**Do Robotics with QuadBot!**



**Outcome**

Control a four-legged robot by making it light up and walk with basic coding.

**You will learn**

* Basic coding with Arduino.
* How to control a robot.

**Time**

These activities will last 45 minutes.

**Activity 1 - LEDs and Colour**

LEDs (Light Emitting Diodes) are often used in electronic circuits to generate light.



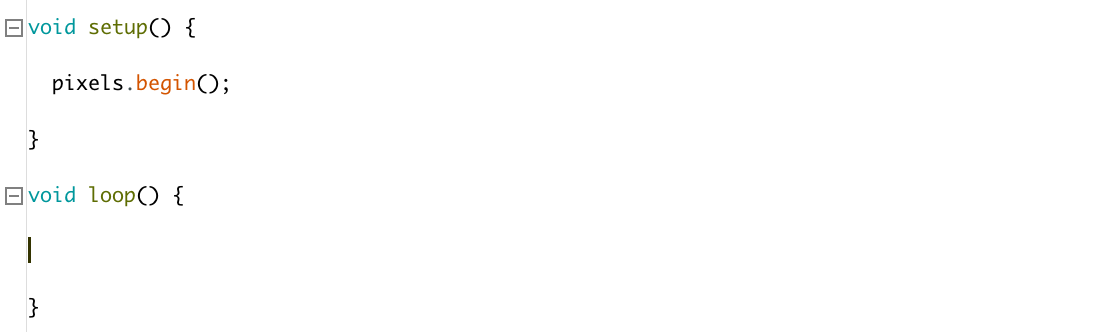
A single LED can only generate a single colour of light. This can be quite limiting as it means you need to use a single LED to achieve each colour you want.

To get around this you can use an RGB LED. An RGB LED looks like a normal LED, but it is in fact three LEDs combined into a single package.

It uses a red, green and blue (RGB) LED inside, and by combining these three colours at different brightness, you can achieve any colour.

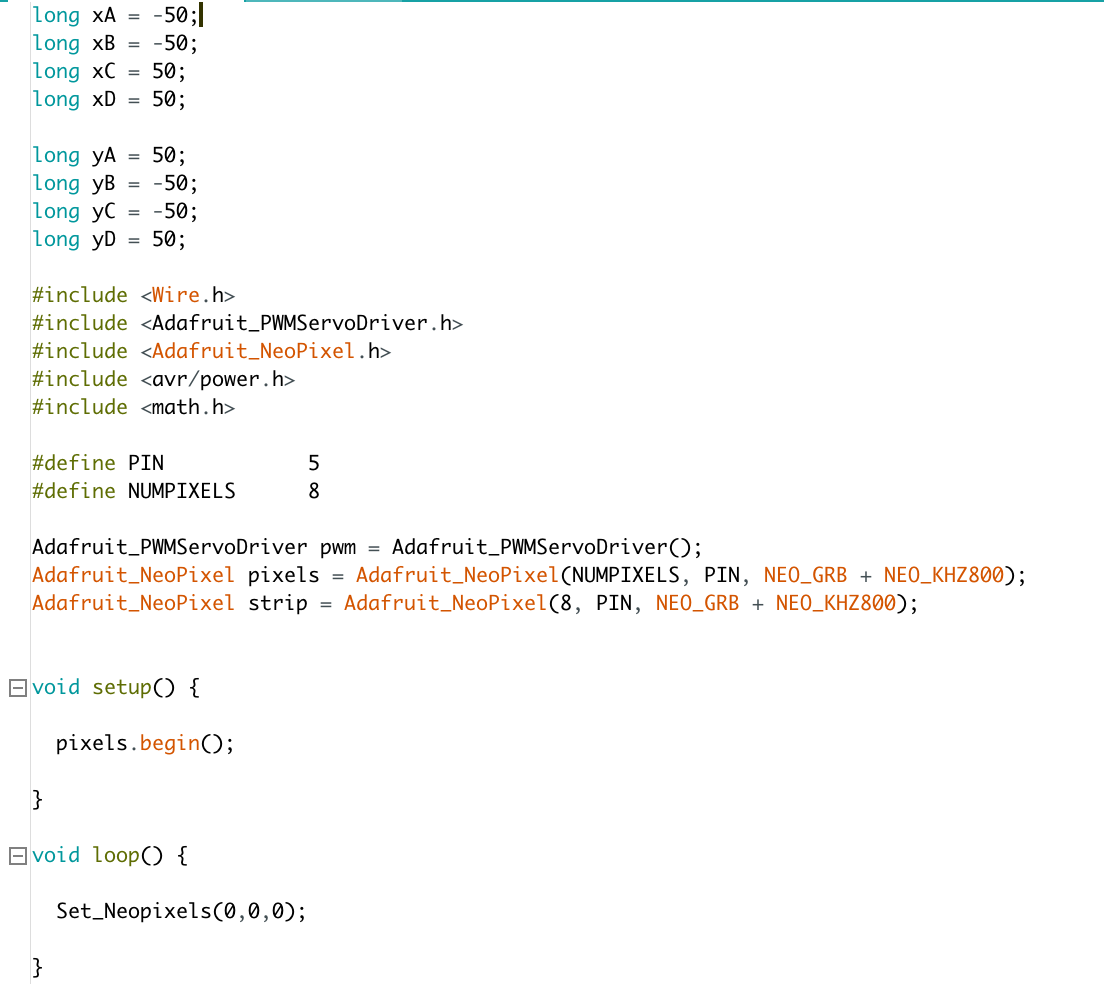
The QuadBoard has eight RGB LEDs onboard (the white squares on the outside of the board).

