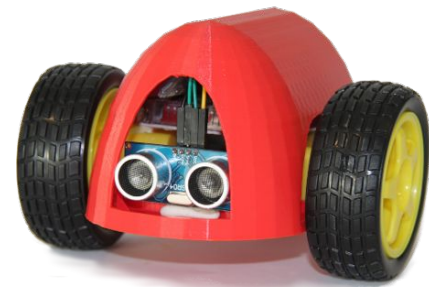
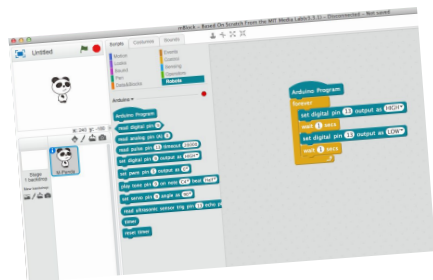
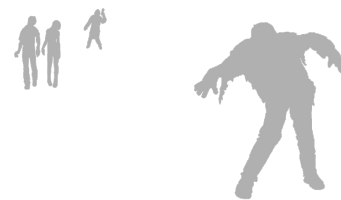


ZOMBIEBOTS!

LEARNING ELECTRONICS, ROBOTICS AND
PROGRAMMING TO SAVE US FROM THE
ZOMBIE APOCALYPSE



OVERVIEW



Chapter One: Unboxing

Chapter Two: Getting Set Up

Chapter Three: Assemble Your Robot

Chapter Four: Practice your skills

Chapter Five: Let's Get Moving

Chapter Six: Line-Following

Chapter Seven: Smart-phone Controlled

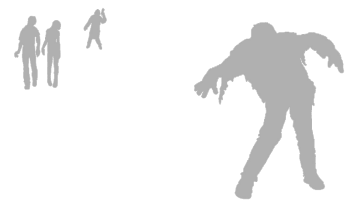
Useful Zombie fact: while slow moving, zombies never seem to tire. Run & frequently make right angle turns to escape.





CHAPTER ONE

UNBOXING

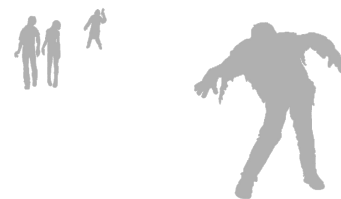


1.1 WHAT ARE ZOMBIEBOTS?

Zombiebots crawled out of the crypt because we believe kiwi kids could do with a hand to become better tinkerers, inventors and innovators. We've devised a fun way for people to learn electronics, robotics and programming while they save us from the zombie menace. The Zombiebot Nano is made out of colour-coded, snap-together parts and programmed with a drag-and-drop interface.

Because if there's a zombie apocalypse, we're going to need robots, right?





1.2 JARGON BUSTER

Arduino: a type of microcontroller (tiny computer) that can be programmed to do things. We're using an Arduino Nano

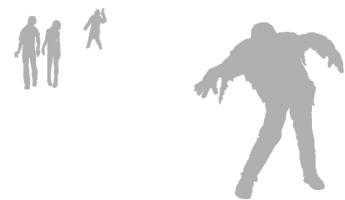
Breadboard: a plastic base that allows you to create circuits without solder. If you'd like to see how they work, [click here](#).

Jumper: a little wire that can be used to make connections between the holes of a breadboard

Microcontroller: a mini computer that you can program to do simple tasks. An Arduino is a microcontroller.

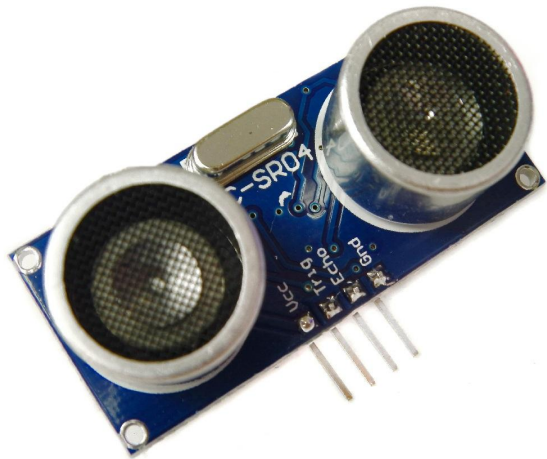
Useful Zombie fact:
they are almost entirely
silent save for the noise
they make while
shuffling around.



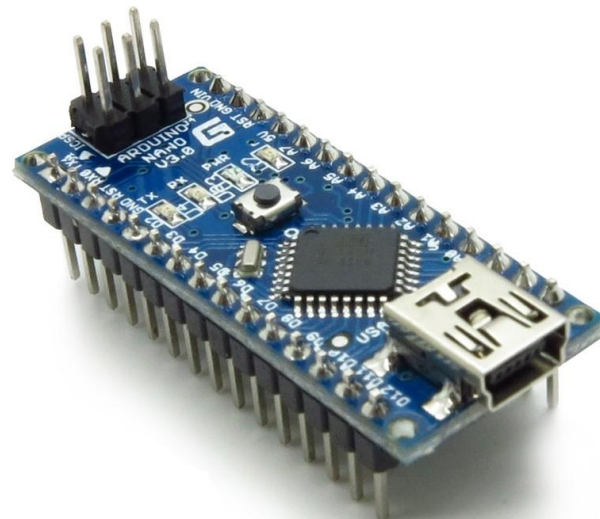


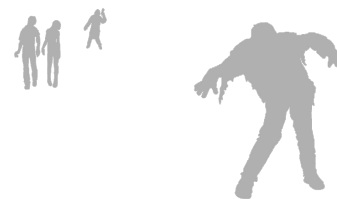
1.3 WHAT'S IN THE BOX? (NANO)

Ultrasonic sensor. Sends a pulse out and counts how long before the pulse comes back to it. Can be used to sense distance.



Arduino Nano. An Arduino is a microcontroller (like a mini computer). We're going to programme it to make lights blink, play a tune, and eventually control our robot.



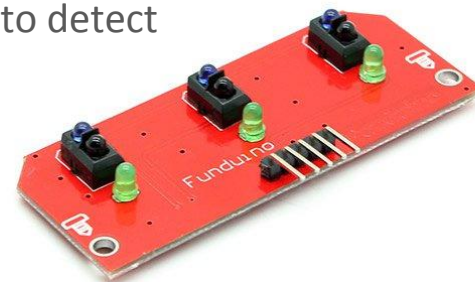


1.4 WHAT'S IN THE BOX? (NANO PLUS)

Bluetooth module. This allows us to communicate with our Arduino via Bluetooth. The easiest way to do this is via an Android smartphone that is Bluetooth capable.



Infrared sensor array. This is a set of three infrared sensors that can detect things like colour. We use it to detect the difference between black and white so your robot can follow a line.



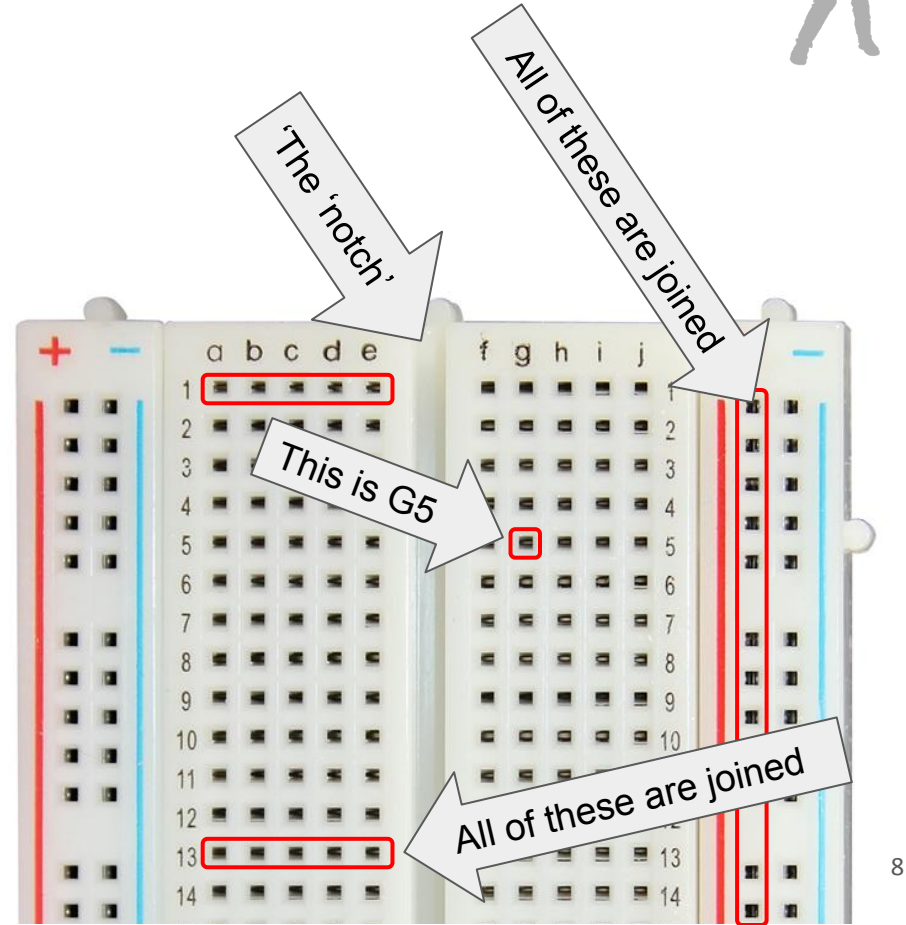
1.5 BREADBOARD

A breadboard is sometimes called a prototyping board and is an easy way to make circuits without soldering. If you put a wire into 1A, and another into 1B (or 1C, 1D or 1E) they are joined by clips inside the breadboard.

The notch separates the two sides of the board, so A-E and F-J are separate.

The 25 pins in each (+) and (-) strip down the sides are also linked together. These are called power rails.

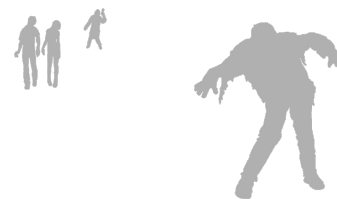
Got the idea? Take our '[How to use a breadboard](#)' video quiz





CHAPTER TWO

GETTING SET UP



2.1 GET THE SOFTWARE

To program our robot, we are going to use software called mBlock.



[Home](#) [Blog](#)

You can download and install it from here:

www.mblock.cc/download



Windows Download

Version 3.3.4
Supports windows XP
Windows 7 and above recommended



Mac Download

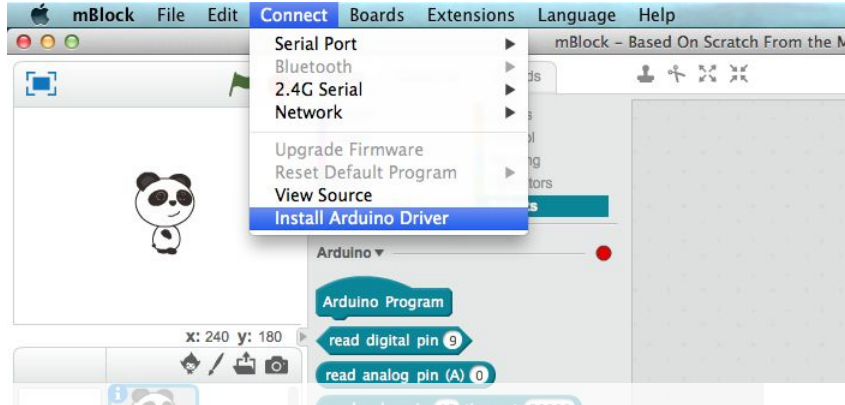
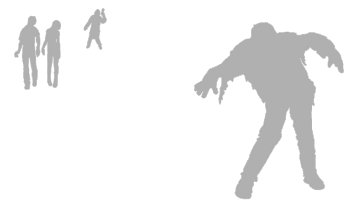
Version 3.3.4
Latest OSX
recommended

Useful Zombie hint:

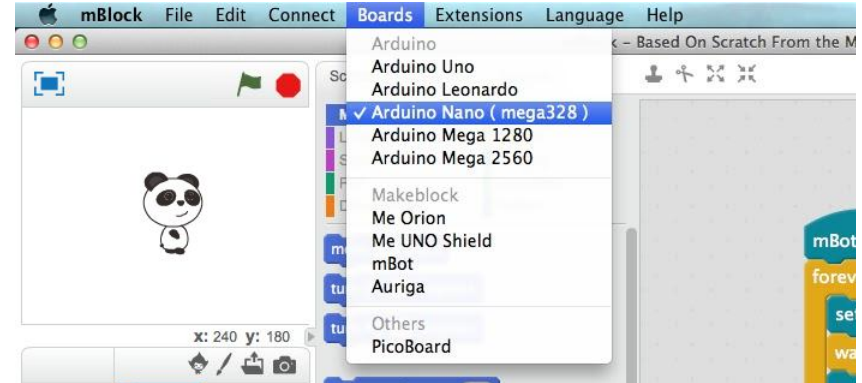
always keep a pair of running shoes at the front
and the back doors.



2.2 SET UP THE SOFTWARE



1. From the 'Connect' menu, choose 'Install Arduino Driver'. This ensures your computer will be able to talk to your Arduino/robot.



2. From the 'Boards' menu, choose Arduino Nano (mega328). This tells your computer what kind of Arduino you're using.



2.3 SIMPLE SCRIPTS

The screenshot shows the mBlock software interface. The top bar indicates 'mBlock - Based On Scratch From the MIT Media Lab(v3.2.2) - Disconnected - Not saved'. The main workspace shows a stage with a panda sprite. The 'Scripts' tab is selected, displaying a list of event blocks: 'when green flag clicked', 'when space key pressed', 'when space key released', 'when this sprite clicked', 'when backdrop switches to backdrop1', 'when loudness > 10', 'when I receive message1', 'broadcast message1', and 'broadcast message1 and wait'. A large white arrow points from the 'when space key pressed' block to the text area on the right.

