ROS

# 安装：

<http://wiki.ros.org/kinetic/Installation/Ubuntu>

# commands

***roswtf*** *# 检测环境变量、安装的文件以及运行的节点*

***rospack list***

***rospack find package-name***

***rosls package-name***

***roscd package-name***

***rosnode list***

***rosnode info node-name***

***rosnode kill node-name***

***rosnode cleanup***

***rostopic list***

***rostopic echo topic-name #*** *rostopic echo /turtle1/cmd\_vel*

***rostopic hz topic-name***

***rostopic bw topic-name***

***rostopic info topic-name***

***rostopic pub –r rate-in-hz topic-name message-type message-content***

***#****rostopic pub –r 1 /turtle1/cmd\_vel geometry\_msgs/Twist ’[0,0,0]’ ’[0,0,1]’*

***rosmsg show message-type-name***

# ROS turtle\_sample

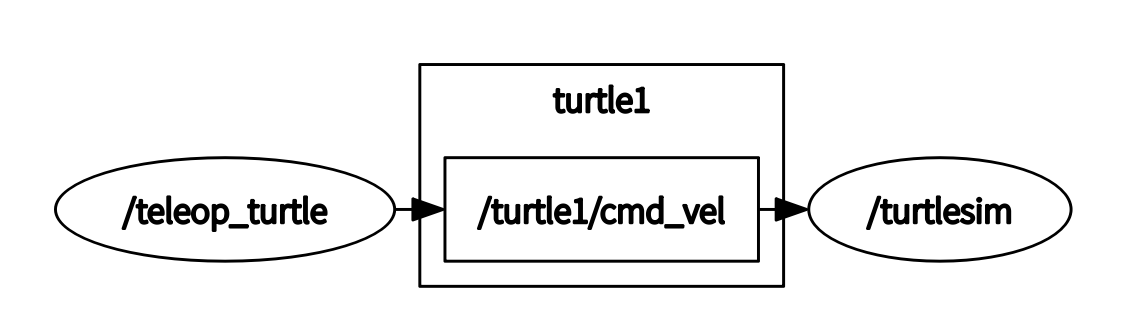
*roscore #启动节点管理器*

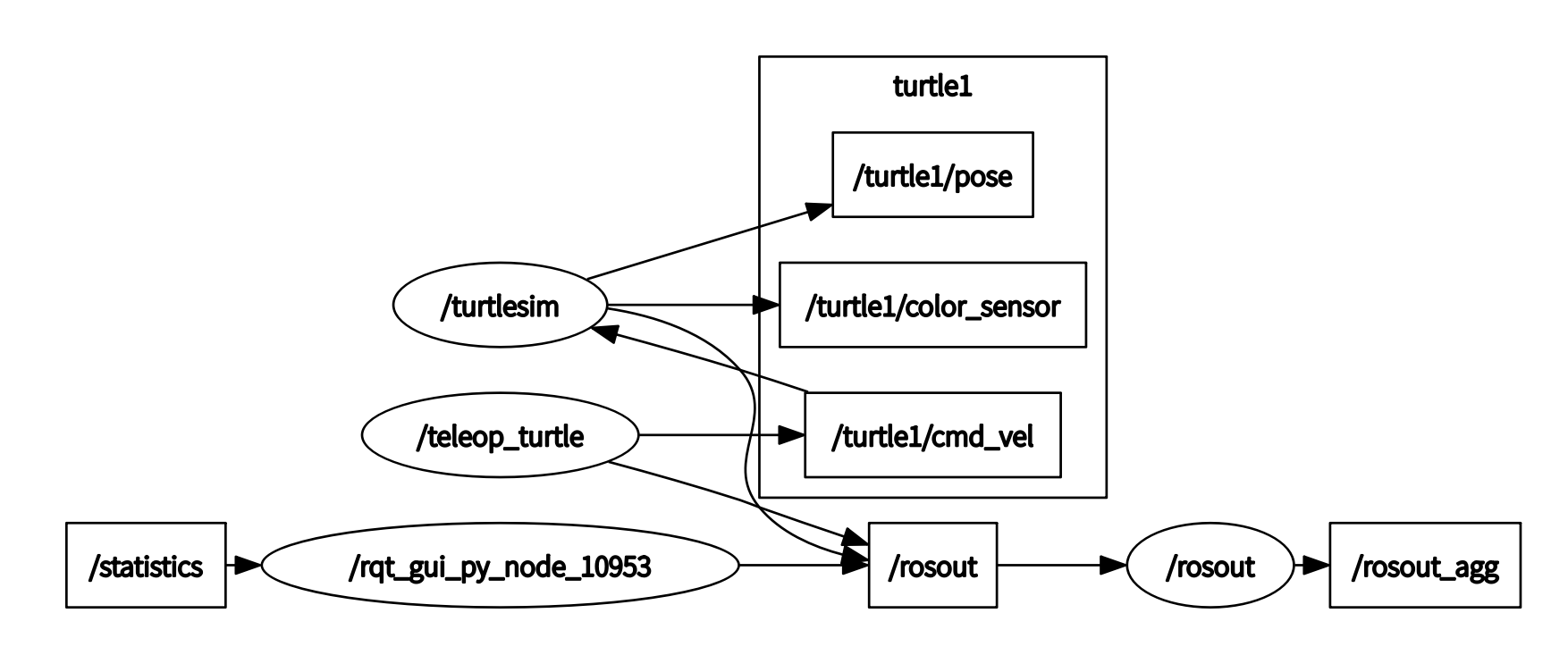
*rosrun turtlesim turtlesim\_node*

*rosrun turtlesim turtle\_teleop\_key*

*#rosrun package-name executable-name*

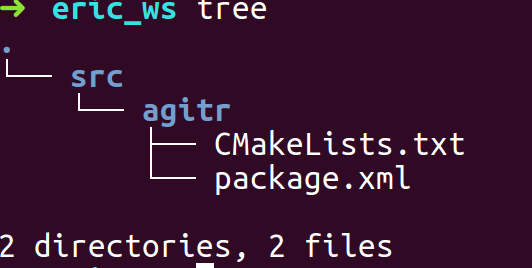
*#rosrun package-name executable-name \_\_name:=node-name*

**

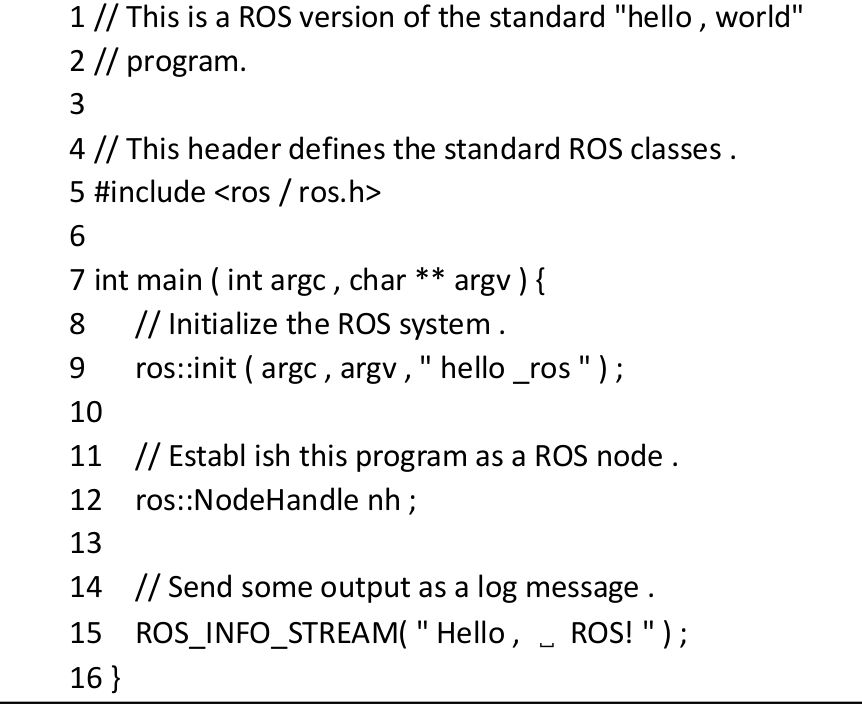
**

# 开发程序

* 创建工作区：mkdir ws\_name、mkdir ws\_name/src
  + 创建功能包：catkin\_create\_pkg package-name #catkin\_create\_pkg agitr

