ROS tutorial

Install & tutorial

ISL

안재원

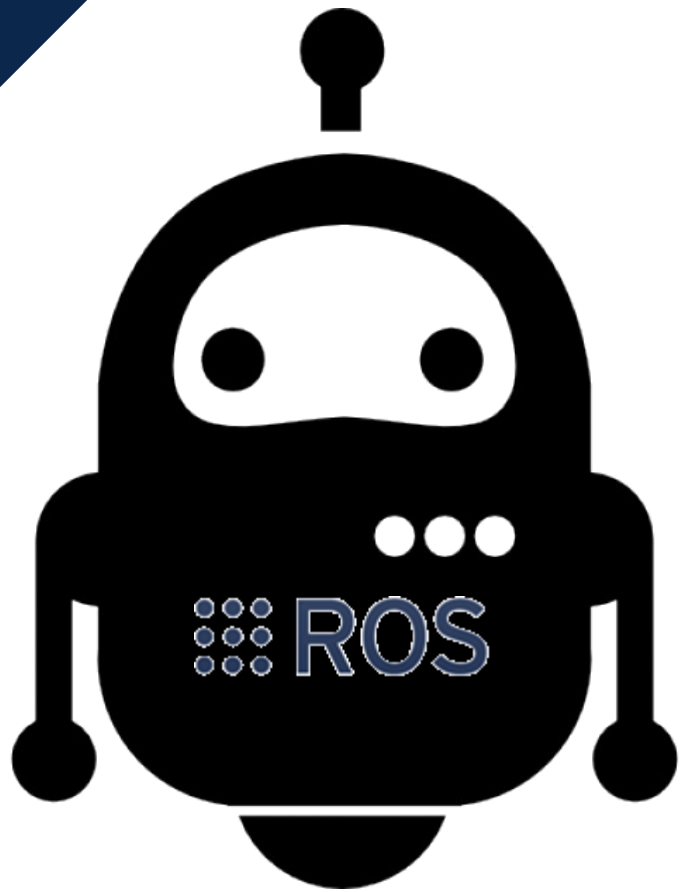
CONTENTS

- ROS?
- Install
- Tutorial

01

ROS?

-Robot Operating System



- 운영체제?

ROS

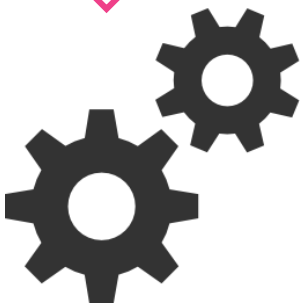
Sensor

Driving part

Etc..

