

UNIVERSIDAD POLITECNICA DE LA ZONA METROPOLITANA DE GUADALAJARA

DINAMICA Y CONTROL DE ROBOTS



TAREA #7

INGENIERIA MECATRONICA 9°B

MAESTRO:

CARLOS ENRIQUE MORAN GARABITO

ALUMNO:

ALEXIS ISRAEL VIORATO ARAMBULA

OBJETIVO:

HACER UNA COMUNICACIÓN ENTRE ROS Y ARDUINO. UNA VEZ TENIENDO LA CONEXIÓN HACER LO MISMO CON UN SENSOR EN OTRO MICROCONTROLADOR.

MATERIAL:

- COMPUTADORA CON UBUNTU
- CABLES DE CONEXIÓN
- FREESCALE
- ARDUINO
- SENSOR LDR

DESARROLLO

1. Abrir la terminal de Ubuntu y hacer la instalación de roserial con el siguiente comando.

```
pi@raspberrypi:~ $ export ROS_IP=192.168.0.15
pi@raspberrypi:~ $ roscore
... logging to /home/pi/.ros/log/fadd46ea-7550-11e8-9f41-b827eba8d49c/roslaunch-raspberrypi-1576.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.0.15:33627/
ros_comm version 1.14.1

SUMMARY
=====

PARAMETERS
* /rostdistro: melodic
* /rosversion: 1.14.1

NODES

auto-starting new master
process[master]: started with pid [1586]
ROS_MASTER_URI=http://192.168.0.15:11311/

setting /run_id to fadd46ea-7550-11e8-9f41-b827eba8d49c
process[rosout-1]: started with pid [1600]
started core service [/rosout]
```

2. Una vez estando en ros debemos de hacer los siguientes pasos:

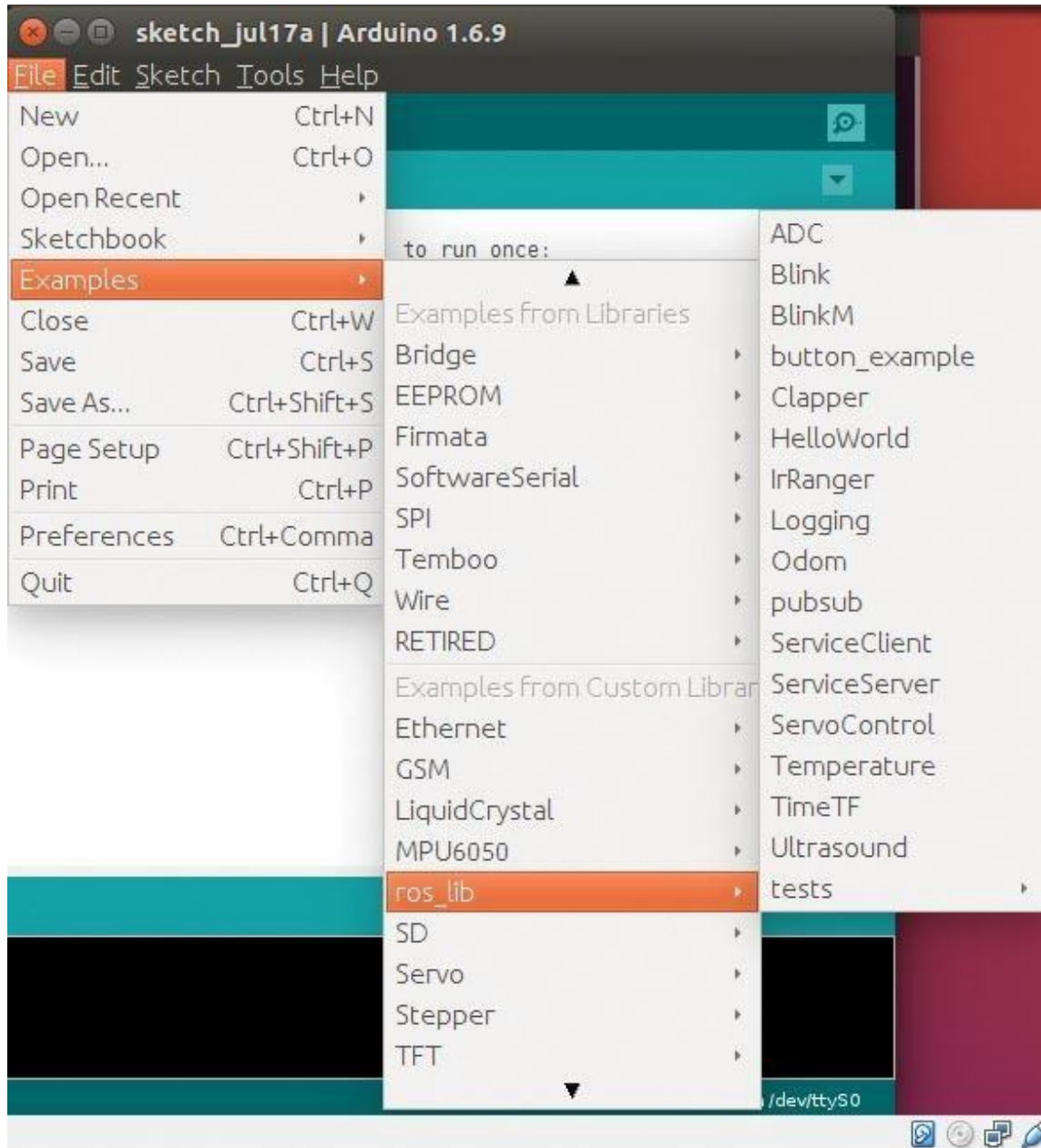
1.- poner esta línea de texto en la terminal para empezar a instalar las librerías de Arduino.

