cmake\_minimum\_required(VERSION 2.8.3)

project(beginner\_tutorials)

## 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

roscpp

rospy

std\_msgs

message\_generation

actionlib\_msgs

actionlib

)

## System dependencies are found with CMake's conventions

# find\_package(Boost REQUIRED COMPONENTS system)

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

# add\_message\_files(

# FILES

# Message1.msg

# Message2.msg

# )

add\_message\_files(

FILES

num.msg

)

## Generate services in the 'srv' folder

# add\_service\_files(

# FILES

# Service1.srv

# Service2.srv

# )

add\_service\_files(

FILES

AddTwoInts.srv

)

## Generate actions in the 'action' folder

# add\_action\_files(

# FILES

# Action1.action

# Action2.action

# )

add\_action\_files(

FILES

Timer.action

DIRECTORY action

)

## Generate added messages and services with any dependencies listed here

# generate\_messages(

# DEPENDENCIES

# std\_msgs

# )

generate\_messages(

DEPENDENCIES

std\_msgs

actionlib\_msgs

)

################################################

## Declare ROS dynamic reconfigure parameters ##

################################################

## To declare and build dynamic reconfigure parameters within this

## package, follow these steps:

## \* In the file package.xml:

## \* add a build\_depend and a exec\_depend tag for "dynamic\_reconfigure"

## \* In this file (CMakeLists.txt):

## \* add "dynamic\_reconfigure" to

## find\_package(catkin REQUIRED COMPONENTS ...)

## \* uncomment the "generate\_dynamic\_reconfigure\_options" section below

## and list every .cfg file to be processed

## Generate dynamic reconfigure parameters in the 'cfg' folder

# generate\_dynamic\_reconfigure\_options(

# cfg/DynReconf1.cfg

# cfg/DynReconf2.cfg

# )

###################################

## catkin specific configuration ##

###################################

## The catkin\_package macro generates cmake config files for your package

## Declare things to be passed to dependent projects

## INCLUDE\_DIRS: uncomment this if your package contains header files

## 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 beginner\_tutorials

# CATKIN\_DEPENDS roscpp rospy std\_msgs

# DEPENDS system\_lib

CATKIN\_DEPENDS roscpp rospy std\_msgs message\_runtime actionlib\_msgs actionlib

LIBRARIES beginner\_tutorials

DEPENDS Boost

)

###########

## Build ##

###########

## Specify additional locations of header files

## Your package locations should be listed before other locations

include\_directories(

# include

${catkin\_INCLUDE\_DIRS}

${Boost\_INCLUDE\_DIRS}

)

## Declare a C++ library

# add\_library(${PROJECT\_NAME}

# src/${PROJECT\_NAME}/beginner\_tutorials.cpp

# )

## Add cmake target dependencies of the library

## as an example, code may need to be generated before libraries

## either from message generation or dynamic reconfigure

# add\_dependencies(${PROJECT\_NAME} ${${PROJECT\_NAME}\_EXPORTED\_TARGETS} ${catkin\_EXPORTED\_TARGETS})

## 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/beginner\_tutorials\_node.cpp)

## 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

## e.g. "rosrun someones\_pkg node" instead of "rosrun someones\_pkg someones\_pkg\_node"

# set\_target\_properties(${PROJECT\_NAME}\_node PROPERTIES OUTPUT\_NAME node PREFIX "")

## Add cmake target dependencies of the executable

## same as for the library above

# add\_dependencies(${PROJECT\_NAME}\_node ${${PROJECT\_NAME}\_EXPORTED\_TARGETS} ${catkin\_EXPORTED\_TARGETS})

## Specify libraries to link a library or executable target against

# target\_link\_libraries(${PROJECT\_NAME}\_node

# ${catkin\_LIBRARIES}

# )

#############

## Install ##

#############

# all install targets should use catkin DESTINATION variables

# See http://ros.org/doc/api/catkin/html/adv\_user\_guide/variables.html

## Mark executable scripts (Python etc.) for installation

## in contrast to setup.py, you can choose the destination

# install(PROGRAMS

# scripts/my\_python\_script

# DESTINATION ${CATKIN\_PACKAGE\_BIN\_DESTINATION}

# )

## Mark executables and/or libraries for installation

# install(TARGETS ${PROJECT\_NAME} ${PROJECT\_NAME}\_node

# ARCHIVE DESTINATION ${CATKIN\_PACKAGE\_LIB\_DESTINATION}

# LIBRARY DESTINATION ${CATKIN\_PACKAGE\_LIB\_DESTINATION}

# RUNTIME DESTINATION ${CATKIN\_PACKAGE\_BIN\_DESTINATION}

# )

## Mark cpp header files for installation

# install(DIRECTORY include/${PROJECT\_NAME}/

# DESTINATION ${CATKIN\_PACKAGE\_INCLUDE\_DESTINATION}

# FILES\_MATCHING PATTERN "\*.h"

# PATTERN ".svn" EXCLUDE

# )

## Mark other files for installation (e.g. launch and bag files, etc.)

# install(FILES

# # myfile1

# # myfile2

# DESTINATION ${CATKIN\_PACKAGE\_SHARE\_DESTINATION}

# )

#############

## Testing ##

#############

## Add gtest based cpp test target and link libraries

# catkin\_add\_gtest(${PROJECT\_NAME}-test test/test\_beginner\_tutorials.cpp)

# if(TARGET ${PROJECT\_NAME}-test)

# target\_link\_libraries(${PROJECT\_NAME}-test ${PROJECT\_NAME})

# endif()

## Add folders to be run by python nosetests

# catkin\_add\_nosetests(test)