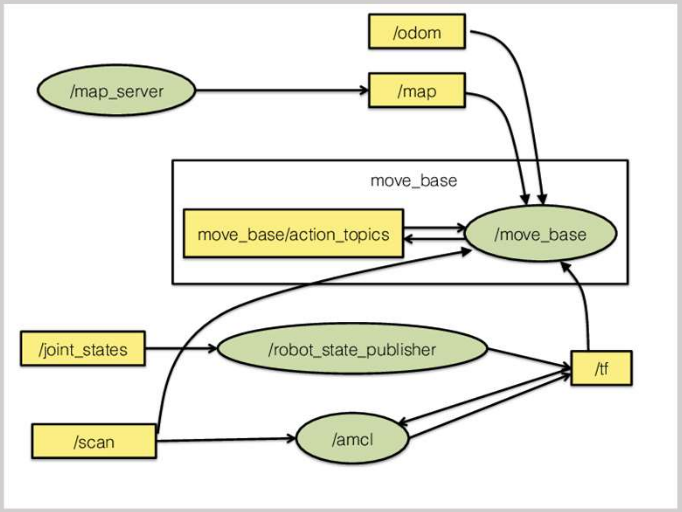
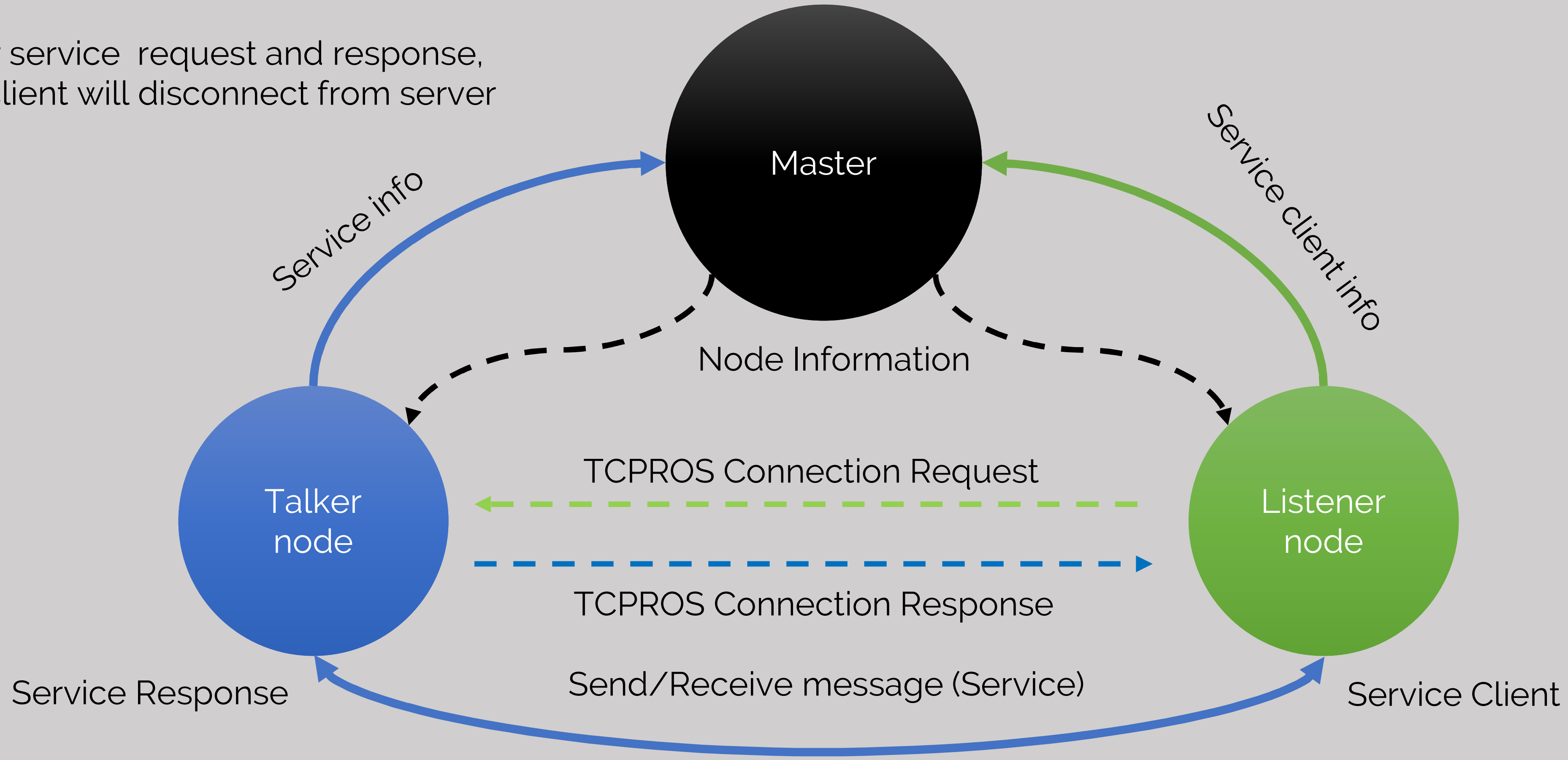


ROS Communication: ROS Services

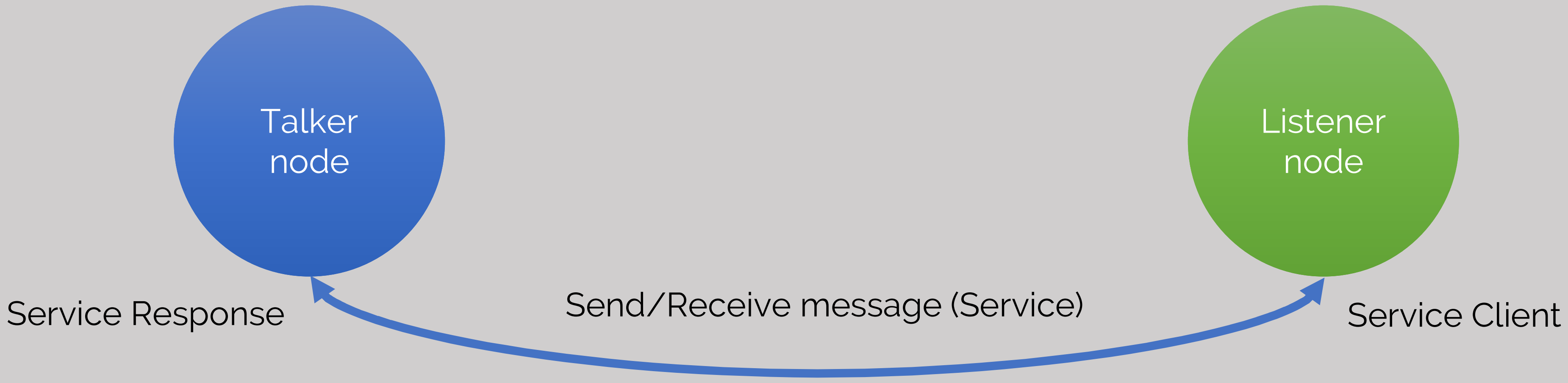
ROS Communication: Services

After service request and response,
the client will disconnect from server



ROS Communication: Services

After service request and response,
the client will disconnect from server



We are Canonical

It is our mission to make open source software available to people everywhere. We believe the best way to fuel innovation is to give the innovators the technology they need.

What is ROS Build System?

