Note: This tutorial assumes that you have completed the previous tutorials: ROS Tutorials (/ROS/Tutorials).

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.

Arduino IDE Setup

Description: This tutorial shows step-by-step how to setup up the Arduino IDE to use rosserial.

Tutorial Level: BEGINNER

Next Tutorial: Hello World (example publisher) (/rosserial_arduino/Tutorials/Hello%20World)

Contents

1. Introduction

The Arduino and Arduino IDE are great tools for quickly and easily programming hardware. Using the rosserial_arduino (/rosserial_arduino) package, you can use ROS directly with the Arduino IDE. rosserial (/rosserial) provides a ROS communication protocol that works over your Arduino's UART. It allows your Arduino to be a full fledged ROS node which can directly publish and subscribe to ROS messages, publish TF transforms, and get the ROS system time.

NOTE: If you do not already have an Arduino IDE installed, download it from the Arduino website (http://arduino.cc/en/Main/Software). It is best to install the application into a folder on the application PATH, the desktop, or home folder. Once installed, launch the application to select your sketchbook location. (See arduino official website (http://www.arduino.cc/en/Guide/Environment), sketchbook is a standard place to store your programs, or sketches). Close the IDE when done.

Our ROS bindings are implemented as an Arduino library. Like all Arduino libraries, `ros_lib` works by putting its library implementation into the *libraries* folder of your sketchbook. If there is not already a *libraries* folder in your sketchbook, make one. You can then install the library using the instructions below.

In order to use the rosserial libraries in your own code, you must first put

```
#include <ros.h>
```

prior to including any other header files, e.g.

```
#include <std_msgs/String.h>
```

otherwise the Arduino IDE will not be able to locate them.

2. Installing the Software

2.1 Installing on the ROS workstation

You have 2 options of how to install related libraries.

2.1.1 (RECOMMENDED) Installing Binaries on the ROS workstation

You can install rosserial for Arduino by running:

```
sudo apt-get install ros-indigo-rosserial-arduino
sudo apt-get install ros-indigo-rosserial
```

Replace indigo with the name of the release you're installing from: e.g. hydro.

2.1.2 Installing from Source onto the ROS workstation

Source build instructions are different for groovy+ (catkin) than for earlier (rosbuild) releases. Select the build system based on your release to see appropriate instructions.

catkin	rosbuild

Rosserial has been catkin-ized since the groovy release, and the workflow is a bit different from fuerte and earlier releases. Rather than running the library generator over each package you want to use, you run it once and generate libraries for all installed messages. In the instructions below, <ws> represents your catkin workspace.

```
cd <ws>/src
git clone https://github.com/ros-drivers/rosserial.git
cd <ws>
catkin_make
```

These commands clone rosserial from the github repository, generate the rosserial_msgs needed for communication, and make the `ros_lib` library in the <ws>/install directory.

Note: currently you HAVE to run catkin_make install, otherwise portions of the ros_lib directory will be missing. This will hopefully be fixed soon.

2.2 Install ros_lib into the Arduino Environment

The preceding installation steps created ros_lib, which must be copied into the Arduino build environment to enable Arduino programs to interact with ROS.

In the steps below, <sketchbook> is the directory where the Linux Arduino environment saves your sketches. Typically this is a directory called *sketchbook* in your home directory. Alternately, you can install into a Windows Arduino environment.

Ros_lib installation instructions are different for groovy source (catkin) than for earlier (rosbuild) or binary releases. Be sure you've selected the correct build system above to see appropriate instructions - catkin for a groovy source build, rosbuild otherwise.

Note: you have to delete libraries/ros_lib in order to regenerate as its existence causes an error.

```
cd <sketchbook>/libraries
rm -rf ros_lib
rosrun rosserial_arduino make_libraries.py .
```

