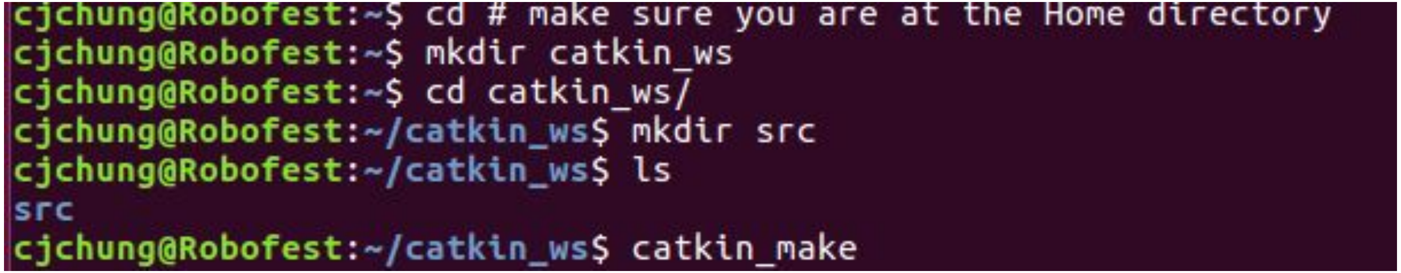
Writing First ROS Programs in Python & C++

IRR S2019

ROS
Robot Operating System

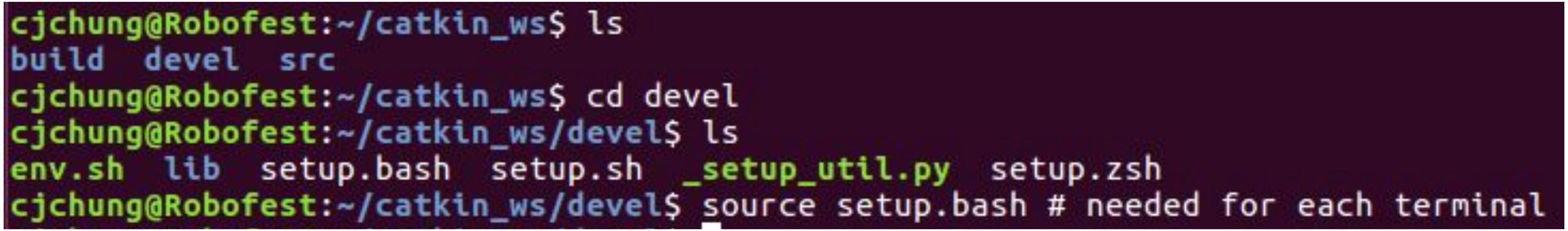
catkin: ROS build system to generate 'targets' from raw source code.
(Old: rosbuild)

Steps to create a ROS Program in Python - 1



```
cjchung@Robofest:~$ cd # make sure you are at the Home directory
cjchung@Robofest:~$ mkdir catkin_ws
cjchung@Robofest:~$ cd catkin_ws/
cjchung@Robofest:~/catkin_ws$ mkdir src
cjchung@Robofest:~/catkin_ws$ ls
src
cjchung@Robofest:~/catkin_ws$ catkin_make
```

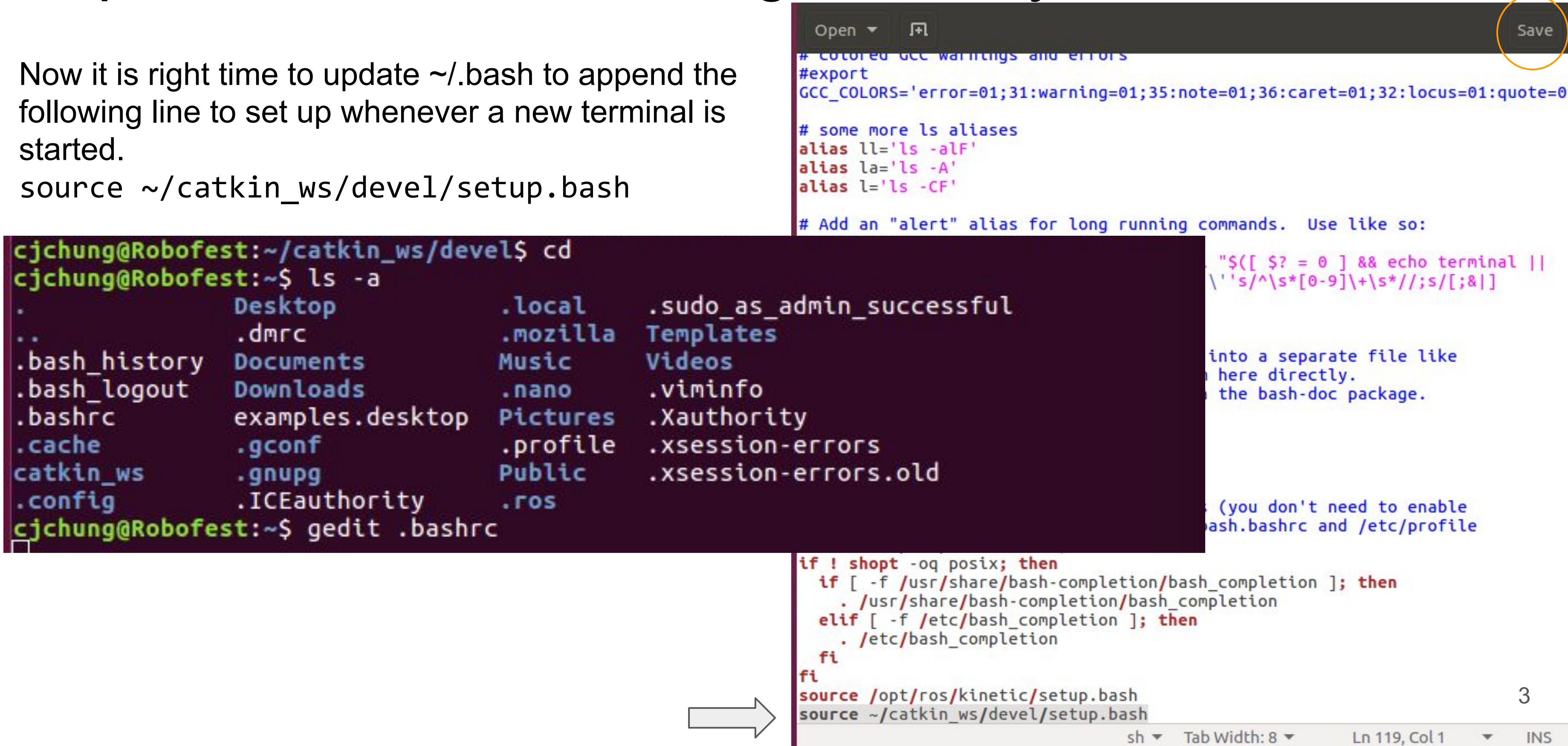
…



```
cjchung@Robofest:~/catkin_ws$ ls
build  devel  src
cjchung@Robofest:~/catkin_ws$ cd devel
cjchung@Robofest:~/catkin_ws/devel$ ls
env.sh  lib  setup.bash  setup.sh  _setup_util.py  setup.zsh
cjchung@Robofest:~/catkin_ws/devel$ source setup.bash # needed for each terminal
```

Steps to create a ROS Program in Python - 2

Now it is right time to update ~/.bash to append the following line to set up whenever a new terminal is started.
`source ~/catkin_ws/devel/setup.bash`