Discussing the basic concepts of ROS build system

ROS Build System

- **ROS Build System:**
  - A build system generate target executable/library from the source code
  - In ROS, we are using ***catkin*** (http://wiki.ros.org/catkin/conceptual_overview)
  - Catkin = CMake Macros (http://www.cmake.org) + Python script
  - The source code in ROS is organized as '*ROS Packages*'.
  - The build system in ROS is used build complex code organization and dependencies
  - The catkin commands used to create ROS packages and workspace

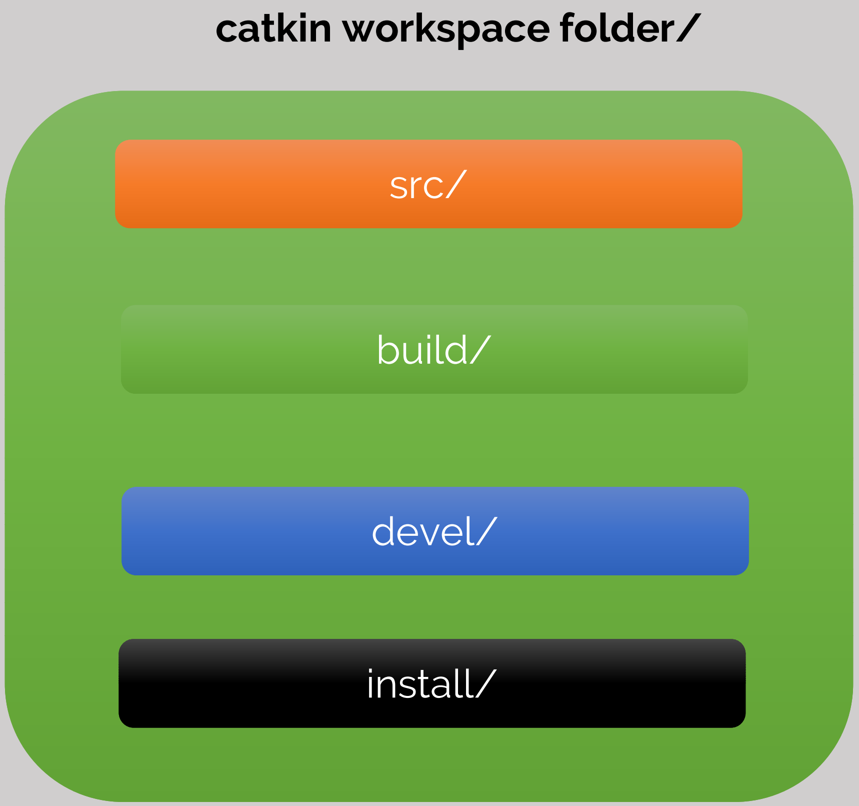
What is ROS Catkin Workspace and ROS Packages?

Discussing the basic concepts of ROS catkin workspace and packages

ROS Catkin workspace and packages

- **ROS Catkin Workspace and packages:**
  - It is the folder where we can modify, build and install **catkin ROS packages**
  - The catkin ROS packages are unit of organizing software in ROS.
  - A ROS package can have ROS nodes, ROS library, configuration files etc.
  - We can create any number of catkin workspace in your PC
  - http://wiki.ros.org/catkin/workspaces

ROS Catkin workspace and packages



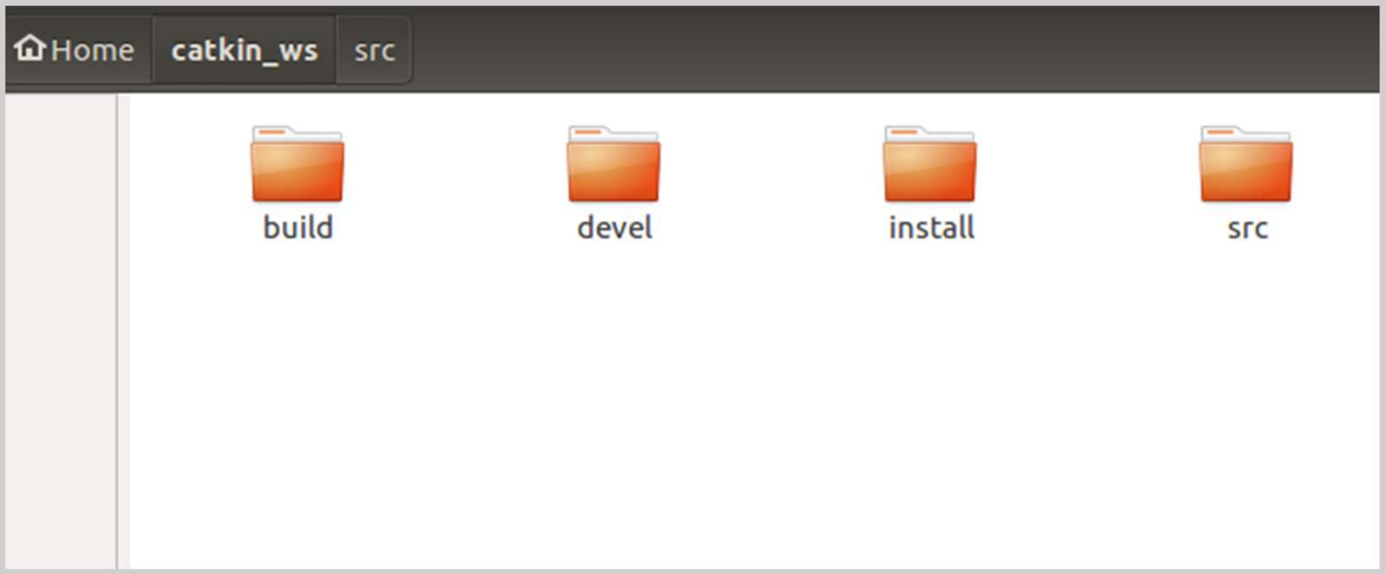
src: The ROS packages are keeping inside src folder.

build: CMake and catkin keep their cache information and other intermediate files here.

devel: Build targets (executable) are stored here

install: The build targets can be installed into this space

ROS Catkin workspace and packages

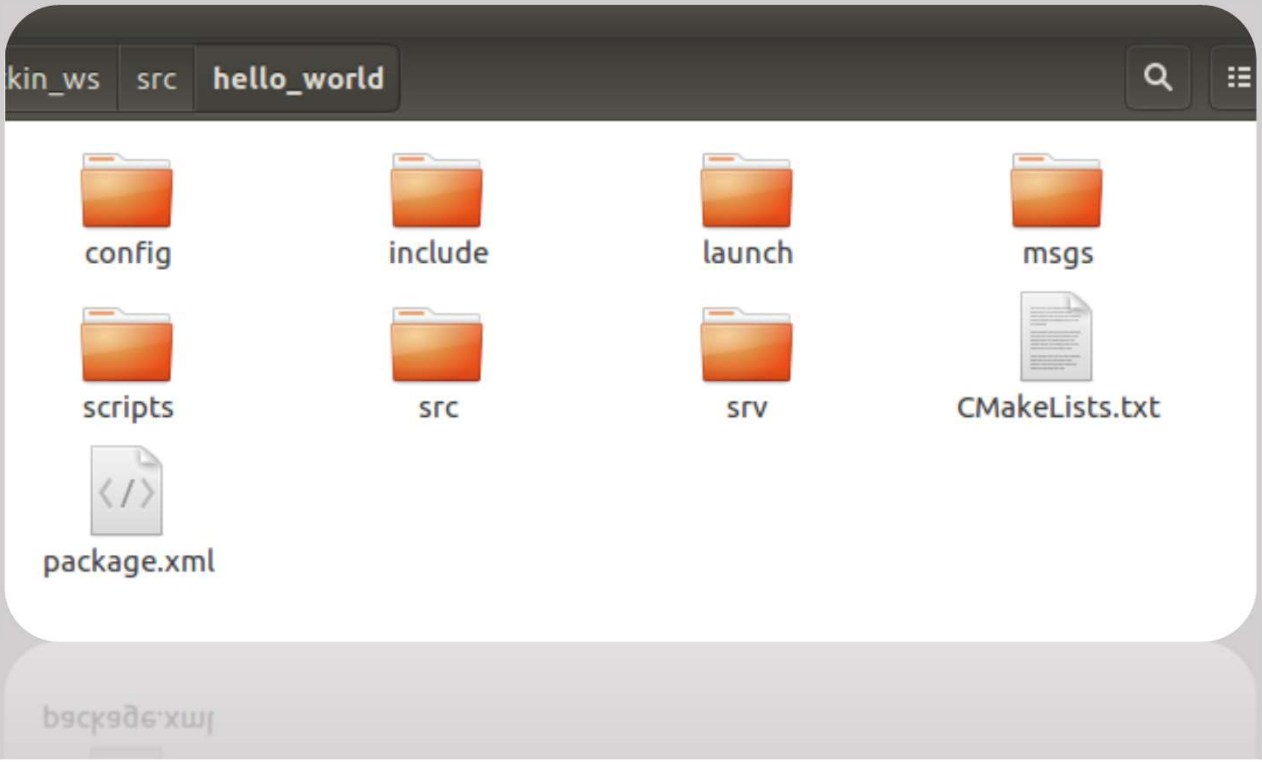


ROS Catkin workspace and packages

- **ROS Package:**
  - The Software in ROS is organized in *packages.*
  - A ROS package may have ROS nodes, library, configuration file etc.
  - The package provides modularity & reusability
  - All ROS packages are keeping inside ROS catkin workspace
  - We can build and customize ROS packages inside a workspace
  - http://wiki.ros.org/Packages

