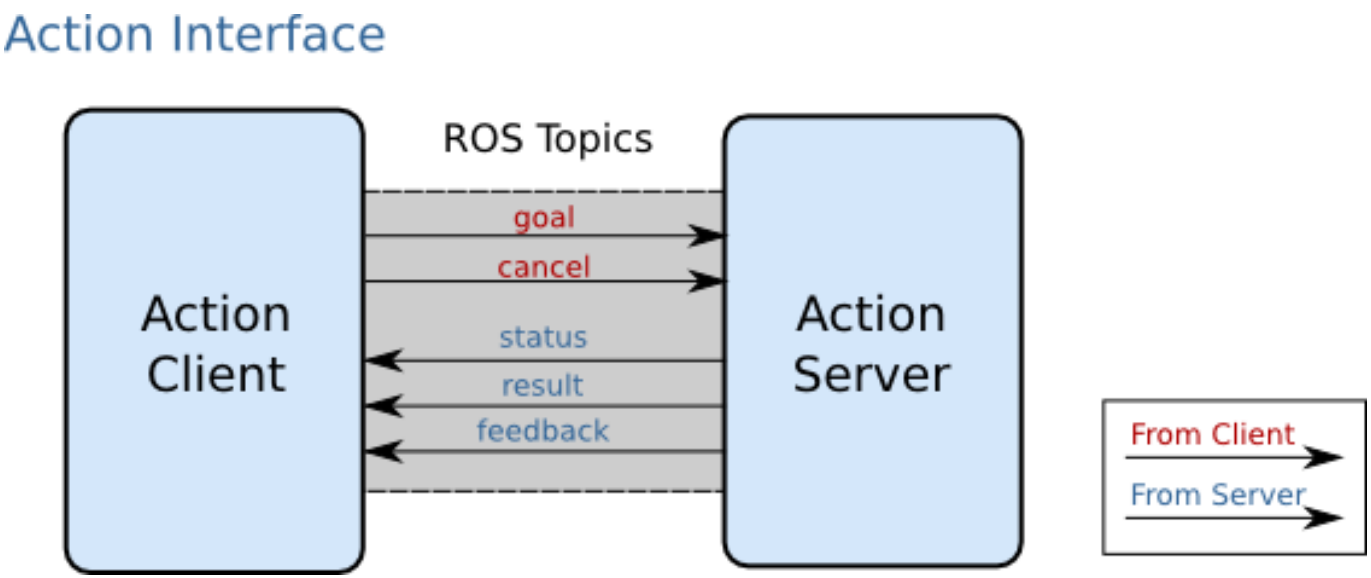
無人載具技術與應用
ROS Action

徐瑋隆
wlhsu304@gmail.com

ROS ACTION

ros action



- goal - client端傳送給server端要執行的任務
- cancel - 若任務進行時間過長，可以傳送取消該任務的指令(由client端發送)
- status - server端發給client關於此server的狀態(pending, active, recalling等)
- feedback - server定期發給client任務的進行狀況
- result - server發給client端任務完成的結果

ros action

Topics

- 應該被使用於連續性的資料流傳遞，例如相機的影像、機器的狀態等。
- 資料流應該要可以被任何獨立的sender/receiver傳遞，支援多對多的串接。
- subscriber接收資料即呼叫callback function進行處理，而publisher應可決定何時發送資料。

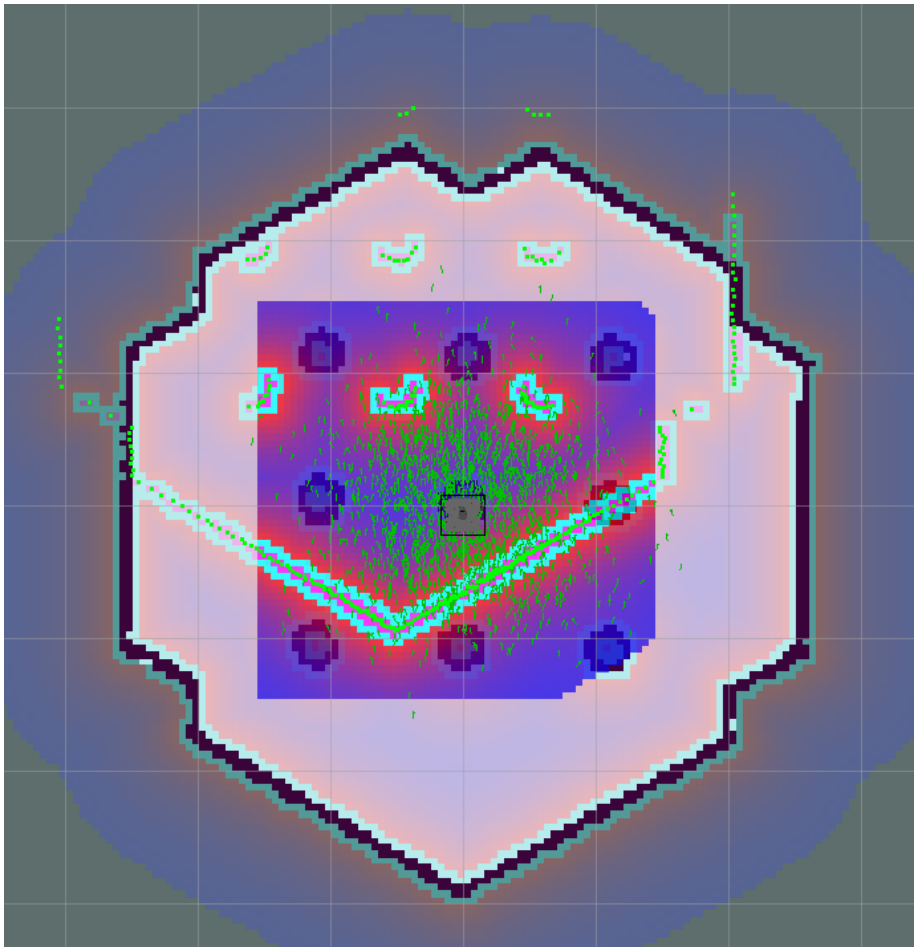
Services

- 應使用於外部呼叫，且呼叫的過程很短暫的情況，例如查表、計算等。
- 不應使用在需要長時間進行處理的作業，否則該作業應該要能被中斷(也就是action的情況)。
- 使用簡易的blocking 進行呼叫，最常使用在需要快速處理比對資料的作業。
- client端發送request，server端接收並回傳response。

Actions

- 應使用於分散式的狀態機處理，並且需要在執行請求的過程中持續發送feedback。
- 跟service很像但不一樣的地方就是他可以因為一些條件而被中斷作業。
- 每個任務的life cycle是獨立的，若有兩個goal在同一個action server上被執行，可以根據它產生的id獨立作業。
- 最好使用的時機為需要花費數秒的工作，或是初始化一些底層控制的模式的時候，可以監聽狀態以及設定終止條件的情況。
- 使用分散式的non-blocking方式做處理，更符合真實世界中作動的感覺，以自駕車為例，車子在往前開的時候若遇障礙物，則需要中斷本來的路線規劃，並執行停止/re-routing的任務。

ros action – amcl 01



roslaunch turtlebot3_gazebo turtlebot3_world.launch

roslaunch turtlebot3_navigation turtlebot3_navigation.launch

ros action – amcl 02

```
user@user-virtual-machine:~$ cd catkin_ws/
user@user-virtual-machine:~/catkin_ws$ ls
build  devel  kk.yaml  src  u123456.pgm  u123456.yaml
user@user-virtual-machine:~/catkin_ws$ rosparam dump a.yaml
user@user-virtual-machine:~/catkin_ws$ gedit a.yaml &
[1] 3186
user@user-virtual-machine:~/catkin_ws$
```