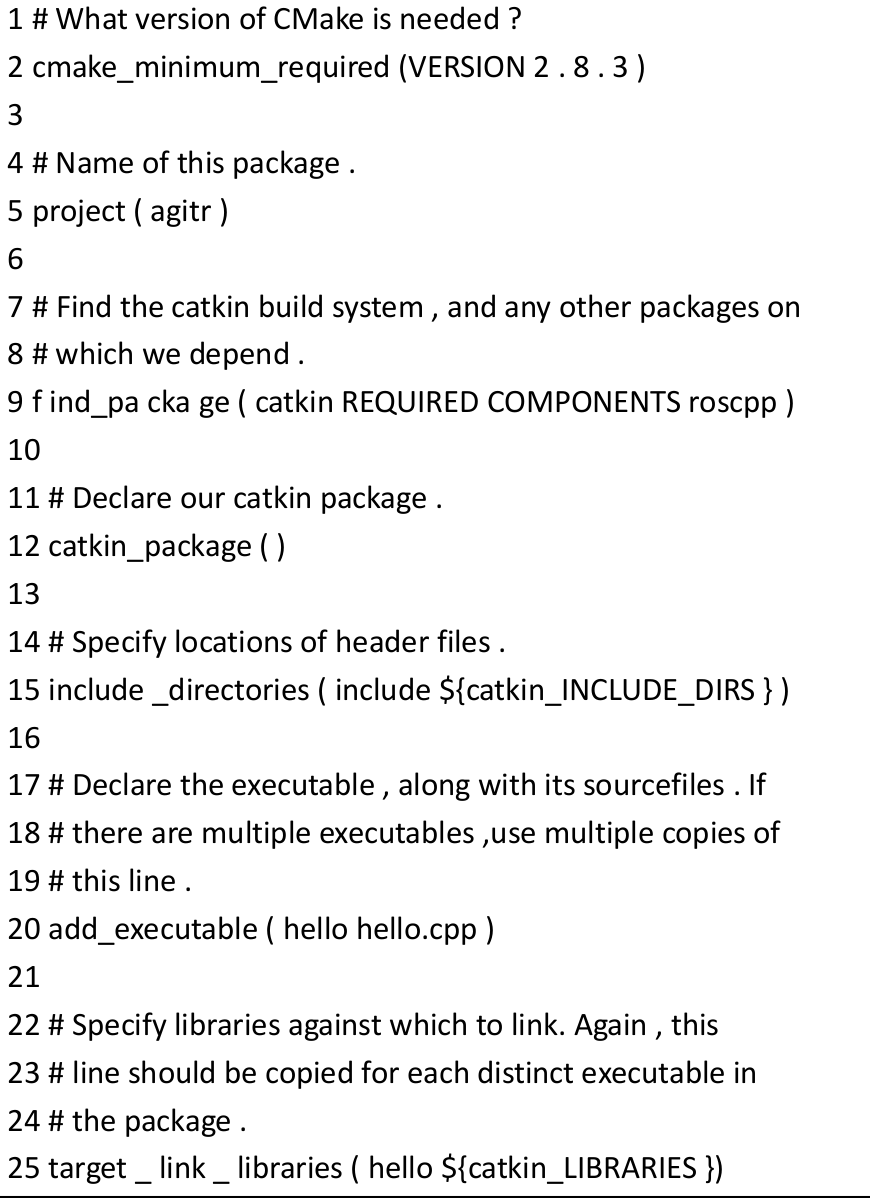


* + 编写代码

**

*创建此对象会将程序注册为ROS节点管理器的节点*

* 编译程序



* 编译工作区

catkin\_make

* Sourcing setup.bash

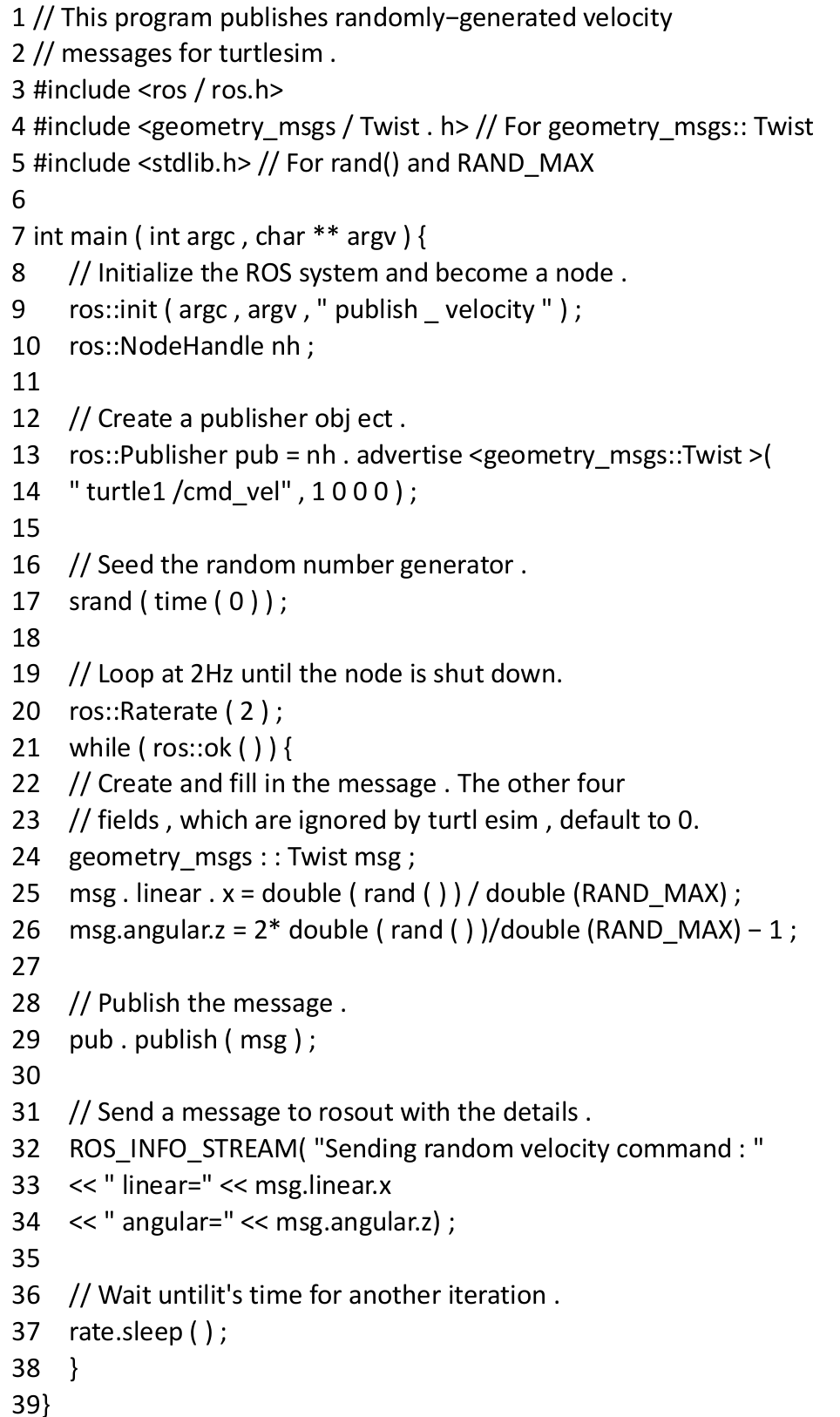
source devel/setup.bash

#*设置了若干环境变量,从而使 ROS 能够找到你创建的功能包和新生成的可执行文件*

* 执行程序

rosrun agitr hello

## 发布消息



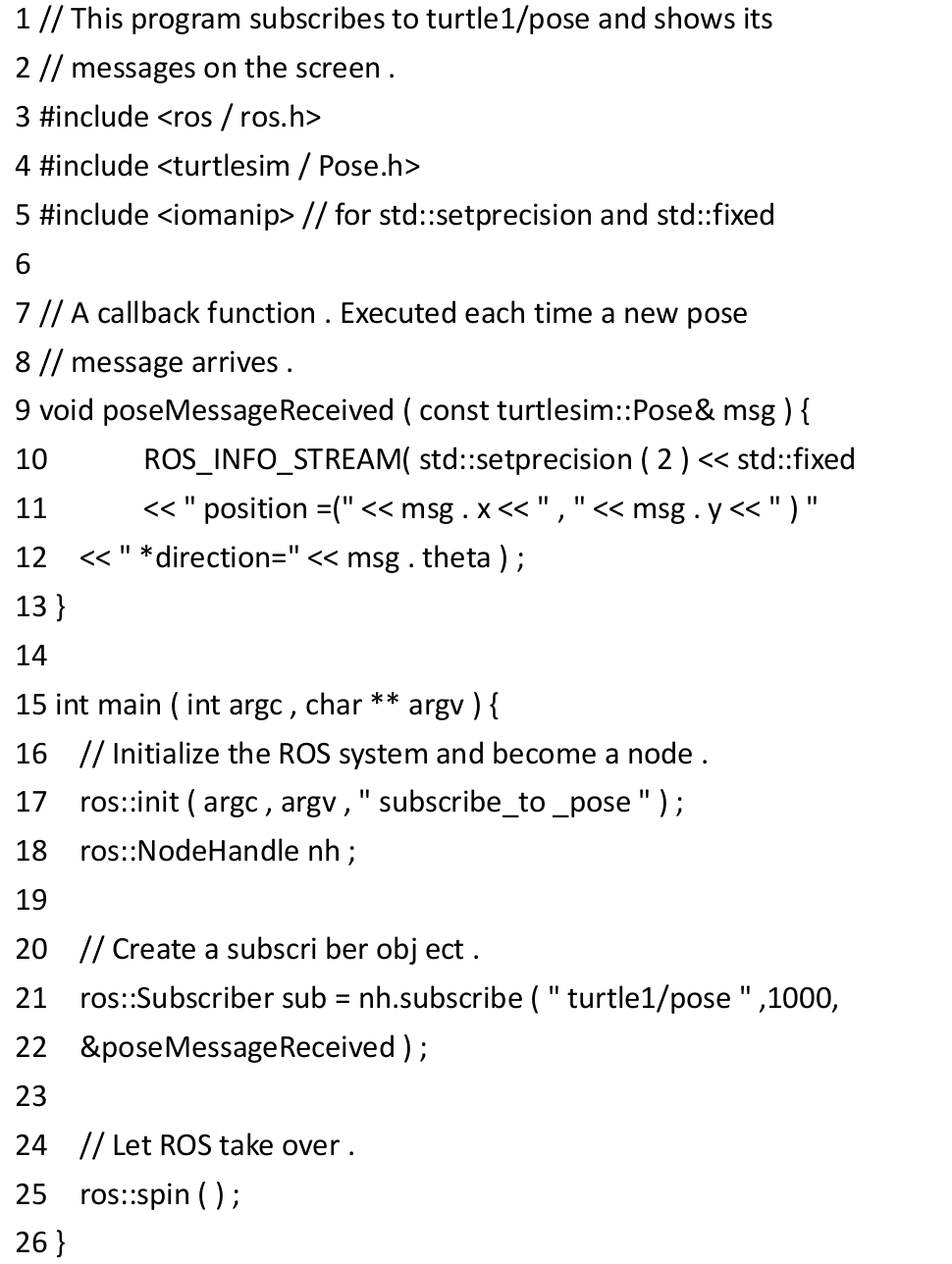
创建发布者：ros::Publisher pub = node\_handle.advertise<message\_type>(

topic\_name, queue\_size);

发布消息

## 订阅消息

* ROS\_\*\*\*\_STREAM #message 可处理 C++中标准输出流(ostream)中的各种表达式如 std::cout
* ROS\_\*\*\* #类似格式输出函数printf
* ROS\_\*\*\*\_STREAM\_ONCE
* ROS\_\*\*\*\_STREAM\_THROTTLE(interval, message) # 频率受控
* \*\*\*：DEBUG/INFO/WARN/ERROR/FATAL
* 输出到控制台



ros::Subscriber sub =node\_handle.subscribe

(topic\_name,queue\_size, pointer\_to\_callback\_function);

程序除了响应回调函数，没有其他重复性工作要做，那么使用 ros::spin(); 否则，合理的选择是写一个循环,做其他需要做的事情,并且周期性地调用 ros::spinOnce()来处理回调

while(ros::ok( ))

{

ros::spinOnce();

}

## 日志

## ros