ROS Catkin workspace and packages

- **A typical ROS Package:**



ROS Catkin workspace and packages

- **A typical ROS Package:**
  - **include/package_name**: C++ include headers
  - **msg/**: Folder containing Message (msg) types for ROS topic communication
  - **src/package_name/**: Mainly C++ Source files
  - **srv/**: Folder containing ROS Service (srv) types

ROS Catkin workspace and packages

- **A typical ROS Package:**
  - **scripts/**: Mainly executable python scripts
  - **CMakeLists.txt**: CMake build file.
  - **package.xml**: Package catkin/package.xml. This file contain complete information of the package and its dependencies.

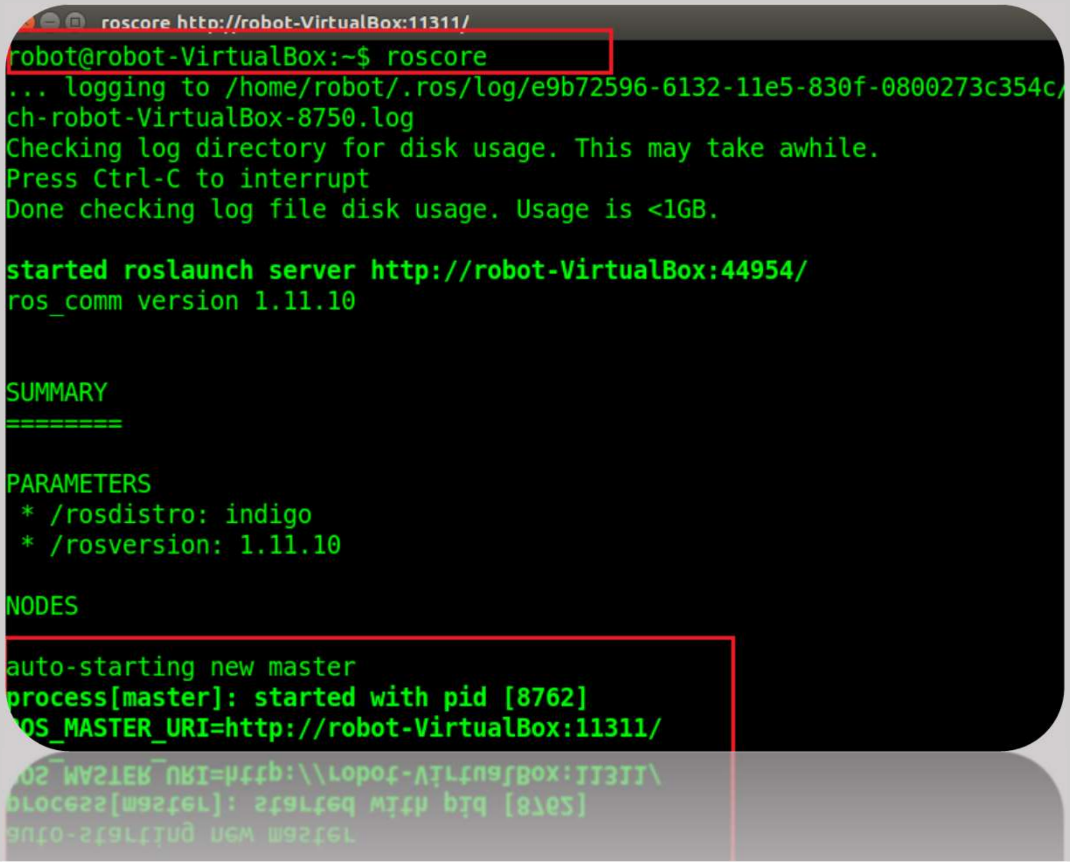
ROS Command tools

Introduction to ROS Command line tools

Command : roscore

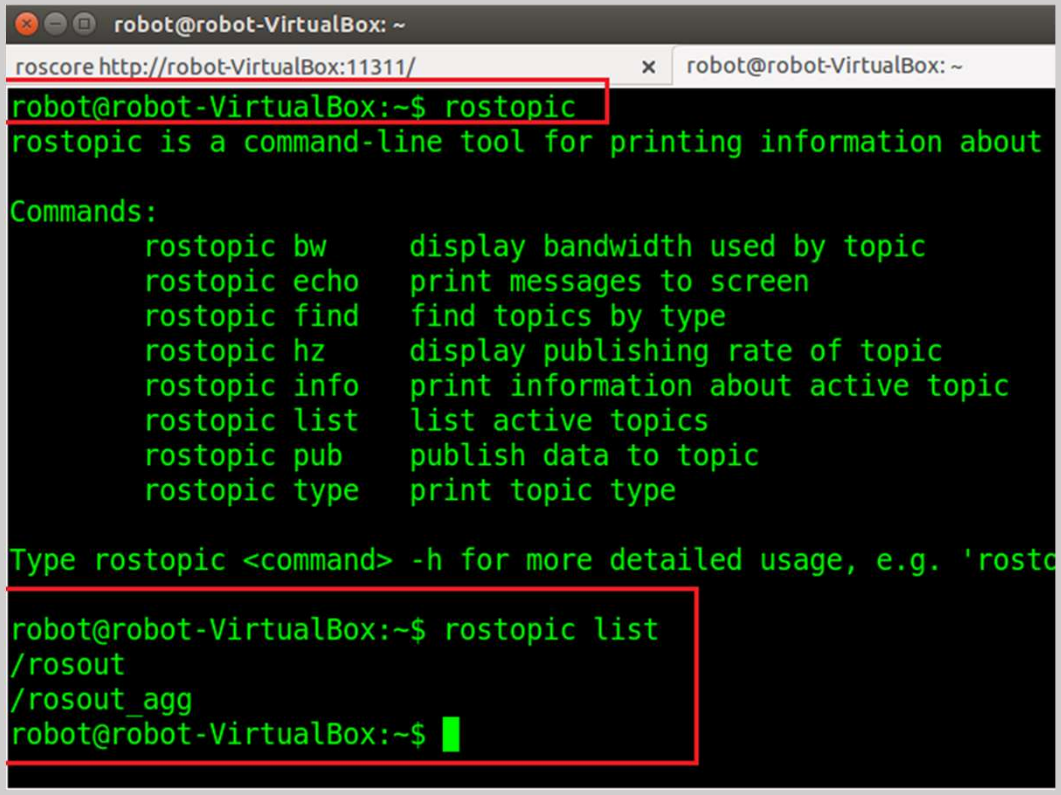
- roscore is a collection of nodes and programs that are pre-requisites of a ROS-based system.
- You **must** have a **roscore** running in order for ROS nodes to communicate. It is launched using the **roscore** command.
- **roscore = rosmaster + parameter server + rosout logging node**

Try roscore in your terminal



```
roscore http://robot-VirtualBox:11311/
robot@robot-VirtualBox:~$ roscore
... logging to /home/robot/.ros/log/e9b72596-6132-11e5-830f-0800273c354c/
ch-robot-VirtualBox-8750.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://robot-VirtualBox:44954/
ros_comm version 1.11.10


SUMMARY
========

PARAMETERS
 * /rosdistro: indigo
 * /rosversion: 1.11.10

NODES

auto-starting new master
process[master]: started with pid [8762]
ROS_MASTER_URI=http://robot-VirtualBox:11311/
```