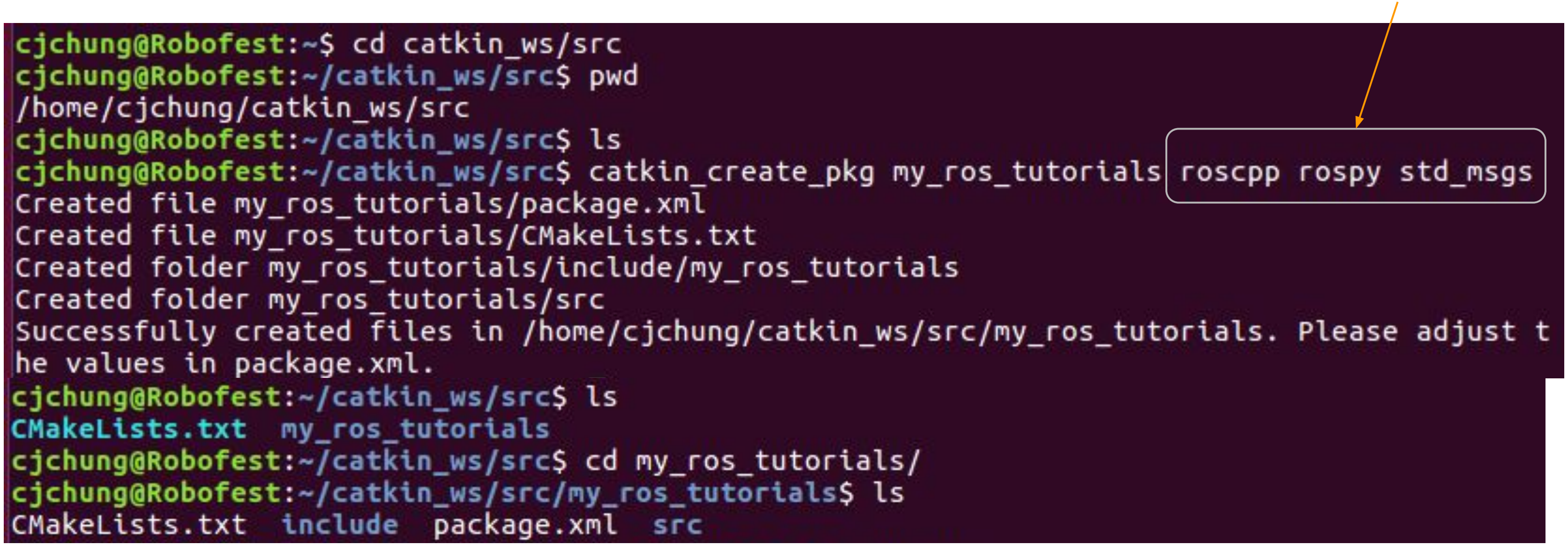
```
cjchung@Robofest:~/catkin_ws/devel$ cd
cjchung@Robofest:~$ ls -a
.               Desktop            .local     .sudo_as_admin_successful
..              .dmrc              .mozilla   Templates
.bash_history   Documents          Music      Videos
.bash_logout    Downloads          .nano      .viminfo
.bashrc         examples.desktop   Pictures   .Xauthority
.cache          .gconf             .profile   .xsession-errors
catkin_ws       .gnupg             Public     .xsession-errors.old
.config         .ICEauthority      .ros
cjchung@Robofest:~$ gedit .bashrc
```

```
# colored GCC warnings and errors
#export
GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=0

# some more ls aliases
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands.  Use like so:

 "$([ $? = 0 ] && echo terminal ||
\''s/^\s*[0-9]\+\s*//;s/[;&|]

into a separate file like
 here directly.
 the bash-doc package.

 (you don't need to enable
ash.bashrc and /etc/profile

if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi
source /opt/ros/kinetic/setup.bash
source ~/catkin_ws/devel/setup.bash
```

Steps to create a ROS Program in Python - 3

Create “`my_ros_tutorials`” **package**

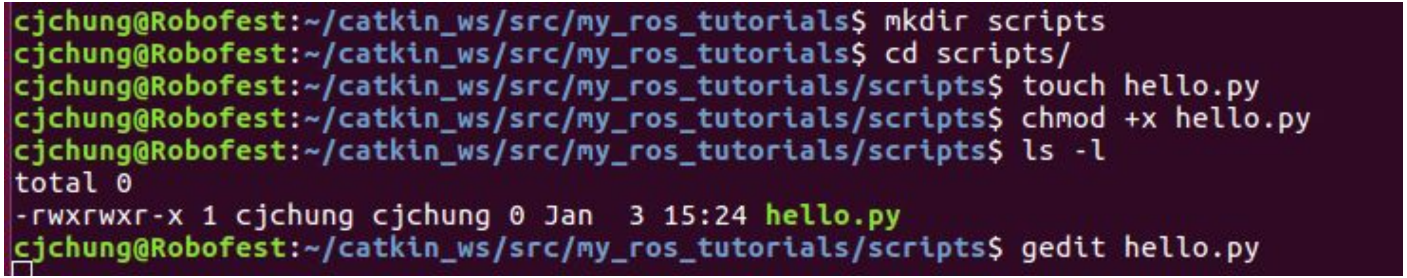
dependencies



```
cjchung@Robofest:~$ cd catkin_ws/src
cjchung@Robofest:~/catkin_ws/src$ pwd
/home/cjchung/catkin_ws/src
cjchung@Robofest:~/catkin_ws/src$ ls
cjchung@Robofest:~/catkin_ws/src$ catkin_create_pkg my_ros_tutorials roscpp rospy std_msgs
Created file my_ros_tutorials/package.xml
Created file my_ros_tutorials/CMakeLists.txt
Created folder my_ros_tutorials/include/my_ros_tutorials
Created folder my_ros_tutorials/src
Successfully created files in /home/cjchung/catkin_ws/src/my_ros_tutorials. Please adjust t
he values in package.xml.
cjchung@Robofest:~/catkin_ws/src$ ls
CMakeLists.txt  my_ros_tutorials
cjchung@Robofest:~/catkin_ws/src$ cd my_ros_tutorials/
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials$ ls
CMakeLists.txt  include  package.xml  src
```

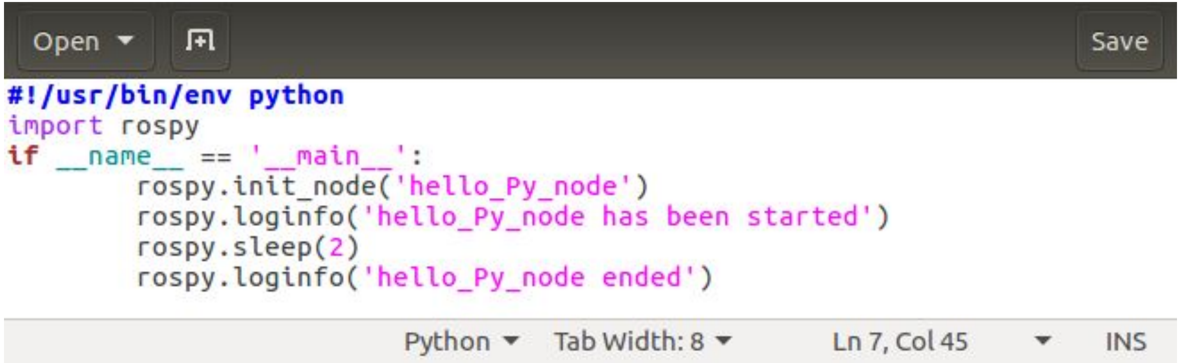
Dependencies: external libraries and tools required outside the package

