《ROS机器人开发技术》

实训讲稿

|  |  |  |
| --- | --- | --- |
| 课程名称 | ： | ROS机器人开发技术 |
| 教师姓名 | ： | 刘梦倩 |
| 提交时间 | ： | 2018年7月15日 |

中国大学MOOC制

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 课程  名称 | 《ROS机器人开发技术》 | | 章 | | 第三章 | | | 课程类型 |
| 节 | | 第一节 | | | 授课( )  实训(√ ) |
| 名称 | ROS通讯架构（一） | | | | | | |
| 教师 |  | 课时 | | | | 1课时 | | |
| 参考  资料 | 1. ROS wiki <http://wiki.ros.org/> 2. 《ROS机器人编程实践》 3. 《ROS机器人开发实践》 | | | | | | | |
| 教学  目的  要求 | 掌握：ROS的通信架构之node,master,launch | | | | | | | |
| 教学  重点  难点 |  | | | PPT页面 | | | 时间分配 | |
| 教学重点 | | | | | | | |
| 1. 创建msg文件 | | | 7~13页 | | | 15分钟 | |
| 1. rosmsg | | | 14~19页 | | | 10分钟 | |
| 1. rostopic | | | 20~30页 | | | 20分钟 | |
| 教学难点 | | | | | | | |
| 1. 创建msg文件 | | | 7~13页 | | | 15分钟 | |
| 1. rosmsg | | | 14~19页 | | | 10分钟 | |
|  | 1. rostopic | | | 20~30页 | | | 20分钟 | |

|  |  |  |  |
| --- | --- | --- | --- |
| 教学方法 | | | |
| 本授课以课堂讲授为主，与课堂实时练习方式相结合 | | | |
| 教学内容 | | 操作演示 | 知识点 |
| **PPT第1页:**  下面我们进行第二部分的实训内容。本节课的主要内容是topic和msg的命令的使用。 | |  |  |
| **PPT第2页:**  下面复习一下授课时候我们讲到的相关知识点：  在ROS中有两个节点，一个是发布者（publisher），一个是订阅者（Subscriber），两个节点分别发布、订阅同一个topic，启动顺序没有强制要求，此处假设publisher先启动，可大致分成以下七个步骤来分析建立通信的详细过程。 | |  |  |
| **PPT第3页:**  …… | |  |  |
| **PPT第4页:**  消息（msg）文件就是一个描述ROS中所使用消息类型的简单文本。它们会被用来生成不同语言的源代码。msg文件存放在package的msg目录下，msg文件其实就是每行声明一个数据类型和变量名，可以使用的数据类型如下：  bool、int8、int16、int32、int64(以及uint)、float、float64、string、time、duration、header、可变长数组array[]、固定长度数组array[C]等等。ROS中还有一个特殊的数据类型：Header，含有时间戳和坐标系信息。在msg文件的第一行经常可以看到Header header的声明。 | |  |  |
| **PPT第5页:**  msg也有相对应的ros命令：  rosmsg  这个命令比较简单，只有两个子命令：  ……  我们可以通过rosmsg list命令 查看系统上所有的消息，  然后从当中随便选择一个msg运行rosmsg show msg\_name命令 | |  |  |
| **PPT第6页:**  关于topic的命令：rostopic  …… | |  |  |
| **PPT第7页:**  虽然系统已经给出了许多msg的文件和类型，但是在不同的应用场景之中，我们需要不同类型的msg文件，因此学会自己创建属于自己的msg文件是十分必要的。  任务2就是创建属于自己的一个msg文件。  下面请大家跟着老师的演示一起，在之前创建的package里面定义一个新的消息文件。  断点1：操作演示**示例代码1**  此时可以打开文件夹下面的Num.msg文件，看看里面的内容。 | | 操作演示：示例代码1 1.演示目的：创建msg文件  2. 打开终端。  3.输入：  $ cd ~/catkin\_ws/src/beginner  $ mkdir msg  $ echo “int64 num” > msg/Num.msg  **结论：创建完毕。** | msg |