**Program the QuadBot LEDs**

Open Activity\_1\_LEDs and find the part of the code below (ignore the code at the top, this is used to initialise the QuadBot is not important to us yet). 

Void setup() will run any code between the {} once, at the start of the program. Void loop() will run any code between the {} forever, until the robot is turned off.

pixels.begin() runs once at the start of the program, and tells the robot to get ready to control the LEDs (the LEDs are called Pixels or Neopixels).

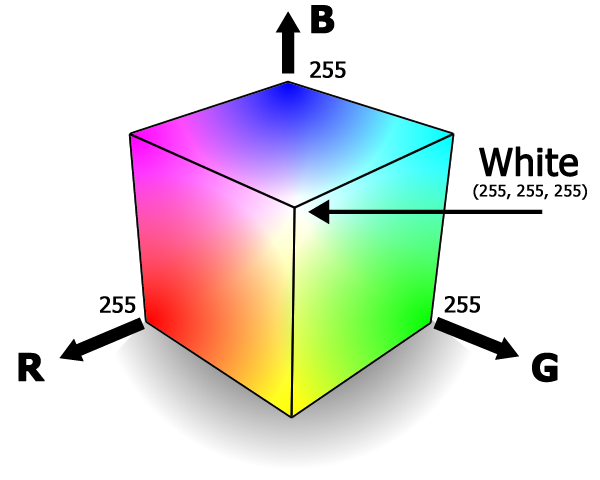
In void loop() type the code as below. 

Set\_Neopixels is a function that will set the colour of all the LEDs (Neopixels) on the robot.

What is your favourite colour? Let’s make the LEDs that same colour. You will need to pick the correct combination of Red, Green and Blue, and put these numbers into the code like this...

Set\_Neopixels(Red, Green, Blue)

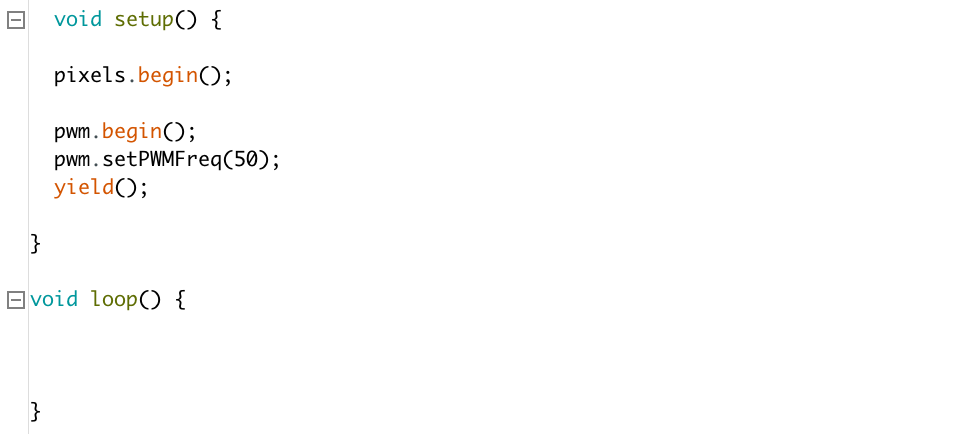
Use the colour cube below to choose the right numbers.