(This is the scripts area. Make the panda do things by dragging actions into here.)

Most computers and microcontrollers use a combination of 'input' and 'output' to get things done. For instance if you press a key on the keyboard (input) the robot will do something (output). Arduino programmes are called sketches, and we're going to build our first sketch now, to control the panda, before moving onto the robot.



2.4 MAKE THE PANDA MOVE

mBlock – Based On Scratch From the MIT Media Lab(v3.3.1) – Disconnected – Not saved

moving panda

Scripts

Motion

move 10 steps

turn 15 degrees

turn 15 degrees

point in direction 90°

point towards

go to x: -9 y: -8

go to mouse-pointer

glide 1 secs to x: -9 y: -8

change x by 10

set x to 0

change y by 10

set y to 0

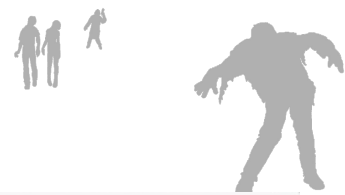
when right arrow key pressed

move 10 steps

when left arrow key pressed

move -10 steps

Click on the Events tab (orange) and drag the 'When space key pressed' block into the scripts area. Change 'space' to 'right arrow'. A key press is our *input*; now we need an *output* (so the panda does something). Click the Motion tab and drag the 'move 10 steps' block and snap it to your orange one. Repeat with -10 steps for left arrow.



2.4 MAKE THE PANDA MOVE

mBlock – Based On Scratch From the MIT Media Lab(v3.3.1) – Disconnected – Not saved

moving panda

Scripts

- when green flag clicked
- when space key pressed
- when space key released
- when this sprite clicked
- when backdrop switches to backdrop1
- when loudness > 10
- when I receive message1
- broadcast message1
- broadcast message1 and wait

Costumes

Sounds

Events

- Control
- Sensing
- Operators
- Robots

when right arrow key pressed

move 10 steps

when left arrow key pressed

move -10 steps

Stage

1 backdrop

New backdrop:

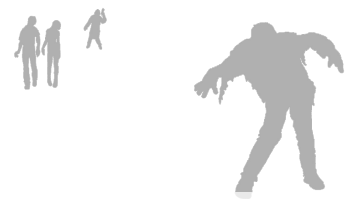
M-Panda

x: 240 y: -180

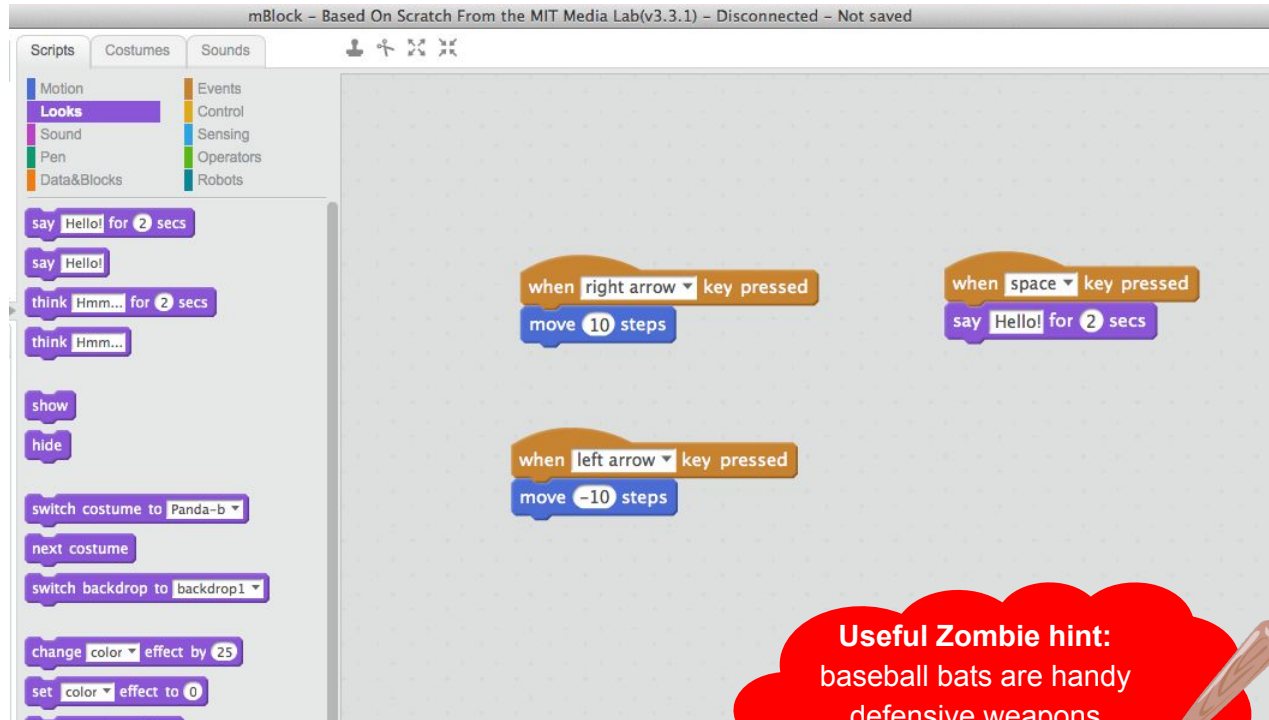
Wherever you see these icons, you can download a copy of the sketch for mBlock or Arduino

Press the right arrow and watch the panda. What happens?

Challenge: can you move the panda left and right, by dragging a second set of Event/Motion blocks into the script area alongside your existing ones?



2.5 MAKE THE PANDA SAY HELLO!



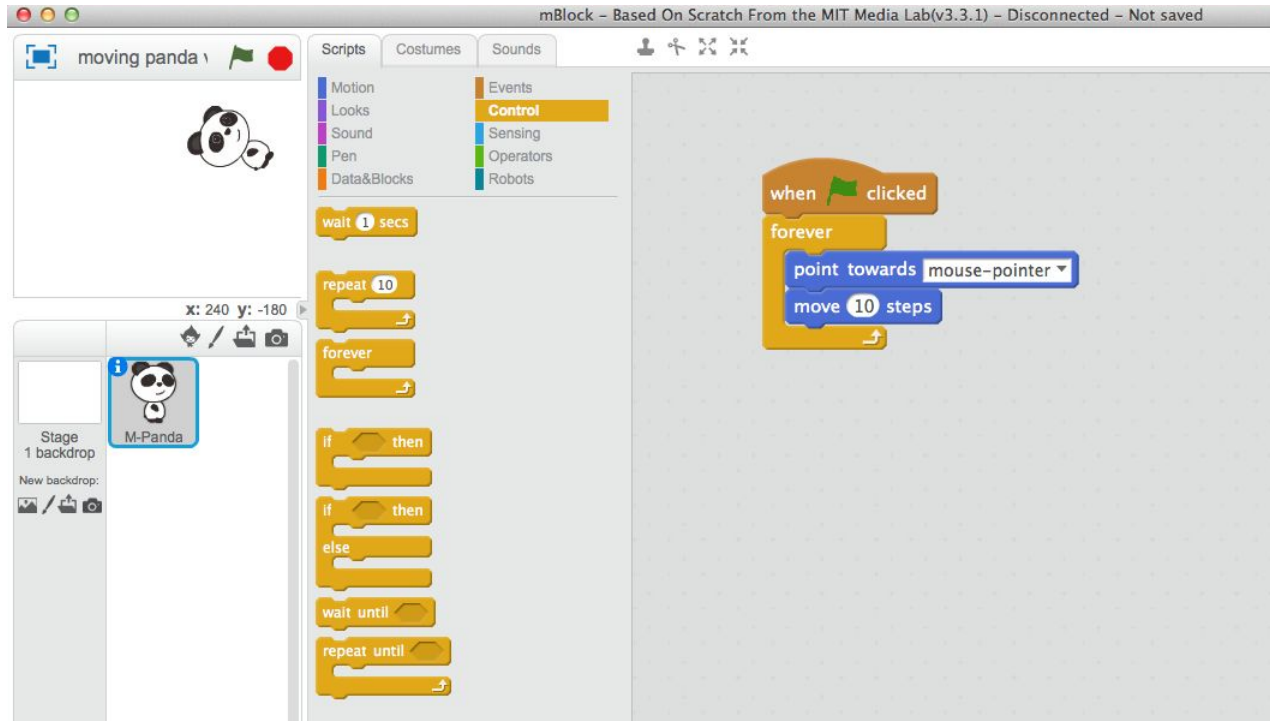
Now we're going to get our panda to do a bit more. Add another space key press event (orange) and this time snap the the 'Say Hello for 2 secs' block (purple) to it. Hit space key to test it out.

Challenge: can you change what the panda says, and make him say it for longer (or shorter)? (Play around with the options.)

Useful Zombie hint:
baseball bats are handy defensive weapons because they don't require ammunition.



2.6 MAKE THE PANDA FOLLOW YOUR MOUSE



Let's get a bit tricky now. Instead of using a key press for input, let's use the mouse.

Find the blocks on the left and assemble them in the correct order. (Choose 'mouse pointer' from the drop down menu in the 'point towards' block). Play with the number of steps to make the panda faster or slower.



2.7 KEEPING SCORE WITH VARIABLES

mBlock – Based On Scratch From the MIT Media Lab(v3.3.1) – Disconnected – Not saved

moving panda

Scripts Costumes Sounds

Motion Looks Sound Pen Data&Blocks Events Control Sensing Operators Robots

wait 1 secs

repeat 10

forever

if then

if then else

wait until

repeat until

x: 240 y: -180

Stage 1 backdrop

New backdrop:

M-Panda

when green flag clicked

set score to 0

forever

point towards mouse-pointer

move 5 steps

if touching edge ? then

change score by 1

if touching mouse-pointer ? then

change score by -1

Variables are containers that can hold store numbers that change (like a score or a countdown).

We're going to make a game that uses a variable to keep score. Using our 'panda chasing the mouse' sketch, we're going to set it up so that if the panda touches the edge of the box we gain a point, and if the panda touches the mouse pointer we lose a point.



2.7 KEEPING SCORE WITH VARIABLES

mBlock – Based On Scratch From the MIT Media Lab(v3.3.1) – Disconnected – Not saved

moving panda

Scripts

Motion

Looks

Sound

Pen

Data&Blocks

Events

Control

Sensing

Operators

Robots

wait 1 secs

repeat 10

forever

if then

if then else

wait until

repeat until

when green flag clicked

set score to 0

forever

point towards mouse-pointer

move 5 steps

if touching edge ? then

change score by 1

if touching mouse-pointer ? then

change score by -1

Stage

1 backdrop

New backdrop:

M-Panda

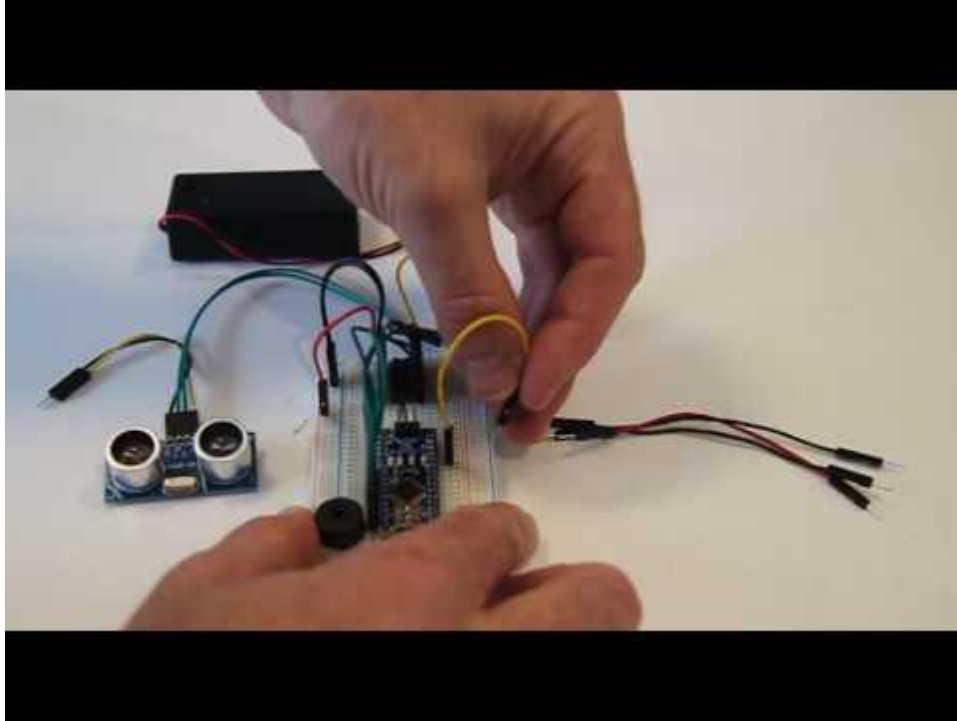
Click on Data&Blocks and click Make a Variable. Call it 'score'. Drag in 'set score to 0' and assemble the rest of the blocks so your sketch looks like the one on the left. (Choose 'mouse pointer' from the drop down menu in the 'point towards' block). Hit the green flag to play! **Challenge:** can you change the difficulty by making the panda faster or slower?

