

## 第十三课 编写服务器和客户端

### 1. 编写 Service 节点

这里，我们将创建一个简单的 service 节点("add\_two\_ints\_server")，该节点将接收到两个整形数字，并返回它们的和。进入先前教程中所创建的 beginner\_tutorials 包所在的目录：

```
cd ~/catkin_ws/src/beginner_tutorials/src
touch add_two_ints_server.cpp
```

在 beginner\_tutorials 包中创建 src/add\_two\_ints\_server.cpp 文件，并添加以下代码：

```
#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;
}
```

代码解析：

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

beginner\_tutorials/AddTwoInts.h 是由编译系统自动根据我们先前创建的 srv 文件生成的对应该 srv 文件的头文件。

```
bool add(beginner_tutorials::AddTwoInts::Request &req, beginner_tutorials::AddTwoInts::Response &res)
```

这个函数提供两个 int 值求和的服务，int 值从 request 里面获取，而返回数据装入 response 内，这些数据类型都定义在 srv 文件内部，函数返回一个 boolean 值。

```
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);
```

现在，两个 int 值已经相加，并存入了 response。然后一些关于 request 和 response 的信息被记录下来。最后，service 完成计算后返回 true 值。

```
ros::ServiceServer service = n.advertiseService("add_two_ints", add)
```

这里，service 已经建立起来，并在 ROS 内发布出来。

## 2. 编写 Client 节点

在 `beginner_tutorials` 包中创建 `src/add_two_ints_client.cpp` 文件，并复制粘贴下面的代码：

```
cd ~/catkin_ws/src/beginner_tutorials/src
touch add_two_ints_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;
}
```

代码解析：

```
ros::ServiceClient client = n.serviceClient<beginner_tutorials::AddTwoInts>("add_two_ints");
```

这段代码为 `add_two_ints` service 创建一个 `client`。`ros::ServiceClient` 对象待会用来调用 service。

```
beginner_tutorials::AddTwoInts srv;
srv.request.a = atoll(argv[1]);
srv.request.b = atoll(argv[2]);
```

这里，我们实例化一个由 ROS 编译系统自动生成的 service 类，并给其 `request` 成员赋值。一个 service 类包含两个成员 `request` 和 `response`。同时也包括两个类定义 `Request` 和 `Response`。

```
if (client.call(srv))
```

这段代码是在调用 service。由于 service 的调用是模态过程（调用的时候占用进程阻止其他代码的执行），一旦调用完成，将返回调用结果。如果 service 调用成功，`call()` 函数将返回 `true`，`srv.response` 里面的值将是合法的值。如果调用失败，`call()` 函数将返回 `false`，`srv.response` 里面的值将是非法的。

## 3. 编译代码

再来编辑一下 `beginner_tutorials` 里面的 `CMakeLists.txt`，文件位于

~/catkin\_ws/src/beginner\_tutorials/CMakeLists.txt, 并将下面的代码添加在文件末尾:

```
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 beginner_tutorials_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 beginner_tutorials_gencpp)
```

```
cd ~/catkin_ws

catkin_make

rospack profile
```

```
[ 96%] Linking CXX executable /home/huanyu/robot_ws/devel/lib/beginner_tutorials/add_two_ints_client
[ 96%] Built target add_two_ints_client
[100%] Linking CXX executable /home/huanyu/robot_ws/devel/lib/beginner_tutorials/add_two_ints_server
[100%] Built target add_two_ints_server
```

#### 4. 测试代码

首先运行 server 节点:

```
huanyu@ubuntu:~$ rosrund beginner_tutorials add_two_ints_server
[ INFO] [1558614440.921226322]: Ready to add two ints.
[ INFO] [1558614453.795479974]: request: x=2, y=5
[ INFO] [1558614453.795549733]: sending back response: [7]
```

然后运行 client 节点:

```
huanyu@ubuntu:~$ rosrund beginner_tutorials add_two_ints_client 2 5
[ INFO] [1558614453.795950474]: Sum: 7
```

#### 5. 总结

本节课程我们学习了如何编译一个 ROS 服务器节点和客户端节点。