- 하드웨어 추상화
- 디바이스 제어
- 패키지 관리
- 메시지 관리

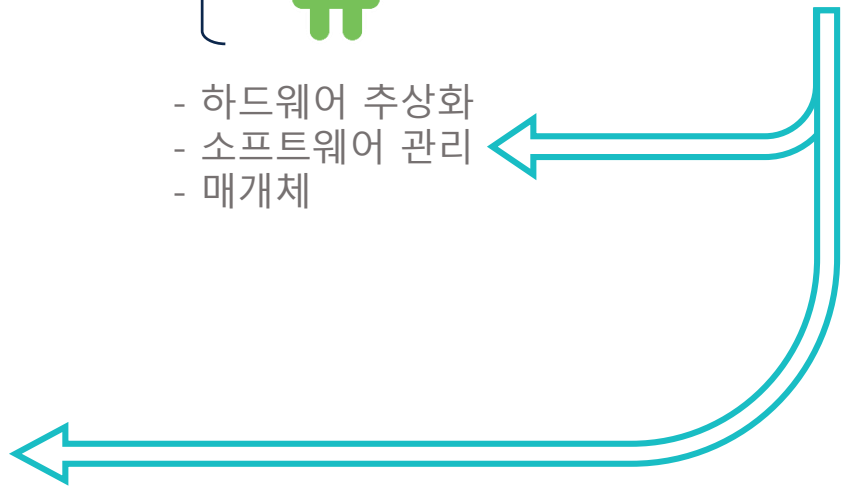
※통신기반 프로그램



- System
- Application

User

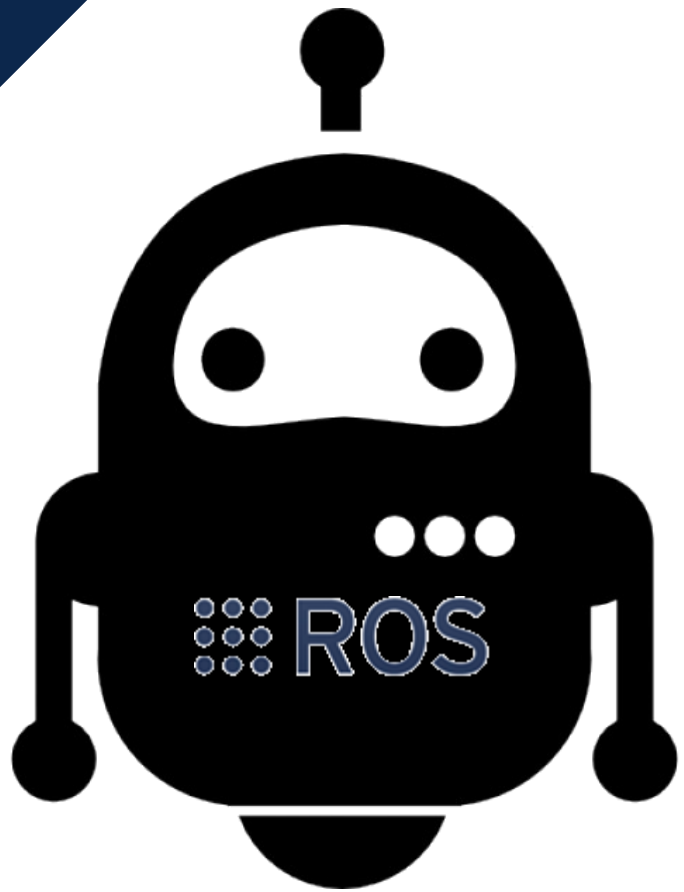
- 하드웨어 추상화
- 소프트웨어 관리
- 매개체



01

ROS?

-Robot Operating System



- 운영체제?

ROS

Sensor

Driving part

Etc..

User

- 하드웨어 추상화
- 소프트웨어 관리
- 매개체

ROS

Sensor System

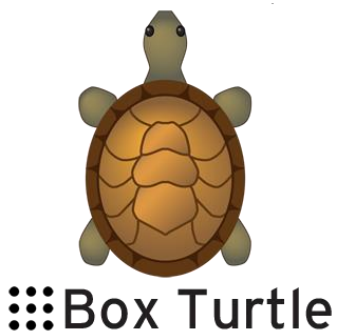
ROS

Driving System

01

ROS?