| **PPT第8页:**  然后为了确保msg文件被转换成C++、Python或其他语言的源代码，查看package.xml，确保它包含以下两条语句：  <build\_depend>message\_generation</build\_depend>  <exec\_depend>message\_runtime</exec\_depend>  没有的话加进去。  断点1：操作演示**示例代码1** | | 操作演示：示例代码1 1.演示目的：配置package.xml  2. 打开package.xml。  3.加入：  <build\_depend>message\_generation</build\_depend>  <exec\_depend>message\_runtime</exec\_depend>  **结论：配置完毕。** | msg |
| **PPT第9页:**  接着进行CMakeLists.txt的配置：  在文件里面找到这几个代码块，改成ppt当中的样子。  断点1：操作演示**示例代码1** | | 操作演示：示例代码1 1.演示目的：CMakeLists.txt的配置  2. 打开CMakeLists.txt。  3.更改：      **结论：配置完毕。** | msg |
| **PPT第10页:**  然后接着配置：  断点1：操作演示**示例代码1**  做完这些，属于自己的消息就创建完毕了，下面通过命令检查ROS是否能够识别创建好的消息。 | | 操作演示：示例代码1 1.演示目的：CMakeLists.txt的配置  2. 打开CMakeLists.txt。  3.更改：      **结论：配置完毕。** | msg |
| **PPT第11页:**  我们可以通过rosmsg list命令 查看系统上所有的消息。  断点1：操作演示**示例代码1**  然后下面会显示出许多的系统当中存在的msg文件，我们可以在其中随便选择一个，使用另一个命令查看他的内容。  例如选择turtle\_actionlib/Velocity这个消息。  同学们可以自己随意挑选一个消息进行下面的操作。  断点1：操作演示**示例代码2** | | 操作演示：示例代码1 1.演示目的：使用rosmsg list命令  2. 打开终端。  3.输入：  $ rosmsg list  **结论：看见list。** 示例代码2 1.演示目的：使用rosmsg show命令  2.输入：  $rosmsg show turtle\_actionlib/Velocity    **结论：看见msg内容。** | rosmsg |
| **PPT第12页:**  我们现在思考一个问题，如果要查看刚刚创建的beginner/Num消息，该如何操作？  大家想一想，是不是还是刚才的操作（是）？  断点1：操作演示**示例代码1**  代码中，beginner：消息所在的package Num：消息名。看看显示的是什么？——是不是我们最开始写到那个文件里面的内容？ | | 示例代码1 1.演示目的：查看刚刚创建的消息  2. 打开终端。  3.输入：  $ rosmsg show beginner/Num #beginner：消息所在的package Num：消息名  显示： int64 num  **结论：看见内容。** | rosmsg |
| **PPT第13页:**  我们再思考一个问题：如果不知道Num消息所在的package，我们怎么找到这个msg？  （过一段时间）  有的同学可能会想出先输入rosmsg list，然后从中找出我们想要的消息。的确可行，但是ros提供给了我们一个更简便的方法，同样是使用rosmsg show命令，直接输入消息名，不加package名，也能找到我们想要的消息。  断点1：操作演示1  看看跟之前那种加上package名的结果有什么不同？  ——显示了package名，相当于一个搜索的功能。 | | 示例代码1 1.演示目的：查看刚刚创建的消息  2. 打开终端。  3.输入：  $ rosmsg show Num  显示：    **结论：看见内容。** | rosmsg |
| **PPT第14页:**  rosmsg的命令很简单，我们就联系到这里为止。下面我们来进入到最难也是最重要的一个部分：rostopic。  我们可以来稍微回忆一下，授课的时候这个命令都有哪些子命令？它们都有着什么功能？  （学生回答一段时间） | |  | rostopic |
| **PPT第15页:**  大家都说的很好，现在给出所有的子命令，大家看看刚刚说的或者是你们自己回忆起来的有什么错误或者是遗漏的地方。  当然，大家记不清也没有关系，毕竟当时只是介绍了一下，只有通过实践才能让大家真正地记住这些命令。  下面我们将逐个演示这些命令的功能。 | |  | rostopic |
