Creating a ROS package:

`catkin_create_pkg <package_name> [depend1] [depend2] [depend…]`

Example: `catkin_create_pkg chatter_demo std_msgs roscpp rospy`

Must run this command inside `catkin_ws/src` (i.e. run `cd ~/catkin_ws/src` before running this command).

Writing Publisher and Subscriber in C++:

http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29

What to add to CMakeLists.txt (all the way at the bottom):

```
add_executable(talker src/talker.cpp)
target_link_libraries(talker ${catkin_LIBRARIES})

add_executable(listener src/listener.cpp)
target_link_libraries(listener ${catkin_LIBRARIES})
```

Change `talker` and `listener` if the names of the C++ files are different!

Whenever you modify a C++ file:

```
cd ~/catkin_ws
catkin_make
```

Writing Publisher and Subscriber in Python:

http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29

After you created the Python scripts (before running the script):

```
cd <location_of_python_script>
chmod +x <the_python_script>
```

No need to redo `catkin_make` after modifying Python files.

Running the `chatter` demo:

Uses 3 terminals (enter each command into a new terminal).

```
roscore
rosrun chatter_demo talker
rosrun chatter_demo listener
```

Some `rostopic` tools (`roscore` must be running):

- `rostopic list`
- `rostopic info /chatter`
- `rostopic echo /chatter`
- `rostopic hz /chatter`

Some `rosmsg` tools (`roscore` must be running):

- `rosmsg info`
- `rosmsg list`

Some `rosnode` tools (`roscore` must be running):

- `rosnode info`
- `rosnode list`

To see how all ROS nodes are connected:

```
rqt_graph
```

Then, open ConstructSim's graphical tool so the GUI window can appear.

Launching a launch file:

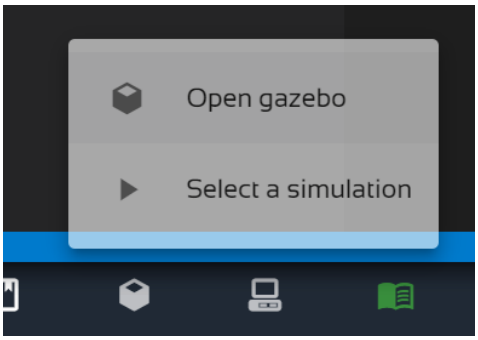
```
roslaunch <package_containing_launch_file>
<name_of_launch_file>
```

Example: `roslaunch chatter_demo chatter.launch`

Launching the Turtlebot demo:

```
roslaunch turtlebot_gazebo main.launch
```

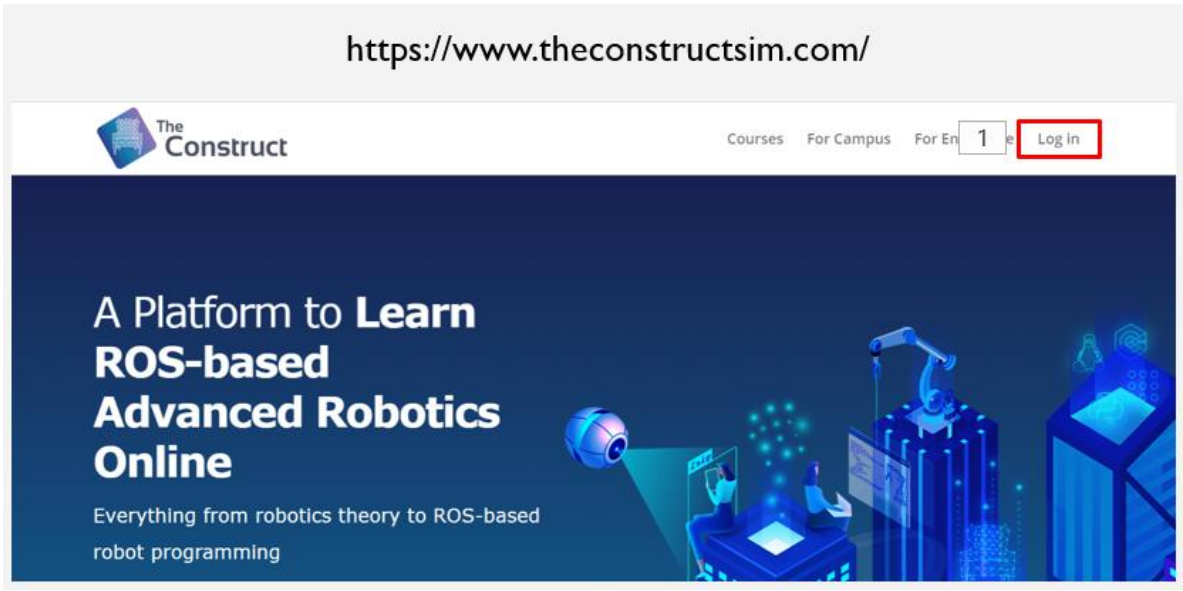
Then, open ConstrucSim's Gazebo window.

