# Python

$ catkin\_create\_pkg beginner\_tutorials std\_msgs rospy roscpp

$ mkdir scripts

$ cd scripts

chmod +x talker.py

#!/usr/bin/env python

# license removed for brevity

import rospy

from std\_msgs.msg import String

def talker():

pub = rospy.Publisher('chatter', String, queue\_size=10)

rospy.init\_node('talker', anonymous=True)

rate = rospy.Rate(10) # 10hz

while not rospy.is\_shutdown():

hello\_str = "hello world %s" % rospy.get\_time()

rospy.loginfo(hello\_str)

pub.publish(hello\_str)

rate.sleep()

if \_\_name\_\_ == '\_\_main\_\_':

try:

talker()

except rospy.ROSInterruptException:

pass

Add the following to your CMakeLists.txt. This makes sure the python script gets installed properly, and uses the right python interpreter.

catkin\_install\_python(PROGRAMS scripts/talker.py

DESTINATION ${CATKIN\_PACKAGE\_BIN\_DESTINATION}

)

[Toggle line numbers](http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29)

#!/usr/bin/env python

import rospy

from std\_msgs.msg import String

def callback(data):

rospy.loginfo(rospy.get\_caller\_id() + "I heard %s", data.data)

def listener():

# In ROS, nodes are uniquely named. If two nodes with the same

# name are launched, the previous one is kicked off. The

# anonymous=True flag means that rospy will choose a unique

# name for our 'listener' node so that multiple listeners can

# run simultaneously.

rospy.init\_node('listener', anonymous=True)

rospy.Subscriber("chatter", String, callback)

# spin() simply keeps python from exiting until this node is stopped

rospy.spin()

if \_\_name\_\_ == '\_\_main\_\_':

listener()

Then, edit the catkin\_install\_python() call in your CMakeLists.txt so it looks like the following:

catkin\_install\_python(PROGRAMS scripts/talker.py scripts/listener.py

DESTINATION ${CATKIN\_PACKAGE\_BIN\_DESTINATION}

)

# C++

mkdir -p src

[Toggle line numbers](http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29)

#include "ros/ros.h"

#include "std\_msgs/String.h"

#include <sstream>

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

{

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

ros::NodeHandle n;

ros::Publisher chatter\_pub = n.advertise<std\_msgs::String>("chatter", 1000);

ros::Rate loop\_rate(10);

int count = 0;

while (ros::ok())

{

std\_msgs::String msg;

std::stringstream ss;

ss << "hello world " << count;

msg.data = ss.str();

ROS\_INFO("%s", msg.data.c\_str());

chatter\_pub.publish(msg);

ros::spinOnce();

loop\_rate.sleep();

++count;

}

return 0;

}

Subscriber Node

#include "ros/ros.h"

#include "std\_msgs/String.h"

void chatterCallback(const std\_msgs::String::ConstPtr& msg)

{

ROS\_INFO("I heard: [%s]", msg->data.c\_str());

}

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

{

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

ros::NodeHandle n;

ros::Subscriber sub = n.subscribe("chatter", 1000, chatterCallback);

ros::spin();

return 0;

}

add\_executable(talker src/talker.cpp)

target\_link\_libraries(talker ${catkin\_LIBRARIES})

add\_dependencies(talker beginner\_tutorials\_generate\_messages\_cpp)

add\_executable(listener src/listener.cpp)

target\_link\_libraries(listener ${catkin\_LIBRARIES})

add\_dependencies(listener beginner\_tutorials\_generate\_messages\_cpp)