| **PPT第16页:**  任务3：使用键盘控制小乌龟运动  我们在前面的章节使用过这个功能：用方向键控制小乌龟行动。当时我们可能不明白命令的意义，今天我们再做一遍，是不是看上去就很明白了？  首先我们先运行小乌龟的程序，不知道大家还记不记得如何操作。请大家自己根据回忆试一下，忘记的也没关系，可以跟着老师一起做。  断点：操作演示1 | | 示例代码1 1.演示目的：运行小乌龟的程序  2. 打开终端。  3.输入：  $ roscore  4.在一个新的终端中运行：  $ rosrun turtlesim turtlesim\_node    **结论：运行小乌龟的程序。** | rostopic |
| **PPT第17页:**    断点：操作演示1  显示的结果如上图，下面就可以通过键盘上的方向键控制turtle运动了。要注意需要选中turtle\_teleop\_key所在的终端窗口以确保按键输入能够被捕获。  下面就要解释通讯到底是怎么做到的：  turtlesim\_node节点和turtle\_teleop\_key节点之间是通过一个topic来互相通信的。turtle\_teleop\_key在一个topic上发布按键输入消息，而turtlesim则订阅该topic以接收该消息。 | | 示例代码1 1.演示目的：用方向键控制小乌龟行动  2. 打开新终端。  3.输入：  $ rosrun turtlesim turtle\_teleop\_key    **结论：用方向键控制小乌龟行动。** | rostopic |
| **PPT第18页:**  为了更直观地看出通讯是怎么做到的，让我们来再次使用rqt\_graph来显示当前运行的节点和话题：  断点：操作演示1  如果将鼠标放在/turtle1/cmd\_vel上方，相应的ROS节点（蓝色和绿色）和话题（红色）就会高亮显示。意味着turtlesim\_node和turtle\_teleop\_key节点正通过一个名为/turtle1/cmd\_vel的话题来互相通信。 | | 示例代码1 1.演示目的：显示当前运行的节点和话题  2. 打开新终端。  3.输入：  $ rosrun rqt\_graph rqt\_graph  或者  $ rqt\_graph    **结论：显示当前运行的节点和话题。** | rostopic |
| PPT第19页: 下面通过操作使大家使用topic命令：  断点：操作演示1  现在再看一下rqt\_graph（可能需要刷新一下，左上角的按键是刷新），显示如下：    正如看到的，rostopic echo（右下角多出来的节点）现在也订阅了/turtle1/cmd\_vel话题。 | | 示例代码1 1.演示目的：rostopic echo  2. 打开新终端。  3.输入：  $ rostopic echo /turtle1/cmd\_vel  这时候可能看不到任何信息，这是因为还没有任何数据发布到该话题上，接下来选中turtle\_teleop\_key节点运行时所在的终端窗口，当按下上方向键的时候应该会看到下面的信息：    **结论：用方向键控制小乌龟行动。** | rostopic |
| PPT第20页: rostopic list能够列出当前所有订阅和发布的话题。  这个命令下有很多的参数，我们可以通过查看帮助来看看到底有哪些参数：  来演示一下：  断点：操作演示1  h：就是查看帮助  b：bagfile：ROS提供的数据记录与回放的功能包：rosbag，可以帮助开发者收集ROS运行时的消息数据，然后再离线状态下回放。（下面的章节中会仔细讲到，这里只是因为提到了，因此作为了解）  v：显示出有关所发布和订阅的话题及其类型的详细信息。 | | 示例代码1 1.演示目的：rostopic list -h  2. 打开新终端。  3.输入：  $ rostopic list -h    **结论：显示了帮助。** | rostopic |
| PPT第21页: 让我们来试一下-v的作用。  断点：操作演示1  可以看到有5个话题正在发布中，被订阅的话题有两个。 | | 示例代码1 1.演示目的：rostopic list -v  2.输入：  $rostopic list -v    **结论：显示详细信息。** | rostopic |
| PPT第22页: 我们再回到15页看这两个参数：  p：只显示发布者  s：只显示订阅者  （大家可以自己试一试这两个命令，试之前先猜猜结果是什么？）  （这部分老师就不演示了） | |  | rostopic |