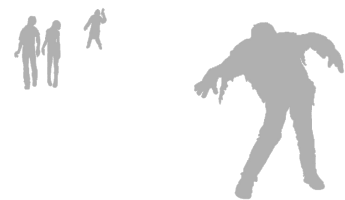


CHAPTER THREE

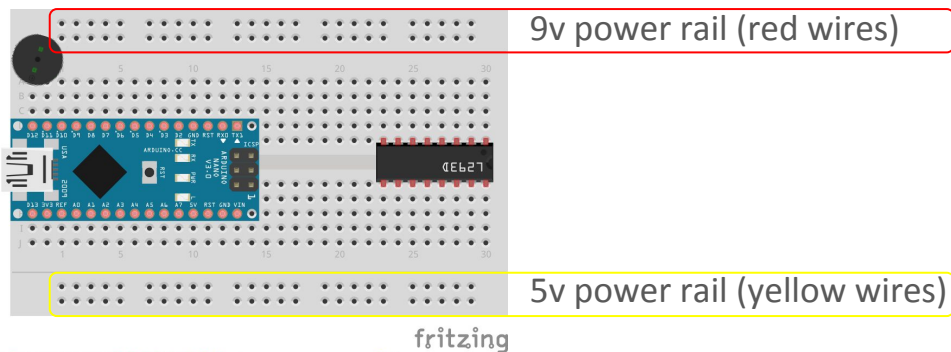
ASSEMBLE YOUR ROBOT

ASSEMBLY VIDEO: STEP ONE





3.1 START WITH THE BREADBOARD

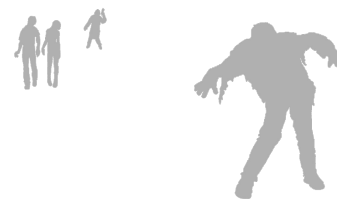


Start with your breadboard and Arduino. You'll notice two other items on the breadboard:

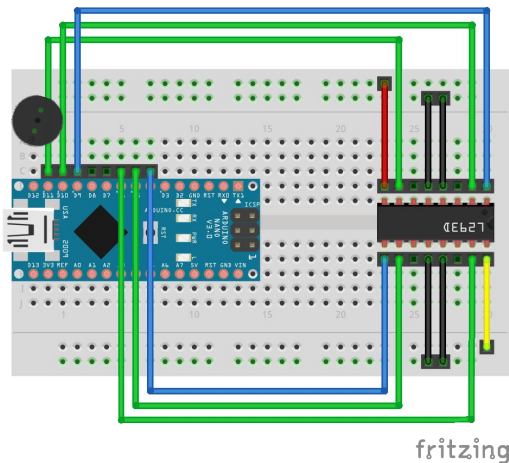
1. the round piezo buzzer (we'll use this to make a bit of noise), and
2. the chip that controls the motors (we'll send signals to the chip to get it to move the motors forward or back).

Useful Zombie hint: have an emergency kit with enough food, water & supplies to last at least two days until you reach a safe zone.





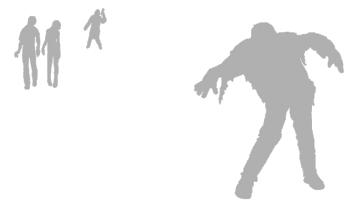
3.2 ADD WIRE ASSEMBLY



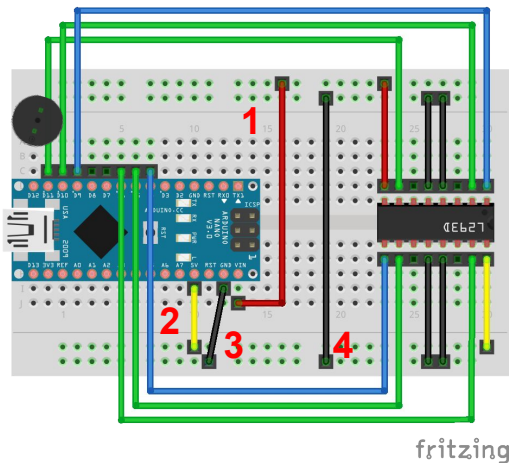
Add the wiring assembly, matching up the colours correctly.

Follow the diagram to the left to ensure the three blocks go in holes:

- C22-C29 (blue and green, with the blue wires in C22 and C27)
- D1-D8 (red wire in D8)
- G1-G8 (yellow wire in G1)



3.3 ADD JUMPER WIRES



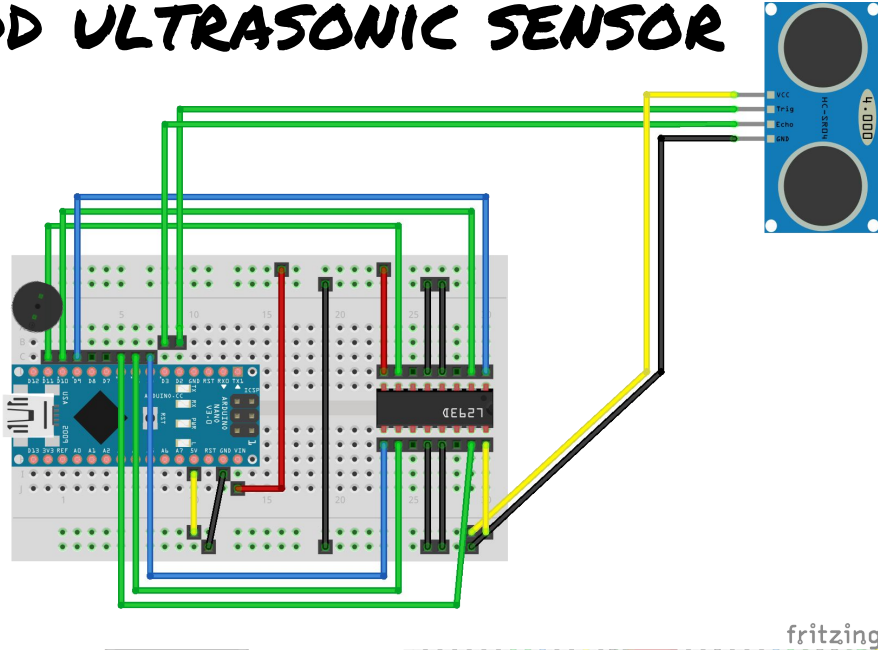
9v power rail (red wires)

5v power rail (yellow wires)

Now add these jumper wires:

1. Red from the red (+) side of the 9v power rail to the VIN pin (J16)
2. Yellow from the 5v pin (J19) to the red (+) side of the 5v power rail.
3. Black from J17 to the blue (-) side of the 5v power rail.
4. Black from the blue (-) side of the 9v power rail to the blue (-) side of the 5v power rail.

3.4 ADD ULTRASONIC SENSOR

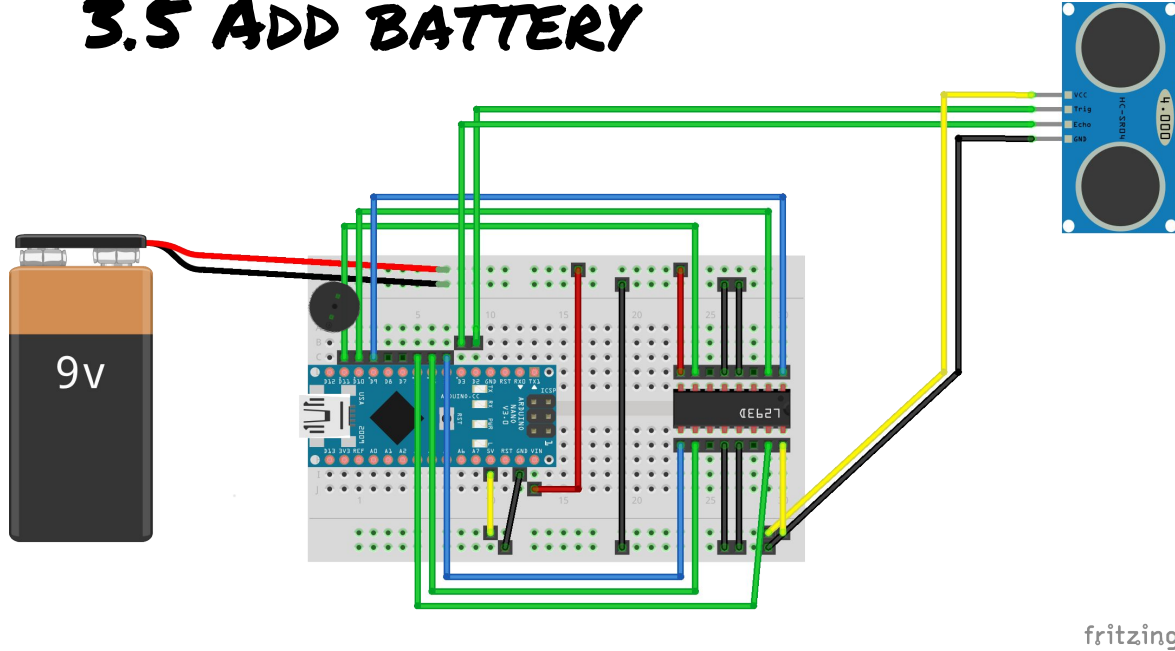


Connect the ultrasonic sensor:

1. Plug the yellow (VCC) & black (GND) wires from the Ultrasonic sensor into the 5v power rail, matching the colours (yellow + , black -).
2. Connect the green TRIG jumper to Arduino pin D2 (B20)
3. Connect the green ECHO jumper to Arduino pin D3 (B21)



3.5 ADD BATTERY

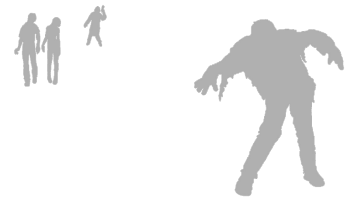


fritzing

Add your battery by pushing the green screw terminal into the 9v power rail, taking care to match the colours: red and black. The battery will slot under the breadboard between the motors.

Your battery can be recharged by taking it out of the black holder and plugging it into a micro USB phone charger.

Important: don't recharge it in a traditional 9v battery charger.



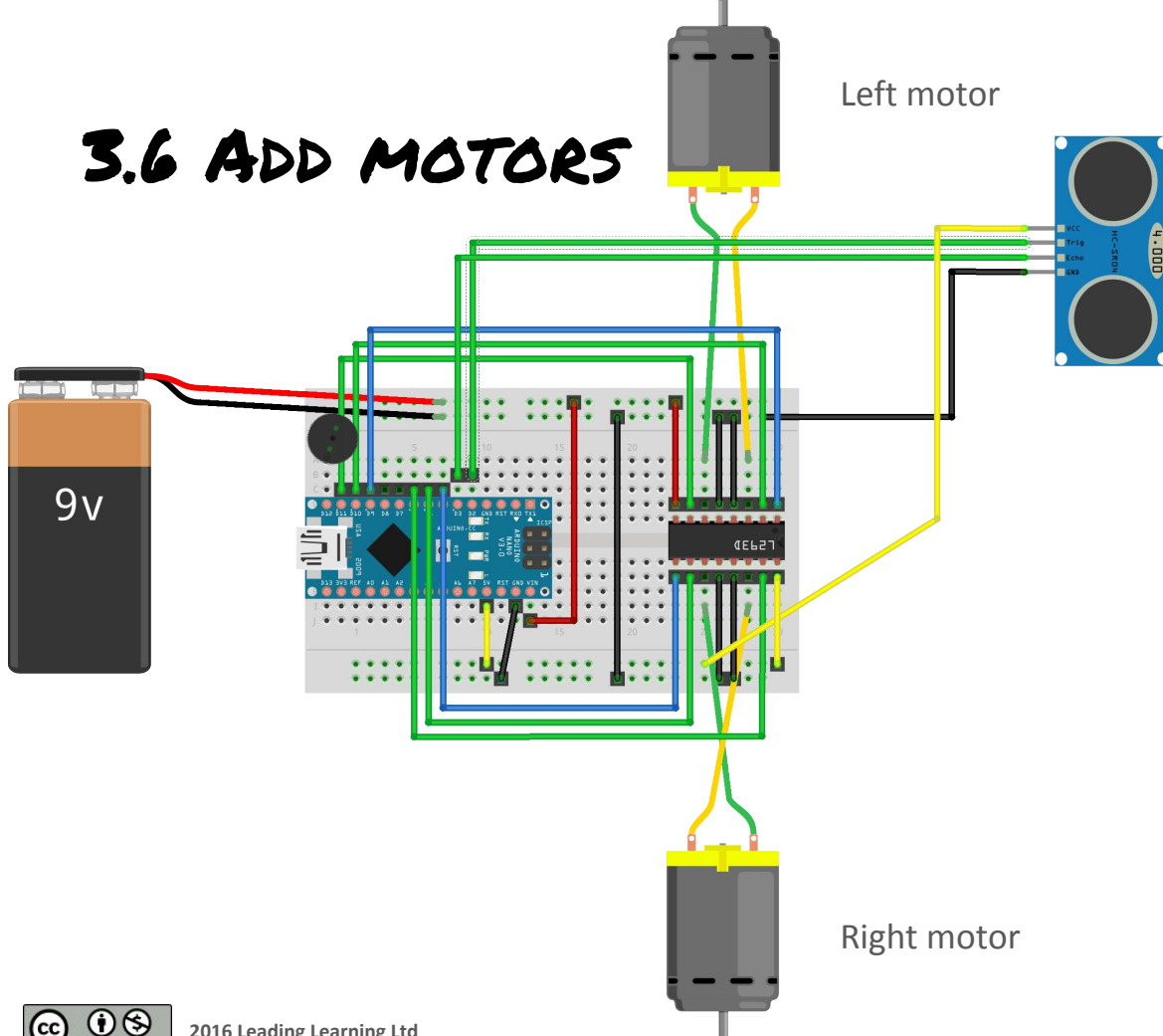
ASSEMBLY VIDEO: STEP TWO



Useful Zombie fact:
sporting goods stores
in shopping malls
make great refuges.