Open | a.yaml ~/catkin_ws | Save

*Untitled Document 1 | a.yaml

```yaml
amcl:
  base_frame_id: base_footprint
  beam_skip_distance: 0.5
  beam_skip_threshold: 0.3
  do_beamskip: false
  first_map_only: false
  force_update_after_initialpose: false
  force_update_after_set_map: false
  global_frame_id: map
  gui_publish_rate: 50.0
  initial_cov_aa: 0.002677220579005033 7
  initial_cov_xx: 0.003302086882758015
  initial_cov_yy: 0.0029586311637792018
  initial_pose_a: 0.4416592204537845
  initial_pose_x: -1.410978432515672
  initial_pose_y: 0.5630480624930594
  kld_err: 0.02
  kld_z: 0.99
  laser_lambda_short: 0.1
  laser_likelihood_max_dist: 2.0
  laser_max_beams: 180
  laser_max_range: 3.5
  laser_min_range: -1.0
  laser_model_type: likelihood_field
  laser_sigma_hit: 0.2
```

rostopic list
rosservice list
rosparam list
rosparam get /
rosparam dump a.yaml

ros action – amcl 03

rostopic list

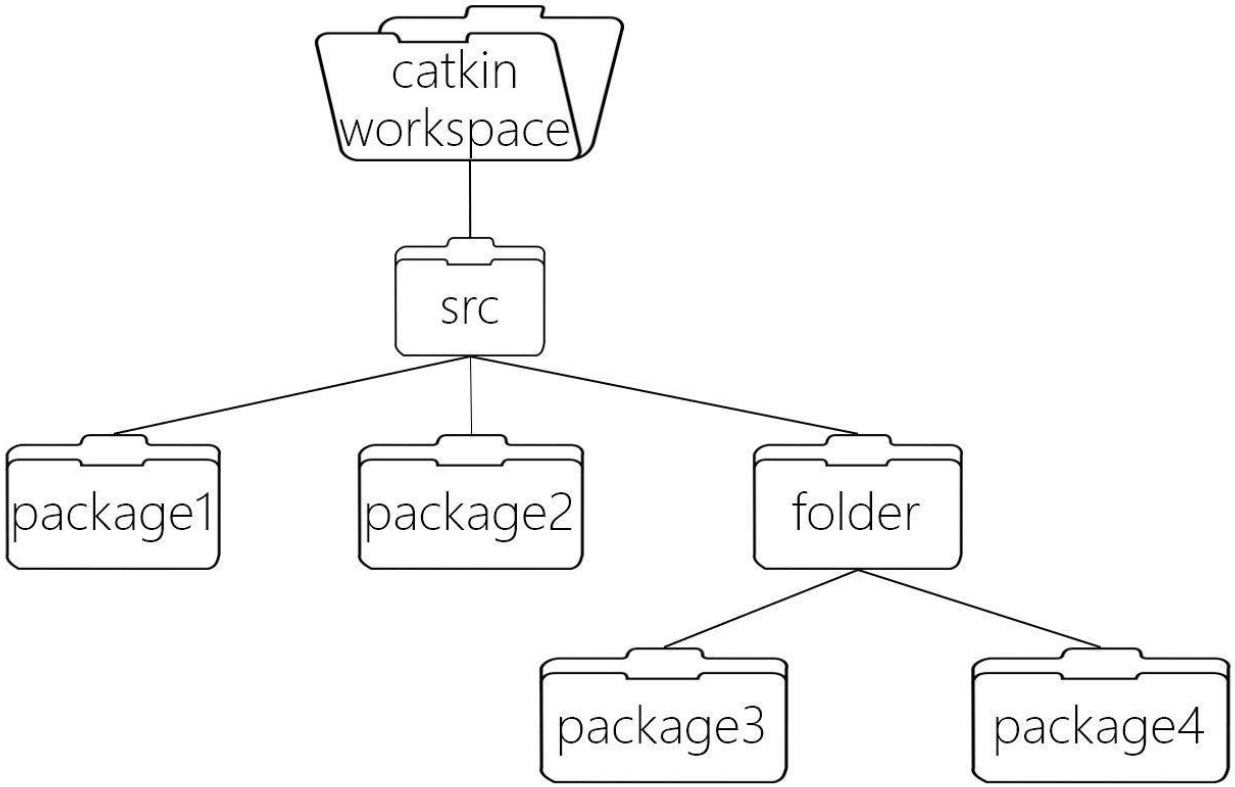
rostopic pub /move_base/goal 

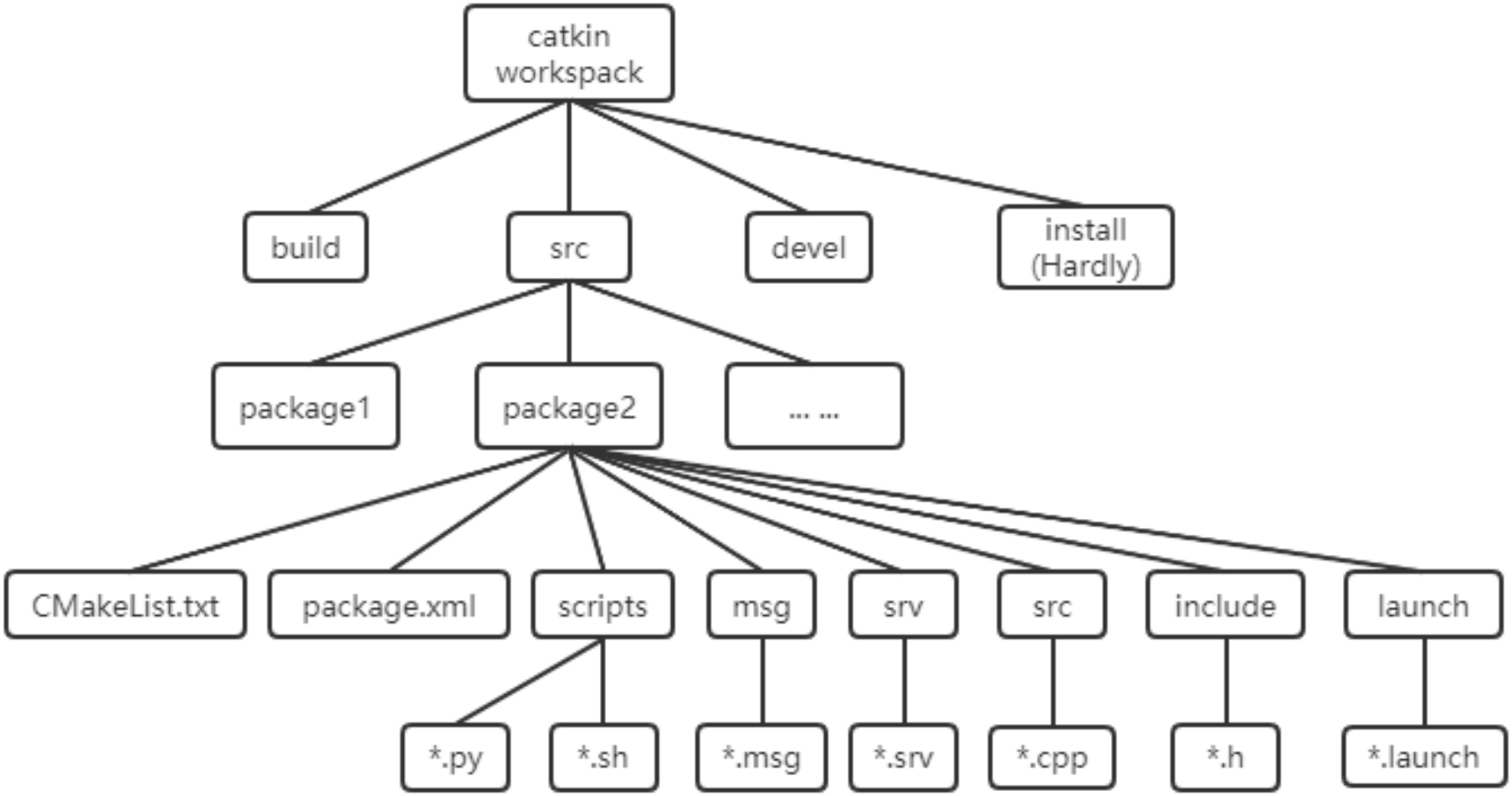
position: x: -2.0 y: 0.5 z: 0.0 orientation: x: 0.0 y: 0.0 z: 0.0 w: 1.0 “

