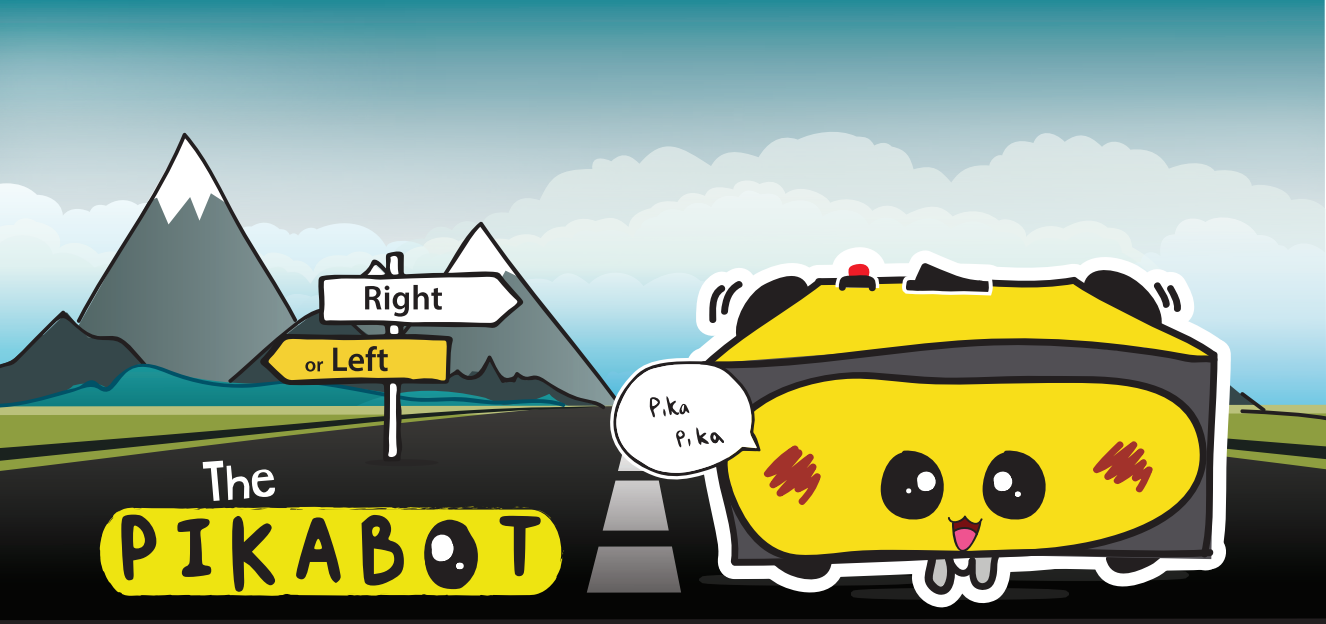
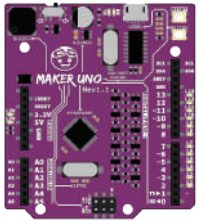


Quick Start Kit

Let's Get Started Building Your Own Smart Car!

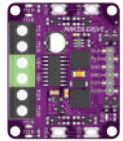
TABLE OF CONTENTS

WHAT'S IN THE BOX?



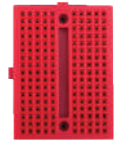
MAKER UNO

x1



MAKER DRIVE

x1



MINI BREADBOARD

x1

TT GEARED MOTOR

x2

RUBBER WHEEL

x2

WHEEL CASTOR

x1

BATTERY HOLDER

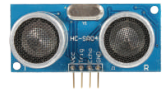
x1

AA BATTERY

x4

IR LINE SENSOR

x2



ULTRASONIC SENSOR

x1

SCREWDRIVER

x1

WHITE RIVET

x2

BLACK RIVET

x10

SCREW

x4

NUT

x4

x1

PUSH BUTTON

x1

ROCKER SWITCH

x1

USB MICRO B CABLE

x1

PRESS TYPE TERMINAL

x40

MALE TO FEMALE JUMPER WIRE

x40

MALE TO MALE JUMPER WIRE

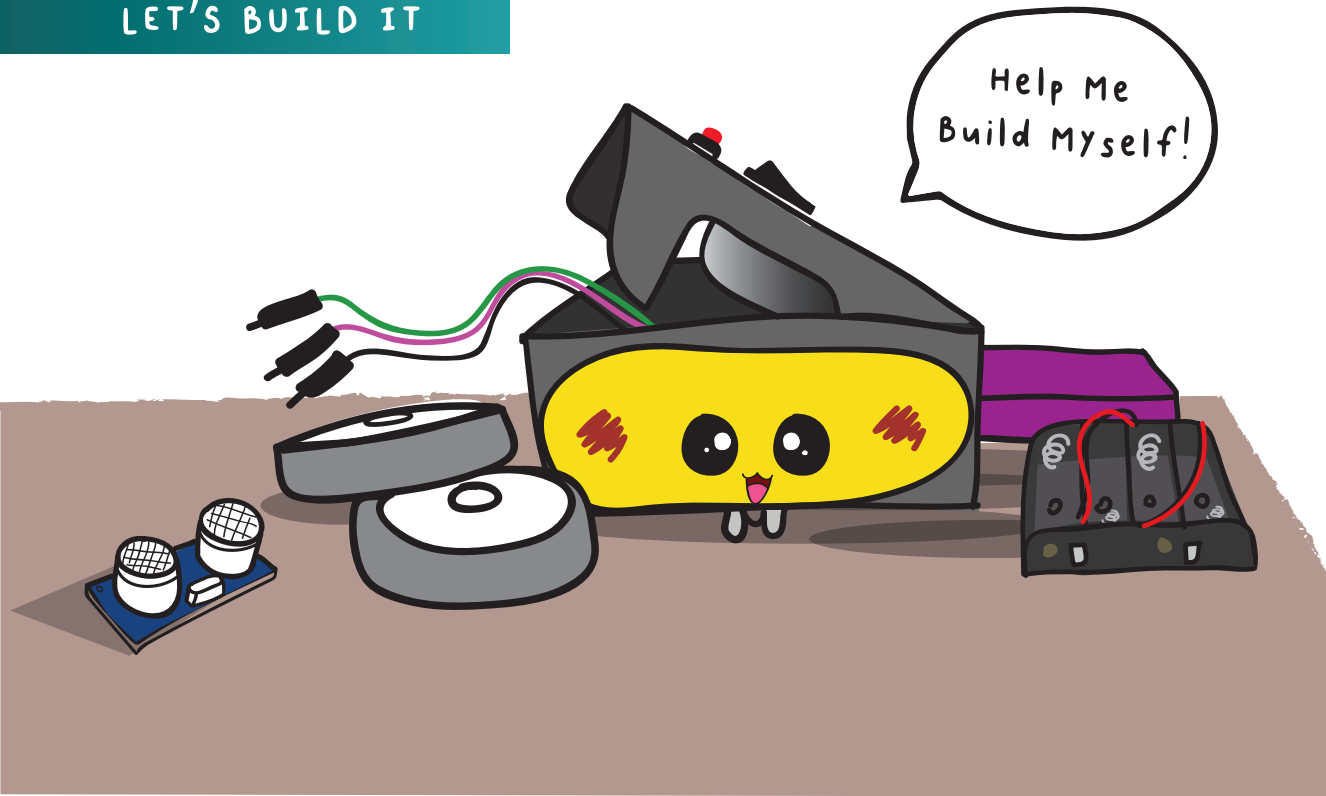
x1

LINE FOLLOWING TRACK

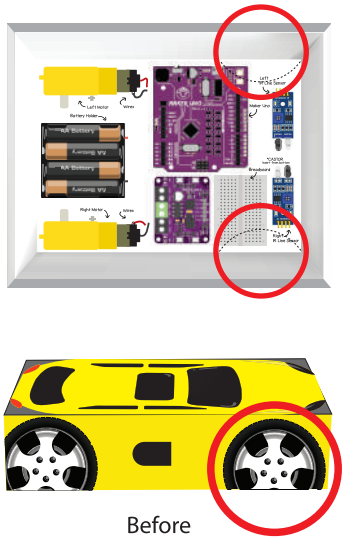
x1

PIKABOT BOOKLET

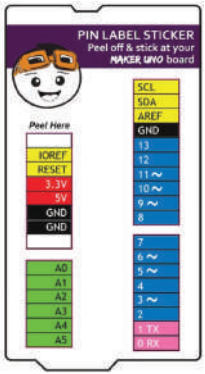
LET'S BUILD IT

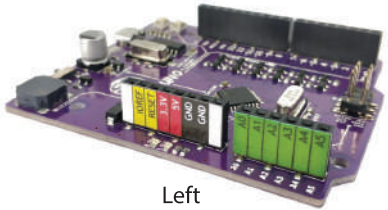


1. Open and empty the box of its contents. Press along the dashed lines to push out the front wheels as shown.

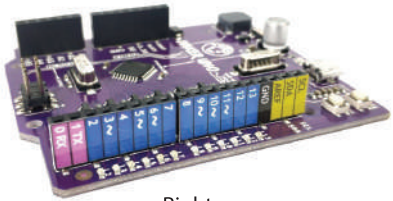


2. Peel off and attach the pin label sticker to the Maker UNO at the indicated position as shown below.



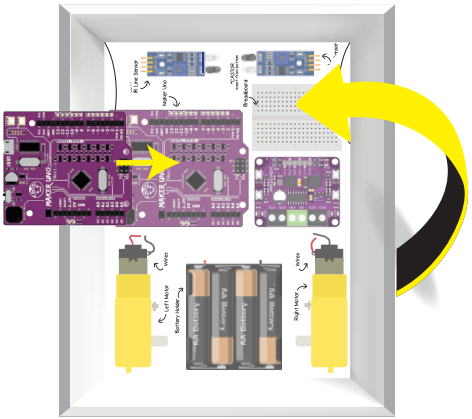


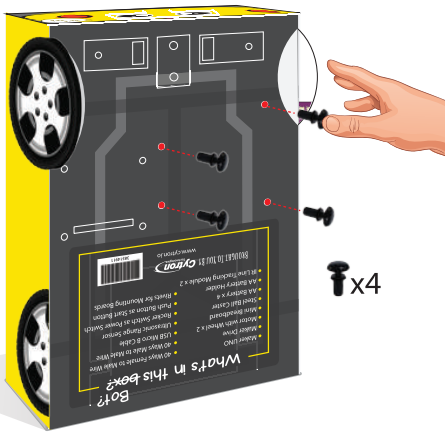
Left



Right

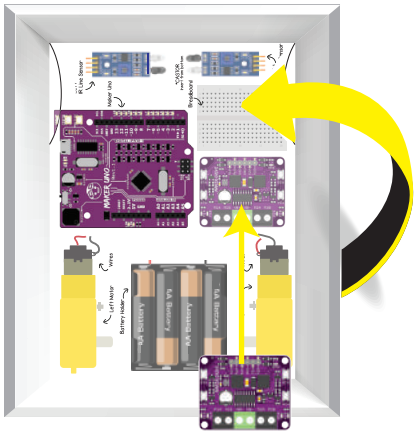
3. Place Maker UNO at its indicated position. Insert **four (4) black rivets** from the bottom of the box to fasten Maker UNO to the box.

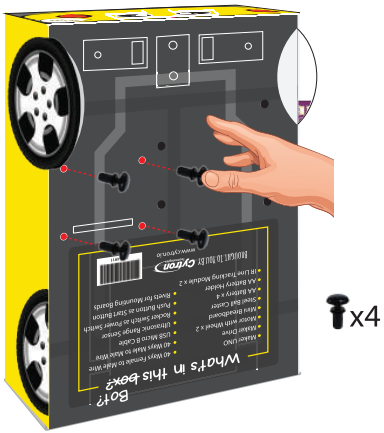




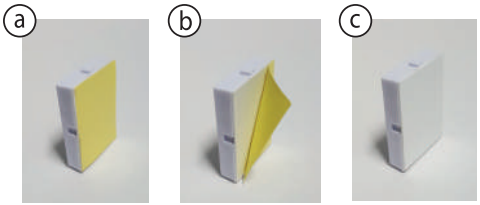
Due to manufacturing constraints, the printed image inside PikaBot box may be misaligned. Please follow the rivet holes for alignment when installing the components.

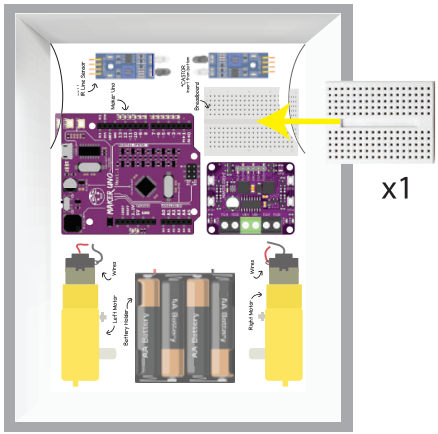
4. Place Maker Drive at its indicated position. Insert **four (4) black rivets** from the bottom of the box to fasten Maker Drive to the box.



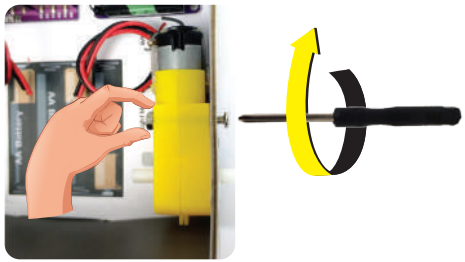


5. Peel off the adhesive backing from the breadboard. Attach the breadboard to the box at the indicated position as shown below.





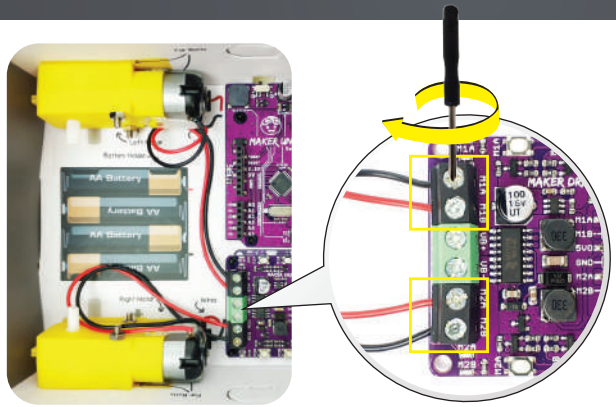
6. Get a TT motor, two screws and two nuts. Use the screwdriver to fasten the motor to the box as shown below. Repeat this step for the other motor.

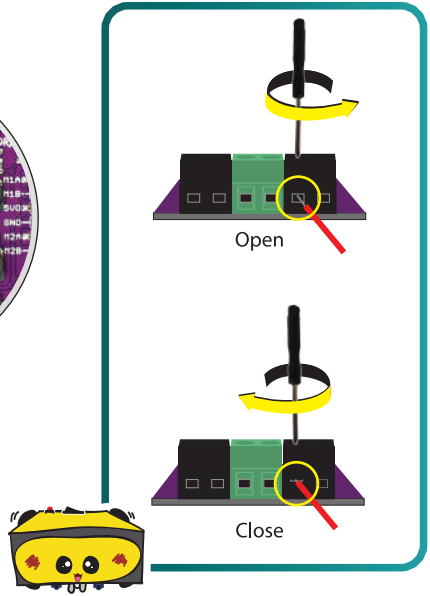


After

The wire should be facing inward and the notch facing outward.

