

# 無人載具技術與應用

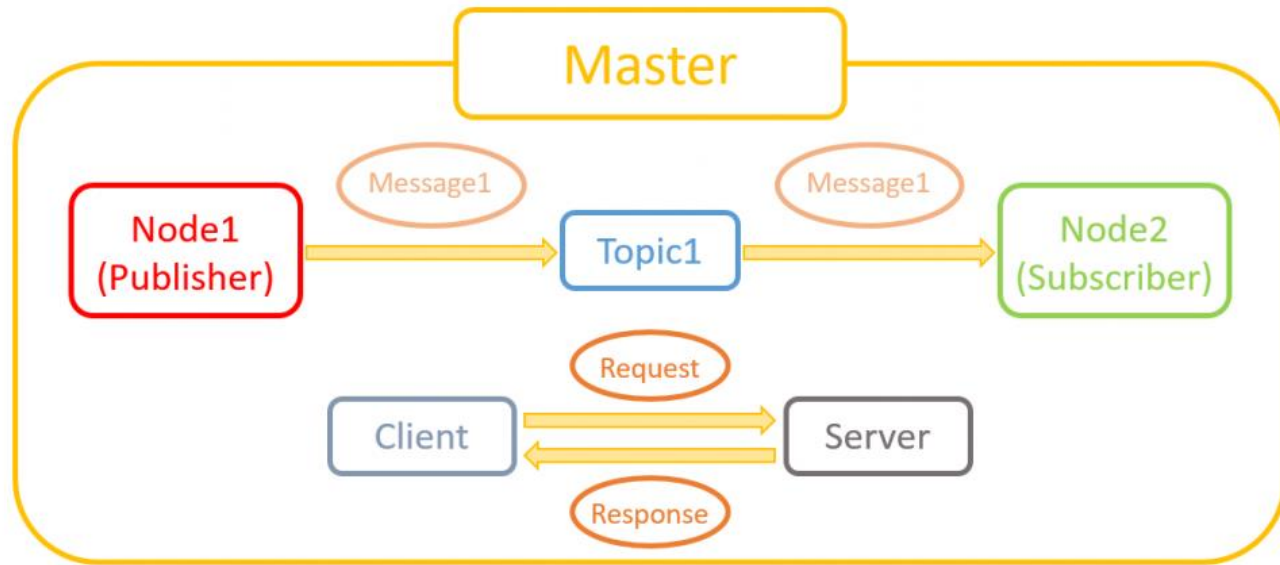
## ROS

徐瑋隆

wlhsu304@gmail.com

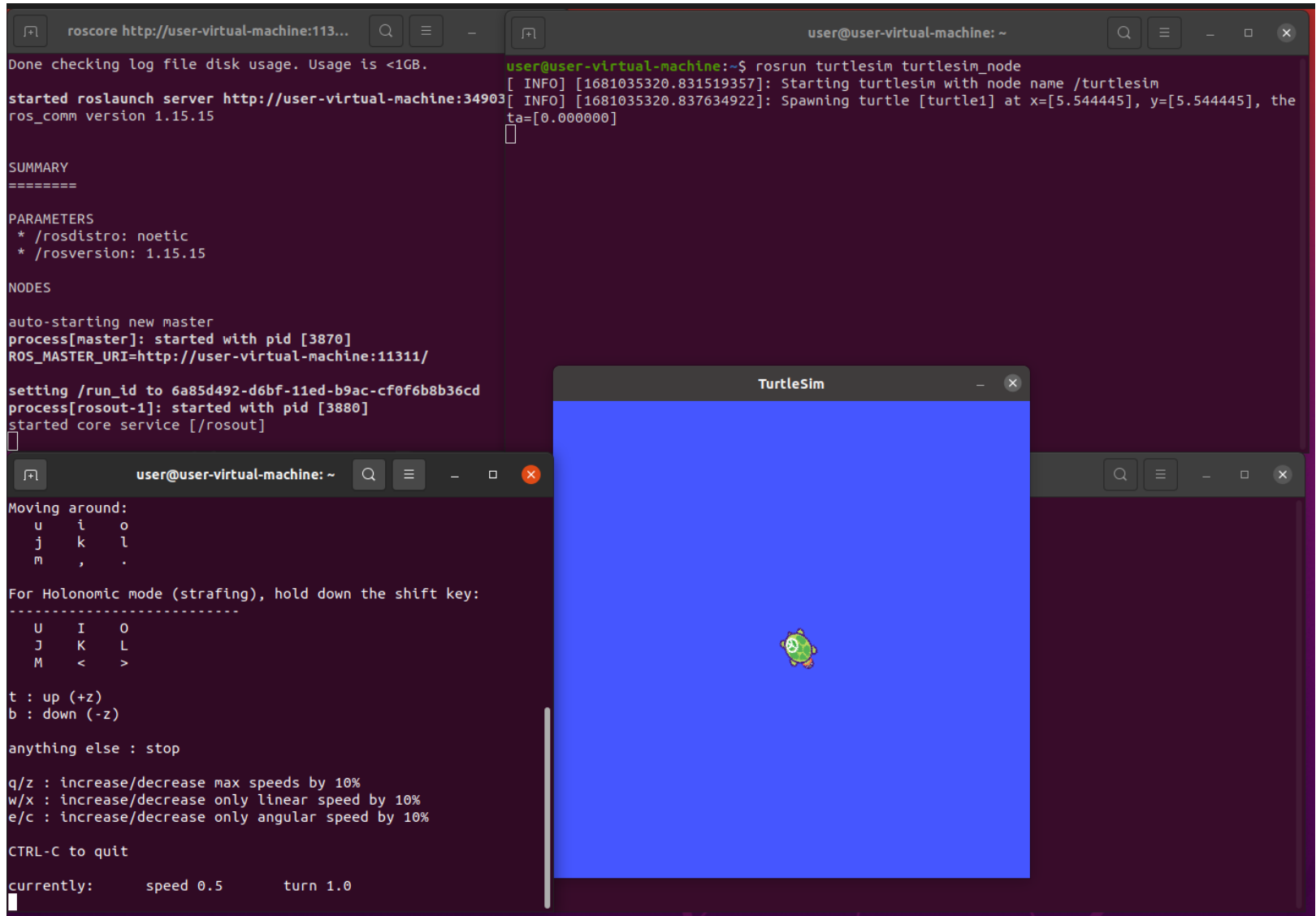
**ROS SERVICE**

# ROS Service



- Services 同時包含了 Request 及 Response 的訊息機制
  - 在 Topic 裡頭有 Publisher 以及 Subscriber，兩者是由 Publisher 向 Topic 發布訊息，Subscriber 才會接收到訊息。
  - Service 裡只有 Server 與 Client，Client 必須透過 Service 向 Server 提出 Request，而 Server 才會回傳對應的 Response 給 Client。

