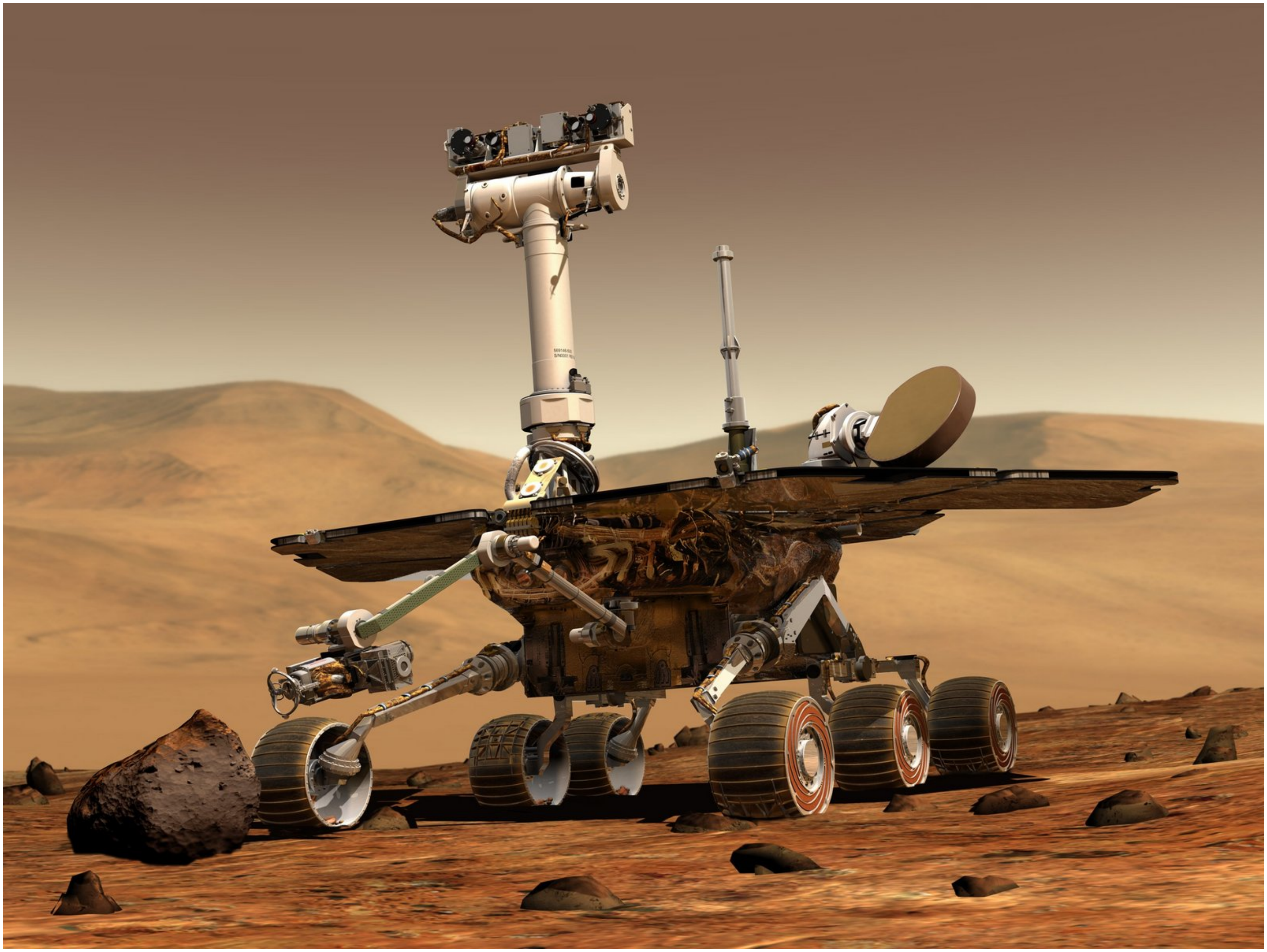
# B - Exploring the Planet

Our robot needs to do more than just move backwards and forwards if we are going to explore Mars - let's learn how to make it turn!



