# Lab Session 3 and 4
## Publishers and Subscribers (rospy) + Custom messages

## 1. Creating the publisher node (rospy)

Before creating our nodes, create a new package with the name pubsub
In any ROS Workspace (make sure to source the workspace in every terminal session)

```
$ cd /catkin_ws/src
$ catkin_create_pkg pubsub std_msgs rospy roscpp
$ cd ..
$ catkin_make
```

Now we can start creating our nodes. In the /src directory create a new python script with the name talker.py and add the following code inside:

```
import rospy
from geometry_msgs.msg import Point
import random

def talker():
    rospy.init_node('Talker_Node', anonymous=True)
    pub = rospy.Publisher('LOCATION', Point, queue_size=10)
    rate = rospy.Rate(5) # five messages per second
    while not rospy.is_shutdown():
        msg = Point()
         msg.x = rospy.get_time()
         msg.y = random.randint(0, 1000)
        pub.publish(msg)
        rate.sleep()

talker() # Call the talker() function
```

In order for the script to be recognized as a node, we need to edit our CMAKELISTS.txt file and add the script we just created by uncommenting lines (162 -> 164) and modifying it to look like:

```
catkin_install_python(PROGRAMS src/talker.py
  DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
)
```

After editing the CMAKELISTS.txt file, we can go ahead and execute $catkin_make inside our workspace.

Initiate roscore in a separate command window

```
$ roscore
```

To run our nodes, we will use the $rosrun command.

```
# This is an example, do not try to run this
# rosrun package_name node_name
```

In our case, we will run:

```
$ rosrun pubsub talker.py
```

To check that the node is running fine and our topic is initiated, we will execute $rosnode list to list all active nodes and $rostopic list to list all active topics.

```
$ rosnode list
$ rostopic list
```

We can display the contents of any active topic straight through our terminal by using the $rostopic echo command

```
$ rostopic echo /LOCATION
```

## 2. Creating the subscriber node (rospy)

In the /src directory create a new python script with the name listener.py and add the following code inside

```
import rospy
from geometry_msgs.msg import Point

def ros_callback(msg):
    rospy.loginfo(f'The Sum of the Points is {msg.x+msg.y}')

def listener():
    rospy.init_node('LISTENER_NODE', anonymous=True)
    rospy.Subscriber("LOCATION", Point, ros_callback)
    rospy.spin() # spin() simply keeps python from exiting

listener() # Call the listener() function
```

In order for the script to be recognized as a node, we need to edit our CMAKELISTS.txt file and add the script we just created by uncommenting lines (162 -> 164) and modifying it to look like:

```
catkin_install_python(PROGRAMS src/talker.py src/listener.py
  DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
)
```

After editing the CMAKELISTS.txt file, we can go ahead and execute $catkin_make inside our workspace.

Then use $rosrun to run the node

```
$ rosrun pubsub listener.py
```

## 3. Creating our own message for our package

It is always better to create our own messages inside our package that specifically meets the needs of our project. To do this we are going to create a new /msg directory in our package and add a new file called person.msg which will hold our custom message. Either open gedit and add the lines manually or use the echo command inside the terminal

```
$ roscd pubsub
$ mkdir msg
$ echo 'string name' >> msg/person.msg
$ echo 'int32 age' >> msg/person.msg
```

Now we need to modify both our CMAKELISTS.txt as well as our package.xml file.
open the package.xml file and add these two lines

```
<build_depend>message_generation</build_depend>
<exec_depend>message_runtime</exec_depend>
```

Next, open the CMAKELISTS.txt file and perform the following:

1. Add message_generation to your list of dependencies (lines 10 -> 14) so it looks like this:

```
find_package(catkin REQUIRED COMPONENTS
  roscpp
  rospy
  std_msgs
  message_generation
)
```

2. Export the message_runtime dependency by adding it lines (106 -> 111) in so it looks like this:

```
catkin_package(
  ...
  CATKIN_DEPENDS message_runtime ...
  ...
)
```

3. Uncomment lines (51 -> 55) and add your message file so it looks like this:

```
add_message_files(
  FILES
  person.msg
)
```

4. Now uncomment lines (72 -> 75) to ensure the generate_messages() function is called upon build, it should look like this:

```
generate_messages(
  DEPENDENCIES
  std_msgs
)
```

After modifying both files, we can now execute $catkin_make inside our workspace.

Now use the $rosmsg show command to ensure our message is visible within our environment

```
$ rosmsg show person
```