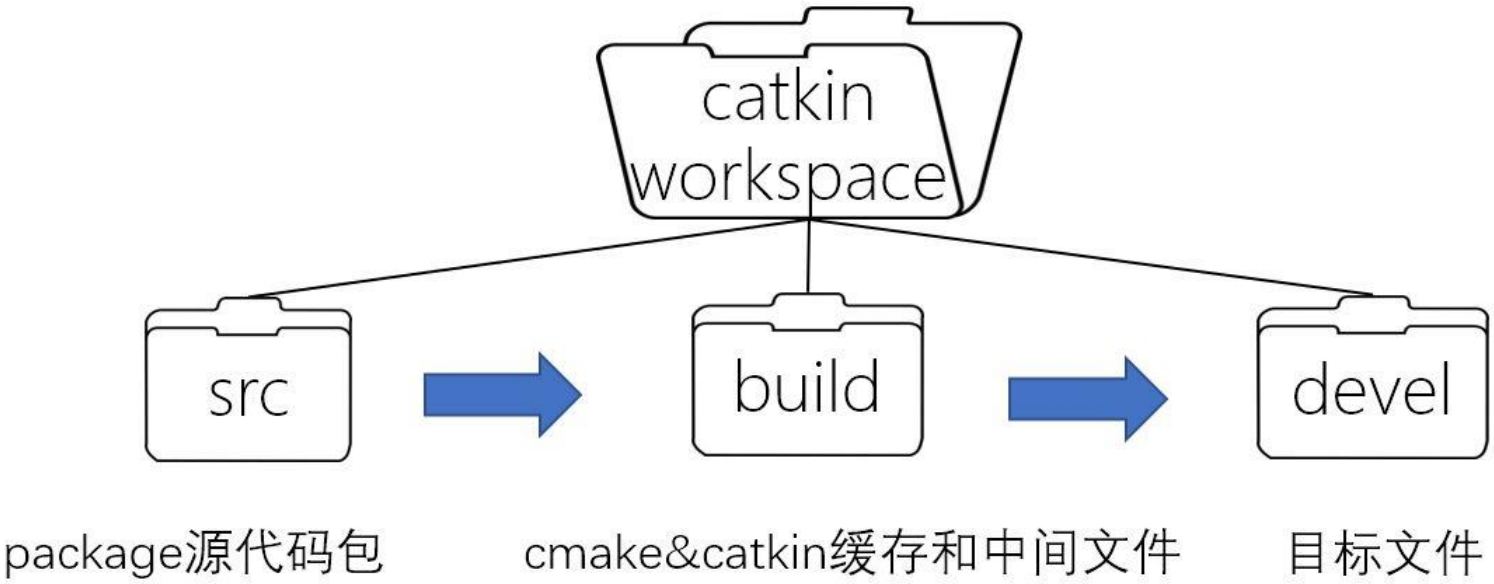
position: x: -2.0 y: -1.0 z: 0.0 orientation: x: 0.0 y: 0.0 z: 0.0 w: 1.0"

