



SUTD ORGANIZATION OF
AUTONOMOUS ROBOTICS

ROS – Arduino workshop

Objective

- Nodes and topics
- ROS node: rosserial
- Arduino IDE: ROS Serial library
- Understanding rosserial
- Code simple pub/sub program

Node and Topics

Nodes



Executable codes that does things. Each node have its own job(s)

Node and Topics

Topic



Usually represented by an arrow or oval. Each topic has a unique message type and is being to communicate between nodes.

Node and Topics



Wood chopper

pile of woods



Factory

wooden boards



Student

Every node only accepts a specific type of message. If the topic's message is wrong, the node cannot understand

Node and Topics



Every node only accepts a specific type of message. If the topic's message is wrong, the node cannot understand

Node and Topics

Wheels

Path
planner

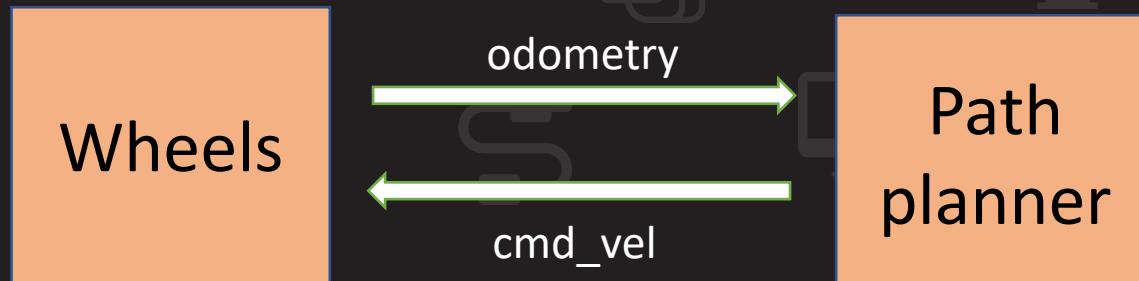
odometry

cmd_vel

odometry - position of the robot
cmd_vel - speed of the motor

Publisher and Subscriber

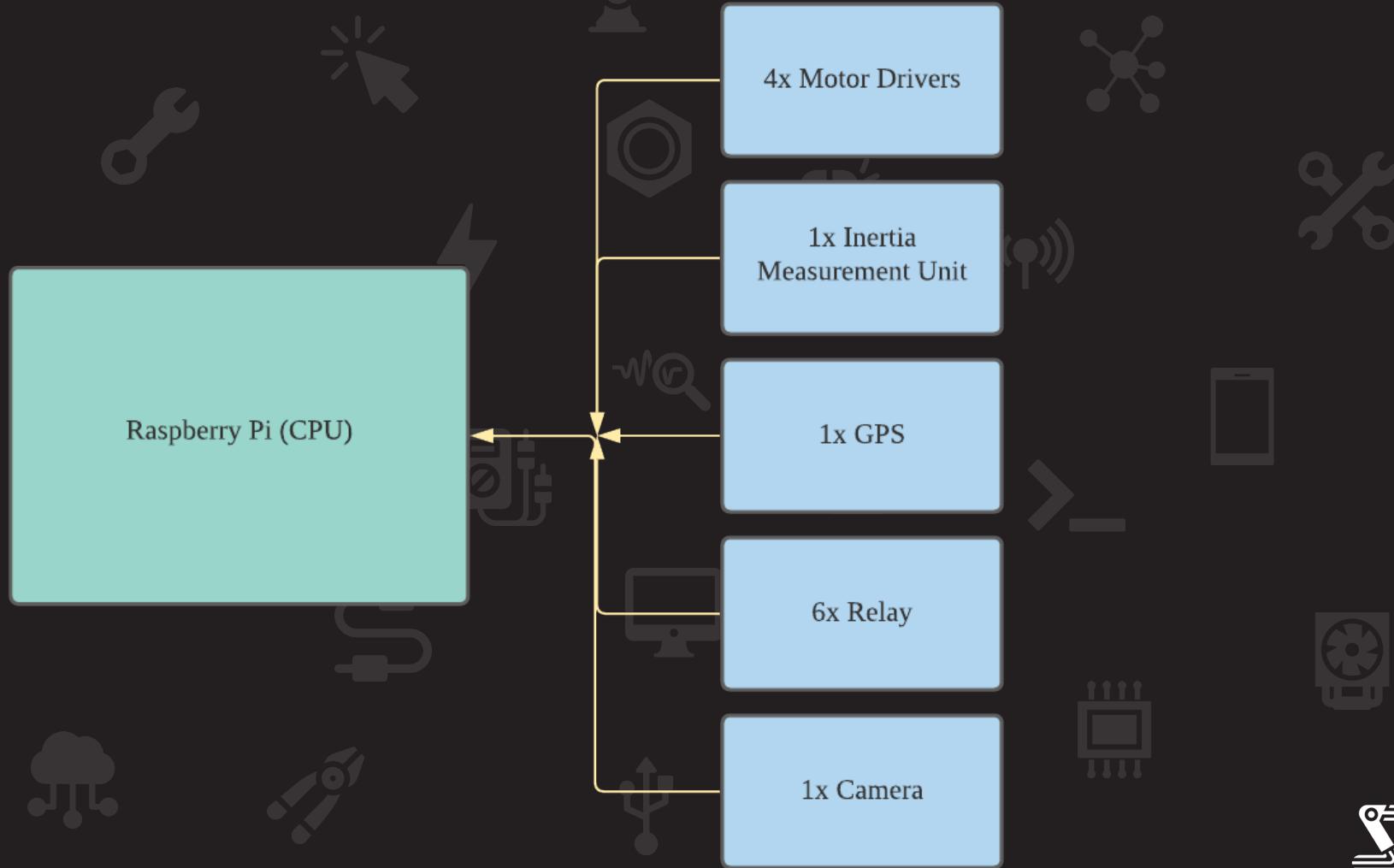
- Nodes can *publish* and *subscribe* to a topic
- “Wheels”(node) publish to “odometry”(topic) and subscribe to “cmd_vel”(topic)



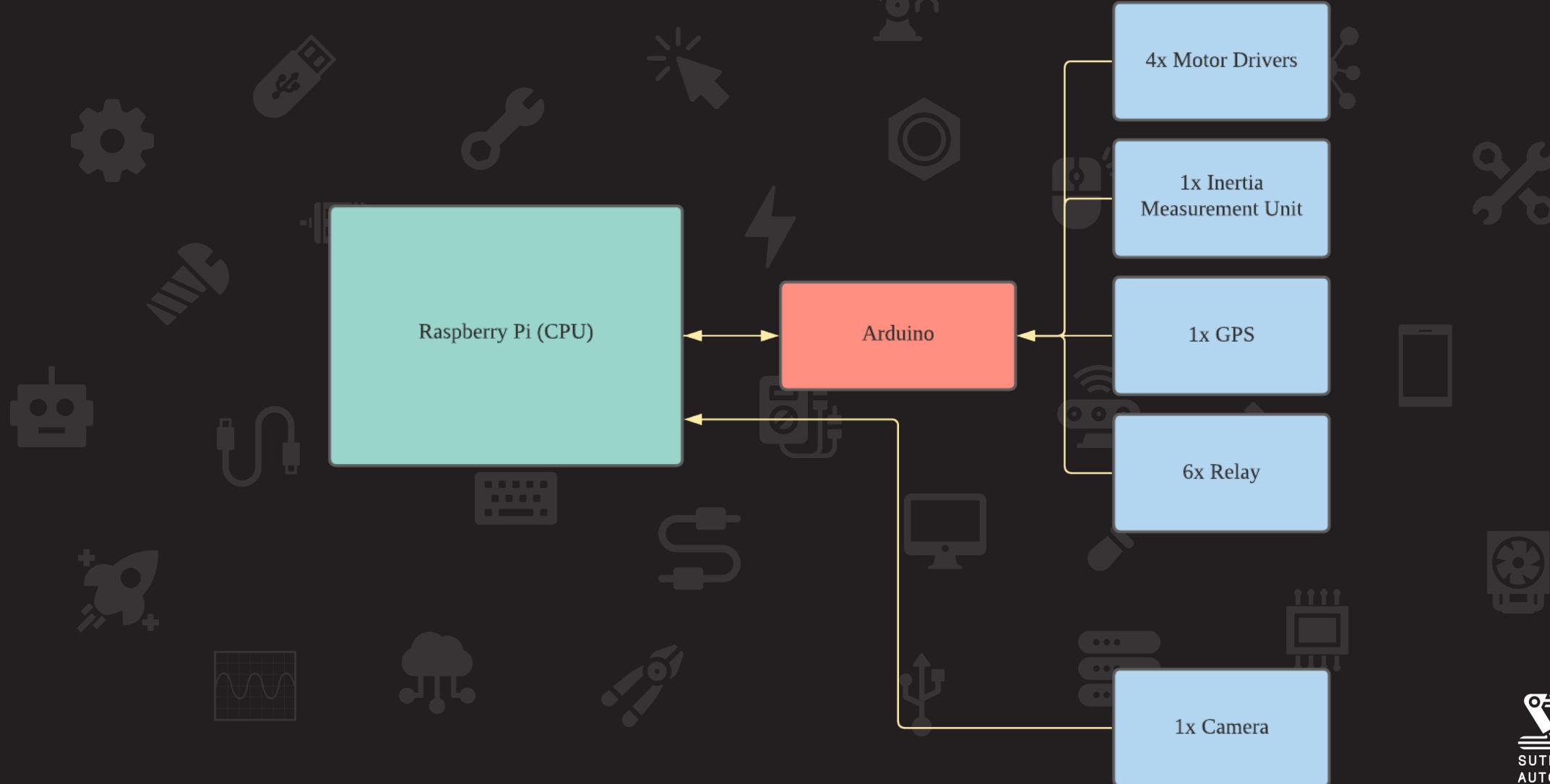
Parcel Delivery Robot



Sensor & Actuators Architecture



Sensor & Actuators Architecture



Motivation

- Unloading processing burden from CPU
- Most CPU only supports 3.3V hardware application whereas most hardware are 5V
- More hardware libraries available in Arduino

```
boonpin@boonpin-VB:~/catkin_ws/src$ git clone https://github.com/ros-drivers/ro  
serial
```