3.6 ADD MOTORS

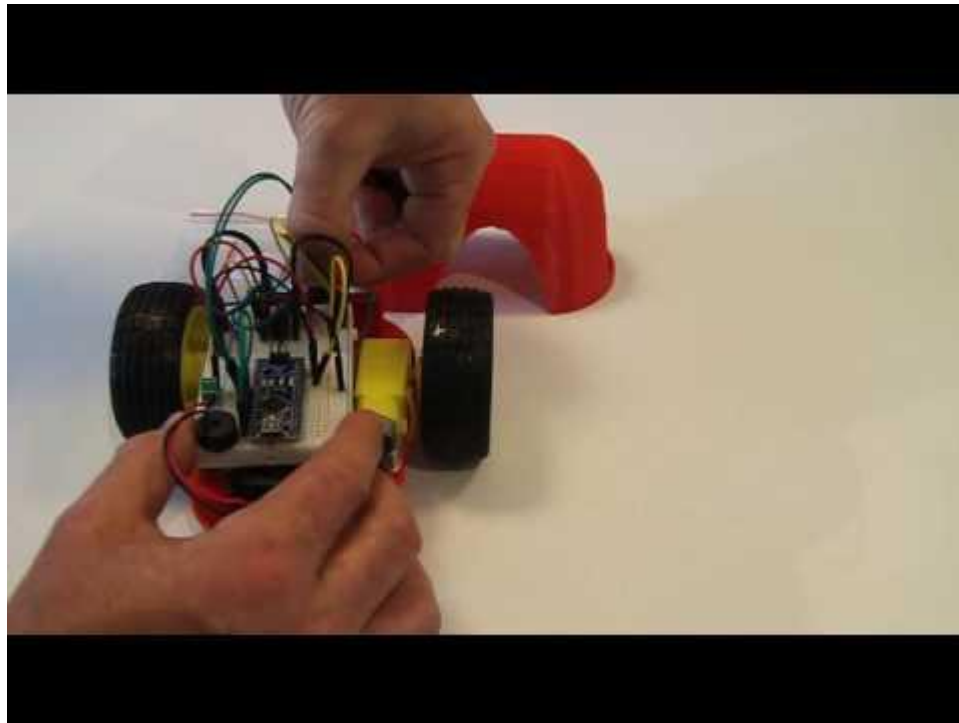


Time to add the motors. Bring the wires forward underneath the breadboard so they are at the front of the base.

1. Connect the two wires from the left motor into breadboard holes C3 (yellow) and C6 (green).
2. Repeat for the right motor using holes H3 (yellow) and H6 (green)



ASSEMBLY VIDEO: STEP THREE





CHAPTER FOUR

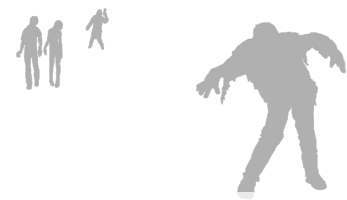
PRACTICE YOUR SKILLS



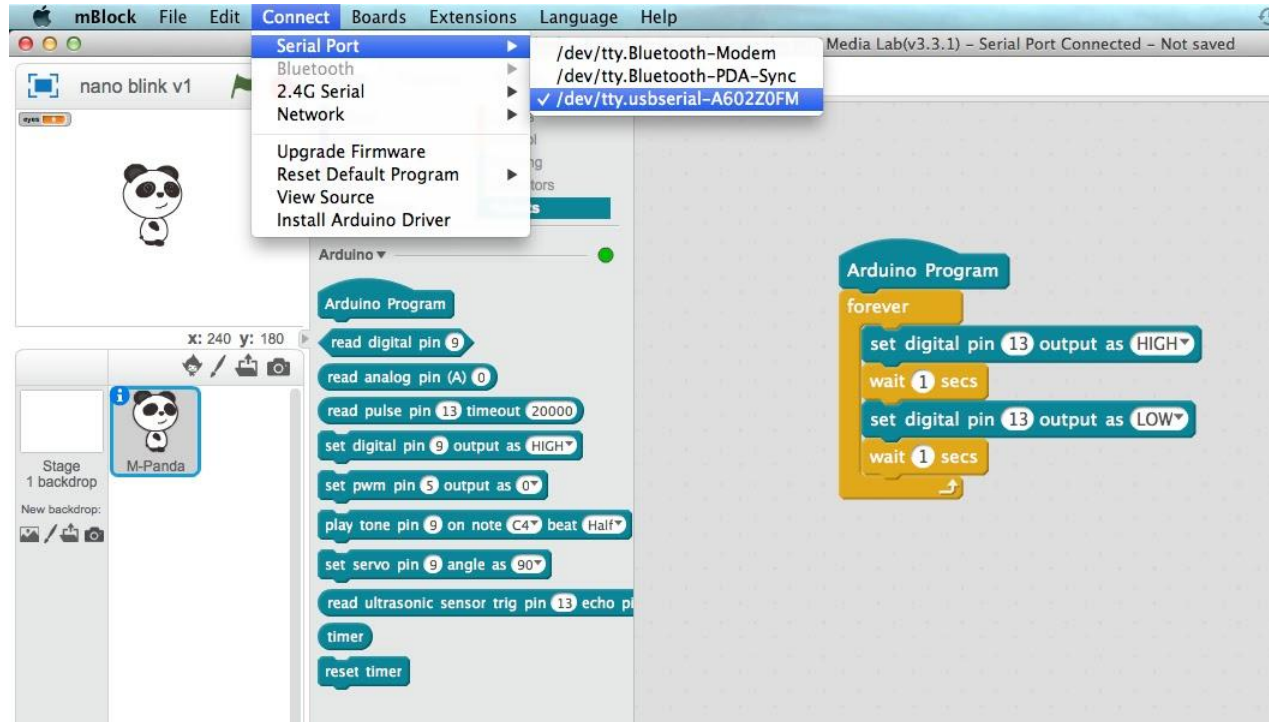
4.1 MAKE A LIGHT BLINK

The screenshot shows the mBlock software interface. The top bar indicates 'mBlock - Based On Scratch From the MIT Media Lab(v3.3.1) - Disconnected - Not saved'. The left pane shows a 'Scripts' tab with a list of categories: Motion, Looks, Sound, Pen, Data&Blocks, Events, Control, Sensing, Operators, and Robots. The 'Robots' category is selected. Below this, the 'Arduino' dropdown is expanded, showing a list of blocks: 'Arduino Program', 'read digital pin 9', 'read analog pin (A) 0', 'read pulse pin 13 timeout 20000', 'set digital pin 9 output as HIGH', 'set pwm pin 5 output as 0%', 'play tone pin 9 on note C4 beat Half', 'set servo pin 9 angle as 90', 'read ultrasonic sensor trig pin 13 echo pin', 'timer', and 'reset timer'. A large white arrow points from the 'set digital pin 9 output as HIGH' block in the left pane to the 'set digital pin 13 output as HIGH' block in the right pane. The right pane shows the 'Arduino Program' block with a 'forever' loop containing the following blocks: 'set digital pin 13 output as HIGH', 'wait 1 secs', 'set digital pin 13 output as LOW', and 'wait 1 secs'. The bottom right corner of the interface features a blue gear icon with a panda face and an 'ARDUINO' logo.

There is an LED light built into your Arduino. This sketch will make it flash on and off a second at a time. From the 'Robots' and 'Control' tabs, drag these actions across and assemble them in this order. You'll need to change the digital pin from 9 to 13 because that's where our LED is.



4.2 CONNECT YOUR NANO



1. From the 'Connect' Menu, choose 'Serial Port' then `dev/tty.usbserial-XXXXXXX` (COM on Windows)
2. Click the 'Robots' tab to make sure the red light has turned to green. This indicates you are connected.

4.3 UPLOAD YOUR SKETCH



mBlock – Based On Scratch From the MIT Media Lab(v3.3.1) – Disconnected – Not saved

nano blink v1

Scripts Costumes Sounds

Motion Looks Sound Pen Data&Blocks Events Control Sensing Operators Robots

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 12

timer

reset timer

1. Right click on the sketch and select 'upload to arduino'

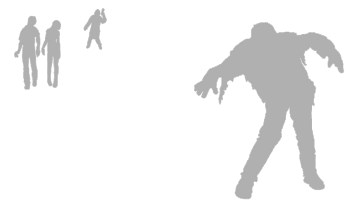
2. Click the 'Upload to Arduino' button

```
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <SoftwareSerial.h>
4
5 double angle_rad = PI/180.0;
6 double angle_deg = 180.0/PI;
7
8 void setup(){
9   pinMode(4,OUTPUT);
10  analogWrite(4,255);
11 }
12
13 void loop(){
14   pinMode(5,OUTPUT);
15   analogWrite(5,0);
16   pinMode(6,OUTPUT);
17   analogWrite(6,150);
18   delay(1000*1);
19   pinMode(5,OUTPUT);
20   analogWrite(5,0);
21   pinMode(6,OUTPUT);
22   analogWrite(6,0);
23   delay(1000*1);
24 }
```

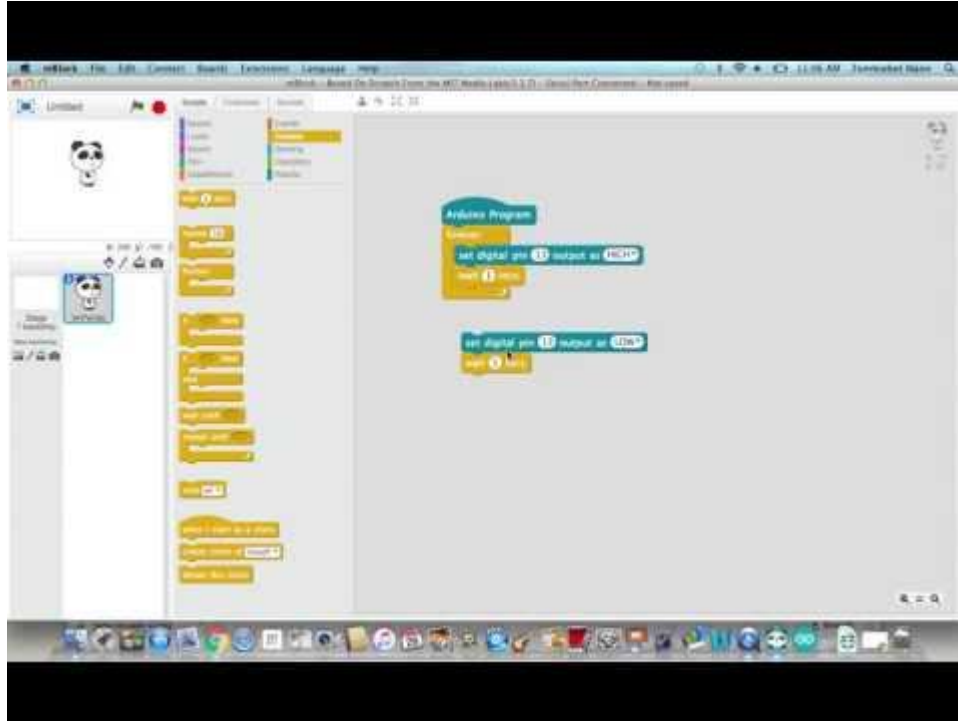
avrdude: load data flash data from input file /var/folders/m/...
avrdude: input file /var/folders/m3/ff_q05hj1517p_3spv38g0qw/
avrdude: reading on-chip flash data:
Reading | #####

Now to upload to your Arduino (make sure it is plugged in and connected).

1. Right click on the sketch and select 'upload to Arduino'
2. Click the 'Upload to Arduino' button.



4.3 UPLOAD YOUR SKETCH



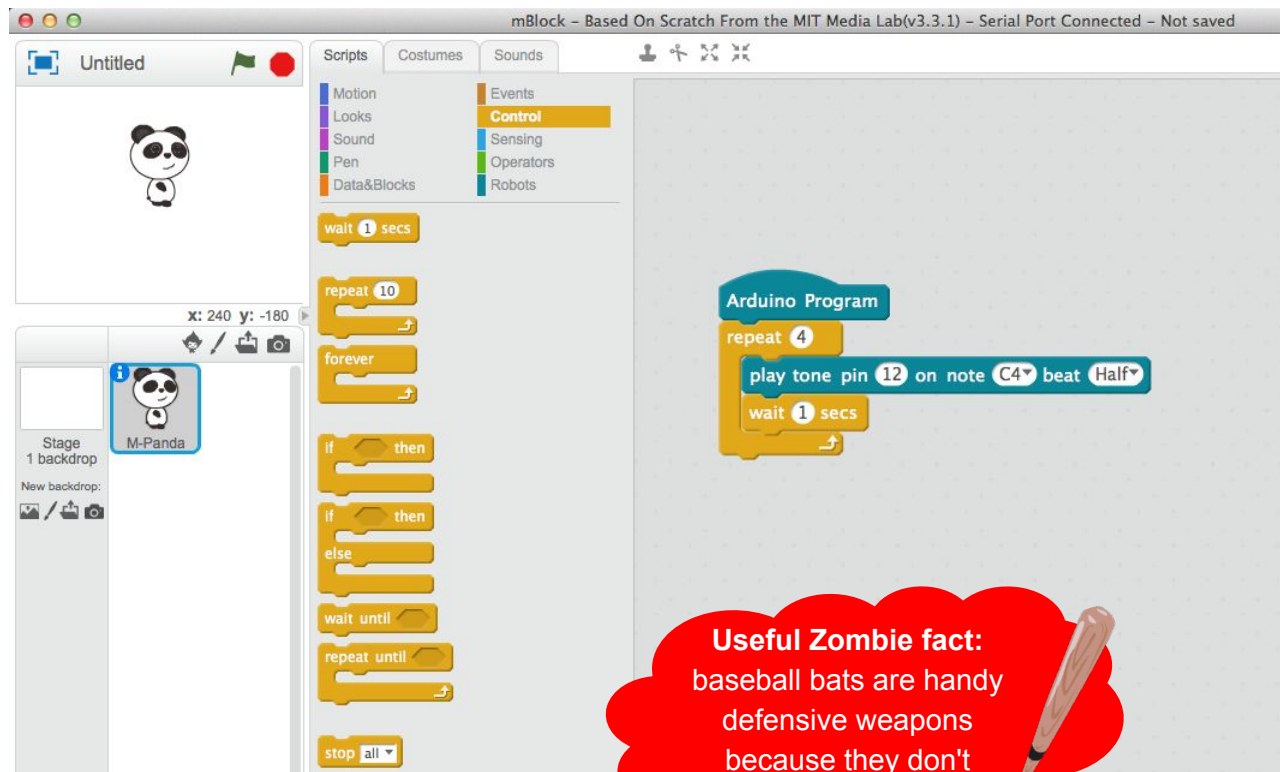
Video tutorial: build a simple sketch to make the LED blink and upload it to your Arduino

Challenge: can you make the LED blink faster or slower? What would you need to change? Re-upload to see if you are right.