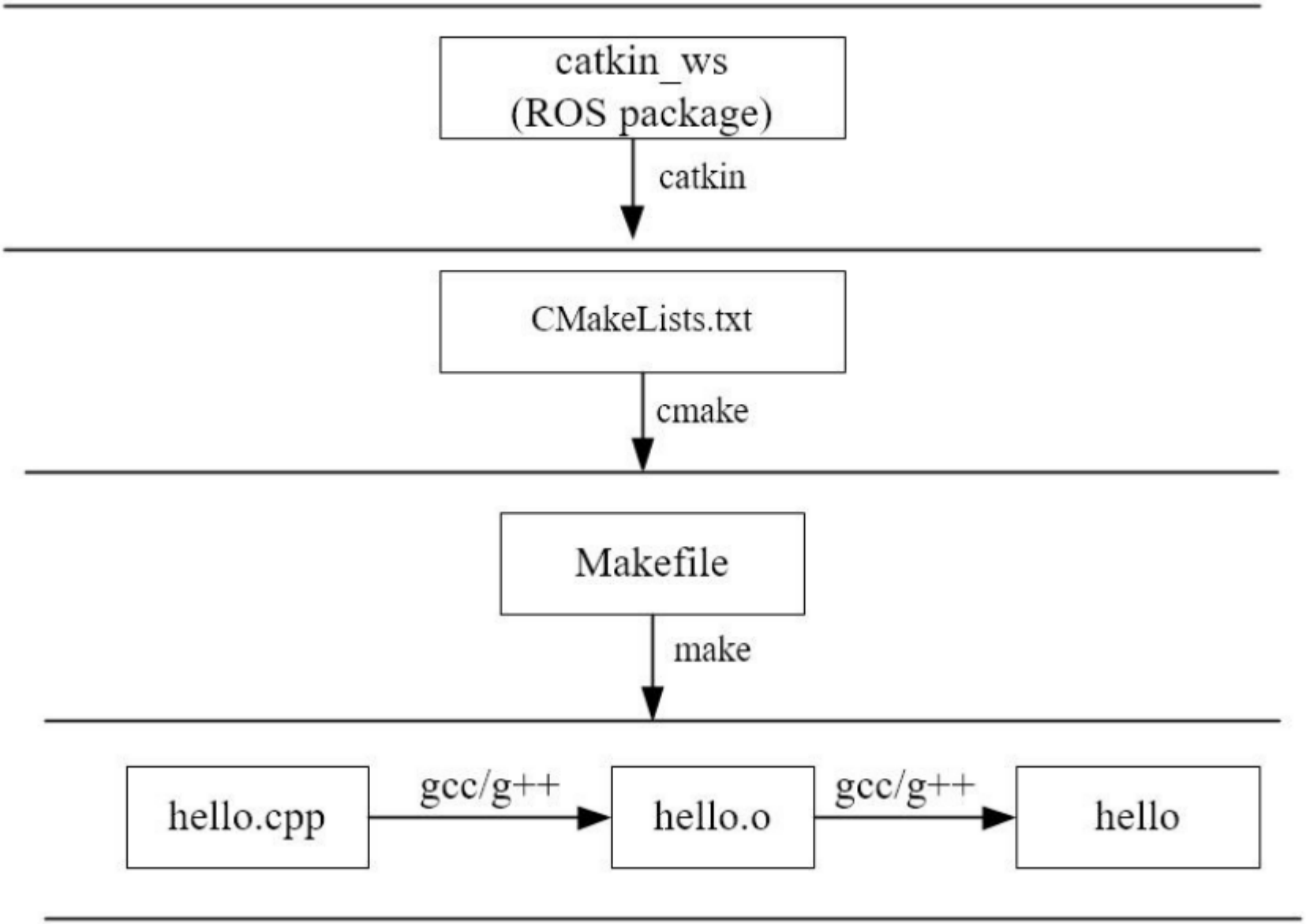
ros node coding

ROS ACTION


catkin
workspace
src
package1
package2
folder
package3
package4


catkin workspack
build
src
devel
install (Hardly)
package1
package2
... ...
CMakeList.txt
package.xml
scripts
msg
srv
src
include
launch
*.py
*.sh
*.msg
*.srv
*.cpp
*.h
*.launch


catkin
workspace
src
build
devel
package源代码包
cmake&catkin缓存和中间文件
目标文件

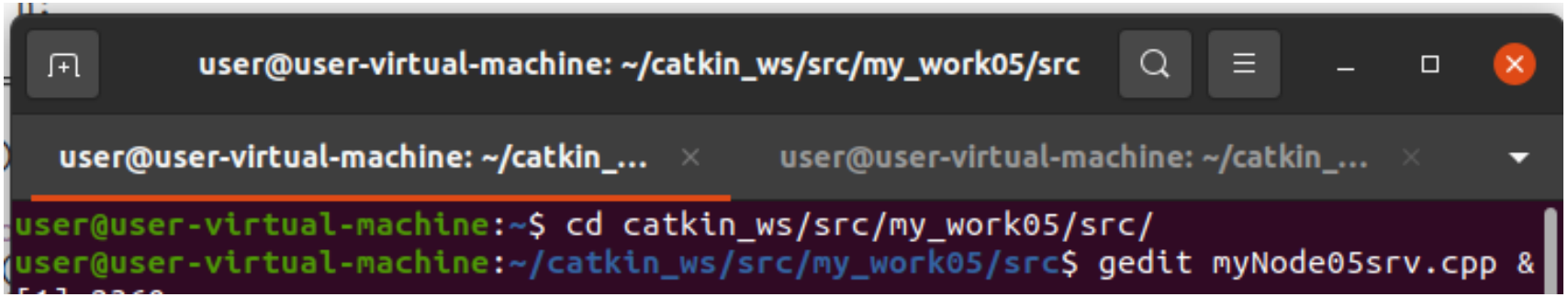

catkin_ws
(ROS package)
catkin
CMakeLists.txt
cmake
Makefile
make
hello.cpp
gcc/g++
hello.o
gcc/g++
hello

ros node 01

```
user@user-virtual-machine:~/catkin_ws/src$ catkin_create_pkg my_work05 roscpp ro
spy std_msgs actionlib_msgs actionlib
Created file my_work05/package.xml
Created file my_work05/CMakeLists.txt
Created folder my_work05/include/my_work05
Created folder my_work05/src
Successfully created files in /home/user/catkin_ws/src/my_work05. Please adjust
the values in package.xml.
user@user-virtual-machine:~/catkin_ws/src$ ls
CMakeLists.txt  my_work01  my_work02  my_work03  my_work04  my_work05
user@user-virtual-machine:~/catkin_ws/src$
```

cd catkin_ws/src/
catkin_create_pkg my_work05 roscpp rospy std_msgs actionlib_msgs actionlib
ls

ros node 02




```
#include "ros/ros.h"

int main(int argc, char **argv)
{
  ros::init(argc, argv, "myNode05srv");
  ros::NodeHandle n;

  ROS_INFO("myNode05srv: hi");
  ros::spin();

  return 0;
}
```

ros node 03

```
user@user-virtual-machine:~/catkin_ws/src/my_work05/src$ cd ..
user@user-virtual-machine:~/catkin_ws/src/my_work05$ gedit CMakeLists.txt &
[2] 2372
```

Open | *CMakeLists.txt ~/catkin_ws/src/my_work05 | Save

myNode05srv.cpp | *CMakeLists.txt

```
## Specify additional locations of header files
## Your package locations should be listed before other locations
include_directories(
# include
  ${catkin_INCLUDE_DIRS}
)

add_executable(myNode05srv src/myNode05srv.cpp)
target_link_libraries(myNode05srv ${catkin_LIBRARIES})
```

add_executable(myNode05srv src/myNode05srv.cpp)
target_link_libraries(myNode05srv ${catkin_LIBRARIES})

```
user@user-virtual-machine:~/catkin_ws$ catkin_make
```

cd ~/catkin_ws
catkin_make 編譯程式

ros node 04

```
user@user-virtual-machine:~/catkin_ws$ rosrun my_work05 myNode05srv
[ INFO] [1682332237.128378698]: myNode05srv: hi
```

roscore
rosrun my_work05 myNode05srv

ros action coding

ACTION

ros action 01

ros_tutorials_action/src/action_server.cpp https://github.com/Waywrong/ros_course/

ros_course/action_server.cpp at

github.com/Waywrong/ros_course/blob/main/ros_tutorials_action/src/action_server.cpp

Search or jump to... / Pull requests Issues Codespaces Marketplace Explore

Waywrong / ros_course Public

Pin Unwatch 2 Fork 0 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Beta Try the new code view

main ros_course / ros_tutorials_action / src / action_server.cpp Go to file

Waywrong my_work05 Latest commit ce7853f 5 minutes ago History

1 contributor

Executable File 87 lines (75 sloc) 3.35 KB Raw Blame

```cpp
#include <ros/ros.h>                                // ROS Default Header File
#include <actionlib/server/simple_action_server.h>// action Library Header File
#include <ros_tutorials_action/FibonacciAction.h> // FibonacciAction Action File Header

class FibonacciAction
{
protected:

  ros::NodeHandle nh_;  // Node handle declaration
  actionlib::SimpleActionServer<ros_tutorials_action::FibonacciAction> as_; // Action server declaration, NodeHandle instance must be created bef
  std::string action_name_; // Use as action name
  // Create messages that are used to published feedback/result
  ros_tutorials_action::FibonacciFeedback feedback_;
  ros_tutorials_action::FibonacciResult result_;