Steps to create a ROS Program in Python - 4



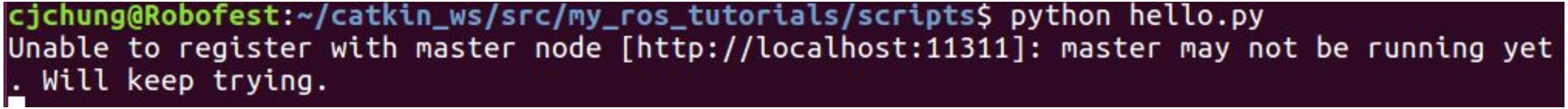
```
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials$ mkdir scripts
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials$ cd scripts/
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$ touch hello.py
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$ chmod +x hello.py
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$ ls -l
total 0
-rwxrwxr-x 1 cjchung cjchung 0 Jan  3 15:24 hello.py
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$ gedit hello.py
```

Save the file!



```python
#!/usr/bin/env python
import rospy
if __name__ == '__main__':
        rospy.init_node('hello_Py_node')
        rospy.loginfo('hello_Py_node has been started')
        rospy.sleep(2)
        rospy.loginfo('hello_Py_node ended')
```

Steps to create a ROS Program in Python - 5

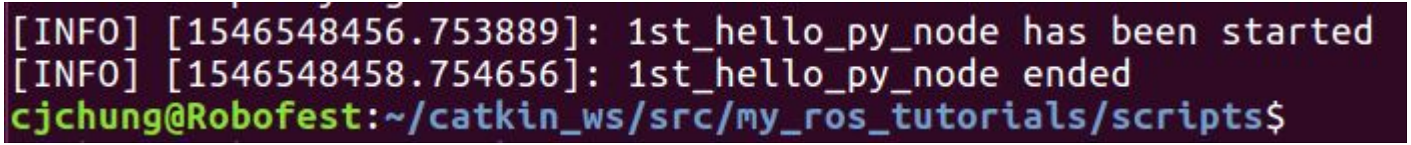


```
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$ python hello.py
Unable to register with master node [http://localhost:11311]: master may not be running yet
. Will keep trying.
```

Why error? No **roscore** is running.

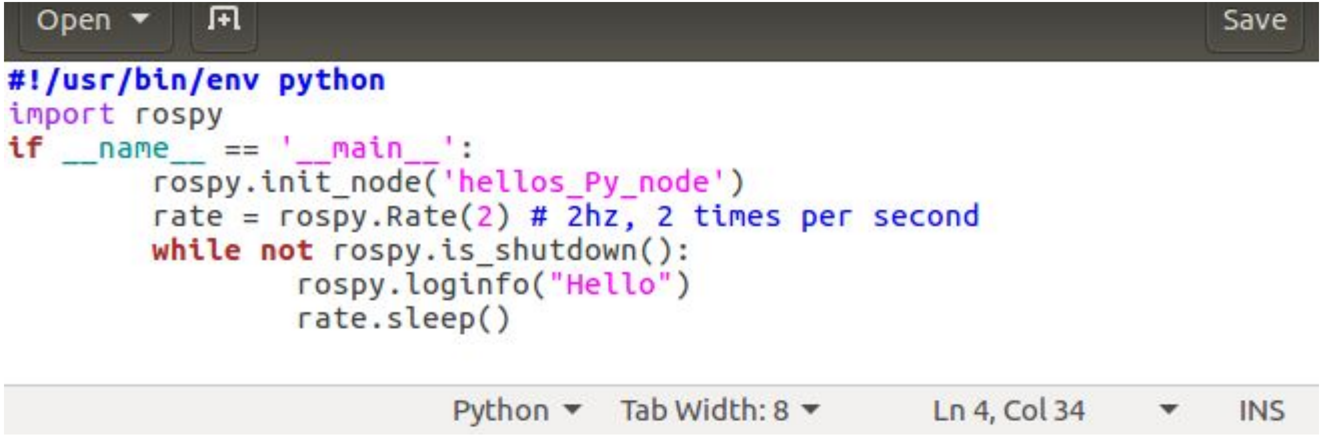
Open another terminal and run `$ roscore`

Then you will see:

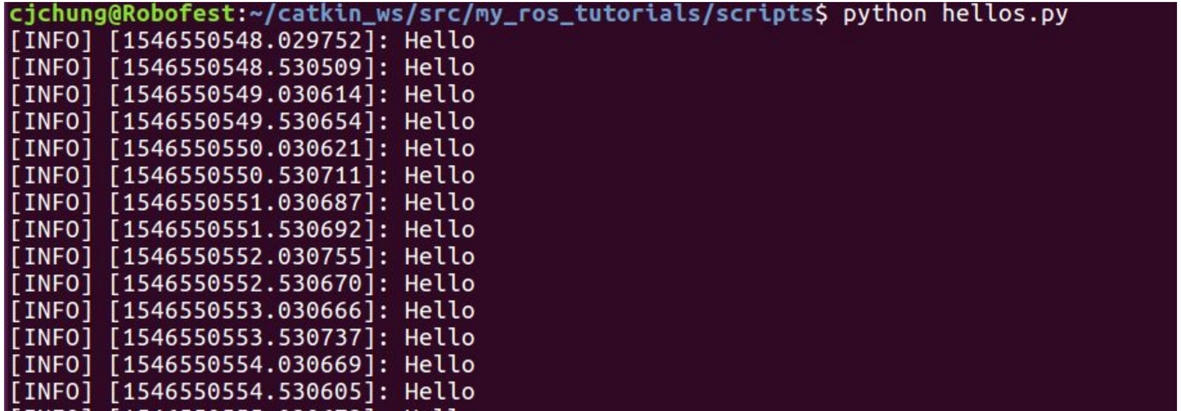


```
[INFO] [1546548456.753889]: 1st_hello_py_node has been started
[INFO] [1546548458.754656]: 1st_hello_py_node ended
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$
```

Another Python Example: hellos.py

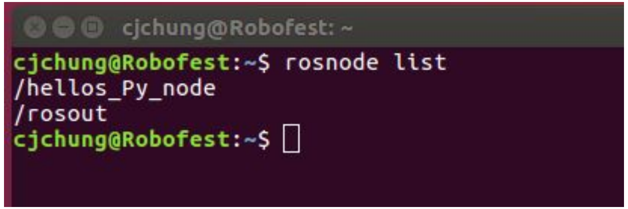


```
#!/usr/bin/env python
import rospy
if __name__ == '__main__':
        rospy.init_node('hellos_Py_node')
        rate = rospy.Rate(2) # 2hz, 2 times per second
        while not rospy.is_shutdown():
                rospy.loginfo("Hello")
                rate.sleep()
```



```
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$ python hellos.py
[INFO] [1546550548.029752]: Hello
[INFO] [1546550548.530509]: Hello
[INFO] [1546550549.030614]: Hello
[INFO] [1546550549.530654]: Hello
[INFO] [1546550550.030621]: Hello
[INFO] [1546550550.530711]: Hello
[INFO] [1546550551.030687]: Hello
[INFO] [1546550551.530692]: Hello
[INFO] [1546550552.030755]: Hello
[INFO] [1546550552.530670]: Hello
[INFO] [1546550553.030666]: Hello
[INFO] [1546550553.530737]: Hello
[INFO] [1546550554.030669]: Hello
[INFO] [1546550554.530605]: Hello
```

To see all the nodes are running:



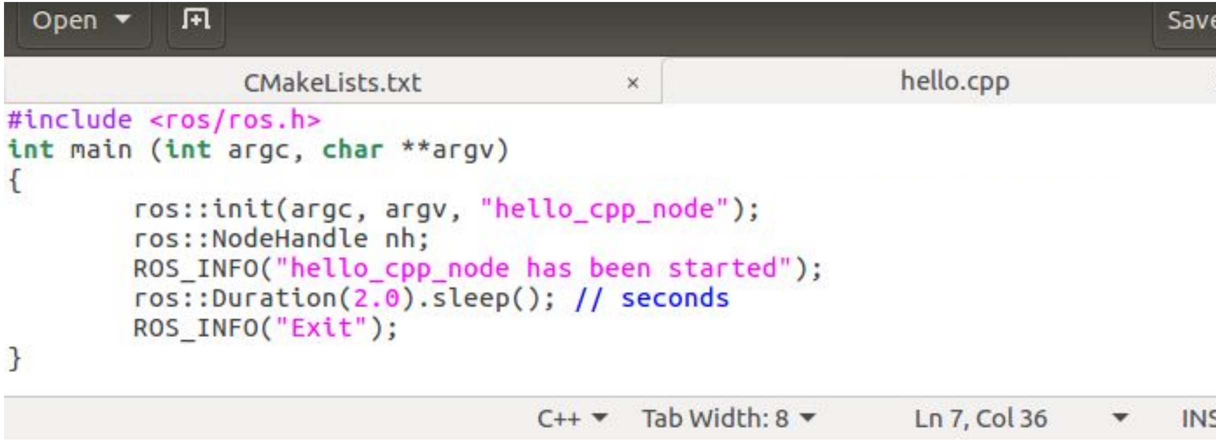
```
cjchung@Robofest:~$ rosnode list
/hellos_Py_node
/rosout
cjchung@Robofest:~$ 
```

To create a ROS Program in C++ - (1 /3)

- Python 1 ~ 3 steps are the same for C++
- Instead of using “script” folder, go to the “**package**”/src folder.

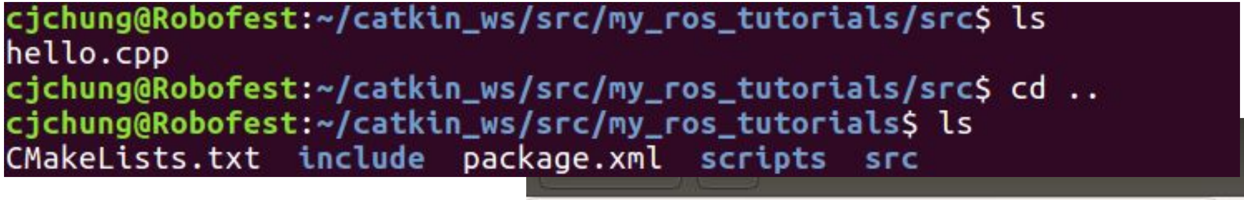
Note: the nodename does not need to match with the nodename in CMakeList.txt
However, must be unique unless anonymous

A NodeHandle is an object which represents your ROS node in roscpp



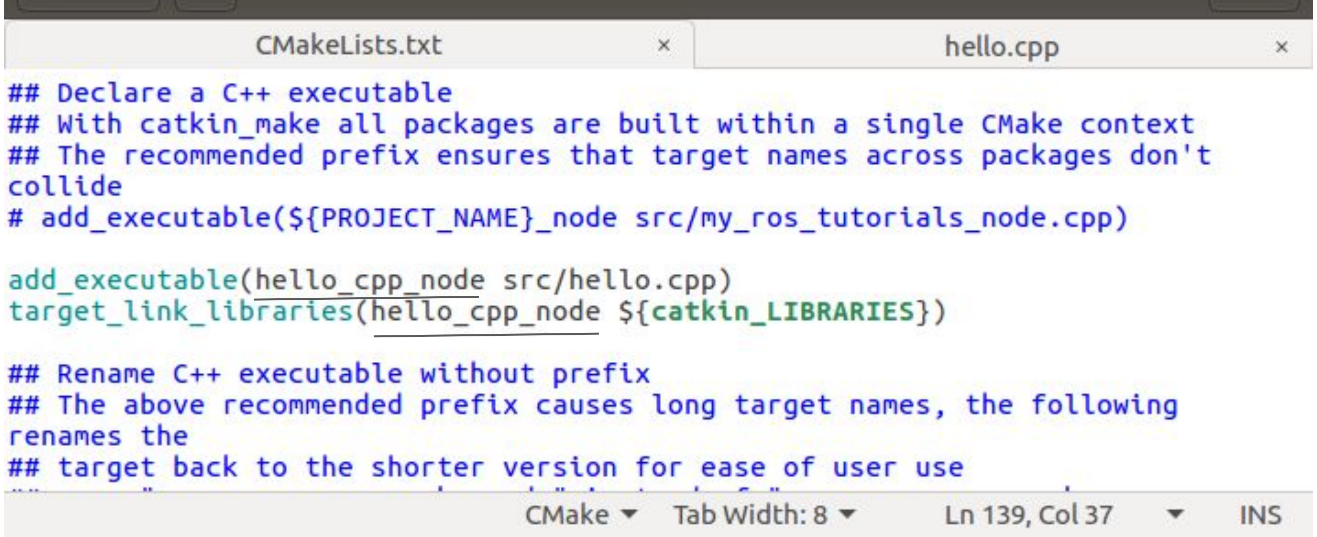
```
#include <ros/ros.h>
int main (int argc, char **argv)
{
        ros::init(argc, argv, "hello_cpp_node");
        ros::NodeHandle nh;
        ROS_INFO("hello_cpp_node has been started");
        ros::Duration(2.0).sleep(); // seconds
        ROS_INFO("Exit");
}
```

To create a ROS Program in C++ - (2 /3)



```
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/src$ ls
hello.cpp
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/src$ cd ..
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials$ ls
CMakeLists.txt  include  package.xml  scripts  src
```