Install rosserial

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

Downloads



Arduino IDE 1.8.13

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file
Windows app Win 8.1 or 10 [Get](#)

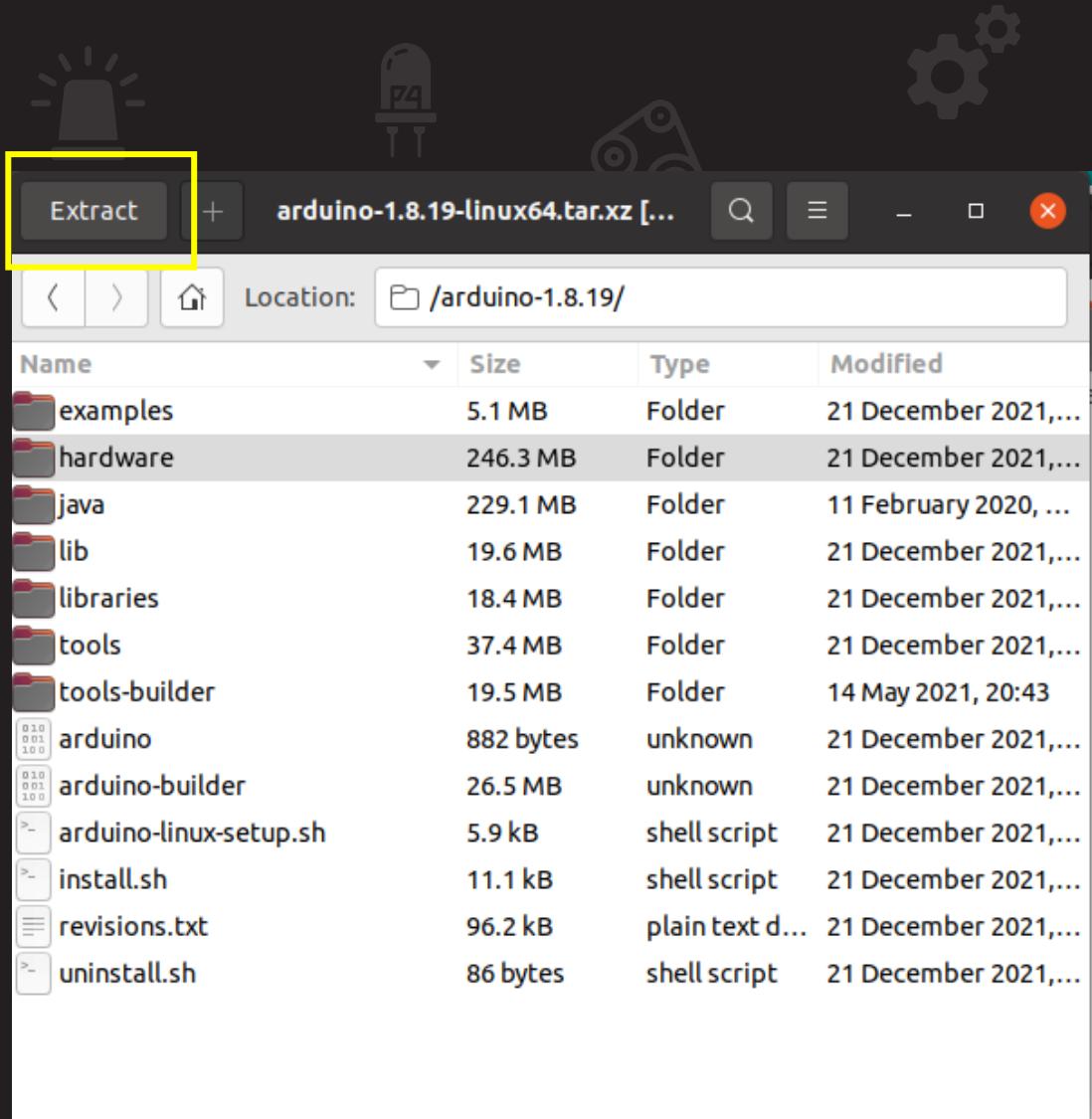
Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

Mac OS X 10.10 or newer

[Release Notes](#) [Checksums \(sha512\)](#)

Install Arduino IDE

<https://www.arduino.cc/en/software/>



Extract the file

".tar.xz" is a compressed file so we must extract it to "Home"

```
boonpin@boonpin-VB:~/arduino-1.8.19$ cd arduino-1.8.19/  
boonpin@boonpin-VB:~/arduino-1.8.19$ sudo ./install.sh
```

Install Arduino

Navigate to the Arduino file and run
“install.sh”

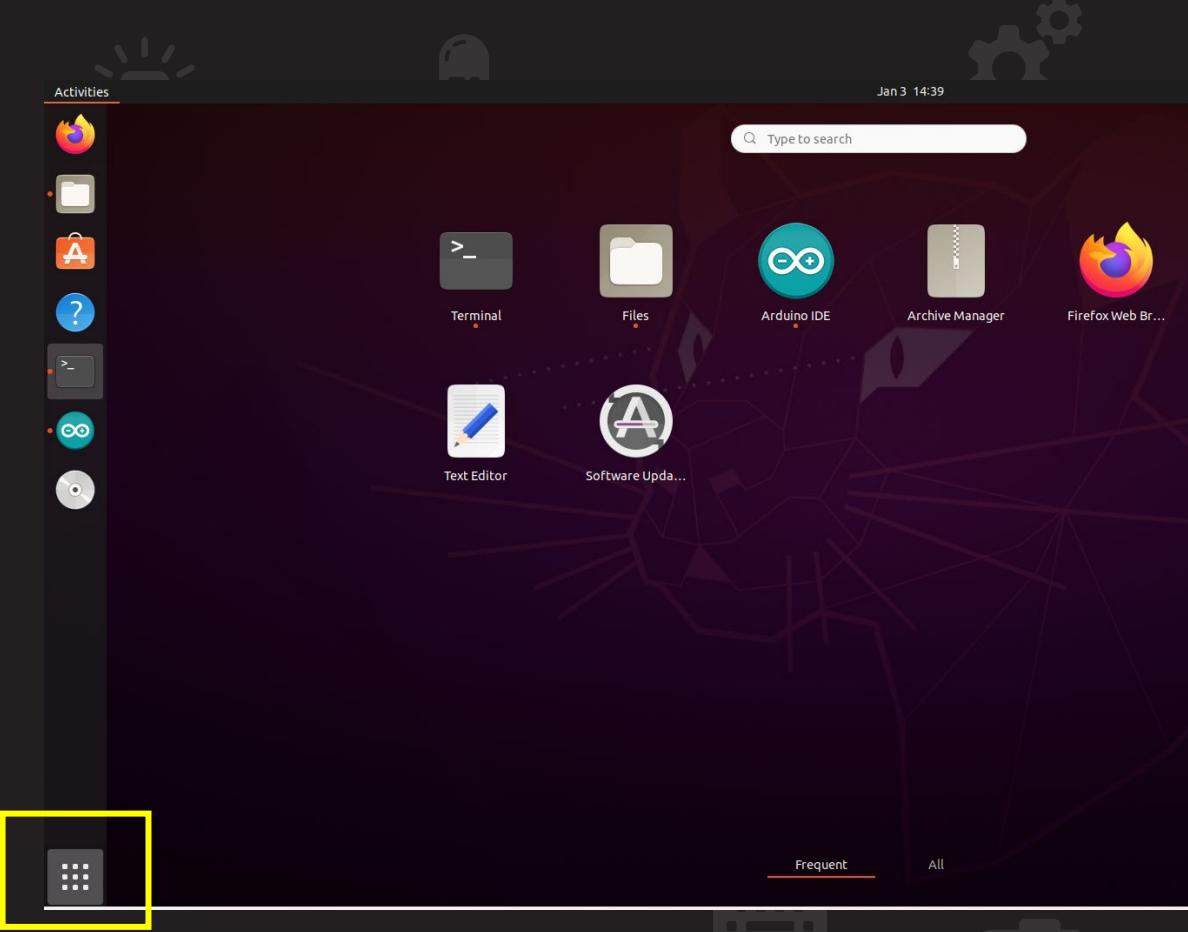
```
cd arduino-1.8.19/  
sudo ./install.sh
```

```
boonpin@boonpin-VB:~/arduino-1.8.19$ cd arduino-1.8.19/  
boonpin@boonpin-VB:~/arduino-1.8.19$ sudo ./install.sh
```

Install Arduino

Navigate to the Arduino file and run
“install.sh”