This document was generated on 2022-01-01 09:46:25 PM (MST).

# INTRODUCTION

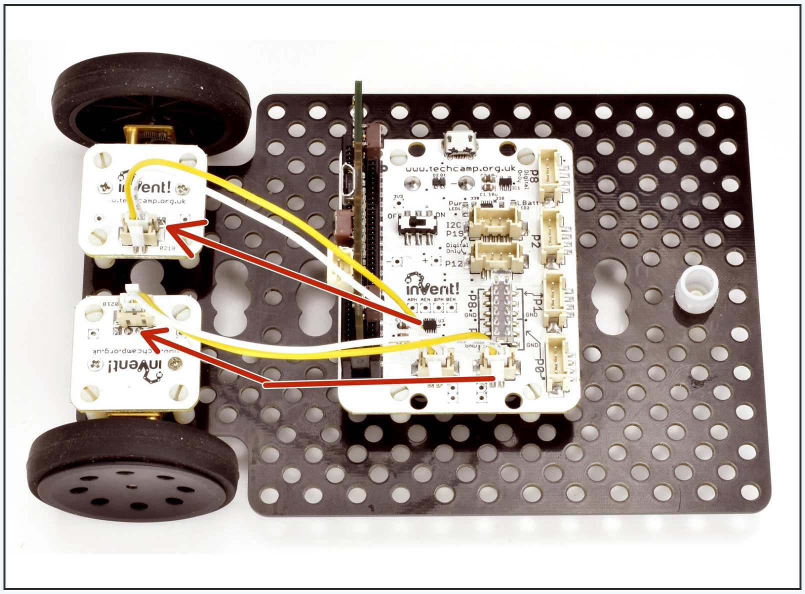
Our robot needs to do more than just move backwards and forwards if we are going to explore Mars - let's learn how to make it turn!

Step 1

## Setup your Robot

- Your robot should be setup in the same way as the previous section!

⚠ Don't forget that the left motor should be plugged into M1, and the right motor into M2!



This document was generated on 2022-01-01 09:46:25 PM (MST).

Step 2

## How to Turn

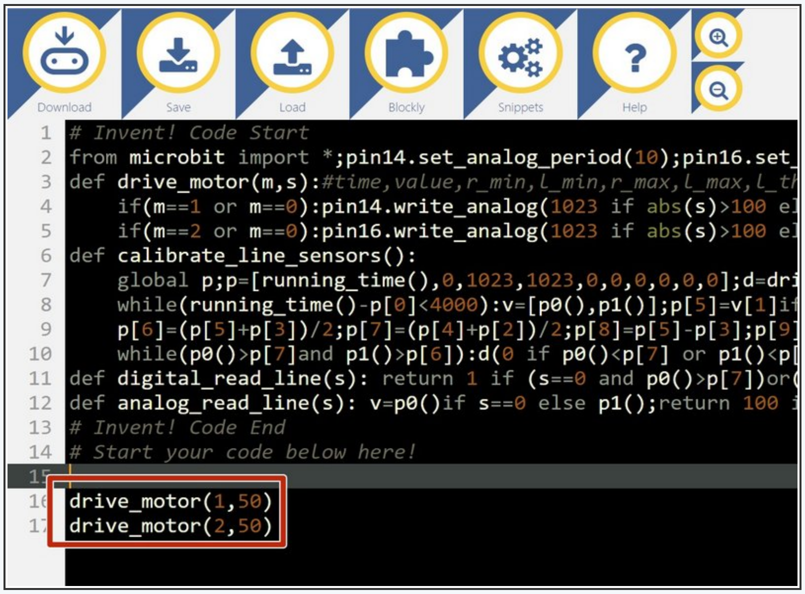
- Turning a car is easy - you can just rotate the wheels!
- However the wheels on most robots are **fixed,** so we have to do something different.
- Fixed wheel machines steer using **'skid steering'** - exactly the same way as a tank! You might have also seen small diggers that have 4 fixed wheels steering like this as well.
- Skid steering is easy - you just **increase or decrease the speed of one wheel**, and the robot will turn!