Now try the code below.

**Activity 2 - Motors and Movement**

There are twelve motors on the QuadBot, let’s learn how to make them move, open Activity\_2\_Motors. Ignore the lines of code at the top, focus on void setup() and void loop().



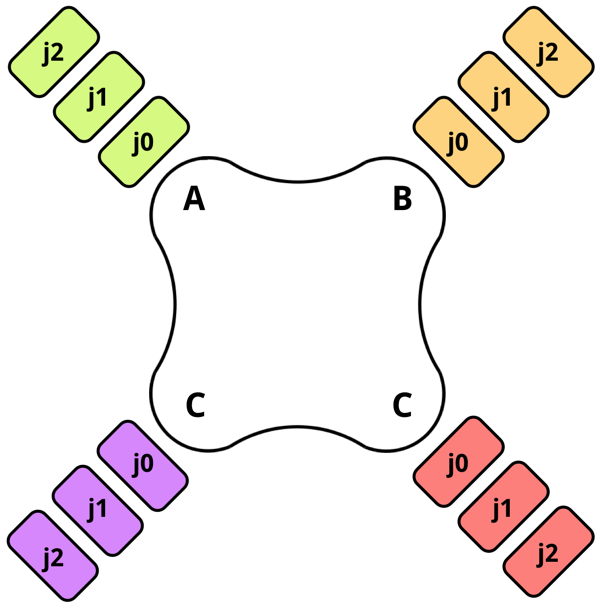
The code in void setup() will get the robot ready to control the LEDs **and** the motors. In void loop() write the following code.

What happens? You should notice that one of the motors inside a robot leg has moved. How? Well let’s think about the motors on the robot. It turns out it is very useful to name them, so we can easily know which leg to control.

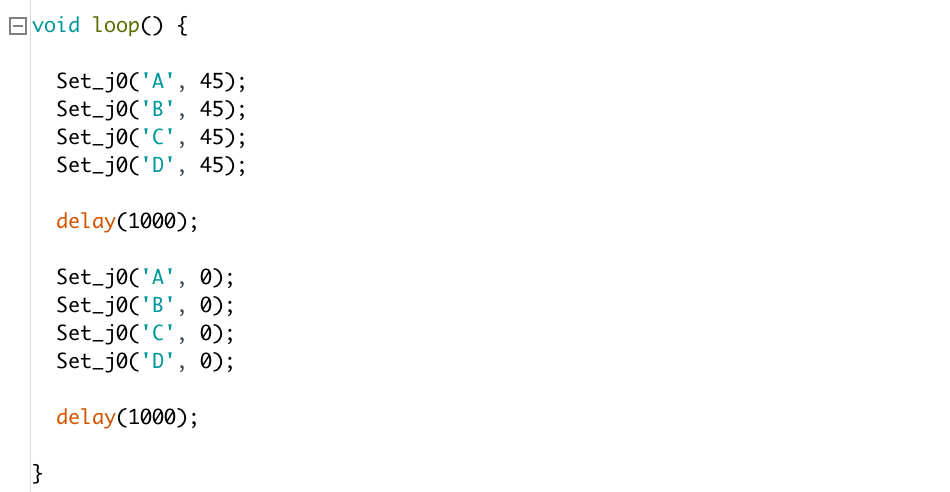
Since the robot has four legs, we name each leg **A**, **B**, **C** and **D**. And since each leg has three motors each (one for each joint) we can name each joint **j0**, **j1**, and **j2**. This means that we can name a motor **A-j0** for example, and know exactly where it is on the robot.

So, Set\_j0(‘A’, 0) will move motor **A-j0**. In this case, it will move motor **A-j0** to an angle of 0°, because the second value between the () is 0. Try another angle like 45° and send the code to the QuadBot.

Here is a diagram to help you remember where each motor is.



Now try this code…

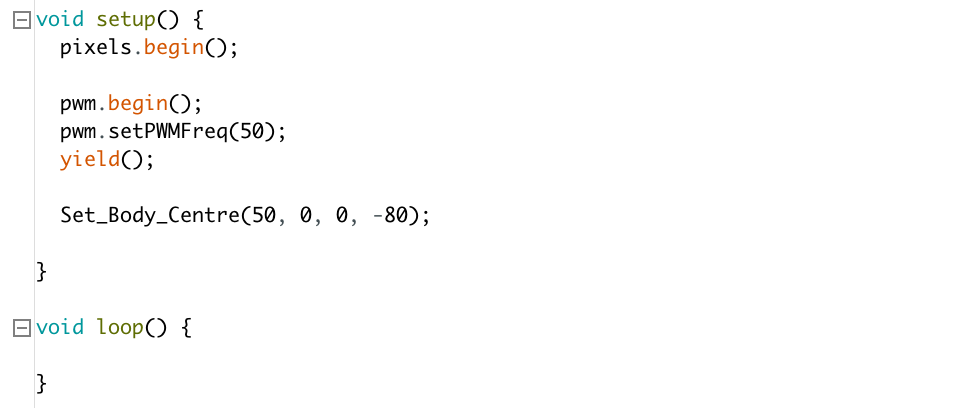


What happened? Notice how the delay(1000) adds a delay between the movement of all the motors. Now try programming other motors.

By programming the motors to move to different positions, at different times we can make the robot move in a certain way.

**Activity 3 - Legs and Walking**

Moving motors is fun, but let’s make this robot walk! Open Activity\_3\_Walking.



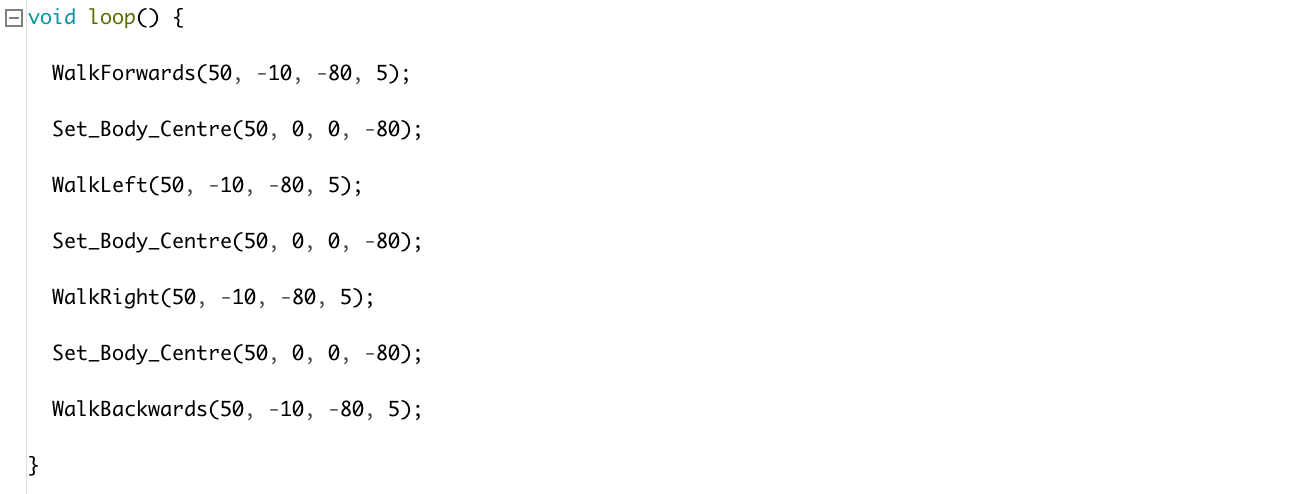
The code Set\_Body\_Centre(50, 0, 0, -80) will set the centre position of the robot. Try playing with the numbers as below and see if you can understand what each number does.

1. Set\_Body\_Centre(50, 15, 0 -80)
2. Set\_Body\_Centre(50, 30, 0, -80)
3. Set\_Body\_Centre(50, 0, 15, -80)
4. Set\_Body\_Centre(50, 0, 30, -80)
5. Set\_Body\_Centre(50, 0, 0, 0)

You should notice that changing these values will shift the body in different directions. Now write the code below.