Command : rostopic

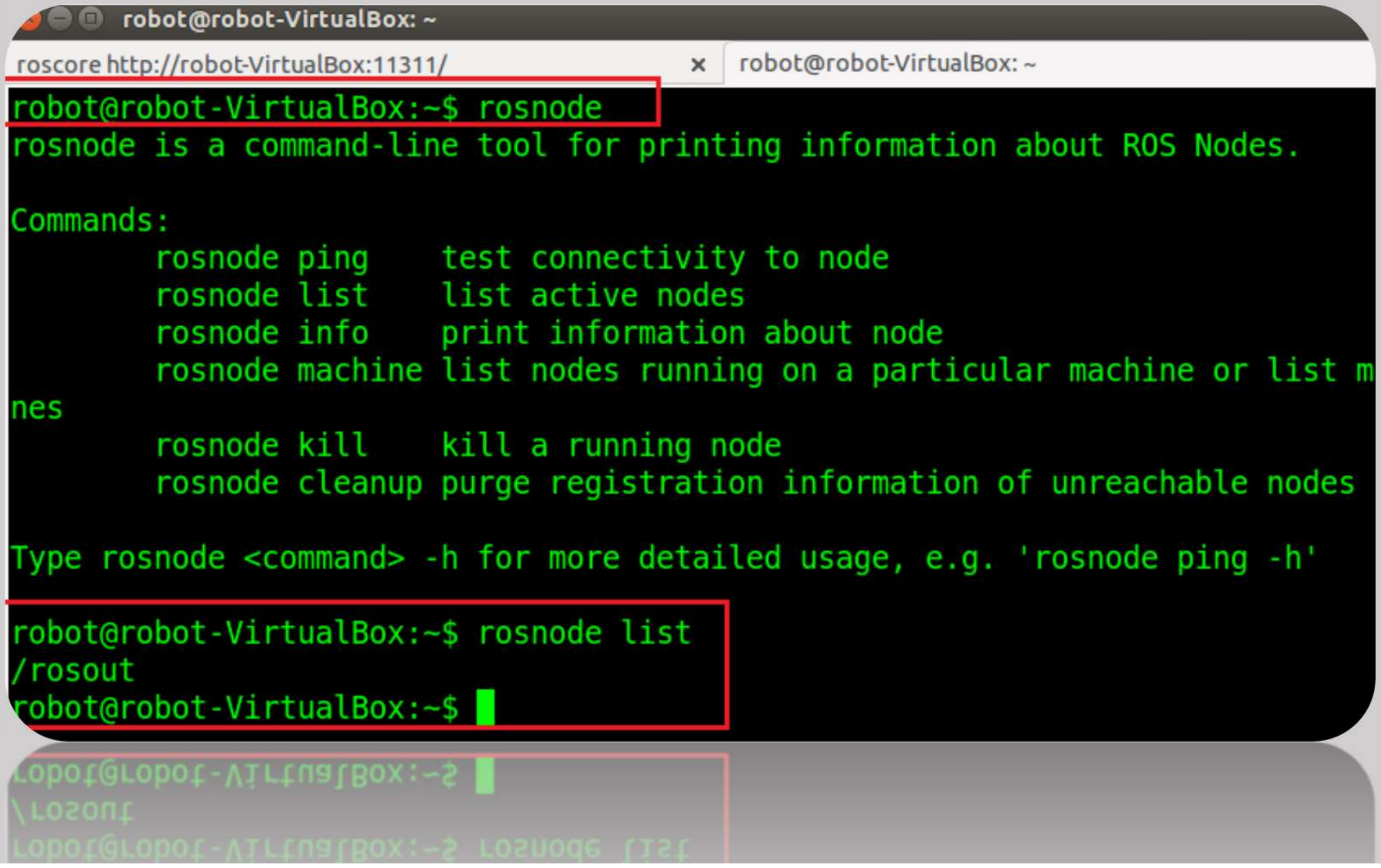


robot@robot-VirtualBox: ~

roscore http://robot-VirtualBox:11311/ × robot@robot-VirtualBox: ~

```
robot@robot-VirtualBox:~$ rostopic
rostopic is a command-line tool for printing information about

Commands:
        rostopic bw     display bandwidth used by topic
        rostopic echo   print messages to screen
        rostopic find   find topics by type
        rostopic hz     display publishing rate of topic
        rostopic info   print information about active topic
        rostopic list   list active topics
        rostopic pub    publish data to topic
        rostopic type   print topic type

Type rostopic <command> -h for more detailed usage, e.g. 'rostc

robot@robot-VirtualBox:~$ rostopic list
/rosout
/rosout_agg
robot@robot-VirtualBox:~$
```

Command : rosnode

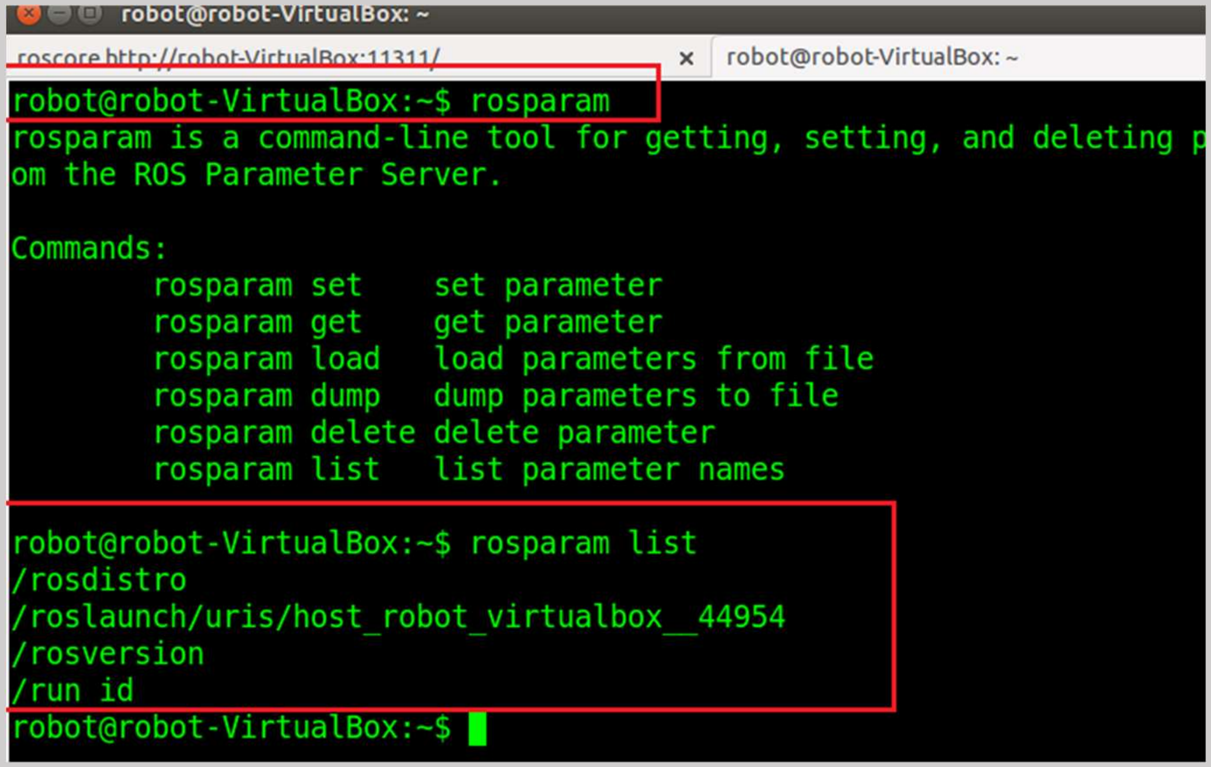


robot@robot-VirtualBox: ~

roscore http://robot-VirtualBox:11311/ | robot@robot-VirtualBox: ~

```
robot@robot-VirtualBox:~$ rosnode
rosnode is a command-line tool for printing information about ROS Nodes.

Commands:
        rosnode ping    test connectivity to node
        rosnode list    list active nodes
        rosnode info    print information about node
        rosnode machine list nodes running on a particular machine or list m
nes
        rosnode kill    kill a running node
        rosnode cleanup purge registration information of unreachable nodes

Type rosnode <command> -h for more detailed usage, e.g. 'rosnode ping -h'

robot@robot-VirtualBox:~$ rosnode list
/rosout
robot@robot-VirtualBox:~$ 
```