```

ros action 02

user@user-virtual-machine: ~/catkin_ws/src/my_work05/src

```
user@user-virtual-machine:~/catkin_ws/src/my_work05/src$ gedit myNode05srv.cpp &
```

Open | *myNode05srv.cpp ~/catkin_ws/src/my_work05/src | Save

*myNode05srv.cpp | action_server.cpp

```
1 #include "ros/ros.h"
2 #include <actionlib/server/simple_action_server.h>// action Library Header File
3 #include <my_work05/FibonacciAction.h> // FibonacciAction Action File Header
4 class FibonacciAction
5 {
6 protected:
7
8   ros::NodeHandle nh_;  // Node handle declaration
9   actionlib::SimpleActionServer<my_work05::FibonacciA
  NodeHandle instance must be created before this line.
10  std::string action_name_; // Use as action name
11  // Create messages that are used to published feedb
12  my_work05::FibonacciFeedback feedback_;
13  my_work05::FibonacciResult result_;
14
15 public:
16  // Initialize action server (Node handle, action na
17  FibonacciAction(std::string name) :
18    as_(nh_, name, boost::bind(&FibonacciAction::exec
19    action_name_(name)
20  {
21    as_.start();
22  }
23
24  ~FibonacciAction(void)
25  {
26  }
27
28  // A function that receives an action goal message and performs a specified
29  // action (in this example, a Fibonacci calculation).
30  void executeCB(const my_work05::FibonacciGoalConstPtr &goal)
31  {
32    ros::Rate r(1);       // Loop Rate: 1Hz
33    bool success = true;  // Used as a variable to store the success or failure of an action
34
```

Find and Replace

Find: ros_tutorials_action

Replace with: my_work05

- [ ] Match case
- [ ] Search backwards
- [ ] Match entire word only
- [x] Wrap around
- [ ] Regular expression

Replace All | Replace | Find

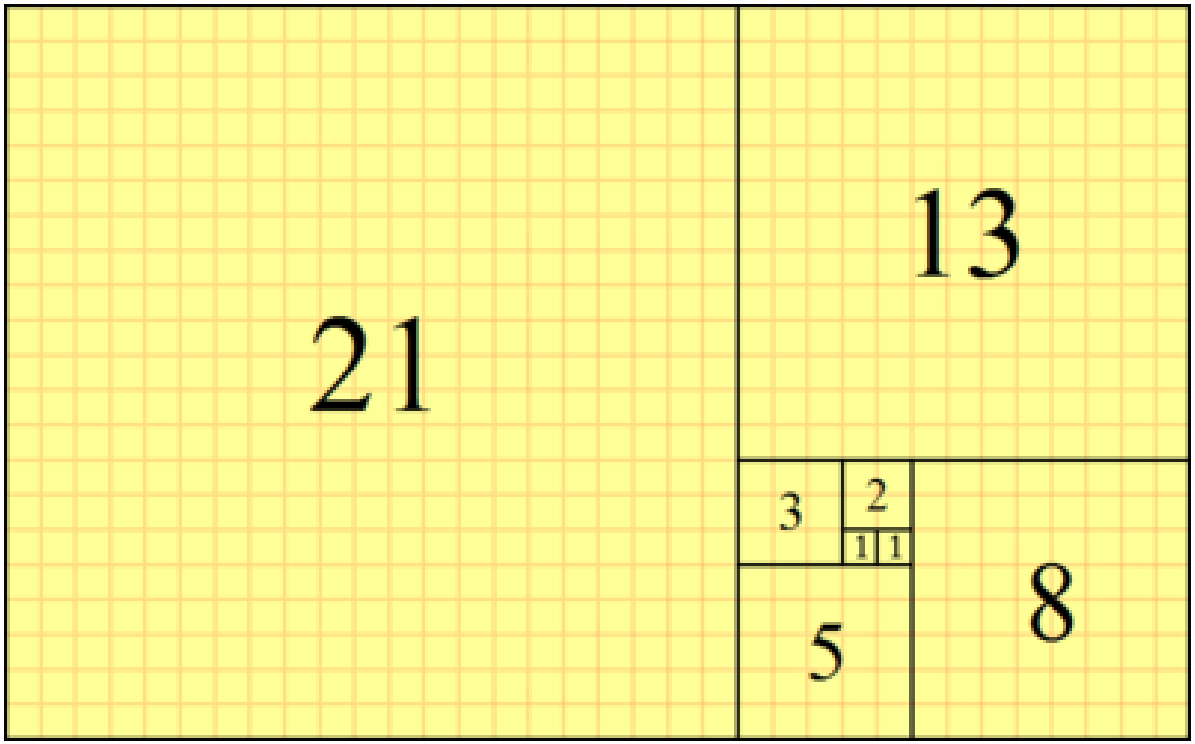
C++ | Tab Width: 8 | Ln 3, Col 20 | INS

```
#include "ros/ros.h"
#include <actionlib/server/simple_action_server.h>// action Library Header File
#include <my_work05/FibonacciAction.h> // FibonacciAction Action File Header
class FibonacciAction
{
protected:

  ros::NodeHandle nh_;  // Node handle declaration
  actionlib::SimpleActionServer<my_work05::FibonacciAction> as_;
  std::string action_name_; // Use as action name
  // Create messages that are used to published feedback/result
  my_work05::FibonacciFeedback feedback_;
  my_work05::FibonacciResult result_;

public:
  // Initialize action server (Node handle, action name, action callback function)
  FibonacciAction(std::string name) :
    as_(nh_, name, boost::bind(&FibonacciAction::executeCB, this, _1), false),
    action_name_(name)
  {
      as_.start();
  }

  ~FibonacciAction(void)
  {  }
```

```
void executeCB(const my_work05::FibonacciGoalConstPtr &goal)
 {
   ros::Rate r(1);       // Loop Rate: 1Hz
   bool success = true;  // Used as a variable to store the success or failure of an action

   // Setting Fibonacci sequence initialization,
   // add first (0) and second message (1) of feedback.
   feedback_.sequence.clear();
   feedback_.sequence.push_back(0);
   feedback_.sequence.push_back(1);

   // Notify the user of action name, goal, initial two values of Fibonacci sequence
   ROS_INFO("%s: Executing, creating fibonacci sequence of order %i with seeds %i, %i",
action_name_.c_str(), goal->order, feedback_.sequence[0], feedback_.sequence[1]);
```

```
  // Action contents
  for(int i=1; i<=goal->order; i++)
  {
    // Confirm action cancellation from action client
    if (as_.isPreemptRequested() || !ros::ok())
    {
      // Notify action cancellation
      ROS_INFO("%s: Preempted", action_name_.c_str());
      // Action cancellation and consider action as failure and save to variable
      as_.setPreempted();
      success = false;
      break;
    }
    feedback_.sequence.push_back(feedback_.sequence[i] + feedback_.sequence[i-1]);
    // publish the feedback
    as_.publishFeedback(feedback_);
    // this sleep is not necessary, the sequence is computed at 1 Hz for demonstration purposes
    r.sleep();
  }
if(success)
  {
    result_.sequence = feedback_.sequence;
    ROS_INFO("%s: Succeeded", action_name_.c_str());
    // set the action state to succeeded
    as_.setSucceeded(result_);
  }
 } };
```

```
int main(int argc, char **argv)
{
  ros::init(argc, argv, "myNode05srv");
  ros::NodeHandle n;

  ROS_INFO("myNode05srv: hi");
  FibonacciAction fibonacci("ros_tutorial_action");
  ros::spin();

  return 0;
}
```