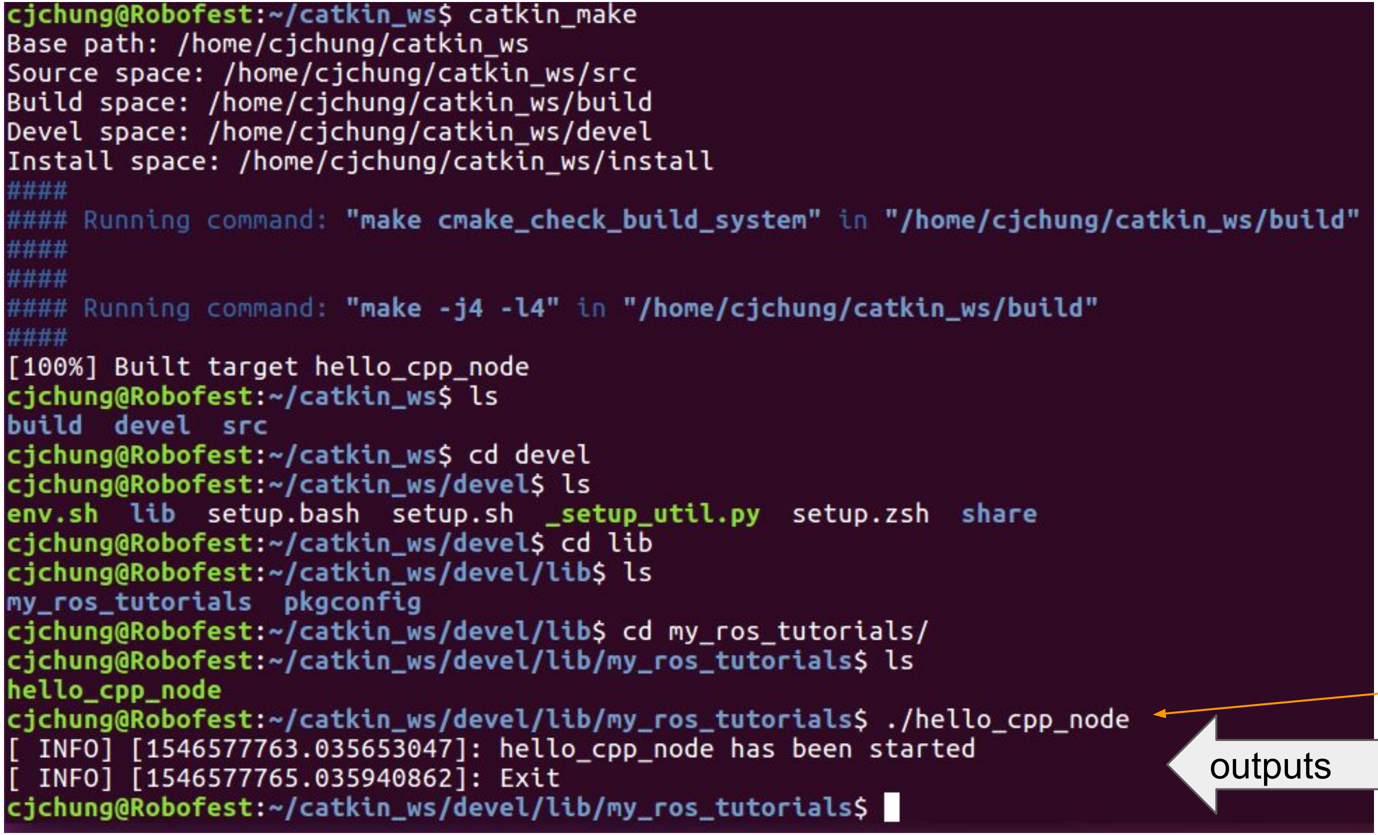
Add the 2 lines in the middle of the `CMakeLists.txt` `file`



```
## 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/my_ros_tutorials_node.cpp)

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

## Rename C++ executable without prefix
## The above recommended prefix causes long target names, the following
renames the
## target back to the shorter version for ease of user use
```

Then go to `~/catkin_ws` and **`$ catkin_make`**

To create a ROS Program in C++ - (3/3)



```
cjchung@Robofest:~/catkin_ws$ catkin_make
Base path: /home/cjchung/catkin_ws
Source space: /home/cjchung/catkin_ws/src
Build space: /home/cjchung/catkin_ws/build
Devel space: /home/cjchung/catkin_ws/devel
Install space: /home/cjchung/catkin_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/cjchung/catkin_ws/build"
####
####
#### Running command: "make -j4 -l4" in "/home/cjchung/catkin_ws/build"
####
[100%] Built target hello_cpp_node
cjchung@Robofest:~/catkin_ws$ ls
build  devel  src
cjchung@Robofest:~/catkin_ws$ cd devel
cjchung@Robofest:~/catkin_ws/devel$ ls
env.sh  lib  setup.bash  setup.sh  _setup_util.py  setup.zsh  share
cjchung@Robofest:~/catkin_ws/devel$ cd lib
cjchung@Robofest:~/catkin_ws/devel/lib$ ls
my_ros_tutorials  pkgconfig
cjchung@Robofest:~/catkin_ws/devel/lib$ cd my_ros_tutorials/
cjchung@Robofest:~/catkin_ws/devel/lib/my_ros_tutorials$ ls
hello_cpp_node
cjchung@Robofest:~/catkin_ws/devel/lib/my_ros_tutorials$ ./hello_cpp_node
[ INFO] [1546577763.035653047]: hello_cpp_node has been started
[ INFO] [1546577765.035940862]: Exit
cjchung@Robofest:~/catkin_ws/devel/lib/my_ros_tutorials$ 
```

outputs

$catkin_make

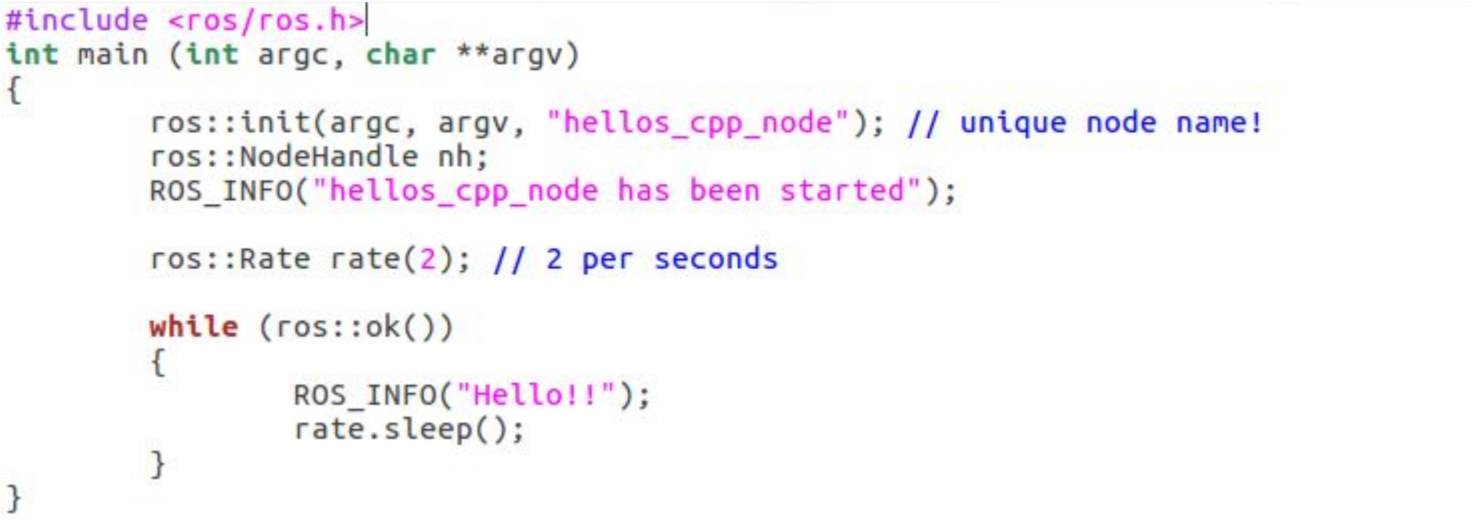
Go to
~/catkin_ws/devel/lib/package

Run executable

Is this a good way to run programs? Better way using "rosrun" later

Another C++ example, hellos.cpp - (1 /3)