7. Connect both TT motor wires to Maker Drive terminals as shown below.





Motors		Maker Drive
Left Motor	Black (-)	M1A
	Red (+)	M1B
Right Motor	Red (+)	M2A
	Black (-)	M2B

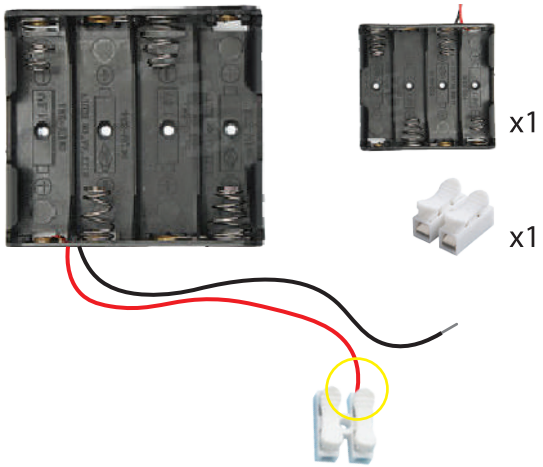
8. Insert the rocker switch through the hole from the top of the box.

x1

Before

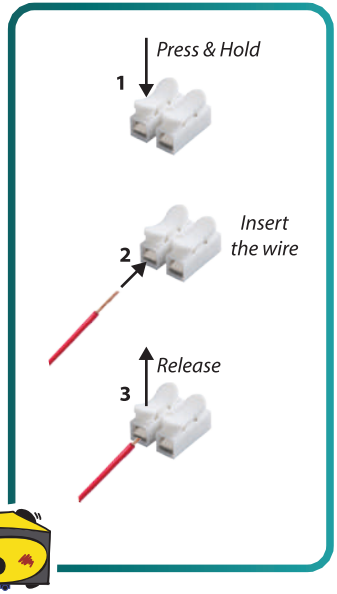
After

9. Connect the **red wire from battery holder** to the press type terminal.

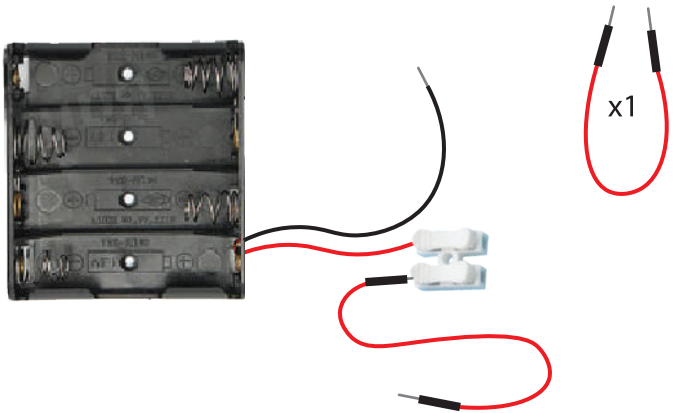


x1

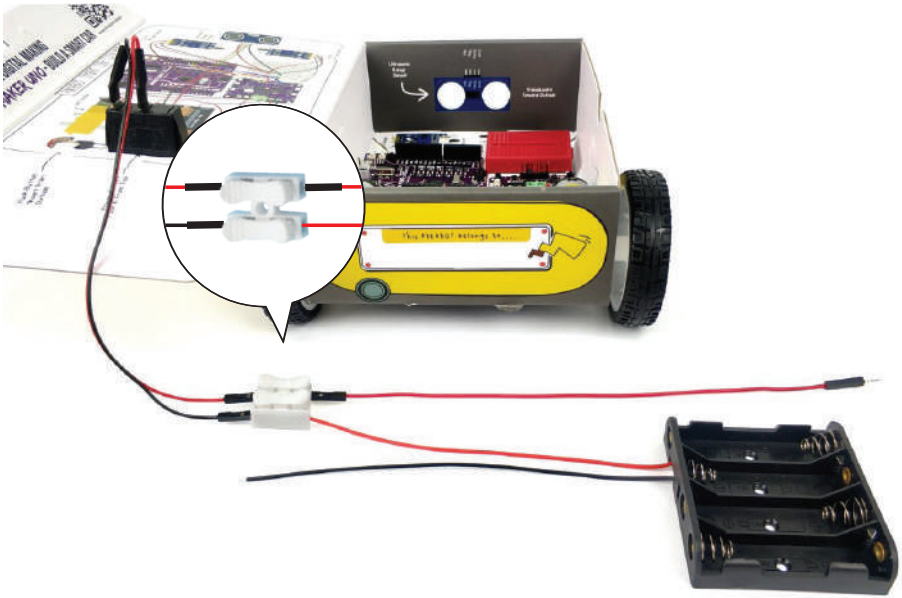
x1



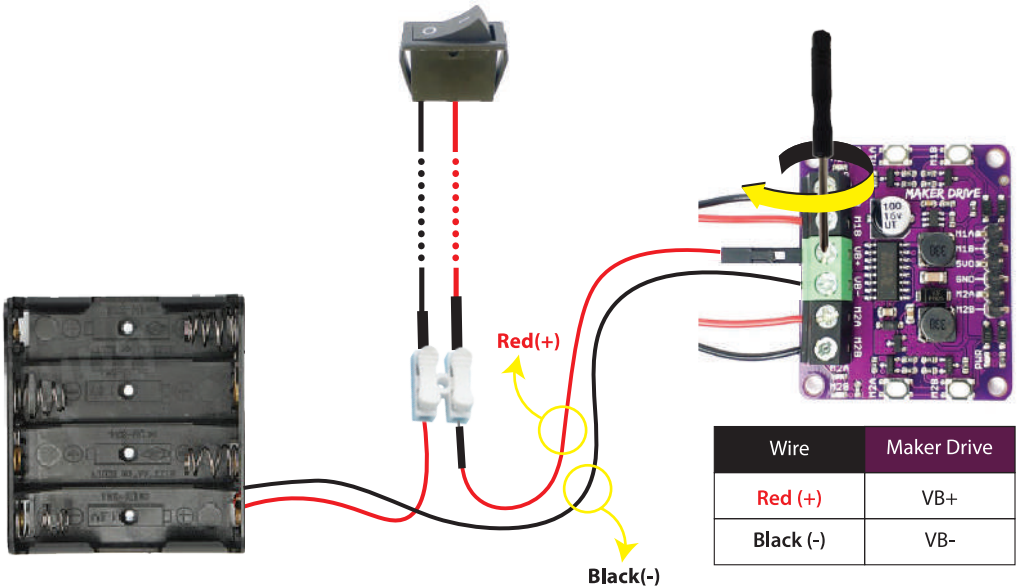
10. Connect **one (1) red male to male jumper wire** to the press type terminal.



11. Connect the rocker switch to the press type terminal as shown below.



12. Connect the wires to Maker Drive - **red wire to VB+ and black wire to VB-.**

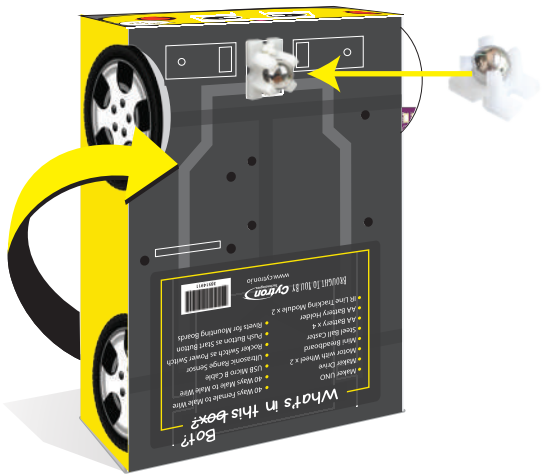


Wire	Maker Drive
Red (+)	VB+
Black (-)	VB-

13. Insert four AA batteries into the battery holder. Then, place the battery holder inside the box at the indicated position.

Please observe the polarity marked in the battery holder to insert the batteries correctly.

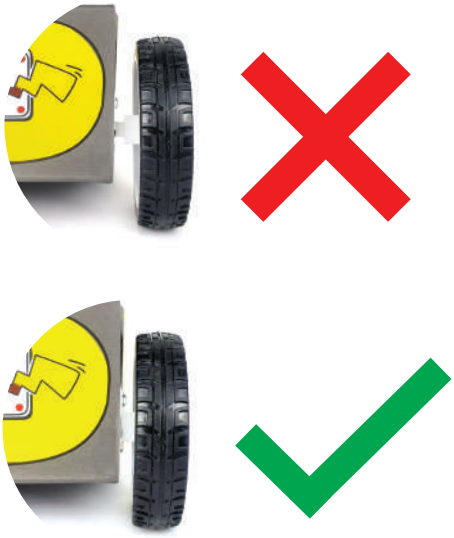
14. Place the wheel castor at the indicated position at the bottom of the box. From inside the box, insert **two (2) white rivets** through the holes and press to fasten the wheel castor to the bottom of the box.



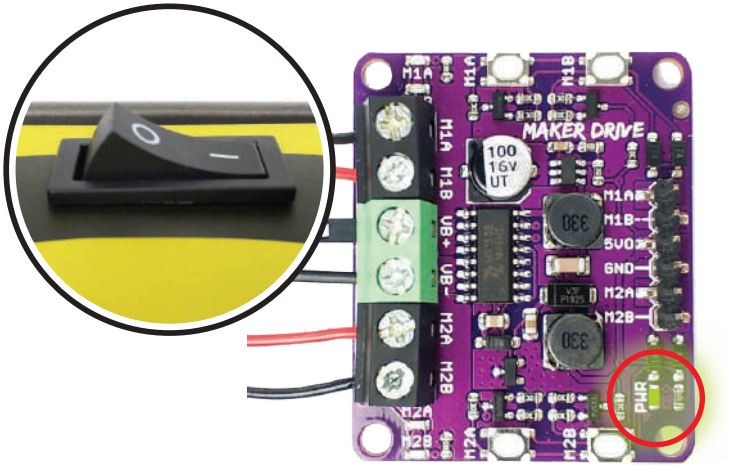
x1

x2

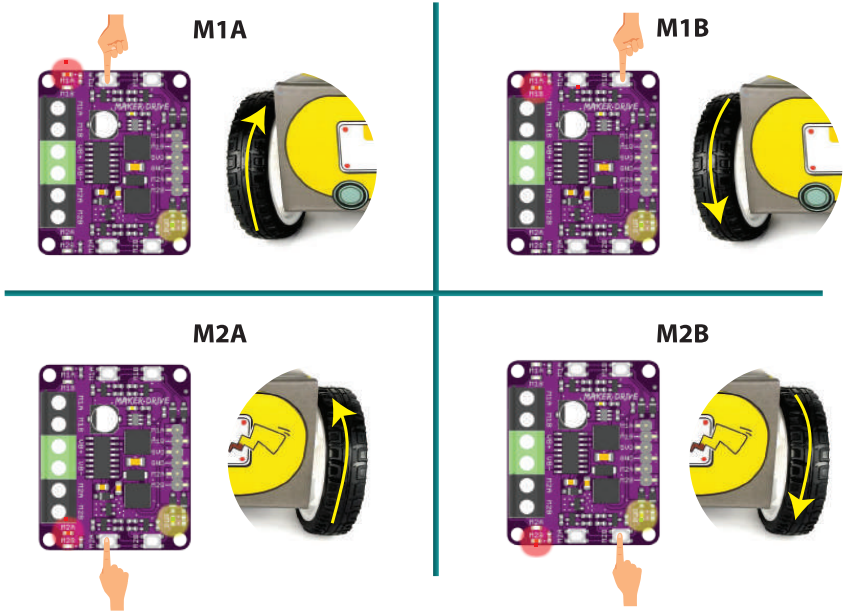
15. Get a wheel and attach it to the TT motor shaft. Repeat for the other side.



16. Turn ON the power by pressing the **rocker switch to I**. You will notice that Maker Drive's power LED will light up.

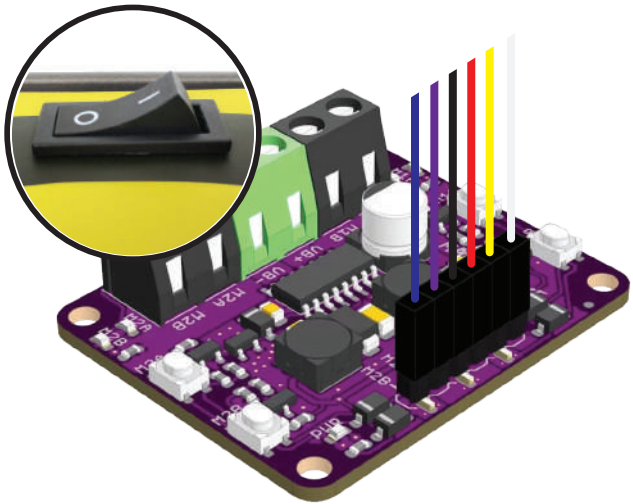


17. Press the M1A, M1B, M2A and M2B buttons on Maker Drive, one by one, to test the spinning direction of the wheels.



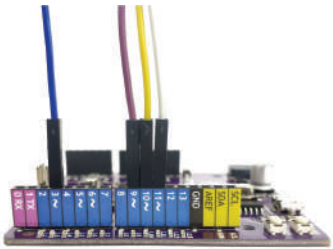
If the wheel is not moving in the same direction as shown above, you need to check the connection of the TT motor to Maker Drive.

18. After testing, turn OFF the power by pressing the **rocker switch to 0**. Get **six (6) male to female jumper wires** and connect them to Maker Drive as shown below.

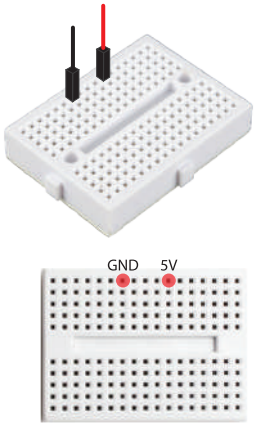


Wire	Maker Drive
White	M1A
Yellow	M1B
Red	5VO
Black	GND
Purple	M2A
Blue	M2B

19. Connect the **six (6) male to female jumper wires** from Maker Drive to Maker UNO and bread-board as shown below.

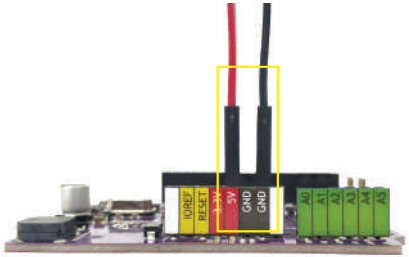


Wire	Maker UNO
White	11
Yellow	10
Purple	9
Blue	3

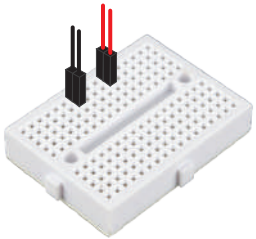


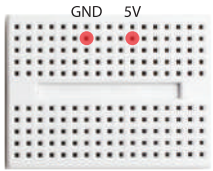
Wire	Breadboard
Red	5V
Black	GND

20. Get **two (2) male to male jumper wires** and connect them to the ground (GND) and 5V pins on Maker UNO. Connect the other ends to the breadboard as shown below.



