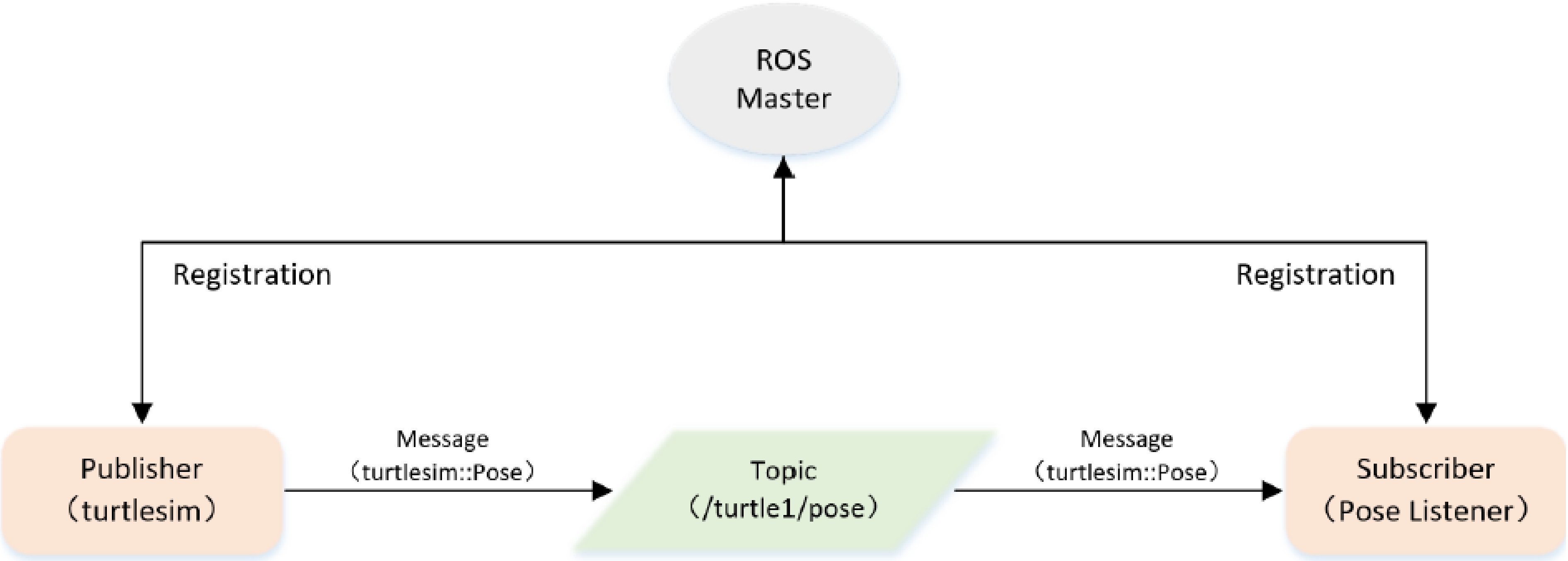
ECE 4703
Mobile Autonomous Robots

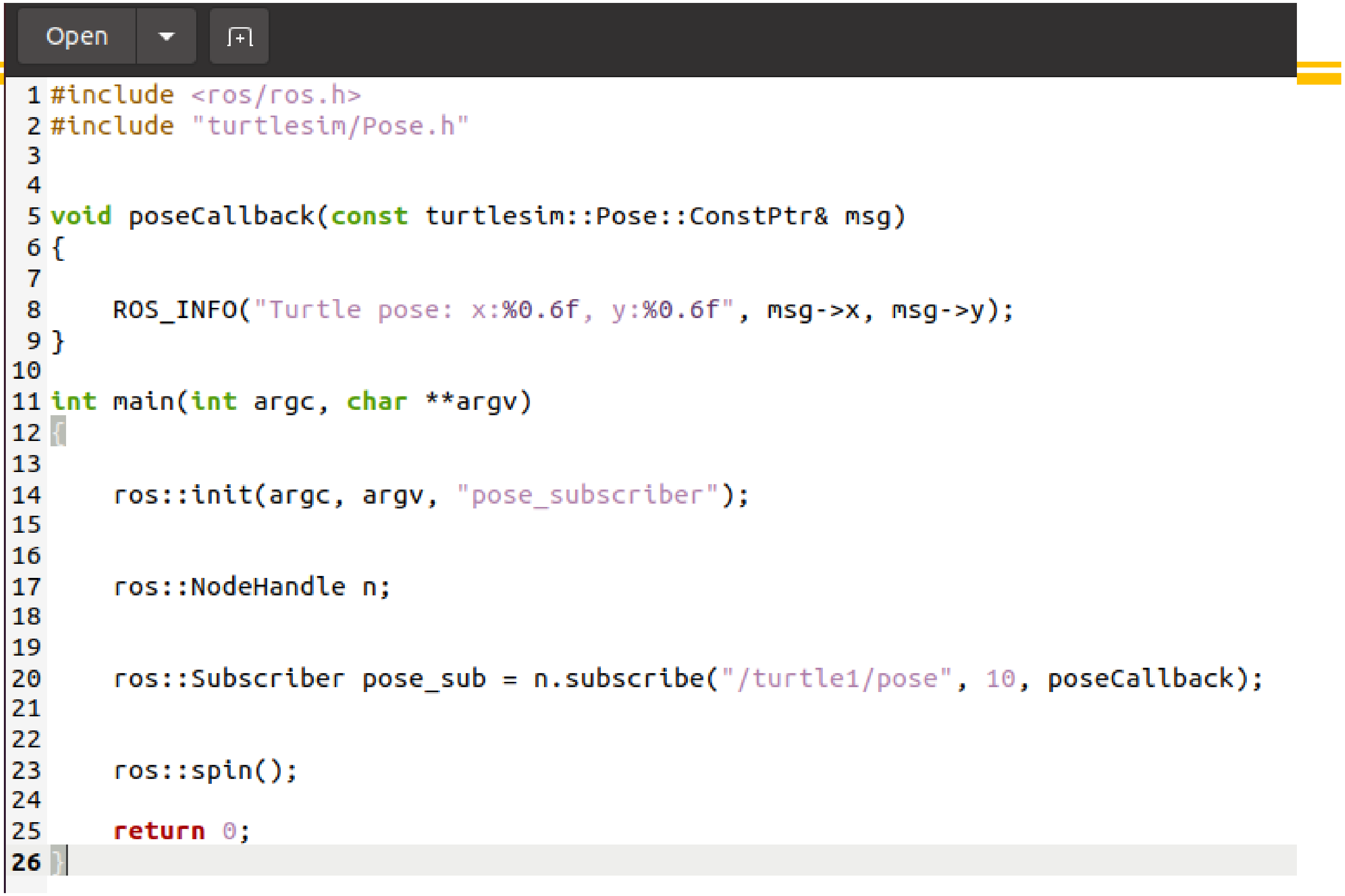
Jenny Zhen Yu
zhenyu@cpp.edu
Department of Electrical and Computer Engineering
California State Polytechnic University, Pomona

Lecture 6: Subscriber Programming

Topic

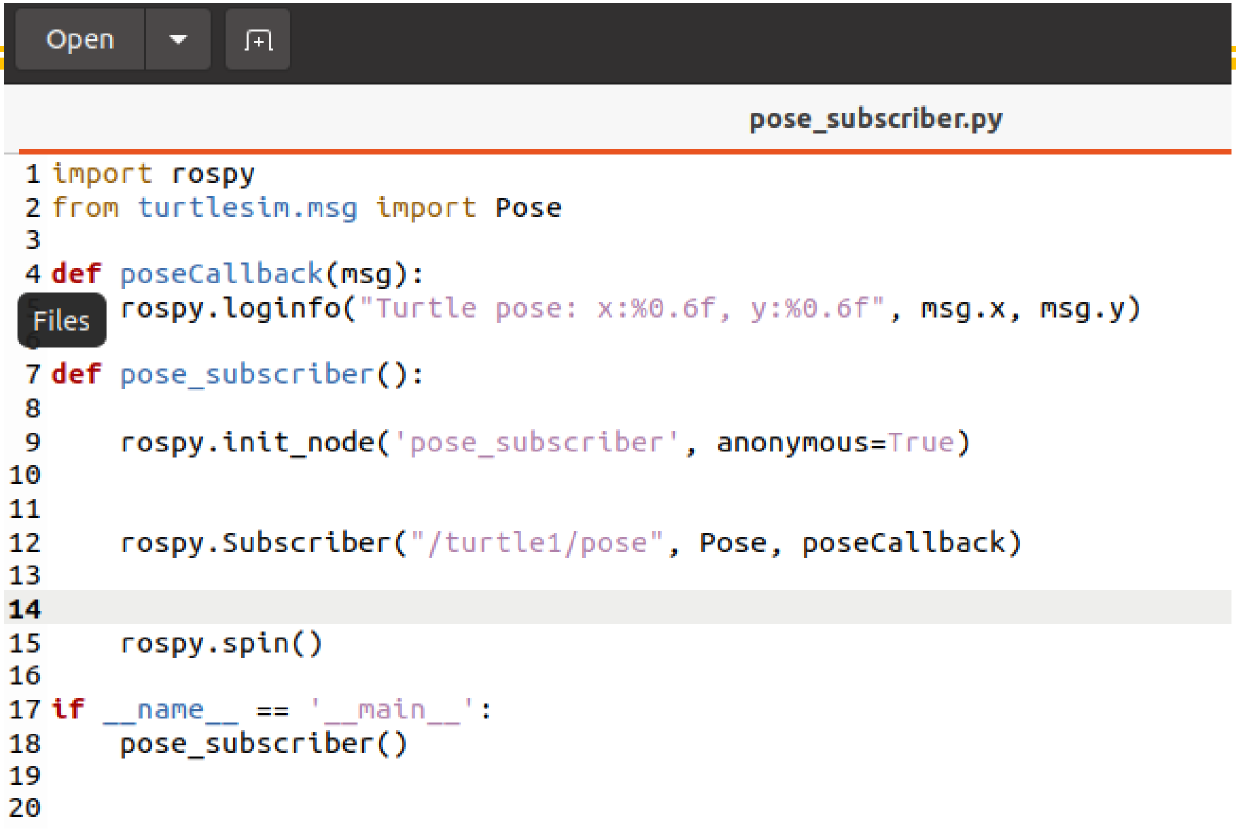

ROS
Master
Registration
Registration
Publisher
（turtlesim）
Message
（turtlesim::Pose）
Topic
（/turtle1/pose）
Message
（turtlesim::Pose）
Subscriber
（Pose Listener）

C++ Code



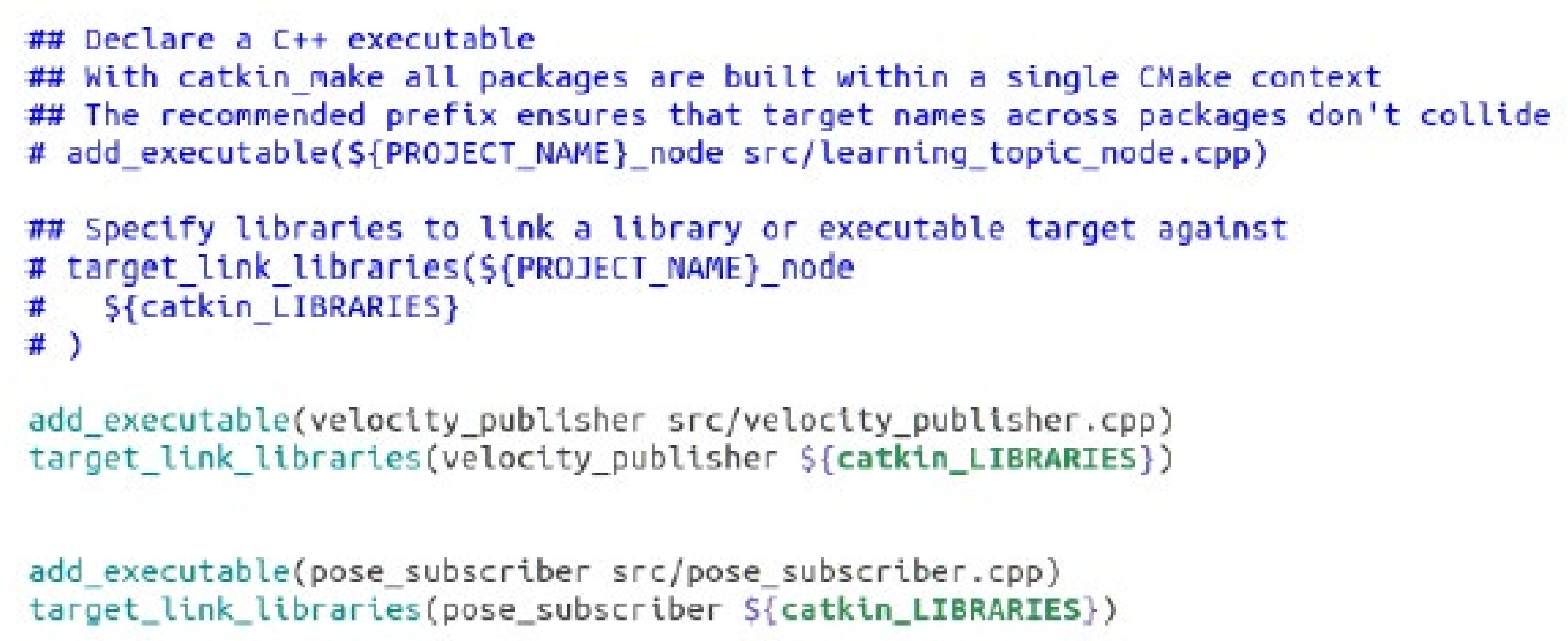
```cpp
#include <ros/ros.h>
#include "turtlesim/Pose.h"


void poseCallback(const turtlesim::Pose::ConstPtr& msg)
{

    ROS_INFO("Turtle pose: x:%0.6f, y:%0.6f", msg->x, msg->y);
}

int main(int argc, char **argv)
{

    ros::init(argc, argv, "pose_subscriber");


    ros::NodeHandle n;


    ros::Subscriber pose_sub = n.subscribe("/turtle1/pose", 10, poseCallback);


    ros::spin();

    return 0;
}
```

Python Code



```python
import rospy
from turtlesim.msg import Pose

def poseCallback(msg):
    rospy.loginfo("Turtle pose: x:%0.6f, y:%0.6f", msg.x, msg.y)

def pose_subscriber():

    rospy.init_node('pose_subscriber', anonymous=True)


    rospy.Subscriber("/turtle1/pose", Pose, poseCallback)


    rospy.spin()

if __name__ == '__main__':
    pose_subscriber()