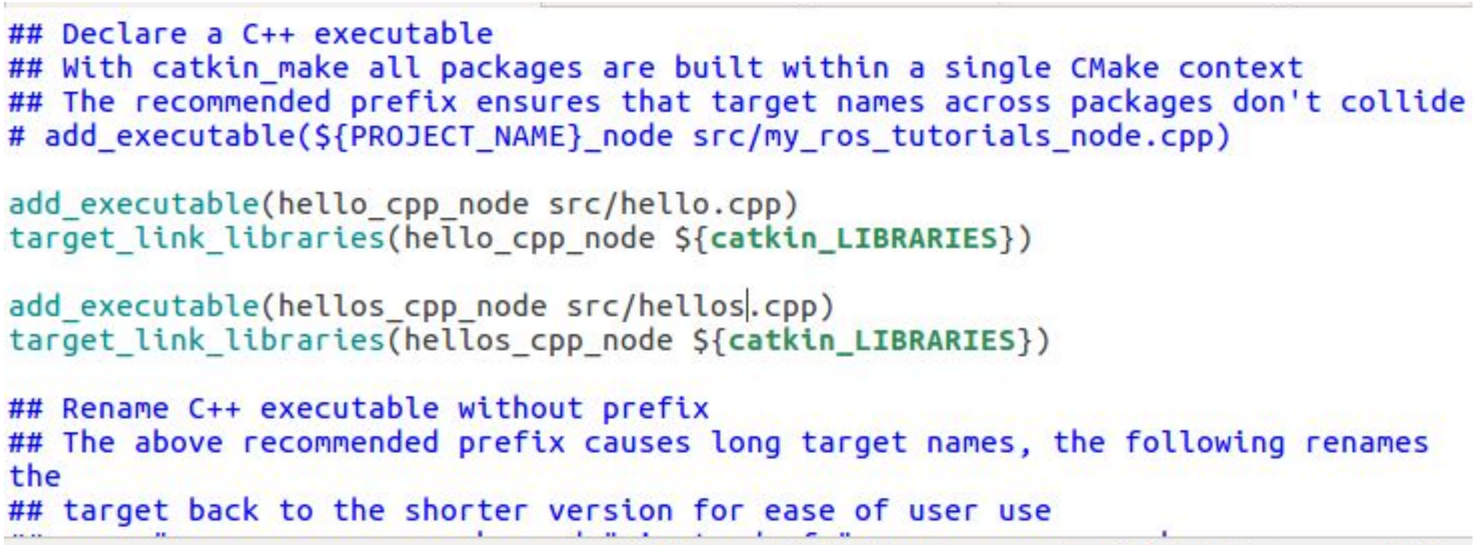
~/catkin_ws/src/my_ros_tutorials/src/hellos.cpp



```
#include <ros/ros.h>
int main (int argc, char **argv)
{
        ros::init(argc, argv, "hellos_cpp_node"); // unique node name!
        ros::NodeHandle nh;
        ROS_INFO("hellos_cpp_node has been started");

        ros::Rate rate(2); // 2 per seconds

        while (ros::ok())
        {
                ROS_INFO("Hello!!");
                rate.sleep();
        }
}
```

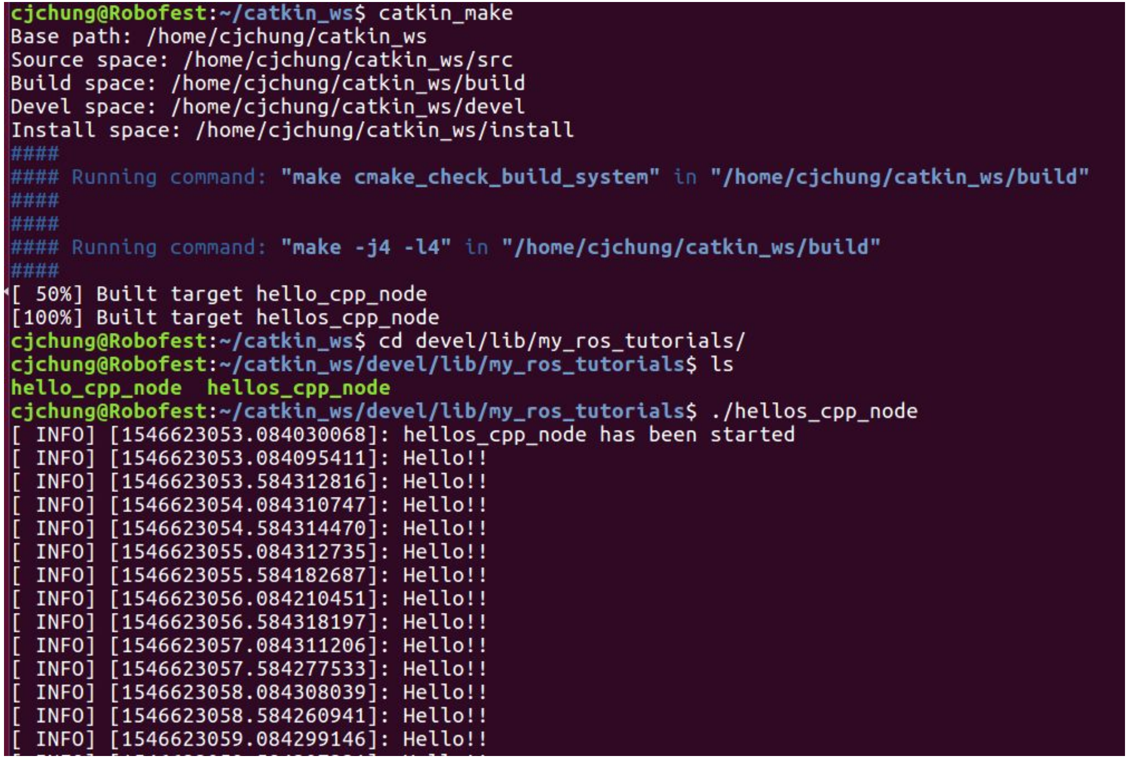
Another C++ example, hellos.cpp - (2 /3)

Update/add ~/catkin_ws/src/my_ros_tutorials/CMakeLists.txt



```
## 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/my_ros_tutorials_node.cpp)

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

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

## Rename C++ executable without prefix
## The above recommended prefix causes long target names, the following renames
the
## target back to the shorter version for ease of user use
```

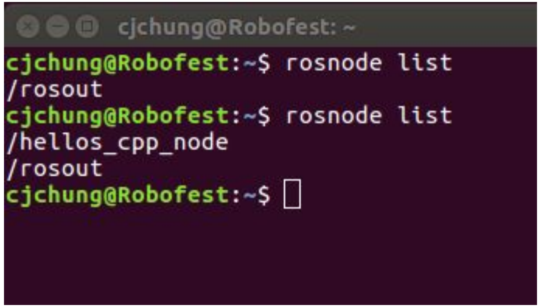
Another C++ example, hellos.cpp - (3 /3)



```
cjchung@Robofest:~/catkin_ws$ catkin_make
Base path: /home/cjchung/catkin_ws
Source space: /home/cjchung/catkin_ws/src
Build space: /home/cjchung/catkin_ws/build
Devel space: /home/cjchung/catkin_ws/devel
Install space: /home/cjchung/catkin_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/cjchung/catkin_ws/build"
####
####
#### Running command: "make -j4 -l4" in "/home/cjchung/catkin_ws/build"
####
[ 50%] Built target hello_cpp_node
[100%] Built target hellos_cpp_node
cjchung@Robofest:~/catkin_ws$ cd devel/lib/my_ros_tutorials/
cjchung@Robofest:~/catkin_ws/devel/lib/my_ros_tutorials$ ls
hello_cpp_node  hellos_cpp_node
cjchung@Robofest:~/catkin_ws/devel/lib/my_ros_tutorials$ ./hellos_cpp_node
[ INFO] [1546623053.084030068]: hellos_cpp_node has been started
[ INFO] [1546623053.084095411]: Hello!!
[ INFO] [1546623053.584312816]: Hello!!
[ INFO] [1546623054.084310747]: Hello!!
[ INFO] [1546623054.584314470]: Hello!!
[ INFO] [1546623055.084312735]: Hello!!
[ INFO] [1546623055.584182687]: Hello!!
[ INFO] [1546623056.084210451]: Hello!!
[ INFO] [1546623056.584318197]: Hello!!
[ INFO] [1546623057.084311206]: Hello!!
[ INFO] [1546623057.584277533]: Hello!!
[ INFO] [1546623058.084308039]: Hello!!
[ INFO] [1546623058.584260941]: Hello!!
[ INFO] [1546623059.084299146]: Hello!!
```