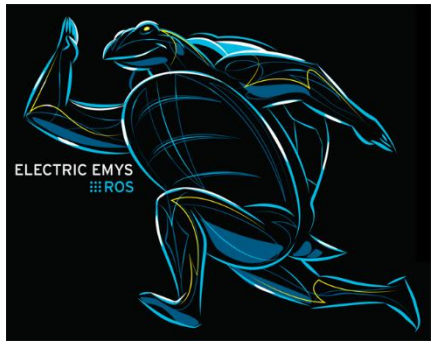
-Robot Operating System



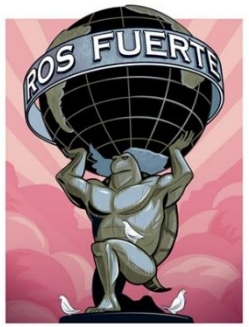
Box Turtle

C Turtle

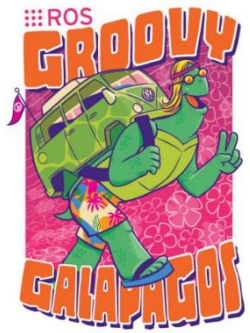
Diamondback



Electric Emys



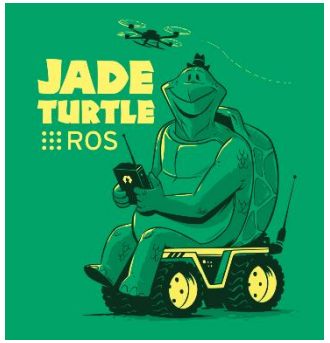
Fuerte Turtle



Groovy Galapagos

Hydro

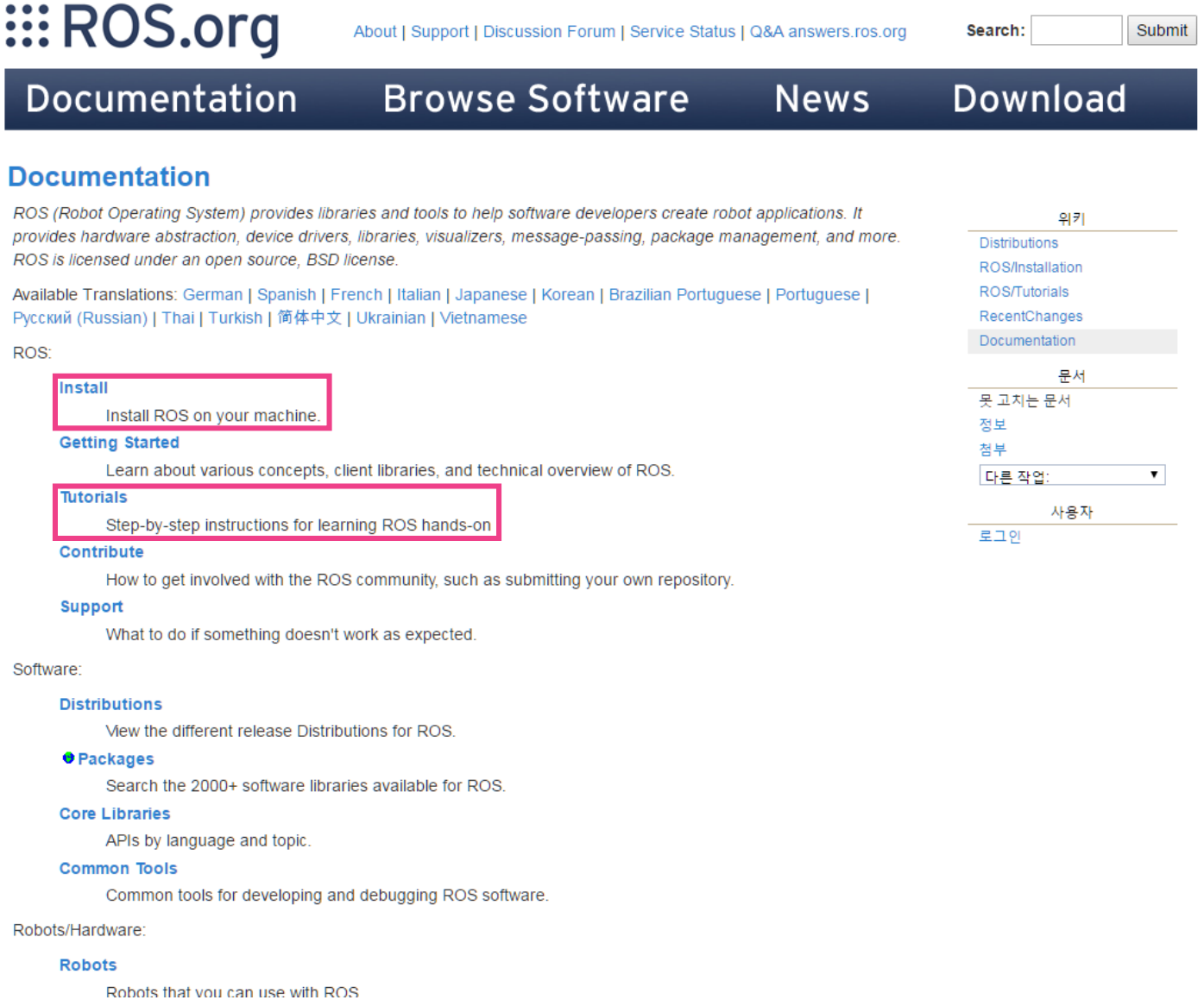
Indigo



Jade

Kinetic Kame

ROS?



ROS.org

About | Support | Discussion Forum | Service Status | Q&A answers.ros.org

Search: Submit

Documentation Browse Software News Download

Documentation

ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license.

Available Translations: German | Spanish | French | Italian | Japanese | Korean | Brazilian Portuguese | Portuguese | Русский (Russian) | Thai | Turkish | 简体中文 | Ukrainian | Vietnamese

ROS:

Install
Install ROS on your machine.

Getting Started
Learn about various concepts, client libraries, and technical overview of ROS.

Tutorials
Step-by-step instructions for learning ROS hands-on

Contribute
How to get involved with the ROS community, such as submitting your own repository.

Support
What to do if something doesn't work as expected.

Software:

Distributions
View the different release Distributions for ROS.

Packages
Search the 2000+ software libraries available for ROS.

Core Libraries
APIs by language and topic.

Common Tools
Common tools for developing and debugging ROS software.

Robots/Hardware:

Robots
Robots that you can use with ROS

위키
Distributions
ROS/Installation
ROS/Tutorials
RecentChanges
Documentation

문서
못 고치는 문서
정보
첨부
다른 작업:

사용자
로그인

Install

ROS.org About | Support | Discussion Forum | Service Status | Q&A answers.ros.org Search: Submit

Documentation Browse Software News Download

ROS/ Installation

ROS Kinetic installation instructions