This document was generated on 2022-01-01 09:46:25 PM (MST).

## Step 3

## Setup the Program

- **Load** up your program from the previous session, and make sure it has the lines of code at the top that we always need.
- Then add the line of code in the picture, which will make both motors go forwards at **50%.**



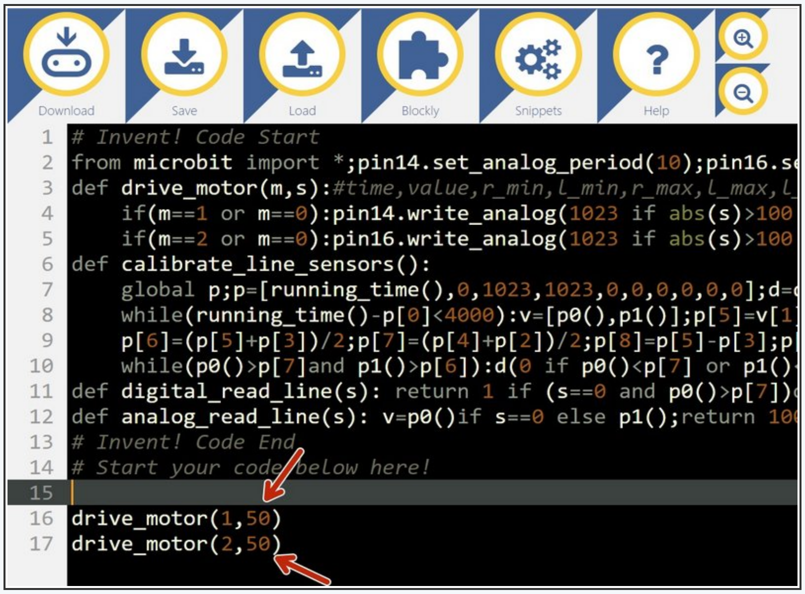
```
# Invent! Code Start
from microbit import *;pin14.set_analog_period(10);pin16.set_
def drive_motor(m,s):#time,value,r_min,l_min,r_max,l_max,l_th
    if(m==1 or m==0):pin14.write_analog(1023 if abs(s)>100 el
    if(m==2 or m==0):pin16.write_analog(1023 if abs(s)>100 el
def calibrate_line_sensors():
    global p;p=[running_time(),0,1023,1023,0,0,0,0,0,0];d=dri
    while(running_time()-p[0]<4000):v=[p0(),p1()];p[5]=v[1]if
    p[6]=(p[5]+p[3])/2;p[7]=(p[4]+p[2])/2;p[8]=p[5]-p[3];p[9]
    while(p0()>p[7]and p1()>p[6]):d(0 if p0()<p[7] or p1()<p[
def digital_read_line(s): return 1 if (s==0 and p0()>p[7])or(
def analog_read_line(s): v=p0()if s==0 else p1();return 100 i
# Invent! Code End
# Start your code below here!

drive_motor(1,50)
drive_motor(2,50)
```

## Step 4

## Motor Functions

- Remember how the **second number" we give to the** drive_motor **function controls the** speed"'?
- **Experiment** with changing the speed of **one motor** and uploading the code to your robot. Which way do you think the robot will turn? How can you make it turn faster?



```
# Invent! Code Start
from microbit import *;pin14.set_analog_period(10);pin16.se
def drive_motor(m,s):#time,value,r_min,l_min,r_max,l_max,l_
    if(m==1 or m==0):pin14.write_analog(1023 if abs(s)>100
    if(m==2 or m==0):pin16.write_analog(1023 if abs(s)>100
def calibrate_line_sensors():
    global p;p=[running_time(),0,1023,1023,0,0,0,0,0,0];d=
    while(running_time()-p[0]<4000):v=[p0(),p1()];p[5]=v[1
    p[6]=(p[5]+p[3])/2;p[7]=(p[4]+p[2])/2;p[8]=p[5]-p[3];p
    while(p0()>p[7]and p1()>p[6]):d(0 if p0()<p[7] or p1()
def digital_read_line(s): return 1 if (s==0 and p0()>p[7])
def analog_read_line(s): v=p0()if s==0 else p1();return 10
# Invent! Code End
# Start your code below here!

drive_motor(1,50)
drive_motor(2,50)
```

