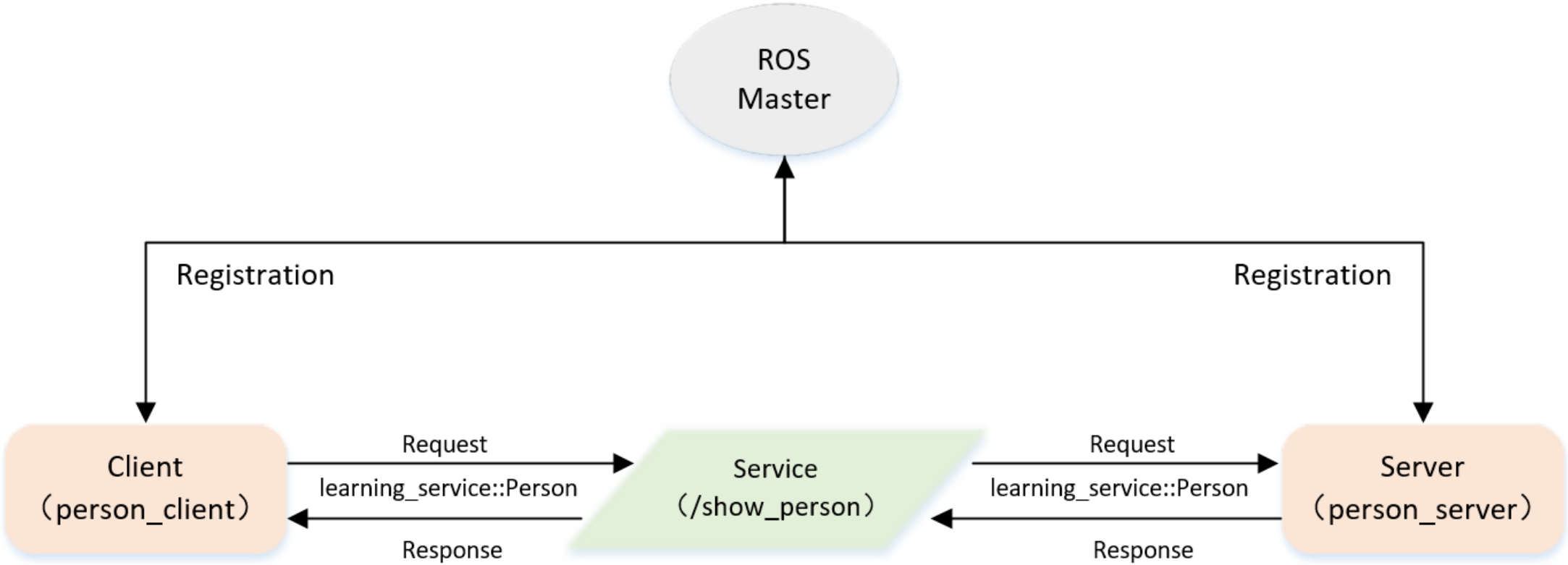
古月居 GYH.AI ROS

ROS入门
21讲

15.服务数据的定义与使用

主讲人：古月

服务模型



服务模型（服务端/客户端）

如何自定义服务数据

```
string name
uint8  age
uint8  sex

uint8 unknown = 0
uint8 male    = 1
uint8 female  = 2
---
string result
```

Person.srv

- **定义srv文件；**
- **在package.xml中添加功能包依赖**

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

- **在CMakeLists.txt添加编译选项**
  - find_package(…… message_generation)
  - add_service_files(FILES Person.srv)
    generate_messages(DEPENDENCIES std_msgs)
  - catkin_package(…… message_runtime)
- **编译生成语言相关文件**

创建服务器代码（C++）

```cpp
/**
 * 该例程将执行/show_person服务，服务数据类型learning_service::Person
 */

#include <ros/ros.h>
#include "learning_service/Person.h"

// service回调函数，输入参数req，输出参数res
bool personCallback(learning_service::Person::Request  &req,
                    learning_service::Person::Response &res)
{
    // 显示请求数据
    ROS_INFO("Person: name:%s  age:%d  sex:%d", req.name.c_str(), req.age, req.sex);

    // 设置反馈数据
    res.result = "OK";

    return true;
}

int main(int argc, char **argv)
{
    // ROS节点初始化
    ros::init(argc, argv, "person_server");

    // 创建节点句柄
    ros::NodeHandle n;

    // 创建一个名为/show_person的server，注册回调函数personCallback
    ros::ServiceServer person_service = n.advertiseService("/show_person", personCallback);

    // 循环等待回调函数
    ROS_INFO("Ready to show person informtion.");
    ros::spin();

    return 0;
}
```