1、1、2、3、5、8、13、21、34、55、89、144、233、377、610、987

ros action 03

https://github.com/Waywrong/ros_course/

ros_tutorials_action/action/Fibonacci.action

main ros_course / ros_tutorials_action / action / Fibonacci.action

Waywrong my_work05

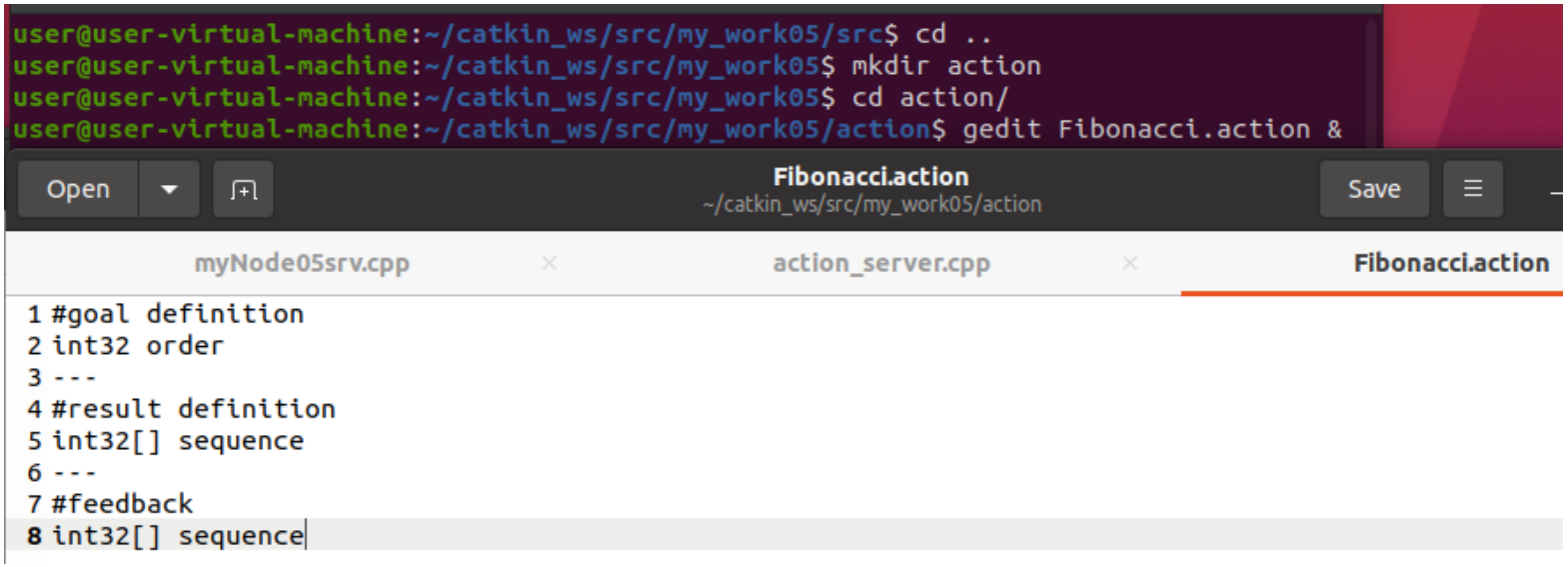
1 contributor

Executable File | 8 lines (8 sloc) | 100 Bytes

```
#goal definition
int32 order
---
#result definition
int32[] sequence
---
#feedback
int32[] sequence
```

ros action 04

cd ..
mkdir action
cd action
gedit Fibonacci.action &



#goal definition
int32 order

#result definition
int32[] sequence

#feedback
int32[] sequence

ros action 05

cd ..
gedit CMakeLists.txt &

```
user@user-virtual-machine: ~/catkin_ws/...    user@user-virtual-machine: ~/catkin_ws
user@user-virtual-machine:~/catkin_ws/src/my_work05/src$ cd ..
user@user-virtual-machine:~/catkin_ws/src/my_work05$ mkdir action
user@user-virtual-machine:~/catkin_ws/src/my_work05$ cd action/
user@user-virtual-machine:~/catkin_ws/src/my_work05/action$ gedit Fibonacci.action &
[1] 2197
user@user-virtual-machine:~/catkin_ws/src/my_work05/action$ cd ..
[1]+  Done                    gedit Fibonacci.action  (wd: ~/catkin_ws/src/my_work05/
action)
(wd now: ~/catkin_ws/src/my_work05)
user@user-virtual-machine:~/catkin_ws/src/my_work05$ gedit CMakeLists.txt
user@user-virtual-machine:~/catkin_ws/src/my_work05$ cd action/
user@user-virtual-machine:~/catkin_ws/src/my_work05/action$ cd ..
user@user-virtual-machine:~/catkin_ws/src/my_work05$ gedit CMakeLists.txt &
```

Open CMakeLists.txt ~/catkin_ws/src/my_work05 Save

myNode05srv.cpp | action_server.cpp | Fibonacci.action | CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0.2)
project(my_work05)

## 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
  actionlib
  actionlib_msgs
  roscpp
  rospy
  std_msgs
)

add_action_files(FILES Fibonacci.action)
generate_messages(DEPENDENCIES actionlib_msgs std_msgs)
```

add_action_files(FILES Fibonacci.action)
generate_messages(DEPENDENCIES actionlib_msgs std_msgs)

ros action 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/b
uild"
```

cd ~/catkin_ws
catkin_make 編譯程式

ros action 07

```
user@user-virtual-machine:~/catkin_ws$ rosrun my_work05 myNode05srv
[ INFO] [1682337220.755509649]: myNode05srv: hi
[ INFO] [1682337318.522453810]: ros_tutorial_action: Executing, creating fibonac
ci sequence of order 5 with seeds 0, 1
[ INFO] [1682337323.522832336]: ros_tutorial_action: Succeeded
```

roscore
rosrun my_work05 myNode05srv

```
user@user-virtual-machine:~$ rostopic list
/ros_tutorial_action/cancel
/ros_tutorial_action/feedback
/ros_tutorial_action/goal
/ros_tutorial_action/result
/ros_tutorial_action/status
/rosout
/rosout_agg
user@user-virtual-machine:~$ rostopic pub /ros_tutorial_action/goal my_work05/Fibonac
ciActionGoal "header:
  seq: 0
  stamp:
    secs: 0
    nsecs: 0
  frame_id: ''
goal_id:
  stamp:
    secs: 0
    nsecs: 0
  id: ''
goal:
  order: 5"
publishing and latching message. Press ctrl-C to terminate
```