```

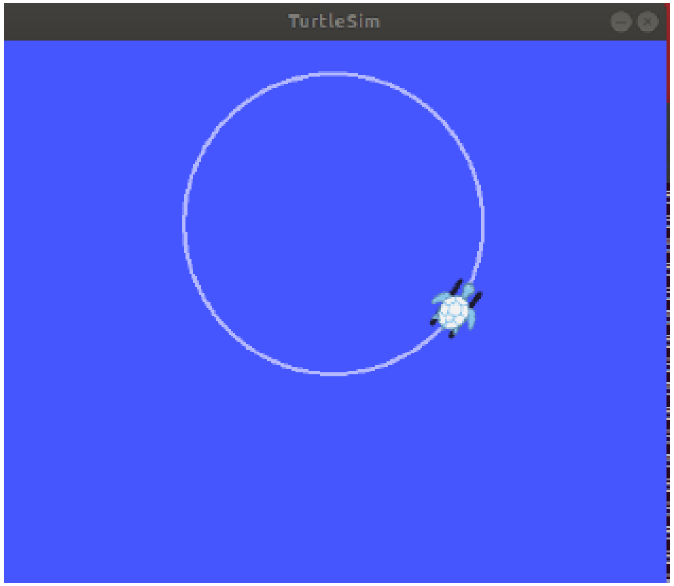
Compiling Code

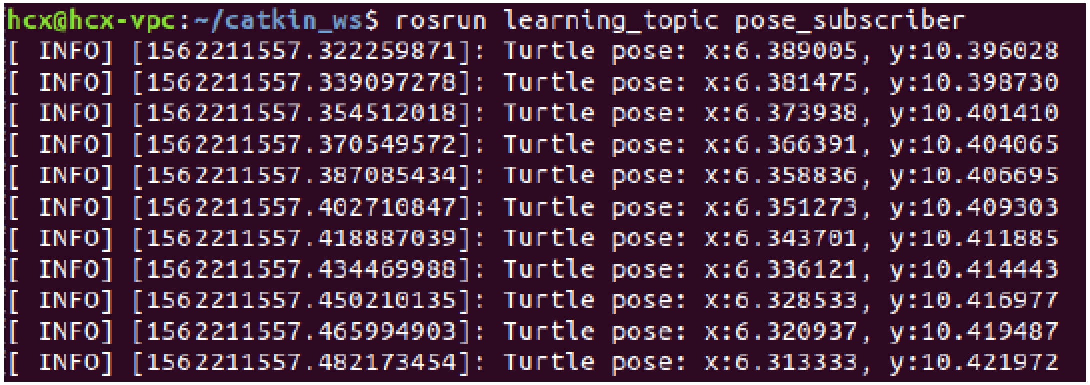


```
## Declare a C++ executable
## With catkin_make all packages are built within a single CMake context
## The recommended prefix ensures that target names across packages don't collide
# add_executable(${PROJECT_NAME}_node src/learning_topic_node.cpp)

## Specify libraries to link a library or executable target against
# target_link_libraries(${PROJECT_NAME}_node
#   ${catkin_LIBRARIES}
# )

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


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

Run Program

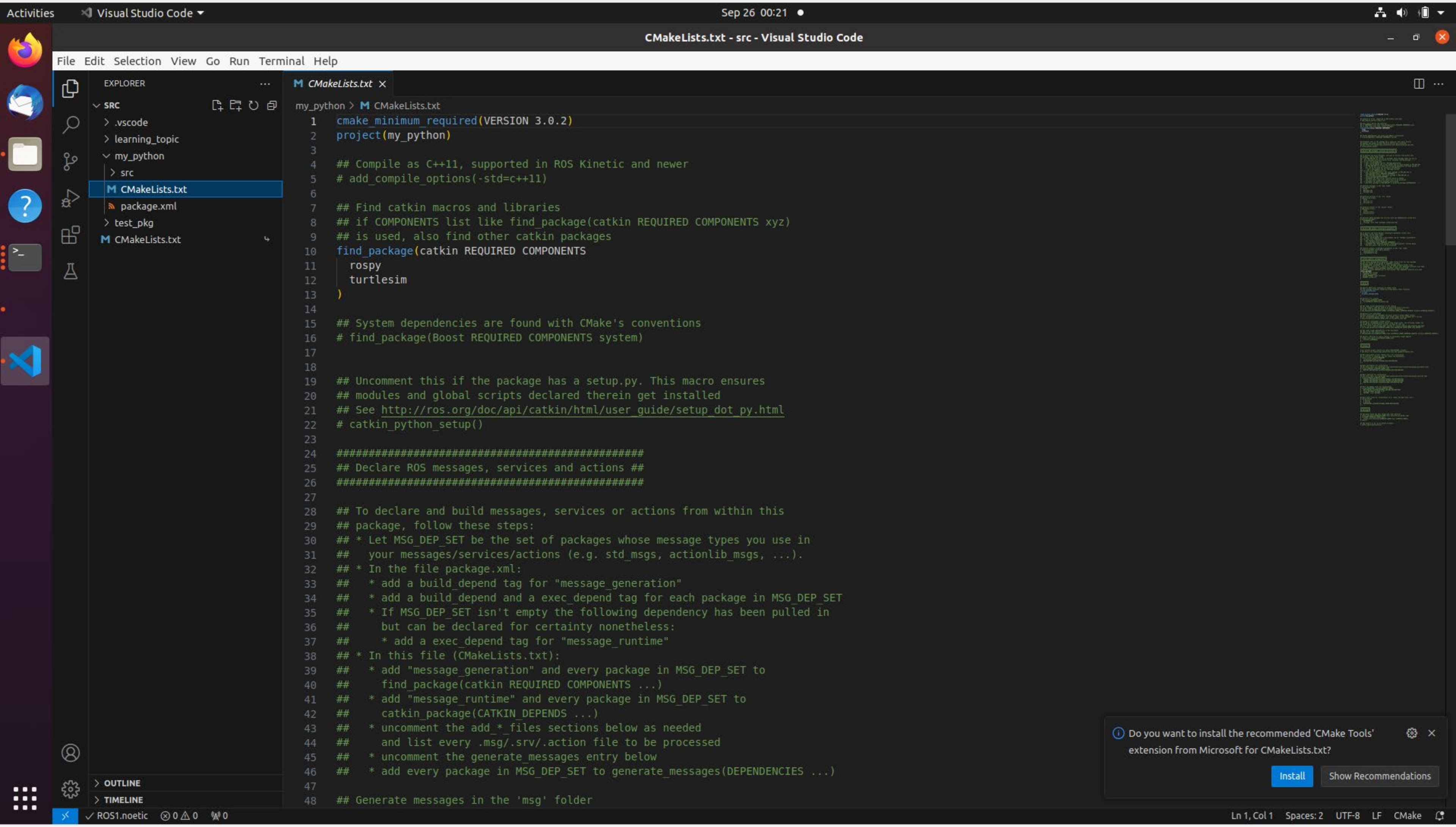
```
$ cd ~/catkin_ws
$ catkin_make
$ source devel/setup.bash
$ roscore
$ rosrun turtlesim turtlesim_node
$ rosrun learning_topic velocity_publisher
```





```
hcx@hcx-vpc:~/catkin_ws$ rosrun learning_topic pose_subscriber
[ INFO] [1562211557.322259871]: Turtle pose: x:6.389005, y:10.396028
[ INFO] [1562211557.339097278]: Turtle pose: x:6.381475, y:10.398730
[ INFO] [1562211557.354512018]: Turtle pose: x:6.373938, y:10.401410
[ INFO] [1562211557.370549572]: Turtle pose: x:6.366391, y:10.404065
[ INFO] [1562211557.387085434]: Turtle pose: x:6.358836, y:10.406695
[ INFO] [1562211557.402710847]: Turtle pose: x:6.351273, y:10.409303
[ INFO] [1562211557.418887039]: Turtle pose: x:6.343701, y:10.411885
[ INFO] [1562211557.434469988]: Turtle pose: x:6.336121, y:10.414443
[ INFO] [1562211557.450210135]: Turtle pose: x:6.328533, y:10.416977
[ INFO] [1562211557.465994903]: Turtle pose: x:6.320937, y:10.419487
[ INFO] [1562211557.482173454]: Turtle pose: x:6.313333, y:10.421972
```

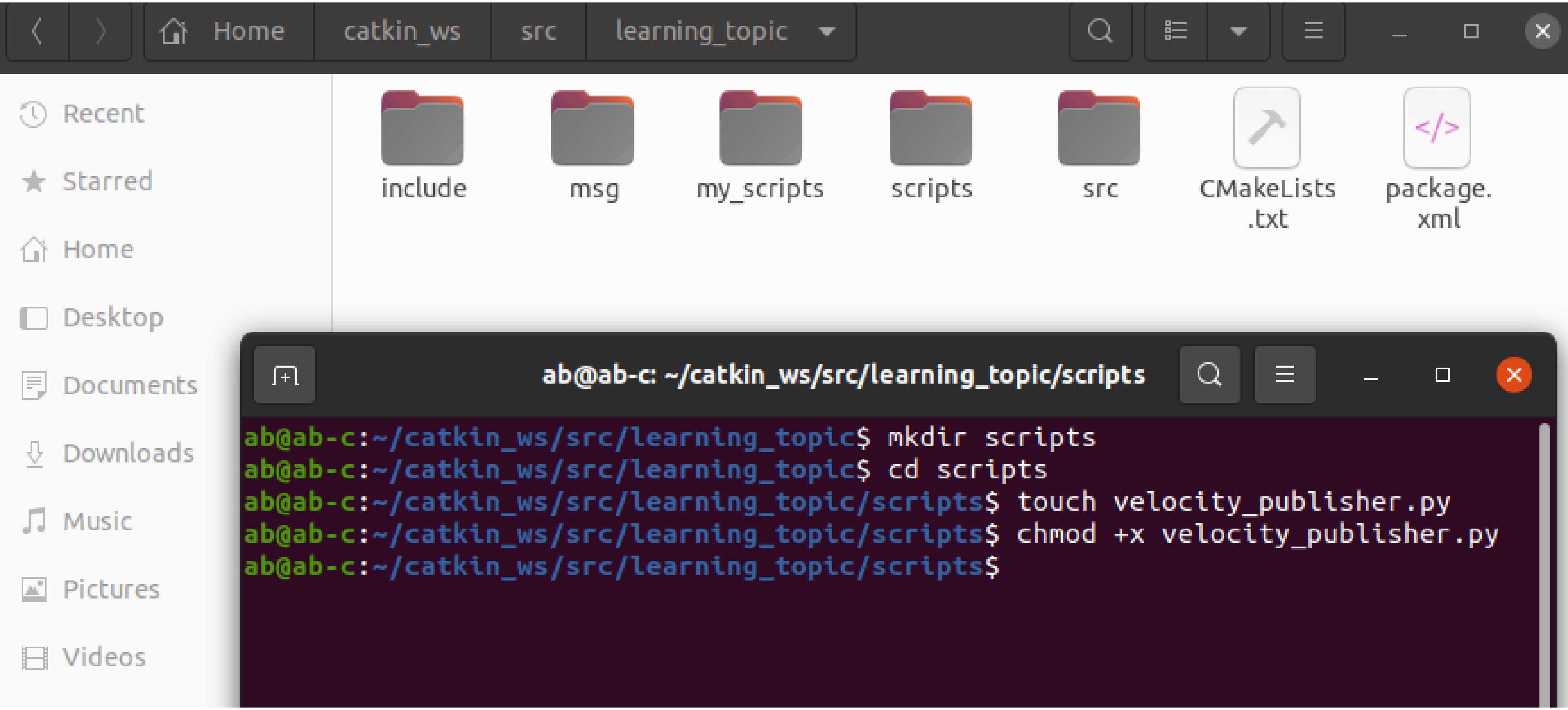
Python



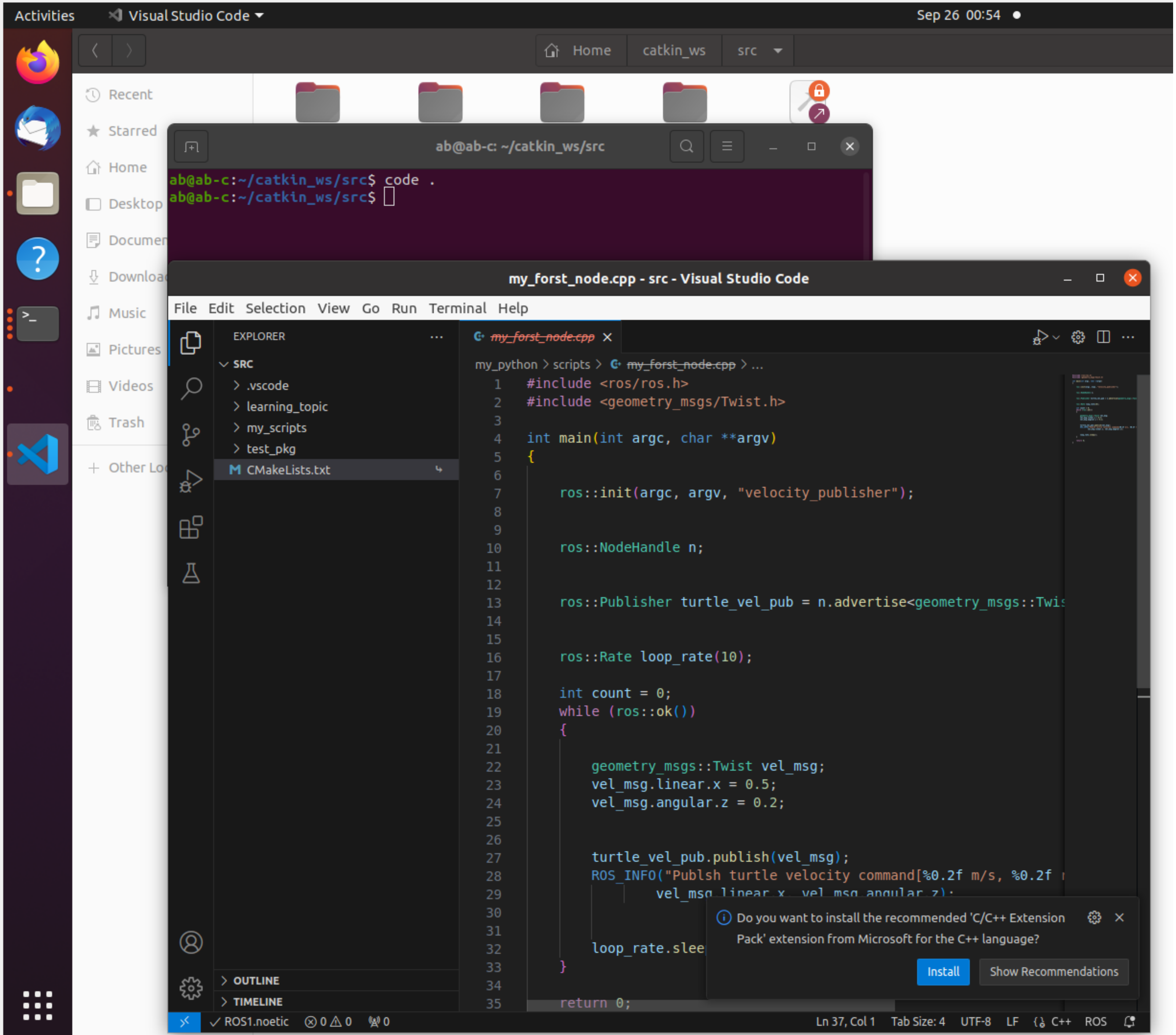
```
cmake_minimum_required(VERSION 3.0.2)
project(my_python)

