Uso del paquete rosserial con Arduino

En este tutorial se explica como utilizar el paquete rosserial con Arduino, cuando el maestro de ROS se encuentra en una Raspberry Pi con Raspbian.

Instalar el IDE de Arduino

En primer lugar hay que instalar el entorno de desarrollo de Arduino

```
$ sudo apt-get install arduino-core
$ sudo apt-get install arduino
$ arduino
```

Para que se cree el espacio de trabajo de Arduino hay que ejecutar el programa una vez. Se hace simplemente con

```
$ arduino
```

Si se está usado Raspbian Lite, es, decir, desde la línea de comando, es necesario instalar al menos un escritorio virtual para poder ejecutar la aplicación.

Más información sobre esto en https://www.raspberrypi.org/forums/viewtopic.php?f=66&t=133691.

Hecho esto se crea un directorio en /pi/home llamado /sketchbook. Ahí se guardan las librerías y sketches de Arduino.

Instalar el paquete comm-msgs

En general para instalar paquetes en ROS se usa la herramienta catkin. Eso se hará con rosserial pero antes hay que instalar el paquete common_msgs. Este paquete es necesario porque hay otros paquetes en rosserial que usan mensajes que no viene por defecto en la instalación de ROS.

Este paquete se puede instalar en el mismo espacio de trabajo de ROS o en cualquier otro, el proceso es el mismo.

En la línea de comando se va hasta el directorio source del espacio de trabajo

```
$ cd ~/ros_catkin_ws/src
```

Se copia el código fuente de la librería

```
$ git clone https://github.com/ros/common_msgs
```

Se vuelve al directorio del espacio de trabajo y se compila con catkin

```
$ cd ~/ros_catkin_ws
$ catkin_make
```

Instalar rosserial

El paquete rosserial se instala igual que el common_msgs.

En la línea de comando se va hasta el directorio source del espacio de trabajo

```
$ cd ~/ros_catkin_ws/src
```

Como siempre después de compilar hay que actualizar las fuentes del bash

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

Se copia el código fuente de la librería

```
$ git clone https://github.com/ros-drivers/rosserial.git
```

Se vuelve al directorio del espacio de trabajo y se compila con catkin_make install porque catkin_make produce algunos errores

```
$ cd ~/ros_catkin_ws
$ catkin_make install
$ source ~/ros_catkin_ws/devel/setup.bash
```

Ahora resta crear eliminar el directorio ros_lib en /sketches/libraries y crear las librerías para arduino usando rosserial.

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

Si se va a usar otra computadora para programar la Arduino hay que exportar la carpeta ~/sketchbook/libraries/ros_lib a dicha computadora.

Cada vez que se crean nuevos paquetes, nodos, mensajes, servicios, etc. que se usen con Arduino hay que repetir el proceso de creación de las librerías.

Usar rosserial

Además de lo hecho, para poder comunicar la raspberry con arduino usando rosserial hay que instalar la librería pyserial en la raspberry

```
$ sudo apt-get install python-serial
```

En cada sketch de Arduino que se quiera usar hay que incluir los archivos de cabecera necesarios de /ros_lib. El único indispensable es ros.h.

Antes de poder iniciar un nodo en Arduino hay que habilitar los permisos de lectura y escritura en el puerto usb al que está conectado la Arduino

```
$ sudo chmod a+rw /dev/ttyACM0 (usar el puerto correspondiente)
```

Para ejecutar el nodo de Arduino se usa en el puerto y con el baudrate correspondiente

```
$ rosrun rosserial_python serial_node.py /dev/ttyACM0 _baud=9600
```

.

Mas información sobre rosserial y Arduino en http://wiki.ros.org/rosserial_arduino/Tutorials.