rostopic list
rostopic pub /ros_tutorial_action/goal 

ros action 08

rostopic echo /ros_tutorial_action/feedback 過程時會有反饋
rostopic echo /ros_tutorial_action/result 整個算完才回報

user@user-virtual-machine: ~/catkin_ws

roscore http://user-virtual-machine:1... | user@user-virtual-machine: ~/catkin_...

```
[ 50%] Built target my_work05_generate_messages_py
[ 70%] Built target my_work05_generate_messages_cpp
[ 74%] Built target my_work05_generate_messages_lisp
[ 86%] Built target my_work05_generate_messages_nodejs
[100%] Built target my_work05_generate_messages_eus
[100%] Built target my_work05_generate_messages
[100%] Built target my_work03_generate_messages
user@user-virtual-machine:~/catkin_ws$ rosrun my_work05 myNode05srv
[ INFO] [1682337220.755509649]: myNode05srv: hi
[ INFO] [1682337318.522453810]: ros_tutorial_action: Executing, creating fibonac
ci sequence of order 5 with seeds 0, 1
[ INFO] [1682337323.522832336]: ros_tutorial_action: Succeeded
[ INFO] [1682337532.361594971]: ros_tutorial_action: Executing, creating fibonac
ci sequence of order 30 with seeds 0, 1
[ INFO] [1682337562.362080855]: ros_tutorial_action: Succeeded
[ INFO] [1682337623.697099116]: ros_tutorial_action: Executing, creating fibonac
ci sequence of order 30 with seeds 0, 1
[ INFO] [1682337653.697941342]: ros_tutorial_action: Succeeded
```

user@user-virtual-machine: ~

```
    nsecs: 0
  id: ''
goal:
  order: 30"
publishing and latching message. Press ctrl-C to terminate
^[[A^[[A^[[B^Cuser@user-virtual-machine:~$ ^C
user@user-virtual-machine:~$ rostopic pub /ros_tutorial_action/goal my_work05/Fibonac
ciActionGoal "header:
  seq: 0
  stamp:
    secs: 0
    nsecs: 0
  frame_id: ''
goal_id:
  stamp:
    secs: 0
    nsecs: 0
  id: ''
goal:
  order: 30"
publishing and latching message. Press ctrl-C to terminate
```

user@user-virtual-machine: ~

```
  sequence: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1
597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811
, 514229, 832040]
---
header:
  seq: 64
  stamp:
    secs: 1682337652
    nsecs: 697485353
  frame_id: ''
status:
  goal_id:
    stamp:
      secs: 1682337623
      nsecs: 697011281
    id: "/myNode05srv-3-1682337623.697011281"
  status: 1
  text: "This goal has been accepted by the simple action server"
feedback:
  sequence: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1
597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811
, 514229, 832040, 1346269]
---
```

user@user-virtual-machine: ~

```
user@user-virtual-machine:~$ rostopic echo /ros_tutorial_action/result
header:
  seq: 2
  stamp:
    secs: 1682337653
    nsecs: 698001935
  frame_id: ''
status:
  goal_id:
    stamp:
      secs: 1682337623
      nsecs: 697011281
    id: "/myNode05srv-3-1682337623.697011281"
  status: 3
  text: ''
result:
  sequence: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1
597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811
, 514229, 832040, 1346269]
---
```

ros action 09

ros_tutorials_action/src/action_client.cpp https://github.com/Waywrong/ros_course/

main ros_course / ros_tutorials_action / src / action_client.cpp

Waywrong my_work05

1 contributor

Executable File 36 lines (29 sloc) 1.53 KB

```cpp
#include <ros/ros.h>                            // ROS Default Header File
#include <actionlib/client/simple_action_client.h>// action Library Header File
#include <actionlib/client/terminal_state.h>      // Action Goal Status Header File
#include <ros_tutorials_action/FibonacciAction.h> // FibonacciAction Action File Header

int main (int argc, char **argv)          // Node Main Function
{
  ros::init(argc, argv, "action_client"); // Node Name Initialization

  // Action Client Declaration (Action Name: ros_tutorial_action)
  actionlib::SimpleActionClient<ros_tutorials_action::FibonacciAction> ac("ros_tutorial_action", true);

  ROS_INFO("Waiting for action server to start.");
  ac.waitForServer(); //wait for the action server to start, will wait for infinite time

  ROS_INFO("Action server started, sending goal.");
  ros_tutorials_action::FibonacciGoal goal; // Declare Action Goal
  goal.order = 20;    // Set Action Goal (Process the Fibonacci sequence 20 times)
  ac.sendGoal(goal);  // Transmit Action Goal

  // Set action time limit (set to 30 seconds)
  bool finished_before_timeout = ac.waitForResult(ros::Duration(30.0));

  // Process when action results are received within the time limit for achieving the action goal
  if (finished_before_timeout)
  {
    // Receive action target status value and display on screen
    actionlib::SimpleClientGoalState state = ac.getState();
    ROS_INFO("Action finished: %s",state.toString().c_str());
  }
  else
    ROS_INFO("Action did not finish before the time out.");

  //exit
  return 0;
}
```

ros action 10

cd catkin_ws/src/my_work05/src

gedit myNode05cli.cpp

```
user@user-virtual-machine:~/catkin_ws/src/my_work05/src$ gedit myNode05cli.cpp
user@user-virtual-machine:~/catkin_ws/src/my_work05/src$
```

myNode05srv.cpp | action_server.cpp | Fibonacci.action | CMakeLists.txt | *myNode05cli.cpp

```
1 #include <ros/ros.h>                                  // ROS Default Header File
2 #include <actionlib/client/simple_action_client.h>// action Library Header File
3 #include <actionlib/client/terminal_state.h>       // Action Goal Status Header File
4 #include <ros_tutorials_action/FibonacciAction.h> // FibonacciAction Action File Header
5
6 int main (int argc, char **argv)          // Node Main Function
7 {
8   ros::init(argc, argv, "action_client"); // Node Name Initialization
9
10  // Action Client Declaration (Action Name: ros_tutorial_action)
11  actionlib::SimpleActionClient<ros_tutorials_action::FibonacciAction> ac("ros_tutorial_action", true);
12
13  ROS_INFO("Waiting for action server to start.");
14  ac.waitForServer(); //wait for the action ser
15
16  ROS_INFO("Action server started, sending goal
17  ros_tutorials_action::FibonacciGoal goal; //
18  goal.order = 20;    // Set Action Goal (Proce
19  ac.sendGoal(goal);  // Transmit Action Goal
20
21  // Set action time limit (set to 30 seconds)
22  bool finished_before_timeout = ac.waitForResu
23
24  // Process when action results are received w
25  if (finished_before_timeout)
26  {
27    // Receive action target status value and c
28    actionlib::SimpleClientGoalState state = ac
29    ROS_INFO("Action finished: %s",state.toString().c_str());
30  }
31  else
32    ROS_INFO("Action did not finish before the time out.");
33
34  //exit
35  return 0;
36 }
```