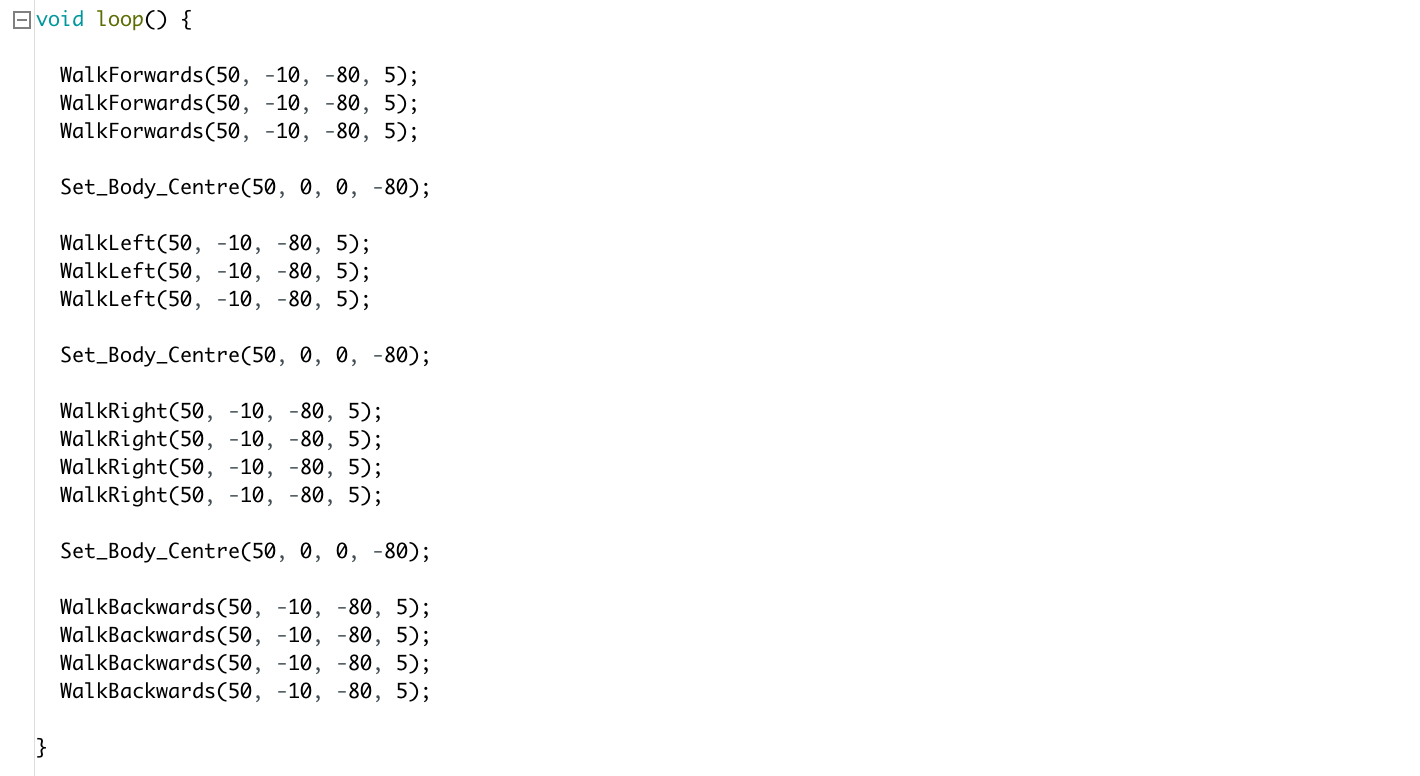


If you did it right, the robot should now be walking. Great job!

To make it walk in different directions, try the code below…

Notice how we use Set\_Body\_Centre() to reset the position of QuadBot before it changes direction.

That’s great, but as you can see, Set\_Body\_Centre() only makes the robot take four steps. To take more steps let’s add in more of each walk function.



Good job!

**Activity 4 - Extra**

Can you combine the LED control with the walking to make QuadBot light a different colour for each walking direction?