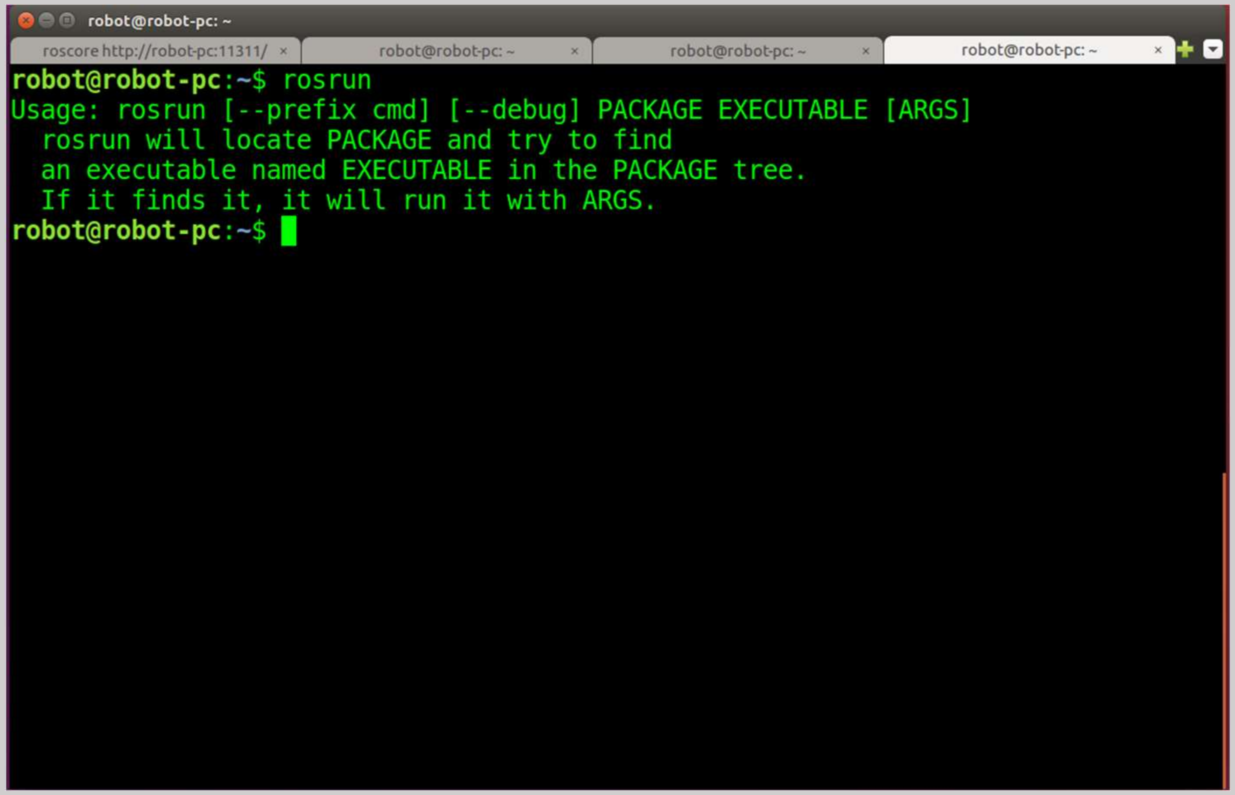
Command : rosparam


robot@robot-VirtualBox: ~

roscore http://robot-VirtualBox:11311/ × robot@robot-VirtualBox: ~

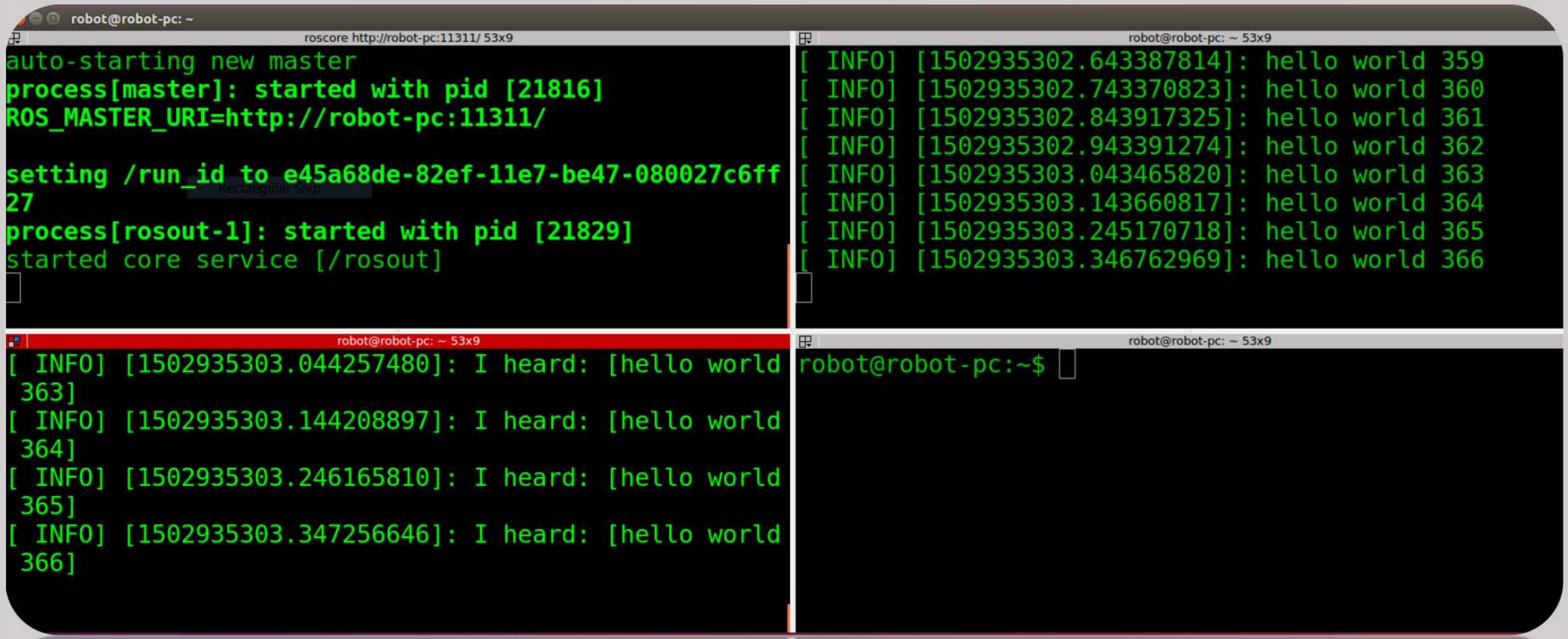
```
robot@robot-VirtualBox:~$ rosparam
rosparam is a command-line tool for getting, setting, and deleting p
om the ROS Parameter Server.

Commands:
        rosparam set    set parameter
        rosparam get    get parameter
        rosparam load   load parameters from file
        rosparam dump   dump parameters to file
        rosparam delete delete parameter
        rosparam list   list parameter names

robot@robot-VirtualBox:~$ rosparam list
/rosdistro
/roslaunch/uris/host_robot_virtualbox__44954
/rosversion
/run_id
robot@robot-VirtualBox:~$ 
```

Command : rosrun



```
robot@robot-pc:~$ rosrun
Usage: rosrun [--prefix cmd] [--debug] PACKAGE EXECUTABLE [ARGS]
  rosrun will locate PACKAGE and try to find
  an executable named EXECUTABLE in the PACKAGE tree.
  If it finds it, it will run it with ARGS.
robot@robot-pc:~$ 
```

roscpp tutorials: talker & listener



```
auto-starting new master
process[master]: started with pid [21816]
ROS_MASTER_URI=http://robot-pc:11311/

setting /run_id to e45a68de-82ef-11e7-be47-080027c6ff
27
process[rosout-1]: started with pid [21829]
started core service [/rosout]
```

```
[ INFO] [1502935302.643387814]: hello world 359
[ INFO] [1502935302.743370823]: hello world 360
[ INFO] [1502935302.843917325]: hello world 361
[ INFO] [1502935302.943391274]: hello world 362
[ INFO] [1502935303.043465820]: hello world 363
[ INFO] [1502935303.143660817]: hello world 364
[ INFO] [1502935303.245170718]: hello world 365
[ INFO] [1502935303.346762969]: hello world 366
```

```
[ INFO] [1502935303.044257480]: I heard: [hello world
 363]
[ INFO] [1502935303.144208897]: I heard: [hello world
 364]
[ INFO] [1502935303.246165810]: I heard: [hello world
 365]
[ INFO] [1502935303.347256646]: I heard: [hello world
 366]
```

```
robot@robot-pc:~$
```

roscpp tutorials: talker & listener [Code explanation]