Can you make the LED blink quickly three times then wait for two seconds then blink quickly three time again?



4.4 MAKE A SOUND



Useful Zombie fact:
baseball bats are handy
defensive weapons
because they don't
require ammunition.



We're going to use the piezo buzzer on our breadboard to make sound now. The software has tones built into the 'Robots' tab. Make a sketch like this one (you'll need to change the pin to 12, because that's where our buzzer is). Connect via USB and upload the sketch. **Challenge:** can you play three different tones in a row?



4.5 PLAY A TUNE

mBlock – Based On Scratch From the MIT Media Lab(v3.3.1) – Serial Port Connected – Not saved

Scripts Costumes Sounds

nanoplay-aton

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 12

timer

reset timer

Arduino Program

repeat 1

play tone pin 12 on note C4 beat Quarter

play tone pin 12 on note G3 beat Eighth

play tone pin 12 on note G3 beat Eighth

play tone pin 12 on note A3 beat Quarter

play tone pin 12 on note G3 beat Quarter

play tone pin 12 on note 0 beat Quarter

play tone pin 12 on note B3 beat Quarter

play tone pin 12 on note C4 beat Quarter

M-Panda

Stage 1 backdrop

New backdrop:

Arduino

Build this sketch and upload it to your Arduino. Recognise the tune?

Challenge: can you get your Arduino to play this tune, wait 1 sec, then play it a second time, then stop? What will you need to add/change to make it work?

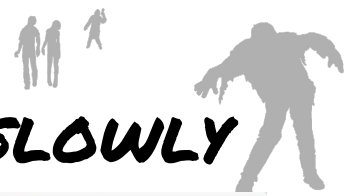




CHAPTER FIVE

LET'S GET MOVING!

5.1 MAKE THE RIGHT MOTOR GO FORWARD SLOWLY



mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Saved

Scripts Costumes Sounds

motion Looks Sound Pen Data&Blocks Events Control Sensing Operators Robots

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 12

timer

reset timer

Arduino Program

set pwm pin 4 output as 255

activate right motor

forever

set pwm pin 5 output as 150

set pwm pin 6 output as 0

wait 1 secs

set pwm pin 5 output as 0

set pwm pin 6 output as 0

wait 1 secs

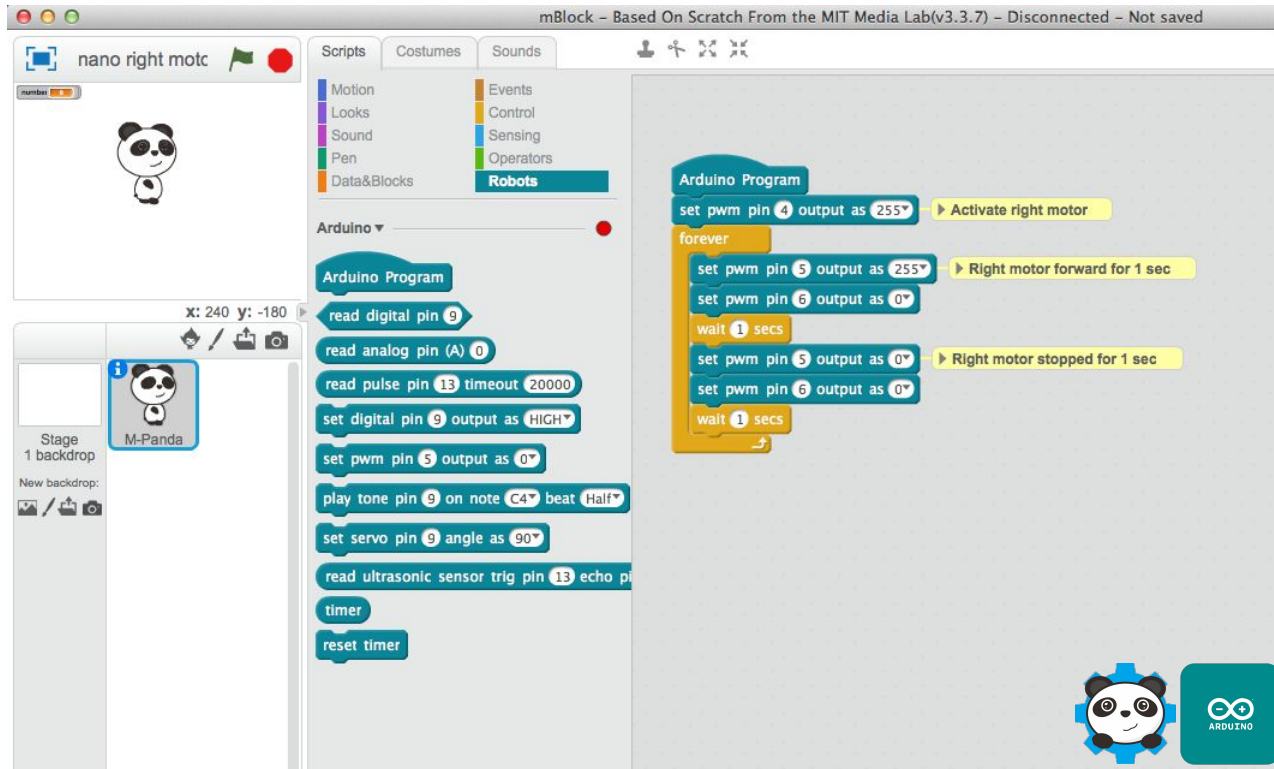
Right motor forward for 1 sec

Right motor stopped for 1 sec

We can control each of our two motors independently of each other. We're going to start by turning the left motor on. (Your robot will move in a circle.)

Build this sketch (taking care the the pwm pin numbers). Attach the USB cable, *connect* and upload this sketch.

5.2 MAKE THE RIGHT MOTOR GO FORWARD QUICKLY

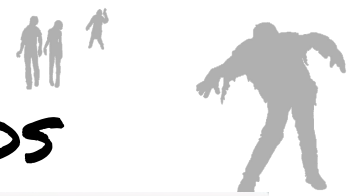


The screenshot shows the mBlock software interface, titled "mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Not saved". The interface includes a stage area on the left with a panda character named "M-Panda" and a "Stage 1 backdrop". The "Scripts" tab is selected, showing a list of blocks: Motion, Looks, Sound, Pen, Data&Blocks, Events, Control, Sensing, Operators, and Robots. The "Robots" block is highlighted. The main workspace displays an "Arduino Program" script with the following blocks:

- set pwm pin 4 output as 255 (comment: Activate right motor)
- forever loop:
 - set pwm pin 5 output as 255 (comment: Right motor forward for 1 sec)
 - set pwm pin 6 output as 0
 - wait 1 secs
 - set pwm pin 5 output as 0 (comment: Right motor stopped for 1 sec)
 - set pwm pin 6 output as 0
 - wait 1 secs

The bottom right corner of the interface features a blue gear icon with a panda face and an "ARDUINO" logo.

To increase the speed of the motor, increase the output number of pin 5 in the first part of the sketch from 150 to 255 (this is the maximum speed).



5.3 MAKE THE RIGHT MOTOR GO BACKWARDS

mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Not saved

nano right motc

Scripts Costumes Sounds

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 12

timer

reset timer

Arduino Program

set pwm pin 4 output as 255

activate right motor

forever

set pwm pin 5 output as 0

set pwm pin 6 output as 150

Right motor backwards for 1 sec

wait 1 secs

set pwm pin 5 output as 0

Right motor stopped for 1 sec

set pwm pin 6 output as 0

wait 1 secs

For the right motor, a value of 0 on pin 5 and a value of more than 100 on pin 6 will make the motor go backwards.

Set your sketch up to make the right motor go backwards for a second and stop for a second.

5.4 MAKE THE LEFT MOTOR GO FORWARD SLOWLY



mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Not saved

nano left motor

Scripts Costumes Sounds

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 12

timer

reset timer

Arduino Program

set pwm pin 9 output as 255

activate left motor

forever

set pwm pin 10 output as 0

set pwm pin 11 output as 150

Left motor forward for 1 sec

wait 1 secs

set pwm pin 10 output as 0

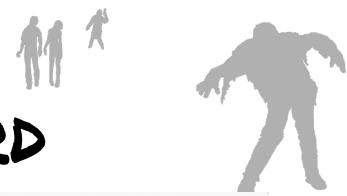
set pwm pin 11 output as 0

Left motor stopped for 1 sec

wait 1 secs

Now repeat the process with the left motor. The left motor is controlled by the pins 9, 10, & 11. Change the pwm pins so your sketch now controls the left motor.

Challenge: can you remember how to speed it up and slow it down? Can you remember how to make it go backwards?



5.5 PUTTING IT TOGETHER: MOVING FORWARD

mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Not saved

nano simple go

Scripts Costumes Sounds

motion Looks Sound Pen Data&Blocks Events Control Sensing Operators Robots

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 12

timer

reset timer

Arduino Program

set pwm pin 4 output as 255

set pwm pin 9 output as 255

forever

set pwm pin 5 output as 150

set pwm pin 6 output as 0

set pwm pin 10 output as 0

set pwm pin 11 output as 150

wait 1 secs

set pwm pin 5 output as 0

set pwm pin 6 output as 0

set pwm pin 10 output as 0

set pwm pin 11 output as 0

wait 1 secs

Put everything you've learnt together to turn both

Challenge: can you make your robot move forward for 5 secs, turn around (using just the left or right motor) then go forward again for 5 secs?

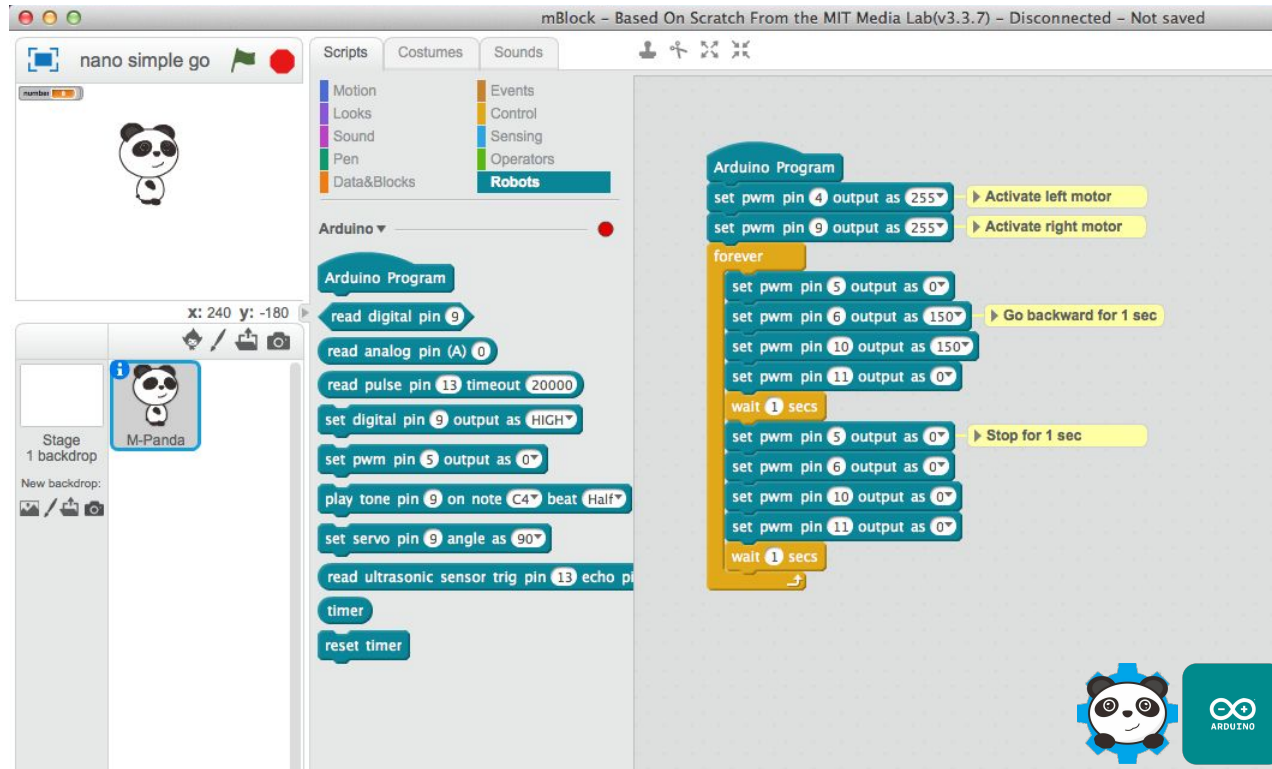
M-Panda

Stage 1 backdrop

New backdrop:

Arduino

5.6 PUTTING IT TOGETHER: MOVING BACKWARD



mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Not saved

Scripts Costumes Sounds

motion Looks Sound Pen Data&Blocks Events Control Sensing Operators Robots

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 12

timer

reset timer

Arduino Program

set pwm pin 4 output as 255

set pwm pin 9 output as 255

forever

set pwm pin 5 output as 0

set pwm pin 6 output as 150

set pwm pin 10 output as 150

set pwm pin 11 output as 0

wait 1 secs

set pwm pin 5 output as 0

set pwm pin 6 output as 0

set pwm pin 10 output as 0

set pwm pin 11 output as 0

wait 1 secs

Stage 1 backdrop

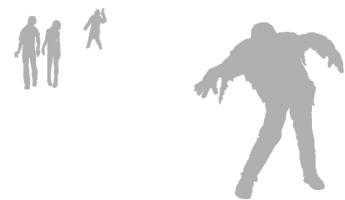
New backdrop:

M-Panda

CC BY NC

Build this sketch to make your robot go backwards for a second, then stop for a second.