## Compile as C++11, supported in ROS Kinetic and newer
# add_compile_options(-std=c++11)

## Find catkin macros and libraries
## if COMPONENTS list like find_package(catkin REQUIRED COMPONENTS xyz)
## is used, also find other catkin packages
find_package(catkin REQUIRED COMPONENTS
  rospy
  turtlesim
)

## System dependencies are found with CMake's conventions
# find_package(Boost REQUIRED COMPONENTS system)


## Uncomment this if the package has a setup.py. This macro ensures
## modules and global scripts declared therein get installed
## See http://ros.org/doc/api/catkin/html/user_guide/setup_dot_py.html
# catkin_python_setup()

################################################
## Declare ROS messages, services and actions ##
################################################

## To declare and build messages, services or actions from within this
## package, follow these steps:
## * Let MSG_DEP_SET be the set of packages whose message types you use in
##   your messages/services/actions (e.g. std_msgs, actionlib_msgs, ...).
## * In the file package.xml:
##   * add a build_depend tag for "message_generation"
##   * add a build_depend and a exec_depend tag for each package in MSG_DEP_SET
##   * If MSG_DEP_SET isn't empty the following dependency has been pulled in
##     but can be declared for certainty nonetheless:
##     * add a exec_depend tag for "message_runtime"
## * In this file (CMakeLists.txt):
##   * add "message_generation" and every package in MSG_DEP_SET to
##     find_package(catkin REQUIRED COMPONENTS ...)
##   * add "message_runtime" and every package in MSG_DEP_SET to
##     catkin_package(CATKIN_DEPENDS ...)
##   * uncomment the add_*_files sections below as needed
##     and list every .msg/.srv/.action file to be processed
##   * uncomment the generate_messages entry below
##   * add every package in MSG_DEP_SET to generate_messages(DEPENDENCIES ...)

## Generate messages in the 'msg' folder
```

Do you want to install the recommended 'CMake Tools' extension from Microsoft for CMakeLists.txt?

Scripts



```
ab@ab-c:~/catkin_ws/src/learning_topic$ mkdir scripts
ab@ab-c:~/catkin_ws/src/learning_topic$ cd scripts
ab@ab-c:~/catkin_ws/src/learning_topic/scripts$ touch velocity_publisher.py
ab@ab-c:~/catkin_ws/src/learning_topic/scripts$ chmod +x velocity_publisher.py
ab@ab-c:~/catkin_ws/src/learning_topic/scripts$
```



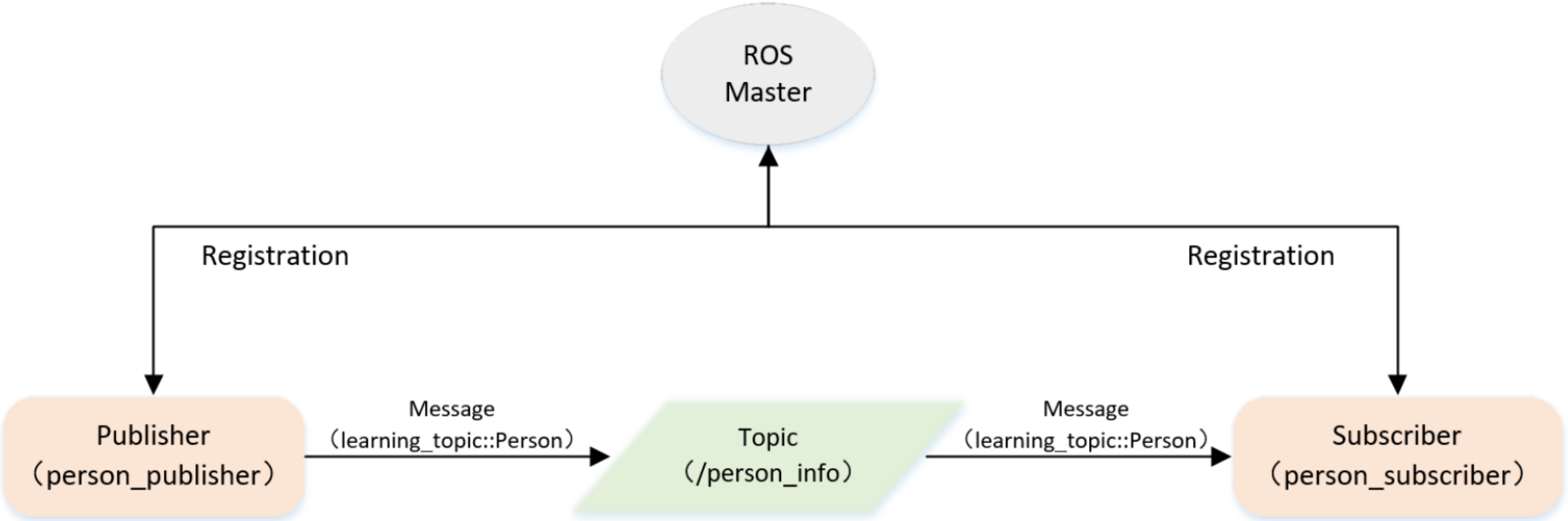
```
ab@ab-c:~/catkin_ws/src$ code .
ab@ab-c:~/catkin_ws/src$ 
```

```cpp
#include <ros/ros.h>
#include <geometry_msgs/Twist.h>

int main(int argc, char **argv)
{

    ros::init(argc, argv, "velocity_publisher");


    ros::NodeHandle n;


    ros::Publisher turtle_vel_pub = n.advertise<geometry_msgs::Twis


    ros::Rate loop_rate(10);

    int count = 0;
    while (ros::ok())
    {

        geometry_msgs::Twist vel_msg;
        vel_msg.linear.x = 0.5;
        vel_msg.angular.z = 0.2;


        turtle_vel_pub.publish(vel_msg);
        ROS_INFO("Publsh turtle velocity command[%0.2f m/s, %0.2f r
                vel_msg.linear.x, vel_msg.angular.z);


        loop_rate.slee
    }

    return 0;
```

Do you want to install the recommended 'C/C++ Extension Pack' extension from Microsoft for the C++ language?

Lecture 7: Topic Message

Topic Model

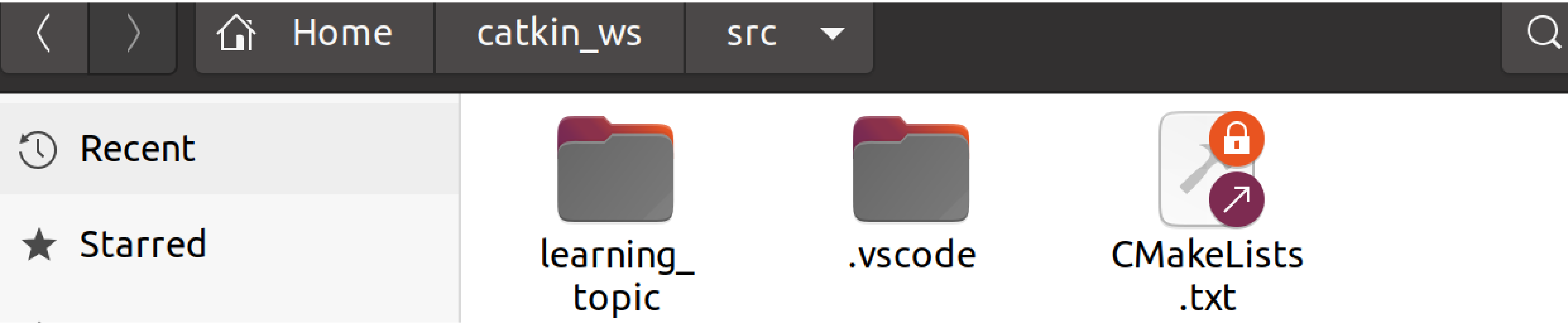

ROS
Master
Registration
Registration
Publisher
（person_publisher）
Message
（learning_topic::Person）
Topic
（/person_info）
Message
（learning_topic::Person）
Subscriber
（person_subscriber）

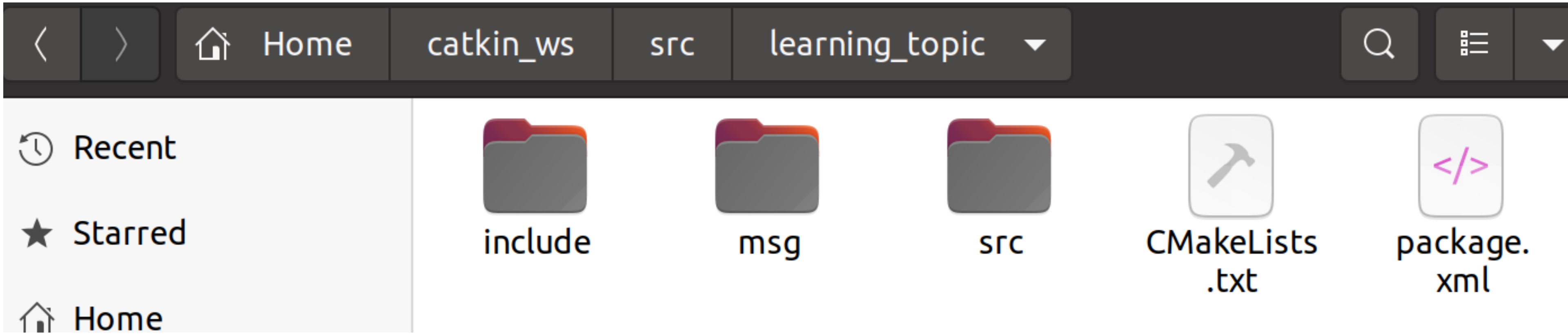
Topic Model