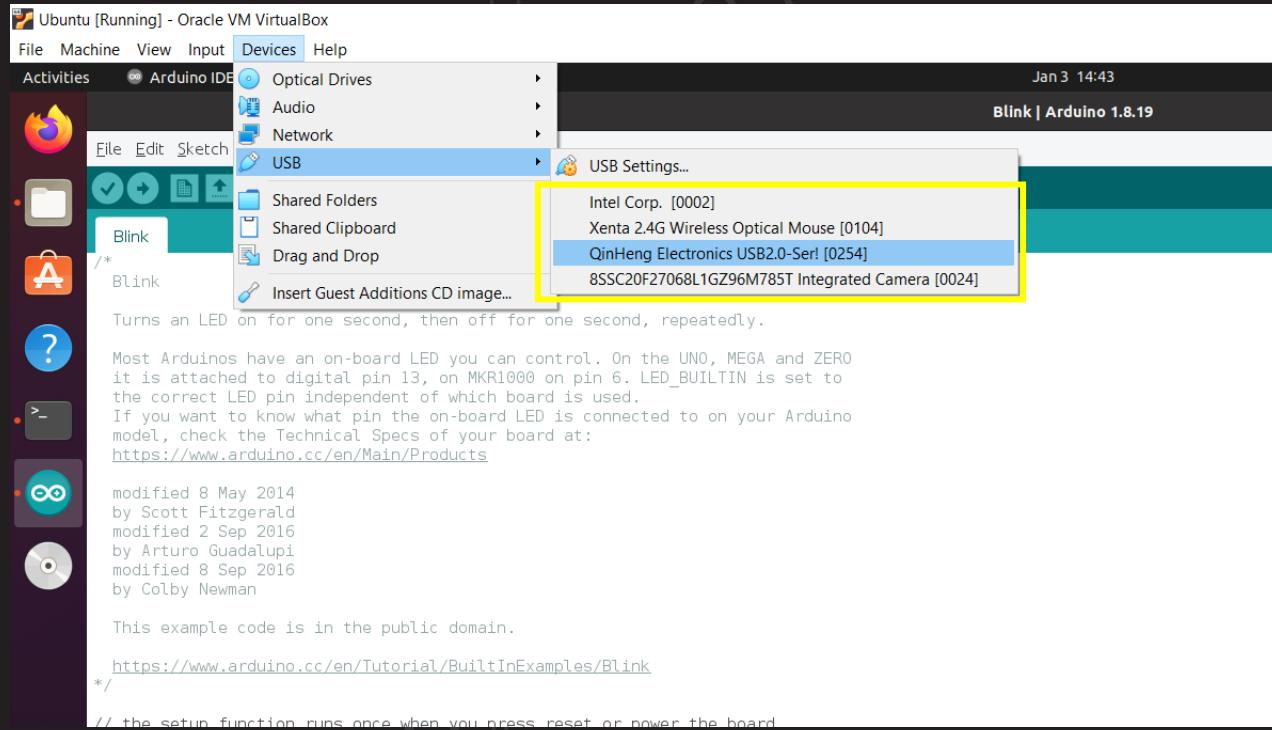
```
cd arduino-1.8.19/  
sudo ./install.sh
```



```
cd arduino-1.8.19/  
sudo ./install.sh
```

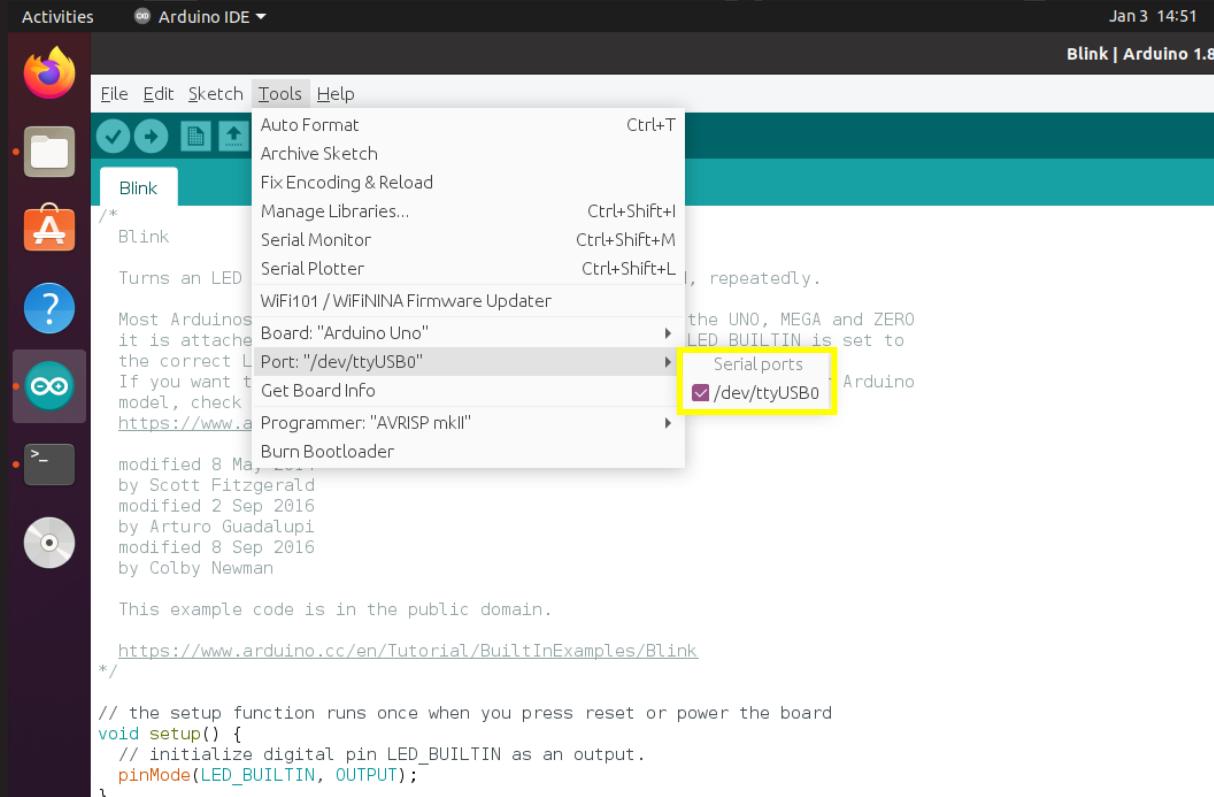
Launch Arduino

Click the icon at the bottom-left corner and search for “Arduino”



USB port

Plug in the Arduino
Select Devices >> USB >> <USB_port>



The screenshot shows the Arduino IDE interface. The title bar reads "Blink | Arduino 1.8.7". The main window displays the "Blink" sketch code. The "Tools" menu is open, and the "Port" dropdown is selected, showing "/dev/ttyUSB0" which is highlighted with a yellow box.

```
/*
 * The setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);    // Turn the LED on (HIGH is the collected voltage)
  delay(1000);                      // Wait for a second
  digitalWrite(LED_BUILTIN, LOW);     // Turn the LED off by setting the voltage LOW
  delay(1000);                      // Wait for a second
}
```

USB port

Go to Tools >> Port

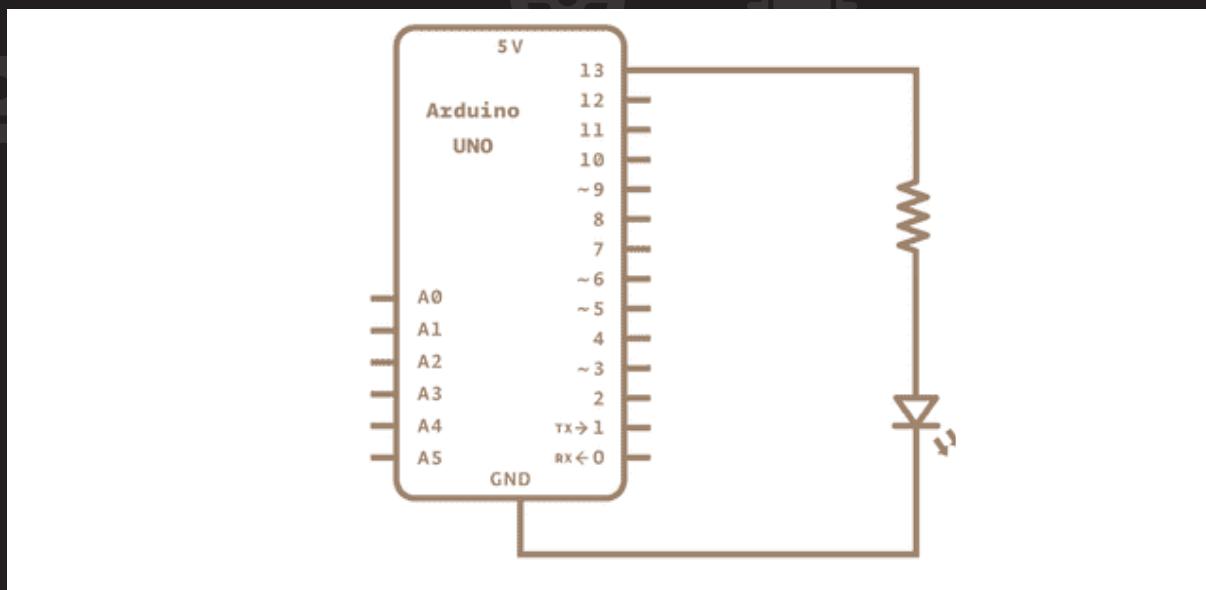
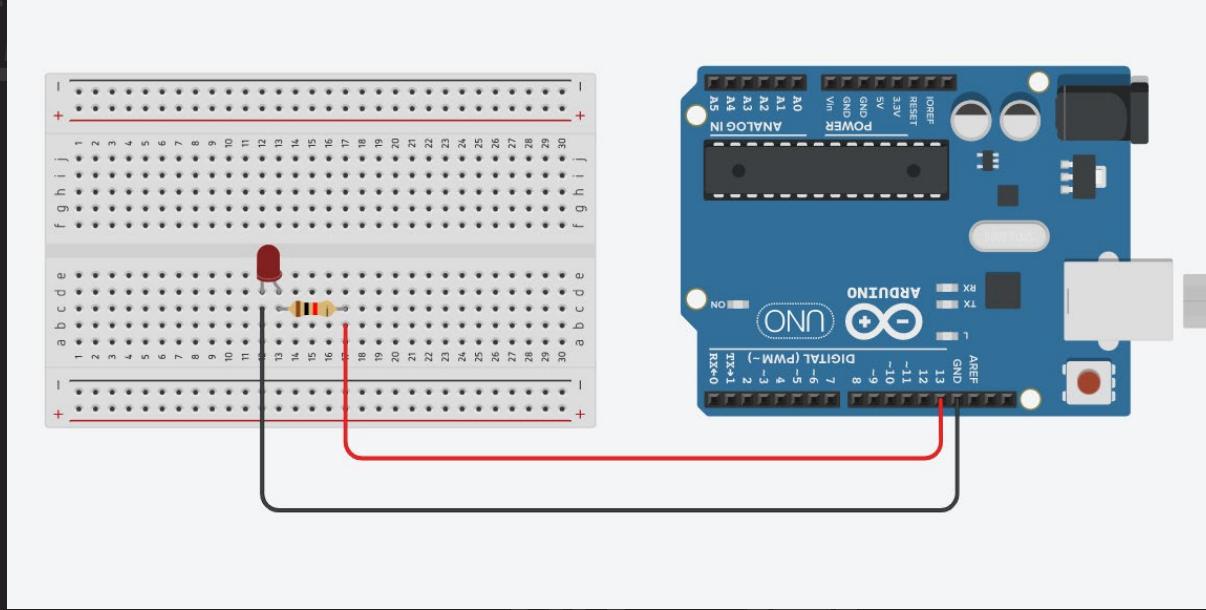
Find out what is the port name for the Arduino
and remember it

```
wargreymon@ubootu: ~
wargreymon@ubootu: ~ 80x24
wargreymon@ubootu:~$ ls -l /dev/ttyACM0
crw-rw----+ 1 root dialout 166, 0 Feb 16 21:29 /dev/ttyACM0
wargreymon@ubootu:~$ sudo usermod -a -G dialout wargreymon
[sudo] password for wargreymon:
wargreymon@ubootu:~$
```

