Turtlesim TurtleBot   remap 09/29/18

Terminal 1

```
harman@D104-45931:~$ roscore
```

Terminal 2

```
harman@D104-45931:~$ rosrun turtlesim turtlesim_node
```

Terminal 3

```
harman@D104-45931:~$ cd Desktop/Turtlebot_Python_Scripts/
harman@D104-45931:~/Desktop$ ls | grep py
        go_in_circleTurtlesim.py
```

Run Turtlesim in a circle:

```
harman@D104-45931:~/Desktop$ python go_in_circleTurtlesim.py
        [INFO] [1517603612.336740]:  Press CTRL+c to stop Turtlesim
        [INFO] [1517603612.337787]:  Set rate 10Hz
        ^C[INFO] [1517603625.581991]: Stopping Turtlesim
```

go_in_circleTurtlesim.py   move_cmd.linear.x = 0.5 m/s   move_cmd.angular.z = 0.5 r/s

```
#!/usr/bin/env python   go_in_circleTurtlesim.py
# Execute as a python script
# Set linear and angular values of Turtlesim's speed and turning.
import rospy                                # Needed to create a ROS node
from geometry_msgs.msg import Twist    # Message that moves base

class ControlTurtlesim():
   def __init__(self):
      # ControlTurtlesim is the name of the node sent to the master
      rospy.init_node('ControlTurtlesim', anonymous=False)

                  # Message to screen
      rospy.loginfo(" Press CTRL+c to stop Turtlesim")

      # Keys CNTL + c will stop script
      rospy.on_shutdown(self.shutdown)

         # Publisher will send Twist message on topic
      # /turtle1/cmd_vel   This is for Turtlesim only

      self.cmd_vel = rospy.Publisher('/turtle1/cmd_vel', Twist, queue_size=10)

            # Turtlesim will receive the message 10 times per second.
```

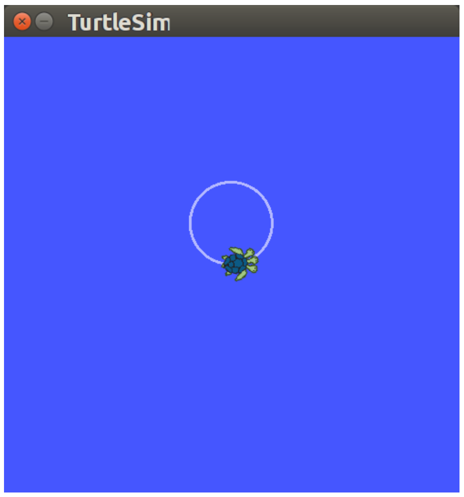
```
        rate = rospy.Rate(10);
              # 10 Hz is fine as long as the processing does not exceed
        #   1/10 second.
        rospy.loginfo(" Set rate 10Hz")
        # Twist is geometry_msgs for linear and angular velocity
        move_cmd = Twist()
              # Linear speed in x in meters/second is + (forward) or
        # - (backwards)
        move_cmd.linear.x = 0.5       # Modify this value to change speed
              # Turn at 0.5 radians/s
        move_cmd.angular.z = 0.5
        # Modify this value to cause rotation rad/s

              # Loop and Turtlesim will move until you type CNTL+c
        while not rospy.is_shutdown():
                # publish Twist values to the Turtlesim node /cmd_vel
            self.cmd_vel.publish(move_cmd)
                # wait for 0.1 seconds (10 HZ) and publish again
            rate.sleep()


    def shutdown(self):
        # You can stop turtlesim by publishing an empty Twist message
        rospy.loginfo("Stopping Turtlesim")

        self.cmd_vel.publish(Twist())
              # Give Turtlesim time to stop
        rospy.sleep(1)

if __name__ == '__main__':
    try:
        ControlTurtlesim()
    except:
        rospy.loginfo("End of the trip for Turtlesim")
```

Ok to 

Kill Poor Turtle Now

Let’s try TurtleBot   Remap /turtle1/cmd_vel To cmd_vel_mux/input/navi   (Book Page103)