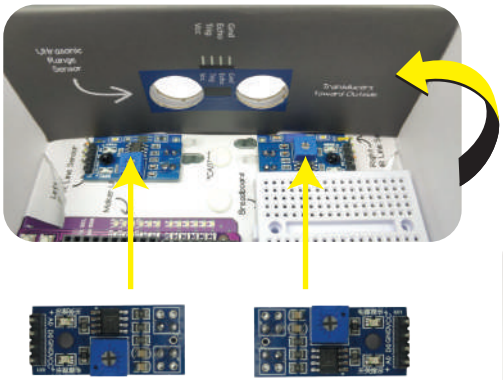
Wire	Maker UNO
Red	5V
Black	GND

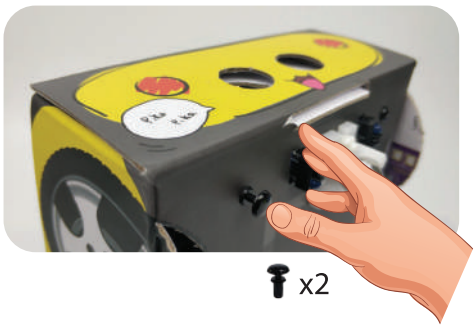




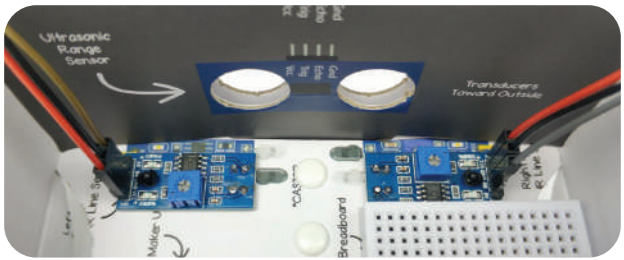
Wire	Breadboard
Red	5V
Black	GND

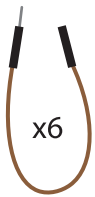
21. Insert both IR sensors through the holes from inside the box at the indicated positions. Insert **two (2) black rivets** from the bottom of the box to fasten the IR sensors in place in the box.



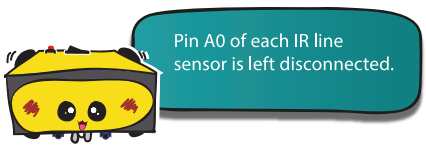


22. Get six **(6) male to female jumper wires** and connect them to the IR sensors as shown below.



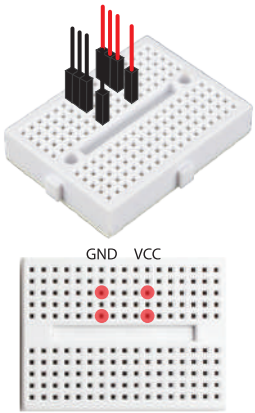


Wire	Left IR Sensor	Right IR Sensor
Red	VCC	VCC
Black	GND	GND
Brown	DO	-
Grey	-	DO



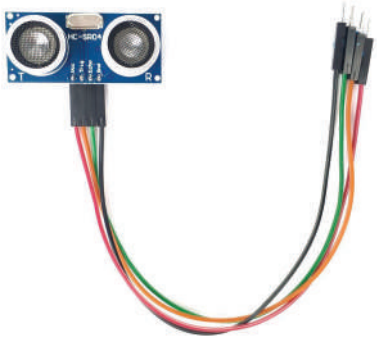
23. Connect the wires to Maker UNO and the breadboard as shown below.

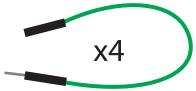
Wire	Maker UNO
Brown	A0
Grey	A1

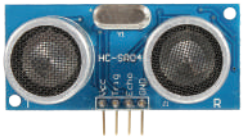


Wire	Breadboard
Red	VCC
Black	GND

24. Get **four (4) male to female jumper wires** and connect them to the ultrasonic sensor.

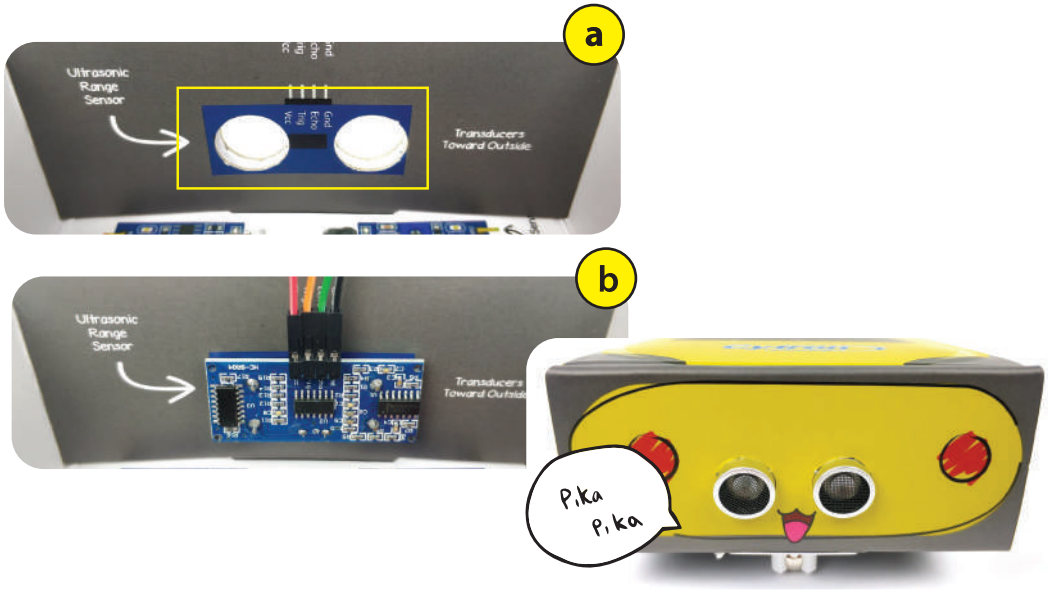




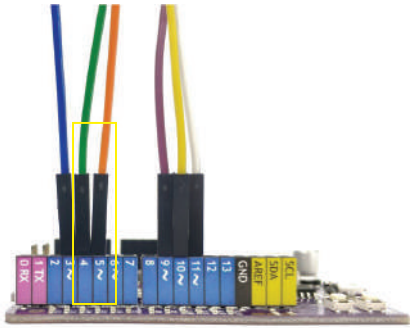


Wire	Ultrasonic Sensor
Red	VCC
Orange	TRIG
Green	ECHO
Black	GND

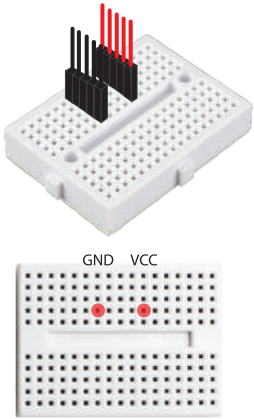
25. Place the ultrasonic sensor at the indicated position and push through the holes from the inside out.



26. Connect the wires from ultrasonic sensor to Maker UNO and the breadboard as shown.



Wire	Maker UNO
Green	4
Orange	5



Wire	Breadboard
Red	VCC
Black	GND

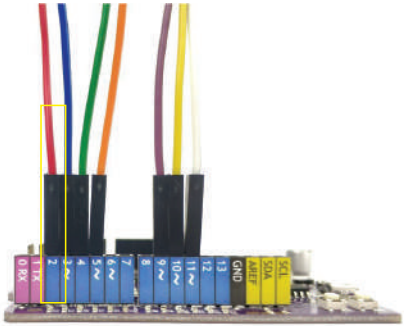
27. Insert the push button through the hole from the top of the box.

x1

Before

After

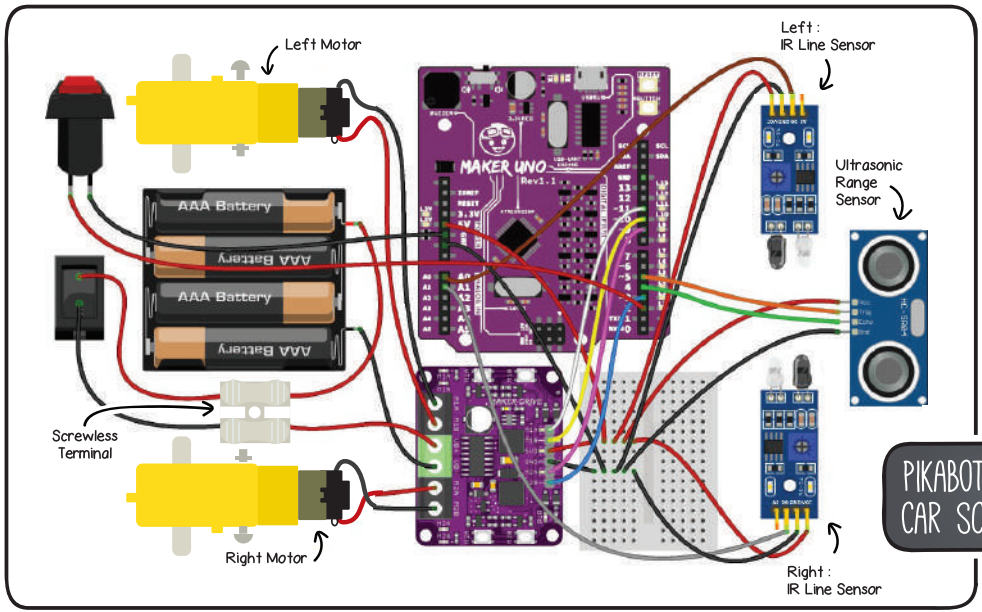
28. Connect the wires to Maker UNO as shown below.



Wire	Maker UNO
Red	2

Wire	Maker UNO
Black	GND

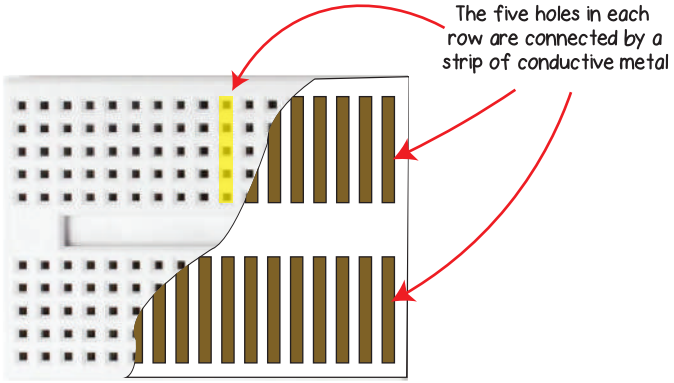
29. Check the wire connections against the schematic diagram. Close the box flap, and you're done!



Fun FACT

A breadboard is commonly used to build prototypes of electronic circuits. It is really convenient to use as you only need to plug your components into the holes on the breadboard to build a circuit - no soldering required!

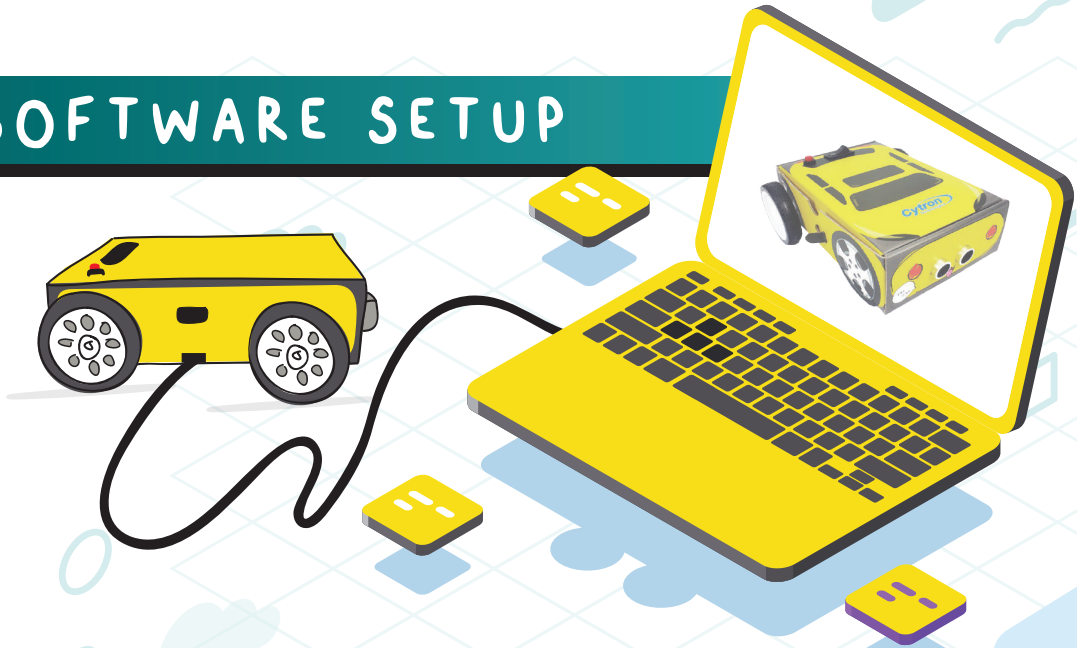
Do take note that not all holes are connected though. Check out the following diagram to determine which holes are connected.



CONGRATULATIONS

You Just Built A PIKABOT!

SOFTWARE SETUP

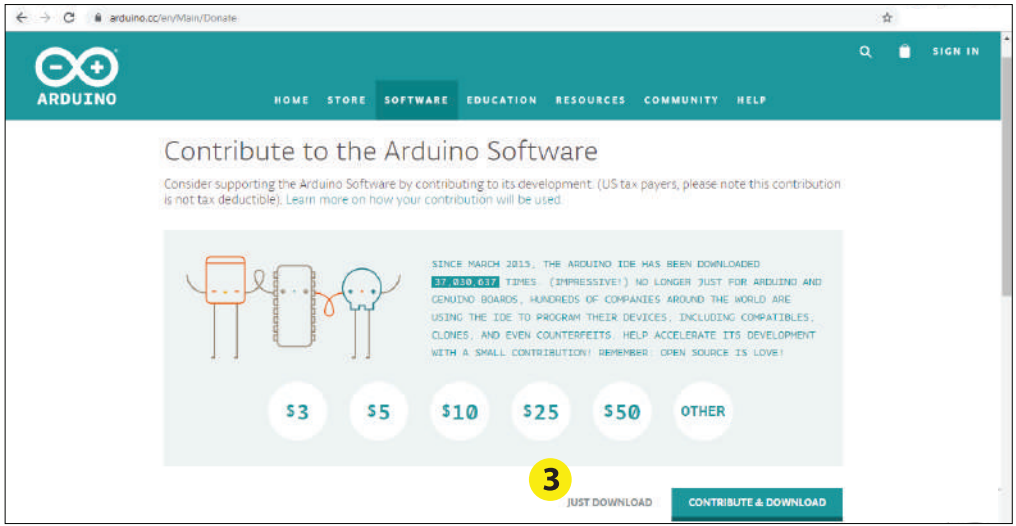


DOWNLOAD ARDUINO IDE

We will program PikaBot using Arduino IDE. Follow this step by step guide to download the Arduino IDE.