Add user to dialout

reboot to take effect

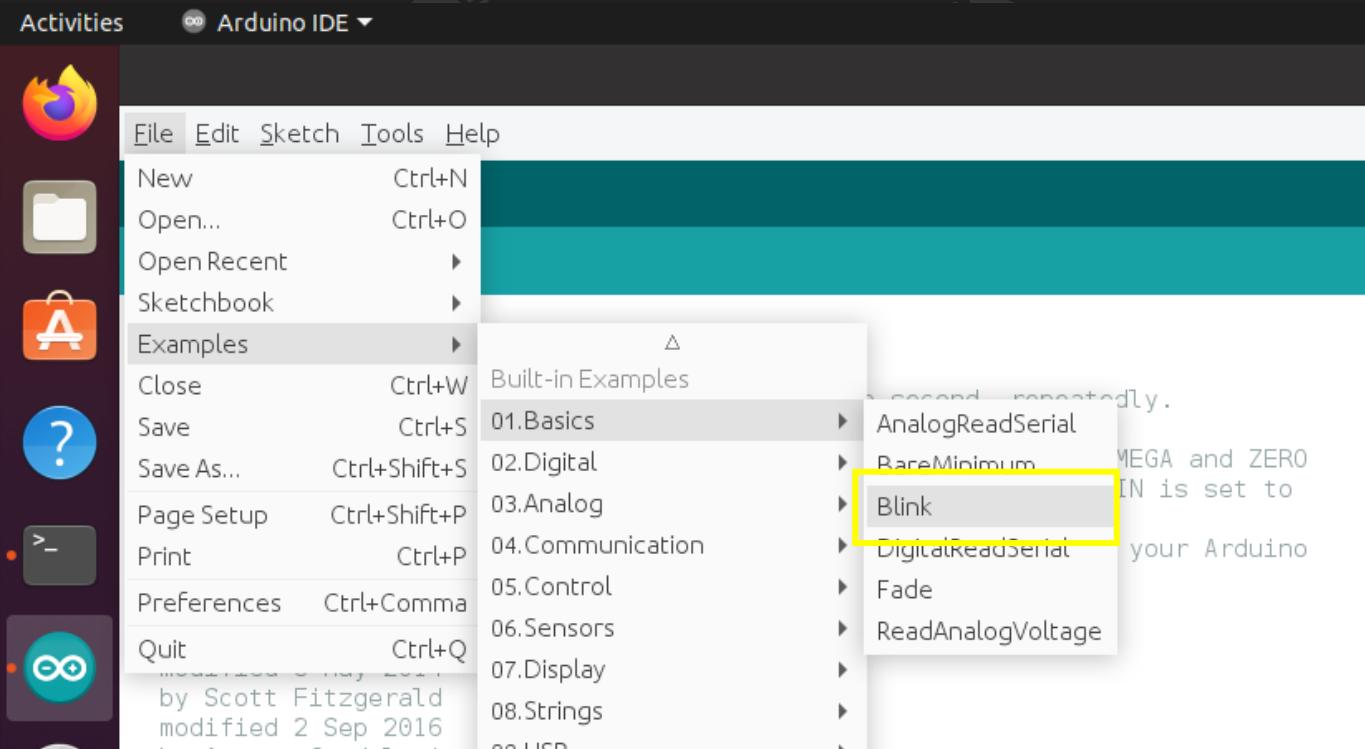
```
ls -l /dev/tty<USB_port> (e.g /dev/ttyUSB0)
sudo usermod -a -G dialout <user>
```



Breadboarding LED

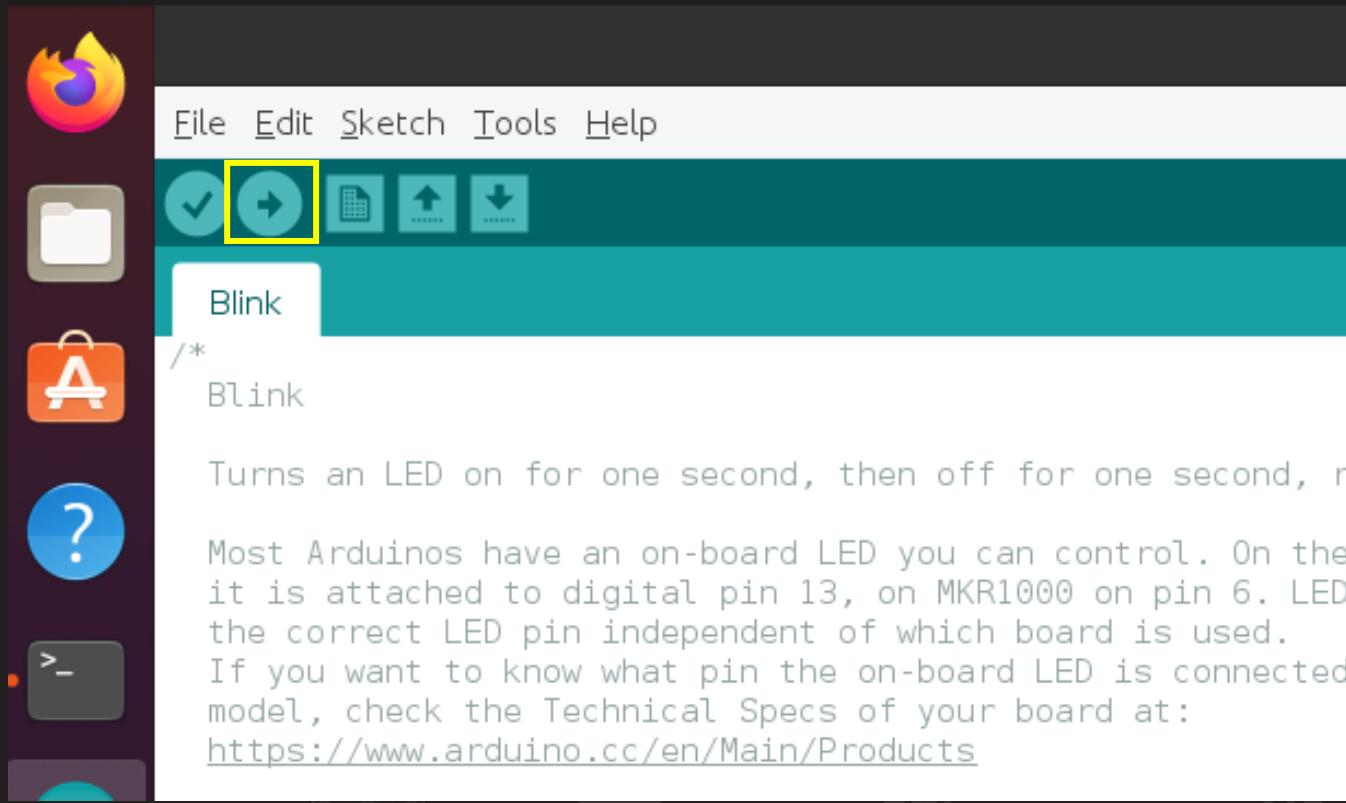
Materials:

- 1x Breadboard
- 1x Arduino (pin 13)
- 1x LEDs
- 1x Resistor



Blinking Code

Open a sample blinking program by going to Examples > Basics > Blink



```
File Edit Sketch Tools Help
Blink
/*
Blink

Turns an LED on for one second, then off for one second, n

Most Arduinos have an on-board LED you can control. On the
it is attached to digital pin 13, on MKR1000 on pin 6. LED
the correct LED pin independent of which board is used.
If you want to know what pin the on-board LED is connected
model, check the Technical Specs of your board at:
https://www.arduino.cc/en/Main/Products
```

Upload

Upload the code by clicking the icon at the top-left corner

Troubleshooting tips:

1. Unplug and plug back in the Arduino
2. Check that you have selected the USB port under Devices >> USB in VirtualBox
3. Check that you have selected the USB port under Tools >> Port in the Arduino software
4. Open terminal and type

sudo chmod 777 /dev/ttyUSB0

Activity 1: Fast blinky LED

- A quick recap of Arduino coding
- Open a new sketch
- Program the LED to blink at an interval of 0.3 second
- 5 minutes

sketch_feb16a | Arduino 1.8.13

File Edit Sketch Tools Help

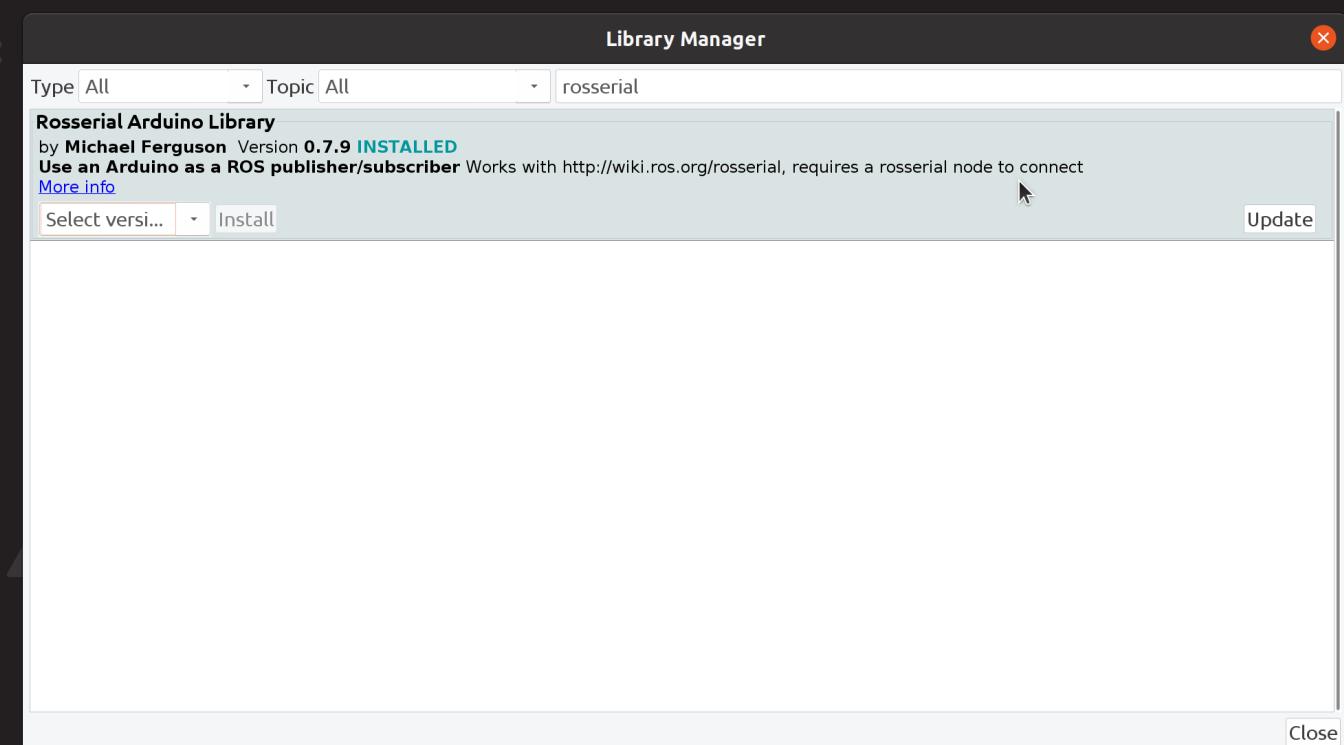
Auto Format Archive Sketch Fix Encoding & Reload Manage Libraries... Serial Monitor Serial Plotter WiFi101 / WiFiNINA Firmware Updater Board: "Arduino Uno" Port Get Board Info Programmer: "AVRISP mkII" Burn Bootloader

sketch_feb16

```
void setup() {  
 // put your setup code here  
}  
  
void loop() {  
 // put your main loop code here  
}
```

1

Arduino Uno

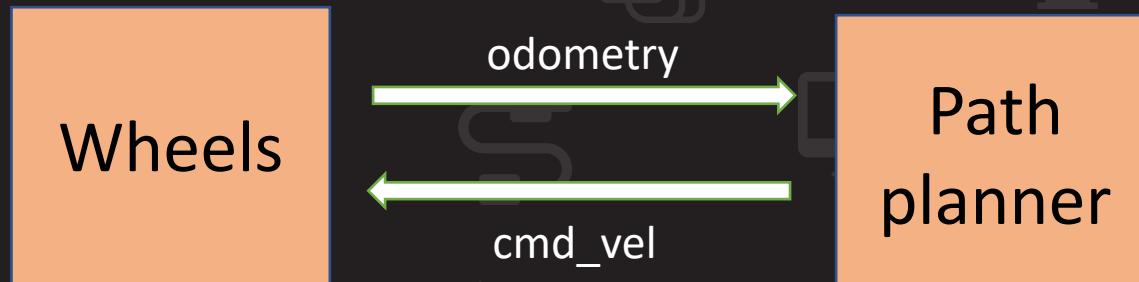


Install Rosserial Arduino Library Version 0.7.9

1. Go to Tools >> Manage Libraries
2. Search “rosserial”
3. Download version 0.7.9
4. Restart Arduino

Recap: Publisher and Subscriber

- Nodes can *publish* and *subscribe* to a topic
- “Wheels”(node) publish to “odometry”(topic) and subscribe to “cmd_vel”(topic)



LED Subscriber

- Serial_node(node) – Arduino node
- /toggle_LED(topic) – topic message to control LED

```
boonpin@boonpin-VB:~/Downloads$ cd Downloads/  
boonpin@boonpin-VB:~/Downloads$ git clone https://github.com/Boonpin97/rosserial_material  
Cloning into 'rosserial_material'...  
remote: Enumerating objects: 8, done.  
remote: Counting objects: 100% (8/8), done.  
remote: Compressing objects: 100% (5/5), done.  
remote: Total 8 (delta 1), reused 8 (delta 1), pack-reused 0  
Unpacking objects: 100% (8/8), 1.50 KiB | 1.50 MiB/s, done.  
boonpin@boonpin-VB:~/Downloads$
```

```
cd Downloads/  
git clone https://github.com/sutd-robotics/ros-ws-2022.git
```

Git clone sample code

File Edit Sketch Tools Help



LED_sub_code_no_comments

```
1 #include <ros.h>
2 #include <std_msgs/String.h>
3
4 ros::NodeHandle nh;
5
6 int led_pin = 13;
7
8 void messageCb(std_msgs::String& toggle_msg) {
9     if (String(toggle_msg.data) == "1") {
10         digitalWrite(led_pin, HIGH);
11     }
12     if (String(toggle_msg.data) == "0") {
13         digitalWrite(led_pin, LOW);
14     }
15 }
16
17 ros::Subscriber<std_msgs::String> sub("toggle_led", messageCb );
```

#include <ros.h>

- include the rosserial library

ros::NodeHandle nh;

- create a handler for this Arduino node

void messageCb(std_msgs::String& toggle_msg)

- message is stored in `toggle_msg.data`
- if `toggle_msg.data` is 1, turn on LED
- if `toggle_msg.data` is 0, turn off LED

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

- create a subscriber object call `sub`
- subscribe to “/toggle_led” topic
- “/toggle_led” is message type of `std_msgs::String`
- once message received, it will run callback function `messageCb`

```
18
19 void setup() {
20 // put your setup code here, to run once:
21 pinMode(led_pin, OUTPUT);
22 nh.initNode();
23 nh.subscribe(sub);
24 }
25
26 void loop() {
27 // put your main code here, to run repeatedly:
28 nh.spinOnce();
29 delay(1);
30 }
```

nh.initNode();

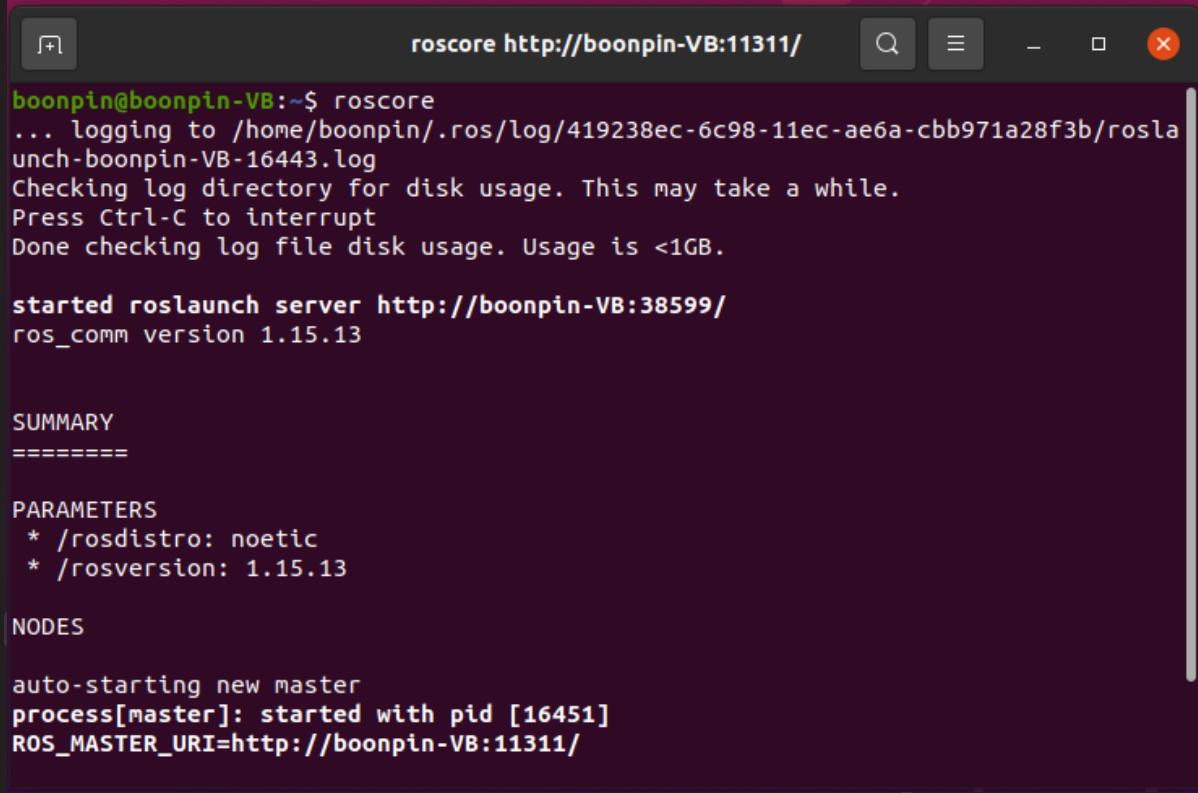
- initialise node

nh.subscribe(sub)