# rosservice – turtlesim 01



The screenshot shows a terminal window with two panes. The left pane displays the output of `roslaunch` for the `turtlesim` package, including log file usage, node spawning details, and a summary of parameters and nodes. The right pane shows the output of `roslaunch` for the `turtlesim` package, including log file usage, node spawning details, and a summary of parameters and nodes. Below the terminal panes, a `TurtleSim` window is visible, showing a blue background with a small green turtle icon in the center.

```
roscore http://user-virtual-machine:113...
Done checking log file disk usage. Usage is <1GB.
started roslaunch server http://user-virtual-machine:34903
ros_comm version 1.15.15

SUMMARY
=====

PARAMETERS
* /rostdistro: noetic
* /rosversion: 1.15.15

NODES
auto-starting new master
process[roscore]: started with pid [3870]
ROS_MASTER_URI=http://user-virtual-machine:11311/

setting /run_id to 6a85d492-d6bf-11ed-b9ac-cf0f6b8b36cd
process[roscout-1]: started with pid [3880]
started core service [/roscout]

user@user-virtual-machine:~$ roslaunch turtlesim turtlesim_node
[ INFO] [1681035320.831519357]: Starting turtlesim with node name /turtlesim
[ INFO] [1681035320.837634922]: Spawning turtle [turtle1] at x=[5.544445], y=[5.544445], the
ta=[0.000000]

Moving around:
u i o
j k l
m , .

For Holonomic mode (strafing), hold down the shift key:
-----
U I O
J K L
M < >

t : up (+z)
b : down (-z)

anything else : stop

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%

CTRL-C to quit

currently: speed 0.5 turn 1.0
```

roscore

roslaunch turtlesim turtlesim\_node

roslaunch teleop\_twist\_keyboard teleop\_twist\_keyboard.py /cmd\_vel:=/turtle1/cmd\_vel

# rosservice – turtlesim 02

rostopic list

rosservice list

rosservice call /reset

rosservice type /reset | rossrv show

rosservice type /spawn | rossrv show

rosservice info /spawn

rosservice call /spawn 1 2 0 kk

```
user@user-virtual-machine:~$ rosservice list
/clear
/kill
/reset
/rosout/get_loggers
/rosout/set_logger_level
/spawn
/teleop_twist_keyboard/get_loggers
/teleop_twist_keyboard/set_logger_level
/turtle1/set_pen
/turtle1/teleport_absolute
/turtle1/teleport_relative
/turtlesim/get_loggers
/turtlesim/set_logger_level
user@user-virtual-machine:~$ rosservice call /reset

user@user-virtual-machine:~$ rosservice call /reset

user@user-virtual-machine:~$ rosservice type /spawn | rossrv show
float32 x
float32 y
float32 theta
string name
---
string name

user@user-virtual-machine:~$ rosservice call /spawn "x: 1.0
y: 3.0
theta: 1.0
name: 'abc'"
name: "abc"
```