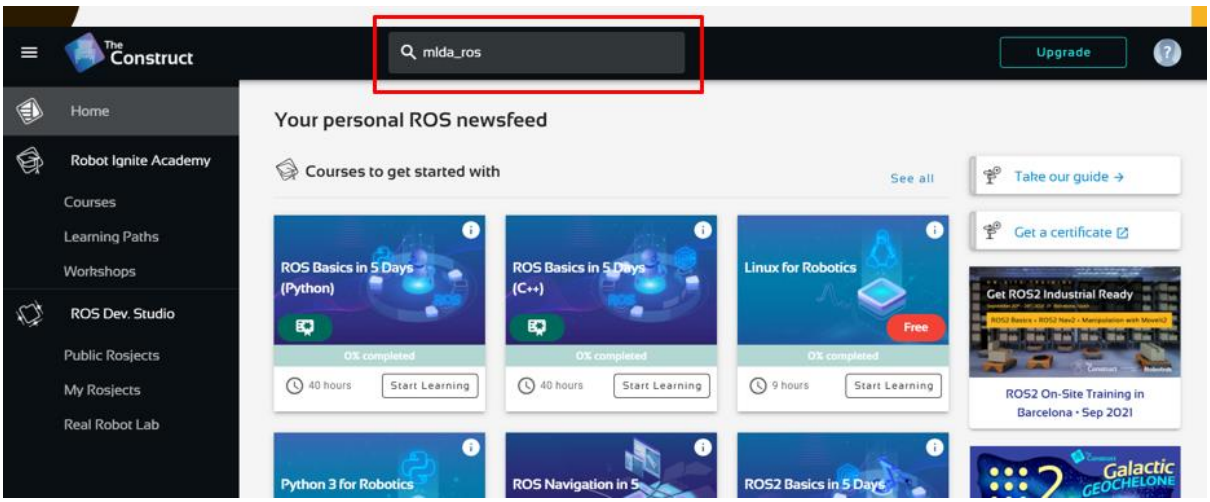


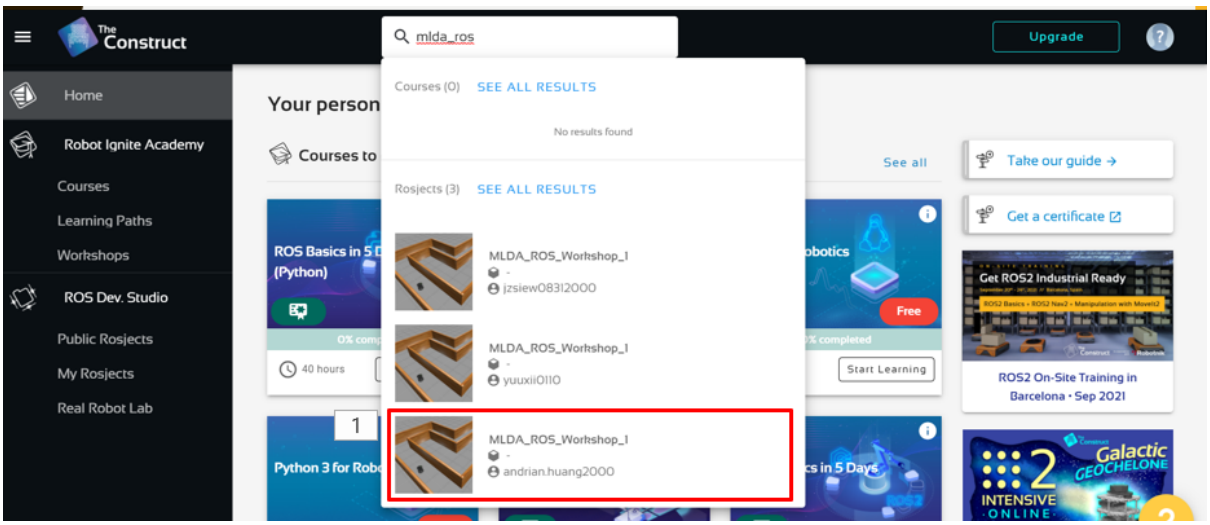

```
rosrun teleop_twist_keyboard teleop_twist_keyboard.py
```

Follow the instruction on the terminal, press the keys to drive the robot around.

Forking this workshop's ROSject:







There should be a “Fork” button like this one (this is a screenshot of another ROSject, not ours):