These instructions will install the **ROS Kinetic Kame** distribution, which is available for Ubuntu Wily (15.10) and Ubuntu Xenial (16.04 LTS), among other platform options.

To install our previous release, **ROS Jade Turtle**, please see the Jade installation instructions.

The previous long-term support release, **ROS Indigo Igloo**, is available for Ubuntu Trusty (14.04 LTS) and many other platforms. Please refer to the Indigo installation instructions if you need to use this version due to robot or platform compatibility reasons.

The links below contain instructions for installing **ROS Kinetic Kame** on various operating systems. You may also wish to look at robot-specific installation options instead.

위키
Distributions
ROS/Installation
ROS/Tutorials
RecentChanges
문서
못 고치는 문서
정보
첨부
다른 작업:
사용자
로그인

Select Your Platform

Supported:

Ubuntu Wily amd64 i386
Xenial amd64 i386 armhf

Debian Jessie amd64 arm64

Source installation

Experimental:

OS X (Homebrew)

Gentoo

Or, Select your robot

Robots:
See all robots supported here: Robots

ROS.org About | Support | Discussion Forum | Service Status | Q&A answers.ros.org Search: Submit

Documentation Browse Software News Download

kinetic/ Installation/ Ubuntu

Ubuntu install of ROS Kinetic

We are building Debian packages for several Ubuntu platforms, listed below. These packages are more efficient than source-based builds and are our preferred installation method for Ubuntu. Note that there are also packages available from Ubuntu upstream. Please see UpstreamPackages to understand the difference.

Ubuntu packages are built for the following distros and architectures.

Distro	amd64	i386	armhf
Wily	X	X	
Xenial	X	X	X

If you need to install from source (**not recommended**), please see source (download-and-compile) installation instructions.

If you rely on these packages, please support OSRF.