# rosservice – turtlesim 03

rostopic list

rosservice info /kill

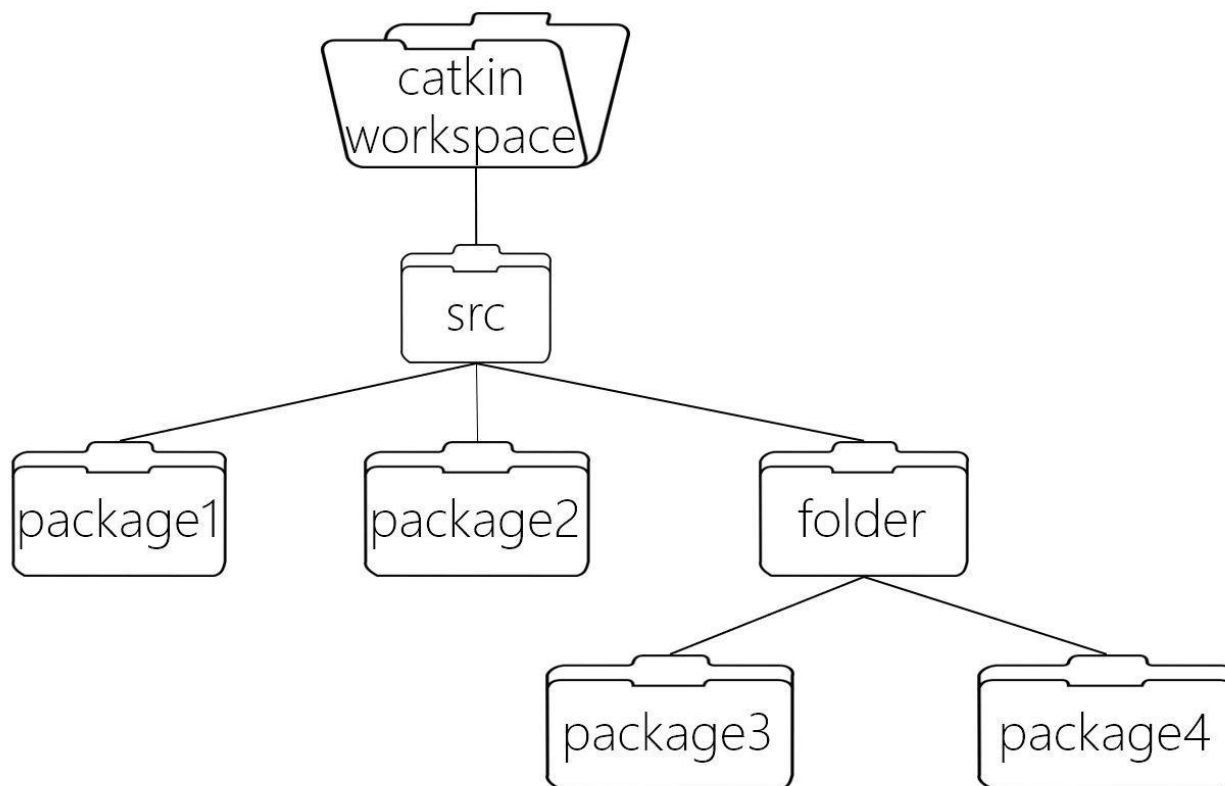
rosservice call /kill kk

```
user@user-virtual-machine:~$ rostopic list
/kk/cmd_vel
/kk/color_sensor
/kk/pose
/rosout
/rosout_agg
/turtle1/cmd_vel
/turtle1/color_sensor
/turtle1/pose
user@user-virtual-machine:~$ rosservice call /kill kk

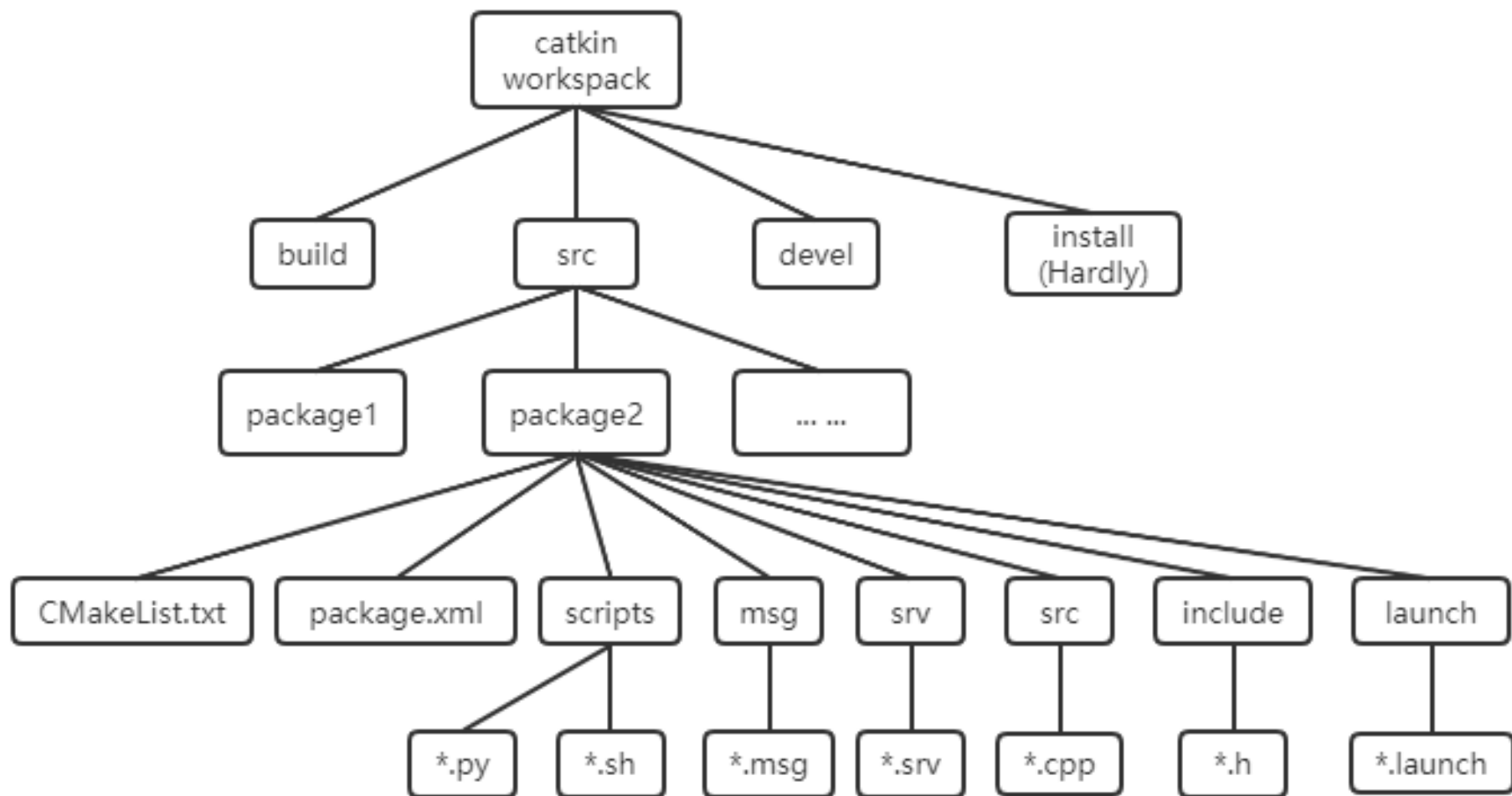
user@user-virtual-machine:~$ rostopic list
/rosout
/rosout_agg
/turtle1/cmd_vel
/turtle1/color_sensor
/turtle1/pose
```

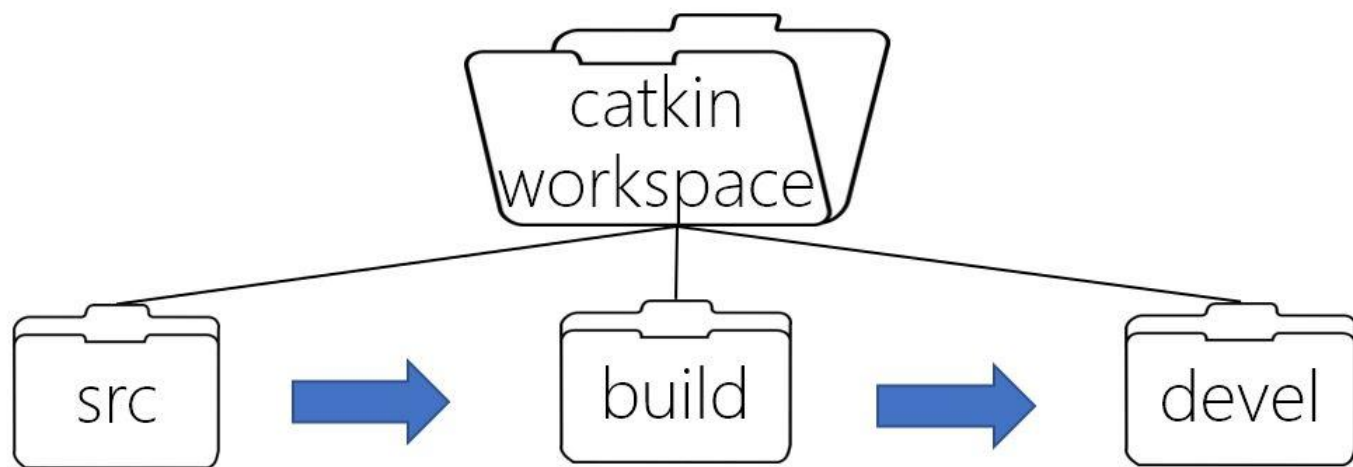
ros node coding

# **ROS SERVICE**









package源代码包

cmake&catkin缓存和中间文件

目标文件

---

catkin\_ws  
(ROS package)