```
string name
uint8 sex
uint8 age

unit8 unknown = 0
uint8 male = 1
uint8 female =2
```

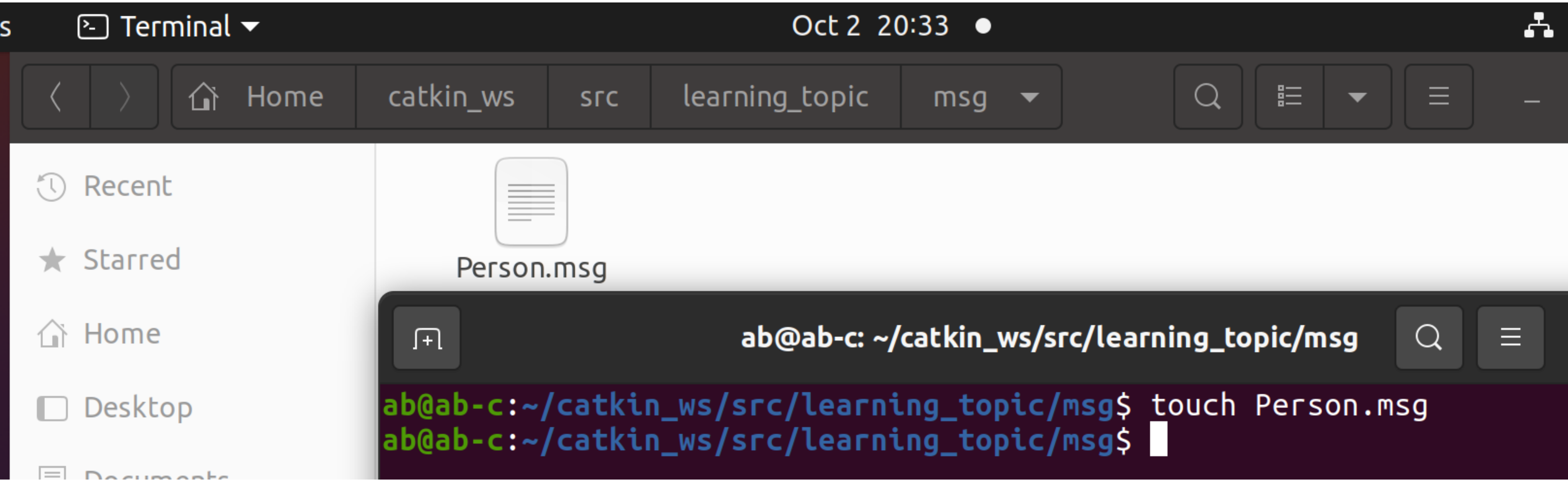
Person.msg

Build msg in learning_topic



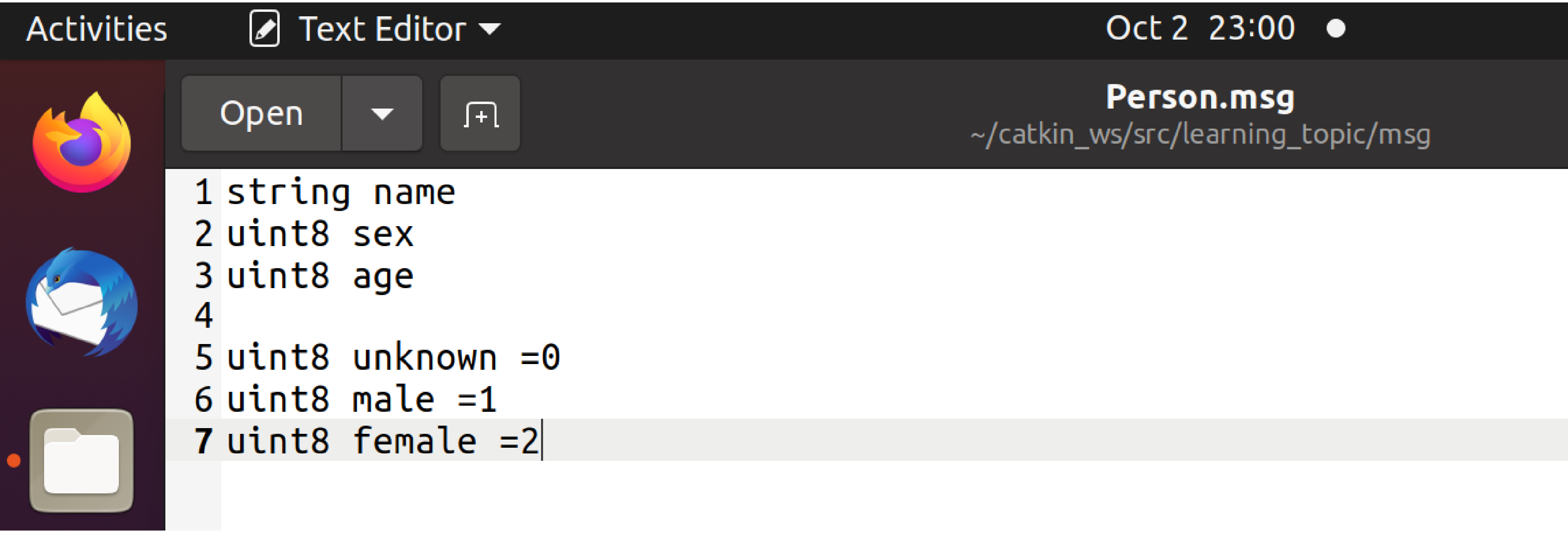


Build Person.msg



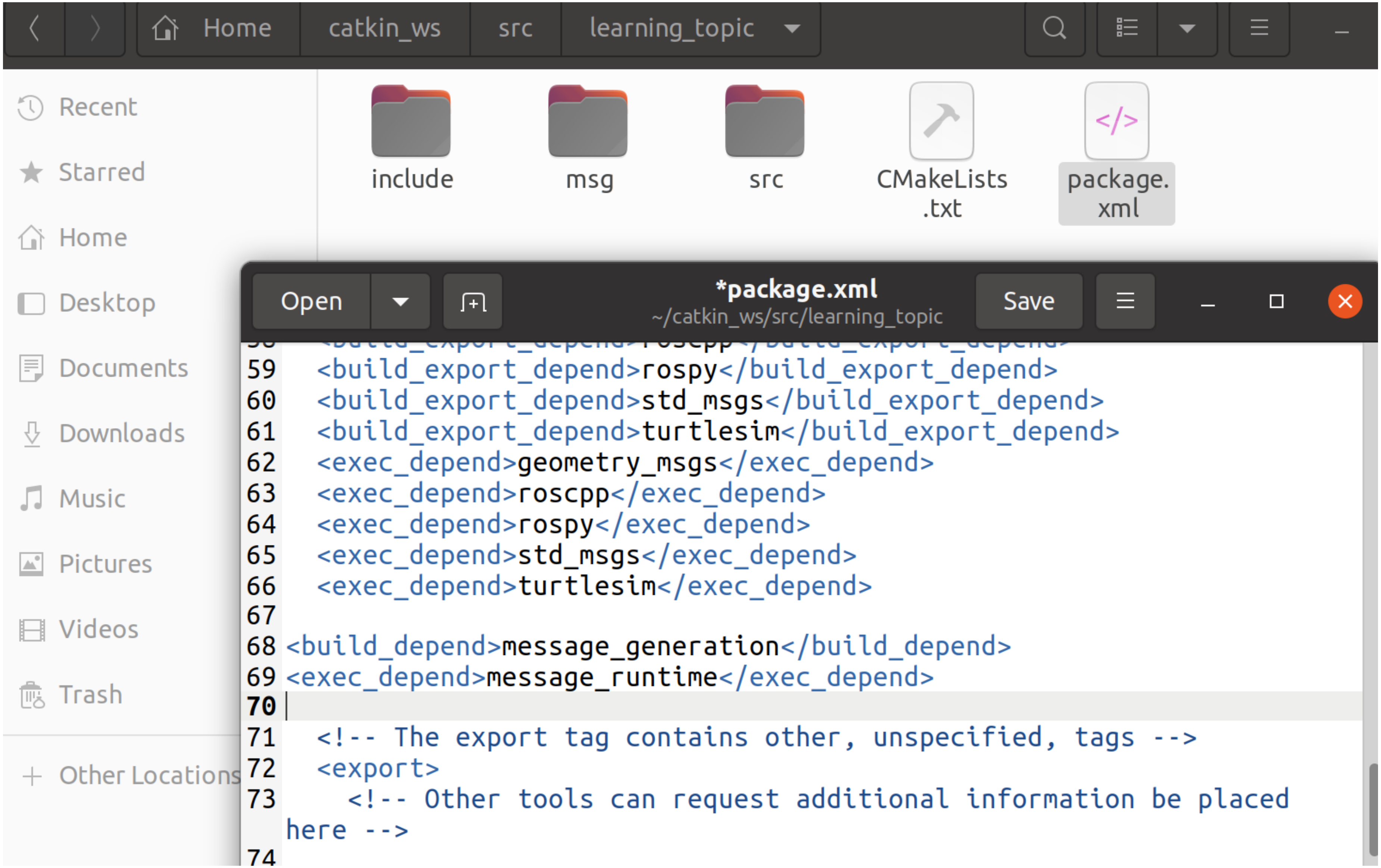
```
ab@ab-c:~/catkin_ws/src/learning_topic/msg$ touch Person.msg
ab@ab-c:~/catkin_ws/src/learning_topic/msg$ 
```

Data Interface Definition Person.msg



```
string name
uint8 sex
uint8 age

uint8 unknown =0
uint8 male =1
uint8 female =2
```

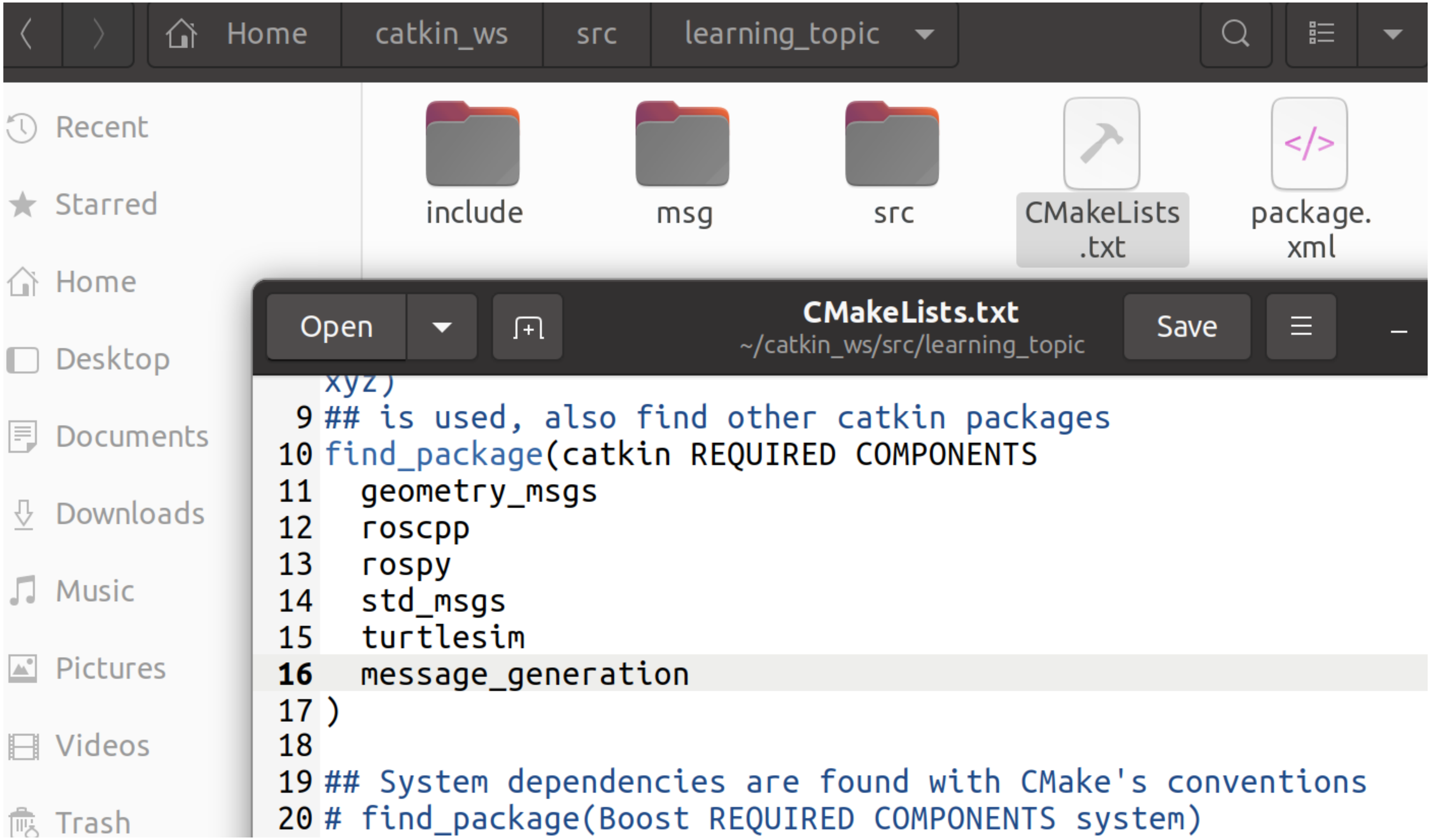
Add Dependence in package.xml



```
  <build_export_depend>roscpp</build_export_depend>
  <build_export_depend>rospy</build_export_depend>
  <build_export_depend>std_msgs</build_export_depend>
  <build_export_depend>turtlesim</build_export_depend>
  <exec_depend>geometry_msgs</exec_depend>
  <exec_depend>roscpp</exec_depend>
  <exec_depend>rospy</exec_depend>
  <exec_depend>std_msgs</exec_depend>
  <exec_depend>turtlesim</exec_depend>

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

  <!-- The export tag contains other, unspecified, tags -->
  <export>
    <!-- Other tools can request additional information be placed here -->
```

Add Compile Options in CMakeLists.txt

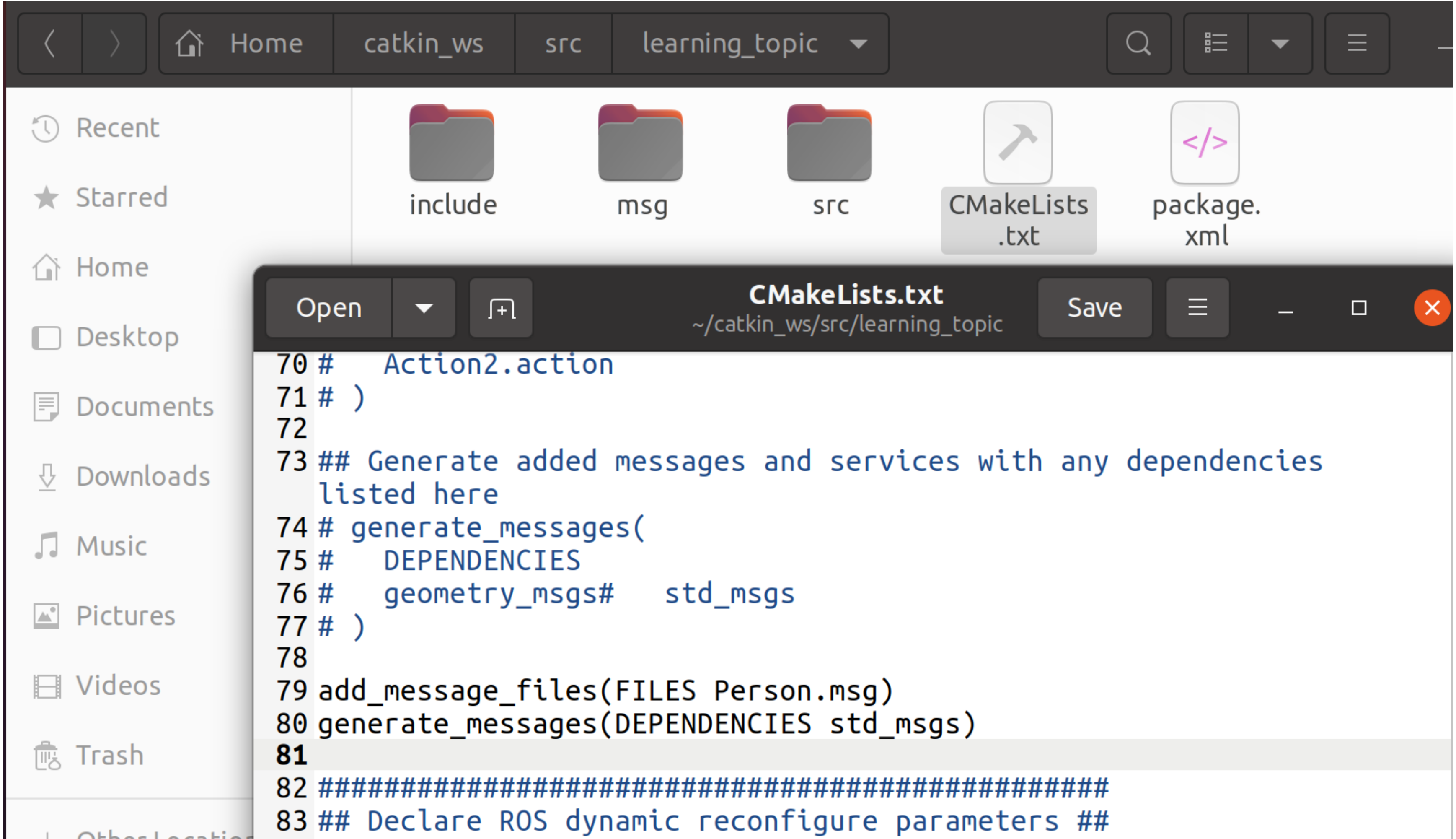
1. message_generation



```
xyz)
## is used, also find other catkin packages
find_package(catkin REQUIRED COMPONENTS
  geometry_msgs
  roscpp
  rospy
  std_msgs
  turtlesim
  message_generation
)