- set **sub** as the node's subscriber

nh.spinOnce()

- check for updates in the rostopic

```
roscore http://boonpin-VB:11311/   
boonpin@boonpin-VB:~$ roscore  
... logging to /home/boonpin/.ros/log/419238ec-6c98-11ec-ae6a-cbb971a28f3b/rosla  
unch-boonpin-VB-16443.log  
Checking log directory for disk usage. This may take a while.  
Press Ctrl-C to interrupt  
Done checking log file disk usage. Usage is <1GB.  
  
started roslaunch server http://boonpin-VB:38599/  
ros_comm version 1.15.13  
  
SUMMARY  
=====
```

PARAMETERS

- * /rosdistro: noetic
- * /rosversion: 1.15.13

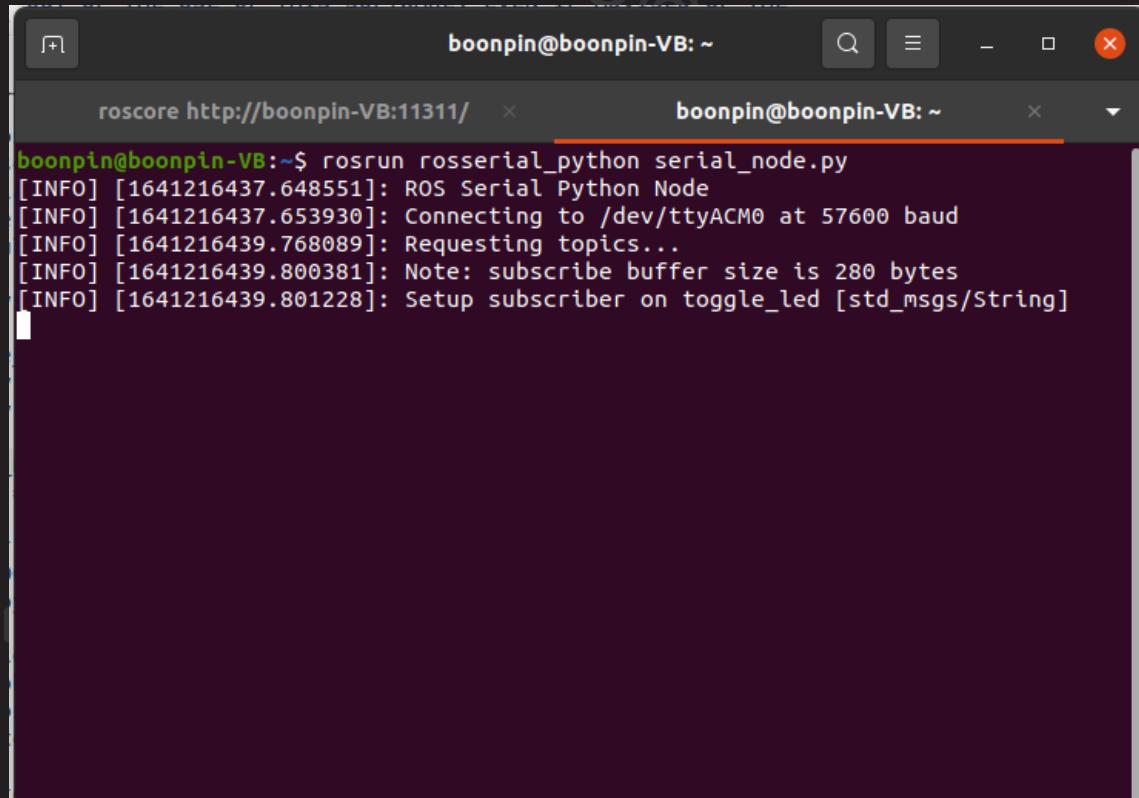
NODES

```
auto-starting new master  
process[master]: started with pid [16451]  
ROS_MASTER_URI=http://boonpin-VB:11311/
```

roscore

Start roscore

Open a terminal and start roscore

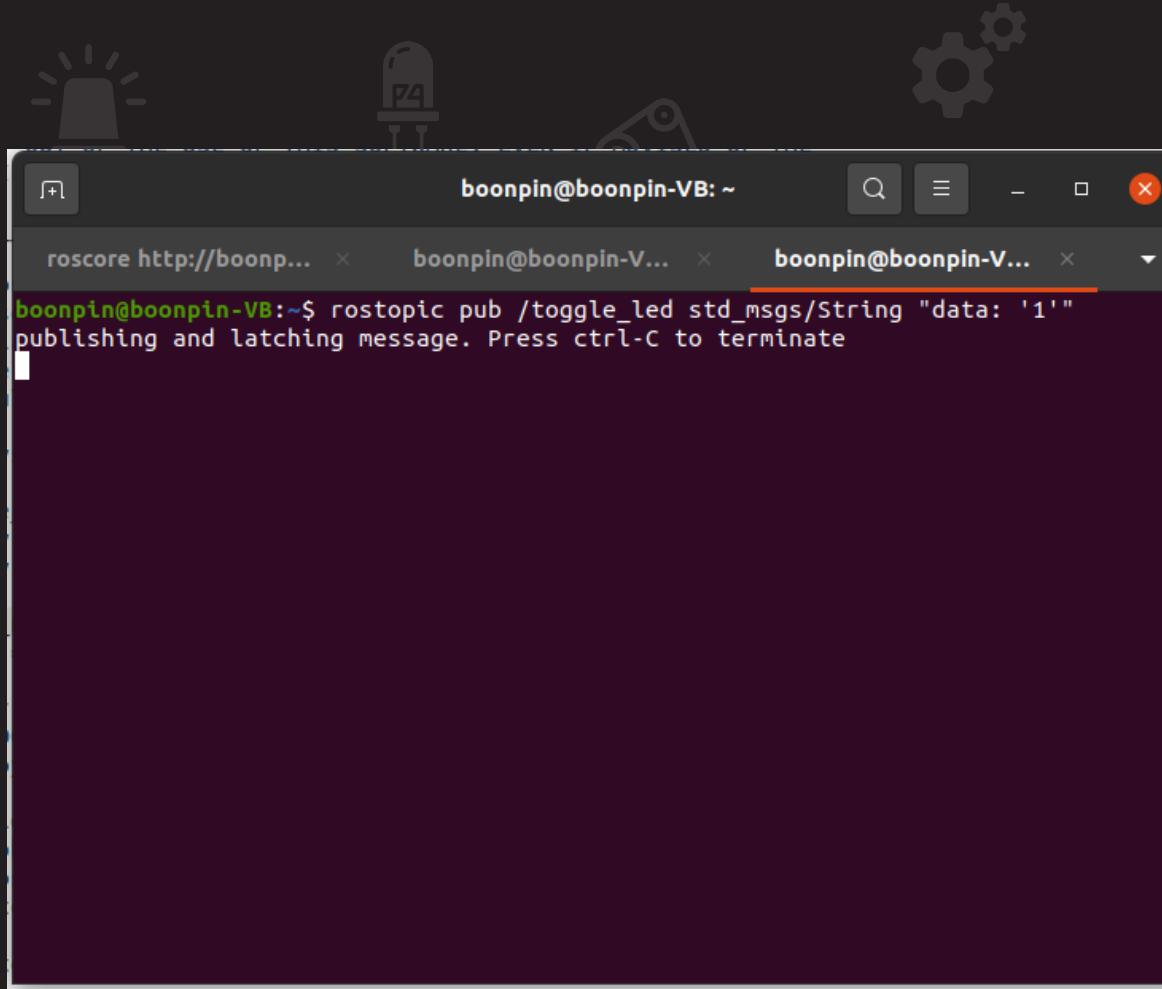


```
boonpin@boonpin-VB: ~ rosrun rosserial_python serial_node.py
[INFO] [1641216437.648551]: ROS Serial Python Node
[INFO] [1641216437.653930]: Connecting to /dev/ttyACM0 at 57600 baud
[INFO] [1641216439.768089]: Requesting topics...
[INFO] [1641216439.800381]: Note: subscribe buffer size is 280 bytes
[INFO] [1641216439.801228]: Setup subscriber on toggle_led [std_msgs/String]
```

Start rosserial

Open a new terminal and run rosserial

```
rosrun rosserial_python serial_node.py /dev/ttyUSB0 (your Arduino USB port)
```



A terminal window titled "boonpin@boonpin-VB: ~" with three tabs. The active tab shows the command:

```
rostopic pub /toggle_led std_msgs/String "data: '1'"
```

The output indicates: "publishing and latching message. Press ctrl-C to terminate".

```
rostopic pub /toggle_led std_msgs/String "data: '1'"
```