| PPT第23页: 我们再回到15页看这个参数：  --host：以hostname为标准分组（环境变量）  可能看不太懂这个作用，让我们来实际操作一下看看：  断点：操作演示1  看到只有一个host。  关于rostopic list的命令到这里就结束了，我们下面看下一条命令： | | 示例代码1 1.演示目的：rostopic list --host  2.输入：  $rostopic list --host    **结论：显示分组信息。** | rostopic |
| PPT第24页: rostopic info：显示某个topic的属性信息  例如：我们查看/turtle1/cmd\_vel这个topic的信息：  断点：操作演示1  看到这个topic的type为：geometry\_msgs/Twist  如果只想要查看这个topic的type信息呢？ | | 示例代码1 1.演示目的：rostopic info  2.输入：  $ rostopic info /turtle1/cmd\_vel    **结论：显示详细信息。** | rostopic |
| PPT第25页: 就要用到别的命令了。  ……  $ rostopic type /cmd\_vel  断点：操作演示1  回顾我们刚刚讲过的rosmsg命令，  可以再使用rosmsg show命令查看消息的详细情况：  断点：操作演示2  可以看出分为三个方向的线速度和角速度。 | | 示例代码1 1.演示目的：rostopic type  2.输入：  $ rostopic type /cmd\_vel    **结论：显示type信息。** 示例代码2 1.演示目的：rosmsg show命令查看消息的详细情况  2.输入：  $ rosmsg show geometry\_msgs/Twist    **结论：显示msg信息。** | rostopic |
| PPT第26页: 介绍下一个命令：rostopic pub  可以把数据发布到当前某个正在广播的话题上。  用法：  rostopic pub [topic] [msg\_type] [args]  例如：  $ rostopic pub -1 /turtle1/cmd\_vel geometry\_msgs/Twist -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, 1.8]'  断点：操作演示1  可以看到乌龟运动了一个弧度。下面分析这个命令：  -1：表示只发布一个命令，然后马上退出  /turtle1/cmd\_vel：消息所发布到的话题的名称  --：告诉命令选项解析器，接下来的参数部分都不是命令选项。 | | 示例代码1 1.演示目的：rostopic type  2.输入：  $ rostopic pub -1 /turtle1/cmd\_vel geometry\_msgs/Twist -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, 1.8]'    **结论：乌龟运动了一个弧度。** | rostopic |
| PPT第27页: 改变下刚刚的命令，改成：  $ rostopic pub /turtle1/cmd\_vel geometry\_msgs/Twist -r 1 -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, 1.8]'  看一下效果  断点：操作演示1  -r：发布一个稳定的命令流，使乌龟不停运动。  1：代表1hz。 | | 示例代码1 1.演示目的：乌龟持续运动  2.输入：  $ rostopic pub /turtle1/cmd\_vel geometry\_msgs/Twist -r 1 -- '[2.0, 0.0, 0.0]' '[0.0, 0.0, 1.8]'    **结论：乌龟持续运动。** | rostopic |
| PPT第28页: 刚刚命令的效果是这样的： | |  | rostopic |
| PPT第29页: 刚刚命令当中，用到了hz，发布命令的频率是1hz，那么怎么查看某个命令的频率呢：  rostopic hz  用来查看数据发布的频率。  用法：  rostopic hz [topic]  比如我们想要查看turtlesim\_node发布/turtle1/cmd\_vel的频率  断点：操作演示1 | | 示例代码1 1.演示目的：显示频率  2.输入：  $ rostopic hz /turtle1/cmd\_vel    **结论：显示频率。** | rostopic |
| PPT第30页： 结果如图。 | |  | rostopic |
| PPT第31页： 到此为止，所有的命令都已经演示过了。接下来的一节课会进行一些综合训练。谢谢同学们。 | |  |  |
| 知识点框图 | | | |
| 结构图 |  | | |