## System dependencies are found with CMake's conventions
# find_package(Boost REQUIRED COMPONENTS system)
```

Add Compile Options in CMakeLists.txt

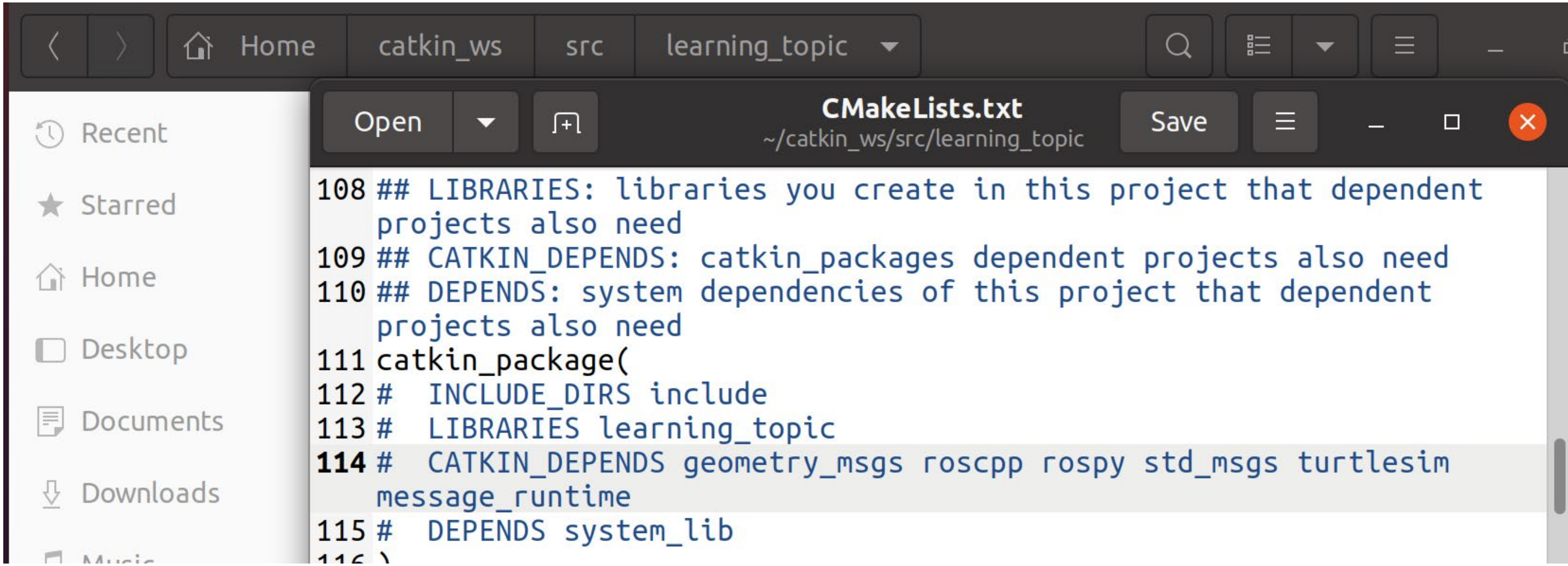
2. add_message_files(FILES Person.msg)
generate_messages(DEPENDENCIES std_msgs)



```
#   Action2.action
# )

## Generate added messages and services with any dependencies listed here
# generate_messages(
#   DEPENDENCIES
#   geometry_msgs#   std_msgs
# )

add_message_files(FILES Person.msg)
generate_messages(DEPENDENCIES std_msgs)

################################################
## Declare ROS dynamic reconfigure parameters ##
```

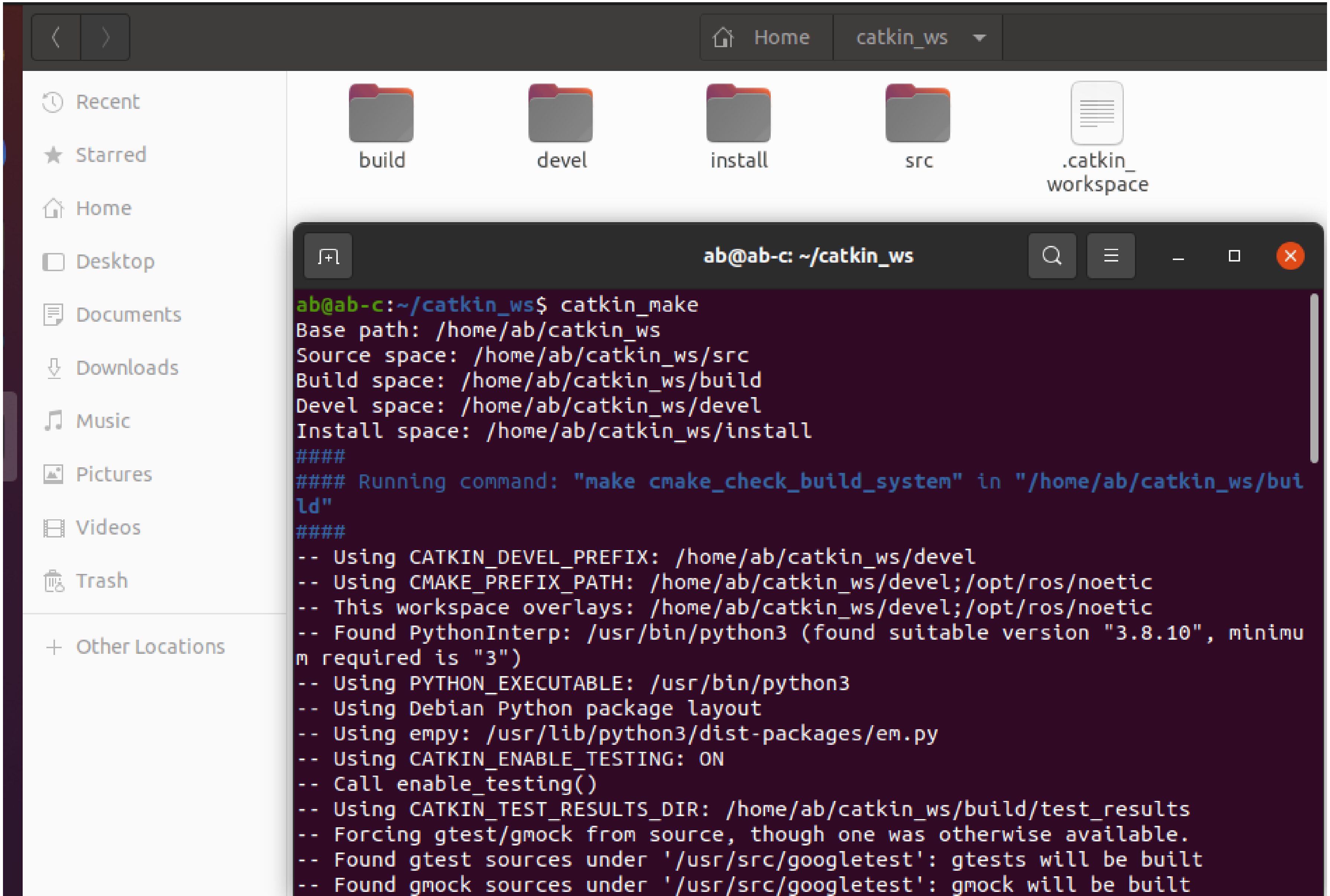
Add Compile Options in CMakeLists.txt

3. message_runtime



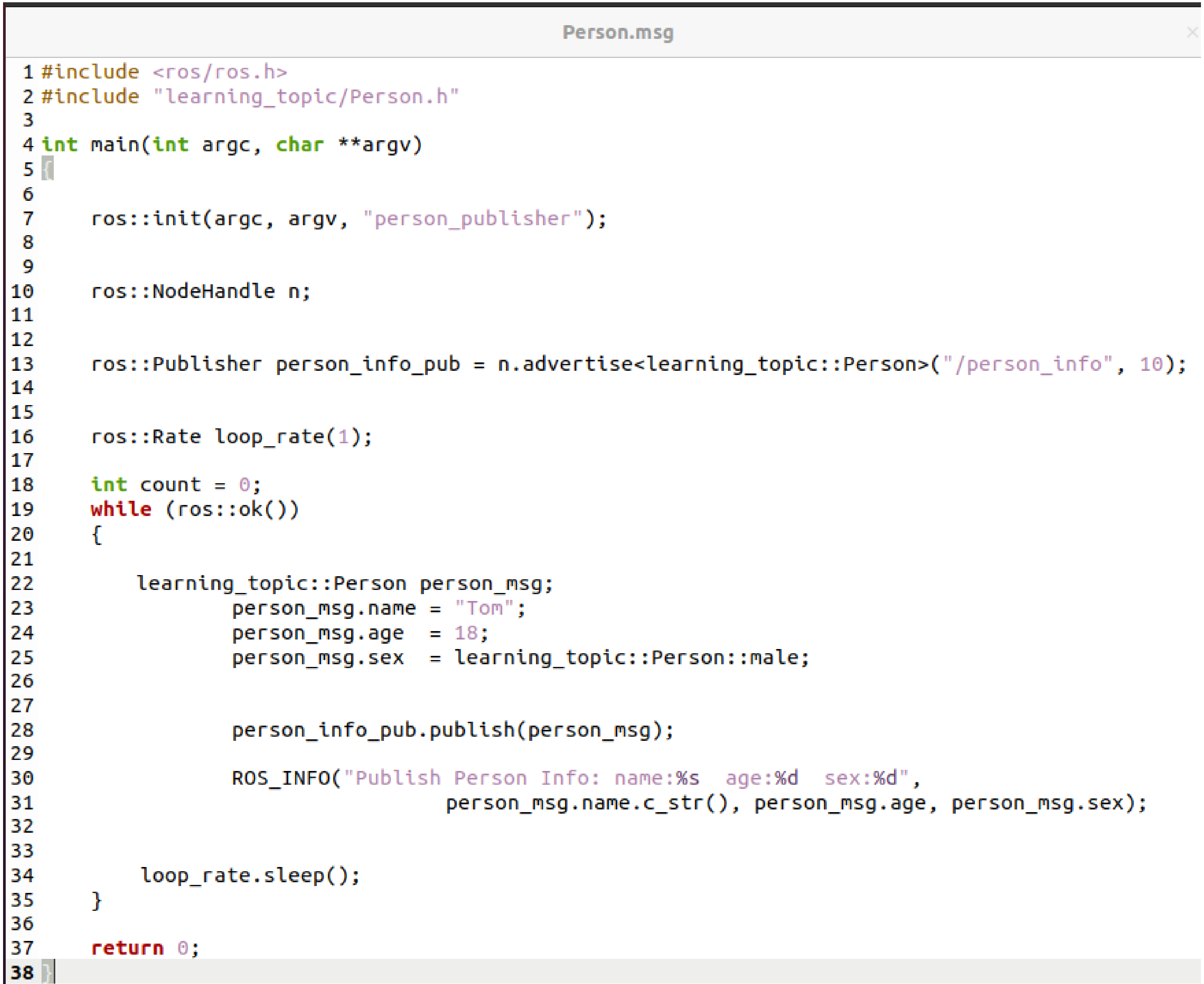
```
## LIBRARIES: libraries you create in this project that dependent projects also need
## CATKIN_DEPENDS: catkin_packages dependent projects also need
## DEPENDS: system dependencies of this project that dependent projects also need
catkin_package(
#  INCLUDE_DIRS include
#  LIBRARIES learning_topic
#  CATKIN_DEPENDS geometry_msgs roscpp rospy std_msgs turtlesim message_runtime
#  DEPENDS system_lib
)
```

catkin_make Compile



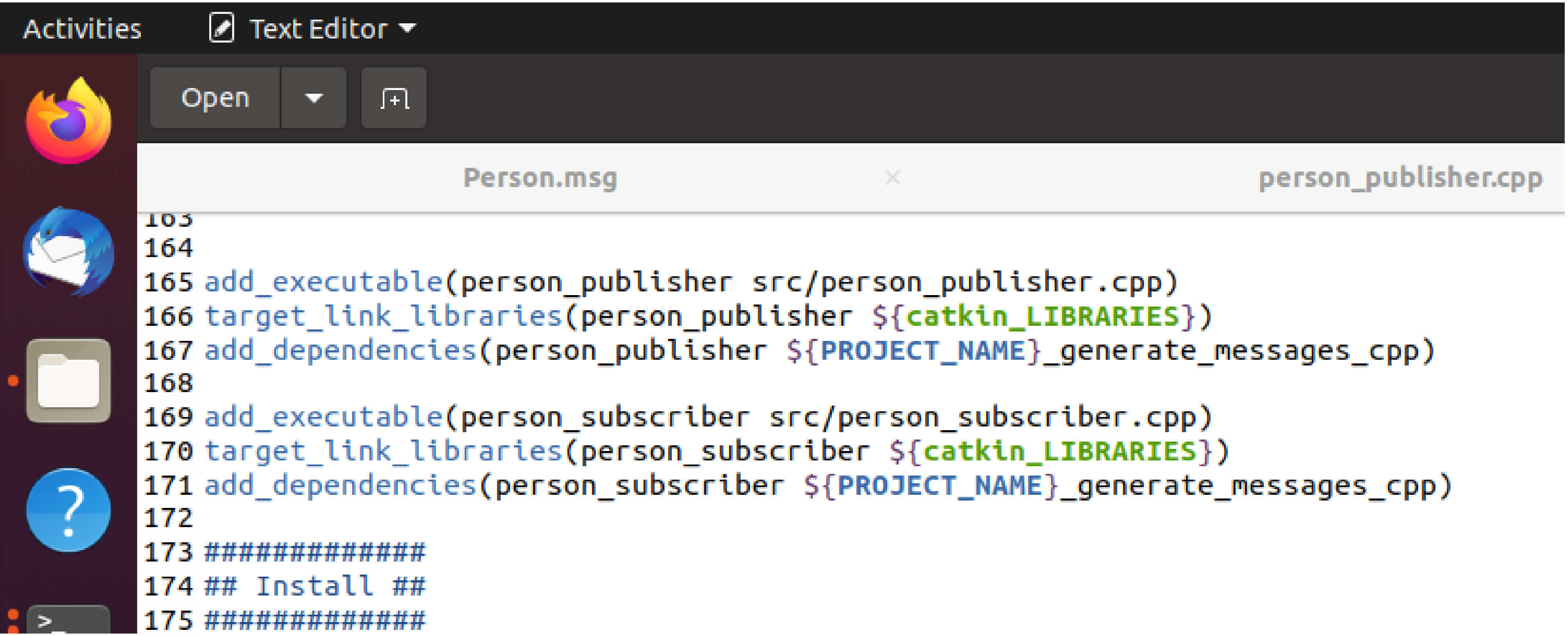
```
ab@ab-c:~/catkin_ws$ catkin_make
Base path: /home/ab/catkin_ws
Source space: /home/ab/catkin_ws/src
Build space: /home/ab/catkin_ws/build
Devel space: /home/ab/catkin_ws/devel
Install space: /home/ab/catkin_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/ab/catkin_ws/bui
ld"
####
-- Using CATKIN_DEVEL_PREFIX: /home/ab/catkin_ws/devel
-- Using CMAKE_PREFIX_PATH: /home/ab/catkin_ws/devel;/opt/ros/noetic
-- This workspace overlays: /home/ab/catkin_ws/devel;/opt/ros/noetic
-- Found PythonInterp: /usr/bin/python3 (found suitable version "3.8.10", minimu
m required is "3")
-- Using PYTHON_EXECUTABLE: /usr/bin/python3
-- Using Debian Python package layout
-- Using empy: /usr/lib/python3/dist-packages/em.py
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()
-- Using CATKIN_TEST_RESULTS_DIR: /home/ab/catkin_ws/build/test_results
-- Forcing gtest/gmock from source, though one was otherwise available.
-- Found gtest sources under '/usr/src/googletest': gtests will be built
-- Found gmock sources under '/usr/src/googletest': gmock will be built
```

Create Publisher Code C++



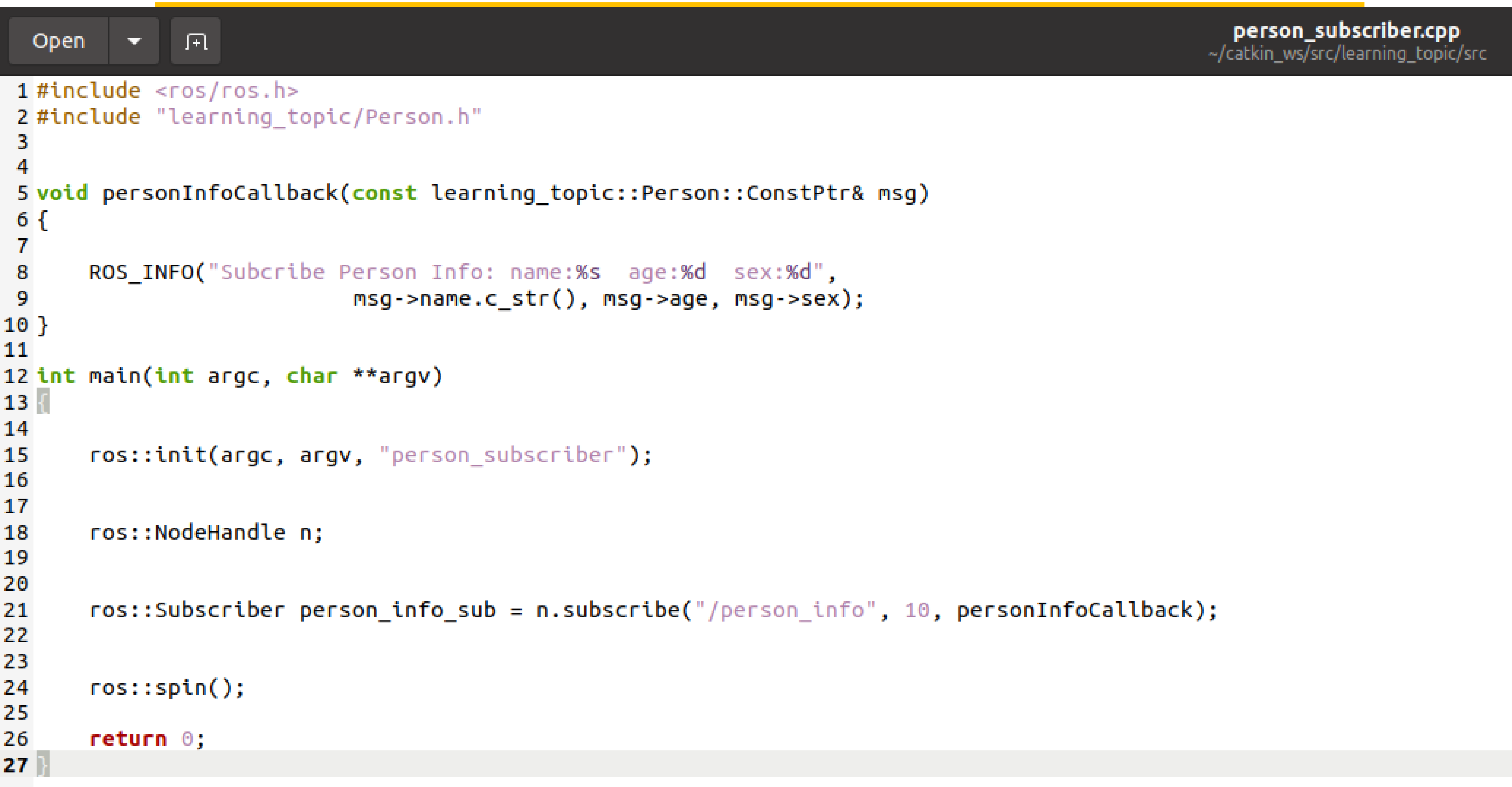
```cpp
#include <ros/ros.h>
#include "learning_topic/Person.h"