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

2.- enseguida creamos un catkin en Arduino con los siguientes comandos:

```
git clone https://github.com/ros-drivers/rosserial.git cd <ws>
catkin_make rosrund rosserial_arduino make_libraries.py .
rosrund rosserial_arduino make_libraries.py .
```

3.- reiniciamos el dispositivo y ya deberíamos de tener las librerías de ros en Arduino.



4.- Abrimos el ejemplo de Blink el cual sería el del led y nos abrirá un código como el siguiente:

This should open the following code in your IDE:

```
[des]activar nros. de línea
1  /*
2  * rosserial Subscriber Example
3  * Blinks an LED on callback
4  */
5
6  #include <ros.h>
7  #include <std_msgs/Empty.h>
8
9  ros::NodeHandle nh;
10
11 void messageCb( const std_msgs::Empty& toggle_msg){
12   digitalWrite(13, HIGH-digitalRead(13)); // blink the led
13 }
14
15 ros::Subscriber<std_msgs::Empty> sub("toggle_led", &messageCb );
16
17 void setup()
18 {
19   pinMode(13, OUTPUT);
20   nh.initNode();
21   nh.subscribe(sub);
22 }
23
24 void loop()
25 {
26   nh.spinOnce();
27   delay(1);
28 }
```

1.2 The Code Explained

Now, let's break the code down.

4.- para correr para correr el programa de Arduino en ros usaremos los siguientes comandos:

```
roscore
```

después ubicamos el puerto y ponemos:

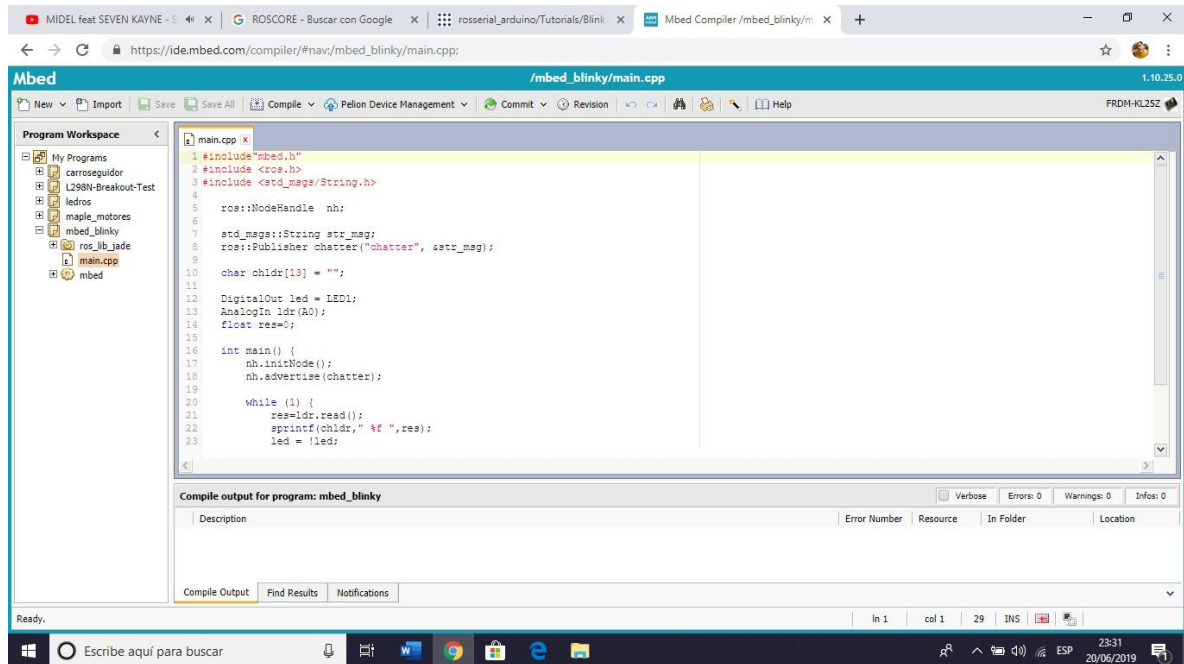
```
roslaunch rosserial_python serial_node.py /dev/ttyUSB0
```

y por ultimo:

```
rostopic pub toggle_led std_msgs/Empty --once
```

5.-. una vez hecho lo de Arduino en mbed importamos la librería de ros y hacemos un código para un sensor. Yo utilice el LDR.

CODIGO:



6.- Ya teniendo el código del sensor en mbed con ros corremos el programa utilizando los mismos comandos que Arduino solo que esta vez imprimiremos caracteres.

```

roscore    rosrund    roserial_python    serial_node.py
/dev/ttyUSB0 rostopic echo chatter

```

y ya queda lista la tarea.

RESULTADO:

