《ROS机器人开发技术》

课程讲稿

|  |  |  |
| --- | --- | --- |
| 课程名称 | ： | ROS机器人开发技术 |
| 教师姓名 | ： |  |
| 提交时间 | ： | 2018年9月9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 | 通信架构（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下运行输出Hello World！ | | | | | | | |
| 教学  重点  难点 |  | | | PPT页面 | | | 时间分配 | |
| 教学重点 | | | | | | | |
| 1. 创建Catkin工作空间 | | | 4~11页 | | | 10分钟 | |
| 1. 创建一个包并创建hello.cpp文件 | | | 13~16页 | | | 10分钟 | |
| 1. 修改配置文件CMkeLists.txt | | | 18~22页 | | | 10分钟 | |
| 教学难点 | | | | | | | |
| 1. 创建Catkin工作空间 | | | 4~11页 | | | 10分钟 | |
| 1. 创建一个包并创建hello.cpp文件 | | | 13~16页 | | | 10分钟 | |
|  | 1. 修改配置文件CMkeLists.txt | | | 18~22页 | | | 10分钟 | |

|  |  |  |
| --- | --- | --- |
| 教学方法 | | |
| 本授课以课堂讲授为主，与课堂演示方式相结合 | | |
| 教学内容 | 操作演示 | 知识点 |
| **PPT第1页：**  本章我们讲ROS的一个实训内容：ros下运行输出Hello world。  这是我们手动编写的第一个ROS程序，所以大家一定要认真听讲，认真对待这部分内容。 |  |  |
| **PPT第2页:**  这是一个提纲。分别总结了本章将要介绍到的一些概念以及要讲解的一些命令。  主要分为四个步骤。  现在先给大家看一下，让大家有一个大概的印象，后面再一一讲解。 |  |  |
| **PPT第3页:**  第一部分，我们先介绍如何创建catkin工作空间，这是我们ROS的首要工作，后续工作都会在catkin工作空间中实现。 |  |  |
| **PPT第4页:**  创建catkin工作空间任务主要分为三个步骤：  **创建一个名为catkin\_ws的工作空间；**  **在工作空间文件夹下创建src文件夹；**  **Catkin编译** |  |  |
| **PPT第5页:**  现在，我们开始创建catkin工作空间，首先我们要创建一个叫catkin\_ws的文件夹，里面要有src目录；打开终端，在终端中输入：  ~$ mkdir -p ~/catkin\_ws/src  这样我们就创建了catkin\_ws的文件夹，并且里面包含了src目录  现在我们就要编译这个catkin工作空间  然后进入这个文件夹：  ~$ cd ~/catkin\_ws  在这个文件夹下运行一个命令：  ~/catkin\_ws$ catkin\_make  catkin\_make 的作用是帮我们初始化工作空间，这样我们的工作空间就建好了  下面我们来介绍几个知识点 |  |  |
| **PPT第6页：**  知识点1：  catkin\_make：是一个非常方便的Catkin工作空间工具。  在工作空间中第一次运行这个命令时，系统会在src目录下创建一个CMakeLists.txt文件。  同时，在catkin\_ws目录下建立build文件夹和devel文件夹以及其中文件  在devel目录里还可以看到系统自动创建了setup.\*sh形式的环境变量设置脚本。  其实catkin\_make不只是初始化工作空间，他最重要的功能是编译, 一般我们写完了代码，然后catkin\_make一下，这样系统就会进行自动构建，注意啊，运行catkin\_make之前一定要回到Workspace目录，否则编译会失败。一定要cd到catkin\_ws那创建完工作空间，catkin\_make之后，这个workspace下就会起变化。  **PPT第7页：**  知识点2：  mkdir命令用来创建目录;  如果在目录名的前面没有加任何路径名，则在当前目录下创建由dirname指定的目录；如果给出了一个已经存在的路径，将会在该目录下创建一个指定的目录。在创建目录时，应保证新建的目录与它所在目录下的文件没有重名。  语法：mkdir（选项）（参数）。  选项：-m 建立目录的同时设置目录的权限;  -p 递归创建多个子目录;  -v 创建新目录都显示信息  参数：  目录：指定要创建的目录列表，多个目录之间用空格隔开。 |  | **node**  **Master** |
| **PPT第8页：**  知识点3：  cd：用来切换工作目录至dirname, 其中dirname表示法可为绝对路径或相对路径。  另外，~表示为home directory的意思，.则表示目前所在的目录，..则表示目前目录位置的上一层目录  cd 表示进入用户主目录  cd .. 表示返回上级目录；  cd ../.. 表示返回上两级目录。 |  | **roscore** |
| **PPT第9页:**  知识点4：  使用source命令，运行这些脚本文件之一，就可以使工作空间中的环境变量生效。  使当前shell读入路径为filepath的shell文件并依次执行文件中的所有语句，通常用于重新执行刚修改的初始化文件，使之立即生效，而不必注销并重新登录。  在终端中输入：  $ source devel/setup.bash |  |  |
| **PPT第10页:**  具体操作是：  在终端中继续输入：$ echo $ROS\_PACKAGE\_PATH  为确认ROS\_PACKAGE\_PATH环境变量包含了我们建立的工作空间目录，可使用上述指令；  运行结果如下图所示 |  |  |
| **PPT第11页:**  现在我们来总结一下如何创建catkin工作空间，首先创建文件夹：  命令是： $ mkdir ~/catkin\_ws/src  然后进入catkin\_ws: $ cd ~/catkin\_ws/  编译工作空间： $ catkin\_make  这样我们可以得到左边的运行结果，这样我们就成功的创建了catkin工作空间，后面的工作我们就可以在这个创建好的空间中实现了。 |  | **rosrun** |
| **PPT第12页:**  现在我们进入本章实训的第二个步骤：创建一个包并创建hello.cpp文件。尽管这部分内容不是很多，但是这一步是我们这次实训的核心内容。下面我们就来学习怎么创建一个包并创建hello.cpp文件 |  | **rosnode** |