- $ **roscore**

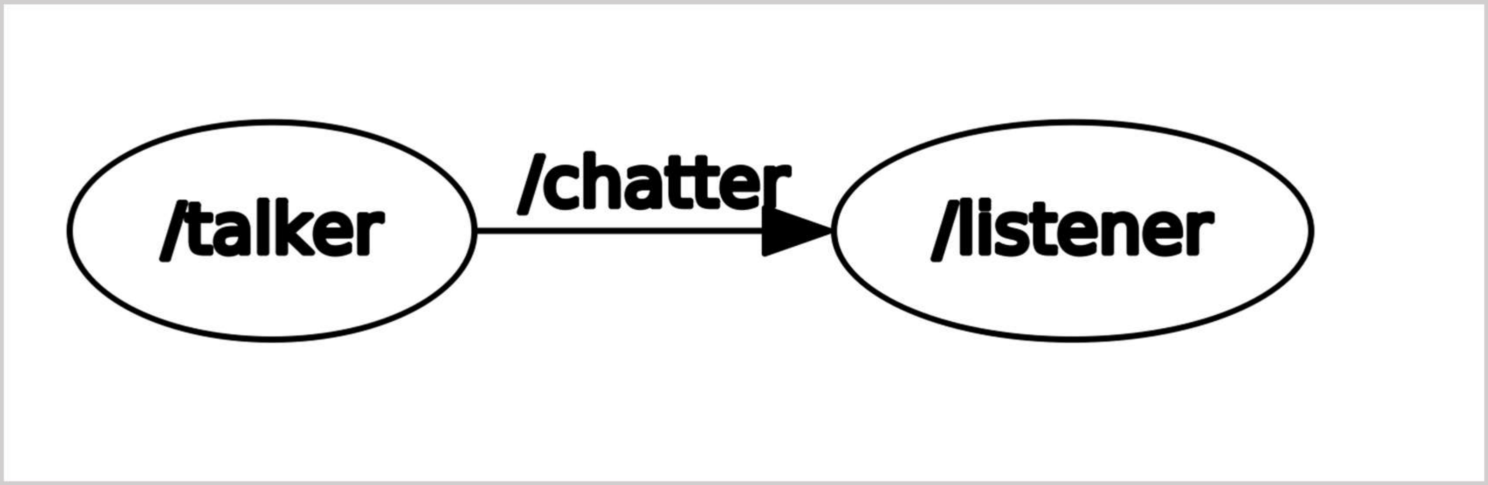
[Publisher]

- $ **rosrun roscpp_tutorials talker**

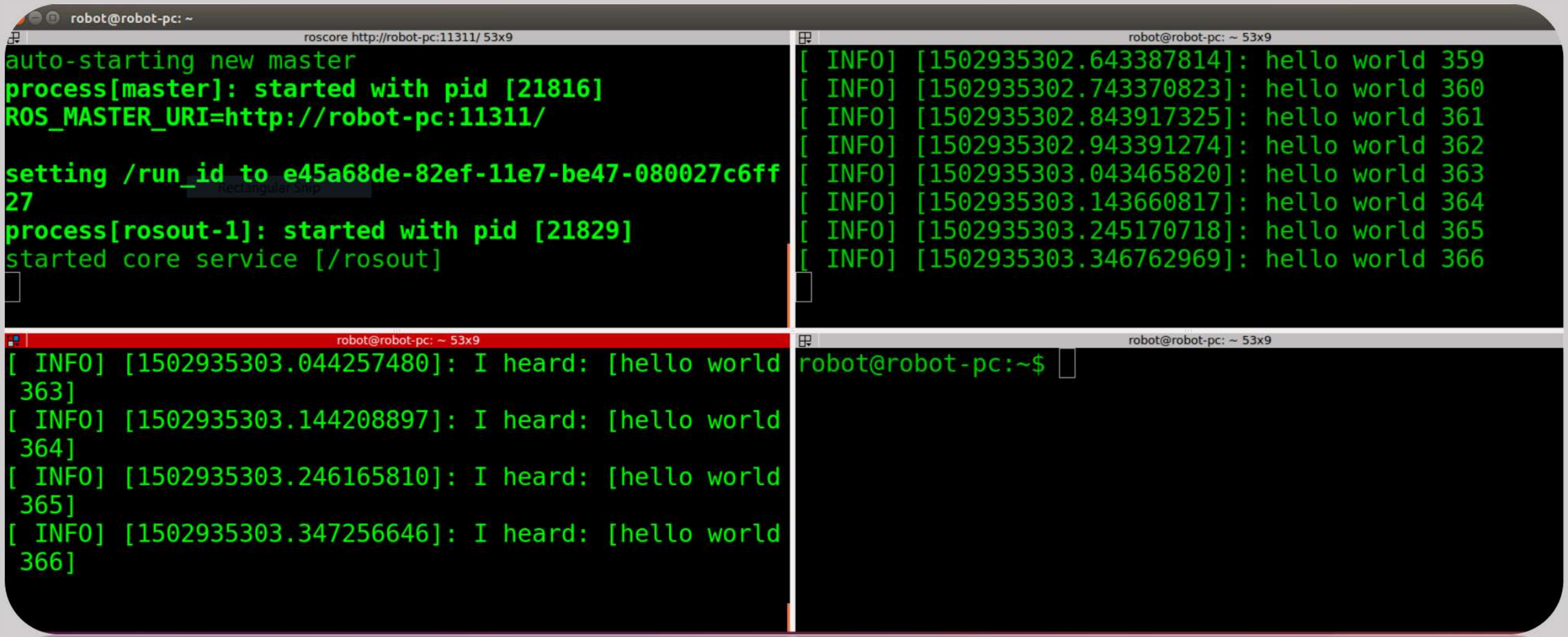
[Subscriber]

- $ **rosrun roscpp_tutorials listener**

roscpp tutorials: talker & listener



Playing with ROS command tools



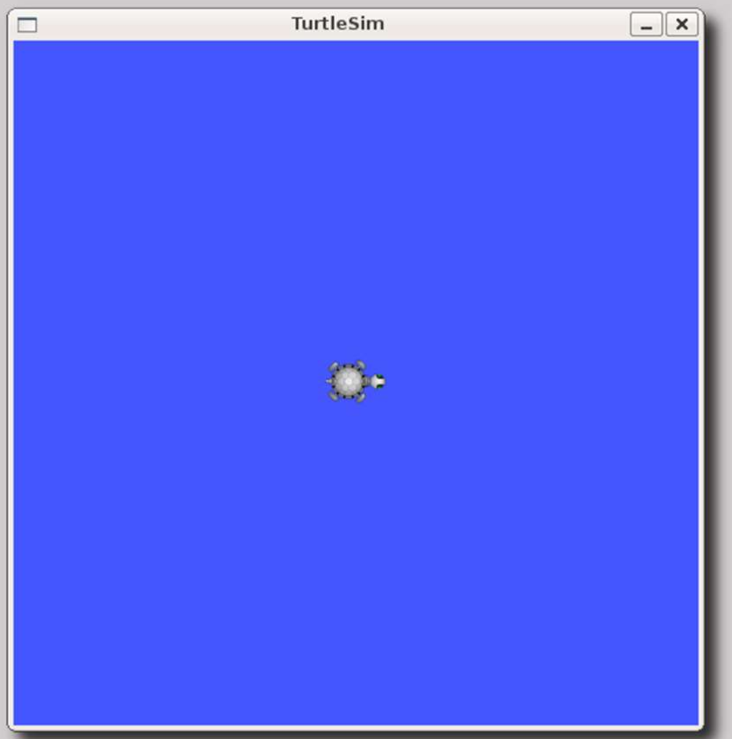
```
auto-starting new master
process[master]: started with pid [21816]
ROS_MASTER_URI=http://robot-pc:11311/

setting /run_id to e45a68de-82ef-11e7-be47-080027c6ff
27
process[rosout-1]: started with pid [21829]
started core service [/rosout]
```

```
[ INFO] [1502935302.643387814]: hello world 359
[ INFO] [1502935302.743370823]: hello world 360
[ INFO] [1502935302.843917325]: hello world 361
[ INFO] [1502935302.943391274]: hello world 362
[ INFO] [1502935303.043465820]: hello world 363
[ INFO] [1502935303.143660817]: hello world 364
[ INFO] [1502935303.245170718]: hello world 365
[ INFO] [1502935303.346762969]: hello world 366
```

```
[ INFO] [1502935303.044257480]: I heard: [hello world
 363]
[ INFO] [1502935303.144208897]: I heard: [hello world
 364]
[ INFO] [1502935303.246165810]: I heard: [hello world
 365]
[ INFO] [1502935303.347256646]: I heard: [hello world
 366]
```

```
robot@robot-pc:~$
```