int main(int argc, char **argv)
{

    ros::init(argc, argv, "person_publisher");


    ros::NodeHandle n;


    ros::Publisher person_info_pub = n.advertise<learning_topic::Person>("/person_info", 10);


    ros::Rate loop_rate(1);

    int count = 0;
    while (ros::ok())
    {

        learning_topic::Person person_msg;
                person_msg.name = "Tom";
                person_msg.age  = 18;
                person_msg.sex  = learning_topic::Person::male;


                person_info_pub.publish(person_msg);

                ROS_INFO("Publish Person Info: name:%s  age:%d  sex:%d",
                                person_msg.name.c_str(), person_msg.age, person_msg.sex);


        loop_rate.sleep();
    }

    return 0;
}
```

Compiling Code



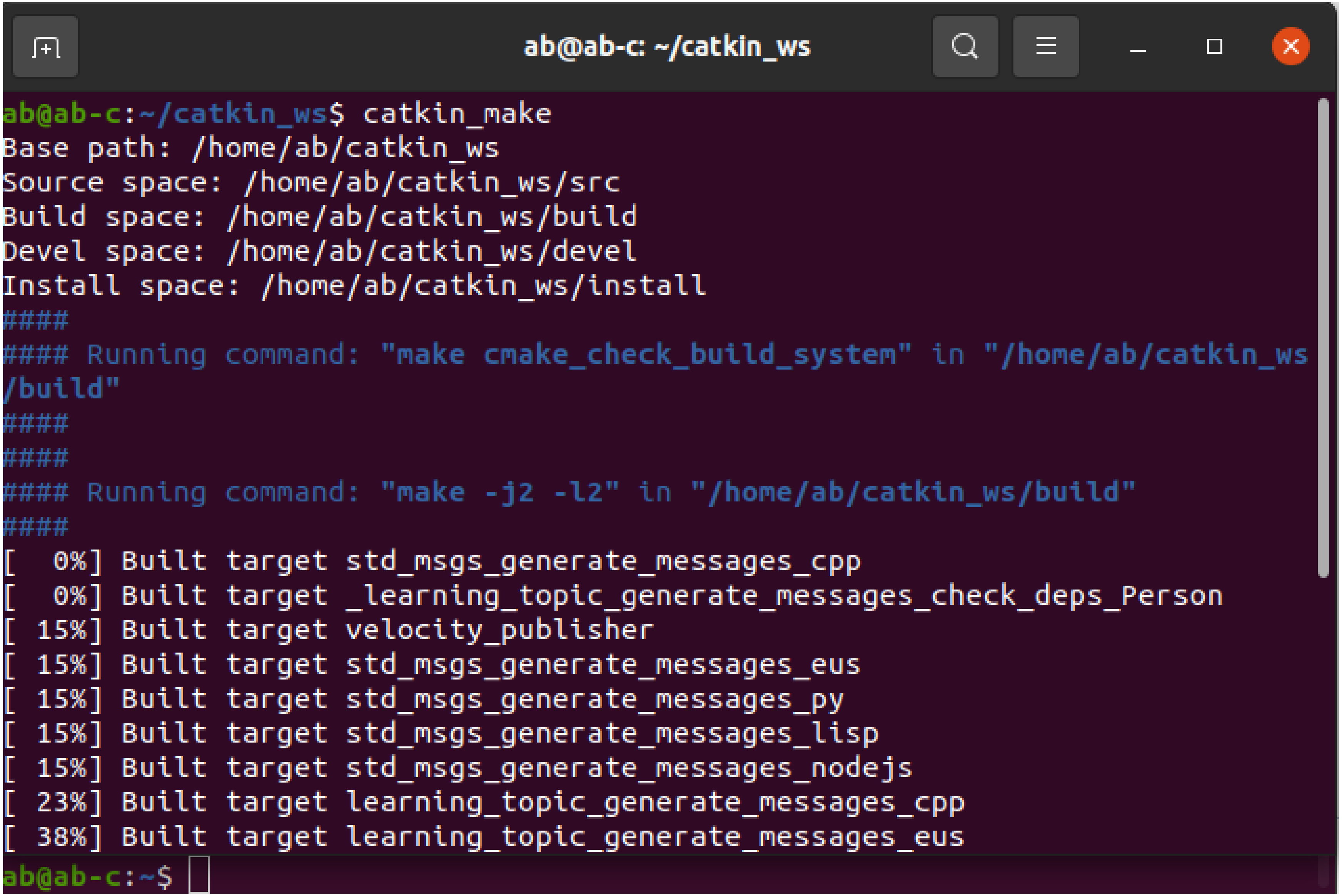
```

add_executable(person_publisher src/person_publisher.cpp)
target_link_libraries(person_publisher ${catkin_LIBRARIES})
add_dependencies(person_publisher ${PROJECT_NAME}_generate_messages_cpp)

add_executable(person_subscriber src/person_subscriber.cpp)
target_link_libraries(person_subscriber ${catkin_LIBRARIES})
add_dependencies(person_subscriber ${PROJECT_NAME}_generate_messages_cpp)

#############
## Install ##
#############
```

Create Subscriber Code C++



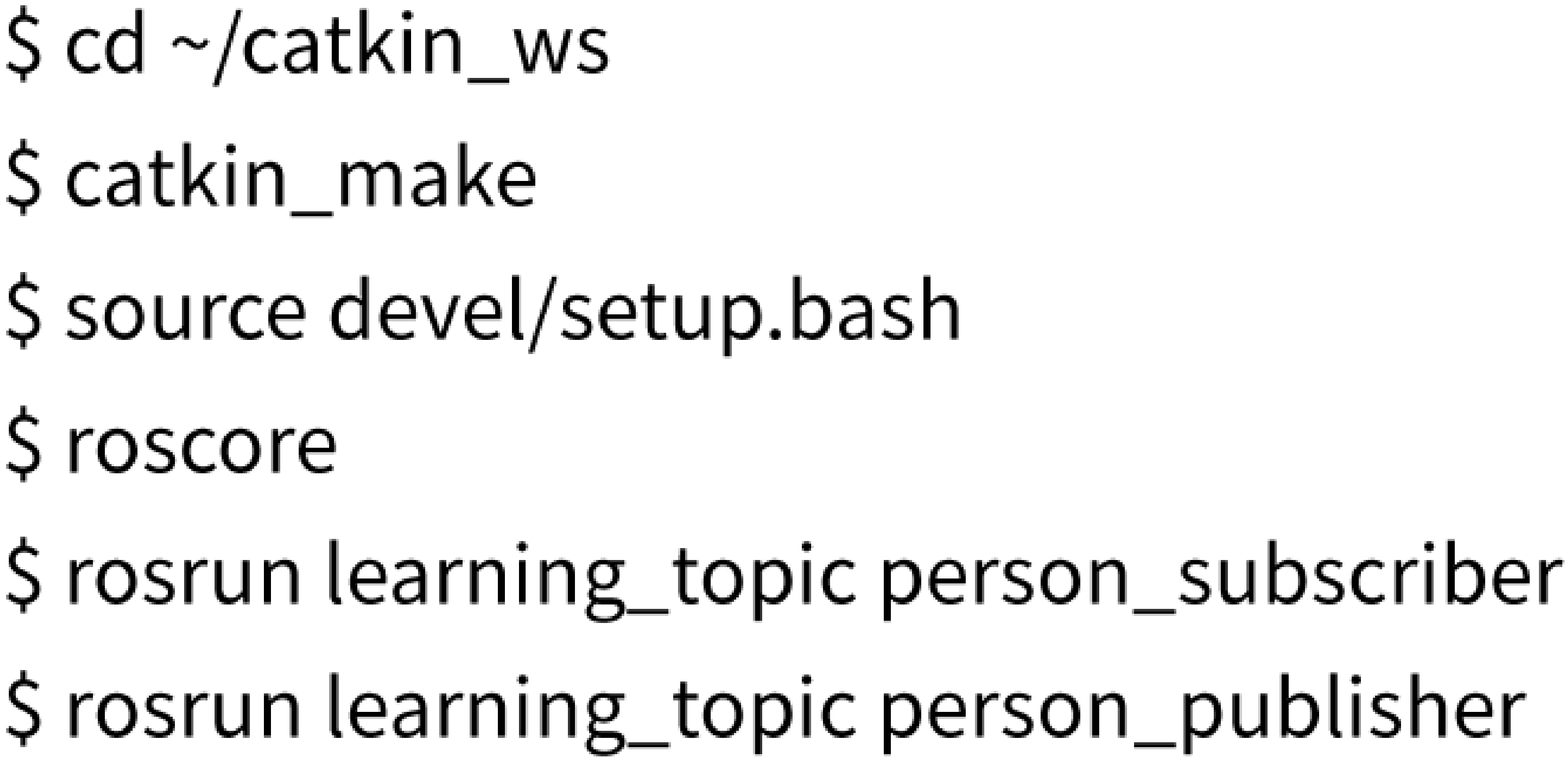
```cpp
#include <ros/ros.h>
#include "learning_topic/Person.h"


void personInfoCallback(const learning_topic::Person::ConstPtr& msg)
{

    ROS_INFO("Subcribe Person Info: name:%s  age:%d  sex:%d",
                        msg->name.c_str(), msg->age, msg->sex);
}

int main(int argc, char **argv)
{

    ros::init(argc, argv, "person_subscriber");


    ros::NodeHandle n;


    ros::Subscriber person_info_sub = n.subscribe("/person_info", 10, personInfoCallback);


    ros::spin();