Topic's name

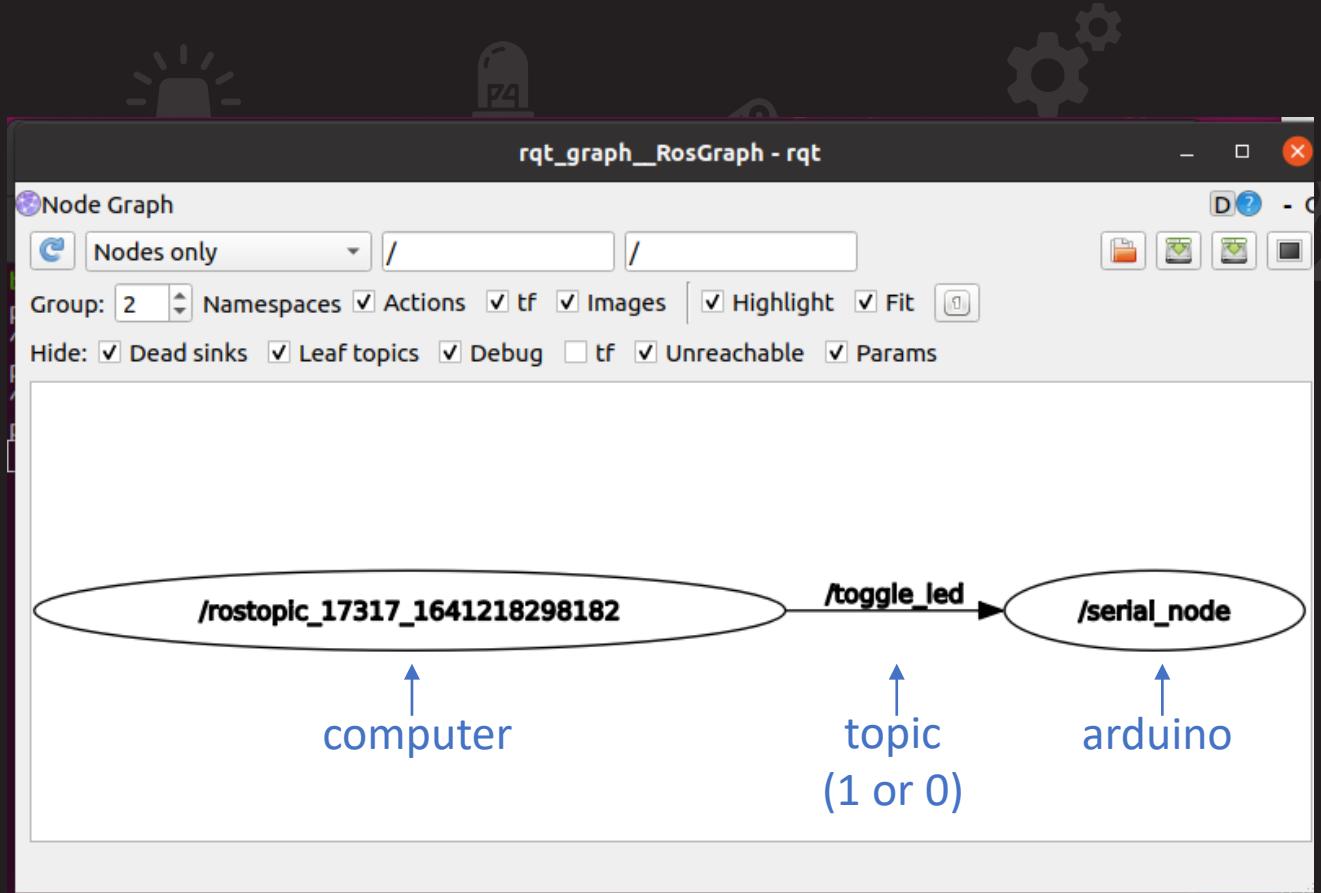
Topic's message type

Message

Publish “/toggle_led”

Publish 0 and 1 to “/toggle_led” to control the LED”

1. Open a new terminal
2. Publish 1 to turn on LED, 0 to turn off



rqt_graph

rqt graph

open rqt graph to understand what nodes and topics are there and how they are communicating with each other

Break Time~

Please be back in 5mins



Chatter Publisher

- Serial_node(node) – Arduino node
- /chatter(topic) – topic message Arduino publishes

```
File Edit Sketch Tools Help
Upload
chatter_pub_code_no_comments
1 #include <ros.h>
2 #include <std_msgs/String.h>
3
4 ros::NodeHandle nh;
5
6 std_msgs::String str_msg;
7 ros::Publisher pub("chatter", &str_msg);
8
9 char msg[13] = "hello world!";
10
11 void setup()
12 {
13     nh.initNode();
14     nh.advertise(pub);
15 }
16
17 void loop()
18 {
19     str_msg.data = msg;
20     pub.publish( &str_msg );
21     nh.spinOnce();
22     delay(500);
23 }
24
```

ros::Publisher pub("chatter",&str_msg);

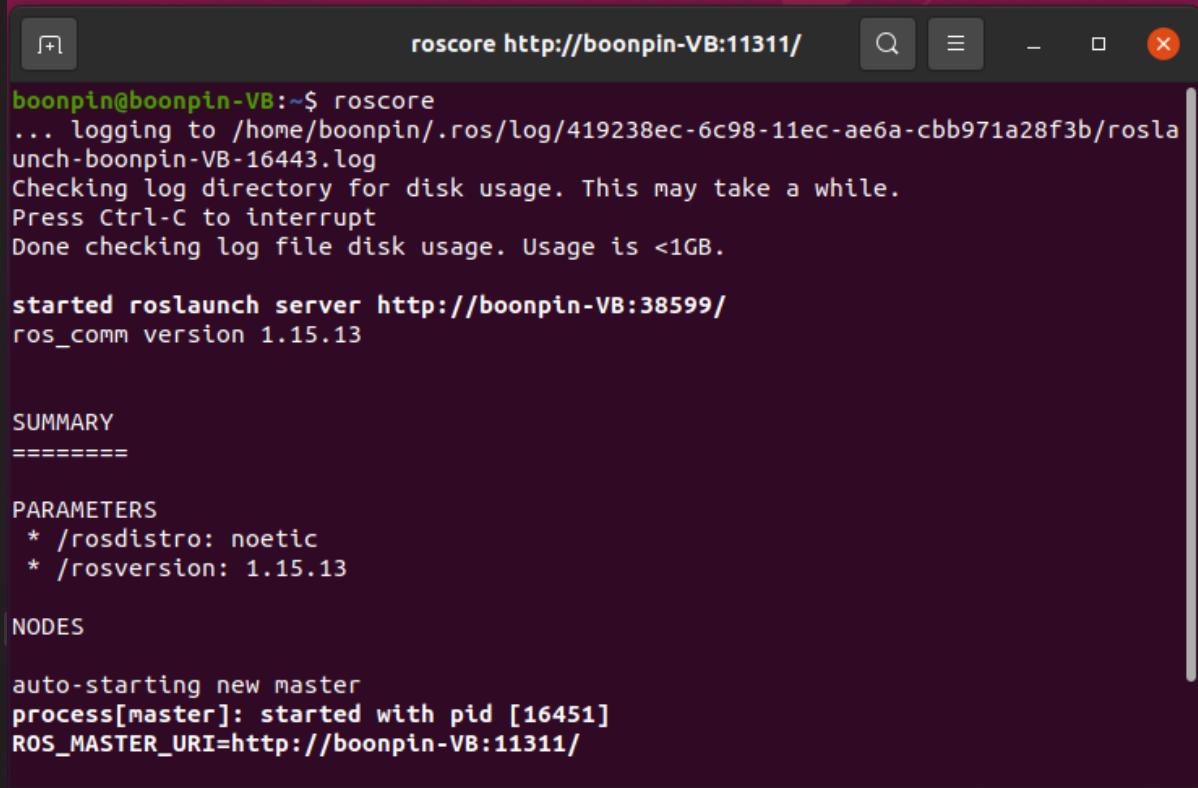
- create a publisher object call pub
- publish a rostopic named “chatter” with the value of &str_msg

nh.advertise(pub);

- set pub as the node's publisher

pub.publish(&str_msg);

- publish pub with the message being &str_msg

```
roscore http://boonpin-VB:11311/   
boonpin@boonpin-VB:~$ roscore  
... logging to /home/boonpin/.ros/log/419238ec-6c98-11ec-ae6a-cbb971a28f3b/rosla  
unch-boonpin-VB-16443.log  
Checking log directory for disk usage. This may take a while.  
Press Ctrl-C to interrupt  
Done checking log file disk usage. Usage is <1GB.  
  
started roslaunch server http://boonpin-VB:38599/  
ros_comm version 1.15.13  
  
SUMMARY  
=====
```

PARAMETERS

- * /rosdistro: noetic
- * /rosversion: 1.15.13

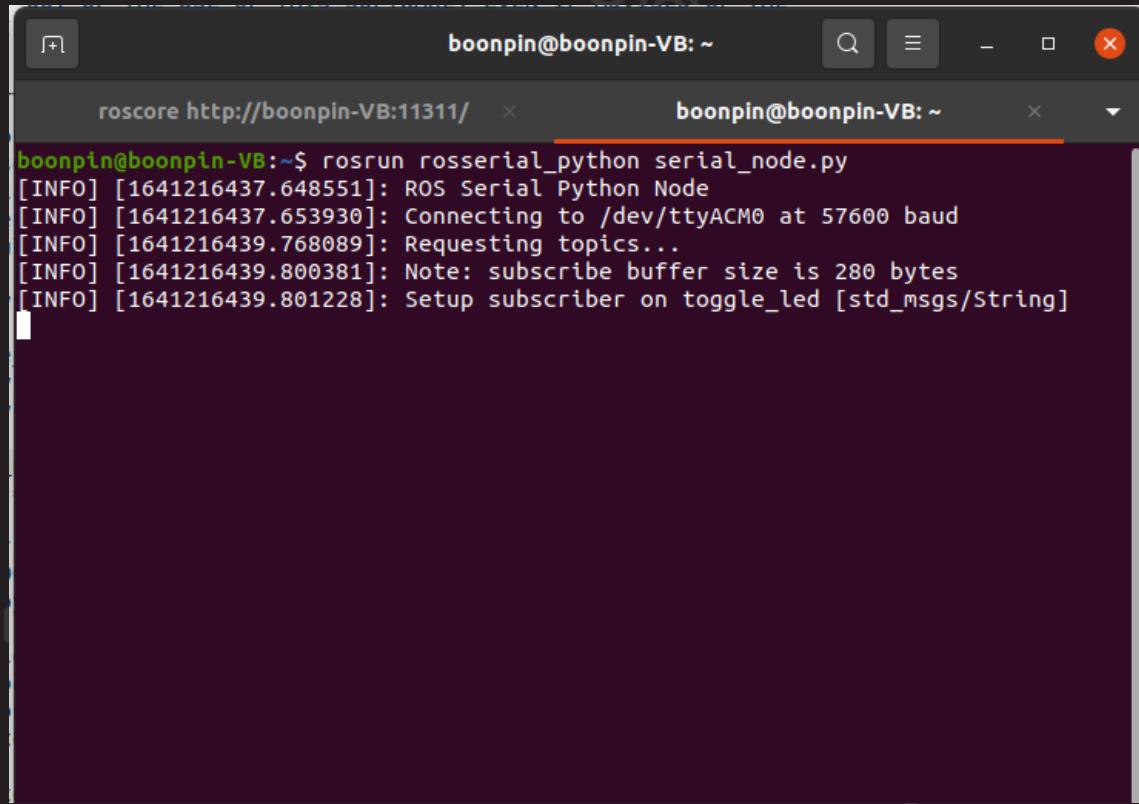
NODES

```
auto-starting new master  
process[master]: started with pid [16451]  
ROS_MASTER_URI=http://boonpin-VB:11311/
```

