Steps of create script python for move the robot TurleBot3

1. Creating a ROS Package

```
# You should have created this in the Creating a Workspace Tutorial
$ cd ~/catkin_ws/src
$ catkin_create_pkg navigation std_msgs rospy roscpp
$ cd ~/catkin_ws
$ catkin_make
$ . ~/catkin_ws/devel/setup.bash
```

2. Write python script
    a. Import the library

```
#/!usr/bin/env python
import rospy
import string
import math
import time
import sys
from std_msgs.msg import String
from move_base_msgs.msg import MoveBaseActionResult
from actionlib_msgs.msg import GoalStatusArray
from geometry_msgs.msg import PoseStamped
```

    b. Define the class

```
class Goal:
    def __init__(self, goalListX, goalListY, retry, map_frame):
        self.sub = rospy.Subscriber('move_base/result', MoveBaseActio
nResult, self.statusCB, queue_size=10)
        self.pub = rospy.Publisher('move_base_simple/goal', PoseStamp
ed, queue_size=10)
 #        params & variables
        self.goalX = goalX
        self.goalY = goalY
        self.retry = retry
        self.goalMsg = PoseStamped()
        self.goalMsg.header.frame_id = map_frame
        self.goalMsg.pose.orientation.z = 0.0
        self.goalMsg.pose.orientation.w = 1.0
 #        Publish the first goal
```

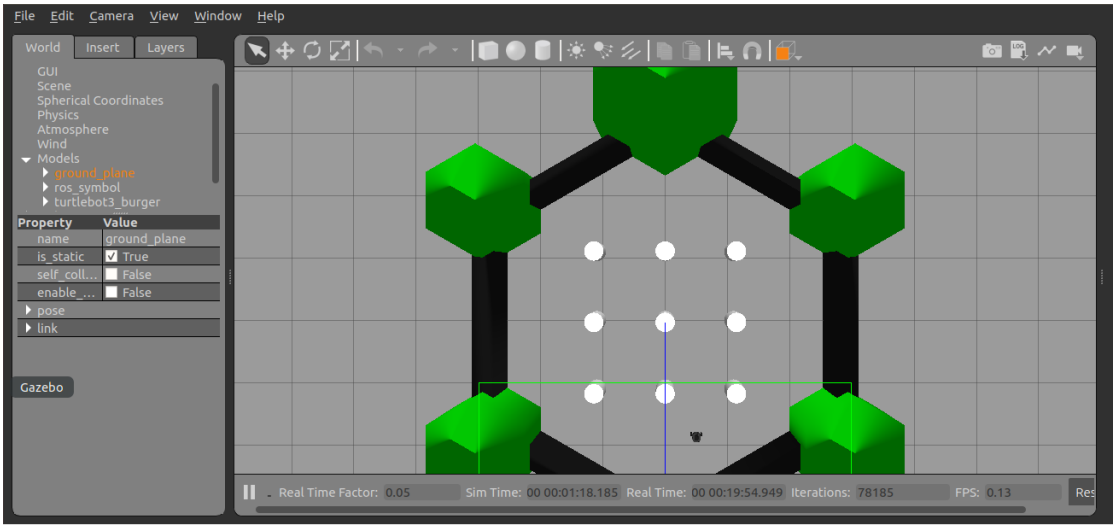
```
        time.sleep(1)
        self.goalMsg.header.stamp = rospy.Time.now()
        self.goalMsg.pose.position.x = self.goalX
        self.goalMsg.pose.position.y = self.goalY
        self.pub.publish(self.goalMsg)
        rospy.loginfo("Initial goal published! Goal ")



    def statusCB(self, data):
        if data.status.status == 3: # reached
            self.goalMsg.header.stamp = rospy.Time.now)
                (
            self.goalMsg.pose.position.x = self.goalX
            self.goalMsg.pose.position.y = self.goalY
            self.pub.publish(self.goalMsg)
            rospy.loginfo("Initial goal published! Goal ")
```

c. Define the code principal

```
if __name__ == "__main:"__
    try    :
 #        ROS Init
        rospy.init_node('goal', anonymous=True)
 #        Get params
        map_frame = rospy.get_param('~map_frame', 'map' )
        retry = rospy.get_param('~retry', '1')
       goalX = 0.5
       goalY = 0.4
        mg = Goal(goalX, goalY, retry, map_frame)
        rospy.spin()


    except KeyboardInterrupt:
        print("shutting down")
```

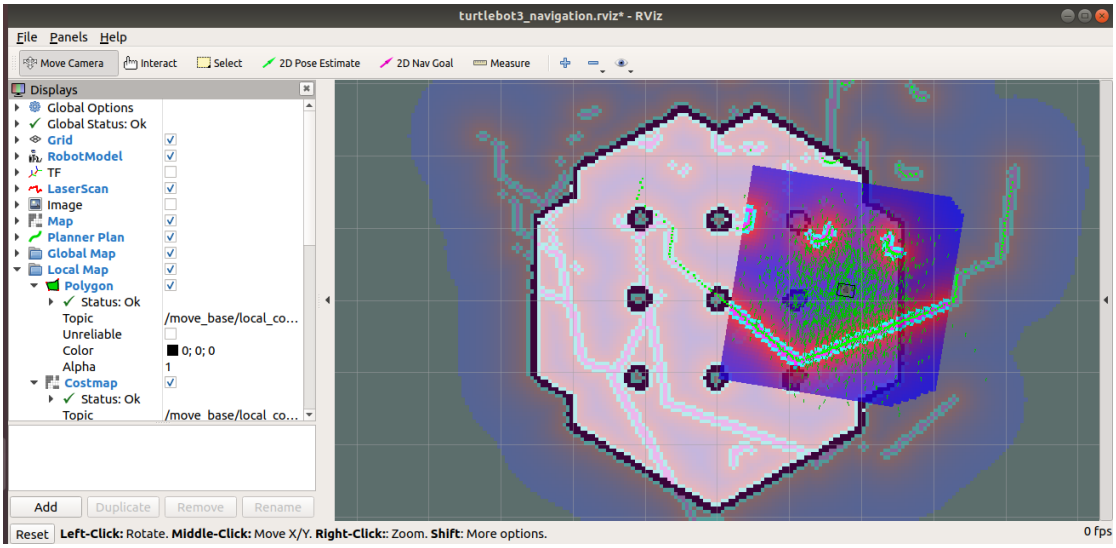
3. Launch the gazebo et Rviz code
    a. In first terminal Launch Simulation World

```
$ export TURTLEBOT3_MODEL=burger

$ roslaunch turtlebot3_gazebo turtlebot3_world.launch
```



    b. In second terminal Run Navigation Node

```
$ export TURTLEBOT3_MODEL=burger

$ roslaunch turtlebot3_navigation turtlebot3_navigation.launch map_fi
le:=$HOME/map.yaml
```



4. Run the node

```
$ rosrun navigation goal.py
```

```
jamal@jamal:~/catkin_ws$ rosrun navigation goal.py
[INFO] [1628862784.448629, 40.747000]: Initial goa
l published! Goal
```