| PPT第13页: 这一节的任务分为两个步骤：1.创建一个包First\_pkg；2.创建hello.cpp文件  在之前的课程中我们已经学习了catkin工作空间的一些基础知识，我们知道要我们的每个包有一个具体实现的功能，多个包可以组成了一个模块。这里我们就不具体介绍了，下面我们就来学习如何创建一个包。 |  | rosnode |
| PPT第14页: 本节将演示如何使用[catkin\_create\_pkg](http://wiki.ros.org/catkin/commands/catkin_create_pkg)命令来创建一个新的catkin程序包  首先，打开终端，进入catkin\_ws工作空间，命令如下：  $ cd ~/catkin\_ws/src  然后使用catkin\_create\_pkg命令来创建一个名为' **First\_pkg** '的新程序包，这个程序包依赖于roscpp和rospy**：**  **~/catkin\_ws/src$ catkin\_create\_pkg First\_pkg roscpp rospy**  这将会创建一个名为 **First\_pkg** 的文件夹，这个文件夹里面包含一个package.xml文件和一个CMakeLists.txt文件，这两个文件都已经自动包含了部分你在执行catkin\_create\_pkg命令时提供的信息**。** |  | launch文件 |
| PPT第15-16页: 现在我们来创建hello.cpp文件：  首先，进入First\_pkg /src目录：  $ cd First\_pkg/src  然后，创建C++程序文件hello.cpp：  ~/catkin\_ws/src/First\_pkg/src$ vim hello.cpp  创建完成后在终端下编写hello world程序：  #include <iostream>  int main()  {  std::cout<<"Hello World!"<<std::endl;  return 0;  }  写完后（：wq）保存。 |  | launch文件 |
| PPT第17页: 现在我们进入本章实训的第三个步骤：修改 CMkeLists.txt |  | topic |
| **PPT第18页:**  首先进入First\_pkg 包，打开 CMkeLists.txt：  ~/catkin\_ws/src/First\_pkg/src$ cd ..  ~/catkin\_ws/src/First\_pkg$ vim CMakeLists.txt  这样我们就打开了 CMakeLists.txt，下面我们来修改 CMakeLists.txt的一些内容 |  |  |
| PPT第19页: 在打开的 CMakeLists.txt添加下面三个函数： add\_executable(test1\_node src/hello.cpp)  add\_dependencies(test1\_node ${hello.cpp\_EXPORTED\_TARGETS} ${catkin\_EXPORTED\_TARGETS})  target\_link\_libraries(test1\_node  ${catkin\_LIBRARIES} )  编写完成后保存退出，命令为（：wq） |  | topic |
| PPT第20页: 具体操作如图所示： |  |  |
| **PPT第21页：**  修改CMakeLists.txt完成后，我们要进入catkin\_ws目录输入catkin\_make命令进行编译：  $ cd ~/catkin\_ws  ~/catkin\_ws$ catkin\_make |  |  |
| **PPT第22页：**  编译后结果如下： |  |  |
| **PPT第23页：**  现在我们进入本节实训的最后一个步骤：编译运行输出hello world  新建终端，进入catkin\_ws目录，输入以下命令：  ~/catkin\_ws$ source ./devel/setup.bash  ~/catkin\_ws$ roscore //运行主节点 |  | topic |
| PPT第24页: 新建终端，进入catkin\_ws目录，输入以下命令：  ~/catkin\_ws$ source ./devel/setup.bash  ~/catkin\_ws$ roscore //运行主节点 |  | msg |
| PPT第25页: 新建终端，执行以下命令：  $source catkin\_ws/devel/setup.bash  $ rosrun First\_pkg test1\_node |  | rosmsg |
| PPT第26页: 这样我们就可以得到输出结果：  **PPT第27页:**  实训的部分就到此结束了，下面我们介绍一下source：  编译完成后必须使用source命令刷新一下工作空间的环境。**初学时可能以为，在工作空间中进行一次source之后就一劳永逸以后都不用source了，这种看法是错的。事实是，每次进行catkin\_make之后，都要进行source。**  进行package创建时，当加入了新的package编译完成后，也要进行source刷新环境，否则会出现找不到“package XXX not found” 的问题  **PPT第28页:**  我们进行实验验证：f*irst\_pkg* 为我们自己创建的package包名, 如果新建终端再进行rosrun就提示package test找不到，因为此时重新编译了但未刷新环境。source之后，就可以运行了。  其结果如图所示：  **PPT第29页:**  这样每次都要source都很麻烦，解决办法是，使用如下命令  echo “source ~/catkin\_ws/devel/setup.bash” >> ~/.bashrc  这样在每次打开终端时，让系统自动刷新工作空间环境。在这个工作空间下的所有package都可以编译后就可以直接运行了，不用再source  注： echo $ROS\_PACKAGE\_PATH可以查看系统包含的package路径  **PPT第30页:**  本次课程就结束了，谢谢各位同学的收听，希望大家在后面多回顾这部分的内容，起到记忆的效果。 |  | rostopic |
| 知识点框图 | | |
| 通讯架构之话题  概念  命令  rostopic  roslaunch  roscore  rosnode  rosrun  rosmsg  launch文件  msg  topic  Master  Node | | |