roscore

Start roscore

Open a terminal and start roscore



```
boonpin@boonpin-VB:~$ rosrun rosserial_python serial_node.py
[INFO] [1641216437.648551]: ROS Serial Python Node
[INFO] [1641216437.653930]: Connecting to /dev/ttyACM0 at 57600 baud
[INFO] [1641216439.768089]: Requesting topics...
[INFO] [1641216439.800381]: Note: subscribe buffer size is 280 bytes
[INFO] [1641216439.801228]: Setup subscriber on toggle_led [std_msgs/String]
```

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

Start rosserial

Open a new terminal and run rosserial

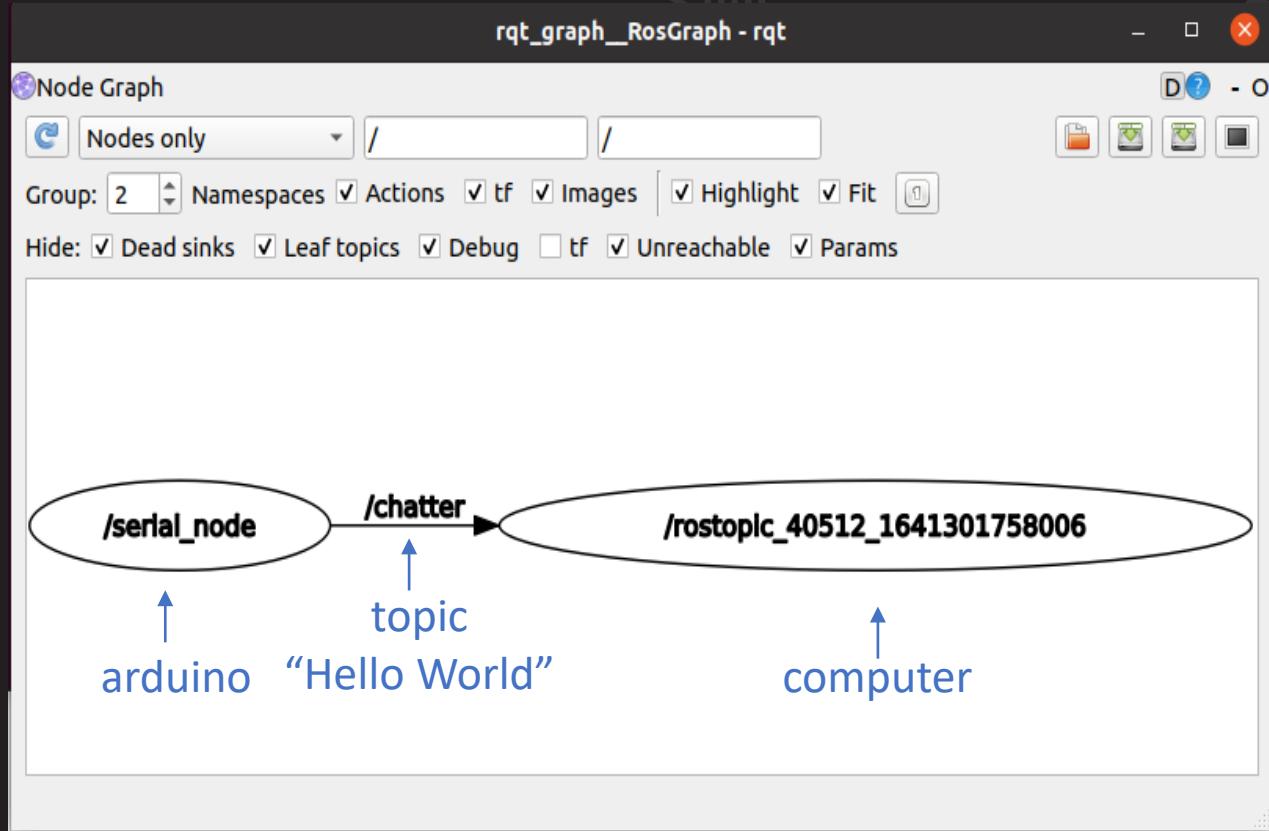
```
rostopic echo /chatter
data: "hello world!"
---
|
```

```
rostopic echo /chatter
```

Topic's name

Echo topic “/chatter”

See what the rostopic “/chatter” is showing



rqt_graph

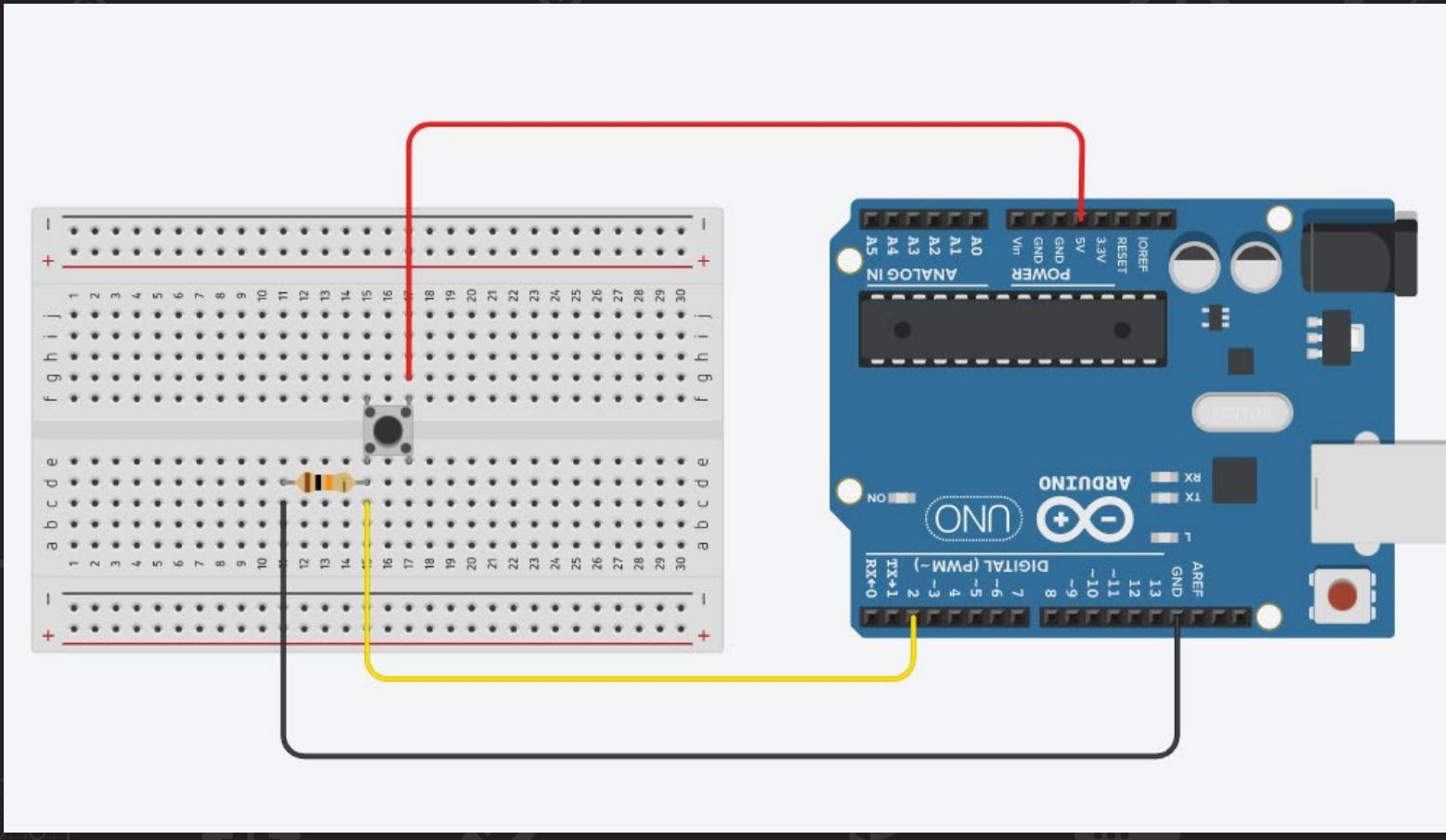
rqt graph

open rqt graph to understand what nodes and topics are there and how they are communicating with each other

Activity 2: Button Publisher

- Create a press button circuit
- Publish a topic of string type called “button_status”
- Topic message should be “pressed” or “not pressed”
- 30 minutes

Activity 2: Button Publisher



Simplified rqt_graph

