

A - Save the Astronaut!

You are the chief programmer for a mission to Mars that has crash landed, and one of your crew is stranded on the other side of the planet!



INTRODUCTION

You are the chief programmer for a mission to Mars that has crash landed, and one of your crew is stranded on the other side of the planet! Let's learn how to make our robot move so we can save them.

Step 1

The Main Board



- Let's assemble our robot so we can get started.
- We need to assemble the main board first this is the brain of the robot, and controls everything it does!
- So we can use it without the cable plugged in, turn the board over and insert the batteries. Make sure they are the right way around.
- Next, plug in the micro:bit to the long black connector - make sure it is the same way round as the picture, with the LEDs and buttons facing forwards.



Step 2

Assemble your robot

- Next, plug the main board and motors into the baseboard, just like the picture.
- Use the two yellow/white cables to connect the two motors to the main board.

Make sure you plug the left motor into the left socket (M1) and the right motor into M2!



Step 3

Open the Editor

- Open the editor which you downloaded in the Getting Started section - you can download it again here (https://drive.google.com/open? id=1hzofVGb9QVIYNQk_N9QCzvCXswRNr0il) if you need to.
- Paste the Invent! starter code into the top of the program - again, you can get it <u>here</u> (<u>https://drive.google.com/file/d/1VgLKH6Qhb47pchbd6QYZijylrgz9yU</u> mo/view?usp=sharing) if you need to.
- **Plug in** your robot with the USB cable.
- Make sure your editor looks like the picture, and we are ready to start programming.
- Don't forget make sure you have this code in **every program you make**, otherwise the motors (and some other modules we get to later) won't work properly.

Dow	hload Save
1	# Invent! Code Start
2	<pre>from microbit import *;pin14.set analog period(10)</pre>
3	<pre>def drive motor(m,s):#time,value,r min,l min,r may</pre>
4	if(m==1 or m==0):pin14.write analog(1023 if al
5	if(m==2 or m==0):pin16.write_analog(1023 if al
6	<pre>def calibrate line sensors():</pre>
7	global p;p=[running time(),0,1023,1023,0,0,0,0
8	while(running time()-p[0]<4000):v=[p0(),p1()]
9	p[6]=(p[5]+p[3])/2;p[7]=(p[4]+p[2])/2;p[8]=p[1]
10	while(p0()>p[7]and p1()>p[6]):d(0 if p0() <p[7]< th=""></p[7]<>
11	def digital read line(s): return 1 if (s==0 and p
12	<pre>def analog read line(s): v=p0()if s==0 else p1();</pre>
13	# Invent! Code End
14	# Start your code below here!
15	

Driving Forwards

- Now we're all setup, let's make a simple program so the robot drives forwards.
- After the **Invent! code**, add the following two lines into your program:
 - drive_motor(1,100)
 - drive_motor(2,100)
- Can you guess what the robot will do? Upload the code and find out!

Download Save Load Blockly Snippets Help
1 # Invent! Code Start
<pre>2 from microbit import *;pin14.set_analog_period(10);pin16.set_a</pre>
<pre>3 def drive_motor(m,s):#time,value,r_min,l_min,r_max,l_max,l_thr</pre>
<pre>4 if(m==1 or m==0):pin14.write_analog(1023 if abs(s)>100 els</pre>
<pre>5 if(m==2 or m==0):pin16.write_analog(1023 if abs(s)>100 els</pre>
<pre>6 def calibrate_line_sensors():</pre>
<pre>7 global p;p=[running_time(),0,1023,1023,0,0,0,0,0,0];d=driv</pre>
<pre>8 while(running_time()-p[0]<4000):v=[p0(),p1()];p[5]=v[1]if</pre>
<pre>9 p[6]=(p[5]+p[3])/2;p[7]=(p[4]+p[2])/2;p[8]=p[5]-p[3];p[9]=</pre>
<pre>10 while(p0()>p[7]and p1()>p[6]):d(0 if p0()<p[7] or="" p1()<p[6]<="" pre=""></p[7]></pre>
<pre>11 def digital_read_line(s): return 1 if (s==0 and p0()>p[7])or(s</pre>
<pre>12 def analog_read_line(s): v=p0()if s==0 else p1();return 100 if</pre>
13 # Invent! Code End
14 # Start your code below here!
15
16 drive_motor(1,100)
17 drive_motor(2,100)

Step 5

Step 4

How does it work?