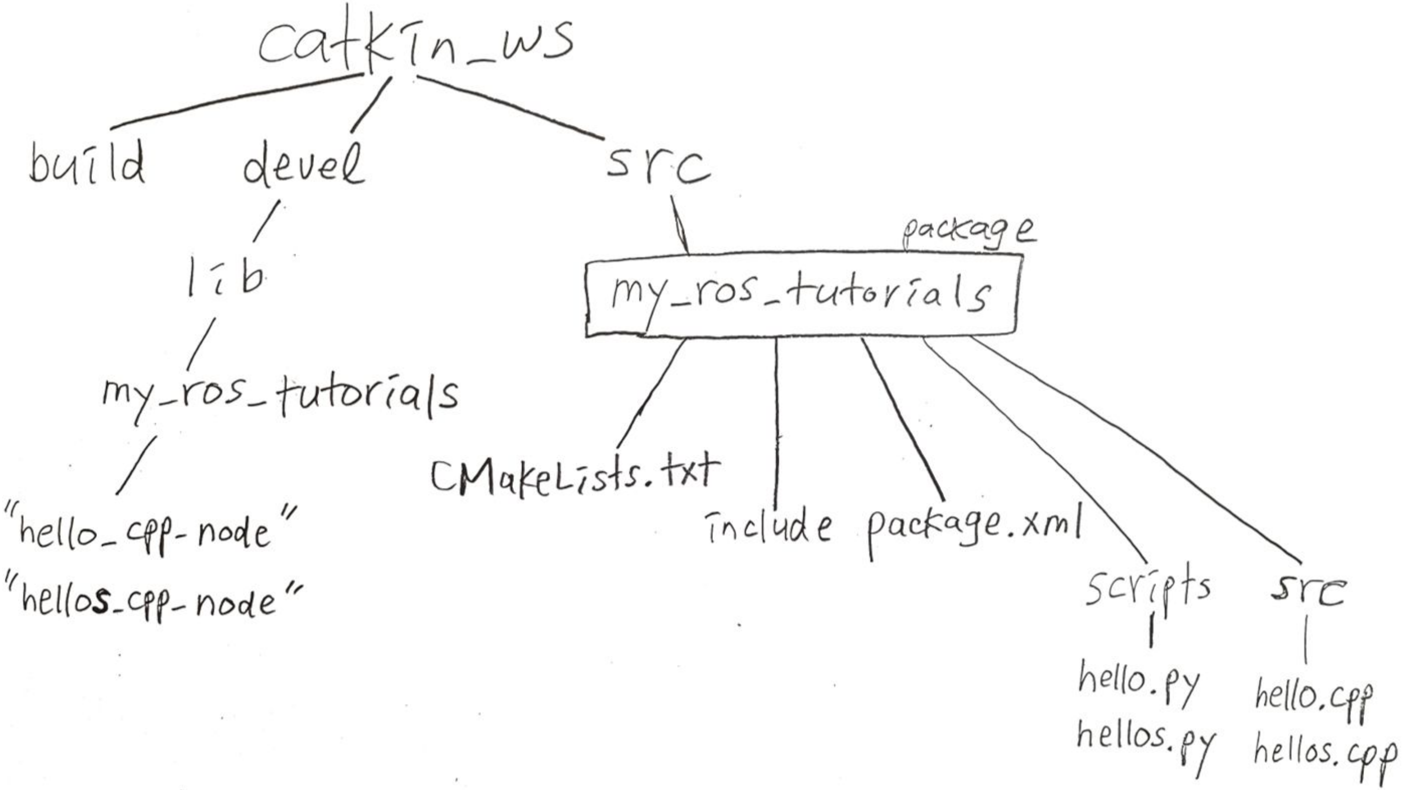
Compile and build

Run! Do not forget roscore first



```
cjchung@Robofest:~$ rosnode list
/rosout
cjchung@Robofest:~$ rosnode list
/hellos_cpp_node
/rosout
cjchung@Robofest:~$
```

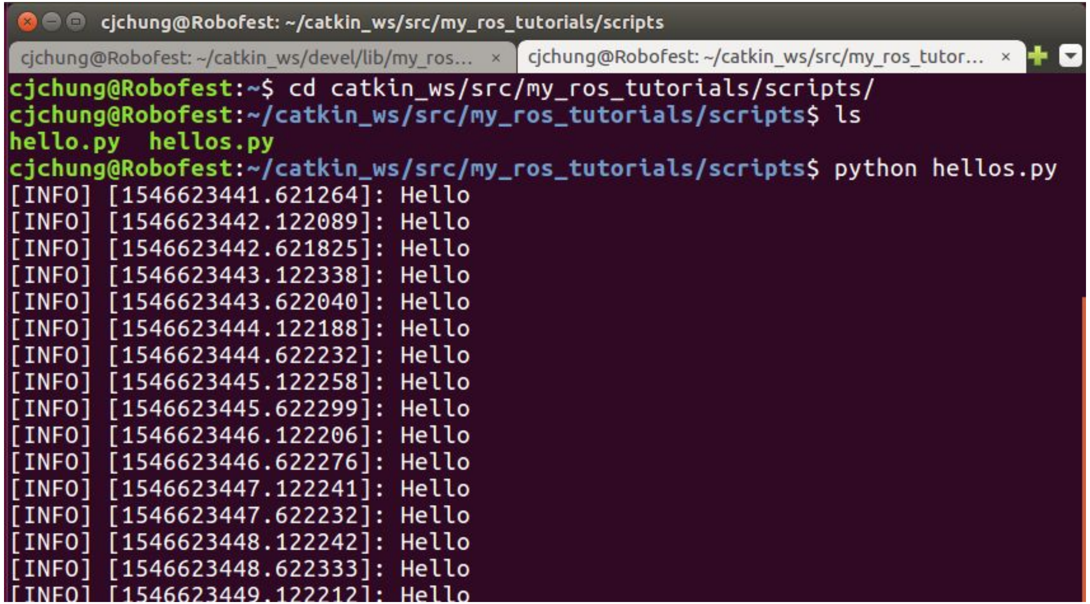
Directory Structure So Far



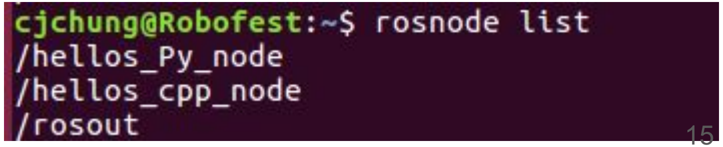
Running Multiple Nodes

^ Shift T to create another terminal inside a terminal

Then run the Python program


cjchung@Robofest: ~/catkin_ws/src/my_ros_tutorials/scripts
cjchung@Robofest: ~/catkin_ws/devel/lib/my_ros...
cjchung@Robofest: ~/catkin_ws/src/my_ros_tutor...

```
cjchung@Robofest:~$ cd catkin_ws/src/my_ros_tutorials/scripts/
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$ ls
hello.py  hellos.py
cjchung@Robofest:~/catkin_ws/src/my_ros_tutorials/scripts$ python hellos.py
[INFO] [1546623441.621264]: Hello
[INFO] [1546623442.122089]: Hello
[INFO] [1546623442.621825]: Hello
[INFO] [1546623443.122338]: Hello
[INFO] [1546623443.622040]: Hello
[INFO] [1546623444.122188]: Hello
[INFO] [1546623444.622232]: Hello
[INFO] [1546623445.122258]: Hello
[INFO] [1546623445.622299]: Hello
[INFO] [1546623446.122206]: Hello
[INFO] [1546623446.622276]: Hello
[INFO] [1546623447.122241]: Hello
[INFO] [1546623447.622232]: Hello
[INFO] [1546623448.122242]: Hello
[INFO] [1546623448.622333]: Hello
[INFO] [1546623449.122212]: Hello
```