These packages are built and hosted on infrastructure maintained and paid for by the Open Source Robotics Foundation, a 501(c)(3) non-profit organization. If OSRF were to receive one penny for each downloaded package for just two months, we could cover our annual costs to manage, update, and host all of our online services. Please consider donating to OSRF today.

위키
Distributions
ROS/Installation
ROS/Tutorials
RecentChanges
Ubuntu
문서
못 고치는 문서
정보
첨부
다른 작업:
사용자
로그인

차례
1. Ubuntu install of ROS Kinetic
   1. Installation
      1. Configure your Ubuntu repositories
      2. Setup your sources.list
      3. Set up your keys
      4. Installation
      5. Initialize rosdep
      6. Environment setup
      7. Getting rosinstall
      8. Build farm status
   2. Tutorials

02

Install

1.4 Installation

First, make sure your Debian package index is up-to-date:

```
sudo apt-get update
```

There are many different libraries and tools in ROS. We provided four default configurations to get you started. You can also install ROS packages individually.

In case of problems with the next step, you can use following repositories instead of the ones mentioned above ros-shadow-fixed

Desktop-Full Install: (Recommended) : ROS, rqt, rviz, robot-generic libraries, 2D/3D simulators, navigation and 2D/3D perception

```
sudo apt-get install ros-kinetic-desktop-full
```

or click here

Desktop Install: ROS, rqt, rviz, and robot-generic libraries

```
sudo apt-get install ros-kinetic-desktop
```

or click here

ROS-Base: (Bare Bones) ROS package, build, and communication libraries. No GUI tools.

```
sudo apt-get install ros-kinetic-ros-base
```

or click here

Individual Package: You can also install a specific ROS package (replace underscores with dashes of the package name):

```
sudo apt-get install ros-kinetic-PACKAGE
```

e.g.

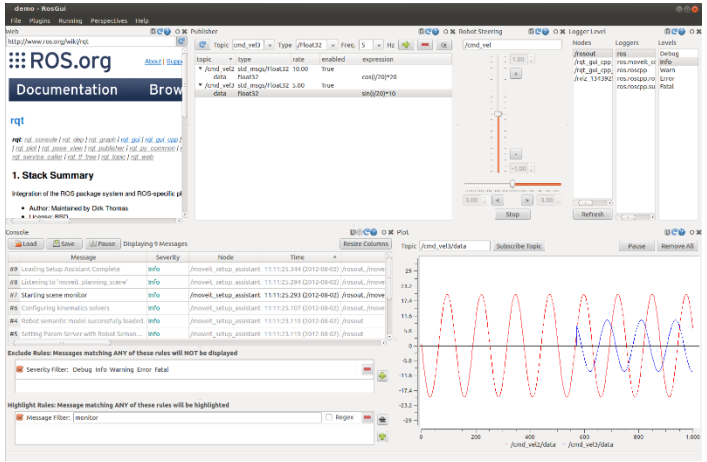
```
sudo apt-get install ros-kinetic-slam-gmapping
```

To find available packages, use:

```
apt-cache search ros-kinetic
```

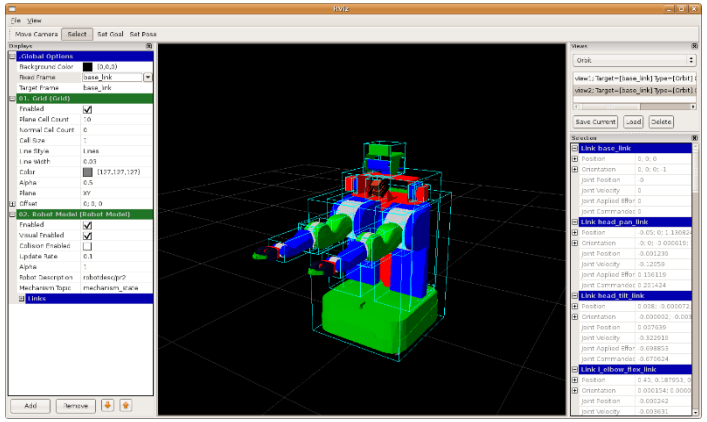
-rqt

Qt-based framework for GUI development



-rviz

3D visualization tool



02

Install

1.5 Initialize rosdep

Before you can use ROS, you will need to initialize rosdep. rosdep enables you to easily install system dependencies for source you want to compile and is required to run some core components in ROS.

```
sudo rosdep init
rosdep update
```

