



ROS

آشنایی با ROS

جلسه سوم : Customization در ROS



balazadeh.elnaz@gmail.com



پیاده سازی و کار با Concept ها

Client node

```
1.  #!/usr/bin/python3
2.
3.  import rospy
4.  from turtlesim.srv import Spawn
5.  import sys
6.
7.
8.  def spawner(x, y, theta, name):
9.      rospy.wait_for_service('/spawn')
10.     try:
11.         Spawner = rospy.ServiceProxy('/spawn', Spawn)
12.         resp = Spawner(x, y, theta, name)
13.         return resp.name
14.     except rospy.ServiceException as e:
15.         print(f"Service call failed: {s}"%e)
16.
17.
18. if __name__ == "__main__":
19.     try:
20.         if len(sys.argv) == 5:
21.             x = int(sys.argv[1])
22.             y = int(sys.argv[2])
23.             theta = int(sys.argv[3])
24.             name = str(sys.argv[4])
25.         else:
26.             sys.exit(1)
27.         print("Turtle Successfully added: ", spawner(x, y, theta, name))
28.     except rospy.ROSInterruptException:
29.         pass
```



پیاده سازی و کار با Concept ها

اجرای Client node

```
$ rosrun [package_name] [node_name]  
$ rosrun my_package turtlesim_client.py
```



Customize کردن Concept ها

ساخت Message دلخواه

```
$ mkdir ~/catkin_ws/src/demo_pkg/msg/
```

```
$ > demo_msg.msg
```

```
# msg_type msg_name
```

1. ساخت دایرکتوری msg

2. ساخت فایل msg

3. تعریف نوع msg



Customize کردن Concept ها

ایجاد تغییرات در فایل package.xml

```
<build_depend>message_generation</build_depend>  
<exec_depend>message_runtime</exec_depend>
```

← Uncomment



Customize کردن Concept ها

ایجاد تغییرات در فایل CMakeLists.txt

```
find_package(catkin REQUIRED  
  
COMPONENTS  
  
    roscpp  
  
    rospy  
  
    std_msgs  
  
    message_generation  
)
```

```
1 cmake_minimum_required(VERSION 3.0.2)
2 project(demo_pkg)
3
4 ## Compile as C++11, supported in ROS Kinetic and newer
5 # add_compile_options(-std=c++11)
6
7 ## Find catkin macros and libraries
8 ## if COMPONENTS list like find_package(catkin REQUIRED COMPONENTS xyz)
9 ## is used, also find other catkin packages
10 find_package(catkin REQUIRED COMPONENTS
11   roscpp
12   std_msgs
13 )
14
15 ## System dependencies are found with CMake's conventions
16 # find_package(Boost REQUIRED COMPONENTS system)
17
18
19 ## Uncomment this if the package has a setup.py. This macro ensures
20 ## modules and global scripts declared therein get installed
21 ## See http://ros.org/doc/api/catkin/html/user\_guide/setup\_dot\_py.html
22 # catkin_python_setup()
23
24 #####
25 ## Declare ROS messages, services and actions ##
26 #####
27
28 ## To declare and build messages, services or actions from within this
29 ## package, follow these steps:
30 ## * Let MSG_DEP_SET be the set of packages whose message types you use in
31 ##   your messages/services/actions (e.g. std_msgs, actionlib_msgs, ...).
32 ## * In the file package.xml:
33 ##   * add a build_depend tag for "message_generation"
34 ##   * add a build_depend and a exec_depend tag for each package in MSG_DEP_SET
35 ##   * If MSG_DEP_SET isn't empty the following dependency has been pulled in
36 ##     but can be declared for certainty nonetheless:
37 ##     * add a exec_depend tag for "message_runtime"
38 ## * To this file (CMakeLists.txt):
```



Customize کردن Concept ها

ایجاد تغییرات در فایل CMakeLists.txt

```
add_message_files(  
  FILES  
  demo.msg  
)
```



Customize کردن Concept ها

ایجاد تغییرات در فایل CMakeLists.txt

```
generate_messages(  
  DEPENDENCIES  
  std_msgs  
)
```




Customize کردن Concept ها

و قدم آخر ...

```
$ cd ~/catkin_ws/  
$ catkin_make
```



Customize کردن Concept ها

ساخت Service دلخواه

```
$ mkdir ~/catkin_ws/src/demo_pkg/srv/
```

```
$ > demo.srv
```

```
# Request
int64 B
---
# Response
int64 Sum
```

1. ساخت دایرکتوری `srv`

2. ساخت فایل `srv`

3. تعریف نوع `srv`



Customize کردن Concept ها

ایجاد تغییرات در فایل package.xml

```
<build_depend>message_generation</build_depend>  
<exec_depend>message_runtime</exec_depend>
```

← Uncomment



Customize کردن Concept ها

ایجاد تغییرات در فایل CMakeLists.txt

```
find_package(catkin REQUIRED  
  
COMPONENTS  
  
    roscpp  
  
    rospy  
  
    std_msgs  
  
    message_generation  
)
```

```
1 cmake_minimum_required(VERSION 3.0.2)
2 project(demo_pkg)
3
4 ## Compile as C++11, supported in ROS Kinetic and newer
5 # add_compile_options(-std=c++11)
6
7 ## Find catkin macros and libraries
8 ## if COMPONENTS list like find_package(catkin REQUIRED COMPONENTS xyz)
9 ## is used, also find other catkin packages
10 find_package(catkin REQUIRED COMPONENTS
11   roscpp
12   std_msgs
13 )
14
15 ## System dependencies are found with CMake's conventions
16 # find_package(Boost REQUIRED COMPONENTS system)
17
18
19 ## Uncomment this if the package has a setup.py. This macro ensures
20 ## modules and global scripts declared therein get installed
21 ## See http://ros.org/doc/api/catkin/html/user\_guide/setup\_dot\_py.html
22 # catkin_python_setup()
23
24 #####
25 ## Declare ROS messages, services and actions ##
26 #####
27
28 ## To declare and build messages, services or actions from within this
29 ## package, follow these steps:
30 ## * Let MSG_DEP_SET be the set of packages whose message types you use in
31 ##   your messages/services/actions (e.g. std_msgs, actionlib_msgs, ...).
32 ## * In the file package.xml:
33 ##   * add a build_depend tag for "message_generation"
34 ##   * add a build_depend and a exec_depend tag for each package in MSG_DEP_SET
35 ##   * If MSG_DEP_SET isn't empty the following dependency has been pulled in
36 ##     but can be declared for certainty nonetheless:
37 ##     * add a exec_depend tag for "message_runtime"
38 ## * To this file (CMakeLists.txt):
```



Customize کردن Concept ها

ایجاد تغییرات در فایل CMakeLists.txt

```
add_service_files(  
  FILES  
  demo.srv  
)
```



Customize کردن Concept ها

ایجاد تغییرات در فایل CMakeLists.txt

```
generate_messages(  
  DEPENDENCIES  
  std_msgs  
)
```



Customize کردن Concept ها

و قدم آخر ...

```
$ cd ~/catkin_ws/  
$ catkin_make
```



Launch Files

لانیچ فایل برای اجرای همزمان چند node و یا تنظیم پارامترها به کار می رود.

```
<launch>
    <node pkg="package_name" type="File_name" name="node_name"
output="screen" />
    <node pkg="package_name" type="File_name" name="node_name" />
    .
    .
    .
</launch>
```




ROS

ممنون از توجه شما