Challenge: can you make your robot go forward for a second, stop for a second, go backward for a second, then stop for a second?



CHAPTER 6

OBJECT-AVOIDING ROBOT



6.1 IF-THEN STATEMENTS

mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Not saved

Scripts Costumes Sounds

motion Looks Sound Pen Data&Blocks Events Control Sensing Operators Robots

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 13

timer

reset timer

Arduino Program

forever

set distance to read ultrasonic sensor trig pin 2 echo pin 3

if distance < 10 then If there's something ahead...

set digital pin 13 output as HIGH Turn the LED on, else

else

set digital pin 9 output as LOW turn the LED off

wait 1 secs

Remember variables from Chapter Two? We're going to use a variable to store the value returned by the ultrasonic sensor. The number will either be high (object is a long way away) or low (the object is close).

From the Data&Blocks tab, make a variable called distance and bring in the blocks to make this sketch.



6.1 IF-THEN STATEMENTS

mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Not saved

Scripts Costumes Sounds

motion Looks Sound Pen Data&Blocks Events Control Sensing Operators Robots

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 3

timer

reset timer

Arduino Program

forever

set distance to read ultrasonic sensor trig pin 2 echo pin 3

if distance < 10 then If there's something ahead...

set digital pin 13 output as HIGH Turn the LED on, else

else

set digital pin 9 output as LOW turn the LED off

wait 1 secs

If-then statements are often used in programming to make things happen: IF a button is pushed THEN do something.

Set up a sketch like this to test the distance between your robot's ultrasonic sensor and any object. IF the object is close THEN the LED will turn on, ELSE the LED will turn off.

6.2 OBJECT-AVOIDING ROBOT

mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Not saved

Scripts Costumes Sounds

motion Looks Sound Pen Data&Blocks Events Control Sensing Operators Robots

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 13

timer

reset timer

Arduino Program

forever

set distance to read ultrasonic sensor trig pin 2 echo pin 3

set pwm pin 4 output as 255 Turn right motor on

set pwm pin 10 output as 255 Turn left motor on

if distance < 10 then If there's something ahead...

set number to pick random 1 to 10

if number < 6 then ...randomly choose to

set pwm pin 5 output as 150 ...turn right

set pwm pin 6 output as 0

set pwm pin 10 output as 0

set pwm pin 11 output as 150

wait 0.5 secs

else

set pwm pin 5 output as 0 ...or turn left

set pwm pin 6 output as 150

set pwm pin 10 output as 150

set pwm pin 11 output as 0

wait 0.5 secs

else

set pwm pin 5 output as 150 Otherwise go forward

set pwm pin 6 output as 0

set pwm pin 10 output as 0

set pwm pin 11 output as 150

wait 0.1 secs

Stage 1 backdrop

M-Panda

New backdrop:

CC BY NC

Build this sketch to make an object-avoiding robot.

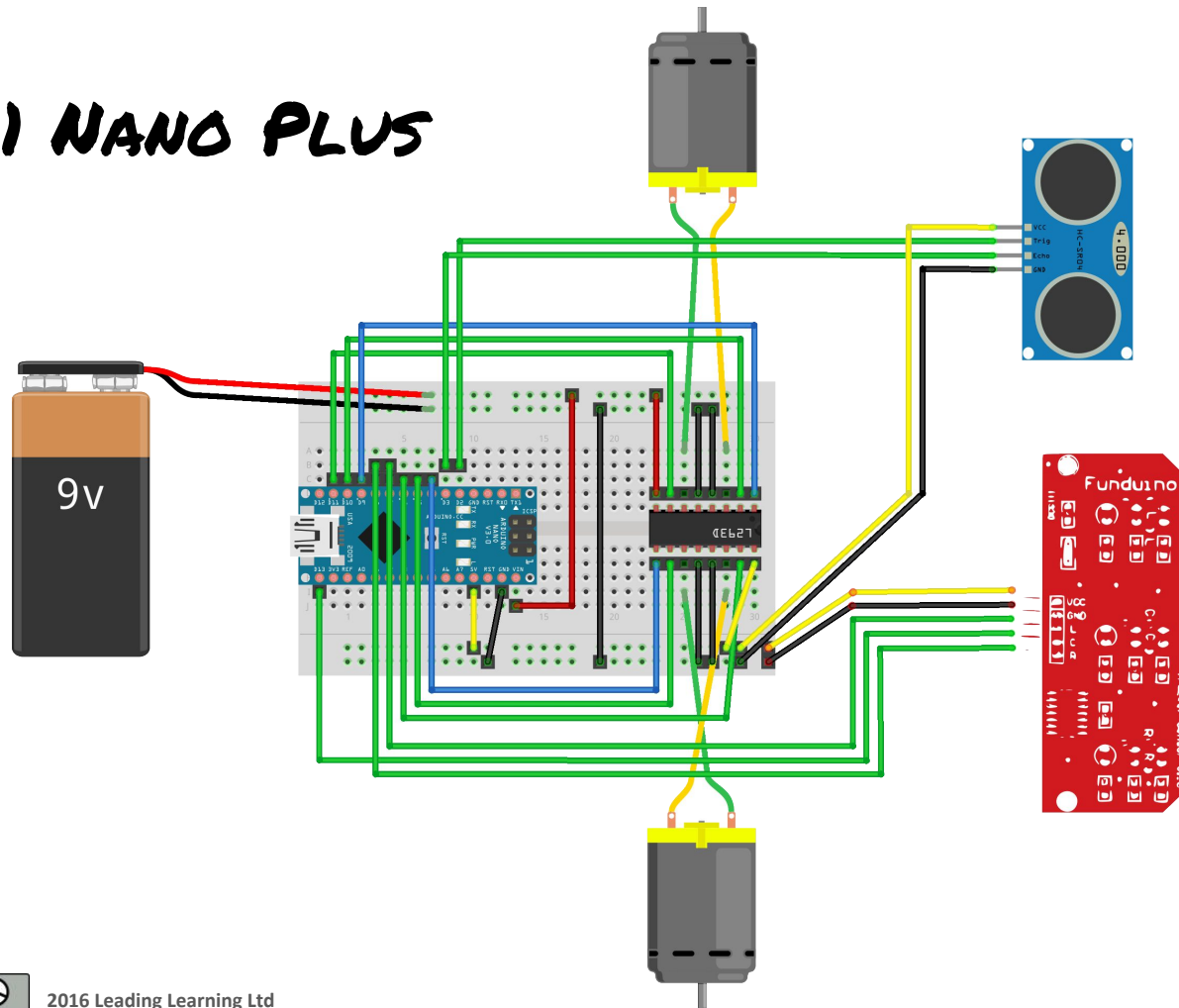
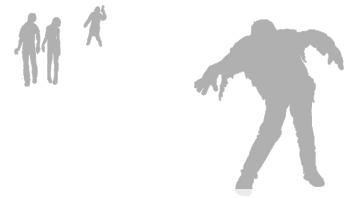
If an object (or wall) comes within a distance of 10, the robot chooses a random number to decide if it will turn left or right for half a second before going forward again.



CHAPTER 7

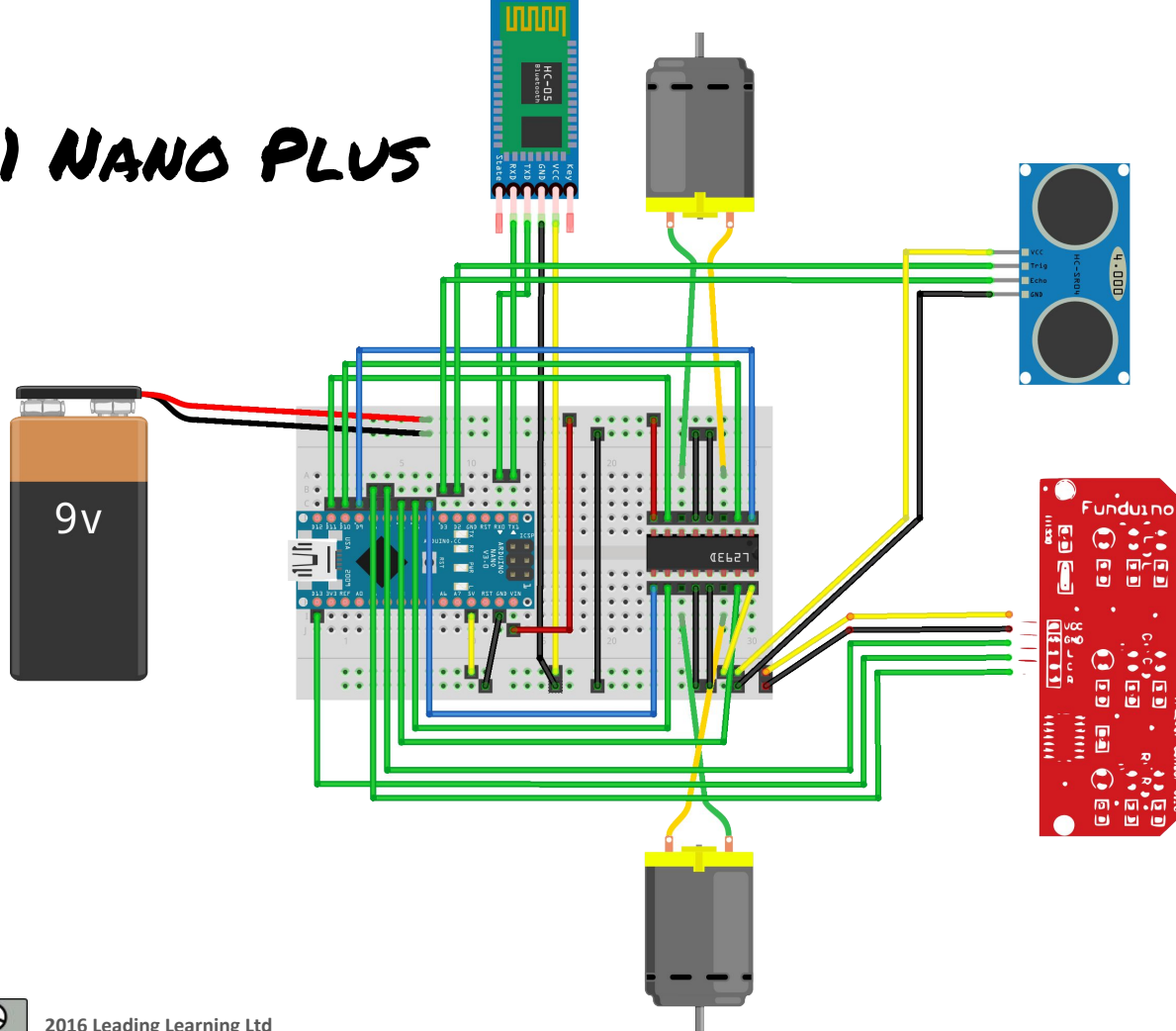
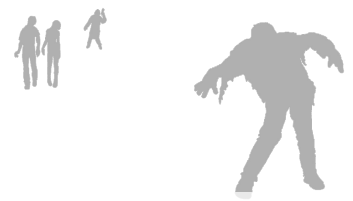
THE NANO PLUS

7.1 NANO PLUS



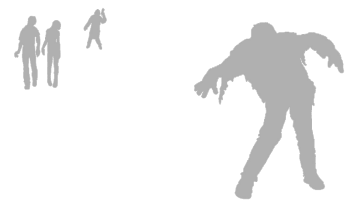
The Nano Plus has an infrared sensor (for detecting colours) and a Bluetooth module (for receiving instructions from an Android Smartphone). Attach the infrared sensor to the bottom of the robot chassis using the screws provided, feed the jumpers through the hole, and connect the jumpers according to this diagram.

7.1 NANO PLUS

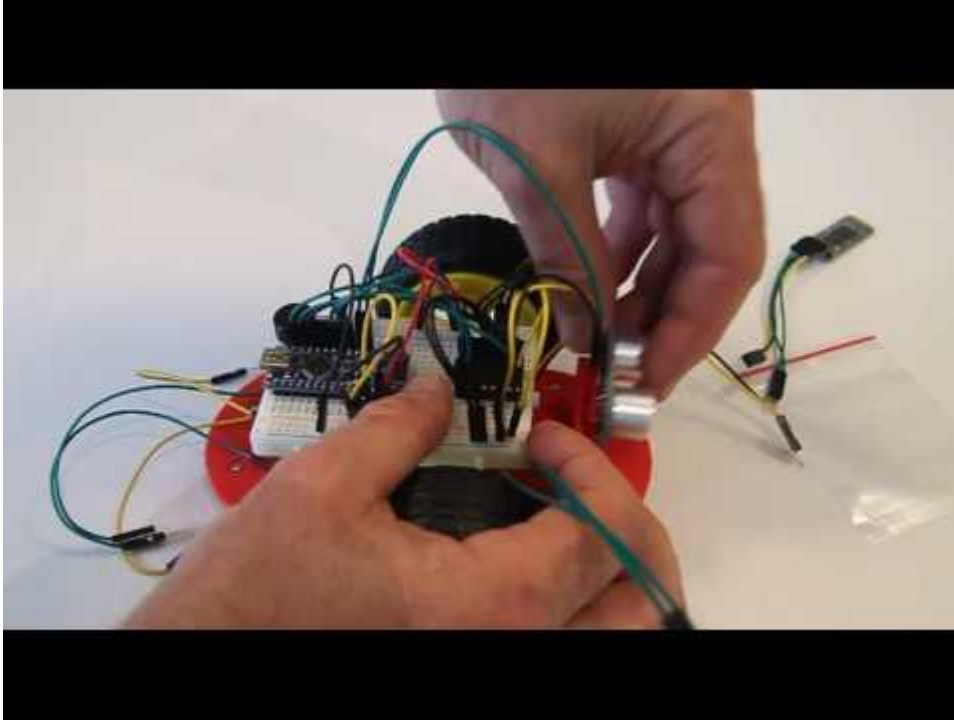


The Bluetooth module requires 5v power (yellow and black) and needs to be connected so the RXD pin on the Bluetooth module is plugged into B16 and the TXD pin is plugged into B17.