1. Go to https://www.arduino.cc/en/Main/Software
2. Choose your Operating System.

3. To download, click on "**JUST DOWNLOAD**". If you want to make a monetary contribution to help fund the development of the Arduino IDE, you can click "**CONTRIBUTE & DOWNLOAD**".



INSTALL ARDUINO IDE

1. Double-click the downloaded installer file and follow onscreen instructions to proceed.

arduino-1.8.10-windows

FOR WINDOWS

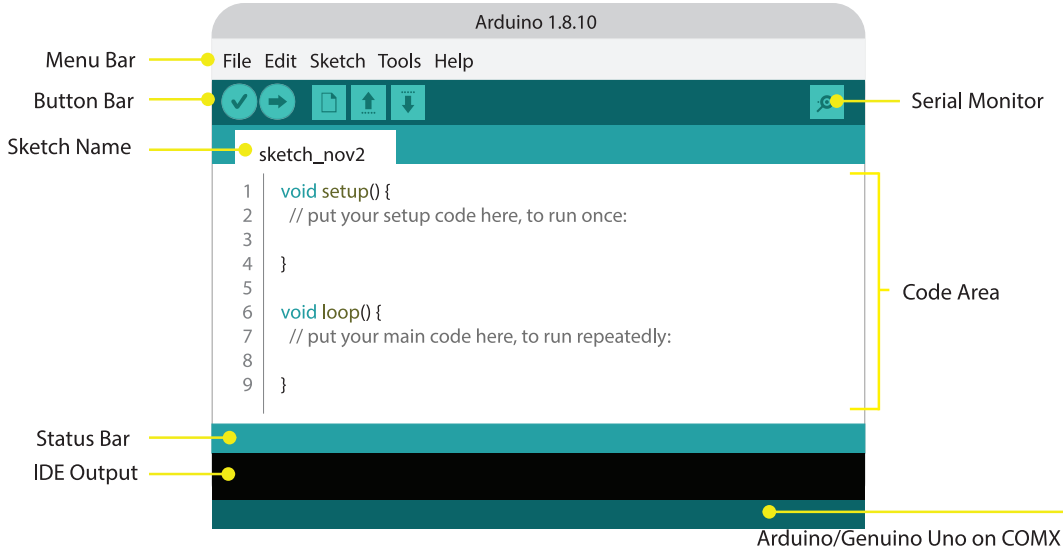
arduino-1.8.10-macosx.zip

FOR MAC

2. Once installation is completed, the Arduino icon will appear on your desktop. Double-click the icon to launch the Arduino IDE.

Arduino

ARDUINO IDE


Arduino 1.8.10
Menu Bar
File Edit Sketch Tools Help
Button Bar
Serial Monitor
Sketch Name
sketch_nov2
1 void setup() {
2 // put your setup code here, to run once:
3
4 }
5
6 void loop() {
7 // put your main code here, to run repeatedly:
8
9 }
Code Area
Status Bar
IDE Output
Arduino/Genuino Uno on COMX

MAKER UNO DRIVER

To use Maker UNO, you need to install CH340 driver. Download the driver from the following link.

Windows - https://cdn.cytron.io/makeruno/CH341SER.EXE
Mac - http://www.wch.cn/download/CH341SER_MAC_ZIP.html
Linux - Normally it is readily installed

1. Connect PikaBot to your computer using the USB micro B cable. Make sure that Maker UNO's power LED is lighted up.

2. Double- click on the downloaded file to proceed with the installation.

FOR WINDOWS

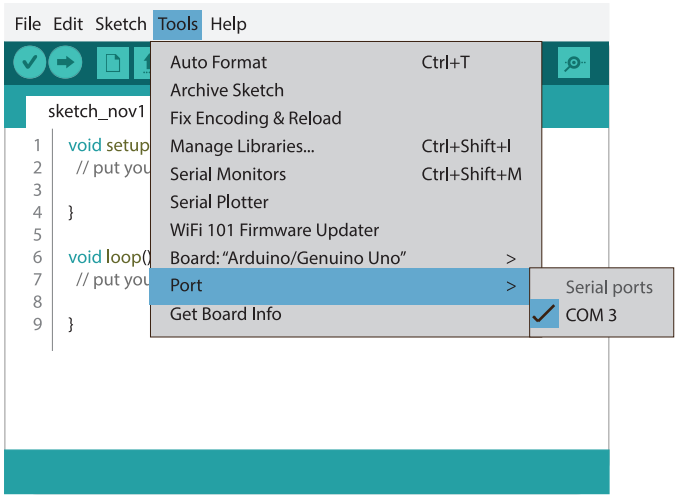
CH341SER

FOR MAC

CH34x_Install_V1.4.pkg

FOR WINDOWS USER

3. Once installation is completed, launch Arduino IDE. Go to **Tools >Ports >COM X**. Select the right COM port.



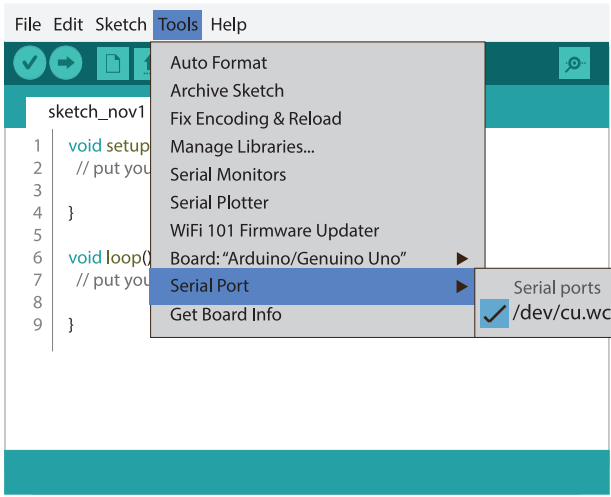
If several COM ports are listed, you can follow the steps below to determine which port your Maker UNO is connected to :

1. **Take a look at the list of available ports.**
2. **Unplug your Maker UNO.**
3. **Check the list again.**

* The entry that "disappeared" is the COM port your Maker UNO was connected to earlier.

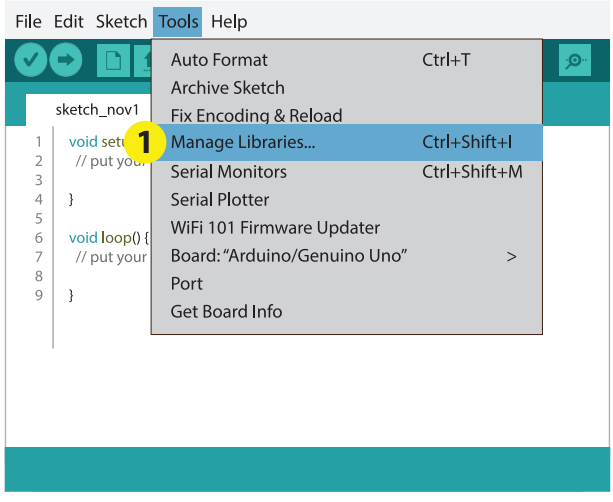
FOR MAC USER

3. Once installation is completed, restart your Mac and launch Arduino IDE. Choose the driver in **Tools > Serial Port > /dev/cu.wchusbserialXXXX**

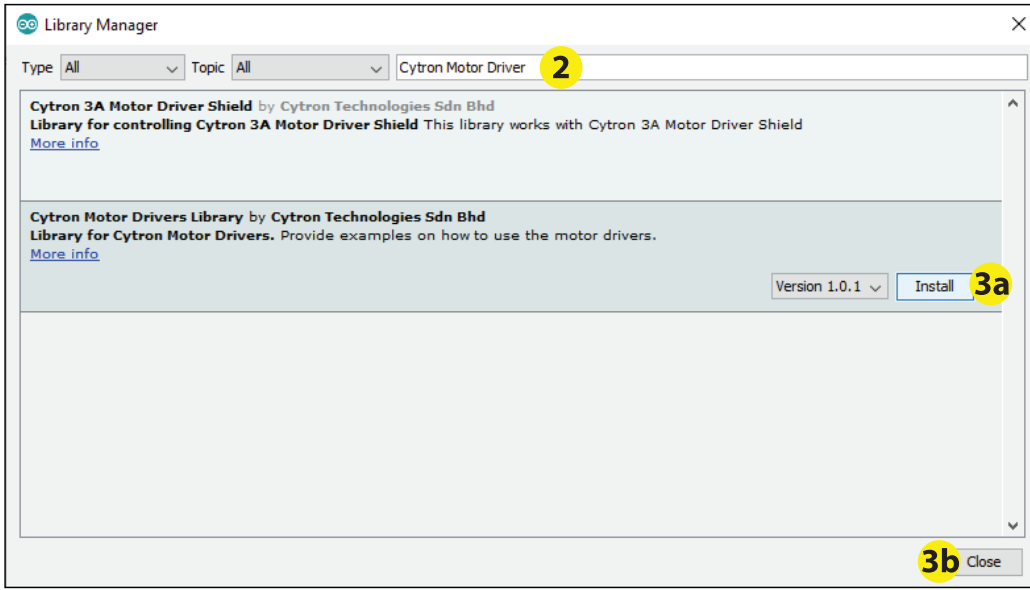


If you're not sure your Maker UNO is connected to which serial port, follow the troubleshooting steps here:
https://i.cytron/CH341-MAC

1. To use Maker Drive, we need to install the Cytron Motor Driver library. Go to **Tools > Manage Libraries..**

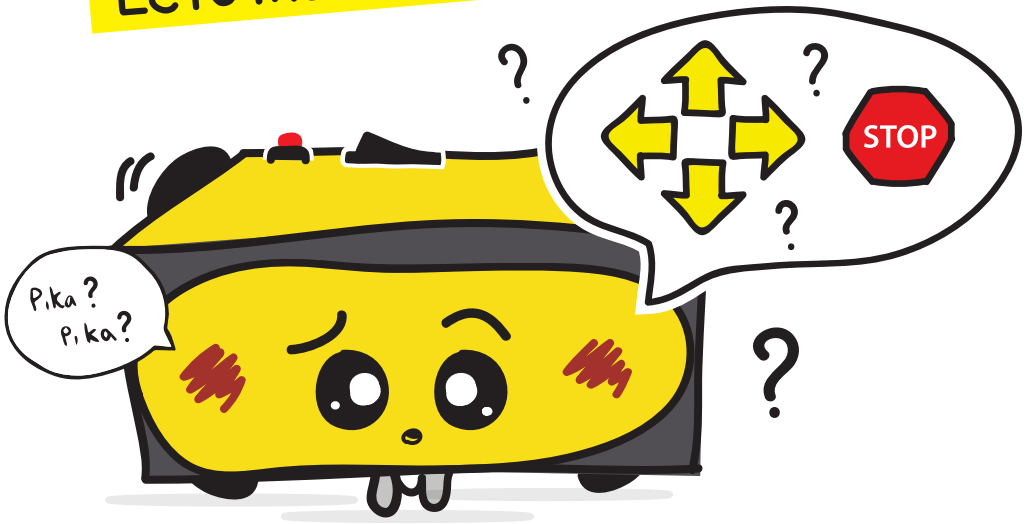


2. Search for "**Cytron Motor Drivers Library**".
3. Click "**Install**" and then "**Close**" after it is done.



PROJECT 1

Let's move it



PROJECT 1-1

Let's Move It!

BASIC NAVIGATION

In this project, we are going to learn the basic movements to control PikaBot :

1. Forward
2. Backward
3. Turn Right
4. Turn Left
5. Stop

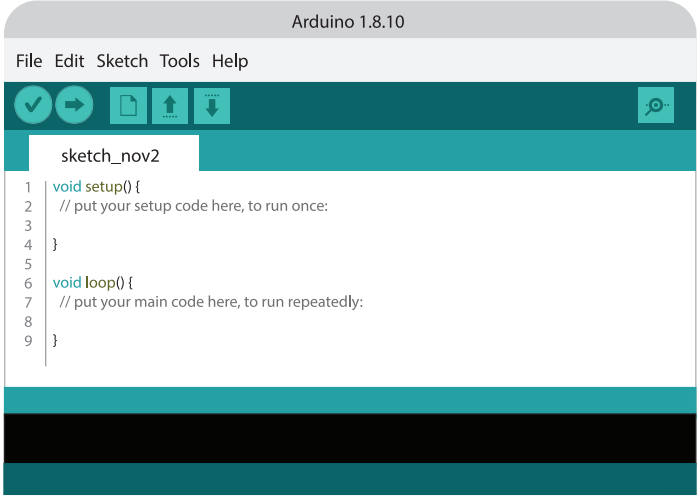
Let's Move It!

1. Connect PikaBot to your computer using the USB micro B cable. Make sure that Maker UNO's power LED is lighted up.

Before you upload the program, make sure the power switch is turned off to protect the USB socket and USB cable from being pulled by the moving robot.

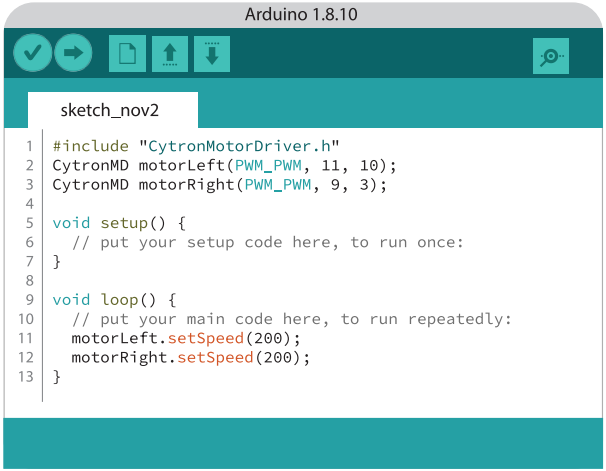
Let's Move It!

2. Launch the Arduino IDE. You will see a new sketch like this.



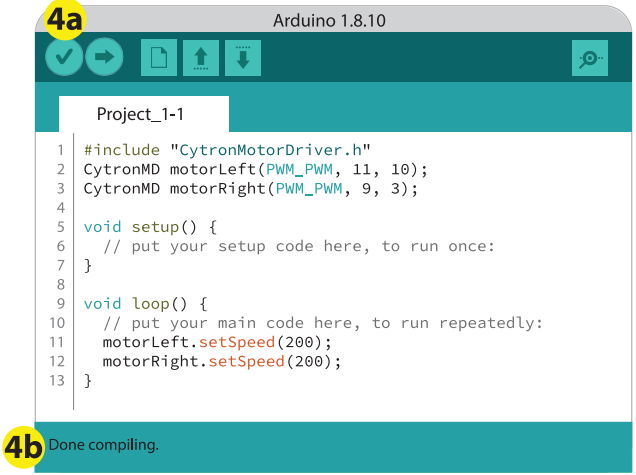
Let's Move It!