    return 0;
}
```

catkin_make Compile



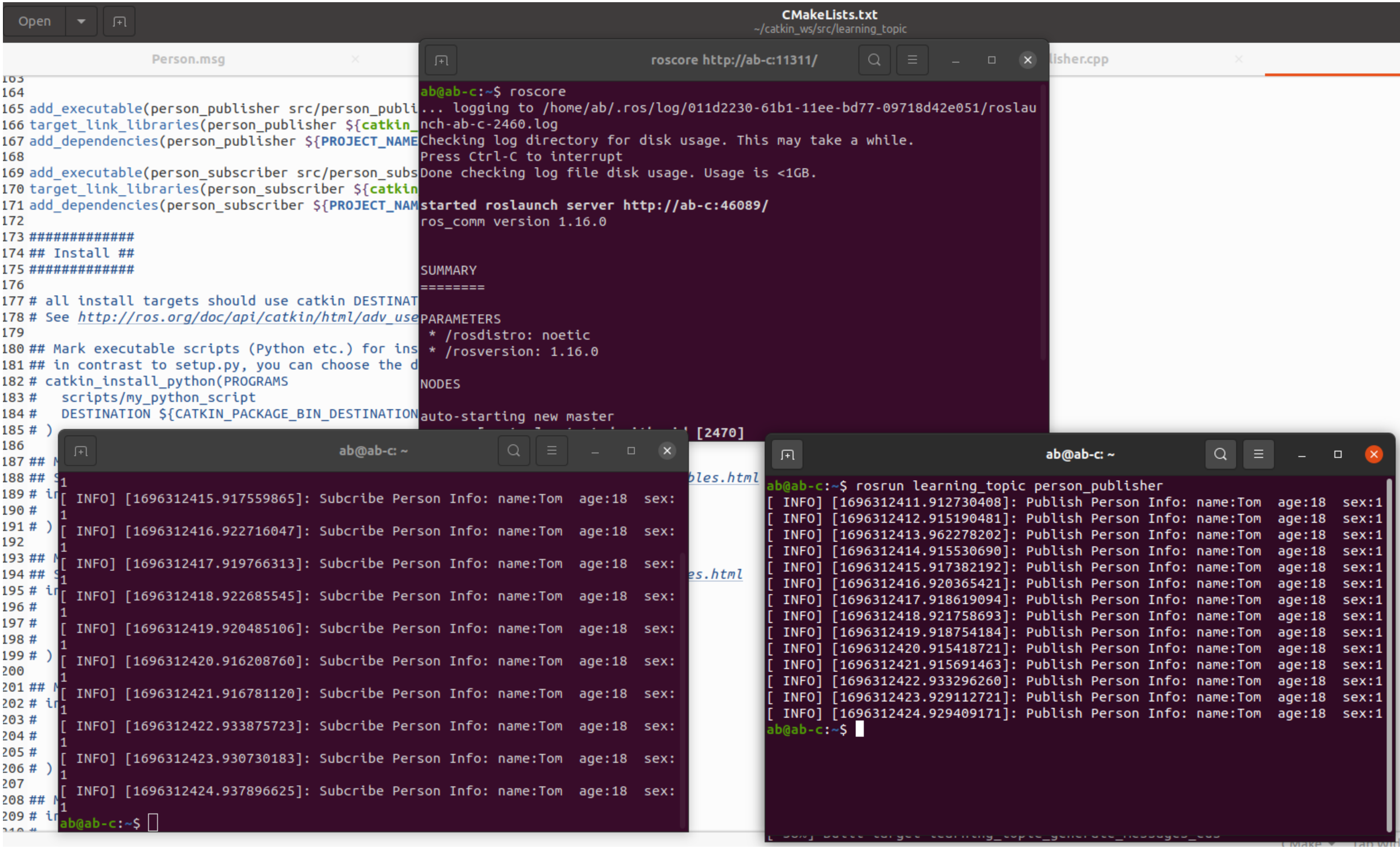
```
ab@ab-c:~/catkin_ws$ catkin_make
Base path: /home/ab/catkin_ws
Source space: /home/ab/catkin_ws/src
Build space: /home/ab/catkin_ws/build
Devel space: /home/ab/catkin_ws/devel
Install space: /home/ab/catkin_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/ab/catkin_ws
/build"
####
####
#### Running command: "make -j2 -l2" in "/home/ab/catkin_ws/build"
####
[  0%] Built target std_msgs_generate_messages_cpp
[  0%] Built target _learning_topic_generate_messages_check_deps_Person
[ 15%] Built target velocity_publisher
[ 15%] Built target std_msgs_generate_messages_eus
[ 15%] Built target std_msgs_generate_messages_py
[ 15%] Built target std_msgs_generate_messages_lisp
[ 15%] Built target std_msgs_generate_messages_nodejs
[ 23%] Built target learning_topic_generate_messages_cpp
[ 38%] Built target learning_topic_generate_messages_eus
ab@ab-c:~$ 
```

Publisher and Subscriber Compile



```
$ cd ~/catkin_ws
$ catkin_make
$ source devel/setup.bash
$ roscore
$ rosrun learning_topic person_subscriber
$ rosrun learning_topic person_publisher
```

Publisher and Subscriber Compile



```
add_executable(person_publisher src/person_publi
target_link_libraries(person_publisher ${catkin_
add_dependencies(person_publisher ${PROJECT_NAME

add_executable(person_subscriber src/person_subs
target_link_libraries(person_subscriber ${catkin
add_dependencies(person_subscriber ${PROJECT_NAM

#############
## Install ##
#############

# all install targets should use catkin DESTINAT
# See http://ros.org/doc/api/catkin/html/adv_use

## Mark executable scripts (Python etc.) for ins
## in contrast to setup.py, you can choose the d
# catkin_install_python(PROGRAMS
#   scripts/my_python_script
#   DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION
# )
```

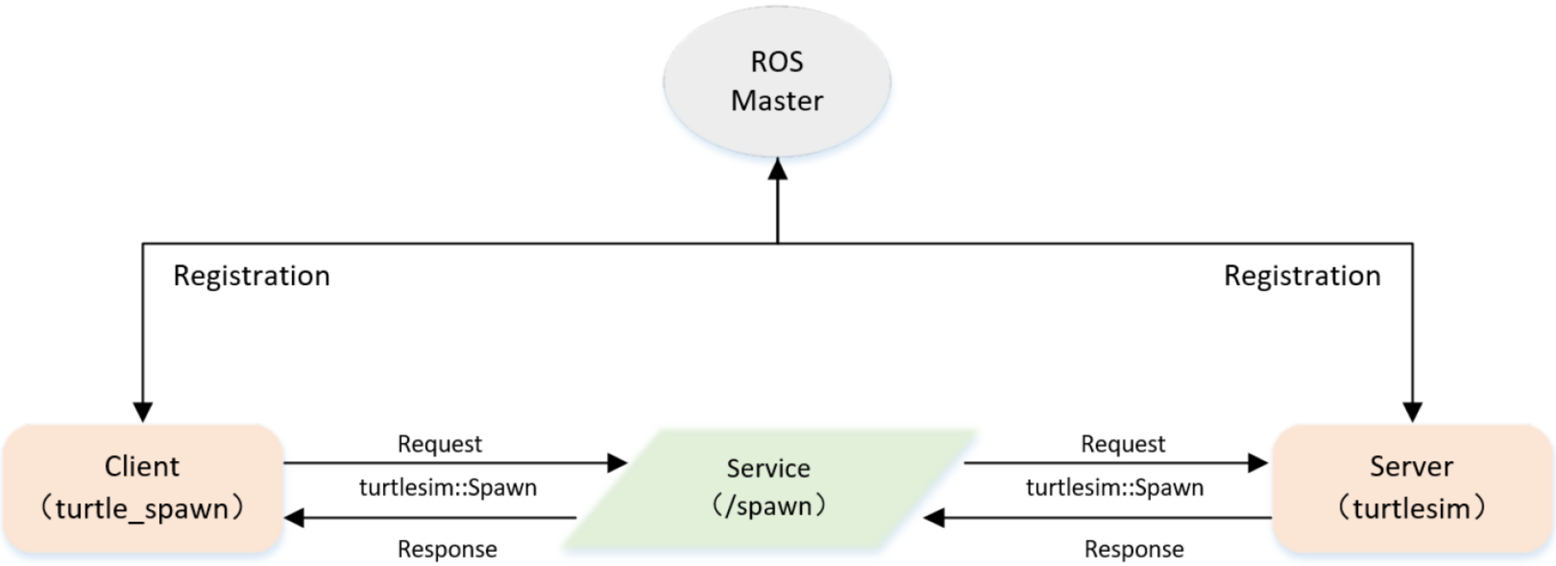
```
ab@ab-c:~$ roscore
... logging to /home/ab/.ros/log/011d2230-61b1-11ee-bd77-09718d42e051/roslau
nch-ab-c-2460.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://ab-c:46089/
ros_comm version 1.16.0


SUMMARY
========

PARAMETERS
 * /rosdistro: noetic
 * /rosversion: 1.16.0

NODES

auto-starting new master
```

```
1
[ INFO] [1696312415.917559865]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312416.922716047]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312417.919766313]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312418.922685545]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312419.920485106]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312420.916208760]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312421.916781120]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312422.933875723]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312423.930730183]: Subcribe Person Info: name:Tom  age:18  sex:
1
[ INFO] [1696312424.937896625]: Subcribe Person Info: name:Tom  age:18  sex:
1
ab@ab-c:~$
```

```
ab@ab-c:~$ rosrun learning_topic person_publisher
[ INFO] [1696312411.912730408]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312412.915190481]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312413.962278202]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312414.915530690]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312415.917382192]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312416.920365421]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312417.918619094]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312418.921758693]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312419.918754184]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312420.915418721]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312421.915691463]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312422.933296260]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312423.929112721]: Publish Person Info: name:Tom  age:18  sex:1
[ INFO] [1696312424.929409171]: Publish Person Info: name:Tom  age:18  sex:1
ab@ab-c:~$
```

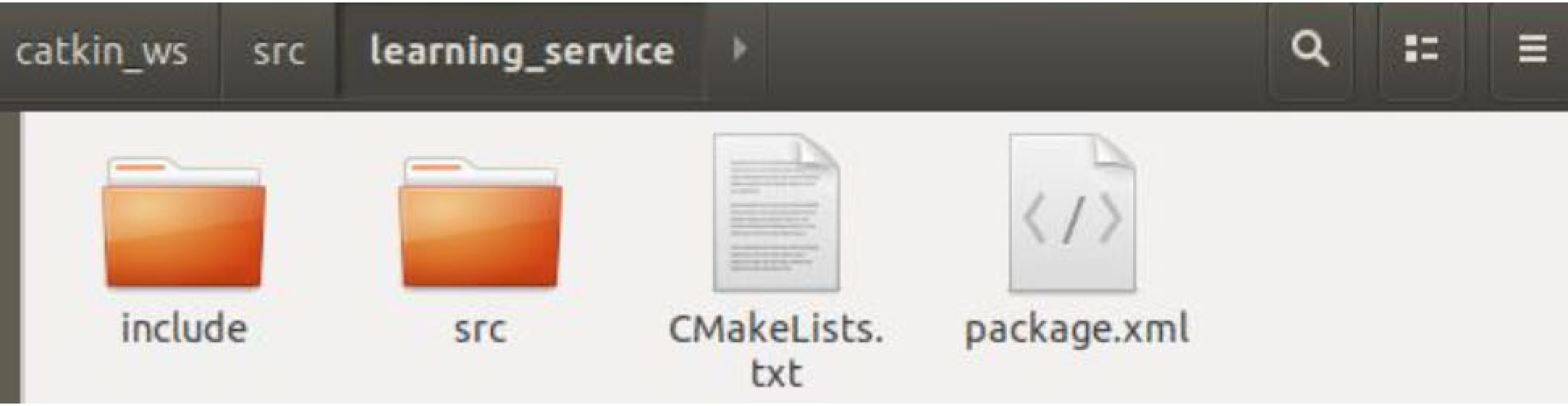
Lecture 8: Client Program

Publisher and Subscriber Compile

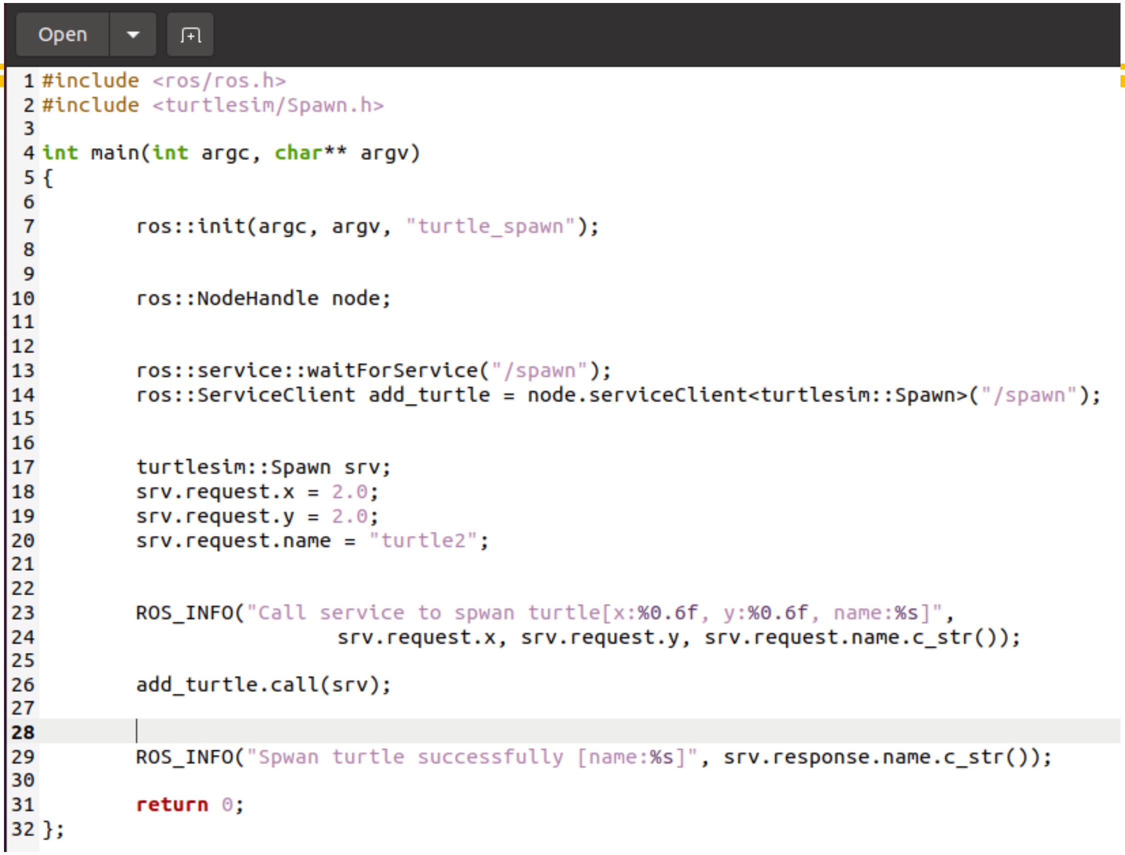


Build Package

