TAREA 7 (LEY DE CONTROL)

DINAMICA DE ROBOTS

UNIVERSIDAD POLITÉCNICA
DE LA ZONA METROPOLITANA DE GUADALAJARA

ALUMNO: FRANCISCO JAVIER HERNANDEZ MORALES
MAESTRO: CARLOS ENRIQUE MORAN GARABITO

DINAMICA DE ROBOTS

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

MATERIAL:

- COMPUTADORA
  - PROGRAMA DE ARDUINO
- CABLES DE CONEXIÓN
- FREESCALE
- ARDUINO
- SENSOR LDR

PROCEDIMIENTO: PRIMERAMENTE, LO QUE SE HACE ES CORRER ROS.

```
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
 * /rosdistro: 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]
```

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

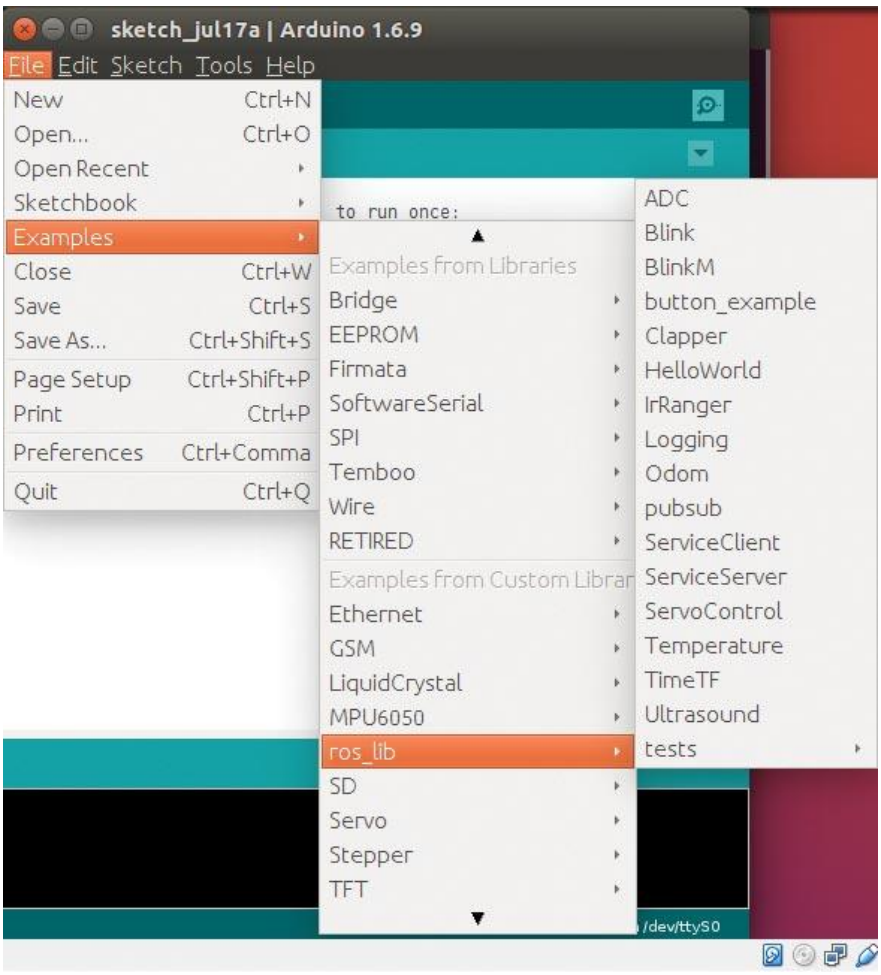
1.- poner esta línea de texto en la terminal para empezar a instalar las librerias de Arduino.

```
sudo apt-get install ros-indigo-rosserial-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
rosrun rosserial_arduino make_libraries.py .
rosrun rosserial_arduino make_libraries.py .
```

3.- reiniciamos el dispositivo y ya deberíamos de tener las librerias 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

```
/*
 * rosserial Subscriber Example
 * Blinks an LED on callback
 */

#include <ros.h>
#include <std_msgs/Empty.h>

ros::NodeHandle nh;

void messageCb( const std_msgs::Empty& toggle_msg){
  digitalWrite(13, HIGH-digitalRead(13));   // blink the led
}

ros::Subscriber<std_msgs::Empty> sub("toggle_led", &messageCb );

void setup()
{
  pinMode(13, OUTPUT);
  nh.initNode();
  nh.subscribe(sub);
}

void loop()
{
  nh.spinOnce();
  delay(1);
}
```

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:

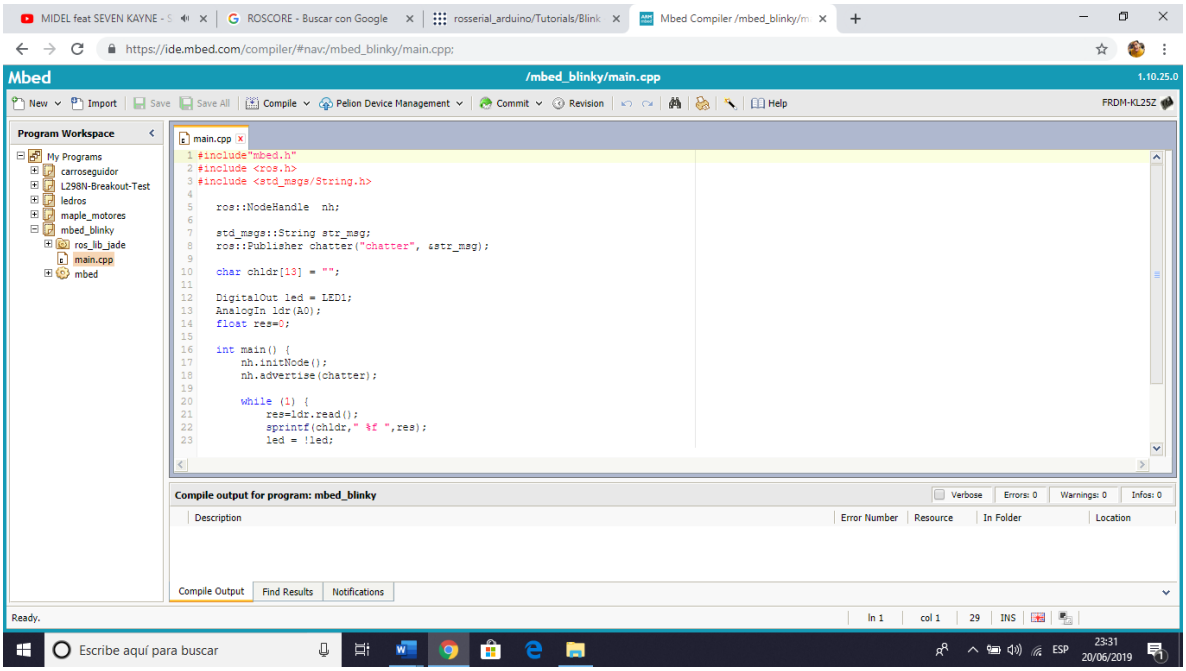
```
rosrun rosserial_python serial_node.py /dev/ttyUSB0
```

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
rosrun rosserial_python serial_node.py /dev/ttyUSB0
rostopic echo chatter
```

y ya queda lista la tarea.

RESULTADO:

DINAMICA DE ROBOTS

FRANCISCO JAVIER HERNANDEZ MORALES