3. Write the following code into your sketch.



Let's Move It!

4. Click on the compile button . Wait for a few seconds until you see "**Done Compiling**" appear at the bottom of the sketch.

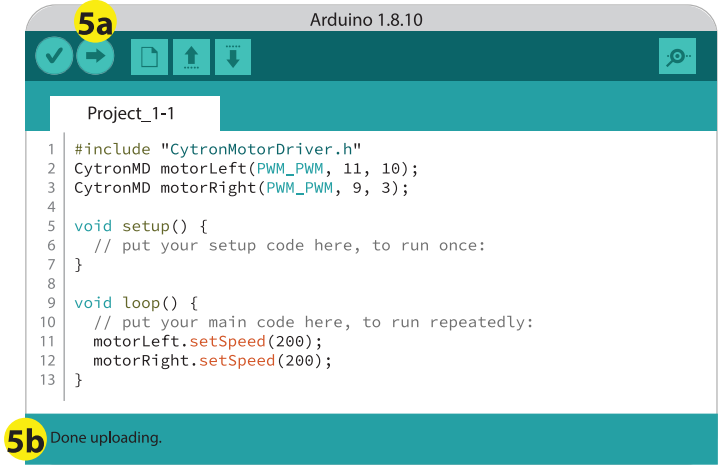


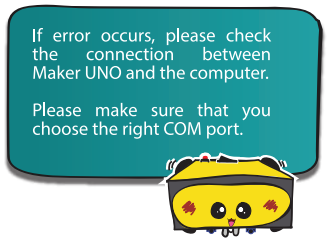
If error occurs, please check the code line by line for missing parentheses () , curly braces {} , or semicolons ; .

Pay attention to the case sensitivity in your code such as ABC or aBc.

Let's Move It!

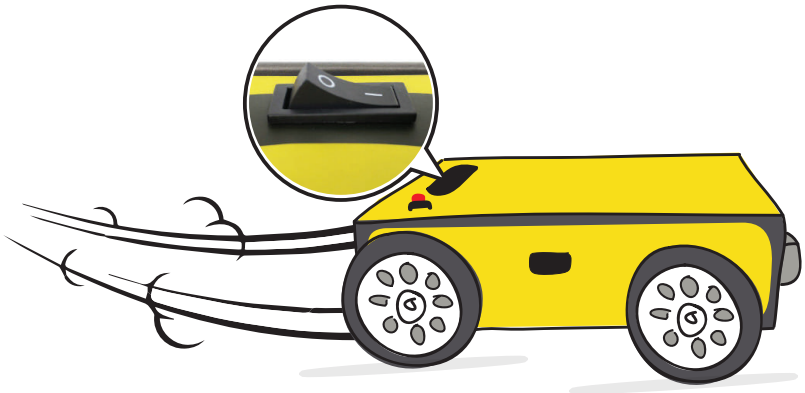
5. Click the upload button . "**Done Uploading**" will appear at the bottom of the sketch after Arduino IDE successfully uploaded the code to the Maker UNO.





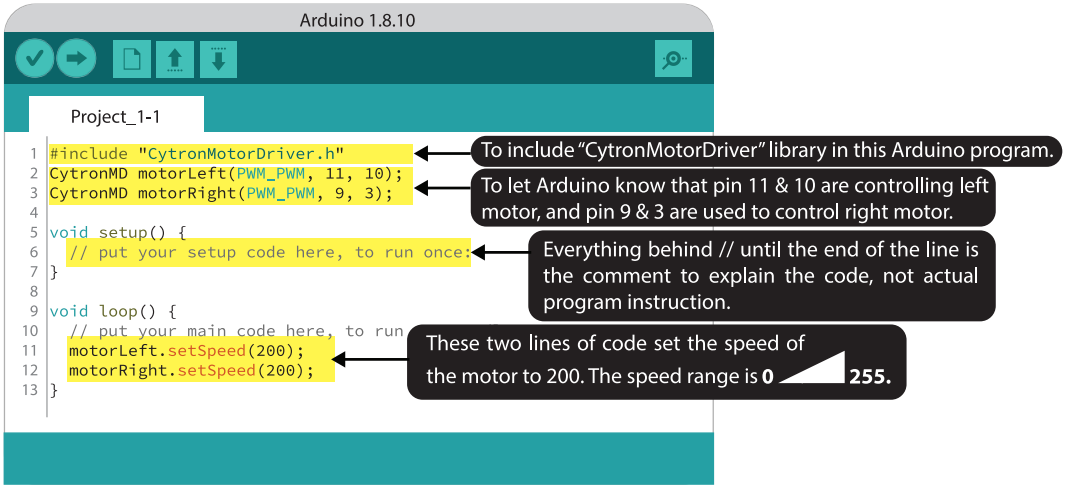
Let's Move It!

6. Check the result.
   PikaBot will move forward continuously when the power is turned on.



Disconnect the USB cable before you turn on the power on PikaBot.

HOW it works!


Arduino 1.8.10
Project_1-1
1 #include "CytronMotorDriver.h"
2 CytronMD motorLeft(PWM_PWM, 11, 10);
3 CytronMD motorRight(PWM_PWM, 9, 3);
4
5 void setup() {
6 // put your setup code here, to run once:
7 }
8
9 void loop() {
10 // put your main code here, to run
11 motorLeft.setSpeed(200);
12 motorRight.setSpeed(200);
13 }
To include "CytronMotorDriver" library in this Arduino program.
To let Arduino know that pin 11 & 10 are controlling left motor, and pin 9 & 3 are used to control right motor.
Everything behind // until the end of the line is the comment to explain the code, not actual program instruction.
These two lines of code set the speed of the motor to 200. The speed range is 0 255.

PROJECT 1-2

Let's Move It!

1. Modify the previous code into this. Then, compile  and upload  the code.

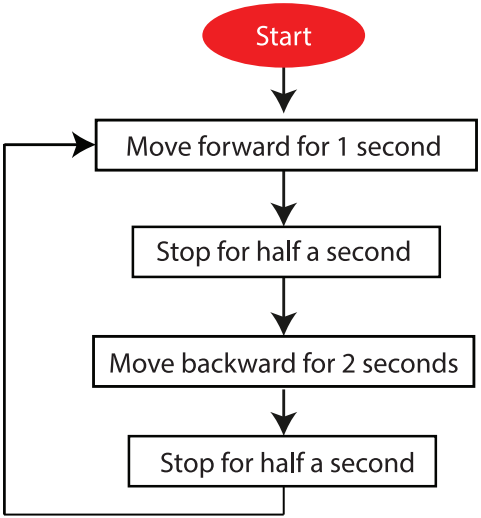
Project_1-2

```
#include "CytronMotorDriver.h"
CytronMD motorLeft(PWM_PWM, 11, 10);
CytronMD motorRight(PWM_PWM, 9, 3);

void setup() {
}

void loop() {
  motorLeft.setSpeed(200);   //moves forward
  motorRight.setSpeed(200);
  delay(1000);

  motorLeft.setSpeed(0);
  motorRight.setSpeed(0);
  delay(500);

  motorLeft.setSpeed(-200);  //moves backward
  motorRight.setSpeed(-200);
  delay(2000);

  motorLeft.setSpeed(0);
  motorRight.setSpeed(0);
  delay(500);
 }
```

HOW it works!

When powered on, PikaBot will move forward for a second and stop for half a second, then move backward for two seconds and stop for half a second.

PikaBot will repeat these movements endlessly until you switch it off.



HOW it works!

Project_1-2

```
1  #include "CytronMotorDriver.h"
2  CytronMD motorLeft(PWM_PWM, 11, 10);
3  CytronMD motorRight(PWM_PWM, 9, 3);
4
5  void setup() {
6  }
7
8  void loop() {
9    motorLeft.setSpeed(200);
10   motorRight.setSpeed(200);
11   delay(1000);
12
13   motorLeft.setSpeed(0);
14   motorRight.setSpeed(0);
15   delay(500);
16
17   motorLeft.setSpeed(-200);
18   motorRight.setSpeed(-200);
19   delay(2000);
20
21   motorLeft.setSpeed(0);
22   motorRight.setSpeed(0);
23   delay(500);
24 }
```

The program inside loop() will continue to run forever

Positive value for the speed means PikaBot is moving forward.

Negative value for the speed means PikaBot is moving backward.

The delay holds/pauses the program in milliseconds(ms). 500ms = 0.5s .

PROJECT 1-3

Let's Move It!

Now we will learn to program PikaBot to turn left and right, using 'functions'.

1. Modify the previous code into this. Then, compile and upload the code.

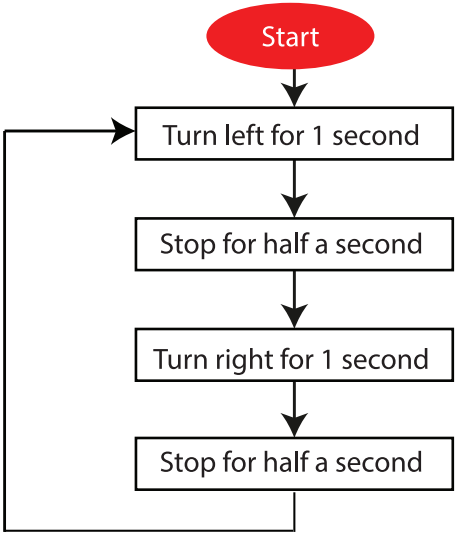
Project_1-3

```
#include "CytronMotorDriver.h"
CytronMD motorLeft(PWM_PWM, 11, 10);
CytronMD motorRight(PWM_PWM, 9, 3);

void robotStop() {
  motorLeft.setSpeed(0);
  motorRight.setSpeed(0);
}
void robotMove(int speedLeft, int speedRight) {
  motorLeft.setSpeed(speedLeft);
  motorRight.setSpeed(speedRight);
}
void setup() {
}
void loop() {
  robotMove(-200, 200); //turn left
  delay(1000);
  robotStop();
  delay(500);
  robotMove(200, -200); //turn right
  delay(1000);
  robotStop();
  delay(500);
}
```

HOW it works!

When powered on, PikaBot will turn left for one second, stop for half a second, turn right for one second, then stop for half a second.

PikaBot will keep turning to the left and then to the right endlessly.



HOW it works!

Project_1-3

```
1  #include "CytronMotorDriver.h"
2  CytronMD motorLeft(PWM_PWM, 11, 10);
3  CytronMD motorRight(PWM_PWM, 9, 3);
4
5  void robotStop() {
6    motorLeft.setSpeed(0);
7    motorRight.setSpeed(0);
8  }
9  void robotMove(int speedLeft, int speedRight) {
10   motorLeft.setSpeed(speedLeft);
11   motorRight.setSpeed(speedRight);
12 }
13 void setup() {
14 }
15 void loop() {
16   robotMove(-200, 200); //turn left
17   delay(1000);
18   robotStop();
19   delay(500);
20   robotMove(200, -200); //turn right
21   delay(1000);
22   robotStop();
23   delay(500);
24 }
```

This is a function. A function is a block of code which is not executed immediately. They are "saved for later use", and will be executed when they are called.

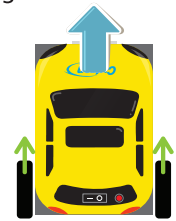
This is also a function and it has two arguments - speedLeft and speedRight. We need to pass two values to it in the bracket when calling this function.
eg: robotMove(185, 220);

This calls the robotStop() function which doesn't accept any values.

This code calls the robotMove() function and passes the speed values for left and right motors.

Fun FACT

If both wheels spin in the same direction at the same speed, the robot will move forward or backward. If the wheels spin in opposite directions at the same speed, the robot will rotate left or right.



FORWARD

Both wheels move forward.

REVERSE

Both wheels move backward.

TURN RIGHT

Left wheel moves forward.
Right wheel moves backward.

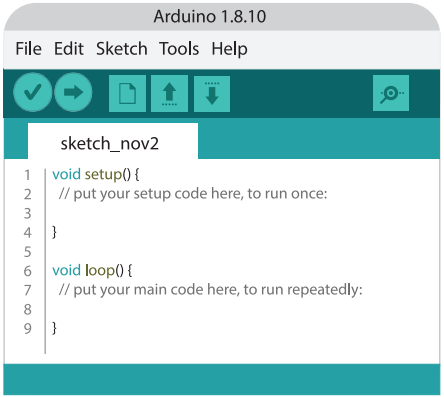
TURN LEFT

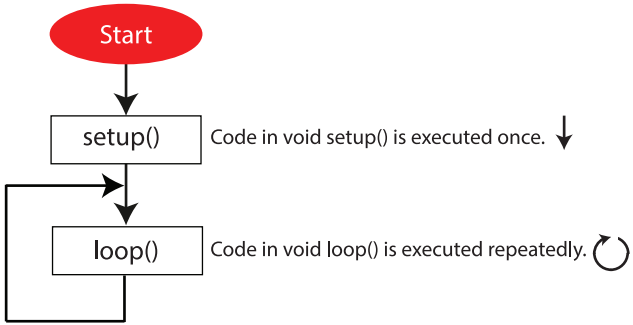
Left wheel moves backward.
Right wheel moves forward.

What about when both wheels spin in the same direction at different speeds? For example left motor at speed 140 and right motor at speed 200.

Fun FACT

This is the basic flow chart for an Arduino sketch.

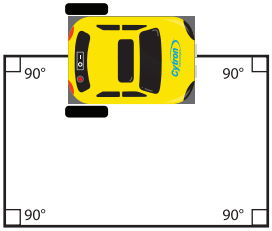




CHALLENGE

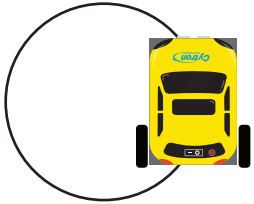
Program your PikaBot to move in the following patterns :

1.



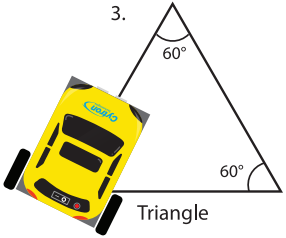
Rectangle

2.



Circle

3.



Triangle

SCAN HERE
FOR EXTRA RESOURCES

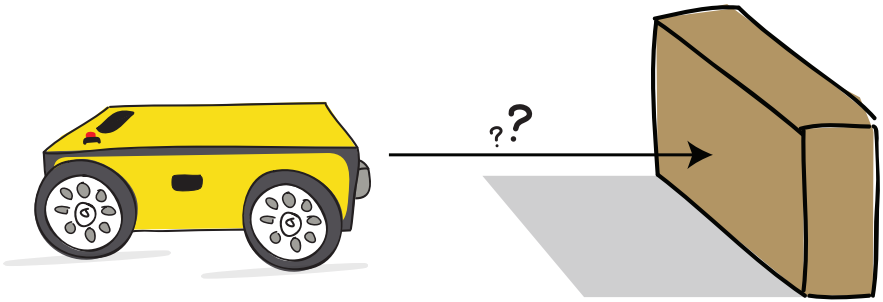
PROJECT 2

Oh No! Stop!

PROJECT 2-1

Oh No! Stop!

