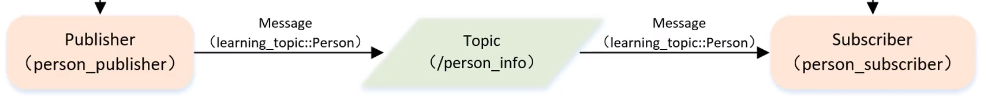
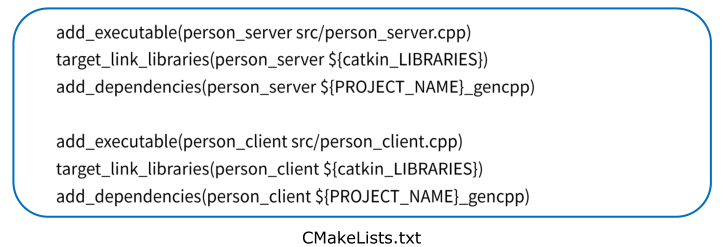
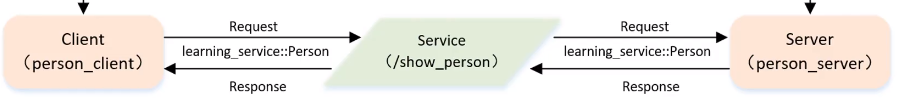
# 【自定义数据】



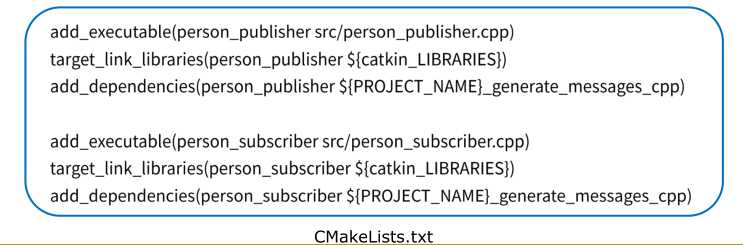


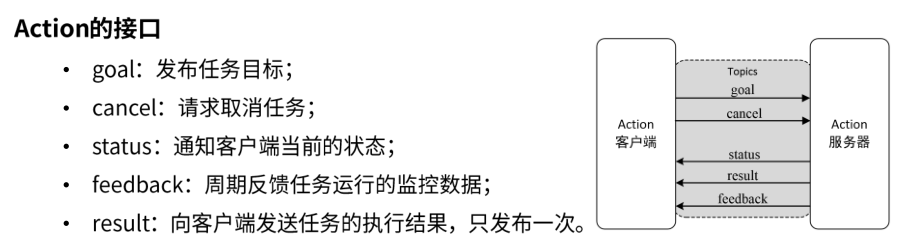


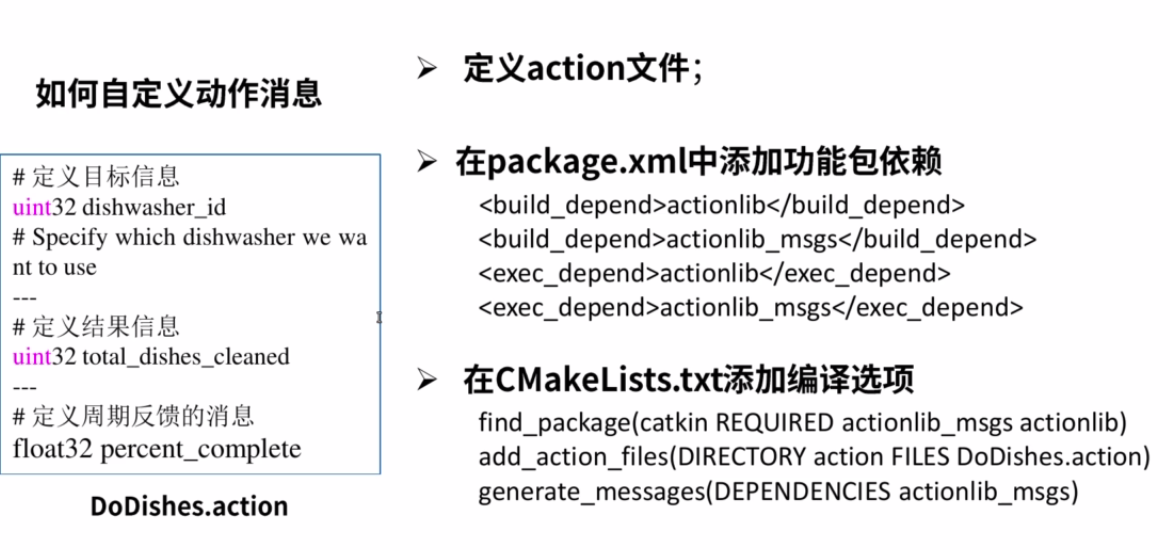
VS



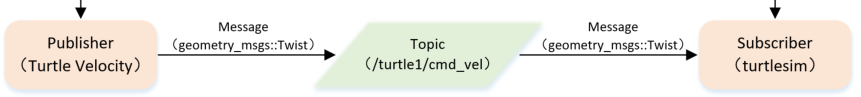








# 【通信类型】

①：发布（使海龟运动）

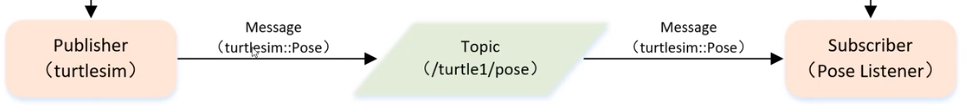
发布者：turtle1/cmd\_vel话题，geometry\_msgs::Twist话题消息类型