```
$ cd ~/catkin_ws/src
$ catkin_create_pkg learning_service roscpp rospy std_msgs geometry_msgs turtlesim
```

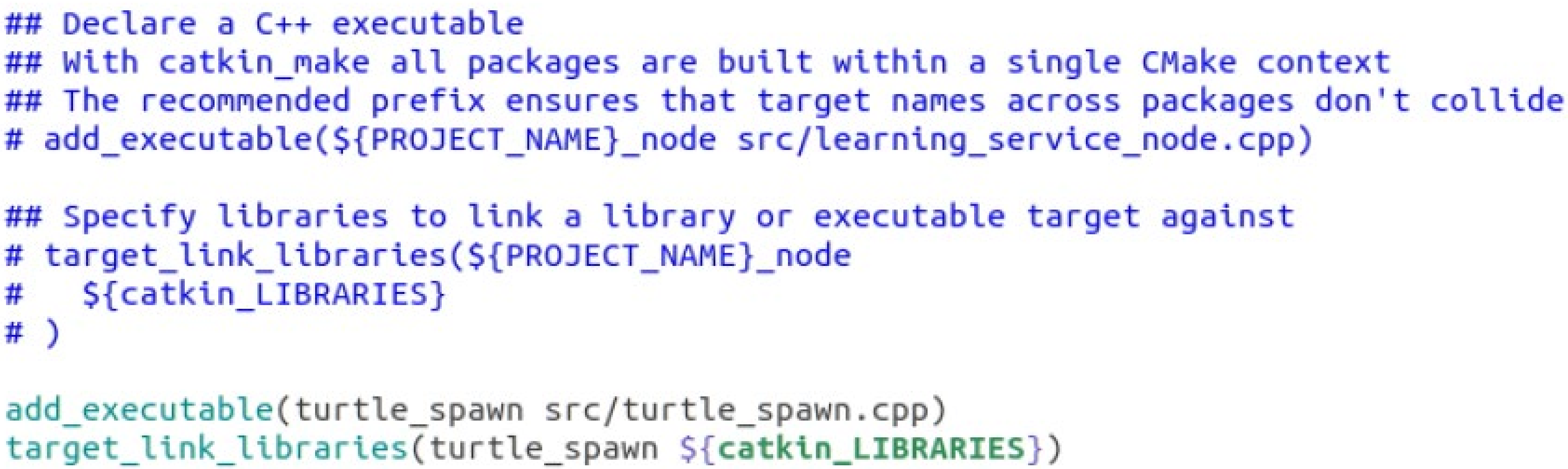


C++ Code



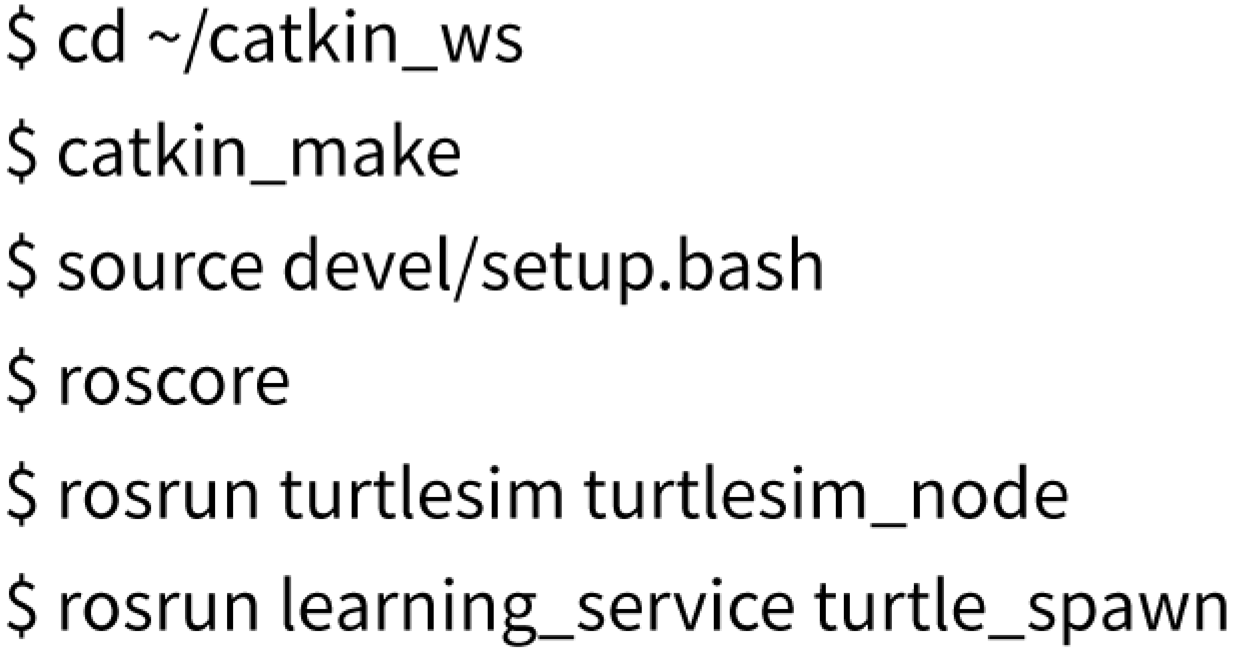
```cpp
#include <ros/ros.h>
#include <turtlesim/Spawn.h>

int main(int argc, char** argv)
{

        ros::init(argc, argv, "turtle_spawn");


        ros::NodeHandle node;


        ros::service::waitForService("/spawn");
        ros::ServiceClient add_turtle = node.serviceClient<turtlesim::Spawn>("/spawn");


        turtlesim::Spawn srv;
        srv.request.x = 2.0;
        srv.request.y = 2.0;
        srv.request.name = "turtle2";


        ROS_INFO("Call service to spwan turtle[x:%0.6f, y:%0.6f, name:%s]",
                         srv.request.x, srv.request.y, srv.request.name.c_str());

        add_turtle.call(srv);


        ROS_INFO("Spwan turtle successfully [name:%s]", srv.response.name.c_str());

        return 0;
};
```

Compiling Code



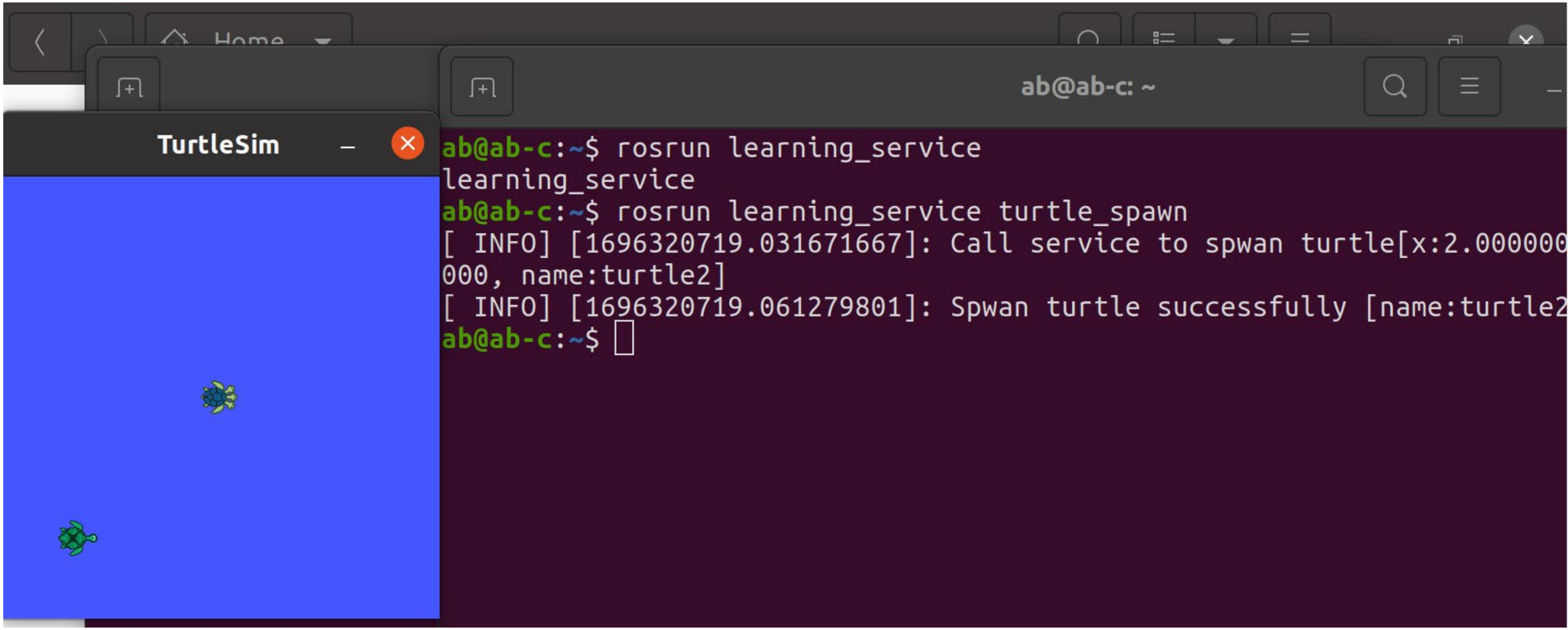
```
## Declare a C++ executable
## With catkin_make all packages are built within a single CMake context
## The recommended prefix ensures that target names across packages don't collide
# add_executable(${PROJECT_NAME}_node src/learning_service_node.cpp)

## Specify libraries to link a library or executable target against
# target_link_libraries(${PROJECT_NAME}_node
#   ${catkin_LIBRARIES}
# )

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

Compiling Code



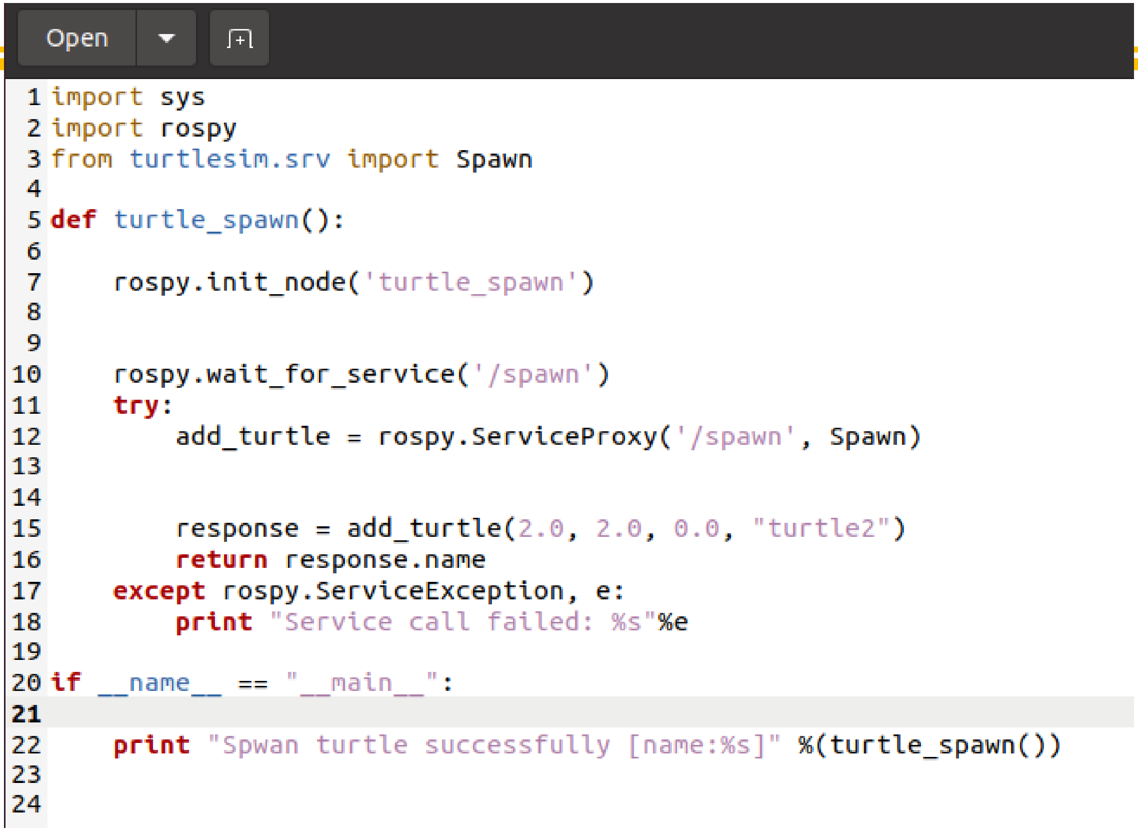
```
$ cd ~/catkin_ws
$ catkin_make
$ source devel/setup.bash
$ roscore
$ rosrun turtlesim turtlesim_node
$ rosrun learning_service turtle_spawn
```

Compiling Code



```
ab@ab-c:~$ rosrun learning_service
learning_service
ab@ab-c:~$ rosrun learning_service turtle_spawn
[ INFO] [1696320719.031671667]: Call service to spwan turtle[x:2.000000
000, name:turtle2]
[ INFO] [1696320719.061279801]: Spwan turtle successfully [name:turtle2
ab@ab-c:~$ 
```

Python Code



```python
import sys
import rospy
from turtlesim.srv import Spawn

def turtle_spawn():

    rospy.init_node('turtle_spawn')


    rospy.wait_for_service('/spawn')
    try:
        add_turtle = rospy.ServiceProxy('/spawn', Spawn)


        response = add_turtle(2.0, 2.0, 0.0, "turtle2")
        return response.name
    except rospy.ServiceException, e:
        print "Service call failed: %s"%e

if __name__ == "__main__":

    print "Spwan turtle successfully [name:%s]" %(turtle_spawn())


```

Reference

✓ Download link

✓ Language:

English, chinese, Japanese, korean

"ROS Robot Programming"

A Handbook is written by TurtleBot3 Developers

Reference

- R. Siegwart, I. R. Nourbakhsh, D. Scaramuzza. Introduction to Autonomous Mobile Robots. MIT Press, 2nd Edition, 2011, ISBN-10: 0262015358.

- Y. Pyo, H. Cho, R. Jung, and T. Lim, ROS Robot Programming, ROBOTIS Co., Ltd., 2017, ISBN 979-11-962307-1-5

- J. O’Kane, A Gentle Introduction to ROS, CreateSpace Independent Publishing Platform, 2013, ISBN-13: 978-1492143239