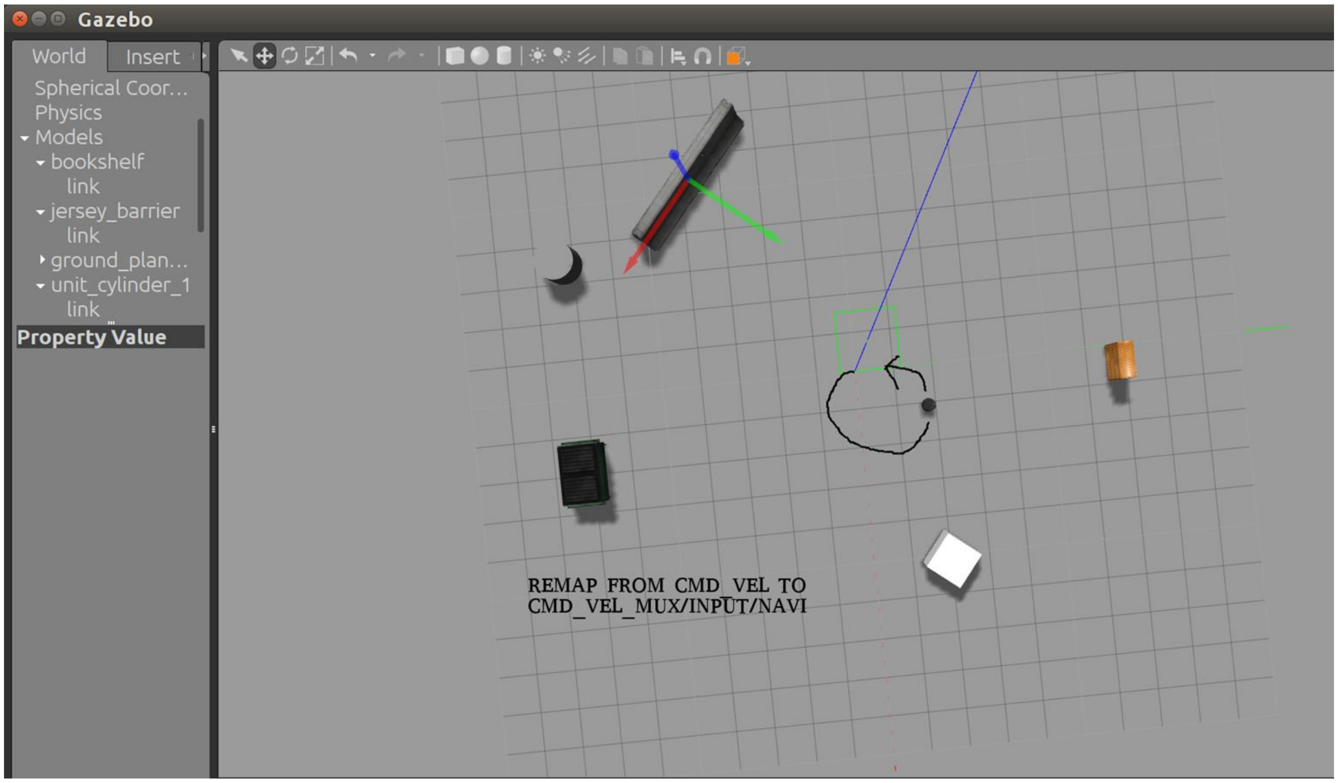
Terminal 1
harman@D104-45931:~/Desktop$ **roslaunch turtlebot_gazebo turtlebot_world.launch**

Terminal 2
harman@D104-45931:~$ cd Desktop/Turtlebot_Python_Scripts/
harman@D104-45931:~/Desktop$ **ls | grep py**

```
        go_in_circleTurtlesim.py
```

Use Turtlesim Code Remapped
harman@D104-45931:~/Desktop/Turtlebot_Python_Scripts$ python go_in_circleTurtlesim.py /turtle1/cmd_vel:=cmd_vel_mux/input/navi

```
        [INFO] [1538268848.480240, 0.000000]:  Press CTRL+c to stop Turtlesim
        [INFO] [1538268848.482112, 0.000000]:  Set rate 10Hz
        ^C[INFO] [1538268860.883950, 178.170000]: Stopping Turtlesim
        [INFO] [1538268862.013880, 179.170000]: End of the trip for Turtlesim
```


Gazebo
World
Insert
Spherical Coor...
Physics
Models
bookshelf
link
jersey_barrier
link
ground_plan...
unit_cylinder_1
link
Property Value
REMAP FROM CMD_VEL TO
CMD_VEL_MUX/INPUT/NAVI