ros::Publisher xxx = n.advertise<话题消息类型>(“话题”, 消息队列长度);

xxx.publish(要发布的消息实体);//前面要定义该实体

ROS\_INFO(“…” , …);//printf

②：订阅（海龟位置信息）



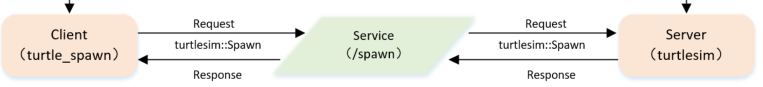
订阅者：turtle1/pose话题，turtlesim::Pose话题数据类型

ros::Subscriber xxx = n.subscribe (“话题”, 消息队列长度, 注册回调函数);

ros::spin();//循环等待，有消息进队列调用回调函数

//-----------------------------------------------------------------------------------------------------------------------------

③：客户端（客户端请求产生新海龟）



客户端：/spawn服务，turtlesim::Spawn服务数据类型

ros::service::waitForService(“服务”);//阻塞等待找到该服务

ros::ServiceClient xxx = n.serviceClient<服务数据类型>(“服务”);

xxx.call(要请求的服务实体); //前面要定义该实体，阻塞，等待反馈

④：服务器（海龟运动/停止）



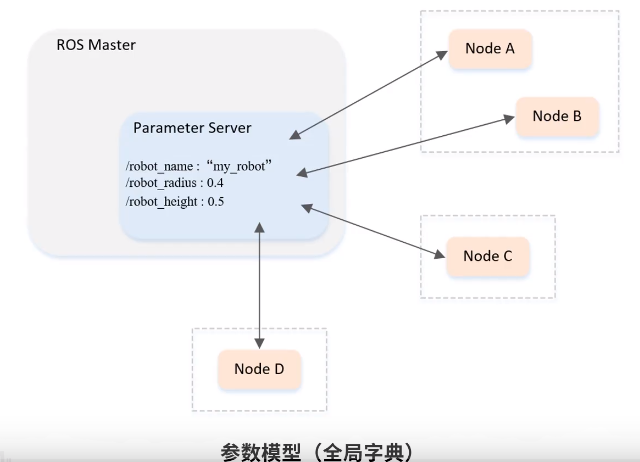
服务器：/turtle\_command服务，std\_srvs::Trigger服务数据类型

ros::ServiceServer xxx = n.advertiseService (“服务”, 注册回调函数);

ros::spinOnce();//查看一次队列，并调用回调

//-----------------------------------------------------------------------------------------------------------------------------

⑤：参数



参数较多时，使用YAML文件集中参数。

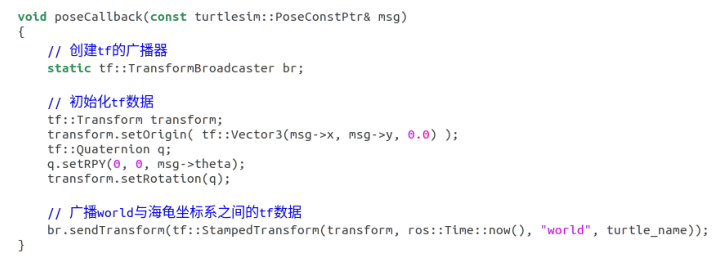
rosparam list

ros::param::get(x1, x2)//获取参数x1存放于x2

ros::param::set(x1, x2)

⑥TF坐标

【TF广播器】：



setOrigin：设置坐标系的平移位置x，y，z ；getOrigin：获取…

setRotation( setRPY(x1, x2, x3 ) )：设置坐标系每个轴旋转角度

StampedTransform(x1, x2, x3, x4)：坐标系关系，时间戳，哪两个坐标系

sendTransform()：发布自身坐标系与world坐标系的tf数据

【TF监听器】：



waitForTransform()：等待是否存t1和t2两个坐标系，ros::Time(0)：实时数据，等待3s无结果，提示错误

transformPoint()：获取某个坐标在另一个坐标系下的坐标轴（用于坐标转换）

lookupTransform()：坐标关系数据保存在transform里

rosrun tf view\_frames //查看tf树坐标系连接关系

# 【lauch文件】（自启动和配置多节点、Master）

<launch>

<node pkg=”…” type=”…” name=”…” args=”…” output=”screen” />

功能：运行节点

pkg: 节点功能包

name：节点名，程序里的init时节点名，若不一样，此名字会覆盖程序里的（rqt\_graph）

type：可执行文件名

output: 输出到屏幕

args: 可执行文件需要输入的参数，可执行文件里可以用main函数的argv[1]来接收

<param name=”…” value=”…” />

功能：加载某个参数（新进参数到master）

name：参数名

value：参数值

<rosparam file=”$(find 功能包名)/config/xx.yaml” command=”load” ns=”params” />

功能：加载参数问价

file：要加载的参数文件

command：指令

ns：参数使用的命名空间

<arg name=”xx” default=”…” />

功能：将lauch文件中的某个参数加载

使用：<param name=”…” value=”$(arg xx)” />

<remap from=”…” to=”…”/>

功能：计算图资源重命名（原来的不会存在了）

<include file=”$(xxx)/…/xx.launch”/>

功能：包含其他launch文件

</launch>

<node /> //在该节点的命名空间下进行

… …

</node>

注释：<!-- …. -->

调用：roslaunch 功能包名 launch文件名（进入catkin工作空间）

# 【ros指令】

**ros安装在Ubuntu其他位置/opt下**

**【第一个小海龟】**

**roscore //运行rosmaster**

**rosrun turtlesim turtlesim\_node**

**rosrun turtlesim turtle\_teleop\_key**

**【常用命令】**

**//只输入以下指令，回车可以看见适配的操作参数！**

**• rostopic**

**• rosservice**

**• rosnode**

**• rosparam**

****

**• rosmsg**

**• rossrv**

**• rosbag**

**//---**

**rqt\_graph //查看计算图，基于qt的可视化工具**

**//---**

**rosnode //查看节点**

**rosnode list //列出系统所有节点**

**info 节点名 //查看节点各项信息**

**//---**

**rostopic list**

**pub 话题名tabtab tab //向某个话题发送指令，修改xyz即可**

**pub -r 数字 话题名tabtab tab //以多少hz速度向某个话题定时发送指令**

**//---**

**rosmsg show 消息数据类型 //显示该消息的结构rosmsg show geometry\_msgs/Twist-**

**//---**

**rosservice list**

**rosservice call 服务名//发布服务rosservice call /spawn tabtab //产生新的海龟**

**//---**

**rossrv show 服务数据类型 //rossrv show std\_srvs/Trigger**

**//---**

**rosbag record -a -O 文件名 //将所有数据保存在指定文件里 .bag文件，**

**rosbag play 文件名 //复现该文件数据，重新开仿真**

**注意：rostopic list话题列表、rostopic info 话题消息类型、rosmsg show 消息数据type**

**rosservice list、rosservice info 服务、rossrv show 服务数据类型type**

**rostopic和rosservice都必须运行功能包才可，rosmsg、rossrv直接可查看某种数据类型**

rosrun tf view\_frames //查看tf树坐标系连接关系

**//---------以上命令都在家目录即可运行！！！！**

**【工作空间】：**

**catkin\_init\_workspace //初始化工作空间，只需手动创建src，进入src进行初始化**

**catkin\_make install //编译后放在install文件里，在主空间进行编译**

**catkin\_create\_pkg 名字 依赖库 //创建功能包！！！ 在src里面创建**

**source devel/setup.bash //设置环境变量**

**echo $ROS\_PACKAGE\_PATH //查看环境变量**

**创建的功能包里重要文件：功能包不允许同名**

**CMakeList.txt ：Cmake的编译规则**

**package.xml：依赖包等信息**

**【话题（Twist/Pose）】功能包里的src存放cpp代码，scripts存放py代码**

**//修改自己的CMakeList.txt ：**

**add\_executable(xx,xxx)将xxx路径文件编译为xx**

**target\_linl\_libraries(xx ${xxx})将编译生成的可执行文件链接到ros库**

**编译生成的运行文件在devel/lib下**

**//每次就不用设置环境变量:**

**在用户主文件夹（家目录），按ctrl+h显示隐藏文件.bashrc，拷贝source /home/gj/devel/setup.bash!!!**

**【自定义话题消息】在功能包里建一个msg文件夹**

**先定义xxx.msg文件，再修改package.xml，再修改CMakeList.txt**

**【tf坐标系】**

**rosrun tf view\_frames //查看tf tree，默认保存在主目录.pdf**

**rosrun tf tf\_echo x1 x2 //获取x1与x2的坐标关系（表示法：四元素、绕xyz弧度、绕xyz角度）**

**解释：实例程序是获取海龟1与world坐标系之间的关系，或者获取海龟2与world坐标系之间的关系**

**rosrun rviz rviz**

**（选择add添加显示类型选项（图像点云等），在确定话题）**

**roslaunch gazebo\_ros tabtab**

**rqt tabtab//查看ros-qt工具**