This enables a smartphone to talk to our Arduino.



7.2 NANO PLUS ASSEMBLY VIDEO



This video might help you to assemble all of the parts of the Nano Plus.

Useful Zombie fact:
you can't reason with
zombies. Don't even try
to talk to them - just run.



7.3 LINE-FOLLOWING ROBOT

mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Disconnected - Saved

nano line follow

number 1

M-Panda

Stage
1 backdrop

New backdrop:

Scripts

Costumes

Sounds

Robots

Arduino Program

read digital pin 9

read analog pin (A) 0

read pulse pin 13 timeout 20000

set digital pin 9 output as HIGH

set pwm pin 5 output as 0

play tone pin 9 on note C4 beat Half

set servo pin 9 angle as 90

read ultrasonic sensor trig pin 13 echo pin 12

timer

reset timer

Arduino Program

forever

if read digital pin 7 = 0 and read digital pin 8 = 1 then If the right sensor sees black...

set pwm pin 4 output as 255 Turn slight right

set pwm pin 9 output as 255

set pwm pin 5 output as 70

set pwm pin 6 output as 0

set pwm pin 10 output as 0

set pwm pin 11 output as 60

wait 0.1 secs

else

if read digital pin 7 = 1 and read digital pin 8 = 0 then If the left sensor sees black...

set pwm pin 4 output as 255 Turn slight left

set pwm pin 9 output as 255

set pwm pin 5 output as 60

set pwm pin 6 output as 0

set pwm pin 10 output as 0

set pwm pin 11 output as 70

wait 0.1 secs

else

if read digital pin 7 = 0 and read digital pin 13 = 1 and read digital pin 8 = 0 then If the centre sensor sees black...

set pwm pin 4 output as 255 Go forward slowly

set pwm pin 9 output as 255

set pwm pin 5 output as 70

set pwm pin 6 output as 0

set pwm pin 10 output as 0

set pwm pin 11 output as 70

wait 0.1 secs

This sketch will set your robot up to follow a thick black line. The infrared sensor under your Nano has 3 'eyes' (Pins 7, 13 & 8) Pin 13 should always read 1 (see black) If pin 7 reads 1 (sees black



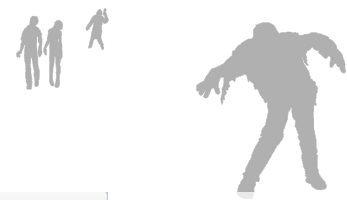
7.3 LINE-FOLLOWING ROBOT

The image shows a Scratch window titled "mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Serial Port Connected - Not saved". The project is named "nano line follow" and features a panda character. The "Scripts" tab is selected, showing an "Arduino Program" block. The "Robots" category in the left sidebar is highlighted. The main workspace contains the following code:

```
Arduino Program
forever
  if then -> If the right sensor sees black...
else
  if then -> If the left sensor sees black...
else
  if then -> If the centre sensor sees black...
  read digital pin 7 = 0
  read digital pin 8 = 1
  and
  read digital pin 7 = 1
  read digital pin 8 = 0
  and
  read digital pin 7 = 0
  read digital pin 13 = 1
  read digital pin 8 = 0
  and
  and
```

The code is a "forever" loop with three "if" statements. The first "if" statement checks "If the right sensor sees black...". The second "if" statement checks "If the left sensor sees black...". The third "if" statement checks "If the centre sensor sees black...". Below the "if" statements, there are several "read digital pin" blocks and "and" blocks. The "read digital pin" blocks are: "read digital pin 7 = 0", "read digital pin 8 = 1", "read digital pin 7 = 1", "read digital pin 8 = 0", "read digital pin 7 = 0", "read digital pin 13 = 1", and "read digital pin 8 = 0". The "and" blocks are: "and", "and", "and", "and", and "and".

Build this set of if/then statements. Note that the bottom one is actually one inside another (look on the next page to see how it should look when finished).



7.3 LINE-FOLLOWING ROBOT

mBlock – Based On Scratch From the MIT Media Lab(v3.3.7) – Disconnected – Not saved

Scripts Costumes Sounds

Motion

- Looks
- Sound
- Pen
- Data&Blocks

Events

- Control
- Sensing
- Operators
- Robots

move 10 steps

turn 15 degrees

turn 15 degrees

point in direction 90°

point towards

go to x: -21 y: 20

go to mouse-pointer

glide 1 secs to x: -21 y: 20

change x by 10

set x to 0

change y by 10

set y to 0

Arduino Program

```
forever
  if read digital pin 7 = 0 and read digital pin 8 = 1 then
    If the right sensor sees black...
  else
    if read digital pin 7 = 1 and read digital pin 8 = 0 then
      If the left sensor sees black...
    else
      if read digital pin 7 = 0 and read digital pin 13 = 1 and read digital pin 8 = 0 then
        If the centre sensor sees black...
```

Your if-then statements should look like this.

x: -21
y: 20

7.4 LINE-FOLLOWING ROBOT



mBlock - Based On Scratch From the MIT Media Lab(v3.3.7) - Serial Port Connected - Not saved

Scripts | Costumes | Sounds

Arduino ▾

Arduino Program

- read digital pin 9
- read analog pin (A) 0
- read pulse pin 13 timeout 20000
- set digital pin 9 output as HIGH
- set pwm pin 5 output as 0
- play tone pin 9 on note C4 beat Half
- set servo pin 6 angle as 90
- read ultrasonic sensor trig pin 13 echo pin 12
- timer
- reset timer

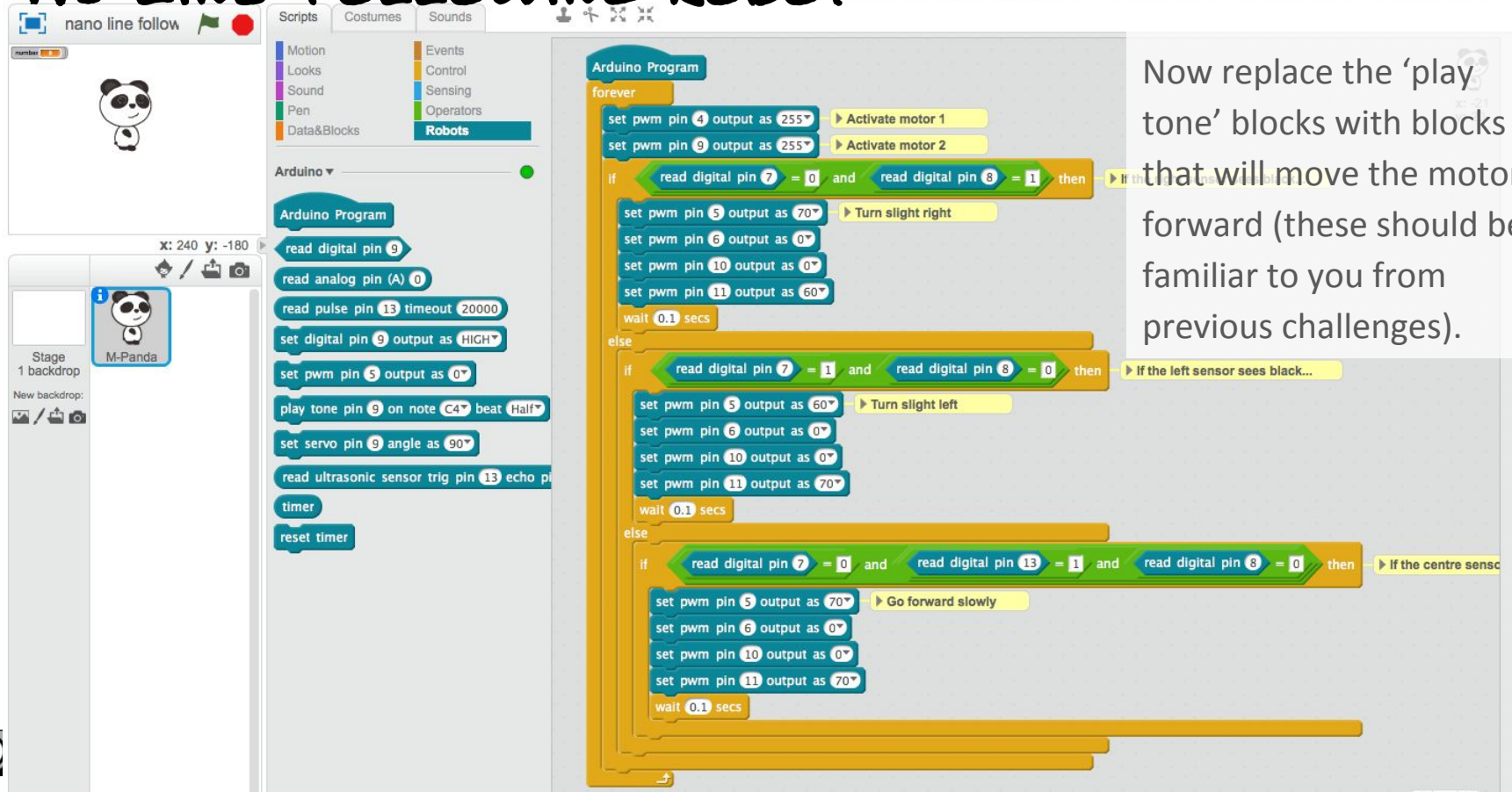
Arduino Program

forever

- if read digital pin 7 = 0 and read digital pin 8 = 1 then If the right sensor sees black...
 - play tone pin 12 on note D8 beat Half
- else if read digital pin 7 = 1 and read digital pin 8 = 0 then If the left sensor sees black...
 - play tone pin 12 on note A2 beat Half
- else if read digital pin 7 = 0 and read digital pin 13 = 1 and read digital pin 8 = 0 then If the centre sensor sees black...
 - play tone pin 12 on note C5 beat Half
- wait 0.1 secs

Build this sketch to test whether your IR sensors are working properly. If the right sensor (8) sees the colour black a high tone will sound, if the left sensor sees the colour black (7) a low tone will sound. The centre sensor (13) will play a mid-range tone.

7.5 LINE-FOLLOWING ROBOT



nano line follow

Scripts

Costumes

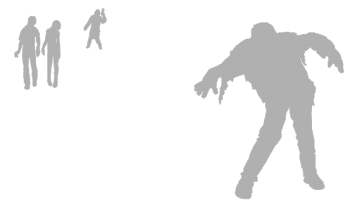
Sounds

Arduino Program

forever

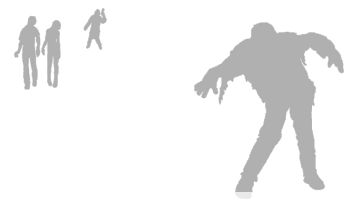
- set pwm pin 4 output as 255° ▶ Activate motor 1
- set pwm pin 9 output as 255° ▶ Activate motor 2
- if read digital pin 7 = 0 and read digital pin 8 = 1 then ▶ If the left sensor sees black...
- set pwm pin 5 output as 70° ▶ Turn slight right
- set pwm pin 6 output as 0°
- set pwm pin 10 output as 0°
- set pwm pin 11 output as 60°
- wait 0.1 secs
- else
- if read digital pin 7 = 1 and read digital pin 8 = 0 then ▶ If the left sensor sees black...
- set pwm pin 5 output as 60° ▶ Turn slight left
- set pwm pin 6 output as 0°
- set pwm pin 10 output as 0°
- set pwm pin 11 output as 70°
- wait 0.1 secs
- else
- if read digital pin 7 = 0 and read digital pin 13 = 1 and read digital pin 8 = 0 then ▶ If the centre sensor...
- set pwm pin 5 output as 70° ▶ Go forward slowly
- set pwm pin 6 output as 0°
- set pwm pin 10 output as 0°
- set pwm pin 11 output as 70°
- wait 0.1 secs
- end if
- end else
- end if
- end else
- end forever

Now replace the 'play tone' blocks with blocks that will move the motors forward (these should be familiar to you from previous challenges).