catkin



---

CMakeLists.txt

cmake



---

Makefile

make



---

hello.cpp

gcc/g++



hello.o

gcc/g++



hello

---

# ros node 01

```
user@user-virtual-machine:~$ cd catkin_ws/src/  
user@user-virtual-machine:~/catkin_ws/src$ catkin_create_pkg my_work03 roscpp ro  
spy std_msgs  
Created file my_work03/package.xml  
Created file my_work03/CMakeLists.txt  
Created folder my_work03/include/my_work03  
Created folder my_work03/src  
Successfully created files in /home/user/catkin_ws/src/my_work03. Please adjust  
the values in package.xml.
```

```
cd catkin_ws/src/  
catkin_create_pkg my_work03 roscpp rospy std_msgs  
ls
```

# ros node 02

```
user@user-virtual-machine:~/catkin_ws/src$ cd my_work03/src/  
user@user-virtual-machine:~/catkin_ws/src/my_work03/src$ gedit myNode01.cpp &  
[1] 34055
```



```
1 #include "ros/ros.h"  
2 |  
3 int main(int argc, char **argv)  
4 {  
5     ros::init(argc, argv, "myNode01");  
6     ros::NodeHandle n;  
7  
8     ros::Rate loop_rate(10);    // 10Hz  
9  
10    while (ros::ok())  
11    {  
12        ros::spinOnce();  
13        loop_rate.sleep();  
14    }  
15    return 0;  
16 }
```

```
cd my_work03/src/  
gedit myNode01.cpp &
```

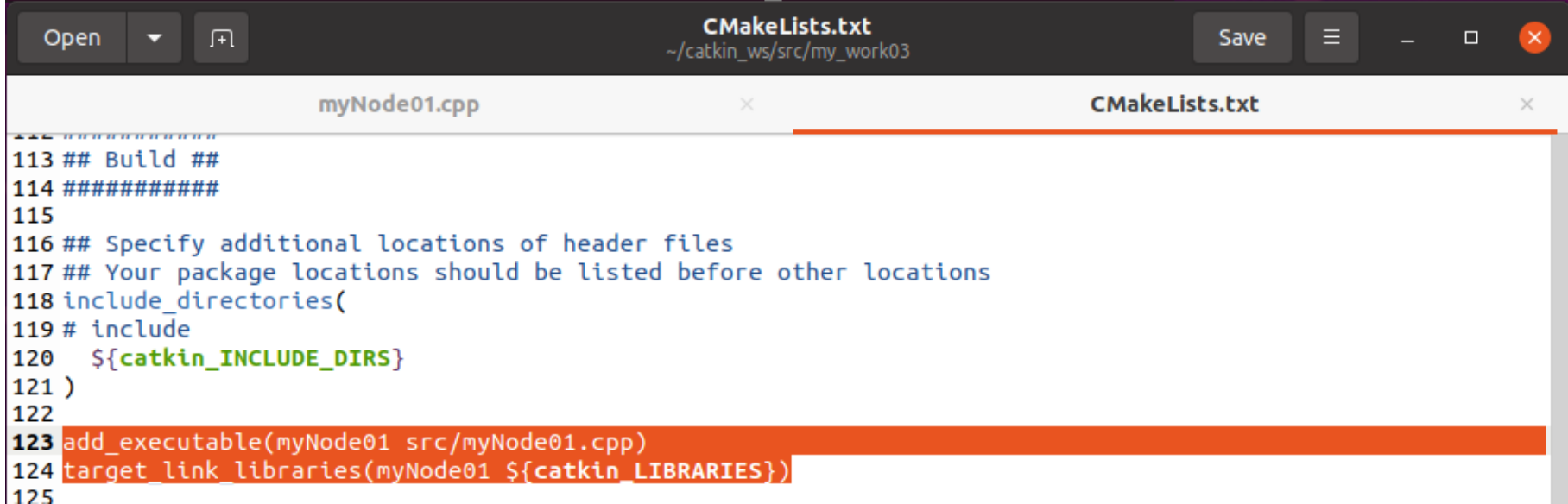
# ros node 03

```
my_work01/ my_work02/ my_work03/  
user@user-virtual-machine:~/catkin_ws/src$ cd my_work02/src/  
user@user-virtual-machine:~/catkin_ws/src/my_work02/src$ gedit myNode01.cpp  
[  
myNode01.cpp  
~/catkin_ws/src/my_work02/src  
Save  
1 #include "ros/ros.h"  
2 |  
3 int main(int argc, char **argv)  
4 {  
5     ros::init(argc, argv, "myNode01");  
6     ros::NodeHandle n;  
7  
8     ros::Rate loop_rate(10);    // 10Hz  
9  
10    while (ros::ok())  
11    {  
12        ros::spinOnce();  
13        loop_rate.sleep();  
14    }  
15    return 0;  
16 }
```

```
#include "ros/ros.h"  
int main(int argc, char **argv)  
{  
    ros::init(argc, argv, "myNode01");  
    ros::NodeHandle n;  
    ros::Rate loop_rate(10); // 10Hz  
    ROS_INFO("myNode01: hi");  
    while (ros::ok()) {  
        ros::spinOnce();  
        loop_rate.sleep();  
    }  
    return 0;  
}
```