- If you guessed drive forwards you are correct! Let's have a detailed look at the drive_motor line and how it works:
- drive_motor is a function (this just means that it does something), which takes 2 inputs:
 - The motor to drive this can be 1 (left motor, M1),
 2 (right motor, M2) or 0 (both motors)
 - What speed to drive it at this can be any number from -100 (full speed backwards) to 100 (full speed forwards)
- We're also going to be using another function sleep. This function just waits for however long you want it to.
- The sleep function takes just 1 input: how many milliseconds to sleep for. In programming, time is usually measured in milliseconds. There are 1000 milliseconds in 1 second.



Step 6

Stopping

- Let's change the code so the robot doesn't drive forever.
- Change your program so it looks like the picture now we are using both the drive_motor and the sleep functions.
- Can you guess what they might do?
 - sleep(1000) just makes the robot wait for 1 second (don't forget - there are 1000 milliseconds in 1 second
 - The final drive_motor line should make the motors
- **Upload and test** the program your robot should drive forwards and then stop!

Ø	wniie(running_time()-p[0]<4000):v=[
9	p[6]=(p[5]+p[3])/2;p[7]=(p[4]+p[2])
10	<pre>while(p0()>p[7]and p1()>p[6]):d(0 i</pre>
11	<pre>def digital_read_line(s): return 1 if (</pre>
12	<pre>def analog_read_line(s): v=p0()if s==0</pre>
13	# Invent! Code End
14	# Start your code below here!
15	
16	drive_motor(1,100)
17	<u>drive motor(2,100</u>)
18	sleep(1000)
19	drive_motor(0,0)

Magnets

Step 7



- To save the astronaut, we need to attach the **magnet module** to the robot so we can pickup the magnetic astronaut.
- Remove the white trackball from the front of the robot - if you have a brand new kit it might be quite hard to get out, so ask your teacher for help if it is too difficult.
- Slot the magnet module over the top of the trackball, with the magnets facing upwards like in the picture.
- Finally put the trackball back into the robot like the picture. You should now be able to pickup an astronaut!



Rescue the Astronaut!





- Now you can make your robot drive and stop, and have some magnets, you need to save the stranded astronaut.
- Write a program to make your robot drive forwards across the planet and pick up the astronaut. You will need to change the sleep time so it drives forwards the right amount!

Step 9

Step 8

Save Your Work

- After each challenge, make sure to save your work
 you might need it later!
- Try to give your programs descriptive names so you know what they do.

save_t	he_astronaut
<pre>ent! UAVS start microbil import *;pin14.set_analog_period(10);pin16.set_analog_period(10);pi rive_mot r(m,s):#time,value,r_min,l_min,r_max,l_max,l_thr,r_thr,L_range,r_t f(m==1 or m==0):pin14.write_analog(1023 if abs(s)>100 else(abs(s)*1023)/100 f(m==2 or l==0):pin16.write_analog(1023 if abs(s)>100 else(abs(s)*1023)/100</pre>	params=[0]*10;p0 <i>range</i> a).w=pin13.write a).w=pin15.write
<pre>pliprate_line sensors(): lobal p;p=[\nning_time(),0,1023,1023,0,0,0,0,0,0];d=drive_motor;d(1,100);q hile(running_time()-p[0]<4000):v=[p0(),p1()];p[5]=v[1]if v[1]>p[5]else p[5] [6]=(p[5]+p[3]\/2;p[7]=(p[4]+p[2])/2;p[8]=p[5]-p[3];p[9]=p[4]-p[2];d(1,-100) hile(p0()>p[7] nd p1()>p[6]):d(0 if p0()<p[7] 3,0)<br="" else="" or="" p1()<p[6]="">foital modul in (0; noture 1 if (s=-0 and p0())p[5])not(s and p1())p[6])else</p[7]></pre>	d(,-100);sleep(];[3]=v[1]if v[0) d(2,100);slee
<pre>lgital_read_line(s): return i if (s==0 and pe()>p())p('s and pi()>p[s])er nalog_read_line(s): v=p0()if s==0 else p1();return 100 if(v>p[4]and s)or(v: ent! Code End rt your code below here! motor(1 100)</pre>	5]and s==0)el
motor(2,100) (1900) motor(0,0)	
	Þ

Bring them Back

Step 10





- For this extension challenge, add some more lines of code to make your robot:
 - Drive forwards
 - Pickup the astronaut
 - Reverse back again

(i) See if you can work out how to change the numbers in the **drive_motor** lines to make the robot reverse - make some changes and **try it out!**