Step 5

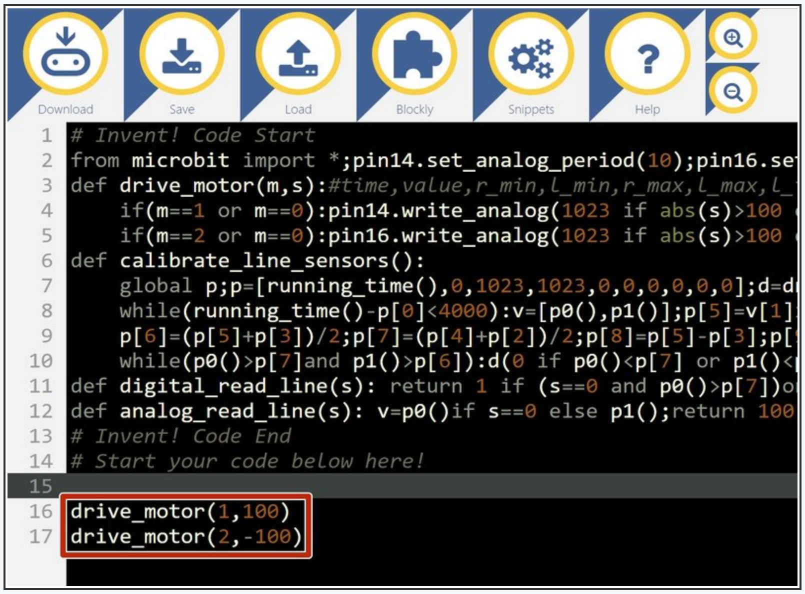
## Turning Challenge

- Let's use what we've learned to **pickup** the stranded astronaut again.
- This time, pickup the astronaut by driving your robot **around the outside of the planet!** Try and stay as close as you can to the edge.
- Once you've picked up the astronaut, use a **sleep()**line and two more **drive_motor** functions, to make your robot **reverse** back around the planet and go **back to where it started.**

Step 6

## Super Fast Turning

- You may have worked this out already, but we can actually make our robot **spin on the spot** if we want to!
- Try making one motor go **forwards** at 100, and the other **backwards** at 100. Your robot should spin on the spot!



```
# Invent! Code Start
from microbit import *;pin14.set_analog_period(10);pin16.se
def drive_motor(m,s):#time,value,r_min,l_min,r_max,l_max,l_
    if(m==1 or m==0):pin14.write_analog(1023 if abs(s)>100 
    if(m==2 or m==0):pin16.write_analog(1023 if abs(s)>100 
def calibrate_line_sensors():
    global p;p=[running_time(),0,1023,1023,0,0,0,0,0,0];d=d
    while(running_time()-p[0]<4000):v=[p0(),p1()];p[5]=v[1]
    p[6]=(p[5]+p[3])/2;p[7]=(p[4]+p[2])/2;p[8]=p[5]-p[3];p[
    while(p0()>p[7]and p1()>p[6]):d(0 if p0()<p[7] or p1()<
def digital_read_line(s): return 1 if (s==0 and p0()>p[7])o
def analog_read_line(s): v=p0()if s==0 else p1();return 100
# Invent! Code End
# Start your code below here!

drive_motor(1,100)
drive_motor(2,-100)
```

Step 7

## Spinning Challenge

- Now we can turn the robot, let's pick up the astronaut by driving across the planet, but come back **facing forwards!**
- Write a program to go and **pick up** the astronaut, **turn** on the spot and drive back to base going **forwards**.

This document was generated on 2022-01-01 09:46:25 PM (MST).