# ros node 04

```
user@user-virtual-machine:~/catkin_ws/src/my_work03/src$ cd ..  
user@user-virtual-machine:~/catkin_ws/src/my_work03$ gedit CMakeLists.txt &  
[2] 24554  
user@user-virtual-machine:~/catkin_ws/src/my_work03$
```



```
113 ## Build ##  
114 #####  
115  
116 ## Specify additional locations of header files  
117 ## Your package locations should be listed before other locations  
118 include_directories(  
119 # include  
120   ${catkin_INCLUDE_DIRS}  
121 )  
122  
123 add_executable(myNode01 src/myNode01.cpp)  
124 target_link_libraries(myNode01 ${catkin_LIBRARIES})  
125
```

```
add_executable(myNode01 src/myNode01.cpp)  
target_link_libraries(myNode01 ${catkin_LIBRARIES})
```

```
user@user-virtual-machine:~/catkin_ws/src/my_work02/src$ cd ../../..  
user@user-virtual-machine:~/catkin_ws$ catkin_make
```

```
cd ~/catkin_ws  
catkin_make
```

編譯程式

# ros node 05

```
user@user-virtual-machine:~/catkin_ws/src/my_work03/src$ rosrund my_work03 myNode01  
[ INFO] [1681044576.929375008]: myNode01: hi
```

roscore

roslun my\_work03 myNode01



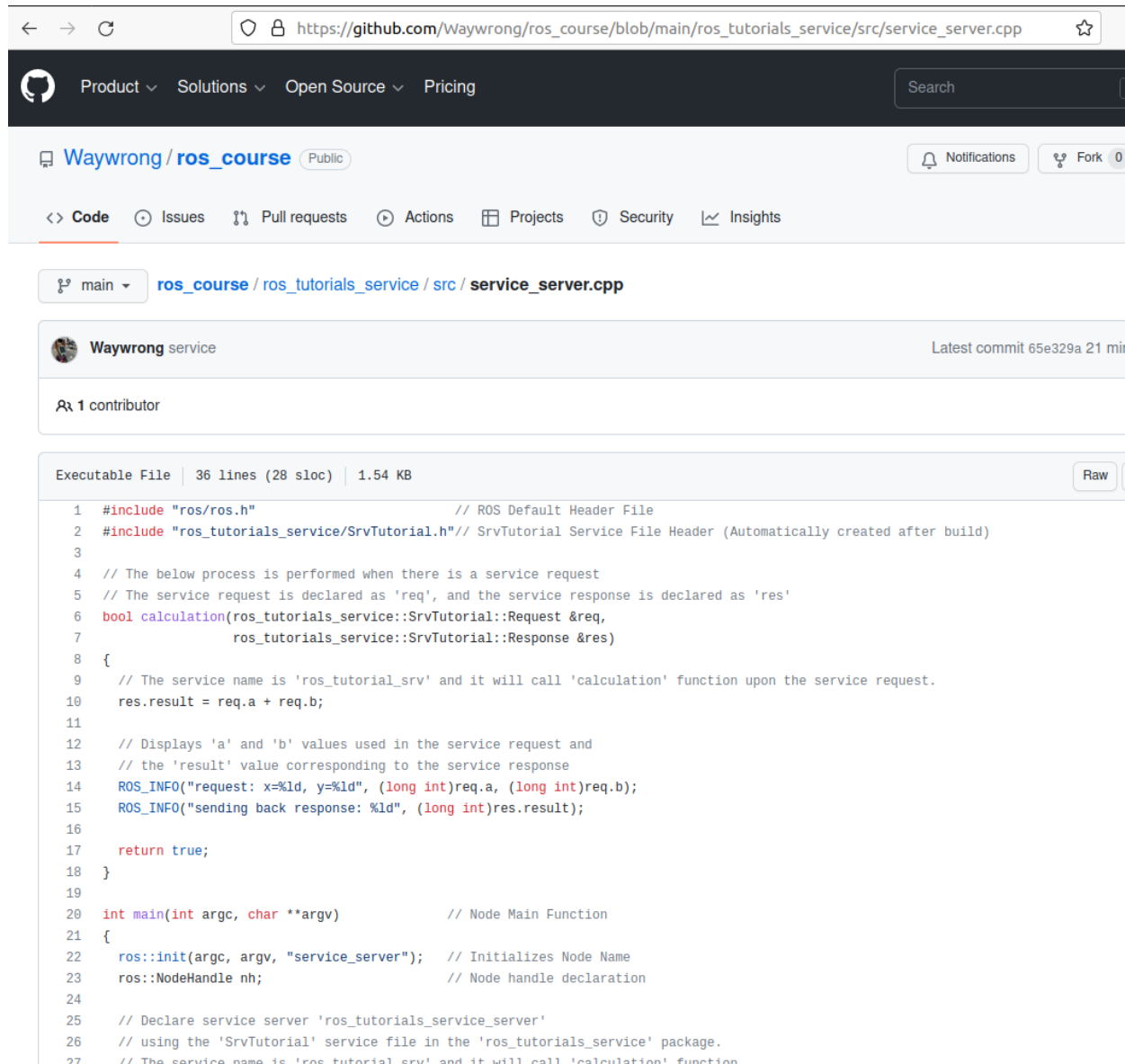
ros node coding

# ROS SERVICE

# ros service 01

ros\_tutorials\_service/src/service\_server.cpp

[https://github.com/Waywrong/ros\\_course/](https://github.com/Waywrong/ros_course/)



```
1 #include "ros/ros.h" // ROS Default Header File
2 #include "ros_tutorials_service/SrvTutorial.h" // SrvTutorial Service File Header (Automatically created after build)
3
4 // The below process is performed when there is a service request
5 // The service request is declared as 'req', and the service response is declared as 'res'
6 bool calculation(ros_tutorials_service::SrvTutorial::Request &req,
7                 ros_tutorials_service::SrvTutorial::Response &res)
8 {
9     // The service name is 'ros_tutorial_srv' and it will call 'calculation' function upon the service request.
10    res.result = req.a + req.b;
11
12    // Displays 'a' and 'b' values used in the service request and
13    // the 'result' value corresponding to the service response
14    ROS_INFO("request: x=%ld, y=%ld", (long int)req.a, (long int)req.b);
15    ROS_INFO("sending back response: %ld", (long int)res.result);
16
17    return true;
18 }
19
20 int main(int argc, char **argv) // Node Main Function
21 {
22     ros::init(argc, argv, "service_server"); // Initializes Node Name
23     ros::NodeHandle nh; // Node handle declaration
24
25     // Declare service server 'ros_tutorials_service_server'
26     // using the 'SrvTutorial' service file in the 'ros_tutorials_service' package.
27     // The service name is 'ros_tutorial_srv' and it will call 'calculation' function
```