Now let's try to measure the distance of an obstacle placed in front of PikaBot using the ultrasonic sensor.



PROJECT 2-1

Oh No! Stop!

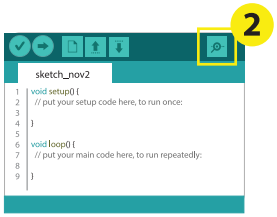
1. Open a new sketch and write the following code. Then, compile and upload the code.

Project_2-1

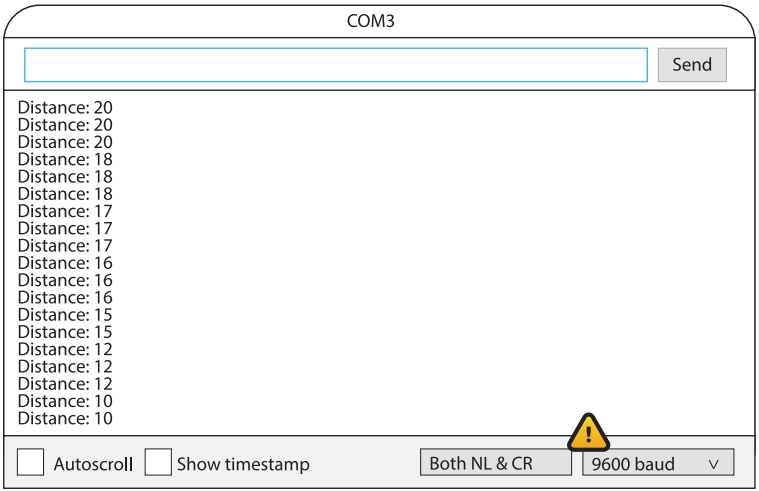
```
long duration;
long distance;

#define TRIGPIN 5
#define ECHOPIN 4

void setup() {
  pinMode(TRIGPIN, OUTPUT);
  pinMode(ECHOPIN, INPUT);
  Serial.begin(9600);
}

void loop() {
  digitalWrite(TRIGPIN, LOW);   // Clears the trigPin
  delayMicroseconds(2);
  digitalWrite(TRIGPIN, HIGH);  // Sets the trigPin on HIGH state for 10 microseconds
  delayMicroseconds(10);
  digitalWrite(TRIGPIN, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(ECHOPIN, HIGH);
  distance = duration * 0.017;
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println("cm");
}
```

Oh No! Stop!

2. Click  to open the serial monitor. The serial monitor window shows the distance measured by the ultrasonic sensor in cm. Place an object in front of the sensor at various distances and observe the reading.



Please make sure that the Serial Monitor's baud rate is 9600 baud.



HOW it works!

Project_2-1

```
1  long duration;
2  long distance;
3
4  #define TRIGPIN 5
5  #define ECHOPIN 4
6
7  void setup() {
8    pinMode(TRIGPIN, OUTPUT);
9    pinMode(ECHOPIN, INPUT);
10   Serial.begin(9600);
11 }
12
13 void loop() {
14   digitalWrite(TRIGPIN, LOW);
15   delayMicroseconds(2);
16   digitalWrite(TRIGPIN, HIGH);
17   delayMicroseconds(10);
18   digitalWrite(TRIGPIN, LOW);
19
20   duration = pulseIn(ECHOPIN, HIGH);
21   distance = duration * 0.017;
22   Serial.print("Distance: ");
23   Serial.print(distance);
24   Serial.println("cm");
25 }
```

To declare two variables which will be used to store long (32-bit) values in our program.

To give a name to a constant value. We can use these reference name in code, instead of memorizing all the constant values.

To configure the specified pin as input or output pin. Here we set pin 5 as output and pin 4 as input pin.

To start a serial communication between Maker UNO and the computer so that we can display data from PikaBot on our computer.

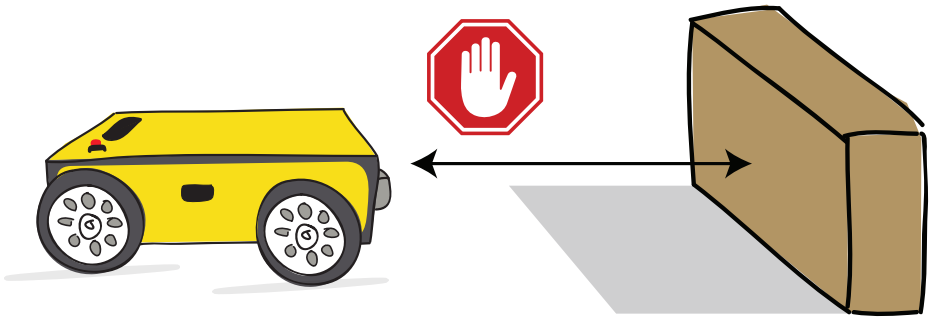
This block of code activates the ultrasonic sensor. 'LOW' = OFF or 0V and HIGH = ON or 5V.

Serial.print() prints the data on the Serial Monitor. While Serial.println() prints the data and moves the cursor to a new line.

PROJECT 2-2

Oh No! Stop!

Since we already know how to measure the distance using ultrasonic sensor, now let's try to stop PikaBot before it bumps into an obstacle.



PROJECT 2-2

Oh No! Stop!

1. Modify the previous code into this. Then, compile and upload the code.

Project_2-2

```
#include "CytronMotorDriver.h"
CytronMD motorLeft(PWM_PWM, 11, 10);
CytronMD motorRight(PWM_PWM, 9, 3);

//Declare variable
long duration;
long distance;
//Define constant
#define BUTTON  2
#define TRIGPIN 5
#define ECHOPIN 4

void robotStop() {
  motorLeft.setSpeed(0);
  motorRight.setSpeed(0);
}
void robotMove(int speedLeft, int speedRight) {
  motorLeft.setSpeed(speedLeft);
  motorRight.setSpeed(speedRight);
}

```

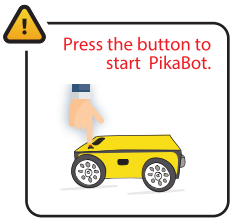
Note : The code continues on the next page

PROJECT 2-2

Oh No! Stop!

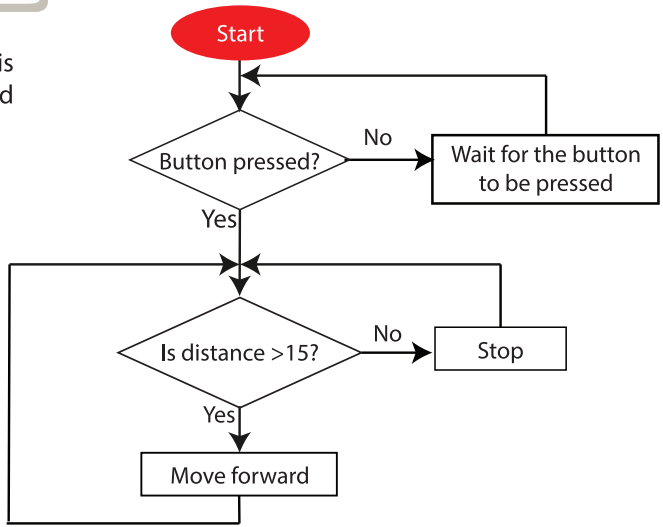
Project_2-2

```
22 void setup() {
23   //Configure input and output pins
24   pinMode(TRIGPIN, OUTPUT);
25   pinMode(ECHOPIN, INPUT);
26   pinMode(BUTTON, INPUT_PULLUP);
27 }
28
29 void loop() {
30   if(digitalRead(BUTTON) == LOW){  // Button is pressed
31
32     while (true){   //Execute the code in this loop forever
33       digitalWrite(TRIGPIN, LOW);
34       delayMicroseconds(2);
35       digitalWrite(TRIGPIN, HIGH);
36       delayMicroseconds(10);
37       digitalWrite(TRIGPIN, LOW);
38
39       duration = pulseIn(ECHOPIN,HIGH);
40       distance = duration*0.017;
41
42       if (distance > 15) {
43         robotMove(200,200);   //Robot move forward at speed 200
44       }
45       else {
46         robotStop();  //Robot stop
47       }
48     }
49   }
50 }
```



HOW it works!

PikaBot waits until the button is pressed. Then it will move forward when there is no obstacle detected.



HOW it works!

Project_2-2

```
22 void setup() {
23   //Configure input and output pins
24   pinMode(TRIGPIN, OUTPUT);
25   pinMode(ECHOPIN, INPUT);
26   pinMode(BUTTON, INPUT_PULLUP);
27 }
28
29 void loop() {
30   if(digitalRead(BUTTON) == LOW){  // Button is pressed
31
32     while (true){   //Execute the code in this loop forever
33       digitalWrite(TRIGPIN, LOW);
34       delayMicroseconds(2);
35       digitalWrite(TRIGPIN, HIGH);
36       delayMicroseconds(10);
37       digitalWrite(TRIGPIN, LOW);
38
39       duration = pulseIn(ECHOPIN,HIGH);
40       distance = duration*0.017;
41
42       if (distance > 15) {
43         robotMove(200,200);    //Robot move forward at speed 200
44       }
45       else {
46         robotStop();  //Robot stop
47       }
48     }
49   }
50 }
```

Configure BUTTON as INPUT_PULLUP - which means the default value for the input is HIGH, unless it is pulled LOW by pressing the push button.

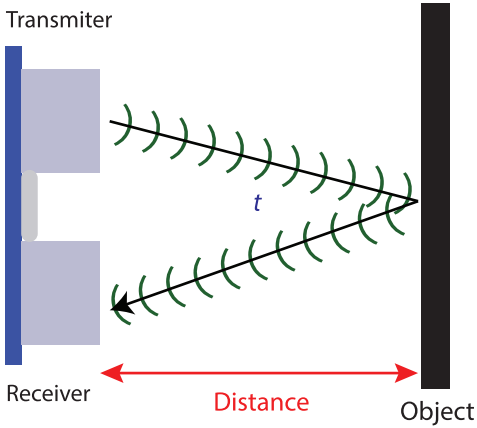
This **if statement** is to check whether the button is pressed.

This if statement is to check if the distance is greater than 15cm.

This else will run if the condition is false.

Fun FACT

The basic idea of how an ultrasonic sensor works is simple. The sensor transmits ultrasound wave which will be bounced back if an object or obstacle is blocking its way. By measuring the duration needed for the reflected ultrasound wave to be detected by the receiver, we are able to estimate the distance of the object from the sensor.



t = Time travel of the sound wave in µs

Speed of Sound = 343 m/s = 0.034cm/µs

$$\text{Distance} = \frac{t}{2} \times \text{Speed of Sound}$$

$$= \frac{t}{2} \times 0.034 \text{ cm}$$

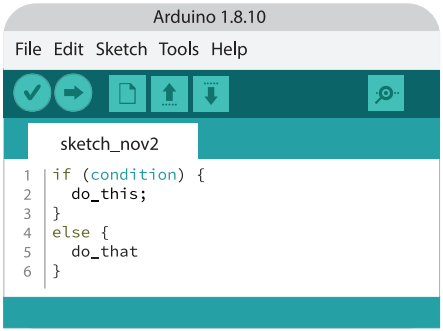
$$= t \times 0.017 \text{ cm}$$

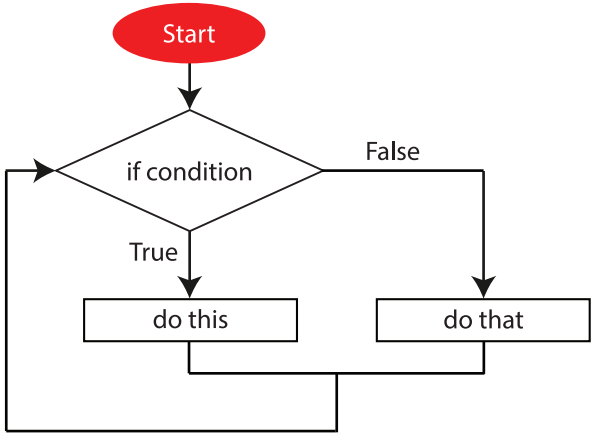
This is the formula that we used in our code :

Distance = t x 0.017cm

Fun FACT

This is the basic flow chart for if-else programming. The if() statement in programming compares two things and determines whether the comparison is true or false. Then it executes the corresponding action you program it to do.

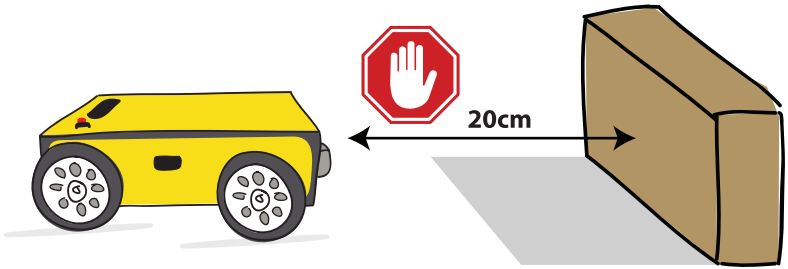




CHALLENGE

Program PikaBot to do the following :

1. Move forward when powered on.
2. Turn to the left more than 90 degrees (90°) when it detects an obstacle within 20cm range.
3. Then continue to move forward.



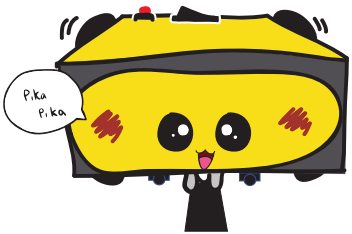
SCAN HERE
FOR EXTRA RESOURCES

PROJECT 3

Black Or White Line?

Black or White Line?

Let's teach PikaBot how to **differentiate between black and white**.



Directly above the black line

Left IR detects black line

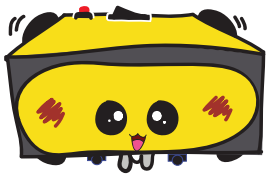
Right IR detects black line

PROJECT 3-1

Black or White Line?

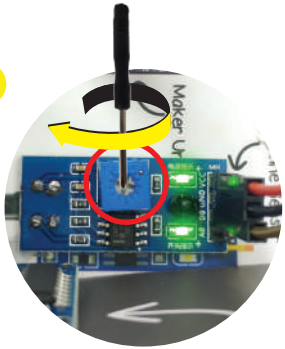
Calibrate IR Line Sensors on the PikaBot track.

1



Open the PikaBot track. Then, place PikaBot on the white surface of the track.

2



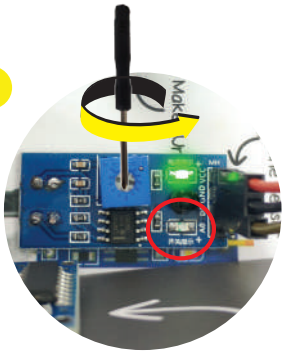
Turn the trimmer clockwise (CW) until it reaches the limit. Both LEDs sholuld light up.

3

Adjust PikaBot so that the left IR line sensor is directly above the black line.

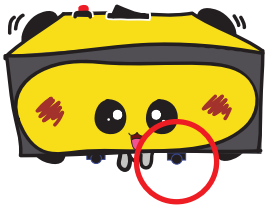
Black or White Line?

