学习ROS通信常用API

学习ROS节点的初始化相关API:

1. **ros::init(argc, argv, "node\_name")**: 该函数用于初始化ROS节点，其中 **argc** 和 **argv** 是命令行参数，"node\_name" 是节点名称。
2. **ros::NodeHandle**: 该类提供了一种访问ROS系统的方法。通过创建一个 **NodeHandle** 对象，可以访问ROS的各种功能，例如发布和订阅话题，提供和调用服务等。
3. **ros::Rate**: 该类用于设置发布频率。通过在循环中调用 **ros::Rate::sleep()** 方法，可以将循环速率限制为指定的频率。

学习NodeHandle的基本使用相关API:

1. **NodeHandle::advertise<msg\_type>("topic\_name", queue\_size)**: 该函数用于创建一个发布者对象，发布指定类型的消息到指定名称的话题上。**msg\_type** 是消息类型，**queue\_size** 是队列大小。
2. **NodeHandle::subscribe<msg\_type>("topic\_name", queue\_size, callback\_function)**: 该函数用于创建一个订阅者对象，订阅指定名称的话题，接收指定类型的消息，并在接收到消息时调用指定的回调函数进行处理。**msg\_type** 是消息类型，**queue\_size** 是队列大小，**callback\_function** 是回调函数。
3. **NodeHandle::serviceServer("service\_name", service\_function)**: 该函数用于创建一个服务服务器对象，提供指定名称的服务，并在请求到来时调用指定的服务函数进行处理。**service\_name** 是服务名称，**service\_function** 是服务函数。

学习话题的发布方，订阅方对象相关API:

1. **ros::Publisher**: 该类用于创建一个发布者对象，发布指定类型的消息到指定名称的话题上。可以使用 **Publisher::publish(msg)** 方法向话题发布消息，其中 **msg** 是要发布的消息。
2. **ros::Subscriber**: 该类用于创建一个订阅者对象，订阅指定名称的话题，接收指定类型的消息，并在接收到消息时调用指定的回调函数进行处理。可以使用 **Subscriber::getNumPublishers()** 方法获取当前订阅者接收到消息的发布者数量。

学习服务的服务端，客户端对象相关API:

1. **ros::ServiceServer**: 该类用于创建一个服务服务器对象，提供指定名称的服务，并在请求到来时调用指定的服务函数进行处理。可以使用 **ServiceServer::advertise()** 方法进行广告服务。
2. **ros::ServiceClient**: 该类用于创建一个服务客户端对象，用于向指定名称的服务发送请求，并等待响应。可以使用 **ServiceClient::call()** 方法发送请求，并在响应到来时接收响应。

学习时间相关API:

1. **ros::Time::now()**: 该函数用于获取当前的ROS时间。
2. **ros::Time::sleepUntil(time)**: 该函数用于使当前进程休眠，直到指定时间。

学习日志输出相关API:

1. **ROS\_INFO()**: 该宏用于输出INFO级别的日志信息。
2. **ROS\_WARN()**: 该宏用于输出WARN级别的日志信息。
3. **ROS\_ERROR()**: 该宏用于输出ERROR级别的日志信息。
4. **ROS\_FATAL()**: 该宏用于输出FATAL级别的日志信息。
5. **ROS\_DEBUG()**: 该宏用于输出DEBUG级别的日志信息。

以上宏都可以使用类似于printf()函数的格式化字符串输出信息，例如:

ROS\_INFO("This is an info message with value %d", value);

ROS\_ERROR("An error occurred: %s", error\_msg.c\_str());

需要注意的是，默认情况下，ROS只输出INFO级别及以上的日志信息。如果需要输出更低级别的信息，可以在启动节点时添加命令行参数 **-v** 或 **-vv**，分别表示输出DEBUG级别和VERBOSE级别的信息。此外，还可以使用 **ros::console::set\_logger\_level()** 函数来动态设置输出级别。