# ros service 02

```
ser-virtual-machine: ~/catkin_ws/src/my_work03/src
l-machine:~$ cd catkin_ws/src/my_work03/src/
l-machine:~/catkin_ws/src/my_work03/src$ gedit myNode01.cpp &

myNode01.cpp
~/catkin_ws/src/my_work03/src

os.h"
rk03/SrvTutorial.h"

n(my_work03::SrvTutorial::Request &req,
  my_work03::SrvTutorial::Response &res)

req.a + req.b;
uest: x=%ld, y=%ld", (long int)req.a, (long int)req.b);
ding back response: %ld", (long int)res.result);

gc, char **argv)

c, argv, "myNode01");
le n;
server ros_tutorials_service_server = nh.advertiseService("ros_tutorial_srv", calculation);
p_rate(10); // 10Hz
ode01: hi");
k())

ce();
leep();
```


```
#include "ros/ros.h"
#include "my_work03/SrvTutorial.h"
bool calculation(my_work03::SrvTutorial::Request &req,
                my_work03::SrvTutorial::Response &res)
{
    res.result = 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.result);
    return true;
}

int main(int argc, char **argv) {
    ros::init(argc, argv, "myNode01");
    ros::NodeHandle n;
    ros::ServiceServer ros_tutorials_service_server = n.advertiseService("ros_tutorial_srv",
calculation);
    ros::Rate loop_rate(10); // 10Hz
    ROS_INFO("myNode01: hi");
    while (ros::ok())
    {
        ros::spinOnce();
        loop_rate.sleep();
    }
    return 0;
}
```

# ros service 03


[https://github.com/Waywrong/ros\\_course/](https://github.com/Waywrong/ros_course/)


[ros\\_tutorials\\_service/srv/SrvTutorial.srv](#)

 **Waywrong** / **ros\_course** Public

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#)

[main](#) [ros\\_course](#) / [ros\\_tutorials\\_service](#) / [srv](#) / **SrvTutorial.srv**

 **Waywrong** service

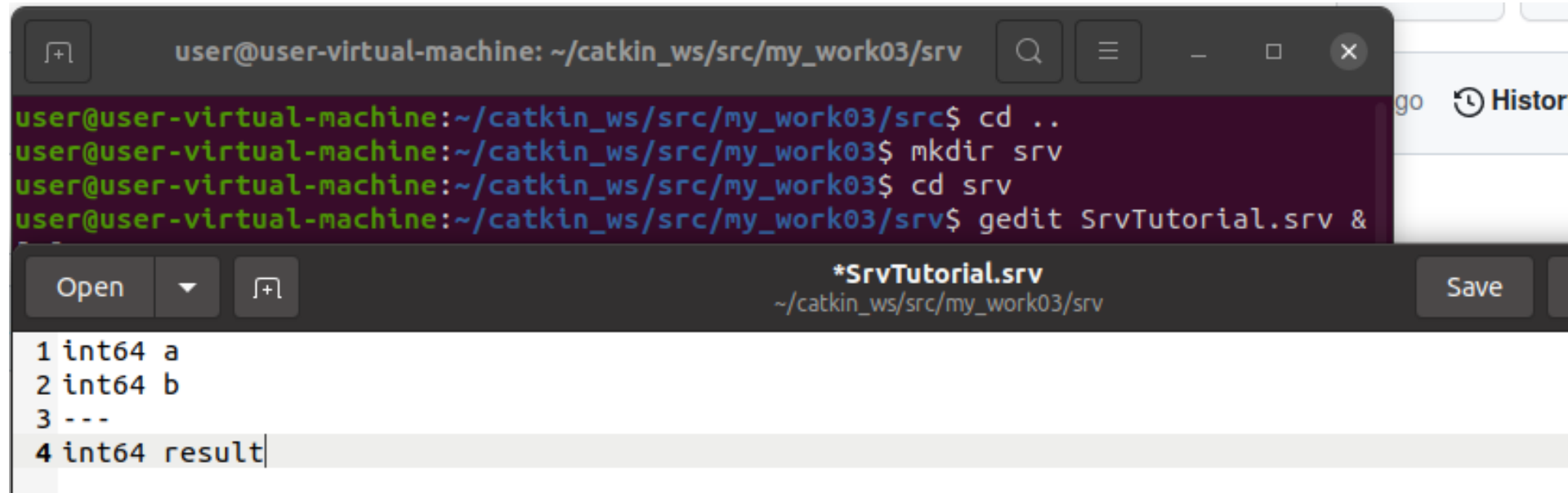
 1 contributor

Executable File | 4 lines (4 sloc) | 33 Bytes

```
1  int64 a
2  int64 b
3  ---
4  int64 result
```

# ros service 04

```
cd ..  
mkdir srv  
cd srv  
gedit SrvTutorial.srv &
```



The screenshot shows a terminal window and a text editor. The terminal window, titled "user@user-virtual-machine: ~/catkin\_ws/src/my\_work03/srv", displays the following commands and their outputs:

```
user@user-virtual-machine:~/catkin_ws/src/my_work03/src$ cd ..  
user@user-virtual-machine:~/catkin_ws/src/my_work03$ mkdir srv  
user@user-virtual-machine:~/catkin_ws/src/my_work03$ cd srv  
user@user-virtual-machine:~/catkin_ws/src/my_work03/srv$ gedit SrvTutorial.srv &
```