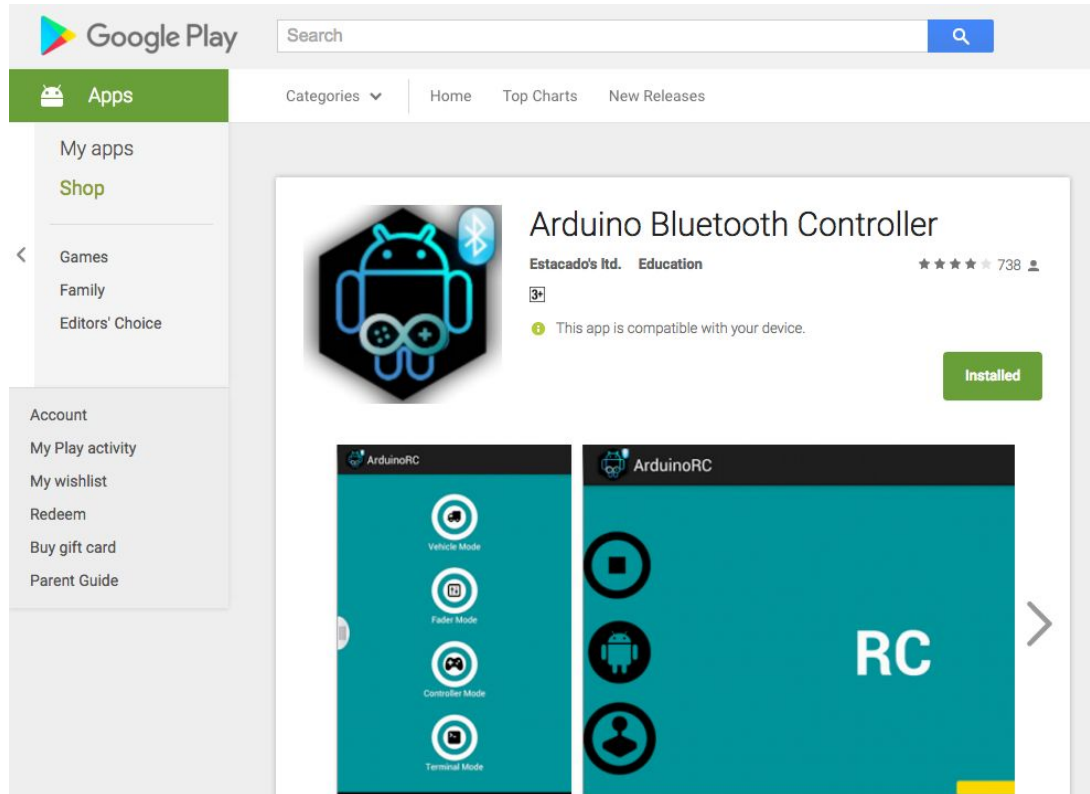


CHAPTER 8

SMARTPHONE-CONTROLLED ROBOT



8.1 INSTALL THE ARDUINO BLUETOOTH APP



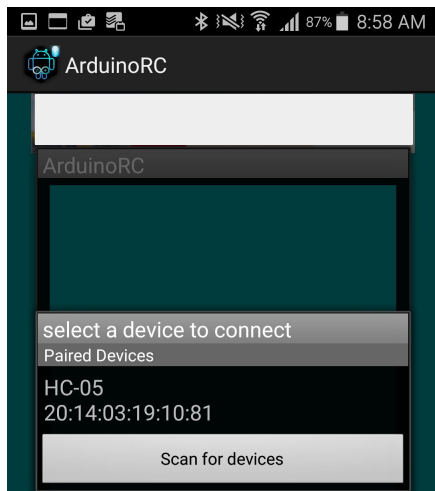
Install the *Arduino Bluetooth Controller* app from the Google Play store.

We're going to use this to control our robot.

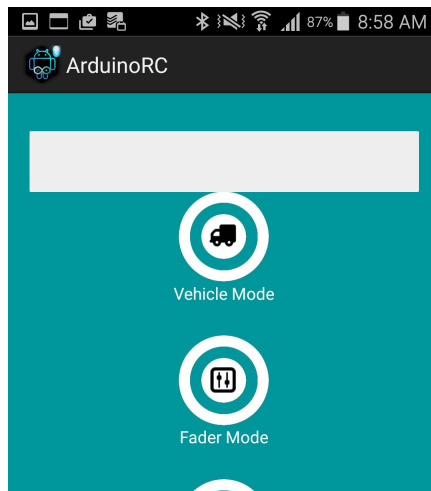
Unfortunately there isn't an app for iOS that we can use.



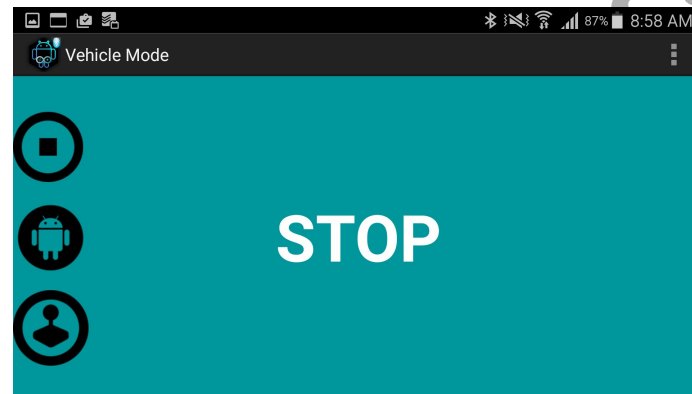
8.2 SET UP THE BLUETOOTH APP



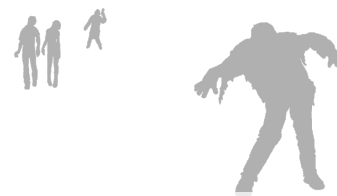
Open the app, confirm your UUID and scan for devices. Look for one called HC-05. Use 1234 for a pairing code if it asks for one. Ensure your robot is powered on.



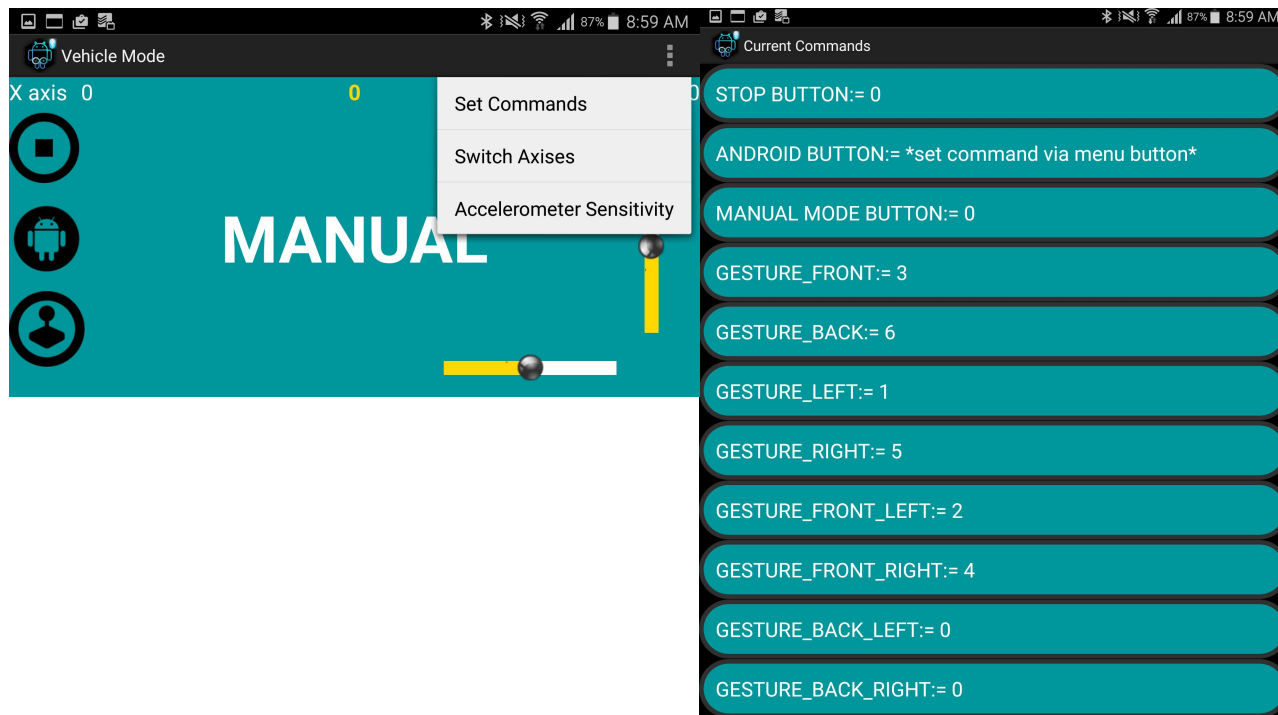
Once your phone is paired with and connected to your Bluetooth module, choose 'Vehicle Mode'.



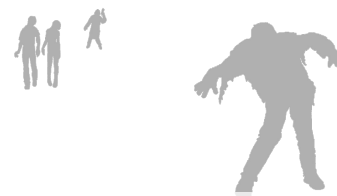
When you see this screen, choose the joystick option (bottom).



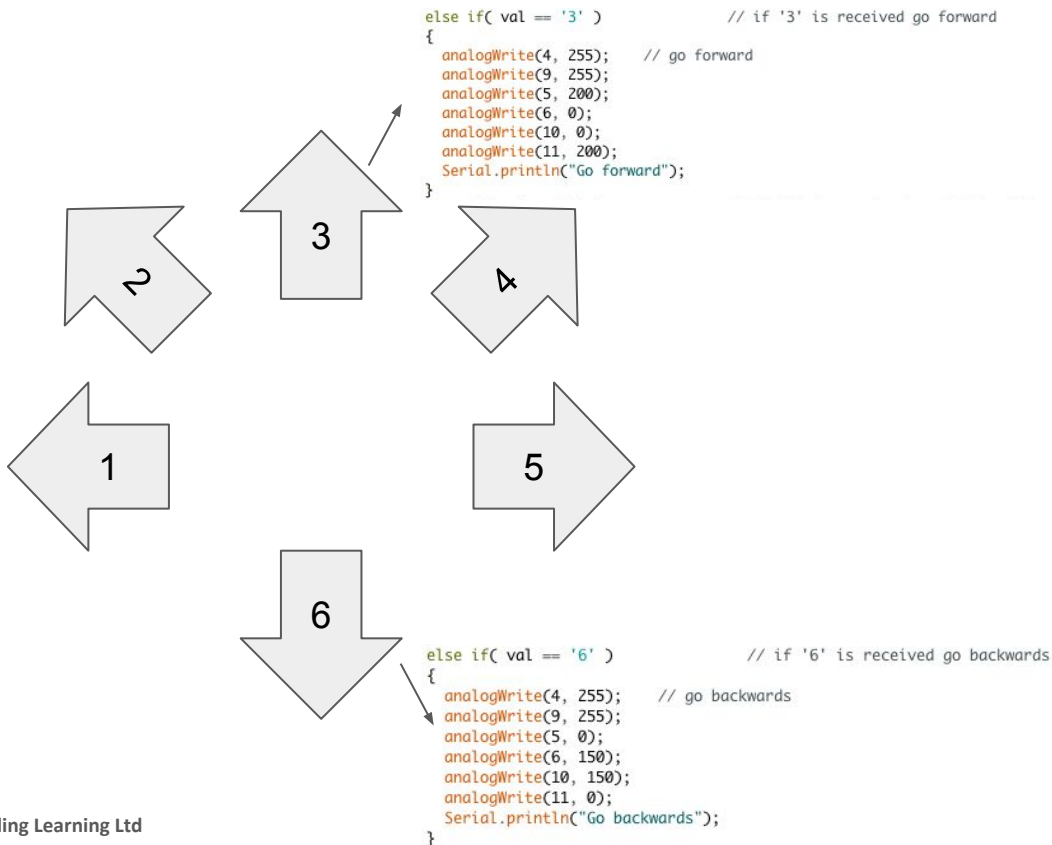
8.2 SET UP THE BLUETOOTH APP



From the menu on the top right, choose 'Set Commands'. Go through the process of setting up each of the commands so they have these numbers attached to them (GESTURE_FRONT should be 3, and so on.)



8.2 SET UP THE BLUETOOTH APP



We're going to send these numbers to our robot and make it move in a certain direction. When it gets a '3' from our phone, it'll go forward; when it gets a '6' it will go backwards, and so on.

You might recognise this code from previous challenges.



8.3 UPLOAD THE SKETCH

```
Nano-Bluetooth-v1 | Arduino 1.6.8

Nano-Bluetooth-v1

/*
simple Bluetooth App sketch
*/

char val;          // variable to receive data from the serial port

void setup()
{
  pinMode(4, OUTPUT); // pin 4 as OUTPUT
  pinMode(5, OUTPUT); // pin 5 as OUTPUT
  pinMode(6, OUTPUT); // pin 6 as OUTPUT
  pinMode(9, OUTPUT); // pin 9 as OUTPUT
  pinMode(10, OUTPUT); // pin 10 as OUTPUT
  pinMode(11, OUTPUT); // pin 11 as OUTPUT

  Serial.begin(9600); // start serial communication at 9600bps
}

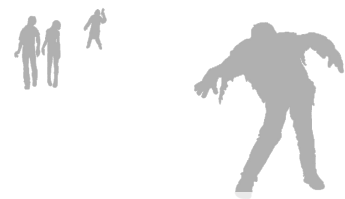
void loop() {
  if( Serial.available() ) // if data is available to read
  {
    ;
  }
  val = Serial.read(); // read it and store it in 'val'

  if( val == '0' ) // if '0' is received stop
  {
    analogWrite(4, 0); // Stop
    analogWrite(5, 0);
    analogWrite(6, 0);
    analogWrite(9, 0);
    analogWrite(10, 0);
    analogWrite(11, 0);
  }
}
```

We're going to use the Arduino software (rather than mBlock) to programme our robot for Smartphone control. You should already have it installed on your computer, but if you don't, you can get it from <https://www.arduino.cc/>



Download this sketch, open it in Arduino and upload to your Nano.



8.4 CONTROL YOUR ZOMBIEBOT



If you've uploaded the sketch and set up your smartphone correctly, you should be able to do this with your smartphone.

Accelerometers are small devices in phones that detect changes in the way you are holding your phone. We're using them to tell your Zombiebot which direction to move.

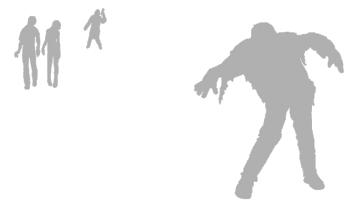


IMAGE CREDITS:

https://commons.wikimedia.org/wiki/File:Emoji_u1f45f.svg

https://commons.wikimedia.org/wiki/File:Baseball_bat.svg

https://commons.wikimedia.org/wiki/File:Sports_icon.png

https://commons.wikimedia.org/wiki/File:Sport_balls.svg

https://upload.wikimedia.org/wikipedia/commons/e/ef/Hockey_Stick_and_Puck.png

https://upload.wikimedia.org/wikipedia/commons/4/4a/Nike_Free%2B_3_running_shoe.jpg

<https://pixabay.com/en/tennis-shoes-running-shoes-shoes-297150/>

https://pixabay.com/static/uploads/photo/2013/07/12/14/09/emergency-doctor-147857_640.png