Turtlesim Demo

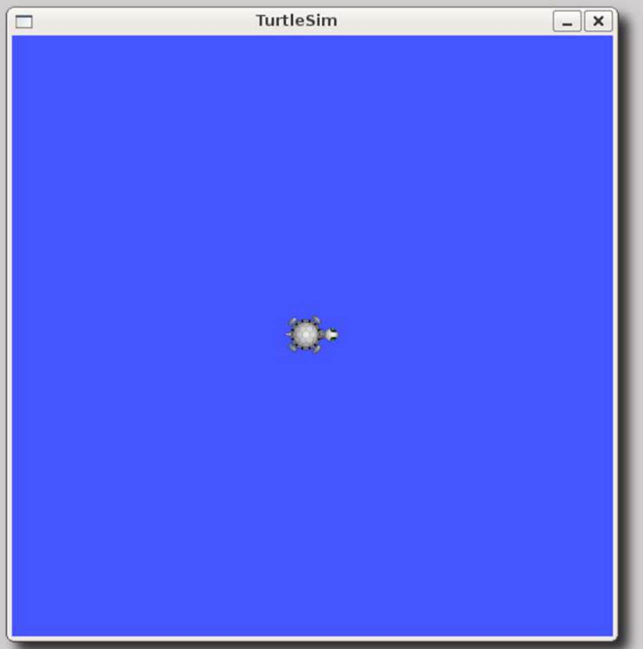


Do you remember Logo Programming?

```
>
```

Learn ROS using Turtlesim

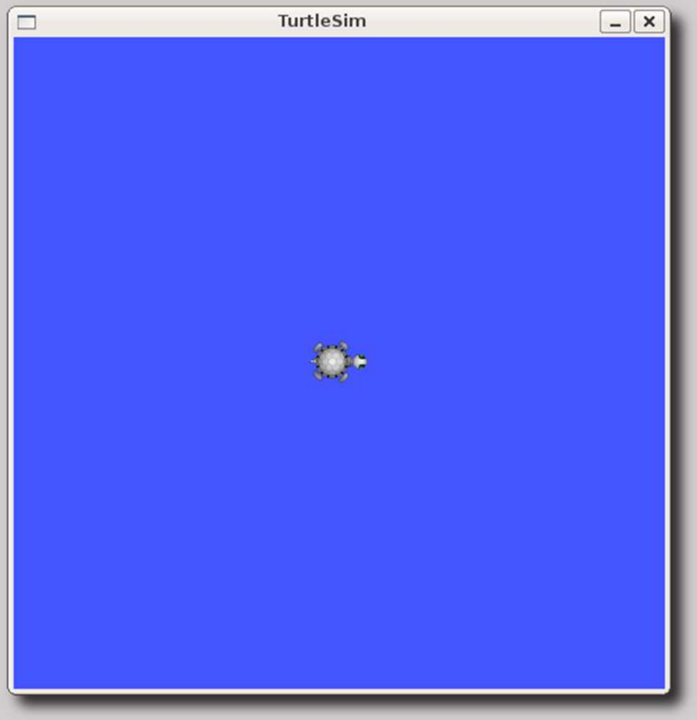
- http://wiki.ros.org/turtlesim



Launching turtlesim

- $ **roscore** [In first terminal]
- **$ rosrun turtlesim turtlesim_node** [In second terminal]