4



Make sure that both LEDs are lighted up. Then use the screwdriver to slowly turn the trimmer counter clockwise (CCW). As you do so, observe the LED next to the 'A0' and stop turning the trimmer right after this LED has turned off.

5



Move left IR sensor away from the track to the white surface. You'll notice that both LEDs will be lighted up again; otherwise repeat steps 1 - 4.

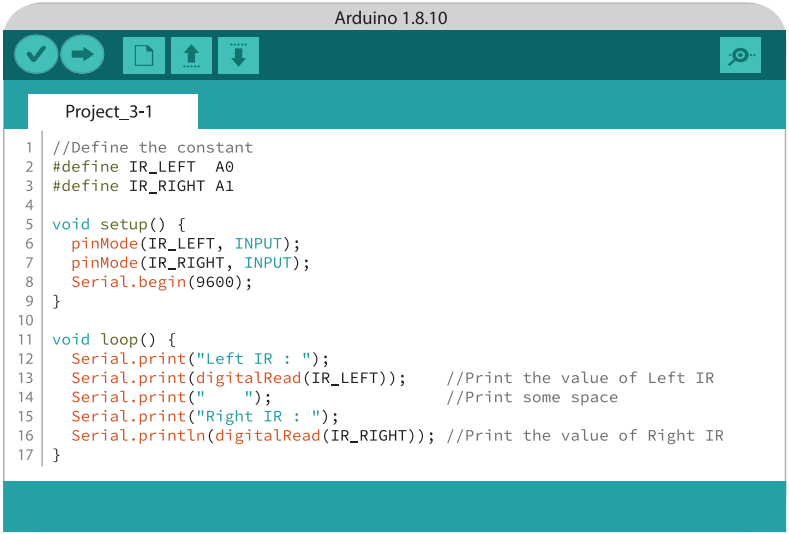
6

Repeat steps 1 - 5 to calibrate the other IR sensor.

PROJECT 3-1

Black or White Line?

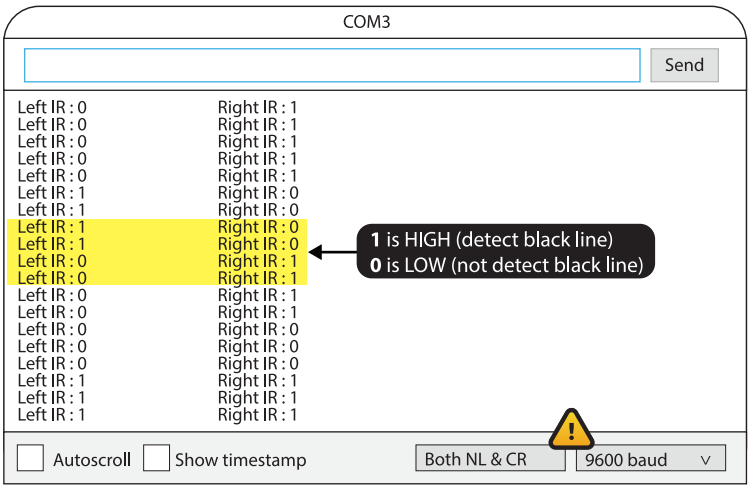
1. Open a new sketch and write the following codes. Then, compile and upload the code.

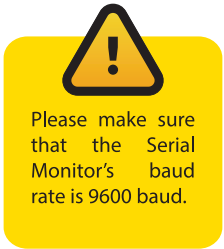



```
//Define the constant
#define IR_LEFT  A0
#define IR_RIGHT A1

void setup() {
  pinMode(IR_LEFT, INPUT);
  pinMode(IR_RIGHT, INPUT);
  Serial.begin(9600);
}

void loop() {
  Serial.print("Left IR : ");
  Serial.print(digitalRead(IR_LEFT));    //Print the value of Left IR
  Serial.print("    ");                  //Print some space
  Serial.print("Right IR : ");
  Serial.println(digitalRead(IR_RIGHT)); //Print the value of Right IR
}
```

Black or White Line?

2. Click to open the serial monitor. Place PikaBot on the track and move it to the right and then to the left of the track. Observe the IR reading.

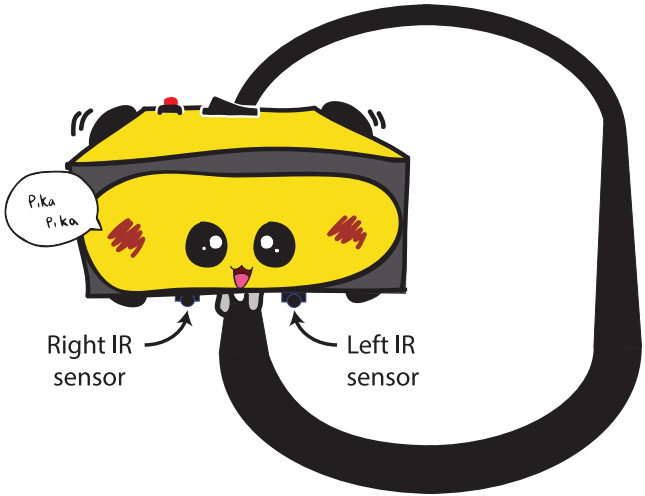




PROJECT 3-2

Black or White Line?

Since PikaBot knows how to differentiate between black and white, now we can program it to move around the track by following the black line.



PROJECT 3-2

Black or White Line?

1. Modify the previous code into this. Then, compile ✓ and upload ➡ the code.

Project_3-2

```
#include "CytronMotorDriver.h"
CytronMD motorLeft(PWM_PWM, 11, 10);
CytronMD motorRight(PWM_PWM, 9, 3);

#define BUTTON   2
#define IR_LEFT  A0
#define IR_RIGHT A1

void robotStop() {
  motorLeft.setSpeed(0);
  motorRight.setSpeed(0);
}
void robotMove(int speedLeft, int speedRight) {
  motorLeft.setSpeed(speedLeft);
  motorRight.setSpeed(speedRight);
}

```

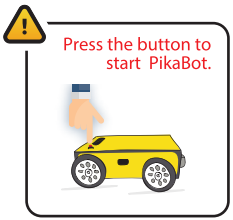
Note : The code continues on the next page.

PROJECT 3-2

Black or White Line?

Project_3-2

```
18 void setup() {
19   pinMode(IR_LEFT, INPUT);
20   pinMode(IR_RIGHT, INPUT);
21   pinMode(BUTTON, INPUT_PULLUP);
22 }
23
24 void loop() {
25   if (digitalRead(BUTTON) == LOW){
26
27     while (true) {
28       if (digitalRead(IR_LEFT) == LOW &&
29           digitalRead(IR_RIGHT) == LOW) {
30         robotMove(200, 200);  //Robot moves forward
31       }
32       else if (digitalRead(IR_LEFT) == HIGH)  {
33         robotMove(-200, 200); //Robot turns left
34       }
35       else if (digitalRead(IR_RIGHT) == HIGH) {
36         robotMove(200, -200); //Robot turns right
37       }
38     }
39   }
40 }
```

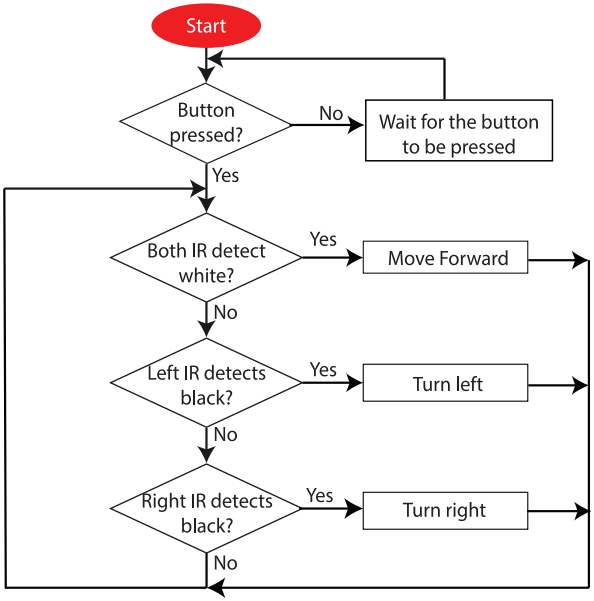


HOW it works!

PikaBot waits until the push button is pressed.

Then it moves forward when both IR detect white surface.

It will turn left when only the left IR detects black line and it will turn right when only the right IR detects black line.



HOW it works!

Project_3-2

```
18 void setup() {
19   pinMode(IR_LEFT, INPUT);
20   pinMode(IR_RIGHT, INPUT);
21   pinMode(BUTTON, INPUT_PULLUP);
22 }
23
24 void loop() {
25   if (digitalRead(BUTTON) == LOW){
26
27     while (true) {
28       if (digitalRead(IR_LEFT) == LOW &&
29           digitalRead(IR_RIGHT) == LOW) {
30         robotMove(200, 200);  //Robot moves forward
31       }
32       else if (digitalRead(IR_LEFT) == HIGH) {
33         robotMove(-200, 200); //Robot turns left
34       }
35       else if (digitalRead(IR_RIGHT) == HIGH)
36         robotMove(200, -200); //Robot turns right
37       }
38     }
39   }
40 }
```

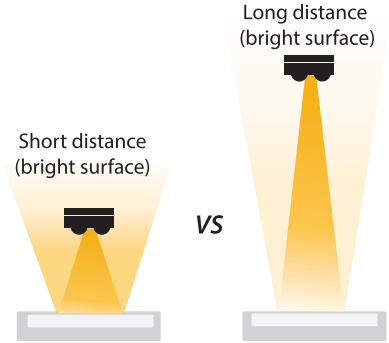
This if statement checks if both IRs are above the white surface

This if statement checks if the left sensor is above the black line.

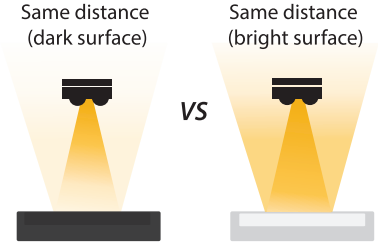
This if statement checks if the right sensor is above the black line.

Fun FACT

An Infrared (IR) sensor consists of two parts - an emitter (IR LED) and a receiver (photodiode). The IR LED emits IR light which will be reflected to the receiver if an object is placed in front of the sensor. If there is no object or the surface is dark (or black), very little or no IR light is reflected to the receiver.



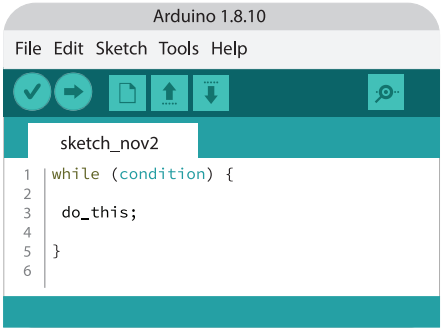
The light reflected to the IR sensor is high when the object is near and the light reflected to the IR sensor is low when the object is far.

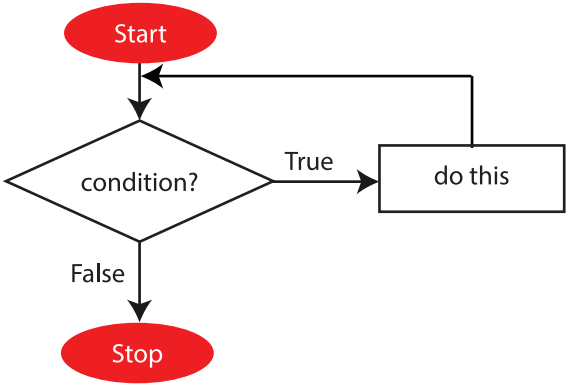


More light is reflected to the IR sensor by the bright surface compared to dark surface.

Fun FACT

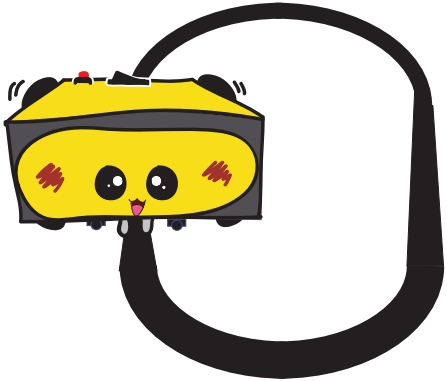
A 'while' loop will loop continuously, and infinitely, until the condition or expression inside the parenthesis () becomes false. Something must change the tested variable, or the while loop will never exit.

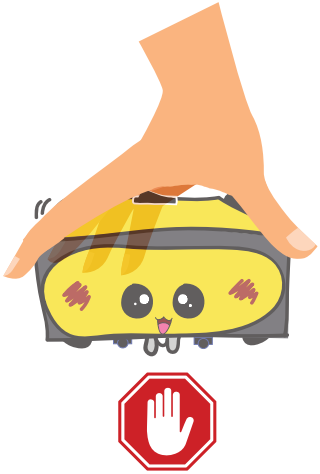




CHALLENGE

Program PikaBot to follow the black line to move around the track but stop moving if it is lifted up.

