Please ask about problems and questions regarding this tutorial on answers.ros.org (http://answers.ros.org). Don't forget to include in your question the link to this page, the versions of your OS & ROS, and also add appropriate tags.

Installing and Configuring Your ROS Environment

Description: This tutorial walks you through installing ROS and setting up the ROS environment on your computer.

Tutorial Level: BEGINNER

Next Tutorial: Navigating the ROS Filesystem (/ROS/Tutorials/NavigatingTheFilesystem)

Contents

1. Install ROS

Before starting these tutorials please complete installation as described in the ROS installation instructions (/ROS/Installation).

Note: If you installed ROS from a package manager like `apt`, then those packages will not be write accessible and should not be edited by you the user. When working with ROS packages from source or when creating a new ROS package, you should always work in a directory that you have access to, like your home folder.

2. Managing Your Environment

During the installation of ROS, you will see that you are prompted to `source` one of several setup.*sh files, or even add this 'sourcing' to your shell startup script. This is required because ROS relies on the notion of combining spaces using the shell environment. This makes developing against different versions of ROS or against different sets of packages easier.

If you are ever having problems finding or using your ROS packages make sure that you have your environment properly setup. A good way to check is to ensure that environment variables (/ROS/EnvironmentVariables) like ROS_ROOT (/ROS/EnvironmentVariables#ROS_ROOT) and ROS_PACKAGE_PATH (/ROS/EnvironmentVariables#ROS_PACKAGE_PATH) are set:

```
$ printenv | grep ROS
```

If they are not then you might need to 'source' some setup.*sh files.

Environment setup files are generated for you, but can come from different places:

- ROS packages installed with package managers provide setup.*sh files
- rosbuild workspaces (http://www.ros.org/wiki/fuerte/Installation/Overlays) provide setup.*sh files using tools like rosws (/rosws)
- Setup.*sh files are created as a by-product of building (/catkin/workspaces#Building_Packages_with_catkin) or installing (/catkin/workspaces#Installing_Packages_with_Catkin) catkin packages

Note: Throughout the tutorials you will see references to rosbuild (/rosbuild) and catkin (/catkin). These are the two available methods for organizing and building your ROS code. rosbuild is not recommended or maintained anymore but kept for legacy. catkin is the recommended way to organise your code, it uses more standard CMake conventions and provides more flexibility especially for people wanting to integrate external code bases or who want to release their software. For a full break down visit catkin or rosbuild (/catkin_or_rosbuild).

If you just installed ROS from `apt` on Ubuntu then you will have setup.*sh files in '`/opt/ros/<distro>/`', and you could source them like so:

```
$ source /opt/ros/<distro>/setup.bash
```

Using the short name of your ROS distribution instead of `<distro>`

If you installed ROS Kinetic, that would be:

```
$ source /opt/ros/kinetic/setup.bash
```

You will need to run this command on every new shell you open to have access to the ROS commands, unless you add this line to your .bashrc. This process allows you to install several ROS distributions (e.g. indigo and kinetic) on the same computer and switch between them.

On other platforms you will find these setup.*sh files wherever you installed ROS.

3. Create a ROS Workspace

catkin | rosbuild

These instructions are for ROS Groovy and later. For ROS Fuerte and earlier, select rosbuild.

Let's create and build a catkin workspace (/catkin/workspaces):

```
$ mkdir -p ~/catkin_ws/src
$ cd ~/catkin_ws/
$ catkin_make
```

The catkin_make (/catkin/commands/catkin_make) command is a convenience tool for working with catkin workspaces (/catkin/workspaces). Running it the first time in your workspace, it will create a `CMakeLists.txt` link in your 'src' folder. Note, for Arch Linux users, the first catkin_make

(/catkin/commands/catkin_make) command in a clean catkin workspace (/catkin/workspaces) must be:

```
$ catkin_make -DPYTHON_EXECUTABLE=/usr/bin/python3 -DPYTHON_INCLUDE_DIR=/usr/include/python3.7m -DPYTHON_LIBRARY=/usr/lib/libpython3.7m.so
```

This will configure catkin_make (/catkin/commands/catkin_make) with Python 3. You may then proceed to use just `catkin_make` for subsequent builds.

Additionally, if you look in your current directory you should now have a 'build' and 'devel' folder. Inside the 'devel' folder you can see that there are now several setup.*sh files. Sourcing any of these files will overlay this workspace on top of your environment. To understand more about this see the general catkin documentation: catkin (/catkin). Before continuing source your new setup.*sh file:

```
$ source devel/setup.bash
```

To make sure your workspace is properly overlayed by the setup script, make sure `ROS_PACKAGE_PATH` environment variable includes the directory you're in.

```
$ echo $ROS_PACKAGE_PATH
/home/youruser/catkin_ws/src:/opt/ros/kinetic/share
```

Now that your environment is setup, continue with the ROS file system tutorial (/ROS/Tutorials/NavigatingTheFilesystem).

Wiki: ROS/Tutorials/InstallingandConfiguringROSEnvironment (last edited 2017-05-22 23:24:18 by Marguedas (/Marguedas))