Note: This tutorial assumes that you have completed the previous tutorials: Arduino IDE Setup (/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup).

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.

Hello World (example publisher)

Description: This tutorial shows step by step how to create a publisher using rosserial.

Tutorial Level: BEGINNER

Next Tutorial: Blink (example subscriber) (/rosserial_arduino/Tutorials/Blink)

Contents

1. Hello World: Creating a Publisher

1.1 The Code

We'll start our exploration into rosserial by creating a "hello world" program for our Arduino. (Note: the Arduino community often calls source code for programs a "sketch", we will use the same convention below). If you have followed the Arduino IDE Setup (/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup) tutorial, you'll be able to open the sketch below by choosing `ros_lib -> HelloWorld` from the Arduino `examples` menu.

This should open the following code in your IDE:

Toggle line numbers

```
 1 /*
 2  * rosserial Publisher Example
 3  * Prints "hello world!"
 4  */
 5
 6 #include <ros.h>
 7 #include <std_msgs/String.h>
 8
 9 ros::NodeHandle nh;
10
11 std_msgs::String str_msg;
12 ros::Publisher chatter("chatter", &str_msg);
13
14 char hello[13] = "hello world!";
15
16 void setup()
17 {
18   nh.initNode();
19   nh.advertise(chatter);
20 }
21
22 void loop()
23 {
24   str_msg.data = hello;
25   chatter.publish( &str_msg );
26   nh.spinOnce();
27   delay(1000);
28 }
```

1.2 The Code Explained

Now, let's break the code down.

Toggle line numbers

```
 6 #include <ros.h>
 7 #include <std_msgs/String.h>
 8
```

As a part of every ROS Arduino program, you need to include the `ros.h` header file and header files for any messages that you will be using.

Toggle line numbers

```
 9 ros::NodeHandle nh;
```

Next, we need to instantiate the node handle, which allows our program to create publishers and subscribers. The node handle also takes care of serial port communications.

Toggle line numbers

```
11 std_msgs::String str_msg;
12 ros::Publisher chatter("chatter", &str_msg);
```

We need to instantiate the publishers and subscribers that we will be using. Here we instantiate a Publisher with a topic name of "chatter". The second parameter to Publisher is a reference to the message instance to be used for publishing.

Toggle line numbers

```
16 void setup()
17 {
18   nh.initNode();
19   nh.advertise(chatter);
20 }
```

In the Arduino setup function you then need to initialize your ROS node handle, advertise any topics being published, and subscribe to any topics you wish to listen to.

Toggle line numbers

```
22 void loop()
23 {
24   str_msg.data = hello;
25   chatter.publish( &str_msg );
26   nh.spinOnce();
27   delay(1000);
28 }
```

Finally, in the loop function, the node publishes "Hello World" and calls `ros::spinOnce()` where all of the ROS communication callbacks are handled.

1.3 Uploading the Code

To upload the code to your Arduino, use the upload function within the Arduino IDE. This is no different from uploading any other sketch.

1.4 Running the Code

Now, launch the roscore (/roscore) in a new terminal window:

```
roscore
```

Next, run the rosserial client application that forwards your Arduino messages to the rest of ROS. Make sure to use the correct serial port:

```
rosrun rosserial_python serial_node.py /dev/ttyUSB0
```

Finally, watch the greetings come in from your Arduino by launching a new terminal window and entering :

```
rostopic echo chatter
```

2. Further Reading

Please see rosserial/Overview (/rosserial/Overview/Publishers%20and%20Subscribers) for more information on publishers and subscribers. Also see limitations (/rosserial/Overview/Limitations) for information about more complex data types.

Wiki: rosserial_arduino/Tutorials/Hello World (last edited 2015-05-29 16:58:26 by MahdiehNejati (/MahdiehNejati))