Understanding ROS

The ROS packages are the basic units of the ROS system. The distribution of ROS which we are using is **Indigo**. We will use the ***catkin build system*** to build ROS packages. A build system is responsible for generating 'targets'(executables/libraries) from a raw source code that can be used by an end user.

The first requirement in creating ROS packages is to create a ROS catkin workspace.

How to create ROS catkin workspace (catkin_ws):

Here is the procedure to build a catkin workspace:

1. Build a workspace folder in the home directory and create an **"src"** folder inside the workspace folder:
   mkdir -p ~/catkin_ws/src
2. Switch to the **"src"** folder. The packages are created inside this package:
   cd ~/catkin_ws/src
3. Initialize a new catkin workspace:
   catkin_init_workspace
4. You can **build the workspace** even if there are no packages. You can use the following command to switch to the workspace folder:
   cd ~/catkin_ws
5. The **catkin_make** command will build the catkin workspace:
   catkin_make
6. After building the empty workspace, you should set the environment of the current workspace to be visible by the ROS system. This process is called overlaying a workspace. You should add the package environment using the following command:
   echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
   source ~/.bashrc

This command will source a bash script called **setup.bash** inside the **"devel"** workspace folder. To set the environment in all bash sessions, you need to add a source command in the **.bashrc** file, which will **source this script** whenever a bash session starts.

After setting the catkin workspace, you can **create your own package** that has sample nodes to demonstrate the working of ROS topics, messages, services, and actionlib.

Creating First Package:

The **catkin_create_pkg** command is used to create a ROS package. This command is used to create your package in which you are going to create tasks of various ROS concepts.

Switch to the catkin workspace **"src"** folder and create the package using the following command:

Syntax of catkin_create_pkg:

catkin_create_pkg [package_name] [dependency1] [dependency2]

Here is the example command to create the sample ROS package:

catkin_create_pkg sample_package roscpp rospy std_msgs

The dependencies in the sample packages are as follows:

- **roscpp**: This is the C++ implementation of ROS. It is a ROS client library which provides APIs to C++ developers to make ROS nodes with ROS topics, services, parameters, and so on. You can include this dependency if you are going to write a ROS C++ node. Any ROS package which uses the C++ node must add this dependency.
- **rospy**: This is the **Python** implementation of ROS. It is a ROS client library which provides APIs to Python developers to make ROS nodes with ROS topics, services, parameters, and so on. You can include this dependency if you are going to write a ROS Python node. In this theme, you have to write the nodes in Python so you must include this dependency in your package.
- **std_msgs**: This package contains basic ROS primitive data types such as integer, float, string, array, and so on. You can directly use these data types in your nodes without defining a new ROS message.

After creating this package, build the package without adding any nodes using the **catkin_make** command. This command must be executed from the catkin workspace path. The following command shows you how to build an empty ROS package:

cd ~/catkin_ws/src/sample_package
mkdir scripts
mkdir launch
cd ~/catkin_ws
catkin_make

After a successful build, you can start **adding nodes** to the **"scripts"** folder inside this sample package. You can also add your **launch files** in **"launch"** folder.

Important: To better understand the commands to run your packages, programs and launch your files, we suggest you go through the chapters from 2 to 6 in "Programming Robots with ROS: A Practical Introduction to the Robot Operating - Brian Gerkey, Morgan L. Quigley, and William D. Smart**".**

Additional resources which may help you:

1. Mastering ROS for Robotics Programming - Lentin Joseph
2. Programming Robots with ROS: A Practical Introduction to the Robot Operating - Brian Gerkey, Morgan L. Quigley, and William D. Smart
3. http://wiki.ros.org/ROS/Tutorials