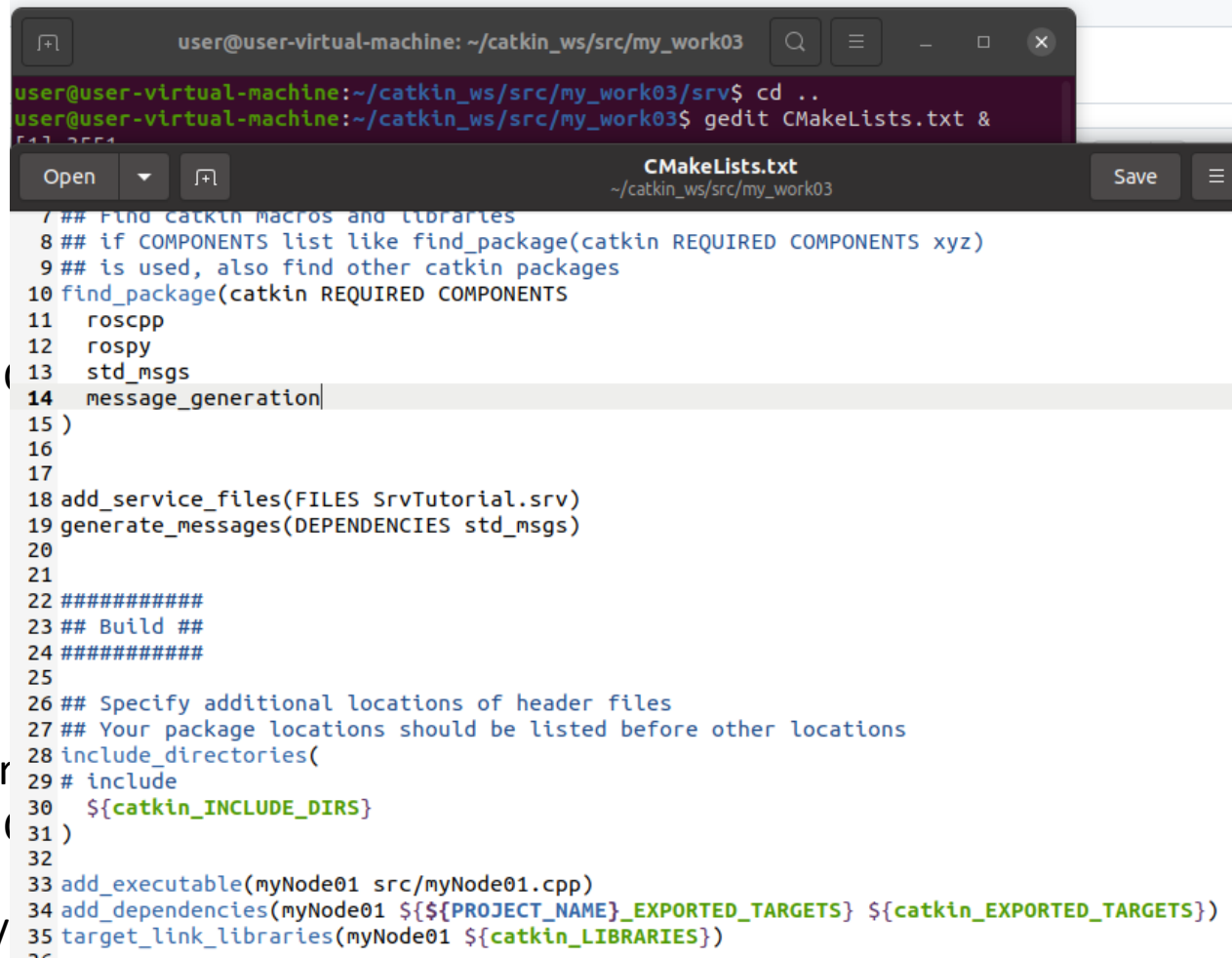
The text editor, titled "\*SrvTutorial.srv" and located at "~/catkin\_ws/src/my\_work03/srv", shows the content of the service file:

```
1 int64 a  
2 int64 b  
3 ---  
4 int64 result
```

```
int64 a  
int64 b  
---  
int64 result
```

# ros service 05

```
cd ..  
gedit CMakeLists.txt &
```



The image shows a terminal window and a CMakeLists.txt file editor. The terminal window is titled "user@user-virtual-machine: ~/catkin\_ws/src/my\_work03" and shows the following commands and output:

```
user@user-virtual-machine:~/catkin_ws/src/my_work03/srv$ cd ..  
user@user-virtual-machine:~/catkin_ws/src/my_work03$ gedit CMakeLists.txt &
```

The CMakeLists.txt file is titled "CMakeLists.txt" and shows the following content:

```
1 ## Find catkin macros and libraries  
2 ## if COMPONENTS list like find_package(catkin REQUIRED COMPONENTS xyz)  
3 ## is used, also find other catkin packages  
4 find_package(catkin REQUIRED COMPONENTS  
5   roscpp  
6   rospy  
7   std_msgs  
8   message_generation  
9 )  
10  
11  
12 add_service_files(FILES SrvTutorial.srv)  
13 generate_messages(DEPENDENCIES std_msgs)  
14  
15  
16 #####  
17 ## Build ##  
18 #####  
19  
20 ## Specify additional locations of header files  
21 ## Your package locations should be listed before other locations  
22 include_directories(  
23   # include  
24   ${catkin_INCLUDE_DIRS}  
25 )  
26  
27 add_executable(myNode01 src/myNode01.cpp)  
28 add_dependencies(myNode01 ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})  
29 target_link_libraries(myNode01 ${catkin_LIBRARIES})
```

```
find_package(catkin REQUIRED COMPONENTS  
  roscpp  
  rospy  
  std_msgs  
  message_generation  
)  
  
add_service_files(FILES SrvTutorial.srv)  
generate_messages(DEPENDENCIES std_msgs)  
  
add_executable(myNode01 src/myNode01.cpp)  
add_dependencies(myNode01  
  ${${PROJECT_NAME}_EXPORTED_TARGETS}  
  ${catkin_EXPORTED_TARGETS})  
target_link_libraries(myNode01  
  ${catkin_LIBRARIES})
```

```
find_package(catkin REQUIRED COMPONENTS
```

```
  roscpp
```

```
  rospy
```

```
  std_msgs
```

```
  message_generation
```

```
)
```

```
add_service_files(FILES SrvTutorial.srv)
```

```
generate_messages(DEPENDENCIES std_msgs)
```

```
add_executable(myNode01 src/myNode01.cpp)
```

```
add_dependencies(myNode01 ${PROJECT_NAME}_EXPORTED_TARGETS ${catkin_EXPORTED_TARGETS})
```

```
target_link_libraries(myNode01 ${catkin_LIBRARIES})
```

表示這個程式編譯時有先後  
相依問題(**SrvTutorial.h**)，無  
此行則需多編譯數次才可成  
功



# ros service 06

```
user@user-virtual-machine:~/catkin_ws$ catkin_make
Base path: /home/user/catkin_ws
Source space: /home/user/catkin_ws/src
Build space: /home/user/catkin_ws/build
Devel space: /home/user/catkin_ws/devel
Install space: /home/user/catkin_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/user/catkin_ws/build"
####
```

```
cd ~/catkin_ws
catkin_make
```