person_server.cpp

如何实现一个服务器

- 初始化ROS节点；
- 创建Server实例；
- 循环等待服务请求，进入回调函数；
- 在回调函数中完成服务功能的处理，并反馈应答数据。

创建客户端代码（C++）

```cpp
/**
 * 该例程将请求/show_person服务，服务数据类型learning_service::Person
 */

#include <ros/ros.h>
#include "learning_service/Person.h"

int main(int argc, char** argv)
{
    // 初始化ROS节点
    ros::init(argc, argv, "person_client");

    // 创建节点句柄
    ros::NodeHandle node;

    // 发现/spawn服务后，创建一个服务客户端，连接名为/spawn的service
    ros::service::waitForService("/show_person");
    ros::ServiceClient person_client = node.serviceClient<learning_service::Person>("/show_person");

    // 初始化learning_service::Person的请求数据
    learning_service::Person srv;
    srv.request.name = "Tom";
    srv.request.age  = 20;
    srv.request.sex  = learning_service::Person::Request::male;

    // 请求服务调用
    ROS_INFO("Call service to show person[name:%s, age:%d, sex:%d]",
             srv.request.name.c_str(), srv.request.age, srv.request.sex);

    person_client.call(srv);

    // 显示服务调用结果
    ROS_INFO("Show person result : %s", srv.response.result.c_str());

    return 0;
};
```

person_client.cpp

如何实现一个客户端

- 初始化ROS节点；
- 创建一个Client实例；
- 发布服务请求数据；
- 等待Server处理之后的应答结果。

配置服务器/客户端代码编译规则

```
## 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/learning_service_node.cpp)

## Specify libraries to link a library or executable target against
# target_link_libraries(${PROJECT_NAME}_node
#   ${catkin_LIBRARIES}
# )

## 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})

add_executable(person_server src/person_server.cpp)
target_link_libraries(person_server ${catkin_LIBRARIES})
add_dependencies(person_server ${PROJECT_NAME}_gencpp)

add_executable(person_client src/person_client.cpp)
target_link_libraries(person_client ${catkin_LIBRARIES})
add_dependencies(person_client ${PROJECT_NAME}_gencpp)
```

如何配置CMakeLists.txt中的编译规则

- 设置需要编译的代码和生成的可执行文件；
- 设置链接库；
- 添加依赖项。

```
add_executable(person_server src/person_server.cpp)
target_link_libraries(person_server ${catkin_LIBRARIES})
add_dependencies(person_server ${PROJECT_NAME}_gencpp)

add_executable(person_client src/person_client.cpp)
target_link_libraries(person_client ${catkin_LIBRARIES})
add_dependencies(person_client ${PROJECT_NAME}_gencpp)
```

CMakeLists.txt

编译并运行客户端和服务端

```
$ cd ~/catkin_ws
$ catkin_make
$ source devel/setup.bash
$ roscore
$ rosrun learning_service person_server
$ rosrun learning_service person_client
```

```
hcx@hcx-vpc:~/catkin_ws$ rosrun learning_service person_server
[ INFO] [1562234385.473929292]: Ready to show person informtion.
[ INFO] [1562234405.584154235]: Person: name:Tom  age:20  sex:1
[ INFO] [1562234411.809871741]: Person: name:Tom  age:20  sex:1
```

```
hcx@hcx-vpc:~/catkin_ws$ rosrun learning_service person_client
[ INFO] [1562234405.582071660]: Call service to show person[name:Tom, age:20, se
x:1]
[ INFO] [1562234405.584514656]: Show person result : OK
hcx@hcx-vpc:~/catkin_ws$ rosrun learning_service person_client
[ INFO] [1562234411.808122249]: Call service to show person[name:Tom, age:20, se
x:1]
[ INFO] [1562234411.810180819]: Show person result : OK
```

创建客户端和服务端代码（Python）

古月居 GYH.AI

person_server.py

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# 该例程将执行/show_person服务，服务数据类型learning_service::Person

import rospy
from learning_service.srv import Person, PersonResponse

def personCallback(req):
    # 显示请求数据
    rospy.loginfo("Person: name:%s  age:%d  sex:%d", req.name, req.age, req.sex)

    # 反馈数据
    return PersonResponse("OK")

def person_server():
    # ROS节点初始化
    rospy.init_node('person_server')

    # 创建一个名为/show_person的server，注册回调函数personCallback
    s = rospy.Service('/show_person', Person, personCallback)

    # 循环等待回调函数
    print "Ready to show person informtion."
    rospy.spin()

if __name__ == "__main__":
    person_server()
```

person_client.py

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# 该例程将请求/show_person服务，服务数据类型learning_service::Person

import sys
import rospy
from learning_service.srv import Person, PersonRequest

def person_client():
    # ROS节点初始化
    rospy.init_node('person_client')

    # 发现/spawn服务后，创建一个服务客户端，连接名为/spawn的service
    rospy.wait_for_service('/show_person')
    try:
        person_client = rospy.ServiceProxy('/show_person', Person)

        # 请求服务调用，输入请求数据
        response = person_client("Tom", 20, PersonRequest.male)
        return response.result
    except rospy.ServiceException, e:
        print "Service call failed: %s"%e

if __name__ == "__main__":
    #服务调用并显示调用结果
    print "Show person result : %s" %(person_client())
```

感谢观看

怕什么真理无穷，进一寸有一寸的欢喜

更多精彩，欢迎关注

古月居

古月春旭

bilibili 古月居GYH

ROS入门
21讲