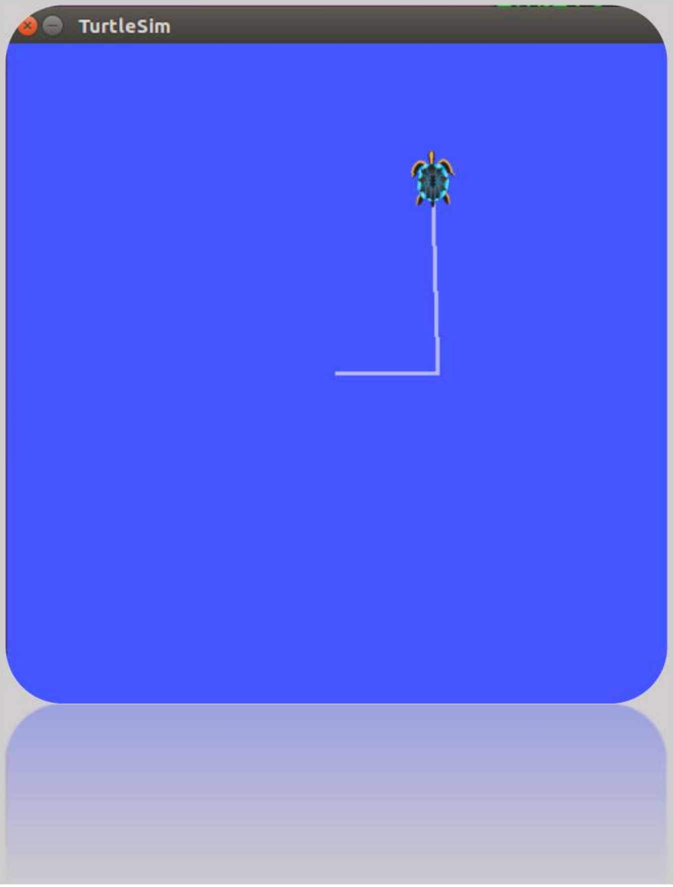
Launching turtlesim



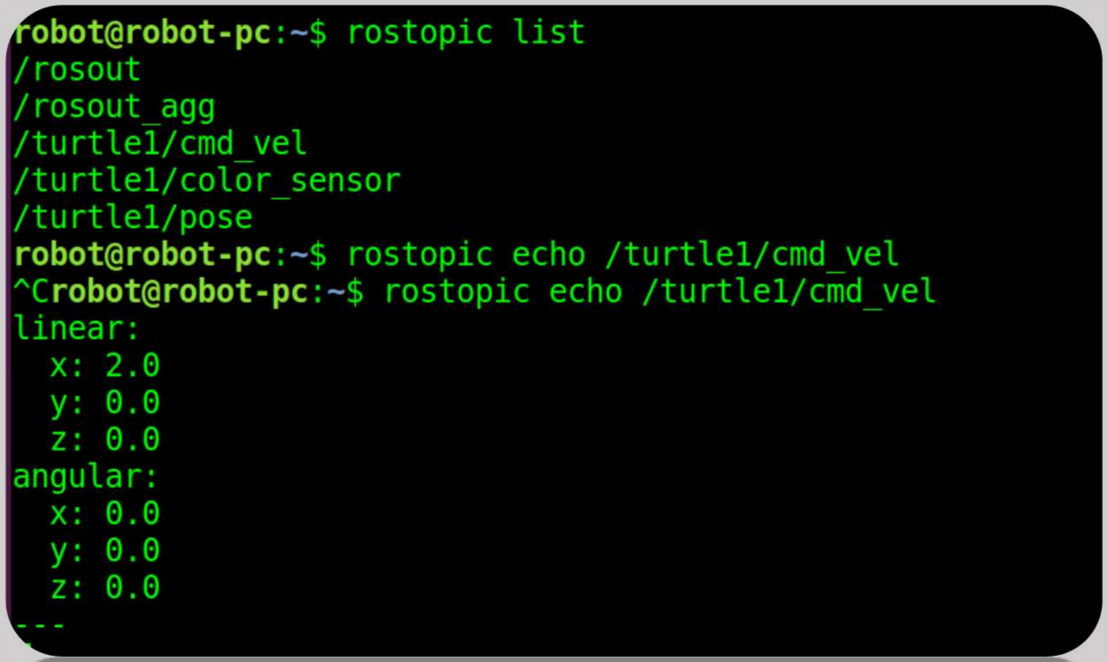
Moving turtle

- $ rosrun turtlesim turtle_teleop_key

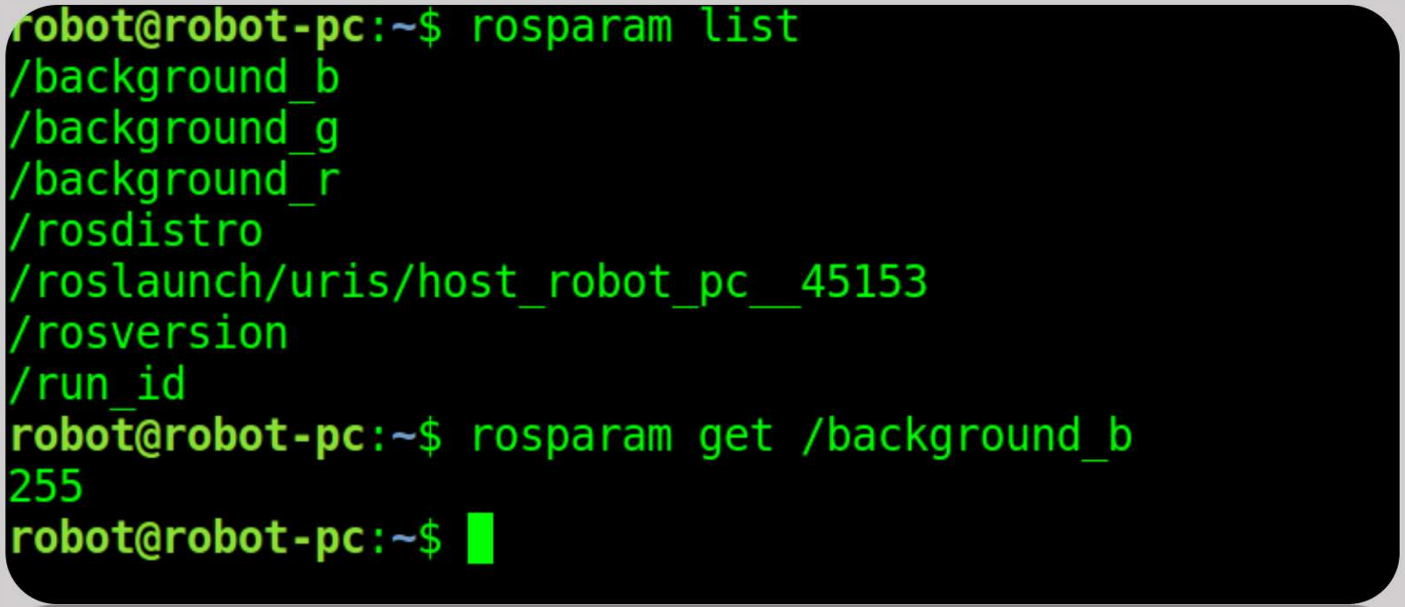
Moving turtle



Inspecting ROS Topic

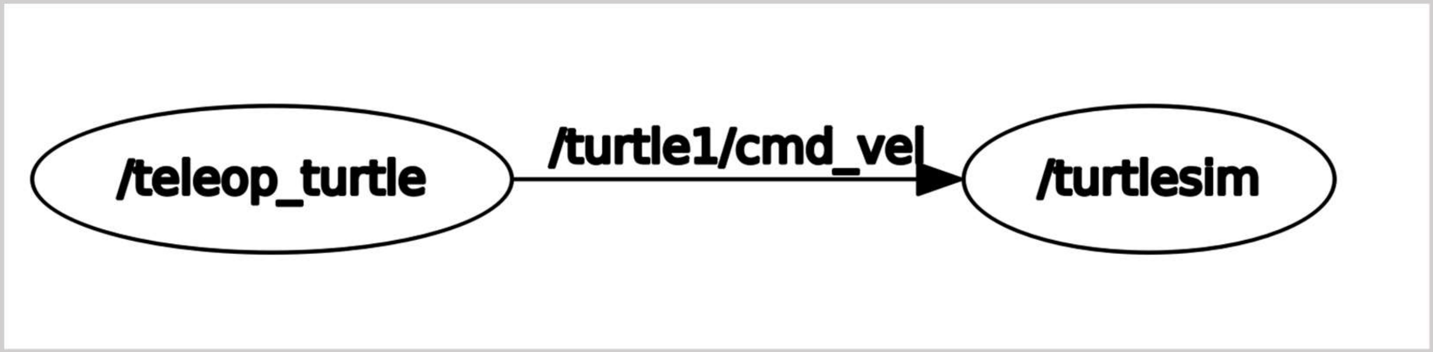

```
robot@robot-pc:~$ rostopic list
/rosout
/rosout_agg
/turtle1/cmd_vel
/turtle1/color_sensor
/turtle1/pose
robot@robot-pc:~$ rostopic echo /turtle1/cmd_vel
^Crobot@robot-pc:~$ rostopic echo /turtle1/cmd_vel
linear:
  x: 2.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.0
---
```

Inspecting ROS Parameter



```
robot@robot-pc:~$ rosparam list
/background_b
/background_g
/background_r
/rosdistro
/roslaunch/uris/host_robot_pc__45153
/rosversion
/run_id
robot@robot-pc:~$ rosparam get /background_b
255
robot@robot-pc:~$ 
```

Communication graph



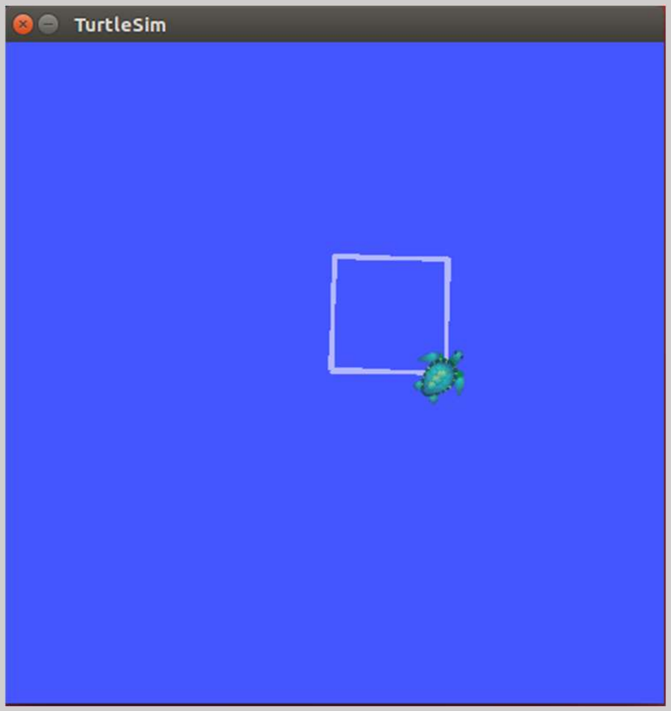
Turtlesim Demo: Draw Square

- $ **roscore**   [In first terminal]

- $ **rosrun turtlesim turtlesim_node** [In second terminal]

- **rosrun turtlesim draw_square**

Turtlesim Demo: Draw Square



Setting your ROS IDE

- http://wiki.ros.org/IDEs

VSCode

KDevelop

Setting your ROS IDE

- VSCode
  - c_cpp_properties.json

```
 "name": "ROS",
         "intelliSenseMode": "clang-x64",
         "compilerPath": "/usr/bin/gcc",
         "cStandard": "c11",
         "cppStandard": "c++14“
```

  - For python intellisense

```
sudo apt install python3-pip
sudo –H pip3 install pylint
```