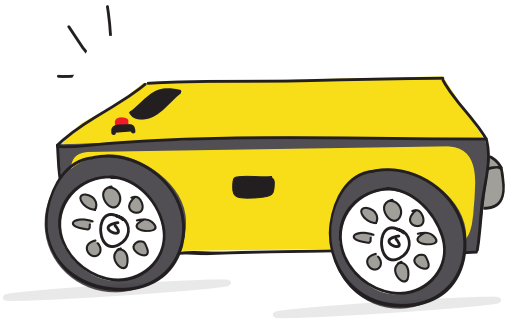




SCAN HERE
FOR EXTRA RESOURCES

PROJECT 4

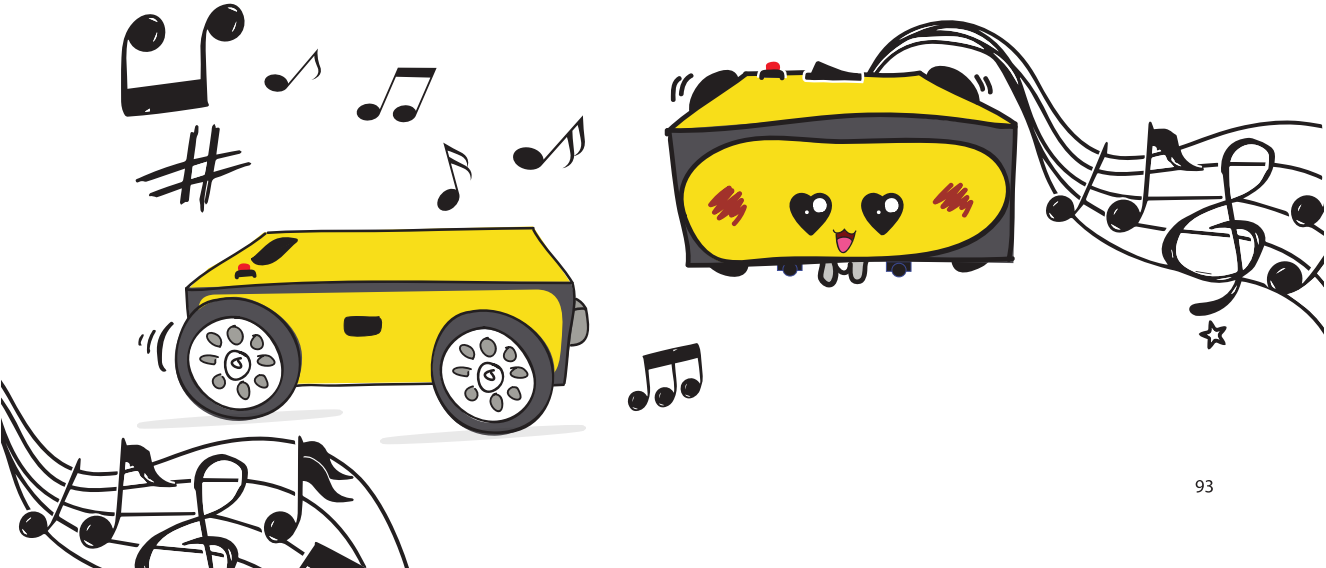
Music On



PROJECT 4

Music On

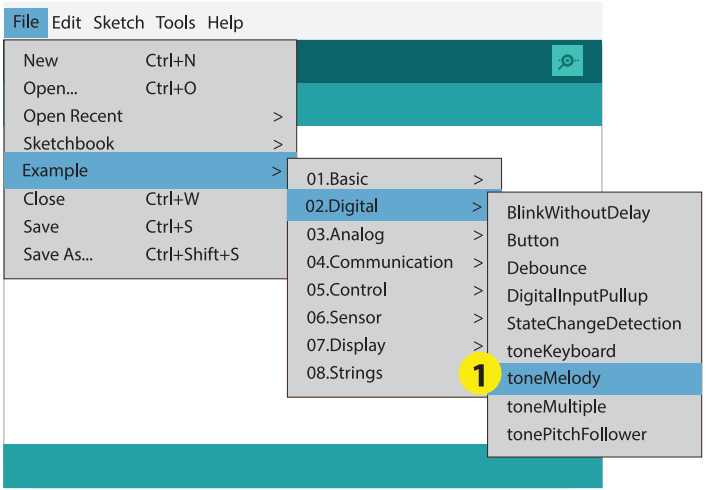
PikaBot loves music.
Let's have some fun by playing some music with PikaBot.



PROJECT 4-1

Music On

1. Open toneMelody sketch from Arduino IDE's Example. Go to **File > Examples > 02 Digital > toneMelody**. You will see a sketch with two tabs - **toneMelody** and **pitches.h**.



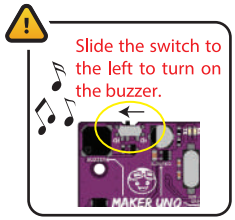
PROJECT 4-1

Music On

2. Click compile and then upload the sample code.

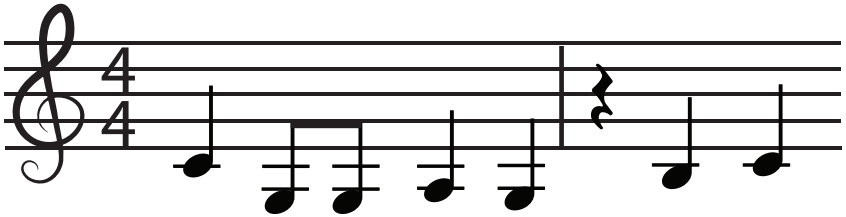
toneMelody | pitches.h

```
#include "pitches.h"

// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8, 4, 4, 4, 4, 4
};

```



HOW it works!

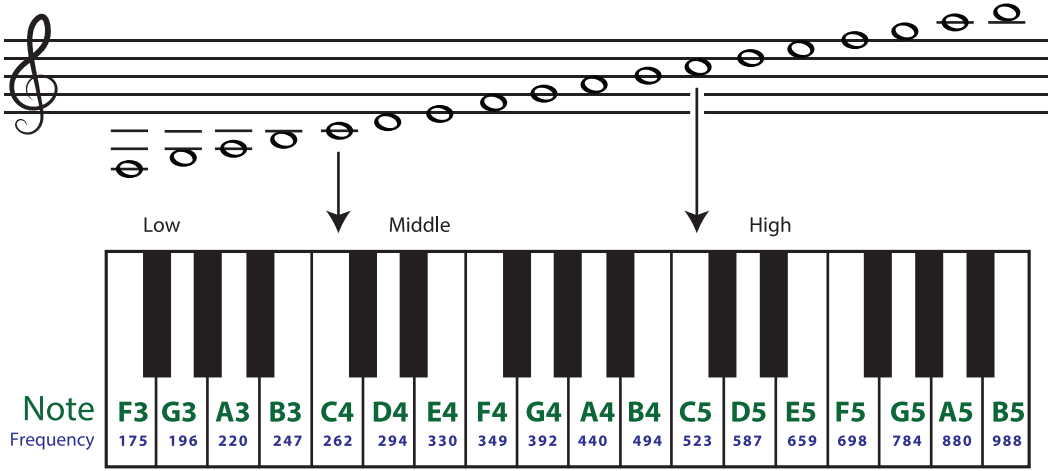
1. The melody in Arduino toneMelody example is the catchy ending of the song “Shave and a Haircut, Two Bits”. Let's take a look at the music score for this simple tune.



Note	C4	G3	G3	A3	G3	-	B3	C4
Note Duration	4	8	8	4	4	4	4	4

Music On

2. The position of a music note on the staff (the five horizontal lines) tells us which tone to play. The higher the note sits on the staff, the higher the frequency of the sound and vice versa. You can click on the pitches.h tab to see the frequencies for all 88 keys on the piano.



3. In Arduino tone library, the duration to play a quarter note is 250ms; assuming that one whole note is played for 1 second.

Relative Length	Whole Note	Half Note	Quarter Note	Eighth Note	Sixteenth Note
noteDurations	1	2	4	8	16
Duration(ms)	1000	500	250	125	63

These music notes and the corresponding note durations are defined in toneMelody sketch.

toneMelody | pitches.h

```
1  #include "pitches.h"
2
3  // notes in the melody:
4  int melody[] = {
5    NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4
6  };
7
8  // note durations: 4 = quarter note, 8 = eighth note, etc.:
9  int noteDurations[] = {
10   4, 8, 8, 4, 4, 4, 4, 4
11 };
12
```

There are a total of 8 notes, including the rest '0' in this melody.

PROJECT 4-1

Music On

toneMelody | pitches.h

```
13 void setup() {
14 // iterate over the notes of the melody:
15   for (int thisNote = 0; thisNote < 8; thisNote++) {
16
17     // to calculate the note duration, take one secon
18     //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
19     int noteDuration = 1000 / noteDurations[thisNote];
20     tone(8, melody[thisNote], noteDuration);
21
22     // to distinguish the notes, set a minimum time between them.
23     // the note's duration + 30% seems to work well:
24     int pauseBetweenNotes = noteDuration * 1.30;
25     delay(pauseBetweenNotes);
26     // stop the tone playing:
27     noTone(8);
28   }
28 }
30
31 void loop() {
32 // no need to repeat the melody.
33 }
```

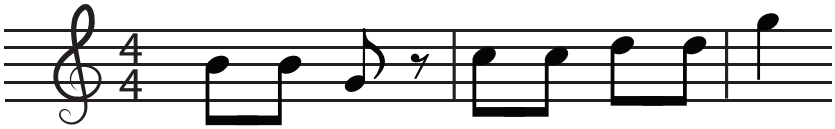
This is the number of notes to be played. We want to play 8 notes in this melody.

This refers to digital pin 8 which is connected to the built in piezo buzzer on Maker UNO.

PROJECT 4-2

Music On

1. After understanding how the program works, let's create a new melody for your PikaBot. Based on this music score, try to "decode" the notes and note durations by referring to pages 97 and 98.



Note	B4		G4	0	C5		D5		G5
Note Duration		8		8		8		8	

After playing the melody, we want PikaBot to turn left continuously.

PROJECT 4-2

Music On

2. Modify the previous code to play this melody. Then, compile and upload the code. Do you hear a new melody? Does PikaBot turn left after playing the melody?

toneMelody | pitches.h

```
1  #include "pitches.h"
2  #include "CytronMotorDriver.h"
3  CytronMD motorLeft(PWM_PWM, 11, 10);
4  CytronMD motorRight(PWM_PWM, 9, 3);
5
6  // notes in the melody:
7  int melody[] = {
8    NOTE_B4, NOTE_B4, NOTE_G4, 0,
9    NOTE_C5, NOTE_C5, NOTE_D5, NOTE_D5, NOTE_G5
10 };
11
12 // note durations: 4 = quarter note, 8 = eighth note, etc.:
13 int noteDurations[] = {
14   8, 8, 8, 8,
15   8, 8, 8, 8, 4
16 };
17
```

Insert music notes based on the sequence you want to play using the predefined notes.

Insert note durations here.

PROJECT 4-2

Music On

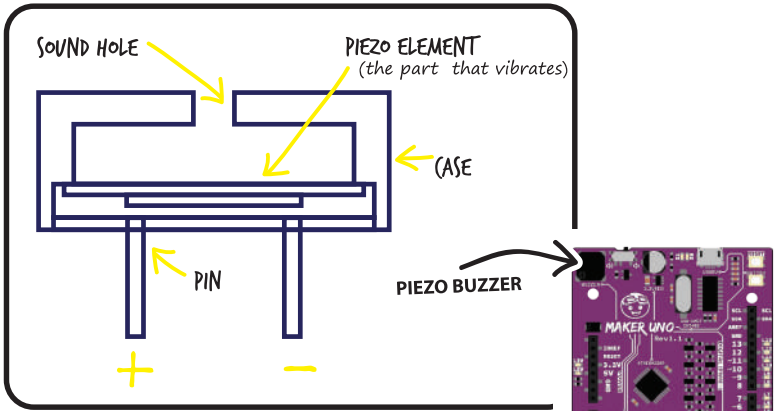
toneMelody | pitches.h

```
18 void robotMove(int speedLeft, int speedRight) {
19   motorLeft.setSpeed(speedLeft);
20   motorRight.setSpeed(speedRight);
21 }
22
23 void setup() {
24   // iterate over the notes of the melody:
25   for (int thisNote = 0; thisNote < 9; thisNote++) {
26
27     // to calculate the note duration, take one second divided by the note type.
28     //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
29     int noteDuration = 1000 / noteDurations[thisNote];
30     tone(8, melody[thisNote], noteDuration);
31
32     // to distinguish the notes, set a minimum time between them.
33     // the note's duration + 30% seems to work well:
34     int pauseBetweenNotes = noteDuration * 1.30;
35     delay(pauseBetweenNotes);
36     // stop the tone playing:
37     noTone(8);
38   }
39 }
40
41 void loop() {
42   robotMove(-200, 200); // turn left
43 }
```

Since the total number of music notes is 9, insert 9 here.

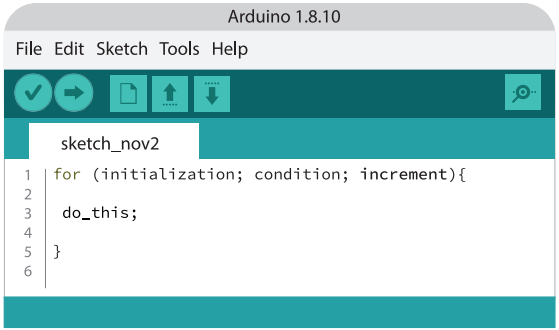
Fun FACT

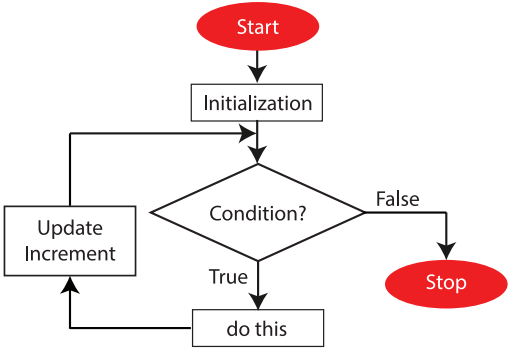
A piezo buzzer is an electronic component that is commonly used to produce sounds. The piezo element in the buzzer vibrates when electric signal passes through it. We can control the pitch of the sound produced by changing the frequency of the electric signal that passes through the piezo buzzer.



Fun FACT

The 'for loop' is used to execute certain codes for a specific number of times. Initialization is executed only once. If the condition is true, codes inside of the body is executed, and the increment is updated. This process goes on until the condition is false, then the loop terminates.

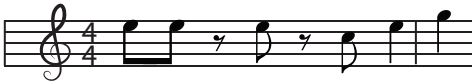




CHALLENGE

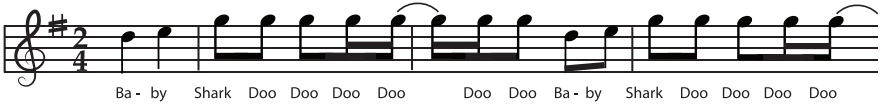
Program PikaBot to play these catchy melodies:

1. Super Mario

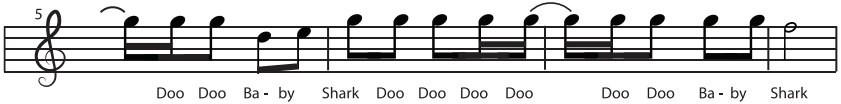


Note								
Note Duration								

2. Baby Shark.



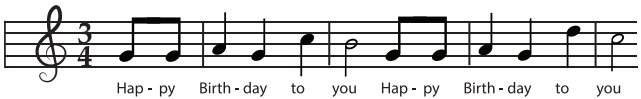
Note																	
Note Duration																	



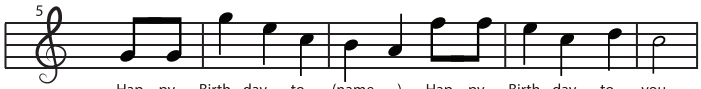
Note																	
Note Duration																	

CHALLENGE

3. Birthday Song



Note												
Note Duration												



Note												
Note Duration												

SCAN HERE
FOR EXTRA RESOURCES

CONGRATULATIONS!

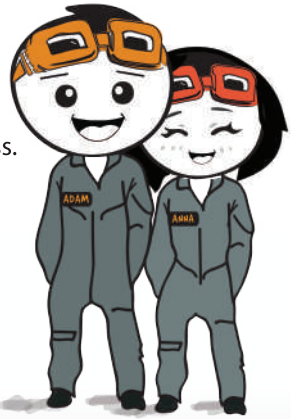
Scan here for extra resources

You've built and programmed your own PikaBot!
We hope that you've had fun along the way.

However this isn't the end of your journey with PikaBot. Do you know that you can apply the knowledge and skills you've acquired to build a new robot?

You can dismantle and reuse PikaBot parts for your next project. Get creative and reuse cardboards, milk cartons, plastic bottles, etc. for the body of the robot Besides, you can also try bringing your old toys to life! The possibilities are endless.

Remember to share your creations with us if you'd like to be featured or just drop us a message anytime. We'd love to hear from you!




The
PIKABOT