-> 소스파일 컴파일을 위한 dependency 초기화 과정

$ sudo rosdep init

```
ERROR: cannot download default sources list from:
https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/sources.list.d/20-default.list
Website may be down.
```

$ sudo c_rehash /etc/ssl/certs -> 네트워크 & 인증서 문제

```
Doing /etc/ssl/certs
WARNING: Skipping duplicate certificate ca-certificates.crt
WARNING: Skipping duplicate certificate ca-certificates.crt
ubuntu@tegra-ubuntu:~$ wget https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/sources.list.d/20-default.list
--2017-03-22 04:59:44--  https://raw.githubusercontent.com/ros/rosdistro/master/rosdep/sources.list.d/20-default.list
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.100.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.100.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 588 [text/plain]
Saving to: '20-default.list'

20-default.list                                     100%[=================================================================>]

2017-03-22 04:59:45 (2.96 MB/s) - '20-default.list' saved [588/588]
```

03

Tutorial

-Beginner level

1. Installing and configuring your ROS environment.
2. Navigating the ROS filesystem.
3. Creating a ROS package.
4. Building a ROS package.
5. Understanding ROS nodes.
6. Understanding ROS Topics.
7. Understanding ROS services and parameters.
8. Using rqt_console and roslaunch.
9. Using rosed to edit files in ROS.
10. Creating a ROS msg and srv.
11. Writing & Examining a simple Publisher and Subscriber.
12. Writing & Examining a simple Service and Client.
13. Recording and playing back data.
14. Getting started with roswtf.

ROS

- 하드웨어 추상화
- 디바이스 제어
- 패키지 관리
- 메시지 관리

-msg

Describe the fields of a ROS message.
<package>/msg/<message file name>.msg

```
자료형 변수명1
자료형 변수명2
.....
```

Example

```
float32 testval
string teststr
.....
```

-srv

Describes a service
<package>/srv/<service file name>.srv

```
자료형 입력변수명1
자료형 입력변수명2
.....
---
자료형 출력변수명
.....
```

Example

```
int64 A
int64 B
---
int64 Sum
```

03

Tutorial

-Publisher and Subscriber

Publisher

```cpp
#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

```cpp
#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;
}
```

Tutorial

03

-Publisher and Subscriber

Edit <package>/CMakeLists.txt

```
## Generate messages in the 'msg' folder
add_message_files(
   FILES
   Num.msg
 )

## Generate services in the 'srv' folder
 add_service_files(
   FILES
   AddTwoInts.srv
 )

## Generate actions in the 'action' folder
# add_action_files(
#   FILES
#   Action1.action
#   Action2.action
# )

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

```
## 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(talker src/talker.cpp)
add_executable(listener src/listener.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(talker beg_generate_messages_cpp)
add_dependencies(listener beg_generate_messages_cpp)

## Specify libraries to link a library or executable target against
target_link_libraries(talker ${catkin_LIBRARIES})
target_link_libraries(listener ${catkin_LIBRARIES})
```

Publisher

```
[ INFO] [1493533811.719219174]: hello world 187
[ INFO] [1493533811.818693321]: hello world 188
[ INFO] [1493533811.919110269]: hello world 189
[ INFO] [1493533812.019314458]: hello world 190
[ INFO] [1493533812.118383091]: hello world 191
[ INFO] [1493533812.219399093]: hello world 192
[ INFO] [1493533812.319497711]: hello world 193
[ INFO] [1493533812.419271542]: hello world 194
```