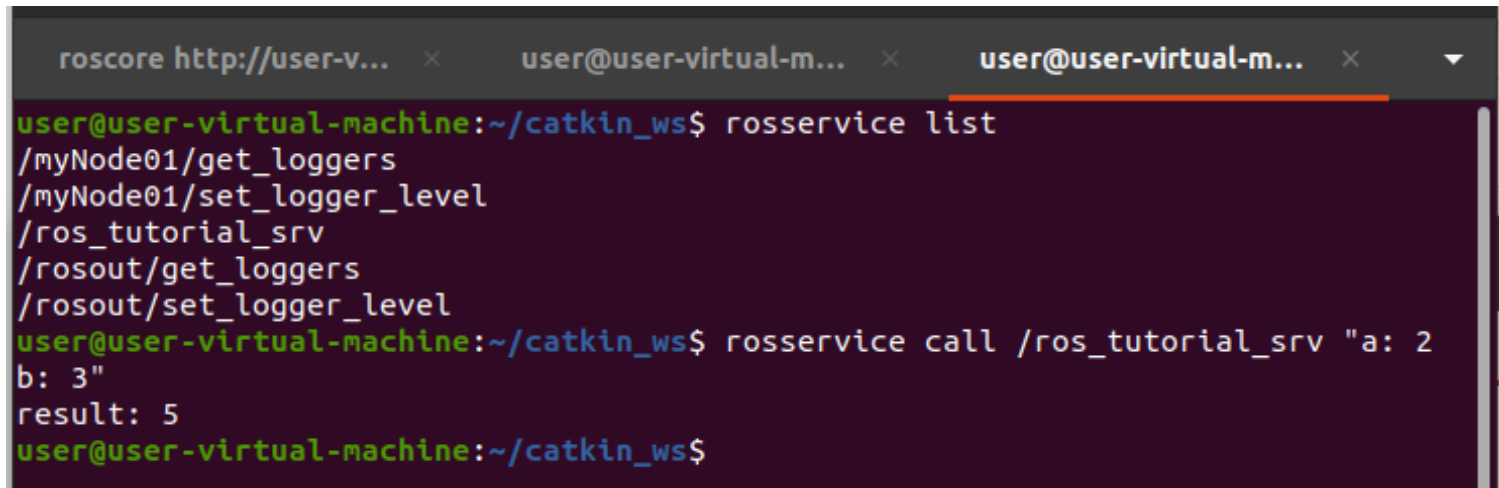
編譯程式

# ros service 07

```
user@user-virtual-machine:~/catkin_ws$ rosrund my_work03 myNode01
[ INFO] [1681050254.623201291]: myNode01: hi
[ INFO] [1681050310.023610107]: request: x=2, y=3
[ INFO] [1681050310.023710495]: sending back response: 5
```

roscore

roslund my\_work03 myNode01

A terminal window with three tabs: 'roscore http://user-v...', 'user@user-virtual-m...', and 'user@user-virtual-m...'. The active tab shows the following commands and output:

```
user@user-virtual-machine:~/catkin_ws$ rosservice list
/myNode01/get_loggers
/myNode01/set_logger_level
/ros_tutorial_srv
/rosout/get_loggers
/rosout/set_logger_level
user@user-virtual-machine:~/catkin_ws$ rosservice call /ros_tutorial_srv "a: 2
b: 3"
result: 5
user@user-virtual-machine:~/catkin_ws$
```

rosservice list

rosservice call /ros\_tutorial\_srv 2 3

# 作業3

- 於本周相同專案內(my\_work03)增加一個service client，並上傳相關檔案
- 參考4/11上課內容，"ROS-Class-6.pdf"
- ros\_tutorials\_service/src/service\_client.cpp，寫一個會送出服務請求(service request)的專案(my\_hw03)
- 計分部分包含
  - 1. 完整性
  - 2. 修改roscpp 名稱
  - 3. 紀錄實驗過程於word檔，紀錄所下的命令與回應，可多利用截圖(圖文並茂加分)
    - roscpp list
    - rosservice list
    - Rossservice info
- 上傳作業包含 (4/25 前上傳):
- 1. CMakeLists.txt
- 2. package.xml
- 3. 修改過的cpp檔
- 4. 實驗紀錄word檔

main ▾ [ros\\_course](#) / [ros\\_tutorials\\_service](#) / [src](#) / [service\\_client.cpp](#)

Go to file



**Waywrong** service

Latest commit 65e329a 1 hour ago 🔗 History

👤 1 contributor

Executable File | 42 lines (35 sloc) | 1.71 KB

Raw

Blame



```
1 #include "ros/ros.h"           // ROS Default Header File
2 #include "ros_tutorials_service/SrvTutorial.h" // SrvTutorial Service File Header (Automatically created after build)
3 #include <cstdlib>              // Library for using the "atoll" function
4
5 int main(int argc, char **argv) // Node Main Function
6 {
7     ros::init(argc, argv, "service_client"); // Initializes Node Name
8
9     if (argc != 3) // Input value error handling
10    {
11        ROS_INFO("cmd : rosrn ros_tutorials_service service_client arg0 arg1");
12        ROS_INFO("arg0: double number, arg1: double number");
13        return 1;
14    }
15
16    ros::NodeHandle nh; // Node handle declaration for communication with ROS system
17
18    // Declares service client 'ros_tutorials_service_client'
19    // using the 'SrvTutorial' service file in the 'ros_tutorials_service' package.
20    // The service name is 'ros_tutorial_srv'
21    ros::ServiceClient ros_tutorials_service_client = nh.serviceClient<ros_tutorials_service::SrvTutorial>("ros_tutorial_srv");
22
23    // Declares the 'srv' service that uses the 'SrvTutorial' service file
24    ros_tutorials_service::SrvTutorial srv;
25
26    // Parameters entered when the node is executed as a service request value are stored at 'a' and 'b'
27    srv.request.a = atol(argv[1]);
28    srv.request.b = atol(argv[2]);
29
30    // Request the service. If the request is accepted, display the response value
31    if (ros_tutorials_service_client.call(srv))
32    {
33        ROS_INFO("send srv, srv.Request.a and b: %ld, %ld", (long int)srv.request.a, (long int)srv.request.b);
34        ROS_INFO("receive srv, srv.Response.result: %ld", (long int)srv.response.result);
35    }
36    else
37    {
38        ROS_ERROR("Failed to call service ros_tutorial_srv");
39        return 1;
40    }
41    return 0;
42 }
```