Find and Replace

Find: ros_tutorials_action

Replace with: my_work05

- [ ] Match case
- [ ] Search backwards
- [ ] Match entire word only
- [x] Wrap around
- [ ] Regular expression

Replace All | Replace | Find

```
#include <ros/ros.h>                          // ROS Default Header File
#include <actionlib/client/simple_action_client.h>// action Library Header File
#include <actionlib/client/terminal_state.h>      // Action Goal Status Header File
#include <my_work05/FibonacciAction.h> // FibonacciAction Action File Header
int main (int argc, char **argv)         // Node Main Function
{
  ros::init(argc, argv, "action_client"); // Node Name Initialization
  actionlib::SimpleActionClient<my_work05::FibonacciAction> ac("ros_tutorial_action", true);
  ROS_INFO("Waiting for action server to start.");
  ac.waitForServer(); //wait for the action server to start, will wait for infinite time
  ROS_INFO("Action server started, sending goal.");
  my_work05::FibonacciGoal goal; // Declare Action Goal
  goal.order = 20;    // Set Action Goal (Process the Fibonacci sequence 20 times)
  ac.sendGoal(goal);  // Transmit Action Goal
  bool finished_before_timeout = ac.waitForResult(ros::Duration(30.0));

  if (finished_before_timeout)
  {
    // Receive action target status value and display on screen
    actionlib::SimpleClientGoalState state = ac.getState();
    ROS_INFO("Action finished: %s",state.toString().c_str());
  }
  else
    ROS_INFO("Action did not finish before the time out.");
return 0;
}
```

ros node 11

```
user@user-virtual-machine:~$ cd ~/catkin_ws/src/my_work05/
user@user-virtual-machine:~/catkin_ws/src/my_work05$ gedit CMakeLists.txt &
```

```
)

add_executable(myNode05srv src/myNode05srv.cpp)
target_link_libraries(myNode05srv ${catkin_LIBRARIES})

add_executable(myNode05cli src/myNode05cli.cpp)
target_link_libraries(myNode05cli ${catkin_LIBRARIES})
```

add_executable(myNode05cli src/myNode05cli.cpp)
target_link_libraries(myNode05cli ${catkin_LIBRARIES})

```
user@user-virtual-machine:~/catkin_ws$ catkin_make
```

cd ~/catkin_ws
catkin_make 編譯程式

ros action 12

```
user@user-virtual-machine: ~/catkin_ws/src/my_work05
roscore http://user-virtual-machine:1...   user@user-virtual-machine: ~/catkin_...
user@user-virtual-machine:~/catkin_ws/src/my_work05$ rosrun my_work05 myNode05sr
v
[ INFO] [1682341547.859487334]: myNode05srv: hi
[ INFO] [1682341574.664698559]: ros_tutorial_action: Executing, creating fibonac
ci sequence of order 20 with seeds 0, 1
[ INFO] [1682341594.665543735]: ros_tutorial_action: Succeeded
```

```
user@user-virtual-machine: ~/catkin_ws/src/my_work05/src
user@user-virtual-machine:~/catkin_ws/src/my_work05/src$ rosrun my_work05 myNode05cli
[ INFO] [1682341574.396979953]: Waiting for action server to start.
[ INFO] [1682341574.664228523]: Action server started, sending goal.
[ INFO] [1682341594.665973279]: Action finished: SUCCEEDED
user@user-virtual-machine:~/catkin_ws/src/my_work05/src$
```

roscore
rosrun my_work05 myNode05srv
rosrun my_work05 myNode05cli

```
user@user-virtual-machine:~/catkin_ws$ rostopic list
/ros_tutorial_action/cancel
/ros_tutorial_action/feedback
/ros_tutorial_action/goal
/ros_tutorial_action/result
/ros_tutorial_action/status
/rosout
/rosout_agg
user@user-virtual-machine:~/catkin_ws$ rostopic echo /ros_tutorial_action/feedback
header:
  seq: 0
  stamp:
    secs: 1682341574
    nsecs: 664783548
  frame_id: ''
status:
  goal_id:
    stamp:
      secs: 1682341574
      nsecs: 664299847
    id: "/action_client-1-1682341574.664299847"
  status: 1
  text: "This goal has been accepted by the simple action server"
feedback:
  sequence: [0, 1, 1]
---
```

rostopic list
rostopic echo /ros_tutorial_action/feedback

作業4

- 於本周相同專案內(my_work05)增加一個action client，目的為在AMCL範例中送出小車目標點，並上傳相關檔案

- 參考4/25上課內容，“ROS-Class-8.pdf“ , p.36-37
- https://github.com/Waywrong/ros_course/blob/main/my_work05/src/amcl_mv_cli.cpp
- 計分部分包含
  - 1. 完整性
  - 2.修改目標點座標 3次 看結果
  - 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)
    - rosnode list
    - rostopic list

- 上傳作業包含 (期末前上傳):
- 1. CMakeLists.txt
- 2. package.xml
- 3. 修改過的cpp檔
- 4. 實驗紀錄word檔

ros action – hw

roslaunch turtlebot3_gazebo turtlebot3_world.launch
roslaunch turtlebot3_navigation turtlebot3_navigation.launch
rostopic pub /move_base/goal 

position: x: -2.0 y: 0.5 z: 0.0 orientation: x: 0.0 y: 0.0 z: 0.0 w: 1.0 “

position: x: -2.0 y: -1.0 z: 0.0 orientation: x: 0.0 y: 0.0 z: 0.0 w: 1.0"

ros action – hw

my_work05/src/amcl_mv_cli.cpp

https://github.com/Waywrong/ros_course/

```
#include <ros/ros.h>                                  // ROS Default Header File
#include <actionlib/client/simple_action_client.h>// action Library Header File
#include <actionlib/client/terminal_state.h>       // Action Goal Status Header File
//#include <my_work05/FibonacciAction.h> // FibonacciAction Action File Header
#include <move_base_msgs/MoveBaseAction.h>
int main (int argc, char **argv)           // Node Main Function
{
  ros::init(argc, argv, "goal_cli"); // Node Name Initialization
  //actionlib::SimpleActionClient<my_work05::FibonacciAction> ac("ros_tutorial_action", true);
  actionlib::SimpleActionClient<move_base_msgs::MoveBaseAction> ac("move_base", true);
  ROS_INFO("Waiting for action server to start.");
  ac.waitForServer(); //wait for the action server to start, will wait for infinite time
  ROS_INFO("Action server started, sending goal.");

  move_base_msgs::MoveBaseGoal goal;
  goal.target_pose.header.frame_id = "map";
  goal.target_pose.header.stamp = ros::Time::now();

  goal.target_pose.pose.position.x = -2.0;
  goal.target_pose.pose.position.y = 0.5;
  goal.target_pose.pose.orientation.w = 1.0;
  ac.sendGoal(goal);

  bool finished_before_timeout = ac.waitForResult(ros::Duration(30.0));
  if (finished_before_timeout)
  {
    // Receive action target status value and display on screen
    actionlib::SimpleClientGoalState state = ac.getState();
    ROS_INFO("Action finished: %s",state.toString().c_str());
  }
  else
    ROS_INFO("Action did not finish before the time out.");
  return 0;
}
```