Subscriber

```
[ INFO] [1493533811.723664167]: I heard: [hello world 187]
[ INFO] [1493533811.823271542]: I heard: [hello world 188]
[ INFO] [1493533811.923743540]: I heard: [hello world 189]
[ INFO] [1493533812.024112624]: I heard: [hello world 190]
[ INFO] [1493533812.118967666]: I heard: [hello world 191]
[ INFO] [1493533812.221569741]: I heard: [hello world 192]
[ INFO] [1493533812.325651744]: I heard: [hello world 193]
[ INFO] [1493533812.425408648]: I heard: [hello world 194]
```

03

Tutorial

-Service and Client

Service

```cpp
#include "ros/ros.h"
#include "beginner_tutorials/AddTwoInts.h"

bool add(beginner_tutorials::AddTwoInts::Request  &req,
         beginner_tutorials::AddTwoInts::Response &res)
{
  res.sum = req.a + req.b;
  ROS_INFO("request: x=%ld, y=%ld", (long int)req.a, (long int)req.b);
  ROS_INFO("sending back response: [%ld]", (long int)res.sum);
  return true;
}

int main(int argc, char **argv)
{
  ros::init(argc, argv, "add_two_ints_server");
  ros::NodeHandle n;

  ros::ServiceServer service = n.advertiseService("add_two_ints", add);
  ROS_INFO("Ready to add two ints.");
  ros::spin();

  return 0;
}
```

Client

```cpp
#include "ros/ros.h"
#include "beginner_tutorials/AddTwoInts.h"
#include <cstdlib>

int main(int argc, char **argv)
{
  ros::init(argc, argv, "add_two_ints_client");
  if (argc != 3)
  {
    ROS_INFO("usage: add_two_ints_client X Y");
    return 1;
  }

  ros::NodeHandle n;
  ros::ServiceClient client = n.serviceClient<beginner_tutorials::AddTwoInts>("add_two_ints");
  beginner_tutorials::AddTwoInts srv;
  srv.request.a = atoll(argv[1]);
  srv.request.b = atoll(argv[2]);
  if (client.call(srv))
  {
    ROS_INFO("Sum: %ld", (long int)srv.response.sum);
  }
  else
  {
    ROS_ERROR("Failed to call service add_two_ints");
    return 1;
  }

  return 0;
}
```

Tutorial

03

-Service and Client

Edit <package>/CMakeLists.txt

```
## Generate messages in the 'msg' folder
add_message_files(
   FILES
   Num.msg
 )

## Generate services in the 'srv' folder
 add_service_files(
   FILES
   AddTwoInts.srv
 )

## Generate actions in the 'action' folder
# add_action_files(
#   FILES
#   Action1.action
#   Action2.action
# )

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

```
add_executable(add_two_ints_server src/add_two_ints_server.cpp)
target_link_libraries(add_two_ints_server ${catkin_LIBRARIES})
add_dependencies(add_two_ints_server beg_gencpp)

add_executable(add_two_ints_client src/add_two_ints_client.cpp)
target_link_libraries(add_two_ints_client ${catkin_LIBRARIES})
add_dependencies(add_two_ints_client beg_gencpp)
```

Service

```
ubuntu@tegra-ubuntu:~/catkin_ws$ rosrun beginner_tutorials add_two_ints_server
[rospack] Error: package 'beginner_tutorials' not found
ubuntu@tegra-ubuntu:~/catkin_ws$ rosrun beg add_two_ints_server
[ INFO] [1493537656.602297588]: Ready to add two ints.
[ INFO] [1493537667.649629245]: request: x=1, y=3
[ INFO] [1493537667.649702838]: sending back response: [4]
```

Client

```
ubuntu@tegra-ubuntu:~/catkin_ws/src/beg$ rosrun beg add_two_ints_client 1 3
[ INFO] [1493537667.650041792]: Sum: 4
ubuntu@tegra-ubuntu:~/catkin_ws/src/beg$
```

Q & A