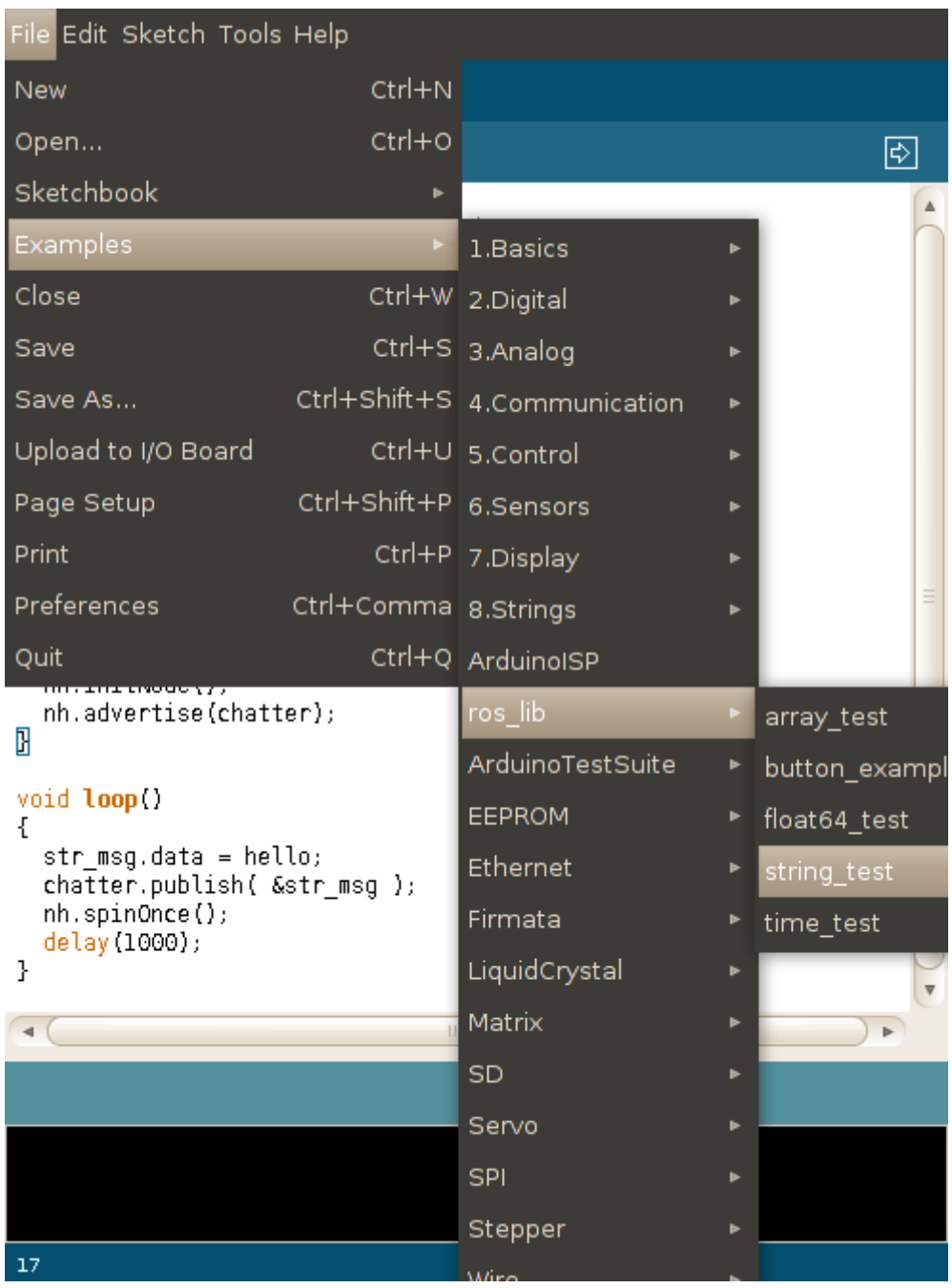
If you are building the Arduino on Windows, you need to create the ros_lib folder in some convenient directory.

```
cd <some_empty_directory>
rosrun rosserial_arduino make_libraries.py .
```

If you are building Arduino on Windows, copy the ros_lib directory from Linux to the Windows sytem's sketchbook/libraries folder (typically found in My Documents).

3. Finishing Up

After restarting your IDE, you should see ros_lib listed under examples:



Wiki: rosserial_arduino/Tutorials/Arduino IDE Setup (last edited 2016-06-16 19:25:52 by Shibhansh Dohare (/Shibhansh%20Dohare))

Brought to you by:  Open Source Robotics Foundation

(http://www.osrfoundation.org)