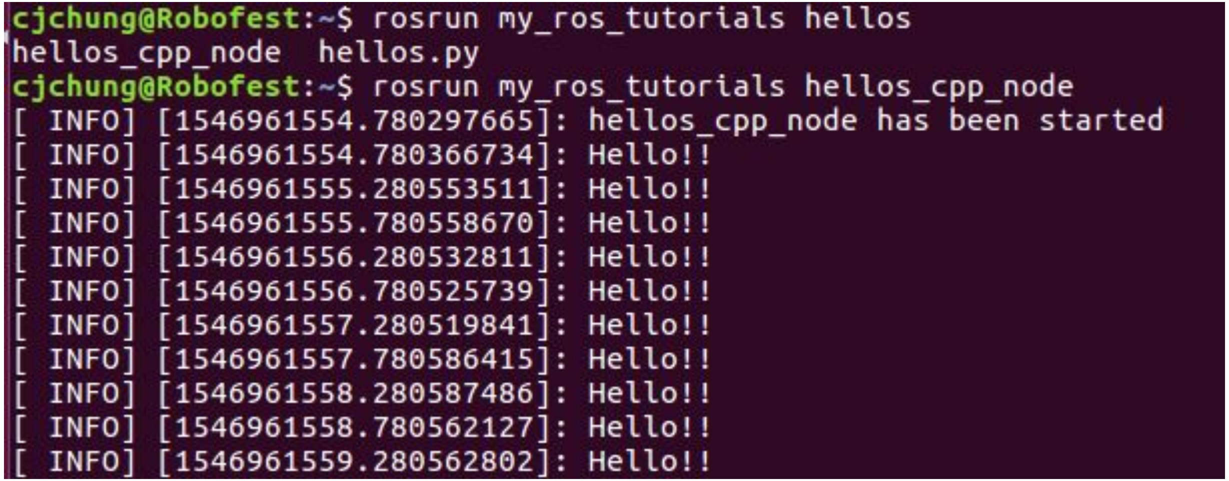


```
cjchung@Robofest:~$ rosnode list
/hellos_Py_node
/hellos_cpp_node
/rosout
```

Run/Debug your nodes with command line tools

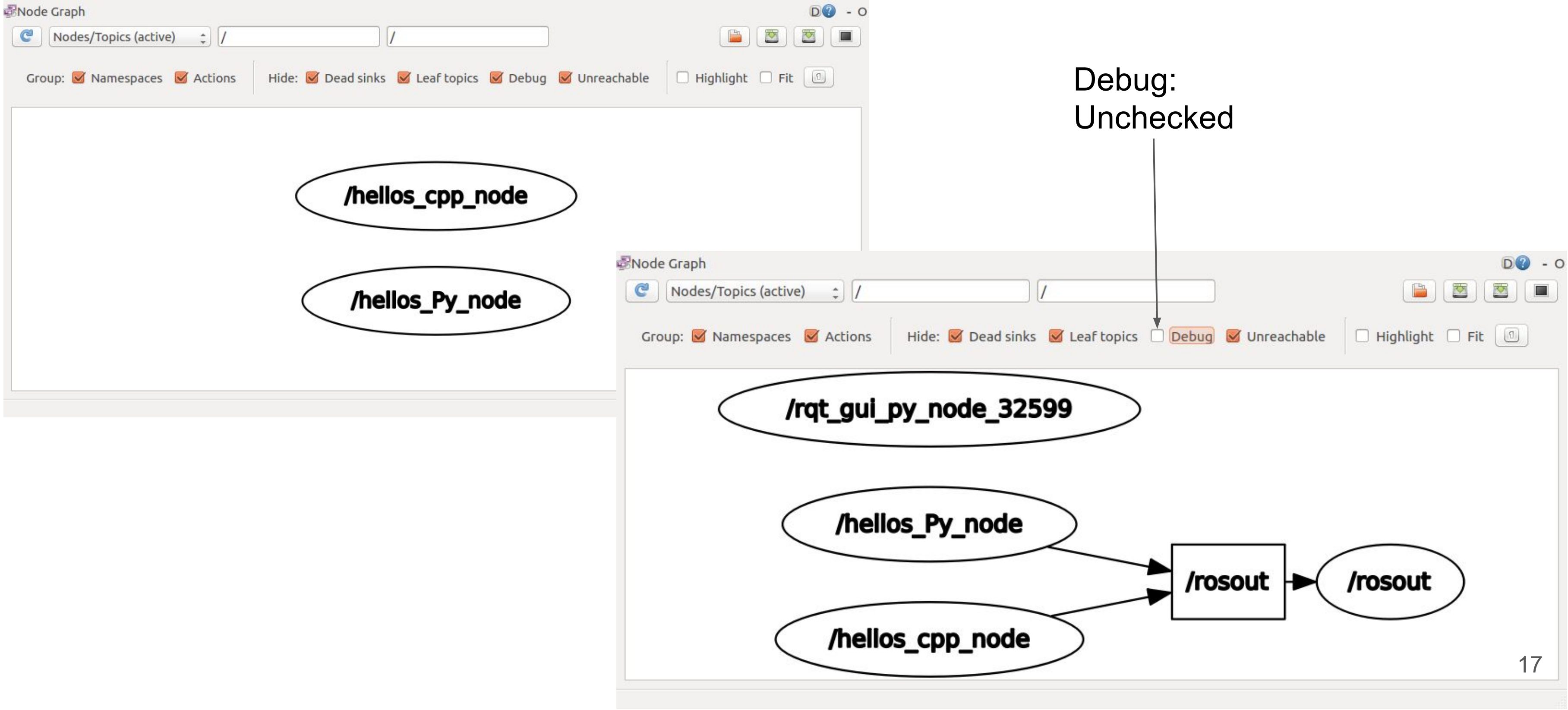
$ rosrun package EXECUTABLE

If you do not remember exe_names in the package, then 2 tabs!!



```
cjchung@Robofest:~$ rosrun my_ros_tutorials hellos
hellos_cpp_node   hellos.py
cjchung@Robofest:~$ rosrun my_ros_tutorials hellos_cpp_node
[ INFO] [1546961554.780297665]: hellos_cpp_node has been started
[ INFO] [1546961554.780366734]: Hello!!
[ INFO] [1546961555.280553511]: Hello!!
[ INFO] [1546961555.780558670]: Hello!!
[ INFO] [1546961556.280532811]: Hello!!
[ INFO] [1546961556.780525739]: Hello!!
[ INFO] [1546961557.280519841]: Hello!!
[ INFO] [1546961557.780586415]: Hello!!
[ INFO] [1546961558.280587486]: Hello!!
[ INFO] [1546961558.780562127]: Hello!!
[ INFO] [1546961559.280562802]: Hello!!
```

$ rqt_graph (when 2 nodes are running)



Node Graph

Nodes/Topics (active) | / | /

Group: ☑ Namespaces ☑ Actions | Hide: ☑ Dead sinks ☑ Leaf topics ☑ Debug ☑ Unreachable | □ Highlight □ Fit

Debug: Unchecked

Node Graph

Nodes/Topics (active) | / | /

Group: ☑ Namespaces ☑ Actions | Hide: ☑ Dead sinks ☑ Leaf topics □ Debug ☑ Unreachable | □ Highlight □ Fit